可逆生成模型及其高效算法研究

(申请清华大学工学博士学位论文)

培 养 单 位： 计算机科学与技术系

学　　科： 计算机科学与技术

研 究 生： 路　　橙

指 导 教 师： 朱　　军　教　授

二〇二三年十二月

可逆生成模型及其高效算法研究

路　橙

Research on Invertible Generative Models and Efficient Algorithms

Dissertation submitted to

Tsinghua University

in partial fulfillment of the requirement

for the degree of

Doctor of Philosophy

in

Computer Science and Technology

by

Lu Cheng

Dissertation Supervisor: Professor Jun Zhu

December, 2023

学位论文公开评阅人和答辩委员会名单

公开评阅人名单

王宏宁	副教授	清华大学
李崇轩	副教授	中国人民大学

答辩委员会名单

主席	赫然	研究员	中科院自动化研究所
委员	朱军	教授	清华大学
	陈键飞	副教授	清华大学
	王宏宁	副教授	清华大学
	李崇轩	副教授	中国人民大学
秘书	李建民	副研究员	清华大学

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：

清华大学拥有在著作权法规定范围内学位论文的使用权，其中包括：（1）已获学位的研究生必须按学校规定提交学位论文，学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（2）为教学和科研目的，学校可以将公开的学位论文作为资料在图书馆、资料室等场所供校内师生阅读，或在校园网上供校内师生浏览部分内容；（3）根据《中华人民共和国学位条例暂行实施办法》及上级教育主管部门具体要求，向国家图书馆报送相应的学位论文。

本人保证遵守上述规定。

作者签名：_____　　导师签名：_____

日　　期：_____　　日　　期：_____

摘 要

随着数字技术的普及，各种设备产生的复杂多样数据呈爆炸式增长，且通常缺乏高质量标注。近年来，基于深度学习技术的深度生成模型在理解和建模大规模无标注数据任务中取得了一系列进展。给定训练数据，深度生成模型旨在通过某种训练算法拟合数据分布，并通过某种推断算法采样新的数据或计算数据的对数似然。这本质上对应了深度生成模型的基本问题：表达能力、训练难度、推断速度与准度。根据采样过程对应映射的可逆性，深度生成模型可以被分为不可逆生成模型和可逆生成模型。其中，不可逆生成模型训练难度较大，且通常难以进行准确的似然推断；反之，可逆生成模型可以计算准确的似然，且表达能力和训练稳定性都较好，因而在众多复杂高维数据建模任务中展现了优异的性能。

然而，不同种类的可逆生成模型的表达能力、训练难度和推断速度都存在瓶颈，可逆生成模型仍存在一些亟待解决的关键基本问题。第一，模型表达能力与推断速度之间存在取舍，推断速度较快的离散时间标准化流模型的表达能力受限。第二，虽然连续时间标准化流模型的表达能力足够强，但是其最大似然训练的开销较大，难以扩展至大规模数据。第三，扩散模型拥有较强的表达能力和较稳定的训练过程，但其采样需要数百至数千次大型神经网络的串行函数调用，因而采样过程较为低效。为解决上述关键问题，本文提出更强表达能力的可逆生成模型和高效的训练与推断算法。本文的主要创新点如下：

- 针对离散时间标准化流模型表达能力受限的问题，提出了基于隐函数定义的隐式标准化流模型。该模型具有利普希兹常数不受限的特性，其函数族严格包含已有模型的函数族，因而表达能力严格强于同等规模的已有模型。
- 针对连续时间标准化流模型的最大似然训练开销较大的问题，提出了一种误差可控的高阶去噪分数匹配算法，避免了连续时间标准化流模型所依赖的常微分方程求解器的昂贵开销，使得其最大似然训练可以被高效地实现。
- 针对无条件扩散模型采样速度慢的问题，提出了一种快速、专用、无需训练的高阶扩散常微分方程求解器，显著提升了无条件扩散模型的采样速度。
- 针对条件扩散模型的采样速度慢及少步采样不稳定的问题，提出了一种针对引导采样的快速、无需训练的采样算法。该算法同时适用于扩散常微分方程和扩散随机微分方程，显著提升了条件扩散模型的采样速度和稳定性。

关键词：可逆生成模型；标准化流模型；扩散模型；最大似然训练；快速采样

Abstract

With the proliferation of digital technology, the diverse and complex data generated by various devices have exploded in growth, and they often lack high-quality annotations. In recent years, deep generative models based on deep learning technologies have made a series of advancements in understanding and modeling large-scale unlabelled data. Given training data, deep generative models aim to fit the data distribution through certain training algorithms and sample new data or calculate the logarithmic likelihood of the data through certain inference algorithms. This essentially corresponds to the fundamental problems of deep generative models: expressive power, training difficulty, inference speed, and inference accuracy. Based on the invertibility of the sampling process mapping, deep generative models can be divided into non-invertible generative models and invertible generative models. The former is more challenging to train and usually struggles to make accurate likelihood inferences. In contrast, the latter can compute accurate likelihoods and have better expressive power and training stability, demonstrating superior performance in many complex high-dimensional data modeling tasks.

However, various types of invertible generative models have bottlenecks in their expressive power, training difficulty, and inference speed. There are some critical fundamental problems still waiting to be solved with invertible generative models. First, there is a trade-off between model expressive power and inference speed. Discrete-time normalizing flows with faster inference speeds have limited expressive power. Second, although continuous-time normalizing flows have sufficient expressive power, their maximum likelihood training is costly and hard to scale to large datasets. Third, diffusion models possess expressive power and stable training processes, but their sampling requires hundreds to thousands of sequential function evaluations of large neural networks, making the sampling process inefficient. To address these key issues, this dissertation proposes an invertible generative model with stronger expressive power and efficient training and inference algorithms. The main contributions are summarized as follows:

- To address the limited expressive power of the discrete-time normalizing flows, implicit normalizing flow is proposed, which is defined by an implicit function. This model has an unrestricted Lipschitz constant, and its function family strictly includes that of the original model, making its expressive power strictly stronger

than the original model of the same scale.

- To address the high overhead of maximum likelihood training for continuous-time normalizing flows, an error-bounded high-order denoising score matching algorithm is proposed, which avoids the expensive overhead of the ordinary differential equation solvers that continuous-time normalizing flows rely on, allowing for efficient maximum likelihood training.
- To address the slow sampling speed of unconditional diffusion models, a fast, dedicated, and training-free high-order ordinary differential equation solver is proposed, which significantly improves the sampling speed of unconditional diffusion models.
- To address the slow sampling speed and the instability issue of conditional diffusion models, a fast, untrained sampling algorithm for guided sampling is proposed. This algorithm is applicable to both diffusion ordinary differential equations and diffusion stochastic differential equations, which significantly enhances the sampling speed and the stability of conditional diffusion models.

Keywords: invertible generative models; normalizing flows; diffusion models; maximum likelihood training; fast sampling

目 录

插图清单

附表清单

符号和缩略语说明

ResFlow	残差标准化流模型（residual normalizing flows）
ImpFlow	隐式标准化流模型（implicit normalizing flows）
SM	分数匹配（score matching）
DSM	去噪分数匹配（denoising score matching）
SNR	信噪比（signal-to-noise ratio）
DPM	扩散概率模型（diffusion probabilistic models）
ODE	常微分方程（ordinary differential equation）
Diffusion ODE	扩散常微分方程（diffusion ordinary differential equation）
SDE	随机微分方程（stochastic differential equation）
Diffusion SDE	扩散随机微分方程（diffusion stochastic differential equation）
GAN	生成对抗网络（generative adversarial networks）
VAE	变分自编码器（variational auto-encoders）
DDIM	去噪扩散隐式模型（denoising diffusion implicit models）
KL 散度	库尔贝克-莱布勒散度（Kullback-Leibler divergence）
Fisher 散度	费舍尔散度（Fisher divergence）
Hessian 矩阵	海森矩阵（Hessian matrix）
Frobenius 范数	弗罗贝尼乌斯范数（Frobenius norm）
MLP	多层感知机（multilayer perceptron）
$\boldsymbol{\epsilon}$	标准高斯噪声
$\mathcal{N}(\boldsymbol{x}\|\mu,\sigma^2\boldsymbol{I})$	以 μ 为均值，σ 为标准差的高斯分布在 x 处的概率密度
$\mathcal{N}(\mathbf{0},\boldsymbol{I})$	标准高斯分布
$\boldsymbol{f}(\cdot),\boldsymbol{g}(\cdot),\boldsymbol{h}(\cdot)$	向量值函数（vector-valued function）
$\boldsymbol{F}(\cdot)$	向量值函数
$\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}$	向量随机变量
$p(\cdot)$	概率密度函数
$p(\cdot\,;\theta)$	由 θ 参数化的概率分布 $p(\cdot)$ 的概率密度函数
$\mathbb{P}(\cdot)$	概率值
$\mathbb{D}_{\mathrm{KL}}(q\,\|\,p)$	分布 q 和 p 之间的 KL 散度
$\mathbb{D}_{\mathrm{F}}(q\,\|\,p)$	分布 q 和 p 之间的 Fisher 散度
α_t	扩散模型前向过程中均值的系数（$p(\boldsymbol{x}_t\|\boldsymbol{x}_0)$ 为 $\mathcal{N}(\boldsymbol{x}_t\|\alpha_t\boldsymbol{x}_0,\sigma_t^2\boldsymbol{I})$）
σ_t	扩散模型前向过程中的标准差（$p(\boldsymbol{x}_t\|\boldsymbol{x}_0)$ 为 $\mathcal{N}(\boldsymbol{x}_t\|\alpha_t\boldsymbol{x}_0,\sigma_t^2\boldsymbol{I})$）

λ_t	扩散模型在时间 t 的对数信噪比（log-SNR）的一半（定义为 $\lambda_t \coloneqq \log \alpha_t - \log \sigma_t$）
t_λ	λ_t 的反函数
$f(t), g(t)$	扩散模型的前向扩散过程中的系数
$\boldsymbol{x}_t$	扩散常微分方程或扩散随机微分方程在时间 t 的精确解
$\tilde{\boldsymbol{x}}_t$	扩散常微分方程或扩散随机微分方程在时间 t 的近似解
$\hat{\boldsymbol{x}}_\lambda$	扩散常微分方程或扩散随机微分方程在 $t = t_\lambda(\lambda)$ 的精确解
$\boldsymbol{w}_t$	维纳过程（Wiener process）在时间 t 的随机变量
$\bar{\boldsymbol{w}}_t$	反向（reverse-time）维纳过程在时间 t 的随机变量
$\hat{\boldsymbol{w}}_\lambda$	维纳过程在时间 $t = t_\lambda(\lambda)$ 的随机变量
$\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$	概率分布 $q_t(\boldsymbol{x}_t)$ 的分数函数
$\boldsymbol{s}_\theta$	由参数 θ 定义的分数模型（score model）
$\boldsymbol{\epsilon}_\theta$	由参数 θ 定义的噪声预测模型（noise-prediction model）
$\hat{\boldsymbol{\epsilon}}_\theta$	噪声预测模型 $\boldsymbol{\epsilon}_\theta$ 关于 λ 换元后的函数
$\boldsymbol{x}_\theta$	由参数 θ 定义的数据预测模型（data-prediction model）
$\hat{\boldsymbol{x}}_\theta$	数据预测模型 $\boldsymbol{x}_\theta$ 关于 λ 换元后的函数
$\mathcal{J}_{\mathrm{SM}}$	加权分数匹配目标函数
$\mathcal{J}_{\mathrm{DSM}}$	加权去噪分数匹配目标函数
$\mathcal{O}$	大 O 符号（big O notation，描述渐进上界）
$\nabla(\cdot)$	求导（梯度）算子
$\nabla \cdot (\cdot)$	散度算子
$\nabla^2(\cdot)$	二阶求导算子
$\boldsymbol{f}^{-1}$	函数 $\boldsymbol{f}$ 的逆函数（inverse function）
$\boldsymbol{J}_{\boldsymbol{f}}(\cdot)$	$\boldsymbol{f}$ 的雅克比矩阵（Jacobian matrix）
$\mathrm{Lip}(\cdot)$	函数的利普希兹常数（Lipschitz constant）
$\|\cdot\|_1$	向量或矩阵的 L_1 范数（L_1 norm）
$\|\cdot\|_2$	向量或矩阵的 L_2 范数（L_2 norm）
$\|\cdot\|_F$	矩阵的弗罗贝尼乌斯范数（Frobenius norm）
$\circ$	函数的复合（composition）
$\odot$	向量的逐元素相乘（element-wise multiplication）
$\boldsymbol{A} \subsetneq \boldsymbol{B}$	$\boldsymbol{A}$ 是 $\boldsymbol{B}$ 的真子集
$\det(\cdot)$	矩阵的行列式（determinant）
$\sigma(\boldsymbol{J})$	矩阵 $\boldsymbol{J}$ 的奇异值（singular value）
Id	恒等映射（identity mapping）
$\mathbb{R}$	实数空间

$\mathbb{R}^+$	正实数空间
$\mathbb{R}^d$	d 维的实数空间
$C^1(\mathbb{R}^d, \mathbb{R}^d)$	所有 $\mathbb{R}^d$ 到 $\mathbb{R}^d$ 的一阶连续可微函数组成的集合
$\mathcal{R}$	单层残差标准化流模型的函数族
$\mathcal{R}_\ell$	ℓ 层残差标准化流模型的函数族
$\mathcal{I}$	单层隐式标准化流模型的函数族
$\mathcal{D}$	所有 $\mathbb{R}^d$ 到 $\mathbb{R}^d$ 的一阶连续可微的双向利普希兹连续（bi-Lipschitz continuous）的可逆函数组成的集合
$\mathcal{F}$	$\mathcal{D}$ 中的单调递增函数组成的集合
$\mathcal{B}_r$	d 维的半径为 r 的球的内部的点组成的空间

第 1 章　绪论

1.1　研究背景及意义

在当今时代，数字技术的广泛传播引发了数据量的爆炸式增长。从智能手机到可穿戴设备的每次交互，都为这股数据洪流增添了一份力量。而社交媒体平台更是源源不断地产生由用户发布的文本、图片和视频等新数据。同样，物联网的诞生也让众多传感器和智能设备不断从家庭、工厂或城市等多种环境中产生大量数据。此外，零售、金融及其他领域的交易数据也为这个日益增长的数字信息库添砖加瓦。这种随处可见的数据生成和获取，常被称为“大数据”，其中蕴含着丰富的信息等待被挖掘。

“大数据”不仅指代大量的数据，还包括了它的复杂性和多样性。如何在合理的时间内有效地管理、处理和理解这些数据，成为了信息时代最重要的难题。近年来，深度学习（deep learning）[1-3]利用大规模的深度神经网络（deep neural networks）[4-5]将复杂数据的信息进行充分压缩，并利用随机梯度下降（stochastic gradient descent）[6-7]对神经网络的参数进行优化，从而实现了数据的高效建模与理解。根据数据类别的不同，深度学习又可以被分为判别式建模（discriminative modeling）[2]和生成式建模（generative modeling）[8]。判别式建模依赖于带有额外标注的数据（例如图像的类别[9]、物体的包围盒（bounding box）[10]等）进行训练，目的是预测未知数据对应的标注，且在图像分类[2]、物体检测与分割[11-12]、语音识别[13-14]等领域取得了超过人类平均水平的性能。然而，为数据进行额外标注通常是一件昂贵且耗时的事情，数据标注的速度远远慢于现代社会数据产生的速度，因此判别式建模很难充分利用互联网上大规模的数据。相反，生成式建模无需额外的数据标注，而是直接建模数据对应的分布，且可以充分利用互联网上的所有数据，例如在互联网文本数据上大规模训练的生成模型[3]可以在无需额外标注的前提下对人类提问进行回答。基于生成式建模的范式和神经网络强大的表达能力，深度生成模型（deep generative models）在高分辨率图像生成[15]、语音合成[16]、文本对话系统[17]等领域都取得了瞩目的进展，且可以完成如绘画[18]、写诗[19]等创造性任务。因此，深度生成模型被认为是迈向通用人工智能的重大进展[20-21]。

具体而言，给定空间 $\mathbb{R}^d$ 上的 d 维数据 $\boldsymbol{x}$ 的分布 $p(\boldsymbol{x})$，深度生成模型旨在学习（learning）由参数空间 $\boldsymbol{\Theta}$ 中的参数 θ 定义的模型分布（model distribution）$p(\boldsymbol{x};\theta)$，使得该模型分布近似于数据分布 $p(\boldsymbol{x})$，并可以从 $p(\boldsymbol{x};\theta)$ 中推断（inference）以估计数据的密度函数和生成新的数据。为了达到这一目标，不同的深度生成模型 $p(\boldsymbol{x};\theta)$

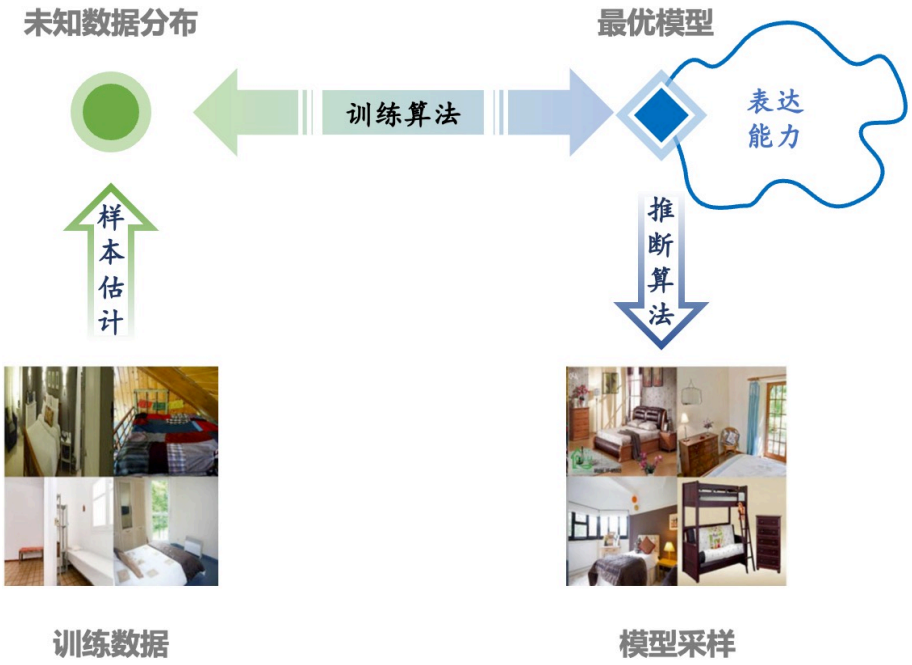


图 1.1　深度生成模型的基本问题

本质上都对应了以下三点基本问题：

1. 表达能力：即如何定义参数空间 Θ 及模型分布 $p(\boldsymbol{x};\theta)$，使得模型分布的函数族尽可能包含数据分布。一般而言，参数空间 Θ 由神经网络结构决定，不同的神经网络（如多层感知机[1]、卷积神经网络[22]、自注意力模型[5]等）对应了不同的参数空间 Θ。而给定 $\theta \in \Theta$ 后，模型分布 $p(\boldsymbol{x};\theta)$ 直接由深度生成模型的类型决定。例如，显式（explicit）深度生成模型直接定义了 $p(\boldsymbol{x};\theta)$，包括自回归模型（autoregressive models）[23-25]、变分自编码器（variational auto-encoders）[26-27]、标准化流模型①（normalizing flows）（包括离散时间（discrete-time）[29-35]和连续时间（continuous-time）[36-38]）、能量模型（energy-based models）[39-42]和扩散模型（diffusion models）[43-45]；隐式（implicit）深度生成模型没有显式的 $p(\boldsymbol{x};\theta)$ 的表达式，而是定义了从 $p(\boldsymbol{x};\theta)$ 中的采样过程，例如生成对抗网络（generative adversarial networks）[8,46-47]。不同类型深度生成模型的表达能力往往不同。总体而言，深度生成模型需要尽可能地提高模型的表达能力，以匹配复杂的数据分布。

2. 训练算法（训练难度）：即如何选择合适的目标函数 L 与算法，使得对于给定的数据集 $\boldsymbol{D}$，算法可以求解出最优的参数 $\hat{\theta}$：

$$\hat{\theta} = \arg\min_{\theta\in\Theta} L(\boldsymbol{D};\theta). \tag{1.1}$$

一般而言，L 会取某种分布之间的度量，用以匹配数据分布与模型分布，例如库尔

① “流”是指一系列简单映射的叠加[28]，“标准化”是指模型旨在将数据经过一系列简单映射后得到的分布服从标准高斯分布[28]。一些工作也将该类模型称为“基于流的生成模型”（flow-based generative models[29]）。

贝克-莱布勒散度（Kullback-Leibler divergence，KL 散度）。由于不同类型的深度生成模型的表达能力不同，其训练难度（寻找到最优解的难度）也不同。例如，生成对抗网络需要引入额外的判别器（discriminator）借助对抗训练（adversarial training）来监督生成器（generator），但其训练很不稳定；变分自编码器需要引入额外的变分后验（variational posterior）来最大化数据对数似然的下界；能量模型需要利用分数匹配（score matching）来估计数据分布的分数函数（score function），但其在低密度区域的训练难度很大[48]；自回归模型由于缺乏对高维度数据的冗余信息的先验，其在图像数据的训练难度通常远远大于其它生成模型[25]；连续时间标准化流模型需要在训练的每一步优化都借助常微分方程（ordinary differential equation）求解器（solver），因此训练较慢；而离散时间标准化流模型可以高效地计算准确的对数似然，因而可以简单地利用最大似然估计（maximum likelihood estimation）准则进行训练，其训练难度通常较低；扩散模型只需要预测加噪数据的噪声，其训练目标函数是一个非常简单且稳定的均方误差，因此容易扩展到高维数据[44]。总体而言，深度生成模型需要尽可能地设计稳定且易于扩展的训练算法，使得模型可以高效地发挥其理论上的表达能力。

3. 推断算法（推断速度与准度）：即如何准确地计算给定数据 $\boldsymbol{x}$ 的对数似然 $\log p_\theta(\boldsymbol{x})$ 以及如何从模型分布 $p_\theta(\boldsymbol{x})$ 中采样得到新的数据。一般而言，由于不同类型的深度生成模型的似然计算与采样过程不同，其推断的难度也有所不同。例如，生成对抗网络、变分自编码器和离散时间标准化流模型都只需要首先采样隐变量（latent variable），接着将其作为生成器的输入即可得到数据采样，因而采样速度很快；但自回归模型、能量模型、扩散模型和连续时间标准化流模型都需要成百上千步串行迭代来进行采样，因此推断速度很慢。此外，由于生成对抗网络和能量模型无法计算数据的对数似然，而变分自编码器只能计算数据的对数似然的一个下界，因而推断准度较低；但自回归模型、标准化流模型和扩散模型都可以准确计算数据的对数似然。总体而言，深度生成模型需要尽可能地设计准确、快速的推断算法，使得模型可以高效地应用于下游任务。

综上所述，已有深度生成模型的表达能力、训练难度、推断速度与准度之间往往存在取舍，针对深度生成模型研究的核心目标为：稳定训练出表达能力强的模型，且可以准确、快速推断，以用于诸多下游任务（例如数据的合成、编辑、增广、压缩等）。具体而言，根据深度生成模型中隐变量和数据变量之间映射的种类，深度生成模型可以被分为可逆生成模型（invertible generative models）和不可逆生成模型（non-invertible generative models）。表 1.1 给出了不同类型的深度生成模型的表达能力、训练难度、推断速度与准度的研究现状比较，其中 ✓ 代表表达能力

表 1.1　不同深度生成模型的表达能力、训练难度、推断速度与推断准度的研究现状

		表达能力	训练难度	推断速度	推断准度
不可逆生成模型	生成对抗网络	✓	○	✓	○
	变分自编码器	○	○	✓	○
	能量模型	✓	○	○	○
	自回归模型	✓	○	○	✓
可逆生成模型（第 2 章）	离散时间的标准化流模型	○（第 3 章）	✓	✓	✓
	连续时间的标准化流模型	✓	○（第 4 章）	○（第 5、6 章）	✓
	扩散模型	✓	✓	○（第 5、6 章）	✓

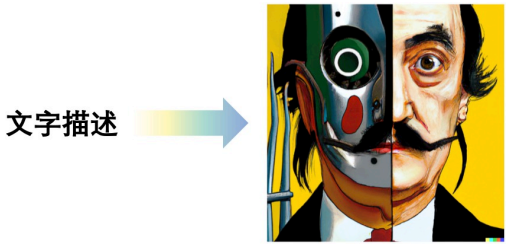
强 / 训练难度低 / 推断速度快 / 可精确推断似然；而 ○ 代表表达能力弱 / 训练难度高 / 推断速度慢 / 不可精确推断似然。可以发现，不可逆生成模型的训练难度都较大，且往往无法进行准确的推断；然而，可逆生成模型的表达能力和训练稳定性都较好，且都可以进行准确的推断，因此可逆生成模型在众多复杂高维数据的建模任务中都有非常不错的表现（详见第 1.1.1 节）。

具体而言，可逆生成模型通过学习由神经网络参数化的可逆映射，将复杂数据分布映射到简单分布（例如标准高斯分布）。由于映射的可逆性，也可以先从简单分布里采样，接着经过逆映射得到数据分布里的采样。可逆生成模型包括标准化流模型和扩散概率模型[43-45]，其中标准化流模型又可以被分为离散时间标准化流模型[29-35]和连续时间标准化流模型[36-37,45]。可逆生成模型在许多任务上的表现都很出色，例如图像生成[49-50]、视频生成[51]、文本到图像生成[18]、语音合成[52-53]和无损压缩[54]。

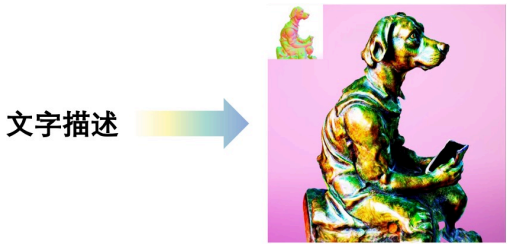
综上所述，可逆生成模型是一类具有前景的深度生成模型，其在建模大规模、复杂、多样的数据分布上具有非常大的优势。

1.1.1　研究价值

可逆生成模型在复杂高维分布的建模任务中具有显著的优势，并且由于其可逆的特性，该模型在高质量、高可控的数据生成中也具有较强的表现。图 1.2 展示

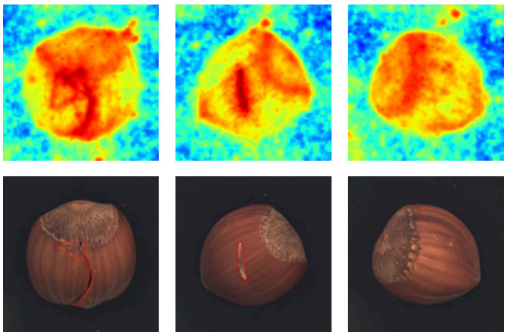


(a) 文本到图像生成[18]

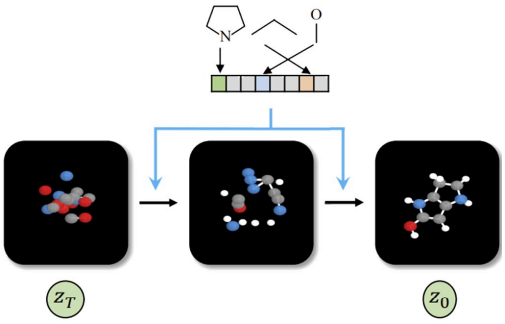


(b) 文本到三维内容生成[55]

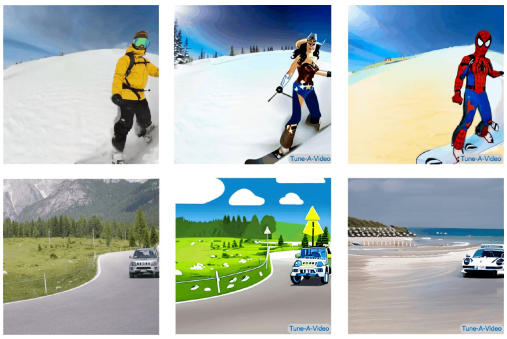
(c) 个性化图像生成[56]



(d) 异常检测[57]



(e) 可控分子生成[58]



(f) 可控视频编辑[59]

图 1.2　可逆深度生成模型取得的应用成果示例

了一些可逆生成模型目前取得的应用成果示例。例如，由于可逆生成模型可以准确地计算数据的似然，该模型在异常检测[57]、不确定性建模[60]等任务中都可以借助准确的似然评估来完成数据的不确定性度量与分析。此外，由于可逆生成模型中的隐变量和数据变量之间的映射是可逆的，通过对给定数据对应的隐变量进行编辑，可逆生成模型可以容易地完成复杂的编辑任务和可控生成，例如可控图像生成[61]、可控视频编辑[59]和可控分子生成[58]。此外，作为可逆生成模型的一种，扩散模型在多模态数据上展现了较强的性能，其在文本到图像生成[18]、个性化图像生成[56]和高质量的文本到三维内容生成[55]任务中都有较高的生成质量。

综上所述，可逆生成模型在多种复杂高维数据的建模与生成任务中具有极强的应用价值，且具有高度的可控性。因此，可逆生成模型的应用前景十分广阔。

1.2　研究现状及难点

尽管可逆生成模型在许多生成任务中取得了成功，其对应的深度生成模型基本问题并未完全解决。表 1.1 总结了目前可逆生成模型的研究现状，其中可逆生成模型的表达能力、训练难度和推断速度都存在瓶颈。本节针对当前可逆生成模型研究现状及难点进行详细讨论。

表达能力。　尽管可逆生成模型可以进行准确的推断，但可逆性通常会带来模型表达能力与推断速度的取舍。尽管离散时间标准化流模型的推断速度较快，但其表达能力较为受限。具体而言，为了保证映射的可逆性，离散时间标准化流模型的每一层映射一般都需要有特殊的结构限制。一个典型的限制是令映射的雅克比矩阵的结构为特殊的对角结构[30-32]或三角结构[62]，以保证其对数行列式可以高效地计算，但这样会限制模型的表达能力。为了解决这一问题，已有的许多工作都致力于提升标准化流模型的表达能力。一种方式是在保证可逆的前提下尽量减少雅克比矩阵的限制，例如设计雅克比矩阵易于计算的可逆映射及其专属的神经网络架构[29,35,63-65]；设计具有结构不受限的雅克比矩阵的可逆映射[33-34,37]等；另一种方式是打破可逆映射的维度限制，例如在更高维的空间中进行数据变换[66-70]。然而，这些方法得到的离散时间标准化流模型的表达能力仍然受限。

训练算法。　对于标准化流模型而言，由于连续时间标准化流模型由常微分方程定义，尽管连续时间标准化流模型的表达能力优于离散时间标准化流模型，所有前人工作[36-37]在其最大似然训练时需要在训练的每一步迭代利用常微分方程求解器来计算给定数据点的对数似然。然而，由于常微分方程求解器的求解通常需要成百上千次神经网络的串行调用，这导致每一步训练的前向与反向计算都是离散时间标准化流模型训练开销的成百上千倍，故连续时间标准化流模型的最大似然训练开销较大。例如，前人工作[45]中基于大型神经网络定义的连续时间标准化流模型在评估单个数据点的对数似然需要花费 2 到 3 分钟，这导致连续时间标准化流模型的训练难以扩展至大规模数据集。此外，Finlay 等人[71]发现，通过简单地最大似然训练连续时间标准化流模型还可能会导致高度不光滑的常微分方程的轨迹（trajectory），这种不光滑的轨迹也可能使得后续的采样和似然计算的时间成本进一步增加。因此，如何高效稳定地基于最大似然估计准则来训练连续时间标准化流模型，仍然是一个未知的问题。

推断算法。　扩散模型拥有较强的表达能力、较小的训练难度和准确的似然推断，因此扩散模型在大规模多模态数据建模中取得了显著的成功[15,18,72-74]。然而，扩

散模型采样一个样本通常需要数百至数千次大型神经网络的串行函数调用（采样步数），这比其它单步生成模型例如生成对抗网络或变分自编码器的采样速度慢得多。这样低效的采样速度成为了扩散模型在下游任务应用的关键瓶颈。因此，如何为扩散模型设计快速采样算法是该领域的核心问题之一。具体而言，现有的扩散模型快速采样算法可被分为两类。第一类包括知识蒸馏[75-76]和噪声水平或样本轨迹学习[77-80]。该类方法带来了不可忽略的甚至较为昂贵的额外训练开销，并且适用性和灵活性都较为受限。例如，使用者需要做大量的额外调整以适应不同的模型、数据集和采样步数。因此，该类方法难以大规模应用至下游任务。第二类为无需训练的采样器[81-83]，这些采样器简单且即插即用，适用于所有预训练的扩散模型。具体而言，无需训练的采样器包括采用隐式[81]或解析方差[83]的生成过程、更先进的微分方程求解器[45,82,84-86]以及对采样时间点的动态规划[80]。然而，这些方法中最快的算法仍然需要大约 50 次采样步数[83]来生成高质量样本（指与普通采样器通过约 1000 次函数调用得到的生成质量相当），因此仍然耗时较久。综上所述，如何对扩散模型进行高效采样，仍然是该领域一个亟待解决的问题。

1.3　研究内容与主要贡献

针对上述研究难点，本文对离散时间标准化流模型表达能力受限的问题、连续时间标准化流模型最大似然训练困难的问题和扩散模型采样速度缓慢的问题进行系统地分析，并提出更强表达能力的离散时间标准化流模型、针对连续时间标准化流模型的高效稳定的训练算法和针对扩散模型的更高效快速的推断算法，以改进可逆生成模型的性能。本文的研究内容和贡献可以总结为四个部分：

第一部分旨在提高离散时间标准化流模型的表达能力，提出了基于隐函数定义的隐式标准化流模型。该部分主要的创新点为：由非线性方程的根隐式地定义模型的可逆映射，从而推广了原有的标准化流模型。隐式标准化流模型基于残差标准化流模型，在可计算性和表达能力之间达到了良好的平衡。理论分析表明，隐式标准化流模型的函数空间具有利普希兹常数（Lipschitz constant）不受限的特性，其函数族比残差标准化流模型的函数族更丰富，特别是在建模具有较大的利普希兹常数的函数方面，因此表达能力严格强于已有模型。此外，基于隐式微分公式，该部分的研究提出了一种可扩展的算法来训练和推断隐式标准化流模型。实验结果表明，隐式标准化流模型在多种密度估计任务中的性能都优于基准线方法。该部分的研究成果发表在 ICLR 2021[87]上，并被评选为亮点论文（Spotlight，接收率约 5.5%）。

第二部分旨在实现扩散常微分方程的高效最大似然训练，提出了一种误差可

控的高阶去噪分数匹配算法。该部分的主要创新点是：通过分析分数匹配目标与数据分布到扩散常微分方程分布的 KL 散度之间的关系，提出可以通过最小化分数模型的一阶、二阶和三阶分数匹配误差来控制 KL 散度的上界。为最小化高阶分数匹配的误差，该部分的研究进一步提出了一种误差可控的高阶去噪分数匹配算法，使得高阶分数匹配误差可以由训练误差和低阶分数匹配误差共同限制，且分数模型的全局最优解仍与扩散模型的原始训练目标相同。所提高阶去噪分数匹配算法避免了连续时间标准化流模型所依赖的常微分方程求解器的昂贵开销，使得其最大似然训练可以被高效地实现。实验证明，该部分所提高阶去噪分数匹配算法可以极大地提高多个不同密度建模基准任务上扩散常微分方程的模型密度，使扩散常微分方程在模拟数据和真实数据都获得最优的似然估计结果。该部分的研究成果发表在 ICML 2022[88] 上。

第三部分旨在解决扩散模型采样速度慢的问题，提出了 DPM-Solver，一种快速、专用、无需训练的扩散常微分方程求解器，只需要大约 10 步左右的函数调用就可以对扩散模型进行快速采样。该部分的主要创新点是：利用了扩散常微分方程的半线性结构，并直接近似扩散常微分方程的精确解的简化表达式，该表达式由噪声预测模型的指数加权积分组成。受指数积分器的数值方法的启发，该部分的研究提出了一阶、二阶和三阶 DPM-Solver 来近似噪声预测模型的指数加权积分，并具有理论上的收敛保证。DPM-Solver 可以应用于连续时间和离散时间的扩散模型。多个数据集上的实验结果表明，DPM-Solver 可以只用 10 到 20 次函数调用就可以生成高质量的样本，并且与以前的最先进的无需训练的采样器相比可以实现 4 到 16 倍的加速。该部分的研究成果发表在 NeurIPS 2022[89] 上，并被评选为口头报告论文（Oral，接收率约 1.7%）。

第四部分旨在解决条件扩散模型的采样速度慢及少步采样不稳定的问题，提出了 *DPM-Solver*++，一种用于条件扩散模型的高效采样算法，能够同时适用于加速扩散常微分方程和扩散随机微分方程的引导采样。DPM-Solver++ 基于数据预测模型对应的参数化形式来求解扩散常微分方程，并可以直接采用多步法和阈值法进一步稳定采样过程。实验结果显示，DPM-Solver++ 只需 15 到 20 步就可以生成高质量的样本，极大地提升了条件扩散模型的采样速度和稳定性。该部分的研究成果被主流的文到图生成模型 Stable Diffusion[15] 应用为官方默认采样算法，在开源社区产生了广泛的影响。

1.4　本文组织结构

本文的章节组织结构见图 1.3，共有 7 个章节。

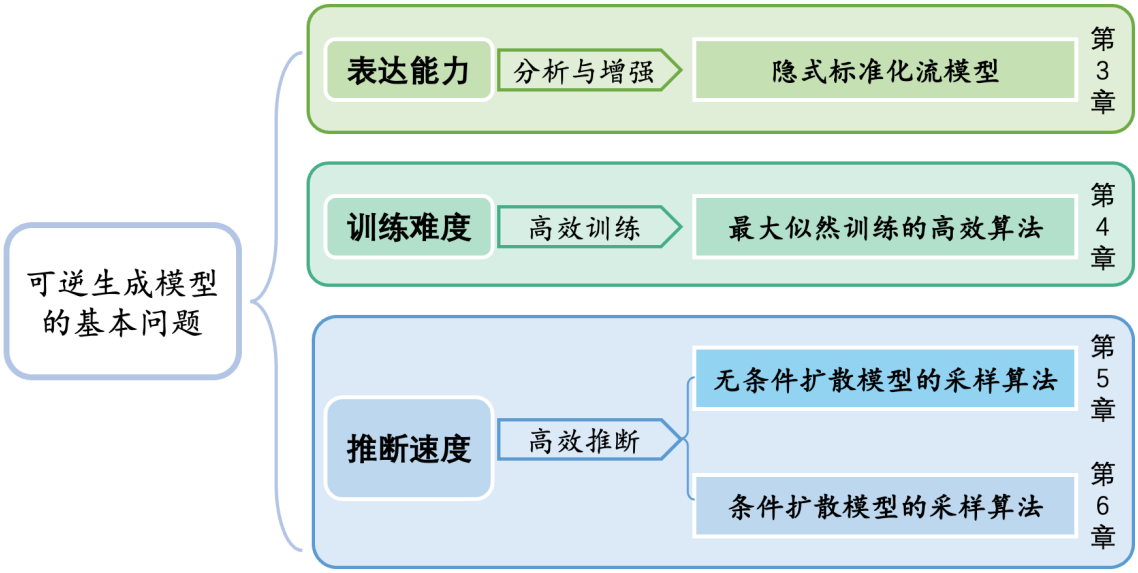


图 1.3　本文的章节组织结构

第 1 章介绍了本文的研究背景及意义，并概括了本文的主要研究内容和贡献。

第 2 章介绍了本文研究工作的背景知识，主要包括离散时间标准化流模型、连续时间标准化流模型和扩散模型的形式化介绍，其中扩散模型包括扩散随机微分方程和扩散常微分方程，以及前人工作中的训练和推断算法。

第 3 章提出了基于隐函数定义的隐式标准化流模型，并从理论上分析了该模型函数族空间与已有模型函数族空间的关系，证明了在同等规模下所提模型的表达能力严格强于已有模型，并且具有利普希兹常数不受限的特性，在多种数据集的密度估计任务中都显著优于已有模型。

第 4 章提出了一种误差可控的高阶去噪分数匹配算法，使得高阶分数匹配误差可以由训练误差和低阶分数匹配误差共同限制，且分数模型的全局最优解仍与扩散模型的原始训练目标相同，使得连续时间的标准化流模型以及扩散常微分方程的最大似然训练可以被高效地实现。

第 5 章提出了快速、专用、无需训练的扩散常微分方程求解器，该方法只需要大约 10 步左右的函数调用就可以对无条件扩散模型进行快速采样，与已有最先进的无需训练采样器相比可以实现 4 到 16 倍的加速，显著提升了无条件扩散模型的采样速度。

第 6 章提出了用于条件扩散模型的高效采样算法，同时适用于加速扩散常微分方程和扩散随机微分方程的引导采样。该方法只需 15 到 20 步就可以得到几乎收敛的采样结果，显著提升了条件扩散模型的采样速度和少步采样的稳定性。

第 7 章总结了本文的所有研究内容，并对未来的研究方向进行了展望。

第 2 章　背景知识

在介绍具体的研究内容之前，本章首先对可逆生成模型涉及的背景知识给出详细的形式化说明，包括标准化流模型（normalizing flows）以及扩散模型（diffusion models）。图 2.1 阐述了标准化流模型和扩散模型的基本原理以及对应的可逆映射。

2.1　标准化流模型

标准化流模型是一类由可逆映射（invertible mapping）定义的深度生成模型。由于映射的可逆性，其概率密度函数可以通过变量替换公式（change-of-variable formula）显式地和准确地计算。根据可逆映射的种类不同，标准化流模型可以被分为离散时间标准化流模型（discrete-time normalizing flows）和连续时间标准化流模型（continuous-time normalizing flows）。本节给出这两种模型的详细定义以及常见的训练方式。

本节首先给出标准化流模型的一般形式。具体而言，令 $\boldsymbol{x} \in \mathbb{R}^d$ 为 d 维实数空间中服从未知数据分布 $q(\boldsymbol{x})$ 的随机变量，$\boldsymbol{z} \in \mathbb{R}^d$ 为 d 维实数空间中服从某个简单分布 $p_{\boldsymbol{z}}(\boldsymbol{z})$（例如标准高斯分布或均匀分布）的随机变量。标准化流模型通过参数 θ 定义了一个可逆映射 $\boldsymbol{f} : \mathbb{R}^d \to \mathbb{R}^d$，来将数据 $\boldsymbol{x}$ 转换为某个对应的隐变量（latent variable）$\boldsymbol{z}$：

$$\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x}; \theta). \tag{2.1}$$

由于 $\boldsymbol{f}$ 的可逆性，标准化流模型进一步基于变量替换公式（change-of-variable formula）定义模型分布 $p(\boldsymbol{x}; \theta)$:

$$\log p(\boldsymbol{x}; \theta) = \log p_{\boldsymbol{z}}(\boldsymbol{z}) + \log \left| \det(J_{\boldsymbol{f}}(\boldsymbol{x})) \right|, \tag{2.2}$$

其中 $J_{\boldsymbol{f}}(\boldsymbol{x})$ 是 $\boldsymbol{f}$ 在 $\boldsymbol{x}$ 处的雅克比矩阵（Jacobian matrix），$\det(J_{\boldsymbol{f}}(\boldsymbol{x}))$ 为该雅克比矩阵的行列式（determinant）。基于式（2.2），标准化流模型的参数 θ 可以通过最大似然估计（maximum likelihood estimation）的方式学习，即：

$$\max_{\theta} \mathbb{E}_{q(\boldsymbol{x})}[\log p(\boldsymbol{x}; \theta)], \tag{2.3}$$

其中对 $q(\boldsymbol{x})$ 的期望可以用训练数据集的蒙特卡洛采样（Monte Carlo sampling）来估计。由于标准化流模型的对数似然 $\log p(\boldsymbol{x}; \theta)$ 可以由式（2.2）准确地计算，因而标准化流模型的训练较为简单且稳定。

此外，由于 $\boldsymbol{f}$ 的可逆性，标准化流模型的采样（即随机采样 $\boldsymbol{x} \sim p(\boldsymbol{x}; \theta)$）也非

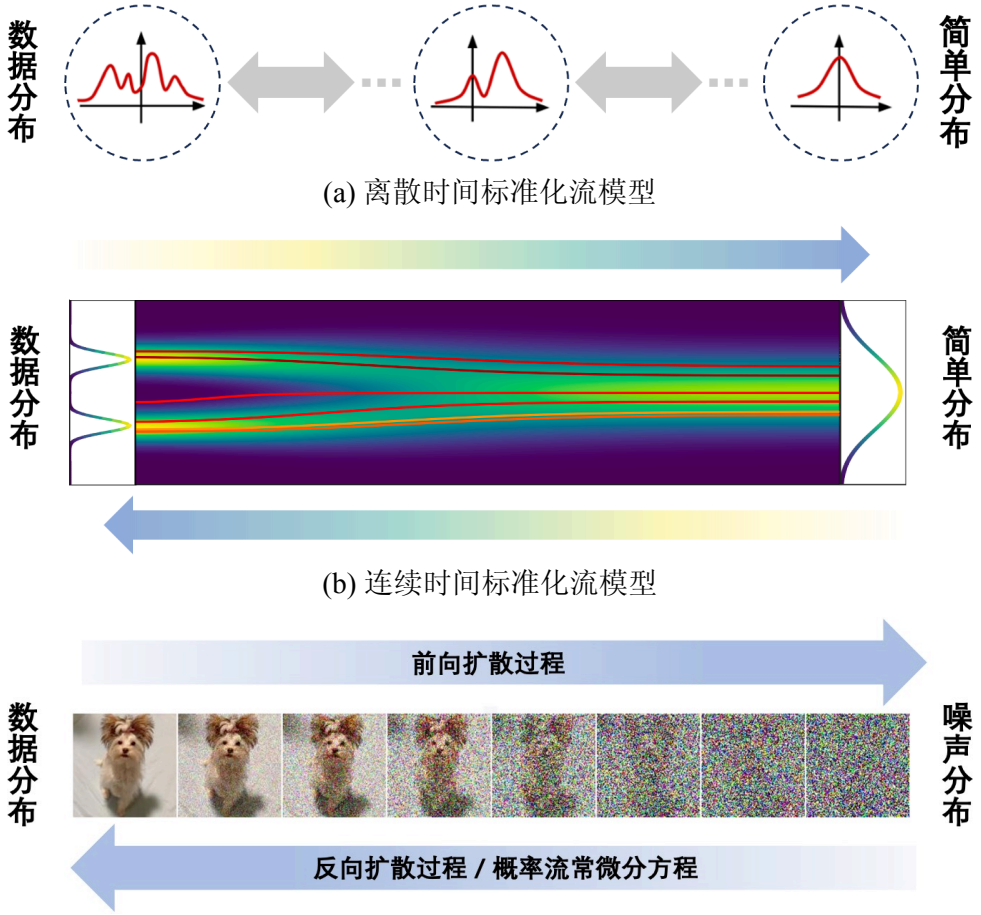


图 2.1　可逆深度生成模型（标准化流模型和扩散模型）的示意图

常容易，只需要首先采样 $\boldsymbol{z} \sim p_z(\boldsymbol{z})$，再对 $\boldsymbol{z}$ 取逆变换即可获得 $p(\boldsymbol{x};\theta)$ 的采样：

$$\boldsymbol{x} = \boldsymbol{f}^{-1}(\boldsymbol{z};\theta). \tag{2.4}$$

综上所述，设计一个标准化流模型主要需要考虑与式 (2.1)、(2.2) 和 (2.4) 的计算相关的三个要素：

1. $\boldsymbol{f}$ 是可逆映射，且 $\boldsymbol{f}(\cdot)$ 易于计算；
2. $\boldsymbol{f}$ 的雅克比矩阵的对数行列式 $\log\left|\det(J_{\boldsymbol{f}}(\boldsymbol{x}))\right|$ 易于计算；
3. 逆函数 $\boldsymbol{f}^{-1}(\cdot)$ 易于计算。

2.1.1　离散时间标准化流模型

离散时间标准化流模型[29-35]通过有限个可逆映射的复合来定义生成模型，其中“离散时间”表示该模型的计算过程是离散的有限次数。具体而言，离散时间标准化流模型定义了一系列易于计算的简单可逆映射 $\boldsymbol{f}_1, \cdots, \boldsymbol{f}_L$ 来将数据 $\boldsymbol{x}$ 转换为

某个对应的隐变量 $\boldsymbol{z}$：

$$\boldsymbol{x} \overset{\boldsymbol{f}_1}{\longleftrightarrow} \boldsymbol{h}_1 \overset{\boldsymbol{f}_2}{\longleftrightarrow} \boldsymbol{h}_2 \cdots \overset{\boldsymbol{f}_L}{\longleftrightarrow} \boldsymbol{z}, \tag{2.5}$$

其中对于每个 $\ell \in \{1, \cdots, L\}$，$\boldsymbol{f}_\ell : \mathbb{R}^d \to \mathbb{R}^d$ 的输入与输出维度都为 d 维，因此每个中间变量 $\boldsymbol{h}_\ell = \boldsymbol{f}_\ell(\boldsymbol{h}_{\ell-1})$ 也为 $\mathbb{R}^d$ 中的随机变量。为了简便起见，此处定义 $\boldsymbol{h}_0 = \boldsymbol{x}$ 和 $\boldsymbol{h}_L = \boldsymbol{z}$。记 $\boldsymbol{f} \coloneqq \boldsymbol{f}_L \circ \cdots \circ \boldsymbol{f}_1$ 是所有 L 个映射的复合，离散时间标准化流模型对应的三个要素分别为：

1. $\boldsymbol{z} = \boldsymbol{h}_L = \boldsymbol{f}(\boldsymbol{x})$ 的计算由每一层 $\boldsymbol{h}_\ell = \boldsymbol{f}_\ell(\boldsymbol{h}_{\ell-1})$ 迭代进行，其可逆性由每一个 $\boldsymbol{f}_\ell$ 的可逆性保证；
2. $\log \left|\det(\boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x}))\right|$ 的计算由雅克比矩阵的*链式法则*（chain rule）得到：

$$\log \left|\det(\boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x}))\right| = \sum_{\ell=1}^{L} \log \left|\det(\boldsymbol{J}_{\boldsymbol{f}_\ell}(\boldsymbol{h}_{\ell-1}))\right|, \tag{2.6}$$

