Journal of Machine Learning Research 20 (2019) 1-52 Submitted 5/16; Revised 10/18; Published 2/19

Transport Analysis of Infinitely Deep Neural Network

Sho Sonoda SHO.SONODA@RIKEN.JP
Center for Advanced Intelligence Project
RIKEN
1–4–1 Nihonbashi, Chuo-ku, Tokyo 103–0027, Japan

Noboru Murata NOBORU.MURATA@EB.WASEDA.AC.JP
School of Advanced Science and Engineering
Waseda University
3–4–1 Okubo, Shinjuku-ku, Tokyo 169–8555, Japan

Editor: Yoshua Bengio

Abstract

We investigated the feature map inside deep neural networks (DNNs) by tracking the transport map. We are interested in the *role of depth*—why do DNNs perform better than shallow models?—and the *interpretation* of DNNs—what do intermediate layers do? Despite the rapid development in their application, DNNs remain analytically unexplained because the hidden layers are nested and the parameters are not faithful. Inspired by the *integral representation* of shallow NNs, which is the continuum limit of the width, or the hidden unit number, we developed the *flow representation* and *transport analysis* of DNNs. The flow representation is the continuum limit of the depth, or the hidden layer number, and it is specified by an ordinary differential equation (ODE) with a vector field. We interpret an ordinary DNN as a *transport map* or an Euler broken line approximation of the flow. Technically speaking, a dynamical system is a natural model for the nested feature maps. In addition, it opens a new way to the coordinate-free treatment of DNNs by avoiding the redundant parametrization of DNNs. Following *Wasserstein geometry*, we analyze a flow in three aspects: dynamical system, continuity equation, and Wasserstein gradient flow. A key finding is that we specified a series of transport maps of the *denoising autoencoder* (DAE), which is a cornerstone for the development of deep learning. Starting from the shallow DAE, this paper develops three topics: the transport map of the deep DAE, the equivalence between the stacked DAE and the composition of DAEs, and the development of the double continuum limit or the integral representation of the flow representation. As partial answers to the research questions, we found that deeper DAEs converge faster and the extracted features are better; in addition, a deep Gaussian DAE transports mass to decrease the Shannon entropy of the data distribution. We expect that further investigations on these questions lead to the development of an interpretable and principled alternatives to DNNs.

Keywords: representation learning, denoising autoencoder, flow representation, continuum limit, backward heat equation, Wasserstein geometry, ridgelet analysis

1. Introduction

Despite the rapid development in their application, deep neural networks (DNN) remain analytically unexplained. We are interested in the *role of depth—why do DNNs perform better than shallow models?*—and the *interpretation* of DNNs—*what do intermediate layers* *do?* To the best of our knowledge, thus far, traditional theories, such as the statistical learning theory (Vapnik, 1998), have not succeeded in completely answering the above questions (Zhang et al., 2018). Existing DNNs lack interpretability; hence, a DNN is often called a *blackbox*. In this study, we propose the *flow representation* and *transport analysis* of DNNs, which provide us with insights into why DNNs can perform better and facilitate our understanding of what DNNs do. We expect that these lines of study lead to the development of an interpretable and principled alternatives to DNNs.

Compared to other *shallow models*, such as kernel methods (Shawe-Taylor and Cristianini, 2004) and ensemble methods (Schapire and Freund, 2012), DNNs have at least two specific technical issues: the *function composition* and the *redundant and complicated parametrization*. First, a DNN is formally a composite $\boldsymbol{g}_L \circ \cdots \circ \boldsymbol{g}_0$ of intermediate maps $\boldsymbol{g}_\ell \, (\ell = 0, \ldots, L)$. Here, each $\boldsymbol{g}_\ell$ corresponds to the ℓ-th hidden layer. Currently, our understanding of learning machines is based on *linear algebra*, i.e., the *basis and coefficients* (Vapnik, 1998). Linear algebra is compatible with shallow models because a shallow model is a linear combination of basis functions. However, it has poor compatibility with deep models because the function composition $(\boldsymbol{f}, \boldsymbol{g}) \mapsto \boldsymbol{f} \circ \boldsymbol{g}$ is not assumed in the standard definition of the linear space. Therefore, we should move to spaces where the function composition is defined, such as monoids, semigroups, and *dynamical systems*. Second, the standard parametrization of the NN, such as $\boldsymbol{g}_\ell(\boldsymbol{x}) = \sum_{j=1}^p \boldsymbol{c}_j^\ell \sigma(\boldsymbol{a}_j^\ell \cdot \boldsymbol{x} - b_j^\ell)$, is redundant because there exist different sets of parameters that specify the same function, which causes technical problems, such as local minima. Furthermore, it is complicated because the interpretation of parameters is usually impossible, which results in the blackbox nature of DNNs. Therefore, we need a new parametrization that is concise in the sense that different parameters specify different functions and simple in the sense that it is easy to understand.

For shallow NNs, the *integral representation theory* (Murata, 1996; Candès, 1998; Sonoda and Murata, 2017a) provides a concise and simple reparametrization. The integral representation is derived by a continuum limit of the *width* or the number of hidden units. Owing to the *ridgelet transform* or a pseudo-inverse operator of the integral representation operator, it is concise and simple (see Section 1.3.2 for further details on the ridgelet transform). Furthermore, in the integral representation, we can compute the parameters of the shallow NN that attains the *global minimum* of the backpropagation training (Sonoda et al., 2018). In the integral representation, thus far, the shallow NNs is no longer a blackbox, and the training is principled. However, the integral representation is again based on linear algebra, the scope of which does not include DNNs.

Inspired by the integral representation theory, we introduced the *flow representation* and developed the *transport analysis* of DNNs. The flow representation is derived by a continuum limit of the *depth* or the number of hidden layers. In the flow representation, we formulate a DNN as a flow of an ordinary differential equation (ODE) $\dot{\boldsymbol{x}}_t = \boldsymbol{v}_t(\boldsymbol{x}_t)$ with vector field $\boldsymbol{v}_t$. In addition, we introduced the *transport map* by which we call a discretization $\boldsymbol{x} \mapsto \boldsymbol{x} + \boldsymbol{f}_t(\boldsymbol{x})$ of the flow. Specifically, we regard the intermediate map $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^n$ of an ordinary DNN as a transport map that transfers the mass at $\boldsymbol{x} \in \mathbb{R}^m$ toward $\boldsymbol{g}(\boldsymbol{x}) \in \mathbb{R}^n$. Since the flow and transport map are independent of coordinates, they enable us the coordinate-free treatment of DNNs. In the transport analysis, following *Wasserstein geometry* (Villani, 2009), we track a flow by analyzing the three profiles of the

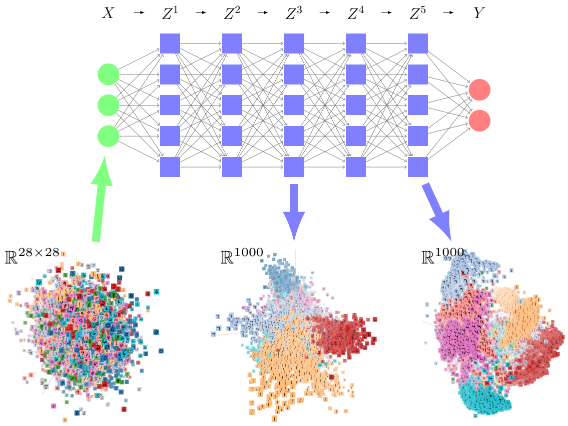


Figure 1: Mass transportation in a deep neural network that classifies images of digits. In the final hidden layer, the feature vectors have to be linearly separable because the output layer is just a linear classifier. Hence, through the network, the same digits gradually accumulate and different digits gradually separate.

flow: *dynamical system*, *pushforward measure*, and *Wasserstein gradient flow* (Ambrosio et al., 2008) (see Section 2 for further details).

We note that when the input and the output differ in dimension, i.e., $m \neq n$, we simply consider that both the input space and the output space are embedded in a common high-dimensional space. As a composite of transport maps leads to another transport map, the transport map has compatibility with deep structures. In this manner, transportation is a universal characteristic of DNNs. For example, let us consider a digit recognition problem with DNNs. We can expect the feature extractor in the DNN to be a transport map that separates the feature vectors of different digits, similar to the separation of *oil and water* (see Figure 1 for example). At the time of the initial submission in 2016, the flow representation seemed to be a novel viewpoint of DNNs. At present, it is the mainstream of development. For example, two important DNNs—residual network (ResNet) (He et al., 2016) and generative adversarial net (GAN) (Goodfellow et al., 2014)—are now considered to be transport maps (see Section 1.2 for a more detailed survey). Instead of directly investigating DNNs in terms of the redundant and complex parametrization, we perform transport analysis associated with the flow representation. We consider that the flow representation is potentially concise and simple because the flow is independent of parametrization, and it is specified by a single vector field $\boldsymbol{v}$.

In this study, we demonstrate transport analysis of the *denoising autoencoder (DAE)*. The DAE was introduced by Vincent et al. (2008) as a heuristic modification to enhance the robustness of the traditional autoencoder. The traditional autoencoder is an NN that is trained as an identity map $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x}$. The hidden layer of the network is used as a feature map, which is often called the "code" because the activation pattern appears to be random, but it surely encodes some information about the input data. On the other hand, the DAE is an NN that is trained as a "denoising" map $\boldsymbol{g}(\widetilde{\boldsymbol{x}}) \approx \boldsymbol{x}$ of deliberately corrupted inputs $\widetilde{\boldsymbol{x}}$. The DAE is a cornerstone for the development of deep learning or representation learning (Bengio et al., 2013a). Although the *corrupt and denoise* principle is simple, it is successful and has inspired many representation learning algorithms (see Section 1.3.1 for example). Furthermore, we investigate *stacking* (Bengio et al., 2007) of DAEs. Because *stacked DAE* (Vincent et al., 2010) runs DAEs on the codes in the hidden layer, it has been less investigated, so far.

The key finding is that when the corruption process is additive, i.e., $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$ with some noise $\boldsymbol{\varepsilon}$, then the DAE $\boldsymbol{g}$ is given by the sum of the traditional autoencoder $\widetilde{\boldsymbol{x}} \mapsto \widetilde{\boldsymbol{x}}$ and a certain denoising term $\widetilde{\boldsymbol{x}} \mapsto \boldsymbol{f}_t(\widetilde{\boldsymbol{x}})$ parametrized by noise variance t:

$$\boldsymbol{g}_t(\widetilde{\boldsymbol{x}}) = \widetilde{\boldsymbol{x}} + \boldsymbol{f}_t(\widetilde{\boldsymbol{x}}). \tag{1}$$

From the statistical viewpoint, this equation is reasonable because the DAE amounts to an estimation problem of the mean parameter. Obviously, (1) is a transport map because the denoising term $\boldsymbol{f}_t$ is a displacement vector from the origin $\widetilde{\boldsymbol{x}}$ and the noise variance t is the transport time. Starting from the shallow DAE, this paper develops three topics: the transport map of the deep DAE, the equivalence between the stacked DAE and the composition of DAEs, and the development of the double continuum limit, or the integral representation of the flow representation.

1.1. Contributions of This Study

In this paper, we introduce the flow representation of DNNs and develop the transport analysis of DAEs. The contributions of this paper are listed below.

- We introduced the flow representation, which can avoid the redundancy and complexity of the ordinary parametrization of DNNs.

- We specified the transport maps of shallow, deep, and infinitely deep DAEs, and provided their statistical interpretations. The shallow DAE is an estimator of the mean, and the deep DAE transports data points to decrease the Shannon entropy of the data distribution. According to analytic and numerical experiments, we showed that deep DAEs can extract much more information than shallow DAEs.

- We proved the equivalence between the stacked DAE and the composition of DAEs. Because of the peculiar construction, it is difficult to formulate and understand stacking. Nevertheless, by tracking the flow, we succeeded in formulating the stacked DAE. Consequently, we can interpret the effect of the *pre-training* as a regularization of hidden layers.

- We provided a new direction for the mathematical modeling of DNNs: the double continuum limit or the integral representation of the flow representation. We presented some examples of the double continuum limit of DAEs. In the integral representation, the shallow NNs is no longer a blackbox, and the training is principled. We consider that further investigations on the double continuum limit lead to the development of an interpretable and principled alternatives to DNNs.

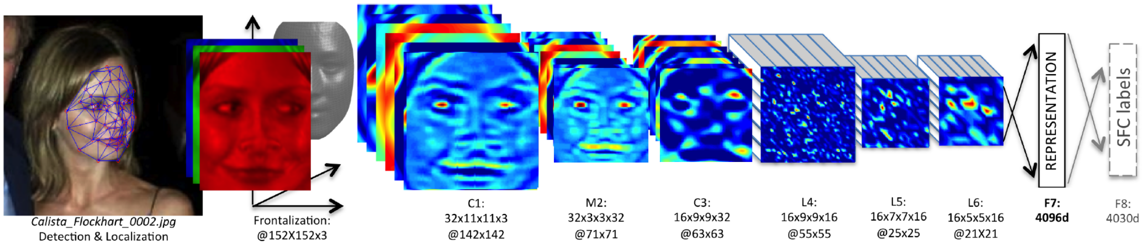


Figure 2: The activation patterns in DeepFace gradually changes (Taigman et al., 2014).

1.2. Related Work

1.2.1. Why Deep?

Before the success of deep learning, traditional theories were skeptical of the depth concept. According to approximation theory, (not only NNs but also) various shallow models can approximate any function (Pinkus, 2005). According to estimation theory, various shallow models can attain the minimax optimal ratio (Tsybakov, 2009). According to optimization theory, the depth does nothing but increase the complexity of loss surfaces unnecessarily (Boyd and Vandenberghe, 2004). In reality, of course, DNNs perform overwhelmingly better than shallow models. Thus far, the learning theory has not succeeded in explaining the gap between theory and reality (Zhang et al., 2017).

In recent years, these theories have changed drastically. For example, many authors claim that the depth increases the expressive power in the exponential order while the width does so in the polynomial order (Telgarsky, 2016; Eldan and Shamir, 2016; Cohen et al., 2016; Yarotsky, 2017), and that DNNs can attain the minimax optimal ratio in wider classes of functions (Schmidt-Hieber, 2017; Imaizumi and Fukumizu, 2019). Radical reviews of the shape of loss surfaces (Dauphin et al., 2014; Choromanska et al., 2015; Kawaguchi, 2016; Soudry and Carmon, 2016), the implicit regularization by stochastic gradient descent (Neyshabur, 2017), and the acceleration effect by over-parametrization (Nguyen and Hein, 2017; Arora et al., 2018) are ongoing. Besides the recent trends toward the rationalization of deep learning, neutral yet interesting studies have been published (Ba and Caruana, 2014; Lin et al., 2017; Poggio et al., 2017). In this study, we found that deep DAEs converge faster and that the extracted features are different from each other.

1.2.2. What Do Deep Layers Do?

Traditionally, DNNs are said to construct the hierarchy of meanings (Hinton, 1989). In convolutional NNs for image recognition, such hierarchies are empirically observed (Lee, 2010; Krizhevsky et al., 2012; Zeiler and Fergus, 2014). The hierarchy hypothesis seems to be acceptable, but it lacks explanations as to how the hierarchy is organized.

Taigman et al. (2014) reported an interesting phenomenon whereby the activation patterns in the hidden layers change by gradation from face-like patterns to codes. Inspired by Figure 2, we came up with the idea of regarding the activation pattern as a coordinate and the depth as the transport time.

1.2.3. Flow Inside Neural Networks

At the time of the initial submission in 2016, the flow representation, especially the continuum limit of the depth and collaboration with Wasserstein geometry, seemed to be a novel viewpoint of DNNs. At present, it is the mainstream of development.

Alain and Bengio (2014) was the first to derive a special case of (1), which motivated our study. Then, Alain et al. (2016) developed the generative model as a probabilistic reformulation of DAE. The generative model was a new frontier at that time; now, it is widely used in variational autoencoders (Kingma and Welling, 2014), generative adversarial nets (GANs) (Goodfellow et al., 2014), minimum probability flows (Sohl-Dickstein et al., 2015), and normalizing flows (Rezende and Mohamed, 2015). Generative models have high compatibility with transport analysis because they are formulated as Markov processes. In particular, the *generator* in GANs is exactly a transport map because it is a change-of-distribution $\boldsymbol{g} : M \to N$ from a normal distribution to a data distribution. From this viewpoint, Arjovsky et al. (2017) succeeded in stabilizing the training process of GANs by introducing Wasserstein geometry.

The *skip connection* in the residual network (ResNet) (He et al., 2016) is considered to be a key structure for training a super-deep network with more than $1,000$ layers. Formally, the skip connection is a transport map because it has an expression $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x})$. From this viewpoint, Nitanda and Suzuki (2018) reformulated the ResNet as a functional gradient and estimated the generalization error, and Lu et al. (2018) unified various ResNets as ODEs. In addition, Chizat and Bach (2018) proved the global convergence of stochastic gradient descent (SGD) using Wasserstein gradient flow. Novel deep learning methods have been proposed by controlling the flow (Ioffe and Szegedy, 2015; Gomez et al., 2017; Haber and Ruthotto, 2018; Li and Hao, 2018; Chen et al., 2018).

We remark that in shrinkage statistics, the expression of the transport map $\boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x})$ is known as Brown's representation of the posterior mean (George et al., 2006). Liu and Wang (2016) analyzed it and proposed a Bayesian inference algorithm, apart from deep learning.

1.3. Background

1.3.1. Denoising Autoencoders

The *denoising autoencoder (DAE)* is a fundamental model for representation learning, the objective of which is to capture a good representation of the data. Vincent et al. (2008) introduced it as a heuristic modification of traditional autoencoders for enhancing robustness. In the setting of traditional autoencoders, we train an NN as an identity map $\boldsymbol{x} \mapsto \boldsymbol{x}$ and extract the hidden layer to obtain the so-called "code." On the other hand, the DAE is trained as a denoising map $\widetilde{\boldsymbol{x}} \mapsto \boldsymbol{x}$ of deliberately corrupted inputs $\widetilde{\boldsymbol{x}}$. Although the *corrupt and denoise* principle is simple, it has inspired many next-generation models. In this study, we analyze DAE variants such as shallow DAE, deep DAE (or composition of DAEs), infinitely deep DAE (or continuous DAE), and stacked DAE. Stacking (Bengio et al., 2007) was proposed in the early stages of deep learning, and it remains a mysterious treatment because it runs DAEs on codes in the hidden layer.

The theoretical justifications and extensions follow from at least five standpoints: manifold learning (Rifai et al., 2011; Alain and Bengio, 2014), generative modeling (Vincent et al., 2010; Bengio et al., 2013b, 2014), infomax principle (Vincent et al., 2010), learning dynamics (Erhan et al., 2010), and score matching (Vincent, 2011). The first three standpoints were already mentioned in the original paper (Vincent et al., 2008). According to these standpoints, a DAE extracts one of the following from the data set: a manifold on which the data are arranged (manifold learning); the latent variables, which often behave as nonlinear coordinates in the feature space, that generate the data (generative modeling); a transformation of the data distribution that maximizes the mutual information (infomax); good initial parameters that allow the training to avoid local minima (learning dynamics); or the data distribution (score matching). A turning point appears to be the finding of the score matching aspect (Vincent, 2011), which reveals that score matching with a special form of the energy function coincides with a DAE. Thus, a DAE is a density estimator of the data distribution μ. In other words, it extracts and stores information as a function of μ. Since then, many researchers have avoided stacking deterministic autoencoders and have developed generative density estimators (Bengio et al., 2013b, 2014) instead.

1.3.2. Integral Representation Theory and Ridgelet Analysis

The flow representation is inspired by the integral representation theory (Murata, 1996; Candès, 1998; Sonoda and Murata, 2017a).

The integral representation

$$S[\gamma](\boldsymbol{x}) = \int \gamma(\boldsymbol{a}, b)\sigma(\boldsymbol{a} \cdot \boldsymbol{x} - b)\mathrm{d}\lambda(\boldsymbol{a}, b) \tag{2}$$

is a continuum limit of a shallow NN $g_p(\boldsymbol{x}) = \sum_{j=1}^{p} c_j \sigma(\boldsymbol{a}_j \cdot \boldsymbol{x} - b_j)$ as the hidden unit number $p \to \infty$. In $S[\gamma]$, every possible nonlinear parameter $(\boldsymbol{a}, b)$ is "integrated out," and only linear parameters c_j remain as a coefficient function $\gamma(\boldsymbol{a}, b)$. Therefore, we do not need to select which $(\boldsymbol{a}, b)$'s to use, which amounts to a non-convex optimization problem. Instead, the coefficient function $\gamma(\boldsymbol{a}, b)$ automatically selects the $(\boldsymbol{a}, b)$'s by weighting them. Similar reparametrization techniques have been proposed for Bayesian NNs (Radford M. Neal, 1996) and convex NNs (Bengio et al., 2006; Bach, 2017a). Once a coefficient function γ is given, we can obtain an ordinary NN g_p that approximates $S[\gamma]$ by numerical integration. We also remark that the integral representation $S[\gamma_p]$ with a singular coefficient $\gamma_p := \sum_{j=1}^{p} c_j \delta_{(\boldsymbol{a}_j, b_j)}$ leads to an ordinary NN g_p.

