國立政治大學應用數學系

博士學位論文

基於深度殘差神經網路的流形嵌入與霍奇排名的連續性之探討

Manifold Embedding with Deep ResNet and A Study of the Continuity of HodgeRank

指導教授：蔡炎龍　博士
劉宣谷　博士

研究生：林澤佑　撰

中 華 民 國　109　年　12　月

DOI:10.6814/NCCU202100296

中文摘要

仿造人腦的功能性，深層神經網路被建立用於萃取高階訊息。從數學的觀點來看，神經網路可以視為在適當定義遇上近似任意函數的近似器。

為了展示深層神經網路的威力，我們在本論文的第一部分考慮神經網路的兩種不同形式的應用。第一種應用是源於衍生性金融商品的定價模型，而另一種應用則是將基於仿射空間的流形重建演算法改寫為一殘差神經網路的學習過程，而這樣的改寫提供了深層神經網路在幾何演算法上的潛在應用。

本論文的第二個部分，我們聚焦在 HodgeRank，一個基於組合霍奇理論的逐對排名演算法。我們首先會回顧組合貨奇理論的背景知識，接著，我們考慮 HodgeRank 在線上同儕互評上之應用。最後，將 HodgeRank 視為 Moore-Penrose 廣義逆算子與矩陣-向量乘法的合成函數，我們可以探討 HodgeRank 的連續性。最後，我們從圖的角度證明了關於 HodgeRank 的一個連續性定理。

關鍵字：深層神經網路、深度殘差網路、移動邊界問題、流形重建、霍奇排名、組合霍奇理論、拉普拉斯矩陣、同儕互評

DOI:10.6814/NCCU202100296

Abstract

Deep neural networks are modeled to extract higher-level information in a way that is like the function of the human brain. From a mathematical perspective, neural networks are function approximators, which can approximate any function on a suitable domain.

In the first part of this dissertation, we consider two different tasks to demonstrate the power of deep neural networks. One task is derived from a option pricing model of financial derivatives while another task is to rewrite an affine subspaces based manifold reconstruction algorithm to a learning process of a deep residual network. Such reformulation offers a possibility for potential application of deep neural networks to various geometrical related algorithms.

In the second part, we focus on the HodgeRank, a pairwise ranking method based on the combinatorial Hodge theory. We first quick review the background of combinatorial Hodge theory, then a real world application of HodgeRank to online peer assessment is provided. Finally, by considering HodgeRank as a composition of Moore-Penrose generalized inverse and matrix-vector product, we can study the continuity of HodgeRank. In terms of graph, a theorem of continuity of the HodgeRank is provided in the end.

Keywords: deep neural network, deep residual network, moving boundary problem, manifold reconstruction, HodgeRank, combinatorial Hodge theory, graph Laplacian, peer assessment

DOI:10.6814/NCCU202100296

Contents

DOI:10.6814/NCCU202100296

DOI:10.6814/NCCU202100296

DOI:10.6814/NCCU202100296

List of Tables

DOI:10.6814/NCCU202100296

List of Figures

DOI:10.6814/NCCU202100296

Chapter 1

Introduction

1.1 Deep Learning

In recent years, artificial intelligence (AI) has been utilized in many applications such as machine translation [1, 89], speech recognition [21, 39], autonomous driving [74], image recognition [28, 54], and game playing [70]. In many cases, developing AI applications start with training deep neural networks [3].

Deep neural network is a class of model of machine learning in AI, which allows the computer to learn features of data with multiple levels of abstraction [38, 55]. Among most used deep neural network architecture, feed-forward neural network, convolutional neural network [56] and recurrent neural network (RNN) [15, 36, 45]. Since the training of a deep neural network is time-consuming and prone to overfitting due to the presence of noise from data, an alternative structure of neural network, called the deep residual network (ResNet) [43], was considered to solve these issues. ResNet contains a structure, called skip-connection, which skips one or more layers. Hence, information is allowed to pass to deeper layers so that the vanishing gradient problem can be prevented [76]. Motivated by differential equations, the structure of skip-connection can be seen as an Euler discretization of a continuous transformation of a time evolving ordinary differential equations [41, 65, 79]. This perspective turns a deep neural network model into a family of a continuous-depth models, which allows end-to-end training of ordinary differential equations [13].

Recently, Transformer [89] plays an important role in the development of natural language processing (NLP), which outperforms most RNN-based NLP models. Several Transformer-

DOI:10.6814/NCCU202100296

based models such as GPT-3 [8], BERT [25], ELMo [75], and ULMFiT [48] are widely used in multiple NLP tasks. However, the Transformer family is beyond the scope of this dissertation. For a comprehensive survey of Transformer, please refer [86].

Theoretically, the well-known universal approximation theorem [20] of neural networks establish a mathematical foundation that neural network with single hidden layer can be applied to approximate any continuous function in any precision. However, the approximation theorem offers no information about the width of certain single hidden layer. Other than above depth-bounded universal approximation theorem, the width-bounded version [66] provides a more interesting result for deep neural networks. The universal approximation theorem for ResNet proposed in [58] which provides a magnificent result for the capability of ResNet.

To demonstrate the power of depth-bounded universal approximation theorem, a neural network approach is applied to solve a moving boundary problem arising in pricing formula in chapter 3.

1.2 Manifold Reconstruction

In many science and engineering fields, high-dimensional data are assumed to be sampled from a manifold of lower dimension embedded in a Euclidean space. This leads to the problem of manifold learning, which aims approximate the underlying manifold of dataset of interest [6]. The goal of manifold learning is to extract the intrinsic information from high-dimensional data using the structure of manifold. In other words, high-dimensional data can be explained by latent variables of lower dimension

A practical technique is to fit data points with a line or a lower dimensional affine space [18, 33, 97]. However, one issue is that data points may not sampled from flat object but from a nonlinear class such as curve or a smooth manifold. The presence of noise is another issue which leads to unexpected results.

The former issue was initially addressed in [14] with a triangulation-based algorithm. In [73], a method based on simplicial complexes was proposed under the presence of noise, which solves both issues. In [4, 5], iterative methods were considered to reduce the computational cost of triangulation of the dataset. Later in a series of works [29, 30, 31], manifold was reconstructed by gluing by affine subspaces, which rely on the computation of

DOI:10.6814/NCCU202100296

affine projections. A parameterization-free projection method was proposed in [61] to avoid the computation of local plane. In [84], a moving least-squares method [57] was applied for manifold reconstruction given the intrinsic dimension of the underlying manifold.

1.3 Pairwise Ranking and Combinatorial Hodge Theory

According to [68], most people are unable rank more than 10 items at the same time. Also, bias occurs when the number of ranked items is large. To prevent such situation, pairwise comparison offers a solution to reduce bias and also provides a way to fill missing values.

Pairwise comparison is a ranking process that compares candidates in pairs to determine which candidate is preferred. In other words, if candidates are assumed with some unknown underlying ranking, one may want to recover such ranking from pair comparisons [27]. In ranking algorithm, each candidate is assigned a score according to their preference, or importance. New candidates can also be sorted quickly once the certain ranking process is built [64].

By measuring candidates in pairs for their relative preference or importance, pairwise comparisons are represented using a pairwise comparison matrix. Under different scale of measurements, comparison matrix could be represented either on the additive scale or on the multiplicative scale [80]. To find a global ranking of all candidates from pairwise comparison matrix, a score vector for candidates is generated to approximate the observed pairwise comparison matrix. Score vector can be generated by various methods under different assumption and conditions. Among all ranking methods, HodgeRank [51] and PerronRank [80] are two of the most applied in real world applications. The connection between these ranking algorithms is being discussed in [87].

HodgeRank has been widely applied in many fields [60, 83]. HodgeRank can be obtained simply by computing the row geometric mean of the comparison matrix [19] on the multiplicative scale. However, on the additive scale, HodgeRank can be viewed as a consistent part of a flow on a finite graph, which is explained by combinatorial Hodge theory [51]. Thus, nice properties of HodgeRank are guaranteed thanks to combinatorial Hodge theory.

Combinatorial Hodge theory is an algebraic topology based theory which decomposes a edge flow on a graph into the direct sum of gradient, harmonic and curl flows. Accordingly, by

DOI:10.6814/NCCU202100296

treating a pairwise ranking as an edge flow on a graph, one can be recover the global ranking from the gradient flow while ranking error can be quantified by other two flows.

1.4 Organization of this Dissertation

The dissertation is organized as follows.

In Part I, we focus on the deep neural networks. In the second part. We will focus on the HodgeRank, a ranking method based on the combinatorial Hodge theory. In the end of this dissertation, we present our conclusions and possibilities for future research including few unsolved problems in this dissertation.

In Part I, the mathematical definition of a neural network is introduced in chapter 2. Next, we recall the well-known universal approximation theorems for feed-forward neural network and some similar results. Then, residual neural network, a neural net that consists of residual blocks, is introduced Also, a "dual" version of universal approximation theorems for residual neural network is stated here. Finally, graph neural network is introduced along with various versions of attention mechanism. In chapter 3, we propose a neural network approach to construct an approximate solution of a moving boundary problem arising from pricing of American volatility options. In chapter 4, a manifold reconstruction algorithm proposed by [30] is considered. Also, combined with manifold interpolation algorithm and the structure of the residual network, we propose a neural network based manifold interpolation algorithm.

In Part II, we will recall some basic knowledge about the combinatorial Hodge theory in chapter 5. In chapter 6, to study the continuity of HodgeRank, we recall the definition of a generalized inverse of a matrix. By analyzing the continuity of the pseudoinverse operator † on graph Laplacian matrices, one condition about the continuity of HodgeRank is proved. In chapter 7, an real world application of HodgeRank to online peer assessment is presented.

In the end, the conclusion of this dissertation is presented chapter 8. Also, two unsolved problems which relate to our research are stated.

DOI:10.6814/NCCU202100296

Part I

Deep Neural Network and Manifold Reconstruction

DOI:10.6814/NCCU202100296

Chapter 2

Deep Learning

In recent year, deep learning has been widely developed and applied in various fields to handle real-world problems such as recommendation system, computer vision, autonomous driving, game playing, etc.

Deep learning is derived from the field of machine learning. However, unlike most shallow-structured algorithms in machine learning algorithm, deep-structured models are considered in deep learning. This difference resulting from the fact that data can be easily accessed and collected nowadays.

A central component of deep learning is the *neural network*. Neural network is a parametric model originally inspired by neurobiology, which imitates the computing power and flexibility of the brain. Parameters can be optimized through the learning process.

In this section, we first briefly introduce the model of neural network. Then we state and prove the well-known universal approximation theorem for shallow networks. Note that there are various universal approximation theorems under different conditions for different purposes.

We will cover some well-known versions of universal approximation theorem here.

2.1 Standard Structure of Neural Network

We first state the definition of a feed-forward neural network between Euclidean spaces. This is a modified definition from the one given in the book [38].

Definition 2.1.1. *A feed-forward neural network (or a neural network simply) of depth* d *is a parametric function* $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$ *defined by a series of alternative compositions of affine and*

DOI:10.6814/NCCU202100296

nonlinear functions with a set of parameters $\theta = \{(W_i, b_i) \mid 0 \le i \le d\} \in \Theta = \bigsqcup_{i=0}^{d} \left(\mathbb{R}^{h_{i+1} \times h_i} \times \mathbb{R}^{h_{i+1}}\right)$*, where* $\bigsqcup$ *is the disjoint union. That is,* f_θ *of the form:*

$$f_\theta = L_d \circ \sigma_{d-1} \circ L_{d-1} \circ \cdots \sigma_0 \circ L_0,$$

where $L_i : \mathbb{R}^{h_i} \to \mathbb{R}^{h_{i+1}}$ *is an affine transformation defined by* $L_i(x) = W_i x + b_i$ *and* $\sigma_i : \mathbb{R}^{h_{i+1}} \to \mathbb{R}^{h_{i+1}}$*, called activation function of hidden layer* i*, is a nonlinear function that operates on vectors component-wisely . That is,* $\sigma_i(x) = \big(\sigma_i(x_1), \sigma_i(x_2), \cdots, \sigma_i(x_k)\big)^T$ *for column vector* $x = (x_1, \cdots, x_k)^T \in \mathbb{R}^k$.

The map $\sigma_0 \circ L_0$ is called the input layer and $\circ L_d$ is called the output layer, and $\sigma_i \circ L_i$ is called the i-th hidden layer for $i = 1, 2, \cdots, d-1$. Each layer $\sigma_i \circ L_i : \mathbb{R}^{h_i} \to \mathbb{R}^{h_{i+1}}$ can be written of the form

$$\sigma_i(W_i x + b_i) = \big[\sigma_i(W_i x + b_i)_j\big]_{1 \le j \le h_{i+1}} = \big[\sigma_i(W_{ij} x + b_{ij})\big]_{1 \le j \le h_{i+1}}$$

where W_{ij} is the j-th row of the $h_{i+1} \times h_i$ matrix W_i and b_{ij} is the j-th component of $b_i \in \mathbb{R}^{h_{i+1}}$.

Hence, to compute $\sigma_i \circ L_i(x)$, it suffices to compute $\sigma(W_{ij} x + b_{ij})$ for all $1 \le j \le h_{i+1}$. Note that $x \mapsto y = \sigma(Wx + b)$, where $W \in \mathbb{R}^{1 \times h}$ and $b \in \mathbb{R}$ is a function from $\mathbb{R}^h$ to $\mathbb{R}$, which can be represented as a diagram shown in Figure 2.1 below. This is called the mathematical model of a single neuron. Here, x is called the input vector, and $y = \sigma(Wx + b)$ is called the output vector defined by computing the inner product of x and a weight vector $W = [w_i]_{1 \le i \le n} \in \mathbb{R}^{1 \times h}$ plus the bias $b \in \mathbb{R}$ under σ.

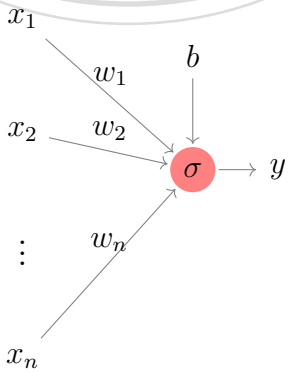


Figure 2.1: Mathematical model for a neuron: $y = \sigma\big(\sum_{i=1}^{n} w_i x_i + b\big)$

DOI:10.6814/NCCU202100296

The following terms can be used to design and characterize a neural network:

$\begin{cases} d\text{: depth of } f_\theta \\ \sigma_i\text{: activation function of layer } i \\ h_i\text{: width of the hidden layer } i \text{ of } f_\theta \text{ for } 1 \leq i \leq d \\ h_0 = n\text{: dimension of the input layer} \\ h_{d+1} = m\text{: dimension of the output layer} \end{cases}$

For a fixed design of neural network, the corresponding parametric model is the collection of all neural networks with the same design

$$\mathcal{H}_\Theta = \{f_\theta : \mathbb{R}^n \to \mathbb{R}^m \mid \theta \in \Theta\}$$

which has an one-to-one correspondence to parameters $\theta \in \Theta$. Since $\Theta = \bigsqcup_{i=0}^{d} \left(\mathbb{R}^{h_{i+1} \times h_i} \times \mathbb{R}^{h_{i+1}}\right)$ is isomorphic to $\mathbb{R}^K$, where $K = \sum_{i=0}^{d} \left[(1 + h_i)h_{i+1}\right]$. That is, each f_θ is fully determined by its design and K parameters.

We can classify a neural network f_θ by its depth d and all its width of each layers $\{h_i\}_{i=0}^{d+1}$.

Definition 2.1.2. *For $N \in \mathbb{N}$, f_θ is called a*

1. *shallow neural network if $d \leq 2$. i.e., f_θ contains at most two hidden layer.*
2. *deep neural network for $d \geq 3$. i.e., f_θ contain at least three layers.*
3. *width-N network if $h_i \leq N$ for all $1 \leq i \leq d$.*

In some literatures [2, 10, 49, 92], a neural network with one hidden layer (that is, $d = 1$) is called a single hidden layer feed-forward neural network (SLFN) instead of a shallow neural network. However, we will use the term shallow nets for the rest of this section. Sometimes, we say f_θ is a narrow net if it is width-N network without indicating its width bound N.

In Table 2.1, some popular types of activation functions used in deep learning out of which commonly used are listed.

DOI:10.6814/NCCU202100296

Table 2.1: Common choices of activation function

Name	Formula	Derivative
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	$\sigma'(x) = \sigma(x)(1-\sigma(x))$
Hyperbolic tangent	$\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$\sigma'(x) = 1 - \sigma^2(x)$
Rectified linear unit (ReLU)	$\sigma(x) = \max\{x, 0\}$	$\sigma'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$
Leaky ReLU [67]	$\sigma(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases}$	$\sigma'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ \alpha & \text{if } x < 0 \end{cases}$

2.1.1 Learning Process

Let $\mathcal{D} = \{(x_i)\}_{i=1}^N$ be data points in $\mathbb{R}^n$. We often aim to extract information on $\mathcal{D}$. Thus, we propose one or multiple mathematical models to work on this dataset.

In supervised learning, we associate each data $x_i \in \mathbb{R}^n$ with a label $y_i \in \mathbb{R}^m$. One common application of neural network in a supervise manner is to interpolate all points in $\mathcal{D}$ with a sutable designed neural network f_θ. i.e., find a f_θ so that

$$f_\theta(x_i) = y_i \text{ for all } 1 \leq i \leq N \tag{2.1.1}$$

For a fixed design Θ of neural network, our goal is to find an optimal $f_{\theta^*} \in \mathcal{H}_\Theta = \{f_\theta : \mathbb{R}^n \to \mathbb{R}^m \mid \theta \in \Theta\}$ (if exists) so that (2.1.1) holds.

For $f_\theta \in \mathcal{H}_\Theta$, we can evaluate how f_θ far from (2.1.1) by a *cost function* or a *loss function*. To measure this gap, we first consider a nonnegative function $L : \mathbb{R}^m \times \mathbb{R}^m \to [0, \infty)$ so that

$$L(f_\theta(x_i), y_i) = 0 \iff f_\theta(x_i) = y_i \text{ for all } 1 \leq i \leq N$$

Such L is called a *loss function* or a *cost function* of f_θ at a point (x_i, y_i), $L(f_\theta(x_i), y_i)$ is called a loss of f_θ at (x_i, y_i) under criterion L. Further, the loss function $\mathcal{L} : \Theta \to [0, \infty)$ assigns each θ a non-negative number by computing the mean of $L(f_\theta(\cdot), \cdot)$ on all data points of $\mathcal{D}$, that is:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} L(f_\theta(x_i), y_i)$$

Therefore, finding a optimal θ^* satisfying (2.1.1) is equivalent to solve the following

DOI:10.6814/NCCU202100296

minimization problem

$$\min_{\theta\in\Theta} \mathcal{L}(\theta) = \mathcal{L}(W_0, b_0, W_1, b_1, \cdots, W_d, b_d) \tag{2.1.2}$$

If L is chosen with nice properties such as convexity or Lipschitz continuity [72, 94]. Then there are several standard optimization techniques can be applied to deal with (2.1.2). The process to find the optimal θ^* using loss function $\mathcal{L}$ is called the *learning process* under criterion $\mathcal{L}$.

Some common used loss functions are listed in Table 2.2

Table 2.2: Common choices of loss function

Name	Formula of $L(x, y)$
Binary loss	$\mathbb{1}_{\{x\neq y\}}$
Mean square error	$\frac{1}{2}\|x-y\|_2^2 = \frac{1}{2}\sum_{i=1}^{n}(x_i-y_i)^2$
Mean absolute error	$\|x-y\|_1 = \sum_{i=1}^{n}\|x_i-y_i\|$
Negative log likelihood	$-\sum_{i=1}^{n}[y_i\log(x_i)+(1-y_i)\log(1-x_i)]$

If the loss function L is chosen to be differentiable, then a common procedure of learning process is to apply gradient descent method from elementary calculus. There are several different flavors of stochastic gradient descent, which can be all seen throughout the literature. There are various learning algorithm based on the gradient method such as Adam [52] or Nesterov momentum [71].

In unsupervised learning, data points are untagged, that is, each x_i associates with no label y_i, that is, $\mathcal{D}$ is the only set we have. Based on the result we intended to achieve, various algorithms would be considered to deal with $\mathcal{D}$. In some case, loss functions could be not analytic or even differentiable. However, in some case, we can reformulate an unsupervised learning problem into a supervised one. For instance, an Autoencoder is to reconstruct a data by learning the identity function. That is, the label of each data x_i is x_i itself. This makes the selection of loss function being reduced into supervised case.

DOI:10.6814/NCCU202100296

2.1.2 Universal Approximation Theorem for Shallow Net

A famous result for neural networks is that that any continuous function on a compact set of $\mathbb{R}^n$ can be approximated by a neural network in any precision. Computationally, any continuous function can be replaced by a neural network without losing precision. To state this theorem, we need the following notations.

Let $I_n = [0,1]^n$ be n-dimensional unit cube in $\mathbb{R}^n$. Denote the set of all signed Radon measures defined on I_n by $M(I_n)$. The set of all continuous functions defined on I_n is denoted by $C(I_n)$. Each $f \in C(I_n)$ associates with a supremum norm

$$\|f\|_\infty = \sup_{x \in I_n} |f(x)|$$

Definition 2.1.3 ([20]). *We say that σ is discriminatory on I_n if $\mu \in M(I_n)$ on I_n so that*

$$\int_{I_n} \sigma(Wx + b) d\mu(x) = 0$$

for any $W \in \mathbb{R}^{1\times n}$, $b \in \mathbb{R}$, then $\mu = 0$

We need the following geometric form of the Hahn-Banach Extension Theorem and the Riesz Representation Theorem.

Theorem 2.1.4 (Hahn-Banach Extension Theorem [7, Corollary 1.8]). *Let $F \subseteq E$ be a linear subspace with $\overline{F} \neq E$. Then there exists a nonzero linear functional L on E such that*

$$L|_F \equiv 0$$

Theorem 2.1.5 (Riesz Representation Theorem [34, Theorem 7.17]). *Let X be a locally compact Hausdorff space. For $\mu \in M(X)$ and $f \in C_0(X)$, define $L_\mu(f) = \int_X f d\mu$. Then $\mu \mapsto L_\mu$ is an isometric isomorphism from $M(X)$ to $C_0(X)^*$.*

Note that the domain we talk about here is $I_n = [0,1]^n$, hence, the set of continuous functions vanish at infinity $C_0(I_n)$ coincides with $C(I_n)$.

The shallow net f_θ we consider below contains one hidden layer of width h, and is of the form

$$f_\theta(x) = S\sigma(Wx + b) = \sum_{i=1}^{h} s_i \sigma(W_i x + b_i) \tag{2.1.3}$$

DOI:10.6814/NCCU202100296

where W_i is the i-th row of W.

Denote $\mathcal{N}_\sigma$ be the set of all shallow net f_θ of the form (2.1.3) of any width $h \in \mathbb{N}$ with a fixed activation function σ. Then we have the following theorem.

Theorem 2.1.6 (Universal Approximation Theorem [20])**.** *If σ is continuous and discriminatory, then $\mathcal{N}_\sigma$ is dense in $C(I_n)$ with respect to the uniform norm $\|\cdot\|_\infty$.*

Proof. Suppose not, then the closure of $\mathcal{N}_\sigma$ is not dense in $C(I_n)$. That is, $\overline{\mathcal{N}_\sigma}$, is a proper closed subspace of $C(I_n)$. By Hahn-Banach Extension Theorem, there exists a nonzero linear functional $L \in C(I_n)^*$ so that

$$L|_{\mathcal{N}_\sigma} \equiv 0 \tag{2.1.4}$$

By Riesz Representation Theorem, there exists $\mu \in M(I_n)$ so that the linear functional L is of the form

$$L(f) = \int_{I_n} f(x) d\mu(x)$$

for any $f \in C(I_n)$.