   因此，需要保证每一层 $\boldsymbol{f}_\ell$ 的雅克比矩阵的对数行列式 $\log \left|\det(\boldsymbol{J}_{\boldsymbol{f}_\ell}(\boldsymbol{h}_{\ell-1}))\right|$ 也是易于计算的；
3. $\boldsymbol{x} = \boldsymbol{h}_0 = \boldsymbol{f}^{-1}(\boldsymbol{z})$ 的计算由反向的每一层 $\boldsymbol{h}_{\ell-1} = \boldsymbol{f}_\ell^{-1}(\boldsymbol{h}_\ell)$ 迭代进行。

因此，设计离散时间标准化流模型只需要分别设计每一层可逆映射 $\boldsymbol{f}_\ell$ 以满足以上三个要求即可。以下列出几种常见的离散时间标准化流模型所采用的单层可逆映射。

仿射耦合层。 仿射耦合层（affine coupling layer）[31]是一种非常常见的可逆变换，其中每个变换 $\boldsymbol{h}_\ell = \boldsymbol{f}_\ell(\boldsymbol{h}_{\ell-1}; \theta)$ 由如下定义：

$$\begin{aligned} &\boldsymbol{x}_1, \boldsymbol{x}_2 = \text{split}(\boldsymbol{h}_{\ell-1}), \\ &\boldsymbol{y}_1 = \boldsymbol{x}_1, \quad \boldsymbol{y}_2 = \boldsymbol{\mu}_\ell(\boldsymbol{x}_1; \theta) + \exp(\boldsymbol{s}_\ell(\boldsymbol{x}_1; \theta)) \odot \boldsymbol{x}_2, \\ &\boldsymbol{h}_\ell = \boldsymbol{f}_\ell(\boldsymbol{h}_{\ell-1}; \theta) = \text{concat}(\boldsymbol{y}_1, \boldsymbol{y}_2), \end{aligned} \tag{2.7}$$

其中 split(·) 是将输入分割成两个不相交部分的操作，concat(·) 是其逆操作，$\boldsymbol{\mu}_\ell(\cdot; \theta)$ 和 $\boldsymbol{s}_\ell(\cdot; \theta)$ 是输入输出维度都是 d 维的神经网络，$\odot$ 表示向量的*逐元素相乘*（element-wise multiplication）。此外，由于 split(·) 的特殊形式，仿射耦合层的雅克比矩阵具有上三角或下三角形式，因此其雅克比矩阵的对数行列式可以在 $\mathcal{O}(d)$ 时间内准确地计算：

$$\log \left|\det(\boldsymbol{J}_{\boldsymbol{f}_\ell}(\boldsymbol{h}_{\ell-1}))\right| = \|\boldsymbol{s}_\ell(\boldsymbol{x}_1)\|_1, \tag{2.8}$$

其中 $\|\cdot\|_1$ 表示向量的 L_1 范数。另外，仿射耦合层的逆变换也非常简单，同样是一个仿射耦合层：

$$\begin{aligned}
&\boldsymbol{y}_1, \boldsymbol{y}_2 = \text{split}(\boldsymbol{h}_\ell), \\
&\boldsymbol{x}_1 = \boldsymbol{y}_1, \quad \boldsymbol{x}_2 = \big(\boldsymbol{y}_2 - \boldsymbol{\mu}_\ell(\boldsymbol{y}_1;\theta)\big) / \exp(\boldsymbol{s}_\ell(\boldsymbol{y}_1;\theta)), \\
&\boldsymbol{h}_{\ell-1} = \boldsymbol{f}_\ell^{-1}(\boldsymbol{h}_\ell;\theta) = \text{concat}(\boldsymbol{x}_1, \boldsymbol{x}_2).
\end{aligned} \tag{2.9}$$

其中由于 $\boldsymbol{x}_1 = \boldsymbol{y}_1$，神经网络 $\boldsymbol{\mu}_\ell$ 和 $\boldsymbol{s}_\ell$ 的输出与正向变换相同，进而保证了可逆性。

可逆 1×1 卷积。　可逆 1×1 卷积[32]由一个在特征通道（channel）上的可逆矩阵 $\boldsymbol{W}_\ell$ 定义：

$$\boldsymbol{h}_{\ell,ij} = \big(\boldsymbol{f}_\ell(\boldsymbol{h}_{\ell-1};\boldsymbol{W}_\ell)\big)_{ij} = \boldsymbol{W}_\ell \boldsymbol{h}_{\ell-1,ij}, \tag{2.10}$$

其中假设 $\boldsymbol{h}_{\ell-1}$ 的维度 $d = H\times W\times C$（例如图片数据），$\boldsymbol{W}_\ell \in \mathbb{R}^{C\times C}$ 为可逆矩阵（由单位矩阵初始化），下标 $_{ij}$ 表示从 $H\times W\times C$ 维特征图中提取位于 (i,j) 位置的通道，因此对应维度为 C 维。由于 $\boldsymbol{W}_\ell$ 的可逆性，其雅克比矩阵的对数行列式可以在 $\mathcal{O}(C^3)$ 时间内计算出来，由于通常情况下 $C \ll H$ 以及 $C \ll W$，因此 $C^3 \ll d$，故计算复杂度可以接受，其具体表达式为：

$$\log\left|\det(\boldsymbol{J}_{\boldsymbol{f}_\ell}(\boldsymbol{h}_{\ell-1}))\right| = H\times W\times \log\left|\det(\boldsymbol{W}_\ell)\right|. \tag{2.11}$$

此外，可逆 1×1 卷积的逆变换只需要使用该可逆矩阵的逆进行线性映射即可：

$$\boldsymbol{h}_{\ell-1,ij} = \big(\boldsymbol{f}_\ell^{-1}(\boldsymbol{h}_\ell;\boldsymbol{W}_\ell)\big)_{ij} = \boldsymbol{W}_\ell^{-1}\boldsymbol{h}_{\ell,ij}. \tag{2.12}$$

残差标准化流模型。　残差标准化流模型是一类由可逆残差网络（invertible residual networks）[33-34]定义的可逆映射，详见第 3.2.2 节。

综上所述，离散时间标准化流模型都需要设计某种特殊的可逆神经网络，以满足可逆性、易于计算的前向和反向变换、易于计算的雅克比矩阵的对数行列式这三点要求。然而，这些要求限制了离散时间标准化流模型的表达能力。例如，仿射耦合层的雅克比矩阵只有三角形式，可逆 1×1 卷积的雅克比矩阵是低秩矩阵，而残差标准化流模型的雅克比矩阵的利普希兹常数受限（详见第 3.2.2 节）。针对这些问题，本文将在第 3 章给出系统性的研究，提出一种表达能力严格更强的离散时间标准化流模型。

2.1.2　连续时间标准化流模型

连续时间标准化流模型[36-38]是离散时间标准化流模型用于生成式建模的推广。这类模型通常将可逆变换视为动力系统（dynamical system），并由常微分方程

（ordinary differential equation，ODE）求解器（solver）近似地求解。由于常微分方程的时间是连续的实数，因此连续时间标准化流模型也被视为“无限层”（infinite-depth）神经网络。此外，由于该常微分方程通常由神经网络定义，故连续时间标准化流模型也被称为“神经常微分方程”（neural ODE）。

具体而言，假设对 $t \in [0,1]$，函数 $\boldsymbol{h}(\cdot,t)：\mathbb{R}^d \to \mathbb{R}^d$ 都为一阶连续可微函数，且函数关于 t 也连续，那么连续时间标准化流模型由如下常微分方程的解定义：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}(\boldsymbol{x}_t, t), \tag{2.13}$$

其中 $\boldsymbol{x}_0 = \boldsymbol{x}$ 为模型分布（拟合数据分布）对应的随机变量，$\boldsymbol{x}_1 = \boldsymbol{z}$ 为简单分布对应的随机变量，其对应的变换等价于：

$$\boldsymbol{x}_0 = \boldsymbol{x}_1 + \int_1^0 \boldsymbol{h}(\boldsymbol{x}_t, t)\mathrm{d}t. \tag{2.14}$$

由于交换积分上下限对应了原积分的相反数，因此上式对应的逆变换为：

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \int_0^1 \boldsymbol{h}(\boldsymbol{x}_t, t)\mathrm{d}t. \tag{2.15}$$

这也意味着连续时间标准化流模型的可逆性直接由常微分方程的性质保证，而不需要额外对神经网络 $\boldsymbol{h}$ 进行更多的可逆性限制，因此连续时间标准化流模型比离散时间标准化流模型更为灵活。然而，连续时间标准化流模型的前向和反向都需要常微分方程求解器，因此计算开销更大。此外，Chen 等人[36]证明了连续时间标准化流模型的对数似然满足一个 $d+1$ 维的常微分方程：

$$\begin{cases} \dfrac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}(\boldsymbol{x}_t, t), \\ \dfrac{\mathrm{d}\log p_t(\boldsymbol{x}_t)}{\mathrm{d}t} = -\mathrm{tr}(\nabla_{\boldsymbol{x}_t}\boldsymbol{h}(\boldsymbol{x}_t, t)). \end{cases} \tag{2.16}$$

因此，连续时间标准化流模型也可以基于最大似然估计的准则进行训练。

最后，虽然连续时间标准化流模型较为灵活，但由于其优化的不稳定性[90-91]和过多的常微分方程求解器的计算步数[71]，训练连续时间标准化流模型的难度很大，因此大规模应用此类模型仍然是一个开放的问题。针对该问题，本文将在第 4 章中给出一种针对连续时间标准化流模型的可扩展的高效训练算法。

2.2　扩散模型

扩散概率模型（diffusion probabilistic models）[43-45]，简称扩散模型（diffusion models），是一种新兴的具有强大表达能力的生成模型。该模型在许多任务上的表现都很出色，例如高分辨率图像生成[49,92]、图像编辑[50,93-94]、视频生成[51]、文本到图像生成[15,18,72-74]、语音合成[16,52-53,95]、分子生成[96-98]、三维数据生成[55,99-100]和数据压缩[54,101]。相较于被广泛使用的生成对抗网络（generative adversarial networks，

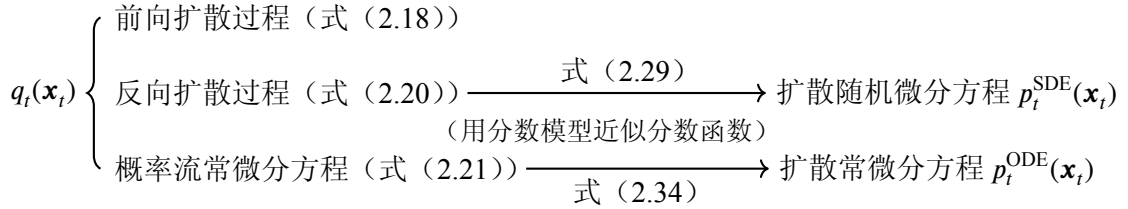


图 2.2　数据分布 q_t、扩散随机微分方程分布 p_t^{SDE} 和扩散常微分方程分布 p_t^{ODE} 之间的关系

GAN）[8]和变分自编码器（variational auto-encoders，VAE）[26]，扩散模型不仅可以计算精确的似然[45]，而且在图像生成方面可以实现更好的样本质量[49]。由于扩散模型也可以被理解为可逆生成模型，因此本节详细介绍扩散模型的定义与基础算法。

扩散模型由离散时间随机过程（stochastic process）[43-44]或连续时间随机微分方程（stochastic differential equation，SDE）[45]定义。具体而言，扩散模型定义了一个前向扩散过程（forward diffusion process），其对数据分布逐渐加噪直到对应的分布变为一个简单的高斯噪声分布。该前向扩散过程有一个解析的等价反向扩散过程（reverse diffusion process）[45,102]和一个解析的等价概率流常微分方程（probability flow ODE），这两个过程都只依赖于前向过程中每个时间对应的加噪数据分布的分数函数（score function，对数概率密度的梯度），且所有时间的边缘分布（marginal distribution）都与前向过程完全相同。扩散模型进而训练了一个参数化的神经网络，称为“分数模型”（score model）[45,48,103-104]，来拟合数据的分数函数，并通过该分数模型进一步定义了概率模型。

在扩散模型中，一个给定的分数模型可以对应两种概率模型。一种是扩散随机微分方程[45]（diffusion SDE），它通过将反向扩散过程中的分数函数替换为分数模型来定义模型，且可以生成高质量的样本。另一种是扩散常微分方程[45,105]（diffusion ODE），它通过将概率流常微分方程中的分数函数替换为分数模型来定义模型。由于扩散常微分方程可以被视为连续时间标准化流模型[36]，所以与扩散随机微分方程不同，扩散常微分方程可以通过常微分方程求解器[37]计算准确的对数似然（基于式（2.16））。

本节对扩散模型及其对应的不同分布进行形式化定义，其中不同分布之间的关系见图 2.2。

2.2.1　扩散过程的三种等价形式

本节对扩散模型中扩散过程的三种等价形式的定义进行详细描述，包括前向扩散过程、反向扩散过程和概率流常微分方程。

前向扩散过程。　假设 d 维随机变量 $\boldsymbol{x}_0 \in \mathbb{R}^d$ 服从未知数据分布 $q_0(\boldsymbol{x}_0)$。扩散模型定义了一个从时间 $t=0$ 到 $t=T>0$（T 通常取 1）的前向扩散过程 $\{\boldsymbol{x}_t\}_{t\in[0,T]}$，使得对于任意 $t\in[0,T]$，$\boldsymbol{x}_t$ 的分布在给定 $\boldsymbol{x}_0$ 的条件下满足

$$q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t|\alpha_t\boldsymbol{x}_0, \sigma_t^2\boldsymbol{I}), \tag{2.17}$$

其中 $\alpha_t, \sigma_t \in \mathbb{R}^+$ 是关于 t 的可微函数，且具有有界的导数。α_t 和 σ_t 的具体定义又被称为扩散模型的噪声时间表（noise schedule），例如线性噪声时间表（linear noise schedule）[44,81]和余弦噪声时间表（cosine noise schedule）[78]。令 $q_t(\boldsymbol{x}_t)$ 为 $\boldsymbol{x}_t$ 的边缘分布，扩散模型选择噪声时间表以保证 $q_T(\boldsymbol{x}_T) \approx \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I})$，并且信噪比（signal-to-noise ratio，SNR）α_t^2/σ_t^2 关于 t 严格递减[54]。此外，Kingma 等人[54]证明以下随机微分方程对于任何 $t\in[0,T]$ 都具有与式（2.17）中相同的条件概率分布 $q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)$，因此也称以下随机微分方程为扩散模型的前向扩散过程：

$$\mathrm{d}\boldsymbol{x}_t = f(t)\boldsymbol{x}_t\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}_t, \quad \boldsymbol{x}_0 \sim q_0(\boldsymbol{x}_0), \tag{2.18}$$

其中 $\boldsymbol{w}_t \in \mathbb{R}^d$ 是标准维纳过程（standard Wiener process），且

$$f(t) = \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}, \quad g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2. \tag{2.19}$$

反向扩散过程。　在一些正则条件下[102]，Song 等人[45]证明了式（2.18）中的前向扩散过程具有一个等价的从时间 T 到 0 的反向扩散过程：

$$\mathrm{d}\boldsymbol{x}_t = [f(t)\boldsymbol{x}_t - g^2(t)\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)]\mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}_t, \quad \boldsymbol{x}_T \sim q_T(\boldsymbol{x}_T), \tag{2.20}$$

其中 $\bar{\boldsymbol{w}}_t$ 是反向时间的标准维纳过程，并且每个时间 t 的 $\boldsymbol{x}_t$ 的边缘分布也是前向扩散过程的边缘分布 $q_t(\boldsymbol{x}_t)$。并且，式（2.20）中唯一的未知项为每个时间 t 的分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$。

概率流常微分方程。　在一些正则条件下[102]，Song 等人[45]证明了式（2.18）中的前向扩散过程也具有一个等价的从时间 T 到 0 的概率流常微分方程：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_q(\boldsymbol{x}_t, t) \coloneqq f(t)\boldsymbol{x}_t - \frac{1}{2}g^2(t)\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t), \quad \boldsymbol{x}_T \sim q_T(\boldsymbol{x}_T), \tag{2.21}$$

其中每个时间 t 的 $\boldsymbol{x}_t$ 的边缘分布也是前向扩散过程的边缘分布 $q_t(\boldsymbol{x}_t)$。与随机微分方程不同，常微分方程的对数似然可以通过式（2.16）精确计算。并且，式（2.21）中唯一的未知项也是每个时间 t 的分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$。

2.2.2　扩散模型的参数化形式

如第 2.2.1 节所述，时间 T 对应的分布 $q_T(\boldsymbol{x}_T)$ 约等于一个简单高斯分布 $\mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0},\sigma_T^2\boldsymbol{I})$。因此，为了得到数据分布 $q_0(\boldsymbol{x}_0)$ 的近似采样，只需要首先从时间 T 开始采样 $\boldsymbol{x}_T\sim\mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0},\sigma_T^2\boldsymbol{I})$，再根据反向扩散过程（式（2.20））或概率流常微分方程（式（2.21））计算得到最终的 $\boldsymbol{x}_0$ 即可。由于反向扩散过程和概率流常微分方程中唯一的未知项都是每个时间 t 的分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$，因此为了得到数据分布的近似采样，只需要估计每个时间 t 的分数函数。本节介绍扩散模型中关于分数函数的估计的不同参数化形式及其对应关系，包含分数模型（score model）、噪声预测模型（noise-prediction model）和数据预测模型（data-prediction model）。

分数模型。　分数模型用于估计所有 $\boldsymbol{x}_t\sim q_t(\boldsymbol{x}_t)$ 和所有 $t\in[0,T]$ 对应的分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$，其可以表示为一个由 θ 参数化的神经网络 $\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)$。分数模型通过不同时间 t 的加权分数匹配（score matching）进行训练：

$$\mathcal{J}_{\mathrm{SM}}(\theta;\omega(\cdot)):=\frac{1}{2}\int_0^T\omega(t)\mathbb{E}_{q_t(\boldsymbol{x}_t)}\Big[\|\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\|_2^2\Big]\mathrm{d}t,\tag{2.22}$$

其中 $\omega(\cdot)>0$ 是一个权重函数（weighting function）。由于 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$ 是未知的，式（2.22）中的分数匹配无法直接被计算。为了解决这个问题，Song 等人[45]基于去噪分数匹配（denoising score matching）[106]提出了式（2.22）的等效目标函数：

$$\mathcal{J}_{\mathrm{DSM}}(\theta;\omega(\cdot)):=\frac{1}{2}\int_0^T\frac{\omega(t)}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}}\Big[\|\sigma_t\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)+\boldsymbol{\epsilon}\|_2^2\Big]\mathrm{d}t,\tag{2.23}$$

其中 $\boldsymbol{x}_0\sim q_0(\boldsymbol{x}_0)$，$\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ 且 $\boldsymbol{x}_t=\alpha_t\boldsymbol{x}_0+\sigma_t\boldsymbol{\epsilon}$。并且可以证明，对于任意权重函数 $\omega(\cdot)$，都有

$$\nabla_\theta\mathcal{J}_{\mathrm{SM}}(\theta;\omega(\cdot))=\nabla_\theta\mathcal{J}_{\mathrm{DSM}}(\theta;\omega(\cdot)).\tag{2.24}$$

因此，分数模型可以通过 $\mathcal{J}_{\mathrm{DSM}}(\theta;\omega(\cdot))$ 进行训练。

噪声预测模型。　式（2.23）中的损失函数可以理解为将线性变换后的分数模型与加噪数据 $\boldsymbol{x}_t$ 中的噪声 $\boldsymbol{\epsilon}$ 进行最小二乘。因此，可以进一步定义噪声预测模型：

$$\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t):=-\sigma_t\boldsymbol{s}_\theta(\boldsymbol{x}_t,t),\tag{2.25}$$

其对应的训练目标函数为：

$$\mathcal{J}_{\mathrm{DSM}}(\theta;\omega(\cdot))=\frac{1}{2}\int_0^T\frac{\omega(t)}{\sigma_t^2}\mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}}\Big[\|\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t)-\boldsymbol{\epsilon}\|_2^2\Big]\mathrm{d}t,\tag{2.26}$$

其中 $\boldsymbol{x}_0\sim q_0(\boldsymbol{x}_0)$，$\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ 且 $\boldsymbol{x}_t=\alpha_t\boldsymbol{x}_0+\sigma_t\boldsymbol{\epsilon}$。由式（2.26）可以发现，噪声预测模型预测了加噪数据 $\boldsymbol{x}_t$ 中的噪声 $\boldsymbol{\epsilon}$。

数据预测模型。　由于 $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}$ 为原始数据 $\boldsymbol{x}_0$ 和噪声 $\boldsymbol{\epsilon}$ 的线性组合，因此可以基于噪声预测模型进一步定义*数据预测模型*：

$$\boldsymbol{x}_\theta(\boldsymbol{x}_t, t) \coloneqq \frac{\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)}{\alpha_t} = \frac{\boldsymbol{x}_t + \sigma_t^2 \boldsymbol{s}_\theta(\boldsymbol{x}_t, t)}{\alpha_t}, \tag{2.27}$$

其对应的训练目标函数为：

$$\mathcal{J}_{\mathrm{DSM}}(\theta; \omega(\cdot)) \coloneqq \frac{1}{2} \int_0^T \frac{\omega(t)\alpha_t^2}{\sigma_t^4} \mathbb{E}_{\boldsymbol{x}_0, \boldsymbol{\epsilon}} \left[\| \boldsymbol{x}_\theta(\boldsymbol{x}_t, t) - \boldsymbol{x}_0 \|_2^2 \right] \mathrm{d}t, \tag{2.28}$$

其中 $\boldsymbol{x}_0 \sim q_0(\boldsymbol{x}_0)$，$\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ 且 $\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}$。由式（2.28）可以发现，数据预测模型预测了加噪数据 $\boldsymbol{x}_t$ 中的原始数据 $\boldsymbol{x}_0$。

综上所述，通过不同的参数化方式可以对分数函数进行近似。因此，经过训练后，分数模型、噪声预测模型和数据预测模型可以将反向扩散过程和概率流常微分方程中的分数函数进行替换，从而得到参数化的生成模型。详细的讨论将在第 2.2.3 节和第 2.2.4 节给出。

2.2.3　扩散随机微分方程

通过用 $\mathcal{N}(\boldsymbol{x}_T | \boldsymbol{0}, \sigma_T^2 \boldsymbol{I})$ 近似 $q_T(\boldsymbol{x}_T)$，并把式（2.20）中的分数函数替换为分数模型（或等价的噪声预测模型和数据预测模型），扩散模型定义了反向扩散过程对应的参数化模型，称为*扩散随机微分方程*（diffusion SDE）：

$$\mathrm{d}\boldsymbol{x}_t = \left[f(t)\boldsymbol{x}_t - g^2(t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \right] \mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}_t, \quad \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T | \boldsymbol{0}, \sigma_T^2 \boldsymbol{I}). \tag{2.29}$$

记式（2.29）中每个时间 t 的解 $\boldsymbol{x}_t$ 对应的概率分布为 $p_t^{\mathrm{SDE}}(\boldsymbol{x}_t)$（省略下标 θ），其中 $p_T^{\mathrm{SDE}}(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T | \boldsymbol{0}, \sigma_T^2 \boldsymbol{I})$。特别地，扩散随机微分方程的模型分布 $p_0^{\mathrm{SDE}}(\boldsymbol{x}_0)$ 与数据分布 $q_0(\boldsymbol{x}_0)$ 的*库尔贝克-莱布勒散度*（Kullback-Leibler divergence，KL 散度）有如下形式的上界[105]：

$$\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{SDE}}) \leqslant \mathbb{D}_{\mathrm{KL}}(q_T \,\|\, p_T^{\mathrm{SDE}}) + \mathcal{J}_{\mathrm{SM}}(\theta; g^2(\cdot)). \tag{2.30}$$

因此，最小化 $\mathcal{J}_{\mathrm{SM}}(\theta; g^2(\cdot))$ 等价于对扩散随机微分方程分布 p_0^{SDE} 的最大似然训练，其中权重函数为 $\omega(\cdot) = g^2(\cdot)$。为简便起见，记

$$\mathcal{J}_{\mathrm{SM}}(\theta) \coloneqq \mathcal{J}_{\mathrm{SM}}(\theta; g^2(\cdot)). \tag{2.31}$$

此外，扩散随机微分方程的采样本质上对应了式（2.29）中的随机微分方程的数值求解，这需要从时间 $t = T$ 到时间 $t = 0$ 离散化该随机微分方程。Song 等人[45]证明，扩散模型传统的*祖先采样*（ancestral sampling）方法[44]可以视为式（2.29）的一阶随机微分方程求解器。然而，这些一阶方法通常需要数百到数千次函数调用才能收敛[45]，从而导致极其缓慢的采样速度。本文将在第 6 章中给出扩散随机微

分方程的高阶稳定的求解器。

为完整起见，以下同时给出扩散随机微分方程在噪声预测模型和数据预测模型参数化下对应的常微分方程，这些方程都与式（2.29）等价。其中，噪声预测模型对应的扩散随机微分方程为：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \left[f(t)\boldsymbol{x}_t + \frac{g^2(t)}{\sigma_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \right] \mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}_t, \quad \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I}). \tag{2.32}$$

数据预测模型对应的扩散随机微分方程为：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \left[\left(f(t) + \frac{g^2(t)}{\sigma_t^2} \right) \boldsymbol{x}_t - \frac{\alpha_t g^2(t)}{\sigma_t^2}\boldsymbol{x}_\theta(\boldsymbol{x}_t, t) \right] \mathrm{d}t + g(t)\mathrm{d}\bar{\boldsymbol{w}}_t, \quad \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I}). \tag{2.33}$$

2.2.4　扩散常微分方程

通过用 $\mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I})$ 近似 $q_T(\boldsymbol{x}_T)$，并把式（2.21）中的分数函数替换为分数模型（或等价的噪声预测模型和数据预测模型），扩散模型定义了概率流常微分方程对应的参数化模型，称为*扩散常微分方程*（diffusion ODE）：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_p(\boldsymbol{x}_t, t) \coloneqq f(t)\boldsymbol{x}_t - \frac{1}{2}g^2(t)\boldsymbol{s}_\theta(\boldsymbol{x}_t, t), \quad \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I}). \tag{2.34}$$

记式（2.29）中每个时间 t 的解 $\boldsymbol{x}_t$ 对应的概率分布为 $p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)$（省略下标 θ），其中 $p_T^{\mathrm{ODE}}(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I})$，因此 $p_T^{\mathrm{ODE}}(\boldsymbol{x}_T) = p_T^{\mathrm{SDE}}(\boldsymbol{x}_T)$。并且，由于扩散常微分方程也属于连续时间标准化流模型，其对数似然可以通过式（2.16）解析地计算。因此，与扩散随机微分方程中的上界（式（2.30））不同，扩散常微分方程可以进行精确的似然估计。

前人工作[45,105]使用扩散随机微分方程的最大似然估计（见式（2.30）定义的上界）同时训练扩散随机微分方程和扩散常微分方程的分数模型。然而，当 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \neq \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t, t)$ 时，分数匹配的目标函数 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 与扩散常微分方程和数据分布的 KL 散度 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 之间的关系仍然是一个未知的问题。本文将在第 4 章中详细讨论 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 与 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 的关系，并给出进一步最小化 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 的理论分析与算法。

此外，扩散常微分方程的采样本质上对应了式（2.34）中的常微分方程的数值求解，这需要从时间 $t = T$ 到时间 $t = 0$ 离散化该常微分方程。与扩散随机微分方程相比，离散化随机微分方程时的步长受维纳过程的随机性的限制[107]，因此较大的采样步长（较小的步数）常常导致结果不收敛的问题；但常微分方程没有随机性，因此可以用更大的采样步长（更少的采样步数）来求解。Song 等人[45]发现基于扩散常微分方程的高阶黑盒（black-box）求解器（RK45 求解器[108]）可以将扩

散模型的采样步数缩减至一百多步。然而，现有的通用常微分方程求解器在少步（约 10 步）的范围内仍无法生成令人满意的样本，因此扩散模型的加速采样仍然是一个关键问题。本文将在第 5 章中给出一种针对扩散常微分方程特殊设计的高阶求解器，并在第 6 章中给出一种更稳定的高阶求解器，使得扩散模型可以在 10 步到 20 步之间得到高质量的样本。

为完整起见，以下同时给出扩散常微分方程在噪声预测模型和数据预测模型参数化下对应的常微分方程，这些方程都与式（2.34）等价。其中，噪声预测模型对应的扩散常微分方程为：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_p(\boldsymbol{x}_t, t) = f(t)\boldsymbol{x}_t + \frac{g^2(t)}{2\sigma_t}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t), \quad \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I}). \tag{2.35}$$

数据预测模型对应的扩散常微分方程为：

$$\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_p(\boldsymbol{x}_t, t) = \left(f(t) + \frac{g^2(t)}{2\sigma_t^2}\right)\boldsymbol{x}_t - \frac{\alpha_t g^2(t)}{2\sigma_t^2}\boldsymbol{x}_\theta(\boldsymbol{x}_t, t), \quad \boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{x}_T|\boldsymbol{0}, \sigma_T^2\boldsymbol{I}). \tag{2.36}$$

2.2.5　条件扩散模型的引导采样

条件扩散模型旨在从给定额外条件变量的条件数据分布中进行采样，例如给定类别下的图像生成[49,109]以及给定文本描述下的图像生成[15,18,72-74]。对于条件扩散模型而言，其条件采样是通过*引导采样*（guided sampling）[49,109]完成的。本节给出常见的引导采样的具体定义。

具体而言，记 $q_0(\boldsymbol{x}_0|c)$ 为给定条件变量 c 的条件数据分布，$q_0(\boldsymbol{x}_0)$ 为其对应的边缘数据分布。再分别记 $q_t(\boldsymbol{x}_t|c)$ 和 $q_t(\boldsymbol{x}_t)$ 为时间 t 对应的条件加噪数据分布和边缘加噪数据分布。根据*贝叶斯定理*（Bayes' theorem），有[45,49]：

$$\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t|c) = \nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t) + \nabla_{\boldsymbol{x}_t}\log q_t(c|\boldsymbol{x}_t). \tag{2.37}$$

因此，只需要将 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t|c)$ 中对应的两项 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$ 和 $\nabla_{\boldsymbol{x}_t}\log q_t(c|\boldsymbol{x}_t)$ 分别近似，即可得到条件扩散模型的条件分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t|c)$ 的近似，进而可以通过扩散随机微分方程或扩散常微分方程进行采样，这就是引导采样的核心思想。由于 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$ 可以由分数模型近似，因此引导采样的关键在于近似 $\nabla_{\boldsymbol{x}_t}\log q_t(c|\boldsymbol{x}_t)$。根据近似方法的不同，引导采样包括*分类器引导*（classifier guidance）和*无分类器引导*（classifier-free guidance）。

分类器引导。　分类器引导[49]利用一个由参数 $\boldsymbol{\phi}$ 进行参数化的预训练分类器 $p_t^{\boldsymbol{\phi}}(c|\boldsymbol{x}_t)$ 的梯度来近似 $\nabla_{\boldsymbol{x}_t}\log q_t(c|\boldsymbol{x}_t)$ 项。因此，给定一个预训练的分数模型

$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$，分类器引导对应的条件分数模型为：

$$\tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_t, t, c) \coloneqq \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) + s \cdot \nabla_{\boldsymbol{x}_t} \log p_t^\phi(c|\boldsymbol{x}_t), \tag{2.38}$$

其中 $s > 0$ 是引导尺度（guidance scale）。实践中通常倾向于使用较大的引导尺度来增强采样结果与条件的匹配程度[15,73]。此外，根据式（2.25）和式（2.27），分类器引导对应的条件噪声预测模型和条件数据预测模型分别为：

$$\tilde{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t, c) \coloneqq \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) - s \cdot \sigma_t \nabla_{\boldsymbol{x}_t} \log p_t^\phi(c|\boldsymbol{x}_t), \tag{2.39}$$

$$\tilde{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t, c) \coloneqq \boldsymbol{x}_\theta(\boldsymbol{x}_t, t) + s \cdot \frac{\sigma_t^2}{\alpha_t} \nabla_{\boldsymbol{x}_t} \log p_t^\phi(c|\boldsymbol{x}_t). \tag{2.40}$$

无分类器引导。　无分类器引导[109]去除了对额外分类器的依赖，而是对于无条件模型和条件模型共享相同的参数化模型（例如噪声预测模型 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, c)$），其中无条件模型的输入 c 是一个特殊的占位符 $\varnothing$。因此，相应的条件分数模型、条件噪声预测模型和条件数据预测模型由如下定义：

$$\tilde{\boldsymbol{s}}_\theta(\boldsymbol{x}_t, t, c) \coloneqq (1 - s) \cdot \boldsymbol{s}_\theta(\boldsymbol{x}_t, t, \varnothing) + s \cdot \boldsymbol{s}_\theta(\boldsymbol{x}_t, t, c), \tag{2.41}$$

$$\tilde{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t, t, c) \coloneqq (1 - s) \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, \varnothing) + s \cdot \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, c), \tag{2.42}$$

$$\tilde{\boldsymbol{x}}_\theta(\boldsymbol{x}_t, t, c) \coloneqq (1 - s) \cdot \boldsymbol{x}_\theta(\boldsymbol{x}_t, t, \varnothing) + s \cdot \boldsymbol{x}_\theta(\boldsymbol{x}_t, t, c). \tag{2.43}$$

综上所述，条件扩散模型的引导采样通过条件模型与无条件模型的线性组合来定义新的条件分数模型或条件噪声预测模型或条件数据预测模型，进而代入扩散随机微分方程或扩散常微分方程中进行采样。其中，引导采样存在一个额外的超参数为引导尺度。由于引导采样在实践中有广泛的应用，因此针对引导采样的加速也是扩散模型的关键问题之一。本文将在第 6 章中详细分析高阶求解器在引导采样中应用时存在的问题，并提出针对引导采样的更稳定的高阶扩散模型采样器。

第 3 章　隐式标准化流模型

标准化流模型是一类经典的可逆生成模型，其通过一个显式的可逆映射来定义模型的概率分布。然而，由于可逆性的限制，标准化流模型的表达能力往往不足以建模复杂的数据分布。因此，建立一种表达能力丰富的标准化流模型是该类模型用于真实场景的关键。本章提出了隐式标准化流模型，该模型的可逆映射由非线性方程的根隐式地定义，从而推广了原有的标准化流模型。理论分析表明，隐式标准化流模型的函数空间具有利普希兹常数（Lipschitz constant）不受限的特性，因此表达能力严格强于已有模型。实验结果表明，隐式标准化流模型在多种密度估计任务中的性能都优于基准线方法。

3.1　本章引言

标准化流模型（normalizing flows）[30,110]通过指定从一个随机变量 $\boldsymbol{x}$ 到另一个随机变量 $\boldsymbol{z}$ 的可逆映射 $\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x})$ 来定义模型分布 $p_{\boldsymbol{x}}(\boldsymbol{x})$。根据变量替换公式（change-of-variable formula），模型的概率密度可以被准确地计算：

$$\log p_{\boldsymbol{x}}(\boldsymbol{x}) = \log p_{\boldsymbol{z}}(\boldsymbol{z}) + \log \left|\det(J_{\boldsymbol{f}}(\boldsymbol{x}))\right|, \tag{3.1}$$

其中 $p_{\boldsymbol{z}}(\boldsymbol{z})$ 遵循一个简单的概率分布，例如高斯分布。标准化流模型需要满足两个要求：（一）$\boldsymbol{x}$ 和 $\boldsymbol{z}$ 之间的映射是可逆的；（二）函数 $\boldsymbol{f}(\boldsymbol{x})$ 的雅克比矩阵（Jacobian matrix）$J_{\boldsymbol{f}}(\boldsymbol{x})$ 的对数行列式（log-determinant）是易于计算的。标准化流模型的研究主要集中在满足这些要求的前提下提高模型的表达能力。对于第二个要求，前人工作如逆自回归流模型（inverse autoregressive flows）[111]和实值非保体积映射模型（real-valued non-volume preserving transformations，RealNVP）[31]将函数族限制为仅仅具有上三角或下三角雅克比矩阵的函数。最近的一些工作提出了具有结构不受限的雅克比矩阵的模型，例如残差标准化流模型（residual normalizing flows，ResFlow）[33-34]。这些工作利用对数行列式的随机估计器（stochastic estimators）放宽了行列式的结构限制，使得函数族可以拓展至非三角结构的可逆映射。然而，为了保证可逆性，该类模型的每一层映射的利普希兹常数会极为受限。在一般情况下，这种限制会不利于概率建模，因为将一个简单的先验分布映射到复杂的数据分布可能需要该映射有一个非常大的利普希兹常数（参见图 3.3 中的二维示例）。更进一步，前述的所有方法都显式地定义了一个正向映射 $\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x})$。然而，并不是所有的可逆映射都具有显式表达形式，因此这种显式的表达形式可能会限制模

型的表达能力。

本章提出了隐式标准化流模型（implicit normalizing flows，ImpFlow）。该模型通过一个关于 $\boldsymbol{z}$ 和 $\boldsymbol{x}$ 的方程 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x})=\boldsymbol{0}$ 隐式地定义可逆映射，从而推广了标准化流模型。具体而言，给定 $\boldsymbol{x}$（或 $\boldsymbol{z}$），另一个变量 $\boldsymbol{z}$（或 $\boldsymbol{x}$）通过隐式的求根过程 $\boldsymbol{z}=\text{RootFind}(\boldsymbol{F}(\cdot,\boldsymbol{x}))$ 计算，而没有显式的表达形式。此外，前人工作中使用的显式映射 $\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})$ 可以视为隐式标准化流模型的特例，对应的方程的形式为 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x})=\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{z}=\boldsymbol{0}$。为了在表达能力和计算复杂度之间取得平衡，本章提出了隐式标准化流模型的特定形式，其中每个映射是一个残差标准化流模型和另一个残差标准化流模型的逆（inverse）的复合（composition）。在理论层面，本章研究了残差标准化流模型和隐式标准化流模型的函数空间的表达能力。由于隐式标准化流模型放宽了利普希兹常数的约束，单层隐式标准化流模型的函数空间严格包含了双层残差标准化流模型的函数空间。并且，对于任何具有固定层数的残差标准化流模型，存在无穷多的可逆函数，使得残差标准化流模型在建模时具有不可忽略的近似误差，但隐式标准化流模型可以精确地建模。

为了用于真实数据的概率建模，本章还提出一种基于隐式微分公式的可扩展算法来估计隐式标准化流模型的概率密度及其梯度，并从模型中采样。并且，隐式标准化流模型的梯度计算开销与残差标准化流模型大致相似，其中隐式求根带来的额外开销是可接受的。在分类任务和生成式建模任务的基准测试上，隐式标准化流模型都优于同等参数的残差标准化流模型。

3.2　本章背景

3.2.1　利普希兹连续函数

对于任何可微函数 $\boldsymbol{f}:\mathbb{R}^d\to\mathbb{R}^d$ 和任何 $\boldsymbol{x}\in\mathbb{R}^d$，记 $\boldsymbol{f}$ 在 $\boldsymbol{x}$ 处的雅克比矩阵为 $\boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\in\mathbb{R}^{d\times d}$。

定义 3.1： 函数 $\boldsymbol{f}:\mathbb{R}^d\to\mathbb{R}^d$ 被称为利普希兹连续函数（Lipschitz continuous function），若存在一个常数 L 使得

$$\|\boldsymbol{f}(\boldsymbol{x}_1)-\boldsymbol{f}(\boldsymbol{x}_2)\|\leqslant L\|\boldsymbol{x}_1-\boldsymbol{x}_2\|,\ \forall \boldsymbol{x}_1,\boldsymbol{x}_2\in\mathbb{R}^d. \tag{3.2}$$

满足上述不等式的最小的 L 被称为 f 的利普希兹常数，记为 $\text{Lip}(\boldsymbol{f})$。

一般来说，$\text{Lip}(\boldsymbol{f})$ 的定义取决于范数 $||\cdot||$ 的选择。在本章中，为简便起见，默认使用 L_2 范数 $||\cdot||_2$。

定义 3.2： 函数 $\boldsymbol{f}:\mathbb{R}^d\to\mathbb{R}^d$ 称为双向利普希兹连续函数（bi-Lipschitz continuous function），若其是利普希兹连续函数，且存在逆映射 $\boldsymbol{f}^{-1}$，该逆映射也是利普希兹

连续函数。

以下给出一个利普希兹常数的等价定义[112]。

命题 3.1： 若函数 $\boldsymbol{f}:\mathbb{R}^d\to\mathbb{R}^d$ 是利普希兹连续函数，则 $\boldsymbol{f}$ 几乎处处可微，且

$$\mathrm{Lip}(\boldsymbol{f})=\sup_{\boldsymbol{x}\in\mathbb{R}^d}\|J_{\boldsymbol{f}}(\boldsymbol{x})\|_2, \tag{3.3}$$

其中 $J_{\boldsymbol{f}}(\boldsymbol{x})$ 为函数 $\boldsymbol{f}$ 在 $\boldsymbol{x}$ 处的雅克比矩阵，$\|\boldsymbol{M}\|_2:=\sup_{\{\boldsymbol{v}:\|\boldsymbol{v}\|_2=1\}}\|\boldsymbol{M}\boldsymbol{v}\|_2$ 是矩阵 $\boldsymbol{M}\in\mathbb{R}^{d\times d}$ 的 L_2 范数（又被称为矩阵 $\boldsymbol{M}$ 的谱范数（spectral norm））。

3.2.2　残差标准化流模型

如式（3.1）所示，标准化流模型 $f:\boldsymbol{x}\mapsto\boldsymbol{z}$ 通过一个可逆映射定义模型概率分布，因此标准化流模型的表达能力取决于可逆映射 $\boldsymbol{f}$ 的表达能力。残差标准化流模型基于 $\boldsymbol{f}=\boldsymbol{f}_L\circ\cdots\circ\boldsymbol{f}_1$ 构造可逆映射，其中每层 $\boldsymbol{f}_\ell$ 的定义如下：

$$\boldsymbol{f}_\ell(\boldsymbol{x})=\boldsymbol{x}+\boldsymbol{g}_\ell(\boldsymbol{x}),\quad \mathrm{Lip}(\boldsymbol{g}_\ell)\leqslant\kappa<1, \tag{3.4}$$

其中 κ 是一个固定的常数，$\mathrm{Lip}(\boldsymbol{g})$ 是函数 $\boldsymbol{g}$ 的利普希兹常数（详见第 3.2.1 节）。可以证明，这样定义的 $\boldsymbol{f}_\ell$ 一定是可逆映射[33]。尽管残差标准化流模型的雅克比矩阵结构不受限，但残差标准化流模型的表达能力仍受可逆映射 $\boldsymbol{f}_\ell$ 的利普希兹常数限制。具体而言，由于每层残差标准化流模型 $\boldsymbol{f}_\ell$ 的利普希兹常数不超过 2（详见 Behrmann 等人[33]的证明），因此 $\boldsymbol{L}$ 层残差标准化流模型的利普希兹常数不超过 2^L。然而，为了将一个简单的先验分布映射到一个复杂的数据分布，这样的映射一般需要足够大的利普希兹常数。因此，仅仅为了满足需要的利普希兹常数约束，残差标准化流模型可能会不可避免地需要足够的深度（参见图 3.3 中的二维示例）。

3.2.3　隐式深度学习

隐式深度学习（implicit deep learning）利用隐式函数来增强神经网络的灵活性，使得可以针对特定问题设计对应的网络结构。例如，Bai 等人[113]提出了深度均衡模型（deep equilibrium models）作为循环神经网络（recurrent neural networks）[1]的更强替代；Amos 等人[114]通过解决一个优化问题来推广神经网络的每一层；Wang 等人[115]将逻辑推理整合到神经网络中；Reshniak 等人[116]利用隐式欧拉方法（implicit Euler method）提高残差网络的前向和后向过程的稳定性；Sitzmann 等人[117]使用周期函数（periodic function）进行表示学习（representation learning）。这些工作都只局限在将隐式函数作为前向神经网络的替代，而本章为标准化流模型设计了可逆的隐式函数（同时包括前向和逆向），使得其可以用于生成式建模。

3.3　模型定义与理论分析

本节将给出隐式标准化流模型的定义，并从理论上分析该模型的表达能力。

3.3.1　模型定义

一般而言，随机变量 $\boldsymbol{x}$ 和 $\boldsymbol{z}$（维度为 d）之间的映射可以由方程 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x})=\boldsymbol{0}$ 的根来隐式地定义，其中 $\boldsymbol{F}$ 是从 $\mathbb{R}^{2d}$ 到 $\mathbb{R}^{d}$ 的函数。特别地，前人工作[32,34]中使用的显式映射 $\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})$ 也可以等价于由方程 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x})=\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{z}=\boldsymbol{0}$ 定义。为了满足标准化流模型所要求的可逆性和对数行列式的可计算性，本章关注以下特定形式的方程定义的隐函数，它在表达能力和可计算性之间的取舍获得了良好的平衡。

定义 3.3： 令 $\boldsymbol{g_z}:\mathbb{R}^d\to\mathbb{R}^d$ and $\boldsymbol{g_x}:\mathbb{R}^d\to\mathbb{R}^d$ 为满足 $\mathrm{Lip}(g_{\boldsymbol{x}})<1$ 和 $\mathrm{Lip}(g_{\boldsymbol{z}})<1$ 的函数，其中 $\mathrm{Lip}(\boldsymbol{g})$ 是函数 $\boldsymbol{g}$ 的利普希兹常数。隐式标准化流模型的定义如下：

$$\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x})=\boldsymbol{0}，\text{其中}\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x})=\boldsymbol{g_x}(\boldsymbol{x})-\boldsymbol{g_z}(\boldsymbol{z})+\boldsymbol{x}-\boldsymbol{z}. \tag{3.5}$$

其中，式（3.5）的根对（root pairs）是 $\mathbb{R}^d\times\mathbb{R}^d$ 空间的一个子集，该子集实际上定义了唯一的可逆函数 $\boldsymbol{f}$ 的赋值规则。具体而言，对于任何给定的 $\boldsymbol{x}_0$，根据定义 3.3，可以构造压缩映射（contraction mapping）$h_{\boldsymbol{x}_0}(\boldsymbol{z})=F(\boldsymbol{z},\boldsymbol{x}_0)+\boldsymbol{z}$，它具有唯一的不动点。该不动点对应于方程 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x}_0)=\boldsymbol{0}$ 关于 $\boldsymbol{z}$ 的唯一根，记为 $\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x}_0)$。同理，在反向过程中，给定 $\boldsymbol{z}_0$，方程 $\boldsymbol{F}(\boldsymbol{z}_0,\boldsymbol{x})=\boldsymbol{0}$ 关于 $\boldsymbol{x}$ 的根也是存在且唯一的，记为 $\boldsymbol{x}=\boldsymbol{f}^{-1}(\boldsymbol{z}_0)$。因此，$\boldsymbol{f}$ 的存在性和可逆性都可被保证，如下述定理所示。

定理 3.1： 式（3.5）定义了一个唯一的映射 $\boldsymbol{f}:\mathbb{R}^d\to\mathbb{R}^d$，$\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})$，且 $\boldsymbol{f}$ 是可逆的。

证明 首先，$\forall\boldsymbol{x}_0\in\mathbb{R}^d$，定义映射

$$\boldsymbol{h}_{\boldsymbol{x}_0}(\boldsymbol{z})=\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x}_0)+\boldsymbol{z}. \tag{3.6}$$

由于 g_z 是利普希兹函数，有

$$\|(\boldsymbol{F}(\boldsymbol{z}_1,\boldsymbol{x}_0)+\boldsymbol{z}_1)-(\boldsymbol{F}(\boldsymbol{z}_2,\boldsymbol{x}_0)+\boldsymbol{z}_2)\|=\|\boldsymbol{g}_z(\boldsymbol{z}_1)-\boldsymbol{g}_z(\boldsymbol{z}_2)\|<\|\boldsymbol{z}_1-\boldsymbol{z}_2\|. \tag{3.7}$$

因此，$\boldsymbol{h}_{\boldsymbol{x}_0}(\boldsymbol{z})$ 是一个压缩映射，故有唯一的不动点，记为 $\boldsymbol{f}(\boldsymbol{x}_0)$：

$$\boldsymbol{h}_{\boldsymbol{x}_0}(\boldsymbol{f}(\boldsymbol{x}_0))=\boldsymbol{f}(\boldsymbol{x}_0)\Leftrightarrow\boldsymbol{F}(\boldsymbol{f}(\boldsymbol{x}_0),\boldsymbol{x}_0)=\boldsymbol{0}. \tag{3.8}$$

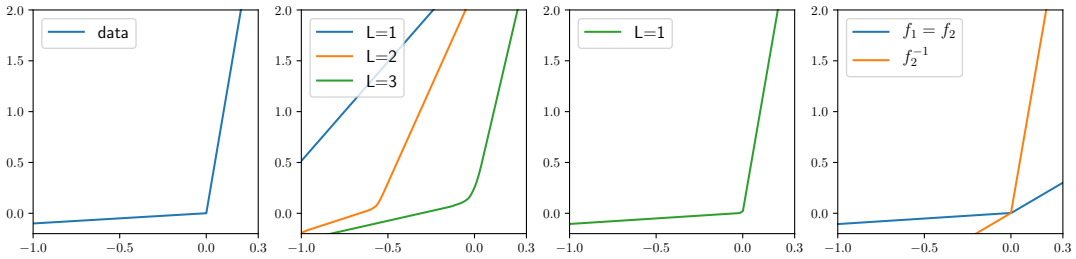
同理，$\forall\boldsymbol{z}_0\in\mathbb{R}^d$，存在唯一的 $g(\boldsymbol{z}_0)$ 满足 $\boldsymbol{F}(\boldsymbol{z}_0,\boldsymbol{g}(\boldsymbol{z}_0))=\boldsymbol{0}$。

更进一步，令 $\boldsymbol{z}_0=\boldsymbol{f}(\boldsymbol{x}_0)$，有 $\boldsymbol{F}(\boldsymbol{f}(\boldsymbol{x}_0),\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_0)))=\boldsymbol{0}$。由于唯一性，有 $\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_0))=\boldsymbol{x}_0,\forall\boldsymbol{x}_0\in\mathbb{R}^d$；同理，$\boldsymbol{f}(\boldsymbol{g}(\boldsymbol{x}_0))=\boldsymbol{x}_0,\forall\boldsymbol{x}_0\in\mathbb{R}^d$。因此，$\boldsymbol{f}$ 是唯一的，且可逆。 ∎