The advantage of the integral representation is that the solution operator—the *ridgelet transform*—to the integral equation $S[\gamma] = f$ and the optimization problem of $L[\gamma] := \|S[\gamma] - f\|^2 + \beta\|\gamma\|^2$ is known. The ridgelet transform with an admissible function ρ is given by

$$R[f](\boldsymbol{a}, b) := \int_{\mathbb{R}^m} f(\boldsymbol{x})\overline{\rho(\boldsymbol{a} \cdot \boldsymbol{x} - b)}\mathrm{d}\boldsymbol{x}. \tag{3}$$

The integral equation $S[\gamma] = f$ is a traditional form of learning, and the ridgelet transform $\gamma = R[f]$ satisfies $S[\gamma] = S[R[f]] = f$ (Murata, 1996; Candès, 1998; Sonoda and Murata, 2017a). The optimization problem of $L[\gamma]$ is a modern form of learning, and a modified version of the ridgelet transform gives the global optimum (Sonoda et al., 2018). These studies imply that a shallow NN is *no longer a blackbox* but a ridgelet transform of the data set. Traditionally, the integral representation has been developed to estimate the approximation and estimation error bounds of shallow NNs g_p (Barron, 1993; Kůrková, 2012; Klusowski and Barron, 2017, 2018; Suzuki, 2018). Recently, the numerical integration methods for $R[f]$ and $S[R[f]]$ were developed (Candès, 1998; Sonoda and Murata, 2014; Bach, 2017b) with various f, including the MNIST classifier. Hence, by computing the ridgelet transform of the data set, we can obtain the global minimizer without gradient descent.

Thus far, the integral representation is known as an efficient reparametrization method to facilitate understanding of the hidden layers, to estimate the approximation and estimation error bounds of shallow NNs, and to calculate the hidden parameters. However, it is based on linear algebra, i.e., it starts by regarding c_j and $\sigma(\boldsymbol{a}_j \cdot \boldsymbol{x} - b_j)$ as coefficients and basis functions, respectively. Therefore, the integral representation for DNNs is not trivial at all.

1.3.3. Optimal Transport Theory and Wasserstein Geometry

The optimal transport theory (Villani, 2009) originated from the practical requirement in the 18th century to transport materials at the minimum cost. At the end of the 20th century, it was transformed into *Wasserstein geometry,* or the geometry on the space of probability distributions. Recently, Wasserstein geometry has attracted considerable attention in statistics and machine learning. One of the reasons for its popularity is that the *Wasserstein distance* can capture the difference between two singular measures, whereas the traditional Kullback-Leibler distance cannot (Arjovsky et al., 2017). Another reason is that it gives a unified perspective on a series of function inequalities, including the concentration inequality. Computation methods for the Wasserstein distance and Wasserstein gradient flow have also been developed (Peyré and Cuturi, 2018; Nitanda and Suzuki, 2018; Zhang et al., 2018). In this study, we employ *Wasserstein gradient flow* (Ambrosio et al., 2008) for the characterization of DNNs.

Given a density μ of materials in $\mathbb{R}^m$, a density ν of final destinations in $\mathbb{R}^m$, and a cost function $c : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ associated with the transportation, under some regularity conditions, there exist some optimal transport map(s) $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^m$ that attain the minimum transportation cost. Let $W(\mu, \nu)$ denote the minimum cost of the transportation problem from μ to ν. Then, it behaves as the distance between two probability densities μ and ν, and it is called the *Wasserstein distance*, which is the start point of Wasserstein geometry.

When the cost function c is given by the ℓ^p-distance, i.e., $c(\boldsymbol{x}, \boldsymbol{y}) = |\boldsymbol{x} - \boldsymbol{y}|_p$, the corresponding Wasserstein distance is called the L^p-Wasserstein distance $W_p(\mu, \nu)$. Let $\mathcal{P}_p(\mathbb{R}^m)$ be the space of probability densities on $\mathbb{R}^m$ that have at least the p-th moment. The distance space $\mathcal{P}_p(\mathbb{R}^m)$ equipped with L^p-Wasserstein distance W_p is called the *L^p-Wasserstein space.* Furthermore, the L^2-Wasserstein space $(\mathcal{P}_2, W_2)$ admits the *Wasserstein metric* $\mathfrak{g}_2$, which is an infinite-dimensional Riemannian metric that induces the L^2-Wasserstein distance as the geodesic distance. Owing to $\mathfrak{g}_2$, the L^2-Wasserstein space is an infinite-dimensional manifold. On $\mathcal{P}_2$, we can introduce the tangent space $T_\mu \mathcal{P}_2$ at $\mu \in \mathcal{P}_2$, and the gradient operator grad, which are fundamentals to define *Wasserstein gradient flow.* See Section 2 for more details.

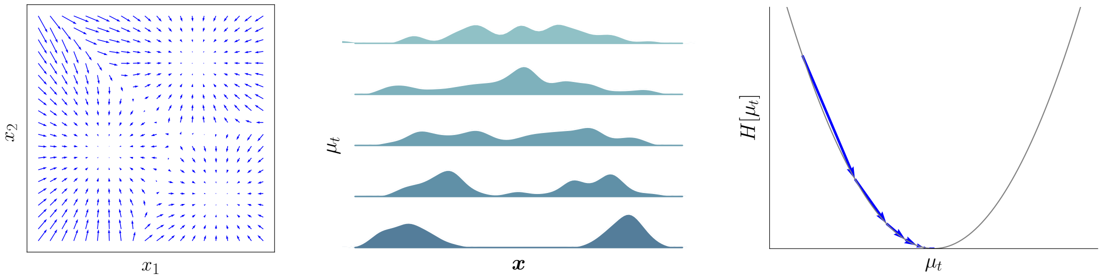


Figure 3: Three profiles of a flow analyzed in the transport analysis: dynamical system in $\mathbb{R}^m$ described by vector field (or transport map) (**left**), pushforward measure described by continuity equation in $\mathbb{R}^m$ (**center**), and Wasserstein gradient flow in $\mathcal{P}_2(\mathbb{R}^m)$ (**right**).

Organization of This Paper

In Section 2, we describe the framework of transport analysis, which combines a quick introduction to dynamical systems theory, optimal transport theory, and Wasserstein gradient flow. In Section 3 and 4, we specify the transport maps of shallow, deep, and infinitely deep DAEs, and we give their statistical interpretations. In Section 5, we present analytic examples and the results of numerical experiments. In Section 6, we prove the equivalence between the stacked DAE and the composition of DAEs. In Section 7, we develop the integral representation of the flow representation.

Remark

After the initial submission of the manuscript in 2016, the present manuscript has been substantially reorganized and updated. The authors presented the digests of some results from Section 3, 4 and 7 in two workshops (Sonoda and Murata, 2017b,c).

2. Transport Analysis of Deep Neural Networks

In the transport analysis, we regard a deep neural network as a transport map, and we track the flow in three scales: microscopic, mesoscopic, and macroscopic. Wasserstein geometry provides a unified framework for bridging these three scales. In each scale, we analyze three profiles of the flow: dynamical system, pushforward measure, and Wasserstein gradient flow.

First, on the microscopic scale, we analyze the transport map $\boldsymbol{g}_t : \mathbb{R}^m \to \mathbb{R}^m$, which simply describes the transportation of every point. In continuum mechanics, this viewpoint corresponds to the Eulerian description. The transport map $\boldsymbol{g}_t$ is often associated with a velocity field $\boldsymbol{v}_t$ that summarizes all the behavior of $\boldsymbol{g}_t$ by an ODE or the continuous dynamical system: $\partial_t \boldsymbol{g}_t(\boldsymbol{g}_t(\boldsymbol{x})) = \boldsymbol{v}_t(\boldsymbol{g}_t(\boldsymbol{x}))$. We note that, as suggested by chaos theory, it is generally difficult to track a continuous dynamics.

Second, on the mesoscopic scale, we analyze the pushforward μ_t or the time evolution of the data distribution. In continuum mechanics, this viewpoint corresponds to the Lagrangian description. When the transport map is associated with a vector field $\boldsymbol{v}_t$, then the corresponding distributions evolve according to a partial differential equation (PDE) or the *continuity equation* $\partial_t \mu_t = -\nabla \cdot [\boldsymbol{v}_t \mu_t]$. We note that, as suggested by fluid dynamics, it is generally difficult to track a continuity equation.

Finally, on the macroscopic scale, we analyze the Wasserstein gradient flow or the trajectories of time evolution of μ_t in the space $\mathcal{P}(\mathbb{R}^m)$ of probability distributions on $\mathbb{R}^m$. When the transport map is associated with a vector field $\boldsymbol{v}_t$, then there exists a time-independent potential functional F on $\mathcal{P}(\mathbb{R}^m)$ such that an evolution equation or the Wasserstein gradient flow $\dot{\mu}_t = -\mathsf{grad}\, F[\mu_t]$ coincides with the continuity equation. We remark that tracking a Wasserstein gradient flow may be easier compared to the two above-mentioned cases, because the potential functional is independent of time.

2.1. Transport Map and Flow

In the broadest sense, a *transport map* is simply a measurable map $\boldsymbol{g} : M \to N$ between two probability spaces M and N (see Definition 1.2 in Villani, 2009, for example). In this study, we use the term as an update rule. Depending on the context, we distinguish the term "flow" from "transport map." While a flow is associated with a continuous dynamical system, a transport map is associated with a discrete dynamical system. We understand that a transport map arises as a discretization of a flow. An ordinary DNN coincides with a transport map, and the depth continuum limit coincides with a flow.

Definition 1 *A transport map $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^m$ is a measurable map given by*

$$\begin{cases} \boldsymbol{g}_t(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{f}_t(\boldsymbol{x}), & \boldsymbol{x} \in \mathbb{R}^m,\ t > 0 \\ \boldsymbol{g}_0(\boldsymbol{x}) = \boldsymbol{x}, & \boldsymbol{x} \in \mathbb{R}^m,\ t = 0, \end{cases} \tag{4}$$

with an update vector $\boldsymbol{f}_t$.

Definition 2 *A flow $\boldsymbol{\varphi}_t$ is given by an ordinary differential equation (ODE),*

$$\begin{cases} \dot{\boldsymbol{\varphi}}_t(\boldsymbol{x}) = \boldsymbol{v}_t(\boldsymbol{\varphi}_t(\boldsymbol{x})), & \boldsymbol{x} \in \mathbb{R}^m,\ t > 0 \\ \boldsymbol{\varphi}_0(\boldsymbol{x}) = \boldsymbol{x}, & \boldsymbol{x} \in \mathbb{R}^m,\ t = 0, \end{cases} \tag{5}$$

with a velocity field $\boldsymbol{v}_t$.

In particular, we are interested in the case when the update rule (4) is a tangent line approximation of a flow (5). i.e., $\boldsymbol{g}_t$ satisfies

$$\lim_{t \to 0} \frac{\boldsymbol{g}_t(\boldsymbol{x}) - \boldsymbol{x}}{t} = \boldsymbol{v}_0(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^m \tag{6}$$

for some $\boldsymbol{v}_t$. In this case, the velocity field $\boldsymbol{v}_t$ is the only parameter that determines the transport map.

2.2. Pushforward Measure and Continuity Equation

In association with the mass transportation $\boldsymbol{x} \mapsto \boldsymbol{g}_t(\boldsymbol{x})$, the data distribution μ_0 itself changes its shape to, say, μ_t (see Figure 4, for example). Technically speaking, μ_t is called (the density of) the *pushforward measure* of μ_0 by $\boldsymbol{g}_t$, and it is denoted by $\boldsymbol{g}_{t\sharp}\mu_0$.

Definition 3 *Let μ be a Borel measure on M and $\boldsymbol{g} : M \to N$ be a measurable map. Then, $\boldsymbol{g}_\sharp \mu$ denotes the image measure (or pushforward) of μ by $\boldsymbol{g}$. It is a measure on N, defined by $(\boldsymbol{g}_\sharp \mu)(B) = \mu \circ \boldsymbol{g}^{-1}(B)$ for every Borel set $B \subset N$.*

The pushforward μ_t is calculated by the change-of-variables formula. In particular, the following extended version by Evans and Gariepy (2015, Theorem 3.9) from geometric measure theory is useful.

Fact 1 *Let $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^n$ be Lipschitz continuous, $m \leq n$, and μ be a probability density on $\mathbb{R}^m$. Then, the pushforward $\boldsymbol{g}_\sharp \mu$ satisfies*

$$\boldsymbol{g}_\sharp \mu \circ \boldsymbol{g}(\boldsymbol{x})[\nabla \boldsymbol{g}](\boldsymbol{x}) = \mu(\boldsymbol{x}), \quad \text{a.e. } \boldsymbol{x}. \tag{7}$$

Here, the Jacobian is defined by

$$[\nabla \boldsymbol{g}] = \sqrt{\det |(\nabla \boldsymbol{g})^* \circ (\nabla \boldsymbol{g})|}. \tag{8}$$

The continuity equation describes the one-to-one relation between a flow and the pushforward.

Fact 2 *Let $\boldsymbol{\varphi}_t$ be the flow of an ODE* (5) *with vector field $\boldsymbol{v}_t$. Then, the pushforward μ_t of the initial distribution μ_0 evolves according to the* continuity equation

$$\partial_t \mu_t(\boldsymbol{x}) = -\nabla \cdot [\mu_t(\boldsymbol{x}) \boldsymbol{v}_t(\boldsymbol{x})], \quad \boldsymbol{x} \in \mathbb{R}^m, \, t \geq 0. \tag{9}$$

Here, $\nabla \cdot$ denotes the divergence operator in $\mathbb{R}^m$.

The continuity equation is also known as the *conservation of mass formula*, and this relation between the partial differential equation (PDE) (9) and the ODE (5) is a well-known fact in continuum physics (Villani, 2009, pp.19). See Appendix B for a sketch of the proof and Ambrosio et al. (2008, § 8) for more detailed discussions.

2.3. Wasserstein Gradient Flow Associated with Continuity Equation

In addition to the ODE and PDE in $\mathbb{R}^m$, we introduce the third profile: the *Wasserstein gradient flow* or the evolution equation in the space of the probability densities on $\mathbb{R}^m$. The Wasserstein gradient flow has a distinct advantage that the potential functional F of the gradient flow is independent of time t; on the other hand, the vector field $\boldsymbol{v}_t$ is usually time-dependent. Furthermore, it often facilitates the understanding of transport maps because we will see that both the Boltzmann entropy and the Renyi entropy are examples of F.

Let $\mathcal{P}_2(\mathbb{R}^m)$ be the L^2-Wasserstein space defined in Section 1.3.3, and let $\mu_t \in \mathcal{P}_2(\mathbb{R}^m)$ be the solution of the continuity equation (9) with initial distribution $\mu_0 \in \mathcal{P}_2(\mathbb{R}^m)$. Then, the map $t \mapsto \mu_t$ plots a curve in $\mathcal{P}_2(\mathbb{R}^m)$. According to the Otto calculus (Villani, 2009, § 23), this curve coincides with a functional gradient flow in $\mathcal{P}_2(\mathbb{R}^m)$, called the Wasserstein gradient flow, with respect to some *potential functional* $F : \mathcal{P}_2(\mathbb{R}^m) \to \mathbb{R}$.

Specifically, we further assume that the vector field $\boldsymbol{v}_t$ is given by the gradient vector field ∇V_t of a potential function $V_t : \mathbb{R}^m \to \mathbb{R}$.

Fact 3 *Assume that μ_t satisfies the continuity equation with the gradient vector field,*

$$\partial_t \mu_t = -\nabla \cdot [\mu_t \nabla V_t], \tag{10}$$

and that we have found F that satisfies the following equation:

$$\frac{\mathrm{d}}{\mathrm{d}t} F[\mu_t] = \int_{\mathbb{R}^m} \nabla V_t(\boldsymbol{x})[\partial_t \mu_t](\boldsymbol{x}) \mathrm{d}\boldsymbol{x}. \tag{11}$$

Then, the Wasserstein gradient flow

$$\frac{\mathrm{d}}{\mathrm{d}t} \mu_t = -\mathsf{grad}\, F[\mu_t], \tag{12}$$

coincides with the continuous equation.

Here, grad denotes the gradient operator on L^2-Wasserstein space $\mathcal{P}_2(\mathbb{R}^m)$ explained in Section 1.3.3. While (12) is an evolution equation or an ODE in $\mathcal{P}_2(\mathbb{R}^m)$, (9) is a PDE in $\mathbb{R}^m$. Hence, we use different notations for the time derivatives, $\frac{\mathrm{d}}{\mathrm{d}t}$ and ∂_t.

3. Denoising Autoencoder

We formulate the denoising autoencoder (DAE) as a variational problem, and we show that the minimizer $\boldsymbol{g}^*$ or the training result is a transport map. Even though the term "DAE" refers to a training procedure of neural networks, we refer to the minimizer of DAE also as a "DAE." We further investigate the initial velocity vector field $\partial_t \boldsymbol{g}_{t=0}$ for mass transportation, and we show that the data distribution μ_t evolves according to the continuity equation.

For the sake of simplicity, we assume that the hidden unit number of NNs is sufficiently large (or infinite), and thus the NNs can always attain the minimum. Furthermore, we assume the the size of data set is sufficiently large (or infinite). In the case when the hidden unit number and the size of data set are both finite, we understand the DAE $\boldsymbol{g}$ is composed of the minimizer $\boldsymbol{g}^*$ and the residual term $\boldsymbol{h}$. Namely, $\boldsymbol{g} = \boldsymbol{g}^* + \boldsymbol{h}$. However, theoretical investigations on the approximation and estimation error $\boldsymbol{h}$ remain as our future work.

3.1. Training Procedure of DAE

Let $\boldsymbol{x}$ be an m-dimensional random vector that is distributed according to the data distribution μ_0, and let $\widetilde{\boldsymbol{x}}$ be its corruption defined by

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}, \quad \boldsymbol{\varepsilon} \sim \nu_t$$

where ν_t denotes the noise distribution parametrized by variance $t \geq 0$. A basic example of ν_t is the Gaussian noise with mean 0 and variance t, i.e., $\nu_t = N(0, tI)$.

The DAE is a function that is trained to remove corruption $\widetilde{\boldsymbol{x}}$ and restore it to the original $\boldsymbol{x}$; this is equivalent to finding a function $\boldsymbol{g}$ that minimizes an objective function, i.e.,

$$L[\boldsymbol{g}] := \mathbb{E}_{\boldsymbol{x},\widetilde{\boldsymbol{x}}} |\boldsymbol{g}(\widetilde{\boldsymbol{x}}) - \boldsymbol{x}|^2. \tag{13}$$

Note that as long as $\boldsymbol{g}$ is a universal approximator and can thus attain the minimum, it need not be a neural network. Specifically, our analysis in this section and the next section is applicable to a wide range of learning machines. Typical examples of $\boldsymbol{g}$ include neural networks with a sufficiently large number of hidden units, splines (Wahba, 1990), kernel machines (Shawe-Taylor and Cristianini, 2004) and ensemble models (Schapire and Freund, 2012).

3.2. Transport Map of DAE

Theorem 4 *(Modification of Theorem 1 by Alain and Bengio, 2014). The global minimum $\boldsymbol{g}_t^*$ of $L[\boldsymbol{g}]$ is attained at*

$$\boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}) = \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{x} \nu_t(\widetilde{\boldsymbol{x}} - \boldsymbol{x}) \mu_0(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}, \tag{14}$$

$$= \widetilde{\boldsymbol{x}} \underbrace{- \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{\varepsilon} \nu_t(\boldsymbol{\varepsilon}) \mu_0(\widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon}) \mathrm{d}\boldsymbol{\varepsilon},}_{=: \boldsymbol{f}_t(\widetilde{\boldsymbol{x}})} \tag{15}$$

where $*$ denotes the convolution operator.

Here, the second equation is simply derived by changing the variable $\boldsymbol{x} \leftarrow \widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon}$ (see Appendix A for the complete proof, where we used the calculus of variations). Note that this calculation first appeared in Alain and Bengio (2014, Theorem 1), where the authors obtained (14).

The DAE $\boldsymbol{g}_t^*(\boldsymbol{x})$ is composed of the identity term $\boldsymbol{x}$ and the denoising term $\boldsymbol{f}_t(\boldsymbol{x})$. If we assume that $\nu_t \to \delta_t$ as $t \to 0$, then in the limit $t \to 0$, the denoising term $\boldsymbol{f}_t(\boldsymbol{x})$ vanishes and DAE reduces to a traditional autoencoder. We reinterpret the DAE $\boldsymbol{g}_t^*(x)$ as a *transport map with transport time* t that transports the mass at $\boldsymbol{x} \in \mathbb{R}^m$ toward $\boldsymbol{x} + \boldsymbol{f}_t(\boldsymbol{x}) \in \mathbb{R}^m$ with displacement vector $\boldsymbol{f}_t(\boldsymbol{x})$.

