Understanding Diffusion Probability Model Interactively

English Chinese Collapse Expand

0. Introduction

The Diffusion Probability Model[1][2] is currently the main method used in image and video generation, but due to its abstruse theory, many engineers are unable to understand it well. This article will provide a very easy-to-understand method to help readers grasp the principles of the Diffusion Model. Specifically, it will illustrate the Diffusion Model using examples of one-dimensional random variables in an interactive way, explaining several interesting properties of the Diffusion Model in an intuitive manner.

The diffusion model is a probabilistic model. Probabilistic models mainly offer two functions: calculating the probability of a given sample appearing; and generating new samples. The diffusion model focuses on the latter aspect, facilitating the production of new samples, thus realizing the task of **generation**.

The diffusion model differs from general probability models (such as GMM), which directly models the probability distribution of random variables. The diffusion model adopts an indirect approach, which utilizes **random variable transform**(shown in Figure 1a) to gradually convert the data distribution (the probability distribution to be modeled) into the **standard normal distribution**, and meanwhile models the posterior probability distribution corresponding to each transformation (Figure 1b-c). Upon obtaining the final standard normal distribution and the posterior probability distributions, one can generate samples of each random variable $Z_T \dots Z_2, Z_1, X$ in reverse order through **Ancestral Sampling**. Simultaneously, initial data distribution $q(x)$ can be determined by employing Bayes theorem and the total probability theorem.

One might wonder: indirect methods require modeling and learning T posterior probability distributions, while direct methods only need to model one probability distribution, Why would we choose the indirect approach? Here's the reasoning: the initial data distribution might be quite complex and hard to represent directly with a probability model. In contrast, the complexity of each posterior probability distribution in indirect methods is significantly simpler, allowing it to be approximated by simple probability models. As we will see later, given certain conditions, posterior probability distributions can closely resemble Gaussian distributions, thus a simple conditional Gaussian model can be used for modeling.

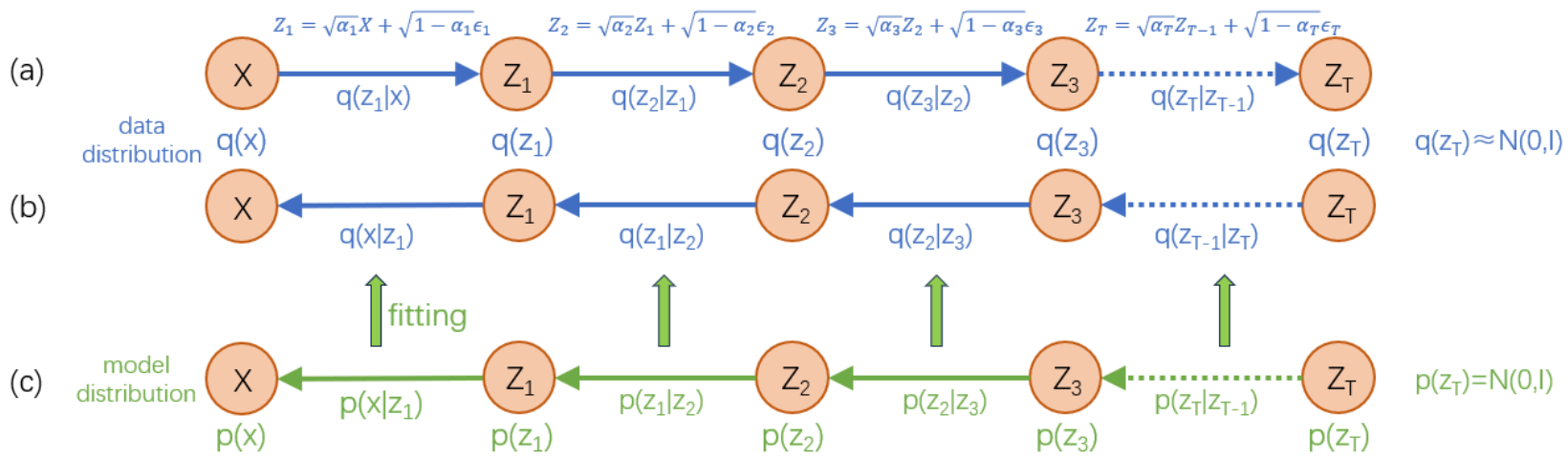


Figure 1: Diffusion probability model schematic

1. How To Transform

To transform the initial data distribution into a simple standard normal distribution, the diffusion model uses the following transformation method:

$$Z = \sqrt{\alpha}X + \sqrt{1-\alpha}\epsilon \qquad where \quad \alpha < 1, \quad \epsilon \sim \mathcal{N}(0, I) \tag{1.1}$$

where $X \sim q(x)$is any random variable， $Z \sim q(Z)$ is the transformed random variable。

This transformation can be divided into two sub-transformations。

The first sub-transformation performs a linear transformation ($\sqrt{\alpha}X$) on the random variable X. According to the conclusion of the literature[3], the linear transformation makes the probability distribution of X **narrower and taller**, and the extent of **narrowing and heightening** is directly proportional to the value of α.

This can be specifically seen in Demo 1, where the first figure depicts a randomly generated one-dimensional probability distribution, and the second figure represents the probability distribution after the linear transformation. It can be observed that the curve of the third figure has become **narrower and taller** compared to the first image. Readers can experiment with different α to gain a more intuitive understanding.

The second sub-transformation is **adding independent random noise**($\sqrt{1-\alpha}\epsilon$). According to the conclusion of the literature[4], **adding independent random variables** is equivalent to performing convolution on the two probability distributions. Since the probability distribution of random noise is Gaussian, it is equivalent to performing a **Gaussian Blur** operation. After blurring, the original probability distribution will become smoother and more similar to the standard normal distribution. The degree of blurring is directly proportional to the noise level ($\sqrt{1-\alpha}$).

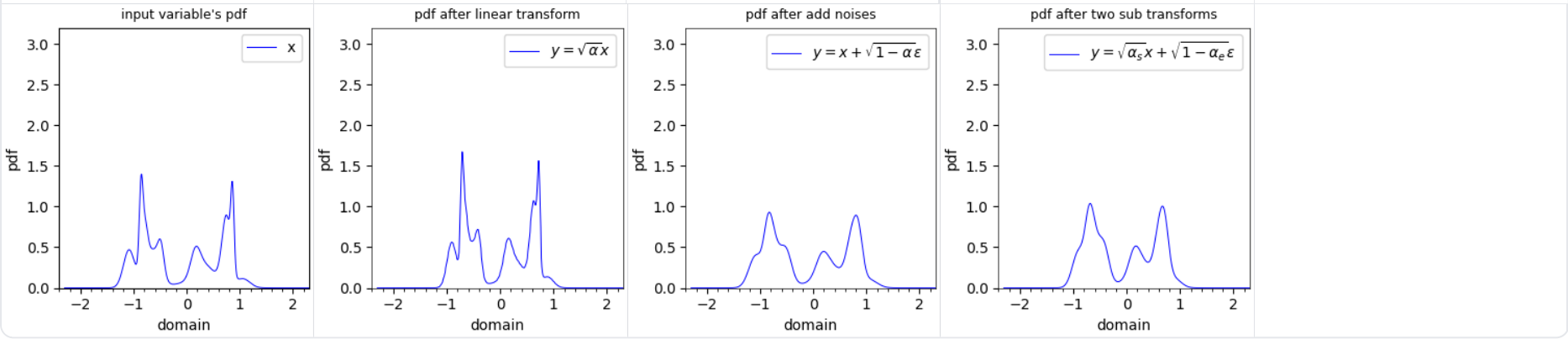
For specifics, one can see Demo 1, where the first figure is a randomly generated one-dimensional probability distribution, and the third figure is the result after the transformation. It can be seen that the transformed probability distribution curve is smoother and there are fewer corners. The readers can test different α values to feel how the noise level affect the shape of the probability distribution. The last figure is the result after applying all two sub-transformations.

Demo 1 - Random Variable Transform In DPM ▼

random seed: 100

alpha of linear transform: 0.7

alpha of add noises: 0.995



2. Likelihood of The Transform ▼

From the transformation method (equation 1.1), it can be seen that the probability distribution of the forward conditional probability $q(z|x)$ is a Gaussian distribution, which is only related to the value of α, regardless of the probability distribution of $q(x)$.

$$q(z|x) = \mathcal{N}(\sqrt{\alpha}x,\ 1-\alpha) \tag{2.1}$$

It can be understood by concrete examples in Demo 2. The third figure depict the shape of $q(z|x)$. From the figure, a uniform slanting line can be observed. This implies that the mean of $q(z|x)$ is linearly related to x, and the variance is fixed. The magnitude of α will determine the width and incline of the slanting line.

3. Posterior of The Transform ▼

The posterior probability distribution does not have a closed form, but its shape can be inferred approximately through some technique.

According to Bayes formula, we have

$$q(x|z) = \frac{q(z|x)q(x)}{q(z)} \tag{3.1}$$

When z takes a fixed value, $q(z)$ is a constant, so the shape of $q(x|z)$ is only related to $q(z|x)q(x)$.

$$q(x|z) \propto q(z|x)q(x) \qquad where\ z\ is\ fixed \tag{3.2}$$

From Equation 2.1, we can see that $q(z|x)$ is a Gaussian distribution, so we have

$$q(x|z) \propto \frac{1}{\sqrt{2\pi(1-\alpha)}} \exp \frac{-(z-\sqrt{\alpha}x)^2}{2(1-\alpha)}\, q(x) \qquad where\ z\ is\ fixed \tag{3.3}$$

$$= \frac{1}{\sqrt{\alpha}} \underbrace{\frac{1}{\sqrt{2\pi}\sigma} \exp \frac{-(x-\mu)^2}{2\sigma^2}}_{\text{GaussFun}}\, q(x) \qquad where\ \mu = \frac{z}{\sqrt{\alpha}} \quad \sigma = \sqrt{\frac{1-\alpha}{\alpha}} \tag{3.4}$$

It can be observed that the **GaussFun** part is a Gaussian function of x, with a mean of $\frac{z}{\sqrt{\alpha}}$ and a variance of $\sqrt{\frac{1-\alpha}{\alpha}}$, so the shape of $q(x|z)$ is determined by **the product of GaussFun and q(x)**.

