On the Crossroads of Scattering Transform and Machine Learning in Image and Signal Processing

By

Wai Ho CHAK

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

APPLIED MATHEMATICS

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Naoki Saito

Naoki Saito

[illegible]

Thomas Strohmer

Soheil Ghiasi

Soheil Ghiasi

Committee in Charge

2023

The thesis is dedicated to people who have supported me throughout my PhD journey, especially during the hardship under the pandemic. Thank you for the support along the way.

Contents

On the Crossroads of Scattering Transform and Machine Learning in Image and Signal Processing

Abstract

Convolutional neural network (CNN) has been effective in solving image and signal processing problems. On the other hand, scattering transform (ST) mathematically formalizes some properties that have made the CNN successful in solving these problems. Its network structure is similar to a CNN, except that it provides interpretable ST coefficients and does not require a gigantic dataset. The ST network generates a robust representation stable to local deformation while keeping essential high-frequency components of an input signal through a cascade of wavelet convolutions with nonlinear operations followed by averaging. In dissertation, motivated by the mathematics behind analyticity and monogenicity, we propose new ST networks to solve these problems.

We present the novel incorporation of the generalized Morse wavelet into the 1-D ST network (Morse-STN) for music genre classification, instead of the commonly-used Morlet wavelet. The reason behind is that the class of generalized Morse wavelets is a superfamily of analytic wavelets, but the Morlet wavelet is only approximately analytic. A significant improvement in the classification accuracy of music genre can be demonstrated in the GTZAN music signal dataset using the generalized Morse wavelet rather than the Morlet wavelet.

A new Monogenic Wavelet Scattering Network (MWSN) is also proposed for 2-D texture image classification, instead of using the 2-D Morlet wavelet in the standard 2-D ST network. Our MWSN extracts valuable hierarchical features with interpretable ST coefficients which help us explain the result. We illustrate the superior performance of our MWSN over the standard STN from the experiment in the CUReT texture image database. The improvement can be explained by the natural extension of 1-D analyticity to 2-D monogenicity.

Lastly, we apply the proposed ST networks in sonar signal classfication. Mine counter-measure (MCM) is crucial for the US Navy, but it relies on accurate mine detection. We synthesize sonar signals using both Gabor and real dolphin signals as sources for sonar classification. Then the generated synthetic aperture sonar (SAS) signals from the underwater objects are classified by the new ST feature extractors. The result can be explained by the ST coefficients together with interpretable simple classifiers such as the logistic regression and the support vector machines, while deep learning approaches lack interpretability.

Acknowledgments

This dissertation marks the climax of my passionate journey in the applied mathematical field. When I was an undergraduate student at the Chinese University of Hong Kong, I often thought the journey towards doctorate in applied mathematics was unattainable. Ultimately, I am so grateful that my passion in mathematics has helped me overcome difficulties in my whole PhD journey.

I am beyond grateful for the enormous support from my family and my local, international and American friends during my PhD journey in the US, especially with the challenges outside academia. I have made the bold decision to leave my home for several years and become a graduate researcher at the University of California, Davis. Although I did not have chance to engage more in activities with colleagues and students on campus during the hardship caused by external factors, I am still grateful to be here in Davis to explore life and interests inside and outside academia.

I would like to express my gratitude to Professor Naoki Saito, who is my dissertation advisor. His patience and guidance has been great throughout my PhD journey. Besides the dissertation research mainly on scattering transform, he paved the way for collaboration with Professor Soheil Ghiasi and PhD student researchers like Begum Kasap and Kourosh Vali in the Electrical and Computer Engineering Department on campus to work on fetal signal extraction for transabdominal fetal pulse oximetry. Most importantly, I remind myself the most on his understanding and support when I encountered struggles outside my PhD research. Besides, I am very thankful to work on a project in microbiology with Professor Mariel Vazquez and Professor Javier Arsuaga in the Graduate Group in Applied Mathematics (GGAM) and in the Microbiology Department with PhD student researcher Tamara Christiani on campus. I would also like to thank Professor Thomas Strohmer and Professor Soheil Ghiasi for being on my PhD dissertation committee.

As for the funding, the dissertation research mainly on the scattering transform network was partially supported by the US National Science Foundation grants DMS-1912747, CCF-1934568, and the US Office of Naval Research (ONR) grant N00014-20-1-2381. The research on the fetal signal extraction with the Electrical and Computer Engineering Department at UCDavis was supported by the National Science Foundation (NSF), the National Institutes of Health (NIH) and the University of California Davis Center for Data Science and Artificial Intelligence Research (CeDAR). The research on the automated detection of DNA organization in cryoEM micrographs of bacteriophage P4 is partially supported in part by NSF grants from Professors Mariel Vazquez and Javier Arsuaga: DMS-1817156, DMS-1716987 and DMS-2054347.

The dissertation research involves prior effort from other researchers and support in the research community. I would like to thank Professor Saito's former PhD student David Weber for his development of the full scattering transform package in Julia programming language, and his suggestions and support in developing the code for the scattering transform network in this dissertation research. The dissertation research would have not been successful without the kind computing support and services, including the GPU computing, from the UC Davis Genome Center. In addition, I am thankful that Dorian Hauser of National Marine Mammal Foundation kindly shared the dolphin recordings to Professor Saito, hence leading to research findings on sonar detection for this dissertation and for the annual presentation for Office of Naval Research (ONR) MCM Virtual Program Review. I want to thank Professor James Bremer (now at University of Toronto) and Dr. Ian Sammis (now at Google) for the development of the Helmholtz equation solver and Professor Saito's former intern Vincent Bodin (now at Dimpl) for his contribution to the main program to generate synthetic sonar signals.

CHAPTER 1

Introduction

The *scattering transform network* (STN) has an architecture similar to the well-known convolutional neural network (CNN). The latter is acclaimed for its ability to extract hierarchical features [**72**] for many applications such as image classification [**68**] and facial recognition [**116**] with gigantic training datasets. Even Mallat [**82**] and Wiatoski [**132**] formulated the mathematical analysis of deep CNNs for feature extraction using scattering transform (ST). Despite the popularity of CNN, the CNN overfits in a dataset without sufficient sample size. On the other hand, the STN works well without being data hungry, and does not require stochastic optimization to learn convolution filters from data, since it employs predefined wavelet filters. Yet, Mallat showed that the STN provides a quasi-translation invariant representation from input signals when the scale tends to infinite resolution, and it is Lipschitz continuous under non-uniform translation [**82**]. Furthermore, Bruna and Mallat demonstrated the power and effectiveness of the STN by the numerical experiments for image classification [**17**, **18**]. Most importantly, the scattering coefficients allow interpretation on machine learning applications, whereas interpretability is a challenge for deep learning methods.

We provide an overview of the content for each chapter after the introduction. The overall theme of the dissertation is the proposal of scattering transform approaches and their machine learning applications. For each proposed scattering transform approach, the mathematics of analyticity and monogenicity are leveraged to motivate the novel design of the scattering transform architectures. The new methods are combined with existing machine learning tools to perform various tasks in image and signal classification. Results from numerical experiments are presented throughout the dissertation.

Chapter 2 reviews the mathematical background of analyticity and monogenicity. These concepts are utilized in developing the novel scattering transform approaches. The chapter starts by introducing analytic signal and *analytic wavelet transform* (AWT), which provides interpretable multiscale instantaneous magnitude and phase information for 1-D signal analysis. In particular, we introduce and compare the Morlet wavelet and generalized Morse wavelets. When one wants to analyze a 2-D signal, the concept of 1-D analyticity needs to be extended properly. A properly extended concept of 1-D analytic signal to higher dimension is called the *monogenic signal* proposed by Felsberg and Sommer [**31**]. Based on the monogenic signal theory, Olhede and Metikas proposed the *monogenic wavelet transform* (MWT) [**91**] generalizing the 1-D AWT. The MWT inherits desirable properties of AWT on higher-dimensional signals, such as providing multiscale instantaneous amplitude, local phase, and orientation information. It would lead to better interpretability of high-dimensional input signals [**91**].

We then discuss the connections from the CNN to ST in Chapter 3. We provide an overview of neural network, in particular the CNN, in the beginning of the chapter. Then we draw a part of the published work written by the dissertation author as the first author on deep neural network: *Turbulence Removal Network* (TRN) [**20**]. While there were multiple contributions in this research, the main contribution discussed in the dissertation is the first proposal of using deep neural network to address the problem of turbulence degradation on videos that are scarcely available in public. Next, we move on to review the architecture of the STN, which has a very similar structure with the CNN. We further connect the two subjects by their theoretical foundation developed by Wiatoski [**132**], with an emphasis on the mathematical theories behind ST. After that, we briefly describe the machine learning models that were utilized in the numerical experiments together with the STN in the latter chapters of the dissertation.

After introducing the mathematical background and the connection between CNN and ST, we discuss the techniques in GPU-enabled STN under the AWT. In particular, we focus on the proposed STN incorporating the generalized Morse wavelets (Morse-STN) in Chapter 4. Some materials in this chapter are extracted from the Reference [**22**] written by the dissertation author, Professor Saito, and Dr. David Weber. Many previous works on the STN used the Morlet wavelet [**81**, Sec. 4.3], which are only approximately analytic. A typical implementation of the STN, such as the Kymatio package [**5**], employed the Morlet wavelet as its base filter. Nevertheless, the analyticity of the wavelet is important for nonstationary and oscillatory input signals since we can represent them in terms of amplitude, phase, and frequency in a multiscale manner [**79**]. In this dissertation, we propose the use of *Generalized Morse Wavelet* (GMW) filters [**80**] as the base filters in the STN instead of the popularly-used Morlet wavelet filters. In particular, we evaluated the performance in music genre classification using the GTZAN music dataset [**122**].

Then we explore the natural extension from the 1-D ST to 2-D ST in Chapter 5. In our prior work in Reference [**21**] written by the dissertation author and Professor Saito, we used MWT to develop our 2-D STN network. The *monogenic scattering transform network* (MWSN) is a novel idea to capture more 2-D geometric features instead of the default 2-D STN using the Morlet wavelets. In addition to being a theoretically-sound 2-D extension of the AWT, there are several desirable properties to make the MWT as a natural candidate for building a 2-D STN network architecture as its base wavelet transform [**91**]. The MWT has been applied to image denoising [**115**] and image compression [**57**]. The experiment in texture image classification on the CUReT texture image dataset [**26**] have shown that we can capture 2-D geometric features more efficiently from the MWSN than what the Morlet wavelets-based STN could provide.

In Chapter 6, we apply the proposed STN from last chapter in sonar signal classification. Content in this chapter contributed to the presentation on “Robust Feature Extraction from Acoustic Wavefields for Object Classification” in the Office of Naval Research (ONR) MCM Virtual Program Review in 2021 and 2022. We outline the procedures for sonar classification using Gabor function as the source signal. We use the Helmholtz equation solver to generate the synthetic aperture sonar (SAS) dataset [**13, 14, 15**]. Since the dolphin-based marine mammal systems (MMS) are adopted as a solution for mine detection and hunting by the US Navy [**86**], we further use real dolphin clicks as a source to generate wavefields for sonar signal classification experiments.

Last but not least, we conclude the common theme and the contributions of this dissertation and highlight the possible directions for future work in Chapter 7. Supplementary information and extra contents that are helpful but not strictly relevant to the common theme of the dissertation topic can be found in the appendix. Appendix A shows the research in multi-detector fetal signal extraction [**63**], with the dissertation author as a contributing author of the publication.

CHAPTER 2

Analyticity and Monogenicity

This chapter is devoted to explain the mathematical concept of analyticity and monogenicity. In particular, we will introduce the definition and the properties of an *analytic signal* in Section 2.1. Later we will describe how analyticity can be extended through the *monogenic wavelet transform* (MWT) to have an interpretable representation in 2-D signal analysis. Some nice properties of the MWT will also be discussed in this chapter.

2.1. Analyticity

A signal is *analytic* if it is a complex-valued function without negative frequency components [**110**]. An *analytic signal* of a real-valued function is an analytical representation providing useful information such as phase and amplitude in signal analysis. The relationship between the real and imaginary components in the Hilbert transform is described below:

Let $g(x)$ be a real-valued signal with its Fourier transform

$$G(\xi) := \mathcal{F}[g(x)](\xi) = \int_{-\infty}^{\infty} g(x)\mathrm{e}^{-2\mathrm{i}\pi\xi x}\,\mathrm{d}x. \tag{2.1}$$

Then there is a Hermitian symmetry in the Fourier transform $G(\xi)$

$$G(-\xi) = G(\xi)^*. \tag{2.2}$$

Note that the function

$$\begin{aligned} G^+(\xi) &:= G(\xi)(1+\operatorname{sgn}(\xi)) \\ &= \begin{cases} 0, & \text{if } \xi < 0, \\ G(\xi), & \text{if } \xi = 0, \\ 2G(\xi), & \text{if } \xi > 0 \end{cases} \end{aligned} \tag{2.3}$$

only contains non-negative frequency components. Due to the Hermitian symmetry, we can express the Fourier transform $G(\xi)$ in terms of $G^+(\xi)$ by

$$G(\xi) = \begin{cases} \frac{1}{2}G^+(-\xi)^*, & \text{if } \xi < 0, \\ G^+(\xi), & \text{if } \xi = 0, \\ \frac{1}{2}G^+(\xi), & \text{if } \xi > 0 \end{cases} \tag{2.4}$$
$$= \frac{1}{2}\Big(G^+(\xi) + G^+(-\xi)^*\Big).$$

The *analytic signal* $g^+(x)$ of $g(x)$ is the inverse Fourier transform of $G^+(\xi)$, i.e.,

$$\begin{aligned} g^+(x) &= \mathcal{F}^{-1}[G^+(\xi)] \\ &= \mathcal{F}^{-1}[G(\xi) + \operatorname{sgn}(\xi)G(\xi)] \\ &= \mathcal{F}^{-1}[G(\xi)] + \mathcal{F}^{-1}[\operatorname{sgn}(\xi)] * \mathcal{F}^{-1}[G(\xi)] \\ &= g(x) + \frac{\mathrm{i}}{\pi x} * g(x) \\ &= g(x) + \frac{\mathrm{i}}{\pi}\ \text{p.v.} \int_{-\infty}^{\infty} \frac{g(y)}{x-y}\,\mathrm{d}y \\ &=: g(x) + \mathrm{i}\mathcal{H}[g](x) \\ &=: g(x) + \mathrm{i}g^{(1)}(x) \end{aligned} \tag{2.5}$$

where $g^{(1)}(x)$ is defined as the Hilbert transform $\mathcal{H}$ of $g(x)$ and p.v. is the Cauchy principal value for an improper integral.

Similarly, we can define the *anti-analytic* signal as

$$g^-(x) := g(x) - \mathrm{i}g^{(1)}(x), \tag{2.6}$$

such that the signal can be expressed as an *analytic decomposition*:

$$g(x) = \frac{1}{2}\Big(g^+(x) + g^-(x)\Big) \tag{2.7}$$

with

$$g^{\pm}(x) = |g^{\pm}(x)| \exp(\pm 2\pi \mathrm{i} \varphi_g(x)), \tag{2.8}$$

where the square of the amplitude, $|g^{\pm}(x)|^2$, is the local energy and the phase $\varphi_g(x)$ is the local phase structural representation. In conclusion, as the Hermitian symmetry shows that the negative frequency components are superfluous, we can obtain the analytic signal $g^+(x)$ with interpretable information such as amplitude and phase for 1-D signal analysis.

2.1.1. Example of Analytic Signal. Let a real-valued function

$$\begin{aligned} g(x) &= 2\cos\Big(2\pi(100)\exp(-x)\Big), \\ &= 2\mathrm{Re}\Big\{\exp\Big(-2\pi\mathrm{i}\ (100)\exp(-x)\Big)\Big\} \end{aligned} \tag{2.9}$$

where $x \in \mathbb{R}$, be a chirp signal. The chirp signal g has a decreasing frequency from 100 Hz with the x-axis as shown in the spectrogram in Figure 2.1. We can numerically compute the Hilbert transform $\mathcal{H}$ of the chirp signal g in the Fourier domain. Analytically, the analytic signal g^+ is

$$g^+(x) = g(x) + \mathrm{i}\mathcal{H}[g](x) = 2\exp\Big(-2\pi\mathrm{i}\ (100)\exp(-x)\Big) \tag{2.10}$$

From the analytic representation, the amplitude of the analytic signal is $|g^+(x)| = 2$ and the phase of the analytic signal $\varphi_g(x) = -100\exp(-x)$. In particular, the amplitude can be computed numerically by taking the modulus of the analytic signal as shown in Figure 2.2. The boundary artifacts from the numerical result are due to the periodization of g via *discrete Fourier transform* (DFT) in numerical computations.

Alternatively, we can find the phase ${\varphi_g}^{(0)}(x)$ of the chirp signal by the instantaneous frequency of the chirp signal by the following relationship:

$$\omega_0(x) = \frac{1}{2\pi}\frac{\mathrm{d}{\varphi_g}^{(0)}(x)}{\mathrm{d}x} \tag{2.11}$$

The instantaneous frequency of the chirp signal is

$$\omega_0(x) = 100\exp(-x). \tag{2.12}$$

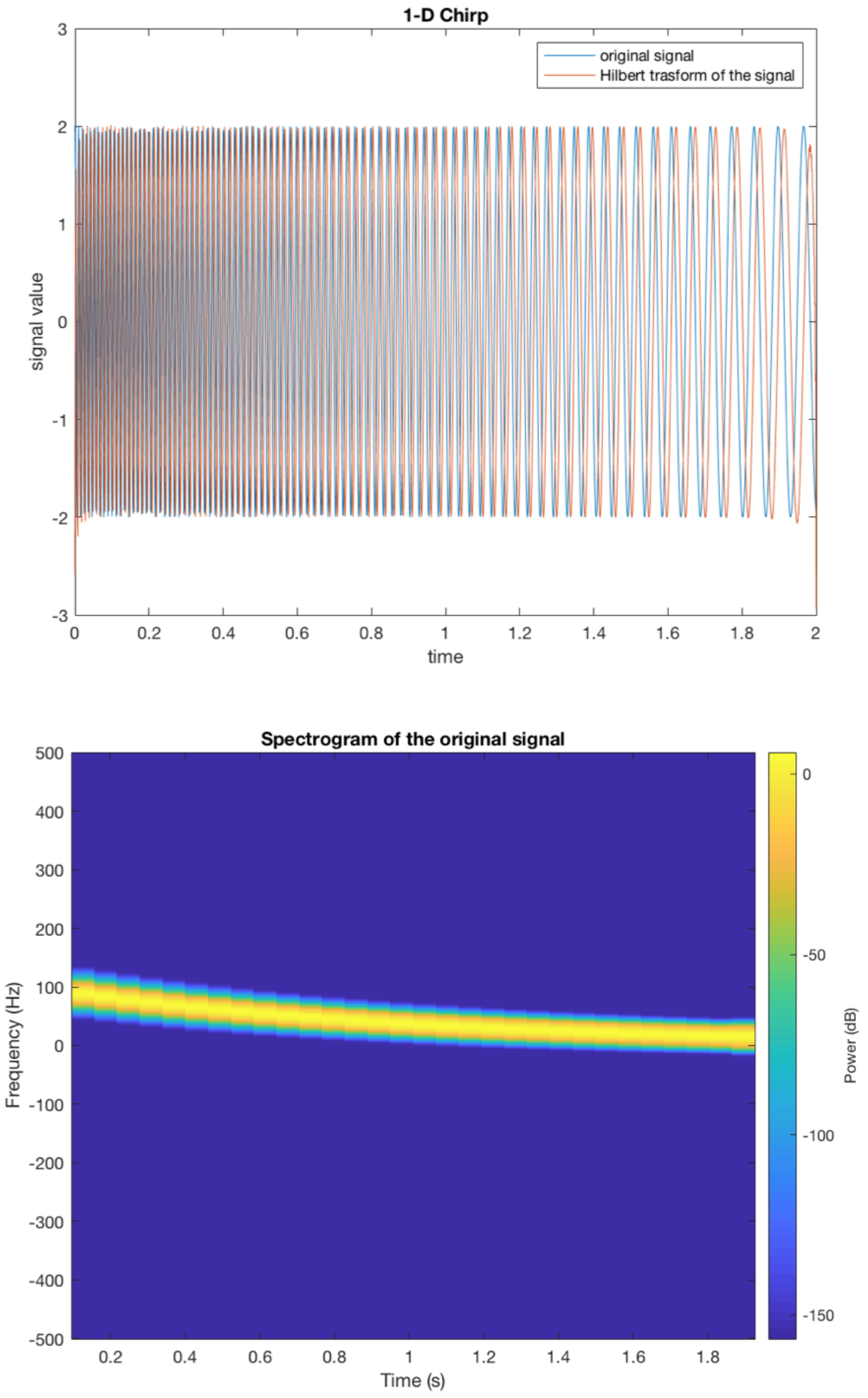


FIGURE 2.1. Example of analytic signal from the 1D chirp signal.

The phase of the chirp signal is the integral of the instantaneous frequency

$$\begin{aligned}\varphi_g^{(0)}(x) &= \varphi_g^{(0)}(0) + 2\pi \int_0^x 100 \exp(-\tau) d\tau \\ &= \varphi_g^{(0)}(0) - 2\pi(100)\exp(-x) + 2\pi(100)\end{aligned} \tag{2.13}$$

The phase in Figure 2.3 is identical to $\varphi_g^{(0)}$ in Equation (2.13) by setting $\varphi_g^{(0)}(0) = 0$. The phase φ_g of the analytic signal g^+ and the phase $\varphi_g^{(0)}$ of the chirp signal are differed only by a constant. The instantaneous frequency of the analytic signal g^+ computed numerically as shown in Figure 2.3 is identical to ω_0 except with boundary artifacts in numerical computations.

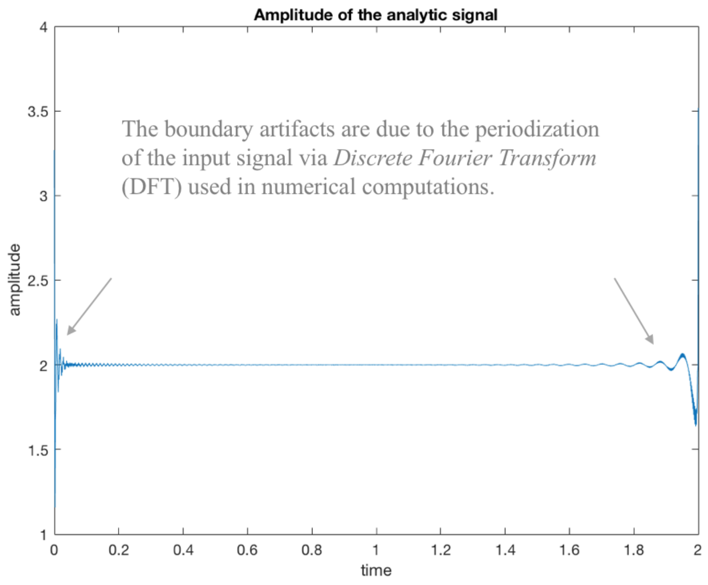


FIGURE 2.2. The amplitude of analytic signal from the 1D chirp signal.

2.2. Continuous and Analytic Wavelet Transform

The notion of wavelet is needed in signal analysis. A wavelet, also known as *mother wavelet*, $\psi(x) \in L^2(\mathbb{R})$, is a function whose translated and dilated versions provide a way to perform localized time-frequency analysis of nonstationary oscillatory signals, such as music and audio signals [**81**, Sec. 4.3]. The wavelet has to satisfy the two *admissibility conditions* in Equations (2.14) and (2.15):

$$C_\psi := \int_{\mathbb{R}} \frac{|\Psi(\xi)|^2}{|\xi|} \mathrm{d}\xi < \infty \tag{2.14}$$

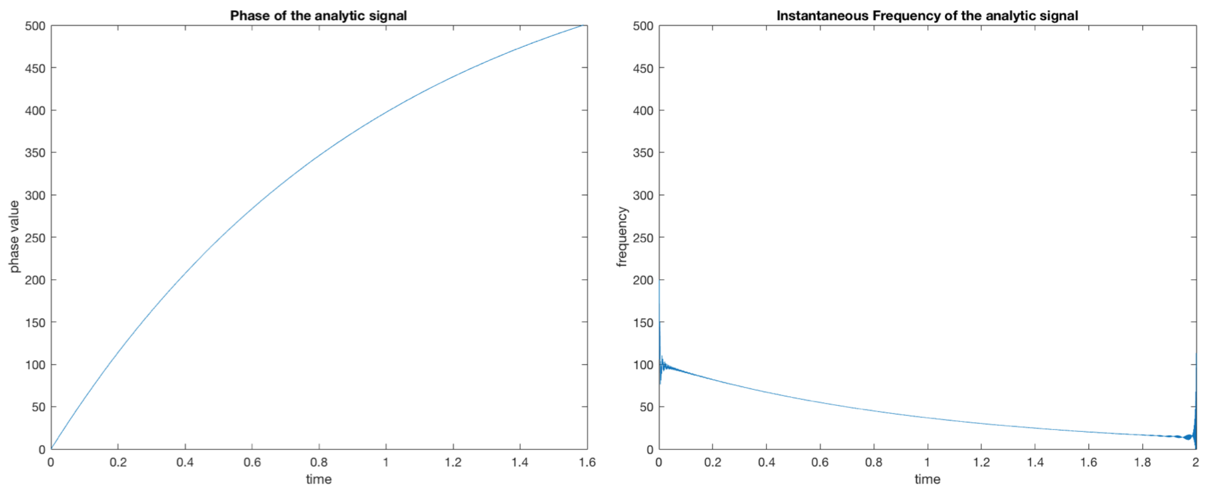


FIGURE 2.3. The phase and the instantaneous frequency of analytic signal from the 1D chirp signal.

$$\int_{\mathbb{R}} |\psi(x)|^2 \, \mathrm{d}x = 1, \tag{2.15}$$

where $\Psi(\xi)$ is the Fourier transform of the wavelet. In particular, a wavelet is called *analytic* if it is complex-valued and vanishes for negative frequencies. That is, $\Psi(\xi) = 0$ for $\xi < 0$.

