

Project 3: Fun with Flow Models

Ray Sun · CS 180 · Fall 2026

Part A explores image generation and editing with PixNerd. Part B builds and trains UNets for single-step denoising and flow matching on MNIST.

[Part A](#) · [Part B](#) · [Website PDF](#)

Part A · Exploring the Power of Flow Models

Pretrained PixNerd · 512×512 images · float32 states and predictions

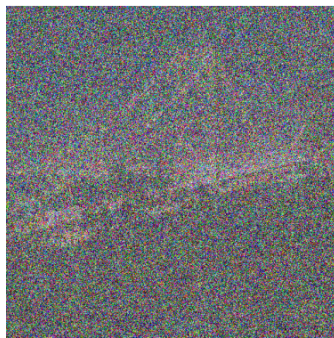
1.1 · Constructing noisy training inputs

I used my nighttime bridge photograph for Sections 1.1-1.5. The input is center-cropped and resized to 512×512 , with RGB values normalized to $[-1, 1]$. I sampled one Gaussian noise tensor with seed 180 and reused it at all three times.

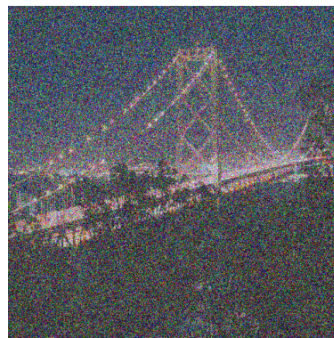
$$\mathbf{x}_t = (1 - t)\mathbf{e} + t\mathbf{x}_1$$



Original



$t = 0.25$



$t = 0.50$



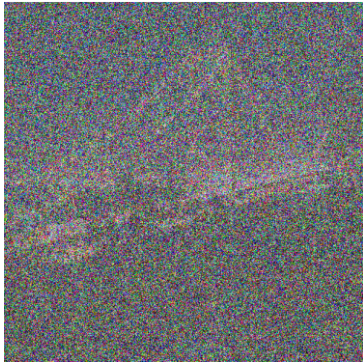
$t = 0.75$

At $t = 0.25$, the image is dominated by noise and the bridge is difficult to recognize. At $t = 0.50$, the tower and lights become visible. At $t = 0.75$, most of the original structure remains, although the sky and dark foreground still contain noise. Increasing t increases the contribution of the original image.

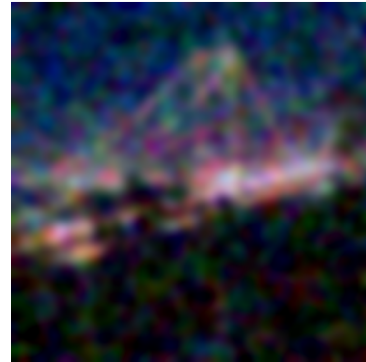
1.2 · Classical denoising

For the classical baseline, I first divided the noisy input by t to restore the image amplitude, then applied Gaussian blur. I tried $\sigma = 1, 2, 4, 8, 16$ and selected the candidate with the lowest mean squared error against the known original, before display clipping.

$t = 0.25$



Noisy input



Gaussian blur, $\sigma = 8$

$t = 0.50$

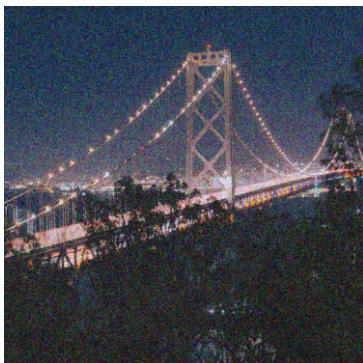


Noisy input



Gaussian blur, $\sigma = 4$

$t = 0.75$



Noisy input



Gaussian blur, $\sigma = 1$

The selected blur becomes weaker as the input becomes cleaner. At $t = 0.25$, $\sigma = 8$ reduces the noise but also removes the bridge cables and fine lights. At $t = 0.50$, $\sigma = 4$ preserves more of the tower, though the result is still visibly blurred. At $t = 0.75$, $\sigma = 1$ retains much more detail. This shows the tradeoff between suppressing noise and preserving edges.

1.3 · One-step denoising

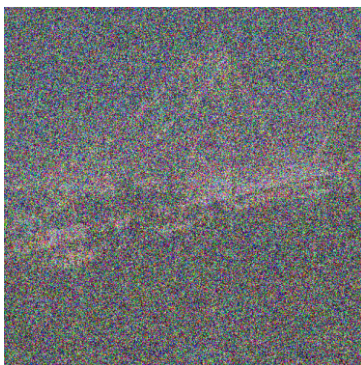
I used the prompt “a high quality photo” to predict the velocity at each noisy input. One-step denoising uses that single prediction for the entire remaining interval to $t = 1$.

$$\hat{x}_1 = x_t + (1 - t)v_\theta(x_t, t, c)$$

$t = 0.25$



Original



Noisy input



One-step estimate

$t = 0.50$



Original



Noisy input

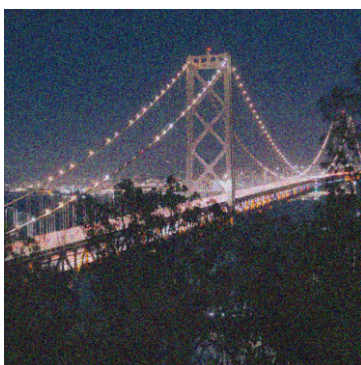


One-step estimate

$t = 0.75$



Original



Noisy input



One-step estimate

The model recovers recognizable bridge structure even at $t = 0.25$, where Gaussian blur leaves mostly broad color patches. However, this low-time estimate is soft, particularly around the cables and foreground. The estimates at $t = 0.50$ and 0.75 retain sharper lights and more of the original structure. The model can infer plausible detail, but a plausible reconstruction is not necessarily an exact recovery of the original pixels.

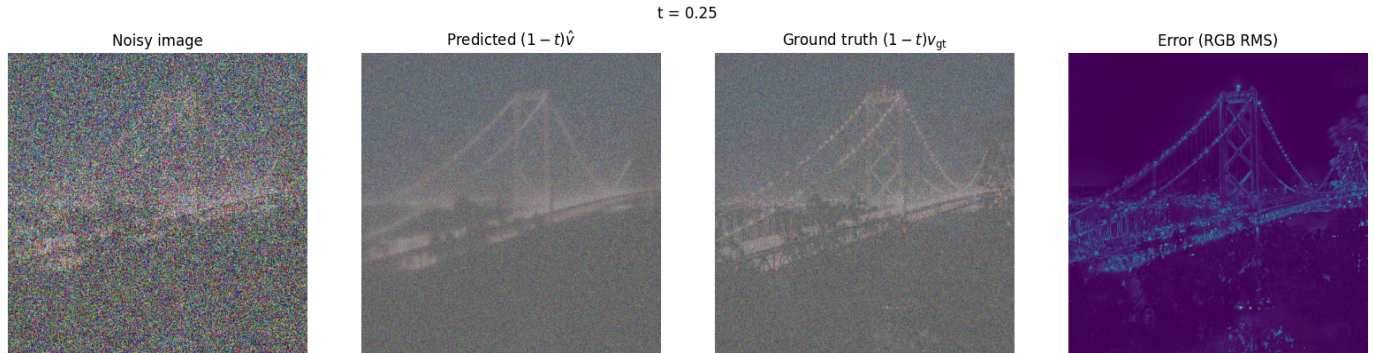
1.4 • Visualizing velocity

I compared the model's proposed update with the exact update available from the original image and saved noise. The original is shown in Section 1.1. At each pixel, I squared the difference between the two updates, averaged across RGB, and took the square root.

