DDIM from First Principles

A Slow, Intuitive Companion to Section 3 of *Step-by-Step Diffusion*

Goal of these notes

These notes are able to read entirely independently, and function like a stand-alone tutorial. However, they evolved from their initial form as a companion guide to Section 3 of "Step-By-Step Diffusion: An Elementary Tutorial" by Nakkiran et al, and may reference that work as "the tutorial" at times.

These notes explain the DDIM construction; they serve to not merely state the DDIM update, but to explain why a deterministic reverse process can possibly work after we spent so much effort motivating a *stochastic* reverse process for DDPM.
The central conceptual distinction is:

DDPM tries to reproduce a reverse conditional distribution.

DDIM only needs to transport one marginal distribution into another.

Those are different requirements. Understanding that distinction is the key to DDIM.

Contents

1 What We Already Know from DDPM

We begin with the simplified Gaussian diffusion process used throughout the tutorial. Let

$$x_0 \sim p^\star,$$

where $p^\star$ is the data distribution we ultimately want to sample from.

The forward process gradually corrupts x_0 by adding Gaussian noise. With continuous-valued time $t \in [0, 1]$ and discretization step Δt,

$$x_{t+\Delta t} = x_t + \eta_t, \qquad \eta_t \sim \mathcal{N}(0, \sigma_q^2 \Delta t I).$$

Because independent Gaussian increments add to another Gaussian, we do not actually need to simulate all the intermediate steps to sample x_t. After elapsed time t,

$$x_t \mid x_0 \sim \mathcal{N}(x_0, \sigma_q^2 t I).$$

Define

$$\sigma_t = \sigma_q \sqrt{t}.$$

Then the same statement can be written

$$\boxed{x_t = x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).}$$

This equation will be extremely important.

1.1 Marginals versus trajectories

There are two different kinds of probabilistic objects here, and DDIM depends on keeping them separate.

First, for any particular time t, we have the conditional marginal

$$p(x_t \mid x_0).$$

It tells us the distribution of the noisy sample at time t, given the original sample.

Second, the complete forward random walk defines a much larger joint distribution,

$$p(x_{\Delta t}, x_{2\Delta t}, \ldots, x_1 \mid x_0).$$

This joint distribution tells us not only what x_t looks like at each time, but also *how the random variables at different times are related to one another.*

These are not the same information.

Simple analogy

Suppose I tell you that two random variables A and B are both standard normal:

$$A \sim \mathcal{N}(0,1), \qquad B \sim \mathcal{N}(0,1).$$

That does not tell you their joint distribution.
For example, all three of the following are possible:

$$B = A, \qquad B = -A, \qquad B \text{ independent of } A.$$

In every case the individual marginals are still $\mathcal{N}(0,1)$, but the relationship between A and B is completely different.

The same principle applies to diffusion. Knowing every distribution $p_t(x)$ does not uniquely determine how a particular particle at time t should be associated with a particle at time $t - \Delta t$. Such a choice of joint relationships is often called a *coupling*.

2 What DDPM Chooses to Reverse

DDPM starts from the particular Markov forward process

$$x_0 \to x_{\Delta t} \to x_{2\Delta t} \to \cdots \to x_1.$$

Because that forward process gives a particular joint distribution, it also defines the conditional reverse distribution

$$p(x_{t-\Delta t} \mid x_t).$$

The ideal DDPM reverse step is:

$$\boxed{x_{t-\Delta t} \sim p(x_{t-\Delta t} \mid x_t).}$$

For sufficiently small Δt, the tutorial argues that this conditional is approximately Gaussian. Its mean is

$$\mu_{t-\Delta t}(x_t) = \mathbb{E}[x_{t-\Delta t} \mid x_t].$$

Thus the simplified DDPM update is approximately

$$x_{t-\Delta t} = \mu_{t-\Delta t}(x_t) + \mathcal{N}(0, \sigma_q^2 \Delta t I).$$

Notice the two pieces:

1. move toward the conditional mean;
2. inject fresh random noise.

The noise is not an arbitrary imperfection. DDPM is trying to sample a *distribution* $p(x_{t-\Delta t} \mid x_t)$, not merely return its mean. If that conditional has nonzero variance, sampling it requires randomness.

3 The Question That Leads to DDIM

Before deriving DDIM, we need to be extremely precise about three different objects that will appear throughout this section:

1. the **true conditional-mean function**

$$\mu_{t-\Delta t}(x_t) = \mathbb{E}[x_{t-\Delta t} \mid x_t],$$

2. the **neural network**

$$f_\theta(x_t, t),$$

which is trained to approximate some conditional expectation such as $\mu_{t-\Delta t}(x_t)$, and

3. the **DDIM transport map**

$$F_t(x_t),$$

which uses the learned information to actually move a sample from time t toward an earlier time.

These are closely related, but they are *not the same object*.

Understanding what information each object receives—particularly information about the timestep—will resolve several apparent ambiguities in the DDIM construction.

3.1 First: What Exactly Is a Timestep?

We normalize diffusion time so that

$$t \in [0, 1].$$

The clean data distribution occurs at

$$t = 0,$$

while the maximally noised distribution occurs at

$$t = 1.$$

Suppose, for the moment, that we discretize this interval into T equally spaced steps. Then

$$\boxed{\Delta t = \frac{1}{T}.}$$

The available timesteps are therefore

$$0, \Delta t, 2\Delta t, \ldots, 1.$$

For example, if

$$T = 1000,$$

then

$$\Delta t = 0.001,$$

and the diffusion times are

$$0,\ 0.001,\ 0.002, \ldots, 0.999,\ 1.$$

The forward diffusion transition is

$$x_{t+\Delta t} = x_t + \eta_t, \qquad \eta_t \sim \mathcal{N}(0, \sigma_q^2 \Delta t I).$$

Thus one forward step takes us from noise level t to noise level

$$t + \Delta t.$$

Generation runs in the opposite direction:

$$1 \to 1 - \Delta t \to 1 - 2\Delta t \to \cdots \to 0.$$

Therefore, when we write

$$x_{t-\Delta t},$$

we mean "the state one timestep earlier than x_t."

3.2 The Distribution at a Particular Time

Although the forward process can be generated one small step at a time, Gaussian noise allows us to write the state at any time directly:

$$\boxed{x_t = x_0 + \sigma_t \epsilon, \qquad \sigma_t = \sigma_q \sqrt{t}, \qquad \epsilon \sim \mathcal{N}(0, I).}$$

Therefore,

$$x_t \mid x_0 \sim \mathcal{N}(x_0, \sigma_t^2 I).$$

Notice that this equation depends on the *absolute time* t, not directly on the discretization step Δt.

For example, if

$$t = 0.7,$$

we can generate $x_{0.7}$ directly from x_0:

$$x_{0.7} = x_0 + \sigma_q \sqrt{0.7}\, \epsilon.$$

We do not need to generate

$$x_{\Delta t}, x_{2\Delta t}, \ldots, x_{0.7-\Delta t}$$

first.

This distinction between the absolute noise level t and the numerical step size Δt will become very important.

3.3 What Does the Ideal Function μ Mean?

For the moment, suppose we have fixed a particular discretization Δt.

At reverse time t, DDPM is interested in the conditional distribution

$$p(x_{t-\Delta t} \mid x_t).$$

Its mean is

$$\boxed{\mu_{t-\Delta t}(x_t) := \mathbb{E}[x_{t-\Delta t} \mid x_t].}$$

This is a mathematical property of the true probability distribution.

It is *not* itself a neural network.

If we somehow knew the complete data distribution $p^\star$, we could in principle calculate this conditional expectation exactly.

In reality, we do not know $p^\star$, so we train a neural network to approximate it.

3.4 What Does the Neural Network Know?

In the tutorial's initial formulation, the neural network receives

$$(x_t, t)$$

as its input:

$$f_\theta(x_t, t).$$

Its goal is to approximate

$$\boxed{f_\theta(x_t, t) \approx \mathbb{E}[x_{t-\Delta t} \mid x_t] = \mu_{t-\Delta t}(x_t).}$$

The input x_t tells the network the current noisy sample.

