**Introduction**
This is the Report for the Neural Network Project in which we try to replicate the results obtained in the paper: *Zero Shot Image Restoration Using Denoising Diffusion Null-Space Model* (Wang et al., 2022) [1]

**Paper Description**
The paper introduces a new zero-shot approach to solve Image Restoration (IR) tasks starting from a pre-trained model. The authors propose a sampling algorithm called DDNM and an enhanced version DDNM+.
To properly understand these new methods, a small background is needed on the following topics:

- the state-of-the-art diffusion model presented in *Denoising Diffusion Probabilistic Model* (DDPM) (Ho et al., 2020) [2]. Indeed, DDPM is the starting point for the improvements in the paper we are working on. It also provides the definitions of the probability distributions $q$ and $p$;
- Range-Null Space Decomposition: any sample $x$ can be decomposed into its range-space and null-space as follows $x \equiv A^\dagger Ax + (I - A^\dagger A)x$. Moreover, some useful properties hold over the components of this decomposition, for example: $AA^\dagger Ax \equiv Ax$ and $A(I - A^\dagger A)x \equiv 0$.

After giving us this initial background, the paper starts by explaining the basic method considering the simple case of noise-free IR. This initial problem can be described as $y = Ax$ where $y$ is the degraded image, $A$ is the degradation operator and $x$ is the ground-truth (GT) image. Therefore, given $y$, the proposed methods aim to produce a $\hat{x}$ that satisfies the constraints of *Consistency* ($A\hat{x} \equiv y$) and *Realness* ($\hat{x} \sim q(x)$).

At this point, the paper explains how to meet these constraints. The *Consistency* constraint is met thanks to the range-null space decomposition. Indeed, if we consider $Ax = y$ and decompose the GT image $x$, we obtain $Ax \equiv AA^\dagger Ax + A(I - A^\dagger A)x \equiv Ax + 0 = y$. But this means that starting from a degraded image $y$ we can obtain a general solution $\hat{x} = A^\dagger y + (I - A^\dagger A)\overline{x}$ that satisfies the *Consistency* constraint whatever value is assigned to $\overline{x}$. Indeed

$$A\hat{x} \equiv A(A^\dagger y + A(I - A^\dagger A)\overline{x}) \equiv AA^\dagger y + A(I - A^\dagger A)\overline{x} \equiv AA^\dagger y = AA^\dagger A\hat{x} \equiv A\hat{x} = y$$

It is possible to see that $\overline{x}$ does not affect *Consistency* but influences the *Realness* $\hat{x} \sim q(x)$. Now, the goal is to find a $\overline{x}$ such that *Realness* is met; this is done using the diffusion model to predict the noise and iteratively clean the degraded image and refine $\overline{x}$ to obtain better results. The resulting sampling algorithm is *Algorithm 1* in [1].

This first approach works, but is limited due to the assumption of noise-free tasks, so the authors propose an enhanced version called DDNM+ that copes with additional noise. Consider a general noisy IR problem described as $y = Ax + n$ where $y, A$ and $x$ have the same meaning as before and $n \sim \mathcal{N}(0, \sigma_y^2 I)$ is Gaussian noise.
Instead of providing an implementation, the paper introduces the Time-Travel Trick as an additional building block to obtain a better *Realness*. The trick consists of reversing $l$ steps at each time step $t$. In this way, we travel back in time to create a better 'past' knowing

the 'present' and this yields to a better 'future'.
At this point DDNM+ and the time travel trick are combined to obtain *Algorithm 2* most general version of the method presented in [1].

Using any version of DDNM the paper also introduces the Mask-Shift Trick that allows to solve IR task of arbitrary dimensions. This is done by dividing the degraded image into patches that are then passed in pairs as input to the algorithm.

The algorithm presented is Zero Shot because it can perform any IR task without needing a model explicitly trained for the task. Only matrix A must be chosen according to the task to be performed.

**Our Approach**
After reading the paper, we decided to implement the sampling process described by Ho et al. (2020) [2] even if only mentioned by Wang et al. (2022) [1]. In this way, we would better understand the inner workings of a state-of-the-art diffusion model, giving us a good starting point from which to understand how we should implement the algorithm proposed by [1] (the implementation can be seen in the Google Colab file under Appendix A).

Once the DDPM sampling was implemented, we moved to implement the algorithms presented by Wang et al.(2022) [1] starting from the standard DDNM. Following the original implementation of DDPM [2] we set $T = 1000$, $\beta_t$ as linearly increasing constants between 0.0001 and 0.02 and following the reparameterization trick from Kingma et al. (2014)[3] we defined $\alpha_t = 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{i=0}^{t} \alpha_i$. So, we start iterating from $t = 1000$ to $t = 1$ and at each time step $t$ we compute the estimate of the final output $x_0$ defined as

$$x_{0|t} = \frac{1}{\sqrt{\overline{\alpha}_t}}(x_t - \mathcal{Z}_\theta(x_t, t)\sqrt{1 - \overline{\alpha}_t}) \tag{1}$$

where $\mathcal{Z}_\theta$ is a denoising diffusion model. In our implementation, we used two models pretrained on ImageNet and CelebA. Then, we computed a rectified estimation in which we fix the range-space and leave untouched the null-space

$$\hat{x}_{0|t} = A^\dagger y + (I - A^\dagger A)x_{0|t} = x_{0|t} - A^\dagger(Ax_{0|t} - y) \quad [1] \tag{2}$$