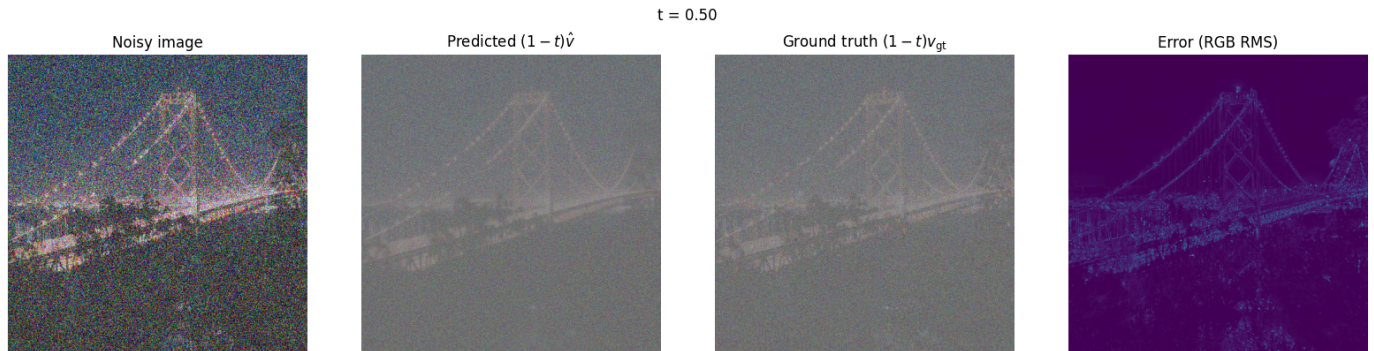
$$\Delta_{\text{pred}} = (1 - t)\hat{v}, \quad \Delta_{\text{gt}} = (1 - t)(x_1 - \epsilon) = x_1 - x_t$$

$$E = \sqrt{\text{mean}_{\text{RGB}}[(\Delta_{\text{pred}} - \Delta_{\text{gt}})^2]}$$

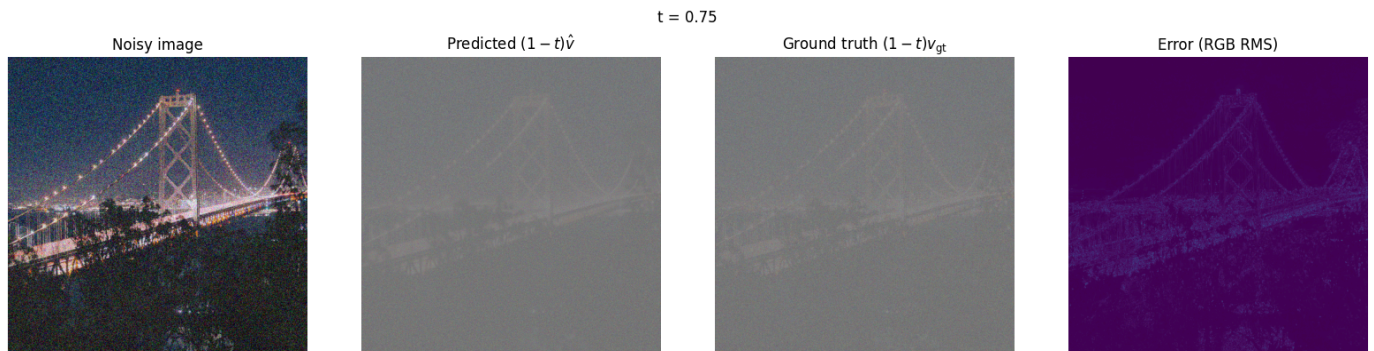
All predicted and ground-truth updates use the same symmetric RGB scale, with zero at middle gray. The heatmaps share one linear scale across all three times; errors are computed before display scaling or clipping.



t = 0.25: noisy input, predicted update, exact update, and RGB RMS error



t = 0.50: noisy input, predicted update, exact update, and RGB RMS error



t = 0.75: noisy input, predicted update, exact update, and RGB RMS error

The predicted and exact updates have similar broad structure, but the error maps trace the bridge tower, cables, lights, and foreground detail. The t = 0.25 error map is more pronounced than those at later times. Fine structures are harder to reconstruct accurately when the input contains less image information.

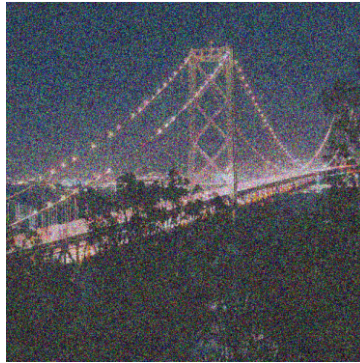
1.5 · Iterative denoising with Euler sampling

Starting from the saved $t = 0.50$ input, I used 25 Euler updates with $h = 0.02$. Each update recomputes the velocity using the current image and time. The sampler shortens the last step when needed to end at exactly $t = 1$.

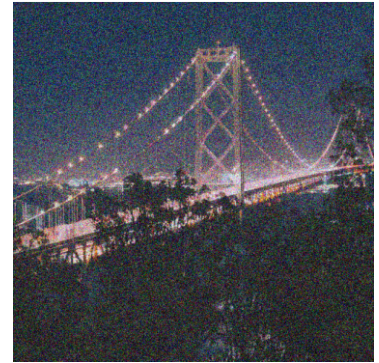
$$x_{t+h} = x_t + hv_{\theta}(x_t, t, c)$$