By (2.1.3), we have

$$0 = L(f_\theta) = \int_{I_n} S\sigma(Wx + b) d\mu(x) = \sum_{i=1}^{h} s_i \int_{I_n} \sigma(W_i x + b_i) d\mu(x)$$

for any $f_\theta = S\sigma(Wx + b) \in \mathcal{N}_\sigma \subseteq C(I_n)$

By choosing suitable s_i, since σ is discriminatory, we can conclude that $\mu = 0$. This implies that $L \equiv 0$ is a trivial linear functional, which leads us to a contradiction. Hence, $\mathcal{N}_\sigma$ is dense in $C(I_n)$. □

Another proof of Theorem 2.1.6 is to apply the Stone-Weierstrass theorem by considering $\mathcal{N}_\sigma$ as the linear span of $\{\sigma(Wx + b) \mid w \in \mathbb{R}^n, b \in \mathbb{R}\}$. However, both are existence proofs, which provide no additional information about how to design width h.

In fact, combined with Lusin's theorem [34, p. 64], one can prove that any finite Borel measurable function on I_n can be well-approximated with a shallow net on a subset of I_n by removing a subset of I_n with small measure.

The universal approximation theorem we state above is called a depth-bounded type. An alternative type of the universal approximation theorem is width-bounded. That is, if we restrict the width of each layer to be bounded, does the universal approximation property holds as the

DOI:10.6814/NCCU202100296

depth goes to infinity? There are various width-bounded universal approximation theorems for deep neural network showing the denseness of particular deep narrow networks.

In [42], it has been shown that deep narrow networks with the ReLU activation function are dense in $C(K, \mathbb{R}^m)$ for any compact set $K \subseteq \mathbb{R}^n$, and require only width $n+m$. A similar result have proved in [66] that deep narrow networks with the ReLU activation function are dense in $L^1(\mathbb{R}^n)$, the set of all Lebesgue integrable functions on $\mathbb{R}^n$, with width $n+4$. Later in 2.2.2, we will see an impressive universal approximation theorem for residual network, which will be introduced soon.

2.2 Residual Network

In this section, we introduce a variant of neural network, called the residual network, which is proposed in [43].

The major difference of residual network from standard feed-forward neural network is the design of *skip connection*. To illustrate this structure, we introduce the residual block below.

2.2.1 Residual Network

Definition 2.2.1. *An residual block (ResBlock) is a function $F_{W,\theta} : \mathbb{R}^n \to \mathbb{R}^m$ defined by*

$$F_{W,\theta}(x) = Wx + f_\theta(x) \tag{2.2.1}$$

where $W \in \mathbb{R}^{m\times n}$ and $f_\theta : \mathbb{R}^n \to \mathbb{R}^m$ is a parametric function. This structure is sometimes called an additive skip-connection.

If f_θ is narrow, then we call $F_{W,\theta}$ a narrow ResBlock.

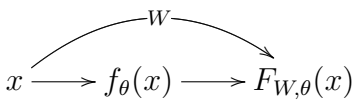


Figure 2.2: Functions of a ResBlcok

Intuitively speaking, $F_{W,\theta}$ not only extract information from x through f_θ, but also preserves some information Wx from x. This makes important information behind x being left for the next layer.

DOI:10.6814/NCCU202100296

Remark 2.2.2. *1. For the case $m = n$, we can fix $W = I_n$ as the identity matrix and ignore it from the subscript of $F_{W,\theta}$. Otherwise, W will be considered as a trainable weight during training process.*

2. Initially, the parametric function f_θ is chosen as a shallow convolutional neural network in [43]. Later in [99], authors consider f_θ as a sum of k shallow nets $f_{\theta_i} : \mathbb{R}^n \to \mathbb{R}^n$ with different design. That is, $F_{\{\theta_i\}_{i=1}^k}(x) = x + \sum_{i=1}^{k} f_{\theta_i}(x)$

Definition 2.2.3. *A residual network (ResNet) is a parametric function containing at least one ResBlock.*

The origin structure of ResNet proposed in [43] consists of ResBlocks only. Two consecutive ResBlocks are connected by multiplying a weight on the output of the former ResBlock, this makes each ResBlocks could handle vectors of different dimensions.

One advantage of ResNet is that we could construct a very deep neural network without gradient vanishing during training process. An interpretation is that a ResNet could be seen as a collection of many paths of various length due to the existence of skip-connections. In other words, a ResNet could be viewed as an ensemble of relative smaller feedforward neural networks which are easier to train. For more detail about the discussion, please refer [90].

There is also an universal approximation theorem for ResNet if we allow its depth goes to infinity with width-bounded ResBlocks. We slightly modify notation and terminology from [58].

2.2.2 Universal Approximation Theorem for Narrow ResNet

Lin and Jegelka [58] have shown that ResNet with ReLU activation and only one neuron is a universal approximator of functions in $L^1(\mathbb{R}^n)$. To state such universal approximation theorem for ResNet, we first consider the following type of ResNet.

Definition 2.2.4. *1. The function $F : \mathbb{R}^n \to \mathbb{R}^n$ is called a k-neuron shallow ResBlock with activation σ if F is of the following form:*

$$F(x) = x + S\sigma(Wx + b)$$

DOI:10.6814/NCCU202100296

where $W \in \mathbb{R}^{k\times n}, b \in \mathbb{R}^k$ and $S \in \mathbb{R}^{n\times k}$. In other words, F is a ResBlock by choosing $W = I_n$ and $f_\theta(x) = S\sigma(Wx + b)$ in (2.2.1).

2. *The set of all k-neuron shallow ResBlocks on $\mathbb{R}^n$ with activation σ is denoted by $\mathcal{N}^{(1)}_{n,k,\sigma}$.*

3. *Denote the set of all composition of d functions in $\mathcal{N}^{(1)}_{n,k,\sigma}$ by $\mathcal{N}^{(d)}_{n,k,\sigma}$.*

   That is, for any $f \in \mathcal{N}^{(d)}_{n,k,\sigma}$, f is of the form

$$f = F_d \circ F_{d-1} \circ \cdots \circ F_1$$

   where $F_1, F_2, \cdots, F_d \in \mathcal{N}^{(1)}_{n,k,\sigma}$.

Then, we have the following universal approximation theorem for narrow ResNet using only ReLU as activation function.

Theorem 2.2.5 (Universal Approximation Theorem for Narrow ResNet [58])**.** *$\mathcal{L}\bigcup_{d=1}^{\infty}\mathcal{N}^{(d)}_{n,1,\mathrm{ReLU}}$ is dense in $L^1(\mathbb{R}^n)$ under the L^1-norm, where $\mathcal{L}X = \{L \circ f \mid L : \mathbb{R}^n \to \mathbb{R}$ is linear , $f \in X\}$*

Proof. See [58, Theorem 3.1] for the proof. □

In fact, we can relax $k = 1$ into arbitrary width.

Corollary 2.2.6. *$\mathcal{L}\bigcup_{d=1}^{\infty}\mathcal{N}^{(d)}_{n,k,\mathrm{ReLU}}$ is dense in $L^1(\mathbb{R}^n)$ under the L^1-norm for any $k \in \mathbb{N}$.*

The central part of the proof of Theorem 2.2.5 is to approximate step function first, then apply the denseness property of step functions in $L^1(\mathbb{R}^n)$. One trick is that shifting is considered to make sure the supports of all ResBlocks are non-overlapping.

However, the proof of Theorem 2.2.5 is given by combining a constructive proof and an existence proof, which provides merely benefits of constructing and training a deep narrow ResNet.

The key advantage of the ResNet is that it contains different paths to pass some of its ResBlocks. Hence, we can consider the local behavior of ResNet other than global. Later in chapter 4, we will see how this property can be applied to solve a manifold interpolation problem.

DOI:10.6814/NCCU202100296

2.3 Graph Neural Network

In various fields, the connection between different objects are recorded and represented as a graph, which consists of nodes and edges to indicate each object and the relationship between them. To extend the capacity of neural network from Euclidean data into non-Euclidean one, graph neural network was proposed in [82] which extends classical neural network models.

Here, we briefly introduce what is the graph neural network and how a graph can be viewed as an input of a graph neural network.

2.3.1 Graph

In this subsection, we recall some basic definitions from graph theory. Some of definitions below will be used in chapter 5. Through this subsection, V will denote a nonempty finite set. The notation $\binom{n}{k}$ denotes by the set of k-elements of subsets of V, and V^k denotes the set of k-tuples of elements of V.

Definition 2.3.1. *A (simple) graph is an ordered pair $G = (V, E)$, where V is called the vertex set and $E \subseteq \binom{n}{2} \triangleq \{(i, j) \in V^2 \mid i \neq j\}$ is called the edge set.*

- *If element in E is not ordered, then we say G is an undirected graph. Otherwise, G is called a directed graph.*
- *If V is a finite set, then G is called a finite graph on n vertices. In this case, we write $V = \{1, 2, \cdots, n\} \triangleq [n]$.*
- *A finite undirected graph G of vertex n, denoted by $G = K_n$, is called a complete graph if $E = \binom{n}{2}$.*

Let G be a graph. Sometimes we use $V(G)$ and $E(G)$ to represent the vertex and the edge set of graph G. These notations are useful when we discuss more than one graph. Now, we see how a graph is associated with a matrix.

Definition 2.3.2. *Let G be a finite graph on n vertices, and $i \in V(G)$.*

DOI:10.6814/NCCU202100296

1. *Let $A = A_G$ be an $n \times n$ matrix defined by matrix of G, which is defined by*

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{if } (i,j) \notin E \end{cases}$$

   A_G is called the adjacent of the graph G.

2. *The set of all neighborhoods of vertex i is denoted by*

$$\mathcal{N}_i = \{j \in V \mid (i,j) \in E\}.$$

3. *The degree of vertex i, denoted by* $\deg(i)$*, is defined by* $\deg(i) = |\mathcal{N}_i|$.

Remark 2.3.3. *Let G be a finite graph on n vertices, and $i \in V(G)$.*

1. *If G is undirected, then its adjacent matrix A is symmetric.*
2. *$A_{ij} = 1$ for all $j \in \mathcal{N}_i$.*
3. *The degree of i can be computed by* $\deg(i) = \sum_{j=1}^{n} A_{ij} = \sum_{j \in \mathcal{N}_i} A_{ij}$.

Figure 2.3 shows an example of a finite undirected graph $G = (V, E)$ of 5 vertices, where $V = \{1, 2, 3, 4, 5\}$, $E = \{(1,2), (2,1), (2,3), (3,2), (2,4), (4,2), (4,5), (5,4)\}$ and

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 2.4 shows an example of a finite directed graph $G = (V, E)$ of 5 vertices, where

DOI:10.6814/NCCU202100296

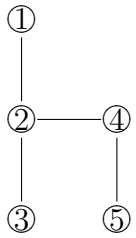


Figure 2.3: Example of an undirected graph

$V = \{1, 2, 3, 4, 5\}$, $E = \{(1, 2), (2, 3), (4, 2), (4, 5), (5, 4)\}$and

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

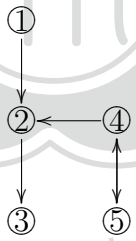


Figure 2.4: Example of a directed graph

Definition 2.3.4. *Let $G = (V, E)$ be a graph with adjacent matrix A_G. If each vertex $i \in V$ is associated with a vector $h_i \in \mathbb{R}^F$. Then we called $(\{h_i\}_{i\in[n]}, A_G)$ a graph-structured data based on the graph G.*

Later, we can see how a graph-structured data be viewed and manipulated as a input data of a neural network.

A natural way to construct a graph-structured data from a graph G is to assign each vertex $i \in V$ with the i-th row (or column) of the adjacent matrix A. Hence, even for a graph with no extrinsic information about each vertex, we can still find a graph-structured data tautologically.

DOI:10.6814/NCCU202100296

2.3.2 Graph Attention Network

Here, we only introduce one type of graph neural network. We first define the traditional attention mechanism in deep learning.

Definition 2.3.5 (Softmax)**.** *The function* softmax : $\mathbb{R}^n \to \mathbb{R}^n$ *is defined by*

$$\text{softmax}(x_1, x_2, \cdots, x_n) = (y_1, y_2, \cdots, y_n)$$

where

$$y_i = \text{softmax}(x_1, x_2, \cdots, x_n)_i = \frac{e^{x_i}}{\sum_{k=1}^{n} e^{x_k}}$$

for $1 \leq i \leq n$*. is called the softmax function.*

Note that $y_i \in [0, 1]$ and $\sum_{i=1}^{n} y_i = 1$. In other words, softmax transforms a vector in $\mathbb{R}^n$ into a probability vector in $\mathbb{R}^n$.

Definition 2.3.6 (Attention mechanism)**.** *Let* $X = \{x_i\}_{i=1}^n$ *and* $Y = \{y_j\}_{j=1}^m$ *be two sets in* $\mathbb{R}^N$ *and* $\mathbb{R}^M$*, respectively. The attention mechanism is to apply to update each* x_i *to* x_i' *by considering the importance of each* y_j *to* x_i*. Usually,* x_i' *is given by* $\sum_{j=1}^{m} \alpha_{ij} y_j$*, a linear combination of* $\{y_j\}$*, where coefficient* α_{ij} *is called the attention score of* i *from* j*, is defined by two steps below.*

For fixed i*, to compute* α_{ij}*, we need to consider all importance between* x_i *and all* y_j*.*

1. *An alignment model* $\{A_{ij}\}$ *on* X *and* Y *is a function that assigns each* $x_i \in X$ *and* $y_j \in Y$ *a scalar* $A_{ij} \in \mathbb{R}$ *for* $1 \leq i \leq n$ *and* $1 \leq j \leq m$*. Sometimes,* A_{ij} *is called the attention coefficient of* x_i *and* y_j*.*

2. *Based on an alignment model* $\{A_{ij}\}$*, an attention score* a_{ij} *from* x_j *to* x_i *is defined by*

$$\alpha_{ij} = \text{softmax}(A_{i1}, A_{i2}, \cdots, A_{im})_j = \frac{e^{A_{ij}}}{\sum_{k=1}^{M} e^{A_{ik}}} \tag{2.3.1}$$

   for $1 \leq i \leq n$ *and* $1 \leq j \leq m$*.*

3. *With* α_{ij}*, we can update each* x_i *by*

$$x_i' = \sum_{j=1}^{m} \alpha_{ij} y_j$$

DOI:10.6814/NCCU202100296

for $1 \leq i \leq n$.

Note that the alignment model A_{ij} between x_i and y_j can be implemented in various way.

Let $(\{h_i\}_{i\in[n]}, A_G)$ be a graph-structured data, our goal is to identify a pattern of each vertex i by studying their feature vector under the structure of graph. This pattern consists of not only hidden information from vertex i, but also from some or all neighborhoods $\mathcal{N}_i$ of vertex i.

In a feed-forward neural network, we compute the linear combination of vectors from the output of one hidden layer as an input of next hidden layer. In a graph neural network, we transform each h_i with a similar computation process to get another feature vector h_i' of vertex i. However, we consider their convex combination instead, and the coefficients are attention scores defined below. This is called the graph attention mechanism, which is slightly different from the traditional one.

Definition 2.3.7 (Graph attention mechanism)**.** *Let* $(\{h_i\}_{i\in[n]}, A_G)$ *be a graph-structured data. Denote* $\overline{E(G)} = E(G) \cup \{(i,i) \mid i \in V(G)\}$ *and* $\overline{\mathcal{N}_i} = \mathcal{N}_i \cup \{i\}$ *for* $i \in V(G)$.

1. *An alignment model* $\{A_{ij}\}$ *between* i *and* h_j *is defined only if* $(i,j) \in E(G)$ *or* $i = j$.

2. *Based on an alignment model* $\{A_{ij}\}_{(i,j)\in\overline{E(G)}}$*, an attention score* a_{ij} *from* h_j *to* h_i *is defined by*

$$\alpha_{ij} = \frac{e^{A_{ij}}}{\sum\limits_{(i,k)\in\overline{E(G)}} e^{A_{ik}}} = \frac{e^{A_{ij}}}{\sum\limits_{k\in\overline{\mathcal{N}_i}} e^{A_{ik}}}$$

   for $1 \leq i \leq n$ *and* $1 \leq j \leq m$.

3. *With* α_{ij}*, we can update each* x_i *by*

$$x_i' = \sum_{(i,j)\in\overline{E(G)}} \alpha_{ij} y_j$$

   for $1 \leq i \leq n$.

To compute the attention score, one need to have an alignment model beforehand. There are various ways to define such alignment model. We introduce two famous examples below.

In [89], a different data structure is considered. Assume that the dataset is of the form $\{Q, K, V\}$, where $Q, K \in \mathbb{R}^{n\times d_k}$ and $V \in \mathbb{R}^{n\times d_n}$, then each data $x_i = (q_i, k_i, v_i)$ is a 3-tuple of i-th row of Q, K and V, respectively.

DOI:10.6814/NCCU202100296

For such data format, the scaled dot-product attention $A = [A_{ij}]_{n\times n}$ is defined

$$A_{ij} = [Q^T K]_{ij}$$

where $1 \leq i, j \leq n$. In this case, the attention score α_{ij} is defined by $(2.3.1)$. Then the attention mechanism is this case is to update v_i by

$$\sum_{j=1}^{n} \alpha_{ij} v_j$$

for $1 \leq i \leq n$.

In [91], the alignment model of a shared attentional mechanism is defined by a shallow neural network $f_\theta : \mathbb{R}^n \to \mathbb{R}^n$ using Leaky ReLU so that

$$A_{ij} = f_\theta(h_i, h_j) = \text{Leaky}(a^T [Wh_i \| Wh_j]) \tag{2.3.2}$$

where $W \in \mathbb{R}^{F\prime\times\prime}$ is a shared weight, $a \in \mathbb{R}^{2F'}$ is a weight vector and $[Wh_i \| Wh_j] \in \mathbb{R}^{2F'}$ be the concatenation operation between two vectors Wh_i and Wh_j of $\mathbb{R}^{F'}$.

Hence, the process of graph attention network between two attention layers can be divided into serval steps:

1. Update all feature vector $\{h_i\}$ in $\mathbb{R}^F$ by multiplying by a common weight matrix W of shape $F' \times F$. In other words, we have $\{Wh_i\}$ in $\mathbb{R}^{F'}$.

2. Compute the alignment or attention coefficient A_{ij} between i and j. One can consider A as a shallow net defined as in (2.3.2).

3. Compute the attention coefficient α_{ij} between i and j by normalizing A_{ij} across all choices of j using softmax function:

$$\alpha_{ij} = \frac{e^{A_{ij}}}{\sum\limits_{k\in\overline{\mathcal{N}_i}} e^{A_{ik}}} \tag{2.3.3}$$

   Then $\alpha_{ij} \in [0, 1]$ and $\sum\limits_{j\in\overline{\mathcal{N}_i}} \alpha_{ij} = 1$ for each i.

4. Update Wh_i to h'_i by consider a convex combination of all its neighborhoods and h_i itself

DOI:10.6814/NCCU202100296

with coefficients α_{ij} for all $j \in \mathcal{N}_i \cup \{i\}$. That is,

$$h_i' = \sigma\Big(\sum_{j \in \mathcal{N}_i \cup \{i\}} \alpha_{ij} W h_j \Big) = \sigma\Big(\sum_{j=1}^{n} \alpha_{ij} (I_n + A)_{ij} W h_j \Big) \tag{2.3.4}$$

where σ is an activation function. Note that this step is to update Wh_i by average of all its neighborhoods.

With $\{h_1, \cdots, h_n\}$ as input, a graph attention network geneates $\{h_1', \cdots, h_n'\}$ by weight matrix W and the graph structure of G in each graph attention layer.

Therefore, a graph neural network generates a series of feature vectors for each vertex i. See Figure 2.5 for an example.

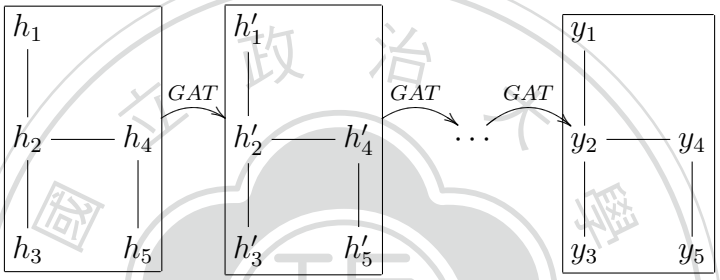


Figure 2.5: Diagram of graph attention neural network

We give an example that illustrates how an attention layer of a graph attention network works in Figure 2.6.

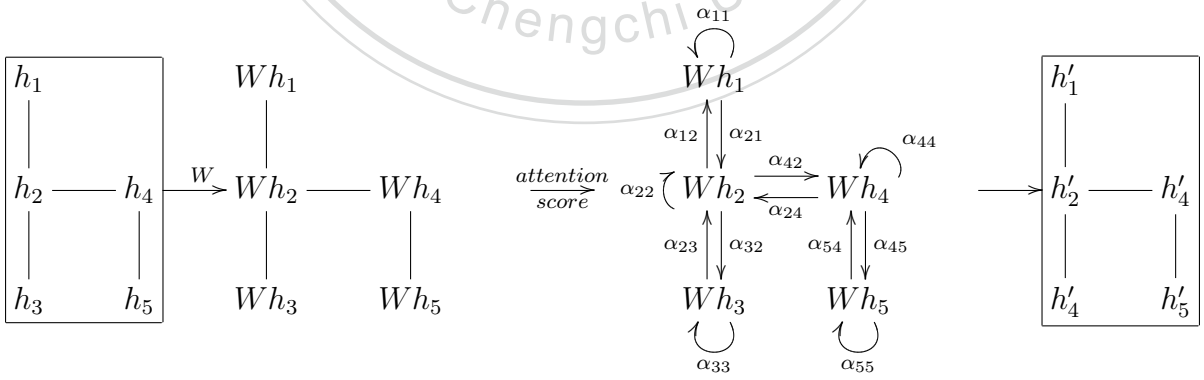


Figure 2.6: Mechanism of graph attention

There are various datasets which could be identified with a graph structure. For example, a citation graph is a directed graph in which each vertex represents a scientific paper and each

DOI:10.6814/NCCU202100296

edge represents a citation from a peer to another. The feature vector of each vertex is formed by a bag-of-words representation of certain paper. For more detail and survey about graph neural networks, please refer [100, 98].

Later in chapter 7, we will see a natural dataset which consist of graph Laplacian matrices of all edge-weighted connected graphs.

DOI:10.6814/NCCU202100296

Chapter 3

Neural Network Approach for Pricing American Volatility Options

In this chapter, we will see how a pricing formula of American power volatility option is derived by solving a moving boundary problem. However, such pricing formula contains a moving boundary term which can only be solved by solving a nonlinear algebraic equation. Therefore, a neural network approach is considered in the end of this chapter. The comparison results demonstrates that the neural network provides an accurate approach to approximate solution for the free boundary problem.

This chapter is partially based on the joint work [63] with my advisors.

3.1 Introduction

In this chapter, we study the properties of the parabolic free-boundary problem arising from pricing of American volatility options in mean-reverting volatility processes. When the volatility index follows the mean-reverting square root process (MRSRP), a closed-form pricing formula for the perpetual American power volatility option can be derived. Moreover, a neural network approach is extended to find an approximate solution of the free boundary problem arising from pricing the perpetual American option.