定理 3.1 说明了在定义 3.3 中引入隐式标准化流模型的有效性。事实上，单层隐式标准化流模型是一个单层残差标准化流模型和另一个单层残差标准化流模型

(a) $\mathcal{R}_2$ 和 $\mathcal{I}$ 的关系　　(b) $\mathcal{R}_\ell$ 和 $\mathcal{I}$ 的关系

图 3.1　本章主要理论结果的图解：残差标准化流模型和隐式标准化流模型的表达能力



(a) 目标函数　(b) 残差标准化流模型　(c) 隐式标准化流模型　(d) 函数复合

图 3.2　残差标准化流模型和隐式标准化流模型对一维函数的建模结果

的逆的复合，该结论将在第 3.3.2 节中正式给出。

3.3.2　模型表达能力分析

本节将给出隐式标准化流模型的表达能力的理论分析，特别是与残差标准化流模型的比较。具体而言，第 3.3.2.1 节中证明了隐式标准化流模型的函数空间严格包含残差标准化流模型的函数空间（如图 3.1 (a) 所示）。此外，对于任何具有固定层数的残差标准化流模型，存在无穷多的可逆函数，使得残差标准化流模型在建模时具有不可忽略的近似误差，但该函数可以由单层隐式标准化流模型精确地建模，如图 3.1 (b) 所示。

3.3.2.1　与双层残差标准化流模型的比较

本节将形式化地给出单层隐式标准化流模型和双层残差标准化流模型的表达能力的比较。本节所有的理论结果如图 3.1 (a) 所示。此外，图 3.2 还展示了一个一维的示例。

一方面，根据残差标准化流模型的定义，单层残差标准化流模型的函数族为

$$\mathcal{R} \coloneqq \{\boldsymbol{f} : \boldsymbol{f} = \boldsymbol{g} + \mathrm{Id},\ \boldsymbol{g} \in \boldsymbol{C}^1(\mathbb{R}^d, \mathbb{R}^d), \mathrm{Lip}(\boldsymbol{g}) < 1\}, \tag{3.9}$$

其中 $\boldsymbol{C}^1(\mathbb{R}^d, \mathbb{R}^d)$ 为所有 $\boldsymbol{R}^d$ 到 $\boldsymbol{R}^d$ 的一阶连续可微函数组成的集合，Id 为恒等映射。此外，ℓ 层残差标准化流模型的函数族由函数复合所定义：

$$\mathcal{R}_\ell \coloneqq \{\boldsymbol{f} : \boldsymbol{f} = \boldsymbol{f}_\ell \circ \cdots \circ \boldsymbol{f}_1,\ \text{其中} \boldsymbol{f}_1, \cdots, \boldsymbol{f}_\ell \in \mathcal{R}\}. \tag{3.10}$$

由式（3.9）和式（3.10）的定义，有 $\mathcal{R}_1 = \mathcal{R}$。

另一方面，根据式（3.5）中隐式标准化流模型的定义，有

$$(\boldsymbol{g}_{\boldsymbol{x}} + \mathrm{Id})(\boldsymbol{x}) = \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}) + \boldsymbol{x} = \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}) + \boldsymbol{z} = (\boldsymbol{g}_{\boldsymbol{z}} + \mathrm{Id})(\boldsymbol{z}), \tag{3.11}$$

其中 $\circ$ 表示函数的复合。等价地，有 $\boldsymbol{z} = \left((\boldsymbol{g}_{\boldsymbol{z}} + \mathrm{Id})^{-1} \circ (\boldsymbol{g}_{\boldsymbol{x}} + \mathrm{Id})\right)(\boldsymbol{x})$, 这意味着单层隐式标准化流模型的函数族为

$$\mathcal{I} = \{\boldsymbol{f} : \boldsymbol{f} = \boldsymbol{f}_2^{-1} \circ \boldsymbol{f}_1, \text{ 其中} \boldsymbol{f}_1, \boldsymbol{f}_2 \in \mathcal{R}\}. \tag{3.12}$$

直观来讲，单层隐式标准化流模型可以等价地理解为一个单层残差标准化流模型和另一个单层残差标准化流模型的逆函数的复合。然而，这种复合可能没有显式的表达式，例如图 3.2 (c) 和 (d) 中的一维示例。因此，一个自然的想法是探究 $\mathcal{I}$ 和 $\mathcal{R}_2$ 之间的关系。为此，以下首先引入一个没有显式利普希兹常数约束的递增函数族，并证明它严格包含 $\mathcal{R}$。

引理 3.1：

$$\mathcal{R} \subsetneq \mathcal{F} \coloneqq \{\boldsymbol{f} \in \mathcal{D} : \inf_{\boldsymbol{x} \in \mathbb{R}^d, \boldsymbol{v} \in \mathbb{R}^d, \|\boldsymbol{v}\|_2 = 1} \boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x}) \boldsymbol{v} > 0\}, \tag{3.13}$$

其中 $\mathcal{D}$ 是所有 $\mathbb{R}^d$ 到 $\mathbb{R}^d$ 的一阶连续可微的双向利普希兹连续（bi-Lipschitz continuous）的可逆函数组成的集合，$A \subsetneq B$ 表示 A 是 B 的真子集。

证明见第 3.5.1 节。

为了更好地理解 $\mathcal{F}$，此处展示一维的特殊情形。根据 Behrmann 等人[33]的证明，所有 $\mathcal{R}$ 中的函数都是双向利普希兹连续的，所以 $\mathcal{R} \subsetneq \mathcal{D}$。在一维数据的情况下，可以得到

$$\mathcal{R} = \{f \in C^1(\mathbb{R}) : \inf_{x \in \mathbb{R}} f'(x) > 0, \sup_{x \in \mathbb{R}} f'(x) < 2\}, \tag{3.14}$$

以及

$$\mathcal{F} = \{f \in C^1(\mathbb{R}) : \inf_{x \in \mathbb{R}} f'(x) > 0\}. \tag{3.15}$$

由此可见，$\mathcal{R}$ 的利普希兹常数是受限的，但 $\mathcal{F}$ 的利普希兹常数完全不受限。更一般地，在高维情况下，$\mathcal{R}$ 和 $\mathcal{F}$ 中的函数很难简单地进行说明，但仍然有类似的结论：$\mathcal{R}$ 中的函数的利普希兹常数是受限的（小于 2）[33]，但 $\mathcal{F}$ 中的函数的利普希兹常数可以任意地大。更进一步，基于引理 3.1，可以证明隐式标准化流模型的函数族 $\mathcal{I}$ 由两个 $\mathcal{F}$ 中的函数的复合构成，因此严格包含了 $\mathcal{R}_2$，如下述定理所示。

定理 3.2 (单层隐式标准化流模型的等价形式)：

$$\mathcal{I} = \mathcal{F}_2 \coloneqq \{\boldsymbol{f} : \boldsymbol{f} = \boldsymbol{f}_2 \circ \boldsymbol{f}_1, \text{ 其中} \boldsymbol{f}_1, \boldsymbol{f}_2 \in \mathcal{F}\}. \tag{3.16}$$

证明见第 3.5.2 节。

注意到恒等映射 $\mathrm{Id} \in \mathcal{F}$，因此有 $\mathcal{F} \subset \mathcal{I}$。由此可得，单层隐式标准化流模型（及其逆）的利普希兹常数可以任意地大。此外，由于 $\mathcal{R} \subsetneq \mathcal{F}$，并且存在一些属于 $\mathcal{I} \setminus \mathcal{R}_2$ 的函数（见第 3.3.2.2 节中构造的示例），易得以下推论：

推论 3.1： $\mathcal{R} \subsetneq \mathcal{R}_2 \subsetneq \mathcal{F}_2 = \mathcal{I}$.

图 3.2 (b) 和 (c) 中一维示例的结果也进一步验证了推论 3.1 的结论。此外，推论 3.1 也可以很容易地推广到 2ℓ 层残差标准化流模型和 ℓ 层隐式标准化流模型的情况，这也能说明隐式标准化流模型的表达能力的优势。

3.3.2.2　与多层残差标准化流模型的比较

本节将进一步探讨 $\mathcal{R}_\ell$（当 $\ell > 2$ 时）与 $\mathcal{I}$ 之间的关系，相关的结论如图 3.1 (b) 所示。

对于任意给定的 ℓ，$\mathcal{R}_\ell$ 中的函数的利普希兹常数仍然是有界的，并且存在无穷多的函数，使得其不在 $\mathcal{R}_\ell$ 中但在 $\mathcal{I}$ 中。具体而言，构造函数族如下：对于任何 $L, r \in \mathbb{R}^+$，定义

$$\mathcal{P}(L,r) = \{\boldsymbol{f} : \boldsymbol{f} \in \mathcal{F}, \exists\, \mathcal{B}_r \subset \mathbb{R}^d, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{B}_r, \|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{y})\|_2 \geqslant L\|\boldsymbol{x} - \boldsymbol{y}\|_2\}, \tag{3.17}$$

其中 $\mathcal{B}_r$ 是一个半径为 r 的 d 维球。显然，$\mathcal{P}(L,r)$ 包含无穷多的元素。以下将证明，$\forall\, 0 < \ell < \log_2(L)$，对于任意 $\mathcal{P}(L,r)$ 中的函数，$\mathcal{R}_\ell$ 有不可忽略的近似误差，但 $\mathcal{I}$ 却可以精确地表示。

定理 3.3： 给定 $L > 0$ 和 $r > 0$，有

- $\mathcal{P}(L,r) \subset \mathcal{I}$.
- $\forall\, 0 < \ell < \log_2(L), \mathcal{P}(L,r) \cap \mathcal{R}_\ell = \varnothing$。并且，对于任意 $\boldsymbol{f} \in \mathcal{P}(L,r)$ 及 d 维球 $\mathcal{B}_r$，用 $\mathcal{R}_\ell$ 中的函数拟合 $\mathcal{B}_r$ 中的函数 $\boldsymbol{f}$ 的最小误差满足

$$\inf_{\boldsymbol{g} \in \mathcal{R}_\ell} \sup_{\boldsymbol{x} \in \mathcal{B}_r} \|\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{g}(\boldsymbol{x})\|_2 \geqslant \frac{r}{2}(L - 2^\ell) \tag{3.18}$$

证明见第 3.5.3 节。

根据定理 3.3，建模 $\boldsymbol{f} \in \mathcal{P}(L,r)$ 只需要一个单层隐式标准化流模型，但至少需要 $\log_2(L)$ 层的残差标准化流模型才可以建模。此外，图 3.2 (b) 展示了一个一维示例函数，该函数无法被 3 层残差标准化流模型建模，但可由单层隐式标准化流模型精确表示。

3.3.3　生成式建模算法

隐式标准化流模型可以由参数化的神经网络复合得到深度生成模型，并用于高维数据分布的建模。本节提出一种可扩展的算法用于隐式标准化流模型的推断、采样和训练。为简便起见，本节的推导过程只关注单层的隐式标准化流模型。

具体而言，单层的参数化的隐式标准化流模型 $\boldsymbol{z} = f(\boldsymbol{x};\theta)$ 定义如下：

$$\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta) = \boldsymbol{0},\ \text{其中}\ \boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta) = \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x};\theta) - \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z};\theta) + \boldsymbol{x} - \boldsymbol{z}, \tag{3.19}$$

且 $\mathrm{Lip}(\boldsymbol{g}_{\boldsymbol{x}}) < 1$，$\mathrm{Lip}(\boldsymbol{g}_{\boldsymbol{z}}) < 1$，$\theta$ 表示 $\boldsymbol{g}_{\boldsymbol{x}}$ 和 $\boldsymbol{g}_{\boldsymbol{z}}$ 中所有的参数（但这并不意味着 $\boldsymbol{g}_{\boldsymbol{x}}$ 和 $\boldsymbol{g}_{\boldsymbol{z}}$ 共享参数）。需要注意的是，$\boldsymbol{x}$ 代表的是该层的输入，而不是输入数据。

单层隐式标准化流模型中，对于给定的 $\boldsymbol{x}$ 计算 $\boldsymbol{z}$ 的*推断过程*对应于求解关于 $\boldsymbol{z}$ 的方程 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta) = \boldsymbol{0}$ 的根。由于隐式标准化流模型基于隐式表示，$\boldsymbol{z}$ 的求解不能被显式地计算。为了解决这个问题，以下采用*拟牛顿法*[118]迭代地求解，每一步的迭代如下：

$$\boldsymbol{z}^{[i+1]} = \boldsymbol{z}^{[i]} - \alpha \boldsymbol{B}\boldsymbol{F}(\boldsymbol{z}^{[i]},\boldsymbol{x};\theta),\ \text{迭代地进行}\ i = 0, 1, \cdots \tag{3.20}$$

其中 $\boldsymbol{B}$ 是雅克比矩阵的逆的*低秩近似*①，α 是*线搜索方法*[118]动态计算的步长，停止准则是 $|\boldsymbol{F}(\boldsymbol{z}^{[i]},\boldsymbol{x};\theta)|_2 < \epsilon_{\boldsymbol{f}}$，其中 $\epsilon_{\boldsymbol{f}}$ 是用于平衡计算时间和计算精度的超参数。由于定理 3.1 保证了根的存在性和唯一性，拟牛顿法的收敛性也得到了保证，且其收敛速度通常快于线性求解算法的收敛速度。

另一个推断问题是估计输入数据 $\boldsymbol{x}$ 的*对数似然*（log-likelihood）。假设 $\boldsymbol{z} \sim p(\boldsymbol{z})$，其中 $p(\boldsymbol{z})$ 是一个简单的先验分布（例如标准高斯分布）。$\boldsymbol{x}$ 的对数似然可以写成

$$\log p(\boldsymbol{x}) = \log p(\boldsymbol{z}) + \log\det(I + J_{g_{\boldsymbol{x}}}(\boldsymbol{x})) - \log\det(I + J_{g_{\boldsymbol{z}}}(\boldsymbol{z})), \tag{3.21}$$

详细推导参见第 3.5.4 节。然而，精确计算对数行列式对应的项需要 $\mathcal{O}(d^3)$ 的时间成本，这难以扩展到高维数据的情形。为了解决这个问题，本节基于 Chen 等人[34]提出的 Skilling-Hutchinson 随机迹（trace）估计方法[119-120]来对 $\log p(\boldsymbol{x})$ 进行无偏估计：

$$\log p(\boldsymbol{x}) = \log p(\boldsymbol{z}) + \mathbb{E}_{n\sim p(N),\boldsymbol{v}\sim\mathcal{N}(0,I)}\left[\sum_{k=1}^{n}\frac{(-1)^{k+1}}{k}\frac{\left(\boldsymbol{v}^T[J_{g_{\boldsymbol{x}}}(\boldsymbol{x})]^k\boldsymbol{v} - \boldsymbol{v}^T[J_{g_{\boldsymbol{z}}}(\boldsymbol{z})]^k\boldsymbol{v}\right)}{\mathbb{P}(N \geqslant k)}\right], \tag{3.22}$$

其中 $p(N)$ 为定义在正整数上的随机变量的概率分布。

*采样过程*需要对给定的 $\boldsymbol{z}$ 计算对应的 $\boldsymbol{x}$，这也可以通过拟牛顿法计算方程 $\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta) = \boldsymbol{0}$ 的根来解决，对应的超参数与推断过程一致。

① 计算细节参见 Broyden[118] 提出的算法。

训练过程基于随机梯度下降（stochastic gradient descent）来最小化训练数据的负对数似然（记为 $\mathcal{L}$），并以反向传播（backpropagation）的方式估计模型参数的梯度。根据链式法则（chain rule）和对数的可加性，每一层隐式标准化流模型的梯度计算都需要估计式（3.21）对 $\boldsymbol{x}$ 和 θ 的梯度。具体而言，梯度计算涉及两项：一项是 $\frac{\partial}{\partial(\cdot)}\log\det(\boldsymbol{I}+\boldsymbol{J}_{\boldsymbol{g}}(\boldsymbol{x};\theta))$，另一项是 $\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial(\cdot)}$，其中 $\boldsymbol{g}$ 是一个满足 $\mathrm{Lip}(\boldsymbol{g})<1$ 的函数，$(\cdot)$ 表示 $\boldsymbol{x}$ 或 θ。第一项的计算参考了 Chen 等人[34]提出的无偏梯度估计方法，具体表述如下：

$$\frac{\partial\log\det(\boldsymbol{I}+\boldsymbol{J}_{\boldsymbol{g}}(\boldsymbol{x};\theta))}{\partial(\cdot)}=\mathbb{E}_{n\sim p(N),\boldsymbol{v}\sim\mathcal{N}(0,I)}\left[\left(\sum_{k=0}^{n}\frac{(-1)^k}{\mathbb{P}(N\geqslant k)}\boldsymbol{v}^T\boldsymbol{J}_{\boldsymbol{g}}(\boldsymbol{x};\theta)^k\right)\frac{\partial\boldsymbol{J}_{\boldsymbol{g}}(\boldsymbol{x};\theta)}{\partial(\cdot)}\boldsymbol{v}\right],\tag{3.23}$$

其中 $p(N)$ 为定义在正整数上的随机变量的概率分布。而第二项 $\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial(\cdot)}$ 可以根据隐函数定理（implicit function theorem）计算，如下所示（详见第 3.5.4 节的推导）：

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial(\cdot)}=\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\boldsymbol{J}_{\boldsymbol{G}}^{-1}(\boldsymbol{z})\frac{\partial\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta)}{\partial(\cdot)},\ \text{其中}\ \boldsymbol{G}(\boldsymbol{z};\theta)\coloneqq\boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z};\theta)+\boldsymbol{z}.\tag{3.24}$$

对于高维数据，精确计算雅克比矩阵的逆的开销很高。为了降低计算开销，本节基于 Bai 等人[113]所提方法来计算 $\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\boldsymbol{J}_{\boldsymbol{G}}^{-1}(\boldsymbol{z})$，即求解如下关于变量 $\boldsymbol{y}$ 的线性方程：

$$\boldsymbol{J}_{\boldsymbol{G}}^{T}(\boldsymbol{z})\boldsymbol{y}^T=(\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}})^T,\tag{3.25}$$

其中等式左边是一个向量-雅克比矩阵乘积（vector-Jacobian product），这可以通过自动微分的编程库有效地计算（而无需计算雅克比矩阵）。求解该线性方程同样也使用拟牛顿法来计算 $\boldsymbol{y}$ 的值，其中停止准则的误差界为 ϵ_b。在实践中，前向计算时拟牛顿法的超参数 $\epsilon_f=10^{-6}$，反向传播时拟牛顿法的超参数 $\epsilon_b=10^{-10}$。前向计算和反向计算的算法流程详见算法 3.1 和算法 3.2。

3.4　实验结果

本节从分类任务和密度建模任务上验证隐式标准化流模型的表达能力。所有实验都使用谱归一化（spectral normalization）[121]来对利普希兹常数进行约束，其中每层的利普希兹常数的上界记为 c。对于伪牛顿法的超参数，训练和测试中都使用 $\epsilon_f=10^{-6}$ 和 $\epsilon_b=10^{-10}$，以在数值上确保训练期间的可逆性和稳定性。

3.4.1　分类任务

本节首先在分类任务上比较残差标准化流模型（ResFlow）和隐式标准化流模型（ImpFlow）的表达能力。本节实验的网络结构全部基于 ResNet-18[4]，并移除了每个残差块（residual block）内所有的批归一化（batch normalization）层，仅保留下

算法 3.1 单层隐式标准化流模型的前向计算算法

输入： 变量 $\boldsymbol{x}$，式（3.5）中的函数 $\boldsymbol{g}_{\boldsymbol{x};\theta}$ 和 $\boldsymbol{g}_{\boldsymbol{z};\theta}$，停止阈值 ϵ_f

输出： 隐变量 $\boldsymbol{z}=\boldsymbol{f}(\boldsymbol{x})$ 和模型的概率密度 $\log p(\boldsymbol{x})$，其中 $\boldsymbol{f}$ 是由 $\boldsymbol{g}_{\boldsymbol{x};\theta}$ 和 $\boldsymbol{g}_{\boldsymbol{z};\theta}$ 定义的隐函数

令 $\boldsymbol{h}(\boldsymbol{z}) \coloneqq \boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta)$，$\boldsymbol{z}\leftarrow\boldsymbol{0}$
while $\|\boldsymbol{h}(\boldsymbol{z})\|_2 \geqslant \epsilon_f$ **do**
    $\boldsymbol{B}\leftarrow$ 伪牛顿法估计的 $\boldsymbol{h}(\boldsymbol{z})$ 的雅克比矩阵的伪逆
    $\alpha\leftarrow$ 线搜索$(\boldsymbol{z},\boldsymbol{h},\boldsymbol{B})$
    $\boldsymbol{z}\leftarrow\boldsymbol{z}-\alpha\boldsymbol{B}\boldsymbol{h}(\boldsymbol{z})$
end while
if 训练 **then**
    基于式（3.23）估计 $\log\det(\boldsymbol{I}+\boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x};\theta))$
    基于式（3.23）估计 $\log\det(\boldsymbol{I}+\boldsymbol{J}_{\boldsymbol{g}_z}(\boldsymbol{z};\theta))$
else
    基于式（3.22）估计 $\log\det(\boldsymbol{I}+\boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x};\theta))$
    基于式（3.22）估计 $\log\det(\boldsymbol{I}+\boldsymbol{J}_{\boldsymbol{g}_z}(\boldsymbol{z};\theta))$
end if
基于式（3.21）计算 $\log p(\boldsymbol{x})$

算法 3.2 单层隐式标准化流模型的反向传播算法

输入： 前向的输入变量 $\boldsymbol{x}$，输出变量 $\boldsymbol{z}$，损失函数对 $\boldsymbol{z}$ 的梯度 $\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}$，式（3.5）中的函数 $\boldsymbol{g}_{\boldsymbol{x};\theta}$ 和 $\boldsymbol{g}_{\boldsymbol{z};\theta}$，停止阈值 ϵ_b。

输出： 反向传播时 $\boldsymbol{z}$ 回传到 $\boldsymbol{x}$ 的梯度 $\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{x}}$ 和回传到 θ 的梯度 $\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial\theta}$

令 $\boldsymbol{G}(\boldsymbol{z};\theta)\coloneqq\boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z};\theta)+\boldsymbol{z}$，$\boldsymbol{h}(\boldsymbol{y})=\boldsymbol{y}\boldsymbol{J}_{\boldsymbol{G}}(\boldsymbol{z})-\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}$，$\boldsymbol{y}\leftarrow\boldsymbol{0}$
while $\|\boldsymbol{h}(\boldsymbol{y})\|_2 \geqslant \epsilon_b$ **do**
    $\boldsymbol{B}\leftarrow$ 伪牛顿法估计的 $\boldsymbol{h}(\boldsymbol{y})$ 的雅克比矩阵的伪逆
    $\alpha\leftarrow$ 线搜索$(\boldsymbol{y},\boldsymbol{h},\boldsymbol{B})$
    $\boldsymbol{y}\leftarrow\boldsymbol{y}-\alpha\boldsymbol{B}\boldsymbol{h}(\boldsymbol{y})$
end while
基于自动微分的编程库，计算 $\frac{\partial\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta)}{\partial\boldsymbol{x}}$ 和 $\frac{\partial\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta)}{\partial\theta}$
$\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial\boldsymbol{x}}\leftarrow\boldsymbol{y}\frac{\partial\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta)}{\partial\boldsymbol{x}}$
$\frac{\partial\mathcal{L}}{\partial\boldsymbol{z}}\frac{\partial\boldsymbol{z}}{\partial\theta}\leftarrow\boldsymbol{y}\frac{\partial\boldsymbol{F}(\boldsymbol{z},\boldsymbol{x};\theta)}{\partial\theta}$

采样（down sampling）层中的批归一化层。此外，由于单层的 ImpFlow 由两个残差块组成，且每个残差块的输入和输出的维度相同，因此本节实验在 ResNet-18 的每个尺度（scale）中将下采样直连（downsampling shortcut）替换为恒等直连（identity shortcut），并在每个尺度的两个残差块后添加具有批归一化层的的下采样层（一个卷积层）。由此一来，每个尺度都由具有相同输入和输出维度的两个残差块组成。所有实验（ResNet、ResFlow、ImpFlow）使用的网络结构都基于上述网络结构（6.5M 参数），而不是原始的 ResNet-18 结构（11.2M 参数），唯一的区别只有是否使用谱归一化以及谱归一化时不同的超参数 c。

本节实验都基于最常用的设置：批大小（batch size）为 128，优化器（optimizer）

表 3.1　在不同的谱归一化超参数 c 下，原始 ResNet、ResFlow 和 ImpFlow 在 CIFAR-10 和 CIFAR-100 测试集上的分类错误率（%）

		原始 ResNet	$c = 0.99$	$c = 0.9$	$c = 0.8$	$c = 0.7$	$c = 0.6$
CIFAR-10	ResFlow	6.61	8.24	8.39	8.69	9.25	9.94
	ImpFlow		**7.29**	**7.41**	**7.94**	**8.44**	**9.22**
CIFAR-100	ResFlow	27.83	31.02	31.88	32.21	33.58	34.48
	ImpFlow		**29.06**	**30.47**	**31.40**	**32.64**	**34.17**

表 3.2　在表格数据集的测试集上的对数似然

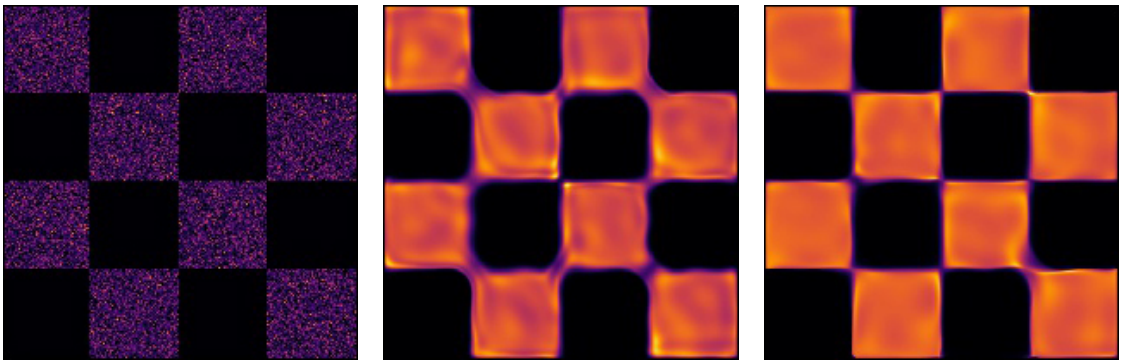
	POWER	GAS	HEPMASS	MINIBOONE	BSDS300
RealNVP[31]	0.17	8.33	-18.71	-13.55	153.28
FFJORD[37]	0.46	8.59	-14.92	-10.43	157.40
MAF[62]	0.24	10.08	-17.70	-11.75	155.69
NAF[122]	**0.62**	11.96	-15.09	**-8.86**	**157.73**
ImpFlow ($L = 20$)	0.61	**12.11**	**-13.95**	-13.32	155.68
ResFlow ($L = 10$)	0.26	6.20	-18.91	-21.81	104.63
ImpFlow ($L = 5$)	**0.30**	**6.94**	**-18.52**	**-21.50**	**113.72**

为 Adam 优化器[7]，学习率（learning rate）为 10^{-3}，没有权重衰减（weight decay），总训练轮数为 150。谱归一化的迭代使用 10^{-3} 的误差界限，与 Chen 等人[34]的设置相同。

与生成式建模相比，分类是评估函数族表达能力的直接方法，因为该类任务可以将函数拟合的能力与其它生成式建模所考虑的因素解耦合（例如对数行列式的估计等）。在以上所述的相同的设置下，本节比较了在 CIFAR-10 和 CIFAR-100[123]上训练的 ResFlow 和 ImpFlow，在测试集上的分类结果如表 3.1 所示。为了进一步探讨利普希兹常数约束的影响，实验中进一步改变不同的谱归一化超参数 c，以比较在相同的利普希兹常数上界下 ResFlow 和 ImpFlow 之间的差异。实验表明，在不同的 c 值下，ImpFlow 的分类结果始终优于同等参数的 ResFlow 的结果。这些结果在经验上验证了本节的主要理论结果，表明了 ImpFlow 的表达能力比 ResFlow 更强。此外，对于较大的 c，ImpFlow 在分类方面与没有利普希兹常数约束的原始的 ResNet 相当，这也进一步说明了 ImpFlow 打破了 ResFlow 的利普希兹常数的约束。

表 3.3　在不同的谱归一化超参数 c 下，ResFlow 和 ImpFlow 在 CIFAR-10 测试集上平均的负对数似然（比特/维度）

	$c = 0.9$	$c = 0.8$	$c = 0.7$	$c = 0.6$
ResFlow ($L = 12$)	3.469	3.533	3.627	3.820
ImpFlow ($L = 6$)	**3.452**	**3.511**	**3.607**	**3.814**



(a) 数据分布（5.00 比特）　(b) ResFlow（5.08 比特）　(c) ImpFlow（5.05 比特）

图 3.3　残差标准化流模型和隐式标准化流模型在棋盘格数据上的密度估计结果

3.4.2　二维模拟数据的密度估计任务

对于密度建模任务，本节在棋盘格模拟数据上评估 ImpFlow 的建模能力。这种模拟数据的概率分布是多峰的，如图 3.3 (a) 所示。为了公平比较，实验遵循与 Chen 等人[34]相同的设置。具体而言，模型使用具有 128 个隐藏单元的全连接层的 4 层多层感知器（multilayer perceptron，MLP）。学习率为 10^{-3}，优化器为权重衰减为 10^{-5} 的 Adam 优化器。此外，实验发现 $\sin(2\pi \boldsymbol{x})/2\pi$ 是一种更好的激活函数，既能保证利普希兹常数为 1，又能避免隐函数的梯度消失，所以在 ResFlow 和 ImpFlow 的所有实验中都使用这种激活函数。模型没有使用任何归一化层。对数行列式项使用与 ResFlow 实验相同的解析计算方式[34]。谱归一化的超参数为 0.999，迭代次数为 20。训练时批大小为 5000，共进行 50000 次迭代。测试时批大小为 10000。

注意到这样的数据分布的支撑（support）是有界的，而模型希望拟合一个将这种数据映射到标准高斯分布的变换 $\boldsymbol{f}$，但标准高斯分布的支撑是无界的。一个完美的 $\boldsymbol{f}$ 需要一个足够大的 $\|J_{\boldsymbol{f}}(\boldsymbol{x})\|_2$ 来保证准确地映射高斯分布中那些远离原点的点。因此，这样的 $\boldsymbol{f}$ 的利普希兹常数需要较大，故无法由 8 层的 ResFlow 拟合（参见图 3.3 (b)）。相反，一个 4 层的 ImpFlow 可以达到 5.05 比特的结果，这优于具有相同参数数量的 8 层 ResFlow 的 5.08 比特。这样的实验结果也与定理 3.2 的理论结果一致。

图 3.4　在 64×64 分辨率的 5 比特 CelebA 数据集上训练的隐式标准化流模型的采样结果

3.4.3　真实数据的密度估计任务

为了进一步验证 ImpFlow 的建模能力，本节在一些真实数据的密度估计任务中进行测试，包括表格数据集[62,124]、CIFAR-10[125]和 5 比特的 64×64 分辨率的 CelebA[32]。所有实验中，对数行列式的无偏估计器都使用 $p = 0.5$ 的几何分布作为 $p(N)$。

对于表格数据，实验在五个数据集上测试了 ImpFlow 的性能，分别为：POWER（$d = 6$），GAS（$d = 8$），HEPMASS（$d = 21$），MINIBOONE（$d = 43$）和 BSDS300（$d = 63$），其中 d 是数据维数。实验使用与 Papamakarios 等人[62]相同的数据预处理，以及与 Chen 等人[34]在二维模拟数据上相同的实验设置，包括 1000 的批大小（训练和测试）和具有 10^{-3} 的学习率 Adam 优化器。残差块是具有 128 个隐藏单元的 4 层 MLP。为了确保参数量相同，ResFlow 使用 10 个残差块，ImpFlow 使用 5 个残差块。此外，20 层的 ImpFlow 可以得到与前人工作可比甚至更好的实验结果。实验使用的激活函数与二维模拟数据的实验相同，即 $\sin(2\pi \boldsymbol{x})/2\pi$，且没有使用任何归一化层。谱归一化的超参数 $c = 0.9$，迭代误差界是 10^{-3}。表 3.2 列出了 ResFlow 和 ImpFlow 在测试集上的平均对数似然。ImpFlow 在所有数据集上都

比 ResFlow 有更好的密度估计结果，这再次证明了 ImpFlow 的有效性。

对于 CIFAR-10 数据集，实验遵循与 Chen 等人[34]相同的设置和架构，其中隐藏通道为 512，批大小为 64，优化器为学习率是 10^{-3} 的 Adam 优化器。谱归一化的迭代误差上限是 10^{-3}。表 3.3 展示了负对数似然（negative log-likelihood，NLL）的结果，单位为比特/维度（bits/dim）。给定不同的谱归一化超参数 c，ImpFlow（6 层）的密度估计结果一致地优于同参数量的 ResFlow（12 层）。

对于 CelebA 数据集，实验遵循与 Chen 等人[34]中 ResFlow 的最终版本完全相同的设置，除了激活函数为 $\sin(2\pi\boldsymbol{x})/2\pi$。ImpFlow 的采样结果如图 3.4 所示。

3.5　定理证明

3.5.1　引理 3.1 的证明

证明（引理 3.1）$\forall \boldsymbol{f} \in \mathcal{R}$，有

$$\sup_{\boldsymbol{x}\in\mathbb{R}^d} \|J_{\boldsymbol{f}}(\boldsymbol{x}) - I\|_2^2 < 1, \tag{3.26}$$

根据矩阵 L_2 范数的定义，上述不等式等价于

$$\sup_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \|(J_{\boldsymbol{f}}(\boldsymbol{x}) - I)\boldsymbol{v}\|_2^2 < 1, \tag{3.27}$$

对不等式左边展开，等价于

$$\sup_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}^T(\boldsymbol{x}) J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} - 2\boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} < 0 \tag{3.28}$$

由于 $\boldsymbol{f}$ 是可逆映射，$J_{\boldsymbol{f}}(\boldsymbol{x})$ 是可逆矩阵，所以对于 $\forall \boldsymbol{x}, \boldsymbol{v} \in \mathbb{R}^d, \|\boldsymbol{v}\|_2 = 1$，有 $\boldsymbol{v}^T J_{\boldsymbol{f}}^T(\boldsymbol{x}) J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} > 0$。因此

$$0 > \sup_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}^T(\boldsymbol{x}) J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} - 2\boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} \geqslant \sup_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} -2\boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}, \tag{3.29}$$

故

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} > 0. \tag{3.30}$$

由于 $\boldsymbol{f}$ 的任意性，故有 $\mathcal{R} \subset \mathcal{F}$。另一方面，令 $\boldsymbol{f}(\boldsymbol{x}) = m\boldsymbol{x}$，其中 $m > 2$，有

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} = \inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T (mI)\boldsymbol{v} = m \tag{3.31}$$

由于 m 可以任意大，而 $\mathcal{R}$ 中的函数的利普希兹常数不超过 2，因此 $\boldsymbol{f} \in \mathcal{F}$ 但 $\boldsymbol{f} \notin \mathcal{R}$，所以 $\mathcal{R} \subsetneq \mathcal{F}$。 ∎

3.5.2　定理 3.2 的证明

为了证明定理 3.2，以下先证明几个必要的引理。

引理 3.2： $\forall \boldsymbol{f} \in \mathcal{D}$，若 $\boldsymbol{f} \in \mathcal{F}$，则 $\boldsymbol{f}^{-1} \in \mathcal{F}$。

证明 首先，

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}^{-1}}(\boldsymbol{x})\boldsymbol{v}$$

（根据*逆函数定理*（inverse function theorem））

$$= \inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}^{-1}(\boldsymbol{f}^{-1}(\boldsymbol{x}))\boldsymbol{v} \tag{3.32}$$

（由于 $\boldsymbol{f}$ 是从 $\mathbb{R}^d$ 到 $\mathbb{R}^d$ 的）

$$= \inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}^{-1}(\boldsymbol{x})\boldsymbol{v}$$

令 $\boldsymbol{u} = J_{\boldsymbol{f}}^{-1}(\boldsymbol{x})\boldsymbol{v}$，$\boldsymbol{v}_0 = \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|_2}$，有 $\|\boldsymbol{v}_0\|_2 = 1$，且

$$\boldsymbol{v}^T J_{\boldsymbol{f}}^{-1}(\boldsymbol{x})\boldsymbol{v} = \boldsymbol{u}^T J_{\boldsymbol{f}}^T(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{u}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{u} = \|\boldsymbol{u}\|_2^2 \boldsymbol{v}_0^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}_0. \tag{3.33}$$

其中，第二个等式是由于如下结论：对于任意 $d \times d$ 维的实矩阵 A，$\forall \boldsymbol{x} \in \mathbb{R}^d$，$\boldsymbol{x}^T A\boldsymbol{x} = (\boldsymbol{x}^T A\boldsymbol{x})^T = \boldsymbol{x}^T A^T \boldsymbol{x}$（因为 $\boldsymbol{x}^T A\boldsymbol{x} \in \mathbb{R}$）。

另一方面，由于 $\boldsymbol{f}$ 是利普希兹连续函数，所以 $\|J_{\boldsymbol{f}}(\boldsymbol{x})\|_2 \leqslant \mathrm{Lip}(\boldsymbol{f})$，故

$$1 = \|\boldsymbol{v}\|_2 \leqslant \|J_{\boldsymbol{f}}(\boldsymbol{x})\|_2 \|\boldsymbol{u}\|_2 \leqslant \mathrm{Lip}(\boldsymbol{f})\|\boldsymbol{u}\|_2, \tag{3.34}$$

这意味着

$$\|\boldsymbol{u}\|_2 \geqslant \frac{1}{\mathrm{Lip}(\boldsymbol{f})}. \tag{3.35}$$

代入式（3.32）和式（3.33），有

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}^{-1}}(\boldsymbol{x})\boldsymbol{v} \geqslant \frac{1}{\mathrm{Lip}(\boldsymbol{f})^2} \inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}_0\in\mathbb{R}^d,\|\boldsymbol{v}_0\|_2=1} \boldsymbol{v}_0^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}_0 > 0. \tag{3.36}$$

因此 $\boldsymbol{f}^{-1} \in \mathcal{F}$。 ∎

引理 3.3： $\forall \boldsymbol{f} \in \mathcal{R}$，有 $\boldsymbol{f}^{-1} \in \mathcal{F}$。

证明 首先，根据引理 3.1，有

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} > 0. \tag{3.37}$$

而由于 $\boldsymbol{f} \in \mathcal{R} \subset \mathcal{D}$，有 $\boldsymbol{f}^{-1} \in \mathcal{D}$，所以根据引理 3.2，有

$$\inf_{\boldsymbol{x}\in\mathbb{R}^d,\boldsymbol{v}\in\mathbb{R}^d,\|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T J_{\boldsymbol{f}^{-1}}(\boldsymbol{x})\boldsymbol{v} > 0. \tag{3.38}$$

因此 $\boldsymbol{f}^{-1} \in \mathcal{F}$。 ∎

引理 3.4： $\forall \boldsymbol{f} \in \mathcal{F}$，$\exists \alpha_0 > 0$，使得 $\forall\, 0 < \alpha < \alpha_0$，有 $\alpha \boldsymbol{f} \in \mathcal{R}$。

证明 由于 $\boldsymbol{f}$ 是利普希兹连续函数，有 $\mathrm{Lip}(\boldsymbol{f}) = \sup_{\boldsymbol{x}\in\mathbb{R}^d} \|\boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\|_2 > 0$。令

$$\beta \coloneqq \inf_{\boldsymbol{x}\in\mathbb{R}^d, \boldsymbol{v}\in\mathbb{R}^d, \|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}, \tag{3.39}$$

那么根据 $\mathcal{F}$ 的定义，有 $\beta > 0$。令

$$\alpha_0 \coloneqq \frac{\beta}{\mathrm{Lip}(\boldsymbol{f})^2}, \tag{3.40}$$

那么 $\alpha_0 > 0$。对于 $\forall\, 0 < \alpha < \alpha_0$，有

$$\sup_{\boldsymbol{x}\in\mathbb{R}^d} \|\alpha \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x}) - \boldsymbol{I}\|_2^2$$

（基于矩阵 L_2 范数的定义而展开）

$$= \sup_{\boldsymbol{x}\in\mathbb{R}^d, \boldsymbol{v}\in\mathbb{R}^d, \|\boldsymbol{v}\|_2=1} \boldsymbol{v}^T (\alpha \boldsymbol{J}_{\boldsymbol{f}}^T(\boldsymbol{x}) - \boldsymbol{I})(\alpha \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x}) - \boldsymbol{I})\boldsymbol{v}$$

（展开左式，且基于如下结论合并同类项：对于任意 $d \times d$ 维的实矩阵 $\boldsymbol{A}$，）

（$\forall \boldsymbol{x} \in \mathbb{R}^d$，$\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = (\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x})^T = \boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{x}$（因为 $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \in \mathbb{R}$））

$$= 1 + \sup_{\boldsymbol{x}\in\mathbb{R}^d, \boldsymbol{v}\in\mathbb{R}^d, \|\boldsymbol{v}\|_2=1} \left(\alpha^2 \boldsymbol{v}^T \boldsymbol{J}_{\boldsymbol{f}}^T(\boldsymbol{x}) \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v} - 2\alpha \boldsymbol{v}^T \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}\right)$$

（利用 sup 的不等式放缩）

$$= 1 + \alpha^2 \sup_{\boldsymbol{x}\in\mathbb{R}^d, \boldsymbol{v}\in\mathbb{R}^d, \|\boldsymbol{v}\|_2=1} \left(\boldsymbol{v}^T \boldsymbol{J}_{\boldsymbol{f}}^T(\boldsymbol{x}) \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}\right) - 2\alpha \inf_{\boldsymbol{x}\in\mathbb{R}^d, \boldsymbol{v}\in\mathbb{R}^d, \|\boldsymbol{v}\|_2=1} \left(\boldsymbol{v}^T \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x})\boldsymbol{v}\right)$$

（基于矩阵 L_2 范数的定义和利普希兹常数的定义）

$$= 1 + \alpha(\alpha \mathrm{Lip}(\boldsymbol{f})^2 - 2\beta)$$

（由于 $\alpha < \alpha_0 = \dfrac{\beta}{\mathrm{Lip}(\boldsymbol{f})^2}$）

$$< 1 - \alpha\beta < 1.$$

因此 $\alpha\boldsymbol{f} \in \mathcal{R}$。 ∎

基于以上引理，以下给出定理 3.2 的具体证明。

证明 (定理 3.2) 令

$$\mathcal{P} \coloneqq \{\boldsymbol{f} \in \mathcal{D} \mid \exists \boldsymbol{f}_1, \boldsymbol{f}_2 \in \mathcal{F}, \text{使得 } \boldsymbol{f} = \boldsymbol{f}_2 \circ \boldsymbol{f}_1\}. \tag{3.41}$$

以下首先证明 $\mathcal{I} \subset \mathcal{P}$。对于 $\forall \boldsymbol{f} \in \mathcal{I}$，存在 $\boldsymbol{f}_1 \in \mathcal{R}$，$\boldsymbol{f}_2^{-1} \in \mathcal{R}$，使得 $\boldsymbol{f} = \boldsymbol{f}_2 \circ \boldsymbol{f}_1$。根据引理 3.1，有 $\boldsymbol{f}_1 \in \mathcal{F}$；而根据引理 3.3，有 $\boldsymbol{f}_2 \in \mathcal{F}$。因此，$\boldsymbol{f} \in \mathcal{P}$。由 $\boldsymbol{f}$ 的任意性，有 $\mathcal{I} \subset \mathcal{P}$。

接下来证明 $\mathcal{P} \subset \mathcal{I}$。对于 $\forall \boldsymbol{f} \in \mathcal{P}$，存在 $\boldsymbol{f}_1, \boldsymbol{f}_2 \in \mathcal{F}$，使得 $\boldsymbol{f} = \boldsymbol{f}_2 \circ \boldsymbol{f}_1$。根据引理 3.2，有 $\boldsymbol{f}_2^{-1} \in \mathcal{F}$。另一方面，根据引理 3.4，$\exists\, \alpha_1 > 0, \alpha_2 > 0$，使得对于

$\forall\, 0 < \alpha < \min\{\alpha_1, \alpha_2\}$，有 $\alpha \boldsymbol{f}_1 \in \mathcal{R}$，$\alpha \boldsymbol{f}_2^{-1} \in \mathcal{R}$。因此，可将 $\boldsymbol{f}$ 拆分为 $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{g}_2 \circ \boldsymbol{g}_1$，其中

$$\begin{aligned} \boldsymbol{g}_1(\boldsymbol{x}) &= \alpha \boldsymbol{f}_1(\boldsymbol{x}), \\ \boldsymbol{g}_2(\boldsymbol{x}) &= \boldsymbol{f}_2(\frac{\boldsymbol{x}}{\alpha}). \end{aligned} \tag{3.42}$$

此时 $\boldsymbol{g}_1 \in \mathcal{R}$，$\boldsymbol{g}_2 \in \mathcal{R}$，所以 $\boldsymbol{f} \in \mathcal{I}$。由 $\boldsymbol{f}$ 的任意性，有 $\mathcal{P} \subset \mathcal{I}$。

综上所述，$\mathcal{I} = \mathcal{P}$。 ∎

3.5.3　定理 3.3 的证明

以下首先给出一个关于双向利普希兹连续函数的引理。

引理 3.5： 若 $\boldsymbol{f} : (\mathbb{R}^d, \|\cdot\|) \to (\mathbb{R}^d, \|\cdot\|)$ 是双向利普希兹连续的，那么

$$\frac{1}{\mathrm{Lip}(\boldsymbol{f}^{-1})} \leqslant \frac{\|\boldsymbol{f}(\boldsymbol{x}_1) - \boldsymbol{f}(\boldsymbol{x}_2)\|}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|} \leqslant \mathrm{Lip}(\boldsymbol{f}),\ \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d, \boldsymbol{x}_1 \neq \boldsymbol{x}_2. \tag{3.43}$$

证明 对于 $\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d, \boldsymbol{x}_1 \neq \boldsymbol{x}_2$，有

$$\|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)\| \leqslant \mathrm{Lip}(\boldsymbol{f}) \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|, \tag{3.44}$$

且

$$\|\boldsymbol{x}_1 - \boldsymbol{x}_2\| = \|\boldsymbol{f}^{-1}(f(\boldsymbol{x}_1)) - \boldsymbol{f}^{-1}(f(\boldsymbol{x}_2))\| \leqslant \mathrm{Lip}(\boldsymbol{f}^{-1}) \|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)\|. \tag{3.45}$$

整理即可得所证结论。 ∎

引理 3.6： 对于 L 层的残差标准化流模型 $\boldsymbol{f} = \boldsymbol{f}_L \circ \cdots \circ \boldsymbol{f}_1$，其中每个函数 $\boldsymbol{f}_\ell$ 满足

$$\boldsymbol{f}_\ell(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{g}_\ell(\boldsymbol{x}),\ \mathrm{Lip}(\boldsymbol{g}_\ell) \leqslant \kappa < 1, \tag{3.46}$$

有

$$(1-\kappa)^L \leqslant \frac{\|\boldsymbol{f}(\boldsymbol{x}_1) - \boldsymbol{f}(\boldsymbol{x}_2)\|}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|} \leqslant (1+\kappa)^L,\ \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d, \boldsymbol{x}_1 \neq \boldsymbol{x}_2. \tag{3.47}$$

证明 根据 Behrmann 等人[33]的结论，有 $\mathrm{Lip}(\boldsymbol{f}_\ell^{-1}) \leqslant \frac{1}{1-\kappa}$。因此根据引理 3.5，有

$$1 - \kappa \leqslant \frac{\|\boldsymbol{f}_\ell(\boldsymbol{x}_1) - \boldsymbol{f}_\ell(\boldsymbol{x}_2)\|}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|} \leqslant 1 + \kappa < 2,\ \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d, \boldsymbol{x}_1 \neq \boldsymbol{x}_2. \tag{3.48}$$

将所有 $\ell = 1, \cdots, L$ 对应的不等式相乘，即可得到所证结论。 ∎

接下来正式给出定理 3.3 的证明。

证明 (定理 3.3)

一方面，根据 $\mathcal{P}(L, r)$ 的定义，有 $\mathcal{P}(L, r) \subset \mathcal{F} \subset \mathcal{I}$。

另一方面，对于 $\forall\, 0 < \ell < \log_2(L)$，有 $L - 2^\ell > 0$。对于 $\forall\, \boldsymbol{g} \in \mathcal{R}_\ell$，根据引

理 3.6，有

$$\|\boldsymbol{g}(\boldsymbol{x})-\boldsymbol{g}(\boldsymbol{y})\|_2 \leqslant 2^{\ell}\|\boldsymbol{x}-\boldsymbol{y}\|_2, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{B}_r. \tag{3.49}$$

因此，$\forall \boldsymbol{f} \in \mathcal{P}(L, r)$，$\forall\, \boldsymbol{x}_0 \in \mathcal{B}_r$，有

$$\begin{aligned}\|\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{g}(\boldsymbol{x})\|_2 &= \|\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{x}_0)+\boldsymbol{g}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x})+\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2 \\ &\quad \text{（三角不等式）} \\ &\geqslant \|f(\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{x}_0)\|_2-\|\boldsymbol{g}(\boldsymbol{x}_0)-g(\boldsymbol{x})\|_2-\|\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2 \\ &\quad \text{（根据 } \mathcal{P}(L, r) \text{ 的定义以及式（3.49））} \\ &\geqslant (L-2^{\ell})\|\boldsymbol{x}-\boldsymbol{x}_0\|_2-\|\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2.\end{aligned} \tag{3.50}$$

所以

$$\begin{aligned}\sup_{\boldsymbol{x}\in\mathcal{B}_r}\|\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{g}(\boldsymbol{x})\|_2 &\geqslant \sup_{\boldsymbol{x}\in\mathcal{B}_r}(L-2^{\ell})\|\boldsymbol{x}-\boldsymbol{x}_0\|_2-\|\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2 \\ &\geqslant (L-2^{\ell})r-\|\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2.\end{aligned} \tag{3.51}$$