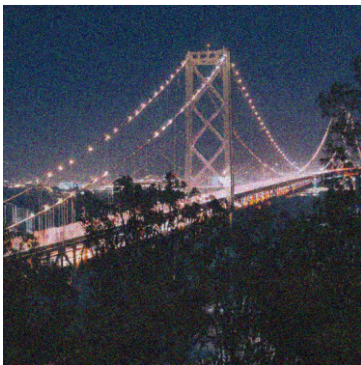
$t = 0.50$



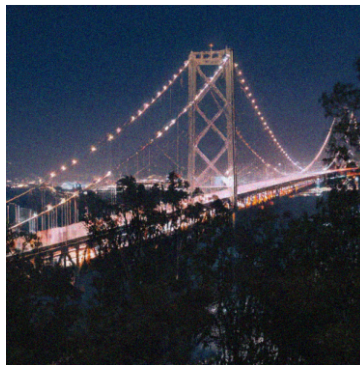
$t = 0.60$



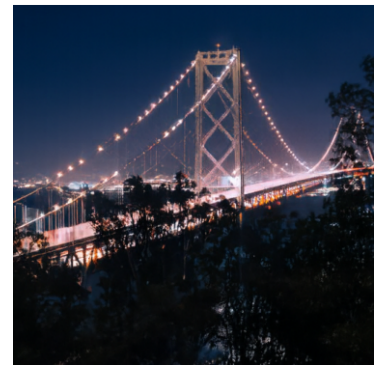
$t = 0.70$



$t = 0.80$



$t = 0.90$



$t = 1.00$

Comparison from the same $t = 0.50$ input



Original



Gaussian blur



One-step



Euler

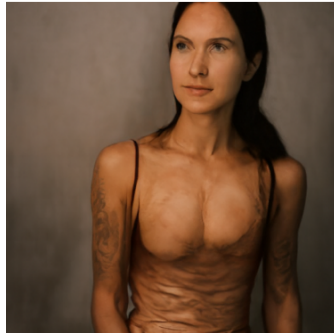
Noise gradually disappears as the trajectory approaches $t = 1$. Gaussian blur has the softest edges and loses the most detail. Both learned methods recover the bridge much more clearly. The one-step and Euler results are visually similar at this starting time, so this example does not show a dramatic advantage for iterative sampling. Euler does allow the predicted direction to change as the image becomes cleaner.

1.6 · Generating images from noise

I started from five independent Gaussian noise tensors, using seeds 180-184, and integrated from $t = 0$ to $t = 1$ with 50 Euler updates. These samples use ordinary conditional prediction with “a high quality photo”, without CFG.



Seed 180



Seed 181



Seed 182



Seed 183



Seed 184

All five samples depict people, even though the prompt does not specify a subject. Several have visible artifacts: seed 180 has an incomplete face, seed 181 has distorted anatomy, and seed 183 has a rough, painterly appearance. Seeds 182 and 184 look more like conventional portraits. In these samples, the generic photo prompt favors people rather than producing a broad range of scene types.

2.1 · Classifier-free guidance

I evaluated the same state and time with the photo prompt and an empty prompt, then amplified the difference between their velocities. I used $w = 7$ and $h = 0.02$, reusing the exact five initial noises from Section 1.6.

$$v_{\text{CFG}} = v_c + w(v_c - v_u)$$



Seed 180, $w = 7$



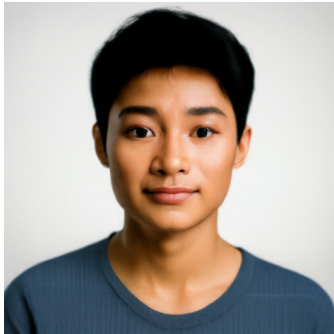
Seed 181, $w = 7$



Seed 182, $w = 7$



Seed 183, $w = 7$



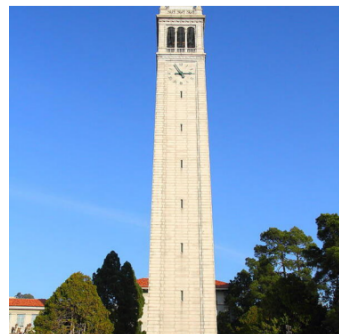
Seed 184, $w = 7$

Compared with the corresponding ordinary conditional samples above, the CFG images have clearer facial features, cleaner lighting, and fewer obvious anatomical artifacts. The incomplete face at seed 180 becomes a recognizable portrait. However, all five results remain portraits, and their compositions are more similar. Guidance improves the photographic appearance in this set, but it does not introduce more subject variety.

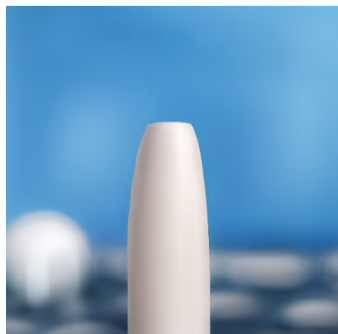
3.1 · Image-to-image editing

For each input, I constructed $x_{\text{start}} = (1 - t_{\text{start}})\epsilon + t_{\text{start}}x_1$, then sampled to $t = 1$ using “a high quality photo”, $w = 7$, and $h = 0.02$. Every input uses the same seed-180 noise and all seven starting times: 0.02, 0.04, 0.06, 0.10, 0.14, 0.25, 0.50.

Campanile



Original



$t = 0.02$



$t = 0.04$



$t = 0.06$



$t = 0.10$



$t = 0.14$



$t = 0.25$



$t = 0.50$

The low-time outputs preserve a pale vertical form against the blue sky, but replace the tower with simplified columns. The upper structure becomes more architectural at $t = 0.10$ and 0.14 . At $t = 0.25$ the belfry returns, and at $t = 0.50$ the clock and stone texture are much closer to the original.

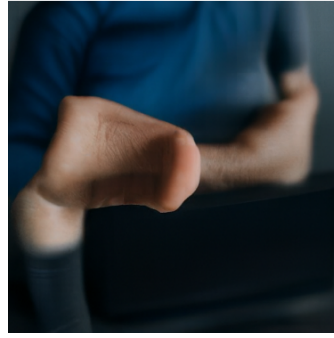
My bridge photograph 1



Original



$t = 0.02$



$t = 0.04$



$t = 0.06$



$t = 0.10$



$t = 0.14$



$t = 0.25$



$t = 0.50$

At $t = 0.02-0.06$, the results become unrelated subjects, including a person, a close-up hand, and furniture. The generic photo prompt does not require the output to remain a bridge. At $t = 0.10-0.14$, illuminated structures return but are interpreted as panels or buildings. At $t = 0.25$ and 0.50 , the bridge is recognizable again.