DOI:10.6814/NCCU202100296

3.2 Problem Definition

3.2.1 The Probability Density Function and Expectation

In the case of mean-reverting square root process (MRSRP), the index process under the Martingale measure Q is presented as

$$dx_t = \beta(m - x_t)dt + \sigma\sqrt{x_t}dw_t$$

with β, m and σ representing the speed of mean-reversion, the long-run mean, and the volatility of the volatility index, respectively [23, 40].

Here, x denotes the index of volatility, t denotes the time-to-maturity and dw_t denotes an increment in the Wiener process under the Martingale probability measure Q. The probability density function of x at the future time T under the current time t is given as (see Cox et al. [17])

$$p(x_T, x_t|\beta, m, \sigma) = ce^{-u-v}(\frac{v}{u})^{\frac{q}{2}}I_q(2\sqrt{uv}), \tag{3.2.1}$$

where $c = \frac{2\beta}{\sigma^2(1-e^{-\beta(T-t)})}$, $u = cx_te^{-\beta(T-t)}$, $v = cx_T$, $q = \frac{2\beta m}{\sigma^2} - 1$ and I_q is the modified Bessel function of the first kind of order q.

In the case of mean-reverting 3/2 volatility process, the index process under the Martingale measure Q is given as follows:

$$dx_t = (\alpha x_t - \beta x_t^2)dt + kx_t^{\frac{3}{2}}dw_t, \tag{3.2.2}$$

where $\alpha > 0$, $\beta > 0$ and $k \neq 0$ are constants. This model has a nonlinear drift so that it exhibits substantial nonlinear mean-reverting behavior when the volatility is above its long-run mean. Hence, after a large volatility spike, the volatility can potentially quickly decrease, while after a low volatility period it can be slow to increase. Applying change of variables $y_t = 1/x_t$, y follows the following MRSRP

$$dy_t = ((k^2 + \beta) - \alpha y_t)dt - k\sqrt{y_t}dw_t.$$

DOI:10.6814/NCCU202100296

Based on (3.2.1), the probability density function of y is then presented as follows:

$$p(y_T, y_t|\alpha, \frac{k^2+\beta}{\alpha}, -k) = ce^{-u-z}(\frac{z}{u})^{\frac{q}{2}} I_q(2\sqrt{uz}), \tag{3.2.3}$$

where $c = \frac{2\alpha}{k^2(1-e^{-\alpha(T-t)})}$, $u = cy_t e^{-\alpha(T-t)}$, $z = cy_T$ and $q = 1 + \frac{2\beta}{k^2}$ (see Goard and Mazur [37]).

Since the probability density function of the MRSRP and the mean-reverting 3/2 processes are given in (3.2.1) and (3.2.3), respectively, the value of a European option can be obtained as

$$V(x,t) = e^{-r(T-t)}\mathbb{E}^Q[\psi(x_T)|x_t = x],$$

where $\psi(x)$ is the payoff function of the European volatility option and $\mathbb{E}^Q$ denotes the expectation under the martingale measure Q.

3.2.2 The Solution of Partial Differential Equations

Except considering the probability density function to find the expectation for the price, the pricing formula of the European volatility option is also the solution of partial differential equations. When the volatility index follows the MRSRP, the pricing equation of the volatility option $V(x,t)$ is presented as

$$(\mathcal{L}_0^M - \frac{\partial}{\partial t})V = 0,\ 0 \le x < \infty,\ \ 0 < t < \infty, \tag{3.2.4}$$

where $\mathcal{L}_0^M$ is defined as

$$\mathcal{L}_0^M \equiv \frac{1}{2}\sigma^2 x\frac{\partial^2}{\partial x^2} + \beta(m-x)\frac{\partial}{\partial x} - r.$$

The fundamental solution of (3.2.4) is given by Feller [32]. When the volatility index follows the mean-reverting 3/2 volatility model, the value $V(x,t)$ of the volatility option can also be obtained by solving the following parabolic equation

$$(\mathcal{L}_0^Q - \frac{\partial}{\partial t})V = 0,\ 0 \le x < \infty,\ \ 0 < t < \infty,$$

DOI:10.6814/NCCU202100296

where the operator $\mathcal{L}_0$ is given in the form

$$\mathcal{L}_0^Q \equiv \frac{1}{2}k^2x^3\frac{\partial^2}{\partial x^2} + (\alpha x - \beta x^2)\frac{\partial}{\partial x} - r. \tag{3.2.5}$$

The coefficients are all continuously differentiable and $\frac{1}{2}k^2x^3 > 0$, for $0 < x < \infty$, $k \neq 0$ and $r > 0$. By setting $x = 1/y$, $\mathcal{L}_0^Q V(x)$ can be changed to $\mathcal{L}_0^M V(y)$.

The closed-form expression for the value of a European volatility call option was proposed by Grunbichler and Longstaff [40], who found that the price of a volatility call option can be below its intrinsic value and that the traditional put-call parity relation does not hold for these options. This is because the volatility is not the price of a traded asset. However, the value of the American style volatility call option, unlike the European option, is bounded below by its early exercise payoff. Evidently, the lower bound is a consequence of the possibility of immediate exercise. Moreover, the European option still has value as the volatility decreases to zero in the MRSRP case. Detemple and Osakwe [23] said that the reason for this difference is the multiplicative impact of the uncertainty of future volatility. They also showed that the price of the American style volatility call ($\psi(x) = \max\{x - K, 0\}$) is an increasing function of the time-to-maturity.

3.2.3 The Free Boundary Problem for Pricing an American Volatility Option

For the American-style option, an entirely satisfactory analytic solution has not been found for the MRSRP model and the mean-reverting 3/2-volatility model, even though several researchers have concentrated on finding the properties of the value as well as the early exercise boundary for American options. Liu [62] proposed the properties of the price and the early exercise boundary for the American volatility put option ($\psi(x) = \max\{K - x, 0\}$) when the volatility index satisfies the mean-reverting 3/2 volatility process.

In this chapter, we consider the pricing problem for the American volatility call with the

DOI:10.6814/NCCU202100296

payoff function $\psi(x) = \max\{x^n - K, 0\}$, $n \in \mathbb{Z}$. Apply $\mathcal{L}_0^M$ to $\psi(x)$ yields that

$$\mathcal{L}_0^M\psi(x) = \begin{cases} -(\beta n + r)x^n + (\frac{1}{2}\sigma^2(n-1) + \beta m)nx^{n-1} + rK, & \text{if } x^n > K, \\ 0, & \text{if } x^n < K. \end{cases}$$

Since $rK > 0$ and $\mathcal{L}_0^M\psi(\xi) \to -\infty$ as $\xi \to \infty$ if $\beta n + r > 0$. This implies that there exists $d > 0$ such that

$$\mathcal{L}_0^M\psi(x) \begin{cases} > 0 \text{ for } 0 < x < d, \\ < 0 \text{ for } d < x < \infty, \end{cases} \tag{3.2.6}$$

Precisely, this chapter examines the following one-dimensional free boundary problem for linear parabolic equations arising from the problem of valuing an American-style volatility option in the models of MRSRP. Define $\mathcal{L} = \mathcal{L}_0^M - \frac{\partial}{\partial t}$.

Let $u(x,t)$ and $s(t)$ be the price and the early exercise price of the American volatility power call. For the case of $n > 0$, we consider the following free boundary problem.

Problem (P)

$$\mathcal{L}u = 0, \qquad 0 < x < s(t),\ 0 < t < \infty, \tag{3.2.7}$$

$$u(x,t) > x^n - K, \qquad 0 < x < s(t),\ 0 < t < \infty,, \tag{3.2.8}$$

$$u(x,0) = x^n - K, \qquad 0 \le x \le s(0), \tag{3.2.9}$$

$$u(s(t),t) = s^n(t) - K, \qquad 0 \le t < \infty, \tag{3.2.10}$$

$$\frac{\partial u}{\partial x}(s(t),t) = \psi'(s(t)), \qquad 0 \le t < \infty, \tag{3.2.11}$$

Since $u(x,t)$ denotes the price of an American volatility power call, the condition $u(0,t) < \infty$ is added in the model. These additional condition will be used in finding the pricing formula of the corresponding perpetual American option.

For the case of $n < 0$, we have $x^n > K$ if $x < \frac{1}{\sqrt[n]{K}}$. Hence, the value of the American volatility option satisfies the following free boundary problem.

DOI:10.6814/NCCU202100296

Problem (Q)

$$\mathcal{L}u = 0, \qquad s(t) < x < \infty,\ 0 < t < \infty, \tag{3.2.12}$$

$$u(x,t) > x^n - K, \qquad s(t) < x < \infty,\ 0 < t < \infty,, \tag{3.2.13}$$

$$u(x,0) = x^n - K, \qquad s(0) \le x \le \infty, \tag{3.2.14}$$

$$u(s(t),t) = s^n(t) - K, \qquad 0 \le t < \infty, \tag{3.2.15}$$

$$\frac{\partial u}{\partial x}(s(t),t) = \psi'(s(t)), \qquad 0 \le t < \infty, \tag{3.2.16}$$

In the mean reverting 3/2 volatility process, the pricing problem for the American volatility option can be considered by changing the variable $x = 1/y$. It would be interesting to consider the properties of the value as well as the early exercise boundary of American volatility power options while the properties have not been mentioned in the case of the MRSRP and the mean-reverting 3/2 volatility process.

In the cases of the MRSRP and the mean-reverting 3/2 volatility process, we derive a closed-pricing formula for the perpetual American volatility power option, where the early exercise price can be obtained iteratively. Moreover, we consider neural network (NN) approach to the solution of the free boundary differential equation arising from pricing a perpetual American volatility option under the MRSRP. The numerical results show that the ANN approach is an accurate approach for pricing the American volatility option in the case of MRSRP. This NN approach can also be applied to approximate the pricing formula of the perpetual American option under the different process. In future studies, our results can be applied to consider the properties of other American-style derivatives with the payoff function satisfying (3.2.6) in the cases of the MRSRP and the mean-reverting 3/2 volatility process.

3.3 Properties of the Solution

Let $\mathcal{T}$ denote the set of all stopping time τ for the process. The value of an American-style option is obtained by evaluating the following optimization problem

$$\hat{u}(x,t) = \sup_{\tau \in \mathcal{T}} \mathbb{E}^Q[\psi(x_\tau)|x(t) = x]. \tag{3.3.1}$$

DOI:10.6814/NCCU202100296

Kotlow [53] showed that the solution (s, u) to Problem (P) resolves the optimization problem (3.3.1) by setting

$$\hat{u}(x,t) = \begin{cases} u(x,t) & \text{if } (x,t) \in C, \\ \psi(x) & \text{if } (x,t) \in \bar{Q} - C, \end{cases}$$

where $\bar{Q} = (0,\infty) \times (0,\infty)$ and $C = \{(x,t)|0 < x < s(t), 0 < t < \infty\}$.

Let $\{s, u\}$ be the solution to Problem (P) and denote C, namely the continuation region, as

$$C = \{(x,t)|0 < x < s(t),\ 0 < t < \infty\}. \tag{3.3.2}$$

Applying results of Kotlow [53] directly to (P), we obtained the following theorems.

Theorem 3.3.1. *Let $\{s, u\}$ be a solution of (P). They have the following properties:*

(a) $u_t > 0$ *in* C.

(b) $s(0) = d$ *and* $s(t) > d$ *for* $0 < t < \infty$.

(c) $s(t)$ *is a non-decreasing function.*

(d) There exists a $s^\infty \in (d, \infty)$ *such that* $s(t) \to s^\infty$ *uniformly as* $t \to \infty$ *if* $\limsup_{\xi \to \infty} [\mathcal{L}_0^M \psi(\xi)] < 0$ *and* $\beta n + r > 0$.

In the case of the American put option, Liu [62] provided the properties for the price as well as the early exercise boundary under the mean-reverting 3/2 volatility model. When the payoff function satisfies (3.2.6), we obtained the following theorem by modifying the proof of Theorem 2.3 in [62]. The following theorem includes the call option and the power call option in the MRSRP or the mean-reverting 3/2 volatility models.

Theorem 3.3.2. *Let $\{s, u\}$ be a solution of (P). Then*

(a) $s(t)$ *is a strictly increasing function.*

(b) $u_x(x,t) > 0$ *for* $(x,t) \in C$.

(c) When $\beta > 0$, $u_x(x,t) < \psi'(x)$ *for* $(x,t) \in C_d$, *where* $C_d = \{(x,t) \in C|x > d)\}$.

DOI:10.6814/NCCU202100296

According to Theorem 3.3.1 and Theorem 3.3.2, we propose properties for the value and the early exercise boundary of an American volatility power option in the MRSRP (Theorem 3.3.3) and the mean-reverting 3/2 volatility process (Theorem 3.3.5). The similar results for the American volatility call can also be founded in Detemple and Kitapbayev [22].

Theorem 3.3.3. *Let $u(x,t)$ and $s(t)$ be the value and the early exercise boundary of an American volatility power option in the MRSRP. When $\beta n + r > 0$, we have the following properties.*

(a) The value $u(x,t)$ increases with an increase in both the time-to-maturity.

(b) The value $u(x,t)$ increases (decreases, respectively) with an increase in the volatility index x for $n > 0$ (for $n < 0$, respectively).

(c) The early exercise boundary $s(t)$ strictly increases (decreases, respectively) with an increase in the time-to-maturity for $n > 0$ (for $n < 0$, respectively).

(d) The early exercise boundary $s(t)$ is bounded by d and s^{∞}, where s^{∞} is the exercise boundary of its corresponding perpetual American option.

(e) The early exercise boundary starts at d, that is $s(0) = d$.

Proof. The coefficients of (3.2.4) are all continuously differentiable and $\frac{1}{2}\sigma^2 x > 0$ for $0 < x < \infty$ and $r > 0$. To show that the value and the early exercise boundary of an American volatility power option satisfy Theorem 3.3.1 and Theorem 3.3.2, it suffices to show that there exists a d in $\mathbb{R}$ such that $\mathcal{L}_0^M \psi(x)$ satisfies

$$\mathcal{L}_0^M \psi(x) \begin{cases} > 0 \text{ for } 0 < x < d, \\ < 0 \text{ for } d < x < \infty, \end{cases}$$

for some $d > 0$.

Applying $\mathcal{L}_0^M$ to $\psi(x) = \max\{x^n - K, 0\}$ for a volatility power option yields that

$$\mathcal{L}_0^M \psi(x) = \begin{cases} -(\beta n + r)x^n + (\frac{1}{2}\sigma^2(n-1) + \beta m)nx^{n-1} + rK, & \text{if } x^n > K, \\ 0, & \text{if } x^n < K. \end{cases}$$

DOI:10.6814/NCCU202100296

Let $f(x) = -(\beta n + r)x^n + (\frac{1}{2}\sigma^2(n-1) + \beta m)nx^{n-1} + rK$. We have $f(0) = rK > 0$ and $\lim_{x\to\infty} f(x) = -\infty$ since $-(\beta n + r) < 0$. Since f is a continuous function on $\mathbb{R}$, f has at least one positive root. Moreover, we have $f'(x) = x^{n-2}\big[-n(\beta n + r)x + (\frac{1}{2}\sigma^2(n-1) + \beta m)n(n-1)\big]$. This implies that $f(x)$ increases as $x < \frac{(\frac{1}{2}\sigma^2(n-1)+\beta m)(n-1))}{\beta n+r}$ and decreases as $x > \frac{(\frac{1}{2}\sigma^2(n-1)+\beta m)(n-1))}{\beta n+r}$. By the continuity of f, we obtained that f has exactly one positive root, say d'. Then we can define $d = \max\{K^{1/n}, d'\}$ for $n > 0$ and $d = \min\{K^{1/n}, d'\}$ for $n < 0$. □

Remark 3.3.4. *For the American volatility call option with $\psi(x) = \max\{x - K, d\}$, we have $d = \max\{K, \frac{r}{r+\beta}K\} = K$ in the case of the MRSRP. This is because $\beta > 0$ and $\mathcal{L}_0^M\psi(x) = -(\beta + r)x + rK$ for $x > K$.*

Now, we consider properties of an American volatility power option in the mean-reverting 3/2 volatility process.

Theorem 3.3.5. *When $\frac{1}{2}k^2(n-1) < \beta$ and $u > 0$, the value $u(x,t)$ and the early exercise boundary $s(t)$ of an American power option have the same properties of (a) to (d) in Theorem 3.3.3 with the volatility following the mean-reverting 3/2 volatility process.*

Proof. Applying $\mathcal{L}_0^Q$ to $\psi(x) = \max\{x^n - K, 0\}$ for a volatility power option yields that

$$\mathcal{L}_0^Q\psi(x) = \begin{cases} (\frac{1}{2}k^2n(n-1) - \beta n)x^{n+1} + (\alpha n - r)nx^n + rK, & \text{if } x > K, \\ 0, & \text{if } x < K. \end{cases}$$

Let $f(x) = nAx^{n+1} + (\alpha n - r)nx^n + rK$, where $A = \frac{1}{2}k^2(n-1) - \beta$. Then $f'(x) = nx^n[(n+1)Ax + (\alpha n - r)]$ and $f(0) = rK > 0$. Since $A < 0$, we have $\lim_{x\to\infty} f(x) = -\infty$ and $f'(x) > 0$ if $x < \frac{r-\alpha n}{(n+1)A}$ and $f'(x) < 0$ if $x > \frac{r-\alpha n}{(n+1)A}$. Hence $f(x)$ increases with $x < \frac{r-\alpha n}{(n+1)A}$ and deceases with $x > \frac{r-\alpha n}{(n+1)A}$. Therefore, we obtained that $f(x)$ has exactly one positive root, say d'. Then we can define $d = \max\{K^{1/n}, d'\}$ for $n > 0$ and $d = \min\{K^{1/n}, d'\}$ for $n < 0$. □

Remark 3.3.6. *According to Theorem 3.3.5, the value of $s(0) = d$ for the American call option with $\psi(x) = \max\{x - K, 0\}$ is obtained as $d = \max\{K, d'\}$, where $d' = \frac{(\alpha-r)+\sqrt{(\alpha-r)^2+4rK\beta}}{2\beta} > 0$.*

DOI:10.6814/NCCU202100296

3.4 Asymptotic Behavior of Exercise Boundary Infinitely Far from Expiry

Since $s(t)$ is a strictly increasing function of the time-to-maturity for the American volatility power option, the lower boundary for the optimal exercise boundary $s(t)$ for $t > 0$ is obtained by $\lim_{t\to 0^+} s(t) = d$ in the MRSRP and the mean-reverting 3/2 volatility process. It would be interesting to explore whether $s(t)$ is bounded or not as $t \to \infty$. At the same time, the pricing formula for the perpetual American volatility power option is obtained in the MRSRP and the mean-reverting 3/2 volatility process.

Before solving the ordinary differential equation arising from pricing the perpetual American volatility option, we first introduce the confluent hypergeometric functions of which the integral representations are given as

$$\Phi(a, b, x) = \frac{\Gamma(b)}{\Gamma(b-a)\Gamma(a)} \int_0^1 e^{xt} t^{a-1} (1-t)^{b-a-1} dt.$$

In the following theorem, we will provide a pricing formula for the perpetual American volatility power option.

Theorem 3.4.1. *Let (v, s) be the value and the early exercise boundary of a perpetual American volatility power option in MRSRP. Assume $\frac{2\beta m}{\sigma^2} > 1$ for the Feller condition. If $n > 0$, then (v, s) solves the following free boundary problem*

$$\begin{aligned} &\mathcal{L}_0^M v(x) = 0, && 0 < x < s, \\ &v(s) = s^n - K, && v'(s) = n s^{n-1}, \end{aligned} \tag{3.4.1}$$

where $\mathcal{L}_0^M$ is defined in (3.2.4). The solution is obtained in the form.

$$v(x) = C_1 \Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2} x), \quad 0 < x < s, \tag{3.4.2}$$

where $C_1 = \frac{s^n - K}{\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2} s)}$ and s is a root of the following equation

$$\frac{\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2} s)}{\frac{\sigma^2 r}{2\beta^2 m} \Phi(\frac{r}{\beta} + 1, \frac{2\beta m}{\sigma^2} + 1; \frac{2\beta}{\sigma^2} s)} = \frac{s^n - K}{n s^{n-1}}. \tag{3.4.3}$$

DOI:10.6814/NCCU202100296

which can be solved iteratively for s.

If $n < 0$, then (v, s) solves the following free boundary problem

$$\begin{aligned} &\mathcal{L}_0^M v(x) = 0, \qquad && s < x < \infty, \\ &v(s) = s^n - K, \qquad && v'(s) = ns^{n-1}, \qquad \lim_{x\to\infty} v(x) = 0, \end{aligned} \tag{3.4.4}$$

where $\mathcal{L}_0^M$ is defined in (3.2.4). The solution is obtained in the form.

$$v(x) = (s^n - K)(\frac{x}{s})^{1-\frac{2\beta m}{\sigma^2}} \frac{\Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x)}{\Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s)}, \quad s < x < \infty. \tag{3.4.5}$$

Here s is a root of the following equation

$$\frac{s^{1-\frac{2\beta m}{\sigma^2}} \Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s)}{\frac{d}{dx}\left[x^{1-\frac{2\beta m}{\sigma^2}} \Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x)\right]|_{x=s}} = \frac{s^n - K}{ns^{n-1}}. \tag{3.4.6}$$

which can be solved iteratively for s.

Proof. By letting $y = \frac{2\beta}{\sigma^2}x$, $\mathcal{L}_0^M v(x) = 0$ is changed to

$$y\frac{d^2v}{dy^2} + (\frac{2\beta m}{\sigma^2} - y)\frac{dv}{dy} - \frac{r}{\beta}v = 0, \quad 0 < y < \frac{2\beta}{\sigma^2}s, \tag{3.4.7}$$

which is regarded as a Kummer's equation.

The solutions of Kummer's equation (3.4.7) are expressed through the confluent hypergeometric function $\Phi(\alpha, \nu, x)$. Precisely, the general solutions are written in the form

$$v(y) = C_1\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; y) + C_2 y^{1-\frac{2\beta m}{\sigma^2}} \Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; y)$$

where C_1 and C_2 are arbitrary constants.

Displaying the solution in terms of x, the equation is rewritten in the form

$$v(x) = C_1\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x) + C_2(\frac{2\beta}{\sigma^2}x)^{1-\frac{2\beta m}{\sigma^2}} \Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x).$$

We first consider the case of $n > 0$. The value $v(x)$ of an American volatility power option is finite in $[0, s]$, that is $v(s) < \infty$ for all $x \in [0, s]$. Since $\beta m > \frac{1}{2}\sigma^2$ and $\Phi(\alpha, \beta; x) \neq 0$

DOI:10.6814/NCCU202100296

as $x \to 0$, we have $(\frac{2\beta}{\sigma^2}x)^{1-\frac{2\beta m}{\sigma^2}} \to \infty$ as $x \to 0$ and $C_2 = 0$. Consequently, the value of the perpetual American volatility option equals to

$$v(x) = C_1\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x). \tag{3.4.8}$$

To determine the free boundary s and the coefficient C_2, we substitute $v(s) = s^n - K$ and $v'(s) = ns^{n-1}$ into (3.4.8) and obtain that

$$C_1\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s) = s^n - K$$

and

$$C_1\frac{d}{dx}\left[\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x)\right]|_{x=s} = ns^{n-1}.$$

Moreover, we have $\frac{d}{dx}\left[\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x)\right]|_{x=s} = \frac{\sigma^2 r}{2\beta^2 m}\Phi(\frac{r}{\beta}+1, \frac{2\beta m}{\sigma^2}+1; \frac{2\beta}{\sigma^2}s)$.