The 1-D *continuous wavelet transform* (CWT) of a signal $g \in L^2(\mathbb{R})$ with respect to the mother wavelet ψ is given by

$$W_\psi g(s, b) := \frac{1}{\sqrt{s}} \int_{\mathbb{R}} g(x) \psi^* \left(\frac{x - b}{s} \right) \mathrm{d}x \tag{2.16}$$

for any scale $s \in \mathbb{R}^+$, time $b \in \mathbb{R}$. In particular, the scale s is a dilation parameter. When s is small, the wavelet tends to be contracted for capturing rapidly changing details and hence higher frequency information of the signal $g(x)$. On the other hand, when s is large, the wavelet tends to be stretched for capturing coarse, slowly varying features and hence lower frequency information of $g(x)$.

Under the admissibility condition for wavelet ψ, if $g \in L^2(\mathbb{R})$, then the CWT is invertible, and the *inverse continuous wavelet transform* (ICWT) is

$$g(x) = \frac{1}{C_\psi} \int_{\mathbb{R}} \int_0^\infty \frac{1}{\sqrt{s}} W_\psi g(s, b) \psi \left(\frac{x - b}{s} \right) \frac{\mathrm{d}s}{s^2} \, \mathrm{d}b. \tag{2.17}$$

The CWT defines the *analytic wavelet transform* (AWT) if ψ is analytic. In this case, any signal $g \in L^2(\mathbb{R})$ can be reconstructed after the AWT. That is,

$$g(x) = \frac{1}{C_\psi} \int_{\mathbb{R}} \int_0^\infty \frac{1}{\sqrt{s}} \mathrm{Re}\left(W_\psi g(s,b) \psi\left(\frac{x-b}{s}\right)\right) \frac{\mathrm{d}s}{s^2} \, \mathrm{d}b. \tag{2.18}$$

The AWT gives an approximation of the instantaneous phase, frequency and amplitude of any signal g in an area surrounding each time-scale location (s, b) [**27**, **81**]. An application of AWT is the detection and characterization of singularities using the modulus maxima of the wavelet transform [**120**]. More applications in signal and image processing with the features from the analytic wavelets are outlined by Selesnick *et al.* [**109**].

2.2.1. Example of analytic wavelet transform. Let

$$g(x) = 1.5 \sin\Big(2\pi(16)x\Big) \mathbb{1}_{[0.1,0.3)} + 2\sin\Big(2\pi(32)x\Big)\mathbb{1}_{[0.5,0.7)} \tag{2.19}$$

be an initial signal, where $x = 0 : \frac{1}{500} : 1$. Figure 2.4 visualizes the signal g, which is supported on $[0.1, 0.3) \cup [0.5, 0.7)$. We will use AWT to illustrate how to extract time-frequency information from the initial signal.

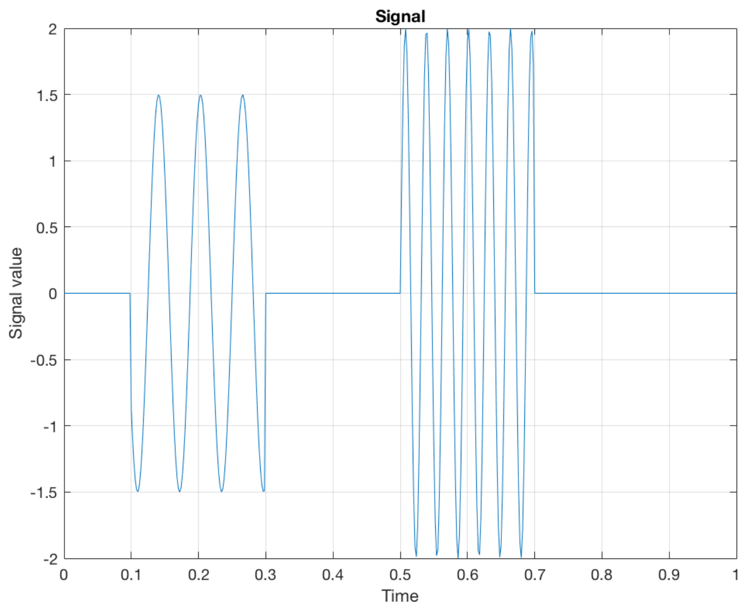


FIGURE 2.4. An example of a signal g.

We can choose an analytic wavelet for AWT. One candidate is *Gabor wavelet*, which is an approximately analytic wavelet defined as

$$\psi(x; s, x_0, k_0) = \mathrm{e}^{-\frac{(x-x_0)^2}{s^2}} \mathrm{e}^{-\mathrm{i}k_0(x-x_0)}, \tag{2.20}$$

where s controls the rate of exponential decay from the center x_0, and k_0 controls the modulation

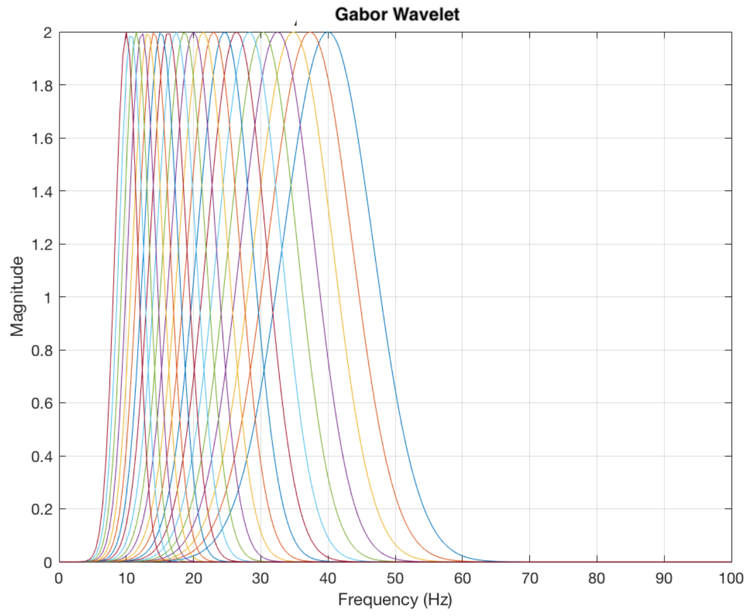


FIGURE 2.5. An example of Gabor wavelets that are approximately analytic.

rate in the complex exponential. Figure 2.5 shows the frequency response of the filter bank from $\psi(x; s, x_0, k_0)$ when $x_0 = 0, s = 2, k_0 = 1$ with frequency limit from 10 Hz to 40 Hz. The wavelet has wider bandwidth when it has a higher center frequency. We will discuss the different wavelet classes more in the next subsection.

The AWT of the signal $g(x)$ under the analytic wavelet $\psi(x; s, x_0, k_0)$ can be visualized by a scalogram in Figure 2.6. We can obtain the time-frequency representation of the initial signal $g(x)$. The scalogram is more interpretable than the plot of the original signal $g(x)$, showing the decay of amplitude at different frequencies from the two supported intervals. Since the time-frequency representation from the scalogram is more structured, the representation is extensively used for feature extraction of the initial 1-D signal.

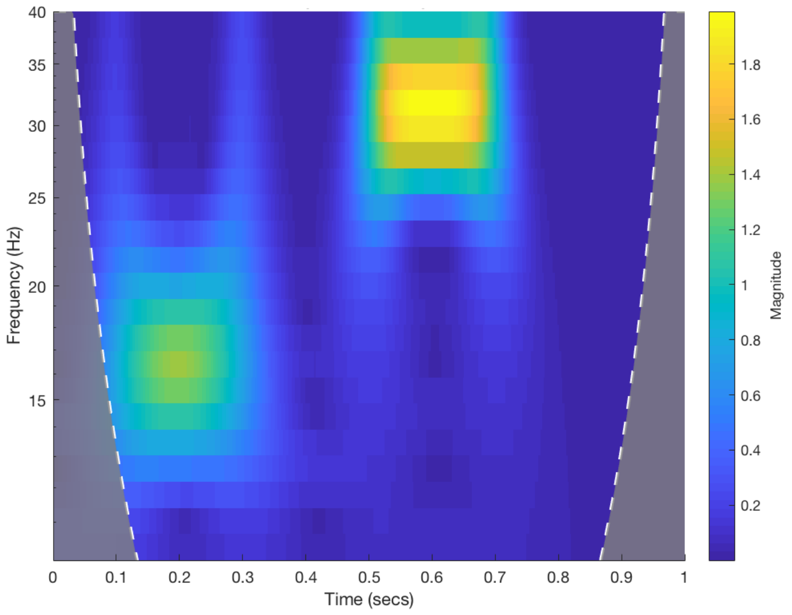


FIGURE 2.6. Scalogram of the AWT.

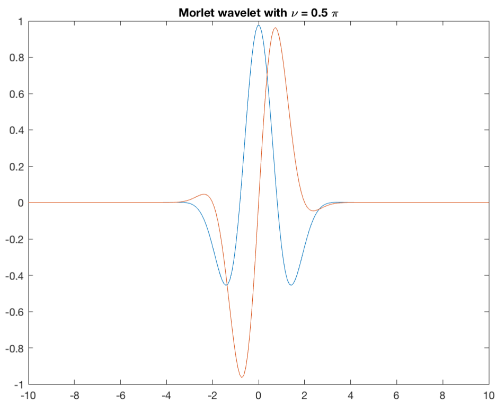
2.3. Morlet and Generalized Morse Wavelets

In the last section, we described the use of analytic wavelets in an example for extracting the time-frequency representation from a signal through wavelet transform. In practice, there are different choices of analytic wavelet in numerical computations. We will also outline the properties of the *Morlet Wavelets* and *Generalized Morse Wavelets* (GMWs).

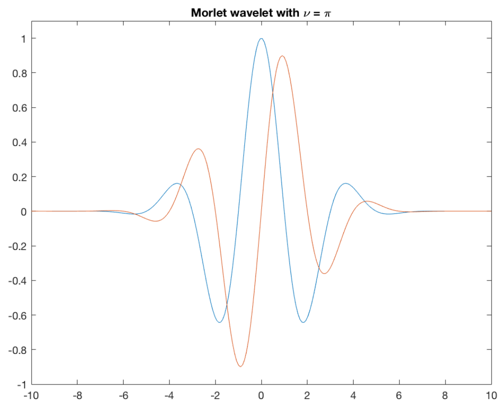
2.3.1. Morlet Wavelet. One of the oldest complex-valued wavelets proposed by Jean Morlet (1984) whose origin can be traced back to Dennis Gabor (1946). Mathematically, the *Morlet wavelet* is defined as [**138**]:

$$\psi_\nu(t) := c_\nu \pi^{-1/4} \mathrm{e}^{-t^2/2}\left(\mathrm{e}^{\mathrm{i}\nu t} - \mathrm{e}^{-\nu^2/2}\right), \tag{2.21}$$

where, $c_\nu := \left(1 + \mathrm{e}^{-\nu^2} - 2\mathrm{e}^{-3\nu^2/4}\right)^{-1/2}$. Figure 2.7 visualizes the Morlet wavelet with $\nu = 0.5\pi$ and $\nu = \pi$. It is basically a *Gabor wavelet* function with a correction to get $\widehat{\psi}_\nu(0) = 0$. In fact, its



(a) Morlet with $\nu = 0.5\pi$



(b) Morlet with $\nu = \pi$

FIGURE 2.7. Examples of Morlet wavelet. The blue and red colours in the plots correspond to real and imaginary parts.

Fourier transform is:

$$\widehat{\psi}_\nu(\omega) = c_\nu \pi^{-1/4} \mathrm{e}^{-(\nu-\omega)^2/2}\Big(1 - \mathrm{e}^{-\nu\omega}\Big), \tag{2.22}$$

which gives us $\widehat{\psi}_\nu(0) = 0$ while $\widehat{\psi}_\nu(-0.43578) \approx -1.70923 \times 10^{-9}$ for $\nu = 2\pi$. Hence the Morlet wavelet is *not strictly analytic*.

The Morlet wavelet was extensively used in the *scattering transform network* (STN) literature and software implementation which we will discuss in the next chapters. However, Lilly and Olhede [**78**, **80**] numerically demonstrated that even small leakage to negative frequencies in the Morlet wavelet can result in abnormal transform phase variation.

2.3.2. Generalized Morse Wavelet (GMW). A follow-up question that we need to address is in practice which analytic wavelet is suitable. It turns out that the *Generalized Morse Wavelets* (GMWs) is a promising superfamily of *truly analytic* wavelets [**80**,**92**]. The GMWs in the frequency domain can be defined as

$$\Psi_{\beta,\gamma}(\omega) := \int_{\mathbb{R}} \psi_{\beta,\gamma}(t)\mathrm{e}^{-\mathrm{i}\omega t}\,\mathrm{d}t = H(\omega)\alpha_{\beta,\gamma}\omega^\beta \mathrm{e}^{-\omega^\gamma}, \tag{2.23}$$

where $\beta > 0, \gamma \geq 1$ are two main parameters controlling the form of wavelet,

$$\alpha_{\beta,\gamma} = 2\left(\frac{\mathrm{e}\gamma}{\beta}\right)^{\beta/\gamma}$$

is a normalization constant, and $H(\omega)$ is the Heaviside step function. Besides showing an additional degree of freedom in the GMWs, the two parameters β and γ control the time-domain and frequency-domain decay respectively. The peak frequency $\omega_{\beta,\gamma} := (\beta/\gamma)^{1/\gamma}$ is the frequency at which the derivative of $\Psi_{\beta,\gamma}$ vanishes [**78**, **80**].

2.3.2.1. *Interpretation of parameters in GMWs.* Next, we will discuss the interpretations of the pair of parameters (β, γ). Figure 2.8 in the time domain and Figure 2.9 in the frequency domain which were generated by the JLab package [**77**] visualize different pairs of (β, γ) in the Morse wavelets.

When $\beta = 0, \gamma > 1$, the GMW becomes

$$\psi_{0,\gamma}(t) = \frac{1}{\pi}\int_0^\infty \mathrm{e}^{-\omega^\gamma}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega, \tag{2.24}$$

Alternatively, we can define a kernel for time-domain transformation

$$K_\gamma(t,u) := \int_0^\infty \frac{1}{2\pi}\mathrm{e}^{\mathrm{i}\omega t - \mathrm{i}\omega^\gamma u}\,\mathrm{d}\omega. \tag{2.25}$$

By the fact that $\Psi_{0,1}(\omega) = 2\mathrm{e}^{-\omega}$, one has

$$\begin{aligned}
\psi_{0,\gamma}(t) &= \frac{1}{2\pi}\int_0^\infty \Psi_{0,1}(\omega^\gamma)\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\
&= \frac{1}{2\pi}\int_0^\infty \left(\int_{-\infty}^\infty \psi_{0,1}(u)\mathrm{e}^{-\mathrm{i}\omega^\gamma u}\,\mathrm{d}u\right)\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\
&= \int_{-\infty}^\infty \psi_{0,1}(u)\left(\int_0^\infty \frac{1}{2\pi}\mathrm{e}^{\mathrm{i}\omega t - \mathrm{i}\omega^\gamma u}\,\mathrm{d}\omega\right)\mathrm{d}u \\
&= \int_{-\infty}^\infty \psi_{0,1}(u) K_\gamma(t,u)\,\mathrm{d}u.
\end{aligned} \tag{2.26}$$

In particular, when $\gamma = 1$, we have

$$K_1(t,u) = \int_0^\infty \frac{1}{2\pi}\mathrm{e}^{\mathrm{i}\omega(t-u)}\,\mathrm{d}\omega = \delta(t-u). \tag{2.27}$$

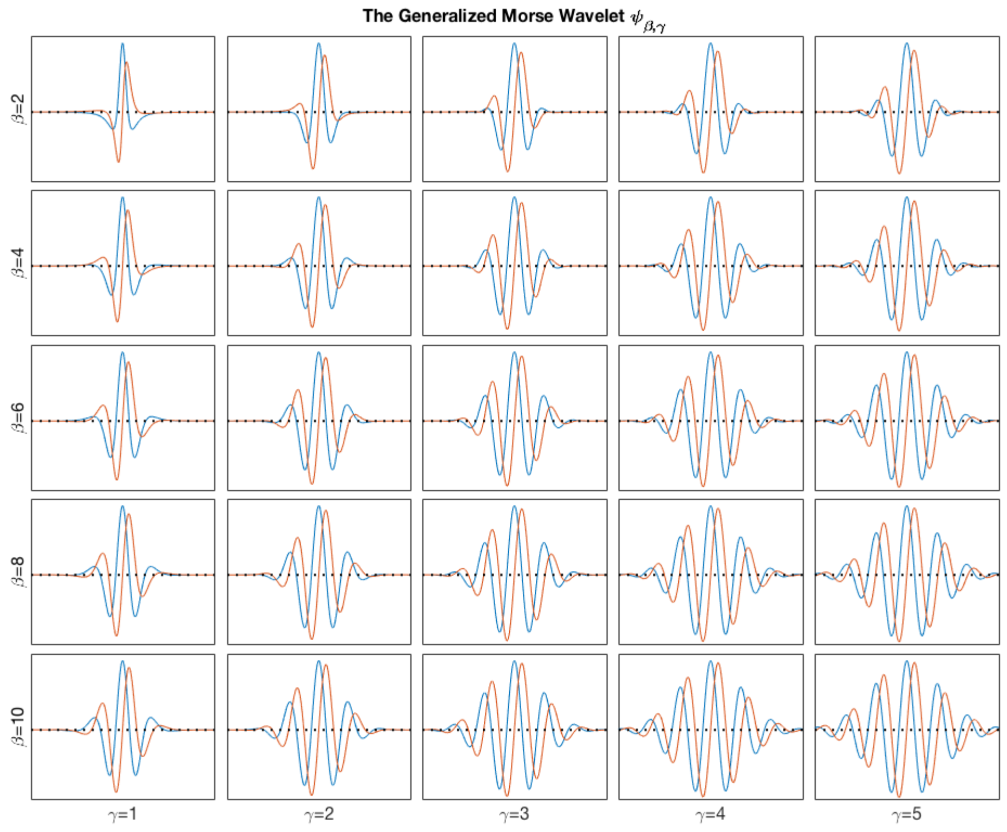


FIGURE 2.8. The GMWs with various parameters in the time domain. The blue and red colours in the plots correspond to real and imaginary parts.

as the Dirac delta function. Therefore, increasing the value of γ from $\psi_{0,\gamma=1}(t)$ can be done by warping in the frequency domain.

For $\beta > 0$, we can obtain $\psi_{\beta,\gamma}(t)$ by differentiation in time domain [**80**]:

$$\psi_{\beta,\gamma}(t) = \alpha_{\beta,\gamma}(-\mathrm{i})^{\beta}\frac{1}{2}\frac{\mathrm{d}^{\beta}}{\mathrm{d}t^{\beta}}\psi_{0,\gamma}(t) \tag{2.28}$$

Hence, the value of β from $\psi_{0,\gamma}(t)$ can be increased by differentiation with respect to β in the time domain. While the parameter γ controls the decay of high-frequency content of the wavelet, we will show that β controls the decay in time domain. Since expression in the second equality of Equation

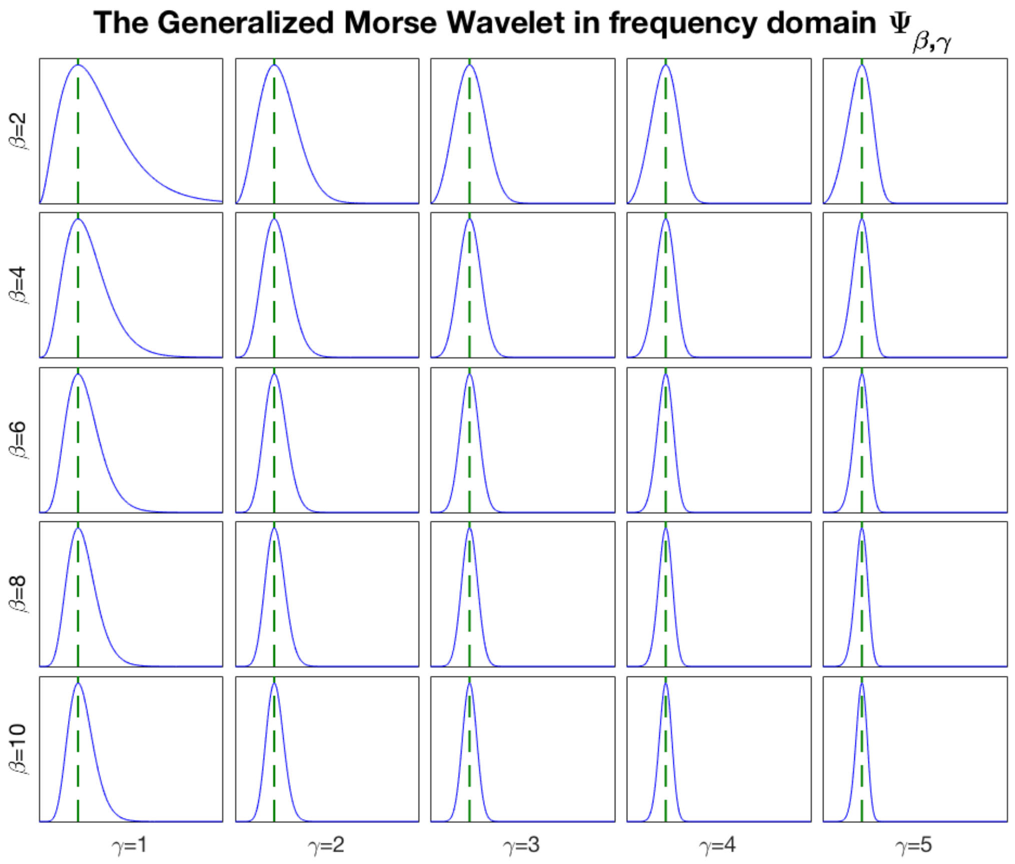


FIGURE 2.9. The GMWs with various parameters in the frequency domain.

(2.29) is Abel summable [**133**], we can interchange the integral and summation such that

$$
\begin{aligned}
\psi_{\beta,\gamma}(t) &= \frac{1}{2\pi}\int_0^\infty \alpha_{\beta,\gamma}\omega^\beta \mathrm{e}^{-\omega^\gamma}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\
&= \frac{1}{2\pi}\int_0^\infty \alpha_{\beta,\gamma}\sum_{s=0}^\infty \frac{(-1)^s}{s!}\omega^{\gamma s+\beta}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\
&= \alpha_{\beta,\gamma}\sum_{s=0}^\infty \frac{(-1)^s}{s!}\frac{1}{2\pi}\int_0^\infty \omega^{\gamma s+\beta}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \qquad (2.29)\\
&= \alpha_{\beta,\gamma}\sum_{s=0}^\infty \frac{(-1)^s}{s!}\mathrm{e}^{\frac{\mathrm{i}\pi(s\gamma+\beta+1)}{2}}\frac{\Gamma(s\gamma+\beta+1)}{t^{s\gamma+\beta+1}} \\
&\sim \alpha_{\beta,\gamma}\mathrm{e}^{\frac{\mathrm{i}\pi(\beta+1)}{2}}\frac{\Gamma(\beta+1)}{t^{\beta+1}},\ \ |t|\to\infty
\end{aligned}
$$

Hence, the wavelet $\psi_{\beta,\gamma}(t)$ has a $O(t^{-(\beta+1)})$ asymptotic behaviour. From Figure 2.8, we see the the more rapid long-time decay as β increases. Figure 2.8 further shows that the filter central portion broadens when β increases. On the other hand, as γ increases, the curvature of the envelope of the filter reduces at the center, therefore leading to a broader central portion, while not altering the long-time decay.

There are several cases on the parameter pair (β, γ) that have been investigated:

(1) When $\gamma = 1$, the GMWs are regarded as "Cauchy" wavelets [**138**], and are equivalent to a solution to Schrödinger equation suggested by Morse [**87**]. The wavelet form in the frequency domain is a generalized Gamma distribution [**75**,**113**]. The generalized Gamma distribution has been reexamined with applications in physics of elementary particles and fields [**52**]. In particular, if $\beta = 0$, the Morse wavelet is analytic Cauchy wavelet

$$\begin{aligned}\psi_{0,1}(t) &= \frac{1}{\pi}\int_0^\infty \mathrm{e}^{-\omega}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\ &= \frac{1}{\pi(1-\mathrm{i}t)} \\ &= \frac{1}{\pi(1+t^2)} + \mathrm{i}\frac{t}{\pi(1+t^2)}.\end{aligned} \tag{2.30}$$

If $\beta \geq 1$, by Equation (2.28), the wavelet has the from

$$\psi_{\beta,1}(t) = \left(\frac{\mathrm{e}}{\beta}\right)^\beta \frac{\Gamma(\beta+1)}{\pi(1-\mathrm{i}t)^{\beta+1}}. \tag{2.31}$$

(2) When $\gamma = 2$, the GMWs are the analytic "Derivative of Gaussian" (DoG) wavelets [**80**]. Lilly and Olhede [**80**] also mentioned that this class of wavelets is less ideal for the analysis of oscillatory signals than the class of wavelets with $\gamma = 3$.

When $\beta = 0$, we have

$$\begin{aligned}\psi_{0,2}(t) &= \frac{1}{\pi}\int_0^\infty \mathrm{e}^{-\omega^2}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\ &= \frac{1}{2\sqrt{\pi}}\mathrm{e}^{-t^2/4} + \frac{\mathrm{i}}{\pi}\mathrm{e}^{-t^2/4}\int_0^{t/2}\mathrm{e}^{u^2}\,\mathrm{d}u.\end{aligned} \tag{2.32}$$

By differentiating β times on the wavelet $\psi_{0,2}$, we obtain

$$\psi_{\beta,2}(t) = \alpha_{\beta,2}(-\mathrm{i})^{\beta}\frac{1}{2}\frac{\mathrm{d}^{\beta}}{\mathrm{d}t^{\beta}}\psi_{0,2}(t). \tag{2.33}$$

The doublet of parameters $(\beta, \gamma) = (2m, 2)$ for $m \in \mathbb{N}$ can be exactly computed using the discrete Fourier transform [**92**].