My bridge photograph 2



Original



$t = 0.02$



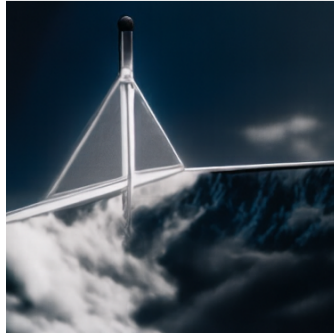
$t = 0.04$



$t = 0.06$



$t = 0.10$



$t = 0.14$



$t = 0.25$



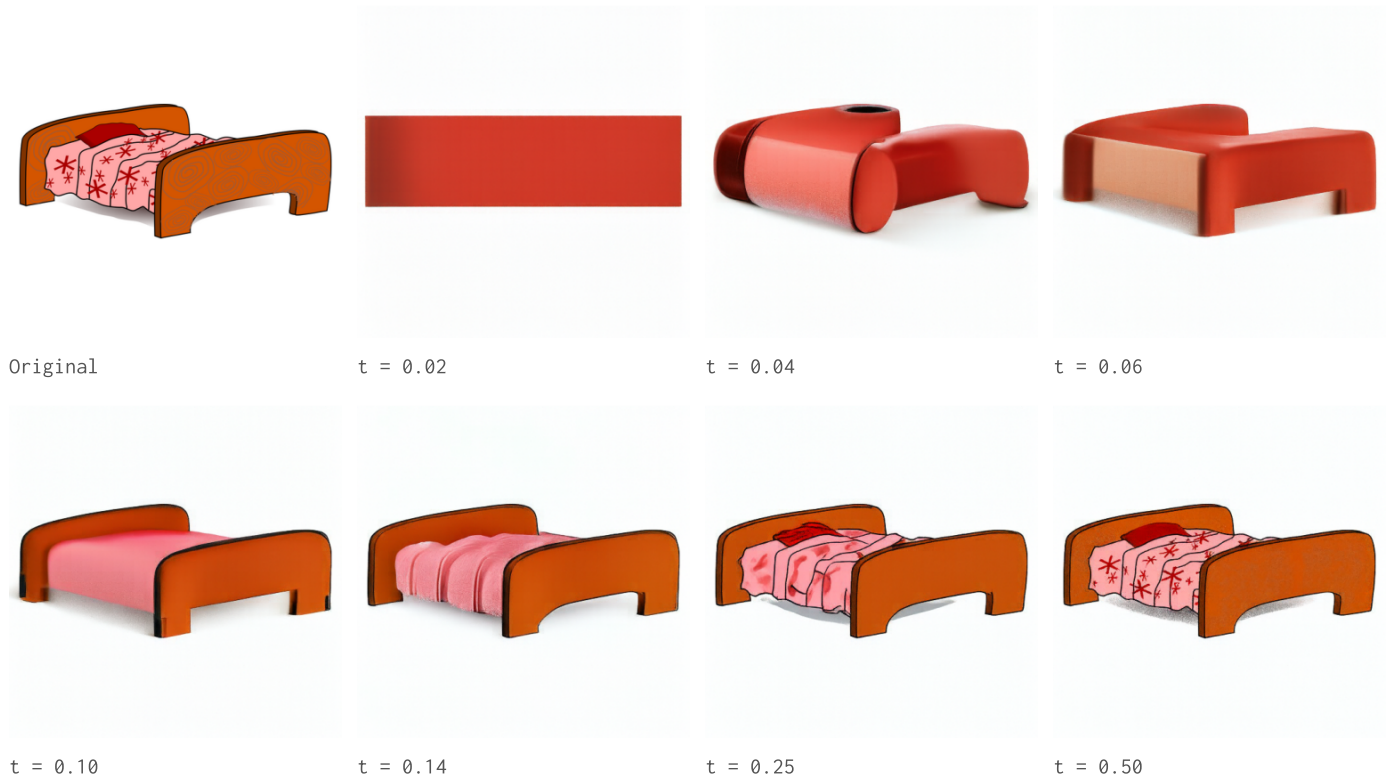
$t = 0.50$

The lowest starting times produce portraits instead of the bridge. Around $t = 0.10$ and 0.14 , the triangular tower shape returns in invented scenes. At $t = 0.25$ the model restores a bridge but changes the surrounding city and water. At $t = 0.50$ it preserves much more of the original tower and curved roadway.

3.1 · Hand-drawn and web images

I repeated the same seven-time sweep for one web illustration and two drawings of my own, using the same generic photo prompt and sampler settings.

Web image: illustrated bed



At $t = 0.02$, the bed collapses into a red rectangular shape. At $t = 0.04$ - 0.06 , the model makes rounded, three-dimensional furniture. At $t = 0.10$ - 0.14 , the headboard, footboard, and bedding are recognizable and have more photographic shading. At $t = 0.25$ - 0.50 , the outlines and patterned blanket increasingly resemble the original illustration.

Bed illustration by biswajyotim, Openclipart. Resized and centered on a white 512 × 512 canvas.

Drawing 1: person adjusting glasses



Original



$t = 0.02$



$t = 0.04$



$t = 0.06$



$t = 0.10$



$t = 0.14$



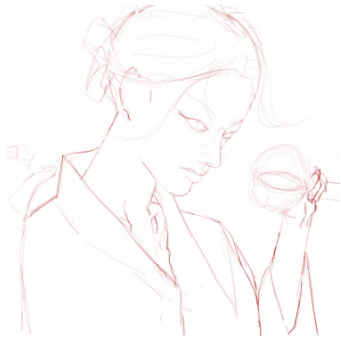
$t = 0.25$



$t = 0.50$

The lowest starting times produce different people, and $t = 0.06$ produces an isolated hand. At $t = 0.10$ - 0.14 , the model produces photographic people with hands near their faces, retaining part of the pose while changing the person's appearance. At $t = 0.25$ and 0.50 , the result remains closer to the original drawing style and composition.

Drawing 2: pale portrait sketch



Original



$t = 0.02$



$t = 0.04$



$t = 0.06$



$t = 0.10$



$t = 0.14$



$t = 0.25$



$t = 0.50$

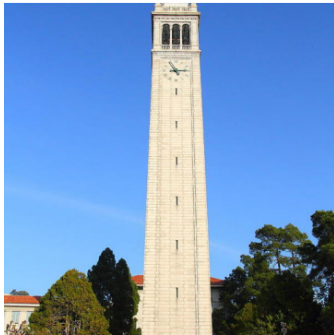
This drawing has much weaker contrast. At $t = 0.02$, the model produces a small white rectangular object, while several other low-time outputs are almost entirely white. A few lines survive at $t = 0.25$, and the portrait becomes recognizable again at $t = 0.50$. This example does not produce a convincing photographic conversion; its faint lines provide little structure after noise is introduced.

3.2 · Text-guided image editing

I reused the Campanile and my two bridge photographs with the same seven starting times, seed, guidance weight, and step size. The change is the conditional prompt, which now describes a specific edit.

Campanile → beer can

Prompt: “a photograph of a can of beer”



Original



$t = 0.02$



$t = 0.04$



$t = 0.06$



$t = 0.10$



$t = 0.14$



$t = 0.25$



$t = 0.50$

At low starting times, the tower becomes a recognizable beverage can with a metallic rim and printed label. At intermediate times, the can becomes unusually tall and starts to share the Campanile’s proportions. At $t = 0.25$, the tower has label-like markings near its top. At $t = 0.50$, the original architecture dominates and the beer-can edit is much weaker.