Because $\hat{x}_{0|t}$ estimates $x_0$, we use it to sample $x_{t-1}$ from $p(x_{t-1}|x_t, \hat{x}_{0|t})$ with the values of $\mu_t(x_t, x_0)$ and $\sigma_t^2$ defined in the article.

$$x_{t-1} = \mu_t(x_t, x_0)\hat{x}_{0|t} + \sigma_t\epsilon = \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1 - \overline{\alpha}_t}\hat{x}_{0|t} + \frac{\sqrt{\alpha_t}(1 - \overline{\alpha}_{t-1})}{1 - \overline{\alpha}_t}x_t + \sigma_t\epsilon \tag{3}$$

with $\epsilon \sim \mathcal{N}(0, I)$. At each step, the noise we add reduces the disharmony between the range-space and null-space resulting in a final output $x_0 \sim q(x)$.

At this point, using the code snippets provided in the paper [1] to implement $A$ and $A^\dagger$, we tested our implementation on the three main IR tasks, obtaining as expected results that are lacking in *Realness*.

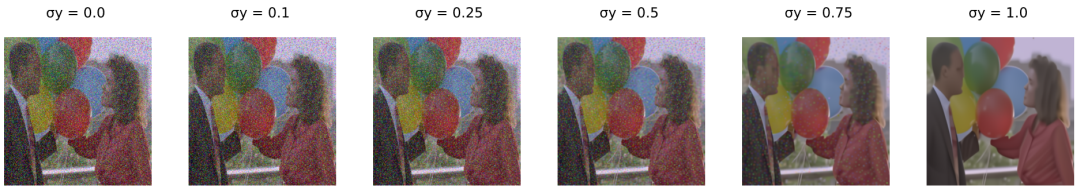
[1] this equality is not explicitly written in [1], we introduced it to better represent how this expression is written in code


Moving to the next algorithm, we decided to implement an intermediate version in which there is only the handling of noisy IR and not the time travel trick.
The implementation was straightforward, we simply needed to introduce $\Sigma_t$ and $\Phi_t$ to, respectively, scale the range-space correction and the added noise. In practice, this meant changing the above-mentioned equation into:

$$\hat{x}_{0|t} = x_{0|t} - \Sigma_t A^{\dagger}(Ax_{0|t} - y) \tag{4}$$

$$x_{t-1} = \frac{\sqrt{\overline{\alpha}_{t-1}}\beta_t}{1-\overline{\alpha}_t}\hat{x}_{0|t} + \frac{\sqrt{\alpha_t}(1-\overline{\alpha}_{t-1})}{1-\overline{\alpha}_t}x_t + \sqrt{\Phi_t}\epsilon \tag{5}$$

Because we are still considering a simple case where the extra noise is approximated to a Gaussian, we can approximate $\Sigma_t$ and $\Phi_t$ using $\gamma_t$ and $\sigma_t$ that are defined in Eq.19 of [1]. In these equations, the value $\sigma_y$ is quite relevant because it represents the amount of noise present in the image.
To implement in code $\Sigma_t$ and $\Phi_t$, we used a part of a given snippet of code.
Once this first version of DDNM+ was implemented, we performed several experiments testing not only the usual IR tasks, but also the denoising capability of the method by trying different values of $\sigma_y$.




After completing the experiments, we proceeded to implement DDNM + with Time-Travel following the pseudocode shown in *Algorithm 2* in [1]. Particular care was needed in "translating" $x_{t+L} \sim q(x_{t+L}|x_t)$ that allows us to compute a new value of the 'past' considering a value from the 'present'.
Considering that the forward process from which we are sampling is defined as a Markov chain, we could write $q(x_{t+L}|x_t) = \prod_{i=t}^{t+L-1} q(x_{i+1}|x_i)$. Moreover considering the reparameterization trick, we have:

$$q(x_{t+L}|x_t) = \mathcal{N}(x_{t+L}; \sqrt{\frac{\overline{\alpha}_{t+L}}{\overline{\alpha}_t}}x_t; (1 - \frac{\overline{\alpha}_{t+L}}{\overline{\alpha}_t})I) \tag{6}$$