The input t tells the network the current noise level.

For example, the same numerical vector x should be interpreted very differently if it occurs at

$$t = 0.05$$

versus

$$t = 0.95.$$

At $t = 0.05$, it should contain relatively little noise.

At $t = 0.95$, it should contain much more noise.

Therefore the network needs t.

However, notice something subtle:

$$f_\theta(x_t, t)$$

does *not* receive Δt as an explicit input.

How, then, does it know whether it should predict

$$x_{t-\Delta t},$$

or

$$x_{t-5\Delta t},$$

or

$$x_{t-10\Delta t}?$$

The answer is:

$$\boxed{\text{In this initial formulation, it does not.}}$$

The training procedure has already fixed what prediction the network is supposed to make. If training uses a fixed step size Δt, then the meaning of

$$f_\theta(x_t, t)$$

is implicitly

$$\boxed{f_\theta(x_t, t) \approx \mathbb{E}[x_{t-\Delta t} \mid x_t].}$$

Thus Δt is effectively built into the *definition of the training target*, even though it is not explicitly passed into the network.

3.5 A Concrete Example

Suppose we choose

$$T = 1000, \qquad \Delta t = 0.001.$$

Consider a training example at

$$t = 0.700.$$

The network receives

$$f_\theta(x_{0.700}, 0.700).$$

Under the tutorial's initial previous-state prediction formulation, the target is

$$x_{0.699}.$$

More precisely, because squared-error regression learns a conditional expectation, the ideal network output is

$$f_\theta(x_{0.700}, 0.700) = \mathbb{E}[x_{0.699} \mid x_{0.700}].$$

Now suppose that after training we suddenly ask this same network for

$$\mathbb{E}[x_{0.690} \mid x_{0.700}].$$

That corresponds to moving backward by

$$10\Delta t.$$

The network has no explicit input telling it that we have changed the destination from 0.699 to 0.690.

Both requests would give it exactly the same inputs:

$$(x_{0.700}, 0.700).$$

Therefore it cannot possibly distinguish between the requests.

This means that, for this particular parameterization,

$$\boxed{f_\theta(x_t, t) \text{ is tied to the } \Delta t \text{ used to define its training target.}}$$

If we wanted a neural network that directly predicted arbitrary destination times, we would need something like

$$f_\theta(x_t, t, s),$$

where $s < t$ specifies the destination timestep.

For example,

$$f_\theta(x_{0.7}, 0.7, 0.699)$$

and

$$f_\theta(x_{0.7}, 0.7, 0.69)$$

would then be distinguishable requests.

This, however, is *not* the strategy we ultimately need.

3.6 Why Predicting x_0 Solves This Problem

The tutorial subsequently introduces a much more useful parameterization.

Instead of training

$$f_\theta(x_t, t) \approx \mathbb{E}[x_{t-\Delta t} \mid x_t],$$

we can train

$$\boxed{f_\theta(x_t, t) \approx \mathbb{E}[x_0 \mid x_t].}$$

This changes the situation substantially.

Now the question answered by the network is always:

> Given my current noisy state x_t, what does it tell me about the original clean state x_0?

There is no destination timestep in this question.

For example, at $t = 0.7$, the network computes

$$f_\theta(x_{0.7}, 0.7) \approx \mathbb{E}[x_0 \mid x_{0.7}].$$

This prediction means the same thing whether the sampler subsequently decides to move to

$$0.699, \qquad 0.69, \qquad \text{or some other earlier noise level.}$$

The *neural network* provides information about the current state.

The *sampler* decides how that information should be used to move to a chosen destination time.

This separation is extremely important:

$$\boxed{\underbrace{f_\theta(x_t, t)}_{\text{What does the current state tell us?}} + \underbrace{\text{chosen destination time}}_{\text{Where do we want to go?}} \longrightarrow \underbrace{\text{sampling update}}_{\text{How far do we move?}}.}$$

3.7 The Neural Network Is Not the DDIM Map

We are now ready to distinguish f_θ from F_t.

The neural network is a learned predictor. In the original previous-state parameterization,

$$f_\theta(x_t, t) \approx \mu_{t-\Delta t}(x_t) = \mathbb{E}[x_{t-\Delta t} \mid x_t].$$

However, from this point onward we will update notation to reflect that we are predicting x_0 rather than $x_{t-\Delta t}$. We therefore define

$$\boxed{\mu_0(x_t) := \mathbb{E}[x_0 \mid x_t],}$$

and train the neural network so that

$$\boxed{f_\theta(x_t, t) \approx \mu_0(x_t) = \mathbb{E}[x_0 \mid x_t].}$$

Thus, f_θ estimates the conditional mean of the original clean sample x_0, rather than the conditional mean of the immediately preceding state $x_{t-\Delta t}$.

The DDIM map, by contrast, is the actual rule that updates the sample. The tutorial originally writes this map in terms of the previous-state conditional mean,

$$F_t(x_t) = x_t + \lambda \left(\mu_{t-\Delta t}(x_t) - x_t\right),$$

where

$$\lambda = \frac{\sigma_t}{\sigma_{t-\Delta t} + \sigma_t}.$$

However, we have chosen to parameterize our neural network as a predictor of the clean state,

$$f_\theta(x_t, t) \approx \mu_0(x_t) = \mathbb{E}[x_0 \mid x_t].$$

We therefore use the relationship

$$\mu_{t-\Delta t}(x_t) = \frac{\Delta t}{t}\mu_0(x_t) + \left(1 - \frac{\Delta t}{t}\right) x_t.$$

Substituting this into the original DDIM update gives

$$\boxed{F_t(x_t) = x_t + \lambda \frac{\Delta t}{t} \left(\mu_0(x_t) - x_t\right).}$$

Since the neural network approximates $\mu_0(x_t)$, the practical update is

$$\boxed{F_t(x_t) \approx x_t + \lambda \frac{\Delta t}{t} \left(f_\theta(x_t, t) - x_t\right).}$$

Thus, the neural network does not directly predict the next state $x_{t-\Delta t}$. It predicts the conditional mean of the clean state x_0. The DDIM sampler then combines this prediction with the known current time t, step size Δt, and noise schedule to determine how far to move toward the predicted clean state.

Thus, schematically,

neural network prediction + known diffusion schedule + chosen timestep $\longrightarrow$ DDIM update.

It would therefore be incorrect to identify

$$f_\theta$$

with

$$F_t.$$

They play different roles.

3.8 Why Does Deterministic DDIM Not Produce Blurry Average Samples?

At first, the deterministic nature of DDIM may seem to contradict an important lesson from DDPM. Recall that the conditional mean

$$\mu_0(x_t) = \mathbb{E}[x_0 \mid x_t]$$

need not itself be a realistic sample. If a noisy state x_t could plausibly have arisen from several different clean samples, then $\mu_0(x_t)$ averages over these possibilities. For images, such an average may be blurry.

This is one reason why DDPM does not simply perform the update

$$x_{t-\Delta t} = \mathbb{E}[x_{t-\Delta t} \mid x_t].$$

Instead, DDPM samples around the conditional mean by adding Gaussian noise, thereby representing the full reverse conditional distribution rather than collapsing it to its mean.

It may therefore seem surprising that DDIM can remove the per-step randomness without suffering from the same averaging problem. The key is that DDIM does *not* replace the current sample with the predicted conditional mean. In our x_0-prediction notation, its update has the form

$$x_{t-\Delta t} = x_t + \alpha_t \left(\mu_0(x_t) - x_t\right),$$

where, for the simplified diffusion considered here,

$$\alpha_t = \lambda \frac{\Delta t}{t}.$$

Thus,

$$\mu_0(x_t) - x_t$$

is used primarily to determine a *direction of motion*. The sample moves only an appropriate distance in this direction; it is not replaced by the potentially blurry conditional mean $\mu_0(x_t)$.