Bridge photograph 1 → mountains

Prompt: “mountains at night in deep blue sky”



Original



$t = 0.02$



$t = 0.04$



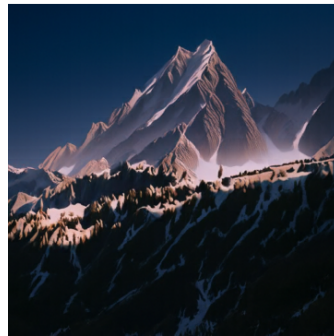
$t = 0.06$



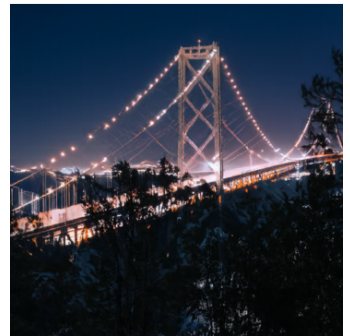
$t = 0.10$



$t = 0.14$



$t = 0.25$



$t = 0.50$

At $t = 0.02$ - 0.14 , mountains replace the bridge while retaining a dark blue palette. At $t = 0.25$, the lit bridge shapes are reinterpreted as a mountain ridge above a dark foreground. At $t = 0.50$, the bridge returns almost unchanged. The intermediate output connects the source composition with the requested subject more strongly than either endpoint.

Bridge photograph 2 → glowing snake

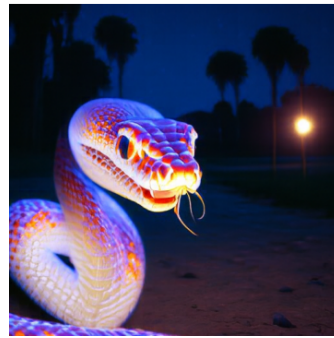
Prompt: “a glowing bright snake at night”



Original



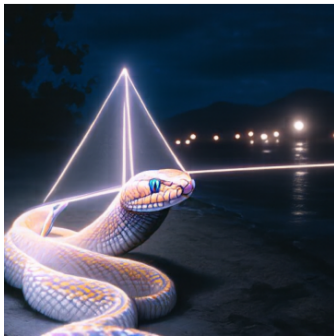
$t = 0.02$



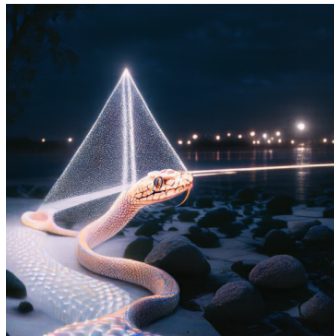
$t = 0.04$



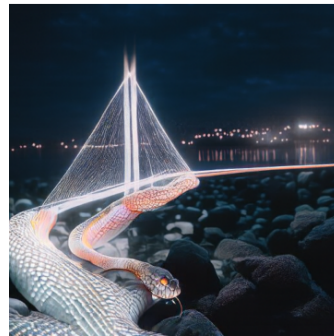
$t = 0.06$



$t = 0.10$



$t = 0.14$



$t = 0.25$



$t = 0.50$

The lowest times produce brightly colored snakes with very little bridge structure remaining. At $t = 0.10$ - 0.25 , the snake and bridge coexist: the bridge tower survives in the background while the curving foreground becomes a snake. At $t = 0.50$, the roadway and tower are much closer to the original, and the requested snake is no longer clearly visible.

3.3 · Visual anagrams

At each step, I computed one CFG velocity for the upright image and another for its 180° rotation using the second prompt. I rotated the second velocity back before averaging the two, so both updates refer to the same pixel positions. I sampled from noise with $w = 7$ and $h = 0.02$.

$$v = [v_{\text{CFG}}(x, t, c_a) + R(v_{\text{CFG}}(R(x), t, c_b))] / 2$$

Illusion 1 · seed 180



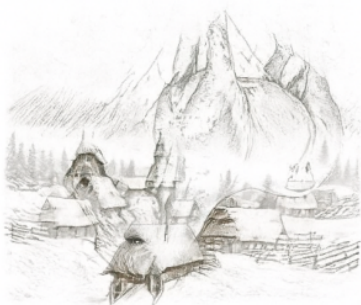
Upright: an oil painting of an old man



Rotated 180°: an oil painting of people around a campfire

Upright, the image forms an old man's face with a bright forehead and dark eye sockets. Rotated, the bright forehead becomes the fire, and figures appear around it. The shared orange lighting and dark background help both readings coexist, although small faces remain visible inside the upright portrait.

Illusion 2 · seed 181



Upright: a lithograph of a snowy mountain village



Rotated 180°: a lithograph of a horse

In the upright view, pointed contours form snowy peaks and roofs. Rotating the same image reveals a horse's head and body. The horse is easier to recognize than the village, so the two interpretations are not equally strong. The pencil-like texture allows the same edges to serve both subjects.

3.4 · Hybrid images

For each Euler update, I combined the low-frequency part of the distant prompt's CFG velocity with the high-frequency part of the close prompt's CFG velocity. The low-pass filter uses a 265×265 Gaussian kernel with $\sigma = 16$. Both predictions use the same image and time, with $w = 7$ and $h = 0.02$.

$$v = G_{16}(v_{\text{distant}}) + v_{\text{close}} - G_{16}(v_{\text{close}})$$

Hybrid 1 · seed 180



Close: a lithograph of a waterfall



Distant, 64×64 : a lithograph of a skull

The waterfall is clear at full size, with vertical streams, dark rocks, and trees. In the reduced view, the broad dark and pale regions become more apparent, but I do not see a clearly defined skull. The close-view subject is stronger than the intended distant-view subject, so this is a limited hybrid rather than a clean perceptual switch.

Hybrid 2 · seed 181



Close: a lithograph of a forest



Distant, 64×64 : a lithograph of a cat

The full-size image clearly shows tree trunks and branches. At reduced size, the grouped canopies and pale regions above them suggest a face-like arrangement, but the cat is ambiguous and the forest remains visible. This result shows that separating the velocity frequencies does not guarantee two equally recognizable subjects.

What I learned

The starting time controls how much the model can rely on the original image. Small starting times allow stronger prompt-driven changes, but they can also remove the subject completely. Larger starting times preserve the input but leave less room for editing. The drawings and hybrids also showed that a correct sampling procedure does not guarantee a convincing visual effect: input contrast, prompt choice, and the compatibility of the two subjects still matter.