Hence, we find that the free boundary satisfies the following nonlinear algebraic equation

$$\frac{\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s)}{\frac{\sigma^2 r}{2\beta^2 m}\Phi(\frac{r}{\beta}+1, \frac{2\beta m}{\sigma^2}+1; \frac{2\beta}{\sigma^2}s)} = \frac{s^n - K}{ns^{n-1}}.$$

When the free boundary s is obtained by the solving the above equation numerically, the coefficient C_1 is expressed as

$$C_1 = \frac{s^n - K}{\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s)}.$$

We then consider the case of $n < 0$. Since $\lim_{x\to\infty} v(x) = 0$, we have $\frac{2\beta}{\sigma^2}x^{1-\frac{2\beta m}{\sigma^2}} \to 0$ as $x \to \infty$ and

$$C_1\Phi(\frac{r}{\beta}, \frac{2\beta m}{\sigma^2}; 0) \to 0 \text{ as } x \to \infty$$

which means $C_1 = 0$. Consequently, we have

$$v(x) = C_2(\frac{2\beta}{\sigma^2}x)^{1-\frac{2\beta m}{\sigma^2}}\Phi(\frac{r}{\beta} + 1 - \frac{2\beta m}{\sigma^2}, 2 - \frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x). \tag{3.4.9}$$

To determine the free boundary s and the coefficient C_2, we substitute $v(s) = s^n - K$ and

DOI:10.6814/NCCU202100296

$v'(s) = ns^{n-1}$ into (3.4.9) and obtain that

$$C_2(\frac{2\beta}{\sigma^2}s)^{1-\frac{2\beta m}{\sigma^2}}\Phi(\frac{r}{\beta}+1-\frac{2\beta m}{\sigma^2}, 2-\frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s) = s^n - K$$

and

$$C_2\frac{d}{dx}\left[(\frac{2\beta}{\sigma^2}x)^{1-\frac{2\beta m}{\sigma^2}}\Phi(\frac{r}{\beta}+1-\frac{2\beta m}{\sigma^2}, 2-\frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x)\right]|_{x=s} = ns^{n-1}.$$

Hence, we find that the free boundary satisfies the following nonlinear algebra equation

$$\frac{(\frac{2\beta}{\sigma^2}s)^{1-\frac{2\beta m}{\sigma^2}}\Phi(\frac{r}{\beta}+1-\frac{2\beta m}{\sigma^2}, 2-\frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s)}{\frac{d}{dx}\left[(\frac{2\beta}{\sigma^2}x)^{1-\frac{2\beta m}{\sigma^2}}\Phi(\frac{r}{\beta}+1-\frac{2\beta m}{\sigma^2}, 2-\frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}x)\right]|_{x=s}} = \frac{s^n - K}{ns^{n-1}}.$$

When the free boundary s is obtained by the solving the above equation numerically, the coefficient C_2 is expressed as

$$C_2 = \frac{s^n - K}{(\frac{2\beta}{\sigma^2}s)^{1-\beta m}\Phi(\frac{r}{\beta}+1-\frac{2\beta m}{\sigma^2}, 2-\frac{2\beta m}{\sigma^2}; \frac{2\beta}{\sigma^2}s)}.$$

□

Remark 3.4.2. *When (v, s) are the value and the early exercise boundary of a perpetual American volatility option in mean reverting 3/2 volatility process, (v, s) is the solution of the following free boundary problem*

$$\begin{aligned} &\mathcal{L}_0^Q v(x) = 0, && 0 < x < s, \\ &v(s) = s - K, && v'(s) = ns^{n-1}, && v(0) = 0 \end{aligned} \tag{3.4.10}$$

where $\mathcal{L}_0^Q$ is defined in (3.2.5). When taking $x = 1/y$, this problem becomes to price a perpetual American volatility option in the MRSRP with $\psi(x) = (1/x - K, 0)^+$. Hence, the price of a perpetual American option becomes to satisfy the following free boundary problem

$$\begin{aligned} &\mathcal{L}_0^M u(y) = 0, && 1/s < y < \infty, \\ &u(s) = 1/s - K, && u'(s) = -1/s^2, && u(y) \to 0 \text{ as } y \to \infty. \end{aligned}$$

DOI:10.6814/NCCU202100296

The solution of this equation can be obtained by changing it to a Kummer's equation. In the mean-reverting 3/2 volatility process, a closed pricing formula for the perpetual American volatility put option is proposed by Liu [62].

This chapter provides a formula for the perpetual American volatility power call in the MRSRP model. Liu [62] provided a formula for the perpetual American volatility put in the mean-reverting 3/2 volatility model. These two papers considered distinct options (call and put) in the different processes (the MRSRP and the mean-reverting 3/2 volatility process). Using the change of variables, the differential equations in both two papers are changed to the same Kummer's differential equation with the different boundary conditions. The general solution is expresses by the combination of the confluent hypergeometric function of the first and second kinds.

For the volatility call, we have a finite initial condition and an upper free boundary; For the volatility put, we have a lower free boundary and assume that the put value tends to zero as the volatility tends to infinite. The different conditions for both put and call induces different pricing formulas. For the volatility call, we eliminate the second independent solution by the finite initial condition. For the volatility put, we eliminate the first independent solution since the put value tends to zero as the volatility tends to infinite. Moreover, the advantage of this chapter is that we add a power to the payoff function, $\psi(x) = \max\{x^n - K, 0\}$. When setting the volatility x to $1/y$ (i.e., choosing $n = -1$), the MRSRP model can be changed to the mean-reverting 3/2 volatility model. In this case, the boundary conditions are changed to a lower free boundary and zero value at the infinite.

3.5 Neural Network Approach

In this section, we consider neural network (NN) approach to the solution of the free boundary differential equation arising from pricing a perpetual American volatility option under the MRSRP.

Alternatively, this NN approach can also be applied to approximate the pricing formula of the perpetual American option under the different process. The definition of a neural network can be recall from Definition 2.1.1.

Now, we go back to the NN approach to our MRSRP problem. Suppose that $\frac{2\beta m}{\sigma^2} > 1$.

DOI:10.6814/NCCU202100296

Let $\{u, s\}$ be a solution the free boundary problem (3.4.4). Transferring $y = \frac{x}{s}$ and defining $w(y) = u(x)$, the free boundary problem can be reformulated as the following boundary value problem

$$\frac{1}{2}\sigma^2 y w'' + \beta(m - sy)w' - rsw = 0, \;\; 0 < y < 1, \tag{3.5.1}$$

with $w(1) = s^n - K$ and $w'(1) = ns^{n-1}$. Substituting $s = (w(1) - K)^{\frac{1}{n}}$ into (3.5.1) yields

$$\frac{1}{2}\sigma^2 y w'' + \beta(m - (w(1) + K)^{\frac{1}{n}} y)w' - r(w(1) + K)^{\frac{1}{n}} w = 0, \;\; 0 < y < 1,$$

with $w'(1) = n(w(1) + K)^{\frac{n-1}{n}}$. We consider the following the trial solution

$$w_A(y) = f_\theta(y),$$

where $f_\theta : \mathbb{R}^2 \to \mathbb{R}$ is a shallow net defined by θ. Then, we obtain $s = (f_\theta(y) + K)^{\frac{1}{n}}$ and $w'_A(1) = n(f_\theta(1) + K)^{\frac{n-1}{n}}$. Therefore, we turn into solve the following equation

$$\frac{1}{2}\sigma^2 y w''_A(y) + \beta(m - (f_\theta(1) + 1 + K)^{\frac{1}{n}} y)w'_A(y) - r(f_\theta(1) + 1 + K)^{\frac{1}{n}} w_A(y) = 0$$

with $w'_A(1) = n(f_\theta(1) + K)^{\frac{n-1}{n}}$.

To find an optimal f_θ to solve the above equations, we consider the following minimization problem.

$$\min_\theta L(\theta),$$

where

$$L(\theta) = \int_0^1 \left[\frac{1}{2}\sigma^2 y w''_A(y) + \beta(m - sy)w'_A(y) - rsw_A(y)\right]^2 + \left[w'_A(1) - ns^{n-1}\right]^2 dy$$

is the L^2-loss between neural network f_θ and the solution of equation and the boundary condition, and $s = (f_\theta(1) + K)^{\frac{1}{n}}$.

In addition, consider a neural network of the form:

$$f_\theta(y) = \sum_{k=1}^{h_d} w_{k,i}^{(d)} \sigma(S_{d,k}).$$

DOI:10.6814/NCCU202100296

Here, the output neuron i in layer j $s_{j,i}$ is defined by

$$\begin{aligned} s_{1,i} = \quad & w_{1,i}^{(0)} y + b_i^{(1)}, \\ s_{j,i} = \quad & \sum_{k=1}^{h_{j-1}} w_{k,i}^{(j-1)} \sigma(s_{j-1,k}) + b_i^{(j)}, \ \ j = 1, 2, \cdots, d, \end{aligned}$$

where $w_{k,i}^{(j)}$ is the weight from neuron k in layer $j-1$ to neuron i in layer j for the network $f_\theta(y)$, and h_j is the number of neurons in layer j and $b_i^{(j)}$ is the bias of neuron i in layer j.

3.5.1 Comparisons

From Theorem 3.4.1, an analytical solution of the free boundary problem (P), where $s = s(t)$ can be only solved numerically. In previous section, we developed a new numerical method to approximate the differential equation with moving boundary by extending the neural network approach. In this section, we will demonstrate the comparison results between the analytical solution and the numerical solution using the neural network approach.

Goard and Mazur [37] used the data of the VIX index value between years of 1990 and 2009 to estimate the parameters the continuous-time model. In the empirical results, the parameters are estimated as $\beta = 3.1637$ and $\beta m = 0.6154$ (see Table 5.1 in [37]) for the MRSRP model. Moreover, the risk-free interest rate and strike are given as $r = 0.05$ and $K = 0.5$, respectively. To satisfy $\frac{2\beta m}{\sigma^2} > 1$, we assume $\sigma^2 = 1$.

To compare the analytical solution and the numerical solution, programs are coded by Python [88] on Google Colab environment. The s of solution $\{u, s\}$ is solved by using the "fsolve" instruction, where the starting estimate for the roots is given as $K + 1$. The network f_θ is constructed by 1 input layer, 1 output layer and 1 hidden layer with 10 neurons in the hidden layer. That is,

$$f_\theta(y) = \sum_{k=1}^{10} w_i^{(1)} \sigma(\sum_{i=1}^{10} w_i^{(0)} y + b_i)$$

The structure of such neural network is shown in Figure 3.1, note that bias b_i are ignored from the figure.

DOI:10.6814/NCCU202100296

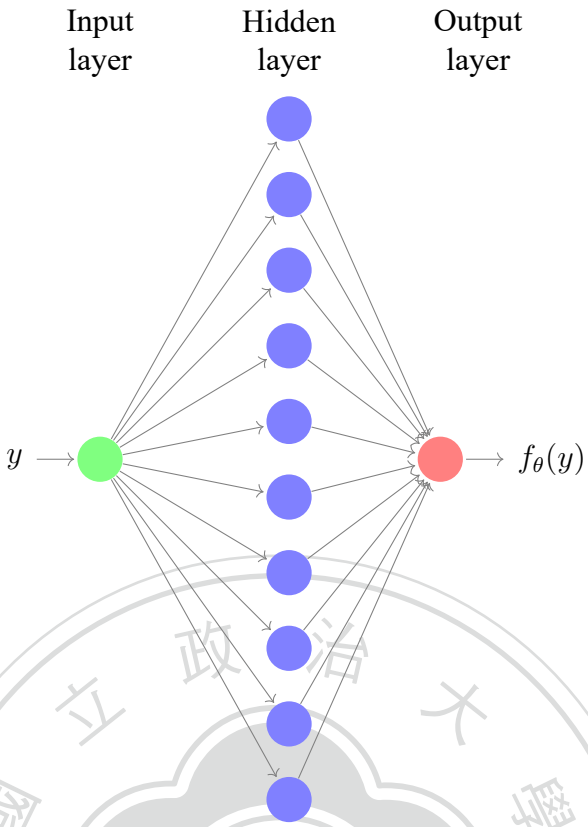


Figure 3.1: Structure of neural network for finding a solution of (3.5.1)

For the neural network approach, the deep learning algorithm, Adam [**?**], is used to minimize the unconstrained optimization problem.

The L^2-losses between the neural network solution and the solution of equation and the boundary condition are $2.31 \times 10{-}4$, 5.19×10^{-5} and 2.34×10^{-5} for 10,000 iterations, 20,000 iterations and 30000 iterations. The L^2-losses for 20,000 iterations and 30,000 iterations are same as $e-5$ and do not reduce so much from 20,000 iterations to 30,000 iterations. The comparison results between the analytical solution and the numerical soltion (20,000 iterations) are demonstrated in Figure 3.2 and Figure 3.3 for 10,000 iterations and 20,000 iterations, respectively.

DOI:10.6814/NCCU202100296

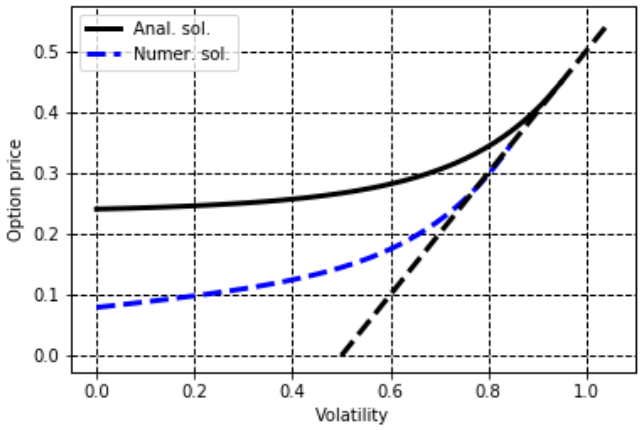


Figure 3.2: Comparison results between analytic solution and the numerical solution (10,000 iterations)

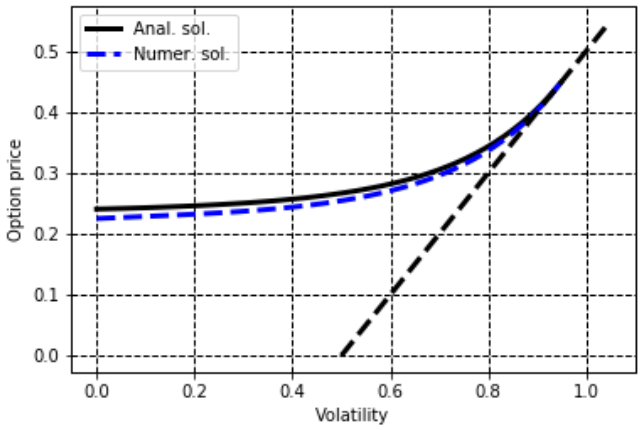


Figure 3.3: The comparison results between analytic solution and the numerical solution (20,000 iterations)

DOI:10.6814/NCCU202100296

Chapter 4

Reconstruction and Interpolation of Manifolds

In this chapter, an algorithmic method of interpolation and reconstruction of Riemannian manifold in Euclidean space will be introduced. These methods are based on a series of works in [29, 30, 31].

4.1 Introduction

High-dimensional data is increasingly available in various fields. To deal with the curse of dimensionality, one approach is to use dimensionality reduction [18, 33, 97] while another approach is to consider the manifold learning in high-dimensional space. When applying methods of dimensionality reduction, some meaningful information might lose since the structure of raw data is largely changed in terms of vector. Another issue is that assumptions of dimensionality reduction are given without considering the geometry of the data. Please refer [50, 59, 78] for comprehensive surveys of manifold learning.

In [30], two methods were proposed to deal with manifold reconstruction problem. The first one is to interpolate a data cloud X in $\mathbb{R}^n$ by a Riemannian submanifold $\mathcal{M}$ in $\mathbb{R}^n$ so that X is closed to $\mathcal{M}$ in some sense. Second, based on the Whitney embedding theorem [96], a metric space can be "almost" embedded as a Riemannian submanifold of some Euclidean space with minimum distance distortion. Under the existence of noisy data, methods in [30] are guaranteed to generate a similar manifold with high probability.

DOI:10.6814/NCCU202100296

In section 4.2, we will briefly introduce the Hausdorff Distance, which provides a measurement of distance between subsets of a metric space. Two theorems about reconstruction and interpolation of manifolds proved in [30] are stated in section 4.3. Some basic knowledge on Riemannian geometry will also be stated in this section. In section 4.4, we will summarize algorithms needed to implement above theorems in practice.

4.2 Hausdorff Distance and δ-closeness

We first introduce the following notations.

Definition 4.2.1. *Let (X, d) be a metric space and $A, B \subseteq X$ and $r > 0$.*

1. *The r-neighborhood of A is defined by*

$$\mathcal{U}_r(A) = \{x \in X \mid d(x, a) < r \text{ for some } a \in A\} = \bigcup_{a \in A} B_X(a, r)$$

   where $B_X(a, r)$ is the open ball of radius r and center a in X.

2. *A is called a r-net in X if $\mathcal{U}_r(A) = X$.*

3. *A is called a maximal r-separated subset of X if A is a r-net of X and*

$$d_X(x, y) \geq r \quad \text{for all } x \neq y \in A.$$

Definition 4.2.2. *Let (X, d) be a metric space and $A, B \subseteq X$ The Hausdorff distance between sets A and B of X is defined by*

$$\begin{aligned} d_H(A, B) &= \inf\{r > 0 \mid A \subseteq \mathcal{U}_r(B) \text{ and } B \subseteq \mathcal{U}_r(A)\} \\ &= \max\left(\sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b)\right) \end{aligned}$$

Note that Hausdorff distance d_H is only a pseudo-metric since $d_H(A, B) = 0$ does not imply $A = B$ in general. For instance, consider $A = [0, 1]$ and $B = (0, 1)$ in $\mathbb{R}^1$. But this does not affect how d_H measure the difference between different sets.

The concept of Hausdorff distance can be extended to measure the difference between sets from different metric spaces by first putting them isometrically into one common metric space.

DOI:10.6814/NCCU202100296

Definition 4.2.3. *The Gromov-Hausdorff distance between metric spaces X and Y is defined by*

$$d_{GH}(X,Y) = \inf\{d_H(f(X),g(Y)) \mid f : X \to Z, g : Y \to Z \text{ are isometries for some metric space } Z\}$$

where d_H is the Hausdorff distance induced by metric space Z.

Definition 4.2.4. *Let $r, \delta > 0$ and $n \in \mathbb{N}$.*

1. *Let X be a metric space. For $r > \delta > 0$, we say that X is δ-close to $\mathbb{R}^n$ at scale r if*

$$d_{GH}(B_X(x,r), B_n(r)) < \delta$$

2. *Let X be a subset of a Hilbert space E. We say that X is δ-close to n-flats at scale r if for any $x \in X$, there exists an n-dimensional affine space $A_x \subseteq E$ through x such that*

$$d_H(X \cap B_E(x,r), A_x \cap B_E(x,r)) \leq \delta.$$

For the following definitions, we assume that X, Y are metric spaces unless specified.

Definition 4.2.5. *Let $\delta > 0$, and $\{x_i\}_{i=1}^N$ be a finite sequence of X.*

1. *$\{x_i\}_{i=1}^N$ is a δ-chain if $d(x_i, x_{i+1}) < \delta$ for all $1 \leq i \leq N-1$.*
2. *$\{x_i\}_{i=1}^N$ is δ-straight if $d(x_i,x_j) + d(x_j,x_k) < d(x_i,x_k) + \delta$ for all $1 \leq i < j < k \leq N$.*
3. *X is δ-intrinsic if for $x, y \in X$, there is a δ-straight δ-chain $\{x_i\}_{i=1}^N$ with $x_1 = x$ and $x_N = y$.*

Intuitively speaking, a δ-intrinsic space X means any two points in X can be connected by an "almost" straight polygon with length of each side smaller than δ.

Definition 4.2.6. *For $\lambda \geq 1$ and $\varepsilon > 0$. A map $f : X \to Y$ is said to be a (λ, ε)-quasi-isometry if $f(X)$ is an ε-net in Y and*

$$\frac{1}{\lambda} d_X(x,y) - \varepsilon \leq d_Y(f(x), f(y)) \leq \lambda d_X(x,y) + \varepsilon$$

DOI:10.6814/NCCU202100296

for all $x, y \in X$. In particular, a $(1, 0)$-quasi-isometry is an isometry.

Remark 4.2.7. *Note that if a map $f : X \to Y$ is (λ, ε)-quasi-isometry, then there exists a $(\lambda, 3\lambda\varepsilon)$-quasi-isometry map $g : Y \to X$. We say that two metric spaces X and Y are (λ, ε)-quasi-isometric if there are (λ, ε)-quasi-isometries in both directions.*

Definition 4.2.8. *Let X, Y be metric spaces and $f : X \to Y$ be a map. The distortion of f, denoted by* dis f*, is defined by*

$$\text{dis } f = \sup_{x,y \in X} |d_Y(f(x), f(y)) - d_X(x, y)|$$

For $\varepsilon > 0$, we say that f is ε-isometry if dis $f < \varepsilon$ *and $f(X)$ is a ε-net in Y.*

Two metric spaces with small Gromov-Hausdorff distance are ε-isometry, moreover, this is true conversely.

Theorem 4.2.9. *Let X, Y be metric spaces.*

1. *If $d_{GH}(X, Y) < \varepsilon$, then there exists a 2ε-isometry from X to Y.*
2. *Conversely, if there is an ε-isometry from X to Y, then $d_{GH}(X, Y) < \varepsilon$. Moreover,*

$$d_{GH}(X, f(X)) \leq \frac{1}{2} \text{dis } f$$

If, in addition, $f(X)$ is a ε-net, then

$$d_{GH}(X, Y) \leq \frac{1}{2} \text{dis } f + \varepsilon \tag{4.2.1}$$

Proof. See [9, Theorem 7.3.25 and Corollary 7.3.28] for proofs of these facts. □

Corollary 4.2.10. *If $f : X \to Y$ is a (λ, ε)-quasi-isometry, then we have*

$$d_{GH}(X, Y) \leq \frac{1}{2}(\lambda - 1) \operatorname{diam}(X) + \frac{3}{2}\varepsilon.$$

Proof. Let $f : X \to Y$ be a (λ, ε)-quasi-isometry. Then we have $f(X)$ is a ϵ-net of Y, and,

$$d_Y(f(x), f(y)) \leq \lambda d_X(x, y) + \varepsilon \tag{4.2.2}$$

DOI:10.6814/NCCU202100296

for any $x, y \in X$.

Subtracting $d_X(x, y)$ from both sides of (4.2.2), we have

$$d_Y(f(x), f(y)) - d_X(x, y) \leq (\lambda - 1)d_X(x, y) + \varepsilon \leq (\lambda - 1)\,\mathrm{diam}(X) + \varepsilon$$

Hence, dis $f \leq (\lambda - 1)\,\mathrm{diam}(X) + \varepsilon$. Combined with $(4.2.1)$, the desired resul follows. □

4.3 Reconstruction and Interpolation of Manifolds

Below, we borrow some definitions about manifolds from the book [12].