Another important point is that DDIM has not eliminated randomness from generation altogether. Generation still begins by sampling

$$x_T \sim \mathcal{N}(0, \sigma_T^2 I).$$

After this initial sample has been drawn, the DDIM trajectory is deterministic. Different initial noise samples therefore follow different trajectories through the learned velocity field:

$$x_T^{(1)} \to x_{T-\Delta t}^{(1)} \to \cdots \to x_0^{(1)},$$

$$x_T^{(2)} \to x_{T-\Delta t}^{(2)} \to \cdots \to x_0^{(2)}.$$

The diversity of generated samples can therefore come from the initial randomness x_T, rather than from injecting additional randomness at every reverse step.

To see how this avoids averaging, consider the simple clean distribution

$$p_0 = \frac{1}{2}\delta_{-a} + \frac{1}{2}\delta_{+a}.$$

At a noisy intermediate time, a point exactly between the two modes may have

$$\mathbb{E}[x_0 \mid x_t = 0] = 0,$$

which is the average of the two possible clean states. However, a sample lying slightly to the right of zero generally has

$$\mathbb{E}[x_0 \mid x_t] > 0,$$

so its DDIM velocity points toward the positive mode. A sample lying slightly to the left has the opposite behavior. As the samples move, their conditional expectations change, and the deterministic flow can separate probability mass toward the two modes rather than collapsing all samples onto their average.

The important distinction is therefore

using a conditional mean as the output $\neq$ using a conditional mean to define a velocity field.

DDPM uses the learned conditional mean to parameterize a stochastic reverse transition and introduces fresh noise at each step. DDIM instead uses the same type of denoising information to construct a deterministic transport of the randomness already present in x_T. Consequently, the conditional mean itself may be blurry without the final samples produced by the DDIM flow being blurry.

3.9 How a Changing Conditional Mean Selects Individual Modes

The previous discussion raises a further question. If DDIM repeatedly moves in the direction of

$$\mu_0(x_t) = \mathbb{E}[x_0 \mid x_t],$$

and this conditional expectation may itself be an average of several possible clean samples, why does the trajectory not simply converge to that average?

The key is that the conditional mean is *not a fixed target*. It is a function of the current state:

$$\boxed{\mu_0(x_t) = \mathbb{E}[x_0 \mid x_t].}$$

After DDIM moves from x_t to a new state x_s, the network is evaluated again, producing

$$\mu_0(x_s) = \mathbb{E}[x_0 \mid x_s].$$

In general,

$$\mu_0(x_s) \neq \mu_0(x_t).$$

Thus, DDIM does not repeatedly move toward one fixed average. It follows a vector field whose direction changes as the sample moves:

$$x_t \xrightarrow{\mu_0(x_t)} x_s \xrightarrow{\mu_0(x_s)} x_r \xrightarrow{\mu_0(x_r)} \cdots .$$

Success case: resolving an ambiguous sample. Consider the simple two-mode clean distribution

$$p_0 = \frac{1}{2}\delta_{-1} + \frac{1}{2}\delta_{+1}.$$

Suppose a noisy sample x_t lies slightly on the positive side of the midpoint. Perhaps its posterior probabilities are

$$P(x_0 = +1 \mid x_t) = 0.55, \qquad P(x_0 = -1 \mid x_t) = 0.45.$$

The clean-state conditional mean is then

$$\mu_0(x_t) = 0.55(+1) + 0.45(-1) = 0.10.$$

The value 0.10 is not itself a valid clean sample; it is an average of the two possible modes. However, DDIM does not simply replace x_t by 0.10. The conditional mean instead contributes to the local direction of the deterministic transport.

After moving in this direction, the new state may provide stronger evidence for the positive mode. For example,

$$P(x_0 = +1 \mid x_s) = 0.65$$

would give

$$\mu_0(x_s) = 0.65(+1) + 0.35(-1) = 0.30.$$

After another step, the posterior might become

$$P(x_0 = +1 \mid x_r) = 0.80,$$

giving

$$\mu_0(x_r) = 0.60.$$

The sequence of conditional means might therefore evolve schematically as

$$0.10 \to 0.30 \to 0.60 \to 0.85 \to 0.97 \to 1.$$

Thus, an initially ambiguous conditional mean can become increasingly mode-specific as the trajectory evolves. A sample initially lying slightly on the negative side can analogously flow toward -1.

The initial randomness

$$x_T \sim \mathcal{N}(0, \sigma_T^2 I)$$

therefore plays an essential role. Different initial noise samples begin at different locations in the deterministic vector field and can consequently follow different streamlines toward different clean modes. DDIM does not need to inject new randomness at every step because randomness is already present in the initial condition.

Edge case: exact symmetry. There is, however, an instructive edge case. Suppose the same two-mode distribution is perfectly symmetric and the trajectory lies exactly at the midpoint,

$$x_t = 0.$$

Then

$$P(x_0 = +1 \mid x_t = 0) = P(x_0 = -1 \mid x_t = 0) = \frac{1}{2},$$

and therefore

$$\mu_0(0) = \frac{1}{2}(+1) + \frac{1}{2}(-1) = 0.$$

The DDIM direction based on

$$\mu_0(x_t) - x_t$$

is then exactly zero:

$$\mu_0(0) - 0 = 0.$$

Under an exact deterministic flow with perfect symmetry, the sample can therefore remain at the ambiguous midpoint:

$$\boxed{0 \to 0 \to 0 \to \cdots \to 0.}$$

In this idealized example, the midpoint is a boundary between the basins flowing toward the two modes. An arbitrarily small displacement to one side can break the symmetry:

$$+\epsilon \to \cdots \to +1,$$

while

$$-\epsilon \to \cdots \to -1.$$

This edge case does not imply that the generated distribution generally collapses onto the average. The initial DDIM state is drawn from a continuous distribution,

$$x_T \sim \mathcal{N}(0, \sigma_T^2 I),$$

and the probability of drawing any one exact point, such as the perfectly symmetric midpoint, is zero:

$$P(x_T = 0) = 0.$$

Almost surely, the initial sample lies on one side or another of such an exact boundary and consequently follows one of the corresponding flow trajectories.

The important distinction is therefore

DDIM does not repeatedly move toward one fixed conditional mean.

Instead, the conditional mean is recomputed as a function of the current state and defines a changing local direction. Initial randomness distributes samples throughout this vector field, while the deterministic DDIM dynamics transport those samples toward different modes of the clean distribution. Only special, measure-zero symmetric initial conditions can remain exactly balanced between modes in the idealized case.

3.10 A Better Notation for Thinking About the Destination

The notation F_t used in the tutorial is convenient when every reverse step has the same fixed size Δt.

If we know in advance that

$$t \longrightarrow t - \Delta t,$$

then specifying t automatically specifies the destination.

For conceptual purposes, however, it is useful to imagine a more explicit notation:

$$\boxed{F_{t \to s}(x_t), \qquad s < t.}$$

Here,

$$t = \text{current noise level},$$

while

$$s = \text{desired next noise level}.$$

With a fixed discretization,

$$s = t - \Delta t,$$

so we can abbreviate

$$F_{t \to t - \Delta t}$$

as simply

$$F_t.$$

But the expanded notation makes the information flow much clearer.

For example,

$$F_{0.7\to 0.699}(x_{0.7})$$

and

$$F_{0.7\to 0.69}(x_{0.7})$$

are two different transport operations.

A sampler must know which destination it is trying to reach.

3.11 So What Is Actually Fixed During Training?

We should now distinguish two parameterizations.

Previous-state prediction. If we train

$$f_\theta(x_t, t) \approx \mathbb{E}[x_{t-\Delta t} \mid x_t],$$

then the destination $t - \Delta t$ is part of the definition of the training target.

Therefore the training step size Δt must be known and fixed unless we explicitly provide the destination or step size to the network.

In this parameterization,

$$\boxed{\Delta t \text{ is implicitly built into } f_\theta.}$$

Clean-state prediction. If instead we train

$$f_\theta(x_t, t) \approx \mathbb{E}[x_0 \mid x_t],$$

the prediction target no longer depends on the next sampling timestep.

The model needs to know the current time t, because t determines the current noise level, but it does not need to know the destination timestep merely to estimate x_0.