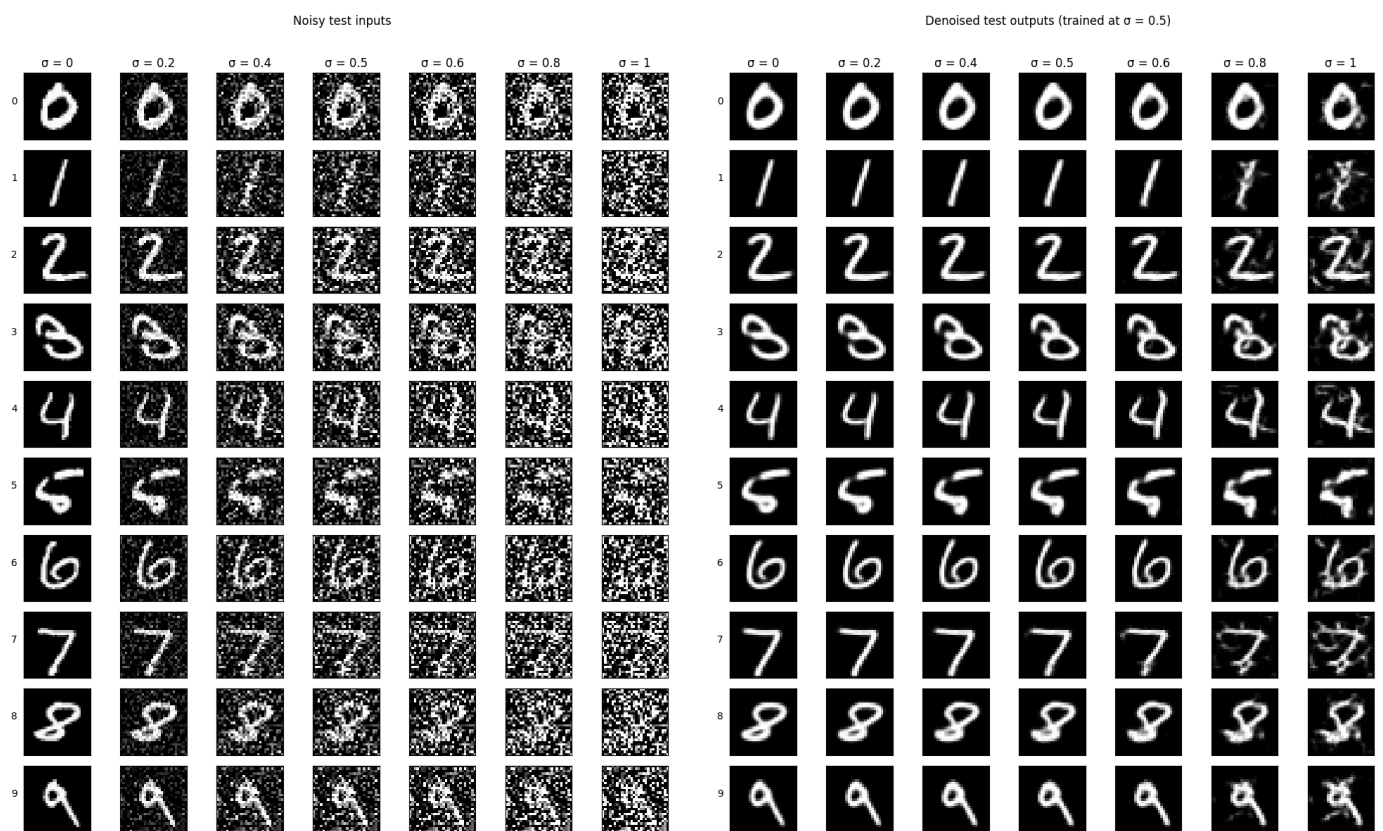
Part B · Flow Matching from Scratch

I trained UNets on the 60,000 MNIST training images, with pixels in $[0, 1]$. Test images are used only for evaluation. First I trained a single-step denoiser, then time-conditioned and class-conditioned models that generate digits by following a learned flow.

The UNet uses convolution, batch normalization, and GELU blocks. Its spatial resolution follows $28 \rightarrow 14 \rightarrow 7 \rightarrow 1 \rightarrow 7 \rightarrow 14 \rightarrow 28$, with concatenated skip connections from the encoder to the decoder. The final convolution produces one grayscale channel.

1.2 & 1.2.2 · Noise levels and OOD testing

For each digit, I keep the clean test image and Gaussian noise tensor fixed while changing σ . The left grid shows $x + \sigma \epsilon$; the right shows the outputs of the denoiser trained at $\sigma = 0.5$ for five epochs. Rows correspond to digits 0-9, and both grids use the same row and column order.



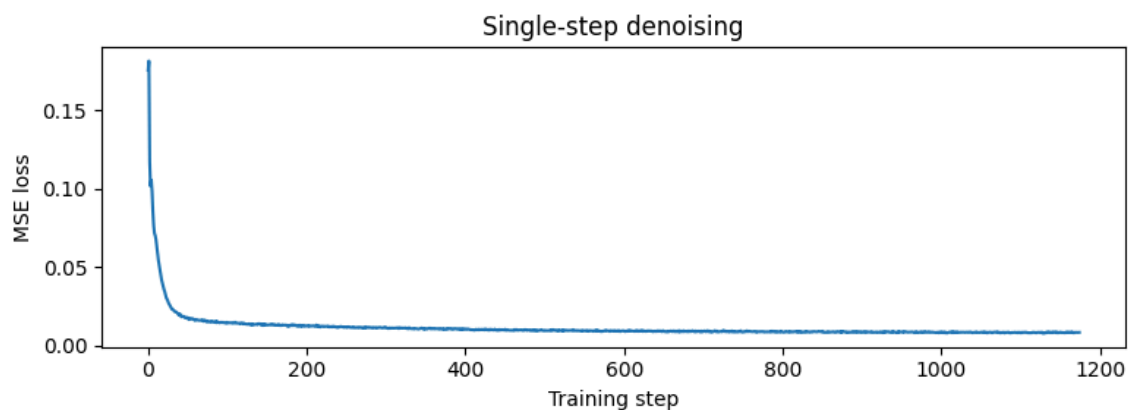
Noisy inputs · $\sigma = 0, 0.2, 0.4, 0.5, 0.6, 0.8, 1.0$

Denoised outputs · the same test digits at every σ

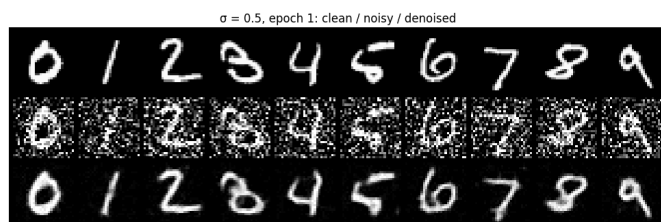
The model handles nearby noise levels reasonably well, although it smooths even clean inputs. At $\sigma = 0.8$ and 1.0 , strokes become distorted and residual noise appears. Training at one noise level does not make the model reliable at every noise level.

1.2.1 · Training a single-step denoiser

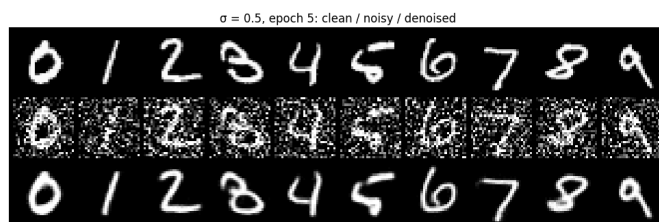
The input is $x + 0.5\epsilon$ and the target is x . I use $D = 128$, batch size 256, Adam with learning rate 10^{-4} , and five epochs. The training set is shuffled, and fresh noise is sampled for each batch.