(3) When $\gamma = 3$, the GMWs are known as “Airy wavelets”. They approximate Gaussian distribution closely while being exactly analytic [**80**]. This family of wavelets can be derived from the inhomogeneous Airy function $\mathrm{Hi}(z)$ [**54**, p. 448]:

$$\mathrm{Hi}(z) = \frac{1}{\pi}\int_0^{\infty} \mathrm{e}^{-u^3/3}\mathrm{e}^{zu}\,\mathrm{d}u. \tag{2.34}$$

When $\beta = 0$, the wavelet has the form

$$\begin{aligned}\psi_{0,3}(t) &= \frac{1}{\pi}\int_0^{\infty} \mathrm{e}^{-\omega^3}\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega \\ &= \frac{1}{3^{1/3}}\mathrm{Hi}\left(\frac{\mathrm{i}t}{3^{1/3}}\right).\end{aligned} \tag{2.35}$$

Again, we can obtain $\psi_{\beta,3}(t)$ for any $\beta > 0$ by Equation (2.28).

(4) When $\gamma = 4$, the GMWs are the “Hyper-Gaussian” wavelets [**80**]. The analytic “Derivative of Gaussian” (DoG) wavelet $\psi_{0,2}(t)$ can be produced by wrapping the analytic Cauchy filter $\Psi_{0,1}$ in the frequency domain such that

$$\psi_{0,2}(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \Psi_{0,1}(\omega^2)\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega. \tag{2.36}$$

Then the “Hyper-Gaussian” wavelet can be given by

$$\psi_{0,4}(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \Psi_{0,2}(\omega^2)\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega. \tag{2.37}$$

We have discussed different family of generalized Morse wavelets above. The numerical implementation and experiment by Lilly and Olhede [**78, 80**] illustrate that the GMWs are supported only on positive frequencies unlike the Morlet wavelets. Thus the statistical properties will not be destroyed due to departures from analyticity if one adopts the GMWs.

There are extensive applications for the estimation of characteristics in non-stationary signals using GMWs. For instance, Olhede and Walden used Morse wavelets to reduce the noise in quadrature Doppler ultrasound blood flow data [**93**]. The Morse wavelets are also used in solar magnetic field data to detect coherent motion [**90**]. Brittain *et al.* applied multiple Morse wavelets to construct spectra and bivariate statistics on neurophysiological data [**16**]. When $\gamma = 1$, the GMWs are applied in physical fading models for communications [**103**, **135**] and threshold autoregressive conditional duration model in econometrics [**136**].

2.4. Monogenicity: Analyticity in Higher Dimension

As suggested by our research goal to analyze the 2-D signals, we are interested in analyzing the high-dimensional signals with an interpretable representation similar to the analytic representation of the 1-D signals. However, it is more difficult to find the analogue of the analytic signal in higher dimensions and hence the analogue of the amplitude and the phase. Fortunately, we can extend the idea to higher dimensions. The 1-D analytic signal is actually the limit of an analytic function which satisfies the *Cauchy-Riemann equations* in the upper half of the complex plane [**49**]. The materials are essential and drawn from Reference [**?**].

Definition 1. (The hyperanalytic function)
A *hyperanalytic function* is a vector-valued function $\mathbb{k}^{+}(\boldsymbol{x}, \boldsymbol{y})$ with a spatial variable $\boldsymbol{x} \in \mathbb{R}^q$, and associated auxiliary variable $\boldsymbol{y} \in \Gamma^{(p)} := \left\{\boldsymbol{y} : y_i > 0 \text{ for } i = 1, \cdots, p\right\}$ that satisfies the generalized Cauchy-Riemann equations for the auxiliary variable $\boldsymbol{y}$. Note that $p, q \in \mathbb{N}$.

A vector-valued function $\boldsymbol{g}^{+}(\boldsymbol{x})$ which can be expressed as the limit of a hyperanalytic function $\mathbb{k}^{+}(\boldsymbol{x}, \boldsymbol{y})$ as $\boldsymbol{y} \to 0^{+}$ is a *hyperanalytic signal.*

The 2-D Riesz system [**114**] is a natural choice of a 2-D generalization to the Cauchy-Riemann equations for functions of variable $z = x + \mathrm{i}y$. The 2-D Riesz system of equations is defined by the three variables (x_1, x_2, y) and is satisfied by a function $\boldsymbol{F} = (f_1, f_2, f_3)$ such that

$$x_3 = y, \ \sum_{k=1}^{3} \frac{\partial f_j}{\partial x_k} = 0, \ \frac{\partial f_j}{\partial x_k} = \frac{\partial f_k}{\partial x_j}, \ 1 \le j, k \le 3.$$

Any solution of the Riesz system in the upper half-space $y > 0$ is said to be a *monogenic* [**30**].

We then need to introduce the *quaternion*, which has the form

$$e := e_1 + e_2\mathbb{i} + e_3\mathbb{j} + e_4\mathbb{k} \in \mathbb{H},$$

where $\mathbb{H}$ is the 4-D real associative algebra of the quaternions, $\mathbb{i}, \mathbb{j}, \mathbb{k}$ are the quaternion units and $e_l \in \mathbb{R},\ l = 1, \cdots, 4$. The quaternion units satisfy the following multiplication rules:

$$\mathbb{i}^2 = \mathbb{j}^2 = \mathbb{k}^2 = -1;\ \mathbb{i}\mathbb{j} = -\mathbb{j}\mathbb{i} = \mathbb{k};\ \mathbb{j}\mathbb{k} = -\mathbb{k}\mathbb{j} = \mathbb{i};\ \mathbb{k}\mathbb{i} = -\mathbb{i}\mathbb{k} = \mathbb{j}.$$

In addition, if $e \in \mathbb{H}$, then we have $e^* = e_1 - e_2\mathbb{i} - e_3\mathbb{j} - e_4\mathbb{k}$, and $\|e\| = \sqrt{ee^*} = \sqrt{e_1^2 + e_2^2 + e_3^2 + e_4^2}$. We need the notion of quaternion when we introduce: (1) the definition of the Fourier transform of a high-dimensional signal by identifying the quaternion unit $\mathbb{j}$ as the imaginary unit i; and (2) the Riesz transform for two orthogonal directions, x_1 and x_2. The Fourier transform of of a d-dimensional signal $g(\boldsymbol{x})$ is expressed by

$$G(\boldsymbol{\xi}) := \int_{\mathbb{R}^d} g(\boldsymbol{x}) \mathrm{e}^{-2\pi\mathbb{j}\boldsymbol{\xi}^\mathsf{T}\boldsymbol{x}} \mathrm{d}\boldsymbol{x}.$$

If $\boldsymbol{k}^+(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathbb{R}^2$, is a monogenic function, then the *anti-monogenic* function

$$\boldsymbol{k}^-(\boldsymbol{x}, y) := \boldsymbol{k}^{+*}(\boldsymbol{x}, y)$$

is a solution of the Riesz system in the lower half-space $y < 0$. Let r_l be the *Riesz kernel* [**114**] [**48**]. Define R_l to be the Fourier transform of r_l, and $\mathcal{R}$ be the *Riesz transform* of a given signal. There exists some constant $c = 1/(2\pi)$ for dimensional normalization such that

$$\begin{aligned} r_l(\boldsymbol{x}) &:= c\,\frac{x_l}{\|\boldsymbol{x}\|^3} \\ R_l(\boldsymbol{\xi}) &:= -\mathbb{j}\frac{\xi_l}{\|\boldsymbol{\xi}\|} \text{ for } l = 1, 2. \end{aligned} \tag{2.38}$$

Also, the Riesz transform is given by

$$
\begin{aligned}
&\mathcal{R}g(\boldsymbol{x}) := \mathbb{i}\mathcal{R}_1 g(\boldsymbol{x}) + \mathbb{j}\mathcal{R}_2 g(\boldsymbol{x}) \\
&\mathcal{R}_l g(\boldsymbol{x}) := g^{(l)}(\boldsymbol{x}) := (r_l * g)(\boldsymbol{x}) = c \text{ p.v.} \int \frac{g(\boldsymbol{x}-\boldsymbol{\tau})\tau_l}{\|\boldsymbol{\tau}\|^3} \mathrm{d}\tau_l \text{ for } l = 1,2.
\end{aligned}
\tag{2.39}
$$

In the 2-D case, if $\boldsymbol{\tau} = \tau_1 + \mathbb{j}\tau_2$, then by Equation (2.38), the Riesz transform

$$
\mathcal{R}g(\boldsymbol{x}) = c \text{ p.v.} \int \frac{g(\boldsymbol{x}-\boldsymbol{\tau})(\tau_1 + \mathbb{j}\tau_2)}{\|\boldsymbol{\tau}\|^3} \mathrm{d}\boldsymbol{\tau}
\tag{2.40}
$$

has the Fourier transform

$$
\mathcal{F}[\mathcal{R}g](\boldsymbol{\xi}) = -\mathbb{j}\frac{\xi_1 + \mathbb{j}\xi_2}{\|\boldsymbol{\xi}\|} G(\boldsymbol{\xi}) = \frac{\xi_2 - \mathbb{j}\xi_1}{\|\boldsymbol{\xi}\|} G(\boldsymbol{\xi}).
\tag{2.41}
$$

Let $y \to 0^+$, the monogenic signal $g^+(\boldsymbol{x})$ can be computed from the hyperanalytic function $\boldsymbol{k}^+(\boldsymbol{x}, y)$. The monogenic signal g^+ and anti-monogenic signal g^- of the signal g are defined by introducing an operator $\mathcal{M}$ [**31**] such that

$$
\begin{aligned}
g^{\pm}(\boldsymbol{x}) &:= \mathcal{M}^{\pm} g(\boldsymbol{x}) \\
&:= g(\boldsymbol{x}) \pm \mathcal{R}g(\boldsymbol{x}) \\
&= g(\boldsymbol{x}) \pm \mathbb{i}g^{(1)}(\boldsymbol{x}) \pm \mathbb{j}g^{(2)}(\boldsymbol{x}).
\end{aligned}
\tag{2.42}
$$

The 1-D analytic signals can be decomposed into amplitude and phase. Similarly, the monogenic signal $g^+(\boldsymbol{x})$ can be further decomposed into amplitude, phase, and phase direction for an easy interpretation [**19**], as described in the following theorem on the local polar representation of the monogenic signal.

THEOREM 2.4.1. *(local polar representation) Let $g^+(\boldsymbol{x})$ be the monogenic signal of the original signal $g(\boldsymbol{x})$. We have the polar representation*

$$
\begin{aligned}
g^+(\boldsymbol{x}) &= \|g^+(\boldsymbol{x})\| \left(\frac{g(\boldsymbol{x})}{\|g^+(\boldsymbol{x})\|} + \frac{\mathbb{i}g^{(1)}(\boldsymbol{x}) + \mathbb{j}g^{(2)}(\boldsymbol{x})}{\|\mathbb{i}g^{(1)}(\boldsymbol{x}) + \mathbb{j}g^{(2)}(\boldsymbol{x})\|} \frac{\|\mathbb{i}g^{(1)}(\boldsymbol{x}) + \mathbb{j}g^{(2)}(\boldsymbol{x})\|}{\|g^+(\boldsymbol{x})\|} \right) \\
&= A(\boldsymbol{x})(\cos\phi(\boldsymbol{x}) + \nu(\boldsymbol{x})\sin\phi(\boldsymbol{x})),
\end{aligned}
\tag{2.43}
$$

where

$$
A(\boldsymbol{x}) := \|g^+(\boldsymbol{x})\| = \sqrt{|g(\boldsymbol{x})|^2 + |g^{(1)}(\boldsymbol{x})|^2 + |g^{(2)}(\boldsymbol{x})|^2}
$$

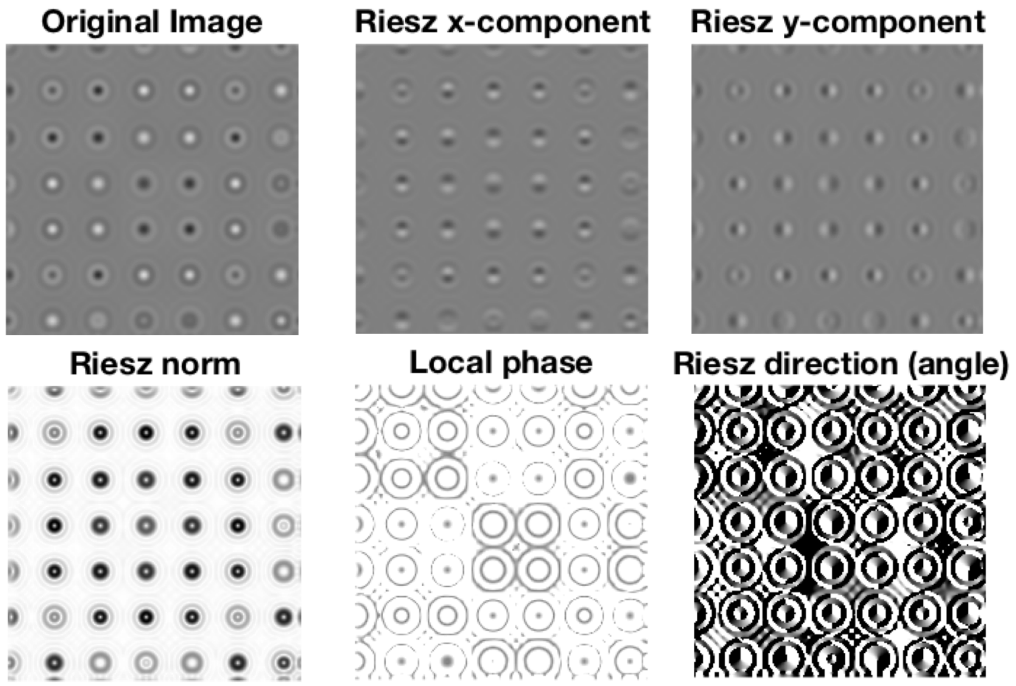


FIGURE 2.10. The monogenic representation of a 2D chirp signal.

denotes the amplitude *of the monogenic signal* $g^+(\boldsymbol{x})$,

$$\phi(\boldsymbol{x}) := \cos^{-1}\left(\frac{g(\boldsymbol{x})}{\|g^+(\boldsymbol{x})\|}\right)$$

denotes the local phase *and*

$$\nu(\boldsymbol{x}) := \frac{\mathbb{i}g^{(1)}(\boldsymbol{x}) + \mathbb{j}g^{(2)}(\boldsymbol{x})}{\|\mathbb{i}g^{(1)}(\boldsymbol{x}) + \mathbb{j}g^{(2)}(\boldsymbol{x})\|}$$

denotes the local phase direction*.*

2.4.1. Example of monogenic representation. Theorem 2.4.1 outlines the local polar representation of a monogenic signal. We provide two examples to illustrate the monogenic representation from some 2-D signals.

The first example is the monogenic representation of a 2-D chirp signal. Let

$$g(x, y) = \cos(1000x^2 + 1000y^2).$$

We sample the variables $x = y = 0 : \frac{1}{100} : 2$. We can compute the mongenic signal $g^+(x, y)$ by Equation (2.41) in the Fourier domain. In particular, the Riesz x-component $g^{(1)}$ corresponds to the $\mathbb{i}$ component of the Riesz transform in Equation (2.39), while the Riesz y-component $g^{(2)}$ corresponds to the $\mathbb{j}$ component of the Riesz transform as shown in Figure 2.10. The figure illustrates that the Riesz x-component analytically extracts the horizontal texture, while the Riesz y-component analytically extracts the vertical texture of the 2-D chirp signal. In other words, the monogenic signal

$$g^{+}(x,y) = g(x,y) + g^{(1)}(x,y)\ \mathbb{i} + g^{(2)}(x,y)\ \mathbb{j}$$

can be visually represented by the top three subfigures in Figure 2.10. By Equation (2.43), we can obtain the Riesz amplitude $A(\boldsymbol{x})$, the local phase $\phi(\boldsymbol{x})$ and the local phase direction (angle) $\nu(\boldsymbol{x})$ from the monogenic signal g^{+}:

(1) The amplitude $A(\boldsymbol{x})$ is the monogenic amplitude for phase-invariant contour detection.

(2) The local phase $\phi(\boldsymbol{x})$ captures and classifies the shape of the contour such as edges and lines.

(3) The local phase direction $\nu(\boldsymbol{x})$ captures the local spatial orientation.

These three new components can be viewed by the bottom three subfigures in Figure 2.10.

We also visually illustrate the monogenic representation of the well-known “Barbara” image in the research community in Figure 2.11. The “Barbara” image was initial applied a low pass filter for signal de-noising. Then we extract the Riesz-x and Riesz-y components from the image that place emphasis respectively on the horizontal and vertical lines or edges. Again, we can compute the amplitude A, the local phase ϕ and the local phase direction (angle) ν from the monogenic signal by Equation (2.43). In particular, the local phase captures sharp edges and lines.

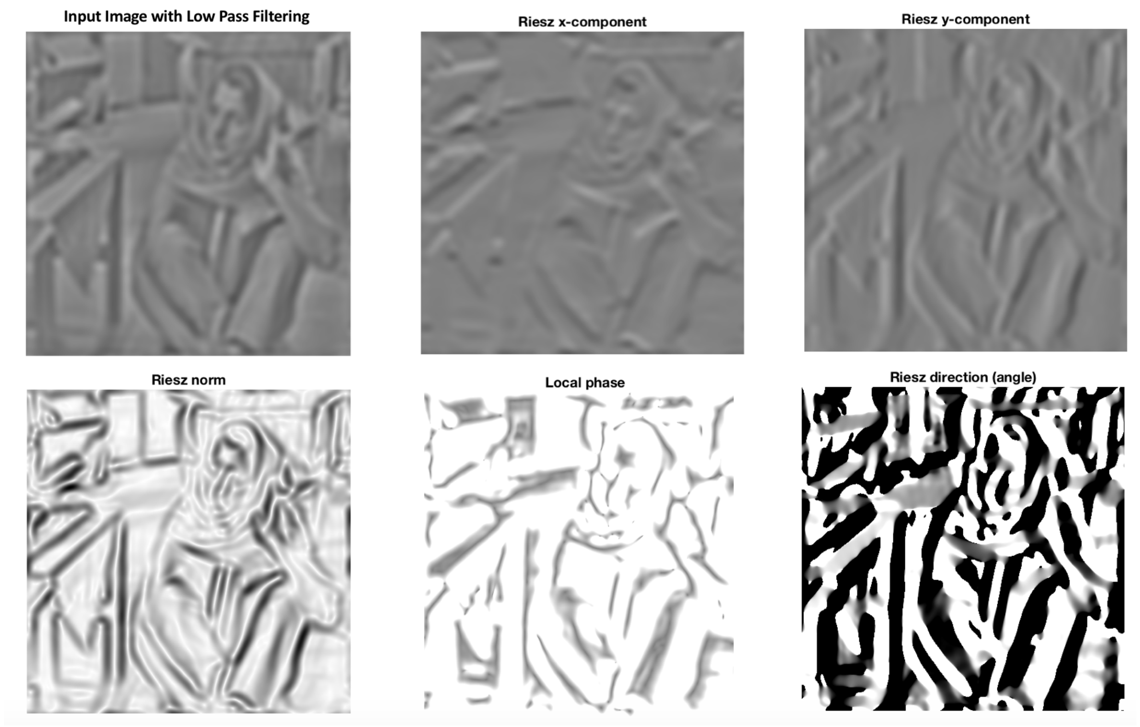


FIGURE 2.11. The monogenic representation of the Barbara image.

CHAPTER 3

From Convolutional Neutral Network to Scattering Transform

With the explosion of data and the availability of cloud computing resources to store and process gigantic amount of data, neural network approach has become popular in image and signal processing. After presenting the concept of neural network and *convolutional neural network* (CNN), we will review one application of convolutional neural network in image restoration from Reference [**20**] authored by the dissertation author as the first author. Then we will discuss *scattering transform* (ST) which was used to develop mathematical theory of deep CNN by Mallat [**82**].

3.1. Overview of Neural Network

3.1.1. Fundamentals of neural network. Neural network is a network of biological neurons that are connected to other neurons with different weights. The inputs taken by the network are combined with different weights through linear combinations. The resulting output then passes through a nonlinear activation layer such as the commonly-known ReLU layer [**1**] to deliver the activated output for the next layer. This algorithmic process is called a *perceptron*. When we combine multiple perceptrons in a sequential order, we can form a feedforward neural network with multiple layers, and the new algorithmic process is known as *multi-layer perceptron* (MLP). The layers that are between the input layer and the output layer are called "hidden" layers.

The process of learning the weights between neurons is called *backpropagation*. The weights are updated backward through stochastic gradient descent (SGD) [**4**], which is a method to iteratively optimize an objective function using gradient descent. Nevertheless, a deep neural network is difficult to train through backpropagation due to the enormous amount of parameters to update. One common issue in training is the *gradient vanishing problem* [**94**], where we completely terminate the learning process as backpropagation employs chain rule to compute gradients through multiplication. Since the magnitudes of gradients are exponentially smaller with more layers, the neural network stops updating these gradients at early stages.

There are some other common stochastic optimization methods to address the issue of small and noisy gradient, such as *Adam* [**65**] which is an algorithm to update stochastic gradients based on adaptive estimates of lower-order moments and *Momentum* [**118**] which is a method to accelerate learning in the directions of low curvature in the surface of the objective function together with SGD, while keeping stability in the direction of high curvature in optimization. Together with the feedforward process, we train a neural network that can require searching considerable amount of hyperparameters and updating a gigantic amount of weights.

However, the complexity of the neural network often leads to *overfitting* when we do not have sufficient amount of data required for training the neural network and addressing variability referring to the estimate on how the model prediction varies given different training data. Increasing the sample size with quality is always the most straightforward solution to reduce variability, but we may not acquire such gigantic amount of data for training the neural network. There are other different ways to reduce variability in model prediction:

(1) Standardize features fed into the neural network to zero mean and unit variance.
(2) Add *dropout layers* [**112**] in the neural network to randomly deactivate some portion of neurons at each step of training.
(3) Perform *batch normalization* [**56**], meaning that we batchwisely normalize the values after activation in order to reduce variability in the hidden layer representation.
(4) Perform *transfer learning* by using pre-trained model, such as a model trained from ImageNet [**28**], and an additional layer with specific training proposes.

A specific type of neural network that has wide range of applications is *convolutional neural network* (CNN), which captures the spatial dependencies of images through filters. In general, the convolutional layers extract underlying features from an image, and the pooling layers extract the most prominent features locally and reduce the dimension from the previous layers [**58**]. Lastly, the intermediate output is mapped into the fully connected layer for classification, regression or other machine learning purposes.

3.1.2. Generative adversarial networks. The CNN structure has been used in many deep networks. For example, Goodfellow at al. [**43**] proposed generative adversarial networks (GANs) that define two different competitors with CNN structures: the generator G_θ and the discriminator

D_ξ, where θ, ξ are parameters in the generator and the discriminator respectively. The generator synthesizes samples from a noise space $\mathcal{Z}$ while a discriminator discriminates generated sample $G_\theta(\boldsymbol{z}_i)$ and actual sample $\boldsymbol{y}_i$. The main goal of the generator is to synthesize samples that are perceptually persuasive and difficult to be distinguished by the real samples. We can describe the competition between the generator G and the discriminator D through the formulation of minimax objective:

$$\min_G \max_D \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}[\log D(\boldsymbol{x})] + \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[\log(1 - D(\tilde{\boldsymbol{x}}))], \tag{3.1}$$

where $\mathbb{P}_r$ denotes the data distribution and $\mathbb{P}_g$ denotes the distribution generated by $\tilde{\boldsymbol{x}} = G(\boldsymbol{z})$, where the vector $\boldsymbol{z}$ is sampled from a noise distribution. i.e., $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, I_n)$. The main benefit of GANs is the capability to synthesize clear samples of high perceptual quality.

However, as described by Salimans *et al.* [**107**], there are undesirable issues such as vanishing gradients, which has been discussed in Section 3.1.1, and mode collapse in the training. The mode collapse problem occurs when the generator synthesizes only one or a few outputs. The conceptional reason of the existence of the problem is that the generator too effectively discovers a few outputs that can deceive the discriminator easily in the minimax competition (3.1). So the generator keeps producing certain outputs only.

The challenges can also be explicated by equivalence of minimizing the objective function for GANs and minimizing the Jensen-Shannon (JS) divergence between the data and model distributions. The latter objective often leads to vanishing gradients. Arjovsky *et al.* [**6**] introduced the weaker Wasserstein-1 distance $W(\mathbb{P}_r, \mathbb{P}_g)$ to address the gradient vanishing problem and provide clear gradients almost everywhere in the GAN model. The competition between the two networks is then reformulated as the following minimax optimization objective:

$$\min_G \max_{D\in\mathcal{D}} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}[D(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}[D(\tilde{\boldsymbol{x}})], \tag{3.2}$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions. The original enforcement of the Lipschitz constraint suggested by Arjovsky *et al.* [**6**] is to clip the weights to $[-c, c]$. Gulrajani *et al.* [**46**] proposed another approach by adding the term for gradient penalty:

$$\lambda \mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}} \sim \mathbb{P}_g} [(\|\nabla_{\tilde{\boldsymbol{x}}} D(\tilde{\boldsymbol{x}})\|_2 - 1)^2]. \tag{3.3}$$

Hyperparameter tuning is not required and the method is robust to the architecture of generator. In contrast to the conventional CNN, GANs can synthesize much clearer images. WGAN-ℓ_1 proposed by Kupyn *et al.* [**69**] has shown effectiveness in deblurring images.

3.2. Application of Convolution Neural Network: Turbulence Removal Network

As discussed in the last section, we sometimes need to deal with the data scarcity problem for training a neural network. For instance, the direct use of deep learning approaches are not feasible for image reconstruction from atmospheric turbulence. The goal for restoring turbulence-distorted images is to remove both geometric distortions and space-time varying blurs. It is challenging to simultaneously discard both the geometric distortions and blurs by mathematical modelling. However, we lack sufficient turbulence-distorted video frames for training a neural network. Therefore a simple and effective data augmentation approach was developed to overcome the data scarcity problem. The approach samples real atmospheric turbulence with deformation fields and different extent of blurs to generate sufficient data for training. A variety of turbulence-distorted videos can be produced from a single image based on the randomly generated artificial turbulence. Normally, the image restoration performance is commensurate with the sample size in training. However, the training sample size requirement is not highly demanding and restrictive in the proposed *Turbulence Removal Network* (TRN) to reconstruct the distorted images under turbulence with data augmentation method. The material of the whole subsection 3.2 is drawn from the previously published work by Chak *et al.* [**20**].