Definition 4.3.1. *A differentiable (C^k, smooth, respectively) manifold of dimension n is a set $\mathcal{M}$ and a family of maps $x_\alpha : U_\alpha \subseteq \mathbb{R}^n \to \mathcal{M}$ of open sets U_α of $\mathbb{R}^n$ into $\mathcal{M}$ so that*

1. $\cup_\alpha x_\alpha(U_\alpha) = \mathcal{N}$.
2. *for any pair α, β with $W = x_\alpha(U_\alpha) \cap x_\beta(U_\beta) \neq \phi$, the sets $x_\alpha^{-1}(W)$ and $x_\beta^{-1}(W)$ are open sets in $\mathbb{R}^n$ and the map $x_\beta^{-1} \circ x_\alpha$ are differentiable (C^k, smooth, respectively).*
3. *The family $\{(U_\alpha, x_\alpha)\}$ is maximal relative to the above two conditions.*

Below, we informally state definitions of tangent space and the Riemannian metric.

Definition 4.3.2. *Let $\mathcal{M}$ be a differentiable manifold.*

1. *A differentiable function $\alpha : (-\varepsilon, \varepsilon) \to \mathcal{M}$ is called a curve in $\mathcal{M}$.*
2. *Suppose that $\alpha(0) = p \in \mathcal{M}$, and let $\mathcal{D}$ be the set of functions on $\mathcal{M}$ that are differentiable at p. The tangent vector to the curve α at $t = 0$ is a function $\alpha'(0) : \mathcal{D} \to \mathbb{R}$ given by*

$$\alpha'(0)f = \frac{d(f \circ \alpha)}{dt}\bigg|_{t=0}, \quad f \in \mathcal{D}$$

3. *A tangent vector at p is the tangent vector at $t = 0$ of some curve $\alpha : (-\varepsilon, \varepsilon) \to \mathcal{M}$ with $\alpha(0) = p$.*
4. *The set of all tangent vectors to $\mathcal{M}$ at p is denoted by $T_p\mathcal{M}$, called the tangent space at p.*

DOI:10.6814/NCCU202100296

5. *A Riemannian metric g on $\mathcal{M}$ is a family of smoothly varying inner products g_p on the tangent spaces $T_p\mathcal{M}$ of $\mathcal{M}$.*

6. *A Riemannian manifold $(\mathcal{M}, g)$ is a differentiable manifold $\mathcal{M}$ equipped with a Riemannian metric g.*

We first formally state the problems of manifold interpolation and manifold reconstruction of data cloud.

1. Let (X, d) be a metric space. The goal of manifold reconstruction problem is to find a Riemannian submanifold $(\mathcal{M}, g)$ so that X and $\mathcal{M}$ are quasi-isometry. That is, $d(x, y) \approx g(x_{\mathcal{M}}, y_{\mathcal{M}})$, where $x_{\mathcal{M}}$ is the image of x under the quasi-isometry.

2. Let $X \subseteq \mathbb{R}^m$ be a data cloud. The goal of the manifold interpolation problem is to find a n-dimensional submanifold $\mathcal{M}$ so that

$$d_H(X, \mathcal{M}) < \varepsilon$$

Two main theorems in [30] provide an initial work about how to construct the desired manifolds algorithmically with sufficient theoretical guarantees.

Theorem 4.3.3 ([30])**.** *For every $n \in \mathbb{N}$, there exists $\sigma_1(n), C_1(n), C_2(n) > 0$ so that the following holds:*

Let $r > 0$ and X be a metric space with $\operatorname{diam}(X) > r$ *and $0 < \delta < \sigma_1 r$. Suppose that X is δ-intrinsic and δ-close to $\mathbb{R}^n$ at scale r. Then there exists a complete n-dimensional Riemannian manifold $\mathcal{M}$ such that*

1. *X and $\mathcal{M}$ is $(1 + C_1\delta r^{-1}, C_1\delta)$-quasi-isometric. Moreover, if* $\operatorname{diam}(X) < \infty$*, then we have*

$$d_{GH}(X, M) \leq 2C_1\delta r^{-1} \operatorname{diam}(X)$$

2. *The modulus of sectional curvature of $\mathcal{M}$ is bounded by $C_2\delta r^{-3}$.*

3. *The injective radius of $\mathcal{M}$ is bounded below by $\frac{r}{2}$, where the injective radius of a Riemannian manifold $(\mathcal{M}, g)$ is the largest $R > 0$ so that*

$$\exp_p : B_{T_p\mathcal{M}}(0, R) \subseteq T_p\mathcal{M} \to \mathcal{M}$$

DOI:10.6814/NCCU202100296

is a diffeomorphism for all $p \in \mathcal{M}$.

Theorem 4.3.4 ([30]). *For every $n, k \in \mathbb{N}$, there exists $\sigma_2(n), C_3(n), C_4(n), C_5(n,k) > 0$ so that the following holds:*

Let X be a subset of a separable Hilbert space E, $r > 0$ and $0 < \delta < \sigma_2\delta$. Suppose that X is δ-close to n-flats at scale r. Then there exists a closed n-dimensional smooth submanifold $\mathcal{M} \subseteq E$ such that

1. $d_H(X, \mathcal{M}) \leq 5\delta$.
2. *The second fundamental form of $\mathcal{M}$ at every point is bounded by $C_3\delta r^{-2}$*
3. *The reach of $\mathcal{M}$ is bounded below by $\frac{r}{3}$, where*

$$\text{Reach}(\mathcal{M}) = \sup\{r > 0 \mid \textit{normal projection} P_{\mathcal{M}} : \mathcal{U}_r(\mathcal{M}) \to \mathcal{M} \textit{ is well-defined}\}.$$

4. *The normal projection $P_{\mathcal{M}} : \mathcal{U}_{r/3}(\mathcal{M}) \to \mathcal{M}$ is smooth and satisfies for all $x \in \mathcal{U}_{r/3}(M)$,*

$$\|d_x^k P_{\mathcal{M}}\| < C_5(n,k)\delta r^{-k}, k \geq 2$$

and

$$\|d_x P_{\mathcal{M}} - P_{T_y\mathcal{M}}\| < C_5(n,k)\delta r^{-1}$$

where $y = P_{\mathcal{M}}(x)$ and $P_{T_y\mathcal{M}}$ is the orthogonal projection onto $T_y\mathcal{M}$

5. *If $x \in X$ and $y = P_{\mathcal{M}}(x)$, then the angle between A_x and the tangent space $T_y\mathcal{M}$ satisfies*

$$\angle(A_x, T_y\mathcal{M}) < C_4\delta r^{-1}$$

Note that the angle $\angle(A, B)$ between n-dimensional linear subspaces of an inner product space $(E, \langle\cdot,\cdot\rangle)$ is defined by

$$\angle(A, B) = \max_{a \in A}\left\{\min_{b \in B}\{\angle(a,b) \mid a, b \neq 0\}\right\}$$

where $\angle(a,b) = \arccos\dfrac{\langle a, b, \rangle}{\|a\| \cdot \|b\|}$ and $\langle\cdot,\cdot\rangle$.

DOI:10.6814/NCCU202100296

In fact, there are serval definitions of the angle between pair of subspaces of a Hilbert space. These definitions are expressed in terms of the orthogonal projection onto these subspaces. For more detail about the angle between subspaces of a Hilbert space, please refer [24].

4.4 Algorithms for Manifold Interpolation

In this section, we describe three algorithms to implement the algorithm of manifold interpolation based on Theorem 4.3.4.

Let X be a finite set of $\mathbb{R}^m$. If X is δ-close to n-flasts at scale 1 (by rescaling factor $1/r$). Then we consider the following steps to generate a Riemannian manifold $\mathcal{M}$ according to theorem :

1. Find a maximal $\frac{1}{100}$-separated subset $X_0 = \{q_i\}_{i=1}^N$ of X.

2. For each $q_i \in X_0$, find an affine subspace A_i of $\mathbb{R}^m$ passing through q_i so that

$$d_H(B_X(q_i, 1), B_{\mathbb{R}^m}(q_i, 1) \cap A_i) < \delta$$

3. For each $q_i \in X_0$, define the affine projection $P_i : \mathbb{R}^m \to A_i$ and

$$\phi_i(x) = \mu_i(x)P_i(x) + (1 - \mu_i(x))x$$

   the convex combination of $P_i(x)$ and x, where $\mu_i(x) = \mu(\|x - q_i\|)$ for a fixed smooth function $\mu : (0, \infty) \to [0, 1]$ satisfying $\mu = 1$ on $[0, \frac{1}{3}]$ and $\mu = 0$ on $[\frac{1}{2}, 1]$

4. Define $f = \phi_N \circ \phi_{N-1} \circ \cdots \circ \phi_1$.

5. Compute $\mathcal{M} = f(\mathcal{U}_\delta(X))$ by sampling method.

In third step, $\mu_i(x) \in [0, 1]$ is used as a coefficient of convex combination of the identity function and $P_i(x)$ to control how $\phi_i(x)$ far from $P_i(x)$ by using the distance between x and q_i. In fact, $\phi_i(x) \to P_i(x)$ as $x \to q_i$. In other words, identity function is homotopic to $P_i(x)$ as $x \to q_i$.

In [30], authors consider a bump function $\mu : \mathbb{R} \to \mathbb{R}$ as follows:

DOI:10.6814/NCCU202100296

First, define $\mu(x) = \frac{e^{(t-1/3)^{-1}}}{e^{(1/2-t)^{-1}} + e^{(t-1/3)^{-1}}}$ on $(\frac{1}{3}, \frac{1}{2})$. Then extend $\mu(x)$ trivially from $(0, \frac{1}{3})$ to $[0, 1]$. One advantage of such μ is that it can be easily implemented as a shallow neural net. In fact, we can replace $\mu : (0, \infty) \to [0, 1]$ with any function which satisfies $\mu = 1$ on $[0, \frac{1}{3}]$ and $\mu = 0$ on $[\frac{1}{2}, 1]$.

Informally speaking, some affine subspaces are constructed as tangent spaces of the desired manifold, and then we glue these tangent space by composing ϕ_i. Then f_N is a map which projects point around X onto the manifold $\mathcal{M}$.

The algorithm of the first step is not provided in [30]. Therefore, we propose an intuitive algorithm to implement as shown in algorithm 1. In the second step, the affine subspace is constructed by a finding an "almost" orthonormal frame. This algorithm is described in algorithm 2. The algorithm 3 consists of rest steps to generate the desired submanifold $\mathcal{M}$.

Algorithm 1: FindMaxSepDet

Input: $\delta > 0, X \subseteq \mathbb{R}^m$ is a finite set.

Output: the set of indices of a maximal δ-separated set of X

Set $I = \{1, 2, \cdots, |X|\}$.

Set $J = \phi$.

Set $i = 1$.

foreach $1 \leq i \leq |X|$ **do**
  Set $I_i = \{j \in I \mid d_{ij} < \delta\}$.

end

while $I \neq \phi$ **do**
  Set $i_0 = \min I$. // Choose the least feasible index
  Set $q_i = x_{i_0}$.
  Set $J = J \cup \{i_0\}$.
  Set $I = I \setminus I_{i_0}$. // Remove all points within a distance δ of q_i
  Set $i = i + 1$.

end

return J

Note that the choice of X_0 depends on the order of elements of X. This might cause a

DOI:10.6814/NCCU202100296

potential issue to generate different manifolds.

Algorithm 2: FindDisc

Input: $n \in \mathbb{N}$, $x \in X$

Output: an affine n-space A_x passing through x

Define $X_1 = X \cap B_1(x)$.

Define $Y = X_1 - x$. // shift all points by x so x is moved to the original

Define $y_1 = \arg\min_{y \in Y} |1 - \|y\||$.

for $1 \leq m \leq n-1$ **do**

$y_{m+1} = \arg\min_{y \in Y \setminus \{y_1, \cdots, y_m\}} \max\{\big|1 - \|y\||, \langle \frac{y_1}{\|y_1\|}, y\rangle, \cdots, \langle \frac{y_m}{\|y_m\|}, y\rangle\}$

end

Define $A = \text{span}\{y_1, \cdots, y_n\}$.

Set $A_x = x + A$ **return** A_x

Let $X \subseteq E = \mathbb{R}^m$ be a δ-close to n-flats at scale r. By rescaling X with the factor $1/r$, we can assume that $r = 1$.

Algorithm 3: SubmanifoldInterpolation

Input: $n, m \in \mathbb{N}, r > 0, \delta \in (0, 1)$, $X \subseteq \mathbb{R}^m$ is δr-close to n-flats at scale 1.

Output: a n-dimensional submanifold $\mathcal{M}$ of $\mathbb{R}^m$.

Find a maximal $\frac{1}{100}$-separated set $X_0 = \{q_i\}_{i=1}^N$ of X.

foreach q_i **do**

Define A_i =FindDisc(n, q_i), the affine n-space of $\mathbb{R}^m$ passing through q_i.

Define $P_i : \mathbb{R}^m \to \mathbb{R}^m$, the orthogonal projection onto A_i.

Define $\mu_i : \mathbb{R}^m \to \mathbb{R}$, a bump function around q_i.

Define $\phi_i(x) = x + \mu_i(x)(P_i(x) - x)$.

end

Define $f = \phi_N \circ \phi_{N-1} \circ \cdots \circ \phi_1$, and $\mathcal{M} = f(\mathcal{U}_\delta(X))$. **return** $\mathcal{M}$

In next section, we will see that the algorithm of manifold interpolation can be reformulated a training process of various independent ResBlocks.

In this chapter, we consider a deep neural network version of the manifold interpolation problem as introduced in chapter 4.

DOI:10.6814/NCCU202100296

4.5 Affine Space and Affine Projection

Recall that, given a data cloud $X \subseteq \mathbb{R}^m$ which is δ-close to n-flats at scale 1. The goal of the manifold interpolation problem is to construct a n-dimensional submanifold $\mathcal{M} \subseteq \mathbb{R}^m$ so that

$$d_H(X, \mathcal{M}) < C\delta$$

where C is a known constant.

The procedure of finding such $\mathcal{M}$ can be decomposed into four steps:

1. Find a maximal $\frac{1}{100}$-separated subset $X_0 = \{q_i\}_{i=1}^{|X_0|} \subseteq X$.
2. For each q_i, find an affine n-space A_i at q_i and the affine projection P_i on A_i.
3. Gluing each A_i smoothly and denote such gluing map by f.
4. Define $\mathcal{M} = f(\mathcal{U}_\delta(X))$ for some $\delta > 0$.

Note that is step 2, an affine projection is constructed on the affine subspace A_x. This motivates us to the study of affine projection on an affine subspace. We first state some basic definitions.

Definition 4.5.1. *Let $(V, \langle\cdot,\cdot\rangle)$ be a inner product space, $P : V \to V$ is a linear operator on V.*

1. *P is a projection if $P^2 = P$. That is, P is idempotent.*
2. *A projection P is called an orthogonal projection if P is self-adjoint. That is, $\langle Px, y\rangle = \langle x, Py\rangle$ for any $x, y \in V$.*
3. *We say P is a projection onto a subspace W of V if $P(V) = W$. Moreover, V can be represented as an direct sum*

$$V = W \bigoplus \ker(P)$$

Theorem 4.5.2. *Let $(V, \langle\cdot,\cdot\rangle)$ be a inner product space and W is a subspace.*

1. *P is an orthogonal projection $\iff$ $I - P$ is an orthogonal projection onto $W^\perp$, where $W^\perp = \{x \in V \mid \langle x, w\rangle = 0$ for all $w \in W\}$, is the orthogonal complement of W in V.*
2. *If P be an orthogonal projection onto W, then $V = W \bigoplus W^\perp$.*

DOI:10.6814/NCCU202100296

To study affine projection other than orthogonal projection, we need to know the translation operation of a set in a vector space.

Definition 4.5.3. *Let S be subset of a vector space V and $a \in S$. The translation of S by is defined by*

$$a + S = \{a + b \mid b \in S\} = S + a$$

Definition 4.5.4. *Let S be subset of a vector space V. We say that S is an affine subspace of V is S is a translation of a vector subspace of V. That is, there exists a subspace $W \subseteq V$ so that*

$$S = a + W$$

for any $a \in S$. We can also define the dimension of S by the vector dimension of W. That is, for any $x \in S$, we have $x - a \in W$.

Note that for any $a, b \in V$, $a + W = b + W \iff a - b \in W$. Hence, the definition of affine subspace is well-defined.

With the relationship between linear subspace and affine subspace, we can define the affine projection as follows:

Definition 4.5.5. *Let S be an affine subspace of V and $S = W + a$ for some linear subspace W of V. The affine projection Π_S is a operator $\Pi_S : V \to V$ defined by*

$$\Pi_S(x) = a + P_W(x - a) = P_W(x) + P_{W^\perp}(a)$$

where P_W is the orthogonal projection onto W.

Theorem 4.5.6. *Let Π be an affine projection onto an affine subspace $S = a + W$.*

(a) $\Pi^2 = \Pi$. *That is, Π is idempotent.*

(b) For any $x \in V$, $\Pi_{a+W^\perp}(x) = a + (I - \Pi)(x)$.

(c) For $k \in \mathbb{N}$, $(I - \Pi)^{k+1} + kP_{W^\perp}(a) = I - \Pi$.

(d) For $x, y \in V$, $\|\Pi_S(x) - \Pi_S(y)\| = \|P_W(x) - P_W(y)\|$, where $S = a + W$.

DOI:10.6814/NCCU202100296

Proof. (a) For any $x \in V$, we have

$$\begin{aligned}
\Pi^2(x) &= \Pi\big(P_W(x) + P_{W^\perp}(a)\big) \\
&= P_W(P_W(x) + P_{W^\perp}(a)) + P_{W^\perp}(a) \\
&= P_W^2(x) + P_W P_{W^\perp}(a) P_{W^\perp}(a) \\
&= P_W(x) + P_{W^\perp}(a) \text{ since } P_W^2 = P_W \text{ and } P_W P_{W^\perp} \equiv 0 \\
&= \Pi_S(x)
\end{aligned}$$

(b) For any $x \in V$, we have

$$\begin{aligned}
(I - \Pi)(x) &= x - P_W(x) - P_{W^\perp}(a) \\
&= (I - P_W)(x - a) \\
&= P_{W^\perp}(x - a)
\end{aligned}$$

(c) Note that for any scalar k, we have

$$\begin{aligned}
\Pi\big(P_{W^\perp}(x-a) - kP_{W^\perp}(a)\big) &= P_W\big(P_{W^\perp}(x-a) - kP_{W^\perp}(a)\big) + P_{W^\perp}(a) \\
&= P_{W^\perp}(a)
\end{aligned}$$

Also, by (b), we have

$$\begin{aligned}
(I - \Pi)^2(x) &= (I - \Pi)\big(P_{W^\perp}(x - a)\big) \\
&= P_{W^\perp}(x - a) - P_{W^\perp}(a) \\
&= (I - \Pi)(x) - P_{W^\perp}(a)
\end{aligned}$$

DOI:10.6814/NCCU202100296

By induction, we have

$$\begin{aligned}(I-\Pi)^{k+1}(x) &= (I-\Pi)\big(P_{W^\perp}(x-a)-(k-1)P_{W^\perp}(a)\big)\\ &= P_{W^\perp}(x-a)-(k-1)P_{W^\perp}(a)-\Pi\big(P_{W^\perp}(x-a)-(k-1)P_{W^\perp}(a)\big)\\ &= P_{W^\perp}(x-a)-(k-1)P_{W^\perp}(a)-P_{W^\perp}(a)\\ &= (I-\Pi)(x)-kP_{W^\perp}(a)\end{aligned}$$

The desired result follows.

(d) Follow from the definition of affine projection.

□

Unlike in the case of orthogonal projections, from Theorem 4.5.6(b) and (c), we can see that $(I-\Pi)$ is neither an affine projection, nor idempotent. These results are derived from the fact that Π is not a linear operator since $\Pi(-x)\neq-\Pi(x)$. For more detail about affine projection, please refer [77].

Note that $\Pi-I$ is also not idempotent, however, the square of $\Pi-I$ is idempotent.

Theorem 4.5.7. *Let Π be an affine projection. Then $(\Pi-I)^2$ is idempotent.*

Proof. For any $x\in V$, we have

$$\begin{aligned}(\Pi-I)^2(x) &= (\Pi-I)\big(\Pi(x)-x\big)\\ &= (\Pi-I)\big(a+P(x-a)-x\big)\\ &= \Pi\big(a+P(x-a)-x\big)-\big(a+P(x-a)-x\big)\\ &= a+P\big(a+P(x-a)-x-a\big)-a-P(x-a)+x\\ &= P^2(x-a)-P(x)-P(x-a)+x\\ &= x-P(x)=P^\perp(x)\end{aligned}$$

Hence, $(\Pi-I)^2$ is idempotent.

□

DOI:10.6814/NCCU202100296

This property would be the central idea to rewrite the algorithm of manifold interpolation into a deep neural network model.

4.6 Manifold Interpolation as Residual Network

Let $X_0 = \{q_i\}_{i=1}^{|X_0|}$ be a maximal $\frac{1}{100}$-separated subset of X. Write $X = \bigcup_{i=1}^{|X_0|} X_i$, where $X_i = B(q_i, \frac{1}{100})$.

Recall in step 2 of the manifold interpolation, for each q_i, we find an affine subspace A_i passing throung q_i, this induces an affine projection P_i on A_i. Then, with μ_i, we define

$$\phi_i(x) = \big(1 - \mu_i(x)\big)x + \mu_i(x)P_i(x) = x + \mu_i(x)\big(P_i(x) - x\big)$$

Define $F_i(x) = mu_i(x)(P_i(x) - x)$, then

$$\phi_i(x) = x + F_i(x) \tag{4.6.1}$$

Note that $F_i(x) = P_i(x) - x$ when $\|x - q_i\| \leq \frac{1}{3}$, by Theorem 4.5.7, we have

$$F_i^4(x) = F_i^2(x) \tag{4.6.2}$$

if $\|x - q_i\| \leq \frac{1}{3}$.

On the other hand,

$$F_i(x) = 0 \tag{4.6.3}$$

if $\|x - q_i\| > \frac{1}{2}$.

Compared with (2.2.1), if we replace F_i in (4.6.1) by a neural network f_{θ_i}, then (4.6.1) becoms $\phi_i(x) = x + f_{\theta_i}(x)$

This reformulation turns ϕ_i into a ResBlock (see (2.2.1)). In particular,

$$f = \phi_N \circ \phi_{N-1} \circ \cdots \circ \phi_1$$

becomes a ResNet under such replacement.

DOI:10.6814/NCCU202100296

Combined with (4.6.2) and (4.6.3), the objective function for f_{θ_i} is then given by

$$L(\theta_i) = \frac{1}{2N_j}\sum_{k=1}^{N_j}\|f_{\theta_i}^4(x_k) - f_{\theta_i}^2(x_k)\|^2 + \frac{1}{2M_j}\sum_{k=1}^{M_j}\|f_{\theta_i}(y_j)\|^2 \tag{4.6.4}$$

The remaining part is to show that

$$S_i = f(\mathcal{U}_\delta(X_i))$$

is a smooth n-dimensional submanifold for some $n \in \mathbb{N}$ and $\delta > 0$. However, the choice of δ is another issue remain to be solved.

One advantage of this formulation is that each θ_i can be trained independently. The cost of training process can be largely reduced and potentially it can be parallelized. This is our ongoing work.