注意到上述不等式对于任意 $\boldsymbol{x}_0 \in \mathcal{B}_r$ 都成立，所以有

$$\begin{aligned}\sup_{\boldsymbol{x}\in\mathcal{B}_r}\|\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{g}(\boldsymbol{x})\|_2 &\geqslant (L-2^{\ell})r-\inf_{\boldsymbol{x}_0\in\mathcal{B}_r}\|\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2 \\ &\geqslant (L-2^{\ell})r-\sup_{\boldsymbol{x}_0\in\mathcal{B}_r}\|\boldsymbol{f}(\boldsymbol{x}_0)-\boldsymbol{g}(\boldsymbol{x}_0)\|_2.\end{aligned} \tag{3.52}$$

由于 $\boldsymbol{x}$ 和 $\boldsymbol{x}_0$ 的任意性，不等式的左右两项可以合并，有

$$\sup_{\boldsymbol{x}\in\mathcal{B}_r}\|\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{g}(\boldsymbol{x})\|_2 \geqslant \frac{r}{2}(L-2^{\ell}), \forall \boldsymbol{g} \in \mathcal{R}_{\ell}. \tag{3.53}$$

由于 $\boldsymbol{g}$ 的任意性，有

$$\inf_{\boldsymbol{g}\in\mathcal{R}_{\ell}}\sup_{\boldsymbol{x}\in\mathcal{B}_r}\|\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{g}(\boldsymbol{x})\|_2 \geqslant \frac{r}{2}(L-2^{\ell})>0. \tag{3.54}$$

因此 $\boldsymbol{f} \notin \mathcal{R}_{\ell}$。又由于 $\boldsymbol{f}$ 的任意性，因此 $\mathcal{R}_{\ell} \cap \mathcal{P}(L, r)=\varnothing$。 ∎

3.5.4　第 3.3.3 节中结果的证明

首先给出式（3.21）的证明。

证明 (式（3.21）) 根据变量替换公式（change-of-variable formula），

$$\log p(\boldsymbol{x})=\log p(\boldsymbol{z})+\log\left|\det\left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right)\right|. \tag{3.55}$$

由于 $\boldsymbol{z}$ 和 $\boldsymbol{x}$ 之间的映射由如下方程定义：

$$\boldsymbol{F}(\boldsymbol{z}, \boldsymbol{x})=\boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x})-\boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z})+\boldsymbol{x}-\boldsymbol{z}=0, \tag{3.56}$$

根据隐函数定理（implicit function theorem），有

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = \boldsymbol{J}_{\boldsymbol{f}}(\boldsymbol{x}) = -[\boldsymbol{J}_{\boldsymbol{F},\boldsymbol{z}}(\boldsymbol{z})]^{-1}[\boldsymbol{J}_{\boldsymbol{F},\boldsymbol{x}}(\boldsymbol{x})] = (\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_z}(\boldsymbol{z}))^{-1}(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x})). \tag{3.57}$$

所以

$$\log|\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}| = \log|\det(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x}))| - \log|\det(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_z}(\boldsymbol{z}))|. \tag{3.58}$$

注意到 $\boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x})$ 的任何一个特征值 λ 都满足 $|\lambda| < \sigma(\boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x})) = \|\boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x})\|_2 < 1$（其中 $\sigma(\boldsymbol{J})$ 表示矩阵 $\boldsymbol{J}$ 的奇异值），所以 $\lambda \in (-1, 1)$，因此 $\det(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x})) > 0$。同理，$\det(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_z}(\boldsymbol{z})) > 0$。所以

$$\log|\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}| = \log\det(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_x}(\boldsymbol{x})) - \log\det(\boldsymbol{I} + \boldsymbol{J}_{\boldsymbol{g}_z}(\boldsymbol{z})). \tag{3.59}$$

∎

其次给出式（3.24）的证明。

证明 (式（3.24）) 对 $\boldsymbol{F}(\boldsymbol{z}, \boldsymbol{x}; \theta) = \boldsymbol{0}$ 两侧关于 $\boldsymbol{x}$ 求全微分（其中 $\boldsymbol{z}$ 是 $\boldsymbol{x}$ 的函数），有

$$\frac{\partial \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}; \theta)}{\partial \boldsymbol{x}} - \frac{\partial \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}; \theta)}{\partial \boldsymbol{z}}\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} + \boldsymbol{I} - \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = \boldsymbol{0}. \tag{3.60}$$

因此

$$\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} = \left(\boldsymbol{I} + \frac{\partial \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}; \theta)}{\partial \boldsymbol{z}}\right)^{-1}\left(\boldsymbol{I} + \frac{\partial \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}; \theta)}{\partial \boldsymbol{x}}\right) = \boldsymbol{J}_{\boldsymbol{G}}^{-1}(\boldsymbol{z})\frac{\partial \boldsymbol{F}(\boldsymbol{z}, \boldsymbol{x}; \theta)}{\partial \boldsymbol{x}}. \tag{3.61}$$

此外，对 $\boldsymbol{F}(\boldsymbol{z}, \boldsymbol{x}; \theta) = \boldsymbol{0}$ 两侧关于 θ 求全微分，有

$$\frac{\partial \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}; \theta)}{\partial \theta} - \frac{\partial \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}; \theta)}{\partial \theta} - \frac{\partial \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}; \theta)}{\partial \boldsymbol{z}}\frac{\partial \boldsymbol{z}}{\partial \theta} - \frac{\partial \boldsymbol{z}}{\partial \theta} = \boldsymbol{0}, \tag{3.62}$$

因此

$$\frac{\partial \boldsymbol{z}}{\partial \theta} = \left(\boldsymbol{I} + \frac{\partial \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}; \theta)}{\partial \boldsymbol{z}}\right)^{-1}\left(\frac{\partial \boldsymbol{g}_{\boldsymbol{x}}(\boldsymbol{x}; \theta)}{\partial \theta} - \frac{\partial \boldsymbol{g}_{\boldsymbol{z}}(\boldsymbol{z}; \theta)}{\partial \theta}\right) = \boldsymbol{J}_{\boldsymbol{G}}^{-1}(\boldsymbol{z})\frac{\partial \boldsymbol{F}(\boldsymbol{z}, \boldsymbol{x}; \theta)}{\partial \theta}. \tag{3.63}$$

因此，从 $\boldsymbol{z}$ 回传到 $(\cdot)$ 的梯度为

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{z}}\frac{\partial \boldsymbol{z}}{\partial (\cdot)} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{z}}\boldsymbol{J}_{\boldsymbol{G}}^{-1}(\boldsymbol{z})\frac{\partial \boldsymbol{F}(\boldsymbol{z}, \boldsymbol{x}; \theta)}{\partial (\cdot)}. \tag{3.64}$$

∎

3.6　本章小结

本章提出了隐式标准化流模型，通过方程的根隐式地定义可逆映射，从而推广了标准化流模型。隐式标准化流模型基于残差标准化流模型，在可计算性和表达能力之间达到了良好的平衡。理论分析表明，隐式标准化流模型的函数族比残差标准化流模型的函数族更丰富，特别是在建模具有较大的利普希兹常数的函数方面。此外，基于隐式微分公式，本章提出了一种可扩展的算法来训练和推断隐

式标准化流模型。实验结果表明，隐式标准化流模型在分类和密度建模基准测试上均优于同等参数量的残差标准化流模型。

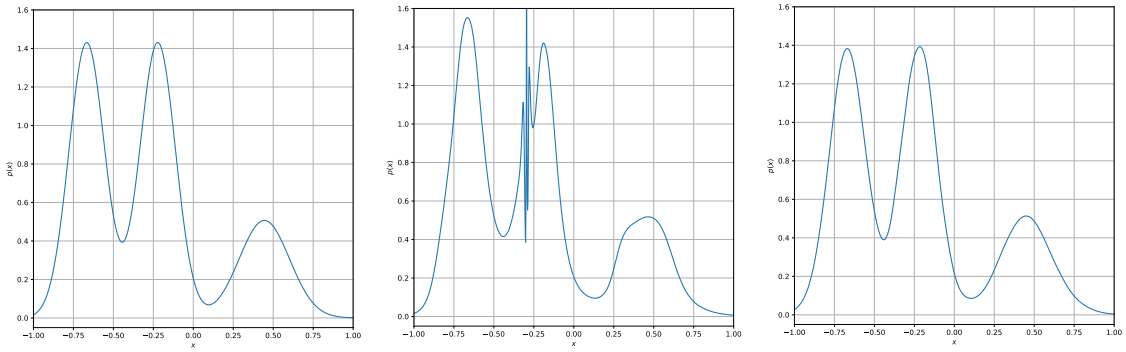
第 4 章　扩散常微分方程最大似然训练的高效算法

第 3 章提出了基于隐函数定义的隐式标准化流模型，旨在提高离散时间标准化流模型的表达能力。本章考虑另一类可逆生成模型，即连续时间标准化流模型，其中可逆映射由一个常微分方程定义。对于一般的连续时间标准化流模型，其训练的每一步迭代都需要基于数值求解器计算常微分方程的解，因而训练的开销较大。本章考虑连续时间标准化流模型的一种特殊参数化形式，称为扩散常微分方程，其采样质量和密度估计性能都具有出色的表现。在前人工作中，该类模型都基于一阶分数匹配（score matching）或等价的一阶去噪分数匹配（denoising score matching）进行训练，避免了昂贵的数值求解器开销。然而，分数匹配与扩散常微分方程的最大似然训练之间的关系仍然是一个未知的问题。本章提出了系统性的理论分析框架，彻底揭示了两者之间的关系，并提出了一种全新的高阶去噪分数匹配算法，以实现扩散常微分方程的高效最大似然训练。所提算法可以保证更高阶的匹配误差严格被训练误差和较低阶的匹配误差所控制，从而保证了高阶分数匹配的收敛性。实验结果表明，通过所提高阶去噪分数匹配算法，扩散常微分方程在模拟数据和真实数据都获得了最优的似然估计结果。

4.1　本章引言

连续时间标准化流模型[36-37]定义了从时间 $t=0$ 到时间 $t=T$（T 一般取 1）的常微分方程（ordinary differential equation，ODE），利用常微分方程的可逆性从而保证了整体映射是可逆的，因而也被称为神经常微分方程（neural ODE）。其中，$t=0$ 对应的分布为模型分布，而 $t=T$ 对应的分布为简单的已知分布（如标准高斯分布）。传统的神经常微分方程的最大似然训练旨在将 $t=0$ 的模型分布与真实数据分布匹配，并通过在每步优化中利用数值求解器估计训练数据的对数似然来训练模型。然而，这种训练方式只能约束 $t=0$ 的模型分布，但无法控制 $0<t<T$ 之间的模型分布。与此相反，扩散常微分方程（diffusion ODE）通过对不同时间 t 的分数匹配来匹配真实数据对应的概率流常微分方程（probability flow ODE）与模型的常微分方程，约束了所有时间 t 的分布和映射轨迹，因而均摊了不同时间 t 的训练开销，易于大规模训练。

在实践中，扩散常微分方程通过最小化一个对不同时间 t 加权求和的去噪分数匹配损失[106,126]来训练分数模型（score model）。Song 等人[105]从理论上证明，对于某个特定的加权函数，最小化对应的一阶去噪分数匹配损失等价于于最大化扩



(a) 数据分布　　(b) 扩散常微分方程（一阶去噪分数匹配）　　(c) 扩散常微分方程(三阶去噪分数匹配)

图 4.1　一阶和三阶去噪分数匹配对一维混合高斯分布的建模结果

散随机微分方程（diffusion SDE）的对数似然。然而，由于扩散随机微分方程与扩散常微分方程并不同，最小化该损失与最大化扩散常微分方程的似然的关系仍然是一个未知的问题。事实上，哪怕为了建模一个一维的简单混合高斯分布（mixture-of-Gaussians），用一阶去噪分数匹配训练的扩散常微分方程的似然估计仍然可能有非常大的误差，如图 4.1 所示。此外，Song 等人[105]发现基于一阶去噪分数匹配的扩散随机微分方程有最先进的采样质量，但相应的扩散常微分方程的对数似然（例如，CIFAR-10 数据集上约为 3.45 比特/维度）比其它可比模型（例如，Song 等人[45]的模型的结果约 3.13 比特/维度）差得多。这样的实验结果表明，p_0^{ODE} 的最大似然训练与分数匹配之间的关系尚未得到很好的理解。由于扩散常微分方程可以用于准确的似然估计，如何使用最大似然估计的准则来训练扩散常微分方程是一个未被解决的重要问题。

本章将系统地分析分数匹配和扩散常微分方程的最大似然训练之间的关系，并提出一种针对扩散常微分方程最大似然训练的高效算法。从理论上，本章给出了数据分布和扩散常微分方程分布之间的库尔贝克-莱布勒散度（Kullback-Leibler divergence，KL 散度）的等价形式①，并明确揭示了 KL 散度和原始的一阶分数匹配之间的差距。此外，为了避免耗时的 ODE 求解器[36]，本章进一步证明了可以通过约束第一、第二和第三阶分数匹配的训练误差来约束 KL 散度的上界，进而约束 KL 散度。由于通过最小化该上界得到的分数模型的最优解仍然是数据分布的分数函数，所以学到的分数模型仍然可以用于原始扩散模型（例如 Song 等人[45]提出的采样器）中的采样方法，用以生成高质量的样本。

基于这些分析，本章将进一步提出一种新的高阶去噪分数匹配算法来训练分数模型，该算法从理论上保证了高阶分数匹配的近似误差是有界的。实验结果表明，所提出的训练方法可以在保持高生成质量的同时，提高扩散常微分方程在不

① 基于最大似然估计的训练等价于基于最小化 KL 散度的训练。

同数据集上的似然估计准确度。

4.2　相关工作

扩散常微分方程是连续时间标准化流模型[36]的一种特殊形式。根据式（2.16），有

$$\log p_0^{\mathrm{ODE}}(\boldsymbol{x}_0) = \log p_T^{\mathrm{ODE}}(\boldsymbol{x}_T) + \int_0^T \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{h}_p(\boldsymbol{x}_t, t))\mathrm{d}t, \tag{4.1}$$

其中 $\boldsymbol{x}_t$ 满足 $\mathrm{d}\boldsymbol{x}_t/\mathrm{d}t = \boldsymbol{h}_p(\boldsymbol{x}_t, t)$。因此，最小化 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 等价于最大化

$$\mathbb{E}_{q_0(\boldsymbol{x}_0)}\left[\log p_0^{\mathrm{ODE}}(\boldsymbol{x}_0)\right] = \mathbb{E}_{q_0(\boldsymbol{x}_0)}\left[\log p_T^{\mathrm{ODE}}(\boldsymbol{x}_T) + \int_0^T \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{h}_p(\boldsymbol{x}_t, t))\mathrm{d}t\right]. \tag{4.2}$$

所有前人工作[36-37]对连续时间标准化流模型的最大似然训练本质上都基于上述表达式，且需要依赖 ODE 求解器来计算给定数据点 $\boldsymbol{x}_0$ 的 $\log p_0^{\mathrm{ODE}}(\boldsymbol{x}_0)$。然而，由于每一步优化都需要这样的计算，因此大型神经网络而言是难以扩展的。例如，Song 等人[45]的扩散常微分方程在评估单个数据点的 $\log p_0^{\mathrm{ODE}}$ 就需要花费 2 ∼ 3 分钟。此外，Finlay 等人[71]发现，通过简单地最大似然训练连续时间标准化流模型可能导致不必要的复杂轨迹，这不利于采样和推断。并且，直接针对 p_0^{ODE} 进行最大似然训练只对 $t = 0$ 时刻的分布进行了约束，因此不能确保模型分布 p_t^{ODE} 在每个时间 $t \in (0, T)$ 也与数据分布 q_t 相似，所以不能确保分数模型 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 的最优解是数据分数函数 $\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$，故不能确保模型 ODE 函数 $\boldsymbol{h}_p(\boldsymbol{x}_t, t)$ 与数据 ODE 函数 $\boldsymbol{h}_q(\boldsymbol{x}_t, t)$ 一致。

最近，Meng 等人[127]提出了一种用于估计数据分布的二阶分数函数的二阶去噪分数匹配方法，其需要同时训练一阶分数模型和二阶分数模型。然而，即使一阶分数匹配的误差很小且二阶分数匹配的训练误差为零，最小化 Meng 等人[127]中的目标函数所得到的最终二阶分数模型与真实的分数函数之间的匹配误差仍然可能是无界的，这并不利于更高阶分数模型的训练。

4.3　分数匹配与扩散常微分方程的 KL 散度之间的关系

准确的似然推断需要计算扩散常微分方程对应的 p_0^{ODE}，但这通常与扩散随机微分方程的分布 p_0^{SDE} 不同。尽管根据式（2.30），一阶分数匹配目标 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 可以限制扩散随机微分方程的 KL 散度上界，但这并不足以限制扩散常微分方程的 KL 散度上界。本节给出了一阶分数匹配目标 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 和扩散常微分方程的 KL 散度 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 之间的关系，并通过引入高阶分数匹配来限制 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 的上界。图 4.2 总结了本节的理论分析以及与前人工作的联系。

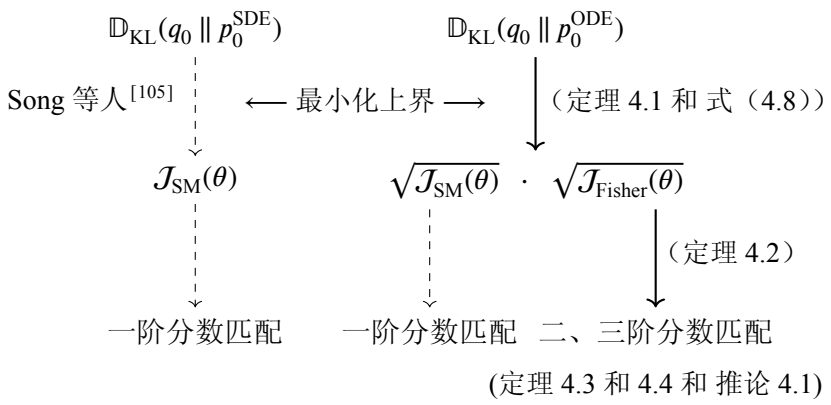


图 4.2　分数匹配目标函数和扩散模型与数据分布的 KL 散度之间的关系

以下首先给出扩散常微分方程的 KL 散度 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 与一阶分数匹配 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 之间的关系。如下述定理所示，二者之间存在严格的非零差距。

定理 4.1： 令 q_0 为数据分布，q_t 为式（2.18）中定义的正向扩散过程在时间 t 的边缘分布，p_t^{ODE} 为通过式（2.34）定义的扩散常微分方程在时间 t 的边缘分布，有

$$\begin{aligned}\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}}) &= \mathbb{D}_{\mathrm{KL}}(q_T \,\|\, p_T^{\mathrm{ODE}}) + \mathcal{J}_{\mathrm{ODE}}(\theta) \\ &= \underbrace{\mathbb{D}_{\mathrm{KL}}(q_T \,\|\, p_T^{\mathrm{ODE}}) + \mathcal{J}_{\mathrm{SM}}(\theta)}_{\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{SDE}}) \text{ 的上界}} + \mathcal{J}_{\mathrm{Diff}}(\theta),\end{aligned} \tag{4.3}$$

其中

$$\Delta_t(\boldsymbol{x}_t, \theta) \coloneqq \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t) \tag{4.4}$$

$$\mathcal{J}_{\mathrm{ODE}}(\theta) \coloneqq \frac{1}{2}\int_0^T g^2(t)\mathbb{E}_{q_t(\boldsymbol{x}_t)}\left[\Delta_t^{\top}(\boldsymbol{x}_t, \theta)\left(\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\right)\right]\mathrm{d}t, \tag{4.5}$$

$$\mathcal{J}_{\mathrm{Diff}}(\theta) \coloneqq \frac{1}{2}\int_0^T g^2(t)\mathbb{E}_{q_t(\boldsymbol{x}_t)}\left[\Delta_t^{\top}(\boldsymbol{x}_t, \theta)\left(\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) - \boldsymbol{s}_\theta(\boldsymbol{x}_t, t)\right)\right]\mathrm{d}t. \tag{4.6}$$

证明见第 4.7.1 节。

由于 p_T^{ODE} 是固定的，最小化 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 等价于最小化 $\mathcal{J}_{\mathrm{ODE}}(\theta)$，其中包括 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 和 $\mathcal{J}_{\mathrm{Diff}}(\theta)$。虽然最小化 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 可以减少 $\boldsymbol{s}_\theta(\cdot, t)$ 和数据分数函数 $\nabla_{\boldsymbol{x}_t} \log q_t(\cdot)$ 之间的差距，但它无法控制 $\mathcal{J}_{\mathrm{Diff}}(\theta)$ 的误差，该误差包括 $\boldsymbol{s}_\theta(\cdot, t)$ 和模型分数函数 $\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\cdot)$ 之间的差距。如图 4.1 所示，仅仅最小化 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 可能导致扩散常微分方程的模型密度出现严重的问题。

定理 4.1 中的 KL 散度包括模型分数函数 $\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)$，然而该函数计算开销很高。为了解决这个问题，以下通过引入高阶分数匹配导出 $\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}})$ 的上界，可以避免计算 $\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)$。令 $\mathbb{D}_{\mathrm{F}}(q \,\|\, p) \coloneqq \mathbb{E}_{q(\boldsymbol{x})}|\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log q(\boldsymbol{x})|_2^2$ 表示两个分布 q 和 p 之间的费舍尔散度（Fisher divergence，简记为 Fisher 散度）。

定义

$$\mathcal{J}_{\text{Fisher}}(\theta) \coloneqq \frac{1}{2}\int_0^T g(t)^2 \mathbb{D}_{\text{F}}(q_t \,\|\, p_t^{\text{ODE}})\mathrm{d}t. \tag{4.7}$$

对式（4.5）直接应用根据柯西-施瓦茨不等式（Cauchy-Schwarz inequality），有

$$\mathcal{J}_{\text{ODE}}(\theta) \leqslant \sqrt{\mathcal{J}_{\text{SM}}(\theta)} \cdot \sqrt{\mathcal{J}_{\text{Fisher}}(\theta)}. \tag{4.8}$$

因此，可以通过最小化 $\mathcal{J}_{\text{SM}}(\theta)$ 和 $\mathcal{J}_{\text{Fisher}}(\theta)$ 来最小化扩散常微分方程的 KL 散度 $\mathbb{D}_{\text{KL}}(q_0 \,\|\, p_0^{\text{ODE}})$。并且，当 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) \equiv \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$ 对所有 $\boldsymbol{x}_t \in \mathbb{R}^d$ 和 $t \in [0, T]$ 都成立时，上述不等式取等号，且有 $\mathcal{J}_{\text{ODE}}(\theta) = \mathcal{J}_{\text{SM}}(\theta) = 0$。然而，直接计算并最小化 $\mathcal{J}_{\text{Fisher}}(\theta)$ 仍然较为困难。以下进一步引入高阶分数匹配来限制 $\mathbb{D}_{\text{F}}(q_t \,\|\, p_t^{\text{ODE}})$，避免 ODE 求解器的计算成本。

定理 4.2： 假设存在 $C > 0$ 使得对于任意 $t \in [0, T]$ 和 $\boldsymbol{x}_t \in \mathbb{R}^d$，都有 $\|\nabla^2_{\boldsymbol{x}_t} \log p_t^{\text{ODE}}(\boldsymbol{x}_t)\|_2 \leqslant C$（其中 ∇^2 表示二阶导（即海森矩阵（Hessian matrix）），$\|\cdot\|_2$ 表示矩阵的谱范数），假设存在 δ_1、δ_2、$\delta_3 > 0$ 使得对于任意 $\boldsymbol{x}_t \in \mathbb{R}^d$ 和 $t \in [0, T]$，分数模型 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 满足

$$\begin{aligned}
&\|\boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\|_2 \leqslant \delta_1, \\
&\|\nabla_{\boldsymbol{x}_t} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla^2_{\boldsymbol{x}} \log q_t(\boldsymbol{x}_t)\|_F \leqslant \delta_2, \\
&\|\nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t)) - \nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla^2_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t))\|_2 \leqslant \delta_3,
\end{aligned} \tag{4.9}$$

其中 $\|\cdot\|_F$ 表示矩阵的弗罗贝尼乌斯范数（Frobenius norm，简记为 Frobenius 范数）。那么，存在与 θ 无关的 $U(t; \delta_1, \delta_2, \delta_3, \boldsymbol{C}, q) \geqslant 0$，使得 $\mathbb{D}_{\text{F}}(q_t \,\|\, p_t^{\text{ODE}}) \leqslant U(t; \delta_1, \delta_2, \delta_3, \boldsymbol{C}, q)$。并且，对于任意 $t \in [0, T]$，若 $g(t) \neq 0$，则 $U(t; \delta_1, \delta_2, \delta_3, \boldsymbol{C}, q)$ 是关于 δ_1、δ_2 和 δ_3 的严格递增函数。

证明见第 4.7.2 节。

定理 4.2 表明，通过限制 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$、$\nabla_{\boldsymbol{x}_t} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 和 $\nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{s}_\theta(\boldsymbol{x}_t, t))$ 与 q_t 的一阶、二阶和三阶分数函数[①]之间的差距，可以限制 $\mathbb{D}_{\text{F}}(q_t \,\|\, p_t^{\text{ODE}})$ 的上界从而限制 $\mathcal{J}_{\text{Fisher}}(\theta)$。由于 $\mathcal{J}_{\text{SM}}(\theta)$ 也可以由 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 的一阶分数匹配误差来限制，故可以通过一阶、二阶和三阶分数匹配来限制 $\mathbb{D}_{\text{KL}}(q_0 \,\|\, p_0^{\text{ODE}})$ 的上界。

需要注意的是，$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 的高阶分数匹配与传统的扩散模型是兼容的，因为在非参的情况下，$\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 的最优解仍然是 $\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$。因此，通过高阶分数匹配最小化 $\mathbb{D}_{\text{KL}}(q_0 \,\|\, p_0^{\text{ODE}})$ 之后，模型仍然可以使用 $\boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$ 作为扩散模型进而采样。相反，通过传统的 ODE 求解器计算 $\log p_0^{\text{ODE}}$ 从而直接最小化 $\mathbb{D}_{\text{KL}}(q_0 \,\|\, p_0^{\text{ODE}})$ 的方法却不能确保这一点。

① 本章中，称 $\nabla^2_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$ 为 q_t 的二阶分数函数，称 $\nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla^2_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t))$ 为 q_t 的三阶分数函数。

4.4　误差有界的高阶去噪分数匹配

第 4.3 节中的分析表明，可以通过控制分数模型 $\boldsymbol{s}_\theta(\cdot,t)$ 在每个时间 $t\in[0,T]$ 的一阶、二阶和三阶分数匹配误差（与其相应的导数）来限制扩散常微分方程的 KL 散度 $\mathbb{D}_{\mathrm{KL}}(q_0\,\|\,p_0^{\mathrm{ODE}})$ 的上界。受此启发，本节将传统的一阶去噪分数匹配[106]推广到二阶和三阶，以最小化高阶分数匹配误差。具体而言，本节提出一种新颖的高阶去噪分数匹配方法，使得高阶分数匹配误差可以被其训练误差和低阶分数匹配误差来限制，从而保证了高阶误差分数匹配的误差可控。

不失一般性，本节只对某个固定的时间 $t\in[0,T]$ 进行讨论，而关于所有 $t\in[0,T]$ 的整体训练目标见第 4.5 节。此外，本节只考虑最常见的扩散模型的情形（详见第 2.2 节），即存在 $\alpha_t\in\mathbb{R}^+$ 和 $\sigma_t\in\mathbb{R}^+$，使得正向过程满足 $q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)=\mathcal{N}(\boldsymbol{x}_t|\alpha_t\boldsymbol{x}_0,\sigma_t^2\boldsymbol{I})$。

4.4.1　一阶去噪分数匹配

本节介绍一些一阶去噪分数匹配的背景知识[45,106]。

一阶分数匹配目标函数 $\mathcal{J}_{\mathrm{SM}}(\theta)$ 需要计算所有 $t\in[0,T]$ 的数据分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$，但它在实际情况中是未知的。为了解决这个问题，扩散模型利用去噪分数匹配方法[106]来训练分数模型。对于固定的时间 t，由 θ 参数化的一阶分数模型 $\boldsymbol{s}_1(\cdot,t;\theta):\mathbb{R}^d\to\mathbb{R}^d$ 对应的一阶分数匹配为：

$$\mathbb{E}_{q_t(\boldsymbol{x}_t)}\left[\left\|\boldsymbol{s}_1(\boldsymbol{x}_t,t;\theta)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\right\|_2^2\right]. \tag{4.10}$$

这可由如下的等价目标函数（一阶去噪分数匹配）进行优化：

$$\theta^*=\operatorname*{argmin}_\theta\mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{\sigma_t^2}\left\|\sigma_t\boldsymbol{s}_1(\boldsymbol{x}_t,t;\theta)+\boldsymbol{\epsilon}\right\|_2^2\right], \tag{4.11}$$

其中 $\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ 以及 $\boldsymbol{x}_t=\alpha_t\boldsymbol{x}_0+\sigma_t\boldsymbol{\epsilon}$。

4.4.2　高阶去噪分数匹配

本节将一阶去噪分数匹配推广到二阶和三阶，以匹配在定理 4.2 中定义的二阶和三阶分数函数。首先介绍如下的二阶方法。

定理 4.3 (误差有界的二阶去噪分数匹配)： 假设 $\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t)$ 是对一阶数据分数函数 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)$ 的估计，$\boldsymbol{s}_2(\cdot,t;\theta):\mathbb{R}^d\to\mathbb{R}^{d\times d}$ 是由 θ 参数化的二阶分数模型，对应的二阶分数匹配为

$$\mathbb{E}_{q_t(\boldsymbol{x}_t)}\left[\left\|\boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta)-\nabla_{\boldsymbol{x}}^2\log q_t(\boldsymbol{x}_t)\right\|_F^2\right]. \tag{4.12}$$

那么，$\boldsymbol{s}_2(\cdot,t;\theta)$ 可由如下的等价目标函数（二阶去噪分数匹配）进行优化：

$$\theta^* = \operatorname*{argmin}_{\theta} \mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}} \left[\frac{1}{\sigma_t^4} \left\| \sigma_t^2 \boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta) + \boldsymbol{I} - \ell_1 \ell_1^\top \right\|_F^2 \right], \tag{4.13}$$

其中

$$\ell_1(\boldsymbol{\epsilon},\boldsymbol{x}_0,t) \coloneqq \sigma_t \hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t) + \boldsymbol{\epsilon}, \quad \boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \sigma_t \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0},\boldsymbol{I}). \tag{4.14}$$

更进一步，记一阶分数匹配的误差为 $\delta_1(\boldsymbol{x}_t,t) \coloneqq \|\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\|_2$，那么对于任意 $\boldsymbol{x}_t$ 和 θ，二阶分数模型 $\boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta)$ 对应的二阶分数匹配误差有如下上界：

$$\left\| \boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta) - \nabla_{\boldsymbol{x}_t}^2 \log q_t(\boldsymbol{x}_t) \right\|_F \leqslant \left\| \boldsymbol{s}_2(\boldsymbol{x}_t,t,\theta) - \boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta^*) \right\|_F + \delta_1^2(\boldsymbol{x}_t,t). \tag{4.15}$$

证明见第 4.7.3 节。

定理 4.3 说明，式（4.13）中的二阶去噪分数匹配目标是二阶分数匹配的有效替代，因为二阶分数模型 $\boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta)$ 和真实的二阶分数函数 $\nabla_{\boldsymbol{x}}^2 \log q_t(\boldsymbol{x}_t)$ 之间的差距可以被训练误差 $\|\boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta) - \boldsymbol{s}_2(\boldsymbol{x}_t,t;\theta^*)\|_F$ 和一阶分数匹配误差 $\delta_1(\boldsymbol{x}_t,t)$ 限制。需要注意的是，Meng 等人[127]提出的二阶去噪分数匹配方法并没有这种误差有界的性质。

此外，式（4.13）中的去噪分数匹配目标需要学习一个矩阵值函数 $\nabla_{\boldsymbol{x}}^2 \log q_t(\boldsymbol{x}_t)$，但有时只需要二阶分数函数的迹（trace）$\operatorname{tr}(\nabla_{\boldsymbol{x}}^2 \log q_t(\boldsymbol{x}_t))$。因此，以下提出一个只匹配二阶分数函数的迹的推论。

推论 4.1 (误差有界的二阶去噪分数迹匹配)： 假设 $\boldsymbol{s}_2^{\text{trace}}(\cdot,t;\theta) : \mathbb{R}^d \to \mathbb{R}$ 是由 θ 参数化的二阶分数迹模型，对应的二阶分数迹匹配为

$$\mathbb{E}_{q_t(\boldsymbol{x}_t)} \left[\left| \boldsymbol{s}_2^{\text{trace}}(\boldsymbol{x}_t,t;\theta) - \operatorname{tr}(\nabla_{\boldsymbol{x}_t}^2 \log q_t(\boldsymbol{x}_t)) \right|^2 \right]. \tag{4.16}$$

那么，$\boldsymbol{s}_2^{\text{trace}}(\cdot,t;\theta)$ 可由如下的等价目标函数（二阶去噪分数迹匹配）进行优化：

$$\theta^* = \operatorname*{argmin}_{\theta} \mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}} \left[\frac{1}{\sigma_t^4} \left| \sigma_t^2 \boldsymbol{s}_2^{\text{trace}}(\boldsymbol{x}_t,t;\theta) + d - \|\ell_1\|_2^2 \right|^2 \right]. \tag{4.17}$$

更进一步，二阶分数迹模型 $\boldsymbol{s}_2^{\text{trace}}(\boldsymbol{x}_t,t;\theta)$ 对应的二阶分数迹匹配误差有如下上界：

$$\left| \boldsymbol{s}_2^{\text{trace}}(\boldsymbol{x}_t,t;\theta) - \operatorname{tr}(\nabla_{\boldsymbol{x}}^2 \log q_t(\boldsymbol{x}_t)) \right| \leqslant \left| \boldsymbol{s}_2^{\text{trace}}(\boldsymbol{x}_t,t;\theta) - \boldsymbol{s}_2^{\text{trace}}(\boldsymbol{x}_t,t;\theta^*) \right| + \delta_1^2(\boldsymbol{x}_t,t). \tag{4.18}$$

最后，以下介绍三阶去噪分数匹配方法，其对应的三阶分数匹配误差可以由训练误差和一、二阶分数匹配误差共同限制。

定理 4.4 (误差有界的三阶去噪分数匹配)： 假设 $\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t)$ 是对一阶分数函数 $\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$ 的估计，$\hat{\boldsymbol{s}}_2(\boldsymbol{x}_t,t)$ 是对二阶分数函数 $\nabla_{\boldsymbol{x}}^2 \log q_t(\boldsymbol{x}_t)$ 的估计，$\boldsymbol{s}_3(\cdot,t;\theta)$ ：

$\mathbb{R}^d \to \mathbb{R}^d$ 是由 θ 参数化的三阶分数模型，对应的三阶分数匹配为

$$\mathbb{E}_{q_t(\boldsymbol{x}_t)}\left[\left\|\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)-\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\right\|_2^2\right]. \tag{4.19}$$

那么，$\boldsymbol{s}_3(\cdot,t;\theta)$ 可由如下的等价目标函数（三阶去噪分数匹配）进行优化：

$$\theta^* = \operatorname*{argmin}_\theta \mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\frac{1}{\sigma_t^6}\left\|\sigma_t^3\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)+\left(\|\ell_1\|_2^2\boldsymbol{I}-\mathrm{tr}(\ell_2)\boldsymbol{I}-2\ell_2\right)\ell_1\right\|_2^2\right], \tag{4.20}$$

其中

$$\begin{aligned}&\ell_1(\boldsymbol{\epsilon},\boldsymbol{x}_0,t)\coloneqq\sigma_t\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t)+\boldsymbol{\epsilon},\quad \ell_2(\boldsymbol{\epsilon},\boldsymbol{x}_0,t)\coloneqq\sigma_t^2\hat{\boldsymbol{s}}_2(\boldsymbol{x}_t,t)+\boldsymbol{I},\\&\boldsymbol{x}_t=\alpha_t\boldsymbol{x}_0+\sigma_t\boldsymbol{\epsilon},\quad \boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I}).\end{aligned} \tag{4.21}$$

更进一步，记一阶分数匹配的误差为 $\delta_1(\boldsymbol{x}_t,t)\coloneqq\|\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\|_2$，二阶分数匹配的误差为 $\delta_2(\boldsymbol{x}_t,t)\coloneqq\|\hat{\boldsymbol{s}}_2(\boldsymbol{x}_t,t)-\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\|_F$，二阶分数迹匹配的误差为 $\delta_{2,\mathrm{tr}}(\boldsymbol{x}_t,t)\coloneqq|\mathrm{tr}(\hat{\boldsymbol{s}}_2(\boldsymbol{x}_t,t))-\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))|$。那么对于任意 $\boldsymbol{x}_t$ 和 θ，三阶分数模型 $\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)$ 对应的三阶分数匹配误差有如下上界：

$$\left\|\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)-\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\right\|_2\leqslant\left\|\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)-\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta^*)\right\|_2+\left(\delta_1^2+\delta_{2,\mathrm{tr}}+2\delta_2\right)\delta_1^2. \tag{4.22}$$

定理 4.4 表明，三阶分数匹配需要一阶分数匹配、二阶分数匹配和二阶分数迹匹配才可以进行训练，并且三阶分数匹配的误差也可以由训练误差和低阶分数匹配的误差限制。

4.5　通过高阶去噪分数匹配训练分数模型

基于第 4.4 节提出的固定时间 t 的高阶去噪分数匹配算法，本节进一步提出针对所有时间 $t\in[0,T]$ 的高效算法，以用于扩散常微分方程的最大似然训练。

通过时间重加权的方差缩减。　理论上，训练所有 $t\in[0,T]$ 的分数函数需要对式 (4.11)、(4.13)、(4.17) 和 (4.20) 从 $t=0$ 积分到 $t=T$。这样的积分可通过蒙特卡洛（Monte Carlo）方法对 $t\in[0,T]$ 进行随机采样从而无偏地估计目标函数。然而，由于 $1/\sigma_t$ 的值在不同 t 下的波动很大，这导致目标函数的方差可能很大。为了减少方差，本节遵循一阶去噪分数匹配[44-45]中的方差缩减技术，分别对第一、第二、第三阶去噪分数匹配目标函数在每个时间 t 乘以 σ_t^2、σ_t^4、σ_t^6 来对时间进行重加权，这也被称为一阶去噪分数匹配的“噪声预测”（noise prediction）技巧[44,54]。

具体而言，假设 $\boldsymbol{x}_0\sim q_0(\boldsymbol{x}_0)$，时间采样的提议分布（proposal distribution）$p(t)=\mathcal{U}[0,T]$ 是 $[0,T]$ 上的均匀分布，随机噪声 $\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ 服从标准高斯分布，令 $\boldsymbol{x}_t=\alpha_t\boldsymbol{x}_0+\sigma_t\boldsymbol{\epsilon}$。将一阶去噪分数匹配的训练目标（式（4.11））对所有 $t\in[0,T]$ 加

权求和，即可得对应的一阶训练目标函数：

$$\mathcal{J}_{\mathrm{DSM}}^{(1)}(\theta) \coloneqq \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\left\|\sigma_t \boldsymbol{s}_\theta(\boldsymbol{x}_t,t)+\boldsymbol{\epsilon}\right\|_2^2\right]. \tag{4.23}$$

且前人工作[44-45]表明，上述目标函数的训练方差较低。

对于第二和第三阶去噪分数匹配目标，本节采取 $\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t) \coloneqq \boldsymbol{s}_\theta(\boldsymbol{x}_t,t)$ 作为一阶分数函数的估计，$\hat{\boldsymbol{s}}_2(\boldsymbol{x}_t,t) \coloneqq \nabla_{\boldsymbol{x}_t}\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)$ 作为二阶分数函数的估计，并且两者都禁用了对 θ 的梯度计算。禁用梯度是因为高阶去噪分数匹配只需要较低阶分数函数的估计值，如定理 4.3 和定理 4.4 所示。综合以上讨论，将二阶去噪分数匹配的训练目标（式（4.13））、二阶去噪分数迹匹配的训练目标（式（4.17））、三阶去噪分数匹配的训练目标（式（4.20））对所有 $t \in [0,T]$ 加权求和，即可得对应的训练目标函数（其中 ℓ_1,ℓ_2 的定义见式（4.21））：

$$\begin{aligned}
&\mathcal{J}_{\mathrm{DSM}}^{(2)}(\theta) \coloneqq \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\left\|\sigma_t^2 \nabla_{\boldsymbol{x}_t}\boldsymbol{s}_\theta(\boldsymbol{x}_t,t)+\boldsymbol{I}-\ell_1\ell_1^{\top}\right\|_F^2\right],\\
&\mathcal{J}_{\mathrm{DSM}}^{(2,\mathrm{tr})}(\theta) \coloneqq \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\left|\sigma_t^2 \mathrm{tr}(\nabla_{\boldsymbol{x}_t}\boldsymbol{s}_\theta(\boldsymbol{x}_t,t))+d-\|\ell_1\|_2^2\right|^2\right],\\
&\mathcal{J}_{\mathrm{DSM}}^{(3)}(\theta) \coloneqq \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\left\|\sigma_t^3 \nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla_{\boldsymbol{x}_t}\boldsymbol{s}_\theta(\boldsymbol{x}_t,t))+\left(\|\ell_1\|_2^2\boldsymbol{I}-\mathrm{tr}(\ell_2)\boldsymbol{I}-2\ell_2\right)\ell_1\right\|_2^2\right].
\end{aligned} \tag{4.24}$$

将各阶训练目标函数进行加权相加，最终的训练目标为：

$$\min_\theta \mathcal{J}_{\mathrm{DSM}}^{(1)}(\theta)+\lambda_1\left(\mathcal{J}_{\mathrm{DSM}}^{(2)}(\theta)+\mathcal{J}_{\mathrm{DSM}}^{(2,tr)}(\theta)\right)+\lambda_2\mathcal{J}_{\mathrm{DSM}}^{(3)}(\theta), \tag{4.25}$$

其中 $\lambda_1,\lambda_2 \geqslant 0$ 为超参数，具体的设置见第 4.6 节。

可扩展性和数值稳定性。　由于二阶和三阶去噪分数匹配目标需要计算神经网络的高阶导数，这对高维数据而言有很大的计算开销。因此，实践中使用*随机迹估计器*（即 Skilling-Hutchinson trace estimator）[119-120]对雅克比矩阵的迹[37]和雅克比矩阵的 Frobenius 范数[71]进行无偏估计。

此外，实践中当 t 接近 0 时存在数值不稳定的问题。因此，本章所有实验遵循 Song 等人[105]的设置，即先选择一个小的起始时间 $\epsilon > 0$，接着所有的训练和测试都是针对 $t \in [\epsilon,T]$ 而不是 $t \in [0,T]$ 进行的。需要注意的是，这样得到的似然计算仍然是精确的，因为这本质上使用了 $p_\epsilon^{\mathrm{ODE}}(\boldsymbol{x}_0)$ 来计算 $\boldsymbol{x}_0$ 的对数似然，它仍然是一个良好定义的概率模型[105]。

4.6　实验结果

本节将从实验角度验证所提出的高阶去噪分数匹配算法可以改善扩散常微分方程的似然估计性能，且同时保持相应扩散随机微分方程的高质量采样性能。具体而言，本节使用*方差爆炸*（variance exploding，VE）[45]类型的扩散模型。在通过

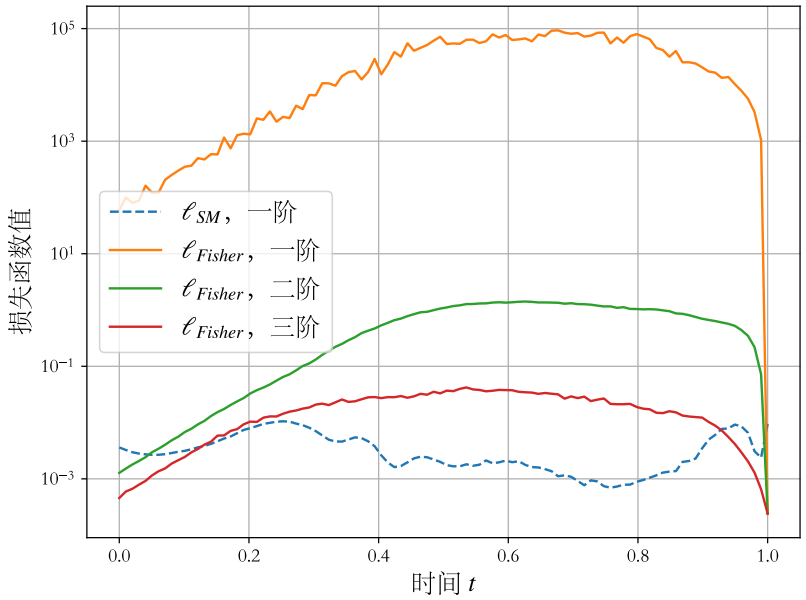


图 4.3　不同阶去噪分数匹配训练得到的扩散常微分方程对应的 $\ell_{\text{Fisher}}(t)$ 和 $\ell_{\text{SM}}(t)$

最小化一阶分数匹配目标 $\mathcal{J}_{\text{SM}}(\theta)$ 训练时，该模型可由扩散随机微分方程得到高质量采样，但由扩散常微分方程计算的数据似然却显著差于其它模型[105]。在所有实验中，起始时间 $\epsilon = 10^{-5}$，这遵循 Song 等人[45]中的默认设置。为简便起见，本节将“分数匹配”称为最小化当 $\lambda_1 = \lambda_2 = 0$ 时的式（4.25）对应的损失函数，将“二阶分数匹配”称为最小化 $\lambda_2 = 0$ 时的式（4.25）对应的损失函数。对于三阶分数匹配，λ_1 和 λ_2 的选取取决于实验数据。对于模拟数据（第 4.6.1、4.6.2 节），实验选择 $\lambda_1 = 0.5$ 和 $\lambda_2 = 0.1$；对于图像数据（第 4.6.3 节），实验选择 $\lambda_1 = \lambda_2 = 1$。

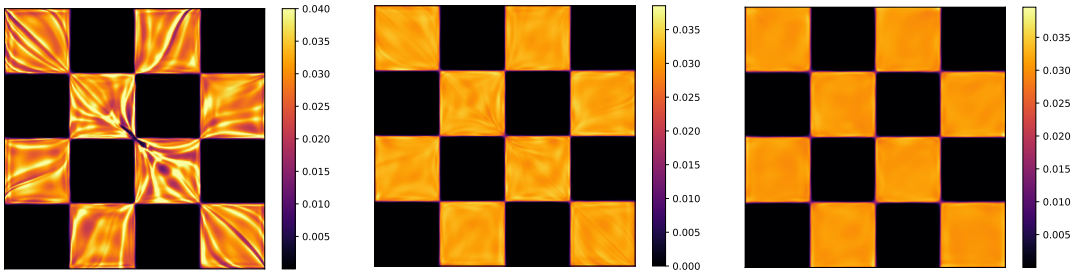
特别地，Song 等人[105]在最小化 $\mathcal{J}_{\text{SM}}(\theta)$ 时使用了另一个提议分布 $t \sim p(t)$ 来对不同时间进行重加权。由于本节的所有实验只考虑 VE 类型的扩散模型，它对应的提议分布恰好是式（4.25）中的 $p(t) = \mathcal{U}[0, T]$，因此本节的实验设置与 Song 等人[105]一致。

4.6.1　示例：一维混合高斯

本节以一个一维示例来说明高阶去噪分数匹配训练如何影响加权 Fisher 散度 $\mathcal{J}_{\text{Fisher}}(\theta)$ 和扩散常微分方程的模型密度。设 q_0 是如图 4.1 (a) 所示的一维混合高斯分布，其对应的密度函数为

$$q_0(\boldsymbol{x}_0) = 0.4 * \mathcal{N}\left(\boldsymbol{x}_0 \middle| -\frac{2}{9}, \frac{1}{9^2}\right) + 0.4 * \mathcal{N}\left(\boldsymbol{x}_0 \middle| -\frac{2}{3}, \frac{1}{9^2}\right) + 0.2 * \mathcal{N}\left(\boldsymbol{x}_0 \middle| \frac{4}{9}, \frac{2}{9^2}\right). \tag{4.26}$$

通过最小化一阶分数匹配 $\mathcal{J}_{\text{SM}}(\theta)$ 训练一个 VE 类型的分数模型，并使用相应的扩散常微分方程计算模型密度，结果如图 4.1 (b) 所示。由此可见，仅通过一阶分数匹配获得的模型密度在某些数据点上与数据分布相差甚远。相反，通过三阶分数



(a) 一阶去噪分数匹配　　(b) 二阶去噪分数匹配　　(c) 三阶去噪分数匹配

图 4.4　在二维棋盘格数据上使用不同阶去噪分数匹配训练得到的扩散常微分方程的模型概率密度

匹配，图 4.1 (c) 中的模型密度与数据分布高度相似，这验证了本章所提算法的有效性。

以下进一步分析一、二、三阶去噪分数匹配训练后模型的 $\mathcal{J}_{\text{Fisher}}(\theta)$ 的差异。由于 q_0 是混合高斯分布，可以证明每个 q_t 也是混合高斯分布，并且 $\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$ 可以解析地计算。令

$$\begin{aligned} \ell_{\text{Fisher}}(t) &\coloneqq \frac{1}{2} g(t)^2 \mathbb{D}_{\text{F}}(q_t \,\|\, p_t^{\text{ODE}}), \\ \ell_{\text{SM}}(t) &\coloneqq \frac{1}{2} g(t)^2 \mathbb{E}_{q_t(\boldsymbol{x}_t)} \| \boldsymbol{s}_\theta(\boldsymbol{x}_t, t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t) \|_2^2, \end{aligned} \tag{4.27}$$