3.2.1. Data Augmentation. Since a massive sample size is required in deep learning, we review a data augmentation algorithm to generate sufficient amount of artificially turbulence-distorted training data. The approach was new, simple and yet effective to reduce variability in deep learning model prediction from initially a small set of data.

To be precise, we transform a single clean image I of height h and width w to another new image I^t with blurs and geometric distortions in the following manner. Firstly, we randomly pick

$(w-2N)\times(h-2N)$ pixel locations, where $N \in \mathbb{N}$. At each randomly picked pixel location (x, y) in the image, we examine a local patch $P^N_{x,y}$ of size $(2N+1)\times(2N+1)$ around the pixel. We then generate a motion vector field $\boldsymbol{V}_{x,y} = (u, v)$ for each local patch $P^N_{x,y}$. For every $p \in P^N_{x,y}$, we sample the stochastic vector $(u(p), v(p))$ from a normal distribution, which is then blurred by a Gaussian kernel and multiplied entry-wisely by a value measuring the distortion strength. The vector field $\boldsymbol{V}_{x,y}$ can be mathematically written as:

$$\boldsymbol{V}_{x,y} = S\ (G_\sigma * \mathcal{N}_1, G_\sigma * \mathcal{N}_2), \tag{3.4}$$

where $\mathcal{N}_1$ and $\mathcal{N}_2$ are random variables drawn from a normal distribution $\mathcal{N}(0,1)$, S is the distortion strength value and G_σ is the Gaussian kernel with standard deviation σ. The vector field $\boldsymbol{V}_{x,y}$ can then be applied to the whole image by padding zero outside the local patch $P^N_{x,y}$. We warp the original image I by $\boldsymbol{V}_{x,y}$ to transform the image, and iterate the whole process by M times.

After M iterations, we obtain the overall motion vector field $\boldsymbol{V} = (u, v)$ by fusing the vector fields on local patches together. In other words,

$$\boldsymbol{V} = \sum_{(x,y)\in\mathcal{Q}} \boldsymbol{V}_{x,y}, \tag{3.5}$$

where $\mathcal{Q}$ is the set of $(w-2N)\times(h-2N)$ randomly picked pixel locations.

Let I^t be an image after transformation according to the previous procedures. Then the transformed image I^t is smoothened by a Gaussian kernel

$$\eta(x) = \mathrm{e}^{-\frac{x^2}{2B^2}},$$

where the parameter B is sampled from the uniform distribution $\mathcal{U}[0.1, 1]$. The transformed image at the final stage under the occurrence of blurs and geometric distortions is given by

$$I_i = \eta * I^t.$$

With these randomized parameters, an original clean image I is transformed into a sequence of video frames $\{I_1, ..., I_n\}$ with blurs and geometric distortions for training a deep neural network. Figure 3.2 displays some images under transformation with different values of distortion strength.

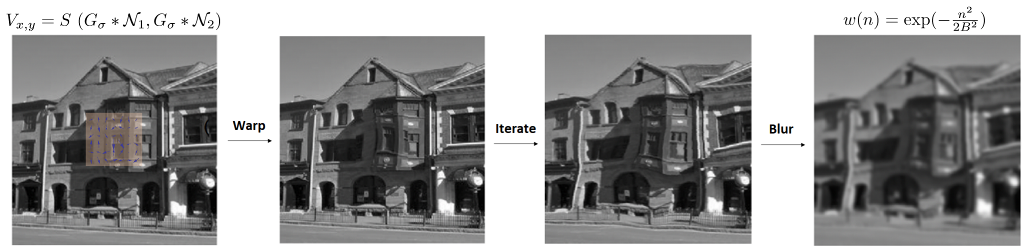
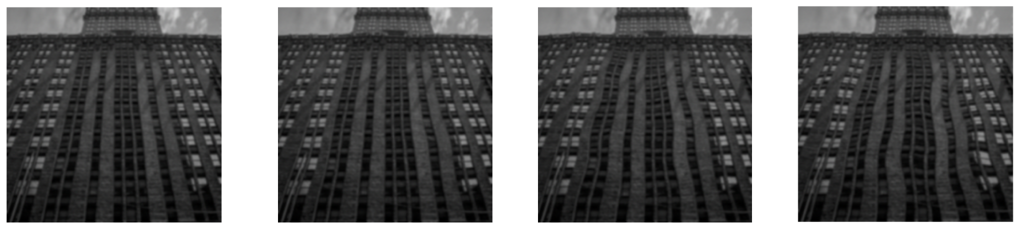

$V_{x,y} = S\ (G_\sigma * \mathcal{N}_1, G_\sigma * \mathcal{N}_2)$
Warp
Iterate
Blur
$w(n) = \exp(-\frac{n^2}{2B^2})$

FIGURE 3.1. The overall procedure of the algorithm for data augmentation.



(a) $S = 0.1$ (b) $S = 0.2$ (c) $S = 0.3$ (d) $S = 0.4$

FIGURE 3.2. Generation of frames under turbulence distortion of different strength S and fixed blur constant $B = 1$ for the propose of illustration. The extent of the distortion on the image can be captured by the strength value S.

Supported by the experimental results later, the data augmentation algorithm successfully covers a wide range of deformations. It also suggests that deep neural network can learn geometric distortions. See Algorithm 1 for the summary of the whole data augmentation algorithm for turbulence-distorted video synthesis. Also, see Figure 3.1 for the overall procedure of the algorithm.

3.2.2. Subsampled Turbulence Removal Network. Based on the fundamentals of the deep CNN as covered in this chapter, we will review the approach, namely, the Turbulence Removal Network (TRN). Figure 3.3 shows that whole network architecture for both our generator network G and the critic network D.

3.2.2.1. *WGAN–ℓ_1 with multiframe input.* The TRN proposed is a WGAN with subsampled multiframe input and the incorporated ℓ_1 penalty in the model. Multiframe input in the TRN is a video created by an original image in greyscale under deformation from turbulence. Then, we train TRN to discard blurs and geometric distortions based on the WGAN architecture. The additional ℓ_1 penalty attempts to extract the important textures of the original image under optimization.

3.2.2.2. *Multiframe input.* The conventional input for GANs is a noise vector $\boldsymbol{z} \in \mathbb{R}^N$ randomly generated from the normal distribution. Then we transform the noise vector $\boldsymbol{z}$ into the desired

Algorithm 1 Blur and Distortion Systhesis

```
    Parameters:
    $N = 32$ : the patch size $2N - 1$
    $M = 1000$ : the number of iterations
    $\sigma$ : the standard deviation of the Gaussian kernel
    $B$ : the blur constant sampled from the uniform distribution $\mathcal{U}[0.1, 1]$
    $S$ : the distortion strength sampled from the uniform distribution $\mathcal{U}[0.1, 0.4]$
    Commands:
    rand: return uniformly distributed random numbers
    randi: return uniformly distributed pseudorandom integers
    randn: returns an $n$-by-$n$ matrix of normally distributed random numbers
 1: procedure DistortBlur (Img, $\sigma$, $N$, $M$, $S$, $B$)
 2:     Create a Gaussian kernel from a Normal distribution $\mathcal{N}(0.2 * \text{rand} - 1, \sigma^2)$.
 3:     for $i = 1 \to M$ do
 4:         $x \leftarrow \text{randi}(-2 * N + \text{width}) + N$
 5:         $y \leftarrow \text{randi}(-2 * N + \text{height}) + N$
 6:         $u(x - N : x + N, y - N : y + N)$
                $\leftarrow u(x - N : x + N, y - N : y + N) + S * \text{randn}$
 7:         $v(x - N : x + N, y - N : y + N)$
                $\leftarrow v(x - N : x + N, y - N : y + N) + S * \text{randn}$
 8:         Convolve the $(u, v)$ vector field with the Gaussian kernel
 9:     end for
10:     Warp the image with $(u, v)$ vector field.
11:     Blur the image by convoluting with a Gaussian smoothing window $w(x) = \mathrm{e}^{-\frac{x^2}{2B^2}}$
12: return Video Frames under Distortion and Blur
13: end procedure
```

output through the generator with a U-Net structure [**101**]. Our network is similar to DeblurGAN [**?**], which has an architecture requiring blurred image as an input and produce an image without blurs. Although space-time varying blur is one of the consequences of turbulence in the frames under observation, one single frame from a turbulence-discarded video as an input is shown to be ineffective in recovering the original image in our experiment.

Hence, using the given architecture from DeblurGAN is not sufficient enough to remove undesirable effects such as the geometric distortions under turbulence. Motivated by this observation, our input in the network is a turbulence-degraded multiframe input originated from a clear image. Therefore, our new architecture is adjusted to use multiple frames as the input. However, instead of using the whole video sequence as the input, only some frames under subsampling are chosen.

With the data augmentation approach described in this subsection, the training data is a multiple frames $I_{\text{TD}} = (I_{\text{TD}}^{(1)}, I_{\text{TD}}^{(2)}, ..., I_{\text{TD}}^{(n)})$ (TD: turbulence-distorted) which are then transformed

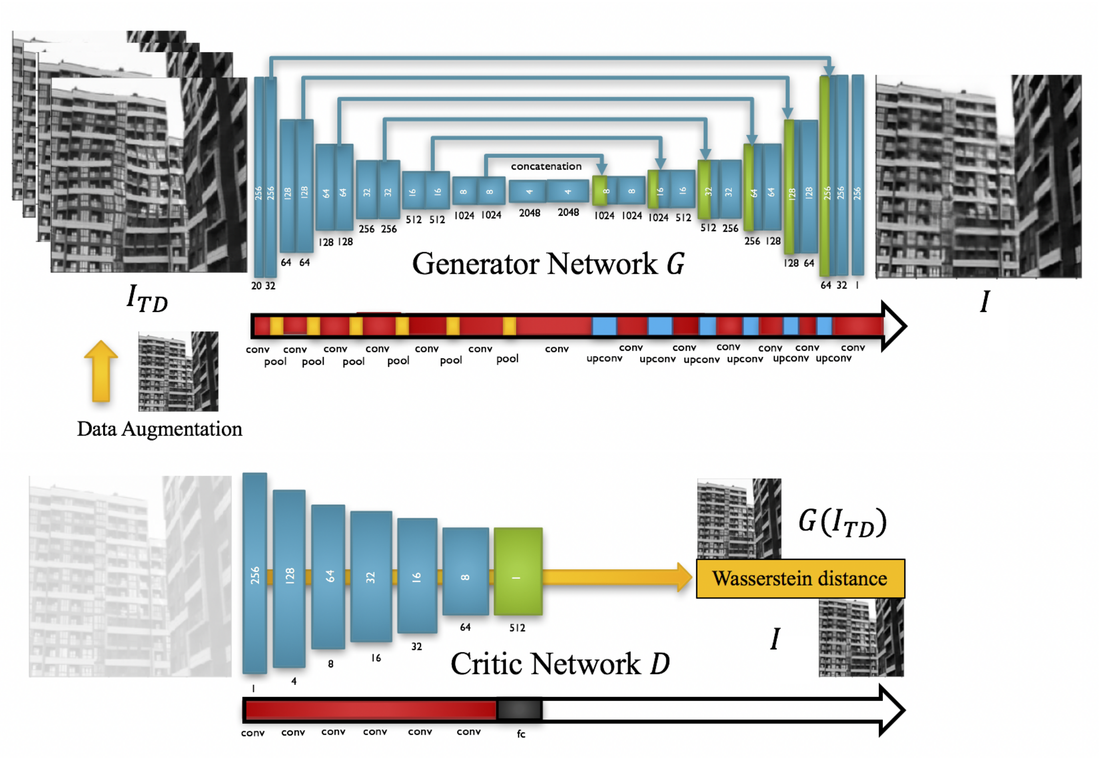


Figure 3.3. The generator network G has the architecture of a U-Net, and the critic network D is the standard conventional CNN. Before passing through into generator network G, the turbulent frames under subsampling are concatenated.

from the original clean image I of size $r \times s$. In TRN, the input is a selected subsampled frames from I_{TD}. Instead of using the whole sequence of frames as an input, we randomly pick $m = 20$ frames from the whole video in training as the generator input in the GAN model. In the testing stage, we combine a subsampling method [**71**] to choose the frames with best quality as the input. The incorporation of the subsampling approach into the network is effective in restoring a significantly better image. As we intend to focus more on the CNN architecture in this dissertation, the section about our proposed subsampling approach will be skipped. See Reference [**20**] for more details.

3.2.2.3. *U-Net architecture in generator network.* The generator network G we employ is the U-Net [**101**], which comprises five different types of layers:

(1) Convolutional layer

(2) De-convolutional layer

(3) Max-pooling layer

(4) ReLU activation layer / Randomized leaky ReLU activation layer ($\alpha = 0.2$) [**134**]

(5) Instance normalization layer [**123**].

The U-Net is known to involve a symmetric expanding path for localization and a contracting path for contextual preservation. It was successful in image segmentation, image denoising and image super-resolution. As a result, we use U-Net as the architecture for our generator network G.

The turbulence-degraded multiframe under subsampling passes through 7 blocks of convolutional layers and 6 blocks of deconvolutional layers to produce a clear image. The first 7 blocks $B_C^{(1)}, B_C^{(2)}, ..., B_C^{(7)}$ consists of convolutional layers, followed by the 6 remaining blocks $B_D^{(1)}, B_D^{(2)}, ..., B_D^{(6)}$ contains deconvolutional layers with kernel size of 2×2 and strike size of 2×2. Each block $B_C^{(i)}$ contains convolutional layers with kernel size of 3×3 and zero padding, nonlinear activation layers and instance normalization layers. The temporal features collected in every block are then down-sampled by max-polling, except for the features of the last block $B_C^{(7)}$. In particular, the temporal features in $B_C^{(6)}$ and $B_C^{(7)}$ are concatenated before passing into the deconvolutional block $B_D^{(1)}$ in order to retain the deep features without too much information loss. The feature extracted in the first block $B_D^{(1)}$ is then concatenated with the feature from the block $B_C^{(5)}$ to output the feature in the second block $B_D^{(2)}$. Repeating the procedure, we obtain a reconstructed clear image I which has the same size as the original undistorted image.

There is no pre-training in the generator network G, since the architecture input is different from the conventional one. The conventional model takes images with the three channels as input. On the other hand, we have subsampled turbulence-degraded video frames $\mathcal{J}$, which are randomly selected in training and chosen by the subsampling method from Reference [**20**] in the testing stage. We then train the generator network G after the critic network D is trained several times to generate a clearer image $I_{\text{TD}} = (I_{\text{TD}}^{(i_1)}, ..., I_{\text{TD}}^{(i_m)})$. The loss function on the generator network G for removing blurs and geometric distortion is defined by

$$L_G = -D(G(I_{\text{TD}})) + \frac{\gamma}{N}\|I - G(I_{\text{TD}})\|_1, \tag{3.6}$$

where γ is a hyperparameter for regularization in the second term. The first term of the total loss function L_G is the adversarial loss which encourages solutions to reside on the manifold of natural images. In order to extract the inherited textures from the turbulence-degraded frames, we further incorporate the ℓ_1 loss into the loss function L_G. The combination of the adversarial loss with the pixel-wise error term has an advantage. It was shown that the minimization of the loss function that consists of only the pixel-wise error term, such as the ℓ_1 or ℓ_2 error, is insufficient to generate a clear image [**74**]. Besides, the ℓ_2 error term can often cause image blur. Therefore, we use ℓ_1 loss, instead of the ℓ_2 loss, in our total loss function L_G to avoid making the image blurry. Experimental results exhibit that the combination of the two terms in the total loss function L_G can effectively discard geometric distortions and undesirable artifacts like image blurs.

3.2.2.4. *Critic network.* In the WGAN [**6**], the critic network D is a deep CNN consisting of convolutional layers, fully-connected layer, ReLU activation layer [**88**], and instance normalization layer [**123**]. We denote the first 6 convolutional layers by $L^{(1)}, L^{(2)}, ..., L^{(6)}$ and the final fully connected layer by $L^{(7)}$. The critic values $D \circ G(I_{\text{TD}})$ and $D \circ I$ are passed through the critic network D to compute the Wasserstein-1 distance

$$\max_D \underset{\boldsymbol{x}\sim\mathbb{P}_r}{\mathbb{E}}[D(\boldsymbol{x})] - \underset{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}{\mathbb{E}}[D(\tilde{\boldsymbol{x}})], \tag{3.7}$$

where D is 1-Lipschitz. We train the critic network D till optimal before updating the generator network G. The loss function of the critic network D in training is given by:

$$\begin{aligned} L_D &= D(G(I_{\text{TD}})) - D(I) \\ &\quad + \lambda\Big(\Big\|\nabla D\Big(\alpha I + (1-\alpha)G(I_{\text{TD}})\Big)\Big\|_2 - 1\Big)^2, \end{aligned} \tag{3.8}$$

where α is randomly generated from the uniform distribution $U[0,1]$. The additional gradient penalty term in Equation (3.8) is robust to the architecture of the generator. Since there is no pre-trained model involved in the generator network G, it takes a longer time foe training the networks. We impose the weight constraint λ further to enforce the 1-Lipschitz assumption in the critic network D. We add the weight constraint by clipping the weights in the interval $[-c, c]$ so that the training process becomes more stable.

3.2.3. Summary of the experimental results from TRN. The image data for training were gathered from Flicker. It comprises 1000 images of chimneys and 1500 images of buildings, each of which is resized to 256×256 and is deformed synthetically by our turbulence generator algorithm. More precisely, each image is deformed to generate hundreds of turbulence-degraded video sequences. Hence, the full dataset is enlarged by a factor of 100. We then test the trained network on more than 400 testing data, which are separated from the training dataset. The testing dataset comprises simulated video sequences as well as the real turbulence-distorted video sequences.

After gathering the data, we conduct experiments in PyTorch [**95**] under a CUDA-enabled GPU. We implement the data augmentation algorithm in MATLAB® before the training process begins. The strength value of distortion S and the blurring parameter B are sampled from uniform distributions $\mathcal{U}[0.1, 0.4]$ and $\mathcal{U}[0.1, 1]$ respectively. We initialize the weights in the convolutional layers and the batch normalization layers by sampling them from the normal distribution $\mathcal{N}(0, 0.01)$. ADAM solver [**65**] is employed as the optimization scheme for gradient descent with a learning rate of 10^{-4}, and ADAM learning parameters $\beta_1 = 0.5$ and $\beta_2 = 0.99$ for both the generator G_θ and the critic network D_ξ. We set 3 gradient descent steps for the critic network D_ξ and then 1 step for the generator G_θ. We then apply the instance normalization and dropout with a dropout rate of 0.5 to improve the training without overfitting. Besides the gradient penalty term [**46**], we enforce the parameters ξ in the range $[-0.01, 0.01]$. For each epoch, we train both the networks with batch size of 1. Then we set $\lambda = 10$ in Equation (3.8) and $\gamma = 1000$ in Equation (3.6). Moreover, we randomly pick $m = 20$ frames from the video sequence as our input. The whole training for 40 epochs takes around 3 days. Figure 3.4 shows that the performance of image restoration is gradually better in the training process. The network firstly removes geometric distortion from the turbulence in the first few epochs and then gradually deblur and preserve the quality and texture of the original image in the remaining epochs. After we thoroughly train the TRN, we test the performance on more than 400 testing data which consist of simulated and real turbulence-degraded videos. The test data before data augmentation are different from the training data. We report some experimental results.

In Figure 3.5, we exhibit the restoration performance on some simulated turbulence-degraded frame sequences that captures various buildings. The first column displays the observed frames from each turbulence-degraded image sequences with both geometric distortions and blurs. The middle column displays the restoration performance by the TRN, without the incorporation of the

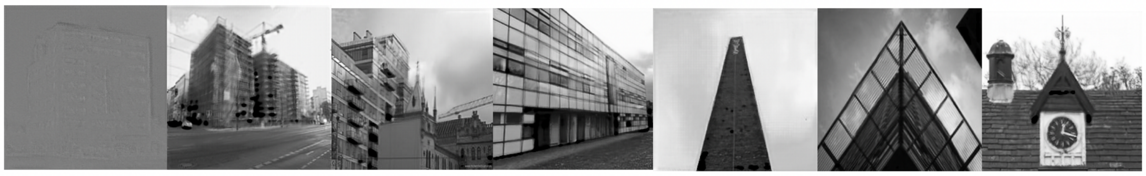


FIGURE 3.4. The training process begins at the 1st epoch (left) to the 7th epoch (right). Each image displayed from its left image, except the first one, are generated with 1-2 epochs. In the first few epochs, the generator network G_θ learns to discard geometric distortion. After removing the distortion to a great extent, it learns to refine the quality and texture of the images, and make it less blurry in the remaining epochs. The performance under training is gradually improved when we increase the number of epoch.

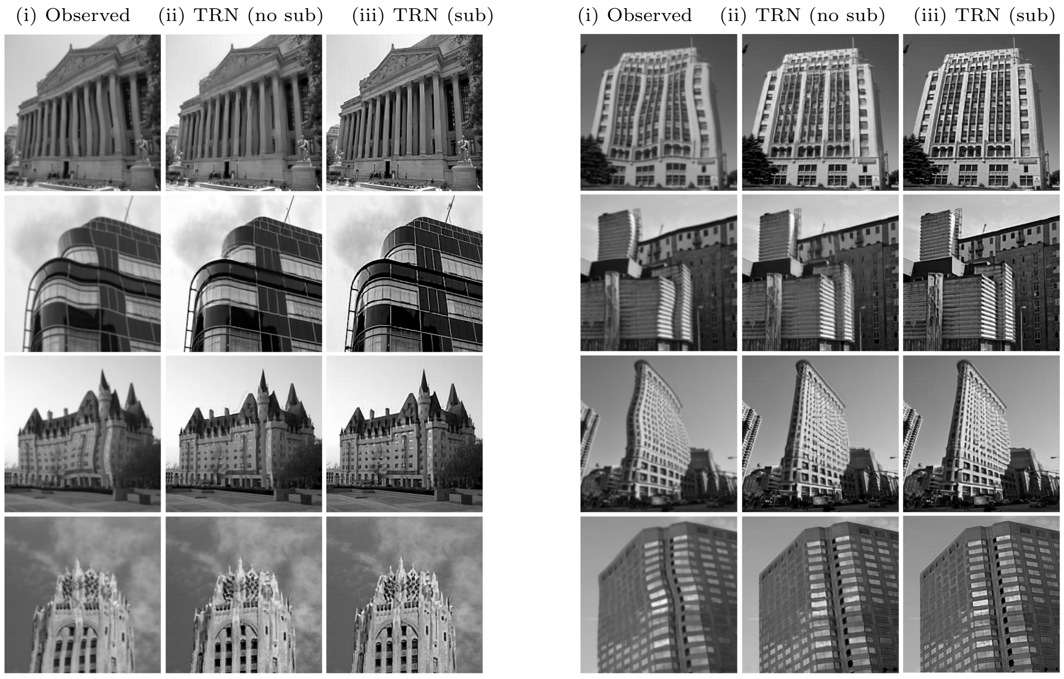


FIGURE 3.5. Restoration of turbulence-distorted "building" images. Column (i) shows the observed frames from each video. Column (ii) shows the restoration results using the proposed TRN without subsampling. Column (iii) shows the restoration results using TRN with subsampling.

proposed subsampling approach. Although some amount of distortions can still be observed, most blurs and geometric distortions are discarded. The right column displays the restoration performance using the TRN with the incorporation of the proposed subsampling approach. With the incorporation of subsampling, blurs and geometric distortions can be discarded more successfully.

Table 3.1. PSNR and SSIM [**126**] of the restored building images with (sub) and without (no sub) subsampling algorithm. The symbol b_i refers to the i-th building image, counting vertically from top-left corner to bottom-right corner.

	b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8
SSIM (no sub)	0.878	0.830	0.846	0.834	0.828	0.806	0.781	0.878
SSIM (sub)	0.904	0.859	0.872	0.861	0.829	0.816	0.830	0.872
PSNR (no sub)	24.8	22.5	26.1	24.0	21.4	22.7	20.2	27.3
PSNR (sub)	25.7	23.4	26.5	24.3	21.6	23.3	20.8	26.9

That is because frames that are highly degraded are successfully filtered out without harming the overall quality of the input. Hence restoration performance is more satisfactory compared to those without subsampling. It demonstrates the benefit of the incorporation of the subsampling method into the deep network. In addition, these visual results are quantitatively evaluated as displayed in Table 3.1.

We also test our deep network on image sequences of chimney. Figure 3.6 shows the restoration performance of some simulated turbulence-distorted image sequences that captures different chimneys. Again, the first column shows the observed frames from each turbulence-degraded image sequences by both geometric distortions and blurs. The middle column shows the restoration results using the TRN without the incorporation of the subsampling approach. The right column shows the restoration results using the TRN with the incorporation of the subsampling approach. With the incorporation of the subsampling method, blurs and geometric distortions can be removed more successfully. The restoration results are more satisfactory compared to those without subsampling. It further demonstrates that the incorporation of the subsampling method into the deep network is beneficial. The results are further shown in Table 3.2.