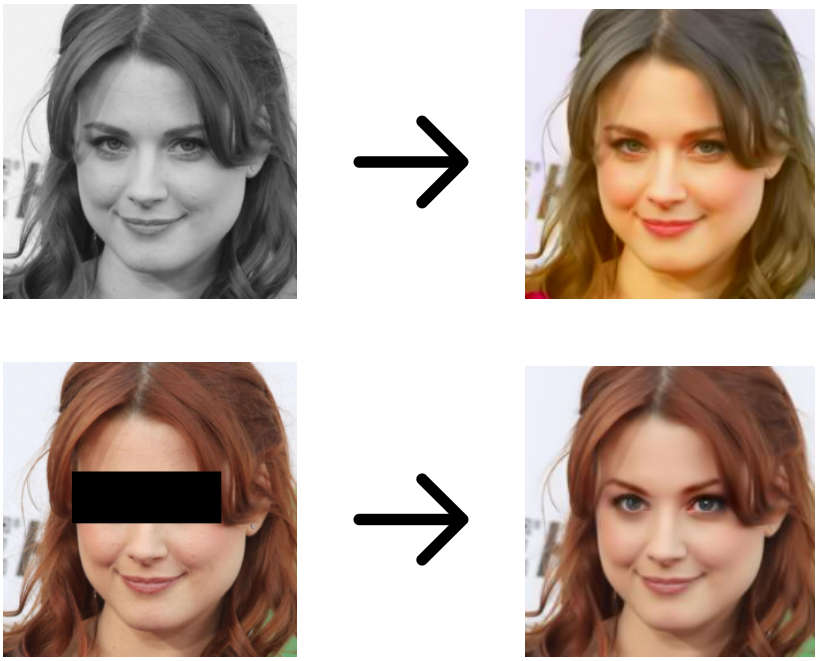
and this means that the sample $x_{t+L}$ is

$$x_{t+L} = \mu_t x_t + \sigma_t \epsilon = \sqrt{\frac{\overline{\alpha}_{t+L}}{\overline{\alpha}_t}}x_t + \sqrt{1 - \frac{\overline{\alpha}_{t+L}}{\overline{\alpha}_t}}\epsilon \tag{7}$$

In this way, "the time-travel trick produces a better 'past', which in turn produces a better 'future'."[1].

Starting from this new past, we recompute all the steps, using basic DDNM+, until we get back to our starting point.

Furthermore, because the paper very briefly mentioned $s$ as a hyper parameter that controls the interval of using the time travel trick, we also implemented it.



Now that we have all the implementations of DDNM, we performed experiments similar to those shown in the paper.
In conclusion, we also implemented the Mask-Shift trick, it is possible to see the result obtained in the image gallery. (The code can be seen in the Google Colab file in Appendix B).

**Evaluation:** To evaluate our implementations, we considered the same metrics used in the paper:

- Fréchet Inception Distance - FID: compares the distribution of the generated image with the ground truth one;
- Peak Signal to Noise Ratio - PSNR: measures the quality of a reconstructed signal (image in our case) compared to the original one;
- Structure Similarity Index Method - SSIM: measures the similarity between two images.
- *Consistency* metric - CONS: measures the absolute value norm of the difference between the degraded image $y$ and the result image $x_0$ to which we apply the degradation operator ($||Ax_0 - y||_1$).

We applied these metrics to the three main tasks (Colorization, Inpainting, and Super Resolution) and to their combination. Moreover, each experiment was performed both with and without denoising.

In doing these experiments, we also considered how the model influences the results. For this reason, for each pre-trained model, we selected a starting input relevant for that model. We chose the following starting images: for CelebA a woman's close-up shot, for ImageNet a lion, and a picture of some lemons (the used model is trained on a subset of the ImageNet dataset containing plants, animals, and naturally occurring objects).

Before showing the results, we should mention that, as explained in [1], the *Consistency* metric has meaning only in the context of a colorization task, so we will consider it only in such cases. Moreover, PSNR and SSIM do not reflect the colorization performance. For this reason, in the experiments that involve colorization, we considered only the FID and the *Consistency* metric.

As you can see in the following tables, CelebA almost always delivers better performance when used in combination with DDNM+ with time-travel. Furthermore, the inpainting and super-resolution tasks had better results when used without any kind of noise removal, that is, using $\sigma_y = 0.0$. It is worth pointing out that the image used for testing does not contain noise.

Considering the tests using ImageNet on a picture of a lion, we can see that the colorization and inpainting tasks had better results when carried out using DDNM without noise removal.

The last experiments were carried out using ImageNet on a picture of some lemons. It stands out that for the task inpainting, better results were obtained using the DDNM algorithm without time travel or denoising.

In all other cases, as you would expect, the best outcomes were achieved using DDNM+ with time-travel in combination with $\sigma_y = 0.1$ for denoising.

The following abbreviations were used in the tables:

- ALL: Colorization + Inpainting + Super resolution 8x
- COL: Colorization
- INP: Inpainting
- SR: Super resolution 16x

- DDNM denotes DDNM
- DDNM+ denotes DDNM+ without time-travel
- DDNM+TT denotes DDNM+ with time-travel

**Woman:** CelebA pre-trained model. DDNM+ with time-travel uses l=5. The mask used for the inpainting task covered the eyes.