3.3. Statistical Interpretation of DAE

In statistics, (15) is known as Brown's representation of the posterior mean (George et al., 2006). This is not just a coincidence, because the DAE $\boldsymbol{g}_t^*$ is an estimator of the mean. Recall that a DAE is trained to retain the original vector $\boldsymbol{x}$, given its corruption $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$. At least in principle, this is nonsense because to retain $\boldsymbol{x}$ from $\widetilde{\boldsymbol{x}}$ means to reverse the random walk $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$ (in Figure 4, the multimodal distributions $\mu_{0.5}$ and $\mu_{1.0}$ indicate its difficulty). Obviously, this is an inverse problem or a statistical estimation problem of the latent vector $\boldsymbol{x}$, given the noised observation $\widetilde{\boldsymbol{x}}$ with the observation model $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\varepsilon}$. According to a fundamental fact of estimation theory, the minimum mean squared error (MMSE) estimator of $\boldsymbol{x}$ given $\widetilde{\boldsymbol{x}}$ is given by the posterior mean $\mathbb{E}[\boldsymbol{x}|\widetilde{\boldsymbol{x}}]$. In our case, the posterior mean equals $\boldsymbol{g}_t^*$.

$$\mathbb{E}[\boldsymbol{x}|\widetilde{\boldsymbol{x}}] = \frac{\int_{\mathbb{R}^m} \boldsymbol{x} p(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}) p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}}{\int_{\mathbb{R}^m} p(\widetilde{\boldsymbol{x}} \mid \boldsymbol{x}') p(\boldsymbol{x}') \mathrm{d}\boldsymbol{x}'} = \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{x} \nu_t(\widetilde{\boldsymbol{x}} - \boldsymbol{x}) \mu_0(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}). \tag{16}$$

Similarly, we can interpret the denoising term $\boldsymbol{f}_t(\widetilde{\boldsymbol{x}})$ as the posterior mean $\mathbb{E}[\boldsymbol{\varepsilon}|\widetilde{\boldsymbol{x}}]$ of noise $\boldsymbol{\varepsilon}$ given observation $\widetilde{\boldsymbol{x}}$.

3.4. Examples: Gaussian DAE

When the noise distribution is Gaussian with mean 0 and covariance tI, i.e.,

$$\nu_t(\boldsymbol{\varepsilon}) = \frac{1}{(2\pi t)^{m/2}} e^{-|\boldsymbol{\varepsilon}|^2/2t},$$

the transport map is calculated as follows.

Theorem 5 *The transport map* $\boldsymbol{g}_t^*$ *of Gaussian DAE is given by*

$$\boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}) = \widetilde{\boldsymbol{x}} + t\nabla \log[\nu_t * \mu_0](\widetilde{\boldsymbol{x}}). \tag{17}$$

Proof The proof is straightforward by using Stein's identity,

$$-t\nabla\nu_t(\boldsymbol{\varepsilon}) = \boldsymbol{\varepsilon}\, \nu_t(\boldsymbol{\varepsilon}),$$

which is known to hold only for Gaussians.

$$\begin{aligned}
\boldsymbol{g}_t^*(\widetilde{\boldsymbol{x}}) &= \widetilde{\boldsymbol{x}} - \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} \boldsymbol{\varepsilon}\nu_t(\boldsymbol{\varepsilon})\mu_0(\widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon})\mathrm{d}\boldsymbol{\varepsilon} \\
&= \widetilde{\boldsymbol{x}} + \frac{1}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \int_{\mathbb{R}^m} t\nabla\nu_t(\boldsymbol{\varepsilon})\mu_0(\widetilde{\boldsymbol{x}} - \boldsymbol{\varepsilon})\mathrm{d}\boldsymbol{\varepsilon} \\
&= \widetilde{\boldsymbol{x}} + \frac{t\nabla\nu_t * \mu_0(\widetilde{\boldsymbol{x}})}{\nu_t * \mu_0(\widetilde{\boldsymbol{x}})} \\
&= \widetilde{\boldsymbol{x}} + t\nabla \log[\nu_t * \mu_0(\widetilde{\boldsymbol{x}})].
\end{aligned}$$

■

Theorem 6 *At the initial moment* $t \to 0$*, the pushforward* μ_t *of Gaussian DAE satisfies the* backward heat equation

$$\partial_t \mu_{t=0}(\boldsymbol{x}) = -\triangle\mu_0(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^m, \tag{18}$$

where $\triangle$ *denotes the Laplacian.*

Proof The initial velocity vector is given by the *Fisher score*

$$\partial_t \boldsymbol{g}_{t=0}^*(\boldsymbol{x}) = \lim_{t\to 0} \frac{\boldsymbol{g}_t^*(\boldsymbol{x}) - \boldsymbol{x}}{t} = \nabla \log \mu_0(\boldsymbol{x}). \tag{19}$$

Hence, by substituting the score (19) in the continuity equation (9), we have

$$\partial_t \mu_{t=0}(\boldsymbol{x}) = -\nabla \cdot [\mu_0(\boldsymbol{x})\nabla \log \mu_0(\boldsymbol{x})] = -\nabla \cdot [\nabla\mu_0(\boldsymbol{x})] = -\triangle\mu_0(\boldsymbol{x}).$$

■

The backward heat equation (BHE) rarely appears in nature. However, of course, the present result is not an error. As mentioned in Section 3.3, the DAE solves an estimation problem. Therefore, in the sense of the mean, the DAE behaves as time reversal. We remark that, as shown by Figure 4, a training result of a DAE with a real NN on a finite data set does not converge to a perfect time reversal of a diffusion process.

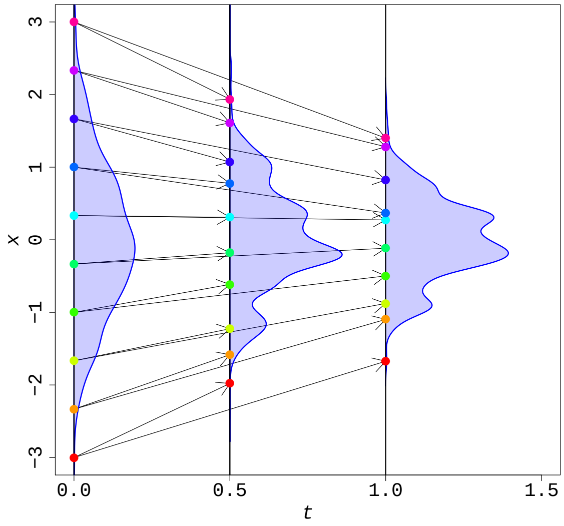


Figure 4: Shallow Gaussian DAE, which is one of the most fundamental versions of DNNs, transports mass, from the left to the right, to decrease the Shannon entropy of data. The x-axis represents the 1-dimensional input/output space, the t-axis represents the variance of the Gaussian noise, and t is the transport time. The leftmost distribution depicts the original data distribution $\mu_0 = N(0,1)$. The middle and rightmost distributions depict the pushforward $\mu_t = \boldsymbol{g}_{t\sharp}\mu_0$, associated with the transportation by two DAEs with noise variance $t = 0.5$ and $t = 1.0$, respectively. As t increases, the variance of the pushforward decreases.

4. Deep DAEs

We introduce the composition $\boldsymbol{g}_L \circ \cdots \circ \boldsymbol{g}_0$ of DAEs $\boldsymbol{g}_\ell : \mathbb{R}^m \to \mathbb{R}^m$ and its continuum limit: the continuous DAE $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$. We can understand the composition of DAEs as the *Euler scheme* or the *broken line approximation* of a continuous DAE.

For the sake of simplicity, we assume that the hidden unit number of NNs is infinite, and that the size of data set is infinite.

4.1. Composition of DAEs

We write $0 = t_0 < t_1 < \cdots < t_{L+1} = t$. We assume that the input vector $\boldsymbol{x}_0 \in \mathbb{R}^m$ is subject to a data distribution μ_0. Let $\boldsymbol{g}_0 : \mathbb{R}^m \to \mathbb{R}^m$ be a DAE that is trained on μ_0 with noise variance $t_1 - t_0$. Then, let $\boldsymbol{x}_1 := \boldsymbol{g}_0(\boldsymbol{x}_0)$, which is a random vector in $\mathbb{R}^m$ that is subject to the pushforward $\mu_1 := \boldsymbol{g}_{0\sharp}\mu_0$. We train another DAE $\boldsymbol{g}_1 : \mathbb{R}^m \to \mathbb{R}^m$ on μ_1 with noise variance $t_2 - t_1$. By repeating the procedure, we obtain $\boldsymbol{g}_\ell(\boldsymbol{x}_\ell)$ from $\boldsymbol{x}_{\ell-1}$ that is subject to $\mu_\ell := \boldsymbol{g}_{(\ell-1)\sharp}\mu_{\ell-1}$.

For the sake of generality, we assume that each component DAE is given by

$$\boldsymbol{g}_\ell(\boldsymbol{x}) = \boldsymbol{x} + (t_{\ell+1} - t_\ell)\nabla V_{t_\ell}(\boldsymbol{x}), \quad (\ell = 0, \ldots, L) \tag{20}$$

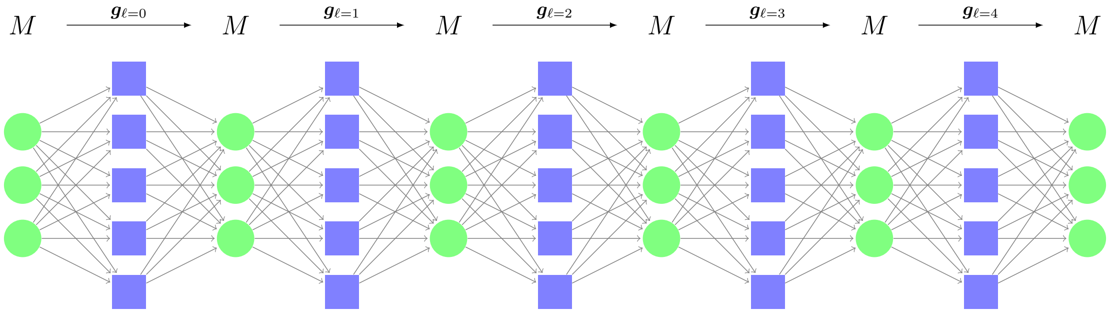


Figure 5: Composition of DAEs $\boldsymbol{g}^t_{0:4} : M \to M$, or the composite of five shallow DAEs $M \to M$, where $M = \mathbb{R}^3$

where V_{t_ℓ} denotes a certain potential function. For example, the Gaussian DAE satisfies the requirement because $V_{t_\ell} = \log[\nu_{t_\ell} * \mu_{t_\ell}]$.

We abbreviate the composition of DAEs by

$$\boldsymbol{g}^t_{0:L}(\boldsymbol{x}) := \boldsymbol{g}_L \circ \cdots \circ \boldsymbol{g}_0(\boldsymbol{x}). \tag{21}$$

By definition, the "velocity" of a composition of DAEs coincides with the vector field

$$\frac{\boldsymbol{g}^{t_{\ell+1}}_{0:\ell}(\boldsymbol{x}) - \boldsymbol{g}^{t_\ell}_{0:(\ell-1)}(\boldsymbol{x})}{t_{\ell+1} - t_\ell} = \nabla V_{t_\ell}(\boldsymbol{x}). \tag{22}$$

4.2. Continuous DAE

We fix the total time t, take the limit $L \to \infty$ of the layer number L, and introduce the continuous DAE as the limit of the "infinite composition of DAEs" $\lim_{L\to\infty} \boldsymbol{g}^t_{0:L}$.

Definition 4 *We call the solution operator or flow $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$ of the following dynamical systems as the* continuous DAE *associated with vector field ∇V_t.*

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \nabla V_t(\boldsymbol{x}(t)), \quad t \geq 0. \tag{23}$$

Proof According to the Cauchy-Lipschitz theorem or the Picard-Lindelöf theorem, when the vector field ∇V_t is continuous in t and Lipschitz in $\boldsymbol{x}$, the limit $\lim_{L\to\infty} \boldsymbol{g}_{0:L}$ converges to a continuous DAE (23) because the trajectory $t \mapsto \boldsymbol{g}_{0:L}(x_0)$ corresponds to a broken line approximation of the integral curve $t \mapsto \boldsymbol{\varphi}_t(x)$. ■

The following properties are immediate from Fact 2 and Fact 3. Let $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$ be the continuous DAE associated with vector field ∇V_t. Given the data distribution μ_0, the pushforward $\mu_t := (\boldsymbol{\varphi}_t)_\sharp \mu_0$ evolves according to the continuity equation

$$\partial_t \mu_t(\boldsymbol{x}) = -\nabla \cdot [\mu_t(\boldsymbol{x}) \nabla V_t(\boldsymbol{x})], \quad t \geq 0 \tag{24}$$

and the Wasserstein gradient flow

$$\frac{\mathrm{d}}{\mathrm{d}t}\mu_t = -\mathsf{grad}\, F[\mu_t], \quad t \geq 0 \tag{25}$$

where F is given by (11).

4.3. Example: Gaussian DAE

We consider a continuous Gaussian DAE $\boldsymbol{\varphi}_t$ trained on $\mu_0 \in \mathcal{P}_2(\mathbb{R}^m)$. Specifically, it satisfies

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}(t) = \nabla \log[\mu_t(\boldsymbol{x}(t))], \quad t \geq 0 \tag{26}$$

with $\mu_t := \boldsymbol{\varphi}_{t\sharp}\mu_0$.

Theorem 7 *The pushforward $\mu_t := \boldsymbol{\varphi}_{t\sharp}\mu_0$ of the continuous Gaussian DAE $\boldsymbol{\varphi}_t$ is the solution to the initial value problem of the backward heat equation (BHE)*

$$\partial_t \mu_t(\boldsymbol{x}) = -\triangle \mu_t(\boldsymbol{x}), \quad \mu_{t=0}(\boldsymbol{x}) = \mu_0(\boldsymbol{x}). \tag{27}$$

The proof is immediate from Theorem 6.

As mentioned after Theorem 6, the BHE appears because the DAE solves an estimation problem. We remark that the BHE is equivalent to the following *final value problem* for the ordinary heat equation:

$$\partial_t u_t(\boldsymbol{x}) = \triangle u_t(\boldsymbol{x}), \quad u_{t=T}(\boldsymbol{x}) = \mu_0(\boldsymbol{x}) \quad \text{for some } T$$

where u_t denotes a probability measure on $\mathbb{R}^m$. Indeed, $\mu_t(\boldsymbol{x}) = u_{T-t}(\boldsymbol{x})$ solves (27). In other words, the backward heat equation describes the time reversal of an ordinary diffusion process.

According to Wasserstein geometry, an ordinary heat equation corresponds to a Wasserstein gradient flow that *increases* the Shannon entropy functional $H[\mu] := -\int \mu(\boldsymbol{x}) \log \mu(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}$ (Villani, 2009, Th. 23.19). Consequently, we can conclude that the continuous Gaussian DAE is a transport map that *decreases* the Shannon entropy of the data distribution.

Theorem 8 *The pushforward $\mu_t := \boldsymbol{\varphi}_{t\sharp}\mu_0$ evolves according to the Wasserstein gradient flow with respect to the Shannon entropy*

$$\frac{\mathrm{d}}{\mathrm{d}t}\mu_t = -\mathsf{grad}\, H[\mu_t], \quad \mu_{t=0} = \mu_0. \tag{28}$$

Proof When $F = H$, then $V_t = -\log \mu_t$; thus,

$$\mathsf{grad}\, H[\mu_t] = \nabla \cdot [\mu_t \nabla \log \mu_t] = \nabla \cdot [\nabla \mu_t] = \triangle \mu_t,$$

which means that the continuity equation reduces to the backward heat equation. ■

4.4. Example: Renyi Entropy

Similarly, when F is the Renyi entropy

$$H^\alpha[\mu] := \int_{\mathbb{R}^m} \frac{\mu^\alpha(\boldsymbol{x}) - \mu(\boldsymbol{x})}{\alpha - 1} \mathrm{d}\boldsymbol{x},$$

then $\mathsf{grad}\, H^\alpha[\mu_t] = \triangle \mu_t^\alpha$ (see Ex. 15.6 in Villani, 2009, for the proof) and thus the continuity equation reduces to the *backward porous medium equation*

$$\partial_t \mu_t(\boldsymbol{x}) = -\triangle \mu_t^\alpha(\boldsymbol{x}). \tag{29}$$

5. Further Investigations on Shallow and Deep DAEs through Examples

5.1. Analytic Examples

We list analytic examples of shallow and continuous DAEs (see Appendix D for further details, including proofs). In all the settings, the continuous DAEs attain a singular measure at some finite $t > 0$ with various singular supports that reflect the initial data distribution μ_0, while the shallow DAEs accept any $t > 0$ and degenerate to a point mass as $t \to \infty$.

5.1.1. Univariate Normal Distribution

When the data distribution is a univariate normal distribution $N(m_0, \sigma_0)$, the transport map and pushforward for the *shallow* DAE are given by

$$g_t(x) = \frac{\sigma_0^2}{\sigma_0^2 + t} x + \frac{t}{\sigma_0^2 + t} m_0, \tag{30}$$

$$\mu_t = N\left(m_0, \frac{\sigma_0^2}{(1 + t/\sigma_0^2)^2} \right), \tag{31}$$

and those of the *continuous* DAE are given by

$$g_t(x) = \sqrt{1 - 2t/\sigma_0^2}(x - m_0) + m_0, \tag{32}$$

$$\mu_t = N(m_0, \sigma_0^2 - 2t). \tag{33}$$

5.1.2. Multivariate Normal Distribution

When the data distribution is a multivariate normal distribution $N(\boldsymbol{m}_0, \Sigma_0)$, the transport map and pushforward for the *shallow* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) = (I + t\Sigma_0^{-1})^{-1} \boldsymbol{x} + (I + t^{-1}\Sigma_0)^{-1} \boldsymbol{m}_0, \tag{34}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0 (I + t\Sigma_0^{-1})^{-2}), \tag{35}$$

and those of the *continuous* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sqrt{I - 2t\Sigma_0^{-1}}(\boldsymbol{x} - \boldsymbol{m}_0) + \boldsymbol{m}_0, \tag{36}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0 - 2tI). \tag{37}$$

5.1.3. Mixture of Multivariate Normal Distributions

When the data distribution is a mixture of multivariate normal distributions $\sum_{k=1}^{K} w_k N(\boldsymbol{m}_k, \Sigma_k)$ with the assumption that it is *well separated*, the transport map and pushforward for the *shallow* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sum_{k=1}^{K} \gamma_{kt}(\boldsymbol{x}) \left\{ (I + t\Sigma_k^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_k)^{-1}\boldsymbol{m}_k \right\}, \tag{38}$$

$$\mu_t \approx \sum_{k=1}^{K} w_k N(\boldsymbol{m}_k, \Sigma_k(I + t\Sigma_k^{-1})^{-2}), \tag{39}$$

with responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^{K} w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}, \tag{40}$$

and those of the *continuous* DAE are given by

$$\boldsymbol{g}_t(\boldsymbol{x}) \approx \sqrt{I - 2t\Sigma_k^{-1}}(\boldsymbol{x} - \boldsymbol{m}_k) + \boldsymbol{m}_k, \tag{41}$$

$$\mu_t = \sum_{k=1}^{K} w_k N(\boldsymbol{m}_k, \Sigma_k - 2tI), \tag{42}$$

with responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}{\sum_{k=1}^{K} w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}. \tag{43}$$

Here, we say that the mixture $\sum_{k=1}^{K} w_k N(\boldsymbol{m}_k, \Sigma_k)$ is well separated when for every cluster center $\boldsymbol{m}_k$, there exists a neighborhood Ω_k of $\boldsymbol{m}_k$ such that $N(\Omega_k; \boldsymbol{m}_k, \Sigma_k) \approx 1$ and $\gamma_{kt} \approx \mathbf{1}_{\Omega_k}$.

5.2. Numerical Example of Trajectories

We employed 2-dimensional examples, in order to visualize the difference of vector fields between the shallow and deep DAEs. In the examples below, every trajectories are drawn into attractors, however the shape of the attractors and the speed of trajectories are significantly different between shallow and deep.

5.2.1. Bivariate Normal Distribution

Figure 6 compares the trajectories of four DAEs trained on the common data distribution

$$\mu_0 = N\left([0, 0], \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\right). \tag{44}$$

The transport maps for computing the trajectories are given by (34) for the shallow DAE and composition of DAEs, and by (36) for the continuous DAE. Here, we applied (34) multiple times for the composition of DAEs.

The continuous DAE converges to an attractor lying on the x-axis at $t = 1/2$. By contrast, the shallow DAE slows down as $t \to \infty$ and never attains the singularity in finite time. As L tends to infinity, $\boldsymbol{g}_{0:L}^t$ plots a trajectory similar to that of the continuous DAE $\boldsymbol{\varphi}_t$; the curvature of the trajectory changes according to Δt.

5.2.2. Mixture of Bivariate Normal Distributions

Figure 7, 8, and 9 compare the trajectories of four DAEs trained on the three common data distributions

$$\mu_0 = 0.5\, N\left([-1,0], \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right) + 0.5\, N\left([1,0], \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right), \tag{45}$$

$$\mu_0 = 0.2\, N\left([-1,0], \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right) + 0.8\, N\left([1,0], \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right), \tag{46}$$

$$\mu_0 = 0.2\, N\left([-1,0], \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\right) + 0.8\, N\left([1,0], \begin{bmatrix}2 & 0\\ 0 & 1\end{bmatrix}\right). \tag{47}$$

respectively.

The transport maps for computing the trajectories are given by (38) for the shallow DAE and composition of DAEs. For the continuous DAE, we compute the trajectories by numerically solving the definition of the continuous Gaussian DAE: $\dot{\boldsymbol{x}} = \nabla \log \mu_t(\boldsymbol{x})$.

In any case, the continuous DAE converges to an attractor at some $t > 0$, but the shape of the attractors and the basins of attraction change according to the initial data distribution. The shallow DAE converges to the origin as $t \to \infty$, and the composition of DAEs plots a curve similar to that of the continuous DAE as L tends to infinity, $\boldsymbol{g}_{0:L}^t$. In particular, in Figure 8, some trajectories of the continuous DAE intersect, which implies that the velocity vector field $\boldsymbol{v}_t$ is time-dependent.