Other than the simulated examples, we evaluate the performance of TRN on *real* turbulence-degraded videos without a clear ground-truth image. Figure 3.7 shows the restoration results of a real turbulence-degraded image sequence of a chimney. An observed frame from the image sequence is shown in Figure 3.7 (a). Then the restoration results using TRN without and with subsampling is displayed in Figure 3.7 (b) and (c) respectively. With the subsampling method, the results are

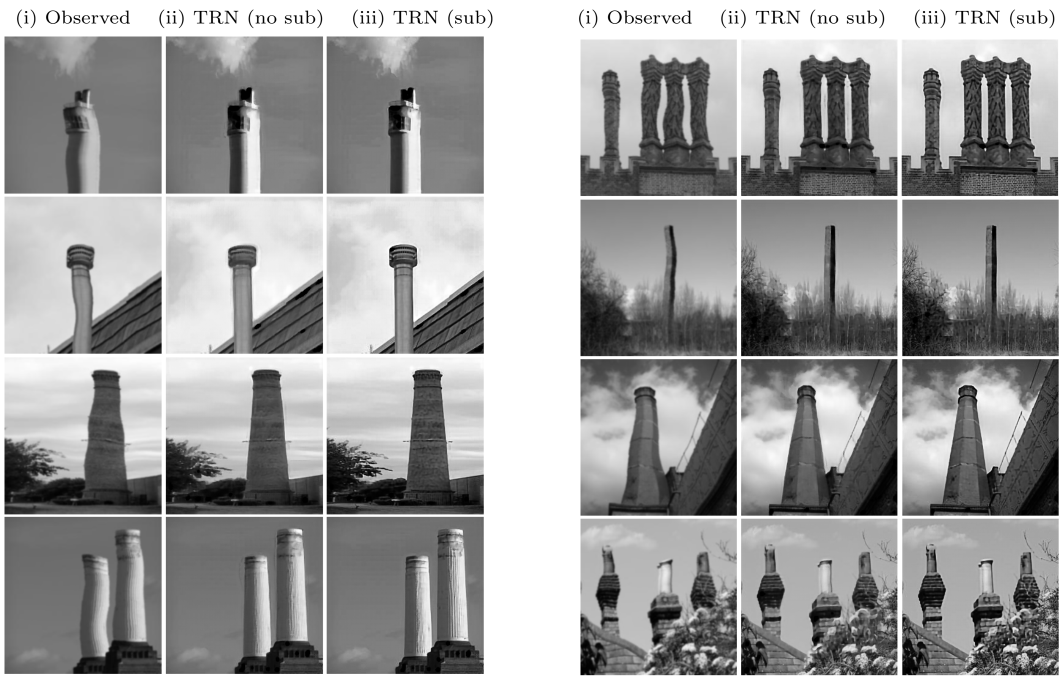


FIGURE 3.6. Restoration of turbulence-distorted "chimney" images. Column (a) shows the observed frames from each video. Column (b) shows the restoration results using the proposed TRN without subsampling. Column (c) shows the restoration results using TRN with subsampling.

TABLE 3.2. PSNR and SSIM of the restored chimney images with (sub) and without (no sub) subsampling method. The symbol c_i corresponds to the i-th Chimney image, counting vertically from top-left corner to bottom-right corner in Figure 3.6.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
SSIM (no sub)	0.948	0.931	0.898	0.897	0.837	0.880	0.896	0.847
SSIM (sub)	0.956	0.951	0.935	0.932	0.914	0.924	0.931	0.931
PSNR (no sub)	25.3	24.5	27.3	27.5	24.7	29.8	25.8	23.6
PSNR (sub)	25.7	26.2	28.7	28.0	25.8	30.4	27.4	24.3

more satisfactory compared to those without subsampling. It again demonstrates the effectiveness of incorporating the subsampling approach into the deep neural network.

Another real example is the turbulence-distorted image sequence capturing a building being restored by the TRN as presented in Figure 3.8. Again, we display an observed frame from the building image sequence, the restored frame using TRN without subsampling and the restored

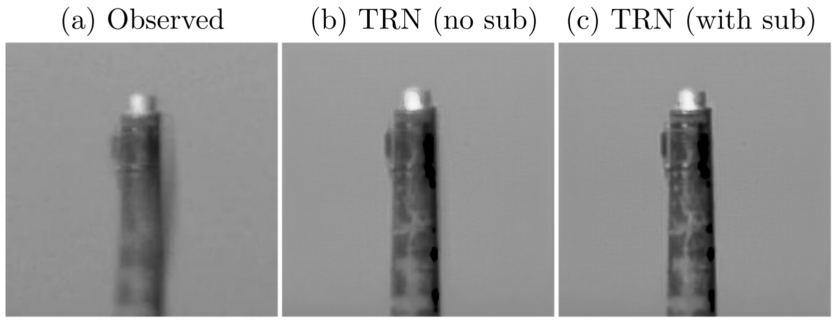


FIGURE 3.7. Restoration of real turbulence-distorted image sequence capturing a chimney. (a) shows an observed frame from the image sequence. (b) shows the restored image using TRN without subsampling. (c) shows the restored image using TRN with subsampling.

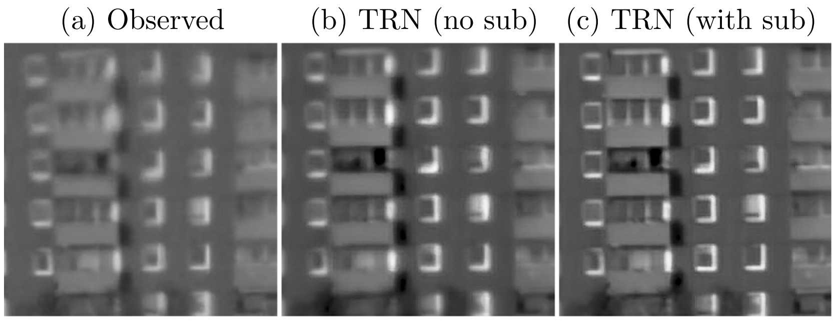


FIGURE 3.8. Restoration of real turbulence-distorted image sequence capturing a building. (a) shows an observed frame from the image sequence. (b) shows the restored image using TRN without subsampling. (c) shows the restored image using TRN with subsampling.

frame using TRN with subsampling in Figure 3.8 (a), (b), and (c) respectively. As before, with subsampling, the results are more satisfactory than those without subsampling. It shows that the turbulent generator is able to synthesize useful data for training and avoid overfitting with just a limited amount of available data. The result shows that it is possible to apply deep learning approach with scarce training data by using data augmentation method. It is similar to human vision in the sense that we learn instead of memorizing knowledge from just a few examples.

3.3. Scattering Transform and the Mathematics of Deep Learning

The data scarcity problem has been widely discussed and addressed by the machine learning community. Often models with high complexity such as a deep neural network may not be necessary in real-life applications. However, the hierarchical manner of locally extracting features like a

CNN is a promising direction to perform tasks in machine learning. We will show that *scattering transform network* (STN) can be an alternative to maintain the advantage of CNN without relying on a huge amount of data for feature extraction. Besides, the STN is a natural choice for interpreting the result from the coefficients of scattering transform.

3.3.1. Importance of More Interpretability in Machine Learning. With the rising applications of data science and modelling, modellers do not only work on minimizing a specific objective function, but have the responsibility to explain or interpret how the model leads to the conclusion, which can be helpful or sometimes even legally necessary [**44**]. Nevertheless, there is often a trade-off between model interpretability and model performance. A linear model is easy to interpret by examining the weight or the p-value of coefficients, but it has a high bias in predicting the true values. An example of linear model is recursive least squares (RLS) adaptive noise cancellation (ANC) filtering for fetal signal extraction, which was used in the published work [**63**] by Kasap *et al.*, with contribution from the dissertation author as a co-author of the work.

Features extracted by some "black-box" machine learning models can be interpreted using the Shapley values [**117**], which are the average marginal contribution of a feature over all combinations of inputs under the game-theoretic formulation. However, the Shapley values can rely heavily on how the machine learning model performs on unrealistic inputs [**38**]. In the setting of deep neural network, it is challenging to interpret the neurons and their significance contributing to the model performance. However, the STN allows interpretability of scattering coefficients and their importance leading to the result.

3.3.2. STN Architecture. A STN has an architecture similar to a CNN with a relatively shallow layers up to M. The STN undergoes layers of wavelet transform which is then followed by the nonlinear and pooling operations. After passing several layers of wavelet transform with nonlinear and pooling operations, we vectorize the resulting coefficients and compress them for feature extraction. We can use linear models or simple machine learning models to perform tasks by feeding these compressed ST coefficients. Figure 3.9 shows the analogue between the STN and CNN.

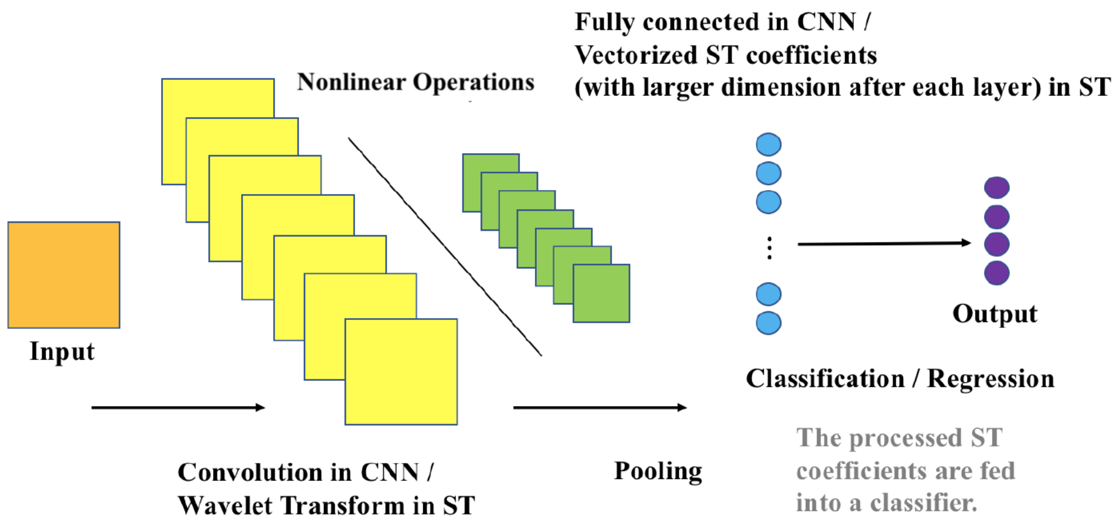


FIGURE 3.9. A comparison between the ST network and the CNN.

We then describe the STN in detail, which can be visualized in Figure 3.10 based on the reference by Mallat [**82**]. At layer m, let $\mathcal{O}_m$ be a finite rotation group in $\mathbb{R}^d$. Let Λ_m be the index set at the m-th layer comprising the rotation $q \in \mathcal{O}_m$ and the scale index $j \in \mathbb{Z}$ and $j > -J$ for some $J \in \mathbb{Z}$. Let ψ be a mother wavelet, such as a Morlet or generalized Morse wavelet. Denote

$$\lambda_m = (q, j) \in \Lambda_m$$

the mult-index of a generator (multiscale directional wavelet filter) which can be acquired by rotating and dilating ψ at layer m. Let ψ_{λ_m} denote the generator, and its formula is provided by

$$\psi_{\lambda_m}(\boldsymbol{x}) := 2^{\frac{2j}{Q_m}} \psi(2^{\frac{j}{Q_m}} q^{-1}\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^d, \tag{3.9}$$

where $Q_m \in \mathbb{R}^+$ is the quality factor for adaptive scale readjustment. We assume the generator $\psi_{\lambda_m} \in L^1(\mathbb{R}^d) \cap L^2(\mathbb{R}^d)$. The low frequency portion not covered by the generators is captured by a father wavelet φ_0 followed by further scaling:

$$\varphi(\boldsymbol{x}) := 2^{2J} \varphi_0(2^J \boldsymbol{x}). \tag{3.10}$$

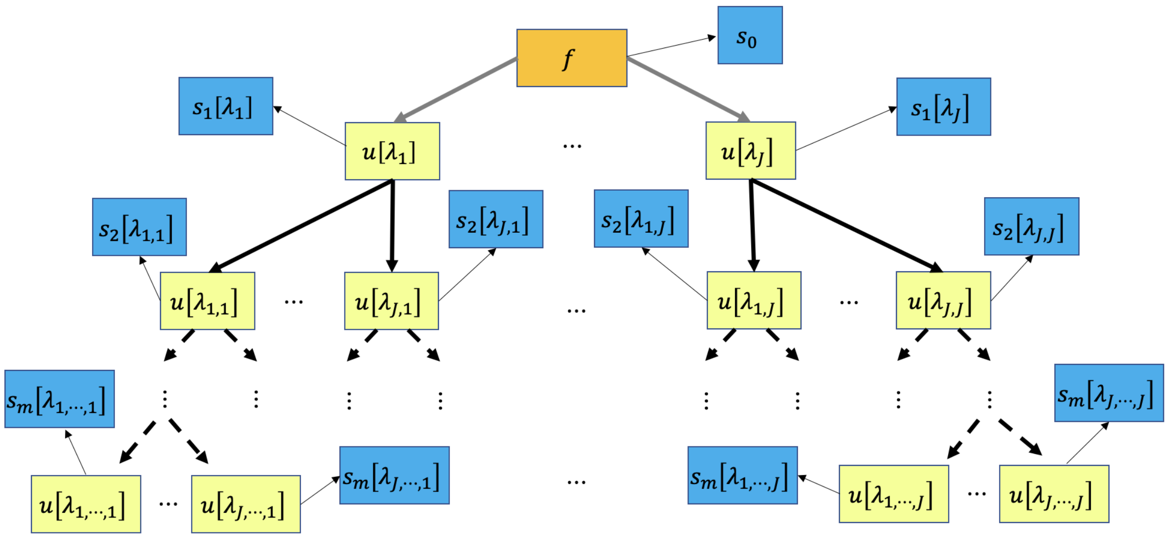


FIGURE 3.10. A typical STN architecture. The multi-index $\lambda_{j_1,j_2,\cdots,j_m}$ corresponds to the scale j_1 in layer 1, j_2 in layer 2, and so on.

There exists frame bounds A_m and B_m such that for any $f \in L^2(\mathbb{R}^d)$,

$$A_m\|f\|_2^2 \le \|f \star \varphi\|_2^2 + \sum_{\lambda\in\Lambda_m} \|f \star \psi_{\lambda_m}\|_2^2 \le B_m\|f\|_2^2. \tag{3.11}$$

In the CNN, we extract features by convolution followed by a nonlinear operation and a pooling operation such as averaging pooling [**58**] and maximum pooling [**58**, **99**]. Meanwhile, we outline the wavelet transform in the setting of ST which is convolution with filters extracted from a mother wavelet with scaling and rotation. To be precise, let $f \in L^2(\mathbb{R}^d)$, and define a *translation* operator

$$T_{\boldsymbol{b}} f(\boldsymbol{x}) := f(\boldsymbol{x} - \boldsymbol{b}), \quad \boldsymbol{b} \in \mathbb{R}^d,$$

and an *involution* operator operator

$$If(\boldsymbol{x}) := \overline{f(-\boldsymbol{x})}.$$

Then, a *frame atom* is defined by

$$\psi_{\boldsymbol{b},\lambda_m} := T_{\boldsymbol{b}} I \psi_{\lambda_m}.$$

which corresponds to a receptive field in a layer of a CNN [**132**]. Note that $\langle f, \psi_{\boldsymbol{b},\lambda_m}\rangle = f * \psi_{\lambda_m}(\boldsymbol{b})$.

A *contraction* operator M_m which is *Lipschitz continuous* with a Lipschitz bound l_m can also be defined as a nonlinear operator, and satisfies

$$M_m f(\boldsymbol{x}) = 0 \Rightarrow f(\boldsymbol{x}) = 0.$$

One particular choice of M_m is the *modulus* operator. That is,

$$M_m f(\boldsymbol{x}) := |f(\boldsymbol{x})|.$$

We define an operator $U_m : \Lambda_m \times L^2(\mathbb{R}^d) \to L^2(\mathbb{R}^d)$ from layer $m-1$ to layer m such that

$$U_m[\lambda_m] f(\boldsymbol{x}) := M_m(f * \psi_{\lambda_m})(r_m \boldsymbol{x}), \tag{3.12}$$

where $r_m \geq 1$ denotes a *subsampling rate* for average pooling as the default setting in the ST package `ContinuousWavelet.jl` [**127**].

The operator U_m is well-defined since

$$\begin{aligned} \|U_m[\lambda_m] f(\boldsymbol{x})\|_2^2 &= \int_{\mathbb{R}^d} |M_m(f * \psi_{\lambda_m})(r_m \boldsymbol{x})|^2 \, \mathrm{d}\boldsymbol{x} \\ &= \frac{1}{r_m^d} \int_{\mathbb{R}^d} |M_m(f * \psi_{\lambda_m})(r_m \boldsymbol{x})|^2 \, \mathrm{d}(r_m \boldsymbol{x}) \\ &= \frac{1}{r_m^d} \int_{\mathbb{R}^d} |M_m(f * \psi_{\lambda_m})(\boldsymbol{y})|^2 \, \mathrm{d}\boldsymbol{y} \\ &= \frac{1}{r_m^d} \|M_m(f * \psi_{\lambda_m})\|_2^2 \\ &\leq \frac{l_m^2}{r_m^d} \|f * \psi_{\lambda_m}\|_2^2 \\ &\leq \frac{B_m l_m^2}{r_m^d} \|f\|_2^2 \end{aligned} \tag{3.13}$$

Therefore, we have a *scattering path* of indices $\boldsymbol{\lambda} \in \Lambda_m \times \cdots \times \Lambda_1$ such that

$$U[\boldsymbol{\lambda}] f(\boldsymbol{x}) := U_m[\lambda_m] U_{m-1}[\lambda_{m-1}] \cdots U_1[\lambda_1] f(\boldsymbol{x}). \tag{3.14}$$

The operator U is also well-defined by the Inequality (3.13):

$$\|U[\boldsymbol{\lambda}] f(\boldsymbol{x})\|_2^2 \leq \left(\prod_{k=1}^{m} \frac{B_k l_k^2}{r_k^d} \right) \|f\|_2^2 \tag{3.15}$$

For each layer m, we define the operators S_m and Φ_m to generate the feature coefficients of the STN for a given input signal $f(\boldsymbol{x})$ by

$$\begin{aligned} S_m[\boldsymbol{\lambda}]f(\boldsymbol{x}) &:= (\varphi * U[\boldsymbol{\lambda}]f)(r'_m \boldsymbol{x}), \\ \Phi_m f(\boldsymbol{x}) &:= \{S_m[\boldsymbol{\lambda}]f(\boldsymbol{x})\}_{\boldsymbol{\lambda} \in \Lambda_m \times \cdots \times \Lambda_1}, \end{aligned} \tag{3.16}$$

where φ_m is an averaging function which is also the father wavelet of a certain scale corresponding to the mother wavelet ψ. The subsampling rate $r'_m \geq 1$ provides another subsampling alternative after the averaging process. Note that for $m = 0$, we have

$$S_0[\emptyset]f(\boldsymbol{x}) = S_0 f(\boldsymbol{x}) := (\varphi_0 * f)(r'_0 \boldsymbol{x}).$$

The feature extractor from the entire scattering transform is denoted by

$$\Phi[f] := \bigcup_{m=0}^{\infty} \Phi_m[f]. \tag{3.17}$$

3.3.3. Theory Behind Scattering Transform. Mallat [**132**] initially formulated the mathematical analysis of deep CNNs for feature extraction. His paper proved a result about translation invariance in the sense that the increasing network depth determines the extent to which the extracted features become progressively more translation invariant.

We assume the *weak admissibility condition* is satisfied in the layers of scattering transform, meaning that the upper bound B_m is sufficiently small relative to the subsampling factor r_m and the Lipschitz bound l_m for the Lipschitz continuous operator M_m.

To be specific, for a m-layer STN, we have

$$B_m \leq \min\{1, l_m^{-2} r_m^{-2}\}. \tag{3.18}$$

Also, the nonlinear Lipschitz continuous operator M_m commutes with the translation operator:

$$M_m T_{\boldsymbol{x}}[f] = T_{\boldsymbol{x}} M_m[f] \tag{3.19}$$

It aligns with the setting in the framework of CNN, where most nonlinearities covered in deep learning literatures are point-wise, hence commuting with the translation operator. These include ReLU [**1**], hyperbolic tangent [**55, 58**] and shifted logistic sigmoid [**42**].

The main difference between the ST in Reference [**132**] and our ST setting is the pooling operator which can be any type of pooling in the reference.

Based on these assumptions, we state the theorems of of Wiatowski and Bölcskei [**132**] on the stability of the resulting feature on small deformation in frequency and space [**131**].

Let $f \in L^2_a(\mathbb{R}^d)$ be a band-limited signal with the support of its Fourier transform limited to $[-a, a]$. Suppose $D_{\boldsymbol{\tau},\boldsymbol{\omega}}$ is a space-frequency deformation operator with respect to a nonlinear distortion $f(\boldsymbol{x} - \boldsymbol{\tau}(\boldsymbol{x}))$ and deformation by modulation $\mathrm{e}^{2\pi \mathrm{i}\boldsymbol{\omega}(\boldsymbol{x})} f(\boldsymbol{x})$ with the form

$$D_{\boldsymbol{\tau},\boldsymbol{\omega}}[f](\boldsymbol{x}) := \mathrm{e}^{2\pi \mathrm{i}\boldsymbol{\omega}(\boldsymbol{x})} f(\boldsymbol{x} - \boldsymbol{\tau}(\boldsymbol{x})) \tag{3.20}$$

For this type of signal class, we have the *deformation sensitivity bound.* The same technique for the bound derivation can be extended to all signal classes that show *deformation insensitivity.*

Definition 2. *A class of signal $\mathcal{C} \subset L^2_R(\mathbb{R}^d)$ is called deformation-insensitive if there exists $C, \alpha_1, \alpha_2 > 0$ such that for any distortion parameter $\tau \in C^1(\mathbb{R}^d, \mathbb{R})$ with $\|\nabla \tau\|_\infty \leq \dfrac{1}{2d}$ and modulation parameter $\boldsymbol{\omega} \in C(\mathbb{R}^d, \mathbb{R})$, we have*

$$\Big\| D_{\boldsymbol{\tau},\boldsymbol{\omega}} f - f \Big\|_2 \leq C\Big(\|\boldsymbol{\tau}\|_\infty^{\alpha_1} + \|\boldsymbol{\omega}\|_\infty^{\alpha_2} \Big) \tag{3.21}$$

Classes of band-limited, cartoon and Lipchitz signals [**45**] are deformation insensitivity. In particular, the band-limited signal $f \in L^2_a(\mathbb{R}^d)$ has the deformation sensitivity bound for the deformation error $\Big\| D_{\boldsymbol{\tau},\boldsymbol{\omega}} f - f \Big\|_2$ given by the following inequality [**132**],

$$\Big\| D_{\boldsymbol{\tau},\boldsymbol{\omega}} f - f \Big\|_2 \leq C \|f\|_2 \Big(a\|\boldsymbol{\tau}\|_\infty + \|\boldsymbol{\omega}\|_\infty \Big). \tag{3.22}$$

To show the deformation-insensitive bound for the feature extractor Φ from the scattering transform, Mallat [**132**] verified that Φ is Lipschitz continuous.

Proposition 3.3.1. *For any function $f \in L^2(\mathbb{R}^d)$, define the norm of the feature extractor Φ by*

$$\Big\|\Phi[f]\Big\| := \sum_{m=1}^{\infty} \sum_{\boldsymbol{q}\in\Lambda^m} \Big\|\varphi_m * U[\boldsymbol{\lambda}]f\Big\|_2. \tag{3.23}$$

By the weak admissibility condition (3.18)*, the feature extractor Φ is Lipschitz continuous. i.e., For any $f, h \in L^2(\mathbb{R}^d)$,*

$$\Big\|\Phi[f] - \Phi[g]\Big\| \leq \|f-h\|_2. \tag{3.24}$$

The immediate result of the Lipschitz continuity of Φ is the robustness with respect to the additive noise $\eta \in L^2(\mathbb{R})$. That is, for any $f \in L^2(\mathbb{R}^d)$,

$$\Big\|\Phi[f+\eta] - \Phi[f]\Big\| \leq \|\eta\|_2. \tag{3.25}$$

By the inequality (3.22) and (3.24), we are ready to establish the deformation-insensitive bound for the feature extractor Φ in the following theorem.

Theorem 3.3.1. *Let $\boldsymbol{\omega} \in C(\mathbb{R}^d, \mathbb{R})$ be the frequency shift and $\boldsymbol{\tau} \in C^1(\mathbb{R}^d, \mathbb{R}^d)$ be the shift in space domain. Define the* shift *operator according to Equation* (3.20)*.*

Denote $L^2_a(\mathbb{R}^d)$ as the set of $L^2(\mathbb{R}^d)$ functions with the support of Fourier transform limited to $[-a, a]$. If $\|\nabla\tau\|_\infty \leq \frac{1}{2d}$, then there exists $C > 0$ which is independent of the feature extractor Φ such that for any function $f \in L^2_a(\mathbb{R}^d)$, we have the deformation sensitivity result given by

$$\Big\|\Phi[D_{\boldsymbol{\tau},\boldsymbol{\omega}}f] - \Phi[f]\Big\| \leq C\|f\|_2\Big(a\|\boldsymbol{\tau}\|_\infty + \|\boldsymbol{\omega}\|_\infty\Big). \tag{3.26}$$

Theorem 3.3.1 also applies to the CNN feature extractor, showing a similar deformation sensitivity bound for CNN with general nonlinear modulus operator and pooling operator which are both Lipschitz-continuous [**82**, **132**].

Then we explore the translation invariance property of scattering transform in deeper layer [**131**]. The extent of translation invariance can be controlled through pooling under hierarchically subsampling.

THEOREM 3.3.2. *Let $f \in L^2(\mathbb{R}^d)$. With the assumptions above, the feature extractor Φ_m in the m-th layer has the property that*

$$\Phi_m[T_{\boldsymbol{x}} f] = T_{\frac{\boldsymbol{x}}{r_1 \cdots r_m}}\Big[\Phi_m[f]\Big]. \tag{3.27}$$

Also, if there exists a global bound K such that the atom φ_m is bounded in the Fourier domain and satisfy the decay condition

$$\|\widehat{\varphi_m}(\boldsymbol{\xi})\|_2 \; \|\boldsymbol{\xi}\|_2 \leq K, \text{ a.e. } \boldsymbol{\xi} \in \mathbb{R}^d \text{ for all } m \in \mathbb{N}, \tag{3.28}$$

then for any $\boldsymbol{x} \in \mathbb{R}^d$,

$$\Big\|\Phi_m[T_{\boldsymbol{x}} f] - \Phi_m[f]\Big\| \leq \frac{2\pi K \|\boldsymbol{x}\|_2}{r_1 \cdots r_m} \|f\|_2. \tag{3.29}$$

The above theorem is also applicable to the study of CNN because the nonlinear operators used mostly in deep learning are pointwise, hence satisfying the property (3.19) to commute with the translation operator. In addition, the pooling operators used mostly in deep learning fulfill the condition (3.19). The details are included in Reference [**132**].