According to the characteristics of *multiplication*, the characteristics of the shape of the $q(x|z)$ function can be summarized.

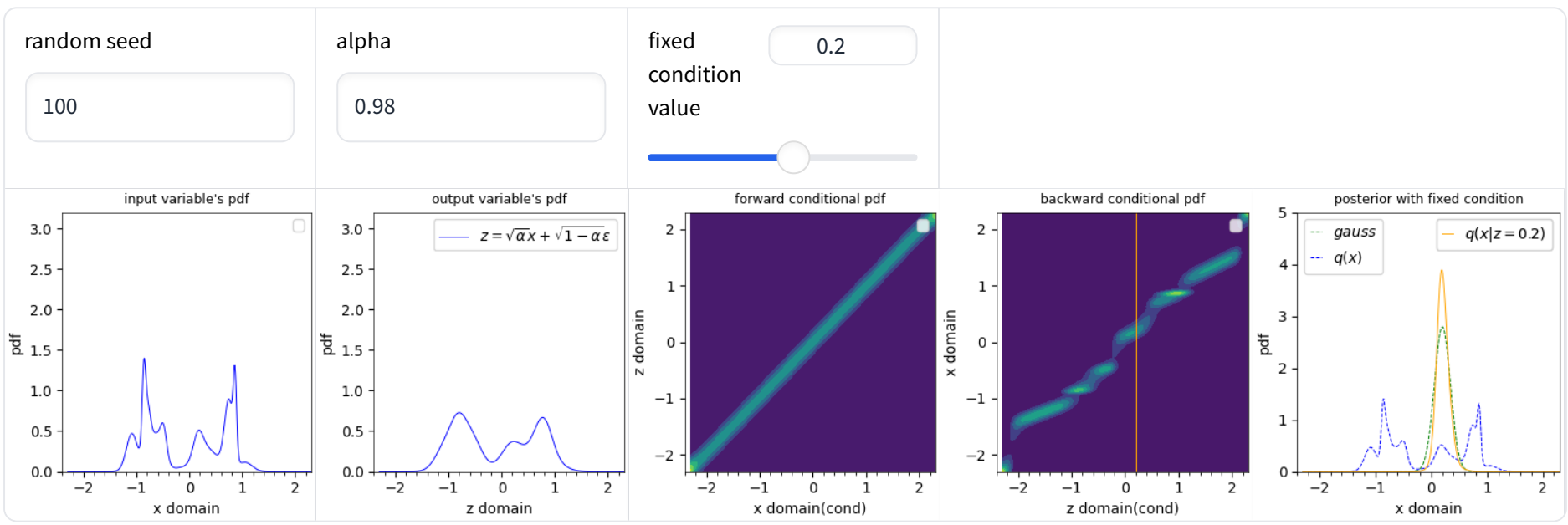
- When the variance of the Gaussian function is small (small noise), or when $q(x)$ changes slowly, the shape of $q(x|z)$ will approximate to the Gaussian function, and have a simpler function form, which is convenient for modeling and learning.
- When the variance of the Gaussian function is large (large noise), or when $q(x)$ changes drastically, the shape of $q(x|z)$ will be more complex, and greatly differ from a Gaussian function, which makes it difficult to model and learn.

The specifics can be seen in Demo 2. The fourth figure present the shape of the posterior $q(x|z)$, which shows an irregular shape and resembles a curved and uneven line. As α increases (noise decreases), the curve tends to be uniform and straight. Readers can adjust different α values and observe the relationship between the shape of posterior and the level of noise. In the last figure, the blue dash line represents $q(x)$, the green dash line represents **GaussFun** in the equation 3.4, and the orange curve represents the result of multiplying the two function and normalizing it, which is the posterior probability $q(x|z = fixed)$ under a fixed z condition. Readers can adjust different values of z to observe how the fluctuation of $q(x)$ affect the shape of the posterior probability $q(x|z)$.

The posterior $q(x|z)$ under two special states are worth considering.

- As $\alpha \to 0$, the variance of **GaussFun** tends to ∞, and $q(x|z)$ for different z almost become identical, and almost the same as $q(x)$. Readers can set α to 0.001 in Demo 2 to observe the specific results.
- As $\alpha \to 1$, the variance of **GaussFun** tends to 0, The $q(x|z)$ for different z values contract into a series of *Dirac delta functions* with different offsets equalling to z. However, there are some exceptions. When there are regions where $q(x)$ is zero, the corresponding $q(x|z)$ will no longer be a Dirac *delta function*, but a zero function. Readers can set α to 0.999 in Demo 2 to observe the specific results.

Demo 2 - Likelihood and Posterior of Transform ▼



4. Transform Data Distribution To Normal Distribution ▼

For any arbitrary data distribution $q(x)$, the transform(equation 2.1) in section 2 can be continuously applied(equation 4.1~4.4). As the number of transforms increases, the output probability distribution will become increasingly closer to the standard normal distribution. For more complex data distributions, more iterations or larger noise are needed.

Specific details can be observed in Demo 3.1. The first figure illustrates a randomly generated one-dimensional probability distribution. After seven transforms, this distribution looks very similar to the standard normal distribution. The degree of similarity increases with the number of iterations and the level of the noise. Given the same degree of similarity, fewer transforms are needed if the noise added at each step is larger (smaller α value). Readers can try different α values and numbers of transforms to see how similar the final probability distribution is.

The complexity of the initial probability distribution tends to be high, but as the number of transforms increases, the complexity of the probability distribution $q(z_t)$ will decrease. As concluded in section 4, a more complex probability distribution corresponds to a more complex posterior probability distribution. Therefore, in order to ensure that the posterior probability distribution is more similar to the Conditional Gaussian function (easier to learn), a larger value of α (smaller noise) should be used in the initial phase, and a smaller value of α (larger noise) can be appropriately used in the later phase to accelerate the transition to the standard normal distribution.

In the example of Demo 3.1, it can be seen that as the number of transforms increases, the corners of $q(z_t)$ become fewer and fewer. Meanwhile, the slanting lines in the plot of the posterior probability distribution $q(z_{t-1}|z_t)$ become increasingly straight and uniform, resembling more and more the conditional Gaussian distribution.

$$
\begin{aligned}
Z_1 &= \sqrt{\alpha_1} X + \sqrt{1-\alpha_1}\epsilon_1 && (4.1)\\
Z_2 &= \sqrt{\alpha_2} Z_1 + \sqrt{1-\alpha_2}\epsilon_2 && (4.2)\\
&\cdots \\
Z_t &= \sqrt{\alpha_t} Z_{t-1} + \sqrt{1-\alpha_t}\epsilon_t && (4.3)\\
&\cdots \\
Z_T &= \sqrt{\alpha_T} Z_{T-1} + \sqrt{1-\alpha_T}\epsilon_T && (4.4)\\
&where \quad \alpha_t < 1 \qquad t \in 1, 2, \ldots, T
\end{aligned}
$$

By substituting Equation 4.1 into Equation 4.2, and utilizing the properties of Gaussian distribution, we can derive the form of $q(z_2|x)$

$$
\begin{aligned}
z_2 &= \sqrt{\alpha_2}(\sqrt{\alpha_1}x + \sqrt{1-\alpha_1}\epsilon_1) + \sqrt{1-\alpha_2}\epsilon_2 && (4.5)\\
&= \sqrt{\alpha_2\alpha_1}x + \sqrt{\alpha_2 - \alpha_2\alpha_1}\epsilon_1 + \sqrt{1-\alpha_2}\epsilon_2 && (4.6)\\
&= \mathcal{N}(\sqrt{\alpha_1\alpha_2}x,\ 1-\alpha_1\alpha_2) && (4.7)
\end{aligned}
$$

In the same way, it can be deduced recursively that

$$
q(z_t|x) = \mathcal{N}(\sqrt{\alpha_1\alpha_2\cdots\alpha_t}x,\ 1-\alpha_1\alpha_2\cdots\alpha_t) = \mathcal{N}(\sqrt{\bar{\alpha}_t}x,\ 1-\bar{\alpha}_t) \qquad where\ \bar{\alpha}_t \triangleq \prod_{j=1}^{t}\alpha_j \tag{4.8}
$$

Comparing the forms of Equation 4.8 and Equation 2.1, it can be found that their forms are completely consistent. If only focusing on the final transformed distribution $q(z_t)$, then the t consective small transformations can be replaced by one large transformation. The α of the large transformation is the accumulation of the α from each small transformation.