This gives a cleaner separation:

network knows:	$x_t,\ t,$
sampler knows:	$f(x_t) = \widehat{x}_0, t,\ s,$ noise schedule,
sampler produces:	$x_s.$

Diagramatically giving us:

$$\underbrace{(x_t, t) \xrightarrow{f_\theta} \hat{x}_0}_{\text{learned denoiser}} \xrightarrow{F_{t\to s}(x_t, \hat{x}_0)} \underbrace{x_s}_{\text{next state}}$$

This distinction will become particularly useful when we discuss taking different-sized steps during deterministic sampling.

3.12 Now We Can Ask the DDIM Question

DDPM begins with the particular Markov forward process

$$x_0 \to x_{\Delta t} \to x_{2\Delta t} \to \cdots \to x_1.$$

That particular stochastic process defines a particular joint distribution over the complete trajectory,

$$p(x_0, x_{\Delta t}, x_{2\Delta t}, \ldots, x_1),$$

and therefore defines particular reverse conditional distributions,

$$p(x_{t-\Delta t} \mid x_t).$$

DDPM attempts to sample these reverse conditionals.
But now ask:

> Do we actually need our generative process to reproduce the particular reverse conditional distribution created by the original random walk?

For generation, what we ultimately require is that samples have the correct marginal distribution. Suppose

$$x_t \sim p_t.$$

We would like an update rule that produces

$$x_{t-\Delta t} \sim p_{t-\Delta t}.$$

If F_t denotes that update, we want

$$\boxed{x_t \sim p_t \quad \Longrightarrow \quad F_t(x_t) \sim p_{t-\Delta t}.}$$

Equivalently, using pushforward notation,

$$\boxed{(F_t)_{\#} p_t = p_{t-\Delta t}.}$$

The crucial point is that this condition does *not* require

$$F_t(x_t) \sim p(x_{t-\Delta t} \mid x_t).$$

Those are two different requirements.

3.13 Pointwise Reversal versus Distributional Transport

DDPM asks:

$$\boxed{\text{Given this particular } x_t, \text{ what are plausible values of } x_{t-\Delta t}?}$$

The answer is a conditional probability distribution,

$$p(x_{t-\Delta t} \mid x_t).$$

Because that distribution generally has nonzero variance, DDPM needs randomness to sample from it.

DDIM asks a different question:

How should all points distributed according to p_t be moved so that their new distribution is $p_{t-\Delta t}$?

This is a transport problem.

A deterministic function may be able to accomplish this even though the original reverse conditional is stochastic.

Therefore,

DDPM:	$x_{t-\Delta t} \sim p(x_{t-\Delta t} \mid x_t)$,
DDIM:	$x_t \sim p_t \Longrightarrow F_t(x_t) \sim p_{t-\Delta t}$.

The first condition specifies what should happen to each individual input probabilistically.

The second only specifies what should happen to the distribution of the population of inputs.

3.14 Why Is DDIM Allowed to Make This Change?

Recall that Gaussian diffusion gives

$$x_t = x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).$$

Therefore we can construct a noisy training example at any desired time t directly from x_0.

We do not need to generate the complete trajectory

$$x_0 \to x_{\Delta t} \to \cdots \to x_t$$

in order to obtain a training example at time t.

Consequently, the denoising information learned from pairs involving x_0 and x_t does not uniquely determine how samples at every pair of intermediate times must be coupled.

This is the freedom DDIM exploits.

There can be multiple processes having the same time-indexed marginal distributions

$$p_0, p_{\Delta t}, p_{2\Delta t}, \ldots, p_1$$

while assigning different trajectories to individual samples.

DDPM uses the stochastic coupling inherited from the original forward random walk.

DDIM instead constructs a deterministic coupling.

3.15 The Information Available During Generation

It is useful to state the generation procedure explicitly.

Suppose we are currently at time t with sample x_t.

At this moment we know:

x_t	current sample,
t	current noise level,
s	next timestep chosen by the sampler,
σ_t, σ_s	known from the predefined noise schedule.

The only important information we *do not* know is the clean sample that corresponds to x_t, or equivalently the exact denoising direction implied by the unknown data distribution.

That is what the neural network estimates.

Thus a useful conceptual picture is

$$(x_t, t) \xrightarrow{f_\theta} \text{denoising information},$$

followed by

$$(x_t,\ \text{denoising information},\ t,\ s,\ \text{noise schedule}) \xrightarrow{\text{DDIM sampler}} x_s.$$

The timestep choice is therefore not mysterious information that the model somehow has to infer from x_t.

It is explicitly known to the sampling algorithm.

3.16 The Central Idea to Carry Forward

The essential DDIM idea can now be stated precisely.

Training teaches us useful information about the family of noisy distributions p_t indexed by t. In DDPM, this information is used to construct a stochastic reverse process that probabilistically moves a noisy sample back toward the data distribution. But generation does not need to reconstruct the particular random trajectory that would have arisen from the original forward Markov chain.

DDIM instead uses the learned denoising information to construct a deterministic transport that moves samples between the same noisy distributions:

$$\boxed{p_t \xrightarrow{\text{deterministic transport}} p_{t-\Delta t}.}$$

In the continuous-time view, this transport is described by a vector field: at each point and noise level, the model determines a direction in which the sample should move toward its clean counterpart. Generating a sample then amounts to integrating this vector field from the noisy distribution back toward the data distribution. If this evolution admitted a known closed-form solution, we could perform the transport directly. In general, it does not, so we approximate the trajectory numerically with a differential-equation solver.

The next section derives this deterministic transport explicitly in the simplest possible setting: a data distribution consisting of a single point.

3.17 The key distinction

DDPM asks for

$$\boxed{F_t(x_t) \sim p(x_{t-\Delta t} \mid x_t) \quad \text{for each particular } x_t.}$$

DDIM instead asks for

$$\boxed{x_t \sim p_t \quad \Longrightarrow \quad F_t(x_t) \sim p_{t-\Delta t}.}$$

The first is a *pointwise conditional* requirement.

The second is only a *marginal transport* requirement.

The second requirement is weaker, so many more maps can satisfy it.

4 Why Training Leaves Us This Freedom

This point is easy to state too quickly, so let us unpack it carefully.

A diffusion model can sample a noisy training example at arbitrary time t directly:

$$x_t = x_0 + \sigma_t \epsilon.$$

Therefore the training problem can be expressed using pairs such as

$$(x_0, x_t),$$

without actually constructing the entire path

$$x_0, x_{\Delta t}, x_{2\Delta t}, \ldots, x_t.$$

For example, practical diffusion parameterizations often learn quantities related to

$$\mathbb{E}[x_0 \mid x_t],$$

or equivalently the noise or score.

The training signal at a selected time t is therefore determined by the distribution of the pair (x_0, x_t), especially the Gaussian conditional

$$p(x_t \mid x_0) = \mathcal{N}(x_0, \sigma_t^2 I).$$

It does not require us to commit to one unique relationship among all the intermediate variables.

The DDIM opportunity

If we can construct another joint process whose time-t marginals are the same ones used during training, then the learned denoising information remains useful, even though the relationship between different timesteps may be different.

This is the conceptual opening behind DDIM.

A useful way to summarize it is

$$\boxed{\text{same marginals } \not\Rightarrow \text{ same joint trajectory.}}$$

DDIM exploits the freedom in the trajectory.

5 Start with the Easiest Possible Data Distribution

The tutorial derives intuition by first considering an absurdly simple dataset: suppose there is only one possible clean sample.

Let

$$p_0 = \delta_{x_0}.$$

That means every clean sample is exactly the same point x_0.

To simplify the algebra even further, first take

$$x_0 = 0.$$

At time t,

$$x_t \sim \mathcal{N}(0, \sigma_t^2 I).$$

At the slightly earlier time $t - \Delta t$,

$$x_{t-\Delta t} \sim \mathcal{N}(0, \sigma_{t-\Delta t}^2 I).$$

Since $t - \Delta t < t$,

$$\sigma_{t-\Delta t} < \sigma_t.$$

So the earlier distribution is simply a narrower Gaussian.