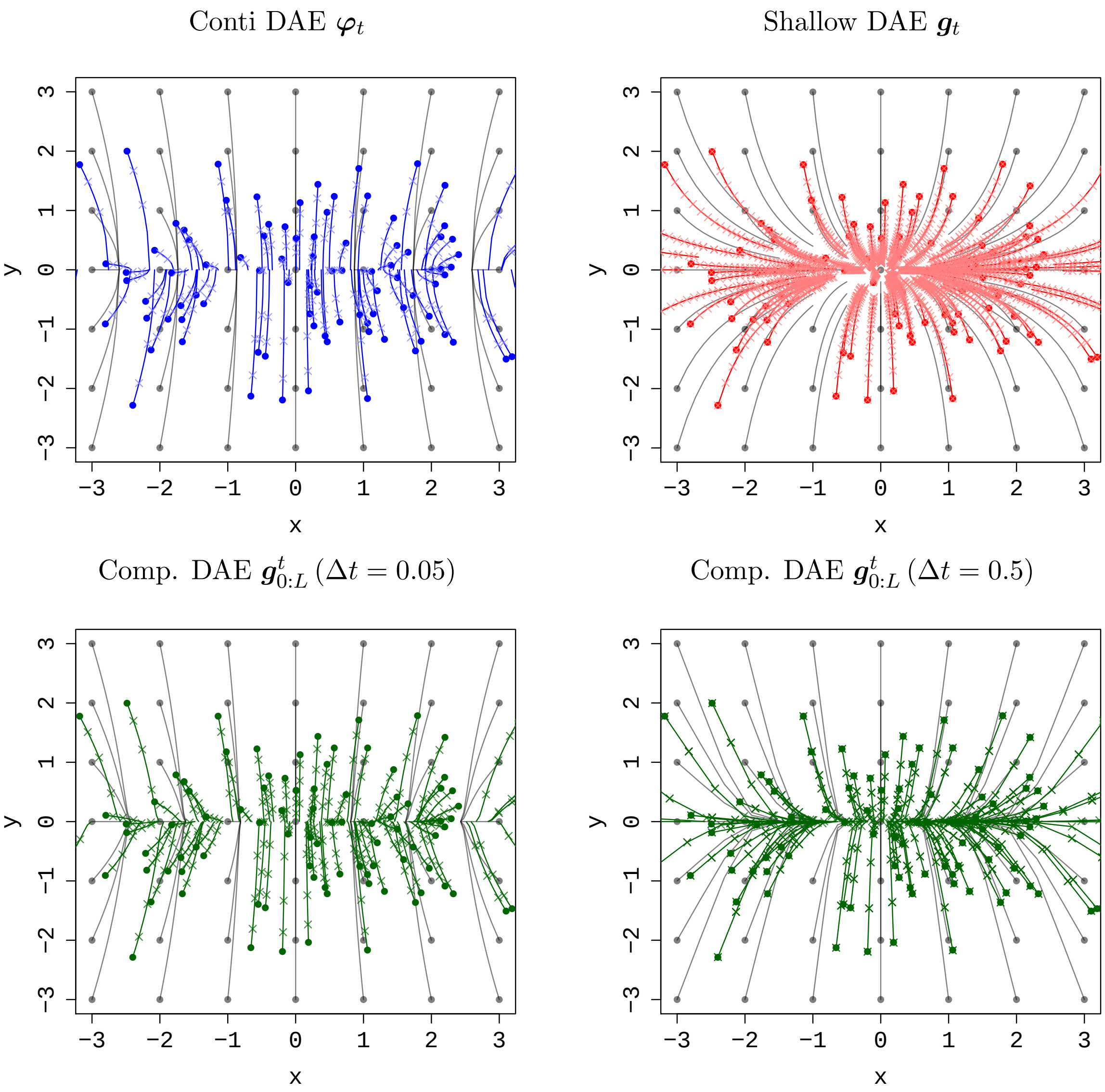


Figure 6: Trajectories of DAEs trained on the common data distribution (44) ($\mu_0 = N([0,0], \mathsf{diag}\,[2,1])$). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. The **midpoints** are plotted every $\Delta t = 0.2$. Every lines are drawn into attractors.

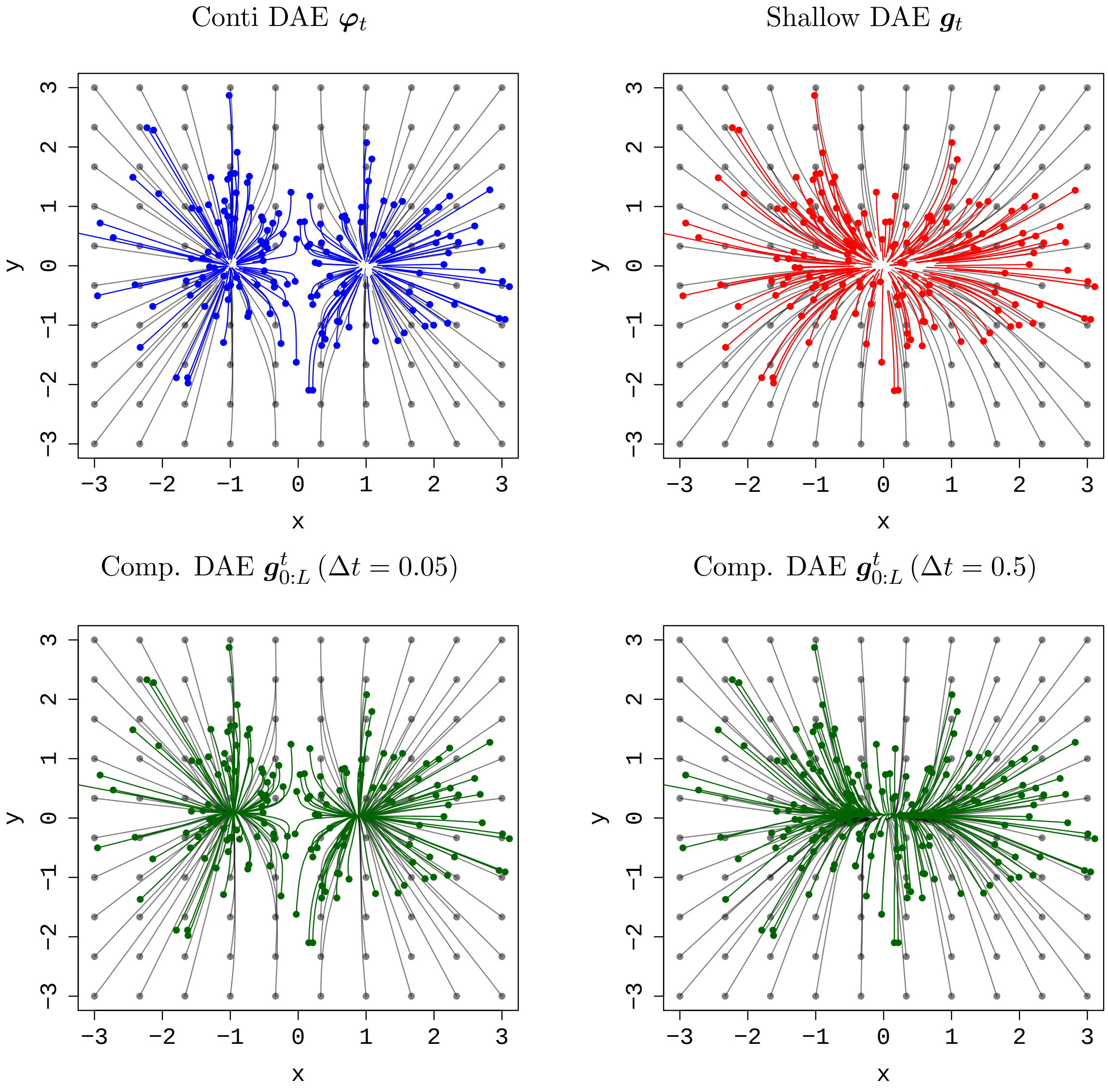
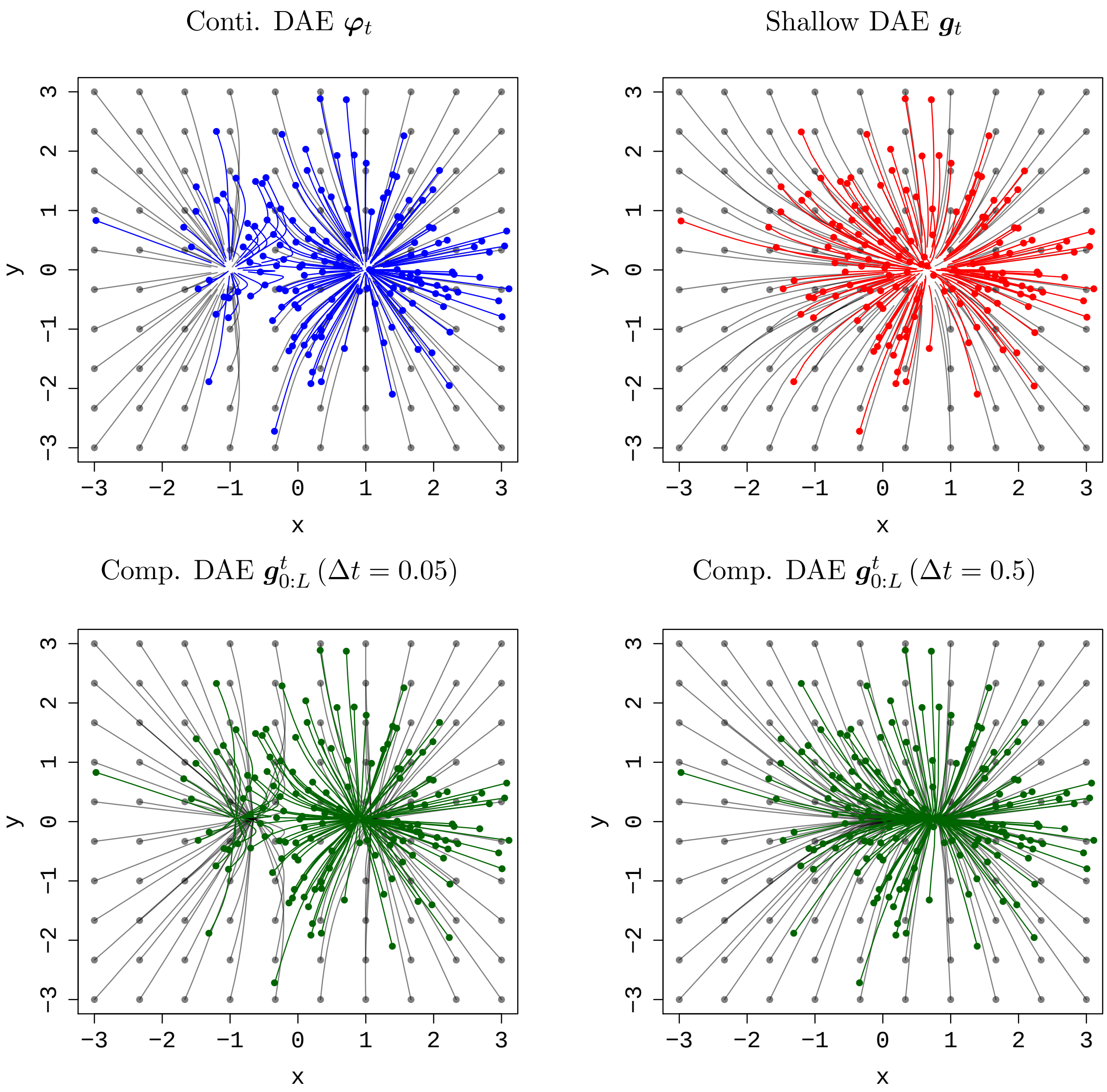


Figure 7: Trajectories of DAEs trained on the common data distribution (45) (a GMM with uniform weight and covariance). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. Every lines are drawn into attractors.



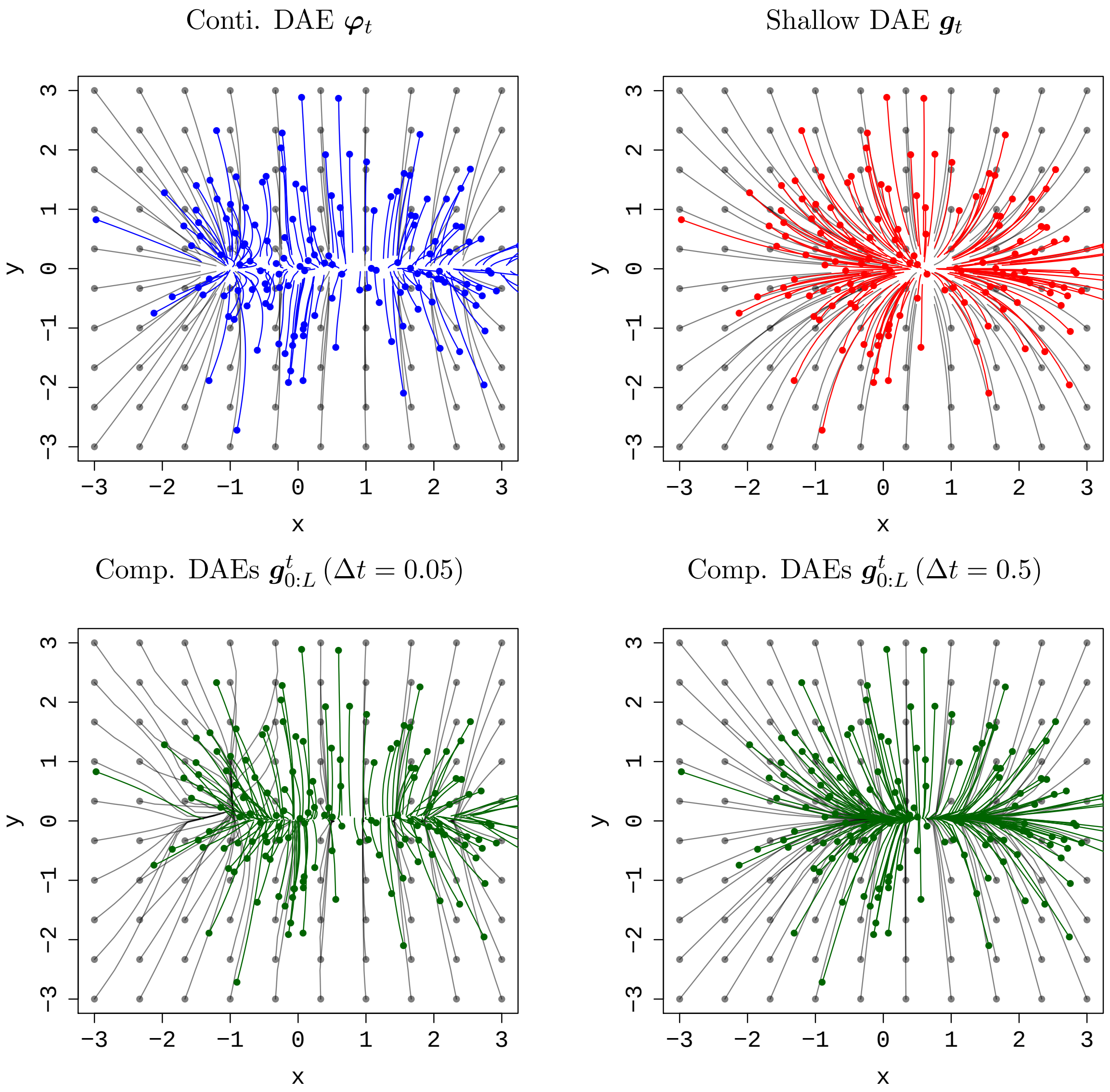
Conti. DAE $\boldsymbol{\varphi}_t$

Shallow DAE $\boldsymbol{g}_t$

Comp. DAE $\boldsymbol{g}^t_{0:L}$ ($\Delta t = 0.05$)

Comp. DAE $\boldsymbol{g}^t_{0:L}$ ($\Delta t = 0.5$)

Figure 8: Trajectories of DAEs trained on the common data distribution (46) (a GMM with non-uniform weight and uniform covariance). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. Every lines are drawn into attractors.



Conti. DAE $\boldsymbol{\varphi}_t$

Shallow DAE $\boldsymbol{g}_t$

Comp. DAEs $\boldsymbol{g}_{0:L}^t$ ($\Delta t = 0.05$)

Comp. DAEs $\boldsymbol{g}_{0:L}^t$ ($\Delta t = 0.5$)

Figure 9: Trajectories of DAEs trained on the common data distribution (47) (a GMM with non-uniform weight and covariance). The **gray lines** start from the regular grid. The **colored lines** start from the samples drawn from μ_0. Every lines are drawn into attractors.

5.3. Numerical Example of Trajectories in Wasserstein Space

We consider the space $\mathcal{Q}$ of bivariate Gaussians:

$$\mathcal{Q} := \left\{ N\left([0,0], \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \right) \,\middle|\, \sigma_1, \sigma_2 > 0 \right\}. \tag{48}$$

Obviously, $\mathcal{Q}$ is a 2-dimensional subspace of L^2-Wasserstein space, and it is closed in the actions of the continuous DAE and shallow DAE because the pushforwards are given by (37) and (35), respectively.

We employ (σ_1, σ_2) as the coordinate of $\mathcal{Q}$. This is reasonable because, in this coordinate, the L^2-Wasserstein distance $W_2(\mu, \nu)$ between two points $\mu = (\sigma_1, \sigma_2)$ and $\nu = (\tau_1, \tau_2)$ is simply given by the "Euclidean distance" $W_2(\mu, \nu) = \sqrt{(\sigma_1 - \tau_1)^2 + (\sigma_2 - \tau_2)^2}$ (see Takatsu, 2011, for the proof). The Shannon entropy is given by

$$H(\sigma_1, \sigma_2) = (1/2) \log |\mathsf{diag}\,[\sigma_1^2, \sigma_2^2]| + \mathit{const.} = \log \sigma_1 + \log \sigma_2 + \mathit{const.} \tag{49}$$

Figure 10 compares the trajectories of the pushforward by DAEs in $\mathcal{Q}$. In the left, we calculated the theoretical trajectories according to the analytic formulas (37) and (35). In the right, we trained real NNs as the composition of DAEs according to the training procedure described in Section 4.1. Even though we always assumed the infinite number of hidden units and the infinite size of data set, the results suggest that our calculus is a good approximation to finite settings.

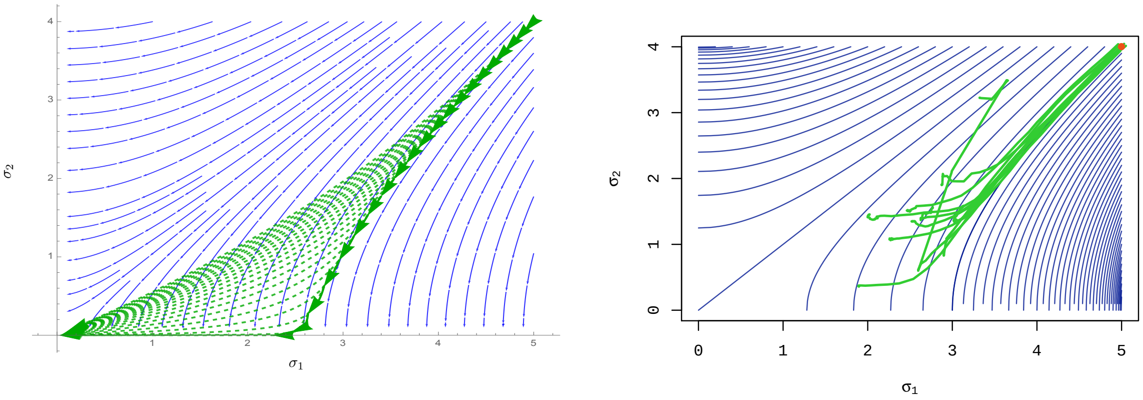


Figure 10: Trajectories of pushforward measures in a space $\mathcal{Q}$ of bivariate Gaussians $N([0,0], \mathsf{diag}\,[\sigma_1^2, \sigma_2^2])$. In **both** sides, the **blue** lines represent the Wasserstein gradient flow with respect to the Shannon entropy. The continuous Gaussian DAE $t \mapsto \boldsymbol{\varphi}_{t\sharp}\mu_0$ always coincides with the blue lines. In the **left-hand side**, the **dashed green** lines represent theoretical trajectories of the shallow DAE $t \mapsto \boldsymbol{g}_{t\sharp}\mu_0$ and the **solid green** line represents a theoretical trajectory of the composition of DAEs $t \mapsto \boldsymbol{g}^t_{0:L\sharp}\mu_0$. Both the green lines gradually leave the gradient flow. In the **right-hand side**, the **solid green** lines represent the trajectories of the composition of DAEs calculated by training real NNs (10 trials). In particular, in the early stage, the trajectories are parallel to the gradient flow.