Alternatively, based on Reference [**102**, Chapter 7], the decay condition (3.28) in the above theorem can be replaced by

$$\sup_m \Big\{ \|\varphi_m\|_1 + \|\nabla \varphi_m\|_1 \Big\} < \infty. \tag{3.30}$$

Moreover, if $\lim\limits_{m\to\infty} r_1 \cdots r_m = \infty$, we can reach translation invariance asymptotically by the Inequality (3.29). That is, for any $f \in L^2(\mathbb{R}^d)$ and $\boldsymbol{x} \in \mathbb{R}^d$,

$$\lim_{m\to\infty} \Big\|\Phi_m[T_{\boldsymbol{x}} f] - \Phi_m[f]\Big\| = 0. \tag{3.31}$$

The result [**82**, Theorem 2.10] that is unique to scattering transform is the asymptotic translation invariance with respect to the scale (or resolution) parameter under the same condition in Theorem 3.3.2, without depending on the depth of the network. i.e.,

$$\lim_{J\to\infty} \Big\|\Phi_m^{(J)}[T_{\boldsymbol{x}} f] - \Phi_m^{(J)}[f]\Big\| = 0. \tag{3.32}$$

3.4. Machine Learning Models for Scattering Coefficient Processing

The machine learning tasks can be performed after collecting extracted features from the scattering transform. These extracted features, or the layer-specific features, are fed into a machine learning model as inputs, as shown in Figure 3.9. Often, we compress the feature vectors before feeding into a machine learning model.

3.4.1. Dimension reduction. The dimension of scattering coefficients is sometimes too high to feed into a machine learning model. Another problem of high-dimensional input is the *curse of dimensionality*, meaning that when the feature dimension grows, we need to exponentially increase the size of the training data to maintain a comparable density of the training data. In particular, Richard Bellman mentioned the problem of the exponential growth in volume due to the increment of dimension to Euclidean space [**10**]. A lower density of our training data in the feature dimension can result in overfitting, as the model cannot generalize well to the test data from the limited training data. When getting sufficient data is not always possible, we can conduct *dimension reduction* which tries to retain as much important information as possible while reducing the feature dimension.

Principal component analysis (PCA) [**96**], which linearly transforms variables with a high feature dimension into a smaller set of new variables by retaining as much variance present in the extracted features as possible, is a common method for dimension reduction. The new set of variables in the lower dimensional space after PCA is called *principal components.*

To be more specific, we have n data points $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n \in \mathbb{R}^p$ which are then projected onto a k-dimensional affine subspace for the best approximation, where $k \ll p$. That is, suppose that $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k\}$ is an orthonormal basis for the k-dimensional affine subspace. Define the $V := [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_k]$ to represent the subspace. Denote the sample mean

$$\boldsymbol{\mu} := \frac{1}{n} \sum_{k=1}^{n} \boldsymbol{x}_k.$$

Then we solve the least square problem for PCA

$$\min_{V^{\mathsf{T}}V=I} \sum_{j=1}^{n} \|(\boldsymbol{x}_j - \boldsymbol{\mu}) - VV^{\mathsf{T}}(\boldsymbol{x}_j - \boldsymbol{\mu})\|_2^2,$$

or equivalently,

$$\max_{V^\intercal V=I} \sum_{j=1}^{n} (\boldsymbol{x}_j - \boldsymbol{\mu})^\intercal V V^\intercal (\boldsymbol{x}_j - \boldsymbol{\mu}) \Leftrightarrow \max_{V^\intercal V=I} \operatorname{Tr}[V^\intercal \Sigma V],$$

where the sample covariance matrix

$$\Sigma := \frac{1}{n-1} \sum_{j=1}^{n} (\boldsymbol{x}_j - \boldsymbol{\mu})(\boldsymbol{x}_j - \boldsymbol{\mu})^\intercal.$$

Hence, we can find the principal components by the eigenvectors corresponding to the k largest eigenvalues of the matrix Σ. The number of principal components can be predetermined by a threshold accounting for the precent of variance explained by the principal components. In particular, Bruna and Mallat [**18**] demonstrated that PCA enhances classification performance by compressing the feature extractor from the scattering transform.

3.4.2. Support vector machine (SVM). The goal of *support vector machine* (SVM) [**130**] is to separate the training data linearly by a hyperplane. The margin is defined as the minimum distance between the decision boundary of SVM to any training data point. The nearest data point to the hyperplane is known as the support vector. For example, the SVM classifier transforms the compressed features from STN further to new representations in which the different classes are separated with margins that are as wide as possible.

To be more specific, suppose that we have the weight $\boldsymbol{w}$, training data $\boldsymbol{x}$ and label $\boldsymbol{y}$. We mimimize the following expression

$$\left[\frac{1}{n} \sum_{i=1}^{n} \max\Big(0, 1 - \boldsymbol{y}_i(\boldsymbol{w}^\intercal \boldsymbol{x}_i - b)\Big)\right] + \lambda \|\boldsymbol{w}\|^2. \tag{3.33}$$

By solving the Lagrangian dual, we obtain the expression

$$\max_{\boldsymbol{\alpha}} \left[\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \boldsymbol{y}_i y_j {\boldsymbol{x}_i}^\intercal \boldsymbol{x}_j\right] \tag{3.34}$$

such that $1 \leq \alpha_i \leq C$ for any $i = 1, 2, \cdots, n$ and $\sum_{i=1}^{n} \alpha_i \boldsymbol{y}_i = 0$.

The expression above can be rewritten with a kernel function,

$$\max_{\boldsymbol{\alpha}} \left[\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \boldsymbol{y}_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)\right], \tag{3.35}$$

with some kernel function such as

(1) Polynomial kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (a + \boldsymbol{x}_i^{\intercal}\boldsymbol{x}_j)^l$ for some degree l.

(2) Gaussian kernel: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$.

In general, the model complexity of SVM is higher than some simple methods like logistic regression. However, the SVM does not work well on a huge dataset since it is computationally challenging to find a hyperplane to separate data points with different target classes that are likely overlapping. In addition, lack of interpretability is a drawback of using SVM.

3.4.3. Lasso and Elastic-Net Regularized Generalized Linear Models (GLMNet). The *Lasso and Elastic-Net Regularized Generalized Linear Models* (GLMNet) fit a generalized linear model with Lasso regularization through penalized maximum likelihood, and the GLMNet coefficients are denoted by $\boldsymbol{\theta}$, which are *sparse* [**51**, Chap. 3].

These models try to solve the following problem:

$$\min_{\theta_0, \boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N} w_i l(\boldsymbol{y}_i, \theta_0 + \boldsymbol{\theta}^{\intercal}\boldsymbol{x}_i) + \lambda\left[\frac{1-\alpha}{2}\|\boldsymbol{\theta}\|_2^2 + \alpha\|\boldsymbol{\theta}\|_1\right],$$

$l(\boldsymbol{y}_i, \boldsymbol{x}_i)$ is the negative log-likelihood for observation i. For instance, if we assume $\boldsymbol{y}_i$ follows a Gaussian distribution, then we can set

$$l(\boldsymbol{y}_i, \boldsymbol{x}_i) = \frac{1}{2}(\boldsymbol{y}_i - \theta_0 + \boldsymbol{\theta}^{\intercal}\boldsymbol{x}_i)^2.$$

The observation y_i is assumed to follow a multinomial distribution for multi-label classification problems. In addition, the regularization path corresponding to the Lasso penalty was computed using cyclic coordinate descent. Here, α is the elastic net penalty and λ controls the strength of the penalty. In particular,

(1) $\alpha = 1$: Lasso regression problem.

(2) $\alpha = 0$: Ridge regression problem.

For the sake of interpretation, the importance of the STN coefficients can be captured by the nonzero entries of $\boldsymbol{\theta}$.

CHAPTER 4

Techniques in Scattering Transform with Analytic Wavelets

We present the techniques in scattering transform with analytic wavelets in this chapter. In particular, the content of the work on the STN with the generalized Morse wavelets will be extracted from the Reference [**22**] written by the dissertation author and Professor Saito. See Chapter 2 for the background of AWT and Chapter 3 for the background of STN.

4.1. Numerical Implementation of scattering transform network

In this subsection, we discuss the implementation of the generalized 1-D scattering transform in the package `ScatteringTransform.jl` [**128**] in the Julia programming language [**11**]. The `ContinuousWavelet.jl` [**129**] covers the 1-D analytic wavelet transforms such as the commonly-used Morlet wavelet transform. The new incorporation of the generalized Morse wavelet into the 1-D scattering transform will be presented in Subsection 4.2.

4.1.1. Wavelet transform in the scattering transform network. As mentioned in Subsection 2.1, wavelets can decompose an input signal into different frequency components. Then each component can be investigated and analyzed with different scales or resolutions in the scattering transform. In particular, the Morlet wavelet was defined in Equation (2.21) in Section 2.3. In our scattering transform implementation, the daughter Morlet wavelet before normalization is given by

$$\widehat{\psi}_0^{(s,W)}(\xi) = c_\xi \pi^{-1/4}\Big[\exp\Big(-\Big|\frac{\frac{\xi}{s}-\sigma}{W}\Big|^2\Big) - K_\sigma \exp\Big(-\frac{1}{2}\Big|\frac{\xi}{s}\Big|^2\Big)\Big], \tag{4.1}$$

where s is the scale, W is the width for variance adjustment, c_ξ is the normalizing constant and K_σ is defined by the admissibility criterion. The effective computation of the values of the scale s and the width for variance adjustment W will be discussed in Subsection 4.1.2.

The continuous wavelet transform (CWT) was introduced in Equation (2.16). In particular, the Morlet wavelet is often used in the actual implementation. In the discrete case, we need to sample the scale parameter s indexed by $k \in \mathbb{N}$. One common way to sample s is uniform sampling by $s_k = x^k$ for $x > 0$ [**60**, Chapter 6]. In particular, one can set $x = 2^{1/Q}$, where Q is the *quality factor* as further explained below. The quality factor Q can be different across different layers in the scattering transform, as presented in Equation (3.9). It allows the adaptive adjustment of the scaling of the wavelets in the discrete setting, as opposed to the conventional convolutional neural network (CNN) with fixed filter size in each layer. There are several parameters in the implementation of the wavelet transform:

(1) **Quality Factor** (Q)

The quality factor (or scaling factor) Q is the number of wavelets between the octaves. For example, the quality factor of 8 is suitable for audio and music signals. Without any extra adjustment on the scale, Q is exactly the number of wavelets in each octave.

(2) **Decreasing Factor** (p)

Since many wavelets with finer scales can sometimes be unnecessary created in the discrete setting, the number of wavelets can be decreased per octave by the decreasing factor p.

(3) **Average Length** (l)

The average length l is the number of octaves that are covered by averaging.

(4) **Averaging Type**

The parameter determines whether the averaging filter is a "Dirac" or a "Father" wavelet. We set the default averaging type as "Father".

(5) **Boundary Condition**

Our boundary condition in the wavelet transform is set to be periodic. That is, we add a flipped version of the signal in the end.

(6) **Normalization factor** (α)

If we set the α to be a positive constant, we normalize the wavelet $\boldsymbol{\psi}_0$ by

$$\boldsymbol{\psi} = \frac{\boldsymbol{\psi}_0}{s^{1/\alpha}}.$$

Otherwise, if we set $\alpha = \infty$, then the normalized wavelet becomes

$$\boldsymbol{\psi} = \frac{\boldsymbol{\psi}_0}{\|\boldsymbol{\psi}_0\|_\infty}.$$

4.1.2. Quality factor computation. If the number of wavelets Q per octave is fixed in each index of wavelet, the wavelets densely and unnecessarily cover low frequencies in many applications such as image and signal processing. Hence, some approach to reduce such number of low frequency wavelets has to be applied. Let Q, l, p be the quality factor, average length and decreasing factor respectively. Instead of the the log-linear distribution of the scale $s_k = 2^{k/Q}$, we sample the scale

$$s_k = 2^{bk^{1/p}}, \tag{4.2}$$

where b can be determined in the following procedure.

Let x be an index of wavelet and y be the logarithm of the scale parameter, Their relationship is given by

$$y(x) = bx^{1/p}.$$

In particular, if $p = 1$, then Q is exactly the number of wavelets in each octave because of the linearity between the index of wavelet and the logarithm of the scale parameter. The wavelets skew more to cover high frequencies when p is increased. There are several additional conditions:

(1) We want the first wavelet to be scaled by 2^l. Hence, the first logarithm of the scale parameter y_1 is given by $y(x_1) = l$. Then the corresponding index of wavelet x_1 is $\left(\frac{l}{b}\right)^p$.

(2) Let O be the number of octaves. Suppose there are m indices of wavelet. Note that for the last wavelet, the the logarithm of the scale parameter is $y(x_m) = O + l$.

(3) To find the slope b, we assume that at the last index,

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{b}{p}x^{\frac{1-p}{p}} = \frac{1}{Q},$$

in order that the "instantaneous" number of wavelets is Q in the last octave.

Therefore, we have a system of equations

$$\begin{cases} O + l & = bx_m^{\frac{1}{p}} \\ b & = \frac{p}{Q}x_m^{\frac{p-1}{p}}. \end{cases} \tag{4.3}$$

By substituting b into the first equation, we find the last index

$$x_m = Q \times \frac{O+l}{p}.$$

When x_m is substituted into the second equation, one has

$$b = \left(\frac{p}{Q}\right)^{1/p} (O+l)^{\frac{p-1}{p}}.$$

On the other hand, for the start of the last octave, we have $y(x_{m-1}) = O + l - 1$. Therefore, the second last index

$$x_{m-1} = \left(\frac{y(x_{m-1})}{b}\right)^p = \left(\frac{O+l-1}{b}\right)^p.$$

As we would like to make sure that there are Q wavelets in the last octave, we define the step size Δx as

$$\Delta x = \frac{x_m - x_{m-1}}{Q}.$$

Since we intend to evenly partition the graph, the number of wavelet will be

$$m = \text{round}\left(\frac{x_m - x_1}{\Delta x}\right) + 1.$$

and the scale s are sampled from $2^{y_1}, 2^{y_2}, \cdots, 2^{y_m}$.

Let u and w be defined by

$$u := \frac{1}{100 + \mathrm{e}^{-O}},$$
$$w := \frac{p^u - 1}{(m-1)^p}.$$

Then the corresponding width for variance adjustment W_k of the k-th wavelet is

$$W_k = 1 + w(m-k)^p, \tag{4.4}$$

where $k = 1, 2, \cdots, m$. Note that W is a polynomial of order p from 1 at the highest frequency to p^u at the lowest frequency. The variable u ensures that W is small when there are fewer octaves.

Figure 4.1 shows an example of the relationship between the index of wavelet x and the logarithm of the scale parameter y. If $O = 7,\ l = 4,\ p = 4,\ Q = 8$, there are $m = 24$ wavelets. The first logarithm of the scale parameter $y(x_1) = l = 4$. Each of these $O = 7$ octaves is separated by the red lines.

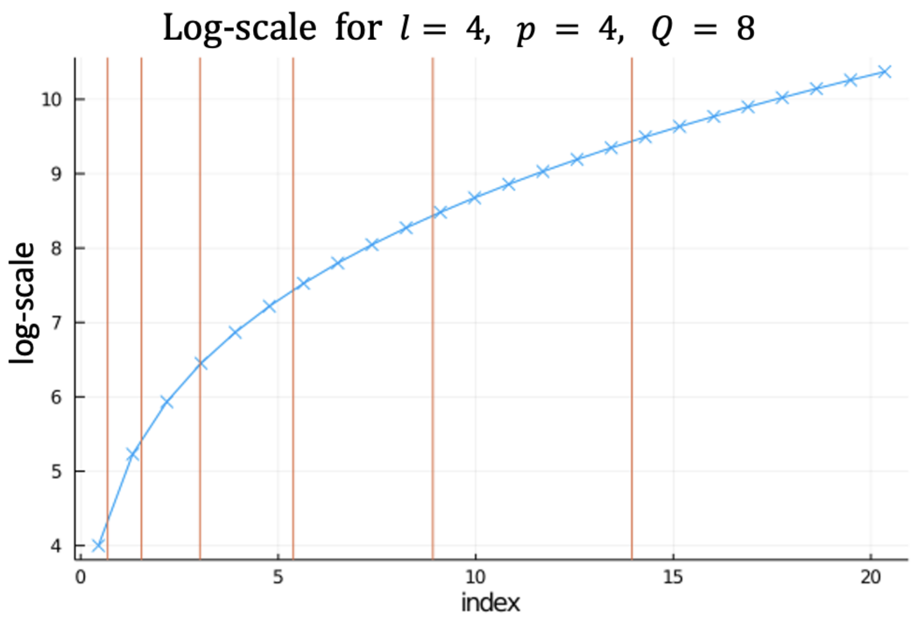


FIGURE 4.1. The logarithm of the scale parameter y for average length l, decreasing factor p and quality factor Q with respect to the index of wavelet x.

4.1.3. Discrete scattering output in the scattering transform. Besides the wavelet transform, pooling and modulus operators are necessary in the scattering transform. In particular, we use the average pooling with a subsampling rate r_m. Unless otherwise specified, the default subsampling rate $r_m = 3/2$, meaning that we take the summation over the sliding local windows of size 2 over 3 consecutive entries. The continuous theory of scattering transform presented in Subsection 3.3.3 does not include the subsampling r'_m for each layer m after averaging with the father wavelet. The scattering output S_m in Equation (3.16) was subsampled to reduce the redundancy in the low-frequency portion of the father wavelet.

On the other hand, the normalization in the scattering transform network for each layer can be processed through

$$S_m^{(1)}[\boldsymbol{\lambda}]f = \frac{S_m^{(0)}[\boldsymbol{\lambda}]f \cdot N_m}{\|\Phi_m f\|_2}, \tag{4.5}$$

where the subscript $\{0\}$ and $\{1\}$ is before and after normalization respectively, N_m is the number of scattering paths Λ_m in the m-th layer. Each time before we compress the scattering output by dimension reduction method such as PCA, we vectorize the scattering transform output and normalize it to have mean zero and unit variance.

4.2. scattering transform network with Generalized Morse Wavelets

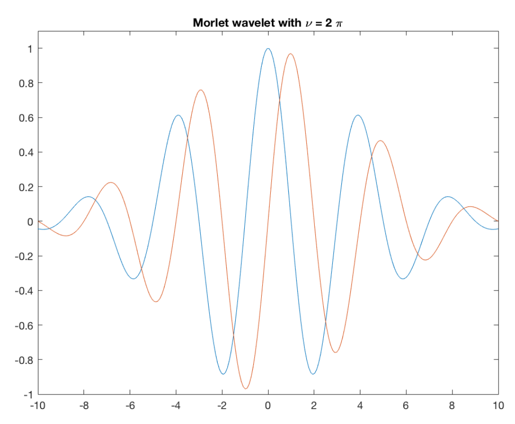
The Generalized Morse Wavelets (GMWs) have been shown to be a superfamily of exactly analytic wavelets in Subsection 2.3. On the other hand, the Morlet wavelet was widely employed in Equation (3.12) for the STN when processing the 1-D signals. However, Lilly *et al.* [**78**] numerically demonstrated that even with a small leakage to negative frequencies by the Morlet wavelet, the wavelet transform could lead to abnormal transform phase variation. The possible leakage to negative frequencies under the Morlet wavelet transform is due to the fact that Morlet wavelet is not exactly analytic. Therefore, we attempt to fill the gap by incorporating the GMWs in the context of scattering transform. The abbreviation GMW-STN refers to *the scattering network incorporating the GMWs* in this subsection.

The GMWs were originally numerically implemented in JLAB, which is a MATLAB® package [**77**]. We further implemented the STN incorporated with both Morlet wavelets and GMWs in Julia [**11**]. Figure 4.2 visualizes the Morlet wavelet with $\nu = 2\pi$ in Equation (2.21) on the left. It also displays the GMW in the time domain on the right, which consists of the real component represented by the blue colour and real component represented by the red color, with two main parameters $(\beta, \gamma) = (4, 2)$ in Equation (2.23).

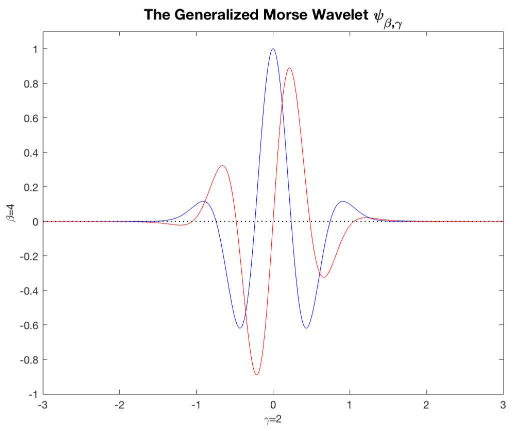
4.2.1. Applications in Music Genre Classification.

Music genre classification is a fundamental research problem in music information retrieval [**100, 124**]. In particular, it is challenging to categorize music into different genres such as classical, country, jazz, hiphop, pop and so on. Part of the reasons is that music genre classification plainly by human judgement can be ineffective and subjective. We would like to distinguish the music information automatically.

The DNN have been shown to be effective to extract hierarchical features for a variety of applications with gigantic training datasets [**72**], including music categorization using the 2D *convolutional recurrent neural network* (CRNN) proposed by Choi *et al.* [**24**]. The local deep feature



(a) Morlet wavelet with $\nu = 2\pi$



(b) GMW with $(\beta, \gamma) = (4, 2)$

FIGURE 4.2. The Morlet wavelet and the GMW used in our experiments

extracted by the CNN and the summarization of the extracted features by the recurrent neural network are shown to achieve effective music classification. Besides, Allamy and Koerich [**3**] also mentioned the effectiveness of 2D CNN [**25**] and suggested the use of 1D residual CNN in the GTZAN dataset [**122**] to automatically classify music genres. While there is a concern about the lack of sufficient data for training a deep network, data augmentation may be able to generate enough data and improve the performance [**3**]. Similarly, we have already described a data augmentation algorithm to synthesize turbulence on images in Subsection 3.2; see also [**20**]. However, the method depends heavily on the quality of the dataset before data augmentation, and it is not clear whether the augmentation technique is appropriate for the GTZAN dataset.

Moreover, data augmentation does not help us understand how to interpret the result from the deep network. It is of interest to know how the music features are interpreted, so that we can explain the high classification rate with a better understanding of the characteristics of different genres. On the other hand, the STN can generate interpretable coefficients unlike the conventional CNN. See Section 3.3 for more details.

Our contributions in the work [**22**] are:

(1) the novel incorporation of the GMW in the STN (GMW-STN) for the analysis of 1-D nonstationary signals such as the music signals;

(2) the demonstration of the superior performance of the GMW in the STN than the Morlet wavelet for classifying music genres, which the result can be explained by the analyticity of the wavelet; and

(3) he demonstration of the interpretability of the STN coefficients computed from the music signals, with which CNNs/DNNs have difficulty.

We now describe the dataset we used for music classification. Below, we will also illustrate how to interpret the hierarchical coefficients from STN, so that we understand how music information can be retrieved for each layer. In this subsection, the doublet of parameters (β, γ) of the GMW in our implementation is set to be $(4, 2)$ as visualized in Figure 4.2 considering the balance of the time-frequency decays.

4.2.2. Data Preparation. In our numerical experiment, we use the GTZAN dataset [**122**] to assess the performance of music genre classification using the STN with the GMW as well as Morlet wavelet. The GTZAN dataset consists of thousands of 30-second 22050Hz audio music tracks. These tracks are distributed evenly into 10 music genres: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, and rock. For each music genre, there are 100 tracks recorded under different conditions.

We split each 30-second track into 15 overlapping music segments, each of which is 5 second long. Let k denote the index of the music segment. The time interval of the k-th music segment is

$$[kL/3 + 1,\ kL/3 + L]$$

for $k = 0 : 14$, where $L = 22050 \cdot 5$ samples. That is, the hop size is $L/3$, i.e., the two adjacent segments have $2/3 \cdot 5 \approx 3.33$ second overlap.

4.2.3. STN Parameter setting. In our STN with 3 layers, we set the inter-layer subsampling rate to $r_m = 8$ and the averaging subsampling rate to $r'_m = 32$ in Equations (3.12) and (3.16). At the nonlinear stage, the modulus operator $M_m(\cdot) = |\cdot|$ was used. We generated the following number of the STN coefficients for each input signal of length 110250: the 0th layer: 3445; the 1st layer: $(431, 33)$; the 2nd layer: $(54, 14, 33)$; and the 3rd layer: $(7, 10, 14, 33)$. Table 4.1 summarizes the STN dimension, which is the same for both the STN incorporating the GMW and the Morlet wavelet.

We also set the quality factors as $(Q_1, Q_2, Q_3) = (8, 4, 4)$, and the maximum scale indices as $(J_1, J_2, J_3) = (32, 13, 9)$. These numbers mean that we used

(1) 33 scales with $j_1 = 0, 1, \cdots, 32$ and the quality factor $Q_1 = 8$ in the 1st layer,

(2) 14 scales with $j_2 = 0, 1, \cdots, 13$ and the quality factor $Q_2 = 4$ in the 2nd layer,

(3) 10 scales with $j_3 = 0, 1, \cdots, 9$ and the quality factor $Q_3 = 4$ in the 3rd layer.

The first numbers in Table 4.1, 3445, 431, 54, and 7, are the size of the output STN coefficients in each path in the respective layers.

TABLE 4.1. Output dimension of STN for each music track.

Layer index	**Size / Length**
Input	110250
0	3445
1	(431, 33)
2	(54, 14, 33)
3	(7, 10, 14, 33)

4.2.4. Interpreting Music Features in different layers. In the 1st layer $m = 1$, we obtained a spectrogram-like output from the GMW-STN as shown in Figure 4.3 for a jazz track. The horizontal and vertical axes for each of these 15 overlapping segments indicate the timestamp $0, 1, \cdots, 430$ (in sample numbers) and the scale $j_1 = 0, 1, \cdots, 32$ indices, respectively. Most of the energy of the signal resides in low to medium frequency bands. Hence, the 1st layer STN reveals spectrogram-like time-frequency information of the original signal that is interpretable and can be used for genre classification. In the context of deep learning, spectrogram extracted from music audio signals can be fed into the 2-D CNN, and the performance on the categorization of music genres is much better than simply feeding the music tracks into a 1-D CNN [**25**]. The result can be explained partly by the additional structured frequency content extracted in spectrogram.