分别对应了 $\mathcal{J}_{\text{SM}}(\theta)$ 和 $\mathcal{J}_{\text{Fisher}}(\theta)$ 中每个时间 t 的被积函数。图 4.3 展示了通过一、二、三阶去噪分数匹配训练后扩散常微分方程从 $t = \epsilon$ 到 $t = T$ 对应的 $\ell_{\text{Fisher}}(t)$ 和 $\ell_{\text{SM}}(t)$（每个时间取 100 个数据点进行平均）。如图 4.3 所示，尽管在最小化去噪分数匹配目标时 $\ell_{\text{SM}}(t)$ 较小，但大多数 t 的 $\ell_{\text{Fisher}}(t)$ 相当大，这意味着 $\mathcal{J}_{\text{Fisher}}(\theta)$ 也大。然而，二阶和三阶去噪分数匹配训练后的扩散常微分方程对应的 $\ell_{\text{Fisher}}(t)$ 在逐渐减小，这表明本章所提的高阶去噪分数匹配训练算法可以减小 $\ell_{\text{Fisher}}(t)$ 从而减小 $\mathcal{J}_{\text{Fisher}}(\theta)$。这些结果从经验上验证了控制高阶分数匹配误差可以控制模型对应的 Fisher 散度，正如定理 4.2 所述。

4.6.2　二维模拟数据的密度估计任务

本节在二维模拟数据（棋盘格数据）上通过一、二、三阶去噪分数匹配训练扩散常微分方程。实验使用 “噪声预测” 模型[54]，即使用神经网络 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ 来建模 $-\sigma_t \boldsymbol{s}_\theta(\boldsymbol{x}_t, t)$。其中，时间嵌入遵循 Song 等人[45]的设置，激活函数为 Swish[45,128]。模型先用两层多层感知机（multilayer perceptron，MLP）对 t 进行编码，并使用两层 MLP 对输入 $\boldsymbol{x}_t$ 进行编码，然后将它们拼接起来输入到另一个两层 MLP 网络，最终输出预测的噪声 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$。优化器为 Adam 优化器[7]，训练的批大小为 5000，共训练 50,000 次迭代。由于本章提出的高阶去噪分数匹配算法具有误差有界的特性，因此实验直接从默认初始化的神经网络开始训练，而不需要对较低阶模型进

表 4.1　CIFAR-10 上不同阶的去噪分数匹配对应的计算时间、训练迭代步数以及显存占用

算法阶数	每步迭代时间（秒）	显存（GB）	训练迭代步数
一阶	0.28	25.84	1200k
二阶	0.33	28.93	1200k（预训练模型）+ 100k
三阶	0.44	49.12	1200k（预训练模型）+ 100k

表 4.2　在 CIFAR-10 和 ImageNet 32×32 上，不同阶去噪分数匹配训练得到的负对数似然（比特/维度）和采样质量（FID 分数）结果

模型	CIFAR-10		ImageNet 32×32
	NLL ↓	FID ↓	NLL ↓
VE（一阶）	3.66	2.42	4.21
VE（二阶）	3.44	**2.37**	4.06
VE（三阶）	**3.38**	2.95	**4.04**
VE-Deep（一阶）	3.45	**2.19**	4.21
VE-Deep（二阶）	3.35	2.43	4.05
VE-Deep（三阶）	**3.27**	2.61	**4.03**

行任何预训练。

如图 4.4 所示，通过一阶去噪分数匹配得到的模型密度相当差，因为它无法控制式（4.6）中 $\mathcal{J}_{\mathrm{Diff}}(\theta)$ 对应的 $\nabla_{\boldsymbol{x}_t}\log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)$ 的值。相反，二阶去噪分数匹配可以控制 $\nabla_{\boldsymbol{x}_t}\log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)$ 的部分值并改善模型密度，而三阶去噪分数匹配可以减小 Fisher 散度 $\mathbb{D}_{\mathrm{F}}(q_t \,\|\, p_t^{\mathrm{ODE}})$ 从而得到与数据分布相当一致的模型密度。这些结果验证了本章所提的误差有界的高阶去噪分数匹配的有效性，并与定理 4.1 和定理 4.2 中的理论分析一致。

4.6.3　图像数据的密度估计任务

本节进一步在 CIFAR-10 数据集[125]和 ImageNet 32×32 数据集[9]上用不同阶的去噪分数匹配算法训练扩散常微分方程。其中，数据似然由扩散常微分方程对应的 $p_\epsilon^{\mathrm{ODE}}$ 进行计算，而计算 FID 分数[129]需要的 50,000 个样本由预测-修正采样器（predictor-corrector sampler）[45]从扩散随机微分方程中获得。

所有实验基于 Song 等人[45]的浅层 1200k 次迭代得到的预训练模型（对应 VE）和深层 600k 次迭代得到的预训练模型（对应 VE-Deep），并进行少量步数（100k）的高阶去噪分数匹配微调。训练遵循与 Song 等人[45]相同的设置，并将指数移动

平均（exponential moving average，EMA）率设置为默认的 0.999。输入图像被预处理归一化到 $[0,1]$ 以用于 VE 类型的模型。微调阶段，二阶去噪分数匹配的批大小为 128，三阶去噪分数匹配的批大小为 48。表 4.1 中列出了在 8 个 NVIDIA GeForce RTX 2080 Ti GPU 卡上（批大小为 48）的 VE 模型（浅模型）的计算时间和峰值显存消耗，其中一阶算法的结果直接采用了 Song 等人[45]的预训练模型。可以发现，二阶去噪分数匹配训练的成本接近一阶去噪分数匹配，而三阶去噪分数匹配训练的成本小于一阶去噪分数匹配训练成本的两倍。因此，本章所提方法在可接受的额外开销下易于扩展到高维数据。

表 4.2 展示了负对数似然（negative log-likelihood，NLL）和 FID 分数的结果，其中 NLL 的单位为比特/维度（bits/dim）。如表中结果所示，本章提出的高阶去噪分数匹配可以改善扩散常微分方程的似然估计性能，且同时保持相应扩散随机微分方程的高采样质量。

4.7　定理证明

本节给出主要定理的具体证明。

4.7.1　定理 4.1 的证明

证明（定理 4.1）根据牛顿-莱布尼茨公式（Newton-Leibniz formula），q_0 与 p_0^{ODE} 之间的 KL 散度可以改写为如下积分形式：

$$\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}}) = \mathbb{D}_{\mathrm{KL}}(q_T \,\|\, p_T^{\mathrm{ODE}}) - \int_0^T \frac{\partial \mathbb{D}_{\mathrm{KL}}(q_t \,\|\, p_t^{\mathrm{ODE}})}{\partial t} \mathrm{d}t. \tag{4.28}$$

给定数据点 $\boldsymbol{x}$，根据福克-普朗克方程（Fokker-Planck equation）[130]，ODE 定义的概率密度函数 q_t 和 p_t^{ODE} 随着时间 t 演化的方程分别为：

$$\frac{\partial q_t(\boldsymbol{x})}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot (\boldsymbol{h}_q(\boldsymbol{x}, t) q_t(\boldsymbol{x})), \quad \frac{\partial p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot (\boldsymbol{h}_p(\boldsymbol{x}, t) p_t^{\mathrm{ODE}}(\boldsymbol{x})). \tag{4.29}$$

因此，将 $\mathbb{D}_{\mathrm{KL}}(q_t \,\|\, p_t^{\mathrm{ODE}})$ 关于时间 t 的导数展开如下：

$$\frac{\partial \mathbb{D}_{\mathrm{KL}}(q_t \,\|\, p_t^{\mathrm{ODE}})}{\partial t} = \frac{\partial}{\partial t} \int q_t(\boldsymbol{x}) \log \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x}$$

（将求导展开）

$$= \int \frac{\partial q_t(\boldsymbol{x})}{\partial t} \log \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x} + \frac{\partial}{\partial t} \int q_t(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} - \int \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \frac{\partial p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} \mathrm{d}\boldsymbol{x}$$

（将 $\dfrac{\partial q_t(\boldsymbol{x})}{\partial t}$ 和 $\dfrac{\partial p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t}$ 代入；且由于 $\displaystyle\int q_t(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = 1$，因此该项求导为 0）

$$= -\int \nabla_{\boldsymbol{x}} \cdot (\boldsymbol{h}_q(\boldsymbol{x},t)q_t(\boldsymbol{x})) \log \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x} + \int \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \nabla_{\boldsymbol{x}} \cdot (\boldsymbol{h}_p(\boldsymbol{x},t)p_t^{\mathrm{ODE}}(\boldsymbol{x})) \mathrm{d}\boldsymbol{x}$$

（分部积分，且代入 $\lim_{\boldsymbol{x}\to\infty} \boldsymbol{h}_q(\boldsymbol{x},t)q_t(\boldsymbol{x}) = 0$ 和 $\lim_{\boldsymbol{x}\to\infty} \boldsymbol{h}_p(\boldsymbol{x},t)p_t^{\mathrm{ODE}}(\boldsymbol{x}) = 0, \forall t \in [0,T]$）

$$= \int (\boldsymbol{h}_q(\boldsymbol{x},t)q_t(\boldsymbol{x}))^{\top} \nabla_{\boldsymbol{x}} \log \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x} - \int (\boldsymbol{h}_p(\boldsymbol{x},t)p_t^{\mathrm{ODE}}(\boldsymbol{x}))^{\top} \nabla_{\boldsymbol{x}} \frac{q_t(\boldsymbol{x})}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x}$$

（将对数求导用链式法则展开，接着合并同类项）

$$= \int q_t(\boldsymbol{x}) \left[\boldsymbol{h}_q^{\top}(\boldsymbol{x},t) - \boldsymbol{h}_p^{\top}(\boldsymbol{x},t)\right] \left[\nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_t^{\mathrm{ODE}}(\boldsymbol{x})\right] \mathrm{d}\boldsymbol{x}$$

（代入 $\boldsymbol{h}_q$ 和 $\boldsymbol{h}_p$ 的定义）

$$= -\frac{1}{2} g(t)^2 \mathbb{E}_{q_t(\boldsymbol{x})} \left[\left(\boldsymbol{s}_\theta(\boldsymbol{x},t) - \nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})\right)^{\top} \left(\nabla_{\boldsymbol{x}} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})\right) \right]$$

（代入 $\boldsymbol{h}_q$ 和 $\boldsymbol{h}_p$ 的定义）

$$= -\mathcal{J}_{\mathrm{ODE}}(\theta).$$

代入（4.28）中，有

$$\mathbb{D}_{\mathrm{KL}}(q_0 \,\|\, p_0^{\mathrm{ODE}}) = \mathbb{D}_{\mathrm{KL}}(q_T \,\|\, p_T^{\mathrm{ODE}}) + \mathcal{J}_{\mathrm{ODE}}(\theta). \tag{4.30}$$

此外，根据 $\mathcal{J}_{\mathrm{ODE}}(\theta)$、$\mathcal{J}_{\mathrm{SM}}(\theta)$ 和 $\mathcal{J}_{\mathrm{Diff}}(\theta)$ 的定义，简单合并同类项即可验证

$$\mathcal{J}_{\mathrm{ODE}}(\theta) = \mathcal{J}_{\mathrm{SM}}(\theta) + \mathcal{J}_{\mathrm{Diff}}(\theta). \tag{4.31}$$

∎

4.7.2　定理 4.2 的证明

以下首先给出一个引理，用于计算微分方程 $\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_q(\boldsymbol{x}_t,t)$ 轨迹上的点在 p_t^{ODE} 和 q_t 的分数函数的差，即 $\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$。

引理 4.1： 假设 $\boldsymbol{x}_t$ 满足常微分方程 $\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_q(\boldsymbol{x}_t,t)$，有

$$\begin{aligned}
&\frac{\mathrm{d}(\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t))}{\mathrm{d}t} \\
&= -\left(\nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{h}_p(\boldsymbol{x}_t,t)) - \nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{h}_q(\boldsymbol{x}_t,t))\right) \\
&\quad - \left(\nabla_{\boldsymbol{x}_t} \boldsymbol{h}_p(\boldsymbol{x}_t,t)^{\top} \nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) - \nabla_{\boldsymbol{x}_t} \boldsymbol{h}_q(\boldsymbol{x}_t,t)^{\top} \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\right) \\
&\quad - \nabla_{\boldsymbol{x}_t}^2 \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)\left(\boldsymbol{h}_p(\boldsymbol{x}_t,t) - \boldsymbol{h}_q(\boldsymbol{x}_t,t)\right),
\end{aligned} \tag{4.32}$$

其中 p_t^{ODE}、q_t、$\boldsymbol{h}_q$、$\boldsymbol{h}_p$ 的定义见第 2.2 节。

证明 对于一个固定的 $\boldsymbol{x}$，根据福克-普朗克方程，ODE 定义的概率密度函数 $p_t^{\mathrm{ODE}}(\boldsymbol{x})$

随着时间 t 演化的方程为：

$$\frac{\partial p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot \left(\boldsymbol{h}_p(\boldsymbol{x},t) p_t^{\mathrm{ODE}}(\boldsymbol{x})\right). \tag{4.33}$$

所以

$$\begin{aligned}\frac{\partial \log p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} &= \frac{1}{p_t^{\mathrm{ODE}}(\boldsymbol{x})} \frac{\partial p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} \\ &= -\mathrm{tr}(\nabla_{\boldsymbol{x}} \boldsymbol{h}_p(\boldsymbol{x},t)) - \boldsymbol{h}_p(\boldsymbol{x},t)^{\top} \nabla_{\boldsymbol{x}} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}).\end{aligned} \tag{4.34}$$

两侧再对 $\boldsymbol{x}$ 求导，有

$$\begin{aligned}\frac{\partial \nabla_{\boldsymbol{x}} \log p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} &= \nabla_{\boldsymbol{x}} \frac{\partial \log p_t^{\mathrm{ODE}}(\boldsymbol{x})}{\partial t} \\ &= -\nabla_{\boldsymbol{x}} \mathrm{tr}(\nabla_{\boldsymbol{x}} \boldsymbol{h}_p(\boldsymbol{x},t)) - \nabla_{\boldsymbol{x}} \left(\boldsymbol{h}_p(\boldsymbol{x},t)^{\top} \nabla_{\boldsymbol{x}} \log p_t^{\mathrm{ODE}}(\boldsymbol{x})\right).\end{aligned} \tag{4.35}$$

另一方面，对于 $\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} = \boldsymbol{h}_q(\boldsymbol{x}_t, t)$ 轨迹上的 $\boldsymbol{x}_t$，$\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)$ 对 t 的全微分为

$$\frac{\mathrm{d}\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)}{\mathrm{d}t}$$

（全微分公式）

$$= \frac{\partial \nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)}{\partial \boldsymbol{x}_t} \frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t} + \frac{\partial \nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)}{\partial t}$$

（代入式（4.35））

$$= \nabla_{\boldsymbol{x}_t}^2 \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) \boldsymbol{h}_q(\boldsymbol{x}_t,t) - \nabla_{\boldsymbol{x}_t} \left(\nabla_{\boldsymbol{x}_t} \cdot \boldsymbol{h}_p(\boldsymbol{x}_t,t)\right) - \nabla_{\boldsymbol{x}_t} \left(\boldsymbol{h}_p(\boldsymbol{x}_t,t)^{\top} \nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x})\right)$$

（将乘积的求导展开）

$$\begin{aligned}&= -\nabla_{\boldsymbol{x}_t} \mathrm{tr}(\nabla_{\boldsymbol{x}_t} \boldsymbol{h}_p(\boldsymbol{x}_t,t)) - \nabla_{\boldsymbol{x}_t} \boldsymbol{h}_p(\boldsymbol{x}_t,t)^{\top} \nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) \\ &\quad - \nabla_{\boldsymbol{x}_t}^2 \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t) \left(\boldsymbol{h}_p(\boldsymbol{x}_t,t) - \boldsymbol{h}_q(\boldsymbol{x}_t,t)\right).\end{aligned}$$

同理，对于 $\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$，有

$$\frac{\mathrm{d}\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)}{\mathrm{d}t} = -\nabla_{\boldsymbol{x}_t} \left(\nabla_{\boldsymbol{x}_t} \cdot \boldsymbol{h}_q(\boldsymbol{x}_t,t)\right) - \nabla_{\boldsymbol{x}_t} \boldsymbol{h}_q(\boldsymbol{x}_t,t)^{\top} \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t). \tag{4.36}$$

综上所述，对 $\frac{\mathrm{d}\nabla_{\boldsymbol{x}_t} \log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)}{\mathrm{d}t}$ 和 $\frac{\mathrm{d}\nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)}{\mathrm{d}t}$ 作差即可得所证结论。 ∎

接下来给出定理 4.2 的证明。

证明（定理 4.2）首先，根据 $\boldsymbol{h}_p$ 和 $\boldsymbol{h}_q$ 的定义，有

$$\boldsymbol{h}_p(\boldsymbol{x}_t,t) - \boldsymbol{h}_q(\boldsymbol{x}_t,t) = -\frac{1}{2} g^2(t) \left(\boldsymbol{s}_\theta(\boldsymbol{x}_t,t) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\right). \tag{4.37}$$

所以

$$\begin{aligned}
&\|\boldsymbol{h}_p(\boldsymbol{x}_t,t)-\boldsymbol{h}_q(\boldsymbol{x}_t,t)\|_2 \leqslant \frac{1}{2}g^2(t)\delta_1,\\
&\|\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_p(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_q(\boldsymbol{x}_t,t)\|_2 \leqslant \|\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_p(\boldsymbol{x}_t,t)-\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_q(\boldsymbol{x}_t,t)\|_F \leqslant \frac{1}{2}g^2(t)\delta_2,\\
&\|\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_p(\boldsymbol{x}_t,t))-\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_q(\boldsymbol{x}_t,t))\|_2 \leqslant \frac{1}{2}g^2(t)\delta_3,
\end{aligned} \tag{4.38}$$

其中 $\|\cdot\|_2$ 表示向量或矩阵的 L_2 范数。对于满足 $\frac{\mathrm{d}\boldsymbol{x}_t}{\mathrm{d}t}=\boldsymbol{h}_q(\boldsymbol{x}_t,t)$ 的 $\boldsymbol{x}_t$，根据引理 4.1，有

$$\begin{aligned}
&\nabla_{\boldsymbol{x}_t}\log p_t^{\mathrm{ODE}}(\boldsymbol{x}_t)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\\
=&\nabla_{\boldsymbol{x}_T}\log p_T^{\mathrm{ODE}}(\boldsymbol{x}_T)-\nabla_{\boldsymbol{x}_T}\log q_T(\boldsymbol{x}_T)\\
&+\int_t^T\Big(\nabla_{\boldsymbol{x}_s}\mathrm{tr}(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(\boldsymbol{x}_s,s))-\nabla_{\boldsymbol{x}_s}\mathrm{tr}(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(\boldsymbol{x}_s,s))\Big)\mathrm{d}s\\
&+\int_t^T\Big(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(\boldsymbol{x}_s,s)^\top\nabla_{\boldsymbol{x}_s}\log p_s^{\mathrm{ODE}}(\boldsymbol{x}_s)-\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(\boldsymbol{x}_s,s)^\top\nabla_{\boldsymbol{x}_s}\log q_s(\boldsymbol{x}_s)\Big)\mathrm{d}s\\
&+\int_t^T\nabla^2_{\boldsymbol{x}_s}\log p_s^{\mathrm{ODE}}(\boldsymbol{x}_s)\big(\boldsymbol{h}_p(\boldsymbol{x}_s,s)-\boldsymbol{h}_q(\boldsymbol{x}_s,s)\big)\mathrm{d}s.
\end{aligned} \tag{4.39}$$

为简便起见，以下记 $\boldsymbol{h}_p(s)\coloneqq\boldsymbol{h}_p(\boldsymbol{x}_s,s)$，$\boldsymbol{h}_q(s)\coloneqq\boldsymbol{h}_q(\boldsymbol{x}_s,s)$，$p_s\coloneqq p_s^{\mathrm{ODE}}(\boldsymbol{x}_s)$ 以及 $q_s\coloneqq q_s(\boldsymbol{x}_s)$。由于

$$\begin{aligned}
&\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(s)^\top\nabla_{\boldsymbol{x}_s}\log p_s-\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)^\top\nabla_{\boldsymbol{x}_s}\log q_s\\
=&\big(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(s)-\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)\big)^\top\big(\nabla_{\boldsymbol{x}_s}\log p_s-\nabla_{\boldsymbol{x}_s}\log q_s\big)\\
&+\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)^\top\big(\nabla_{\boldsymbol{x}_s}\log p_s-\nabla_{\boldsymbol{x}_s}\log q_s\big)+\big(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(s)-\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(\boldsymbol{s})\big)^\top\nabla_{\boldsymbol{x}_s}\log q_s,
\end{aligned} \tag{4.40}$$

代入式（4.39），有

$$\begin{aligned}
&\|\nabla_{\boldsymbol{x}_t}\log p_t-\nabla_{\boldsymbol{x}_t}\log q_t\|\\
\leqslant&\|\nabla_{\boldsymbol{x}_T}\log p_T-\nabla_{\boldsymbol{x}_T}\log q_T\|+\int_t^T\big\|\nabla_{\boldsymbol{x}_s}\mathrm{tr}(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(s))-\nabla_{\boldsymbol{x}_s}\mathrm{tr}(\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s))\big\|\mathrm{d}s\\
&+\int_t^T\Big(\big\|\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(s)-\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)\big\|+\big\|\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)\big\|\Big)\big\|\nabla_{\boldsymbol{x}_s}\log p_s-\nabla_{\boldsymbol{x}_s}\log q_s\big\|\mathrm{d}s\\
&+\int_t^T\big\|\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_p(s)-\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)\big\|\big\|\nabla_{\boldsymbol{x}_s}\log q_s\big\|\mathrm{d}s+\int_t^T\|\nabla^2_{\boldsymbol{x}_s}\log p_s\|\,\big\|\boldsymbol{h}_p(s)-\boldsymbol{h}_q(s)\big\|\,\mathrm{d}s\\
\leqslant&\|\nabla_{\boldsymbol{x}_s}\log p_T-\nabla_{\boldsymbol{x}_s}\log q_T\|+\frac{1}{2}\int_t^T g^2(s)\Big(\delta_3+\delta_2\|\nabla_{\boldsymbol{x}_s}\log q_s\|+\delta_1 C\Big)\mathrm{d}s\\
&+\int_t^T\left(\frac{\delta_2}{2}g^2(s)+\|\nabla_{\boldsymbol{x}_s}\boldsymbol{h}_q(s)\|\right)\|\nabla_{\boldsymbol{x}_s}\log p_s-\nabla_{\boldsymbol{x}_s}\log q_s\|\mathrm{d}s.
\end{aligned} \tag{4.41}$$

为简便起见，记

$$\begin{aligned}
u(t) &\coloneqq \|\nabla_{\boldsymbol{x}_t}\log p_t - \nabla_{\boldsymbol{x}_t}\log q_t\|,\\
\alpha(t) &\coloneqq \|\nabla_{\boldsymbol{x}_T}\log p_T - \nabla_{\boldsymbol{x}_T}\log q_T\| + \frac{1}{2}\int_t^T g^2(s)\Big(\delta_3 + \delta_2\|\nabla_{\boldsymbol{x}_s}\log q_s\| + \delta_1 C\Big)\mathrm{d}s,\\
\beta(t) &\coloneqq \frac{\delta_2}{2}g^2(t) + \|\nabla_{\boldsymbol{x}_t}\boldsymbol{h}_q(t)\|.
\end{aligned} \tag{4.42}$$

由于 q_s 与模型无关，故 $\alpha(t)\geqslant 0, \beta(t)\geqslant 0$ 都与 θ 无关，且有

$$u(t) \leqslant \alpha(t) + \int_t^T \beta(s)u(s)\mathrm{d}s. \tag{4.43}$$

根据格朗瓦尔不等式（Grönwall's inequality），有

$$u(t) \leqslant \alpha(t) + \int_t^T \alpha(s)\beta(s)\exp\left(\int_t^s \beta(r)\mathrm{d}r\right)\mathrm{d}s, \tag{4.44}$$

因此

$$\mathbb{D}_{\mathrm{F}}(q_t \,\|\, p_t) = \mathbb{E}_{q_t}[u(t)^2] \leqslant \mathbb{E}_{q_t}\left[\left(\alpha(t) + \int_t^T \alpha(s)\beta(s)\exp\left(\int_t^s \beta(r)\mathrm{d}r\right)\mathrm{d}s\right)^2\right], \tag{4.45}$$

其中不等式的右边只依赖于 δ_1、δ_2、δ_3、C 以及一些只与前向过程 q_t 有关的常数。令

$$U(t;\delta_1,\delta_2,\delta_3,C,q) \coloneqq \left(\alpha(t) + \int_t^T \alpha(s)\beta(s)\exp\left(\int_t^s \beta(r)\mathrm{d}r\right)\mathrm{d}s\right)^2. \tag{4.46}$$

由于 U 关于 $\alpha(t)\geqslant 0$ 和 $\beta(t)\geqslant 0$ 单调递增，而 $\alpha(t)$ 和 $\beta(t)$ 关于 δ_1、δ_2 和 δ_3 单调递增，因此 U 也关于 δ_1、δ_2 和 δ_3 单调递增。 ∎

4.7.3　定理 4.3 的证明

为了证明定理 4.3，以下先引入几个关于一阶分数函数和二阶分数函数的引理。

引理 4.2： 若 $(\boldsymbol{x}_t, \boldsymbol{x}_0)\sim q(\boldsymbol{x}_t, \boldsymbol{x}_0)$，则

$$\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t) = \mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\left[\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\right]. \tag{4.47}$$

证明

$$\begin{aligned}
\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t) &= \frac{\nabla_{\boldsymbol{x}_t}\int q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)q_0(\boldsymbol{x}_0)\mathrm{d}\boldsymbol{x}_0}{q_t(\boldsymbol{x}_t)} = \int \frac{q_0(\boldsymbol{x}_0)\nabla_{\boldsymbol{x}_t}q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q_t(\boldsymbol{x}_t)}\mathrm{d}\boldsymbol{x}_0\\
&= \int q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\frac{\nabla_{\boldsymbol{x}_t}q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\mathrm{d}\boldsymbol{x}_0 = \mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\left[\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\right].
\end{aligned} \tag{4.48}$$

∎

引理 4.3： 若 $(\boldsymbol{x}_t, \boldsymbol{x}_0) \sim q(\boldsymbol{x}_t, \boldsymbol{x}_0)$，则

$$\begin{aligned}\nabla^2_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t) = \mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[&\nabla^2_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) \\ &+ \big(\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\big)\big(\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\big)^\top\Big].\end{aligned} \tag{4.49}$$

证明 首先，$q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)$ 关于 $\boldsymbol{x}_t$ 的梯度有如下等价形式：

$$\begin{aligned}\nabla_{\boldsymbol{x}_t} q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t) &= q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t} \log q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t) \\ &= q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\Big(\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\Big).\end{aligned} \tag{4.50}$$

根据引理 4.2，对等式两侧再次求导，有

$$\begin{aligned}&\nabla^2_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t) = \nabla_{\boldsymbol{x}_t}\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\Big] \\ &= \int q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\nabla^2_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}_t} q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)^\top \mathrm{d}\boldsymbol{x}_0 \\ &\text{（代入式（4.50））} \\ &= \mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\nabla^2_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) \\ &\quad + \Big(\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\Big)\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)^\top\Big] \\ &\text{（根据引理 4.2，合并同类项）} \\ &= \mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\nabla^2_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) \\ &\quad + \big(\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\big)\big(\nabla_{\boldsymbol{x}_t} \log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)\big)^\top\Big].\end{aligned}$$

∎

以下给出定理 4.3 的具体证明。

证明（定理 4.3）为简单起见，记 $q_{t0} \coloneqq q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)$，$q_{0t} \coloneqq q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)$，$q_t \coloneqq q_t(\boldsymbol{x}_t)$，$q_0 \coloneqq q_0(\boldsymbol{x}_0)$，$\hat{\boldsymbol{s}}_1 \coloneqq \hat{\boldsymbol{s}}_1(\boldsymbol{x}_t, t)$，$\boldsymbol{s}_2(\theta) \coloneqq \boldsymbol{s}_2(\boldsymbol{x}_t, t; \theta)$。

由于 $\nabla_{\boldsymbol{x}_t} \log q_{t0} = -\frac{\boldsymbol{\epsilon}}{\sigma_t}$ 以及 $\nabla^2_{\boldsymbol{x}_t} \log q_{t0} = -\frac{1}{\sigma_t^2}\boldsymbol{I}$，式（4.13）可以重写为：

$$\theta^* = \operatorname*{argmin}_\theta \mathbb{E}_{q_t}\mathbb{E}_{q_{0t}}\left[\left\|\boldsymbol{s}_2(\theta) - \nabla^2_{\boldsymbol{x}_t} \log q_{t0} - (\nabla_{\boldsymbol{x}_t} \log q_{t0} - \hat{\boldsymbol{s}}_1)(\nabla_{\boldsymbol{x}_t} \log q_{t0} - \hat{\boldsymbol{s}}_1)^\top\right\|_F^2\right].$$

对于固定的 t 和 $\boldsymbol{x}_t$，最小化上式内部的期望等价于一个关于 $\boldsymbol{s}_2(\theta)$ 最小二乘问题，所以最优的 θ^* 满足

$$\boldsymbol{s}_2(\theta^*) = \mathbb{E}_{q_{0t}}\left[\nabla^2_{\boldsymbol{x}_t} \log q_{t0} + \left(\nabla_{\boldsymbol{x}_t} \log q_{t0} - \hat{\boldsymbol{s}}_1\right)\left(\nabla_{\boldsymbol{x}_t} \log q_{t0} - \hat{\boldsymbol{s}}_1\right)^\top\right]. \tag{4.51}$$

这样的最优模型与真实的二阶分数函数之间的差距为

$$\begin{aligned}&\boldsymbol{s}_2(\theta^*) - \nabla^2_{\boldsymbol{x}_t} \log q_t \\ &\text{（根据引理 4.3，将二阶分数函数展开）}\end{aligned}$$

$$
\begin{aligned}
&= \mathbb{E}_{q_{0t}}\Big[\hat{\boldsymbol{s}}_1\hat{\boldsymbol{s}}_1^\top - \hat{\boldsymbol{s}}_1\nabla_{\boldsymbol{x}_t}\log q_{t0}^\top - \nabla_{\boldsymbol{x}_t}\log q_{t0}\hat{\boldsymbol{s}}_1^\top - \nabla_{\boldsymbol{x}_t}\log q_t\nabla_{\boldsymbol{x}_t}\log q_t^\top \\
&\quad + \nabla_{\boldsymbol{x}_t}\log q_{t0}\nabla_{\boldsymbol{x}_t}\log q_t^\top + \nabla_{\boldsymbol{x}_t}\log q_t\nabla_{\boldsymbol{x}_t}\log q_{t0}^\top\Big] \\
&\text{（根据引理 4.2，将一阶分数函数抵消）} \\
&= (\hat{\boldsymbol{s}}_1 - \nabla_{\boldsymbol{x}_t}\log q_t)(\hat{\boldsymbol{s}}_1 - \nabla_{\boldsymbol{x}_t}\log q_t)^\top.
\end{aligned}
$$

因此有

$$
\begin{aligned}
\|\boldsymbol{s}_2(\theta) - \nabla_{\boldsymbol{x}_t}^2\log q_t\|_F &\leqslant \|\boldsymbol{s}_2(\theta) - \boldsymbol{s}_2(\theta^*)\|_F + \|\boldsymbol{s}_2(\theta^*) - \nabla_{\boldsymbol{x}_t}^2\log q_t\|_F \\
&= \|\boldsymbol{s}_2(\theta) - \boldsymbol{s}_2(\theta^*)\|_F + \|\hat{\boldsymbol{s}}_1 - \nabla_{\boldsymbol{x}_t}\log q_t\|_2^2 \\
&= \big\|\boldsymbol{s}_2(\theta) - \boldsymbol{s}_2(\theta^*)\big\|_F + \delta_1^2.
\end{aligned} \tag{4.52}
$$

更进一步，根据最小二乘的性质，式（4.13）也等价于：

$$
\theta^* = \operatorname*{argmin}_\theta \mathbb{E}_{q_t(\boldsymbol{x}_t)}\|\boldsymbol{s}_2(\boldsymbol{x}_t, t;\theta) - \boldsymbol{s}_2(\boldsymbol{x}_t, t;\theta^*)\|_F^2. \tag{4.53}
$$

所以 $\|\boldsymbol{s}_2(\boldsymbol{x}_t, t;\theta) - \boldsymbol{s}_2(\boldsymbol{x}_t, t;\theta^*)\|_F$ 对应的即是训练误差。 ∎

4.7.4　定理 4.4 的证明

为了证明定理 4.4，以下先引入几个关于二阶分数函数和三阶分数函数的引理。

引理 4.4： 若 $(\boldsymbol{x}_t, \boldsymbol{x}_0) \sim q(\boldsymbol{x}_t, \boldsymbol{x}_0)$，则

$$
\begin{aligned}
&\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)^\top + \nabla_{\boldsymbol{x}_t}^2\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\Big] \\
&= \nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)^\top + \nabla_{\boldsymbol{x}_t}^2\log q_t(\boldsymbol{x}_t),
\end{aligned} \tag{4.54}
$$

且

$$
\begin{aligned}
&\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)\big\|_2^2 + \operatorname{tr}(\nabla_{\boldsymbol{x}_t}^2\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0))\Big] \\
&= \big\|\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2 + \operatorname{tr}(\nabla_{\boldsymbol{x}_t}^2\log q_t(\boldsymbol{x}_t)).
\end{aligned} \tag{4.55}
$$

证明 在式（4.50）两侧同加 $\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)^\top$，再直接利用引理 4.2即可得到式（4.54）。更进一步，对等式两侧同取 $\operatorname{tr}(\cdot)$ 即可得到式（4.55）。 ∎

引理 4.5： 若 $(\boldsymbol{x}_t, \boldsymbol{x}_0) \sim q(\boldsymbol{x}_t, \boldsymbol{x}_0)$ 且 $\nabla_{\boldsymbol{x}_t}^3\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = \boldsymbol{0}$，则

$$
\begin{aligned}
&\nabla_{\boldsymbol{x}_t}\operatorname{tr}(\nabla_{\boldsymbol{x}_t}^2\log q_t(\boldsymbol{x}_t)) \\
&= \mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2\big(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) - \nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big)\Big].
\end{aligned}
$$

证明 首先，由于 $\nabla_{\boldsymbol{x}_t}^3\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = \boldsymbol{0}$，故 $\nabla_{\boldsymbol{x}_t}^2\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ 为常数（与 $\boldsymbol{x}_0$ 和 $\boldsymbol{x}_t$ 无

关），因此 $\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}[\nabla^2_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)]$ 也为常数。因此，在式（4.49）等价于

$$\begin{aligned}&\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\\&=\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))^{\top}\Big].\end{aligned}$$

两侧取 tr(·) 并对 $\boldsymbol{x}_t$ 求梯度，有

$$\begin{aligned}&\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\\&=\nabla_{\boldsymbol{x}_t}\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2\Big]\end{aligned}$$

（利用乘积的求导原则进行展开）

$$\begin{aligned}&=\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2\nabla_{\boldsymbol{x}_t}\log q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)\Big]\\&\quad+\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\nabla_{\boldsymbol{x}_t}\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2\Big]\end{aligned}$$

（利用贝叶斯公式，以及对求导进行展开）

$$\begin{aligned}&=\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\Big]\\&\quad+\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[2(\nabla^2_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))^{\top}(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\Big]\end{aligned}$$

（由于 $\nabla^2_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)$ 是常数，故 $(\nabla^2_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))$ 与 $\boldsymbol{x}_0$ 无关）

$$\begin{aligned}&=\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\Big]\\&\quad+2(\nabla^2_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla^2_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))^{\top}\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\Big]\end{aligned}$$

（根据引理 4.2，上式第二项为零）

$$=\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)\big\|_2^2(\nabla_{\boldsymbol{x}_t}\log q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)-\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t))\Big].$$

∎

基于以上两个引理，以下证明定理 4.4。

证明（定理 4.4）为简单起见，以下记 $q_{t0}\coloneqq q_{t0}(\boldsymbol{x}_t|\boldsymbol{x}_0)$，$q_{0t}\coloneqq q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)$，$q_t\coloneqq q_t(\boldsymbol{x}_t)$，$q_0\coloneqq q_0(\boldsymbol{x}_0)$，$\hat{\boldsymbol{s}}_1\coloneqq\hat{\boldsymbol{s}}_1(\boldsymbol{x}_t,t)$，$\hat{\boldsymbol{s}}_2\coloneqq\hat{\boldsymbol{s}}_2(\boldsymbol{x}_t,t)$，$\boldsymbol{s}_3(\theta)\coloneqq\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)$。

由于 $\nabla_{\boldsymbol{x}_t}\log q_{t0}=-\frac{\epsilon}{\sigma_t}$ 以及 $\nabla^2_{\boldsymbol{x}_t}\log q_{t0}=-\frac{1}{\sigma_t^2}\boldsymbol{I}$，式（4.20）等价于

$$\begin{aligned}\theta^*=\mathop{\mathrm{argmin}}_{\theta}\mathbb{E}_{q_t(\boldsymbol{x}_t)}\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}\Bigg[\Bigg\|\boldsymbol{s}_3(\theta)-\Big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\Big\|_2^2\Big(\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\Big)\\+\Big(\Big(\mathrm{tr}(\hat{\boldsymbol{s}}_2)-\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_{t0})\Big)\boldsymbol{I}+2\Big(\hat{\boldsymbol{s}}_2-\nabla^2_{\boldsymbol{x}_t}\log q_{t0}\Big)\Big)\Big(\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\Big)\Big\|_2^2\Bigg].\end{aligned}$$

对于固定的 t 和 $\boldsymbol{x}_t$，最小化上式内部的期望等价于一个关于 $\boldsymbol{s}_3(\theta)$ 最小二乘问题，

所以最优的 θ^* 满足

$$\begin{aligned}\boldsymbol{s}_3(\theta^*) &= \mathbb{E}_{q_{0t}}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\big\|_2^2\big(\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\big)\Big]\\ &\quad-\mathbb{E}_{q_{0t}}\Big[\Big(\big(\mathrm{tr}(\hat{\boldsymbol{s}}_2)-\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_{t0})\big)\boldsymbol{I}+2\big(\hat{\boldsymbol{s}}_2-\nabla^2_{\boldsymbol{x}_t}\log q_{t0}\big)\Big)\Big(\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\Big)\Big].\end{aligned}$$

由于 $\nabla^2_{\boldsymbol{x}_t}\log q_{t0}=-\frac{1}{\sigma_t^2}\boldsymbol{I}$ 与 $\boldsymbol{x}_0$ 无关，根据引理 4.2，有

$$\begin{aligned}\boldsymbol{s}_3(\theta^*) &= \mathbb{E}_{q_{0t}}\Big[\big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\big\|_2^2\big(\nabla_{\boldsymbol{x}_t}\log q_{t0}-\hat{\boldsymbol{s}}_1\big)\Big]\\ &\quad-\Big(\big(\mathrm{tr}(\hat{\boldsymbol{s}}_2)-\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_{t0})\big)\boldsymbol{I}+2\big(\hat{\boldsymbol{s}}_2-\nabla^2_{\boldsymbol{x}_t}\log q_{t0}\big)\Big)\Big(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\Big).\end{aligned}\tag{4.56}$$

因此，这样的最优模型与真实的三阶分数函数之间的差距为

$$\begin{aligned}&\boldsymbol{s}_3(\theta^*)-\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t)\\ &=\mathbb{E}_{q_{0t}}\Big[\Big(2\nabla_{\boldsymbol{x}_t}\log q_{t0}\nabla_{\boldsymbol{x}_t}\log q_{t0}^{\top}+2\nabla^2_{\boldsymbol{x}_t}\log q_{t0}-2\hat{\boldsymbol{s}}_2\Big)\big(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\big)\Big]\\ &\quad+\mathbb{E}_{q_{0t}}\Big[\Big(\Big\|\nabla_{\boldsymbol{x}_t}\log q_{t0}\Big\|_2^2+\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_{t0})-\mathrm{tr}(\hat{\boldsymbol{s}}_2)\Big)\big(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\big)\Big]\\ &\quad+(\|\hat{\boldsymbol{s}}_1\|_2^2-\|\nabla_{\boldsymbol{x}_t}\log q_t\|_2^2)\mathbb{E}_{q_{0t}}[\nabla_{\boldsymbol{x}_t}\log q_{t0}]\\ &\quad+2\mathbb{E}_{q_{0t}}\Big[(\nabla_{\boldsymbol{x}_t}\log q_{t0}^{\top}\hat{\boldsymbol{s}}_1)\hat{\boldsymbol{s}}_1-(\nabla_{\boldsymbol{x}_t}\log q_{t0}^{\top}\nabla_{\boldsymbol{x}_t}\log q_t)\nabla_{\boldsymbol{x}_t}\log q_t\Big]\\ &\quad+\|\nabla_{\boldsymbol{x}_t}\log q_t\|_2^2\nabla_{\boldsymbol{x}_t}\log q_t-\|\hat{\boldsymbol{s}}_1\|_2^2\hat{\boldsymbol{s}}_1\\ &\text{（根据引理 4.2 以及引理 4.4，有）}\\ &=2\Big(\nabla_{\boldsymbol{x}_t}\log q_t\nabla_{\boldsymbol{x}_t}\log q_t^{\top}+\nabla^2_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_2\Big)\big(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\big)\\ &\quad+\Big(\Big\|\nabla_{\boldsymbol{x}_t}\log q_t\Big\|_2^2+\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t)-\mathrm{tr}(\hat{\boldsymbol{s}}_2)\Big)\big(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\big)\\ &\quad+(\|\hat{\boldsymbol{s}}_1\|_2^2-\|\nabla_{\boldsymbol{x}_t}\log q_t\|_2^2)\nabla_{\boldsymbol{x}_t}\log q_t\\ &\quad+2\Big((\nabla_{\boldsymbol{x}_t}\log q_t^{\top}\hat{\boldsymbol{s}}_1)\hat{\boldsymbol{s}}_1-\|\nabla_{\boldsymbol{x}_t}\log q_t\|_2^2\nabla_{\boldsymbol{x}_t}\log q_t\Big)\\ &\quad+\|\nabla_{\boldsymbol{x}_t}\log q_t\|_2^2\nabla_{\boldsymbol{x}_t}\log q_t-\|\hat{\boldsymbol{s}}_1\|_2^2\hat{\boldsymbol{s}}_1\\ &\text{（根据引理 4.3，有）}\\ &=\Big(2\big(\nabla^2_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_2\big)+\big(\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t)-\mathrm{tr}(\hat{\boldsymbol{s}}_2)\big)\Big)(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1)\\ &\quad+\|\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\|_2^2(\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1)\end{aligned}$$

因此，对上式两侧都取 $\boldsymbol{L}_2$ 范数并利用三角不等式，有

$$\begin{aligned}&\|\boldsymbol{s}_3(\theta)-\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t)\|_2\\ &\leqslant\|\boldsymbol{s}_3(\theta)-\boldsymbol{s}_3(\theta^*)\|_2+\|\boldsymbol{s}_3(\theta^*)-\nabla_{\boldsymbol{x}_t}\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t)\|_2\\ &\leqslant\|\boldsymbol{s}_3(\theta)-\boldsymbol{s}_3(\theta^*)\|_2+\|\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\|_2^3\end{aligned}$$

$$+\left(2\left\|\nabla^2_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_2\right\|_F+\left|\mathrm{tr}(\nabla^2_{\boldsymbol{x}_t}\log q_t)-\mathrm{tr}(\hat{\boldsymbol{s}}_2)\right|\right)\|\nabla_{\boldsymbol{x}_t}\log q_t-\hat{\boldsymbol{s}}_1\|_2$$
$$=\left\|\boldsymbol{s}_3(\theta)-\boldsymbol{s}_3(\theta^*)\right\|_2+\left(\delta_1^2+\delta_{2,\mathrm{tr}}+2\delta_2\right)\delta_1^2.$$

更进一步，根据最小二乘的性质，式（4.20）也等价于：

$$\theta^*=\underset{\theta}{\mathrm{argmin}}\,\mathbb{E}_{q_t(\boldsymbol{x}_t)}\|\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)-\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta^*)\|_2^2, \tag{4.57}$$

所以 $\|\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta)-\boldsymbol{s}_3(\boldsymbol{x}_t,t;\theta^*)\|_2$ 对应的即是训练误差。 ∎

4.8　本章小结

本章通过新颖的高阶去噪分数匹配方法提出了一种对扩散常微分方程的最大似然训练的高效算法。通过分析分数匹配目标与数据分布到扩散常微分方程分布的 KL 散度之间的关系，本章提出可以通过最小化分数模型的一阶、二阶和三阶分数匹配误差来控制 KL 散度的上界。为最小化高阶分数匹配的误差，本章进一步提出了一种误差可控的高阶去噪分数匹配算法，使得高阶分数匹配误差可以由训练误差和低阶分数匹配误差共同限制，且分数模型的全局最优解仍与扩散模型的原始训练目标相同。实验证明，本章所提方法可以极大地提高多个不同密度建模基准任务上扩散常微分方程的模型密度。

第 5 章　针对无条件扩散模型的高效采样算法

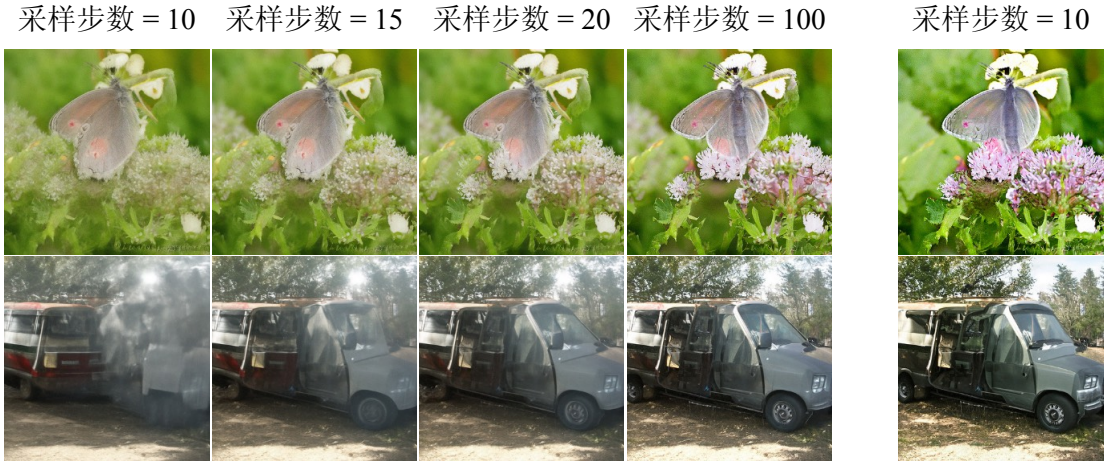
扩散常微分方程共有两大核心应用：准确的似然计算和高质量的样本生成。第 4 章提出了扩散常微分方程的最大似然训练的高效算法，使得扩散常微分方程可以达到最先进的似然估计性能。本章考虑扩散常微分方程的另一重要问题，即如何快速采样得到高质量的样本。总体而言，从扩散常微分方程中采样的过程可以被视为使用常微分方程数值求解器（solver）求解对应的微分方程。然而，目前扩散常微分方程使用的求解器仍然需要成百上千步串行的函数调用才可以得到高质量的采样，这极大限制了模型在下游任务中的应用。本章推导出了扩散常微分方程的解的精确表达式，并基于该表达式提出了 *DPM-Solver*，一种专为扩散常微分方程设计的且具有收敛阶数保证的快速高阶求解器。实验上，DPM-Solver 可以在各种数据集上在无需任何额外训练前提下仅用 10 到 20 次函数调用就可以生成高质量样本，其无条件采样速度是先前最快速的无需训练的采样器的 4 到 16 倍。

5.1　本章引言

扩散概率模型（diffusion probabilistic models，DPM，以下简称扩散模型）是一种新兴的具有强大表达能力的生成模型。尽管该模型具有高质量的采样性能，但该类模型采样一个样本通常需要数百至数千次大型神经网络的串行函数调用（采样步数），这比其它单步生成模型例如生成对抗网络（generative adversarial networks）[8]或变分自编码器（variational auto-encoders）[26]的采样速度慢得多。这样低效的采样速度成为了扩散模型在下游任务应用的关键瓶颈。因此，如何为扩散模型设计快速采样算法是该领域的核心问题之一。

具体而言，现有的扩散模型快速采样算法可被分为两类。第一类包括知识蒸馏[75-76]和噪声水平或样本轨迹学习[77-80]。上述方法需要较为昂贵的额外训练，这导致其额外开销是不容忽视的。其次，上述方法的适用性和灵活性都较为受限，使用者需要做大量的额外调整以适应不同的模型、数据集和采样步数。第二类为无需训练的采样器[81-83]，这些采样器简单且即插即用，适用于所有预训练的扩散模型。具体而言，无需训练的采样器包括更换隐式[81]或解析[83]的生成过程、更先进的微分方程求解器[45,82,84-86]以及对采样时间点的动态规划[80]。然而，这些方法中最快的算法仍然需要大约 50 次函数调用[83]来生成高质量样本（指与普通采样器通过约 1000 次函数调用得到的生成质量相当），因此仍然耗时较久。

为了解决这一问题，本章考虑在“少步采样”（few-step sampling）范围内生成



(a) DDIM[81]　　(b) DPM-Solver

图 5.1　ImageNet 256×256 数据集上 DDIM 和 DPM-Solver 在不同采样步数下的采样结果

高质量样本的问题，其中少步采样是指在约 10 步串行函数调用内完成采样。由于扩散模型的采样可以通过求解对应的扩散常微分方程（diffusion ODE）来完成，因此本章集中在针对扩散常微分方程的特殊结构设计专用的快速求解器。具体而言，扩散常微分方程由数据变量的线性函数和由神经网络参数化的非线性函数共同组成，因而具有半线性（semi-linear）结构。然而，先前的无需训练的采样器[45,82]忽略了上述结构，而是直接使用黑盒（black-box）的微分方程求解器。为了利用这样的半线性结构，本章通过解析地计算扩散常微分方程解的线性部分，推导出扩散常微分方程解的精确表达式，从而避免了相应的离散化误差。接着，通过应用变量替换公式（change-of-variable formula），上述解可以等价地简化为关于神经网络的指数加权积分（exponentially weighted integral）。这种积分非常特殊，可通过指数积分器（exponential integrators）[131]的数值方法进行高效近似。