MSE loss across all 1,175 training steps



Epoch 1 · clean / noisy / denoised

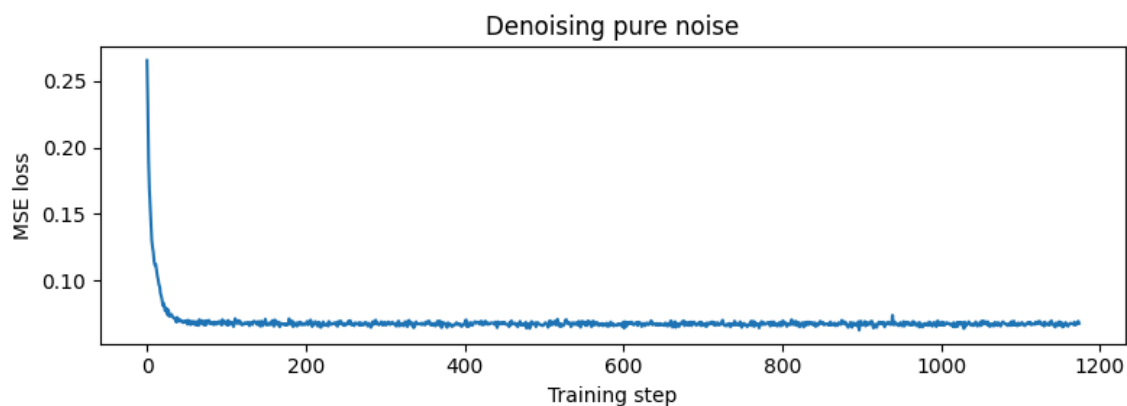


Epoch 5 · clean / noisy / denoised

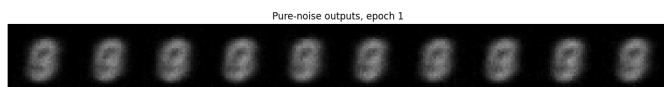
The first epoch already recovers most digit shapes. By epoch 5, the outputs have cleaner, more continuous strokes, while some fine details remain smoothed. Over all 10,000 test images, pixel MSE falls from 0.25007 for the noisy input to 0.00828 after denoising.

1.2.3 · Denoising pure noise

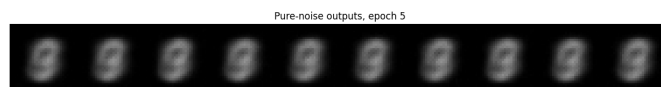
I train a fresh $D = 128$ UNet with the same batch size, optimizer, learning rate, and five-epoch schedule. This time, each input is independent Gaussian noise ε and its target is a training digit. The two sample grids reuse the same held-out noise inputs.



Pure-noise training loss · all 1,175 steps



Epoch 1 · ten independent noise inputs



Epoch 5 · the same ten noise inputs

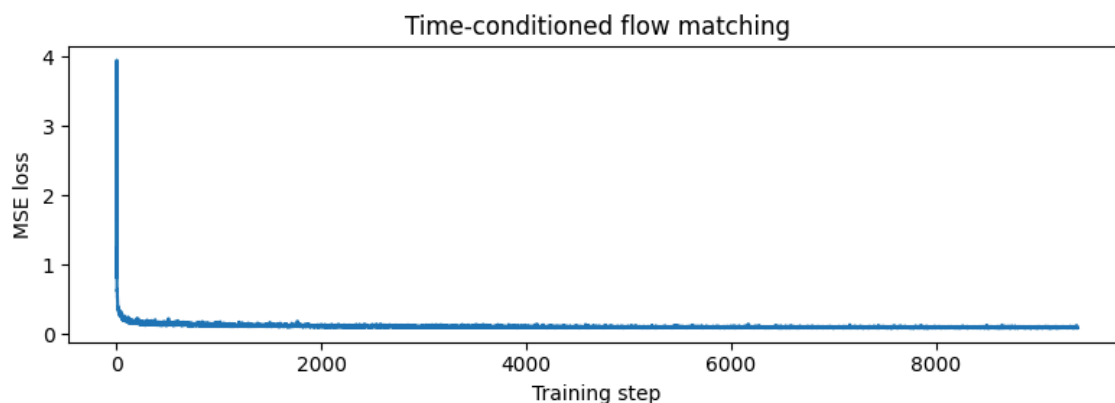
The outputs are nearly identical blurry shapes, with a bright oval center and faint overlapping strokes. They resemble an average of several digits rather than distinct samples of 0-9. Because the noise and target are independent, the input gives no information about which digit to predict. The MSE-optimal prediction is $E[x \mid \varepsilon] = E[x]$, the pixelwise mean of the training images. Averaging the different digit shapes explains both the blur and the lack of variation.

2.1-2.2 · Time-conditioned flow matching

I sample x_0 from Gaussian noise, x_1 from MNIST, and t uniformly from $[0, 1]$. The network learns the velocity along a straight path between them.

$$x_t = (1 - t)x_0 + tx_1 \cdot \text{target velocity} = x_1 - x_0$$

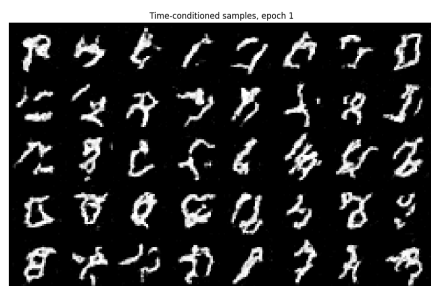
Two fully connected blocks embed t and multiply the unflattened bottleneck and first upsampling block. I use $D = 64$, batch size 64, ten epochs, and Adam starting at 10^{-2} . After each epoch, the learning rate is multiplied by $0.1^{1/10}$, reaching 10^{-3} after the final scheduler step. Both noise and time are resampled for each batch.



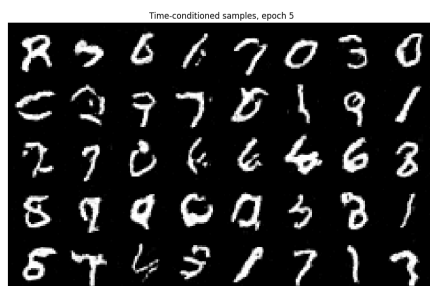
Time-conditioned velocity MSE · all 9,380 training steps

2.3 · Sampling without class labels

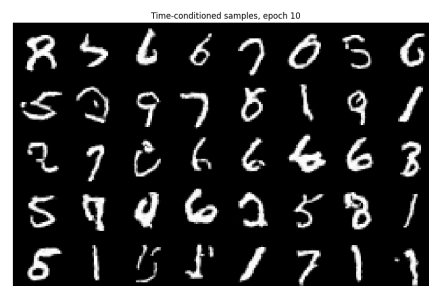
Each checkpoint starts from the same 40 Gaussian noise images (seed 191). I take 50 Euler steps, $x \leftarrow x + u_0(x, t)/50$, from $t = 0$ to 1.