4.6.1 Further Relaxation

Note that the first step of the manifold interpolation is to find a maximal separated subset $\{q_i\}$ and the rest step relies on the existence of $\{q_i\}$. In fact, this process can be replaced by singular value decomposition (SVD) if X can be decomposed to $X = \bigcup X_i$ so that $\text{diam}(X_i) \leq \frac{1}{100}$ for all i and $d_{\mathbb{R}^m}(\overline{X_i}, \overline{X_j}) \geq \frac{1}{100}$ for any $i \neq j$, where $\overline{X_i} = \frac{1}{|X_i|}\sum_{x\in X_i}$ is the mean position of X_i.

One simple way is setting $z_i = \overline{X_i}$ and apply SVD to $X_i - z_i$ to get the best low-dimensional representation of matrix formed by $x - z_i$ for $x \in X_i$. Therefore, we can find a linear n-space W_i which fits $X_i - z_i$ the most. Hence, $A_i = z_i + W_i$ fits X_i in the same manner. In this case, z_i lies in A_i but z_i may not lie in X_i in general.

We can replace the original z_i with the mean of X_i and process the same composition of ϕ_i as before to reconstruct manifold. However, we still do not have any theoretical analysis yet.

In this case, we can consider an objective function similar to (4.6.4)

$$L(\theta_i) = \frac{1}{2|X_j|}\sum_{x\in X_i}\left\|F_i^2(x) + F_i(x)\right\|^2 + \sum_{x\notin X_i}\|F_i(x)\|^2 \tag{4.6.5}$$

DOI:10.6814/NCCU202100296

4.7 Conclusion

We have already shown that the manifold interpolation algorithm proposed in [30] can be reformulated as serval ResBlocks. However, before we dig deeper into the theoretical analysis of this ResNet based interpolation method.

We are now working on some dataset to validate our concept. Part of this work has been summited into the SIAM Conference on Mathematics of Data Science (MDS20).

DOI:10.6814/NCCU202100296

Part II

HodgeRank and its Continuity

DOI:10.6814/NCCU202100296

Chapter 5

Pairwise Comparison and Combinatorial Hodge Theory

In this chapter, we introduce the ranking problem based on the pairwise ranking data and the combinatorial Hodge theory with its relevance in pairwise ranking problem.

5.1 Introductions

Ranking is an essential part in recommendation system. A recommendation system is a system which provides a personalized list of items ranked by ranking algorithms.

However, humans are unable to make a precise preference decision on a set, which contains more than 10 distinct items at the same time [68]. However, people can easily compare two items

For $n \in \mathbb{N}$, we denote $[n] = \{1, 2, \cdots, n\}$ be the number of items to be ranked by a voter. A ranking to these n items may not be given directly. Instead, we may assume that, for any two alternative items $i \neq j$ in $[n]$, a voter either

- prefer i to j
- prefer j to i
- is indifferent to i and j.

To quantify preferences between different items, we define a real number a_{ij} to indicate the preference between i and j. Pairwise comparison can be represented either on the *additive scale* or on the *multiplicative scale*.

DOI:10.6814/NCCU202100296

Definition 5.1.1. *Let $E = \{(i,j) \mid i \text{ and } j \text{ are compared}\}$. We say that $A_E = \{a_{ij} \mid (i,j) \in E\}$ is a pairwise comparison of $[n]$ on the*

- *additive scale if $a_{ij} \in \mathbb{R}$ and $a_{ij} + a_{ji} = 0$ for all $(i,j) \in E$.*

  Also,

$$a_{ij} \begin{cases} > 0 & \text{if } i \text{ is preferred to } j \\ < 0 & \text{if } j \text{ is preferred to } i \\ = 0 & \text{if } i \text{ and } j \text{ are equally preferred} \end{cases}$$

  In this case, a_{ij} is the difference of measurement of preference between i and j.

- *multiplicative additive scale if $a_{ij} \in (0,\infty)$ and $a_{ij} \cdot a_{ji} = 1$ for all $(i,j) \in E$.*

  Also,

$$a_{ij} \begin{cases} > 1 & \text{if } i \text{ is preferred to } j \\ < 1 & \text{if } j \text{ is preferred to } i \\ = 1 & \text{if } i \text{ and } j \text{ are equally preferred} \end{cases}$$

  That is, a_{ij} measures the multiplicative preference of i over j.

Note that $a_{ij} \in A_E$ is defined for $(i,j) \in E$. However, if we treat i and j are indifferent if they are not compared. Then we can fill A_E with undefined a_{ij} to form a matrix A.

More precisely, we cane define an $n \times n$ matrix A by

$$A_{ij} = \begin{cases} a_{ij} & \text{if } i \text{ and } j \text{ are compared.} \\ 0 & \text{if } i \text{ and } j \text{ are not compared.} \end{cases}$$

That is, A_E induces a *skew-symmetric* matrix A.

If A is an additive comparison matrix, we can define an $n \times n$ matrix A by

$$A_{ij} = \begin{cases} a_{ij} & \text{if } i \text{ and } j \text{ are compared.} \\ 1 & \text{if } i \text{ and } j \text{ are not compared.} \end{cases}$$

DOI:10.6814/NCCU202100296

That is, A_E induces a *symmetrically reciprocal* matrix A.

In either scale, the induced matrix is called a pairwise comparison matrix on the certain scale. Note that pairwise comparison matrix was initially used in a multi-criteria decision making method, called analytic hierarchy process. Due to the development of the analytic hierarchy process, pairwise comparison matrix was proposed and studied deeply on the multiplicative scale more than on an additive scale. However, latter one provides more insight from the theory of matrix algebra.

Remark 5.1.2. *A multiplicative pairwise comparison matrix has an one-to-one correspondence to an additive pairwise comparison matrix. This can be easily proved by elementary properties of logarithmic functions.*

Proof. Let $B = [b_{ij}]$ be a multiplicative pairwise comparison matrix, define $a_{ij} = \log\ b_{ij}$. Then we have

$$a_{ij} + a_{ji} = \log\ b_{ij} + \log\ b_{ji} = \log\,(b_{ij} \cdot b_{ji}) = \log\ 1 = 0$$

and

$$a_{ij} = \log\ b_{ij} = -\log\ b_{ji}^{-1} = \log\ b_{ji} = a_{ji}$$

Coversely, let $A = [a_{ij}]$ be an additive pairwise comparison matrix, define $b_{ij} = e^{a_{ij}}$. Then we have

$$b_{ij} \cdot b_{ji} = e^{a_{ij}} \cdot e^{a_{ji}} = e^{a_{ij}+aji} = e^0 = 1$$

and

$$b_{ij} = e^{a_{ij}} = e^{-a_{ji}} = \frac{1}{e^{a_{ji}}} = b_{ji}^{-1}$$

□

The core concept of the pairwise ranking problem is that, given an additive pairwise comparison matrix $A \in \mathbb{R}^{n\times n}$, could we assign each $i \in [n]$ a score $s_i \in \mathbb{R}$ so that

$$A_{ij} = s_i - s_j \tag{5.1.1}$$

for all $i, j \in [n]$.

In other words, could we find a function $s : [n] \to \mathbb{R}$ so that $\sum_{i,j=1}^{n} |A_{ij} - \big(s(i) - s(j)\big)| = 0$. Unfortunately, the equation (5.1.1) does not hold for some pairwise comparison matrix A.

DOI:10.6814/NCCU202100296

Consider the following example,

$$A = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$$

If there exists $s : [n] \to \mathbb{R}$ such that (5.1.1) hold. Then

$$1 = A_{12} = s_2 - s_1 = (s_2 - s_3) + (s_3 - s_1) = A_{32} + A_{13} = 0$$

which leads to a contradiction. That is, (5.1.1) is impossible to be satisfied for any skew-symmetric matrix X. Therefore, we should consider the least square solution of (5.1.1) instead. That is,

$$\min_{s:[n]\to\mathbb{R}} \sum_{i,j=1}^{n} |A_{ij} - \big(s(i) - s(j)\big)|^2 \tag{5.1.2}$$

Remark 5.1.3. *Note that the solution of (5.1.2) is unique up to an additive constant. That is, let s be a function that minimizes $\sum_{i,j=1}^{n} |A_{ij} - \big(s(i) - s(j)\big)|^2$. Since $s(i) - s(j) = \big(s(i) + C\big) - \big(s(j) + C\big)$, the shift of s by any $C \in$ is also a solution that minimizes $\sum_{i,j=1}^{n} |A_{ij} - \big(s(i) - s(j)\big)|^2$.*

Hence, we need to consider the minimum norm solution of (5.1.2) instead. Since this problem is a least-squares problem, the minimum norm solution can be obtained in a standard way.

Now, we consider another point of view of pairwise comparison.

Let A_E be a pairwise comparison on $[n]$, where E is the set of all 2-tuples (i, j) so that i and j are compared.

If we consider $[n]$ and E as vertex set and edge set, respectively. Then $([n], E)$ is a finite directed graph. Moreover, A_E induces a weight on E, that is, $w(i, j) = |[A_E]_{ij}|$ for $(i, j) \in E$. Therefore, there is an edge-weighted graph $([n], E, w)$ induced associated with comparison matrix A_E.

Later, we will see how to solve problem (5.1.2) using an algebraic topology based approach derived from the point of graph $([n], E, w)$.

DOI:10.6814/NCCU202100296

5.2 Combinatorial Hodge Theory and HodgeRank

Continuing from subsection 2.3.1, we introduce some more basic terminologies of graph.

Definition 5.2.1. *Let G be a graph. Denote $\binom{n}{3} = \{(i,j,k) \in V^3 \mid i,j,k \text{ are all distinct}\}$.*

- *A graph H is called a subgraph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$.*
- *A nonempty subgraph K_m of a undirected graph G is called a m-clique of G.*
- *The set of all 3-cliques of $G = (V, E)$ is denoted by*

$$T(E) = \{(i,j,k) \in \binom{n}{3} \mid (i,j),(j,k),(k,i) \in E\}$$

For more detail about other types of graph and the graph theory, please refer [95].

Given a pairwise comparison matrix A on $[n]$, we can associate it with a graph. Assume that A is on the additive scale, let $V = [n]$ and $E = \{(i,j) \in \binom{V}{2} \mid a_{ij} \neq 0\}$, then $G = (V, E)$ is a graph, we call it a comparison graph.

Definition 5.2.2 (Edge Flow)**.** *Let G be a graph (not necessary undirected). An edge flow on G is a function $X : V \times V \to \mathbb{R}$ so that*

$$\begin{cases} X(i,j) = -X(j,i) & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

An example of an edge flow on a graph of 5 vertices is shown in Figure 5.1.

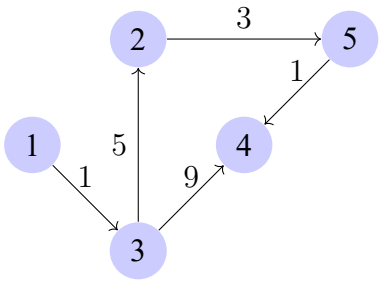


Figure 5.1: An example of edge flow

DOI:10.6814/NCCU202100296

Remark 5.2.3. *Let G be a graph on n vertices, and Y be an $n \times n$ skew-symmetric matrix. Then we can define a edge flow X on G by assigning $X(i,j) = Y_{ij}$. Therefore, the set of edge flows on G has an one-to-one correspondence to the set of $n \times n$ skew-symmetric matrices satisfying*

$$\{X \in \mathbb{R}^{n\times n} \mid X^T = -X \text{ and } X_{ij} = 0 \text{ if } (i,j) \notin E(G)\}$$

Note that, the edge flow shown in Figure 5.1 can be induced from the following skew-symmetric matrix.

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -5 & 0 & 3 \\ -1 & 5 & 0 & 9 & 0 \\ 0 & 0 & -9 & 0 & -1 \\ 0 & -3 & 0 & 1 & 0 \end{bmatrix}$$

We start from the definition of abstract simplicial complex.

Definition 5.2.4. *A collection of K of finite sets is called an (abstract) simplicial complex if $\sigma \in K$ implies that $\tau \in K$ for all $\tau \subseteq \sigma$.*

- *A element $\sigma \in K$ is called a simplex.*
- *The dimension of $\sigma \in K$ is defined as $|K| - 1$. A σ of dimension k is called a k-simplex.*
- *The dimension of K is the highest dimension among all simplex contained in K.*
- *The set of all k-simplex is denoted by Σ_k.*

Example 5.2.5. *Let $G = (V, E)$ be a finite graph on n vertices.*

- *V is a 0-simplex if we identify each element $i \in V$ with $\{i\}$.*
- *E is a 1-simplex.*
- *Hence, $V \cup E$ is a simplicial complex of dimension 1.*

DOI:10.6814/NCCU202100296

Definition 5.2.6. *A function $f : V^{k+1} \to \mathbb{R}$ is called a k-dimensional cochain (or a k-cochain) on K if it satisfies*

1. *f is alternating on Σ_k. That is,*

$$f\big(i_{\sigma(0)}, i_{\sigma(1)}, \cdots, i_{\sigma(k)}\big) = \text{sign}(\sigma) \cdot f(i_0, i_1, \cdots, i_k)$$

   for all $(i_0, i_1, \cdots, i_k) \in \Sigma_k$ and for all permutation $\sigma \in \text{Sym}(k+1)$, the symmetric group.

2. *$f(i_0, i_1, \cdots, i_k) = 0$ if $(i_0, i_1, \cdots, i_k) \notin \Sigma_k$*

The set of all k-cochains on K is denoted by $C^k(K, \mathbb{R})$ or C^k for simplicity.

Now, we define a map between C^k and C^{k+1}.

Definition 5.2.7. *The k-th coboundary operator $\delta_k : C^k \to C^{k+1}$ is a linear map that maps a k-cochin f into a $(k+1)$-cochain $\delta_k f$ defined as follows:*

$$(\delta_k f)(i_0, i_1, \cdots, i_{k+1}) = \sum_{j=0}^{k+1} (-1)^j f(i_0, \cdots, i_{j-1}, i_{j+1}, \cdots, i_{k+1})$$

for $(i_0, i_1, \cdots, i_{k+1}) \in K$.

For example, if $f \in C^0$ and $g \in C^1$, then $(\delta_0 f)(i, j) = f(j) - f(i)$ and $(\delta_1 g)(i, j, j) = g(i, j) - g(j, k) + g(i, j)$.

For $f, g \in C^k$, we can define an inner product on C^k by summing over all values $f(\cdot)g(\cdot)$ on Σ_k. That is,

$$\langle f, g \rangle_{C^k} = \sum_{(i_0, \cdots, i_k) \in \Sigma_k} f(i_0, \cdots, i_k) g(i_0, \cdots, i_k)$$

Let $\omega : \Sigma_k \to [0, \infty)$ be a weight function on Σ_k, we can consider the weighted inner product

$$\langle f, g \rangle_{C^k, \omega} = \sum_{(i_0, \cdots, i_k) \in \Sigma_k} \omega(i_0, \cdots, i_k) f(i_0, \cdots, i_k) g(i_0, \cdots, i_k)$$

Under the existence of weight ω, we need to modify Σ_k by

$$\Sigma_k^\omega = \{(i_0, \cdots, i_k) \in \Sigma_k \mid \omega(i_0, \cdots, i_k) \neq 0\}$$

DOI:10.6814/NCCU202100296

and $f^{\omega}(i_0, \cdots, i_k) = \begin{cases} f(i_0, \cdots, i_k) & \text{if } (i_0, \cdots, i_k) \in \Sigma_k^{\omega} \\ 0 & \text{if } (i_0, \cdots, i_k) \notin \Sigma_k^{\omega} \end{cases}$ for any $f \in C^k$.

so that $\langle \cdot, \cdot \rangle_{C^k,\omega}$ is an inner product on $C^{k,\omega} = \{f^{\omega} \mid f \in C^k\}$.

From elementary linear algebra, we can define the adjoint of a linear map between inner product spaces. Hence, for each $\delta_k : C^k \to C^{k+1}$, we denote its adjoint map $\delta_k^* : C^{k+1} \to C^k$ by d_k. Then d_k satisfies

$$\langle \delta_k f, g \rangle_{C^{k+1}} = \langle f, d_k g \rangle_{C^k}$$

for any $f \in C^k$ and $g \in C^{k+1}$.

Definition 5.2.8. *d_k is called the k-th boundary operator from C^{k+1} to C^k.*

Theorem 5.2.9. *$d_{k-1} \circ d_k = 0$ and $\delta_{k+1} \circ \delta_k = 0$.*

Proof. This can be proved by direct computation. □

Definition 5.2.10. *Let K be a simplicial complex. The k-dimensional combinatorial Laplacian is the operator $\Delta_k : C^k \to C^k$ defined by*

$$\Delta_k = d_k \circ \delta_k + \delta_{k-1} \circ d_{k-1}$$

Theorem 5.2.11 (Combinatorial Hodge Theorem [51])**.** *Let K be a simplicial complex. Then the space C^k admits an orthogonal decomposition*

$$C^k(K, \mathbb{R}) = \text{im}(\delta_{k-1}) \oplus \ker(\Delta_k) \oplus \text{im}(d_k)$$

Further, $\ker(\Delta_k) = \ker(\delta_k) \cap \ker(d_{k-1})$.

With above notations, we can start to work on the comparison graph $G = (V, E)$ induced by a pairwise comparison matrix A on $[n]$. For $k = 0, 1, \cdots, n$, denote the set of all k-cliques of G by Σ_k. Then $\Sigma_G^k = \Sigma_0 \cup \Sigma_1 \cup \cdots \cup \Sigma_k$ is a simplicial complex of dimension k , called the k-clique complex. Note that $\Sigma_0 = V$ and $\Sigma_1 = E$ and $\Sigma_G^1 = V \cup E$ is a simplicial complex of dimension 1.

By an abuse of notation, we write $\Sigma_G^k = (\Sigma_0, \Sigma_1, \cdots, \Sigma_k)$. In particular, $G = \Sigma_G^1$.

With a graph G, $C^0 = \mathcal{F}(V, \mathbb{R}) \cong \mathbb{R}^n$, $C^1 = \mathcal{A} = \{X \in \mathbb{R}^{n \times n} \mid X^T = -X\}$ is the set of all skew-symmetric matrices of order n. Let $T(E) = \{(i, j, k) \in \binom{V}{3} \mid (i, j), (j, k), (k, i) \in E\}$.

DOI:10.6814/NCCU202100296

Definition 5.2.12. *[51] Let $G = (V, E, T(E))$ be a 2-dimensional simplicial complex.*

1. *The combinatorial gradient operator* grad : $\mathcal{F}(V, \mathbb{R}) \to \mathcal{A}$ *is defined by*

$$\text{grad}(s)(i, j) = s_j - s_i.$$

for $i, j \in V$, then grad *is a linear map from C^0 to C^1.*

2. *The image of $\mathcal{F}(V, \mathbb{R})$ under* grad *is denoted by*

$$\mathcal{M}_G = \{X \in \mathcal{A} \mid X_{ij} = s_i - s_j \text{ for some } s : V \to \mathbb{R}\}$$

3. *The combinatorial curl operator* curl : $\mathcal{A} \to C^2$ *is defined by*

$$(\text{curl } X)(i, j, k) = \begin{cases} X_{ij} + X_{jk} + X_{ki} & \text{if } (i, j, k) \in T(E) \\ 0 & \text{otherwise} \end{cases}$$

for $(i, j, k) \in \Sigma_3$, then curl *is a linear map from $\mathcal{A}$ to C^2.*

4. *The set of T-consistent matrices is denoted by*

$$\mathcal{M}_T = \{X \in \mathcal{A} \mid X_{ik} + X_{jk} + X_{ki} = 0 \text{ for all } (i, j, k) \in T\}$$

Then $\ker(\text{curl}) = \mathcal{M}_T$ *and* $\mathcal{M}_G \subseteq \mathcal{M}_T \subseteq \mathcal{A}$

Consider the case that $K = K_G$ for an undirected graph $G = (V, E)$ and $k = 1$, then Theorem 5.2.11 becomes:

Theorem 5.2.13 (Helmholtz Decomposition Theorem [51])**.** *The space $C^1(K_G, \mathbb{R})$ admits an orthogonal decomposition*

$$C^1(K_G, \mathbb{R}) = \text{im}(\text{grad}) \oplus \ker(\Delta_1) \oplus \text{im}(\text{curl}^*)$$

Further, $\ker(\Delta_1) = \ker(\text{curl}) \cap \ker(\text{div})$.

Back to the story of pairwise ranking problem, if we consider $X \in \mathcal{M}_G$ in equation (5.1.1), then there exists $s : V \to \mathbb{R}$ so that (5.1.1) holds.

DOI:10.6814/NCCU202100296

Definition 5.2.14. *(Consistency) [51] A pairwise ranking matrix X on $G = (V, E)$ is called*

1. *consistency on $(i, j, k) \in \binom{V}{3}$ if $(i, j, k) \in T(E)$ and $X \in \mathcal{M}_T$*
2. *globally consistency if $X = \text{grad}(s)$ for some $s \in \mathcal{F}(V, \mathbb{R})$*
3. *locally consistency if* curl $X \equiv 0$.
4. *a cyclic ranking if there exists $i, j, k, \cdots, p, q \in V$ so that*

$$X_{ij} + X_{jk} + \cdots X_{pq} + X_{qi} \neq 0$$

Note that if X is globally consistency, then X is consistency on any 3-clique $(i, j, k) \in T(E)$. Now, consider the weighted trace induced by w. i.e.,

$$\langle X, Y \rangle_w = \sum_{(i,j)\in E} w_{ij} X_{ij} Y_{ij} = \text{tr}\left(X^T (W \odot Y)\right)$$

for $X, Y \in \mathcal{A}$, where $\odot$ represents the Hadamard product or elementwise product.

With respect to a weighted inner product, we obtain two orthogonal complement of $\mathcal{A}$

$$\mathcal{A} = \mathcal{M}_G \oplus \mathcal{M}_G^{\perp} = \mathcal{M}_T \oplus \mathcal{M}_T^{\perp}$$

where $\mathcal{M}_*^{\perp} = \{X \in \mathbb{R}^{n\times n} \mid \langle X, Y \rangle_W = 0 \text{ for all } Y \in \mathcal{M}_*^{\perp}\}$, $* \in \{G, T\}$.

By a simple fact of orthogonal complement, since $\mathcal{M}_G \subseteq \mathcal{M}_T$, we have $\mathcal{M}_G^{\perp} \supseteq \mathcal{M}_T^{\perp}$ and we can get further orthogonal direct sum decomposition of $\mathcal{A}$ as follows:

$$\mathcal{A} = \mathcal{M}_G \oplus \mathcal{M}_H \oplus \mathcal{M}_T^{\perp},$$

where $\mathcal{M}_H = \mathcal{M}_T \cap \mathcal{M}_G^{\perp}$.

This decomposition is called the combinatorial Hodge decomposition. For more detail about the theory of combinatorial Hodge decomposition, please refer [51] for more detail.

With notations above, we can reformulate problem 5.1.2 into

$$\min_{Y_{ij}\in\mathcal{M}_G} \sum_{i,j=1}^{n} w_{ij} |A_{ij} - Y_{ij}|^2 \tag{5.2.1}$$

DOI:10.6814/NCCU202100296

We now state one useful theorem in [51].

Theorem 5.2.15. [51] Let $G = (V, E, w)$ be an edge-weighted graph induced by a pairwise comparison matrix A. Then

1. The minimum norm solution s of (5.1.2) is the solution of the normal equation:

$$\Delta_0 s = -\operatorname{div}(A),$$

where

$$\Delta_0(i,j) = \begin{cases} \sum_{k:(i,k)\in E} w_{ik} & \text{if } i = j \\ -w_{ij} & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{5.2.2}$$

and

$$\operatorname{div}(A)(i) = \sum_{j:(i,j)\in E} w_{ij} A_{ij}$$

is the combinatorial divergence operator of A.