6. Equivalence between Stacked DAE and Compositions of DAEs

As an application of transport analysis, we shed light on the equivalence of the stacked DAE (SDAE) and the composition of DAEs (CDAE), provided that the definition of DAEs is generalized to *L-DAE*, which is defined below. In SDAE, we apply the DAE to the features vectors obtained from the hidden layer of an NN to obtain higher-order feature vectors. Therefore, the feature vectors obtained from the SDAE and CDAE are different from each other. Nevertheless, we can prove that the trajectories generated by the SDAE and CDAE are *topologically conjugate*, which means that there exists a homeomorphism between the trajectories. Moreover, we can transform the trajectory of an SDAE into that of a CDAE by using a *linear map*, which is obtained from the decoder of the SDAE. Thus, we can synthesize the feature vectors of the SDAE by using CDAEs.

6.1. Definitions

To begin with, we introduce a generalized version of shallow DAE.

Definition 5 (L-DAE) *Let L be an elliptic operator on the domain Ω in $\mathbb{R}^m$, μ be a probability density on Ω, and D be a positive definite matrix. The L-DAE with diffusion coefficient D and initial data μ is defined by*

$$\mathsf{id} + tD\nabla \log e^{tL}\mu, \quad t > 0. \tag{50}$$

Here, e^{tL} is the semigroup generated by the elliptic operator L. Specifically, let $\mu_t := e^{tL}\mu$; then, μ_t satisfies the parabolic equation $\partial_t \mu_t = L\mu_t$. The original Gaussian DAE corresponds to a special case when $D \equiv I$ and $L = \triangle$.

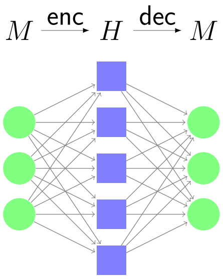
By dae, we denote a DAE realized by a shallow NN (Figure 11). Specifically,

$$\mathsf{dae}(\boldsymbol{x}) = \sum_{j=1}^{p} \boldsymbol{c}_j \sigma(\boldsymbol{a}_j \cdot \boldsymbol{x} - b_j). \tag{51}$$

By enc and dec, we denote the encoder and decoder of dae, respectively. Specifically,

$$\mathsf{enc}_j(\boldsymbol{x}) = \sigma(\boldsymbol{a}_j \cdot \boldsymbol{x} - b_j), \quad j = 1, \ldots, p \tag{52}$$

$$\mathsf{dec}(\boldsymbol{z}) = \sum_{j=1}^{p} \boldsymbol{c}_j z_j, \tag{53}$$



$$\begin{tikzcd}
& H \arrow[dr, "\mathsf{dec}"] & \\
M \arrow[ur, "\mathsf{enc}"] \arrow[rr, "\mathsf{dae}"] & & M
\end{tikzcd}$$

Figure 11: enc and dec correspond to the hidden layer and output layer, respectively.

where z_j denotes the j-th element of $\boldsymbol{z} = \mathsf{enc}(\boldsymbol{x})$. Obviously, $\mathsf{dae} = \mathsf{dec} \circ \mathsf{enc}$.

For the sake of simipicity, even though we introduced the finite number p of hidden units, we assume that p is large, and thus dae approximately equals L-DAE for some L.

6.2. Training Procedure of Stacked DAE (SDAE)

Let $M := \mathbb{R}^m$ be the space of input vectors with probability density μ, and let $\mathsf{dae} : M \to M$ be a shallow NN with p hidden units. We assume that dae is trained as the Gaussian DAE with μ, and it thus approximates the DAE $\mathsf{id} + t\nabla \log[e^{t\triangle}\mu]$. Let $H := \mathbb{R}^p$. Then, the encoder and decoder of dae are the maps $\mathsf{enc} : M \to H$ and $\mathsf{dec} : H \to M$, respectively.

In the SDAE, we apply the DAE to $\boldsymbol{z}$. Specifically, let $\widetilde{\mu}$ be the density of hidden feature vectors $\boldsymbol{z} = \mathsf{enc}(\boldsymbol{x})$, and let $\widetilde{\mathsf{dae}} : H \to H$ be a shallow NN with $\widetilde{p}$ hidden units,

$$\widetilde{\mathsf{dae}}(\boldsymbol{z}) := \sum_{\widetilde{j}=1}^{\widetilde{p}} \widetilde{\boldsymbol{c}}_{\widetilde{j}} \sigma(\widetilde{\boldsymbol{a}}_{\widetilde{j}} \cdot \boldsymbol{z} - \widetilde{b}_{\widetilde{j}}).$$

We train $\widetilde{\mathsf{dae}}$ by using the Gaussian DAE with $\widetilde{\mu}$, where the network is decomposed as $\widetilde{\mathsf{dae}} = \widetilde{\mathsf{dec}} \circ \widetilde{\mathsf{enc}}$ with $\widetilde{\mathsf{enc}} : H \to \widetilde{H}$ and $\widetilde{\mathsf{dec}} : \widetilde{H} \to H$, and we obtain the feature vectors $\widetilde{\boldsymbol{z}} := \widetilde{\mathsf{enc}}(\boldsymbol{z}) \in \widetilde{H} = \mathbb{R}^{\widetilde{p}}$. By iterating the stacking procedure, we can obtain more abstract feature vectors (Figure 12).

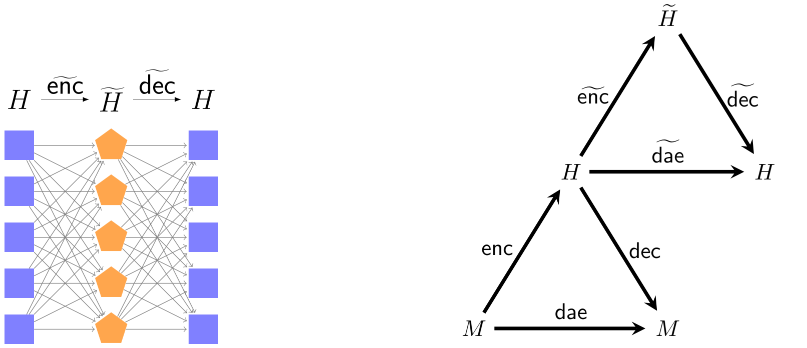

H
$\widetilde{\mathsf{enc}}$
$\widetilde{H}$
$\widetilde{\mathsf{dec}}$
H
$\widetilde{H}$
$\widetilde{\mathsf{enc}}$
$\widetilde{\mathsf{dec}}$
H
$\widetilde{\mathsf{dae}}$
H
enc
dec
M
dae
M

Figure 12: The (feature map of) SDAE $\widetilde{\mathsf{enc}} \circ \mathsf{enc}$ is built on the hidden layer.

Technically speaking, $\widetilde{\mu}$ is (the density of) the pushforward $\mathsf{dae}_\sharp \mu$, and its support is contained in the image $\widetilde{M} := \mathsf{enc}(M)$. In general, we assume that $\dim \widetilde{M} (= \dim M) \leq \dim H$; thus, the support of $\widetilde{\mu}$ is singular (i.e., the density vanishes outside $\widetilde{M}$) (see Fact 1 for further details).

6.3. Topological Conjugacy

The transport map of the feature vector $\widetilde{\mathsf{enc}} \circ \mathsf{enc} : M \to H \to \widetilde{H}$ is somewhat unclear. According to Theorem 9 and 10, the transport map of $\widetilde{\mathsf{enc}} \circ \mathsf{enc}$ can be transformed or projected to the ground space M by applying $\mathsf{dec} \circ \widetilde{\mathsf{dec}}$ (Figure 13). Specifically, there exists an L-DAE $\mathsf{dae}' : M \to M$ such that

$$\mathsf{dec} \circ \widetilde{\mathsf{dec}} \circ \widetilde{\mathsf{enc}} \circ \mathsf{enc} = \mathsf{dae}' \circ \mathsf{dae}. \tag{54}$$

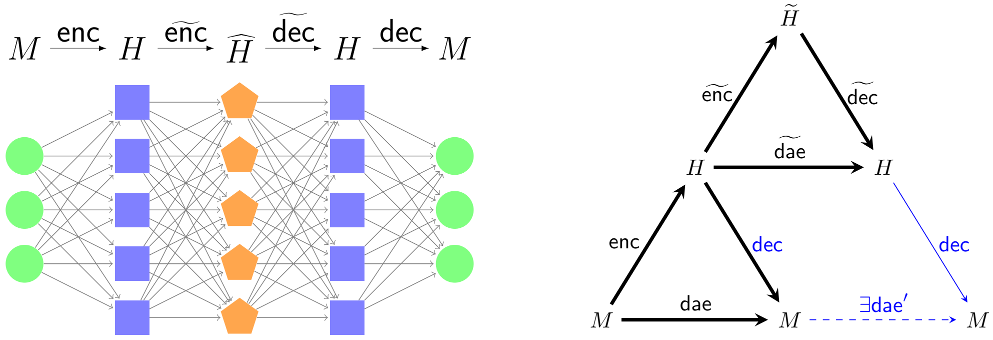

M
enc
H
$\widetilde{\mathsf{enc}}$
$\widehat{H}$
$\widetilde{\mathsf{dec}}$
H
dec
M
$\widetilde{H}$
$\widetilde{\mathsf{enc}}$
$\widetilde{\mathsf{dec}}$
H
$\widetilde{\mathsf{dae}}$
H
enc
dec
dec
M
dae
M
$\exists \mathsf{dae}'$
M

Figure 13: By reusing dec, we can transform the SDAE $\widetilde{\mathsf{enc}} \circ \mathsf{enc}$ into a CDAE $\mathsf{dae}' \circ \mathsf{dae}$.

Theorem 9 *Let H and $\widetilde{H}$ be vector spaces, $\dim H \geq \dim \widetilde{H}$, let M_0 be an m-dimensional smooth Riemannian manifold embedded in H, and let μ_0 be a C^2 probability density on M_0. Let $\boldsymbol{f} : H \to H$ be an L_t-DAE:*

$$\boldsymbol{f} := \mathsf{id}_H + tD\nabla \log e^{tL_t}\mu_0,$$

with diffusion coefficient D and time-dependent elliptic operator L_t on H, where ∇ is the gradient operator in H.

Let $T : H \to \widetilde{H}$ be a linear map. If $T|_M$ is injective, then there exists an $\widetilde{L}_t$-DAE $\widetilde{\boldsymbol{f}} : \widetilde{H} \to \widetilde{H}$ with diffusion coefficient $\widetilde{D}$ such that

$$T \circ \boldsymbol{f}|_M = \widetilde{\boldsymbol{f}} \circ T|_M. \tag{55}$$

In other words, the following diagram commutes. Here we denoted $M_1 := \boldsymbol{f}(M_0)$ and

$$\begin{tikzcd}
H & (M_0, \mu_0) \arrow[r, "\boldsymbol{f}"] \arrow[dr, "T"'] & (M_1, \mu_1) \arrow[dr, "T"] & \\
\widetilde{H} & & (\widetilde{M_0}, \widetilde{\mu_0}) \arrow[r, dashed, "\exists \widetilde{\boldsymbol{f}}"] & (\widetilde{M_1}, \widetilde{\mu_1})
\end{tikzcd}$$

$\mu_1 := \boldsymbol{f}_\sharp \mu_0$. See Appendix C for the proof. The statement is general in that the choice of a linear map T is independent of the DAEs, as long as it is injective.

We note that the trajectory of the equivalent DAE $\widetilde{\boldsymbol{f}}$ may be complicated, because the "equivalence" we mean here is simply the topological conjugacy. Actually, as the proof suggests, $\widetilde{D}$ and $\widetilde{L}_t$ contain the non-linearity of activation functions via the pseudo-inverse $T^\dagger$ of T. Nevertheless, $\widetilde{\boldsymbol{f}}$ may not be much complicated because it is simply a linear projection of the high-dimensional trajectory of L_t-DAE. According to Theorem 6, a Gaussian DAE solves backward heat equation (at least when $t \to 0$). Hence, its projection to low dimension should also solve backward heat equation in low dimension spaces.

6.4. Equivalence between SDAE and CDAE

To clarify the statement, we prepare the notation. Figure 14 summarizes the symbols and procedures.

First, we rewrite the input vector as $\boldsymbol{z}^0$ instead of $\boldsymbol{x}$, the input space as $H^0 = M_0^0 (= \mathbb{R}^m)$ instead of M, and the density as μ_0^0 instead of μ. We iteratively train the ℓ-th NN $\mathsf{dae}_\ell^\ell : H^\ell \to H^\ell$ with a data distribution μ_ℓ^ℓ, obtain the encoder $\mathsf{enc}^\ell : H^\ell \to H^{\ell+1}$ and decoder $\mathsf{dec}^\ell : H^{\ell+1} \to H^\ell$, and update the feature $\boldsymbol{z}^{\ell+1} := \mathsf{enc}^\ell(\boldsymbol{z}^\ell)$, the image $M_{\ell+1}^{\ell+1} := \mathsf{enc}^\ell(M_\ell^\ell) \subset H^{\ell+1}$, and the distribution $\mu_{\ell+1}^{\ell+1} := (\mathsf{enc}^\ell)_\sharp \mu_\mu^\ell$.

For simplicity, we abbreviate

$$\begin{aligned} \mathsf{enc}^{\ell:n} &:= \mathsf{enc}^n \circ \cdots \circ \mathsf{enc}^\ell, \\ \mathsf{dec}^{n:\ell} &:= \mathsf{dec}^\ell \circ \cdots \circ \mathsf{dec}^n. \end{aligned}$$

In addition, we introduce auxiliary objects.

$$\begin{aligned} M_{\ell+1}^n &:= \mathsf{dec}^{\ell:n}(M_{\ell+1}^{\ell+1}), \quad n = 0, \cdots, \ell \\ \mu_{\ell+1}^n &:= \mathsf{dec}_\sharp^{\ell:n} \mu_{\ell+1}^{\ell+1}, \quad n = 0, \cdots, \ell. \end{aligned}$$

By construction, M_n^ℓ is an at most m-dimensional submanifold in H^ℓ, and the support of μ_n^ℓ is in M_n^ℓ.

Finally, we denote the map $\mathsf{dae}_n^\ell : M_n^\ell \to M_{n+1}^\ell$ that is (not "trained by DAE" but) defined by

$$\mathsf{dae}_n^\ell := (\mathsf{dec}^{n:\ell} \circ \mathsf{enc}^{0:n}) \circ (\mathsf{dec}^{(n-1):\ell} \circ \mathsf{enc}^{0:(n-1)})^{-1} : M_n^\ell \to M_{n+1}^\ell.$$

By Theorem 9, if $\mathsf{dae}_n^{\ell+1}$ is an $L_n^{\ell+1}$-DAE, then dae_n^ℓ exists and it is an L_n^ℓ-DAE.

Theorem 10 *If every* $\mathsf{enc}^\ell|_{M_\ell^\ell}$ *is a continuous injection and every* $\mathsf{dec}^\ell|_{M_n^{\ell+1}}$ *is an injection, then*

$$\mathsf{dec}^{L:0} \circ \mathsf{enc}^{0:L} = \mathsf{dae}_L^0 \circ \cdots \circ \mathsf{dae}_0^0. \tag{56}$$

Proof By repeatedly applying the topological conjugacy in Theorem 9,

$$\mathsf{dec}^\ell \circ \mathsf{dae}_n^{\ell+1} = \mathsf{dae}_n^\ell \circ \mathsf{dec}^\ell,$$

we have

$$\begin{aligned} &\mathsf{dec}^{L:0} \circ \mathsf{enc}^{0:L} \\ &\quad = \mathsf{dec}^{(L-2):0} \circ \mathsf{dec}^{L-1} \circ \mathsf{dae}_L^L \circ \mathsf{enc}^{L-1} \circ \mathsf{enc}^{0:(L-2)} \\ &\quad = \mathsf{dec}^{(L-2):0} \circ \mathsf{dae}_L^{L-1} \circ \mathsf{dec}^{L-1} \circ \mathsf{enc}^{L-1} \circ \mathsf{enc}^{0:(L-2)} \\ &\quad = \mathsf{dec}^{(L-2):0} \circ \mathsf{dae}_L^{L-1} \circ \mathsf{dae}_{L-1}^{L-1} \circ \mathsf{enc}^{0:(L-2)} \\ &\cdots \\ &\quad = \mathsf{dae}_L^0 \circ \mathsf{dae}_{L-1}^0 \circ \cdots \circ \mathsf{dae}_0^0. \end{aligned}$$

■

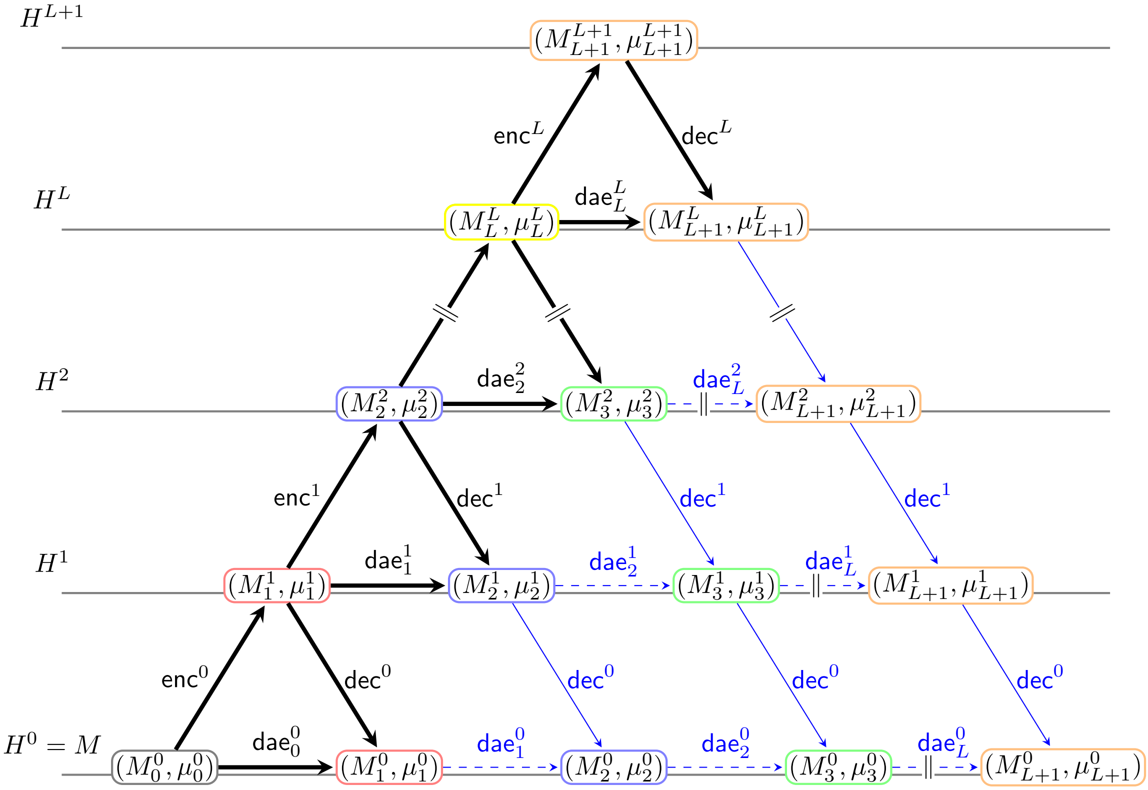


Figure 14: By using decoders, an SDAE is transformed or projected into a CDAE. The leftmost arrows correspond to the SDAE $\mathsf{enc}^{0:L}$, the rightmost arrows correspond to the decoders $\mathsf{dec}^{L:0}$, and the bottom arrows correspond to the CDAE $\mathsf{dae}^0_L \circ \cdots \circ \mathsf{dae}^0_0$.

6.5. Numerical Example

Figure 15 compares the transportation results of the 2-dimensional swissroll data by the DAEs. In both the cases, the swissroll becomes thinner by the action of transportation. We remark that to test the topological conjugacy by numerical experiments is difficult. Here, we display Figure 15 to see typical trajectories by an SDAE and a CDAE.

In the left-hand side, we trained an SDAE $\mathsf{enc}^1 \circ \mathsf{enc}^0$ by using real NNs. Specifically, we first trained a shallow DAE dae^0_0 on the swissroll data $\boldsymbol{x}_0$. Second, writing $\mathsf{dae}^0_0 = \mathsf{dec}_0 \circ \mathsf{enc}_0$ and letting $\boldsymbol{z}^1 := \mathsf{enc}_0(\boldsymbol{x}_0)$, we trained a shallow DAE dae^1_1 on the feature vectors $\boldsymbol{z}^1$. Then, writing $\mathsf{dae}^1_1 = \mathsf{dec}_1 \circ \mathsf{enc}_1$, we obtained $\boldsymbol{x}_1 := \mathsf{dae}^0_0(\boldsymbol{x}_0)$ and $\boldsymbol{x}_2 := \mathsf{dec}^0 \circ \mathsf{dec}^1 \circ \mathsf{enc}^1 \circ \mathsf{enc}^0$. The black points represent the input vectors $\boldsymbol{x}_0$, and the red and blue points represent the first and second transportation results $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively. In other words, the distribution of $\boldsymbol{x}_0, \boldsymbol{x}_1$ and $\boldsymbol{x}_2$ correspond to μ^0_0, μ^0_1 and μ^0_2 in Figure 14, respectively.

In the right-hand side, we trained a CDAE $\mathsf{dae}^1_0 \circ \mathsf{dae}^0_0$ by using real NNs. Specifically, we first trained a shallow DAE dae^0_0 on the swissroll data $\boldsymbol{x}_0$. Second, writing $\boldsymbol{x}_1 := \mathsf{dae}^0_0(\boldsymbol{x}_0)$, we trained a shallow DAE dae^0_1 on the transported vectors $\boldsymbol{x}^0_1$. Then, we obtained $\boldsymbol{x}_2 := \mathsf{dae}^1_0(\boldsymbol{x}_1) = \mathsf{dae}^1_0 \circ \mathsf{dae}^0_0(\boldsymbol{x}_0)$. The black points represent the input vectors $\boldsymbol{x}_0$, and the red and blue points represent the first and second transportation results $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively.