5.1 How can we deterministically turn one Gaussian into the other?

Suppose

$$X \sim \mathcal{N}(0, \sigma_t^2 I).$$

Scale X by

$$\frac{\sigma_{t-\Delta t}}{\sigma_t}.$$

Define

$$G_t(X) = \frac{\sigma_{t-\Delta t}}{\sigma_t} X.$$

Recall that if

$$X \sim \mathcal{N}(0, \sigma^2 I),$$

then

$$aX \sim \mathcal{N}(0, a^2 \sigma^2 I).$$

Therefore,

$$G_t(X) \sim \mathcal{N}\left(0, \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} \sigma_t^2 I\right) = \mathcal{N}(0, \sigma_{t-\Delta t}^2 I).$$

Thus

$$\boxed{G_t(x_t) = \frac{\sigma_{t-\Delta t}}{\sigma_t} x_t}$$

is a perfectly valid *deterministic* reverse sampler for this simple case.
No random noise is required.

5.2 Why this does not contradict DDPM

DDPM asks:

$$\text{“Given this exact } x_t, \text{ what could } x_{t-\Delta t} \text{ have been?”}$$

There are many possible answers, because the forward process was stochastic.

DDIM asks:

“How should I move the entire population of samples distributed as p_t so that it becomes $p_{t-\Delta t}$?”

For the one-point example, simply shrinking every sample toward the origin does the job.

The deterministic DDIM path therefore does *not* need to reproduce the actual historical forward path of any sample.

6 Connecting the Scaling Map to a Denoiser

The previous section gave us the desired deterministic map

$$G_t(x_t) = \frac{\sigma_{t-\Delta t}}{\sigma_t} x_t.$$

At this point, no neural network is actually necessary. This is because we deliberately chose the simplest possible data distribution: there is only one clean point, and we placed that point at $x_0 = 0$. Consequently, we already know exactly where the noisy Gaussian should contract toward.

To see what information will be missing in a realistic problem, suppose instead that the single clean point is located at $x_0 = a$. The corresponding deterministic transport is

$$\boxed{G_t(x_t) = a + \frac{\sigma_{t-\Delta t}}{\sigma_t}(x_t - a).}$$

Thus, if the clean point a were known, deterministic denoising would again be straightforward: simply contract the noisy sample toward a by the appropriate ratio of standard deviations.

In a realistic generative problem, however, the clean distribution contains many possible values of x_0. Given only a noisy sample x_t, we do not know which clean point should determine its motion. We therefore need to express the desired transport using a statistical quantity that can be estimated from x_t. This is where conditional expectations, and ultimately the learned denoiser, enter the construction.

6.1 Deriving the Conditional Mean for the Single-Point Case

We now want to connect the deterministic transport derived above to a quantity that will later be useful for denoising. The tutorial considers

$$\mu_{t-\Delta t}(x_t) = \mathbb{E}[x_{t-\Delta t} \mid x_t].$$

In words, this asks:

> Given that we observe the noisy state x_t, what is the expected value of the state one diffusion step earlier, $x_{t-\Delta t}$?

For now, we remain in the extremely simple case in which the clean distribution consists of the single point

$$x_0 = 0.$$

This assumption is important. In this special case, all randomness in $x_{t-\Delta t}$ and x_t comes entirely from the Gaussian noise added by the forward diffusion process.

6.1.1 The Distributions of the Two States

Recall that our forward diffusion satisfies

$$x_t \mid x_0 \sim \mathcal{N}(x_0, \sigma_t^2 I).$$

Since we have assumed $x_0 = 0$,

$$x_{t-\Delta t} \sim \mathcal{N}(0, \sigma_{t-\Delta t}^2 I)$$

and

$$x_t \sim \mathcal{N}(0, \sigma_t^2 I).$$

Furthermore, the final forward step from $t - \Delta t$ to t is

$$\boxed{x_t = x_{t-\Delta t} + \eta,}$$

where

$$\eta \sim \mathcal{N}(0, \sigma_q^2 \Delta t\, I)$$

is newly sampled Gaussian noise that is independent of $x_{t-\Delta t}$.

Thus x_t and $x_{t-\Delta t}$ are not independent: x_t literally contains $x_{t-\Delta t}$ plus some additional independent noise.

6.1.2 Why Are They Jointly Gaussian?

It is not enough to say that $x_{t-\Delta t}$ and x_t are each individually Gaussian. We will use a formula that requires them to be *jointly Gaussian*.

Two random vectors X and Y are jointly Gaussian if their concatenation

$$\begin{bmatrix} X \\ Y \end{bmatrix}$$

has a multivariate Gaussian distribution.

To see why this is true in our case, begin with

$$x_{t-\Delta t} \sim \mathcal{N}(0, \sigma_{t-\Delta t}^2 I)$$

and

$$\eta \sim \mathcal{N}(0, \sigma_q^2 \Delta t\, I).$$

Because these two Gaussian random variables are independent, their concatenation is jointly Gaussian:

$$\begin{bmatrix} x_{t-\Delta t} \\ \eta \end{bmatrix}.$$

Now recall that

$$x_t = x_{t-\Delta t} + \eta.$$

Therefore,

$$\begin{bmatrix} x_{t-\Delta t} \\ x_t \end{bmatrix} = \begin{bmatrix} I & 0 \\ I & I \end{bmatrix} \begin{bmatrix} x_{t-\Delta t} \\ \eta \end{bmatrix}.$$

A linear transformation of a Gaussian random vector is also Gaussian. Consequently,

$$\boxed{(x_{t-\Delta t}, x_t) \text{ are jointly Gaussian.}}$$

This fact is useful because jointly Gaussian random variables have a particularly simple conditional expectation.

6.1.3 The Conditional-Expectation Formula

For zero-mean jointly Gaussian random vectors X and Y,

$$\boxed{\mathbb{E}[X \mid Y = y] = \mathrm{Cov}(X, Y)\,\mathrm{Var}(Y)^{-1} y.}$$

We will apply this formula using

$$X = x_{t-\Delta t}, \qquad Y = x_t.$$

Thus,

$$\mathbb{E}[x_{t-\Delta t} \mid x_t] = \mathrm{Cov}(x_{t-\Delta t}, x_t)\,\mathrm{Var}(x_t)^{-1} x_t.$$

We therefore need to calculate two quantities:

$$\mathrm{Cov}(x_{t-\Delta t}, x_t)$$

and

$$\mathrm{Var}(x_t).$$

6.1.4 Calculating the Covariance

Recall again that

$$x_t = x_{t-\Delta t} + \eta.$$

Therefore,

$$\mathrm{Cov}(x_{t-\Delta t}, x_t) = \mathrm{Cov}\left(x_{t-\Delta t}, x_{t-\Delta t} + \eta\right).$$

Covariance is linear in its arguments, so

$$\mathrm{Cov}(x_{t-\Delta t}, x_t) = \mathrm{Cov}(x_{t-\Delta t}, x_{t-\Delta t}) + \mathrm{Cov}(x_{t-\Delta t}, \eta).$$

The first term is simply the variance:

$$\mathrm{Cov}(x_{t-\Delta t}, x_{t-\Delta t}) = \mathrm{Var}(x_{t-\Delta t}).$$

The second term is zero because the newly added noise η is independent of $x_{t-\Delta t}$:

$$\mathrm{Cov}(x_{t-\Delta t}, \eta) = 0.$$

Therefore,

$$\mathrm{Cov}(x_{t-\Delta t}, x_t) = \mathrm{Var}(x_{t-\Delta t}).$$

Since

$$x_{t-\Delta t} \sim \mathcal{N}(0, \sigma^2_{t-\Delta t} I),$$

we have

$$\boxed{\mathrm{Cov}(x_{t-\Delta t}, x_t) = \sigma^2_{t-\Delta t} I.}$$

This has a useful intuitive interpretation. The part of x_t that is correlated with $x_{t-\Delta t}$ is precisely the $x_{t-\Delta t}$ component itself. The newly added noise contributes additional variance to x_t, but no additional covariance with the previous state.