2. The minimum norm solution s of (5.1.2) is

$$s^* = -\Delta_0^\dagger \operatorname{div}(A), \tag{5.2.3}$$

where $\Delta_0^\dagger$ represents the Moore-Penrose pseudo inverse of the matrix Δ_0.

The Hodge decomposition indicates the solution of $(5.1.2)$, while Theorem 5.2.15 shows that such solution can be calculated by solving the normal equation. This solution is called the HodgeRank on the pairwise comparison graph $G = (V, E, w)$.

Definition 5.2.16. *Let G be a pairwise comparison graph. Then the minimum norm solution (5.2.3) of the minimization problem (5.1.2) is called the HodgeRank of G.*

In fact, Δ_0 is the (unnormalized) graph Laplacian matrix of the graph induced by the comparison matrix A.

Note that a pairwise comparison graph corresponds to a skew-symmetric matrix. Hence, given an $n \times n$ skew-symmetric matrix X, HodgeRank returns the minimum norm solution

DOI:10.6814/NCCU202100296

$s \in \mathbb{R}^n$ of the optimization problem (5.1.2). This point of view makes HodgeRank a function defined from the set of all $n \times n$ skew-symmetric matrices to $\mathbb{R}^n$. To study the HodgeRank in this sense, we need more terminologies about the graph Laplacian and the generalized inverse.

DOI:10.6814/NCCU202100296

Chapter 6

On Continuity of the HodgeRank

In previous chapter, we mentioned that HodgeRank can be viewed as a function defined from the set of all $n \times n$ skew-symmetric matrices to $\mathbb{R}^n$. In this chapter, we will see how to study the continuity of the HodgeRank in this point of view.

We first recall the definition of the graph Laplacian matrix and some of its properties. Also, we will see the Moore-Penrose pseudo inverse of a matrix and how to compute it by singular value decomposition.

6.1 Graph Laplacian and Generalized Inverse

Definition 6.1.1. *Let G be a simple undirected graph on n vertices with adjacent matrix A. Denote $\deg_i$ be the degree of vertex i and define the diagonal matrix $D = \mathrm{diag}(\deg_1, \deg_2, \cdots, \deg_n)$. Then the (unnormalized) graph Laplacian of G is defined by*

$$L = D - A$$

Theorem 6.1.2. *Let L be the graph Laplacian matrix of a graph G. Then we have:*

1. *L is symmetric and positive-semidefinite.*
2. *Every row sum and column sum of L is zero.*
3. *The vector with all 1 is an eigenvector of L corresponding to eigenvalue 0.*

DOI:10.6814/NCCU202100296

4. *The dimension of the null space of L is the number of connected components of G. In particular, if G is connected, then* $\text{rank}(L) = n - 1$.

5. $\text{tr}(L) = 2|E|$.

Definition 6.1.3. *Let A be an $n \times m$ matrix. Consider the following four matrix equations:*

1. $ABA = A$

2. $BAB = B$

3. $(AB)^* = AB$

4. $(BA)^* = BA$

A $m \times n$ matrix $A^\dagger$ is called

- *a generalized inverse of A if it satisfies first equation.*
- *a reflexive generalized inverse of A if it satisfies first two equations.*
- *the Moore-Penrose pseudo inverse of A if it satisfies all four equations.*

To see how to compute of the Moore-Penrose pseudo inverse of a matrix, we first consider simplest case.

First, the Moore-Penrose pseudo inverse of a scalar a (not necessary real) is

$$a^\dagger = \begin{cases} a^{-1} & \text{if } a \neq 0 \\ 0 & \text{if } a = 0 \end{cases}$$

Then, we consider matrix of the form

$$D = \text{diag}(d_1, d_2, \cdots, d_n)_{n \times m} \tag{6.1.1}$$

Then, the Moore-Penrose pseudo inverse of D is

$$D^\dagger = \text{diag}(d_1^\dagger, d_2^\dagger, \cdots, d_n^\dagger)_{m \times n}$$

DOI:10.6814/NCCU202100296

Now, for general $n \times m$ matrix A, we first consider its singular value decomposition

$$A = U\Sigma V^*$$

where Σ is of the form (6.1.1) and U^* is the Hermitian transpose of U.

The Moore-Penrose pseudo inverse of A can be expressed as

$$A^\dagger = V\Sigma^\dagger U^*$$

One nice property is that Moore-Penrose pseudo inverse preserves the rank.

Theorem 6.1.4. *The matrix rank of A and $A^\dagger$ are the same.*

Proof. This is a direct consequence from the singular value decomposition. □

There is another type of generalized inverse called the Drazin inverse. Drazin inverse are discussed in the algebraic sense, where fruitful tools from the theory of C^*-algebra are introduced to study its properties. For more detail about Drazin inverse, please refer [26].

6.2 HodgeRank as a Composition Function

Now, given an additive pairwise comparison data X on n vertices, we can associate it with an edge-weighted graph $G = (V, E, w)$. We recall how to construct such graph G below.

Note that X can be extended into an $n \times n$ matrix. Set $V = [n] = \{1, 2, \cdots, n\}$, $E = \{(i, j) \in \binom{V}{2} \mid X_{ij} \neq 0\}$. Define $w : E \to [0, \infty)$ by $w_{ij} = |X_{ij}|$, then $G_X = (V, E, w)$ is an dege-weighted graph on n vertices.

HodgeRank returns a global ranking $s^* : V \to \mathbb{R}$ so that s^* is the minimizer of the least square problem

$$\min_{s:V\to\mathbb{R}} \sum_{(i,j)\in E} |X_{ij} - \big(s(i) - s(j)\big)|^2$$

whose 2-norm is minimized. The property of minimum 2-norm is guaranteed since s^* is computed by solving a graph Laplacian problem.

This s^* is called the HodgeRank of the pairwise comparison data X. In fact, to get s^* from the pairwise comparison data X, we have the following process:

DOI:10.6814/NCCU202100296

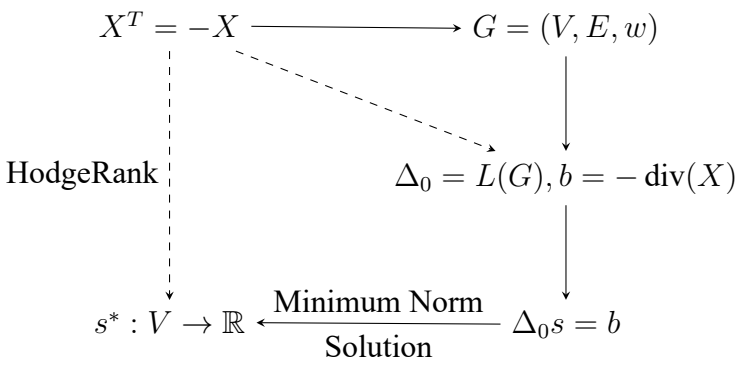
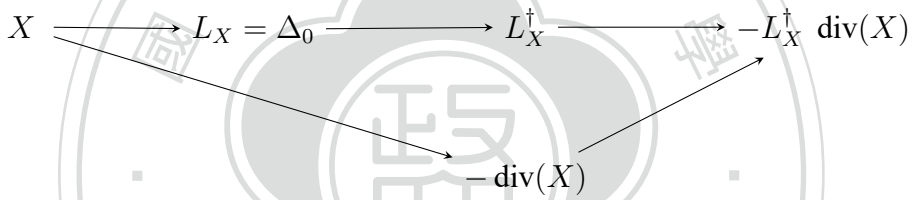


Figure 6.1: HodgeRank from Pairwise Comparison Data

Note that $-\operatorname{div}(X)$ is linear since $\operatorname{div}(X) = X \cdot \mathbf{1}$, where $\mathbf{1} \in \mathbb{R}^n$ is the vector of all ones. By viewing $X \mapsto s^*$ as a function from $\mathcal{A}$ to $\mathcal{F}(V, \mathbb{R}) \cong \mathbb{R}^{|V|}$, we can consider HodgeRank as a composition function, in fact, from any X with $X^T = -X$, HodgeRank of X can be obtained as follows:



where L_X is the graph Laplacian matrix associated with the weighted graph induced by X, and $\dagger$ represents the Moore-Penrose pseudoinverse operator of a matrix.

A natural question is whether the function HodgeRank is continuous in the following sense: Let $X_k \to X$ be a convergent sequence in $\mathcal{A}$ in the sense of maximum norm or Frobenius norm of $\mathbb{R}^{n \times n}$. Then HodgeRank of X_k converges to the HodgeRank of X under the maximum norm of $\mathbb{R}^n$.

As we mentioned above, div is a linear function, hence, it is continuous. The continuity of

$$\left(L_X^\dagger, -\operatorname{div}(X)\right) \mapsto -L_X^\dagger \operatorname{div}(X)$$

is obvious since it is a matrix-vector product. The remain questions are whether $X \mapsto L_X$ and $X \mapsto X^\dagger$ are continuous or not. We would wonder, under what conditions, they are continuous function. We will answer these questions in next two sections.

DOI:10.6814/NCCU202100296

6.2.1 Continuity of $X \mapsto L_X$

Recall that the graph Laplacian matrix L_X induced by a pairwise comparison X is defined based on the degree of each vertex as follows:

$$L_X(i,j) = \begin{cases} \sum\limits_{k:(i,k)\in E} w_{ik} & \text{if } i = j \\ -w_{ij} & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Note that $w_{ij} = \text{sign}(|X_{ij}|)$, hence, the continuity of $X \mapsto L_X$ is determined by the continuity of sign function. Note that the only discontinuity point of sign is 0.

Now, let $\{X_k\}$ be a sequence of comparison matrices in $\mathcal{A}$ which converges to X in any matrix-norm. Denote G_{X_k} be the graph induced by X_k. If there exists $N \in \mathbb{N}$ so that

$$E(G_{X_k}) = E(G_X) \tag{6.2.1}$$

for $k \geq N$, then sign is a continuous function on each (i,j) entry, therefore, $X \mapsto L_X$ is continuous in this sense.

If we write $X_k = [X_{ij}^{(k)}]$, then condition (6.2.1) is equivalent to

$$\text{sign}(X_{ij}^{(k)}) = \text{sign}(X_{ij}) \text{ for all } (i,j) \tag{6.2.2}$$

since $E(G_{X_k}) = \{(i,j) \in \binom{V}{2} \mid X_{ij}^{(k)} \neq 0\}$ and $X_{ij}^{(k)} \to X_{ij}$ as $k \to \infty$ for all (i,j).

6.2.2 Continuity of the Pseudoinverse Operator

Let X be an $n \times m$ matrix over $\mathbb{R}$. We can find its Moore-Penrose pseudoinverse $X^\dagger$ by using the singular value decomposition. A natural question arises: is the map $X \in \mathbb{R}^{n\times m} \mapsto X^\dagger \in \mathbb{R}^{m\times n}$ a continuous function? In other words, does

$$\big(\lim_{k\to\infty} X_k\big)^\dagger = \big(\lim_{k\to\infty} X_k^\dagger\big)$$

hold when the convergence of $X_k \to X$ is in the sense of maximum norm or Frobenius norm?

DOI:10.6814/NCCU202100296

In general, this is not true. Consider the following example:

Example 6.2.1. *Let* $X_\varepsilon = \begin{bmatrix} 1 & 0 \\ 0 & \varepsilon \end{bmatrix}$ *and* $X = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ *Then* $X_\varepsilon \to X$ *as* $\varepsilon \to 0$.

However, the pseudoinverse of X_ε *is*

$$X_\varepsilon^\dagger = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{\varepsilon} \end{bmatrix}$$

In this example, the limit $\lim_{\varepsilon \to 0} X_\varepsilon^\dagger$ does not exists. That is, $X_k \to X$ does not guarantee the convergence of $X_k^\dagger \to X^\dagger$. That is, the pseudoinverse operator is not continuous.

However, a well-known result proved in [85] says that the necessary and sufficient condition of the continuity of † operator

Theorem 6.2.2 ([85])**.** *Let* $\{X_k\}$ *be a sequence of matrices so that* $X_k \to X$ *under the maximum norm or Frobenius norm. Then* $X_k^\dagger \to X^\dagger$*, if and only if, there exists* $N \in \mathbb{N}$ *so that*

$$\text{rank}(X_k) = \text{rank}(X) \text{ for } k \geq N. \tag{6.2.3}$$

In Example 6.2.1, $X_\varepsilon^\dagger \not\to (\lim_{\varepsilon \to 0} X_\varepsilon)^\dagger$ since $\text{rank}(X) = 1$ and $\text{rank}(X_\varepsilon) = 2$ for any $\varepsilon \neq 0$.

Hence, when discussing the continuity of the pseudoinverse operator † defined on $\mathbb{R}^{n \times n}$, the set of all $n \times n$ matrices, we must restrict the domain as a set of matrices of constant rank.

Based on above discussion, we have the following consequence about sufficient condition of the continuity of the HodgeRank on $\mathcal{A}$.

Theorem 6.2.3. *Let* $\{X_k\}$ *be a sequence in* $\mathcal{A}$ *which converges to a matrix* $X \in \mathcal{A}$*. Then condition (6.2.1) implies the condition (6.2.3).*

Proof. Since the underlying graphs of X_k and X have the same edge set for k large enough, then X_k and X have the same number of connected components. By Theorem 6.1.2, the nullity of L_{X_k} and L_X are the same. By dimension theorem, for $n \geq N$, $\text{rank}(L_{X_k}) = \text{rank}(L_X)$. □

Corollary 6.2.4. *Let* $\{X_k\}$ *be a sequence in* $\mathcal{A}$ *which converges to a matrix* $X \in \mathcal{A}$*. If the condition (6.2.1) holds for* $\{X_k\}$*, then HodgeRank is continuous at* X.

DOI:10.6814/NCCU202100296

6.3 Class of Graph Laplacian Matrices

Moore-Penrose pseudoinverse of the graph Laplacian matrix is not only necessary in the computation of the HodgeRank, it also plays an important role in spectral graph theory [16, 46] and machine learning [44, 81]. Hence, we consider a more general condition other than HodgeRank in this section.

Let $\mathcal{G}_n = \{L \in \mathbb{R}^{n\times n} \mid L = L(G) \text{ for some graph } G \text{ of } n \text{ vertices}\}$. Then the pseudoinverse operator $\dagger : \mathcal{G}_n \to \mathbb{R}^{n\times n}$ is well-defined.

In fact, we can partition $\mathcal{G}_n$ by matrix rank, we have

$$\mathcal{G}_n = \bigcup_{k=0}^{n-1} \{L \in \mathcal{G}_n \mid \text{rank}(L) = k\}$$

To understand the behavior of $\dagger$ on $\mathcal{G}_n$, we must study how $\dagger$ behaves on each $\{L \in \mathcal{G}_n \mid \text{rank}(L) = k\}$. For convenience, we can relax each component into $\mathbb{R}^{n\times n}$. This extends the graph Laplacain matrix into the case of edge-weighted graph.

For fixed n, denote $\mathcal{G}_{n,k} = \{L \in \mathbb{R}^{n\times n} \mid \text{rank}(L) = k\}$ for $0 \leq k \leq n-1$. By Theorem 6.2.2, we have $\dagger$ is continuous on each $\mathcal{G}_{n,k}$. By Theorem 6.1.4, L and $L^\dagger$ are of the same rank, hence, $\dagger$ is an operator on $\mathcal{G}_{n,k}$.

A natural question arises, does there exist a graph attention neural network (or a general graph neural network) so that if we input L into such graph neural network, then it outputs $L^\dagger$? Moreover, will the i-th column of L be mapped to the i-th column of $L^\dagger$ for all $L \in \mathcal{G}_{n,k}$?

6.4 Perspective from Grassmannian Manifold

Now, we consider a more general case to see whether the previous question can be solved or not.

For each $L \in \mathcal{G}_{n,k}$, there exists k columns of L which are independent. Each L associates a k-dimensional subspace of $\mathbb{R}^n$.

Therefore, there is an map $\pi : \mathcal{G}_{n,k} \to \text{Gr}(k, n)$ by sending $L \in \mathcal{G}_{n,k}$ to the k-dimensional subspace generated by columns of L, where $\text{Gr}(k, n)$ is the space of all k-dimensional subspace of $\mathbb{R}^n$, called the Grassmannian manifold. There are some advantages to treat a graph Laplacian matrix as a point on a Grassmannian manifold. The first one is one can introduce the theory

DOI:10.6814/NCCU202100296

of Riemannian manifold to deal with such problem. Besides, topological properties of $\mathrm{Gr}(k,n)$ are well-studied.

Theorem 6.4.1 ([69, p. 57-59])**.** $\mathrm{Gr}(k,n)$ *is a Hausdorff, compact, connected smooth manifold of dimension* $k(n-k)$.

However, one disadvantage is that two matrices of the same rank may generate the same subspace of $\mathbb{R}^n$. The pseudoinverse operator † may not be well-defined in this case. Even if we can modify † so that it is well-defined on $\mathcal{G}_{n,k}$, properties of † on $\mathrm{Gr}(k,n)$ may not hold on $\mathcal{G}_{n,k}$ under π. Therefore, we leave above issues as our future work to extend the HodgeRank as a function of the graph Laplacian matrix into an operator on a Grassmannian manifold.

6.5 Conclusion

In this chapter, we review some knowledge of the generalized inverse. Also, by viewing HodgeRank as a function of skew-symmetric matrix, we discuss the continuity of the HodgeRank. A theorem for the continuity of the HodgeRank is provided. In the end of chapter, some ongoing works that the notation of generalized inverse † is extended to an operator on the Grassmannian is introduced. This demonstrates the connection of graph Laplacian and the graph neural network. In next chapter, we will see an application of the HodgeRank to a real world problem.

DOI:10.6814/NCCU202100296

Chapter 7

Online Peer Assessment Problem

This chapter is based on the publication [60]. In this work, we apply HodgeRank to deal with an online peer assessment problem to eliminate the bias and heterogeneity.

7.1 Introduction

Bias and heterogeneity in peer assessment can lead to the issue of unfair scoring in the educational field. To deal with this problem, we propose a reference ranking method for an online peer assessment system using HodgeRank.

A peer assessment system is used to enhance students' learning process, especially in higher education. Through such a system, students are given the opportunity to not only learn knowledge from textbooks and instructors, but also from the process of making judgements on assignments completed by their peers. This process helps them understand the weaknesses and strengths in the work of others, and then to review their own.

However, there are some practical issues associated with a peer assignment system. For example, students tend to give significantly higher grades than senior graders or professionals (see [35] for more details). Also, students have a tendency to give grades within a range, with the center of such a range often being based on the first grade they gave. Therefore, bias and heterogeneity can occur in a peer assessment system.

There are various ranking methods on peer assessment problem, such as PeerRank [93] and Borda-like aggregation algorithm [11]. PeerRank, a famous method based on a iterative process to solve the fixed-point equation. PeerRank has many interesting properties from the

DOI:10.6814/NCCU202100296

view of linear algebra. Borda-like aggregation algorithm, a random method based on the theory of random graphs and voting theory, which provides some probabilistic explanation on peer assessment problem.

We propose another ranking scheme to deal with peer assessment problems that uses HodgeRank, a statistical preference aggregation problem from pairwise comparison data. The purpose of HodgeRank is to find a global ranking system based on pairwise comparison data. HodgeRank can not only generate a ranking order, but also highlight inconsistencies in the comparisons (see [51] for more detail). We apply HodgeRank to the problems in online assessment and display ranking results from HodgeRank and PeerRank in turn.

7.2 Problem Definition

Let V represents the set of students to be ranked by their peers.

Denote Λ to be the number of assignments. Then each assignment $\alpha \in \Lambda$, for students $i, j \in V$, if their cumulative score at assignment α are X_i^α and X_j^α, respectively. Then, score different can be defined as $Y_{ij}^\alpha = X_i^\alpha - X_j^\alpha$. We also define $Y_{ij}^\alpha = 0$ i and j are not compared in assignment α. For example, $Y_{ij}^\alpha \in [-100, 100]$ on hundred-mark system.

Hence, each $\alpha \in \Lambda$ associates with an edge-weighted graph $G_\alpha = (V, E_\alpha, Y^\alpha)$ of assignment α, where $(i, j) \in E_\alpha$ if students i and j are compared at assignment α .

For each comparison $(i, j) \in E_\alpha$, we consider a weight w^α be the number of comparison between student i and j in assignment α. That is, $w_\alpha(i, j) = 0$ if $(i, j) \notin E_\alpha$.

Define $E = \bigcup_{\alpha\in\Lambda} E_\alpha$, $Y = \sum_{\alpha\in\Lambda} Y^\alpha$ and $w = \sum_{\alpha\in\Lambda} w^\alpha$ then we can get a pairwise comparison graph $G = (V, E, Y)$.

The goal of the HodgeRank on G is to find a skew-symmetric $X \in \mathcal{M}_G$ which is the minimizer of the following optimization problem:

$$\min_{X\in\mathcal{M}_G} \sum_{(i,j)\in E} w_{ij}(X_{ij} - Y_{ij})^2 \tag{7.2.1}$$

The minimizer X^* of the problem 7.2.1 can be represented as $X_{ij}^* = \text{grad}(s^*)(i, j)$ for some $s^* : V \to \mathbb{R}$. Such s^* is the HodgeRank of G, which provides a reference score which is compatible with the score different with minimum error.

DOI:10.6814/NCCU202100296

7.3 Data Description

As previously mentioned, bias and heterogeneity can lead to unfair scoring in online peer assessments. Students usually grade other students based on the first score they gave, which causes bias. However, since scores are usually compared with others, we can use this comparison behavior to reconstruct true ranking.

The data we used in this section were collected from an undergraduate calculus course. In this course, 133 students were asked to upload their GeoGebra [47] assignments. Each student was then asked to review five randomly chosen assignments completed by their peers to receive partial credits in return. There are 13 assignments during one semester.

For each assignment $m \in \Lambda = [13]$, each student $k \in V$ is asked to evaluate homework of at most 5 other students. Let $\mathcal{N}_k^m$ be the set of students be graded by k at assignment m. Then $\binom{\mathcal{N}_k^\alpha}{2}$ forms a subset of $V \times V$. For $n \in [13]$, define $E_n = \bigcup_{m=1}^{n} \binom{\mathcal{N}_k^\alpha}{2}$, then $G_n = (V, E_n)$ forms a graph. Set $G = G_{13}$.

One key point of the HodgeRank is the connectedness of the graph G generated by pairwise comparison data. Note that $E_n \subseteq E_{n+1}$ for $1 \leq n \leq 12$. Hence, If G_n is a connected graph, then so is G_m for all $m \geq n$.

In Table 7.1, we compute the number of connected components of each G_n to see when G_n is connected. We can easily find that after half the semester passed (that is, $n \geq 7$), comparison data between students forms a connected graph. Hence, we can apply HodgeRank to calculate the ranking of all the students after assignment $n = 7$.

Table 7.1: Number of components of G_n for $n \in [13]$

n	1	2	3	4	5	6	$7 \sim 13$
# of components in G_n	21	5	4	3	2	2	1

7.4 Numerical Results

The traditional method for finalizing peer assessment consists of either using an average cumulative score or a truncated average score. Although these approaches might have their own statistical meaning, they cannot avoid bias and heterogeneity in peer assessment.

We first state some ranking methods below

DOI:10.6814/NCCU202100296

Definition 7.4.1. *For $i \in V$, $\alpha \in \Lambda$, let X_i^α be the grade of student i obtained in assignment α. Then the weighted cumulative score of student i is given by*

$$X_i = \sum_{\alpha \in \Lambda} w^\alpha X_i^\alpha$$

where $w^\alpha \geq 0$ is the weight among all assignments so that $\sum_{\alpha \in \Lambda} w^\alpha = 1$.