基于上述解析的表达式，本章提出了 *DPM-Solver*，一种专为扩散常微分方程设计的快速采样器，包括了具有收敛阶数保证的一、二、三阶求解器。并且，DPM-Solver 同时适用于连续时间和离散时间的扩散模型，无需任何额外训练。图 5.1 展示了在 ImageNet 256 × 256 数据集[9]上基准线方法去噪扩散隐式模型（denoising diffusion implicit models，DDIM）[81]和 DPM-Solver 在不同的采样步数的加速性能，实验基于 Dhariwal 等人[49]的预训练模型。结果表明，DPM-Solver 可以在仅仅 10 次函数调用下生成高质量样本，其采样速度远超 DDIM。此外，本章的其它实验结果也表明，DPM-Solver 可以显著提高扩散模型的采样速度，并且可以在 10 到 20 次函数调用内实现出色的采样质量，其速度远超以往所有无需训练的采样器。

5.2　算法设计

先前的工作中使用的通用的黑盒常微分方程求解器[45]在实践中无法在少量采样步数内收敛。为了解决这个问题，本节将详细讨论如何为扩散常微分方程设计专用的求解器，以实现快速和高质量的少步采样。

5.2.1　扩散常微分方程的精确解

本节将说明，给定时间 $s > 0$ 的初始值 $\boldsymbol{x}_s$，扩散常微分方程在式（2.35）中的每个时间 $t < s$ 的解 $\boldsymbol{x}_t$ 可以简化成一个非常特殊的精确表达式。

首先，本节的第一个关键发现是，由于扩散常微分方程有特定的半线性结构，t 时间的解 $\boldsymbol{x}_t$ 的一部分可以被解析地计算。具体而言，扩散常微分方程对应的式（2.35）的右侧由两部分组成：一部分 $f(t)\boldsymbol{x}_t$ 是 $\boldsymbol{x}_t$ 的线性函数，而另一部分 $(g^2(t)/2\sigma_t)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ 通常是 $\boldsymbol{x}_t$ 的非线性函数，因为神经网络 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ 是高度非线性的。这类 ODE 被称为半线性 ODE。先前工作采用的黑盒常微分方程求解器[45]并没有考虑到这种半线性结构，因为这类求解器将式（2.35）中的 $\boldsymbol{h}_p(\boldsymbol{x}_t, t)$ 作为一个整体用于求解器中，这导致线性项和非线性项的都存在不可忽略的离散化误差。然而，对于半线性 ODE，可以通过常数变易定理（variation-of-constants formula）[132]得到时间 t 的精确解：

$$\boldsymbol{x}_t = e^{\int_s^t f(\tau)\mathrm{d}\tau}\boldsymbol{x}_s + \int_s^t \left(e^{\int_\tau^t f(r)\mathrm{d}r} \frac{g^2(\tau)}{2\sigma_\tau} \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau) \right) \mathrm{d}\tau. \tag{5.1}$$

该表达式将线性部分和非线性部分巧妙地分离开来。与黑盒的 ODE 求解器相比，该表达式的线性部分是解析计算的，从而消除了线性项对应的的离散化误差。

其次，由于非线性部分的积分耦合了扩散模型相关的系数（即 $f(\tau)$、$g(\tau)$、σ_τ）和复杂的神经网络 $\boldsymbol{\epsilon}_\theta$，因而该积分仍然较为复杂且难以近似。为了解决这个问题，本节的第二个关键发现是，通过引入一个特殊的变量，该非线性部分的积分可以大幅度简化。具体而言，令 $\lambda_t \coloneqq \log(\alpha_t/\sigma_t)$，对应的是对数信噪比（log-signal-to-noise ratio，log-SNR）的一半，那么 λ_t 是 t 的严格递减函数（根据扩散模型的定义，详见第 2.2 节）。此时，式（2.19）中的 $g(t)$ 有如下等价形式：

$$g^2(t) = \frac{\mathrm{d}\sigma_t^2}{\mathrm{d}t} - 2\frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\sigma_t^2 = 2\sigma_t^2\left(\frac{\mathrm{d}\log\sigma_t}{\mathrm{d}t} - \frac{\mathrm{d}\log\alpha_t}{\mathrm{d}t}\right) = -2\sigma_t^2\frac{\mathrm{d}\lambda_t}{\mathrm{d}t}. \tag{5.2}$$

结合式（2.19）中的 $f(t) = \mathrm{d}\log\alpha_t/\mathrm{d}t$，故式（5.1）可以重写为

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \alpha_t\int_s^t \left(\frac{\mathrm{d}\lambda_\tau}{\mathrm{d}\tau}\right)\frac{\sigma_\tau}{\alpha_\tau}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau)\mathrm{d}\tau. \tag{5.3}$$

由于 $\lambda(t) = \lambda_t$ 是 t 的严格递减函数，故其存在一个对应的反函数 $t_\lambda(\cdot)$ 满足 $t = t_\lambda(\lambda(t))$。由此，进一步将 $\boldsymbol{x}$ 和 $\boldsymbol{\epsilon}_\theta$ 的下标从 t 换元为 λ，并记 $\hat{\boldsymbol{x}}_\lambda \coloneqq \boldsymbol{x}_{t_\lambda(\lambda)}$、$\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda) \coloneqq$

$\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{t_\lambda(\lambda)}, t_\lambda(\lambda))$ 为关于 λ 的函数。通过关于 λ 的“变量替换公式”（change-of-variable formula）重写式（5.3），可以得出如下命题：

命题 5.1 (扩散常微分方程的精确解)： 给定时间 $s > 0$ 的初始解 $\boldsymbol{x}_s$，式（2.35）中的扩散常微分方程在时间 $t \in [0, s]$ 的解 $\boldsymbol{x}_t$ 为：

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \alpha_t \int_{\lambda_s}^{\lambda_t} e^{-\lambda}\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)\mathrm{d}\lambda. \tag{5.4}$$

为简便起见，以下称 $\int e^{-\lambda}\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)\mathrm{d}\lambda$ 为 $\hat{\boldsymbol{\epsilon}}_\theta$ 的指数加权积分（exponentially weighted integral），因为该表达式与常微分方程求解器文献中的指数积分器（exponential integrators）[131]高度相关。此外，命题 5.1中的精确解在扩散模型的先前工作中尚未被揭示。

式（5.4）为近似扩散常微分方程的解提供了一个新的视角。具体而言，给定时间 s 的初始解 $\boldsymbol{x}_s$，根据式（5.4），对时间 t 的解 $\boldsymbol{x}_t$ 进行近似等价于直接近似从 λ_s 到 λ_t 的 $\hat{\boldsymbol{\epsilon}}_\theta$ 的指数加权积分。这样的等价形式避免了线性项的离散化误差，且非线性项的近似在指数积分器的文献中也有深入的研究[131,133]。基于这一发现，以下将提出扩散常微分方程的专用高阶求解器。

5.2.2　扩散常微分方程的高阶求解器

本节基于式（5.4）提出具有收敛阶数保证的扩散常微分方程的高阶求解器。该求解器的设计和阶数分析受常微分方程文献中指数积分器方法[131,133]的高度启发。

具体而言，给定时间 T 的初始值 $\boldsymbol{x}_T$ 和从 $t_0 = T$ 递减到 $t_M = 0$ 的 $M + 1$ 个时间点 $\{t_i\}_{i=0}^M$，其中 $\tilde{\boldsymbol{x}}_{t_0} = \boldsymbol{x}_T$ 为给定的初始值。本节所提出的求解器使用 M 步迭代计算序列 $\{\tilde{\boldsymbol{x}}_{t_i}\}_{i=0}^M$ 来近似在时间点 $\{t_i\}_{i=0}^M$ 的准确解。特别地，最后一步迭代得到的 $\tilde{\boldsymbol{x}}_{t_M}$ 为时间 0 对应的近似采样。

为了减少 $\tilde{\boldsymbol{x}}_{t_M}$ 和时间 0 的真实解之间的近似误差，求解器需要减少每一步中 $\tilde{\boldsymbol{x}}_{t_i}$ 的近似误差[132]。对于每个 i，从时间 t_{i-1} 的近似解 $\tilde{\boldsymbol{x}}_{t_{i-1}}$ 开始，根据式（5.4），时间 t_i 的准确解 $\boldsymbol{x}_{t_{i-1}\to t_i}$ 为：

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i} \int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} e^{-\lambda}\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)\mathrm{d}\lambda. \tag{5.5}$$

因此，为了计算 $\boldsymbol{x}_{t_{i-1}\to t_i}$ 对应的近似值 $\tilde{\boldsymbol{x}}_{t_i}$，需要近似从 $\lambda_{t_{i-1}}$ 到 λ_{t_i} 的 $\hat{\boldsymbol{\epsilon}}_\theta$ 的指数加权积分。记 $h_i \coloneqq \lambda_{t_i} - \lambda_{t_{i-1}}$ 为积分步长，$\hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_\lambda, \lambda) \coloneqq \mathrm{d}^n\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)/\mathrm{d}\lambda^n$ 为 $\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)$ 关于 λ 的 n 阶全导数（total derivative）。对于 $k \geqslant 1$，$\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)$ 关于 λ 在 $\lambda_{t_{i-1}}$ 处的 $(k-1)$

阶泰勒展开（Taylor expansion）为

$$\hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda) = \sum_{n=0}^{k-1} \frac{(\lambda - \lambda_{t_{i-1}})^n}{n!} \hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) + \mathcal{O}((\lambda - \lambda_{t_{i-1}})^k). \tag{5.6}$$

将上述泰勒展开代入式（5.5），有

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}} \tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i} \sum_{n=0}^{k-1} \hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) \int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} e^{-\lambda} \frac{(\lambda - \lambda_{t_{i-1}})^n}{n!} \mathrm{d}\lambda + \mathcal{O}(h_i^{k+1}), \tag{5.7}$$

其中积分 $\int e^{-\lambda}((\lambda - \lambda_{t_{i-1}})^n/n!)\mathrm{d}\lambda$ 可以通过反复应用 n 次分部积分（integration-by-parts）得到解析的计算结果。具体而言，定义[133]：

$$\varphi_k(h) := \int_0^1 e^{(1-\delta)h} \frac{\delta^{k-1}}{(k-1)!} \mathrm{d}\delta, \qquad \varphi_0(h) = e^h, \tag{5.8}$$

其满足 $\varphi_k(0) = 1/k!$ 和递推关系 $\varphi_{k+1}(h) = (\varphi_k(h) - \varphi_k(0))/h$，因此每个 k 对应的 $\varphi_k(h)$ 都有解析形式。例如，以下列出 $k = 1, 2, 3$ 的 φ_k 对应的解析形式：

$$\begin{aligned} \varphi_1(h) &= \frac{e^h - 1}{h}, \\ \varphi_2(h) &= \frac{e^h - h - 1}{h^2}, \\ \varphi_3(h) &= \frac{e^h - \frac{h^2}{2} - h - 1}{h^3}. \end{aligned} \tag{5.9}$$

因此，式（5.7）中 $\boldsymbol{x}_{t_{i-1}\to t_i}$ 可以被等价地改写为：

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}} \tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i} \sum_{n=0}^{k-1} h_i^{n+1} \varphi_{n+1}(h_i) \hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) + \mathcal{O}(h_i^{k+1}), \tag{5.10}$$

其中 h_i 和 $\varphi_{n+1}(h_i)$ 都为解析的标量（scalar）。因此，为了近似 $\boldsymbol{x}_{t_{i-1}\to t_i}$，只需近似神经网络的 n 阶全导数 $\hat{\boldsymbol{\epsilon}}_\theta^{(n)}(\hat{\boldsymbol{x}}_\lambda, \lambda)$，其中 $n \leqslant k-1$。这种近似恰好是 ODE 求解器研究领域中的标准问题，已有非常成熟的基于有限差分（finite difference）的解决技巧[133-134]。该技巧不需要对神经网络求导，而只需要对不同时间的神经网络 $\hat{\boldsymbol{\epsilon}}_\theta$ 的输出进行线性组合即可。因此，通过忽略高阶误差项 $\mathcal{O}(h_i^{k+1})$ 并使用神经网络的有限差分近似前 $(k-1)$ 阶全导数[133-134]，可以推导出扩散常微分方程的 k 阶求解器。本章将这样得到的求解器命名为 *DPM-Solver*，并将对应的 k 阶求解器命名为 *DPM-Solver-k*。

以下以 $k = 1$ 为例进行简要说明。在这种情况下，式（5.10）为

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}} \tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}(e^{h_i} - 1)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) + \mathcal{O}(h_i^2). \tag{5.11}$$

通过舍弃高阶误差项 $\mathcal{O}(h_i^2)$，可以得到 $\boldsymbol{x}_{t_{i-1}\to t_i}$ 对应的一阶近似解。由于此处 $k = 1$，该求解器被称为 *DPM-Solver-1*，对应的详细算法如下所述，算法流程见算法 5.1。

算法 5.1 DPM-Solver-1 算法流程

输入： 时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^M$，噪声预测模型 $\boldsymbol{\epsilon}_\theta$
输出： 扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$
for $i \leftarrow 1$ to M **do**
　$h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}\left(e^{h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

算法 5.2 DPM-Solver-2 算法流程

输入： 时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^M$，噪声预测模型 $\boldsymbol{\epsilon}_\theta$，超参数 $r_1 = 0.5$
输出： 扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$
for $i \leftarrow 1$ to M **do**
　$h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
　$s_i \leftarrow t_\lambda\left(\lambda_{t_{i-1}} + r_1 h_i\right)$
　$\boldsymbol{u}_i \leftarrow \frac{\alpha_{s_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{s_i}\left(e^{r_1 h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}(e^{h_i}-1)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \frac{\sigma_{t_i}}{2r_1}(e^{h_i}-1)(\boldsymbol{\epsilon}_\theta(\boldsymbol{u}_i, s_i) - \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}))$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

DPM-Solver-1. 给定时间 T 的初始值 $\boldsymbol{x}_T$ 和从 $t_0 = T$ 递减到 $t_M = 0$ 的 $M+1$ 个时间点 $\{t_i\}_{i=0}^M$。从 $\tilde{\boldsymbol{x}}_{t_0} = \boldsymbol{x}_T$ 开始，序列 $\{\tilde{\boldsymbol{x}}_{t_i}\}_{i=1}^M$ 按照以下方式迭代计算：

$$\tilde{\boldsymbol{x}}_{t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}(e^{h_i}-1)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}), \quad \text{其中} h_i = \lambda_{t_i} - \lambda_{t_{i-1}}. \tag{5.12}$$

对于 $k \geqslant 2$，近似泰勒展开的前 k 项需要在 t 和 s 之间取额外的中间时间点[133]，如算法 5.2 和算法 5.3 所示，其中 DPM-Solver-2 所选择的中间点为 $(s_i, \boldsymbol{u}_i)$，DPM-Solver-3 所选择的中间点为 $(s_{2i-1}, \boldsymbol{u}_{2i-1})$ 和 $(s_{2i}, \boldsymbol{u}_{2i})$。

5.2.3　实现细节

时间点的选取。 第 5.2.2 节中提出的求解器需要首先指定中间时间点 $\{t_i\}_{i=0}^M$。本章实验中使用的时间点为区间 $[\lambda_T, \lambda_0]$ 的均匀分割，即对于 $i = 0, \cdots, M$，每个 t_i 满足 $\lambda_{t_i} = \lambda_T + (i/M)(\lambda_0 - \lambda_T)$。值得注意的是，这与先前的工作[44-45]不同，因为先前的工作选择了关于 t_i 的均匀步长，而本章实验为关于 λ_{t_i} 的均匀步长。从经验上看，具有均匀时间步长 λ_{t_i} 的 DPM-Solver 已经可以在很少的步数中生成相当好的样本，详见第 5.4 节。此外，当采样步数小于等于 20 时，需要结合不同阶数的 DPM-Solver。具体而言，当采样步数不能被 3 整除时，首先需尽可能多地应用

算法 5.3 DPM-Solver-3 算法流程

输入： 时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^M$，噪声预测模型 $\boldsymbol{\epsilon}_\theta$，超参数 $r_1=\frac{1}{3}, r_2=\frac{2}{3}$
输出： 扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$
for $i \leftarrow 1$ to M **do**
　$h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
　$s_{2i-1} \leftarrow t_\lambda\left(\lambda_{t_{i-1}} + r_1 h_i\right), \quad s_{2i} \leftarrow t_\lambda\left(\lambda_{t_{i-1}} + r_2 h_i\right)$
　$\boldsymbol{u}_{2i-1} \leftarrow \frac{\alpha_{s_{2i-1}}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{s_{2i-1}}\left(e^{r_1 h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
　$\boldsymbol{D}_{2i-1} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{u}_{2i-1}, s_{2i-1}) - \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
　$\boldsymbol{u}_{2i} \leftarrow \frac{\alpha_{s_{2i}}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{s_{2i}}\left(e^{r_2 h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \frac{\sigma_{s_{2i}} r_2}{r_1}\left(\frac{e^{r_2 h_i}-1}{r_2 h_i}-1\right)\boldsymbol{D}_{2i-1}$
　$\boldsymbol{D}_{2i} \leftarrow \boldsymbol{\epsilon}_\theta(\boldsymbol{u}_{2i}, s_{2i}) - \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}\left(e^{h_i}-1\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \frac{\sigma_{t_i}}{r_2}\left(\frac{e^{h_i}-1}{h}-1\right)\boldsymbol{D}_{2i}$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

DPM-Solver-3，然后在最后一步应用 DPM-Solver-1 或 DPM-Solver-2（取决于 K 除以 3 的余数）。

t_λ 的计算。　目前常用的扩散模型[44-45,78]都预先定义了 N 个时间点 $\{t_n\}_{n=1}^N$ 对应的 α_n 和 σ_n。为了将这些离散的 α_n 推广到连续版本，本章所有实验对 $\log\alpha_n$ 函数直接进行线性插值（linear interpolation）。具体而言，对任意 $t\in[t_n, t_{n+1}]$，定义

$$\log\alpha_t := \log\alpha_n + \frac{\log\alpha_{n+1}-\log\alpha_n}{t_{n+1}-t_n}(t-t_n). \tag{5.13}$$

由此可以对所有 $t\in[0,T]$ 定义关于 t 的连续函数 α_t 以及对应的 $\sigma_t=\sqrt{1-\alpha_t^2}$ 和 $\lambda_t=\log\alpha_t-\log\sigma_t$。此外，由于 λ_t 关于 t 是单调递减的，因此 λ_t 仍然有对应的反函数 t_λ，且该反函数可以根据式（5.13）解析地计算。

5.3　与现有快速采样算法的比较

本节进一步讨论 DPM-Solver 和现有基于扩散常微分方程的快速采样方法之间的关系，并突出二者的差异。

DDIM 是 DPM-Solver 的一阶情形。　去噪扩散隐式模型（denoising diffusion implicit models，DDIM）[81]设计了一种确定性的算法，用于从扩散模型中快速采样。对于两个相邻的时间 t_{i-1} 和 t_i，从 t_{i-1} 的初始值 $\tilde{\boldsymbol{x}}_{t_{i-1}}$ 开始，DDIM 得到的 t_{i-1} 的

近似解为

$$\tilde{\boldsymbol{x}}_{t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\left(\frac{\sigma_{t_{i-1}}}{\alpha_{t_{i-1}}} - \frac{\sigma_{t_i}}{\alpha_{t_i}}\right)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}). \tag{5.14}$$

根据 λ 的定义，有 $\sigma_{t_{i-1}}/\alpha_{t_{i-1}} = e^{-\lambda_{t_{i-1}}}$ 和 $\sigma_{t_i}/\alpha_{t_i} = e^{-\lambda_{t_i}}$。将这些和 $h_i = \lambda_{t_i} - \lambda_{t_{i-1}}$ 代入式（5.14）得出的结果与式（5.12）中 DPM-Solver-1 完全一致。因此，DDIM 恰好为 DPM-Solver 的一阶情形。并且，DPM-Solver 给出了系统的高阶求解器设计方法和收敛阶分析，因此推广了 DDIM。

此外，近期研究[75]通过对式（5.14）的两边进行微分，也证明了 DDIM 是扩散常微分方程的一阶离散化。然而，该方法无法解释 DDIM 与扩散常微分方程的一阶欧拉法（Euler's method）离散化之间的区别。相反，本节证明了 DDIM 是 DPM-Solver 的一阶特例，因此 DDIM 充分利用了扩散常微分方程的半线性性质，这进一步可以解释其相对于传统欧拉法的优越性。

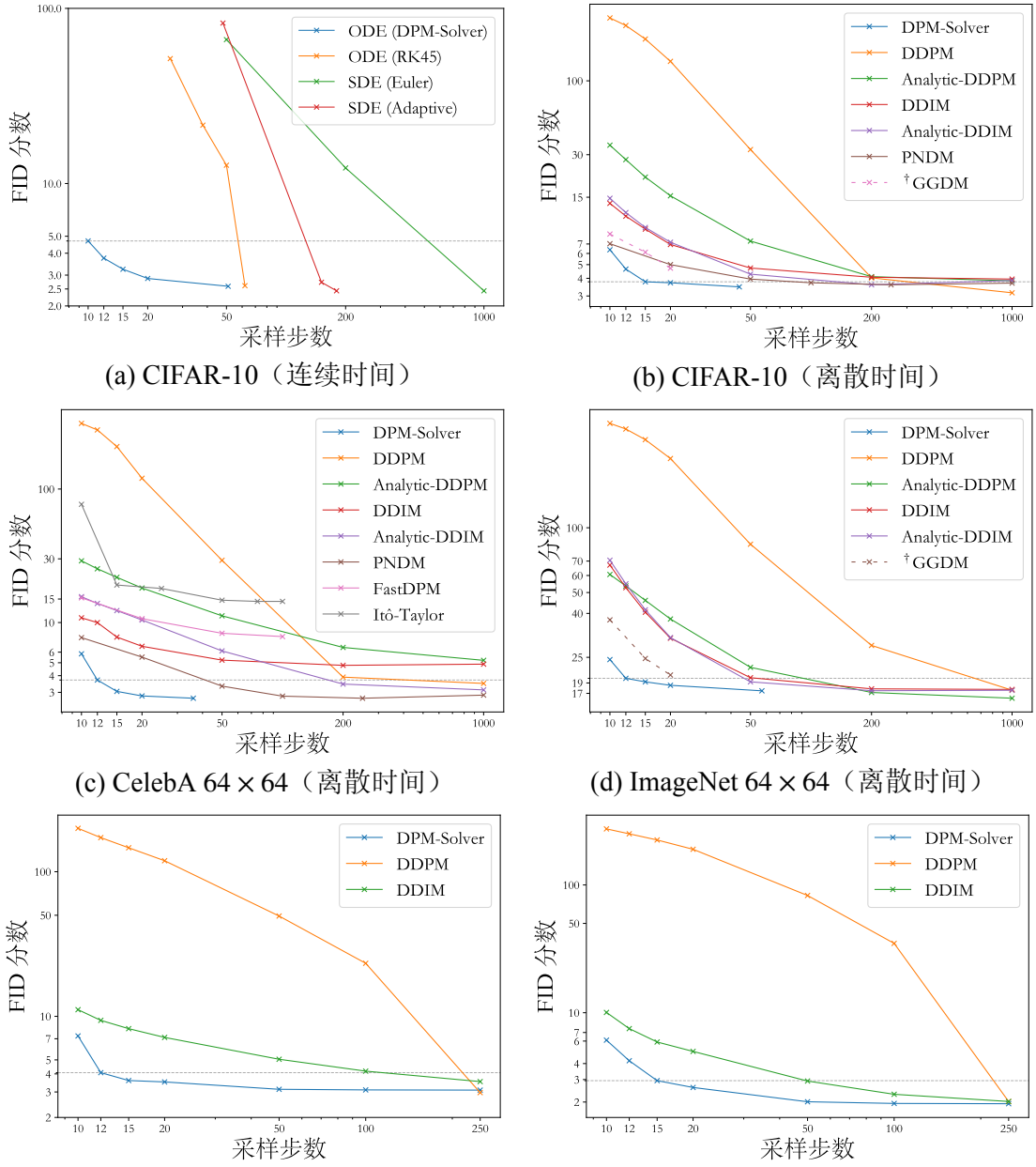
与传统龙格-库塔法的比较。　直接将传统的龙格-库塔（Runge-Kutta，RK）方法应用于式（2.35）中的扩散常微分方程，也可以获得一个高阶 ODE 求解器。具体而言，RK 方法将式（2.35）的解写成以下的积分形式：

$$\boldsymbol{x}_t = \boldsymbol{x}_s + \int_s^t \boldsymbol{h}_p(\boldsymbol{x}_\tau, \tau)\mathrm{d}\tau = \boldsymbol{x}_s + \int_s^t \left(f(\tau)\boldsymbol{x}_\tau + \frac{g^2(\tau)}{2\sigma_\tau}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_\tau, \tau)\right)\mathrm{d}\tau, \tag{5.15}$$

并结合 $\boldsymbol{h}_p$ 在 $[t, s]$ 之间的一些中间时间点的值来近似整个积分。因此，RK 方法的近似误差取决于 $\boldsymbol{h}_p$，其包含线性项 $f(\tau)\boldsymbol{x}_\tau$ 对应的误差和非线性噪声预测模型 $\boldsymbol{\epsilon}_\theta$ 对应的误差。然而，由于线性项的精确解具有指数系数（如式（5.1）所示），线性项的误差可能呈指数增长。大量实证证据[131,133]表明，直接使用 RK 方法求解半线性 ODE 可能会因较大的步长而遭受不稳定的数值问题。对于扩散常微分方程，第 5.4.1 节中比较了所提出的 DPM-Solver 和传统 RK 方法，表明 DPM-Solver 具有比同阶 RK 方法更小的离散化误差。

依赖额外训练的快速采样算法。　需要额外训练或优化的采样算法包括知识蒸馏[75-76]，学习噪声水平或方差[77-78,135]，以及学习样本轨迹[79-80]。尽管渐进式蒸馏方法[75]可以在 4 步内获得较高质量的样本，但该方法需要进一步的训练成本，并且在原始扩散模型中丢失了部分信息（例如，在蒸馏后，噪声预测模型无法预测 $[0, T]$ 之间每个时间步的噪声（即对应于分数函数））。相比之下，无需额外训练的采样器可以保留原始模型的所有信息，从而可以通过结合原始模型和额外的分类器[49]直接扩展到条件采样。

除了直接为扩散模型设计快速采样器外，一些工作还提出了支持更快采样的



(a) CIFAR-10（连续时间）　(b) CIFAR-10（离散时间）

(c) CelebA 64 × 64（离散时间）　(d) ImageNet 64 × 64（离散时间）

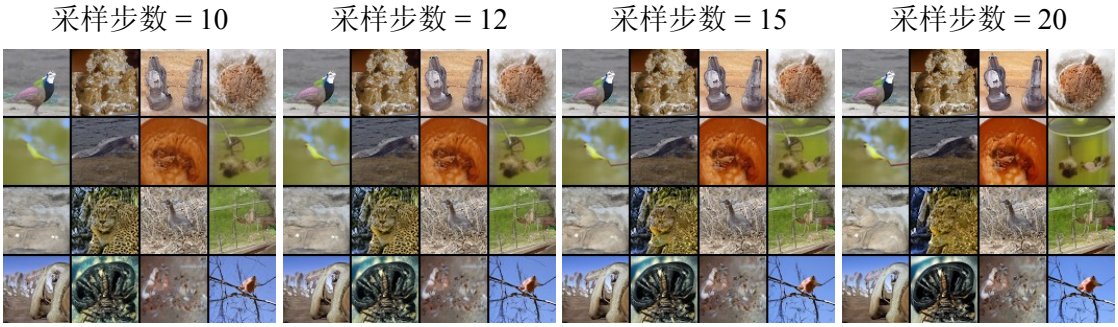
(e) ImageNet 128 × 128（离散时间）　(f) LSUN bedroom 256 × 256（离散时间）

图 5.2　在多个数据集上，不同采样方法使用不同采样步数所得结果的 FID 分数的比较

新型扩散模型。例如，为扩散模型定义低维潜变量[136]；设计具有有界分数函数的特殊扩散过程[137]；将生成对抗网络与扩散模型的逆过程结合[138]。本章所提出的 DPM-Solver 也可能适用于加速这些模型的采样，留待后续深入研究。

5.4　实验结果

本节从实验层面展示，作为一种无需训练的采样器，DPM-Solver 可以极大地加速现有预训练扩散模型的采样，既包括连续时间和离散时间的扩散模型，也包括线



(a) DDIM[81]

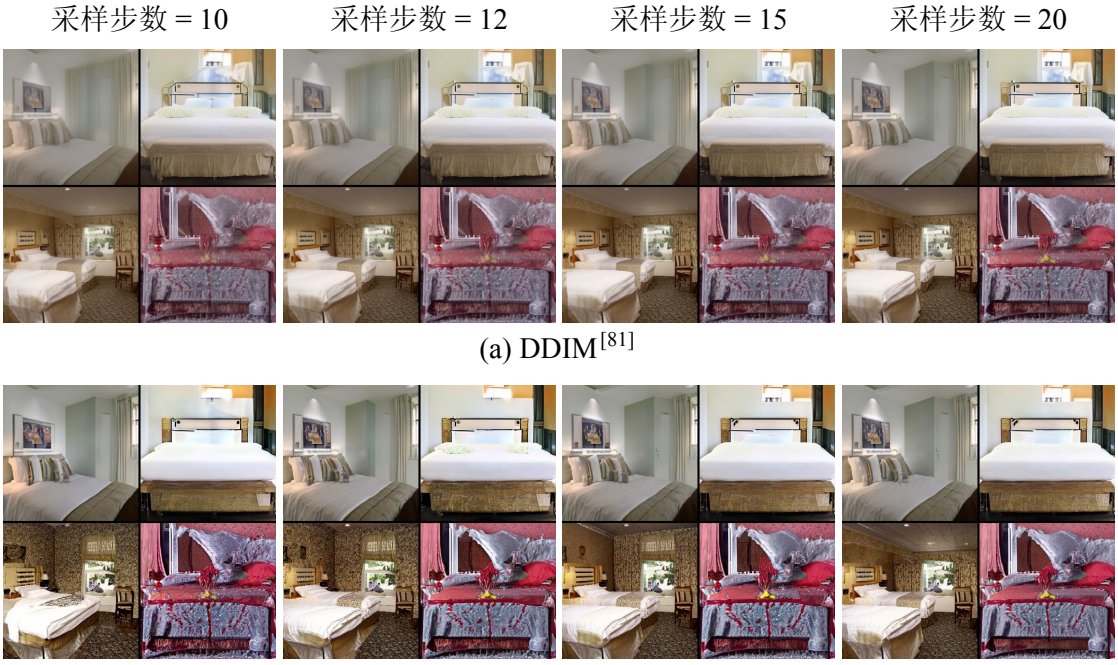
(b) DPM-Solver

图 5.3　基于 ImageNet 64×64 数据集上的预训练模型，DDIM 和 DPM-Solver 在同样的随机数种子下，采样步数为 10、12、15 和 20 的采样结果

表 5.1　不同阶数的 RK 方法和 DPM-Solver 在 CIFAR-10 上使用不同采样步数所得结果的 FID 分数

采样算法 \ 采样步数	12	18	24	30	36	42	48
RK2 (t)	16.40	7.25	3.90	3.63	3.58	3.59	3.54
RK2 (λ)	107.81	42.04	17.71	7.65	4.62	3.58	3.17
DPM-Solver-2	**5.28**	**3.43**	**3.02**	**2.85**	**2.78**	**2.72**	**2.69**
RK3 (t)	48.75	21.86	10.90	6.96	5.22	4.56	4.12
RK3 (λ)	34.29	4.90	3.50	3.03	2.85	2.74	2.69
DPM-Solver-3	**6.03**	**2.90**	**2.75**	**2.70**	**2.67**	**2.65**	**2.65**

性噪声时间表（linear noise schedule）[44,81]和余弦噪声时间表（cosine noise schedule）[78]。本节比较了 DPM-Solver 和其它采样算法的样本质量，所有采样步数指的是计算噪声预测模型 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ 的函数调用次数（number of function evaluations）。所有实验都采样五万个样本，并使用 FID 分数[129]来评估样本质量，其中较低的 FID 通常意味着更好的样本质量。



(a) DDIM[81]

(b) DPM-Solver

图 5.4　基于 LSUN bedroom 256 × 256 数据集上的预训练模型，DDIM 和 DPM-Solver 在同样的随机数种子下，采样步数为 10、12、15 和 20 的采样结果

5.4.1　与连续时间的采样算法的比较

本节将 DPM-Solver 与其它针对连续时间扩散模型的采样方法进行比较，比较的方法包括扩散随机微分方程的 Euler-Maruyama 离散化[45]，扩散随机微分方程的自适应步长求解器（adaptive step size solver）[82]以及式（2.35）中扩散常微分方程的 RK 方法[45,108]。本节实验在 CIFAR-10 数据集[125]上采样预训练的连续时间的“VP deep” 模型[45]，该模型采用线性噪声时间表。

实验结果如图 5.2 (a) 所示。具体而言，实验使用 50、200、1000 的采样步数的均匀时间步对 Euler 离散化的扩散随机微分方程进行采样，并调整自适应步长随机微分方程求解器[82]和 RK45 常微分方程求解器[108]的容差（tolerance）超参数[45,82]来控制采样步数。DPM-Solver 在大约 10 步内就可以生成不错的样本，但其他求解器即使在 50 步也有较大的离散化误差，这表明 DPM-Solver 可以达到以前最好的求解器几乎 5 倍的速度。值得一提的是，DPM-Solver 在采样步数为 10 时达到了 4.70 的 FID 分数，在采样步数为 12 时达到了 3.75 的 FID 分数，在采样步数为 15 时达到了 3.24 的 FID 分数，在采样步数为 20 时达到了 2.87 的 FID 分数，这也是目前 CIFAR-10 上最快的采样器。

此外，作为一项消融研究，本节还对二阶和三阶的 DPM-Solver 与对应阶数的 RK 方法进行了比较，如表 5.1 所示。通过对式（2.35）应用变量替换，实验比较了扩散常微分方程关于 t 的 RK 方法和关于 λ 的 RK 方法。结果显示，在给定相同采

表 5.2　在多个数据集上，DDIM 和 DPM-Solver 在单张 NVIDIA A40 上单次采样运行时间的平均值与标准差

采样算法 \ 采样步数	10	20	100
CIFAR-10 32×32 数据集（批大小为 128）			
DDIM	0.956(±0.011)	1.924(±0.016)	9.668(±0.013)
DPM-Solver	0.923(±0.006)	1.833(±0.004)	9.204(±0.011)
CelebA 64×64 数据集（批大小为 128）			
DDIM	3.253(±0.015)	6.438(±0.029)	32.255(±0.044)
DPM-Solver	3.126(±0.003)	6.272(±0.006)	31.269(±0.012)
ImageNet 64×64 数据集（批大小为 128）			
DDIM	5.084(±0.018)	10.194(±0.022)	50.926(±0.042)
DPM-Solver	4.992(±0.004)	9.991(±0.003)	49.835(±0.028)
ImageNet 128×128 数据集（批大小为 128）			
DDIM	29.082(±0.015)	58.159(±0.012)	290.874(±0.134)
DPM-Solver	28.865(±0.011)	57.645(±0.008)	288.157(±0.022)
LSUN bedroom 256×256 数据集（批大小为 64）			
DDIM	37.700(±0.005)	75.316(±0.013)	378.790(±0.105)
DPM-Solver	36.996(±0.039)	73.873(±0.023)	369.090(±0.076)

样步数的情况下，DPM-Solver 的样本质量始终优于具有相同阶数的 RK 方法。此外，在采样步数小于 15 时，DPM-Solver 的高效性尤为明显，而 RK 方法则具有相当大的离散化误差。这主要是因为 DPM-Solver 解析地计算了线性项，避免了相应的离散化误差。此外，更高阶的 DPM-Solver-3 比 DPM-Solver-2 收敛得更快，也符合定理 5.1 中的阶数分析。

5.4.2　与离散时间的采样算法的比较

本节基于第 5.2.3 节中的方法将 DPM-Solver 用于离散时间扩散模型，并将 DPM-Solver 与其他离散时间的采样器进行比较，包括 DDPM[44]、DDIM[81]、Analytic-DDPM[83]、Analytic-DDIM[83]、PNDM[84]、FastDPM[139]和 Itô-Taylor 方法[86]。此外，还将 DPM-Solver 与 GGDM[80]进行了比较，该方法使用相同的预

训练模型但需要进一步训练采样轨迹。实验比较了采样步数从 10 到 1000 所得样本的采样质量。如图 5.2 所示，在所有数据集上，DPM-Solver 可以在 12 步内获得不错的采样质量（CIFAR-10 上的 FID 为 4.65，CelebA 64×64 上的 FID 为 3.71，ImageNet 64×64 上的 FID 为 19.97，ImageNet 128×128 上的 FID 为 4.08），这是之前最快无需训练采样器的采样速度的 4 到 16 倍，甚至优于需要额外训练的 GGDM。这样的实验结果充分说明了 DPM-Solver 的通用性和高效性。

5.4.3　与已有算法运行时间的比较

本节比较 DPM-Solver 与已有算法 DDIM 的运行时间，以说明 DPM-Solver 的实际加速效果。

理论上，对于相同的采样步数，DPM-Solver 和 DDIM 的运行时间几乎相同（与采样步数几乎成线性关系），这是因为主要的计算时间开销是大型神经网络 $\boldsymbol{\epsilon}_\theta$ 的串行调用，其他系数都是通过可忽略的计算成本解析地计算的。

本节从定量实验中验证了这一点。具体而言，表 5.2 展示了在多个数据集和不同的采样步数下，DPM-Solver 和 DDIM 在单个 NVIDIA A40 上的运行时间，其中每个实验对 8 次采样的运行时间计算平均值和标准差。由于 GPU 的内存限制，实验中对 LSUN bedroom 256×256 数据集使用 64 的批大小，而对其他数据集使用 128 的批大小。实验采用 DDIM 的官方实现①。经发现，DPM-Solver 的实现减少了系数的一些重复计算，因此在相同采样步数下，DPM-Solver 比官方的实现的 DDIM 稍快。尽管如此，实验结果显示，对于相同的采样步数，DPM-Solver 和 DDIM 的运行时间几乎相同，且运行时间大致与采样步数成线性关系。因此，采样步数的加速几乎是运行时间的实际加速，所以本章所提出的 DPM-Solver 可以极大地加速扩散模型的采样。

5.5　收敛阶数的理论保证

本节给出 DPM-Solver 收敛阶数的理论保证及其证明。

首先列出一些关于神经网络的正则性假设。记 $\boldsymbol{x}_s$ 为从 $\boldsymbol{x}_T$ 开始的扩散常微分方程（式（2.35））在时间 s 的精确解。本节做出以下假设：

假设 5.1： 对于任意 $0 \leqslant j \leqslant k+1$，噪声预测模型 $\hat{\boldsymbol{\epsilon}}_\theta$ 关于 λ 的 j 阶全导数 $\mathrm{d}^j \hat{\boldsymbol{\epsilon}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda)/\mathrm{d}\lambda^j$ 存在且连续。

假设 5.2： 函数 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}, s)$ 关于 $\boldsymbol{x}$ 是利普希兹连续的。

假设 5.3： 定义最大步长 $h_{\max} \coloneqq \max_{1 \leqslant i \leqslant M}(\lambda_{t_i} - \lambda_{t_{i-1}})$，那么 $h_{\max} = \mathcal{O}(1/M)$。

① https://github.com/ermongroup/ddim

其中，假设 5.1 保证了式（5.7）中的泰勒展开的存在性；假设 5.2 用于将 $\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_s, s)$ 替换为 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s) + \mathcal{O}(\|\boldsymbol{x}_s - \tilde{\boldsymbol{x}}_s\|)$，以便关于 λ_s 的泰勒展开是适用的；假设 5.3 用于排除一个显著大的步长，而这对于本章所选取的步长而言是显然成立的。

基于上述假设，以下定理说明，DPM-Solver-k 是扩散常微分方程的 k 阶求解器。

定理 5.1： 若 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ 满足假设 5.1 至 5.3，那么对于 $k = 1, 2, 3$，由 DPM-Solver-k 求解得到的 0 时间的近似解 $\tilde{\boldsymbol{x}}_{t_M}$ 满足 $\|\tilde{\boldsymbol{x}}_{t_M} - \boldsymbol{x}_0\| = \mathcal{O}(h_{\max}^k)$。

证明 以下分别给出 $k = 1, 2, 3$ 的相应的证明。

$k = 1$ 的证明。 在式（5.10）中取 $n = 0, t = t_i, s = t_{i-1}$，可得对应的精确解为

$$\boldsymbol{x}_{t_i} = \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\boldsymbol{x}_{t_{i-1}} - \sigma_t(e^{h_i} - 1)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{t_{i-1}}, t_{i-1}) + \mathcal{O}(h_i^2). \tag{5.16}$$

另一方面，由 DPM-Solver-1 得到的近似解满足

$$\begin{aligned}
\tilde{\boldsymbol{x}}_{t_i} &= \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}(e^{h_i} - 1)\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) \\
&\text{（根据假设 5.2）} \\
&= \frac{\alpha_{t_i}}{\alpha_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \sigma_{t_i}(e^{h_i} - 1)\left(\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{t_{i-1}}, t_{i-1}) + \mathcal{O}(\|\tilde{\boldsymbol{x}}_{t_{i-1}} - \boldsymbol{x}_{t_{i-1}}\|)\right) \\
&\text{（代入式（5.16））} \\
&= \boldsymbol{x}_{t_i} + \mathcal{O}(h_{max}^2) + \mathcal{O}(\|\tilde{\boldsymbol{x}}_{t_{i-1}} - \boldsymbol{x}_{t_{i-1}}\|).
\end{aligned}$$

因此有

$$\|\tilde{\boldsymbol{x}}_{t_i} - \boldsymbol{x}_{t_i}\| = \mathcal{O}(h_{max}^2) + \mathcal{O}(\|\tilde{\boldsymbol{x}}_{t_{i-1}} - \boldsymbol{x}_{t_{i-1}}\|). \tag{5.17}$$

重复以上迭代，即可得

$$\|\tilde{\boldsymbol{x}}_{t_M} - \boldsymbol{x}_{t_0}\| = \mathcal{O}(M h_{max}^2) = \mathcal{O}(h_{max}), \tag{5.18}$$

其中最后一个等式基于假设 5.3。因此，DPM-Solver-1 为扩散常微分方程的 1 阶求解器。

对于更高阶的求解器，也可以用类似的方法证明，其证明核心步骤为：先证明局部的离散化误差为 $k+1$ 阶，接着迭代相加即可得到最终的收敛阶数为 k。因此，以下证明只关注从时间 s 到时间 t 的局部离散化误差，并证明 DPM-Solver-k 的局部误差为 $k+1$ 阶。具体而言，考虑 $0 < t < s < T$，令 $h := \lambda_t - \lambda_s$，且 s 时间的初始值为 $\boldsymbol{x}_s$。

$k=2$ 的证明。　首先，在式（5.10）中取 $n=1$，可得对应的精确解为

$$\boldsymbol{x}_t=\frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s-\sigma_t h\varphi_1(h)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s)-\sigma_t h^2\varphi_2(h)\hat{\boldsymbol{\epsilon}}_\theta^{(1)}(\hat{\boldsymbol{x}}_{\lambda_s},\lambda_s)+\mathcal{O}(h^3). \tag{5.19}$$

而 DPM-Solver-2 计算时间 t 的近似解 $\tilde{\boldsymbol{x}}_t$ 的更新规则为：

$$\begin{aligned}
s_1&=t_\lambda\left(\lambda_s+r_1h\right),\\
\tilde{\boldsymbol{u}}&=\frac{\alpha_{s_1}}{\alpha_s}\boldsymbol{x}_s-\sigma_{s_1}\left(e^{r_1h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s),\\
\tilde{\boldsymbol{x}}_t&=\frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s-\sigma_t\left(e^h-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s)-\frac{\sigma_t}{2r_1}(e^h-1)(\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{u}},s_1)-\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s))\\
&\quad\text{（将 }\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{s_1},s_1)\text{ 在点 }\lambda_s\text{ 处泰勒展开）}\\
&=\frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s-\sigma_t\left(e^h-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s)-\underbrace{\frac{\sigma_t}{2r_1}\left(e^h-1\right)\left[\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{u}},s_1)-\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{s_1},s_1)\right]}_{(1)}\\
&\quad-\frac{\sigma_t}{2r_1}\left(e^h-1\right)\left[(\lambda_{s_1}-\lambda_s)\hat{\boldsymbol{\epsilon}}_\theta^{(1)}(\hat{\boldsymbol{x}}_{\lambda_s},\lambda_s)+\mathcal{O}(h^2)\right].
\end{aligned}$$

另一方面，根据假设 5.2，有

$$\|\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{u}},s_1)-\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{s_1},s_1)\|=\mathcal{O}(\|\tilde{\boldsymbol{u}}-\boldsymbol{x}_{s_1}\|)=\mathcal{O}(h^2), \tag{5.20}$$

其中第二个等式基于 $k=1$ 的收敛性。此外，由于 $e^h-1=\mathcal{O}(h)$，故 (1) 是 $\mathcal{O}(h^3)$ 的。故 DPM-Solver-2 等价于

$$\tilde{\boldsymbol{x}}_t=\frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s-\sigma_t\left(e^h-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s)-\frac{\sigma_t}{2r_1}\left(e^h-1\right)(\lambda_{s_1}-\lambda_s)\hat{\boldsymbol{\epsilon}}_\theta^{(1)}(\hat{\boldsymbol{x}}_{\lambda_s},\lambda_s)+\mathcal{O}(h^3). \tag{5.21}$$

将式（5.21）与式（5.19）比较，可得

$$\boldsymbol{x}_t-\tilde{\boldsymbol{x}}_t=\sigma_t\left[h^2\varphi_2(h)-(e^h-1)\frac{\lambda_{s_1}-\lambda_s}{2r_1}\right]\hat{\boldsymbol{\epsilon}}_\theta^{(1)}(\hat{\boldsymbol{x}}_{\lambda_s},\lambda_s)+\mathcal{O}(h^3). \tag{5.22}$$

由于 $\lambda_{s_1}-\lambda_s=r_1h$ 且 $\varphi_2(h)=(e^h-h-1)/h^2$，有

$$h^2\varphi_2(h)-(e^h-1)\frac{\lambda_{s_1}-\lambda_s}{2r_1}=(2e^h-h-2-he^h)/2=\mathcal{O}(h^3). \tag{5.23}$$

代入式（5.22），有 $\tilde{\boldsymbol{x}}_t=\boldsymbol{x}_t+\mathcal{O}(h^3)$，从而完成了证明。

$k=3$ 的证明。　DPM-Solver-3 计算时间 t 的近似解 $\tilde{\boldsymbol{x}}_t$ 的更新规则为：

$$\begin{aligned}
s_1&=t_\lambda\left(\lambda_s+r_1h\right),\quad s_2=t_\lambda\left(\lambda_s+r_2h\right),\\
\tilde{\boldsymbol{u}}_1&=\frac{\alpha_{s_1}}{\alpha_s}\boldsymbol{x}_s-\sigma_{s_1}\left(e^{r_1h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s),\\
\boldsymbol{D}_1&=\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{u}}_1,s_1)-\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s,s),
\end{aligned}$$

$$\begin{aligned}
\tilde{\boldsymbol{u}}_2 &= \frac{\alpha_{s_2}}{\alpha_s}\boldsymbol{x}_s - \sigma_{s_2}\left(e^{r_2 h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s) - \frac{\sigma_{s_2} r_2}{r_1}\left(\frac{e^{r_2 h}-1}{r_2 h}-1\right)\boldsymbol{D}_1,\\
\boldsymbol{D}_2 &= \boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{u}}_2, s_2) - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s),\\
\tilde{\boldsymbol{x}}_t &= \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \sigma_t\left(e^{h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s) - \frac{\sigma_t}{r_2}\left(\frac{e^{h}-1}{h}-1\right)\boldsymbol{D}_2.
\end{aligned}$$

首先证明

$$\tilde{\boldsymbol{u}}_2 = \boldsymbol{x}_{s_2} + \mathcal{O}(h^3), \tag{5.24}$$

这与 $k=2$ 的证明类似。具体而言，由于 $((e^{r_2 h}-1)/r_2 h)-1=\mathcal{O}(h)$ 和 $\tilde{\boldsymbol{u}}_1 = \boldsymbol{x}_{s_1} + \mathcal{O}(h^2)$，有

$$\begin{aligned}
\tilde{\boldsymbol{u}}_2 &= \frac{\alpha_{s_2}}{\alpha_s}\boldsymbol{x}_s - \sigma_{s_2}\left(e^{r_2 h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s)\\
&\quad - \sigma_{s_2}\frac{r_2}{r_1}\left(\frac{e^{r_2 h}-1}{r_2 h}-1\right)\left(\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{s_1}, s_1) - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s)\right) + \mathcal{O}(h^3)\\
&= \frac{\alpha_{s_2}}{\alpha_s}\boldsymbol{x}_s - \sigma_{s_2}\left(e^{r_2 h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s)\\
&\quad - \sigma_{s_2}\frac{r_2}{r_1}\left(\frac{e^{r_2 h}-1}{r_2 h}-1\right)\boldsymbol{\epsilon}_\theta^{(1)}(\boldsymbol{x}_s, s)(\lambda_{s_1}-\lambda_s) + \mathcal{O}(h^3).
\end{aligned} \tag{5.25}$$

令 $h_2 = r_2 h$。与 $k=2$ 的证明思路类似，只需通过泰勒展开检查如下等式即可：

$$\begin{aligned}
&\varphi_1(h_2)h_2 = e^{h_2}-1,\\
&\varphi_2(h_2)h_2^2 = \frac{r_2}{r_1}\left(\frac{e^{h_2}-1}{h_2}-1\right)(\lambda_{s_1}-\lambda_s) + \mathcal{O}(h^3).
\end{aligned} \tag{5.26}$$

因此，式（5.24）成立。将 $\tilde{\boldsymbol{u}}_2 = \boldsymbol{x}_{s_2} + \mathcal{O}(h^3)$ 和 $\lambda_{s_2} - \lambda_s = r_2 h = 2h/3$ 代入 DPM-Solver-3 的更新式中，有

$$\begin{aligned}
\tilde{\boldsymbol{x}}_t &= \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \sigma_t\left(e^{h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s) - \frac{\sigma_t}{r_2}\left(\frac{e^{h}-1}{h}-1\right)\left(\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{u}}_2, s_2) - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s)\right)\\
&= \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \sigma_t\left(e^{h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s) - \frac{\sigma_t}{r_2}\left(\frac{e^{h}-1}{h}-1\right)\left(\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{s_2}, s_2) - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s)\right) + \mathcal{O}(h^4)\\
&= \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \sigma_t\left(e^{h}-1\right)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s)\\
&\quad - \sigma_t\frac{1}{r_2}\left(\frac{e^{h}-1}{h}-1\right)\left(\boldsymbol{\epsilon}_\theta^{(1)}(\boldsymbol{x}_s, s) r_2 h + \frac{1}{2}\boldsymbol{\epsilon}_\theta^{(2)}(\boldsymbol{x}_s, s) r_2^2 h^2\right) + \mathcal{O}(h^4).
\end{aligned}$$