6.1.5 Calculating the Variance of x_t

From the forward process,

$$x_t \sim \mathcal{N}(0, \sigma_t^2 I),$$

so immediately

$$\boxed{\mathrm{Var}(x_t) = \sigma_t^2 I.}$$

We could also derive this directly from

$$x_t = x_{t-\Delta t} + \eta.$$

Because the two terms are independent,

$$\mathrm{Var}(x_t) = \mathrm{Var}(x_{t-\Delta t}) + \mathrm{Var}(\eta),$$

giving

$$\sigma_t^2 I = \sigma^2_{t-\Delta t} I + \sigma_q^2 \Delta t \, I.$$

For the variance schedule used in this tutorial,

$$\sigma_t^2 = \sigma_q^2 t,$$

so this is exactly

$$\sigma_q^2 t = \sigma_q^2 (t - \Delta t) + \sigma_q^2 \Delta t.$$

6.1.6 Putting the Pieces Together

We can now return to the Gaussian conditional-expectation formula:

$$\mathbb{E}[x_{t-\Delta t} \mid x_t] = \mathrm{Cov}(x_{t-\Delta t}, x_t)\,\mathrm{Var}(x_t)^{-1} x_t.$$

Substituting our two results gives

$$\mathbb{E}[x_{t-\Delta t} \mid x_t] = \left(\sigma_{t-\Delta t}^2 I\right)\left(\sigma_t^2 I\right)^{-1} x_t.$$

Since

$$\left(\sigma_t^2 I\right)^{-1} = \frac{1}{\sigma_t^2} I,$$

we obtain

$$\mathbb{E}[x_{t-\Delta t} \mid x_t] = \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t.$$

Thus,

$$\boxed{\mu_{t-\Delta t}(x_t) = \mathbb{E}[x_{t-\Delta t} \mid x_t] = \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t.}$$

6.1.7 Why Is This Not Already Our DDIM Transport?

This result looks very similar to the deterministic transport map derived previously, but there is an important difference.

The deterministic transport that correctly changes

$$\mathcal{N}(0, \sigma_t^2 I)$$

into

$$\mathcal{N}(0, \sigma_{t-\Delta t}^2 I)$$

is

$$\boxed{G_t(x_t) = \frac{\sigma_{t-\Delta t}}{\sigma_t} x_t.}$$

By contrast, the conditional mean is

$$\boxed{\mu_{t-\Delta t}(x_t) = \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t.}$$

In general,

$$\frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} \neq \frac{\sigma_{t-\Delta t}}{\sigma_t}.$$

Therefore,

$$\boxed{\mu_{t-\Delta t}(x_t) \neq G_t(x_t).}$$

The conditional mean moves the sample too far toward the center.
To see this explicitly, suppose we simply used

$$x_{t-\Delta t} = \mu_{t-\Delta t}(x_t) = \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t.$$

Since

$$x_t \sim \mathcal{N}(0, \sigma_t^2 I),$$

the variance of the resulting random variable would be

$$\begin{aligned}
\mathrm{Var}(x_{t-\Delta t}) &= \mathrm{Var}\left(\frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t\right) \\
&= \left(\frac{\sigma_{t-\Delta t}^2}{\sigma_t^2}\right)^2 \mathrm{Var}(x_t) \\
&= \frac{\sigma_{t-\Delta t}^4}{\sigma_t^4} \sigma_t^2 I \\
&= \boxed{\frac{\sigma_{t-\Delta t}^4}{\sigma_t^2} I.}
\end{aligned}$$

But the distribution we actually want at time $t - \Delta t$ has variance

$$\boxed{\sigma_{t-\Delta t}^2 I.}$$

These are not equal.
By comparison, the desired deterministic transport gives

$$G_t(x_t) = \frac{\sigma_{t-\Delta t}}{\sigma_t} x_t,$$

whose variance is

$$\begin{aligned}
\mathrm{Var}(G_t(x_t)) &= \left(\frac{\sigma_{t-\Delta t}}{\sigma_t}\right)^2 \sigma_t^2 I \\
&= \sigma_{t-\Delta t}^2 I,
\end{aligned}$$

which is exactly correct.
This distinction is fundamental:

conditional mean	$\neq$	deterministic transport
$\mathbb{E}[x_{t-\Delta t} \mid x_t]$		$G_t(x_t)$

The conditional mean tells us useful information about the *direction* in which the sample should move, but simply replacing the sample by that mean would contract the distribution too strongly.

This is precisely why DDIM is not obtained by taking the DDPM update and merely deleting its random-noise term. Instead, DDIM will use the conditional-mean information to construct a different deterministic update whose action on the *whole distribution* has the correct variance.

7 Deriving the Tutorial's DDIM Update

The tutorial proposes an update of the form

$$x_{t-\Delta t} = x_t + \lambda \left(\mu_{t-\Delta t}(x_t) - x_t \right).$$

Interpret this geometrically.
The vector

$$\mu_{t-\Delta t}(x_t) - x_t$$

points from the current point toward the denoiser's conditional-mean prediction.
DDIM moves only some fraction λ along that direction.
For the one-point case,

$$\mu_{t-\Delta t}(x_t) = \frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t.$$

Substitute this:

$$x_{t-\Delta t} = x_t + \lambda \left(\frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} x_t - x_t \right).$$

Factor out x_t:

$$x_{t-\Delta t} = \left[1 + \lambda \left(\frac{\sigma_{t-\Delta t}^2}{\sigma_t^2} - 1 \right) \right] x_t.$$

We want this to equal the correct scaling transport

$$\frac{\sigma_{t-\Delta t}}{\sigma_t} x_t.$$

Let

$$a = \sigma_{t-\Delta t}, \qquad b = \sigma_t.$$

Then we require

$$1 + \lambda \left(\frac{a^2}{b^2} - 1 \right) = \frac{a}{b}.$$

Since

$$\frac{a^2}{b^2} - 1 = \frac{(a-b)(a+b)}{b^2},$$

we obtain

$$\lambda = \frac{b}{a+b}.$$

Restoring the original notation,

$$\boxed{\lambda = \frac{\sigma_t}{\sigma_{t-\Delta t} + \sigma_t}.}$$

Thus the DDIM-like update in the tutorial is

$$\boxed{x_{t-\Delta t} = x_t + \frac{\sigma_t}{\sigma_{t-\Delta t} + \sigma_t} \left(\mathbb{E}[x_{t-\Delta t} \mid x_t] - x_t \right).}$$

For the one-point distribution, this exactly reproduces the deterministic scaling map.

8 A Numerical Example

Suppose for simplicity we are in one dimension and

$$\sigma_t = 4, \qquad \sigma_{t-\Delta t} = 3.$$

Then

$$x_t \sim \mathcal{N}(0, 16),$$

and we want

$$x_{t-\Delta t} \sim \mathcal{N}(0, 9).$$

The obvious deterministic transport is

$$x_{t-\Delta t} = \frac{3}{4} x_t.$$

Suppose our current sample happens to be

$$x_t = 8.$$

Then the desired deterministic output is

$$x_{t-\Delta t} = 6.$$

What is the conditional mean?

$$\mathbb{E}[x_{t-\Delta t} \mid x_t = 8] = \frac{3^2}{4^2}(8) = \frac{9}{16}(8) = 4.5.$$

So simply using the conditional mean would move

$$8 \longrightarrow 4.5,$$

which is too far.
DDIM instead computes

$$\lambda = \frac{4}{3+4} = \frac{4}{7}.$$

Then

$$x_{t-\Delta t} = 8 + \frac{4}{7}(4.5 - 8) = 8 + \frac{4}{7}(-3.5) = 8 - 2 = 6.$$

Exactly the desired answer.

Important lesson

DDIM does use the conditional mean learned by denoising, but it does *not* simply set the next state equal to that conditional mean.
The conditional mean supplies a useful *direction*. The DDIM coefficient chooses how far to move in that direction so that the entire distribution is transported correctly.