| $\sigma_y = 0.1$ | ALL | COL | | INP | | | SR | | |
|---|---|---|---|---|---|---|---|---|---|
| metrics | fid↓ | fid↓ | *Cons* ↓ | fid↓ | psnr↑ | ssim↑ | fid↓ | psnr↑ | ssim↑ |
| $A^\dagger y$ | 8.47 | 6.93 | 0.0 | 0.35 | 15.86 | 0.86 | 4.67 | 19.56 | 0.47 |
| DDNM | 87.11 | 83.30 | 901.8 | 0.10 | 27.55 | 0.95 | 1.50 | 20.97 | 0.55 |
| DDNM+ | 70.79 | 70.73 | 1266 | 0.25 | 26.93 | 0.90 | 0.24 | 21.06 | 0.57 |
| DDNM+TT | 3.93 | 1.56 | 733.7 | 0.59 | 30.32 | 0.90 | 1.26 | 22.24 | 0.60 |

Table 1: Experiments on a woman image using $\sigma_y = 0.1$

| $\sigma_y = 0.0$ | ALL | COL | | INP | | | SR | | |
|---|---|---|---|---|---|---|---|---|---|
| metrics | fid↓ | fid↓ | *Cons* ↓ | fid↓ | psnr↑ | ssim↑ | fid↓ | psnr↑ | ssim↑ |
| $A^\dagger y$ | 8.47 | 6.93 | 0.0 | 0.35 | 15.86 | 0.86 | 4.67 | 19.56 | 0.47 |
| DDNM | 87.11 | 83.30 | 901.8 | 0.10 | 27.55 | 0.95 | 1.50 | 20.97 | 0.55 |
| DDNM+ | 87.11 | 83.30 | 901.8 | 0.10 | 27.55 | 0.95 | 1.50 | 20.97 | 0.55 |
| DDNM+TT | 4.28 | 3.31 | 137.7 | 0.004 | 31.72 | 0.96 | 0.41 | 21.96 | 0.61 |

Table 2: Experiments on a woman image using $\sigma_y = 0.0$

**Lion:** ImageNet pre-trained model. DDNM+ with time-travel uses l=5. The mask used for the inpainting task covered the snout.

| $\sigma_y = 0.1$ | ALL | COL | | INP | | | SR | | |
|---|---|---|---|---|---|---|---|---|---|
| metrics | fid↓ | fid↓ | *Cons* ↓ | fid↓ | psnr↑ | ssim↑ | fid↓ | psnr↑ | ssim↑ |
| $A^\dagger y$ | 15.44 | 7.19 | 0.0 | 1.83 | 13.85 | 0.82 | 19.39 | 19.18 | 0.34 |
| DDNM | 81.16 | 3.00 | 47.94 | 0.10 | 18.70 | 0.86 | 114.2 | 12.27 | 0.14 |
| DDNM+ | 76.78 | 2.84 | 447.9 | 0.60 | 18.70 | 0.85 | 116.3 | 12.35 | 0.15 |
| DDNM+TT | 22.88 | 9.76 | 622.5 | 2.08 | 22.55 | 0.85 | 19.50 | 19.45 | 0.43 |

Table 3: Experiments on a lion image using $\sigma_y = 0.1$

| $\sigma_y = 0.0$ | ALL | COL | | INP | | | SR | | |
|---|---|---|---|---|---|---|---|---|---|
| metrics | fid↓ | fid↓ | *Cons* ↓ | fid↓ | psnr↑ | ssim↑ | fid↓ | psnr↑ | ssim↑ |
| $A^\dagger y$ | 15.44 | 7.19 | 0.0 | 1.83 | 13.85 | 0.82 | 19.39 | 19.18 | 0.34 |
| DDNM | 81.16 | 3.00 | 47.94 | 0.10 | 18.70 | 0.86 | 114.2 | 12.27 | 0.14 |
| DDNM+ | 81.16 | 3.00 | 47.94 | 0.10 | 18.70 | 0.86 | 114.2 | 12.27 | 0.14 |
| DDNM+TT | 21.08 | 3.00 | 47.46 | 0.14 | 22.53 | 0.87 | 11.53 | 19.22 | 0.41 |

Table 4: Experiments on a lion image using $\sigma_y = 0.0$

**Lemons:** ImageNet pre-trained model. DDNM+ with time-travel uses l=5. The mask used for the inpainting task covered a lemon and some leaves.

| $\sigma_y = 0.1$ | ALL | COL | | INP | | | SR | | |
|---|---|---|---|---|---|---|---|---|---|
| metrics | fid↓ | fid↓ | *Cons* ↓ | fid↓ | psnr↑ | ssim↑ | fid↓ | psnr↑ | ssim↑ |
| $A^\dagger y$ | 22.85 | 21.13 | 0.0 | 0.55 | 14.41 | 0.89 | 6.68 | 19.10 | 0.47 |
| DDNM | 24.15 | 25.11 | 65.7 | 0.03 | 25.74 | 0.94 | 42.26 | 16.66 | 0.34 |
| DDNM+ | 24.33 | 27.41 | 449.7 | 0.96 | 25.60 | 0.90 | 31.84 | 16.81 | 0.36 |
| DDNM+TT | 22.33 | 24.38 | 528.2 | 1.90 | 25.56 | 0.89 | 2.51 | 19.90 | 0.56 |