In the DDPM[2] paper, the authors used 1000 steps (T=1000) to transform the data distribution $q(x)$ to $q(z_T)$. The probability distribution of $q(z_T|x)$ is as follows:

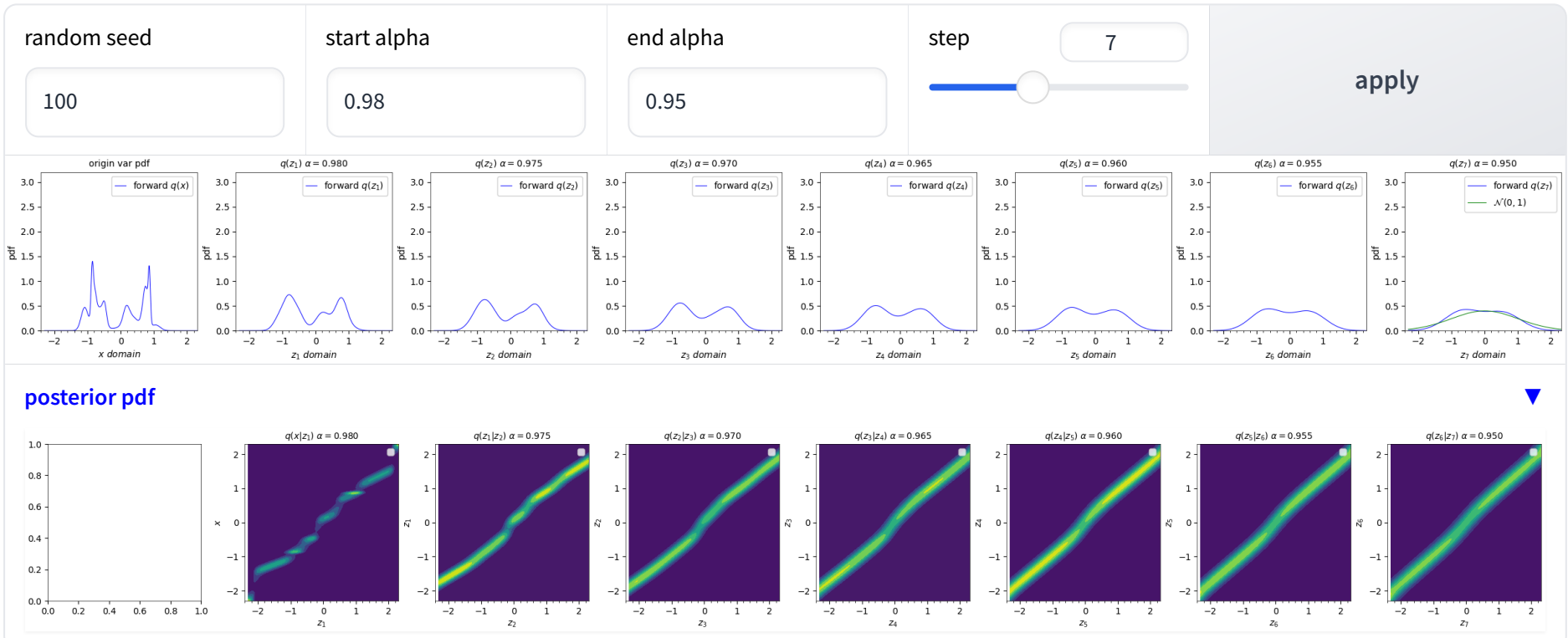
$$
q(z_T|x) = \mathcal{N}(0.00635\ x,\ 0.99998) \tag{4.9}
$$

If considering only marginal distribution $q(z_T)$, a single transformation can also be used, which is as follows:

$$
Z_T = \sqrt{0.0000403}\ X + \sqrt{1-0.0000403}\ \epsilon = 0.00635\ X + 0.99998\ \epsilon \tag{4.10}
$$

It can be seen that, after applying two transforms, the transformed distributions $q(z_T|x)$ are the same. Thus, $q(z_T)$ is also the same.

Demo 3.1 - Transform To Normal Distribution Iteratively ▼



5. Restore Data Distribution From Normal Distribution ▼

If the final probability distribution $q(z_T)$ and the posterior probabilities of each transform $q(x|z), q(z_{t-1}|z_t)$ are known, the data distribution $q(x)$ can be recovered through the Bayes Theorem and the Law of Total Probability, as shown in equations 5.1~5.4. When the final probability distribution $q(z_T)$ is very similar to the standard normal distribution, the standard normal distribution can be used as a substitute.

Specifics can be seen in Demo 3.2. In the example, $q(z_T)$ substitutes $\mathcal{N}(0,1)$, and the error magnitude is given through JS Divergence. The restored probability distribution $q(z_t)$ and $q(x)$ are identified by the green curve, and the original probability distribution is identified by the blue curve. It can be observed that the data distribution $q(x)$ can be well restored, and the error (JS Divergence) will

be smaller than the error caused by the standard normal distribution replacing $q(z_T)$.

$$q(z_{T-1}) = \int q(z_{T-1}, z_T) dz_T = \int q(z_{T-1}|z_T) q(z_T) dz_T \tag{5.1}$$

$$\dots$$

$$q(z_{t-1}) = \int q(z_{t-1}, z_t) dz_t = \int q(z_{t-1}|z_t) q(z_t) dz_t \tag{5.2}$$

$$\dots$$

$$q(z_1) = \int q(z_1, z_2) dz_1 = \int q(z_1|z_2) q(z_2) dz_2 \tag{5.3}$$

$$q(x) = \int q(x, z_1) dz_1 = \int q(x|z_1) q(z_1) dz_1 \tag{5.4}$$

In this article, the aforementioned transform is referred to as the **Posterior Transform**. For example, in equation 5.4, the input of the transform is the probability distribution function $q(z_1)$, and the output is the probability distribution function $q(x)$.The entire transform is determined by the posterior $q(x|z_1)$. This transform can also be considered as the linear weighted sum of a set of basis functions, where the basis functions are $q(x|z_1)$ under different z_1, and the weights of each basis function are $q(z_1)$. Some interesting properties of this transform will be introduced in Section 7.

In Section 3, we have considered two special posterior probability distributions. Next, we analyze their corresponding *posterior transforms*.

- When $\alpha \to 0$, the $q(x|z)$ for different z are almost the same as $q(x)$. In other words, the basis functions of linear weighted sum are almost the same. In this state, no matter how the input changes, the output of the transformation is always $q(x)$.
- When $\alpha \to 1$, the $q(x|z)$ for different z values becomes a series of Dirac delta functions and zero functions. In this state, as long as the *support set* of the input distribution is included in the *support set* of $q(x)$, the output of the transformation will remain the same with the input.

In Section 4, it is mentioned that the 1000 transformations used in the DDPM[2] can be represented using a single transformation

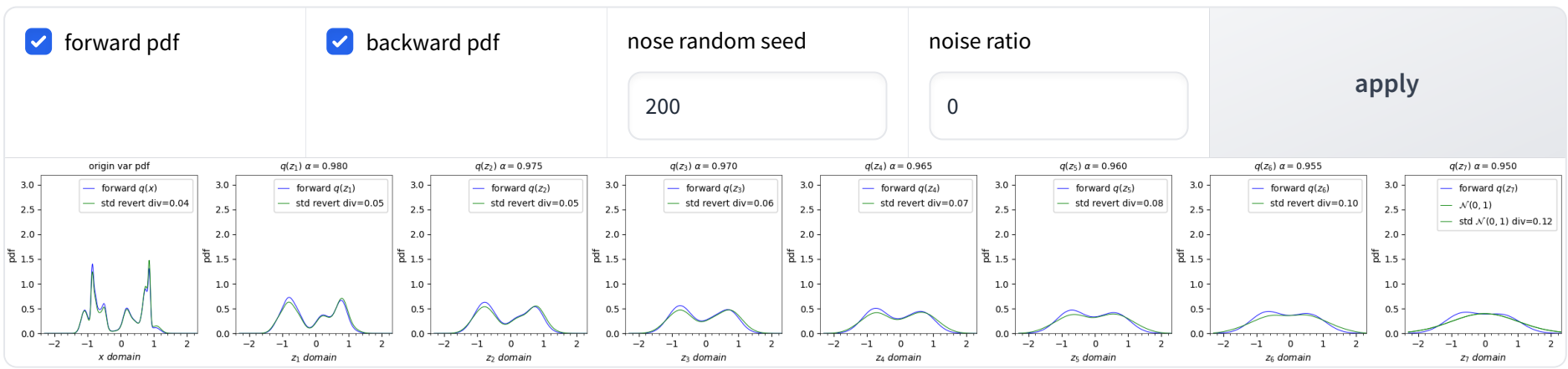
$$Z_T = \sqrt{0.0000403}\, X + \sqrt{1 - 0.0000403}\, \epsilon = 0.00635\, X + 0.99998\, \epsilon \tag{5.5}$$

Since $\alpha = 0.0000403$ is very small, the corresponding standard deviation of GaussFun (Equation 3.4) reaches 157.52. However, the range of X is limited within $[-1, 1]$, which is far smaller than the standard deviation of GaussFun. Within the range of $x \in [-1, 1]$, GaussFun should be close to a constant, showing little variation. Therefore, the $q(x|z_T)$ corresponding to different z_T are almost the same as $q(x)$. In this state, the posterior transform corresponding to $q(x|z_T)$ does not depend on the input distribution, the output distribution will always be $q(x)$.

Therefore, theoretically, in the DDPM model, it is not necessary to use the standard normal distribution to replace $q(z_T)$. Any other arbitrary distributions can also be used as a substitute.

Readers can conduct a similar experiment themselves. In Demo 3.1, set *start_alpha* to 0.25, *end_alpha* to 0.25, and *step* to 7. At this point, $q(z_7) = \sqrt{0.000061}X + \sqrt{1 - 0.000061}\epsilon$, which is roughly equivalent to DDPM's $q(z_T)$. Click on **apply** to perform the forward transform (plotted using blue curves), which prepares for the subsequent restoring process. In Demo 3.2, set the *noise_ratio* to 1, introducing 100% noise into the *tail distribution* $q(z_7)$. Changing the value of *nose_random_seed* will change the distribution of noise. Deselect *backward_pdf* to reduce screen clutter. Click on **apply** to restore $q(x)$ through posterior transform. You will see that, no matter what the shape of input $q(z_7)$ may be, the restored $q(x)$ is always exactly the same as the original $q(x)$. The JS Divergence is zero. The restoration process is plotted using a red curve.

Demo 3.2 - Recover From Normal Distribution Iteratively



6. Fitting Posterior With Conditional Gaussian Model

From the front part of Section 3, it is known that the posterior probability distributions are unknown and related to $q(x)$. Therefore, in order to recover the data distribution or sample from it, it is necessary to learn and estimate each posterior probability distribution.

From the latter part of Section 3, it can be understood that when certain conditions are met, each posterior probability distribution $q(x|z), q(z_{t-1}|z_t)$ approximates the Gaussian probability distribution. Therefore, by constructing a set of conditional Gaussian probability models $p(x|z), p(z_{t-1}|z_t)$, we can learn to fit the corresponding $q(x|z), q(z_{t-1}|z_t)$.

Due to the limitations of the model's representative and learning capabilities, there will be certain errors in the fitting process, which will further impact the accuracy of restored $q(x)$. The size of the fitting error is related to the complexity of the posterior probability distribution. As can be seen from Section 3, when $q(x)$ is more complex or the added noise is large, the posterior probability distribution will be more complex, and it will differ greatly from the Gaussian distribution, thus leading to fitting errors and further affecting the restoration of $q(x)$.