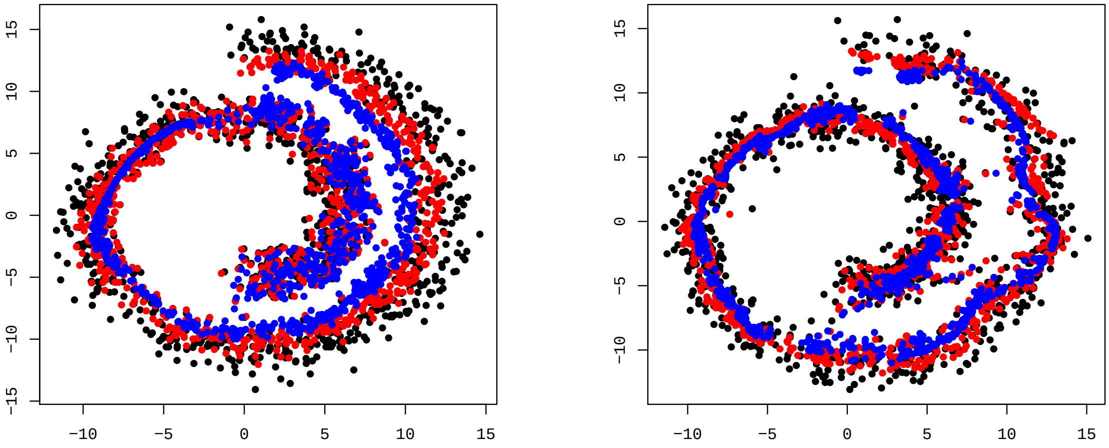


Figure 15: Typical transportation results of the 2-dimensional swissroll data by an SDAE (left) and a CDAE (right). In **both** the sides, the **black** points represent the input vectors $\boldsymbol{x}_0 \in \mathbb{R}^2$, and the **red** and **blue** points represent the first and second transportation results $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, respectively.

7. Integral Representation of the Flow Representation

In this section, we aim to develop the double continuum limit: a combination of the depth continuum limit, or the flow representation, and the width continuum limit, or the integral representation.

To facilitate visualization, we write the hidden parameters as $\boldsymbol{\theta}$ instead of $(\boldsymbol{a}, b)$, the k-th element of the coefficient function as $\gamma(\boldsymbol{\theta}, k)$ or $\gamma_k(\boldsymbol{\theta})$ instead of the boldface $\boldsymbol{\gamma}(\boldsymbol{\theta})$, and the integral representation as

$$S[\gamma_k](\boldsymbol{x}) = \int \gamma(\boldsymbol{\theta}, k)\sigma(\boldsymbol{x}; \boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}. \tag{57}$$

Furthermore, by using a singular measure $\gamma_k^p(\boldsymbol{\theta}) := \sum_{j=1}^p c_{jk}\delta_{\boldsymbol{\theta}_j}(\boldsymbol{\theta})$, we write an ordinary shallow NN as

$$S[\gamma_k^p](\boldsymbol{x}) = \int \gamma^p(\boldsymbol{\theta}, k)\sigma(\boldsymbol{x}; \boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} = \sum_{j=1}^p c_{jk}\sigma(\boldsymbol{x}; \boldsymbol{\theta}_j). \tag{58}$$

If there is no risk of confusion, we omit writing the superscript p. Specifically, we write "$S[\gamma_k]$" without distinction between an infinite NN (57) and a finite NN (58).

7.1. Encoder and Decoder in the Integral Representation

First, we consider a finite case. Suppose that a shallow DAE is realized by a finite NN $\sum_{j=1}^{p} c_{jk}\sigma(\boldsymbol{x};\boldsymbol{\theta}_j)$. Then, the encoder is given by

$$z(\boldsymbol{\theta}_j) = \mathsf{enc}(\boldsymbol{x}, \boldsymbol{\theta}_j) = \sigma(\boldsymbol{x};\boldsymbol{\theta}_j), \quad j = 1, \ldots, p;$$

and the decoder is given by

$$\mathsf{dec}(\boldsymbol{z}, k) = \sum_{j=1}^{p} c_{jk} z(\boldsymbol{\theta}_j).$$

Therefore, supposing that a shallow DAE is realized by $S[\gamma]$, the encoder and decoder in the integral representation are given by

$$\mathsf{enc}(\boldsymbol{x}, \boldsymbol{\theta}) := \sigma(\boldsymbol{x};\boldsymbol{\theta}), \tag{59}$$

$$\mathsf{dec}(\boldsymbol{z}, k) := \int \gamma(\boldsymbol{\theta}, k) z(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}, \tag{60}$$

where "the $\boldsymbol{\theta}$-th element" of $\boldsymbol{z}$ is given by $z(\boldsymbol{\theta})$.

Next, we consider the stacked DAE built on $\boldsymbol{z}$. Suppose that the stacked DAE is realized by $S[\widetilde{\gamma}_{\boldsymbol{\theta}}](\boldsymbol{z}) = \int \widetilde{\gamma}(\boldsymbol{\omega}, \boldsymbol{\theta})\sigma(\boldsymbol{z};\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}$; then, the encoder and decoder are given by

$$\widetilde{\mathsf{enc}}(\boldsymbol{z}, \boldsymbol{\omega}) := \sigma(\boldsymbol{z};\boldsymbol{\omega}), \tag{61}$$

$$\widetilde{\mathsf{dec}}(\boldsymbol{u}, \boldsymbol{\theta}) := \int \widetilde{\gamma}(\boldsymbol{\omega}, \boldsymbol{\theta}) u(\boldsymbol{\omega}) \mathrm{d}\boldsymbol{\omega}, \tag{62}$$

where the $\boldsymbol{\omega}$-th element of $\boldsymbol{u}$ is given by $u(\boldsymbol{\omega})$, and the $\boldsymbol{\theta}$-th element of $\boldsymbol{\omega}$ is given by $\omega(\boldsymbol{\theta})$.

In this notation, for example, the topological conjugacy (55) claims that there exists γ' such that

$$\int \gamma(\boldsymbol{\theta}, k) \int \widetilde{\gamma}(\boldsymbol{\omega}, \boldsymbol{\theta})\sigma(\sigma(\boldsymbol{x};\cdot);\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega}\mathrm{d}\boldsymbol{\theta} = \int \gamma'(\boldsymbol{\theta}', k)\sigma\left(\int \gamma(\boldsymbol{\theta}, \cdot)\sigma(\boldsymbol{x};\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta};\boldsymbol{\theta}'\right)\mathrm{d}\boldsymbol{\theta}'. \tag{63}$$

7.2. Ridgelet Transform of Flows

Let $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$ be a flow that satisfies $\boldsymbol{\varphi}_t \circ \boldsymbol{\varphi}_s = \boldsymbol{\varphi}_{t+s}$. Then, the following formula holds:

$$\int R[\boldsymbol{\varphi}_t](\boldsymbol{\theta}, k)\sigma\left(\int R[\boldsymbol{\varphi}_s](\boldsymbol{\theta}, \cdot)\sigma(\boldsymbol{x};\boldsymbol{\theta}')\mathrm{d}\boldsymbol{\theta}'\right)\mathrm{d}\boldsymbol{\theta} = \int R[\boldsymbol{\varphi}_{t+s}](\boldsymbol{\theta}, k)\sigma(\boldsymbol{x};\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}. \tag{64}$$

In other words, $S[R[\boldsymbol{\varphi}_t]] \circ S[R[\boldsymbol{\varphi}_s]] = S[R[\boldsymbol{\varphi}_{t+s}]]$. According to Barron's bound (Kůrková, 2012, Cor.5.4), the discretization error $\|S[\gamma] - S[\gamma^p]\|_2$ between $S[\gamma]$ and $S[\gamma^p]$ is bounded by $\|\gamma\|_1/\sqrt{p}$. Hence, $\|R[\boldsymbol{\varphi}_t]\|_1 + \|R[\boldsymbol{\varphi}_s]\|_1 \leq \|R[\boldsymbol{\varphi}_{t+s}]\|_1$ for some t and s, which implies the expressive efficiency of the DNN.

Consider a special case when $\boldsymbol{\varphi} : \mathbb{R}^m \to \mathbb{R}^m$ is given by the gradient of a potential function V. Specifically, $\boldsymbol{\varphi} = \nabla V$. We note that according to the polar decomposition theorem by Brenier (1991), any optimal transport map $\boldsymbol{\varphi}_t : [0, 1] \times \mathbb{R}^m \to \mathbb{R}^m$ can be written as $\boldsymbol{\varphi}_t = \mathsf{id} + t\nabla U$ with some potential function U. Hence, by letting $V = |\cdot|^2/2 + U$, we can understand $\boldsymbol{\varphi} := \boldsymbol{\varphi}_1 = \nabla V$ as an optimal transport map.

Then, we have an integration-by-parts formula for the vector ridgelet transform.

Theorem 11 *Let $K \subset \mathbb{R}^m$ be a compact set with smooth boundary ∂K. Given that a smooth scalar potential V is supported in K, the ridgelet transform of the potential vector field ∇V is calculated by*

$$R_\rho[\nabla V](\boldsymbol{a}, b) = -\boldsymbol{a} R_{\rho'}[V](\boldsymbol{a}, b). \tag{65}$$

Here, R_ρ and $R_{\rho'}$ denote the ridgelet transform with respect to ρ and ρ', respectively.

Proof

$$\begin{aligned} R_\rho[\nabla V](\boldsymbol{a}, b) &= \int_K \nabla V(\boldsymbol{x}) \overline{\rho(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \mathrm{d}\boldsymbol{x} \\ &= \left[\int_{\partial K} V(\boldsymbol{x}) \overline{\rho(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \boldsymbol{n}(\boldsymbol{x}) \mathrm{d}S - \boldsymbol{a} \int_K V(\boldsymbol{x}) \overline{\rho'(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \mathrm{d}\boldsymbol{x} \right] \\ &= 0 - \boldsymbol{a}\, R_{\rho'}[V](\boldsymbol{a}, b). \end{aligned}$$

■

The left-hand side (LHS) of (65) denotes a vector ridgelet transform defined by element-wise mapping, whereas the right-hand side (RHS) consists of a scalar ridgelet transform. We can understand the RHS given that the network shares common knowledge among element-wise tasks.

7.3. Example: Autoencoder

As the most fundamental transport map, we consider a smooth "truncated" autoencoder $\mathsf{id}_{r,\delta}$. We denote by $\mathbb{B}^m(\boldsymbol{z}; r)$ a closed ball in $\mathbb{R}^m$ with center $\boldsymbol{z}$ and radius r. We assume that $\mathsf{id}_{r,\delta}$ is (1) smooth, (2) equal to the identity map id when it is restricted to $\mathbb{B}^m(r)$, and (3) truncated to be supported in $\mathbb{B}^m(r+\delta)$ with a small positive number $\delta > 0$. Let $\nabla V_{r,\delta}$ be a smooth function that satisfies

$$V_{r,\delta}(\boldsymbol{x}) := \begin{cases} \frac{1}{2}|\boldsymbol{x}|^2 & \boldsymbol{x} \in \mathbb{B}^m(0; r), \\ \text{(smooth map)} & \boldsymbol{x} \in \mathbb{B}(0; r+\delta) \setminus \mathbb{B}(0; r), \\ 0 & \boldsymbol{x} \notin \mathbb{B}^m(0; r+\delta), \end{cases}$$

and let

$$\mathsf{id}_{r,\delta} := \nabla V_{r,\delta}.$$

Note that we can construct $\mathsf{id}_{r,\delta}$ and $\nabla V_{r,\delta}$ by using mollifiers; thus, such maps exist.

The ridgelet transform of the truncated autoencoder is given by

$$R_\rho[\mathsf{id}_{r,\delta}](\boldsymbol{a}, b) \approx -K \boldsymbol{a} \overline{\rho'(-b)} \quad \text{as} \quad \delta \to 0 \tag{66}$$

with a certain constant K (see Appendix E for the proof).

8. Discussion

We performed transport analysis of denoising autoencoders by introducing the flow representation. The flow representation $\boldsymbol{\varphi}_t$ is the depth continuum limit of a DNN, specified by an ODE with vector field $\boldsymbol{v}_t$. We interpreted an ordinary DNN $\boldsymbol{g}_t$ as a transport map or an Euler broken line approximation of $\boldsymbol{\varphi}_t$. The advantages of the flow representation are that it provides the coordinate-free treatment of DNNs, avoiding the redundancy of the ordinary parametrization of DNNs, and that it facilitates our understanding of what DNNs do—it is the mass transportation controlled by $\boldsymbol{v}_t$. In addition, the advantage of the interpretation as mass transportation is that it can handle function composition. In the transport analysis, we analyzed a flow in three aspects: a dynamical system described by a transport map or vector field, a pushforward measure described by a continuity equation, and Wasserstein gradient flow. From the results in Wasserstein geometry, these aspects are closely connected, and the hyperparameter $\boldsymbol{v}_t$ plays a central role as an intermediary. For example, in the transport analysis of continuous DAEs, the potential functional of the Wasserstein gradient flow often facilitates our understanding of the flow because it is the Shannon entropy, which is a fundamental quantity in statistics and machine learning.

In Section 3 and 4, we specified the transport maps of shallow, deep, and infinitely deep DAEs, and we gave their statistical interpretations. The shallow DAE is an estimator of the mean, while the deep DAE transports data points to decrease the Shannon entropy of the data distribution, which gives a partial answer to our research question "what do hidden layers do?" In Section 5, according to analytic and numerical experiments, we showed that deep DAEs converge faster and that the extracted features are different from each other, which gives a partial answer to the other question "why do DNNs perform better?" In Section 6, we proved the equivalence between the stacked DAE and the composition of DAEs. Because of the peculiar construction, it is difficult to formulate and understand stacking. Nevertheless, by tracking the flow, we succeeded in formulating the stacked DAE. In Section 7, we developed the double continuum limits, or the width continuum limit of the depth continuum limit. We presented some examples of the integral representation of the flow, such as encoder, decoder, and traditional autoencoder.

As a consequence of the equivalence, we can understand the so-called *pre-training* and *fine-tuning* strategy (Bengio et al., 2007; Erhan et al., 2010) as an *optimal control* problem. Namely, write a DNN as a composite $\boldsymbol{\psi} \circ \boldsymbol{\varphi}_t$ of classifier $\boldsymbol{\psi} : \mathbb{R}^m \to [0,1]^n$ and flow $\boldsymbol{\varphi}_t : \mathbb{R}^m \to \mathbb{R}^m$. If $\boldsymbol{\varphi}_t$ stays closer to the identity, $\boldsymbol{\psi}$ has to be more complex—and vice versa. The pre-training regularizes the behavior of hidden layers by

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\varphi}_t(\boldsymbol{x}) = \boldsymbol{v}_t(\boldsymbol{\varphi}_t(\boldsymbol{x})), \quad \boldsymbol{x} \in \mathbb{R}^m,\, t > 0 \tag{67}$$

and the fine-tuning specifies the relation between input and output by

$$\text{Minimize} \quad \mathbb{E}_{X,Y}|Y - \boldsymbol{\psi} \circ \boldsymbol{\varphi}_{t=1}(X)|^2 \quad \text{w.r.t NN } \boldsymbol{\psi} \circ \boldsymbol{\varphi}_{t=1}. \tag{68}$$

Overall, we can understand the strategy as the control problem of system (67) under restriction (68). Owing to ridgelet transform, shallow NNs are interpretable and principled. Development of a "solution operator" to the control problem in the flow representation would open the way to the interpretable and principled alternative to DNNs.

Acknowledgments

The authors thank the editor and reviewers for their supportive and insightful comments, which have improved the clarity of the argument significantly. The authors also acknowledge fruitful discussions with Dr. Shotaro Akaho, Dr. Kohei Yatabe, Dr. Keisuke Yano, and Mr. Kentaro Minami. This work was supported by JSPS KAKENHI (15J07517 and 18K18113).

Appendix A. Proof of Theorem 4

By $L^1_{\text{loc}}(\mathbb{R}^m)$ and $C_c^\infty(\mathbb{R}^m)$, we denote the spaces of locally integrable functions and compactly supported smooth functions, respectively. We assume that $\boldsymbol{g} : \mathbb{R}^m \to \mathbb{R}^m$ is locally integrable (L^1_{loc}).

Proof The proof follows from the calculus of variations. Let

$$\begin{aligned} L[\boldsymbol{g}] &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} |\boldsymbol{g}(\boldsymbol{x}+\boldsymbol{\varepsilon}) - \boldsymbol{x}|^2 \mu_0(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{g}(\boldsymbol{x}') - \boldsymbol{x}' + \boldsymbol{\varepsilon}|^2 \mu_0(\boldsymbol{x}' - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x}', \quad \boldsymbol{x}' \leftarrow \boldsymbol{x} + \boldsymbol{\varepsilon}. \end{aligned}$$

Here, $L[\boldsymbol{g}]$ always exists because $\boldsymbol{g} \in L^1_{\text{loc}}(\mathbb{R}^m) \subset L^2(\mu * \nu)$. Then, for an arbitrary function $\boldsymbol{h} \in C_c^\infty(\mathbb{R}^m)$, the first variation $\delta L[\boldsymbol{h}]$ is given by

$$\begin{aligned} \delta L[\boldsymbol{h}] &= \frac{d}{ds} L[\boldsymbol{g} + s\boldsymbol{h}] \Big|_{s=0} \\ &= \int_{\mathbb{R}^m} \frac{\partial}{\partial s} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{g}(\boldsymbol{x}) + s\boldsymbol{h}(\boldsymbol{x}) - \boldsymbol{x} + \boldsymbol{\varepsilon}|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} \Big|_{s=0} \\ &= 2 \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{x} + \boldsymbol{\varepsilon}) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \boldsymbol{h}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}. \end{aligned}$$

At a critical point $\boldsymbol{g}^*$ of L, $\delta L[\boldsymbol{h}] \equiv 0$ for every $\boldsymbol{h}$. Hence,

$$\mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{g}^*(\boldsymbol{x}) - \boldsymbol{x} + \boldsymbol{\varepsilon}) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] = 0, \quad \text{a.e. } \boldsymbol{x},$$

by the fundamental lemma of calculus of variations for integrable functions, and we have

$$\begin{aligned} \boldsymbol{g}^*(\boldsymbol{x}) &= \frac{\mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{x} - \boldsymbol{\varepsilon}) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]}{\mathbb{E}_{\boldsymbol{\varepsilon}} [\mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]} = (14) \\ &= \boldsymbol{x} - \frac{\mathbb{E}_{\boldsymbol{\varepsilon}} [\boldsymbol{\varepsilon} \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]}{\mathbb{E}_{\boldsymbol{\varepsilon}} [\mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})]} = (15). \end{aligned}$$

Note that $\boldsymbol{g}^*$ attains the global minimum, because, for every function $\boldsymbol{h}$,

$$\begin{aligned} L[\boldsymbol{g}^* + \boldsymbol{h}] &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{\varepsilon} - \mathbb{E}_t[\boldsymbol{\varepsilon}|\boldsymbol{x}] + \boldsymbol{h}(\boldsymbol{x})|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} \\ &= \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{\varepsilon} - \mathbb{E}_t[\boldsymbol{\varepsilon}|x]|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} + \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [|\boldsymbol{h}(\boldsymbol{x})|^2 \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \mathrm{d}\boldsymbol{x} \\ &\quad + 2 \int_{\mathbb{R}^m} \mathbb{E}_{\boldsymbol{\varepsilon}} [(\boldsymbol{\varepsilon} - \mathbb{E}_t[\boldsymbol{\varepsilon}|\boldsymbol{x}]) \mu_0(\boldsymbol{x} - \boldsymbol{\varepsilon})] \boldsymbol{h}(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} \\ &= L[\boldsymbol{g}^*] + L[\boldsymbol{h}] + 2 \cdot 0 \geq L[\boldsymbol{g}^*]. \end{aligned}$$

■

Appendix B. Proof of Fact 2

For simplicity, we assume that $\boldsymbol{g}, \boldsymbol{v}$, and μ are smooth. See Ambrosio et al. (2008, § 8.1) for more generalized conditions on the continuity equation.

Proof To facilitate visualization, we write $\boldsymbol{g}(\boldsymbol{x}, t), \boldsymbol{v}(\boldsymbol{x}, t)$, and $\mu(\boldsymbol{x}, t)$ instead of $\boldsymbol{g}_t(\boldsymbol{x}), \boldsymbol{v}_t(\boldsymbol{x})$, and $\mu_t(\boldsymbol{x})$, respectively.

By definition,

$$\begin{cases} \partial_t \boldsymbol{g}(\boldsymbol{g}(\boldsymbol{x}, t), t) = \boldsymbol{v}(\boldsymbol{g}(\boldsymbol{x}, t), t), & \boldsymbol{x} \in \mathbb{R}^m,\ t > 0 \\ \boldsymbol{g}(\boldsymbol{x}, 0) = 0, & \boldsymbol{x} \in \mathbb{R}^m. \end{cases}$$

In particular,

$$\nabla \boldsymbol{g}(\boldsymbol{x}, 0) = I.$$

According to the change-of-variables formula, for any $\boldsymbol{x} \in \mathbb{R}^m$ and $t > s > 0$,

$$\mu(\boldsymbol{g}(\boldsymbol{x}, t), t) \cdot |\nabla \boldsymbol{g}(\boldsymbol{x}, t)| = \mu(\boldsymbol{x}, s),$$

where $|\cdot|$ denotes the determinant.