Epoch 1



Epoch 5



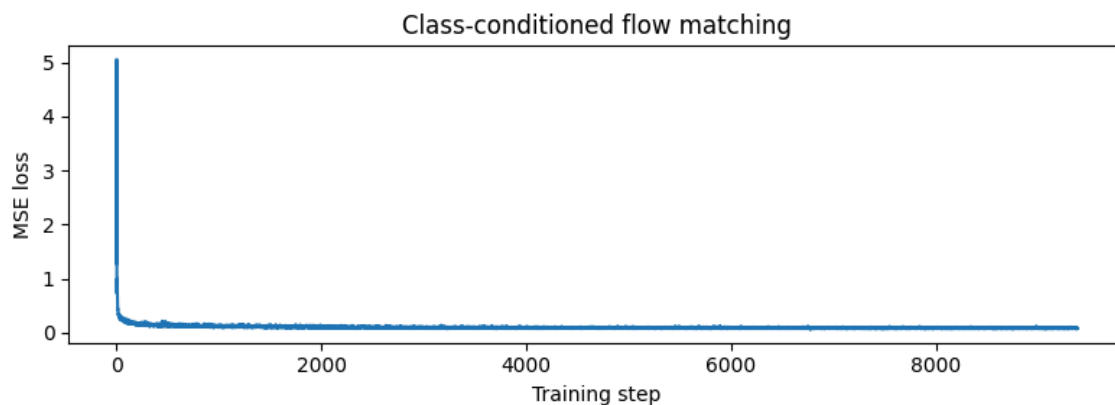
Epoch 10

Epoch 1 produces broken and ambiguous strokes. By epoch 5, many recognizable digits appear; epoch 10 cleans up more of the outlines. Some samples still have extra strokes or unclear identities. The model produces varied digits, but it has no label telling it which digit to generate.

2.4-2.5 · Adding class conditioning

I embed a ten-dimensional one-hot digit label with two additional fully connected blocks. At the bottleneck and first upsampling stage, conditioning becomes $c_1 \cdot \text{unflatten} + t_1$ and $c_2 \cdot \text{up1} + t_2$. For 10% of training examples, the label vector is set to zero so the same network learns an unconditional prediction.

The architecture width, batch size, ten-epoch schedule, Adam learning rate, and exponential decay match the time-conditioned experiment. The training objective still predicts $x_1 - x_0$.

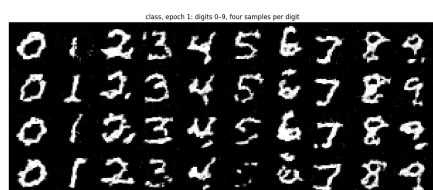


Class-conditioned velocity MSE · all 9,380 training steps

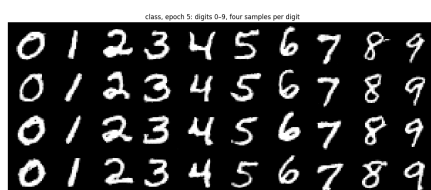
2.6 · Class-guided sampling

Each row requests digits 0-9, giving four instances of every digit. All checkpoints use the same initial noise (seed 201), 300 Euler steps, and guidance scale $\gamma = 5$.

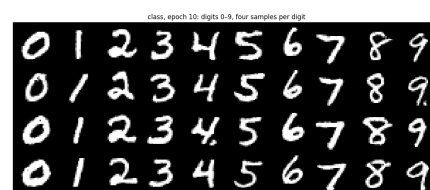
$$u_{\text{guided}} = u_{\text{uncond}} + 5(u_{\text{cond}} - u_{\text{uncond}})$$



Epoch 1



Epoch 5

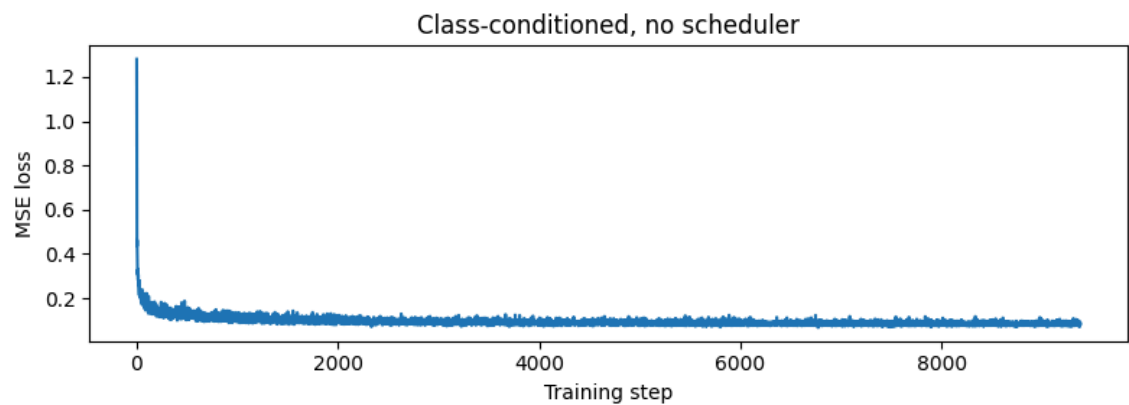


Epoch 10

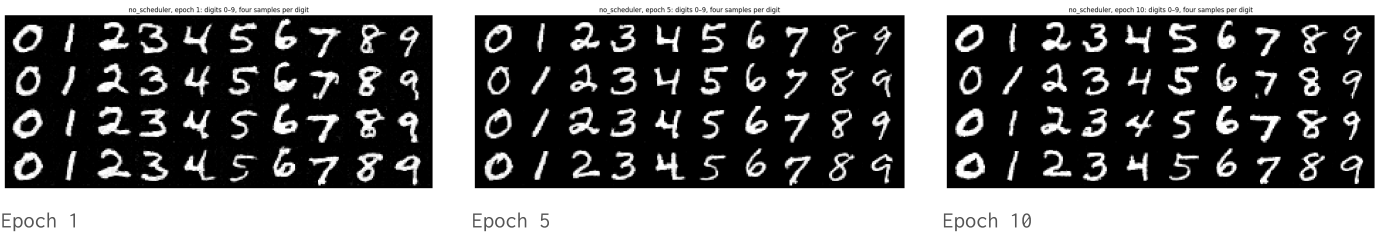
The requested classes are visible even at epoch 1, although several strokes are noisy or incomplete. Epochs 5 and 10 give cleaner, more consistent digits in the requested columns. Class conditioning provides much more control than the time-only model.

2.6 · Removing the scheduler

I train another model from scratch with a constant Adam learning rate of 10^{-3} . This lower fixed rate replaces the scheduled run's larger initial rate of 10^{-2} . The initialization seed, data order, architecture, batch size, label dropout, and ten-epoch budget are kept the same. Sampling again uses seed 201, 300 Euler steps, and $\gamma = 5$.



No scheduler · velocity MSE across all 9,380 steps



Class-conditioned training	Epoch 10 mean training MSE
Exponential decay, $10^{-2} \rightarrow 10^{-3}$	0.084764
Constant 10^{-3}	0.084282

The constant-rate model already produces cleaner digits at epoch 1. By epoch 10, both runs produce clear 0-9 grids with similar visual quality, and their final training losses are close. For this run, a smaller constant learning rate maintains comparable results without a scheduler. This is a visual and training-loss comparison, not a test-set measure of generative quality.