Refer to Demo 3.3 for the specifics. The reader can test different $q(x)$ and α, observe the fitting degree of the posterior probability distribution $q(z_{t-1}|z_t)$ and the accuracy of restored $q(x)$. The restored probability distribution is ploted with orange, and the error is also measured by JS divergence.

Regarding the objective function for fitting, similar to other probability models, the cross-entropy loss can be optimized to make $p(z_{t-1}|z_t)$ approaching $q(z_{t-1}|z_t)$. Since $(z_{t-1}|z_t)$ is a conditional probability, it is necessary to fully consider all conditions. This can be achieved by averaging the cross-entropy corresponding to each condition weighted by the probability of each condition happening. The final form of the loss function is as follows.

$$loss = -\int q(z_t) \overbrace{\int q(z_{t-1}|z_t) \log p(z_{t-1}|z_t) dz_{t-1}}^{\text{Cross Entropy}} dz_t \tag{6.1}$$

$$= -\iint q(z_{t-1}, z_t) \log p(z_{t-1}|z_t) dz_{t-1} dz_t \tag{6.2}$$

KL divergence can also be optimized as the objective function. KL divergence and cross-entropy are equivalent[10]

$$loss = \int q(z_t) KL(q(z_{t-1}|z_t) \| p(z_{t-1}|z_t)) dz_t \tag{6.3}$$

$$= \int q(z_t) \int q(z_{t-1}|z_t) \log \frac{q(z_{t-1}|z_t)}{p(z_{t-1}|z_t)} dz_{t-1} dz_t \tag{6.4}$$

$$= -\int q(z_t) \underbrace{\int q(z_{t-1}|z_t) \log p(z_{t-1}|z_t) dz_{t-1}}_{\text{Cross Entropy}} dz_t + \underbrace{\int q(z_t) \int q(z_{t-1}|z_t) \log q(z_{t-1}|z_t) dz}_{\text{Is Constant}} \tag{6.5}$$

The integral in equation 6.2 does not have a closed form and cannot be directly optimized. The Monte Carlo integration can be used for approximate calculation. The new objective function is as follows:

$$loss = -\iint q(z_{t-1}, z_t) \log p(z_{t-1}|z_t) dz_{t-1} dz_t \tag{6.6}$$

$$\approx -\sum_{i=0}^{N} \log p(Z_{t-1}^i | Z_t^i) \qquad where \quad (Z_{t-1}^i, Z_t^i) \sim q(z_{t-1}, z_t) \tag{6.7}$$

The aforementioned samples (Z_{t-1}^i, Z_t^i) follow a joint probability distribution $q(z_{t-1}, z_t)$, which can be sampled via an **Ancestral Sampling**. The specific method is as follows: sample $X, Z_1, Z_2 \dots Z_{t-1}, Z_t$ step by step through forward transforms (Formulas 4.1~4.4), and then reserve (Z_{t-1}, Z_t) as a sample. This sampling process is relatively slow. To speed up the sampling, we can take advantage of the known features of the probability distribution $q(z_t|x)$ (Formula 4.8). First, sample X from $q(x)$, then sample Z_{t-1} from $q(z_{t-1}|x)$, and finally sample Z_t from $q(z_t|z_{t-1})$. Thus, a sample (Z_{t-1}, Z_t) is obtained.

Some people may question that the objective function in Equation 6.3 seems different from those in the DPM[1] and DDPM[2] papers. In fact, these two objective functions are equivalent, and the proof is given below.

For **Consistent Terms**, the proof is as follows:

$$loss = -\iint q(z_{t-1}, z_t) \ \log p(z_{t-1}|z_t) dz_{t-1} dz_t \tag{6.8}$$

$$= -\iint \int q(x) q(z_{t-1}, z_t|x) dx \ \log p(z_{t-1}|z_t) dz_{t-1} dz_t \tag{6.9}$$

$$= \overbrace{\iint \int q(x) q(z_{t-1}, z_t|x) \log q(z_{t-1}|z_t, x) dx dz_{t-1} dz_t}^{\text{This Term Is Constant And Is Denoted As } C_1} \tag{6.10}$$

$$- \iint \int q(x) q(z_{t-1}, z_t|x) \log p(z_{t-1}|z_t) dx dz_{t-1} dz_t - C_1 \tag{6.11}$$

$$= \iint \int q(x) q(z_{t-1}, z_t|x) \log \frac{q(z_{t-1}|z_t, x)}{p(z_{t-1}|z_t)} dx dz_{t-1} dz_t - C_1 \tag{6.12}$$

$$= \iint q(x) q(z_t|x) \int q(z_{t-1}|z_t, x) \log \frac{q(z_{t-1}|z_t, x)}{p(z_{t-1}|z_t)} dz_{t-1} \ dz_t dx - C_1 \tag{6.13}$$

$$= \iint q(x) q(z_t|x) KL[q(z_{t-1}|z_t, x) \| p(z_{t-1}|z_t)] dz_t dx - C_1 \tag{6.14}$$

$$\propto \iint q(x) q(z_t|x) KL(q(z_{t-1}|z_t, x) \| p(z_{t-1}|z_t)) dz_t dx \tag{6.15}$$

In the above formula, the term C_1 is a fixed value, which does not contain parameters to be optimized. Here, $q(x)$ is a fixed probability distribution, and $q(z_{t-1}|z_t)$ is also a fixed probability distribution, whose specific form is determined by $q(x)$ and the coefficient α.

For the **Reconstruction Term**, it can be proven in a similar way.

$$loss = -\int q(z_1) \overbrace{\int q(x|z_1) \log p(x|z_1) dx}^{\text{Cross Entropy}} dz_1 \tag{6.16}$$

$$= -\iint q(z_1, x) \log p(x|z_1) dx dz_1 \tag{6.17}$$

$$= -\int q(x) \int q(z_1|x) \log p(x|z_1) dz_1 \, dx \tag{6.18}$$

Therefore, the objective function in equation 6.1 is equivalent with the DPM original objective function.

Based on the conclusion of the Consistent Terms proof and the relationship between cross entropy and KL divergence, an interesting conclusion can be drawn:

$$\min_p \int q(z_t) KL(q(z_{t-1}|z_t) \| p(z_{t-1}|z_t)) dz_t \iff \min_p \iint q(z_t) q(x|z_t) KL(q(z_{t-1}|z_t, x) \| p(z_{t-1}|z_t)) dx dz_t \tag{6.19}$$

By comparing the expressions on the left and right, it can be observed that the objective function on the right side includes an additional variable X compared to the left side. At the same time, there is an additional integral with respect to X, with the occurrence probability of X, denoted as $q(x|z_t)$, serving as the weighting coefficient for the integral.

Following a similar proof method, a more general relationship can be derived:

$$\min_p KL(q(z) \| p(z)) \iff \min_p \int q(x) KL(q(z|x) \| p(z)) dx \tag{6.20}$$

A detailed derivation of this conclusion can be found in Appendix A.

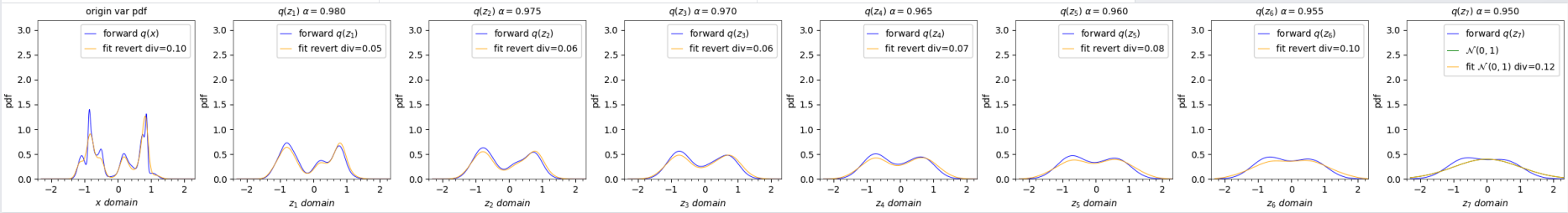
Demo 3.3 - Fitting Posterior with Conditional Gaussian Model

show forward pdf — forward pdf

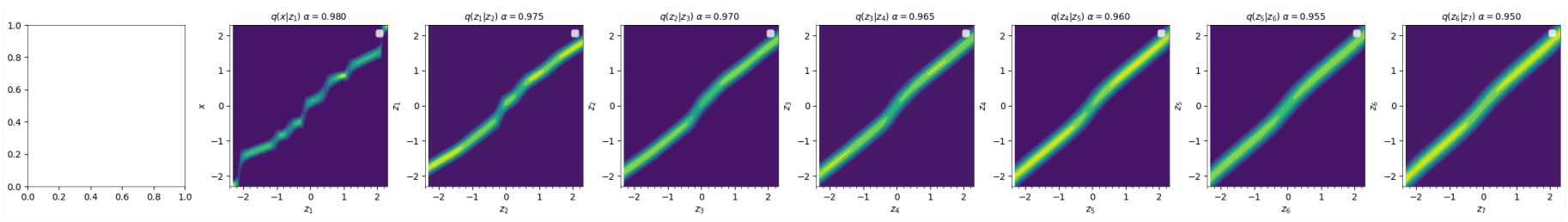
show origin backward pdf — backward pdf

show backward pdf after fitting posterior with conditonal Gaussian — fitted posterior

apply



fitted posterior's pdf



7. Posterior Transform

Contraction Mapping and Converging Point

$$q(x) = \int q(x, z) dz = \int q(x|z) q(z) dz \tag{7.1}$$

Through extensive experiments with one-dimensional random variables, it was found that the **Posterior Transform** exhibits the characteristics of **Contraction Mapping**. This means that, for any two probability distributions $q_{i1}(z)$ and $q_{i2}(z)$, after posterior transform, we get $q_{o1}(x)$ and $q_{o2}(x)$. The distance between $q_{o1}(x)$ and $q_{o2}(x)$ is always less than the distance between $q_{i1}(x)$ and $q_{i2}(x)$. Here, the distance can be measured using JS divergence or Total Variance. Furthermore, the contractive ratio of this contraction mapping is positively related to the size of the added noise.

$$dist(q_{o1}(z),\ q_{o2}(z)) < dist(q_{i1}(x),\ q_{i2}(x)) \tag{7.2}$$

Readers can refer to Demo 4.1, where the first three figure present a transform process. The first figure is an arbitrary data distribution $q(x)$, the third figure is the transformed probability distribution, and second figure is the posterior probability distribution $q(x|z)$. You can change the random seed to generate a new data distribution$q(x)$, and adjust the value of α to introduce different degrees of noise.