The weighted cumulative score is irrelevant to the structure of peer assessment. However, there is another ranking method based on the peer assessment, called the PeerRank. The PeerRank is to iterate the cumulative score to find a fixed point of the iteration process.

Definition 7.4.2 (PeerRank [93])**.** *For $i, j \in V$, define $A_{ij} \in [0, 1]$ be the normalized grade of i given by j. For $\alpha \in (0, 1)$, we consider the following iteration process:*

Let $X^0 \in \mathbb{R}^n$ defined by $X_i^0 = \frac{1}{n} \sum_{j=1}^{n} A_{ij}$. Then, for $k \geq 0$,

$$X^{k+1} = (1 - \alpha)X + \frac{\alpha}{\|X_k\|_1} AX^k$$

The sequence $\{X^k\}$ converges to $X^* \in [0, 1]^n$ as $k \to \infty$, X^* is called the PeerRank.

Note that PeerRank exists and are the same for any $\alpha \in (0, 1)$, so we consider a finite iteration X^k of the PeerRank instead of X^*

In practice, we can consider $A_{ij} = \sum_{\alpha \in \Lambda} w^\alpha A_{ij}^\alpha$, where A_{ij}^α is the grade of student i given by j in assignment α.

Below, we compute the rankings based on the cumulative score, PeerRank under different α and the HodgeRank. Figure 7.1 displays the cumulative score, PeerRank with $\alpha = \{0.25, 0.5, 0.75\}$ and the HodgeRank, respectively. For each α, we compute 15 iterations instead.

We normalize each ranking to the interval $[0, 1]$ linearly and sorted in ascending order to compare ranking from different methods. In addition, to reveal the tendency of each ranking method, a steady line was plotted on the graph. There are some interesting implications that can be observed from this figure.

First, the cumulative score offers a ranking higher than the steady line. This reflects the existence of bias and heterogeneity in the cumulative average method. Second, PeerRank can be viewed as a modification of the average scoring. Third, sorted ranking result from HodgeRank

DOI:10.6814/NCCU202100296

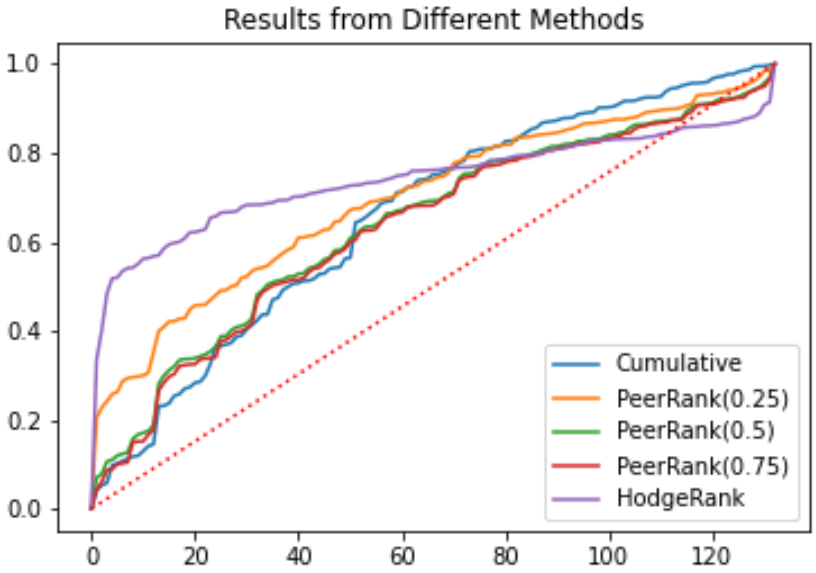


Figure 7.1: Final results using different ranking methods

is a normal distributed curve. This result can might be an explanation why HodgeRank can be solution to eliminate bias and heterogeneity by the normality.

Note that the reason why HodgeRank and PeerRank provides different results is that their conclusion base are totally different, while former method relies on the pairwise comparison data and latter one is applied on the average score as an initial ranking. Hence, HodgeRank provides instructors with an objective scoring reference using score difference rather than cumulative or average score.

In conclusion, this is the first time HodgeRank has been applied in the field of education. While numerical results were processed using real world data in this study, certain issues, such as how to aggregate the HodgeRank ranking method into a peer assessment system, remain unsolved. This task will be attempted as part of our future work.

DOI:10.6814/NCCU202100296

Chapter 8

Conclusions and Future Works

8.1 Conclusions

In this dissertation, we review background theory of deep learning and HodgeRank. These theories can be applied into some real world problems. Below, we review of each chapter which apply these theories in our work.

In chapter 2, we recall some basic definitions of neural networks including universal approximation theorems of depth-bounded type and of width-bounded type. To connect the generalized inverse operator †, we also introduce the graph neural network along with the attention mechanism.

In chapter 3, we see how neural network models can be applied to solve a moving boundary problem, by reformulating such problem, we can apply neural network as a numerical solution of the certain problem. In our numerical test, the analytical solution of the free boundary problem is well approximated by our neural network solution. This shows the capacity of the neural network models as numerical solver.

In chapter 4, we review manifold reconstruction algorithms proposed in [30]. With properties of affine projections, such manifold interpolation algorithm can be modified as a learning process of a residual network. Hence, we propose our deep learning version manifold interpolation algorithm by considering properties of affine projections into the objective function.

If we view HodgeRank which is introduced in chapter 5 as a function by sending a skew-symmetric matrix into a global ranking, then we can talk about its continuity. This issue is

DOI:10.6814/NCCU202100296

studied in chapter 6. Also, a potential linkage between graph and manifold is provided but without further result yet. To see how HodgeRank can be applied in practice, in chapter 7, we consider a peer assessment problem we faced in real world. By treating peer assessment as a pairwise comparison data. Then HodgeRank provides an alternative reference ranking generated based on the combinatorial Hodge theory.

In the end of this dissertation, we propose some unsolved problems which are close to our research above.

8.2 Unsolved Problems and Future works

8.2.1 Dimension of the Reconstructed Manifold

Note that manifold interpolation algorithm is guaranteed theoretically. However, some criteria should be checked beforehand. These practical issues may cause unexpected issue when reconstructing the desired manifold.

We first state some issues about assumptions of the manifold interpolation algorithm below. Let $X \subseteq \mathbb{R}^m$ be finite or countable with $\text{diam}(X) < \infty$. For $(\delta, n, r) \in (0, r) \times [m-1] \times (0, \text{diam}(X))$, we use the notation $X \in P(\delta, n, r)$ if X is δ-close to n-flats at scale r.

The following questions are essential to determine the dimension of the reconstructed manifold.

1. How to check if $X \in P(\delta, n, r)$? Also, is the set $\{(\delta, n, r) \mid X \in P(\delta, n, r)\}$ non-empty?
2. For each $n \in [m-1]$, how to check if $\{(\delta, r) \mid X \in P(\delta, n, r)\} \neq \phi$?
3. Given $r > 0, n < m$, is $\{(\delta, n) \in (0, r) \times [m-1] \mid X \in P(\delta, n, r)\}$ non-empty?
4. Given $\delta > 0, n < m$, is $\{(r, n) \in (\delta, \text{diam}(X)) \times [m-1] \mid X \in P(\delta, n, r)\}$ non-empty?
5. Given $0 < \delta < r$, is $\{n \in [m-1] \mid X \in P(\delta, n, r)\}$ non-empty? If so, how to compute

$$\min\{n \in [m-1] \mid X \in P(\delta, n, r)\}?$$

If we expect to interpolate a data cloud by a manifold using the interpolation algorithm, then above issues should be concerned before we process the algorithm. In fact, let X be a data

DOI:10.6814/NCCU202100296

cloud in $\mathbb{R}^m$ and write $X = \bigcup_{k=1}^{N} X_k$. If the last question can be answered, then we can associate for each X_k a dimension $n_k \in [m-1]$ so that $X_k \in P(\delta, n_k, r)$. Therefore, the dimension of the resulting manifold $\mathcal{M}$ is $\max_{1 \le k \le N} n_k$

8.2.2 HodgeRank and Graph Neural Network

There are two questions about HodgeRank. The first one whether HodgeRank is that whether it can be approximated by a neural network (especially, a graph-type neural network) or not. If this can be proved, then we can generate a local graph Laplacian solver accordingly. This can largely reduce the computational cost of the traditional solver for linear systems.

Besides, if one can extend such problem into Grassmannian, then deep learning techniques can be applied as a tool for Riemannian optimization. This may provide the insight of the deep learning into the optimization on differential geometry and vice versa.

DOI:10.6814/NCCU202100296

Bibliography

[1] D. Bahdanau, K. H. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, in 3rd International Conference on Learning Representations, ICLR 2015, 2015.

[2] S. Belciug and F. Gorunescu, *Learning a single-hidden layer feedforward neural network using a rank correlation-based strategy with application to high dimensional gene expression and proteomic spectra datasets in cancer detection*, Journal of biomedical informatics, 83 (2018), pp. 159–166.

[3] Y. Bengio, *Learning deep architectures for AI*, Now Publishers Inc, 2009.

[4] J.-D. Boissonnat and A. Ghosh, *Manifold reconstruction using tangential delaunay complexes*, Discrete & Computational Geometry, 51 (2014), pp. 221–267.

[5] J.-D. Boissonnat, L. J. Guibas, and S. Y. Oudot, *Manifold reconstruction in arbitrary dimensions using witness complexes*, Discrete & Computational Geometry, 42 (2009), pp. 37–70.

[6] C. Bregler and S. M. Omohundro, *Nonlinear manifold learning for visual speech recognition*, in Proceedings of IEEE International Conference on Computer Vision, IEEE, 1995, pp. 494–499.

[7] H. Brezis, *Functional analysis, Sobolev spaces and partial differential equations*, Springer Science & Business Media, 2010.

[8] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., *Language models are few-shot learners*, arXiv preprint arXiv:2005.14165, (2020).

DOI:10.6814/NCCU202100296

[9] D. Burago, Y. Burago, and S. Ivanov, *A course in metric geometry*, vol. 33, American Mathematical Soc., 2001.

[10] J. Cao and Z. Lin, *Extreme learning machines on high dimensional and large data applications: a survey*, Mathematical Problems in Engineering, (2015).

[11] I. Caragiannis, G. A. Krimpas, and A. A. Voudouris, *Aggregating partial rankings with applications to peer grading in massive online open courses*, in Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, 2015, pp. 675–683.

[12] M. P. d. Carmo, *Riemannian geometry*, Birkhäuser, 1992.

[13] T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, *Neural ordinary differential equations*, in Advances in Neural Information Processing Systems, 2018, pp. 6572–6583.

[14] S.-W. Cheng, T. K. Dey, and E. A. Ramos, *Manifold reconstruction from point samples.*, in SODA, vol. 5, 2005, pp. 1018–1027.

[15] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, in EMNLP, 2014.

[16] F. R. Chung and F. C. Graham, *Spectral graph theory*, no. 92, American Mathematical Soc., 1997.

[17] J. C. Cox, J. E. Ingersoll Jr, and S. A. Ross, *A theory of the term structure of interest rates*, in Theory of valuation, World Scientific, 2005, pp. 129–164.

[18] M. A. Cox and T. F. Cox, *Multidimensional scaling*, in Handbook of data visualization, Springer, 2008, pp. 315–347.

[19] G. Crawford, *The geometric mean procedure for estimating the scale of a judgement matrix*, Mathematical Modelling, 9 (1987), pp. 327–334.

[20] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Mathematics of control, signals and systems, 2 (1989), pp. 303–314.

DOI:10.6814/NCCU202100296

[21] L. Deng, G. Hinton, and B. Kingsbury, *New types of deep neural network learning for speech recognition and related applications: An overview*, in 2013 IEEE international conference on acoustics, speech and signal processing, IEEE, 2013, pp. 8599–8603.

[22] J. Detemple and Y. Kitapbayev, *On American VIX options under the generalized 3/2 and 1/2 models*, Mathematical Finance, 28 (2018), pp. 550–581.

[23] J. Detemple and C. Osakwe, *The valuation of volatility options*, Review of Finance, 4 (2000), pp. 21–50.

[24] F. Deutsch, *The angle between subspaces of a Hilbert space*, in Approximation theory, wavelets and applications, Springer, 1995, pp. 107–130.

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.

[26] M. Drazin, *Pseudo-inverses in associative rings and semigroups*, The American mathematical monthly, 65 (1958), pp. 506–514.

[27] B. Eriksson, *Learning to top-k search using pairwise comparisons*, in Artificial Intelligence and Statistics, PMLR, 2013, pp. 265–273.

[28] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, *Learning hierarchical features for scene labeling*, IEEE transactions on pattern analysis and machine intelligence, 35 (2012), pp. 1915–1929.

[29] C. Fefferman, S. Ivanov, Y. Kurylev, M. Lassas, and H. Narayanan, *Fitting a putative manifold to noisy data*, in Proceedings of COLT 2018 (Conference On Learning Theory), 2018.

[30] ——, *Reconstruction and interpolation of manifolds. I: The geometric Whitney problem*, Foundations of Computational Mathematics, (2019), pp. 1–99.

DOI:10.6814/NCCU202100296

[31] C. Fefferman, S. Ivanov, M. Lassas, and H. Narayanan, *Reconstruction of a Riemannian manifold from noisy intrinsic distances*, SIAM Journal on Mathematics of Data Science, 2 (2020), pp. 770–808.

[32] W. Feller, *Two singular diffusion problems*, Annals of mathematics, (1951), pp. 173–182.

[33] R. A. Fisher, *The use of multiple measurements in taxonomic problems*, Annals of eugenics, 7 (1936), pp. 179–188.

[34] G. B. Folland, *Real analysis: modern techniques and their applications*, vol. 40, John Wiley & Sons, 1999.

[35] S. Freeman and J. W. Parks, *How accurate is peer grading?*, CBE-Life Sciences Education, 9 (2010), pp. 482–488.

[36] F. A. Gers, J. Schmidhuber, and F. Cummins, *Learning to forget: Continual prediction with LSTM*, (1999).

[37] J. Goard and M. Mazur, *Stochastic volatility models and the pricing of VIX options*, Mathematical Finance: An International Journal of Mathematics, Statistics and Financial Economics, 23 (2013), pp. 439–458.

[38] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1, MIT press Cambridge, 2016.

[39] A. Graves, A.-r. Mohamed, and G. Hinton, *Speech recognition with deep recurrent neural networks*, in 2013 IEEE international conference on acoustics, speech and signal processing, Ieee, 2013, pp. 6645–6649.

[40] A. Grünbichler and F. A. Longstaff, *Valuing futures and options on volatility*, Journal of Banking & Finance, 20 (1996), pp. 985–1001.

[41] E. Haber and L. Ruthotto, *Stable architectures for deep neural networks*, Inverse Problems, 34 (2017), p. 014004.

[42] B. Hanin and M. Sellke, *Approximating continuous functions by relu nets of minimal width*, arXiv preprint arXiv:1710.11278, (2017).

DOI:10.6814/NCCU202100296

[43] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[44] M. Herbster, M. Pontil, and L. Wainer, *Online learning over graphs*, in Proceedings of the 22nd international conference on Machine learning, 2005, pp. 305–312.

[45] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation, 9 (1997), pp. 1735–1780.

[46] L. Hogben, *Spectral graph theory and the inverse eigenvalue problem of a graph*, The Electronic Journal of Linear Algebra, 14 (2005).

[47] M. Hohenwarter, *Geogebra–ein softwaresystem für dynamische geometrie und algebra der ebene [geogebra–a software system for dynamic geometry and algebra in the plane] (unpublished master's thesis)*, Universität Salzburg, Salzburg, Austria, (2002).

[48] J. Howard and S. Ruder, *Universal language model fine-tuning for text classification*, in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2018, pp. 328–339.

[49] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, *Extreme learning machine: a new learning scheme of feedforward neural networks*, in 2004 IEEE international joint conference on neural networks (IEEE Cat. No. 04CH37541), vol. 2, Ieee, 2004, pp. 985–990.

[50] X. Huo, X. S. Ni, and A. K. Smith, *A survey of manifold-based learning methods*, Recent advances in data mining of enterprise data, (2007), pp. 691–745.

[51] X. Jiang, L.-H. Lim, Y. Yao, and Y. Ye, *Statistical ranking and combinatorial Hodge theory*, Mathematical Programming, 127 (2011), pp. 203–244.

[52] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in ICLR (Poster), 2015.

[53] D. B. Kotlow, *A free boundary problem connected with the optimal stopping problem for diffusion processes*, Transactions of the American Mathematical Society, 184 (1973), pp. 457–478.

DOI:10.6814/NCCU202100296

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks*, in Advances in neural information processing systems, 2012, pp. 1097–1105.

[55] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, nature, 521 (2015), pp. 436–444.

[56] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE, 86 (1998), pp. 2278–2324.

[57] D. Levin, *The approximation power of moving least-squares*, Mathematics of computation, 67 (1998), pp. 1517–1531.

[58] H. Lin and S. Jegelka, *Resnet with one-neuron hidden layers is a universal approximator*, Advances in neural information processing systems, 31 (2018), pp. 6169–6178.

[59] T. Lin and H. Zha, *Riemannian manifold learning*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 30 (2008), pp. 796–809.

[60] T.-Y. Lin and Y.-L. Tsai, *An application of Hodgerank to online peer assessment*, in Proceedings of the 2018 SIAM Workshop on Machine Learning for Recommender Systems (MLRec), 2018.

[61] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer, *Parameterization-free projection for geometry reconstruction*, ACM Transactions on Graphics (TOG), 26 (2007), pp. 22–es.

[62] H.-K. Liu, *Properties of American volatility options in the mean-reverting 3/2 volatility model*, SIAM Journal on Financial Mathematics, 6 (2015), pp. 53–65.

[63] H.-K. Liu, T.-Y. Lin, and Y.-L. Tsai, *On the pricing formula for the perpetual American volatility option under the mean-reverting processes*, Taiwanese Journal of Mathematics, (2020).

[64] T.-Y. Liu, *Learning to rank for information retrieval*, (2011).

DOI:10.6814/NCCU202100296

[65] Y. Lu, A. Zhong, Q. Li, and B. Dong, *Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations*, in International Conference on Machine Learning, PMLR, 2018, pp. 3276–3285.

[66] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, *The expressive power of neural networks: A view from the width*, Advances in neural information processing systems, 30 (2017), pp. 6231–6239.

[67] A. L. Maas, A. Y. Hannun, and A. Y. Ng, *Rectifier nonlinearities improve neural network acoustic models*, in ICML Workshop on Deep Learning for Audio, Speech and Language Processing, 2013.

[68] G. A. Miller, *The magical number seven, plus or minus two: Some limits on our capacity for processing information.*, Psychological review, 63 (1956), p. 81.

[69] J. W. Milnor and J. D. Stasheff, *Characteristic Classes*, no. 76, Princeton University Press, 1974.

[70] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, *Playing Atari with deep reinforcement learning*, (2013). cite arxiv: 1312.5602Comment: NIPS Deep Learning Workshop 2013.

[71] Y. Nesterov, *A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2)*, in Doklady an ussr, vol. 269, 1983, pp. 543–547.

[72] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, *Exploring generalization in deep learning*, in Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 5949–5958.

[73] P. Niyogi, S. Smale, and S. Weinberger, *Finding the homology of submanifolds with high confidence from random samples*, Discrete & Computational Geometry, 39 (2008), pp. 419–441.

[74] Y. Parmar, S. Natarajan, and G. Sobha, *Deeprange: deep-learning-based object detection and ranging in autonomous driving*, IET Intelligent Transport Systems, 13 (2019), pp. 1256–1264.

DOI:10.6814/NCCU202100296

[75] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, *Deep contextualized word representations*, in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 2227–2237.

[76] G. Philipp, D. Song, and J. G. Carbonell, *Gradients explode-deep networks are shallow-resnet explained*, (2018).

[77] R. Piziak and P. Odell, *Affine projections*, Computers & Mathematics with Applications, 48 (2004), pp. 177–190.

[78] R. Pless and R. Souvenir, *A survey of manifold learning for images*, IPSJ Transactions on Computer Vision and Applications, 1 (2009), pp. 83–94.

[79] L. Ruthotto and E. Haber, *Deep neural networks motivated by partial differential equations*, Journal of Mathematical Imaging and Vision, (2019), pp. 1–13.

[80] R. W. Saaty, *The analytic hierarchy process—what it is and how it is used*, Mathematical modelling, 9 (1987), pp. 161–176.

[81] M. Saerens, F. Fouss, L. Yen, and P. Dupont, *The principal components analysis of a graph, and its relationships to spectral clustering*, in European conference on machine learning, Springer, 2004, pp. 371–383.

[82] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, *The graph neural network model*, IEEE Transactions on Neural Networks, 20 (2008), pp. 61–80.

[83] R. K. Sizemore, *HodgeRank: Applying combinatorial Hodge theory to sports ranking*, PhD thesis, Wake Forest University, 2013.

[84] B. Sober and D. Levin, *Manifold approximation by moving least-squares projection (MMLS)*, Constructive Approximation, (2019), pp. 1–46.

[85] G. Stewart, *On the continuity of the generalized inverse*, SIAM Journal on Applied Mathematics, 17 (1969), pp. 33–45.

DOI:10.6814/NCCU202100296

[86] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, *Efficient transformers: A survey*, arXiv preprint arXiv:2009.06732, (2020).

[87] N. M. Tran, *Pairwise ranking: choice of method can produce arbitrarily different rank order*, Linear Algebra and its Applications, 438 (2013), pp. 1012–1024.

[88] G. Van Rossum and F. L. Drake Jr, *Python tutorial*, vol. 620, Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[89] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need*, Advances in neural information processing systems, 30 (2017), pp. 5998–6008.

[90] A. Veit, M. J. Wilber, and S. Belongie, *Residual networks behave like ensembles of relatively shallow networks*, Advances in neural information processing systems, 29 (2016), pp. 550–558.

[91] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, *Graph attention networks*, in International Conference on Learning Representations, 2018.

[92] V. P. Vishwakarma and M. Gupta, *A new learning algorithm for single hidden layer feedforward neural networks*, International Journal of Computer Applications, 28 (2011), pp. 26–33.

[93] T. Walsh, *The PeerRank method for peer assessment*, in Proceedings of the Twenty-first European Conference on Artificial Intelligence, 2014, pp. 909–914.

[94] T.-W. Weng, H. Zhang, P.-Y. Chen, J. Yi, D. Su, Y. Gao, C.-J. Hsieh, and L. Daniel, *Evaluating the robustness of neural networks: An extreme value theory approach*, in International Conference on Learning Representations, 2018.

[95] D. B. West, *Introduction to graph theory*, vol. 2, Prentice hall Upper Saddle River, 2001.

[96] H. Whitney, *Differentiable manifolds*, Annals of Mathematics, (1936), pp. 645–680.

[97] S. Wold, K. Esbensen, and P. Geladi, *Principal component analysis*, Chemometrics and intelligent laboratory systems, 2 (1987), pp. 37–52.

DOI:10.6814/NCCU202100296

[98] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, *A comprehensive survey on graph neural networks*, IEEE Transactions on Neural Networks and Learning Systems, (2020).

[99] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, *Aggregated residual transformations for deep neural networks*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1492–1500.

[100] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, *How powerful are graph neural networks?*, in International Conference on Learning Representations, 2018.

DOI:10.6814/NCCU202100296