Table 5: Experiments on a lemon image using $\sigma_y = 0.1$

| $\sigma_y = 0.0$ | ALL | COL | | INP | | | SR | | |
|---|---|---|---|---|---|---|---|---|---|
| metrics | fid↓ | fid↓ | *Cons* ↓ | fid↓ | psnr↑ | ssim↑ | fid↓ | psnr↑ | ssim↑ |
| $A^\dagger y$ | 22.85 | 21.13 | 0.0 | 0.55 | 14.41 | 0.89 | 6.68 | 19.10 | 0.47 |
| DDNM | 24.15 | 25.11 | 65.7 | 0.03 | 25.74 | 0.94 | 42.26 | 16.66 | 0.34 |
| DDNM+ | 24.15 | 25.11 | 65.7 | 0.03 | 25.74 | 0.94 | 42.26 | 16.66 | 0.34 |
| DDNM+TT | 23.80 | 22.33 | 67.44 | 0.04 | 25.73 | 0.95 | 1.81 | 19.69 | 0.52 |

Table 6: Experiments on a lemon image using $\sigma_y = 0.0$

**Conclusions** This report reflects the process we went through to analyze and replicate the chosen paper [1]. It allowed us to study in depth a state-of-the-art approach for Image Restoration and, more in general, for Image Generation. It also allows us to experiment with new technologies and learn the importance of setting the correct hyperparameters to balance between computing time and result quality. A selection of generated images can be found in the gallery that follows.

## References

[1] Yinhuai Wang, Jiwen Yu, and Jian Zhang. *Zero-Shot Image Restoration Using Denoising Diffusion Null-Space Model*. 2022. arXiv: 2212.00490 [cs.CV].

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. *Denoising Diffusion Probabilistic Models*. 2020. DOI: 10.48550/ARXIV.2006.11239. URL: https://arxiv.org/abs/2006.11239.

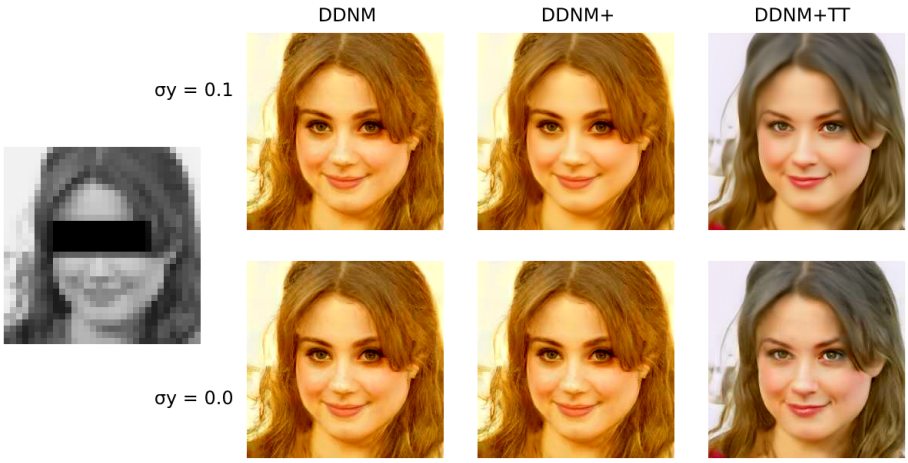
[3] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2013 (v1), last revised 10 Dec 2022 (this version, v11). DOI: 10.48550/ARXIV.1312.6114. URL: https://arxiv.org/abs/1312.6114.