另一方面，在式（5.10）中取 $n=2$，有

$$\boldsymbol{x}_t = \frac{\alpha_t}{\alpha_s}\boldsymbol{x}_s - \sigma_t h\varphi_1(h)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_s, s) - \sigma_t h^2\varphi_2(h)\boldsymbol{\epsilon}_\theta^{(1)}(\boldsymbol{x}_s, s) - \sigma_t h^3\varphi_3(h)\boldsymbol{\epsilon}_\theta^{(2)}(\boldsymbol{x}_s, s) + \mathcal{O}(h^4).$$

比较 $\boldsymbol{x}_t$ 和 $\tilde{\boldsymbol{x}}_t$，只需要证明：

$$\begin{aligned} h\varphi_1(h) &= e^h - 1, \\ h^2\varphi_2(h) &= \left(\frac{e^h-1}{h} - 1\right) h, \\ h^3\varphi_3(h) &= \left(\frac{e^h-1}{h} - 1\right) \frac{r_2 h^2}{2} + \mathcal{O}(h^4). \end{aligned} \tag{5.27}$$

其中前两个条件已在 $k=2$ 中证明。而对于第三个条件，只需要注意到

$$h^3\varphi_3(h) = e^h - 1 - h - \frac{h^2}{2} = \frac{h^3}{6} + \mathcal{O}(h^4) = \left(\frac{e^h-1}{h} - 1\right) \frac{r_2 h^2}{2}. \tag{5.28}$$

因此，$\|\tilde{\boldsymbol{x}}_t - \boldsymbol{x}_t\| = \mathcal{O}(h^4)$。 ∎

5.6　本章小结

本章针对扩散模型的采样速度慢的问题提出了 DPM-Solver，一种快速、专用、无需训练的扩散常微分方程求解器，只需要大约 10 步左右的函数调用就可以对扩散模型进行快速采样。DPM-Solver 利用了扩散常微分方程的半线性结构，并直接近似扩散常微分方程的精确解的简化表达式，该表达式由噪声预测模型的指数加权积分组成。受指数积分器的数值方法的启发，本章提出了一阶、二阶和三阶 DPM-Solver 来近似噪声预测模型的指数加权积分，并具有理论上的收敛保证。DPM-Solver 可以应用于连续时间和离散时间的扩散模型。多个数据集上的实验结果表明，DPM-Solver 可以只用 10 到 20 次函数调用就生成高质量的样本，并且与以前的最先进的无需训练的采样器相比，可以实现 4 到 16 倍的加速。

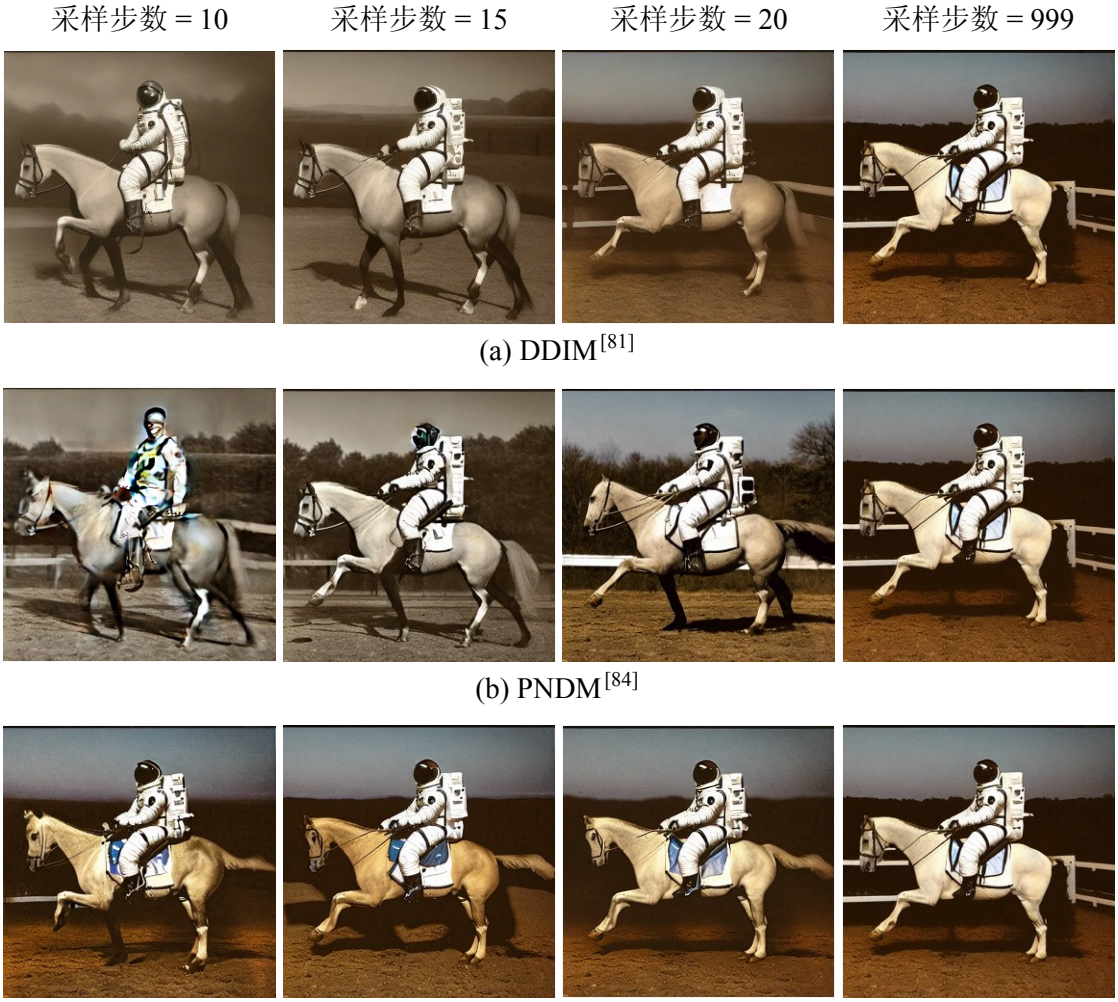
第 6 章　针对条件扩散模型的高效采样算法

第 5 章提出了扩散常微分方程的专用快速高阶求解器，使得扩散常微分方程可以在仅仅 10 到 20 次函数调用下获得较高质量的采样。然而，该类快速采样器仅仅针对无条件扩散模型有很好的效果，对于条件扩散模型的效果仍不确定（详见图 6.3）。条件扩散模型的采样基于引导采样，其同时结合了无条件模型和条件模型，并且往往需要一个较大的引导尺度来确保最佳的采样质量。本章仔细探讨了条件扩散模型的快速采样问题，发现已有的高阶快速采样器在少步采样时都存在不稳定的问题，并且当引导尺度变大时，已有的高阶采样器甚至变得比一阶采样器更慢。为了解决这一问题并进一步加速引导采样，本章提出了 *DPM-Solver*++，一种用于条件扩散模型的高效采样算法。DPM-Solver++ 基于数据预测模型来求解扩散常微分方程，并采用阈值法保证解的范围符合训练数据分布。实验表明，DPM-Solver++ 可以在像素空间和隐空间扩散模型的引导采样中仅用 15 到 20 步就可以生成高质量的样本，极大地提升了条件扩散模型的采样速度和稳定性。

6.1　本章引言

与其它深度生成模型（如生成对抗网络（generative adversarial networks，GAN）[8]或变分自编码器（variational auto-encoders，VAE）[26]相比，由于扩散模型建模了数据分布的分数函数（score function），通过利用一种称为引导采样（guided sampling）[49,109]的技术，扩散模型可以达到比传统条件生成模型更好的采样质量，且很容易扩展到大规模的多模态生成任务中[15,18,73]。具体而言，引导采样将无条件扩散模型和额外的引导模型（guidance）进行线性组合，通过额外的超参数引导尺度（guidance scale）来控制额外监督信号的强弱，以此提高生成样本的保真度和条件与样本的匹配度。由于引导模型可以引入不同模态的条件信息，扩散模型在文本到图像、图像到图像的任务中能生成与给定条件高度相关的高分辨率艺术图像，这也引领了人工智能艺术绘画的新趋势[15,18]。

扩散模型的采样过程需要逐渐从纯高斯随机变量中去除噪声以获得清晰的数据，该过程也可以看作是对由参数化神经网络定义的噪声预测模型（noise-prediction model）或数据预测模型[44,54]（data-prediction model）定义的扩散随机微分方程[44-45]或扩散常微分方程[45,81]的离散化（discretization）。对于引导采样而言，以往常用的采样方法是去噪扩散隐式模型（denoising diffusion implicit models，DDIM）[81]。正如第 5 章所述，该类方法等价于一阶扩散常微分方程求解器[75,89]。



(a) DDIM[81]

(b) PNDM[84]

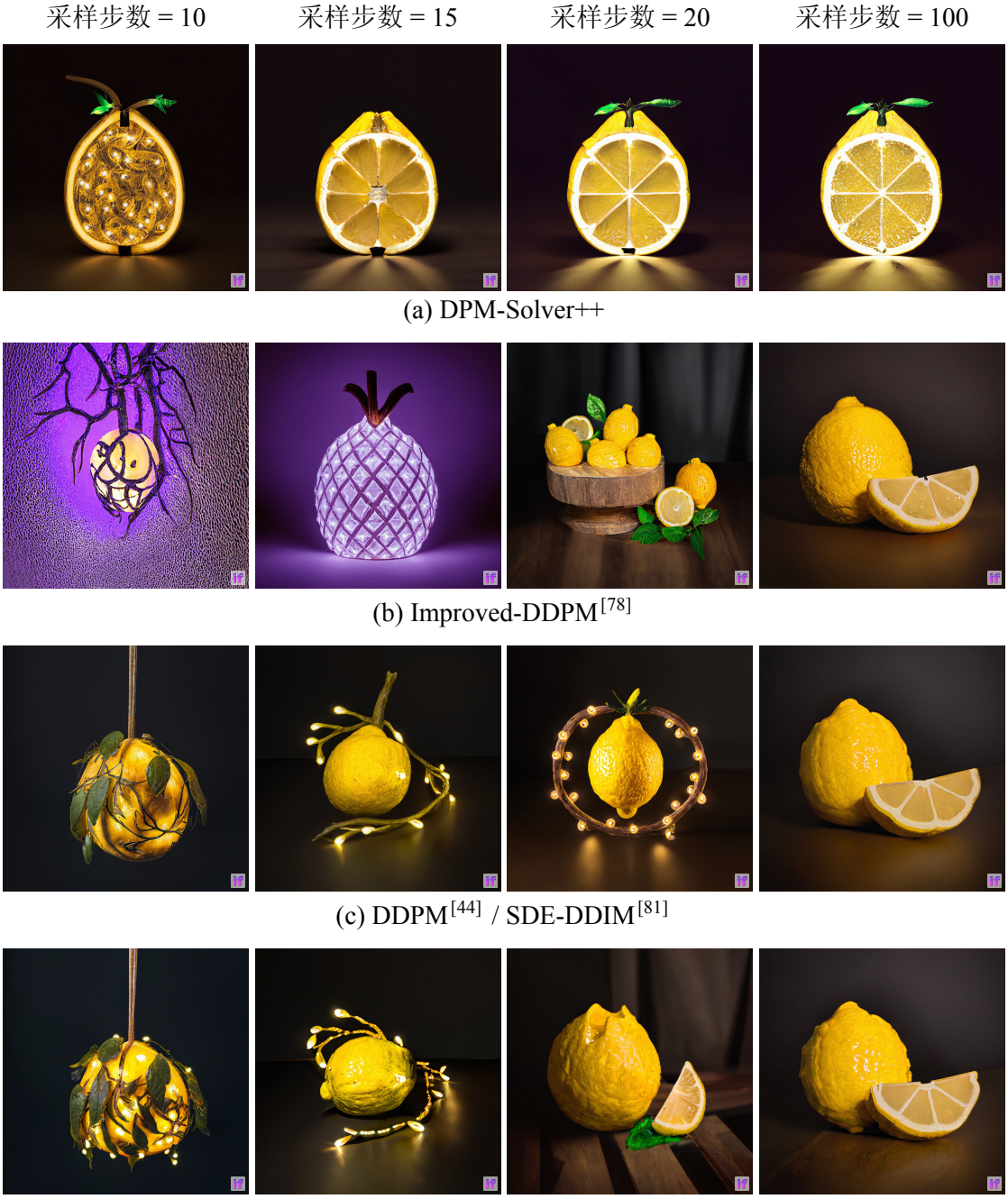
(c) DPM-Solver++

图 6.1　不同采样器在 Stable Diffusion 模型上的采样结果

对于引导采样而言，该类方法通常需要 100 到 250 步的大型神经网络的串行调用才能完成收敛，因而非常耗时。

如第 5 章中提出的专用的高阶扩散常微分方程求解器[89,140]可以在没有引导（即无条件模型）的情况下在 10 到 20 步内生成高质量的样本。然而，该类工作在引导采样中的有效性仍是不确定的，如图 6.3 所示。本章仔细探讨了高阶扩散常微分方程求解器在引导采样中面临的问题，发现先前针对扩散常微分方程的高阶求解器在引导采样下生成的样本并不理想，甚至比简单的一阶求解器 DDIM 更差。这一反直觉的现象本质上反映了高阶求解器在引导采样时的额外困难。具体而言，本章提出了将高阶求解器应用于引导采样时面临的两个挑战：（1）较大的引导尺度会进一步缩小高阶求解器的收敛半径，使其变得非常不稳定；（2）收敛的解可能与原始数据并不在相同的范围内（也称为“训练-测试不匹配”[73]）。

为了解决这些问题，本章提出了 *DPM-Solver++*，一种用于引导采样且无需训



(a) DPM-Solver++

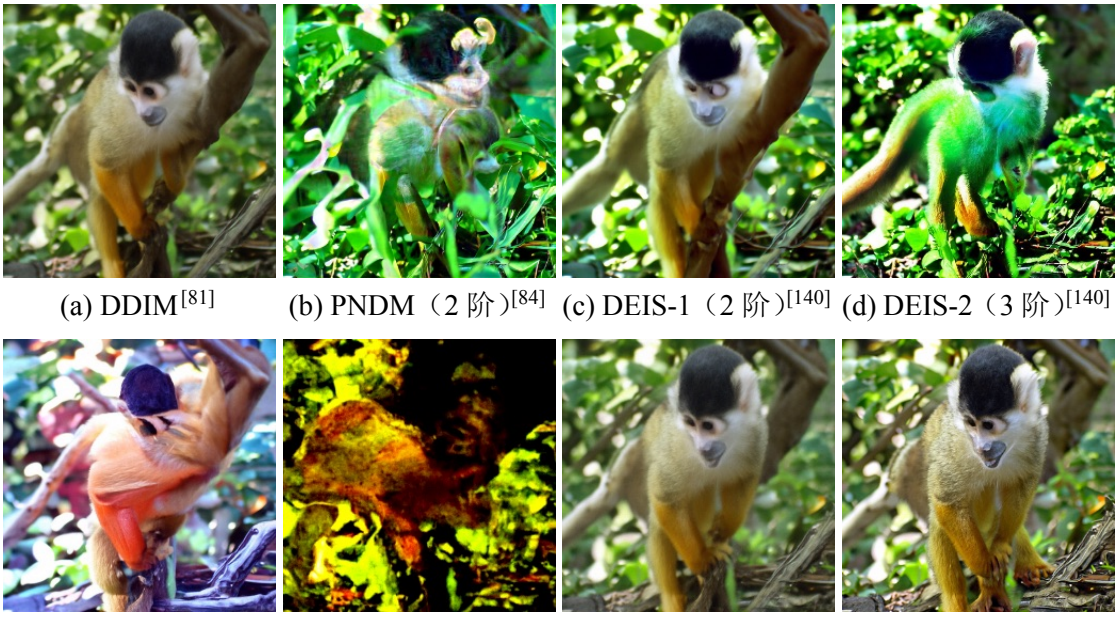
(b) Improved-DDPM[78]

(c) DDPM[44] / SDE-DDIM[81]

(d) SDE-DPM-Solver++

图 6.2　不同 ODE 采样器和 SDE 采样器在 DeepFloyd-IF 模型上的采样结果

练的快速扩散模型采样器，且可以同时适用于扩散常微分方程和扩散随机微分方程。具体而言，本章的一个核心发现是，扩散模型的参数化（parameterization）对采样器的采样质量有着关键的影响。对于引导采样而言，由于引导模型在初始噪声时间的引导至关重要，因此需要重点减少初始时间附近的离散化误差。本章从理论上分析了扩散模型采样的最优参数化形式，并针对由数据预测模型定义的扩散常微分方程和扩散随机微分方程设计了专用的求解器。为了进一步提高生成质



(a) DDIM[81]　(b) PNDM（2 阶）[84]　(c) DEIS-1（2 阶）[140]　(d) DEIS-2（3 阶）[140]

(e) DPM-Solver-2[89]　(f) DPM-Solver-3[89]　(g) †DDIM　(h) DPM-Solver++2M

图 6.3　DPM-Solver++ 与已有采样器在条件扩散模型的引导采样下的结果比较，其中 † 表示采用动态阈值法

量，本章还采用了*动态阈值法*（dynamic thresholding）[73]来缓解训练-测试不匹配的问题。此外，本章还提出一种多步（multi-step）求解器，可以使用更小的步长来解决不稳定性问题。

图 6.1 展示了*隐空间*（latent-space）的文到图模型 Stable Diffusion[15]的采样结果，针对扩散常微分方程设计的 DPM-Solver++ 可以在 10 到 20 步内生成较高质量的样本，远超已有的扩散常微分方程求解器。此外，图 6.2 也展示了所提出的扩散常微分方程求解器（DPM-Solver++）和扩散随机微分方程求解器（SDE-DPM-Solver++）在*像素空间*（pixel-space）的文到图模型 DeepFloyd-IF 上的采样结果，其中 SDE 版本的 DPM-Solver++ 获得了最高的采样质量。

6.2　引导采样时高阶采样器面临的挑战

在提出新的快速求解器之前，本节首先对现有的高阶扩散常微分方程求解器在引导采样时的性能进行了仔细验证，并突出了所面临的挑战。

第一个挑战是*较大的引导尺度会导致高阶求解器不稳定*。如图 6.3 所示，对于较大的引导尺度（数值等于 8.0）和仅仅使用 15 次函数调用，先前的高阶扩散常微分方程求解器[84,89,140]生成的图像质量都较差，且甚至比对应的一阶求解器 DDIM 更差。更为严重的是，求解器的阶数越高，采样质量反而越差。从直觉上来看，较大的引导尺度可能会同时放大模型 $\tilde{\epsilon}_\theta$ 在式（2.39）中的输出和对应的导数。由于

模型的导数影响了微分方程求解器的收敛范围，这样的放大效果可能导致高阶微分方程求解器需要更小的步长来收敛。因此，当步长较大时（采样步数较少时），高阶求解器的性能反而会不如一阶求解器。此外，由于高阶求解器需要近似高阶导数，这些高阶导数通常对放大效果更为敏感，这导致收敛半径的范围被进一步地缩小。

第二个挑战是“训练-测试不匹配”问题[73]。一般而言，训练数据都位于一个有界区间内（例如，图像数据位于 $[-1,1]^d$ 中，其中 d 是数据维度）。然而，较大的引导尺度会将条件噪声预测模型 $\tilde{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t,t,c)$ 推离真实噪声，这反过来使样本（即扩散常微分方程的收敛解 $\boldsymbol{x}_0$）超出了数据所在范围的边界。在这种情况下，生成的图像会变得过饱和且不自然[73]。尽管前人工作提出了*动态阈值法*（dynamic thresholding），但这种方法在少步采样的情形下效果仍不理想。如图 6.3 所示，采用了动态阈值法的一阶 DDIM 在 15 步的采样仍然较为模糊。

为了解决这两个问题，本章将系统地分析引导采样带来的额外问题，并提出一种新的二阶采样算法 DPM-Solver++。如图 6.3 所示，本章所提出的 DPM-Solver++ 可以在 15 步函数调用下获得非常清晰的采样结果，远超所有先前的快速采样算法。

6.3　为引导采样设计无需训练的快速采样器

如第 6.2 节所讨论，先前的高阶求解器对于较大的引导尺度存在不稳定性的问题，以及存在“训练-测试不匹配”问题。为了解决这些问题，本节系统地设计了新型的高阶扩散常微分方程求解器，包括使用数据预测模型来求解扩散常微分方程、设计多步求解器、采用动态阈值法等，从而可实现更快的引导采样。

6.3.1　针对引导采样设计模型的参数化形式

对于引导采样而言，不同时间 t 对应的引导的强弱通常非常不同。Balaji 等人[141]发现，在引导采样时，靠近 $t=T$（即噪声分布对应的时间）的引导模型的*注意力激活*（attention map activation）较大，但靠近 $t=0$（即数据分布对应的时间）的引导模型的注意力激活较弱。此外，$t=T$ 的误差也会在后续所有采样步骤中都造成误差累积。因此，为了让引导采样的数值更稳定，采样器需要对靠近 $t=T$ 附近的求解更为精准。

以下详细分析 $t=T$ 附近扩散模型的参数化形式。首先，根据*特威迪公式*（Tweedie’s formula），有

$$\nabla_{\boldsymbol{x}_t}\log q_t(\boldsymbol{x}_t)=-\frac{\boldsymbol{x}_t-\alpha_t\mathbb{E}_{q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t)}[\boldsymbol{x}_0]}{\sigma_t^2}. \tag{6.1}$$

当 $t \approx T$ 时，$\boldsymbol{x}_0$ 与 $\boldsymbol{x}_t$ 几乎是条件独立的，因此有 $q_{0t}(\boldsymbol{x}_0|\boldsymbol{x}_t) \approx q_0(\boldsymbol{x}_0)$。此时，由于噪声预测模型的全局最优解[45]为 $-\sigma_t \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)$，故噪声预测模型可以被如下方式近似：

$$\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \approx -\sigma_t \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t) \approx \frac{\boldsymbol{x}_t - \alpha_t \mathbb{E}_{q_0(\boldsymbol{x}_0)}[\boldsymbol{x}_0]}{\sigma_t}. \tag{6.2}$$

此时，由于 $\mathbb{E}_{q_0(\boldsymbol{x}_0)}[\boldsymbol{x}_0]$ 与输入 $\boldsymbol{x}_t$ 无关，故噪声预测模型可以被近似为关于 $\boldsymbol{x}_t$ 的一个线性函数。然而，正如第 5.2.1 节中所讨论的，这样额外的线性函数的近似反而会导致更多的累积误差，这也可以解释为什么在第 5 章中针对噪声预测模型设计的 DPM-Solver 无法应用于引导采样。

因此，提高高阶采样器在引导采样中的稳定性，需要将噪声预测模型的参数化形式更改为更稳定的形式。基于以上讨论以及第 5.2.1 节中的结论，可以发现本质上的解决方式是需要将所有的线性项都解析地计算。受式（6.2）启发，将所有线性项减掉后，剩余的项正比于 $\mathbb{E}_{q_0(\boldsymbol{x}_0)}[\boldsymbol{x}_0]$，这恰好对应了数据预测模型。具体而言，当 $t \approx T$ 时，有

$$\boldsymbol{x}_\theta(\boldsymbol{x}_t, t) \approx \frac{\boldsymbol{x}_t + \sigma_t^2 \nabla_{\boldsymbol{x}_t} \log q_t(\boldsymbol{x}_t)}{\alpha_t} \approx \mathbb{E}_{q_0(\boldsymbol{x}_0)}[\boldsymbol{x}_0]. \tag{6.3}$$

换而言之，当 $t \approx T$ 时，数据预测模型 $\boldsymbol{x}_\theta(\boldsymbol{x}_t, t)$ 可以被近似为一个常数。理论上，对这样常数的积分的离散化对应的离散化误差会远小于对线性函数（噪声预测模型）的离散化误差。受此启发，本章对数据预测模型设计扩散模型的专用求解器，并验证了该方法对于引导采样的有效性。

6.3.2　基于数据预测模型设计高阶求解器

本节针对数据预测模型参数化的扩散常微分方程设计专用的快速高阶求解器。具体而言，本节遵循第 5 章中的符号表示。

具体而言，给定时间 T 的初始值 $\boldsymbol{x}_T$ 和从 $t_0 = T$ 递减到 $t_M = 0$ 的 $M+1$ 个时间点 $\{t_i\}_{i=0}^M$，其中 $\tilde{\boldsymbol{x}}_{t_0} = \boldsymbol{x}_T$ 为给定的初始值。求解器的目标是迭代地计算一个序列 $\{\tilde{\boldsymbol{x}}_{t_i}\}_{i=0}^M$ 以近似每个时间 t_i 的精确解，其中最终的值 $\tilde{\boldsymbol{x}}_{t_M}$ 即为扩散常微分方程的近似采样。对 $i = 1, \cdots, M$，记 $h_i \coloneqq \lambda_{t_i} - \lambda_{t_{i-1}}$。此外，为了简便起见，记 $\hat{\boldsymbol{x}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda) \coloneqq \boldsymbol{x}_\theta(\boldsymbol{x}_{t_\lambda(\lambda)}, t_\lambda(\lambda))$ 为数据预测模型 $\boldsymbol{x}_\theta$ 的 λ 的变量替换形式。

式（2.36）给出了关于数据预测模型 $\boldsymbol{x}_\theta$ 的扩散常微分方程的表达式。与第 5.2.1 节中的推导类似，由于该表达式也有半线性的结构，因此可以将该半线性部分解析地计算。再次利用关于 λ 的变量替换公式（change-of-variable formula），可以得到由数据预测模型参数化的扩散常微分方程的精确解，如下述命题所示。

命题 6.1 (由数据预测模型参数化的扩散常微分方程的精确解)： 给定时间 $s > 0$ 的

初始解 $\boldsymbol{x}_s$，式（2.36）中的扩散常微分方程在时间 $t \in [0, s]$ 的解 $\boldsymbol{x}_t$ 为：

$$\boldsymbol{x}_t = \frac{\sigma_t}{\sigma_s}\boldsymbol{x}_s + \sigma_t \int_{\lambda_s}^{\lambda_t} e^{\lambda} \hat{\boldsymbol{x}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda) \mathrm{d}\lambda. \tag{6.4}$$

需要注意的是，由于式（2.35）和式（2.36）中的扩散常微分方程是等价的，因此式（5.4）和式（6.4）中的精确解的公式也是等价的。然而，从设计扩散常微分方程求解器的角度来看，这两个公式是不同的。首先，式（5.4）精确地计算了线性项 $(\alpha_t/\alpha_s)\boldsymbol{x}_s$，而式（6.4）精确地计算了另一个线性项 $(\sigma_t/\sigma_s)\boldsymbol{x}_s$。根据第 6.3.1 节的讨论，本章提出的关于数据预测模型的参数化形式的精确解（式（6.4））几乎把所有的线性项进行了精确计算，对应为 $(\sigma_t/\sigma_s)\boldsymbol{x}_s$，因此进一步减少了线性项带来的离散化误差。其次，为了设计常微分方程求解器，式（5.4）需要近似关于噪声预测模型的指数加权积分 $\int e^{-\lambda}\hat{\boldsymbol{\epsilon}}_\theta \mathrm{d}\lambda$，而式（6.4）需要近似关于数据预测模型的另一个指数加权积分 $\int e^{\lambda}\hat{\boldsymbol{x}}_\theta \mathrm{d}\lambda$，显然这两个积分是不同的（因为 $\boldsymbol{x}_\theta(\boldsymbol{x}_t, t) \coloneqq (\boldsymbol{x}_t - \sigma_t \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t))/\alpha_t$）。因此，基于式（5.4）和式（6.4）设计的高阶求解器是本质不同的。受此启发，以下进一步提出了基于式（6.4）设计扩散常微分方程的高阶求解器的通用方法。

给定时间 t_{i-1} 处的初始值 $\tilde{\boldsymbol{x}}_{t_{i-1}}$，求解器的目标是近似时间 t_i 处的精确解 $\boldsymbol{x}_{t_{i-1}\to t_i}$。记 $\hat{\boldsymbol{x}}_\theta^{(n)}(\lambda) \coloneqq \mathrm{d}^n \hat{\boldsymbol{x}}_\theta(\boldsymbol{x}_\lambda, \lambda)/\mathrm{d}\lambda^n$ 为数据预测模型 $\hat{\boldsymbol{x}}_\theta$ 关于 λ 的 n 阶全导数（total derivative）。与第 5.2.2 节中的推导类似，以下的推导同样基于泰勒展开。具体而言，对于 $k \geqslant 1$，对 $\lambda \in [\lambda_{t_{i-1}}, \lambda_{t_i}]$ 处的 $\hat{\boldsymbol{x}}_\theta$ 关于 $\lambda = \lambda_{t_{i-1}}$ 取 $(k-1)$ 阶泰勒展开，并将其代入式（6.4）中（取 $s = t_{i-1}$ 和 $t = t_i$），有：

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} + \sigma_{t_i}\sum_{n=0}^{k-1} \hat{\boldsymbol{x}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) \int_{\lambda_{t_{i-1}}}^{\lambda_{t_i}} e^{\lambda} \frac{(\lambda - \lambda_{t_{i-1}})^n}{n!} \mathrm{d}\lambda + \mathcal{O}(h_i^{k+1}). \tag{6.5}$$

接着，通过引入第 5.2.2 节中的函数 $\varphi_k(h)$，上式可以被化简为：

$$\boldsymbol{x}_{t_{i-1}\to t_i} = \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} + \alpha_{t_i}\sum_{n=0}^{k-1} h_i^{n+1}\varphi_{n+1}(-h_i)\hat{\boldsymbol{x}}_\theta^{(n)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) + \mathcal{O}(h_i^{k+1}), \tag{6.6}$$

其中 $\varphi_{n+1}(-h_i)$ 可被解析地计算。因此，与第 5.2.2 节中的推导同理，只需要在省略 $\mathcal{O}(h_i^{k+1})$ 的高阶误差项后用有限差分（finite difference）近似 n 阶导数 $\hat{\boldsymbol{x}}_\theta^{(n)}(\lambda_{t_{i-1}})$ 即可。本章将这样设计的求解器命名为 *DPM-Solver++*。值得注意的是，当 $k = 1$ 时，DPM-Solver++ 与第 5 章中的 DPM-Solver 完全相同，且都等价于 DDIM[81]。这将在第 6.5 节中详细讨论。

而当 $k \geqslant 2$ 时，一种做法是使用与第 5 章中 DPM-Solver-2 相同的技术来估计导数 $\hat{\boldsymbol{x}}_\theta^{(1)}(\hat{\boldsymbol{x}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}})$，即在 t_{i-1} 和 t_i 之间引入一个额外的中间时间步 s_i，并组合 s_i 和 t_{i-1} 的函数值来近似导数，这也被称为单步（single-step）求解器[132]。这种求解器共需要 $2M+1$ 个时间点（$\{t_i\}_{i=0}^M$ 和 $\{s_i\}_{i=1}^M$），满足 $t_0 > s_1 > t_1 > \cdots > t_{M-1} > s_M > t_M$。

算法 6.1 DPM-Solver++(2S) 算法流程

输入：时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^M$ 和 $\{s_i\}_{i=1}^M$，数据预测模型 $\boldsymbol{x}_\theta$
输出：扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$
for $i \leftarrow 1$ to M **do**
　$h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
　$r_i \leftarrow \frac{\lambda_{s_i} - \lambda_{t_{i-1}}}{h_i}$
　$\boldsymbol{u}_i \leftarrow \frac{\sigma_{s_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{s_i}\left(e^{-r_i h_i} - 1\right)\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\left(e^{-h_i} - 1\right)\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \alpha_{t_i}\left(e^{-h_i} - 1\right)\frac{\boldsymbol{x}_\theta(\boldsymbol{u}_i, s_i) - \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})}{2r_i}$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

算法 6.2 DPM-Solver++(2M) 算法流程

输入：时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^M$，数据预测模型 $\boldsymbol{x}_\theta$。
输出：扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

对于 $i = 1, \cdots, M$，令 $h_i \coloneqq \lambda_{t_i} - \lambda_{t_{i-1}}$
$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$，并初始化一个缓冲队列 Q
$Q \xleftarrow{\text{buffer}} \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_0}, t_0)$
$\tilde{\boldsymbol{x}}_{t_1} \leftarrow \frac{\sigma_{t_1}}{\sigma_{t_0}}\tilde{\boldsymbol{x}}_0 - \alpha_{t_1}\left(e^{-h_1} - 1\right)\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_0}, t_0)$
$Q \xleftarrow{\text{buffer}} \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_1}, t_1)$
for $i \leftarrow 2$ to M **do**
　$r_i \leftarrow \frac{h_{i-1}}{h_i}$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}} - \alpha_{t_i}\left(e^{-h_i} - 1\right)\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \alpha_{t_i}\left(e^{-h_i} - 1\right)\frac{\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-2}}, t_{i-2})}{2r_i}$
　若 $i < M$，则 $Q \xleftarrow{\text{buffer}} \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

具体的流程见算法 6.1，其中每一步将时间 t_{i-1} 处的值 $\tilde{\boldsymbol{x}}_{t_{i-1}}$ 与时间 s_i 处的中间值 $\boldsymbol{u}_i$ 结合来计算时间 t_i 处的值 $\tilde{\boldsymbol{x}}_{t_i}$。本章将该算法命名为 DPM-Solver++(2S)，意味着所提出的求解器是一种二阶单步方法。该算法的收敛性的证明与第 5 章中的证明一致，因此本章不再赘述。此外，对于 $k \geqslant 3$，如第 6.2 节所讨论的，高阶求解器可能不适合较大的引导尺度，因此本章只考虑 $k = 2$。

6.3.3　基于多步法的求解器

第 6.3.2 节提出了基于单步法的二阶求解器，该求解器的每一步（从 t_{i-1} 到 t_i）需要神经网络 $\boldsymbol{x}_\theta$ 的两次函数调用，而且中间值 $\boldsymbol{u}_i$ 只使用一次就被丢弃。这种方法丢失了先前的信息，因而效果不一定是最优的。为了进一步提高求解器的效果，本节提出了基于多步法（multi-step method）的二阶扩散常微分方程求解器，其每

(a) DPM-Solver-2
（$\boldsymbol{\epsilon}_\theta$ 参数化，单步法）

(b) DPM-Solver++(2S)
（$\boldsymbol{x}_\theta$ 参数化，单步法）

(c) DPM-Solver++(2M)
（$\boldsymbol{x}_\theta$ 参数化，多步法，阈值法）

图 6.4　在 ImageNet 256 × 256 数据集上使用 8.0 的引导尺度时不同采样算法的采样结果

一步都会用到先前的计算结果。

具体而言，对于 $n \geqslant 1$，为了近似式（6.6）中的导数 $\hat{\boldsymbol{x}}_\theta^{(n)}$，有另一种主流方法[132]：*多步法*。给定时间 t_{i-1} 处的先前值 $\{\tilde{\boldsymbol{x}}_{t_j}\}_{j=0}^{i-1}$，多步法重复使用了先前值来近似高阶导数，因而每一步只需要一次函数调用。从经验上看，多步法的确比单步方法更高效，特别是对于有限数量的函数评估[132]。

本节将多步求解器的设计技巧与式（6.5）中的泰勒展开相结合，进一步提出了一种用于数据预测模型 $\boldsymbol{x}_\theta$ 参数化的扩散常微分方程的多步二阶求解器，如算法 6.2 所述。其中，该算法为先前值维护了一个缓冲区，每一步需要结合先前的值 $\tilde{\boldsymbol{x}}_{t_{i-1}}$ 和 $\tilde{\boldsymbol{x}}_{t_{i-2}}$ 来计算值 $\tilde{\boldsymbol{x}}_{t_i}$，而无需额外的中间值。该算法被命名为 DPM-Solver++(2M)，意味着所提出的求解器是一种多步二阶求解器。

多步法的一个优势是，对于固定的采样步数，多步法每一步的步长会更小，因而离散化损失可能会更小。具体而言，对于固定的 N 次函数调用总次数，多步法可以使用 $M = N$ 步，而 k 阶单步方法最多只能使用 $M = N/k$ 步。因此，多步法的每一步大小 h_i 约为单步方法的 $1/k$，所以多步法在式（6.5）中的高阶误差项 $\mathcal{O}(h_i^k)$ 可能小于单步方法对应的步长。第 6.6.1 节的定量实验结果也表明，多步法的 DPM-Solver++ 略优于单步法的 DPM-Solver++。

6.3.4　将阈值法与高阶求解器结合

对于有界数据（例如图像数据）的分布，阈值法[44,73]可以将越界样本限制至训练数据的范围内，从而一定程度上减少由较大引导尺度带来的不利影响[73]。具体而言，阈值法通过逐元素地将数据预测模型 $\boldsymbol{x}_\theta$ 削减在数据范围内，从而定义了一个被削减后的数据预测模型。由于 DPM-Solver++ 是为数据预测模型 $\boldsymbol{x}_\theta$ 专属设计的，因此可以直接与阈值法结合。

综合以上所提的方法，图 6.4 展示了一个在 ImageNet 256 × 256 上使用 15 步

算法 6.3 SDE-DPM-Solver++1 算法流程

输入：时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^{M}$，数据预测模型 $\boldsymbol{x}_\theta$。
输出：扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$
for $i \leftarrow 1$ to M **do**
　$h_i \leftarrow \lambda_{t_i} - \lambda_{t_{i-1}}$
　$\boldsymbol{z}_{t_i} \sim \mathcal{N}(0, \boldsymbol{I})$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}} e^{-h_i} \tilde{\boldsymbol{x}}_{i-1} + \alpha_{t_i}\left(1 - e^{-2h_i}\right) \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) + \sigma_{t_i}\sqrt{1 - e^{-2h_i}}\, z_{t_i}$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

算法 6.4 SDE-DPM-Solver++(2M) 算法流程

输入：时间 T 的初始值 $\boldsymbol{x}_T$，中间时间点 $\{t_i\}_{i=0}^{M}$，数据预测模型 $\boldsymbol{x}_\theta$。
输出：扩散常微分方程的近似采样 $\tilde{\boldsymbol{x}}_{t_M}$

对于 $i = 1, \cdots, M$，令 $h_i \coloneqq \lambda_{t_i} - \lambda_{t_{i-1}}$
$\tilde{\boldsymbol{x}}_{t_0} \leftarrow \boldsymbol{x}_T$，并初始化一个缓冲队列 Q
$Q \xleftarrow{\text{添加缓冲}} \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_0}, t_0)$
$\boldsymbol{z}_{t_1} \sim \mathcal{N}(0, \boldsymbol{I})$
$\tilde{\boldsymbol{x}}_{t_1} \leftarrow \frac{\sigma_{t_1}}{\sigma_{t_0}} e^{-h_1} \tilde{\boldsymbol{x}}_0 + \alpha_{t_1}\left(1 - e^{-2h_1}\right) \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_0}, t_0) + \sigma_{t_1}\sqrt{1 - e^{-2h_1}}\, z_{t_1}$
$Q \xleftarrow{\text{添加缓冲}} \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_1}, t_1)$
for $i \leftarrow 2$ to M **do**
　$r_i \leftarrow \frac{h_{i-1}}{h_i}$
　$\boldsymbol{z}_{t_i} \sim \mathcal{N}(0, \boldsymbol{I})$
　$\tilde{\boldsymbol{x}}_{t_i} \leftarrow \frac{\sigma_{t_i}}{\sigma_{t_{i-1}}} e^{-h_i} \tilde{\boldsymbol{x}}_{t_{i-1}} + \alpha_{t_i}\left(1 - e^{-2h_i}\right) \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1})$
　　$+ \alpha_{t_i}\left(1 - e^{-2h_i}\right) \frac{\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) - \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-2}}, t_{i-2})}{2r_i} + \sigma_{t_i}\sqrt{1 - e^{-2h_i}}\, z_{t_i}$
　若 $i < M$，则 $Q \xleftarrow{\text{添加缓冲}} \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_i}, t_i)$
end for
return $\tilde{\boldsymbol{x}}_{t_M}$

求解器的采样结果示例。可以看到，分别改变参数化形式（从 $\boldsymbol{\epsilon}_\theta$ 参数化到 $\boldsymbol{x}_\theta$ 参数化）和应用多步法与阈值法，DPM-Solver++ 仅仅在 15 步就可以获得较高的采样质量。

6.4　扩散随机微分方程的快速求解器

对于引导采样（条件扩散模型）而言，基于扩散随机微分方程的采样方法在某些情况下可以获得比基于扩散常微分方程采样质量更好的结果。如图 6.2 所示，尽管基于扩散常微分方程的采样方法可以在较少步数下收敛，但其收敛后的图像质量可能较差；相反，前人针对扩散随机微分方程的采样方法在步数较少时可能

较差，但步数较多时的采样结果可能有较高的质量。为了在尽量少的步数下获得较高质量的采样结果，本节进一步为扩散随机微分方程设计快速求解器。

具体而言，令 $\mathrm{d}\hat{\boldsymbol{w}}_\lambda \coloneqq \sqrt{-(\mathrm{d}\lambda_t/\mathrm{d}t)}\mathrm{d}\bar{\boldsymbol{w}}_{t_\lambda(\lambda)}$ 为扩散模型中的*维纳过程*（Wiener process）关于 λ 的换元。为简要起见，令 $\hat{\alpha}_\lambda \coloneqq \alpha_{t_\lambda(\lambda)}$ 和 $\hat{\sigma}_\lambda \coloneqq \sigma_{t_\lambda(\lambda)}$ 为 α_t 和 σ_t 关于 λ 的换元。对于常用的扩散模型，即“*方差保留*”（variance preserving，VP）类型，有 $\hat{\alpha}_\lambda^2 + \hat{\sigma}_\lambda^2 = 1$，此时有

$$\begin{aligned}\frac{\mathrm{d}\log\hat{\alpha}_\lambda}{\mathrm{d}\lambda} &= \hat{\sigma}_\lambda^2,\\ \frac{\mathrm{d}\log\hat{\sigma}_\lambda}{\mathrm{d}\lambda} &= -\hat{\alpha}_\lambda^2.\end{aligned} \tag{6.7}$$

由于扩散模型的系数满足 $f(t) = \mathrm{d}\log\alpha_t/\mathrm{d}t$ 以及 $g(t) = \sigma_t\sqrt{-2(\mathrm{d}\lambda_t/\mathrm{d}t)}$（详见第 2.2 节），因此扩散随机微分方程关于 λ 换元后的随机微分方程为：

$$\mathrm{d}\hat{\boldsymbol{x}}_\lambda = [-(1+\hat{\alpha}_\lambda^2)\hat{\boldsymbol{x}}_\lambda + 2\hat{\alpha}_\lambda\hat{\boldsymbol{x}}_\theta(\hat{\boldsymbol{x}}_\lambda,\lambda)]\mathrm{d}\lambda + \sqrt{2}\hat{\sigma}_\lambda\mathrm{d}\hat{\boldsymbol{w}}_\lambda. \tag{6.8}$$

与第 5.2.1 节同理，通过应用*常数变易定理*（variation-of-constants formula），以下命题给出了扩散随机微分方程的精确解的表达式。

命题 6.2 (由数据预测模型参数化的扩散随机微分方程的精确解)： 给定时间 $s > 0$ 的初始解 $\boldsymbol{x}_s$，式（6.8）中的扩散随机微分方程在时间 $t \in [0, s]$ 的解 $\boldsymbol{x}_t$ 为：

$$\boldsymbol{x}_t = \frac{\sigma_t}{\sigma_s}e^{-(\lambda_t-\lambda_s)}\boldsymbol{x}_s + 2\alpha_t\int_{\lambda_s}^{\lambda_t} e^{2(\lambda-\lambda_t)}\hat{\boldsymbol{x}}_\theta(\hat{\boldsymbol{x}}_\lambda,\lambda)\mathrm{d}\lambda + \sqrt{2}\sigma_t\int_{\lambda_s}^{\lambda_t} e^{\lambda-\lambda_t}\mathrm{d}\hat{\boldsymbol{w}}_\lambda. \tag{6.9}$$

根据上述命题，以下具体推导出针对扩散随机微分方程的求解器。首先，关于 $\hat{\boldsymbol{w}}_\lambda$ 的积分属于*伊藤积分*（Itô integral），它可以被如下等价地计算：

$$\int_{\lambda_s}^{\lambda_t} e^{\lambda-\lambda_t}\mathrm{d}\boldsymbol{w}_\lambda = \left(\sqrt{\int_{\lambda_s}^{\lambda_t} e^{2(\lambda-\lambda_t)}\mathrm{d}\lambda}\right)\boldsymbol{z}_t = \frac{1}{\sqrt{2}}\sqrt{1-e^{-2(\lambda_t-\lambda_s)}}\boldsymbol{z}_t, \tag{6.10}$$

其中 $\boldsymbol{z}_t \sim \mathcal{N}(0, \boldsymbol{I})$ 为服从标准高斯分布的噪声。因此，为了离散化式（6.9），只需要离散化关于 $\hat{\boldsymbol{x}}_\theta$ 的指数加权积分即可。为简便起见，记 $h \coloneqq \lambda_t - \lambda_s$。通过应用第 6.3.2 节类似的推导，可以得到以下的一阶和二阶求解器。由于这些求解器是为扩散随机微分方程设计的，因此分别将一阶和二阶求解器命名为 *SDE-DPM-Solver++1* 和 *SDE-DPM-Solver++2M*。

SDE-DPM-Solver++1　令 $\boldsymbol{z}_t \sim \mathcal{N}(0, \boldsymbol{I})$。通过假设 $\hat{\boldsymbol{x}}_\theta(\hat{\boldsymbol{x}}_\lambda,\lambda) \approx \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_s, s)$，有

$$\tilde{\boldsymbol{x}}_t = \frac{\sigma_t}{\sigma_s}e^{-h}\tilde{\boldsymbol{x}}_s + \alpha_t(1-e^{-2h})\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_s, s) + \sigma_t\sqrt{1-e^{-2h}}\boldsymbol{z}_t. \tag{6.11}$$

表 6.1　基于指数积分器的高阶扩散模型求解器之间的比较

采样算法	一阶特例	参数化	泰勒展开	求解器类型	解析形式
DEIS[140]	DDIM ($\eta=0$)	$\boldsymbol{\epsilon}_\theta$	$\boldsymbol{\epsilon}_\theta$ 对 t	多步法	×
DPM-Solver[89]	DDIM ($\eta=0$)	$\boldsymbol{\epsilon}_\theta$	$\hat{\boldsymbol{\epsilon}}_\theta$ 对 λ	单步法	✓
DPM-Solver++	DDIM ($\eta=0$)	$\boldsymbol{x}_\theta$	$\hat{\boldsymbol{x}}_\theta$ 对 λ	单、多步法	✓
SDE-DPM-Solver++	DDIM ($\eta=1$)	$\boldsymbol{x}_\theta$	$\hat{\boldsymbol{x}}_\theta$ 对 λ	多步法	✓

SDE-DPM-Solver++(2M)　令 $\boldsymbol{z}_t \sim \mathcal{N}(0, \boldsymbol{I})$。假设已有在时间 $r < t$ 处的近似值 $\tilde{\boldsymbol{x}}_r$ 和 $\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_r, r)$。记 $r_1 \coloneqq (\lambda_r - \lambda_s)/h$。通过假设 $\hat{\boldsymbol{x}}_\theta(\hat{\boldsymbol{x}}_\lambda, \lambda) \approx \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_s, s) + ((\lambda-\lambda_s)/r_1 h)(\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_r, r) - \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_s, s))$，有

$$\begin{aligned}\tilde{\boldsymbol{x}}_t = &\frac{\sigma_t}{\sigma_s}e^{-h}\tilde{\boldsymbol{x}}_s + \alpha_t(1-e^{-2h})\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_s, s)\\ &+ \alpha_t(1-e^{-2h})\left(\frac{\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_r, r) - \boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_s, s)}{2r_1}\right) + \sigma_t\sqrt{1-e^{-2h}}\boldsymbol{z}_t.\end{aligned} \tag{6.12}$$

迭代执行以上算法，即可获得最终的采样。全部的算法流程见算法 6.3 和 6.4。

6.5　与已有针对引导采样的快速采样算法比较

本质上，所有用于扩散模型的无需训练的采样方法都可以理解为离散化对应的扩散随机微分方程[44-45,82-83,86,139,142]或扩散常微分方程[45,81,84,89,140]。基于第 5.3 节的讨论，本节进一步讨论 DPM-Solver++ 与其它现有算法的联系和区别。

与基于指数积分器的求解器的比较。　本节首先讨论一般形式的 DDIM[81]与 DPM-Solver++ 的关系。令 $\tilde{\sigma}_{t_i}(\eta) \coloneqq \eta\sigma_{t_i}\sqrt{1-e^{-2h_i}}$，那么对于 $\eta \geqslant 0$，一般形式的 DDIM 的更新公式为：

$$\tilde{\boldsymbol{x}}_{t_i} = \alpha_{t_i}\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) + \sqrt{\sigma_{t_i}^2 - \tilde{\sigma}_{t_i}^2(\eta)}\boldsymbol{\epsilon}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}}, t_{i-1}) + \tilde{\sigma}_{t_i}(\eta)\boldsymbol{z}_{t_i}. \tag{6.13}$$

已有最先进的快速扩散常微分方程求解器[89,140]都利用指数积分器（exponential integrators）来求解由噪声预测模型 $\boldsymbol{\epsilon}_\theta$ 参数化的扩散常微分方程。换言之，这些求解器本质上都在近似第 5 章中式（5.4）中的精确解，并包括 $\eta=0$ 的 DDIM[81]作为一阶特例。更进一步，以下证明 DPM-Solver++ 的一阶特例也是 $\eta=0$ 的 DDIM，且 SDE-DPM-Solver++ 的一阶特例恰好为 $\eta=1$ 的 DDIM（也等价于原始 DDPM[44]）。