The paths from the 2nd layer have two different frequency measurements, which are indexed by the tuple (j_2, j_1) where $j_2 = 0, 1, \cdots, 13$ and $j_1 = 0, 1, \cdots, 32$. Figure 4.4 displays the 2nd layer output from the GMW-STN, with a small number of timestamps $0, 1, \cdots, 53$ in each segment due

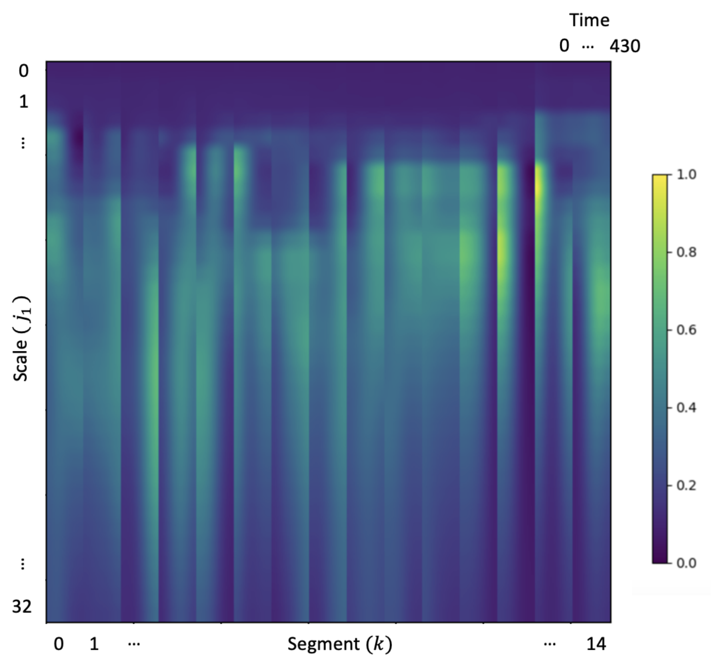


FIGURE 4.3. The normalized 1st layer output of GMW-STN from a Jazz track.

to subsampling between layers. Each block in this figure represents the 2nd layer output from the GMW-STN associated with the scale of the 2nd layer j_2 and the kth segment of the jazz track, and the row index within each block corresponds to the scale from the 1st layer $j_1 = 0, 1, \cdots, 32$. The features corresponding to the 2nd layer scale $j_2 = 0$ has more variations than those features with a larger value of j_2 that represents a finer scale. In other words, the larger the value of j_2 is, the more stable and less noisy the outputs from the GMW-STN become. The hierarchical features in the deeper layer of the GMW-STN become more invariant (or stable) to the local deformations, compared to the hierarchical features in the 1st layer of the GMW-STN. However, there are still noticeable variations across different overlapping segments in the 2nd layer output from the GMW-STN due to nonstationarity of the jazz music signal.

In the GMW-STN, the 3rd layer paths add a third scale variation, indexed by (j_3, j_2, j_1), where $j_3 = 0, 1, \cdots, 9$. Figure 4.5 displays output from the 3rd layer of the GMW-STN. Each block in this figure indicates the 3rd layer output indexed by the 3rd layer scale j_3 and the kth segment of this jazz track. The row index within each block corresponds to the 2nd layer scale $j_2 = 0, 1, \cdots, 13$, while the columns within each block are arranged by the 1st layer scale $j_1 = 0, 1, \cdots, 32$. For each

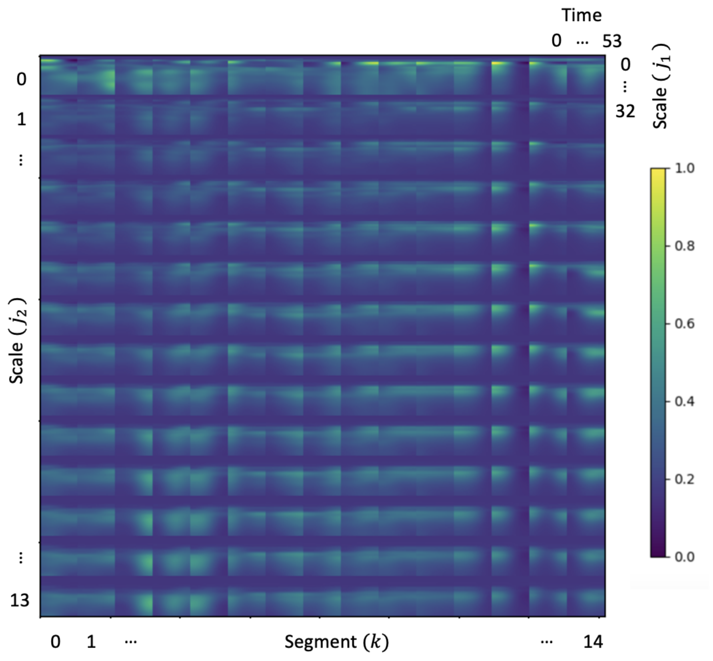


FIGURE 4.4. The normalized 2nd layer output of GMW-STN from a Jazz track.

j_1, there are 7 coefficients after subsampling between the 2nd layer and the 3rd layer. Hence, there are $33 \times 7 = 231$ columns in each block of the figure. The major advantage of using the 3rd layer in the GMW-STN is the increased quasi-translation invariance of the extracted feature. Similar to what we observe in the 2nd layer, we see more stable patterns when the 3rd layer scale j_3 is larger. The main difference from the 2nd layer is that there are more similar patterns across different music segments in the 3rd layer. These suggest that the 3rd layer output from the GMW-STN can capture intrinsic features of the music genre. Thus, we expect the hierarchical features to have more discriminant power to categorize music signals into different music genres than the original music tracks. The interpretation of the higher extent of translation invariance in the deeper layer is connected to Theorem 3.3.2.

4.2.5. Results for Music Classification. This subsection outlines the results of our numerical experiment for classification of music genres on the GTZAN dataset [**122**]. We compare the classification performance between the *STN using Morlet wavelet* (Morlet-STN) and our newly proposed GMW-STN. Moreover, we will explain the numerical results based on the information retrieved from each layer of STNs.

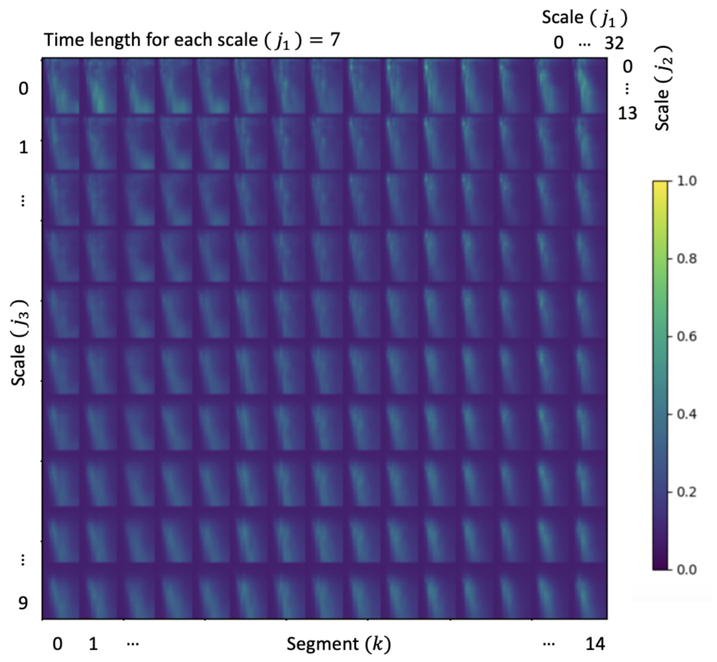


FIGURE 4.5. The normalized 3rd layer output of GMW-STN from a Jazz track.

4.2.5.1. *Training and Testing Stage.* We used the three-fold cross validation scheme in our ten different experiments. In each experiment, the 1000 music tracks were shuffled and distributed into three folds by 340, 330 and 330 samples. We use two folds for training and one fold for testing. Then we iterated the process three times by permutating the folds for each experiment. In other words, we ran the experiments 30 times in total. Each fold of the experiment contains all music genres that are evenly distributed. For instance, there are 34 music files for each music genre in the first fold that consists of 340 files. In each fold, we split each music track further into 15 segments as described in Subsection 4.2.2.

In the training stage, we extracted the STN outputs from all the music segments in the training set. Then these outputs were compressed using the top 1000 principal components using PCA as implemented in [**59**] based on the experimental performance, and are fed into a classifier. Then in the testing stage, each music signal was assigned a label based on the majority vote among the labels of its 15 music segments predicted by the trained classifier.

4.2.5.2. *Classifier.* In our numerical experiments, we mainly use the *Support Vector Machine* (SVM) [**51**, Sec. 3.6] as a classifier for feeding the STN outputs. The SVM classifier transforms the input features further to new representations in which the different classes are separated with margins as wide as possible. The STN coordinates that were compressed under PCA were fed to the SVM classifier of a polynomial kernel of degree 1 implemented in the `LIBSVM.jl` package [**67**] which is built from the C++ library `LIBSVM` [**23**].

Our work contributes further to the interpretability of the information retrieved for the music genre classification. In order to further interpret the classification result, we used GLMNet that we explained in Section 3.4.3 (see also [**51**, Chap. 3]) in our experiments. The GLMNet fits a generalized linear model with Lasso or ElasticNet regularization through penalized maximum likelihood, and the GLMNet weight is parameterized by $\boldsymbol{\theta}$. The regularization path corresponding to the Lasso penalty was computed by cyclic coordinate descent. We capture the significance of the scattering coefficients in distinguishing the music genre by the $\boldsymbol{\theta}$ weight parameter, when the mean loss is minimized by the `GLMNet.jl` package [**66**].

4.2.5.3. *Classification results and evaluation.* We evaluated the music classification performance of various methods (Morlet-STN vs GMW-STN with different classifiers) by comparing the predicted labels and the ground truth of the music tracks. We computed the classification accuracy by first computing the mean accuracy under one experiment of the three-fold cross validation, and then computing the average of the mean accuracies of these ten repeated experiments.

Table 4.2 indicates the superior performance of the GMW-STN with the SVM classifier compared to the Morlet-STN with the SVM classifier. The novel incorporation of the GMW into the STN improved the test accuracy by more than 4% in the third layer. Moreover, this table indicates that as the number of layers of the STN increases, the classification accuracy also increases regardless of the wavelet filters. The increase in accuracy is most significant from the first layer to the second layer.

TABLE 4.2. Average classification accuracy on music genres using GMW-STN and Morlet-STN.

	GMW-STN		Morlet-STN
Layer	GLMNet	SVM	SVM
1	52.3711%	53.0529%	48.8776%
2	70.2504%	73.7329%	70.0517%
3	74.5500%	**77.9088%**	73.7178%

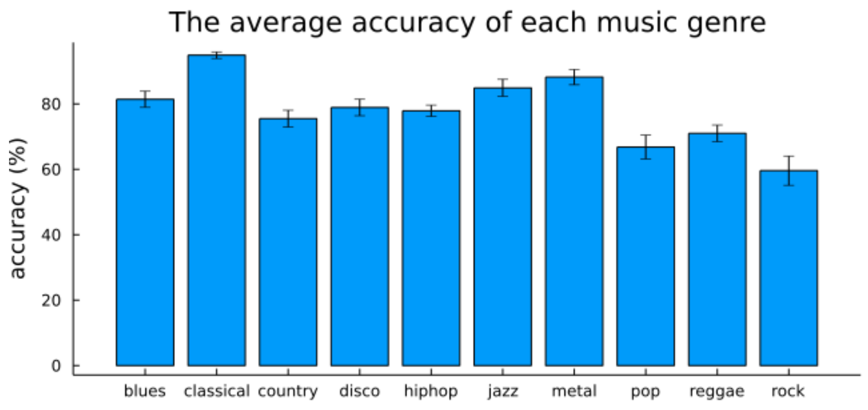


FIGURE 4.6. The test accuracy of all music genres.

In comparison to the SVM classifier, the GLMNet classifier performed slightly worse in tagging music genres by the GMW-STN ($\sim$3%). Nevertheless, we will show in Subsection 4.2.6 that the GLMNet classifier can explain the numerical results, and shed light on the music information extracted by the STN coefficients, which is impossible with the SVM classifier.

We further show the performance of the GMW-STN with the SVM classifier for each individual music genre in Figure 4.6. The GMW-STN with SVM performed best in classifying classical music (94.9%) followed by metal (88.2%), jazz (84.9%), and blues (81.4%). However, it did not perform well in classifying pop (66.8%) and rock music (59.6%). See Subsection 4.2.6 for the explanation on the difference of the accuracies among the music genres.

Our result is comparable to the reported accuracy (76.02%) without data augmentation and (80.93%) with data augmentation using 1D CNN [**3**]. However, we can uniquely provide the explanation of the numerical results based on the additional music information retrieved in deeper layers. These structured information from the music tracks can be visualized and interpreted from the STN outputs. On the other hand, it is quite difficult to explain the results and interpret the intermediate representations in deep learning. Thus, interpretability is a major advantage of our method over CNNs.

4.2.6. Significance of STN Coefficients. As we mentioned in the previous subsection, the SVM classifier cannot provide the explanation of which part of features extracted from the GMW-STN coefficients mainly contribute to the correct music genre classification. Unlike the SVM

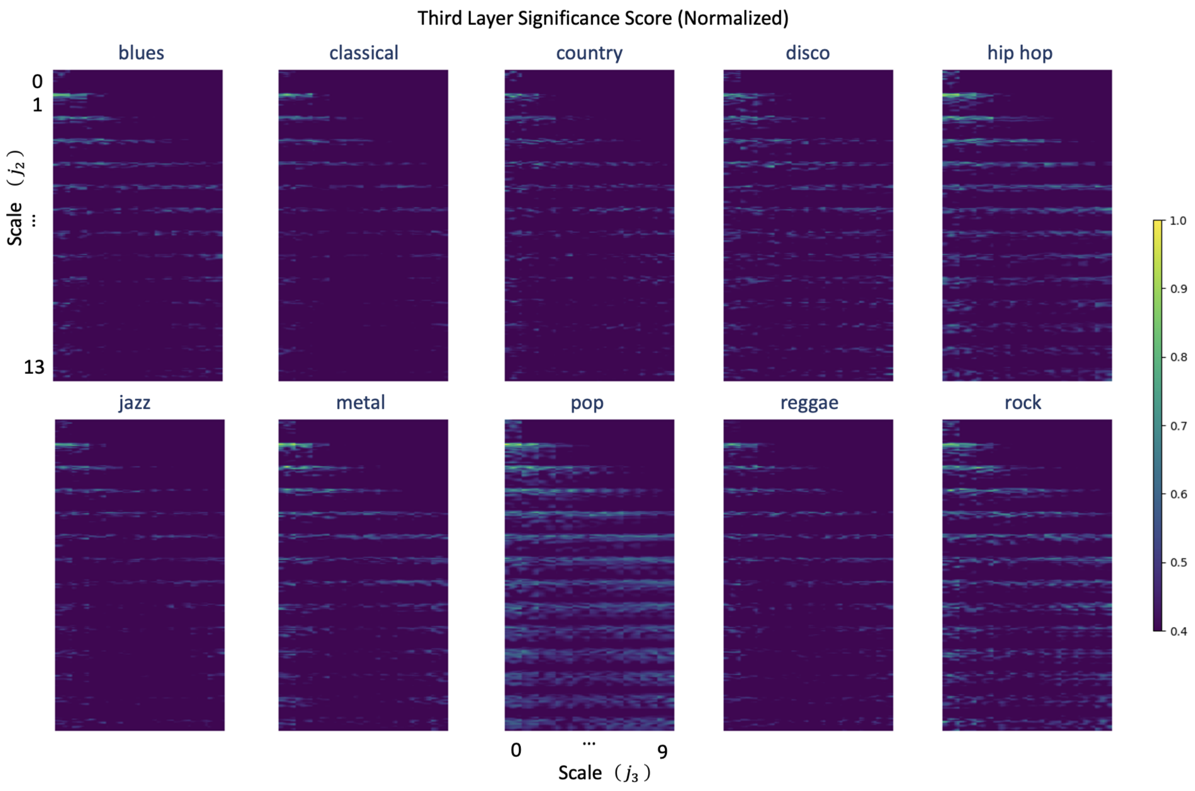


FIGURE 4.7. The normalized significance score from the third layer scattering coefficients for each genre.

classifier, the GLMNet classifier helps us explain how the information retrieved from the STN coefficients relate to the music genre classification.

Since the GLMNet coefficient vector $\boldsymbol{\theta}$ for each genre was computed on the top 1000 PCA components of the third layer GMW-STN coefficients, we inverted the PCA and projected the PCA components back to get a set of coefficients corresponding to the third layer GMW-STN. We then normalized these coefficients so that the maximum value became 1, which we call the *significance scores*, and display them in Figure 5.7, with the lower bound clamped to 0.4. The significance scores of ten music genres from Figure 5.7 were computed to illustrate which music information collected from the STN coefficients contribute to the classification result. The $\boldsymbol{\theta}$ parameter in GLMNet associated to the significance of GMW-STN coefficients can be utilized for the explanation of the music genre classification.

Figure 5.7 shows that information quantified in the third layer GMW-STN is critical in the low frequency portion, especially in the $(j_3, j_2) = (0, 1)$ blocks. In general, the concentration in the lower frequency region is positively associated to the high classification rate. For example, from Figure 5.7, the classical music has the highest scores in the lower frequency portion of the 3rd layer coefficients, while pop and rock have more dispersed score distributions. The dispersion in the score distribution can be attributed to the variations of the patterns in these music genres, which in turn may have contributed to the lower classification accuracies for these music genres.

In short, we demonstrated that the GMW-STN outperformed the conventional Morlet-STN. It can be explained by the importance of analyticity of the underlying wavelet transform. Since the input signals here are digital music signals, which are nonstationary and oscillatory, we should be able to observe the advantages of using the GMWs over the Morlet wavelets. Moreover, the classification accuracy became higher with deeper layers in the STN since we could retrieve additional relevant music information that are stable with respect to local deformations. We could demonstrate the connection between the music information retrieved from the GMW-STN and the classification results in our experiment, which would be impossible when using the CNNs. In addition, it turned out that the lower frequency portion of the retrieved music information from the 3rd layer GMW-STN coefficients attributed mainly to the performance of music genre classification.

4.3. GMW-STN with Different Parameters (β, γ)

After evaluating the performance of the GMW-STN using the parameter pair $(\beta, \gamma) = (4, 2)$, we evaluated the performance with different pairs of (β, γ). Particularly, the "Airy" wavelets corresponding to $\gamma = 3$ is of interest and also is recommended by the Lilly *et al.* [**80**]. This family of the GMWs approximate the Gaussian distribution closely while being exactly analytic [**80**]. See Subsection 2.3.2.1 for more details about different families of the GMWs. Therefore, we decided to conduct more experiments in this subsection on the pairs of parameters $(\beta, \gamma = 3)$ in the GMW-STN. We still used the GTZAN dataset [**122**] for music genre classification.

Our experimental settings in this section are the same as the last section, except the choice of the pairs of parameters (β, γ) in the GMWs. In the same three-fold cross validation scheme, the test accuracy using the GMW-STN with the SVM classifier is still the best when $(\beta, \gamma) = (4, 2)$.

The GMW family with $\gamma = 3$ had the test accuracy of less than 75% in classifying music genres along with $\beta = 3, 9, 27$, as indicated in Table 4.3. It suggests experimentally that the analytic "Derivative of Gaussian" (DoG) wavelets ($\gamma = 2$) is more suitable for analyzing the music signals than the Airy wavelets ($\gamma = 3$).

TABLE 4.3. Average accuracy using the GMW-STN with different pairs of (β, γ) in music genre classification.

(β, γ)	**Test Accuracy**
(4, 2)	**77.9088%**
(3, 2)	74.4058%
(9, 2)	74.6138%
(27, 2)	75.6488%
(3, 3)	74.4052%
(9, 3)	74.7891%
(27, 3)	73.7472%

As concluded by Lilly *et al.* [**80**], the class of wavelets for $\gamma = 2$ is less ideal for the analysis of oscillations than the Airy wavelets with $\gamma = 3$. Perhaps the music signals in general are not as oscillatory as signals that can be analyzed through analytic wavelets with an approximate Gaussian distribution, we can use the parameter $\gamma = 2$ to analyze the music signals. We recall from Subsection 2.3.2.1 that the wavelet $\boldsymbol{\psi}_{\beta,2}$ can be obtained by differentiating $\boldsymbol{\psi}_{0,2}$ with respect to β in the time domain according to Equation (2.28). We see the central portion of the filter increases when β increases and β is not asymptotically large. Therefore, the Morse wavelet with $\beta = 4$ and $\gamma = 2$ (i.e., $\boldsymbol{\psi}_{4,2}$) allows more sufficient analysis of the signal with larger central portion of the filter, than the GMW $\beta = 0$ and $\gamma = 2$ (i.e., $\boldsymbol{\psi}_{0,2}$).

CHAPTER 5

Monogenic Wavelet Scattering Network

This chapter is devoted to present the novel STN, which is called *Monogenic Wavelet Scattering Network* (MWSN), for 2-D signals (e.g., images) by taking the advantage of monogenicity. We extend the 1-D STN discussed in Chapter 4, to 2-D STN by the natural extension of analyticity to monogenicity. See Subsection 2.4 for the mathematical background on monogenicity in 2-D signals. We extract materials extensively from Reference [**21**] written by the dissertation author and Professor Saito in this chapter. The topic was also presented in the CeDAR Research Symposium 2022.

5.1. Architecture for Monogenic Wavelet Scattering Network

The generalized STN was initially implemented by Weber [**128**] in the Julia package. After that, the dissertation author implemented the new MWSN based on the foundation of the STN, with the ability to hierarchically extract "analytic" features in higher dimension among scattering layers. The MWSN architecture as a whole can be described in Figure 5.1. We will then briefly discuss how to incorporate the *monogenic wavelet transform* (MWT) in the STN.

We follow the strategy of Soulard and Carré [**111**] for the MWT implementation as follows. Firstly, we create an $M \times N$ grid of Fourier coordinates. Let

$$X_1 := \frac{1}{M}\Big[-\lfloor M/2 \rfloor, -\lfloor M/2 \rfloor + 1, \cdots, \lceil M/2 \rceil - 1 \Big] \mathbb{1}_{1,N},$$

$$X_2 := \frac{1}{N}\mathbb{1}_{M,1}\Big[-\lfloor N/2 \rfloor, -\lfloor N/2 \rfloor + 1, \cdots, \lceil N/2 \rceil - 1 \Big],$$

where $\mathbb{1}_{1,N}$ is a $1 \times N$ row vector of all ones, and $\mathbb{1}_{M,1}$ is a $M \times 1$ column vector of all ones. We define a variable $\boldsymbol{\eta} \in \mathbb{R}^{M \times N}$ such that

$$\eta[i,j] = 2\pi\sqrt{X_1[i,j]^2 + X_2[i,j]^2}.$$

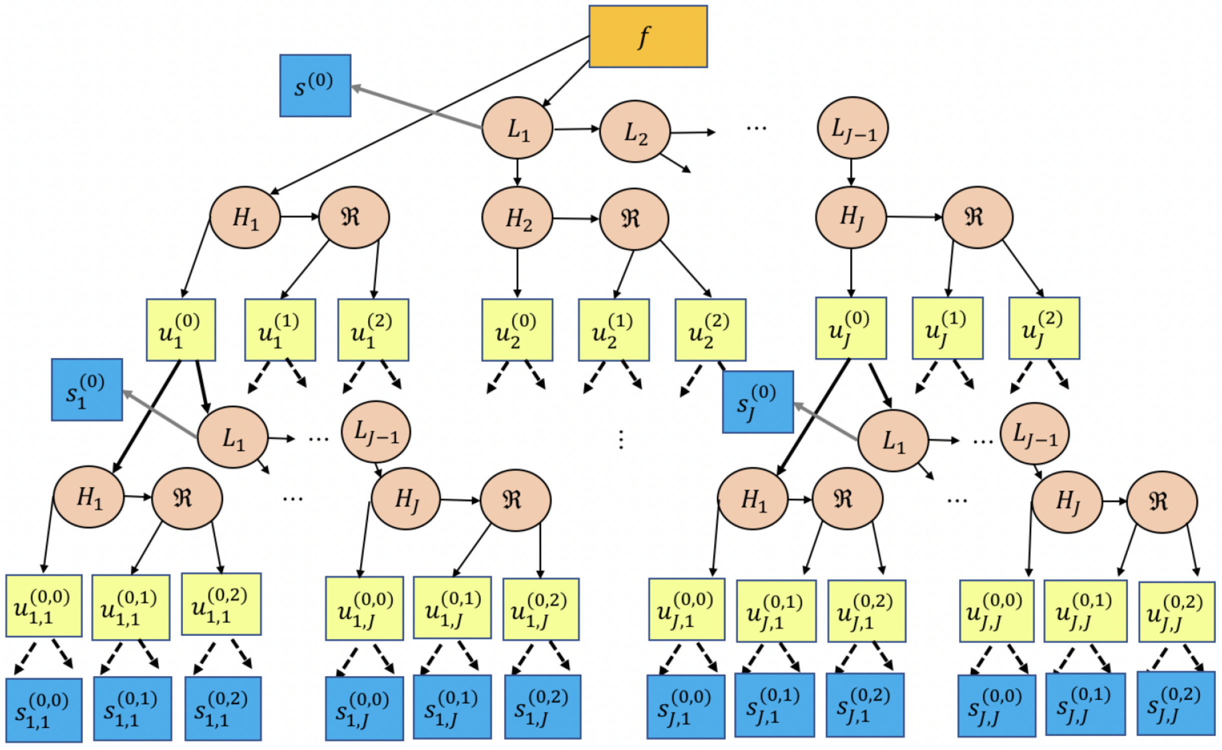

f
$s^{(0)}$
L_1
L_2
L_{J-1}
H_1
H_2
H_J
$\Re$
$u_1^{(0)}$ $u_1^{(1)}$ $u_1^{(2)}$
$u_2^{(0)}$ $u_2^{(1)}$ $u_2^{(2)}$
$u_J^{(0)}$ $u_J^{(1)}$ $u_J^{(2)}$
$s_1^{(0)}$
$s_J^{(0)}$
$u_{1,1}^{(0,0)}$ $u_{1,1}^{(0,1)}$ $u_{1,1}^{(0,2)}$
$u_{1,J}^{(0,0)}$ $u_{1,J}^{(0,1)}$ $u_{1,J}^{(0,2)}$
$u_{J,1}^{(0,0)}$ $u_{J,1}^{(0,1)}$ $u_{J,1}^{(0,2)}$
$u_{J,J}^{(0,0)}$ $u_{J,J}^{(0,1)}$ $u_{J,J}^{(0,2)}$
$s_{1,1}^{(0,0)}$ $s_{1,1}^{(0,1)}$ $s_{1,1}^{(0,2)}$
$s_{1,J}^{(0,0)}$ $s_{1,J}^{(0,1)}$ $s_{1,J}^{(0,2)}$
$s_{J,1}^{(0,0)}$ $s_{J,1}^{(0,1)}$ $s_{J,1}^{(0,2)}$
$s_{J,J}^{(0,0)}$ $s_{J,J}^{(0,1)}$ $s_{J,J}^{(0,2)}$

FIGURE 5.1. The MWSN architecture

Let W be the inverse zero-frequency shift (ifftshift) operator in 2-D. Then we can compute the radial frequency coordinate $\boldsymbol{\xi}$ by

$$\boldsymbol{\xi} = W(\boldsymbol{\eta}).$$

We define a Gaussian high-pass filter H in the 2-D Fourier domain by

$$H(\boldsymbol{\xi}) := 1 - \mathrm{e}^{-\frac{\|\boldsymbol{\xi}\|_2^2}{2}}.$$

Then we define the Gaussian high-pass filter at scale j, denoted by H_j by

$$H_j(\boldsymbol{\xi}) := H(2^{j-1}\boldsymbol{\xi}), \quad j \in \{1, \dots, J\}, \tag{5.1}$$

where we typically use the maximum scale $J = 4$ in practice.