9 From One Clean Point to Two Clean Points

Real datasets obviously contain more than one possible clean sample.

The tutorial next considers

$$p_0 = \frac{1}{2}\delta_a + \frac{1}{2}\delta_b.$$

After adding Gaussian noise,

$$p_t(x) = \frac{1}{2}\mathcal{N}(a, \sigma_t^2 I) + \frac{1}{2}\mathcal{N}(b, \sigma_t^2 I).$$

So we now have two overlapping clouds.

If we knew that a particular noisy sample came from a, we could use the single-point velocity field associated with a.

Likewise, if we knew it came from b, we could use the field associated with b.

But given only x_t, we generally do not know which clean point generated it.

9.1 The correct combination is posterior-weighted

Let

$$v_t^{[a]}(x)$$

be the velocity field appropriate to the a-cloud, and let

$$v_t^{[b]}(x)$$

be the velocity field appropriate to the b-cloud.

At location x_t, the effective velocity is

$$v_t(x_t) = p(x_0 = a \mid x_t) v_t^{[a]}(x_t) + p(x_0 = b \mid x_t) v_t^{[b]}(x_t).$$

This weighting makes intuitive sense.

If x_t lies very close to a, then

$$p(x_0 = a \mid x_t)$$

will usually be large, so the a-field dominates.

If x_t lies near the overlap between the two noisy clouds, both possibilities matter.

Using the law of total expectation,

$$\mathbb{E}[x_{t-\Delta t} - x_t \mid x_t] = \sum_{c \in \{a,b\}} p(x_0 = c \mid x_t) \mathbb{E}[x_{t-\Delta t} - x_t \mid x_t, x_0 = c].$$

Therefore the posterior-weighted combination of the individual velocity fields is exactly the velocity field built from the overall conditional expectation.

This is why the same denoising quantity that worked for one clean point also produces the appropriate field for a mixture.

10 The Gas Analogy

The tutorial's gas analogy is worth slowing down because it captures the central idea.

Imagine that $p_t(x)$ describes the density of a gas.

A very large collection of particles is distributed according to p_t. We want to move these particles so that, a moment later, their density is $p_{t-\Delta t}$.

There are at least two fundamentally different ways particles could move.

10.1 DDPM: random microscopic motion

Each particle undergoes a stochastic update.

Even if two particles begin at exactly the same x_t, their next positions can differ because fresh random noise is sampled.

This resembles Brownian motion.

10.2 DDIM: deterministic flow

Instead, assign every location x a velocity

$$v_t(x).$$

Every particle currently at x moves according to that velocity:

$$x_{t-\Delta t} = x_t + v_t(x_t)\Delta t.$$

No new randomness is required.

Particles at different locations can have different velocities, so the cloud can contract, stretch, bend, and split its flow in complicated ways.

The only requirement is that after the movement, the *density* of the particles is the desired density $p_{t-\Delta t}$.

Microscopic paths versus macroscopic density

DDPM and DDIM need not give the same trajectory to an individual sample.
They are designed so that the *distribution of many samples* evolves through the desired family of marginals.

11 Velocity-Field Form of DDIM

The tutorial defines

$$v_t(x_t) = \frac{\lambda}{\Delta t}\left(\mathbb{E}[x_{t-\Delta t} \mid x_t] - x_t\right).$$

Then

$$x_{t-\Delta t} = x_t + v_t(x_t)\Delta t.$$

This is just the familiar Euler update for an ordinary differential equation:

$$x(t - \Delta t) \approx x(t) - \Delta t \frac{\mathrm{d}x}{\mathrm{d}t},$$

with signs depending on whether we describe time in the forward or reverse direction.

The important conceptual shift is that DDIM can now be viewed as numerical integration of a deterministic flow.

Once we adopt that perspective, "sampling" looks less like repeatedly undoing individual random noise additions and more like moving a point through a learned vector field from the noise distribution toward the data distribution.

12 Why the Same Model Can Support DDPM and DDIM

This is one of the most surprising aspects of DDIM.

Suppose a model gives us the conditional expectation

$$\mathbb{E}[x_{t-\Delta t} \mid x_t],$$

or equivalently a quantity such as

$$\mathbb{E}[x_0 \mid x_t],$$

the noise prediction, or the score.

DDPM can use this information to specify the mean of a stochastic reverse conditional and then add noise.

DDIM can use related information to construct a deterministic velocity field.

Thus the *learned information* can be shared even though the *sampling dynamics* are different.

In the simplified tutorial setting, Claim 2 gives

$$\mathbb{E}[x_{t-\Delta t} - x_t \mid x_t] = \frac{\Delta t}{t}\mathbb{E}[x_0 - x_t \mid x_t].$$

Therefore,

$$\mathbb{E}[x_{t-\Delta t} \mid x_t] = x_t + \frac{\Delta t}{t}\left(\mathbb{E}[x_0 \mid x_t] - x_t\right).$$

So a model trained to estimate the clean sample's conditional expectation can supply the information required for the DDIM direction.

13 DDIM Is Not "DDPM with the Noise Removed"

This misconception is worth isolating.

The DDPM-like update is

$$x_{t-\Delta t} = \mu_{t-\Delta t}(x_t) + \eta,$$

where

$$\eta \sim \mathcal{N}(0, \sigma_q^2 \Delta t I).$$

Naively deleting the noise gives

$$x_{t-\Delta t} = \mu_{t-\Delta t}(x_t).$$

But, as the single-point example showed, that generally does not preserve the correct marginal variance.

DDIM instead uses

$$x_{t-\Delta t} = x_t + \lambda \left(\mu_{t-\Delta t}(x_t) - x_t \right).$$

So the correct comparison is

DDPM:	conditional mean + fresh noise,
DDIM:	carefully scaled deterministic motion toward the conditional mean.

They happen to use closely related learned quantities, but they are different reverse constructions.

14 Does DDIM Mean We Can Skip Timesteps?

There are two ideas here that should not be conflated.

14.1 Idea 1: deterministic reverse transport

This is the conceptual heart of the tutorial's Section 3.

Instead of reproducing the stochastic DDPM reverse conditional, construct a deterministic map that transports

$$p_t \longrightarrow p_{t-\Delta t}.$$

14.2 Idea 2: using fewer numerical steps

Once the reverse process is understood as a deterministic flow, we can consider numerically integrating that flow with a coarser time grid.

For example, instead of evaluating at

$$1,\ 0.999,\ 0.998, \ldots, 0,$$

we might evaluate at a much smaller collection of times.

This is related to DDIM's practical usefulness, but it should not be taken as the logical starting point of the derivation.

The deeper point is that we are no longer required to reproduce the exact step-by-step stochastic history of the original forward Markov chain.

15 Probability Flow ODE

The tutorial later connects DDIM to the probability flow ODE.

The forward diffusion can be represented in continuous time as the SDE

$$\mathrm{d}x = \sigma_q \, \mathrm{d}w,$$

where w is Brownian motion.

The corresponding probability flow ODE is

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\frac{1}{2}\sigma_q^2 \nabla_x \log p_t(x).$$

The score

$$\nabla_x \log p_t(x)$$

points toward regions of increasing probability density.

For the Gaussian corruption process, Tweedie's formula relates the score to the conditional expectation:

$$\mathbb{E}[x_0 \mid x_t] = x_t + \sigma_t^2 \nabla_x \log p_t(x_t).$$

Since

$$\sigma_t^2 = \sigma_q^2 t,$$

we obtain

$$\sigma_q^2 \nabla_x \log p_t(x_t) = \frac{1}{t}\left(\mathbb{E}[x_0 \mid x_t] - x_t\right).$$

Therefore the probability flow ODE can be written

$$\boxed{\frac{\mathrm{d}x}{\mathrm{d}t} = -\frac{1}{2t}\left(\mathbb{E}[x_0 \mid x_t] - x_t\right).}$$

When we integrate in the reverse-time direction, this moves samples toward the data distribution.

The tutorial shows that discretizing this ODE recovers the DDIM-like update in the small-step limit.

16 Why a Deterministic ODE Can Match a Stochastic SDE

This can initially sound impossible.