对于 $k=1$，式（6.6）在忽略高阶误差项 $\mathcal{O}(h_i^{k+1})$ 后等价于：

$$\tilde{\boldsymbol{x}}_{t_i}=\frac{\sigma_{t_i}}{\sigma_{t_{i-1}}}\tilde{\boldsymbol{x}}_{t_{i-1}}-\alpha_{t_i}(e^{-h_i}-1)\boldsymbol{x}_\theta(\tilde{\boldsymbol{x}}_{t_{i-1}},t_{i-1}). \tag{6.14}$$

由于 $\boldsymbol{x}_\theta(\boldsymbol{x}_t,t)=(\boldsymbol{x}_t-\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t))/\alpha_t$，容易验证上式等价于式（5.12）中的 DPM-Solver-1。此外，也容易验证式（6.11）等价于式（6.13）中 $\eta=1$ 时的情形。

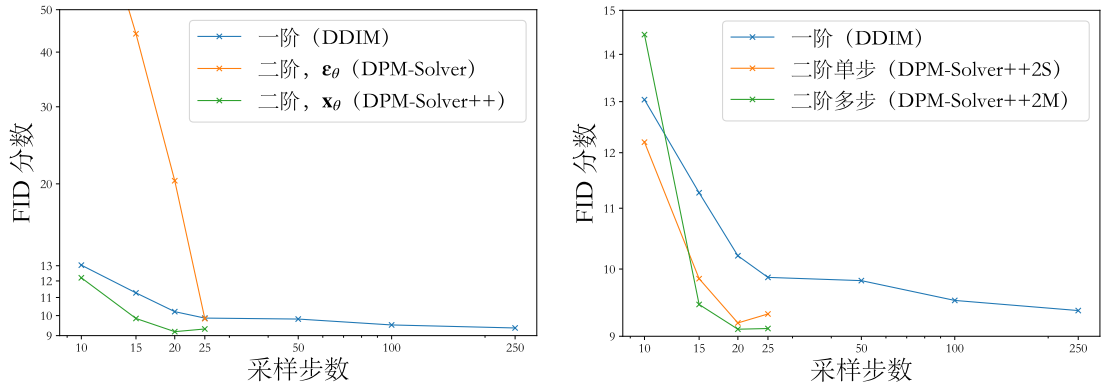
6.5 节 中列出了基于指数积分器的已有高阶求解器和 DPM-Solver++ 之间的区别。尽管这些求解器的一阶版本是等价的，但高阶版本却并不相同。由于 DPM-Solver++ 可以几乎将所有线性项都解析地计算，因此 DPM-Solver++ 和 SDE-DPM-Solver++ 在理论上都优于已有的求解器，且可以视为对 DDIM 的直接拓展。

其它快速采样方法。　基于扩散随机微分方程的采样器[44-45,82-83,86,139,142]通常需要比基于扩散常微分方程的采样器[89]更多的步骤来收敛，因为扩散随机微分方程引入了更多的随机性，使得去噪更为困难。基于额外训练的采样器包括模型蒸馏[75-76]、学习逆过程方差[77-78,135]和学习采样步骤[79-80]。然而，基于训练的采样器很难扩展到预训练的大型扩散模型[15,18,73]。通过将原始扩散模型修改到隐空间[136]或引入动量[137]也可以得到理论上收敛更快的扩散模型。另外，将扩散模型与生成对抗网络结合[138,143]也可以提高生成对抗网络的采样质量和扩散模型的采样速度。

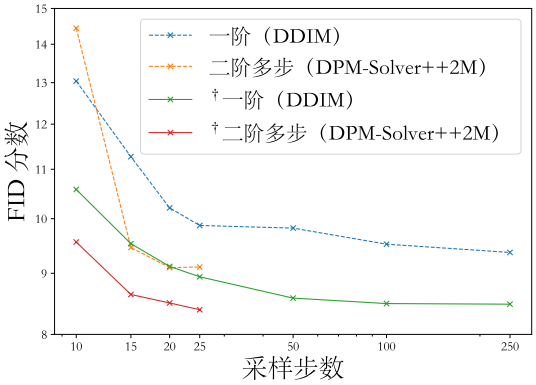
6.6　实验结果

本节从实验上验证 DPM-Solver++ 对于条件扩散模型的加速效果，包括像素空间扩散模型和隐空间扩散模型的引导采样。所有实验通过改变不同数量的采样步数，即调用模型 $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t,t,c)$ 或 $\boldsymbol{x}_\theta(\boldsymbol{x}_t,t,c)$ 的次数，并将 DPM-Solver++ 与先前最先进的扩散模型快速采样器进行比较，包括第 5 章提出的 DPM-Solver[89]，DEIS[140]，PNDM[84]和 DDIM[81]。对于离散时间扩散模型，实验同样采用了第 5.2.3 节中的实验设置。此外，由于先前的求解器没有测试其在引导采样中的性能，本节实验还通过对不同的步长调度策略（即时间步 $\{t_i\}_{i=0}^M$ 的选择）和求解器阶数仔细搜索，来确保基线采样器的最佳性能。经过搜索后有以下发现：

- 对于步长调度，实验搜索了如下策略：对 t 的均匀步长[44-45]、对 λ 的均匀步长[89]、对 t 的指数幂的均匀步长[140]。实验发现，对于引导采样而言，对 t 的均匀步长所得的结果最好。
- 对于较大的引导尺度，所有先前的高阶采样器中表现最好的采样器的阶数都是二阶。



(a) ϵ_θ 与 x_θ 参数化的区别　　(b) 单步法与多步法的区别



(c) 动态阈值法的效果

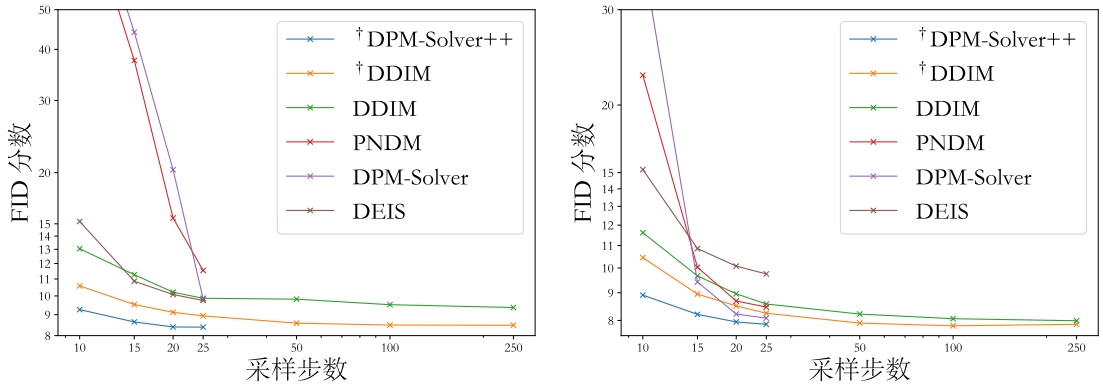
图 6.5　DPM-Solver++ 相比于 DPM-Solver 的具体改进对采样质量（FID 分数）的影响

因此，所有高阶的基准线方法都采用了对 t 的均匀步长的二阶采样器作为最优的基准线结果。

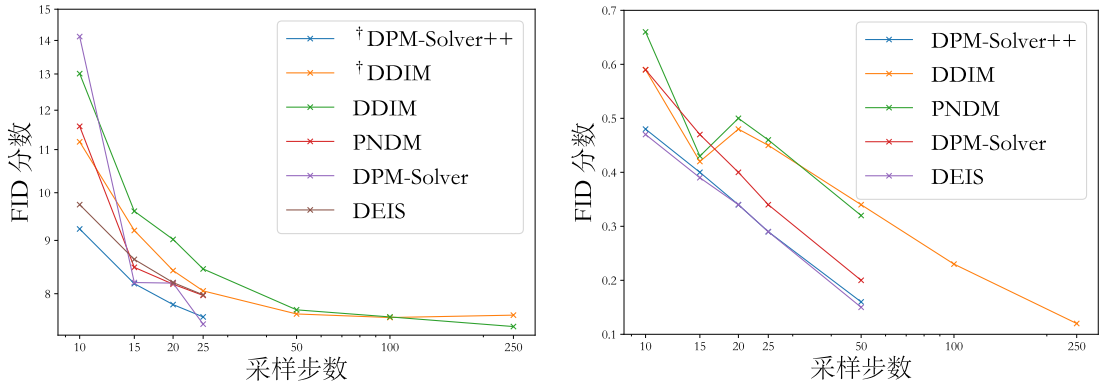
6.6.1　像素空间条件扩散模型的实验结果

实验首先将 DPM-Solver++ 与其它采样器在分类器引导（classifier guidance）的引导采样下进行比较，使用的模型是 ImageNet 256×256 数据集上预训练的扩散模型[49]。具体而言，实验通过绘制一万个采样样本并计算 FID 分数[129]来衡量采样质量，其中较低的 FID 分数通常意味着更好的采样质量。此外，实验还额外比较了 DDIM 和 DPM-Solver++ 在采用动态阈值法[73]时的采样质量。通过改变引导尺度 s 的值（分别为 8.0、4.0 和 2.0），最终的实验结果如图 6.6 (a-c) 所示。实验发现，对于较大的引导尺度，所有先前的高阶采样器（DEIS、PNDM 和 DPM-Solver）的收敛速度都比一阶 DDIM 更慢，这表明先前的高阶采样器不稳定。相反，DPM-Solver++ 在大引导尺度和小引导尺度上都获得了最佳的加速性能。尤其是对于大引导尺度，DPM-Solver++ 几乎只在 15 步内即可收敛，其生成质量与其余采样器 100 步采样所得结果一致。这样的实验结果说明了 DPM-Solver++ 在引导采样中的有效性。

作为消融实验，本节还比较了单步 DPM-Solver-2、单步 DPM-Solver++(2S) 和



(a) 像素空间（$s = 8.0$）　(b) 像素空间（$s = 4.0$）



(c) 像素空间（$s = 2.0$）　(d) 隐空间（$s = 7.5$）

图 6.6　像素空间和隐空间的条件扩散模型在不同求解器下的采样质量

多步 DPM-Solver++(2M)，以验证所提方法的有效性。具体而言，实验使用较大的引导尺度（$s = 8.0$），并进行以下对比实验：

- 比较 $\boldsymbol{\epsilon}_\theta$ 与 $\boldsymbol{x}_\theta$ 参数化的区别：如图 6.5 (a) 所示，通过将扩散常微分方程参数化从 $\boldsymbol{\epsilon}_\theta$ 换成 $\boldsymbol{x}_\theta$（即从 DPM-Solver-2 到 DPM-Solver++(2S) 并进行采样），DPM-Solver++(2S) 可以获得稳定的相对于一阶 DDIM 的加速效果，但原始的 DPM-Solver 比 DDIM 更差。因此，针对 $\boldsymbol{x}_\theta$ 设计的高阶求解器更适合引导采样。
- 比较单步法与多步法的区别：如图 6.5 (b) 所示，多步法的 DPM-Solver++ 比单步法的 DPM-Solver++ 的收敛速度稍快，并且可以在 15 到 20 步内完全收敛。因此，对于引导采样而言，多步法的离散化误差更小，收敛速度更快。
- 动态阈值法带来的效果：图 6.5 (c) 展示了动态阈值法对采样质量的影响，其中 † 表示使用了动态阈值法。需要注意的是，动态阈值法改变了数据预测模型 $\boldsymbol{x}_\theta$，因此改变了扩散常微分方程的收敛解。首先，实验发现通过应用动态阈值法，扩散常微分方程可以得到更高质量的采样结果，这也与前人工作[73]的结论一致；其次，应用了动态阈值法的 DPM-Solver++ 显著优于应用了动态阈值法的 DDIM 和没有应用动态阈值法的 DPM-Solver++，这也说明

了 DPM-Solver++ 与动态阈值法高度适配。

此外，这样的定量结果也与图 6.4 中展示的定性结果一致。

6.6.2　隐空间条件扩散模型的实验结果

本节进一步对 DPM-Solver++ 在隐空间扩散模型[15]上的加速性能进行验证。具体而言，实验基于开源社区最受欢迎的文到图扩散模型 Stable Diffusion[15]，且使用官方代码中默认的引导尺度 $s = 7.5$。隐空间扩散模型通过训练一对*编码器*（encoder）和*解码器*（decoder）将图像空间与隐空间关联起来，然后为隐空间的数据训练一个扩散模型。由于隐空间是无界的（一般在隐空间取高斯分布为先验），因此隐空间扩散模型无法应用动态阈值法。实验比较了 DPM-Solver++ 与不同求解器在隐空间的收敛速度，如图 6.6 (d) 所示。

具体而言，实验随机抽取了 MS-COCO-2014 数据集中验证集里的一万个标题-图像对，并使用标题作为条件从预训练的 Stable Diffusion 模型中采样一万个图像，其中每个标题只采样一个图像样本（这遵循了 Rombach 等人[15], Nichol 等人[72]的标准评估流程）。实验发现，所有求解器在仅仅 10 步内就可以达到大约 15.0 至 16.04 的 FID 分数，这是因为 Stable Diffusion 的强大的预训练解码器可以将未收敛的隐空间数据映射到一个好的图像样本，尽管它与收敛的图像相距甚远。因此，FID 分数难以衡量隐空间扩散模型的收敛速度。

为了解决这个问题，验证实验需要对隐空间扩散模型设计一个专用的评测标准。由于不同的扩散常微分方程求解器直接影响了隐空间上的收敛速度，而隐空间的预训练范式是基于高斯分布（对应于隐空间的 L_2 范数），因此为了进一步比较隐空间扩散模型的不同采样器，本节实验根据隐空间上采样的 $\boldsymbol{x}_0$ 和真实解 $\boldsymbol{x}_0^*$ 之间的 L_2 范数 $\|\boldsymbol{x}_0 - \boldsymbol{x}_0^*\|_2/\sqrt{D}$ 直接比较不同的求解器。具体而言，实验首先从标准正态分布中采样一万个噪声变量并固定，接着使用不同的扩散模型采样器从这一万个固定噪声变量开始采样一万个隐变量。由于所有这些求解器都可以理解为离散化扩散常微分方程，因此实验通过 999 步 DDIM 得到的真实解 $\boldsymbol{x}_0^*$ 与不同采样器在不同采样步数下的采样结果 $\boldsymbol{x}_0$ 比较并计算 L_2 范数，结果如图 6.6 (d) 所示。可以发现，Stable Diffusion 中支持的快速采样器（DDIM 和 PNDM）比 DPM-Solver++ 和 DEIS 收敛慢得多。此外，二阶多步 DPM-Solver++ 和 DEIS 在隐空间上实现了非常接近的加速。由于 Stable Diffusion 默认使用 50 步的 PNDM，而 DPM-Solver++ 只需 15 至 20 步就可以实现类似的收敛误差，因此 DPM-Solver++ 可以对 Stable Diffusion 达到非常快的加速性能。图 6.1 也展示了不同求解器对于 Stable Diffusion 的加速效果。通过对采样图像的定性比较，可以发现 DPM-Solver++ 确实可以在仅 15 至 20 步内生成相当不错的图像样本。

6.7　本章小结

本章对条件扩散模型的引导采样的加速问题进行了深入研究。理论分析和实验验证表明，先前的基于噪声预测模型的高阶求解器在较大引导尺度的引导采样中存在不稳定的问题，其生成质量甚至比一阶求解器 DDIM 更差。为了解决该问题并加速引导采样，本章提出了 DPM-Solver++，一种无需训练的快速扩散模型采样器，同时适用于加速扩散常微分方程和扩散随机微分方程的引导采样。DPM-Solver++ 基于数据预测模型对应的参数化形式，并可以直接采用多步法和阈值法进一步稳定采样过程。实验结果显示，基于 DPM-Solver++ 的条件扩散模型可以在很少步数下生成高质量的样本，并且几乎只需 15 到 20 步就可以收敛。

第 7 章　总结与展望

7.1　本文总结

为了解决可逆生成模型在表达能力、训练难度和推断速度上存在的关键基本问题，本文系统地针对可逆生成模型及其训练与推断算法进行了研究与开发，取得的主要创新性成果如下：

第 3 章提出了*隐式标准化流模型*，通过非线性方程的根隐式地定义可逆生成模型，提高了离散时间标准化流模型的表达能力，在同等参数下的密度估计性能显著优于已有模型。

第 4 章提出了*扩散常微分方程最大似然训练的高效算法*，通过分析扩散常微分方程的最大似然训练与分数匹配的关系，证明了原有一阶去噪分数匹配算法不足以保证扩散常微分方程的最大似然训练，并进一步提出一种误差可控的高阶去噪分数匹配算法，使得扩散常微分方程以及连续时间的标准化流模型的最大似然训练可以被高效地实现。

第 5 章提出了*无条件扩散模型的高效采样算法*，通过解析地计算扩散常微分方程的线性项及对应的系数，提出了一种专为扩散常微分方程设计的具有收敛阶数保证的高阶求解器，在无需训练的前提下将无条件扩散模型的采样加速了 4 到 16 倍。

第 6 章提出了*条件扩散模型的高效采样算法*，通过分析引导采样的不稳定性的因素，提出了一种专为引导采样设计的高阶采样器，该采样器同时适用于扩散随机微分方程和扩散常微分方程，在无需训练的前提下极大地提升了条件扩散模型的采样稳定性和采样速度，并可用于大规模的多模态扩散模型的加速采样。

7.2　未来工作展望

本文系统地研究了可逆生成模型在表达能力、训练难度和推断速度上存在的问题，分别针对其表达能力不足、训练开销昂贵、推断速度极慢等问题，设计更强表达能力的可逆生成模型，开发高效的训练和推断算法，一定程度上解决了上述问题，取得了阶段性成果。但是可逆生成模型的建模、训练与推断等方面还有很多极具挑战性的问题待解决，具体可以概括为以下三方面：

- **对离散数据（如文本）的高效建模：**目前，可逆生成模型在大规模的高维连续数据建模任务中取得了一系列重大进展，其建模效果显著优于其它类型的

深度生成模型。然而，由于可逆生成模型依赖于实数空间的可逆映射，其无法直接应用于离散数据类型例如文本数据。尽管已有工作针对文本数据单独设计了离散类型的扩散模型[144-145]或隐空间编码器[146]，但可逆生成模型的建模效果仍然弱于自回归模型[147-148]。由于文本数据是多模态大模型的核心模态，如何基于可逆生成模型高效地建模文本数据将是未来多模态大模型研究的核心问题之一，也是本文后续工作的一大重点。

- **建立表达能力强、训练稳定、推断快速的深度生成模型：**由于扩散模型和自回归模型都有表达能力强、训练稳定的优势，其在文本、图像、视频、音频等诸多任务中已经展现了惊人的应用价值。对于自回归模型而言，串行推断是该类模型本质的特性，因此长文本的缓慢推断速度和昂贵开销是不可避免的问题；对于扩散模型而言，尽管本文提出的 DPM-Solver 和 DPM-Solver++ 可以在仅仅 10 步到 20 步之间完成扩散模型的高效采样，但当采样步数进一步减少时，所有已有方法的采样质量都急剧下降。因此，当前深度生成模型研究的最本质问题之一即为如何设计一类同时具有表达能力强、训练稳定、推断快速这三大特性的深度生成模型。一种可行的思路是基于扩散常微分方程函数解的蒸馏[149]或一致性训练[150]，但其生成效果仍然弱于原始扩散模型。如何在保证推断速度的前提下进一步提升该类模型的生成质量，是当前领域的研究热点，也是迈向大一统的深度生成模型的必经之路。
- **基于大规模多模态预训练的通用人工智能：**已有的大规模预训练模型通常基于单独的文本数据或文本-图像数据对进行训练，其多模态能力极大地依赖于文本模态的建模能力，而这通常与其余模态的训练是解耦合的。如何基于将大规模、无标注的多模态数据对生成模型充分地进行预训练并提高其在不同模态的建模能力，解决生成模型的 “幻觉” 问题并增强模型对真实物理世界的对齐，是未来生成模型研究的长远目标。这既需要基于上述两个方面的科研突破，也需要根据不同模态的数据类型设计对应的神经网络架构及对齐算法，是机器学习与计算机系统工程结合的重要研究课题之一。

参考文献

[1] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553): 436-444.

[2] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]//Advances in Neural Information Processing Systems: volume 25. 2012.

[3] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners[C]//Advances in Neural Information Processing Systems: volume 33. 2020: 1877-1901.

[4] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2016: 770-778.

[5] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in Neural Information Processing Systems: volume 30. 2017.

[6] Bottou L. Stochastic gradient descent tricks[M]//Neural Networks: Tricks of the Trade: Second Edition. Springer, 2012: 421-436.

[7] Kingma D P, Ba J. Adam: A method for stochastic optimization[C]//International Conference on Learning Representations. 2015.

[8] Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets[C]//Advances in Neural Information Processing Systems: volume 27. 2014: 2672-2680.

[9] Deng J, Dong W, Socher R, et al. ImageNet: A large-scale hierarchical image database[C]// 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2009: 248-255.

[10] Lin T Y, Maire M, Belongie S, et al. Microsoft COCO: Common objects in context[C]// Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. Springer, 2014: 740-755.

[11] He K, Gkioxari G, Dollár P, et al. Mask R-CNN[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 2961-2969.

[12] Girshick R. Fast R-CNN[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 1440-1448.

[13] Nassif A B, Shahin I, Attili I, et al. Speech recognition using deep neural networks: A systematic review[J]. IEEE access, 2019, 7: 19143-19165.

[14] Radford A, Kim J W, Xu T, et al. Robust speech recognition via large-scale weak supervision [C]//International Conference on Machine Learning. PMLR, 2023: 28492-28518.

[15] Rombach R, Blattmann A, Lorenz D, et al. High-resolution image synthesis with latent diffusion models[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 10684-10695.

[16] Kong Z, Ping W, Huang J, et al. Diffwave: A versatile diffusion model for audio synthesis[C]// International Conference on Learning Representations. 2021.

[17] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback[C]//Advances in Neural Information Processing Systems: volume 35. 2022: 27730-27744.

[18] Ramesh A, Dhariwal P, Nichol A, et al. Hierarchical text-conditional image generation with CLIP latents[A]. 2022.

[19] Zhipeng G, Yi X, Sun M, et al. Jiuge: A human-machine collaborative chinese classical poetry generation system[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations. 2019: 25-30.

[20] Zhang C, Zhang C, Li C, et al. One small step for generative AI, one giant leap for AGI: A complete survey on ChatGPT in AIGC era[A]. 2023.

[21] 张钹, 朱军, 苏航. 迈向第三代人工智能[J]. 中国科学: 信息科学, 2020, 50(9): 1281-1302.

[22] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition [J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[23] Uria B, Murray I, Larochelle H. RNADE: The real-valued neural autoregressive density-estimator[C]//Advances in Neural Information Processing Systems: volume 26. 2013.

[24] Menick J, Kalchbrenner N. Generating high fidelity images with subscale pixel networks and multidimensional upscaling[C]//International Conference on Learning Representations. 2019.

[25] Salimans T, Karpathy A, Chen X, et al. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications[C]//International Conference on Learning Representations. 2017.

[26] Kingma D P, Welling M. Auto-encoding variational bayes[C]//International Conference on Learning Representations. 2014.

[27] Vahdat A, Kautz J. NVAE: A deep hierarchical variational autoencoder[C]//Advances in Neural Information Processing Systems: volume 33. 2020: 19667-19679.

[28] Tabak E G, Turner C V. A family of nonparametric density estimation algorithms[J]. Communications on Pure and Applied Mathematics, 2013, 66(2): 145-164.

[29] Ho J, Chen X, Srinivas A, et al. Flow++: Improving flow-based generative models with variational dequantization and architecture design[C]//International Conference on Machine Learning. 2019: 2722-2730.

[30] Dinh L, Krueger D, Bengio Y. NICE: Non-linear independent components estimation[C]// International Conference on Learning Representations Workshop. 2014.

[31] Dinh L, Sohl-Dickstein J, Bengio S. Density estimation using Real NVP[C]//International Conference on Learning Representations. 2017.

[32] Kingma D P, Dhariwal P. Glow: Generative flow with invertible 1x1 convolutions[C]// Advances in Neural Information Processing Systems. 2018: 10215-10224.

[33] Behrmann J, Grathwohl W, Chen R T, et al. Invertible residual networks[C]//International Conference on Machine Learning. 2019: 573-582.

[34] Chen R T, Behrmann J, Duvenaud D K, et al. Residual flows for invertible generative modeling [C]//Advances in Neural Information Processing Systems. 2019: 9916-9926.

[35] Hoogeboom E, Van Den Berg R, Welling M. Emerging convolutions for generative normalizing flows[C]//International Conference on Machine Learning. 2019: 2771-2780.

[36] Chen R T, Rubanova Y, Bettencourt J, et al. Neural ordinary differential equations[C]//Advances in neural information processing systems: volume 31. 2018.

[37] Grathwohl W, Chen R T, Bettencourt J, et al. Ffjord: Free-form continuous dynamics for scalable reversible generative models[C]//International Conference on Learning Representations. 2019.

[38] Chen C, Li C, Chen L, et al. Continuous-time flows for efficient inference and density estimation [C]//International Conference on Machine Learning. 2018: 824-833.

[39] LeCun Y, Chopra S, Hadsell R, et al. A tutorial on energy-based learning[J]. Predicting Structured Data, 2006, 1(0).

[40] Du Y, Mordatch I. Implicit generation and modeling with energy based models[C]//Advances in Neural Information Processing Systems: volume 32. 2019.

[41] Yu L, Song Y, Song J, et al. Training deep energy-based models with f-divergence minimization [C]//International Conference on Machine Learning. PMLR, 2020: 10957-10967.

[42] Song Y, Kingma D P. How to train your energy-based models[A]. 2021.

[43] Sohl-Dickstein J, Weiss E, Maheswaranathan N, et al. Deep unsupervised learning using nonequilibrium thermodynamics[C]//International Conference on Machine Learning. PMLR, 2015: 2256-2265.

[44] Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models[C]//Advances in Neural Information Processing Systems: volume 33. 2020: 6840-6851.

[45] Song Y, Sohl-Dickstein J, Kingma D P, et al. Score-based generative modeling through stochastic differential equations[C]//International Conference on Learning Representations. 2021.

[46] Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks[A]. 2015.

[47] Karras T, Laine S, Aila T. A style-based generator architecture for generative adversarial networks[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 4401-4410.

[48] Song Y, Ermon S. Generative modeling by estimating gradients of the data distribution[C]// Advances in Neural Information Processing Systems: volume 32. 2019: 11895-11907.

[49] Dhariwal P, Nichol A Q. Diffusion models beat GANs on image synthesis[C]//Advances in Neural Information Processing Systems: volume 34. 2021: 8780-8794.

[50] Meng C, Song Y, Song J, et al. SDEdit: Image synthesis and editing with stochastic differential equations[C]//International Conference on Learning Representations. 2022.

[51] Ho J, Salimans T, Gritsenko A, et al. Video diffusion models[A]. 2022.

[52] Chen N, Zhang Y, Zen H, et al. Wavegrad: Estimating gradients for waveform generation[C]// International Conference on Learning Representations. 2021.

[53] Chen N, Zhang Y, Zen H, et al. Wavegrad 2: Iterative refinement for text-to-speech synthesis [C]//International Speech Communication Association. 2021: 3765-3769.

[54] Kingma D P, Salimans T, Poole B, et al. Variational diffusion models[C]//Advances in Neural Information Processing Systems. 2021.

[55] Wang Z, Lu C, Wang Y, et al. ProlificDreamer: High-fidelity and diverse text-to-3D generation with variational score distillation[A]. 2023.

[56] Ruiz N, Li Y, Jampani V, et al. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023: 22500-22510.

[57] Gudovskiy D, Ishizaka S, Kozuka K. CFLOW-AD: Real-time unsupervised anomaly detection with localization via conditional normalizing flows[C]//Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2022: 98-107.

[58] Bao F, Zhao M, Hao Z, et al. Equivariant energy-guided SDE for inverse molecular design[C]// International Conference on Learning Representations. 2023.

[59] Wu J Z, Ge Y, Wang X, et al. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation[C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023: 7623-7633.

[60] Selvan R, Faye F, Middleton J, et al. Uncertainty quantification in medical image segmentation with normalizing flows[C]//Machine Learning in Medical Imaging: 11th International Workshop, MLMI 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4, 2020, Proceedings 11. Springer, 2020: 80-90.

[61] Zhang L, Rao A, Agrawala M. Adding conditional control to text-to-image diffusion models [C]//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023: 3836-3847.

[62] Papamakarios G, Pavlakou T, Murray I. Masked autoregressive flow for density estimation[C]// Advances in Neural Information Processing Systems. 2017: 2338-2347.

[63] Song Y, Meng C, Ermon S. Mintnet: Building invertible neural networks with masked convolutions[C]//Advances in Neural Information Processing Systems. 2019: 11002-11012.

[64] De Cao N, Aziz W, Titov I. Block neural autoregressive flow[C]//Uncertainty in Artificial Intelligence. PMLR, 2020: 1263-1273.

[65] Durkan C, Bekasov A, Murray I, et al. Neural spline flows[C]//Advances in Neural Information Processing Systems. 2019: 7511-7522.

[66] Chen J, Lu C, Chenli B, et al. VFlow: More expressive generative flows with variational data augmentation[C]//International Conference on Machine Learning. 2020.

[67] Huang C W, Dinh L, Courville A. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models[A]. 2020.

[68] Cornish R, Caterini A L, Deligiannidis G, et al. Relaxing bijectivity constraints with continuously indexed normalising flows[C]//International Conference on Machine Learning. 2020.

[69] Nielsen D, Jaini P, Hoogeboom E, et al. Survae flows: Surjections to bridge the gap between vaes and flows[C]//Advances in Neural Information Processing Systems: volume 33. 2020: 12685-12696.

[70] Zhang H, Gao X, Unterman J, et al. Approximation capabilities of neural ODEs and invertible residual networks[C]//International Conference on Machine Learning. 2020.

[71] Finlay C, Jacobsen J H, Nurbekyan L, et al. How to train your Neural ODE: the world of Jacobian and kinetic regularization[C]//International Conference on Machine Learning. PMLR, 2020: 3154-3164.

[72] Nichol A Q, Dhariwal P, Ramesh A, et al. Glide: Towards photorealistic image generation and editing with text-guided diffusion models[C]//International Conference on Machine Learning. PMLR, 2022: 16784-16804.

[73] Saharia C, Chan W, Saxena S, et al. Photorealistic text-to-image diffusion models with deep language understanding[C]//Advances in Neural Information Processing Systems: volume 35. 2022: 36479-36494.

[74] Gu S, Chen D, Bao J, et al. Vector quantized diffusion model for text-to-image synthesis[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 10696-10706.

[75] Salimans T, Ho J. Progressive distillation for fast sampling of diffusion models[C]//International Conference on Learning Representations. 2022.

[76] Luhman E, Luhman T. Knowledge distillation in iterative generative models for improved sampling speed[A]. 2021.

[77] San-Roman R, Nachmani E, Wolf L. Noise estimation for generative diffusion models[A]. 2021.

[78] Nichol A Q, Dhariwal P. Improved denoising diffusion probabilistic models[C]//International Conference on Machine Learning. PMLR, 2021: 8162-8171.

[79] Lam M W, Wang J, Su D, et al. BDDM: Bilateral denoising diffusion models for fast and high-quality speech synthesis[C]//International Conference on Learning Representations. 2022.

[80] Watson D, Chan W, Ho J, et al. Learning fast samplers for diffusion models by differentiating through sample quality[C]//International Conference on Learning Representations. 2022.

[81] Song J, Meng C, Ermon S. Denoising diffusion implicit models[C]//International Conference on Learning Representations. 2021.

[82] Jolicoeur-Martineau A, Li K, Piché-Taillefer R, et al. Gotta go fast when generating data with score-based models[A]. 2021.

[83] Bao F, Li C, Zhu J, et al. Analytic-DPM: An analytic estimate of the optimal reverse variance in diffusion probabilistic models[C]//International Conference on Learning Representations. 2022.

[84] Liu L, Ren Y, Lin Z, et al. Pseudo numerical methods for diffusion models on manifolds[C]// International Conference on Learning Representations. 2022.

[85] Popov V, Vovk I, Gogoryan V, et al. Diffusion-based voice conversion with fast maximum likelihood sampling scheme[C]//International Conference on Learning Representations. 2022.

[86] Tachibana H, Go M, Inahara M, et al. Itô-Taylor sampling scheme for denoising diffusion probabilistic models using ideal derivatives[A]. 2021.

[87] Lu C, Chen J, Li C, et al. Implicit normalizing flows[C]//International Conference on Learning Representations. 2021.

[88] Lu C, Zheng K, Bao F, et al. Maximum likelihood training for score-based diffusion ODEs by high order denoising score matching[C]//International Conference on Machine Learning. PMLR, 2022: 14429-14460.

[89] Lu C, Zhou Y, Bao F, et al. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps[C]//Advances in Neural Information Processing Systems: volume 35. 2022: 5775-5787.

[90] Liu X, Xiao T, Si S, et al. How does noise help robustness? explanation and exploration under the neural SDE framework[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 282-290.

[91] Massaroli S, Poli M, Bin M, et al. Stable neural flows[A]. 2020.

[92] Ho J, Saharia C, Chan W, et al. Cascaded diffusion models for high fidelity image generation [J]. Journal of Machine Learning Research, 2022, 23(47): 1-33.

[93] Saharia C, Chan W, Chang H, et al. Palette: Image-to-image diffusion models[C]//ACM SIGGRAPH 2022 Conference Proceedings. 2022: 1-10.

[94] Zhao M, Bao F, Li C, et al. Egsde: Unpaired image-to-image translation via energy-guided stochastic differential equations[C]//Advances in Neural Information Processing Systems: volume 35. 2022: 3609-3623.

[95] Liu J, Li C, Ren Y, et al. DiffSinger: Singing voice synthesis via shallow diffusion mechanism [C]//Proceedings of the AAAI Conference on Artificial Intelligence: volume 36. 2022: 11020-11028.

[96] Xu M, Yu L, Song Y, et al. GeoDiff: A geometric diffusion model for molecular conformation generation[C]//International Conference on Learning Representations. 2022.

[97] Hoogeboom E, Satorras V G, Vignac C, et al. Equivariant diffusion for molecule generation in 3D[C]//International Conference on Machine Learning. PMLR, 2022: 8867-8887.

[98] Wu L, Gong C, Liu X, et al. Diffusion-based molecule generation with informative prior bridges [C]//Advances in Neural Information Processing Systems: volume 35. 2022: 36533-36545.

[99] Luo S, Hu W. Diffusion probabilistic models for 3D point cloud generation[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 2837-2845.

[100] Zhou L, Du Y, Wu J. 3D shape generation and completion through point-voxel diffusion[C]// Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 5826-5835.

[101] Theis L, Salimans T, Hoffman M D, et al. Lossy compression with Gaussian diffusion[A]. 2022.

[102] Anderson B D. Reverse-time diffusion equation models[J]. Stochastic Processes and their Applications, 1982, 12(3): 313-326.

[103] Song Y, Garg S, Shi J, et al. Sliced score matching: A scalable approach to density and score estimation[C]//Uncertainty in Artificial Intelligence. PMLR, 2020: 574-584.

[104] Song Y, Ermon S. Improved techniques for training score-based generative models[C]// Advances in neural information processing systems: volume 33. 2020: 12438-12448.

[105] Song Y, Durkan C, Murray I, et al. Maximum likelihood training of score-based diffusion models[C]//Advances in Neural Information Processing Systems: volume 34. 2021: 1415-1428.

[106] Vincent P. A connection between score matching and denoising autoencoders[J]. Neural computation, 2011, 23(7): 1661-1674.

[107] Kloeden P E, Platen E. Numerical solution of stochastic differential equations[M]. Springer, 1992.

[108] Dormand J R, Prince P J. A family of embedded Runge-Kutta formulae[J]. Journal of Computational and Applied Mathematics, 1980, 6(1): 19-26.

[109] Ho J, Salimans T. Classifier-free diffusion guidance[C]//NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications. 2021.

[110] Rezende D, Mohamed S. Variational inference with normalizing flows[C]//International Conference on Machine Learning. 2015: 1530-1538.

[111] Kingma D P, Salimans T, Jozefowicz R, et al. Improved variational inference with inverse autoregressive flow[C]//Advances in Neural Information Processing Systems. 2016: 4743-4751.

[112] Federer H. Grundlehren der mathematischen wissenschaften[M]//Geometric measure theory: volume 153. Springer New York, 1969.

[113] Bai S, Kolter J Z, Koltun V. Deep equilibrium models[C]//Advances in Neural Information Processing Systems. 2019.

[114] Amos B, Kolter J Z. OptNet: Differentiable optimization as a layer in neural networks[C]// International Conference on Machine Learning. 2017: 136-145.

[115] Wang P W, Donti P, Wilder B, et al. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver[C]//International Conference on Machine Learning. 2019: 6545-6554.

[116] Reshniak V, Webster C G. Robust learning with implicit residual networks[J]. Machine Learning and Knowledge Extraction, 2020, 3(1): 34-55.

[117] Sitzmann V, Martel J, Bergman A, et al. Implicit neural representations with periodic activation functions[C]//Advances in neural information processing systems: volume 33. 2020: 7462-7473.

[118] Broyden C G. A class of methods for solving nonlinear simultaneous equations[J]. Mathematics of Computation, 1965, 19(92): 577-593.

[119] Skilling J. The eigenvalues of mega-dimensional matrices[M]//Maximum Entropy and Bayesian Methods. Springer, 1989: 455-466.

[120] Hutchinson M F. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines[J]. Communications in Statistics-Simulation and Computation, 1989, 18 (3): 1059-1076.

[121] Miyato T, Kataoka T, Koyama M, et al. Spectral normalization for generative adversarial networks[C]//International Conference on Learning Representations. 2018.

[122] Huang C W, Krueger D, Lacoste A, et al. Neural autoregressive flows[C]//International Conference on Machine Learning. 2018: 2078-2087.

[123] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[R]. University of Toronto, 2009.

[124] Dua D, Graff C. UCI machine learning repository[EB/OL]. University of California, Irvine, School of Information and Computer Sciences, 2017. http://archive.ics.uci.edu/ml.

[125] Krizhevsky A. Learning multiple layers of features from tiny images[R]. 2009.

[126] Hyvärinen A, Dayan P. Estimation of non-normalized statistical models by score matching.[J]. Journal of Machine Learning Research, 2005, 6(4).

[127] Meng C, Song Y, Li W, et al. Estimating high order gradients of the data distribution by denoising[C]//Advances in Neural Information Processing Systems: volume 34. 2021.

[128] Ramachandran P, Zoph B, Le Q V. Searching for activation functions[A]. 2017.

[129] Heusel M, Ramsauer H, Unterthiner T, et al. GANs trained by a two time-scale update rule converge to a local Nash equilibrium[C]//Guyon I, von Luxburg U, Bengio S, et al. Advances in Neural Information Processing Systems: volume 30. 2017: 6626-6637.

[130] Risken H, Risken H. Fokker-Planck equation[M]. Springer, 1996.

[131] Hochbruck M, Ostermann A. Exponential integrators[J]. Acta Numerica, 2010, 19: 209-286.

[132] Atkinson K, Han W, Stewart D E. Numerical solution of ordinary differential equations: volume 108[M]. John Wiley & Sons, 2011.

[133] Hochbruck M, Ostermann A. Explicit exponential Runge-Kutta methods for semilinear parabolic problems[J]. SIAM Journal on Numerical Analysis, 2005, 43(3): 1069-1090.

[134] Luan V T. Efficient exponential Runge-Kutta methods of high order: Construction and implementation[J]. BIT Numerical Mathematics, 2021, 61(2): 535-560.

[135] Bao F, Li C, Sun J, et al. Estimating the optimal covariance with imperfect mean in diffusion probabilistic models[C]//International Conference on Machine Learning. PMLR, 2022: 1555-1584.

[136] Vahdat A, Kreis K, Kautz J. Score-based generative modeling in latent space[C]//Advances in Neural Information Processing Systems: volume 34. 2021: 11287-11302.

[137] Dockhorn T, Vahdat A, Kreis K. Score-based generative modeling with critically-damped Langevin diffusion[C]//International Conference on Learning Representations. 2022.

[138] Xiao Z, Kreis K, Vahdat A. Tackling the generative learning trilemma with denoising diffusion GANs[C]//International Conference on Learning Representations. 2022.

[139] Kong Z, Ping W. On fast sampling of diffusion probabilistic models[A]. 2021.

[140] Zhang Q, Chen Y. Fast sampling of diffusion models with exponential integrator[C]// International Conference on Learning Representations. 2023.

[141] Balaji Y, Nah S, Huang X, et al. eDiff-I: Text-to-image diffusion models with an ensemble of expert denoisers[A]. 2022.

[142] Zhang Q, Tao M, Chen Y. gDDIM: Generalized denoising diffusion implicit models[C]// International Conference on Learning Representations. 2023.

[143] Wang Z, Zheng H, He P, et al. Diffusion-GAN: Training GANs with diffusion[C]//International Conference on Learning Representations. 2023.

[144] Austin J, Johnson D D, Ho J, et al. Structured denoising diffusion models in discrete state-spaces [C]//Advances in Neural Information Processing Systems: volume 34. 2021: 17981-17993.

[145] Chen T, ZHANG R, Hinton G. Analog Bits: Generating discrete data using diffusion models with self-conditioning[C]//International Conference on Learning Representations. 2023.

[146] Li X, Thickstun J, Gulrajani I, et al. Diffusion-LM improves controllable text generation[C]// Advances in Neural Information Processing Systems: volume 35. 2022: 4328-4343.

[147] Zou H, Kim Z M, Kang D. Diffusion models in NLP: A survey[A]. 2023.

[148] Li Y, Zhou K, Zhao W X, et al. Diffusion models for non-autoregressive text generation: A survey[A]. 2023.

[149] Song Y, Dhariwal P, Chen M, et al. Consistency models[A]. 2023.

[150] Song Y, Dhariwal P. Improved techniques for training consistency models[A]. 2023.

致 谢

在博士毕业之际，衷心感谢我的导师朱军教授对我在学术和人生等多方面的指导和帮助。在我刚进入课题组时，可逆生成模型很少有人关注，且缺乏实际落地的场景，从事该领域科研的不确定性极大，但我还是坚持了自己的兴趣和老师的指引。现在回看，深感导师对该领域的洞察和远见，让我有幸经历了一个方向几乎从无到有的过程，并从中受益匪浅。朱老师严谨专注、勤奋务实的品质深深地鼓舞着我。此外，在我科研受阻、人生迷茫时，朱老师多次与我谈心，帮我指导未来规划，我由衷地感谢。在日后的人生道路中，我会以朱老师为榜样，争取为人类科技进步贡献自己的一份力。

感谢张钹教授对整个课题组的无私奉献，张老师数十年如一日地坚持在人工智能科研的第一线，他的勤奋和热情深深鼓舞着我。同时，张老师心系家国，深忧“卡脖子”问题，每学期第一次组会都会鼓励我们坚持做最难、最有价值的问题，以世界最顶尖科研院所的研究成果为标准要求自己。为了保证课题组在读学生能够专心科研，张老师和朱老师付出了大量的努力，为我们营造了纯粹的学术氛围和不受干扰的科研条件，衷心感谢两位老师。

感谢实验室的陈键飞师兄和李崇轩师兄在我的科研道路中提供的耐心帮助和细致指导，我取得的学术成果离不开他们各方面的支持。感谢周聿浩、鲍凡、郑凯文、陈华玉、王征翊在与我合作期间做出的贡献和给予的帮助，感谢许堃、石佳欣、庞天宇、邓志杰、杜超、王子昱、程书宇、苏克、王思宇、宋世虹、付祈安、崔鹏在科研和生活上对我的支持。感谢实验室全体同学创造的浓厚科研氛围。

我热爱音乐和唱歌，感谢丁惟迟、张塞、王泽胤、谭立言、杨诚乐、孟晴、余竞航、周钰翔在我校园歌手比赛期间的帮助，感谢 Maj9 人声乐团的涂轶翔、沈稚栋、陈宇豪、蒲心悦、潘紫琪、李健实、景彦森与我在音乐和人生中的交流和帮助。

本人博士期间多次获得了清华大学计算机科学与技术系的奖学金，并受到了字节跳动的资助，特此致谢。

特别感谢我的女朋友叶茗雨，在我无数次低谷和迷茫时，她用爱宽慰我、鼓舞我、支持我，愿意为我牺牲自己，耐心地包容我的任性，体贴地照顾我的生活，坚定地信任并支持我的选择。我取得的学术成果离不开她在背后的默默付出，她的温柔与热烈每时每刻都包围着我，遇到她是我人生中最幸运的事。

最后，感谢我的父母路存喜和郇广利的养育之恩，感谢他们一直以来对我的无私包容和支持理解，让我有勇气面对科研和人生道路上未知的困难。

声　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____ 日　期：_____

个人简历、在学期间完成的相关学术成果

个人简历

1997 年 10 月 11 日出生于山西省阳高县。

2015 年 9 月考入清华大学土木工程系，2016 年 9 月转专业至清华大学计算机科学与技术系，2019 年 7 月本科毕业并获得工学学士学位。

2019 年 9 月免试进入清华大学计算机科学与技术系攻读工学博士学位至今。

在学期间完成的相关学术成果

学术论文：

[1] **Cheng Lu**, Jianfei Chen, Chongxuan Li, Qiuhao Wang, Jun Zhu. Implicit Normalizing Flows. In *International Conference on Learning Representations* (**ICLR**), 2021. [**Spotlight**]（清华大学 TH-CPL 推荐 A 类会议）

[2] **Cheng Lu**, Kaiwen Zheng, Fan Bao, Chongxuan Li, Jianfei Chen, Jun Zhu. Maximum Likelihood Training for Score-Based Diffusion ODEs by High-Order Denoising Score Matching. In *International Conference on Machine Learning* (**ICML**), 2022.（清华大学 TH-CPL 推荐 A 类会议）

[3] **Cheng Lu**, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, Jun Zhu. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. In *Conference on Neural Information Processing Systems* (**NeurIPS**), 2022. [**Oral**]（清华大学 TH-CPL 推荐 A 类会议）

[4] **Cheng Lu***, Huayu Chen*, Jianfei Chen, Hang Su, Chongxuan Li, Jun Zhu. Contrastive Energy Prediction for Exact Energy-Guided Diffusion Sampling in Offline Reinforcement Learning. In *International Conference on Machine Learning* (**ICML**), 2023.（清华大学 TH-CPL 推荐 A 类会议）

[5] Zhengyi Wang*, **Cheng Lu***, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, Jun Zhu. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. In *Conference on Neural Information Processing Systems* (**NeurIPS**), 2023. [**Spotlight**]（清华大学 TH-CPL 推荐 A 类会议）

[6] Kaiwen Zheng*, **Cheng Lu***, Jianfei Chen, Jun Zhu. Improved Techniques for Maximum Likelihood Estimation for Diffusion ODEs. In *International Conference on Machine Learning* (**ICML**), 2023.（清华大学 TH-CPL 推荐 A 类会议）

[7] Kaiwen Zheng*, **Cheng Lu***, Jianfei Chen, Jun Zhu. DPM-Solver-v3: Improved Diffusion ODE Solvers with Empirical Model Statistics. In *Conference on Neural*

Information Processing Systems (**NeurIPS**), 2023.（清华大学 TH-CPL 推荐 A 类会议）

[8] Jianfei Chen, **Cheng Lu**, Biqi Chenli, Jun Zhu, Tian Tian. VFlow: More Expressive Generative Flows with Variational Data Augmentation. In *International Conference on Machine Learning* (**ICML**), 2020.（清华大学 TH-CPL 推荐 A 类会议）

[9] Huayu Chen, **Cheng Lu**, Chengyang Ying, Hang Su, Jun Zhu. Offline Reinforcement Learning via High-Fidelity Generative Behavior Modeling. In *International Conference on Learning Representations* (**ICLR**), 2023.（清华大学 TH-CPL 推荐 A 类会议）

指导教师评语

深度生成模型是一类将深度神经网络与贝叶斯方法有机融合的机器学习模型，能够描述复杂数据（如图像、文本等）的分布，并用于新数据的生成。该论文针对可逆生成模型的若干基本问题开展研究，设计灵活的新模型和高效的新算法，论文选题具有重要的理论意义和实用价值。论文的主要创新成果为：

1. 针对离散时间标准化流模型表达能力受限的问题，提出了一种基于隐函数定义的隐式标准化流模型，显著提升了模型表达能力和描述图像数据的性能。
2. 针对连续时间标准化流模型的最大似然训练开销较大的问题，提出了一种误差可控的高阶去噪分数匹配算法，避免了连续时间标准化流模型所依赖的常微分方程求解器的昂贵开销。
3. 针对无条件扩散模型采样速度慢的问题，提出了一种无需训练的快速高阶扩散常微分方程求解器，显著提升了无条件扩散模型的采样速度。
4. 针对条件扩散模型的采样速度慢以及少步采样不稳定的问题，提出了一种求解扩散常微分方程和扩散随机微分方程的无需训练的快速采样算法，显著提升了条件扩散模型的采样速度和稳定性。

该论文成果丰富，理论研究与实际应用相结合。论文提出的部分高效算法成果已被开源社区广泛引用，是求解扩散模型的最快采样算法之一。论文结构合理，书写规范，描述清晰，是一篇优秀的博士学位论文。

答辩委员会决议书

论文研究可逆深度生成式模型的表达和计算问题，选题具有重要的理论意义和应用价值。

论文的主要创新性成果如下：

1. 针对离散时间流模型，提出了一种基于隐函数定义的隐式流模型，增强了其表达能力；
2. 针对连续时间流模型，提出了一种误差可控的高阶去噪分数匹配算法，提高了扩散常微分方程最大似然训练的效率；
3. 针对无条件扩散模型，利用扩散常微分方程的半线性特性，提出了一种无需训练的高阶常微分方程求解器，显著提升了该类模型的采样速度；
4. 针对条件扩散模型，基于数据预测的重参数化形式，提出了一种高阶引导采样算法，显著提升了该类模型的采样速度和稳定性。

论文工作表明，作者已掌握本学科领域坚实宽广的基础理论和系统深入的专门知识，独立从事科研工作的能力强。论文写作规范，结构合理，叙述清楚。答辩过程中表述流畅，回答问题正确，是一篇优秀的博士论文。答辩委员会经无记名投票表决，一致同意通过论文答辩，并建议授予路橙同学工学博士学位。