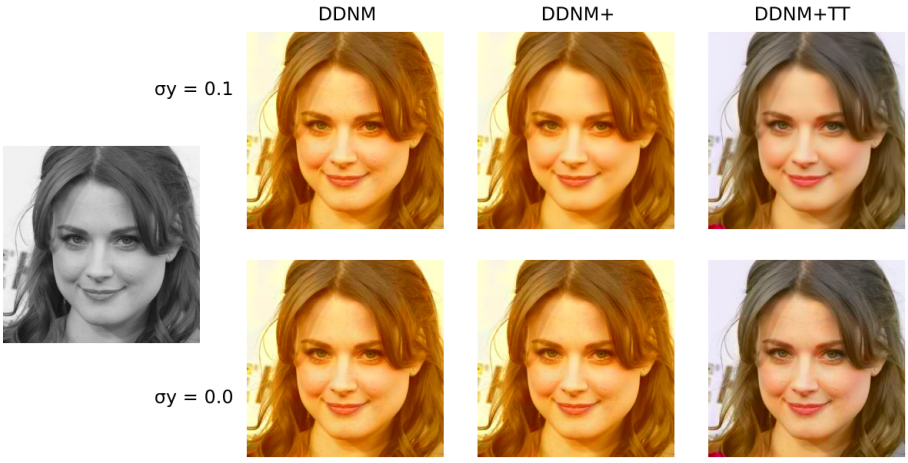
# Image gallery

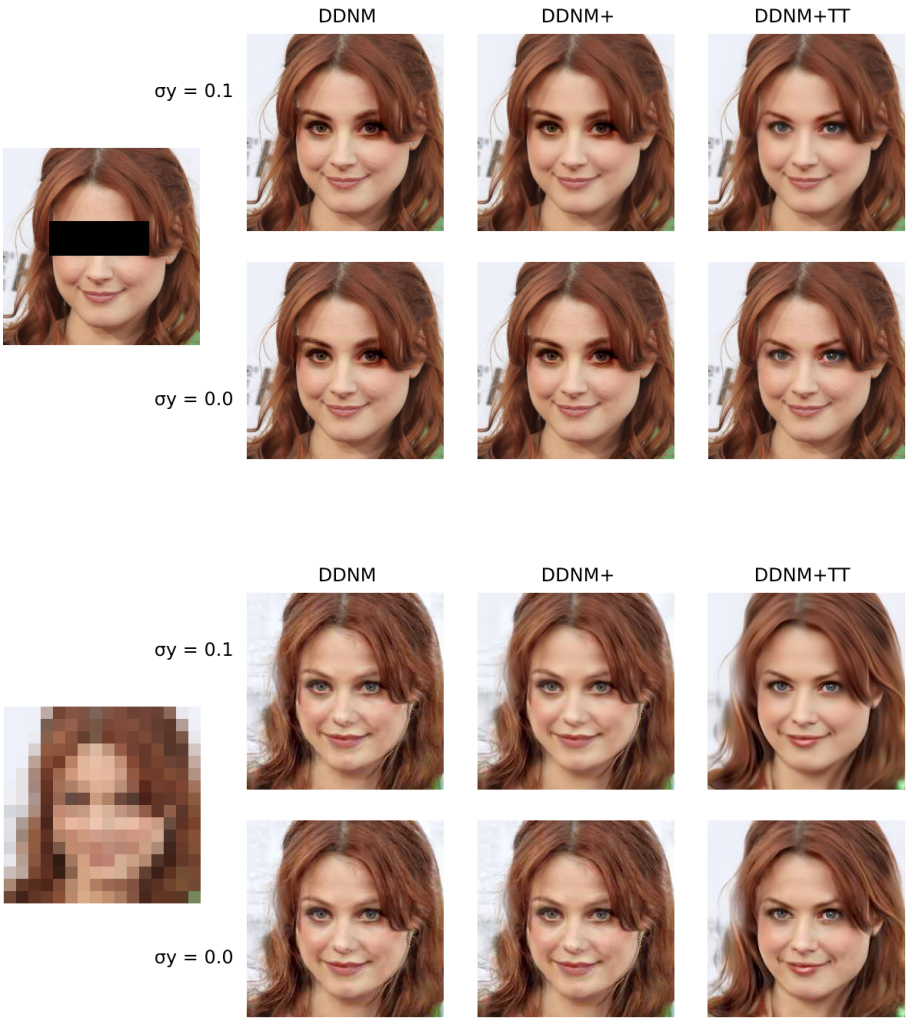
### Original images