The SDE has random Brownian motion:

$$\mathrm{d}x = f(x,t)\,\mathrm{d}t + g(t)\,\mathrm{d}w.$$

The probability flow ODE has no random term:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \widetilde{f}(x,t).$$

How can they be equivalent?

They are *not* equivalent at the level of individual trajectories.

A particle following the SDE wiggles randomly. A particle following the ODE follows a deterministic streamline.

The equivalence is at the level of *marginal probability distributions*:

$$\boxed{p_t^{\mathrm{SDE}} = p_t^{\mathrm{ODE}} \quad \text{for every } t.}$$

This is exactly the same distinction we have been emphasizing throughout DDIM:

$$\text{individual path} \quad \neq \quad \text{distribution at a given time.}$$

A stochastic microscopic process and a deterministic transport field can evolve the same density.

17 What Happens to Randomness in DDIM?

DDIM is deterministic *conditional on its initial noise sample.*

Generation still begins with

$$x_1 \sim \mathcal{N}(0, \sigma_q^2 I).$$

Once x_1 has been chosen, however, the deterministic DDIM sampler gives a fixed trajectory:

$$x_1 \to x_{1-\Delta t} \to \cdots \to x_0.$$

Thus we can write the final sample schematically as

$$x_0 = F(x_1).$$

Different initial noise vectors generally produce different generated samples, but repeatedly using the exact same initial x_1 gives the same DDIM output.

DDPM behaves differently because it injects additional random variables during the reverse process.

18 DDPM versus DDIM

The conceptual comparison can now be summarized.

	DDPM	**DDIM-like sampler**
Reverse step	Stochastic	Deterministic
Goal	Approximate $p(x_{t-\Delta t} \mid x_t)$	Transport p_t to $p_{t-\Delta t}$
Fresh reverse noise	Yes	No
Given identical initial noise	Can produce different trajectories because later noise is sampled	Produces the same trajectory
What must be correct	Pointwise reverse conditional, approximately	Marginal distribution after transport
Continuous-time view	Reverse-time SDE	Probability flow ODE

The tutorial emphasizes that although both updates happen to move in a direction proportional to

$$\mu_{t-\Delta t}(x_t) - x_t,$$

their interpretations are fundamentally different.

19 A More Precise Version of the "Same Marginals" Argument

It is tempting to summarize DDIM by saying:

> "The loss only depends on $p(x_t \mid x_0)$, so we can change the forward process."

That statement captures an important idea, but it can hide several logical steps.
A more careful version is:

1. The Gaussian forward process gives tractable time-indexed conditionals

$$p(x_t \mid x_0).$$

2. Training can obtain the required denoising information from these time-indexed noisy samples without simulating the entire Markov trajectory.
3. Therefore training does not uniquely identify one coupling among all intermediate random variables.
4. Many joint processes can share the same relevant marginals while differing in how samples at different times are paired.
5. A reverse sampler does not fundamentally need to reconstruct the historical forward trajectory. It needs to transform the terminal distribution back into the target distribution.
6. DDPM chooses a stochastic reverse process associated with the original Markov diffusion.
7. DDIM chooses a deterministic transport consistent with the same family of marginal distributions.

That is the conceptual chain worth remembering.

20 What the Neural Network Is Actually Providing

A neural network is not learning an entire DDIM trajectory for each training example.
Instead, at each noisy state x_t and time t, it estimates a local statistical quantity.
Depending on parameterization, that may be presented as

$$\mathbb{E}[x_0 \mid x_t],$$

a noise prediction,

$$\mathbb{E}[\epsilon \mid x_t],$$

or a score,

$$\nabla_x \log p_t(x_t).$$

These quantities can be algebraically related in the Gaussian setting.

DDIM interprets the learned information as defining a direction of deterministic motion through probability space.

So it is useful to separate two objects:

$$\boxed{\text{learned field}}$$

and

$$\boxed{\text{numerical sampler that follows the field.}}$$

The same learned model can support more than one sampler.

21 A Useful Mental Model

Imagine a complicated cloud of points representing noisy images.

At high noise, the cloud is broad and approximately Gaussian. At low noise, the cloud concentrates around the manifold of realistic images.

DDPM says:

> For each point, randomly sample a plausible predecessor according to the learned reverse conditional.

DDIM says:

> Construct a deterministic velocity field that continuously reshapes the entire cloud from the high-noise density into the low-noise density.

The DDIM trajectory of one particle is not meant to be the literal inverse of the random forward trajectory that produced it.

It is a streamline through a probability flow.

22 Common Misconceptions

Misconception 1: DDIM exactly reverses the forward noise realization

No. In general, DDIM does not recover the particular Gaussian increments that were used in some hypothetical forward trajectory.

Its goal is distributional transport.

Misconception 2: DDIM is just DDPM without adding noise

No. Removing the DDPM noise while leaving its conditional-mean update unchanged generally gives the wrong marginal distribution.

DDIM changes the deterministic update itself.

Misconception 3: deterministic means every generation is identical

No. The initial condition

$$x_1 \sim \mathcal{N}(0, \sigma_q^2 I)$$

is still random.

DDIM is deterministic only after x_1 has been chosen.

Misconception 4: the denoiser predicts a clean realistic image at every step

Not necessarily. A squared-error predictor of x_0 estimates

$$\mathbb{E}[x_0 \mid x_t].$$

At high noise this conditional expectation may average over many possible clean samples and can therefore be blurry or otherwise unlike a typical sample.

The sampler uses this expectation to construct a useful direction; it does not require the expectation itself to be a final realistic sample.

Misconception 5: DDIM's main mathematical idea is simply skipping steps

Skipping steps is a practical consequence of treating reverse generation as a deterministic flow that can be numerically integrated on different time grids.

The deeper idea is the existence of a deterministic transport with the desired marginal evolution.

23 The Entire Story in Equations

The forward corruption is

$$x_t = x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).$$

Hence we have a family of noisy marginal distributions

$$p_0, p_{\Delta t}, p_{2\Delta t}, \ldots, p_1.$$

DDPM attempts to reverse the original stochastic coupling:

$$x_{t-\Delta t} \sim p(x_{t-\Delta t} \mid x_t).$$

For small steps this becomes approximately

$$x_{t-\Delta t} = \mathbb{E}[x_{t-\Delta t} \mid x_t] + \mathcal{N}(0, \sigma_q^2 \Delta t I).$$

DDIM instead seeks a deterministic map F_t satisfying

$$(F_t)_{\#} p_t = p_{t-\Delta t}.$$

In the tutorial's simplified setting,

$$F_t(x_t) = x_t + \lambda \left(\mathbb{E}[x_{t-\Delta t} \mid x_t] - x_t \right),$$

where

$$\lambda = \frac{\sigma_t}{\sigma_{t-\Delta t} + \sigma_t}.$$

Equivalently, define the velocity field

$$v_t(x_t) = \frac{\lambda}{\Delta t} \left(\mathbb{E}[x_{t-\Delta t} \mid x_t] - x_t \right),$$

so that

$$x_{t-\Delta t} = x_t + v_t(x_t)\Delta t.$$

In the continuous-time limit this is connected to the probability flow ODE,

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\frac{1}{2}\sigma_q^2 \nabla_x \log p_t(x).$$

The reverse-time integration of this deterministic field transports Gaussian noise toward the target data distribution.

24 Final Conceptual Summary

The most important idea is not a particular coefficient or equation.

It is that *a probability distribution can be evolved in more than one way.*

The original forward diffusion uses stochastic Brownian-like motion. DDPM constructs a stochastic reverse process associated with that diffusion.

But generation does not require us to retrace each microscopic random step. It only requires us to transform the probability distribution correctly.

DDIM therefore constructs a deterministic flow whose particles follow different paths but whose marginal density evolves through the desired distributions.

In one line,

DDPM reverses a stochastic process; DDIM constructs a deterministic transport.

And the reason we can reuse the same learned denoising information is that the training procedure gives us information about the time-indexed noisy distributions without uniquely fixing the coupling between every pair of timesteps.