The last two figures show contraction of the transform. The fourth figure displays two randomly generated input distributions and their distance, div_{in}. The fifth figure displays the two output distributions after transform, with the distance denoted as div_{out}.

Readers can change the input random seed to toggle different inputs. It can be observed from the figures that div_{in} is always smaller than div_{out} for any input. Additionally, if you change the value of α, you will see that the smaller the α(larger noise), the smaller the ratio of div_{out}/div_{in}, indicating a larger rate of contraction.

According to the Banach fixed-point theorem[5], a contraction mapping has a unique fixed point (converged point). That is to say, for any input distribution, the **Posterior Transform** can be applied continuously through iterations, and as long as the number of iterations is sufficient, the final output would be the same distribution. After a large number of one-dimensional random variable experiments, it was found that the fixed point (converged point) is **located near** $q(x)$. Also, the location is related to the value of α; the smaller α (larger noise), the closer it is.

Readers can refer to Demo 4.2, which illustrates an example of applying posterior transform iteratively. Choose an appropriate number of iterations, and click on the button of *Apply*, and the iteration process will be draw step by step. Each subplot shows the transformed output distribution(green curve) from each transform, with the reference distribution $q(x)$ expressed as a blue curve, as well as the distance div between the output distribution and $q(x)$. It can be seen that as the number of iterations increases, the output distribution becomes more and more similar to $q(x)$, and will eventually stabilize near $q(x)$. For more complicated distributions, more iterations or greater noise may be required. The maximum number of iterations can be set to tens of thousands, but it'll take longer.

For the one-dimensional discrete case, $q(x|z)$ is discretized into a matrix (denoted as $Q_{x|z}$), $q(z)$ is discretized into a vector (denoted as q_i). The integration operation $\int q(x|z)q(z)dz$ is discretized into a **matrix-vector** multiplication operation, thus the posterior transform can be written as

$$q_o = Q_{x|z}\ q_i \qquad \text{1 iteration} \tag{7.3}$$
$$q_o = Q_{x|z}\ Q_{x|z}\ q_i \qquad \text{2 iteration} \tag{7.4}$$
$$\cdots$$
$$q_o = (Q_{x|z})^n\ q_i \qquad \text{n iteration} \tag{7.5}$$

In order to better understand the property of the transform, the matrix $(Q_{x|z})^n$ is also plotted in Demo 4.2. From the demo we can see that, as the iterations converge, the row vectors of the matrix $(Q_{x|z})^n$ will become a constant vector, that is, all components of the vector will be the same, which will appear as a horizontal line in the denisty plot.

In the Appendix B, a proof will be provided that, when $q(x)$ and α satisfy some conditions, the posterior transform is a strict Contraction Mapping.

The relationship between the converged distribution and the input distribution q(x) cannot be rigorously proven at present.

Anti-noise Capacity In Restoring Data Distribution

From the above analysis, we know that when certain conditions are satisfied, the *posterior transform* is a contraction mapping. Therefore, the following relationship exists:

$$dist(q(x),\ q_o(x)) < dist(q(z),\ q_i(z)) \tag{7.12}$$

Wherein, $q(z)$ is the ideal input distribution, $q(x)$ is the ideal output distribution, $q_i(x)$ is any arbitrary input distribution, and $q_o(x)$ is the output distribution obtained after transforming $q_i(z)$.

The above equation indicates that the distance between the output distribution $q_o(x)$ and the ideal output distribution q(x) will always be **less than** the distance between the input distribution $q_i(z)$ and the ideal input distribution q(x). Hence, the *posterior transform* has certain resistance to noise. This means that during the process of restoring $q(x)$(Section 5), even if the *tail distribution* $q(z_T)$ contains some error, the error of the outputed distribution $q(x)$ will be smaller than the error of input after undergoing a series of transform.

Refer specifically to Demo 3.2, where by increasing the value of the **noise ratio**, noise can be added to the *tail distribution* $q(z_T)$. Clicking the "apply" button will gradually draw out the restoring process, with the restored distribution represented by a red curve, and the error size will be computed by the JS divergence. You will see that the error of restored $q(x)$ is always less than the error of $q(z_T)$.

From the above discussion, we know that the smaller the α (the larger the noise used in the transform process), the greater the contractive ratio of the contraction mapping, and thus, the stronger the ability to resist noise.

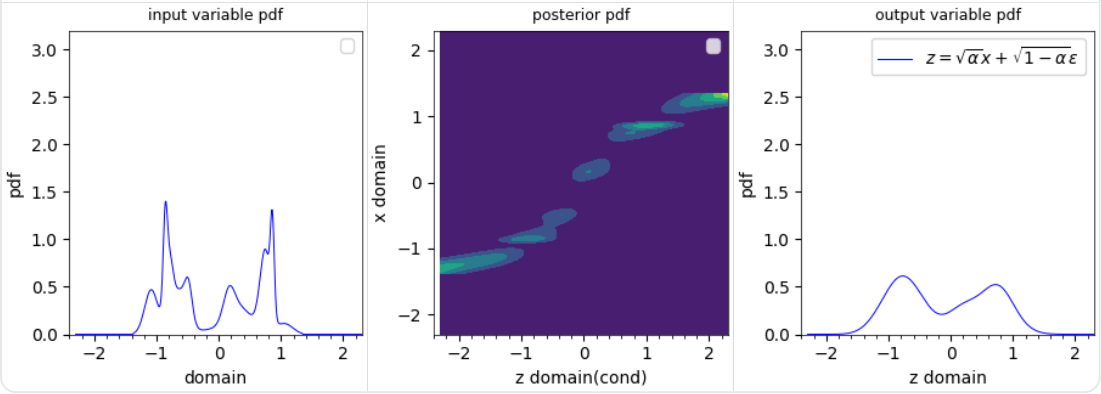
Demo 4.1 - Posterior Transform is a Contraction Mapping

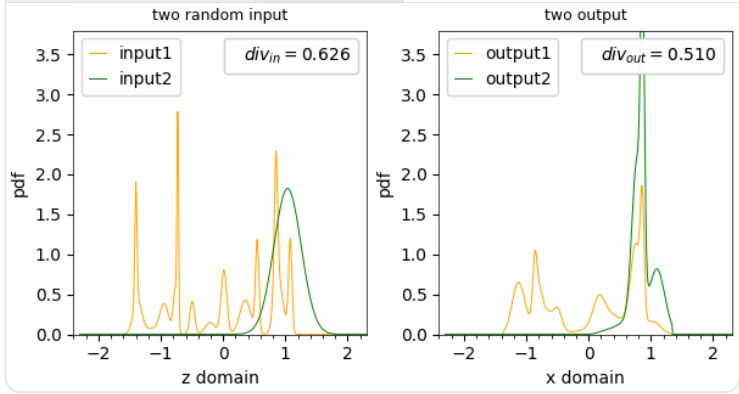
random seed: 100

alpha: 0.95

change inputs seed

two inputs random seed: 865677





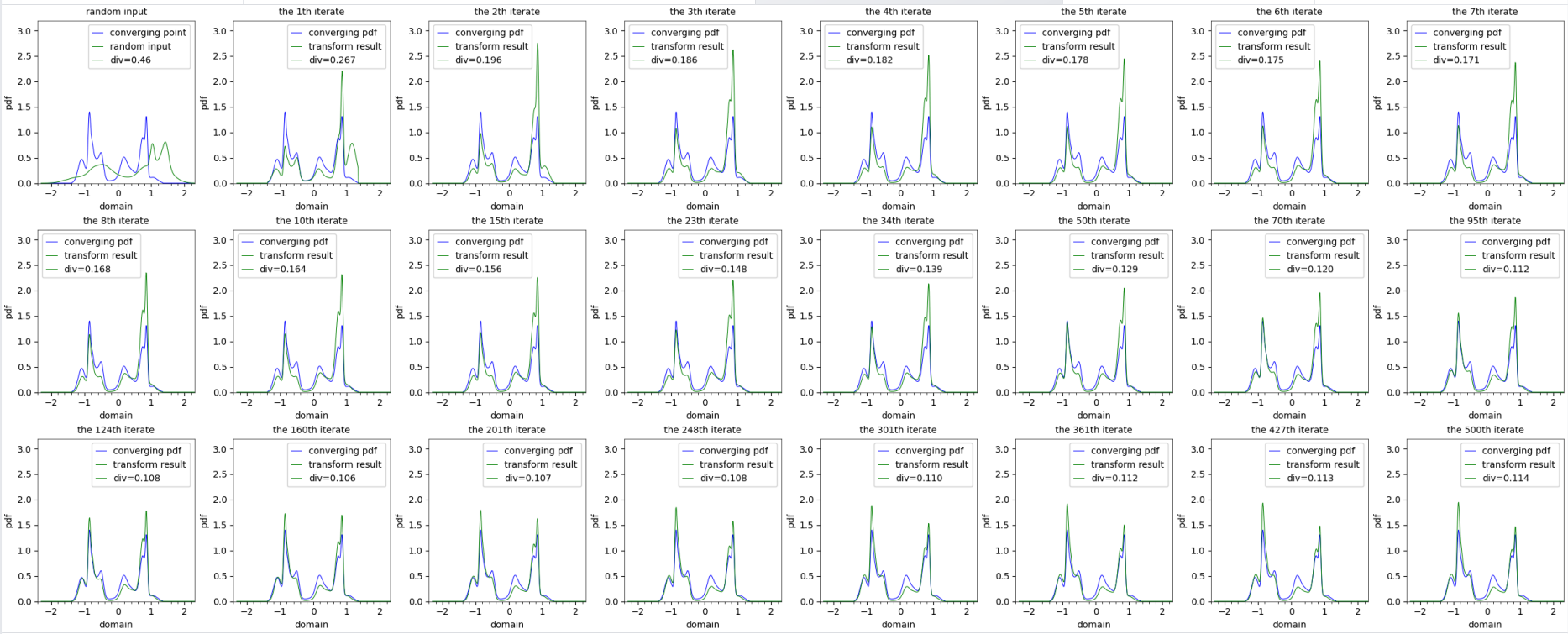
Demo 4.2 - Posterior Transform Have a Converging Point