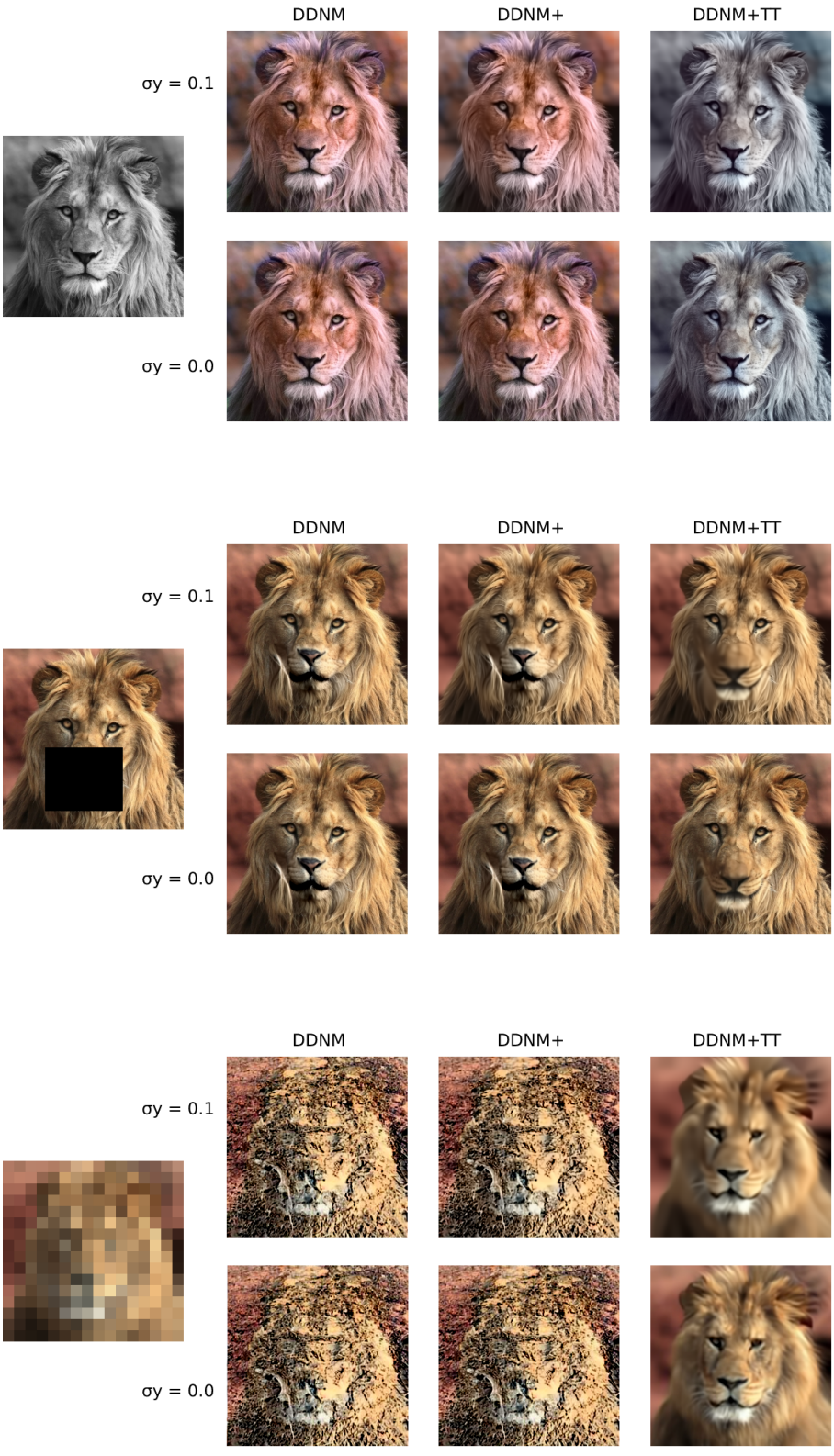
**Woman** In order: all, colorization, inpainting, super resolution 16x