Take the logarithm on both sides and then differentiate with respect to t. Then, the RHS vanishes and the LHS is calculated as follows:

$$\begin{aligned} \partial_t \log[\mu(\boldsymbol{g}(\boldsymbol{x}, t), t) \cdot |\nabla \boldsymbol{g}(\boldsymbol{x}, t)|] &= \frac{\partial_t[\mu(\boldsymbol{g}(\boldsymbol{x}, t), t)]}{\mu(\boldsymbol{g}(\boldsymbol{x}, t), t)} + \partial_t \log |\nabla \boldsymbol{g}(\boldsymbol{x}, t)| \\ &= \frac{(\nabla \mu)(\boldsymbol{g}(\boldsymbol{x}, t), t) \cdot \partial_t \boldsymbol{g}(\boldsymbol{x}, t) + (\partial_t \mu)(\boldsymbol{g}(\boldsymbol{x}, t), t)}{\mu(\boldsymbol{g}(\boldsymbol{x}, t), t)} \\ &\qquad + \mathsf{tr}\,[(\nabla \boldsymbol{g}(\boldsymbol{x}, t))^{-1} \nabla \partial_t \boldsymbol{g}(\boldsymbol{x}, t)], \end{aligned}$$

where the second term follows a differentiation formula by Petersen and Pedersen (2012, Eq. 43)

$$\partial \log |J| = \mathsf{tr}\,[J^{-1} \partial J].$$

By letting $t \to s + 0$,

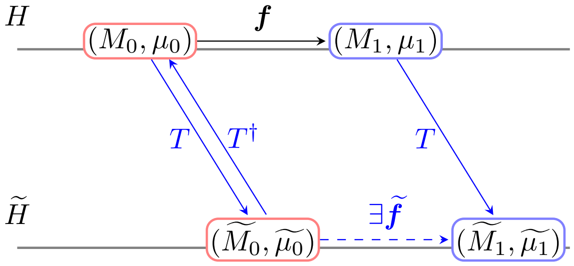
$$\frac{\nabla \mu(\boldsymbol{x}, t) \cdot \boldsymbol{v}(\boldsymbol{x}, t) + (\partial_t \mu)(\boldsymbol{x}, t)}{\mu(\boldsymbol{x}, t)} + \mathsf{tr}\,[\nabla \boldsymbol{v}(\boldsymbol{x}, t)] = 0,$$

which gives

$$\partial_t \mu(\boldsymbol{x}, t) = -\nabla \cdot [\mu(\boldsymbol{x}, t) \boldsymbol{v}(\boldsymbol{x}, t)]. \qquad \blacksquare$$

Appendix C. Proof of Theorem 9

We show that the diagram commutes. Observe that $\boldsymbol{f} = \mathsf{id} + tD\nabla \log e^{tL_t}\mu$ is the sum of the present position id and the gradient ∇V of potential $V = \log e^{tL_t}\mu$. We calculate the pushforward $\widetilde{\nabla}\widetilde{V}$ and show that it coincides with $\widetilde{L}_t$-DAE.


H
(M_0, μ_0)
$\boldsymbol{f}$
(M_1, μ_1)
T
$T^\dagger$
T
$\widetilde{H}$
$(\widetilde{M_0}, \widetilde{\mu_0})$
$\exists \widetilde{\boldsymbol{f}}$
$(\widetilde{M_1}, \widetilde{\mu_1})$

Proof We suppose that L_t is expressed as

$$L_t u := \boldsymbol{a}_t^\top (\nabla^2 u) \boldsymbol{a}_t + \boldsymbol{b}_t^\top \nabla u + c_t u, \quad u \in C^2(H) \tag{69}$$

and T is expressed as

$$T(\boldsymbol{z}) = A\boldsymbol{z} \tag{70}$$

with a matrix A.

By the assumption that the restriction $T|_{M_0}$ is injective, it has a left inverse $T^\dagger$ such that $T^\dagger \circ T|_{M_0} = \mathsf{id}_{M_0}$. Note that it is not a linear map but an abstract nonlinear map, which means that there is no matrix A that realizes $T^\dagger$.

Step. 1

We show that

$$T \circ \boldsymbol{f} \circ T^\dagger = \mathsf{id} + t\widetilde{D}\widetilde{\nabla}\widetilde{V} \quad \text{in } \widetilde{M_0} \tag{71}$$

where $\widetilde{D} = ADA^\top$ and $\widetilde{V} = V \circ T^\dagger$.

For an arbitrary $U \in C^2(M_0)$, write $T_* U := U \circ T^\dagger \in C^2(\widetilde{M_0})$, and

$$\nabla U(T^\dagger(\boldsymbol{x})) = A^\top \widetilde{\nabla} T_* U(\boldsymbol{x}), \quad \boldsymbol{x} \in \widetilde{M_0} \tag{72}$$

because the i-th element of $\widetilde{\nabla} T_* U$ is given by

$$\frac{\partial U \circ T^\dagger}{\partial x_i}(\boldsymbol{x}) = \sum_p \frac{\partial U}{\partial z_p}(T^\dagger(\boldsymbol{x})) \frac{\partial T_p^\dagger}{\partial x_i}(\boldsymbol{x}).$$

Thus, the q-th element of $A^\top \widetilde{\nabla} T_* U$ is given by

$$\begin{aligned}
\sum_i A_{iq} \frac{\partial U \circ T^\dagger}{\partial x_i}(\boldsymbol{x}) &= \sum_p \frac{\partial U}{\partial z_p}(T^\dagger(\boldsymbol{x})) \sum_i A_{iq} \frac{\partial T_p^\dagger}{\partial x_i}(\boldsymbol{x}) \\
&= \sum_p \frac{\partial U}{\partial z_p}(T^\dagger(\boldsymbol{x})) \delta_{pq} \\
&= \frac{\partial U}{\partial z_q}(T^\dagger(\boldsymbol{x})).
\end{aligned}$$

Therefore, by substituting U with $V = \log e^{tL_t}\mu_0$,

$$\begin{aligned} T \circ \boldsymbol{f} \circ T^\dagger(\boldsymbol{x}) &= \boldsymbol{x} + A(tD\nabla V(T^\dagger(\boldsymbol{x}))) \\ &= \boldsymbol{x} + tADA^\top \widetilde{\nabla} T_* V(\boldsymbol{x})) \\ &= \boldsymbol{x} + t\widetilde{D}\widetilde{\nabla}\widetilde{V}(\boldsymbol{x}). \end{aligned}$$

Step. 2

We show that

$$\widetilde{V} = \log e^{t\widetilde{L}_t}\widetilde{\mu}_0 + (const.), \quad \text{in } M_0 \tag{73}$$

where

$$\widetilde{L}_t\widetilde{u} := \widetilde{\boldsymbol{a}}_t^\top(\widetilde{\nabla}^2\widetilde{u})\widetilde{\boldsymbol{a}}_t + \widetilde{\boldsymbol{b}}_t^\top\widetilde{\nabla}\widetilde{u} + \widetilde{c}_t\widetilde{u}, \quad \widetilde{u} \in C^2(\widetilde{H}) \tag{74}$$

with $\widetilde{\boldsymbol{a}}_t = A\boldsymbol{a}_t \circ T^\dagger, \widetilde{\boldsymbol{b}}_t = A\boldsymbol{b}_t \circ T^\dagger$, and $\widetilde{c}_t = c_t \circ T^\dagger$.

Let

$$u_t := e^{tL_t}\mu_0. \tag{75}$$

By the definition of semigroup e^{tL_t}, $u_0 = \mu_0$ and $\partial_t u_t = L_t u_t$ (however, u_1 is different from μ_1).

Given u_t, let

$$\widetilde{u}_t := T_\sharp u_t. \tag{76}$$

According to the change-of-variables formula (7),

$$\widetilde{u}_t = [A]^{-1} T_* u_t, \tag{77}$$

where $[A] := \sqrt{\det|A^\top A|}$ and $T_* u_t := u_t \circ T^\dagger$. In particular, $\widetilde{u}_0 = \widetilde{\mu}_0$ and $\log\widetilde{u}_t = \widetilde{V}$.

Furthermore,

$$\partial_t\widetilde{u}_t = \widetilde{L}_t\widetilde{u}_t, \quad \text{in } \widetilde{M_0}, \tag{78}$$

because

$$\begin{aligned} \partial_t\widetilde{u}_t(\boldsymbol{x}) &= [A]^{-1}\partial_t[u_t(T^\dagger(\boldsymbol{x}))] \\ &= [A]^{-1}L_t[u_t](T^\dagger(\boldsymbol{x})), \end{aligned}$$

and

$$\begin{aligned} &[A]^{-1}\boldsymbol{a}_t(T^\dagger(\boldsymbol{x}))^\top(\nabla^2 u_t(T^\dagger(\boldsymbol{x})))\boldsymbol{a}_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \boldsymbol{a}_t(T^\dagger(\boldsymbol{x}))^\top(A^\top\widetilde{\nabla}^2[[A]^{-1}T_* u_t](\boldsymbol{x})A)\boldsymbol{a}_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \widetilde{\boldsymbol{a}}_t(\boldsymbol{x})^\top(\widetilde{\nabla}^2[\widetilde{u}_t](\boldsymbol{x}))\widetilde{\boldsymbol{a}}_t(\boldsymbol{x}), \\ &[A]^{-1}\boldsymbol{b}_t(T^\dagger(\boldsymbol{x}))^\top\nabla u_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \boldsymbol{b}_t(T^\dagger(\boldsymbol{x}))^\top A^\top\widetilde{\nabla}[[A]^{-1}T_* u_t](\boldsymbol{x}) \\ &\quad = \widetilde{\boldsymbol{b}}_t(\boldsymbol{x})^\top\widetilde{\nabla}\widetilde{u}_t(\boldsymbol{x}), \\ &[A]^{-1}c_t(T^\dagger(\boldsymbol{x}))u_t(T^\dagger(\boldsymbol{x})) \\ &\quad = \widetilde{c}_t(\boldsymbol{x})\widetilde{u}_t(\boldsymbol{x}). \end{aligned}$$

Thus,

$$\partial_t \widetilde{u}_t(\boldsymbol{x}) = [A]^{-1} L_t[u_t](T^\dagger(\boldsymbol{x})) = \widetilde{L}_t \widetilde{u}_t(\boldsymbol{x}).$$

Hence, $\widetilde{u}_t$ is the solution of the initial value problem $\partial_t \widetilde{u}_t = \widetilde{L}_t \widetilde{u}_t$ with $\widetilde{u}_0 = \widetilde{\mu}_0$. By the uniqueness of the solution, $\widetilde{u}_t = e^{t\widetilde{L}_t}\widetilde{\mu}_0$. On the other hand, $\log \widetilde{u}_t = \widetilde{V}$. Therefore, $\widetilde{V} = \log \widetilde{u}_t = e^{t\widetilde{L}_t}\widetilde{\mu}_0$.

To sum up the two steps,

$$T \circ \boldsymbol{f} \circ T^\dagger = \mathsf{id} + t\widetilde{D}\widetilde{\nabla} \log e^{t\widetilde{L}_t}\widetilde{\mu}_0 =: \widetilde{\boldsymbol{f}},$$

and we have the topological conjugacy

$$T \circ \boldsymbol{f} = \widetilde{\boldsymbol{f}} \circ T.$$

■

Appendix D. Proofs for Analytic Examples

D.1. Univariate Normal Distribution

We calculate the case for a univariate normal distribution $N(m_0, \sigma_0^2)$.

D.1.1. Shallow DAE

We show that

$$g_t(x) = \frac{\sigma_0^2}{\sigma_0^2 + t} x + \frac{t}{\sigma_0^2 + t} m_0, \tag{30}$$

$$\mu_t = N\left(m_0, \frac{\sigma_0^2}{(1 + t/\sigma_0^2)^2}\right). \tag{31}$$

Proof The proof is immediate from (17). First, write $\phi_t(x, y) = (4\pi t)^{-1/2} \exp(-|x-y|^2/4t)$,

$$\phi_{t/2} * N(m_0, \sigma_0^2) = N(m_0, \sigma_0^2 + t).$$

Hence,

$$g_t(x) = x + t\nabla \log[N(m_0, \sigma_0^2 + t)] = \frac{\sigma_0^2}{\sigma_0^2 + t} x + \frac{t}{\sigma_0^2 + t} m_0.$$

As g_t is affine, the pushforward is immediate. ■

D.1.2. Continuous DAE

We show that

$$g_t(x) = \sqrt{1 - 2t/\sigma_0^2}(x - m_0) + m_0, \tag{32}$$

$$\mu_t = N(m_0, \sigma_0^2 - 2t), \quad 0 \leq t < \sigma_0^2/2. \tag{33}$$

Proof $[\mu_t]$ Write the pushforward as $N(m_t, \sigma_t^2)$. By using the heat kernel $\phi_t(\boldsymbol{x}, \boldsymbol{y}) = (4\pi t)^{-m/2} \exp(-|\boldsymbol{x} - \boldsymbol{y}|^2/4t)$, for some $T > 0$,

$$\begin{aligned} N(m_t, \sigma_t^2) &= \phi_{T-t} * N(m_T, \sigma_T^2) \\ &= N(m_T, \sigma_T^2 + 2(T-t)). \end{aligned}$$

By eliminating T by the initial conditions, we have

$$N(m_t, \sigma_t^2) = N(m_0, \sigma_0^2 - 2t).$$

By the positivity of σ_t^2, we can determine the largest possible T as $T = \sigma_0^2/2$. ■

Proof $[g_t]$ Fix an arbitrary point x_0. Write $x_t := g_t(x_0)$ and $\dot{x}_t := \partial_t g_t(x_0)$. Recall that $\dot{m}_t \equiv 0$, because m_t is a constant. According to (24),

$$\dot{x}_t = -\frac{x_t - m_t}{\sigma_t^2}.$$

By dividing both sides by x_t and integrating them,

$$\begin{aligned} \log \left| \frac{x_t - m_t}{x_0 - m_0} \right| &= -\int_0^t \frac{\mathrm{d}s}{\sigma_s^2} \\ &= \frac{1}{2} \int_0^t \frac{\mathrm{d}s}{s - T} \\ &= \frac{1}{2} \log \left| \frac{T-t}{T} \right|, \end{aligned}$$

which concludes the proof. ■

D.2. Multivariate Normal Distribution

We calculate the case for a multivariate normal distribution $N(\boldsymbol{m}_0, \Sigma_0)$.

D.2.1. Shallow DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) = (I + t\Sigma_0^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_0)^{-1}\boldsymbol{m}_0, \tag{34}$$

$$\mu_t = N(\boldsymbol{m}_0, \Sigma_0(I + t\Sigma_0^{-1})^{-2}). \tag{35}$$

Proof Calculate (17) directly as in the univariate case. First, by writing $\phi_t(\boldsymbol{x}, \boldsymbol{y}) = (4\pi t)^{-m/2} \exp(-|\boldsymbol{x} - \boldsymbol{y}|^2/4t)$,

$$\phi_{t/2} * N(\boldsymbol{m}_0, \Sigma_0) = N(\boldsymbol{m}_0, \Sigma_0 + tI).$$

Hence,

$$\begin{aligned} \boldsymbol{g}_t(\boldsymbol{x}) &= \boldsymbol{x} + t\nabla \log[N(\boldsymbol{m}_0, \Sigma_0 + tI)] \\ &= \boldsymbol{x} + t\nabla \left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_0)^\top (\Sigma_0 + tI)^{-1}(\boldsymbol{x} - \boldsymbol{m}_0) \right] \\ &= (I + t\Sigma_0^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_0)^{-1}\boldsymbol{m}_0. \end{aligned}$$

As $\boldsymbol{g}_t$ is affine, the pushforward is immediate. ■

D.2.2. Continuous DAE

We show that

$$\begin{aligned} \boldsymbol{g}_t(\boldsymbol{x}) &= \sqrt{I - 2t\Sigma_0^{-1}}(\boldsymbol{x} - \boldsymbol{m}_0) + \boldsymbol{m}_0, && (36) \\ \mu_t &= N(\boldsymbol{m}_0, \Sigma_0 - 2tI). && (37) \end{aligned}$$

Proof Write $\phi_t(\boldsymbol{x}, \boldsymbol{y}) = (4\pi t)^{-m/2} \exp(-|\boldsymbol{x} - \boldsymbol{y}|^2/4t)$, and recall that $\phi_t * N(\boldsymbol{m}, \Sigma) = N(\boldsymbol{m}, \Sigma + 2tI)$. Thus, the pushforward $N(\boldsymbol{m}_t, \Sigma_t)$ is obtained as follows in a manner similar to the univariate case.

$$N(\boldsymbol{m}_t, \Sigma_t) = N\left(\boldsymbol{m}_0, \Sigma_0 - 2tI\right).$$

Suppose that $\boldsymbol{g}_t(\boldsymbol{x})$ is an affine transform $A_t(\boldsymbol{x} - \boldsymbol{m}_0) + \boldsymbol{m}_0$ analogous to the univariate case. Recall that, if $X \sim N(\boldsymbol{m}, \Sigma)$, then $AX + b \sim N(A\boldsymbol{m} + b, A\Sigma A^\top)$. Hence, for our case, $\Sigma_t = A_t\Sigma_0 A_t^\top$ and we can determine

$$A_t = \sqrt{\Sigma_t\Sigma_0^{-1}} = \sqrt{I - 2t\Sigma_0^{-1}}.$$

Finally, we check whether $\boldsymbol{g}_t$ satisfies (24). As Σ_0 is symmetric, we can always diagonalize $\Sigma_0 = UD_0U^\top$ with an orthogonal matrix U and a diagonal matrix D_0. Observe that with the same U, we can simultaneously diagonalize Σ_t and A_t as

$$\begin{aligned} \Sigma_t &= UD_tU^\top, \quad D_t := D_0 - 2tI \\ A_t &= UD_t^{1/2}D_0^{-1/2}U^\top. \end{aligned}$$

Without loss of generality, we can assume that $U = I$; therefore, Σ_t and A_t are diagonal and $\boldsymbol{m}_t \equiv 0$. Fix an index j and denote the j-th diagonal element of Σ_t and A_t by σ_t^2 and a_t, respectively. Then, our goal is reduced to showing that $\partial_t[a_t x] = \nabla \log \mu_t(a_t x)$ for every fixed $x \in \mathbb{R}$.

By definition,

$$\begin{aligned} \sigma_t^2 &= \sigma_0^2 - 2t, \\ a_t &= \sigma_t\sigma_0^{-1} = \sqrt{1 - 2t\sigma_0^{-2}}. \end{aligned}$$

Thus, the LHS is

$$\partial_t[a_t x] = -\frac{1}{\sigma_0\sqrt{\sigma_0^2 - 2t}} x = -\sigma_0^{-1}\sigma_t^{-1} x,$$

and the RHS is

$$\nabla \log \mu_t(a_t x) = -\frac{a_t x}{\sigma_t^2} = -\sigma_0^{-1}\sigma_t^{-1} x.$$

Hence, the LHS equals the RHS. ■

D.3. Mixture of Multivariate Normal Distributions

We calculate the case for the mixture of multivariate normal distributions $\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k)$, with the assumption that it is *well separated* (see Section 5.1.3 for the definition).

D.3.1. Shallow DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sum_{k=1}^K \gamma_{kt}(\boldsymbol{x}) \left\{ (I + t\Sigma_k^{-1})^{-1}\boldsymbol{x} + (I + t^{-1}\Sigma_k)^{-1}\boldsymbol{m}_k \right\}, \tag{38}$$

$$\mu_t \approx \sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k(I + t\Sigma_k^{-1})^{-2}), \quad \text{if well separated} \tag{39}$$

with the responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k + tI)}. \tag{40}$$

Proof Directly calculate (17). By the linearity of the heat kernel,

$$\begin{aligned}
\boldsymbol{g}_t &:= \mathsf{id} + t\sum_{k=1}^K \frac{w_k \nabla N(\boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k + tI)}, \\
&= \mathsf{id} + \sum_{k=1}^K \frac{w_k N(\boldsymbol{m}_k, \Sigma_k + tI)}{\sum_{k=1}^K w_k N(\boldsymbol{m}_k, \Sigma_k + tI)} \cdot t\nabla \log N(\boldsymbol{m}_k, \Sigma_k + tI), \\
&= \mathsf{id} + \sum_{k=1}^K \gamma_{kt}(\boldsymbol{g}_{kt} - \mathsf{id}), \\
&= \sum_{k=1}^K \gamma_{kt}\boldsymbol{g}_{kt},
\end{aligned}$$

where $\boldsymbol{g}_{kt}$ exactly coincides with the flow induced by the individual k-th component.