input seed: 200

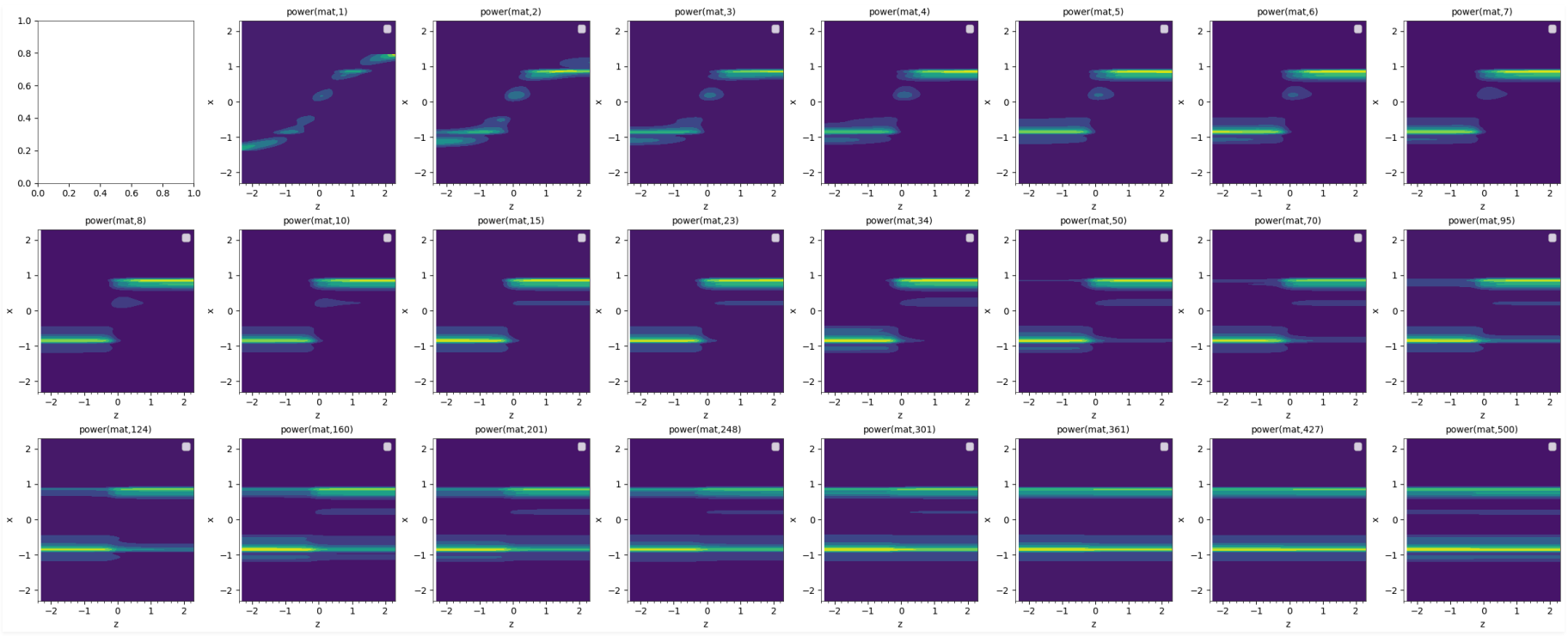
iterate number: 500

show power matrix

apply iteration transform



power matrix of posterior



8. Can the data distribution be restored by deconvolution?

As mentioned in the Section 1, the transform of Equation 2.1 can be divided into two sub-transforms, the first one being a linear transform and the second being adding independent Gaussian noise. The linear transform is equivalent to a scaling transform of the probability distribution, so it has an inverse transformation. Adding independent Gaussian noise is equivalent to the execution of a convolution operation on the probability distribution, which can be restored through **deconvolution**. Therefore, theoretically, the data distribution $q(x)$ can be recovered from the final probability distribution $q(z_T)$ through **inverse linear transform** and **deconvolution**.

However, in actuality, some problems do exist. Due to the extreme sensitivity of deconvolution to errors, having high input sensitivity, even a small amount of input noise can lead to significant changes in output[11][12]. Meanwhile, in the diffusion model, the standard normal distribution is used as an approximation to replace $q(z_T)$, thus, noise is introduced at the initial stage of restoring. Although the noise is relatively small, because of the sensitivity of deconvolution, the noise will gradually amplify, affecting the restoring.

In addition, the infeasibility of **deconvolution restoring** can be understood from another perspective. Since the process of forward transform (equations 4.1 to 4.4) is fixed, the convolution kernel is fixed. Therefore, the corresponding deconvolution transform is also fixed. Since the initial data distribution $q(x)$ is arbitrary, any probability distribution can be transformed into an approximation of $\mathcal{N}(0, I)$ through a series of fixed linear transforms and convolutions. If **deconvolution restoring** is feasible, it means that a fixed deconvolution can be used to restore any data distribution $q(x)$ from the $\mathcal{N}(0, I)$, this is clearly **paradoxical**. The same input, the same transform, cannot have multiple different outputs.

Appendix A Conditional KL Divergence

This section mainly introduces the relationship between **KL divergence** and **conditional KL divergence**. Before the formal introduction, we will briefly introduce the definitions of **Entropy** and **Conditional Entropy**, as well as the inequality relationship between them, in preparation for the subsequent proof.

Entropy and Conditional Entropy

For any two random variables Z, X, the **Entropy** is defined as follows[16]:

$$\mathrm{H}(Z) = \int -p(z)\log p(z)dz \tag{A.1}$$

The **Conditional Entropy** is defined as followed [17]:

$$\mathrm{H}(Z|X) = \int p(x)\overbrace{\int -p(z|x)\log p(z|x)dz}^{\text{Entropy}}\,dx \tag{A.2}$$

The following inequality relationship exists between the two:

$$\mathrm{H}(Z|X) \leq \mathrm{H}(Z) \tag{A.3}$$

It is to say that **the Conditional Entropy is always less than or equal to the Entropy**, and they are equal only when X and Z are independent. The proof of this relationship can be found in the literature [17].

KL Divergence and Conditional KL Divergence

In the same manner as the definition of Conditional Entropy, we introduce a new definition, **Conditional KL Divergence**, denoted as $KL_{\mathcal{C}}$. Since KL Divergence is non-symmetric, there exist two forms as follows.

$$KL_{\mathcal{C}}(q(z|x)\|p(z)) = \int q(x)KL(q(z|x)\|p(z))dx \tag{A.4}$$

$$KL_{\mathcal{C}}(q(z)\|p(z|x)) = \int p(x)KL(q(z)\|p(z|x))dx \tag{A.5}$$

Similar to Conditional Entropy, there also exists a similar inequality relationship for **both forms of Conditional KL Divergence**:

$$KL_{\mathcal{C}}(q(z|x)\|p(z)) \geq KL(q(z)\|p(z)) \tag{A.6}$$

$$KL_{\mathcal{C}}(q(z)\|p(z|x)) \geq KL(q(z)\|p(z)) \tag{A.7}$$

It is to say that **the Conditional KL Divergence is always less than or equal to the KL Divergence**, and they are equal only when X and Z are independent.

The following provides proofs for the conclusions on Equation A.5 and Equation A.6 respectively.

For equation A.6, the proof is as follows:

$$
\begin{aligned}
KL_{\mathcal{C}}(q(z|x)\|p(z)) &= \int q(x)KL(q(z|x)\|p(z))dx && \text{(A.8)}\\
&= \iint q(x)q(z|x)\log\frac{q(z|x)}{p(z)}dzdx && \text{(A.9)}\\
&= -\overbrace{\iint -q(x)q(z|x)\log q(z|x)dzdx}^{\text{Conditional Entropy } \mathrm{H}_q(Z|X)} - \iint q(x)q(z|x)\log p(z)dzdx && \text{(A.10)}\\
&= -\mathrm{H}_q(Z|X) - \int\left\{\int q(x)q(z|x)dx\right\}\log p(z)dz && \text{(A.11)}\\
&= -\mathrm{H}_q(Z|X) + \overbrace{\int -q(z)\log p(z)dz}^{\text{Cross Entropy}} && \text{(A.12)}\\
&= -\mathrm{H}_q(Z|X) + \int q(z)\left\{\log\frac{q(z)}{p(z)} - \log q(z)\right\}dz && \text{(A.13)}\\
&= -\mathrm{H}_q(Z|X) + \int q(z)\log\frac{q(z)}{p(z)}dz + \overbrace{\int -q(z)\log q(z)dz}^{\text{Entropy } \mathrm{H}_q(Z)} && \text{(A.14)}\\
&= KL(q(z)\|p(z)) + \overbrace{\mathrm{H}_q(Z) - \mathrm{H}_q(Z|X)}^{\geq 0} && \text{(A.15)}\\
&\leq KL(q(z)\|p(z)) && \text{(A.16)}
\end{aligned}
$$

In this context, equation A.15 applies the conclusion that **Conditional Entropy is always less than or equal to Entropy**. Thus, the relationship in equation A.6 is derived.