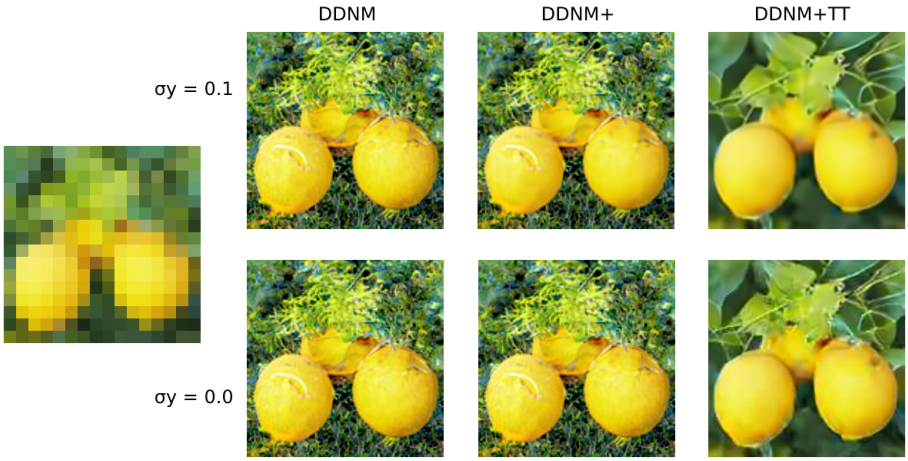





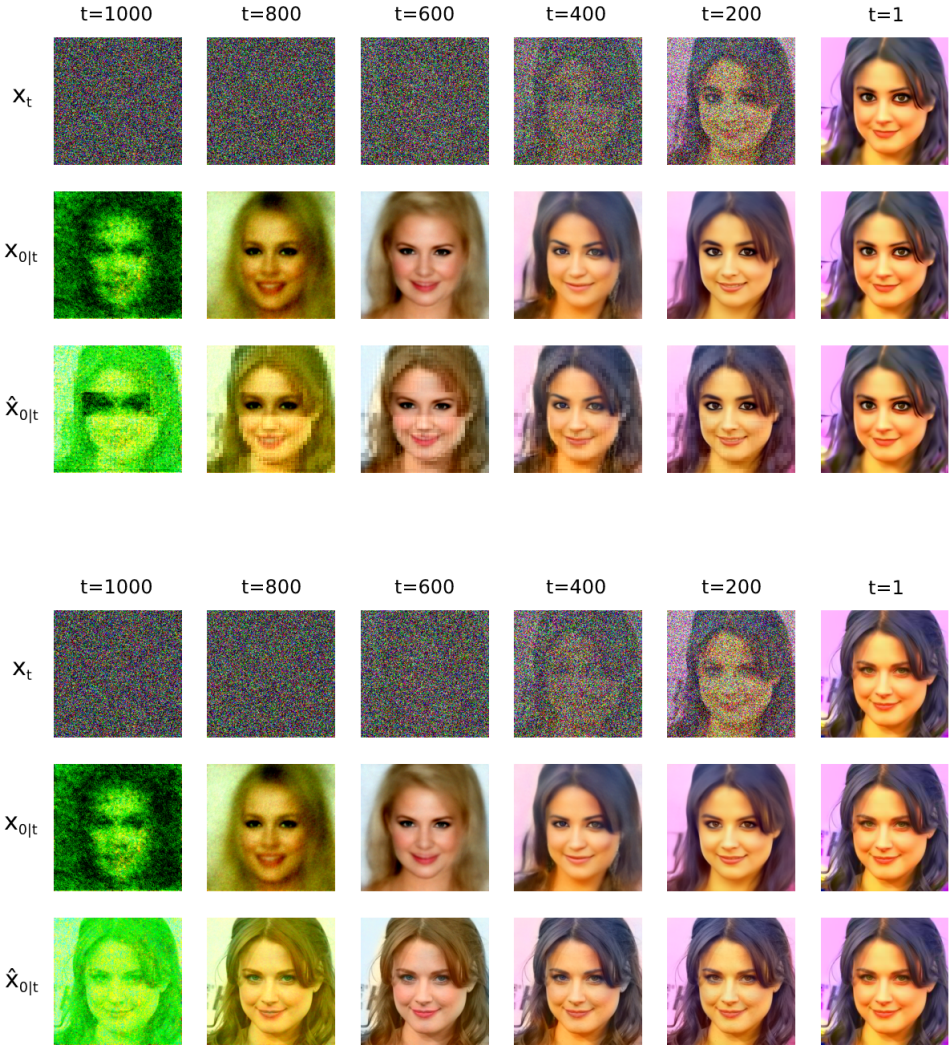

**Lion** In order: all, colorization, inpainting, 16x super resolution

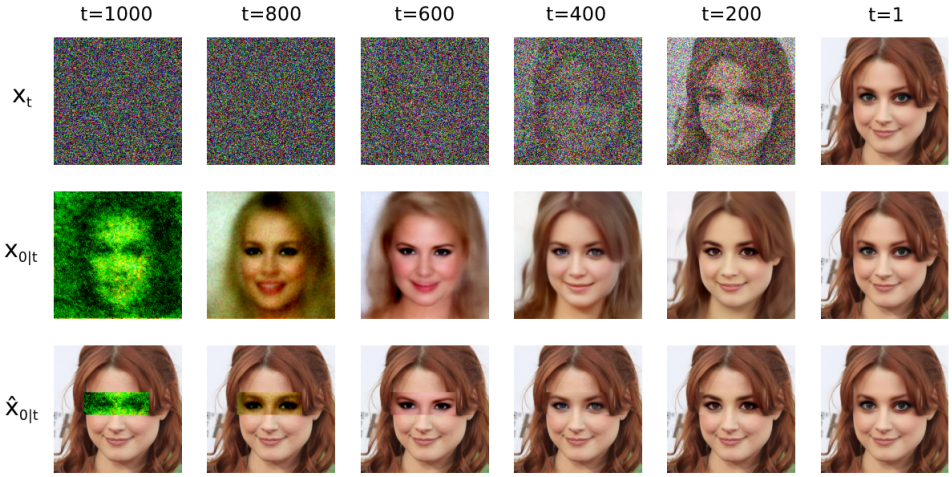

DDNM
DDNM+
DDNM+TT
σy = 0.1
σy = 0.0
DDNM
DDNM+
DDNM+TT
σy = 0.1
σy = 0.0
DDNM
DDNM+
DDNM+TT
σy = 0.1
σy = 0.0

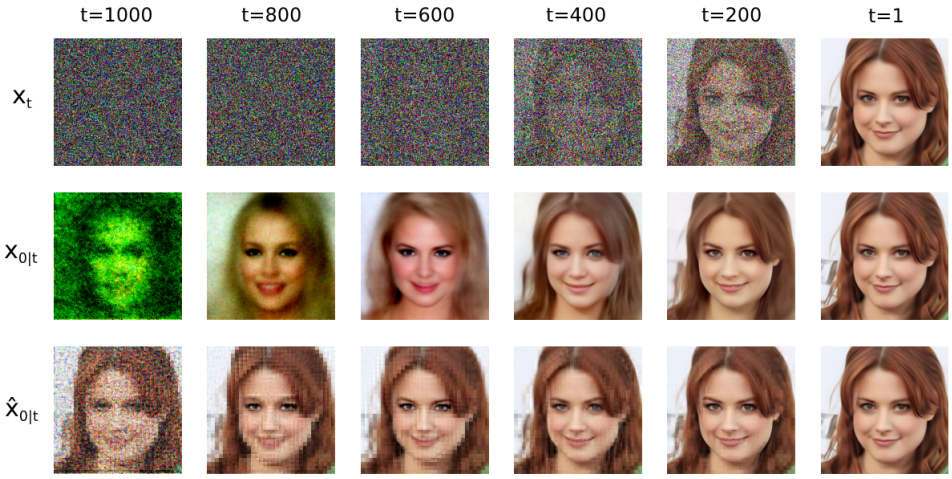
**Lemons** In order: all, colorization, inpainting, super resolution 16x




**Evolution** It shows the progression of the images during the sampling process. In order: all, colorization, inpainting, super resolution 16x






### Mask-shift trick