To calculate the pushforward, we introduce some auxiliary variables. Write $w(k) := w_k,\ \gamma(k \mid \cdot) := \gamma_{kt}(\cdot)$ and

$$\begin{aligned} \mu_t(\cdot \mid k) &:= N(\boldsymbol{m}_k, \Sigma_k + tI), \\ \mu_t &:= \sum_k w(k)\mu_t(\cdot \mid k). \end{aligned}$$

Let $\tau_k(\cdot \mid \boldsymbol{x})$ be a probability measure that satisfies

$$\int_M \tau_k(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x} \mid k)\mathrm{d}\boldsymbol{x} = \mu_t(y \mid k).$$

Note that τ_k is not unique. Recall that by definition, if $X \sim \mu_0(\cdot \mid k)$, then $Y = \boldsymbol{g}_{kt}(X) \sim \mu_t(\cdot \mid k)$. Hence, τ_k is a stochastic alternative to $\boldsymbol{g}_{kt}$.

Consider a probability measure

$$\sigma(\cdot \mid \boldsymbol{x}) := \sum_{k=1}^K \gamma(k \mid \boldsymbol{x})\tau_k(\cdot \mid \boldsymbol{x}).$$

Clearly, this is a stochastic alternative to $\boldsymbol{g}_t$. We show that

$$\int_M \sigma(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \approx \mu_t(y).$$

The LHS is reduced to

$$\begin{aligned} \int_M \sigma(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x} &= \int_M \sum_{k=1}^K \gamma(k \mid \boldsymbol{x})\tau_k(y \mid \boldsymbol{x}) \sum_\ell w(\ell)\mu_0(\boldsymbol{x} \mid \ell)\mathrm{d}\boldsymbol{x} \\ &= \sum_\ell w(\ell) \sum_{k=1}^K \int_M \gamma(k \mid \boldsymbol{x})\tau_k(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x} \mid \ell)\mathrm{d}\boldsymbol{x}. \qquad (79) \end{aligned}$$

Suppose that $\gamma(k \mid \boldsymbol{x})$ is an indicator function of a domain Ω_k, where $\int_{\Omega_k} \mu_0(\cdot \mid k) \approx 1$. Then,

$$\begin{aligned} (79) &\approx \sum_\ell w(\ell) \int_{\Omega_\ell} \tau_k(y \mid \boldsymbol{x})\mu_0(\boldsymbol{x} \mid \ell)\mathrm{d}\boldsymbol{x} \\ &\approx \sum_\ell w(\ell)\mu_t(y \mid \ell) = \mu_t(y). \end{aligned}$$

This concludes the claim. ■

D.3.2. Continuous DAE

We show that

$$\boldsymbol{g}_t(\boldsymbol{x}) \approx \sqrt{I - 2t\Sigma_k^{-1}}\,(\boldsymbol{x} - \boldsymbol{m}_k) + \boldsymbol{m}_k, \quad \boldsymbol{x} \in \Omega_k, \text{ if well separated} \qquad (41)$$

$$\mu_t = \sum_{k=1}^K w_k N\left(\boldsymbol{m}_k, \Sigma_k - 2tI\right), \qquad (42)$$

with the responsibility function

$$\gamma_{kt}(\boldsymbol{x}) := \frac{w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}{\sum_{k=1}^{K} w_k N(\boldsymbol{x}; \boldsymbol{m}_k, \Sigma_k - 2tI)}. \tag{43}$$

Proof The pushforward is immediate by the linearity of the heat kernel. The dynamical system (24) for our case is reduced to

$$\partial_t \boldsymbol{g}_t(\boldsymbol{x}) = -\sum_{k=1}^{K} \gamma_{kt} \circ \boldsymbol{g}_t(\boldsymbol{x})(\Sigma_k - 2tI)^{-1}(\boldsymbol{g}_t(\boldsymbol{x}) - \boldsymbol{m}_k).$$

By the assumption that μ_0 is well separated, we can take an open neighborhood Ω_k of $\boldsymbol{m}_k$ and an open time interval I that contains t such that $\gamma_{kt} \circ \boldsymbol{g}_t(\boldsymbol{x}) \equiv 1$ for every $(\boldsymbol{x}, t) \in \Omega_k \times I$. In this restricted domain, the dynamical system is reduced to a single-component version:

$$\partial_t \boldsymbol{g}_t(\boldsymbol{x}) = -(\Sigma_k - 2tI)^{-1}(\boldsymbol{g}_t(\boldsymbol{x}) - \boldsymbol{m}_k), \quad (\boldsymbol{x}, t) \in \Omega_k \times I.$$

According to the previous results, we have exactly

$$\boldsymbol{g}_t(\boldsymbol{x}) = \sqrt{I - 2t\Sigma_k^{-1}}(\boldsymbol{x} - \boldsymbol{m}_k) + \boldsymbol{m}_k, \quad (\boldsymbol{x}, t) \in \Omega_k \times I. \qquad \blacksquare$$

Appendix E. Proof of (66)

Let $\delta \to 0$. Then, the ridgelet transform of the truncated autoencoder $\mathsf{id}_{r,\delta}$ is given by

$$\begin{aligned} R_\rho[\mathsf{id}_{r,0}](\boldsymbol{a}, b) &= -\frac{A_{m-1}}{2(m+1)} \int_{|p|<r} (r^2 - p^2)^{\frac{m-1}{2}} \left\{ \frac{2}{m-1} p^2 + r^2 \right\} \overline{\rho'(|\boldsymbol{a}|p - b)} \boldsymbol{a} \mathrm{d}p \qquad (80) \\ &\approx -K\boldsymbol{a}\overline{\rho'(-b)}, \qquad (81) \end{aligned}$$

where $A_{m-1} := \frac{2\pi^{\frac{m-1}{2}}}{\Gamma(\frac{m-1}{2})}$ is the surface area of $\mathbb{S}^{m-1}$, and K is given by (87).

Proof Let $\delta \to 0$. Then, the connecting annulus $\mathbb{B}(0; r+\delta) \setminus \mathbb{B}(0; r)$ vanishes as follows:

$$\begin{aligned} R_\rho[\mathsf{id}_{r,\delta}](\boldsymbol{a}, b) &= -\boldsymbol{a} R_{\rho'}[V_{r,\delta}](\boldsymbol{a}, b) \\ &\to -\boldsymbol{a} \int_{\mathbb{B}^m(r)} \frac{1}{2} |\boldsymbol{x}|^2 \overline{\rho'(\boldsymbol{a} \cdot \boldsymbol{x} - b)} \mathrm{d}\boldsymbol{x} \\ &= -\boldsymbol{a} R_{\rho'}[V_{r,0}](\boldsymbol{a}, b). \end{aligned}$$

Hence, we omit considering the annulus.

In the following, we use a spherical coordinate defined by

$$\boldsymbol{u} := \boldsymbol{a}/|\boldsymbol{a}|, \quad \alpha := 1/|\boldsymbol{a}|, \quad \beta := b/|\boldsymbol{a}|,$$

where $\boldsymbol{u} \in \mathbb{S}^{m-1}$ denotes the direction, $\alpha \in \mathbb{R}_+$ denotes the scale, and $\beta \in \mathbb{R}$ denotes the (scaled) shift parameters.

The ridgelet transform in the spherical coordinate (Sonoda and Murata, 2017a) is given by

$$R_\rho f(\boldsymbol{u}/\alpha, \beta/\alpha) = \int_{\mathbb{R}} \mathrm{Rad}[f](\boldsymbol{u}, p)\overline{\rho_\alpha(p-\beta)}\mathrm{d}p,$$

where $\mathrm{Rad}[f](\boldsymbol{u}, p)$ denotes the Radon transform

$$\mathrm{Rad}[f](\boldsymbol{u}, p) := \int_{(\mathbb{R}\boldsymbol{u})^\perp} f(p\boldsymbol{u} + \boldsymbol{y})\mathrm{d}\boldsymbol{y}$$

of the function $f \in L^1(\mathbb{R}^m)$ at direction $\boldsymbol{u} \in \mathbb{S}^{m-1}$ and position $p \in \mathbb{R}$, and

$$\rho_\alpha(p) := \rho(p/\alpha).$$

The Radon transform $\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p)$ for $|p| < r$ is calculated as follows. Because $V_{r,\delta}$ is a radial function, $\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p)$ does not depend on the direction $\boldsymbol{u}$. Hence, it is sufficient to consider a special case when $(\mathbb{R}\boldsymbol{u})^\perp = \mathbb{R}^{m-1}$. Therefore,

$$\begin{aligned}
\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p) &= \int_{\mathbb{R}^{m-1}} V_{r,0}(p\boldsymbol{u} + y)\mathrm{d}\boldsymbol{y}, \quad \boldsymbol{u} \perp \boldsymbol{y} \\
&= \int_{\mathbb{R}^{m-1}} \frac{1}{2}|p\boldsymbol{u} + \boldsymbol{y}|^2 \mathbf{1}_{\mathbb{B}^m(0;r)}(p\boldsymbol{u} + \boldsymbol{y})\mathrm{d}\boldsymbol{y} \\
&= \frac{1}{2}\int_{\mathbb{B}^{m-1}\left(0;\sqrt{r^2-p^2}\right)} \left\{p^2 + |\boldsymbol{y}|^2\right\} \mathrm{d}\boldsymbol{y}, \qquad (82)
\end{aligned}$$

where the third equation follows by the orthogonality $|p\boldsymbol{u} + \boldsymbol{y}|_m^2 = p^2 + |\boldsymbol{y}|_{m-1}^2$ and a geometric consideration as follows:

$$\begin{aligned}
\int_{\mathbb{R}^{m-1}} [\,\cdot\,]\mathbf{1}_{\mathbb{B}^m(0;r)}(p\boldsymbol{u} + \boldsymbol{y})\mathrm{d}\boldsymbol{y} &= \int_{\mathbb{R}^{m-1}} [\,\cdot\,]\mathbf{1}_{\mathbb{B}^m(-p\boldsymbol{u};r)}(\boldsymbol{y})\mathrm{d}\boldsymbol{y} \\
&= \int_{\mathbb{R}^{m-1}\cap\mathbb{B}^m(-p\boldsymbol{u};r)} [\,\cdot\,]\mathrm{d}\boldsymbol{y} \\
&= \int_{\mathbb{B}^{m-1}(0;\sqrt{r^2-p^2})} [\,\cdot\,]\mathrm{d}\boldsymbol{y}.
\end{aligned}$$

The first integral in (82) is calculated as follows:

$$\begin{aligned}
\int_{\mathbb{B}^{m-1}\left(0;\sqrt{r^2-p^2}\right)} p^2\mathrm{d}\boldsymbol{y} &= p^2 \,\mathsf{vol}\left[\mathbb{B}^{m-1}(0;\sqrt{r^2-p^2})\right] \\
&= \frac{\pi^{\frac{m-1}{2}}}{2\Gamma\left(\frac{m-1}{2}+1\right)} p^2(r^2-p^2)^{\frac{m-1}{2}}. \qquad (83)
\end{aligned}$$

The second integral in (82) is calculated as follows:

$$\begin{aligned}\int_{\mathbb{B}^{m-1}\left(0;\sqrt{r^2-p^2}\right)} |\boldsymbol{y}|^2 \mathrm{d}\boldsymbol{y} &= \int_{\mathbb{S}^{m-2}} \int_0^{\sqrt{r^2-p^2}} |\rho\omega|^2 \rho^{m-2} \mathrm{d}\rho \mathrm{d}\omega \\ &= \int_{\mathbb{S}^{m-2}} \mathrm{d}\omega \int_0^{\sqrt{r^2-p^2}} \rho^m \mathrm{d}\rho \\ &= \frac{\pi^{\frac{m-1}{2}}}{(m+1)\Gamma\left(\frac{m-1}{2}\right)} (r^2 - p^2)^{\frac{m+1}{2}}. \end{aligned} \tag{84}$$

Hence, by combining the first and second integrals, we have

$$\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p) = \begin{cases} \frac{A_{m-1}}{2(m+1)} (r^2 - p^2)^{\frac{m-1}{2}} \left\{ \frac{2}{m-1} p^2 + r^2 \right\} & |p| < r \\ 0 & |p| \geq r. \end{cases} \tag{85}$$

The ridgelet transform $R_{\rho'}[V_{r,0}]$ is given by

$$R_{\rho'}[V_{r,0}](\boldsymbol{u}/\alpha, \beta/\alpha) = \int_{|p|<r} k(p) \overline{\rho'_\alpha(p - \beta)} \mathrm{d}p, \tag{86}$$

where we define

$$k(p) := \mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p).$$

Recall that $\mathrm{Rad}[V_{r,0}](\boldsymbol{u}, p)$ does not depend on the direction $\boldsymbol{u}$; thus, the definition of k is reasonable. According to (85), k is a compactly supported bump function. Consequently, k is summable; thus, the integral

$$K := \int_{\mathbb{R}} k(p) \mathrm{d}p \tag{87}$$

always exists. Recall that the convolution results in smoothing, i.e.,

$$\int_{|p|<r} k(p) \overline{\rho'_\alpha(p - \beta)} \mathrm{d}p \approx K \overline{\rho'_\alpha(-\beta)}. \tag{88}$$

In summary, we have presented the following:

$$R_\rho[\mathsf{id}_{r,0}](\boldsymbol{a}, b) = -\boldsymbol{a} R_{\rho'}[V_{r,0}](\boldsymbol{a}, b) \approx -K \boldsymbol{a} \overline{\rho'(-b)}. \qquad \blacksquare$$

References

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data generating distribution. *Journal of Machine Learning Research*, 15(Nov):3743–3773, 2014.

Guillaume Alain, Yoshua Bengio, Li Yao, Jason Yosinski, Eric Thibodeau-Laufer, Saizheng Zhang, and Pascal Vincent. GSNs: Generative stochastic networks. *Information and Inference*, 5(2):210–249, 2016.

Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient Flows in Metric Spaces and in the Space of Probability Measures.* Birkhäuser, 2008.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *Proceedings of The 34th International Conference on Machine Learning*, volume 70, pages 214–223, Sydney, Australia, 2017.

Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *The 36th International Conference on Machine Learning*, volume 80, pages 244–253, Stockholm, Sweden, 2018.

Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems 27*, pages 2654–2662, Montréal, BC, 2014.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18(19):1–53, 2017a.

Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18(21):1–38, 2017b.

Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Yoshua Bengio, Nicolas Le Roux, Pascal Vincent, Olivier Delalleau, and Patrice Marcotte. Convex neural networks. In *Advances in Neural Information Processing Systems 18*, pages 123–130, Vancouver, BC, 2006.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19*, pages 153–160, Vancouver, BC, 2007.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013a.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising autoencoders as generative models. In *Advances in Neural Information Processing Systems 26*, pages 899–907, Lake Tahoe, USA, 2013b.

Yoshua Bengio, Éric Thibodeau-Laufer, Guillaume Alain, and Jason Yosinski. Deep generative stochastic networks trainable by backprop. In *Proceedings of The 31st International Conference on Machine Learning*, volume 32, pages 226–234, Beijing, China, 2014.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

Yann Brenier. Polar factorization and monotone rearrangement of vector-valued functions. *Communications on Pure and Applied Mathematics*, 44(4):375–417, 1991.

Emmanuel Jean Candès. *Ridgelets: Theory and Applications.* PhD thesis, Standford University, 1998.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems 31*, pages 6572—-6583, Montréal, BC, 2018.

Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in Neural Information Processing Systems 32*, Montréal, BC, 2018.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Proceedings of The 18th International Conference on Artificial Intelligence and Statistics 2015*, volume 38, pages 192–204, San Diego, USA, 2015.

Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. In *29th Annual Conference on Learning Theory*, volume 49, pages 1–31, 2016.

Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems 27*, pages 2933–2941, Montréal, BC, 2014.

Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. In *29th Annual Conference on Learning Theory*, volume 49, pages 1–34, 2016.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

Lawrence C. Evans and Ronald F. Gariepy. *Measure Theory and Fine Properties of Functions*. CRC Press, revised edition, 2015.

Edward I. George, Feng Liang, and Xinyi Xu. Improved minimax predictive densities under Kullback-Leibler loss. *Annals of Statistics*, 34(1):78–91, 2006.

Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B. Grosse. The reversible residual network: Backpropagation without storing activations. In *Advances in Neural Information Processing Systems 30*, pages 2214–2224, Long Beach, USA, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680, Montréal, BC, 2014.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):1–22, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, Las Vegas, USA, 2016.

Geoffrey E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40(1-3): 185–234, 1989.

Masaaki Imaizumi and Kenji Fukumizu. Deep neural networks learn non-smooth functions effectively. In *Proceedings of The 22nd International Conference on Artificial Intelligence and Statistics 2019*, Okinawa, Japan, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37, pages 448–456, Lille, France, 2015.

Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems 29*, pages 586–594, Barcelona, Spain, 2016.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations 2014*, pages 1–14, Banff, BC, 2014.

Jason M. Klusowski and Andrew R. Barron. Minimax lower bounds for ridge combinations including neural nets. In *2017 IEEE International Symposium on Information Theory*, pages 1376–1380, 2017.

Jason M. Klusowski and Andrew R. Barron. Approximation by combinations of ReLU and squared ReLU ridge functions with ℓ^1 and ℓ^0 controls. *IEEE Transactions on Information Theory*, 64(12):7649–7656, 2018.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105, Lake Tahoe, USA, 2012.

Věra Kůrková. Complexity estimates based on integral transforms induced by computational units. *Neural Networks*, 33:160–167, 2012.

Honglak Lee. *Unsupervised Feature Learning via Sparse Hierarchical Representations*. PhD thesis, Stanford University, 2010.

Qianxiao Li and Shuji Hao. An optimal control approach to deep learning and applications to discrete-weight neural networks. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 2985–2994, Stockholm, Sweden, 2018.

Henry W. Lin, Max Tegmark, and David Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in Neural Information Processing Systems 29*, pages 2378–2386, Barcelona, Spain, 2016.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 3276–3285, Stockholm, Sweden, 2018.

Noboru Murata. An integral representation of functions using three-layered betworks and their approximation bounds. *Neural Networks*, 9(6):947–956, 1996.

Behnam Neyshabur. *Implicit Regularization in Deep Learning.* PhD thesis, Toyota Technological Institute at Chicago, 2017.

Quynh Nguyen and Matthias Hein. The loss surface of deep and wide neural networks. In *Proceedings of The 34th International Conference on Machine Learning*, volume 70, pages 2603–2612, Sydney, Australia, 2017.

Atsushi Nitanda and Taiji Suzuki. Functional gradient boosting based on residual network perception. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 3819–3828, Stockholm, Sweden, 2018.

Kaare Brandt Petersen and Michael Syskind Pedersen. The matrix cookbook, version: November 15, 2012. Technical report, Technical University of Denmark, 2012.

Gabriel Peyré and Marco Cuturi. Computational Optimal Transport. 2018.

Allan Pinkus. Density in approximation theory. *Surveys in Approximation Theory*, 1:1–45, 2005.

Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, 2017.

Radford M. Neal. *Bayesian Learning for Neural Networks.* Springer-Verlag New York, 1996.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37, pages 1530–1538, Lille, France, 2015.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of The 28th International Conference on Machine Learning*, pages 833–840, Belleview, USA, 2011.

Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms.* MIT Press, 2012.

Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with ReLU activation function. 2017.

John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis.* Cambridge University Press, 2004.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37, pages 2256–2265, Lille, France, 2015.

Sho Sonoda and Noboru Murata. Sampling hidden parameters from oracle distribution. In *24th International Conference on Artificial Neural Networks*, pages 539–546, Hamburg, Germany, 2014.

Sho Sonoda and Noboru Murata. Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis*, 43(2):233–268, 2017a.

Sho Sonoda and Noboru Murata. Double continuum limit of deep neural networks. In *ICML 2017 Workshop on Principled Approaches to Deep Learning*, pages 1–5, Sydney, Australia, 2017b.

Sho Sonoda and Noboru Murata. Transportation analysis of denoising autoencoders: A novel method for analyzing deep neural networks. In *NIPS 2017 Workshop on Optimal Transport & Machine Learning*, pages 1–10, Long Beach, USA, 2017c.

Sho Sonoda, Isao Ishikawa, Masahiro Ikeda, Kei Hagihara, Yoshihiro Sawano, Takuo Matsubara, and Noboru Murata. Integral representation of shallow neural network that attains the global minimum. 2018.

Daniel Soudry and Yair Carmon. No bad local minima: Data independent training error guarantees for multilayer neural networks. 2016.

Taiji Suzuki. Fast generalization error bound of deep learning from a kernel perspective. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, volume 84, pages 1397–1406, Playa Blanca, Lanzarote, Canary Islands, 2018.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, Columbus, USA, 2014.

Asuka Takatsu. Wasserstein geometry of Gaussian measures. *Osaka Journal of Mathematics*, 48(4):1005–1026, 2011.

Matus Telgarsky. Benefits of depth in neural networks. In *29th Annual Conference on Learning Theory*, pages 1–23, 2016.

Alexandre B Tsybakov. *Introduction to Nonparametric Estimation.* Springer-Verlag New York, 2009.

Vladimir Naumovich Vapnik. *Statistical Learning Theory.* Wiley, 1998.

Cédric Villani. *Optimal Transport: Old and New.* Springer-Verlag Berlin Heidelberg, 2009.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of The 25th International Conference on Machine Learning*, pages 1096–1103, Helsinki, Finland, 2008.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

Grace Wahba. *Spline Models for Observational Data*. SIAM, 1990.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833, Zurich, Switzerland, 2014.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations 2017*, Toulon, France, 2017.

Ruiyi Zhang, Changyou Chen, Chunyuan Li, and Lawrence Carin. Policy optimization as Wasserstein gradient flows. In *Proceedings of The 35th International Conference on Machine Learning*, volume 80, pages 5737–5746, Stockholm, Sweden, 2018.