For equation A.6, the proof is as follows:

$$
\begin{aligned}
KL(q(z)\|p(z)) &= \int q(z)\log\frac{q(z)}{p(z)}dz && \text{(A.15)}\\
&= \int q(z)\log\frac{q(z)}{\int p(z|x)p(x)dx}dz && \text{(A.16)}\\
&= \int p(x)dx\int q(z)\log q(z)dz - \int q(z)\log\int p(z|x)p(x)dxdz \qquad \int p(x)dx = 1 && \text{(A.17)}\\
&\leq \iint p(x)q(z)\log q(z)dzdx - \int q(z)\int p(x)\log p(z|x)dxdz \qquad \text{jensen inequality} && \text{(A.18)}\\
&= \iint p(x)q(z)\log q(z)dzdx - \iint p(x)q(z)\log p(z|x)dzdx && \text{(A.19)}\\
&= \iint p(x)q(z)(\log q(z) - \log p(z|x))dzdx && \text{(A.20)}\\
&= \iint p(x)q(z)\log\frac{q(z)}{p(z|x)}dzdx && \text{(A.21)}\\
&= \int p(x)\left\{\int q(z)\log\frac{q(z)}{p(z|x)}dz\right\}dx && \text{(A.22)}\\
&= \int p(x)KL(q(z)\|p(z|x))dx && \text{(A.23)}\\
&= KL_{\mathcal{C}}(q(z)\|p(z|x)) && \text{(A.24)}
\end{aligned}
$$

Thus, the relationship in equation A.7 is obtained.

Another **important conclusion** can be drawn from equation A.15.

The KL Divergence is often used to fit the distribution of data. In this scenario, the distribution of the data is denoted by $q(z)$ and the parameterized model distribution is denoted by $p_\theta(z)$. During the optimization process, since both $q(z|x)$ and $q(x)$ remain constant, the term $\mathrm{H}(Z) - \mathrm{H}(Z|X)$ in Equation A.15 is a constant. Thus, the following relationship is obtained:

$$
\min_{p_\theta} KL(q(z)\|p_\theta(z)) \iff \min_{p_\theta}\int q(x)KL(q(z|x)\|p_\theta(z))dx \qquad \text{(A.25)}
$$

Comparing the above relationship with **Denoised Score Matching** [18](equation A.26), some similarities can be observed. Both introduce a new variable X, and substitute the targeted fitting distribution q(z) with q(z|x). After the substitution, since q(z|x) is a conditional probability distribution, both consider all conditions and perform a weighted sum using the probability of the conditions occurring, $q(x)$, as the weight coefficient.

$$
\min_{\psi_\theta}\frac{1}{2}\int q(z)\left\|\psi_\theta(z) - \frac{\partial q(z)}{\partial z}\right\|^2 dz \iff \min_{\psi_\theta}\int q(x)\overbrace{\frac{1}{2}\int q(z|x)\left\|\psi_\theta(z) - \frac{\partial q(z|x)}{\partial z}\right\|^2 dz}^{\text{Score Matching of } q(z|x)}dx \qquad \text{(A.26)}
$$

The operation of the above weighted sum is somewhat similar to *Elimination by Total Probability Formula* .

$$
q(z) = \int q(z,x)dx = \int q(x)q(z|x)dx \qquad \text{(A.27)}
$$

Appendix B Proof of Contraction

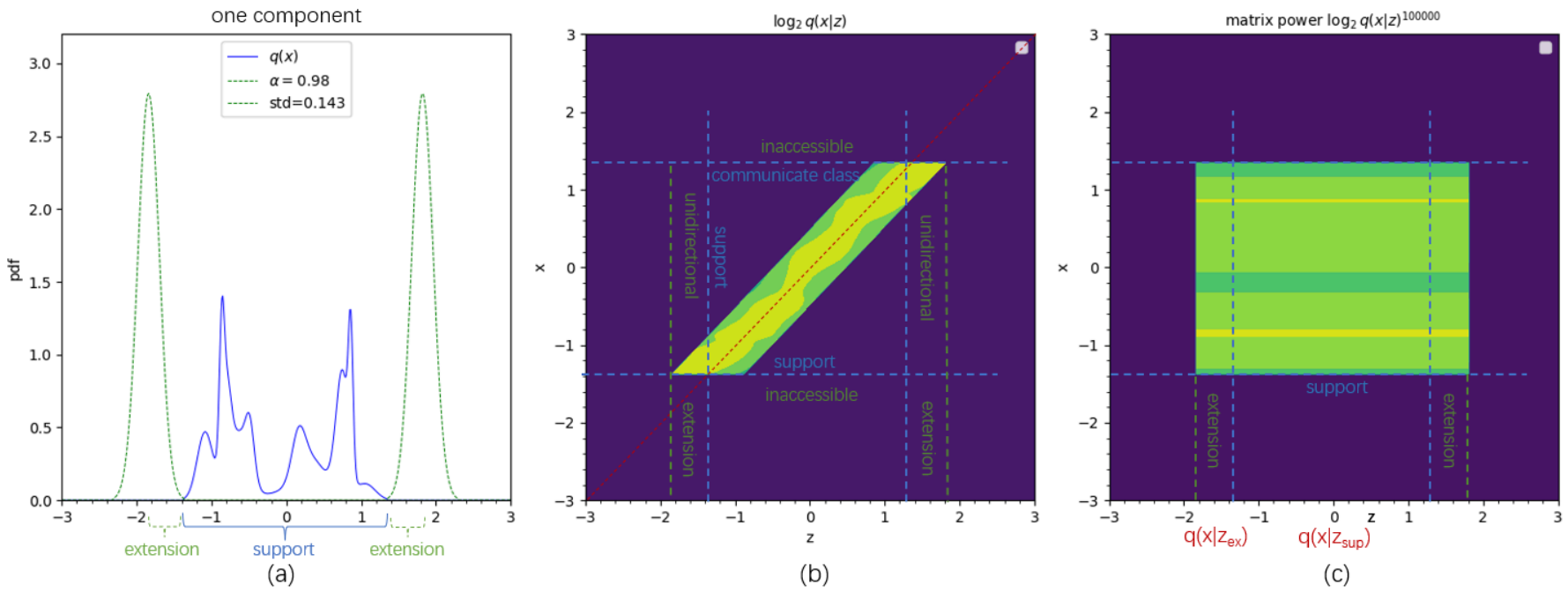


Figure 2: Only one component in support

The following will prove that with some conditions, the posterior transform is a contraction mapping, and there exists a unique point, which is also the converged point.

The proof will be divided into several cases, and assumes that the random variable is discrete, so the posterior transform can be regarded as a single step transition of a **discrete Markov Chain**. The posterior $q(x|z)$ corresponds to the **transfer matrix**. Continuous variables can be considered as discrete variables with infinite states.

1. When $q(x)$ is greater than 0, the posterior transform matrix $q(x|z)$ will be greater than 0 too. Therefore, this matrix is the transition matrix of an irreducible aperiodic Markov Chain. According to the conclusion of the literature [13], this transformation is a contraction mapping with respect to Total Variance metric. Therefore, according to the Banach fixed-point theorem, this transformation has a unique fixed point(converged point).
2. When $q(x)$ is partially greater than 0, and the support of $q(x)$ (the region where $q(x)$ is greater than 0) consists only one connected component (Figure 2), several conclusions can be drawn from equation (3.4):
    a. When z and x are within the support set, since both $q(x)$ and GaussFun are greater than 0, the diagonal elements of the transfer matrix $\{q(x|z)|z = x\}$ are greater than 0. This means that the state within the support set is aperiodic.
    b. When z and x are within the support set, since GaussFun's support set has a certain range, elements above and below the diagonal $\{q(x|z)|x = z + \epsilon\}$is also greater than 0. This means that states within the support set are accessible to each other, forming a Communication Class[14], see in Figure 2b.
    c. When z *is within the support set* and x *is outside the support set*, $q(x|z)$ is entirely 0. This means that the state within the support set is *inaccessible* to the state outside the support set (Inaccessible Region in Figure 2b)
    d. When z *is outside the support set* and x *is inside the support set*, due to the existence of a certain range of the support set of GaussFun, there are some extension areas (Extension Region in Figure 2b), where the corresponding $\{q(x|z)|x \in support\}$ is not all zero. This means that the state of this part of the extension area can *unidirectionally* access the state inside the support set (Unidirectional Region in Figure 2b).
    e. When z *is outside the support set* and x *is outside the support set*, the corresponding $q(x|z)$ is entirely zero. This implies that, states outside the support set will not transit to states outside the support set. In other words, states outside the support set only originate from states within the support set.

    From (c), we know that states within the support set will not transition to states outside of the support set. From (a) and (b), we know that the states within the support set are non-periodic and form a Communicate Class. Therefore, the states within the support set independently form an irreducible and non-periodic Markov Chain. According to the conclusion of Theorem 11.4.1 in reference [7], as $n \to \infty$, $q(x|z)^n$ will converge to a constant matrix, with each column vector in the matrix being identical. This implies that for different values of z, $q(x|z)^n$ are the same (as seen in Figure 2c). In Addition, according to (d) and (e), there exist some states z, which are outside of the support set, that can transition into the support set and will carry information from within the support set back to the outside. Thus, the corresponding $q(x|z)^n$ for these z states (the $q(x|z_{ex})$ region in Figure 2c) will equal the corresponding $q(x|z)^n$ in the support set (the $q(x|z_{sup})$ region in Figure 2c).

    Therefore, it can be concluded that when the state is confined within the support set and two extension regions, $\lim_{n\to\infty} q(x|z)^n$ will converge to a fixed matrix, and each column vector is identical. Hence, for any input distribution, if posterior transforms are continuously applied, it will eventually converge to a fixed distribution, which is equal to the column vector of the converged matrix. Based on the conclusion from the literature [9], when a iterative transform converges to a unique fixed point, this transform is a Contraction Mapping with respect to a certain metric.
3. When $q(x)$ is partially greater than 0, and multiple connected component exist in the support set of $q(x)$, and the maximum distance of each connected component **can** be covered by the support set of corresponding GaussFun, the states within each

connected domain **constitute only one Communicate Class**. As shown in Figure 3, $q(x)$ has two connected component. On the edge of the first component, the support set of GaussFun corresponding to $q(x|z=-0.3)$ can span the gap to reach the second component, so the states of the first component can *access* the states of the second component. On the edge of the second component, the support set of GaussFun corresponding to $q(x|z=0)$ can also span the gap to reach the first. Thus, the states of the second component can *access* the states of the first component, so these two component form a Communicate Class. Therefore, similar to the case with a single component, when states are confined to each component, gaps, and extension areas, the posterior transform has a unique iterative convergence point, which is a contraction mapping with respect to a certain metric.

4. When $q(x)$ is partially greater than 0, and multiple connected component exist in the support set of $q(x)$, and the maximum distance of each connected component **cannot** be covered by the support set of corresponding GaussFun, the states within each component **constitute multiple Communicate Classes**, as shown in Figure 4. Under such circumstances, as $n \to \infty$, $q(x|z)^n$ will also converge to a fixed matrix, but not all the column vectors are identical. Therefore, the posterior transforma is not a strict contraction mapping. However, when the state of the input distribution is confined to a single Communicate Class and its corresponding extension, the posterior transform is also a contraction mapping with a unique convergence point.

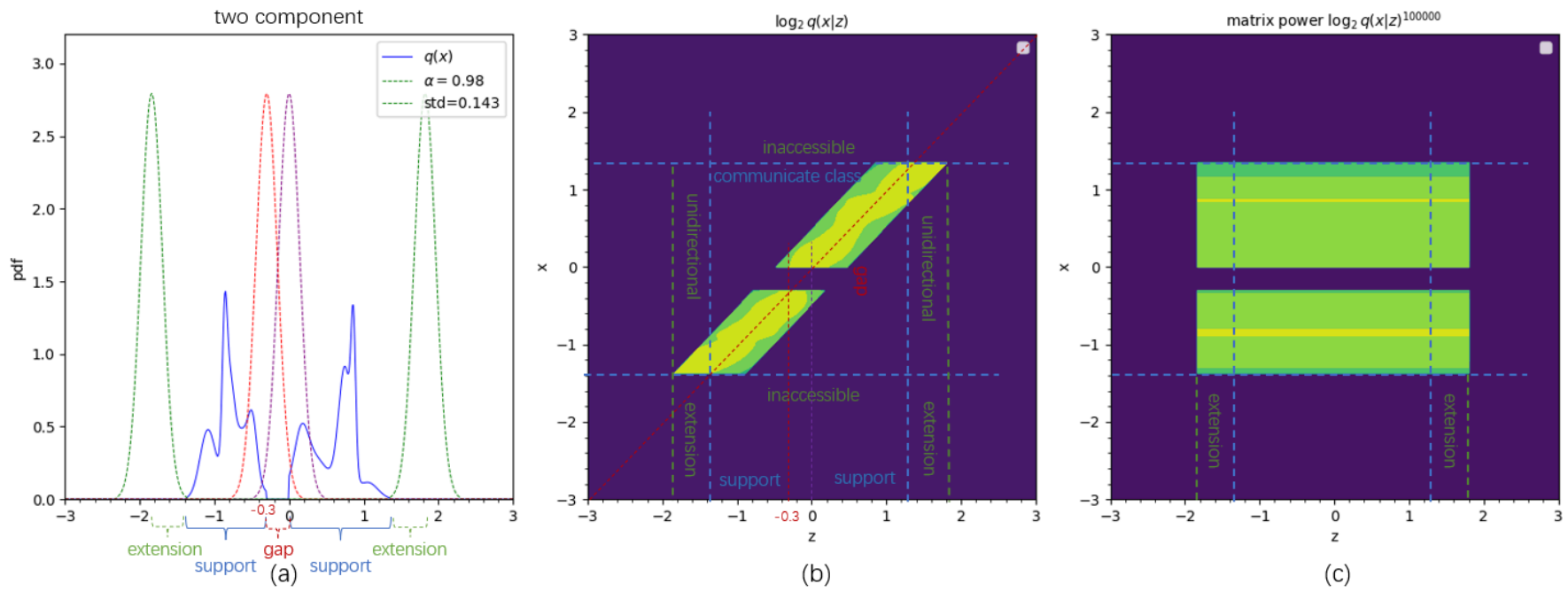


Figure 3: Two components which can communicate with each other

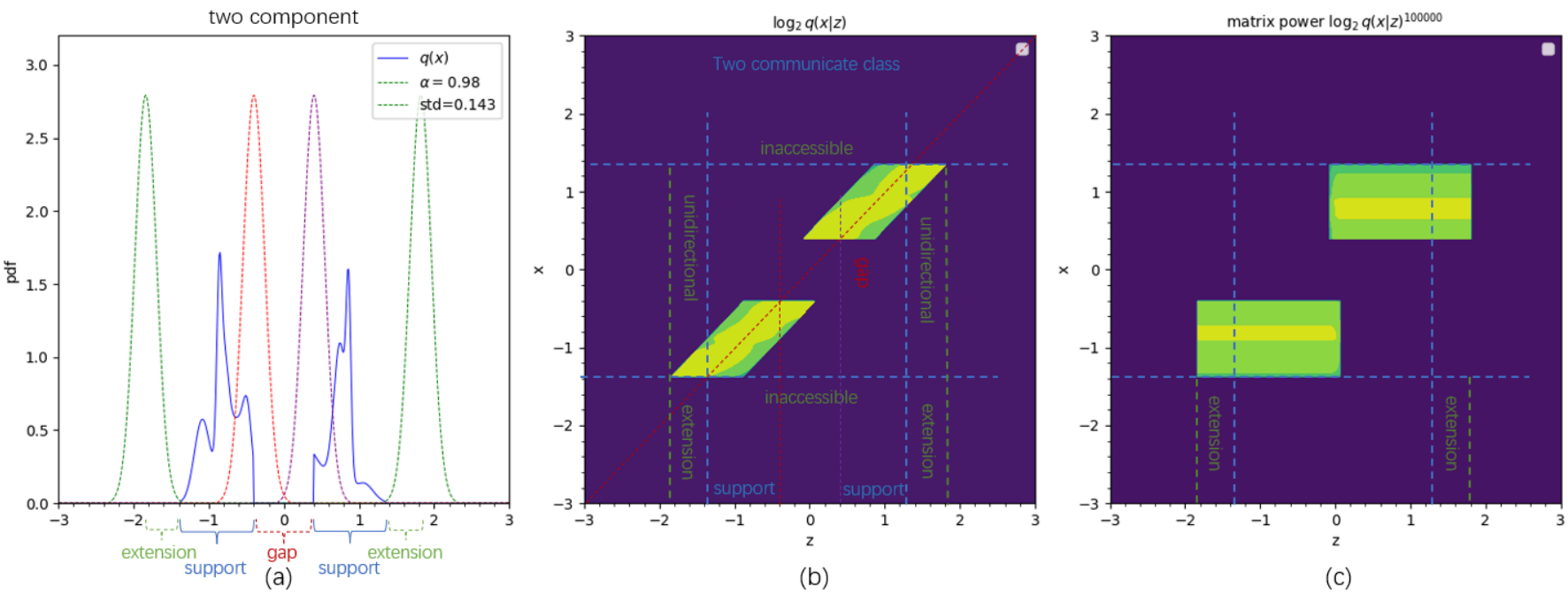


Figure 4: Two components which **cannot** communicate with each other

Additionally, there exists a more generalized relation about the posterior transform that is independent of $q(x|z)$: the Total Variance distance between two output distributions will always be **less than or equal to** the Total Variance distance between their corresponding input distributions, that is

$$dist(q_{o1}(x),\ q_{o2}(x)) <= dist(q_{i1}(z),\ q_{i2}(z)) \tag{B.1}$$

The proof is given below in discrete form:

$$\begin{aligned}
\|q_{o1} - q_{o2}\|_{TV} &= \|Q_{x|z}q_{i1} - Q_{x|z}q_{i2}\|_{TV} && (B.2)\\
&= \sum_m |\sum_n Q_{x|z}(m,n)q_{i1}(n) - \sum_n Q_{x|z}(m,n)q_{i2}(n)| && (B.3)\\
&= \sum_m |\sum_n Q_{x|z}(m,n)(q_{i1}(n) - q_{i2}(n))| && (B.4)\\
&\le \sum_m \sum_n Q_{x|z}(m,n)|(q_{i1}(n) - q_{i2}(n))| && \text{Absolute value inequality} \quad (B.5)\\
&= \sum_n |(q_{i1}(n) - q_{i2}(n))| \sum_m Q_{x|z}(m,n) && \sum_m Q_{x|z}(m,n) = 1 \quad (B.6)\\
&= \sum_n |(q_{i1}(n) - q_{i2}(n))| && (B.7)
\end{aligned}$$

In this context, $Q_{x|z}(m,n)$ represents the element at the m-th row and n-th column of the matrix $Q_{x|z}$, and $q_{i1}(n)$ represents the n-th element of the vector q_{i1}.

Reference

[1] Deep Unsupervised Learning Using Nonequilibrium Thermodynami

[2] Denoising Diffusion Probabilistic Models

[3] Linear Transformations of Random Variable

[4] Sums and Convolution

[5] Banach fixed-point theorem

[6] Contraction mapping

[7] Fundamental Limit Theorem for Regular Chains

[8] Markov Chain:Basic Theory - Proposition 6

[9] A Converse to Banach's Fixed Point Theorem and its CLS Completeness

[10] Cross-entropy minimization

[11] Deconvolution Using Frequency-Domain Division

[12] deconvolution-by-division-in-the-frequency-domain

[13] Markov Chain:Basic Theory - Theorem 7

[14] Markov Chain:Basic Theory - Definition 4

[15] Variational Diffusion Models

About

APP: This Web APP is developed using Gradio and deployed on HuggingFace. Due to limited resources (2 cores, 16G memory), the response may be slow. For a better experience, it is recommended to clone the source code from github and run it locally. This program only relies on Gradio, SciPy, and Matplotlib.

Author: Zhenxin Zheng, Senior computer vision engineer with ten years of algorithm development experience, Formerly employed by Tencent and JD.com, currently focusing on image and video generation.

Email: blair.star@163.com.