A corresponding low-pass filter L_j at scale j can then be defined as

$$L_j(\boldsymbol{\xi}) := \sqrt{1 - (H_j(\boldsymbol{\xi}))^2}.$$

In the MWSN framework, the MWT filter bank has intensive interaction with the feature from the previous layer. In Figure 5.1, the salmon-pink disks represent operators, i.e., H_j, L_j, and $\mathcal{R}$. Note that the convolution with the father wavelet φ_m at the mth layer in Equation (3.16) in the conventional STN corresponds to the low-pass filtering with L_1 at every layer in the MWSN. The zeroth ($m = 0$) layer output is indicated by the blue box $S_0 f(\boldsymbol{x})$ ($\boldsymbol{s}^{(0)}$ for short) after the low-pass filtering with L_1 of the input image f followed by subsampling. The superscript (0) indicates the isotropic filtering is applied. In the first ($m = 1$) layer, the vectors $U_1[\lambda_1]f(\boldsymbol{x})$, $\lambda_1 \in \Lambda_1$ in the conventional STN of Equation (3.12) is now denoted by the vectors in the yellow boxes, $\boldsymbol{u}_j^{(l)}$, $j \in \{1, \ldots, J\}$, $l \in \{0, 1, 2\}$, where j is the scale parameter, and $l \in \{0, 1, 2\}$ indicates the isotropic component, the vertical and horizontal Riesz components obtained by $\mathcal{R}_1$, $\mathcal{R}_2$, respectively. The output vectors of the first layer, indicated by blue boxes such as $\boldsymbol{s}_1^{(0)}$ and $\boldsymbol{s}_J^{(0)}$, are obtained by subsampling $\boldsymbol{u}_j^{(l)}$, low-pass filtering with L_1, and yet another subsampling. Note that the other first-layer outputs, i.e., $\boldsymbol{s}_j^{(0)}, j \in \{2, \ldots, J-1\}$ and $\boldsymbol{s}_j^{(l)}$, $l \in \{1, 2\}$, $j \in \{1, \ldots, J\}$ are omitted due to the crowded graphics. Now, in the second ($m = 2$) layer, the vector $U[\boldsymbol{\lambda}]f(\boldsymbol{x}) = U[\lambda_2]U[\lambda_1]f(\boldsymbol{x})$ in the conventional STN of Equation (3.14) is denoted by $\boldsymbol{u}_{j_1,j_2}^{(l_1,l_2)}$, $j_k \in \{1, \ldots, J\}$, $l_k \in \{0, 1, 2\}$, $k = 1, 2$, where l_1, j_1 indicate the inherited first layer path information whereas l_2, j_2 are the parameters specified in the second layer. The outputs of the second layer are again obtained by applying the same procedure as the first layer to $\boldsymbol{u}_{j_1,j_2}^{(l_1,l_2)}$, which are indicated by blue boxes $\boldsymbol{s}_{j_1,j_2}^{(l_1,l_2)}$. Finally, the arrows in this diagram show the flow of the data; in addition, the thick arrows indicate that the subsampling operations are performed before reaching the destination disks or boxes while their color (gray or black) suggests that a potentially different subsampling rate can be set.

5.1.1. A Simple illustration of MWSN using MNIST dataset. The features extracted by the MWSN can be visualized and interpreted on MNIST [**73**] dataset. The MNST dataset contains 70000 handwritten digit images, which we split into 60000 images for the training set and 10000 images for the test set in our classification experiments. Each 28×28 image indicates a digit from 0 to 9 with the corresponding digit label. Figures 5.2 (digit 0) and 5.3 (digit 1) show the hierarchical features extracted from the MWSN with the maximum scale $J = 4$ in the first and second layer respectively and without applying the modulus operator. While the subsampling rate r_m between layers is 3/2, there is no subsampling in the averaging stage. That is, $r'_m = 1$.

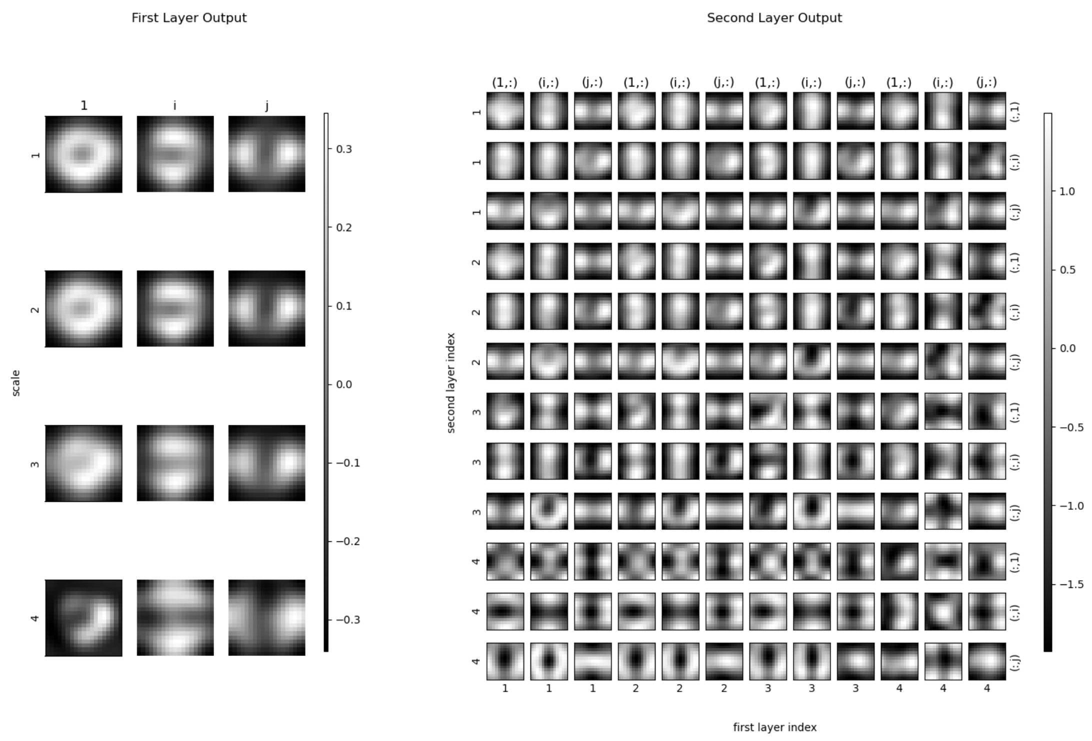


FIGURE 5.2. The MWSN layer1 (left) and layer 2 (right), scale 4 representation of a digit 0 image in the MNIST dataset.

The first column of the left figures in Figures 5.2 and 5.3 corresponds to the isotropic component under the MWT, with increasing scales indicated by the row indices. The second and third columns correspond to the Riesz components in the MWT. In particular, the second column represents the Riesz components in the x-direction. The resulting features in the second column emphasize on the horizontal edges. Meanwhile, the resulting features in the third column, which correspond to the Riesz components in the y-direction, emphasize the vertical edges. Together we know the overall magnitude, phase and orientation based on the quaternion expression as shown in Equation (2.43).

The second layer in the right figures in Figures 5.2 and 5.3 has more contextual details than the first layer. Each row index s in the figure has the corresponding scale $\lceil s/3 \rceil$ and the corresponding contextual emphasis in the $1, \mathbb{i}$ or $\mathbb{j}$ components. The columns within each block are arranged by the 1st layer scale $j = 1, \cdots, 4$ and monogenic components. The implication is that we can look into the resulting features with different directional contexts in a broad perspective from the first layer. We can visualize the contextual details further with different resolutions and different

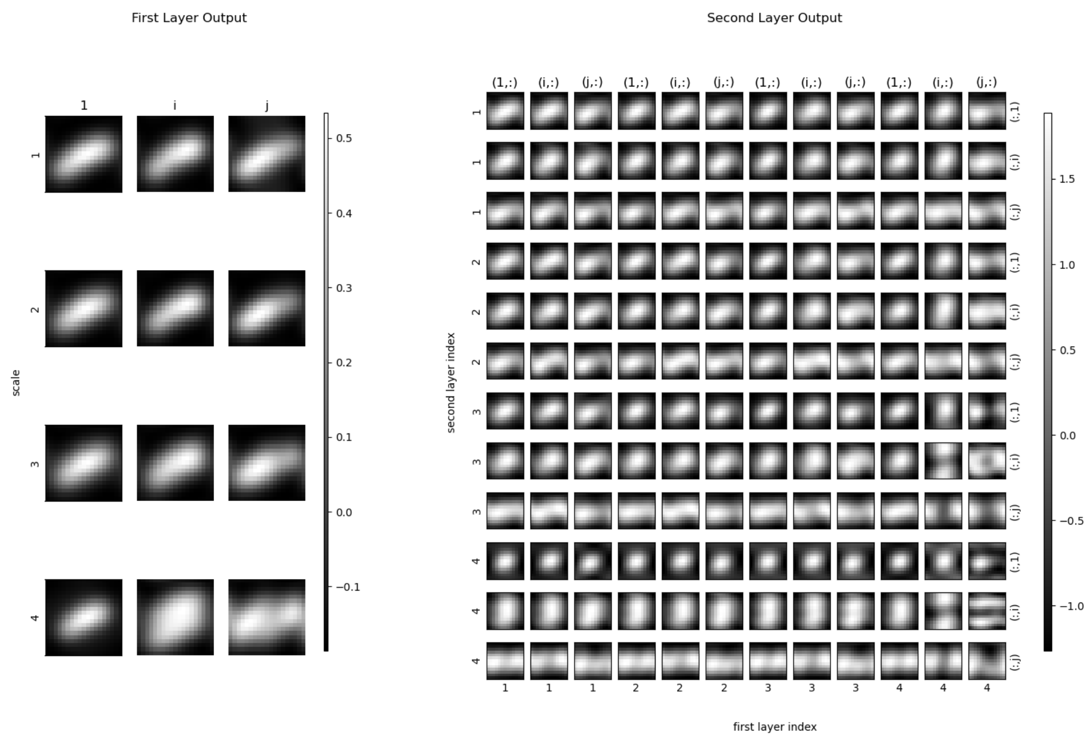


FIGURE 5.3. The MWSN layer1 (left) and layer 2 (right), representation of a digit 1 image in the MNIST dataset (with $J = 4$).

directional contextual emphasis in the second layer. We will show that the additional hierarchical features allow us to better distinguish the digits.

In our experiment on the MNIST dataset, we used different sample sizes to compare the classification performance among the original images, the hierarchical features of the first layer in the MWSN, and the corresponding PCA features with no more than 1000 PCA dimensions. The Julia package for the SVM with the Gaussian kernel [**23**] $\left(\gamma = \dfrac{1}{\text{sample size}}\right)$ was used as the classifier.

To demonstrate that the MWSN does not require a gigantic dataset, we conducted numerical experiments and evaluated the performance on small training sample sizes. Let M be the sample size. For each of the 100 random trials, we randomly selected M training samples and 10000 testing samples. For each training set, we sampled each digit evenly in the MNIST dataset. That is, if $M = 500$, we sampled 50 images for each digit. Then we computed the average training and test accuracies among the 100 random trials. In Figure 5.4, the test accuracy with the PCA hierarchical

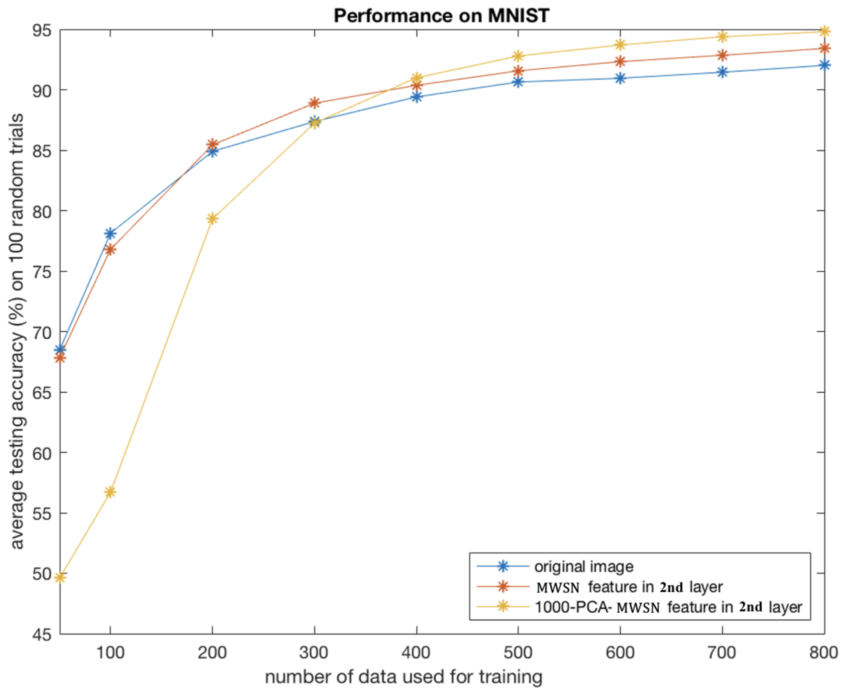


FIGURE 5.4. The performance on the MNIST dataset.

features from MWSN using the SVM was the highest among all the other inputs (i.e., the original images and the 1st-layer MWSN coefficients) if the sample size is greater than 500.

5.2. Rotated Monogenic Wavelet Scattering Network

In the implementation of the MWSN, the dissertation author further incorporated the rotation operators in the MWSN. Unlike the 1-D signals as discussed in Chapter 4, 2-D signals can be analyzed through multiple directions such as the horizontal, vertical and diagonal directions. The additional rotation operator allows us to extract the Riesz directions in the MWSN effectively along directions other than the horizontal and vertical directions.

Let f be a 2-D signal. According to the notation we used in Section 2.4 about monogenicity, we can write the monogenic signal as

$$f^{+}(\boldsymbol{x}) := f(\boldsymbol{x}) + \mathbb{i} f^{(1)}(\boldsymbol{x}) + \mathbb{j} f^{(2)}(\boldsymbol{x}),$$

where $f^{(1)}(\boldsymbol{x})$ and $f^{(2)}(\boldsymbol{x})$ corresponds to the Riesz-x and Riesz-y components respectively.

Suppose that $\mathbf{O}_\phi$ is a rotation operator which rotates points about the origin in the 2-D Euclidean space in the counterclockwise manner through an angle ϕ with respect to the positive x-axis. To be specific,

$$\mathbf{O}_\phi(\boldsymbol{x}) := \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{5.2}$$

We define

$$h_\phi(\boldsymbol{x}) := \mathbf{O}_\phi h(\boldsymbol{x}) = h(\mathbf{O}_{-\phi}\boldsymbol{x})$$

for any given 2-D signal h.

In reality, the 2-D signal f may be rotated to recognize different Riesz directions under orientations. The rotated monogenic and anti-monogenic signals $f_\phi^{\pm}(\boldsymbol{x})$ along the rotation angle ϕ are given by

$$\begin{aligned} f_\phi^{\pm}(\boldsymbol{x}) &= \mathbf{O}_\phi[f(\boldsymbol{x})\pm\mathcal{R}f(\boldsymbol{x})] \\ &= \mathbf{O}_\phi f(\boldsymbol{x})\pm\mathbf{O}_\phi\mathcal{R}f(\boldsymbol{x}) \\ &= \mathbf{O}_\phi f(\boldsymbol{x})\pm\Big(\mathbb{i}\mathbf{O}_\phi f^{(1)}(\boldsymbol{x}) + \mathbb{j}\mathbf{O}_\phi f^{(2)}(\boldsymbol{x})\Big). \end{aligned} \tag{5.3}$$

Note that the rotation operator $\mathbf{O}_\phi$ and the Riesz transform $\mathcal{R}$ do not commute [**19**]. That is,

$$\mathbf{O}_\phi\mathcal{R}f(\boldsymbol{x}) \neq \mathcal{R}\mathbf{O}_\phi f(\boldsymbol{x}) = \mathcal{R}f(\mathbf{O}_{-\phi}\boldsymbol{x}).$$

Therefore, we have the following property:

$$\mathbf{O}_{-\phi}\mathcal{R}\mathbf{O}_\phi \neq \mathcal{R}. \tag{5.4}$$

Based on this property, we define instead

$$
\begin{aligned}
f^{\pm}_{-\phi,\phi}(\boldsymbol{x}) &:= \mathbf{O}_{-\phi}[f(\mathbf{O}_{-\phi}\boldsymbol{x})\pm\mathcal{R}f(\mathbf{O}_{-\phi}\boldsymbol{x})] \\
&= \mathbf{O}_{-\phi}f(\mathbf{O}_{-\phi}\boldsymbol{x})\pm\mathbf{O}_{-\phi}\mathcal{R}f(\mathbf{O}_{-\phi}\boldsymbol{x}) \\
&= \mathbf{O}_{-\phi}\mathbf{O}_{\phi}f(\boldsymbol{x})\pm\mathbf{O}_{-\phi}\Big(\mathbb{i}f^{(1)}(\mathbf{O}_{-\phi}\boldsymbol{x})+\mathbb{j}f^{(2)}(\mathbf{O}_{-\phi}\boldsymbol{x})\Big) \\
&= f(\boldsymbol{x})\pm\Big(\mathbb{i}\mathbf{O}_{-\phi}f^{(1)}(\mathbf{O}_{-\phi}\boldsymbol{x})+\mathbb{j}\mathbf{O}_{-\phi}f^{(2)}(\mathbf{O}_{-\phi}\boldsymbol{x})\Big).
\end{aligned}
\tag{5.5}
$$

Here $\mathbf{O}_{-\phi}f^{(1)}(\mathbf{O}_{-\phi}\boldsymbol{x})$ is the component corresponding to the Riesz direction along the rotation angle ϕ from the x-axis, while $\mathbf{O}_{-\phi}f^{(2)}(\mathbf{O}_{-\phi}\boldsymbol{x})$ is the component corresponding to the Riesz direction along the rotation angle ϕ from the y-axis in the counterclockwise direction. Therefore, we can extract new feature by the new operator $\mathbf{O}_{-\phi}\mathcal{R}\mathbf{O}_{\phi}$.

In the MWSN, we can implement its rotated version inspired by the calculations above. Before passing the signal into the network, we rotate the signal f by the rotation operator $\mathbf{O}_{\phi}$. Then we process the rotated signal in the network. After that, we rotate the resulting features back by $\mathbf{O}_{-\phi}$. The new $\mathbb{i}, \mathbb{j}$ components of the rotated MWSN in the first layer are associated to the $\mathbb{i}, \mathbb{j}$ components in the rotated monogenic representation $f^{+}_{-\phi,\phi}$. The second layer of the rotated MWSN encourages extraction of hierarchical features along the new components. We can concatenate the MWSN features with the rotated MWSN features by the rotation operator $\mathbf{O}_{\phi_1}, \mathbf{O}_{\phi_2}, \cdots, \mathbf{O}_{\phi_l}$, with $2l+2$ Riesz directions. We write RMWSN as the abbreviation of the rotated MWSN.

5.3. Applications in Texture Image Classification

The content of the whole subsection was investigated in Reference [**21**].

5.3.1. Dataset. The texture image dataset we used in this subsection is the Columbia-Utrecht Reflectance and Texture (CUReT) database [**26**]. For each of 61 texture classes in the CUReT dataset, we selected 92 texture images that were then cropped to retain a central region of size 200×200, and converted to grayscale from RGB. Therefore, the total number of texture images available is $5,612$.

5.3.2. Experimental setting. We evaluated the classification performance of our MWSN and compared with that of the standard 2-D STN based on Morlet wavelets [**17, 18**]. The 2-D

Method	Dimension before PCA	Classification Accuracy
MWSN	90,000	**97.34%**
RMWSN	180,000	**98.19%**
Kymatio-STN (L = 2)	11,875	94.50%
Kymatio-STN (L = 4)	38,125	96.18%
Kymatio-STN (L = 6)	79,375	96.56%
Kymatio-STN (L = 8)	135,625	96.61%

Table 5.1. Table of average test accuracy over 10 experiments

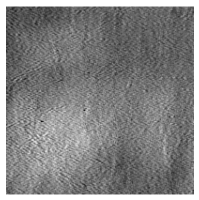
STN was implemented by Andreux *et al.* [**5**] in the the Kymatio package (Kymatio-STN) using the Python programming language. All the other codes for our experiments were conducted on the Julia programming language [**11**]. We also compared the performance of our MWSN and our rotated MWSN (RMWSN) with a rotation operator $\mathbf{O}_{\pi/4}$. In the RMWSN, we concatenated the features and rotated features by $\mathbf{O}_{\pi/4}$ from the original MWSN.

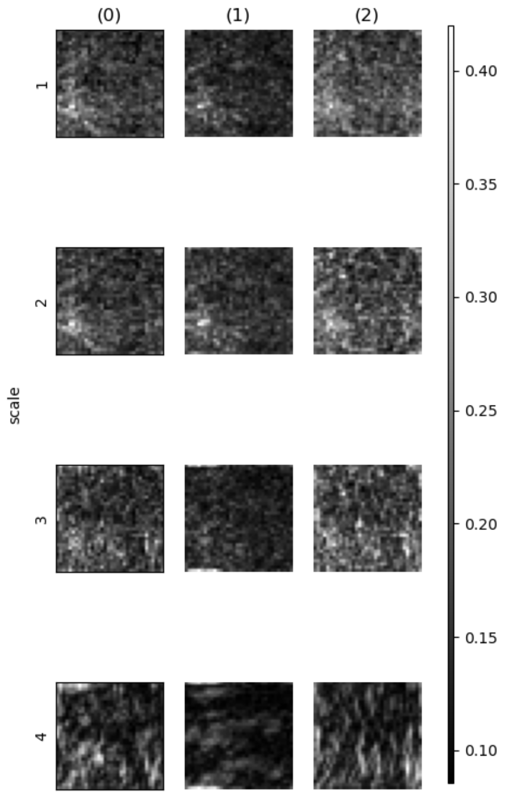
In the MWSN, we set the maximum scale parameter $J = 4$ in Equation (5.1) , which exactly corresponds to $J = 3$ in the Kymatio package. For both methods, we only used the 2nd layer outputs since they contain the most relevant information in the input texture images.

We set the subsampling rates in the MWSN (and RMWSN) to 2 regardless of the layers, while the default values were chosen in the Kymatio-STN. Since the Kymatio-STN allows the users to select the number of orientations of the Morlet wavelets, we tried the number of orientations $L = 2, 4, 6, 8$. In each scenario, we used the PCA method implemented in the `MultivariateStats.jl` package [**59**] to reduce the dimension of the scattering coefficients / feature vectors of the MWSN (and RMWSN) and the Kymatio-STN. After the experiments, we decided to use the top 30 PCA coordinates in all cases.

Next, we fed those coordinates to the SVM classifier with a polynomial kernel of degree 1 implemented in the `LIBSVM.jl` package [**67**] which was build from the C++ library `LIBSVM` [**23**]. We then compared the predicted classes of the texture images by the SVM with the ground truth.

5.3.3. Experimental Results. Table 5.1 lists the average test accuracy by repeating two-fold cross validation 10 times in each case together with the coefficient / feature vector before the PCA was applied. The accuracy achieved by our proposed MWSN was better than the Kymatio-STN even with the higher number of orientations due to the natural extension of analyticity in 1-D to


(a) An input image



(b) The 1st-layer output

FIGURE 5.5. A CUReT image and its MWSN first layer output

monogenicity in 2-D. Because of the "leak" of the energy to the negative frequency range, the CWT with Morlet wavelets used in the Kymatio-STN retains less properties than the MWT. Together with the fact that the Riesz kernels are effective 2-D edge detectors, fewer contextual directions in the MWSN could still extract sufficient textural information and achieved better classification result than the Kymatio-STN. In addition, the best accuracy could be achieved by the RMWSN, showing that our proposed network performed even better with more number of orientations.

The output coefficients have a total size of $(25, 25, 12, 12)$ for each image, of our proposed MWSN were also "interpretable". Interpretability was rarely considered in the earlier studies on texture classification. Figure 5.5b and Figure 5.6 exhibit the additional orientation information of an image shown in Figure 5.5a captured by the Riesz transforms, $\mathcal{R}_1$ and $\mathcal{R}_2$ with different scales

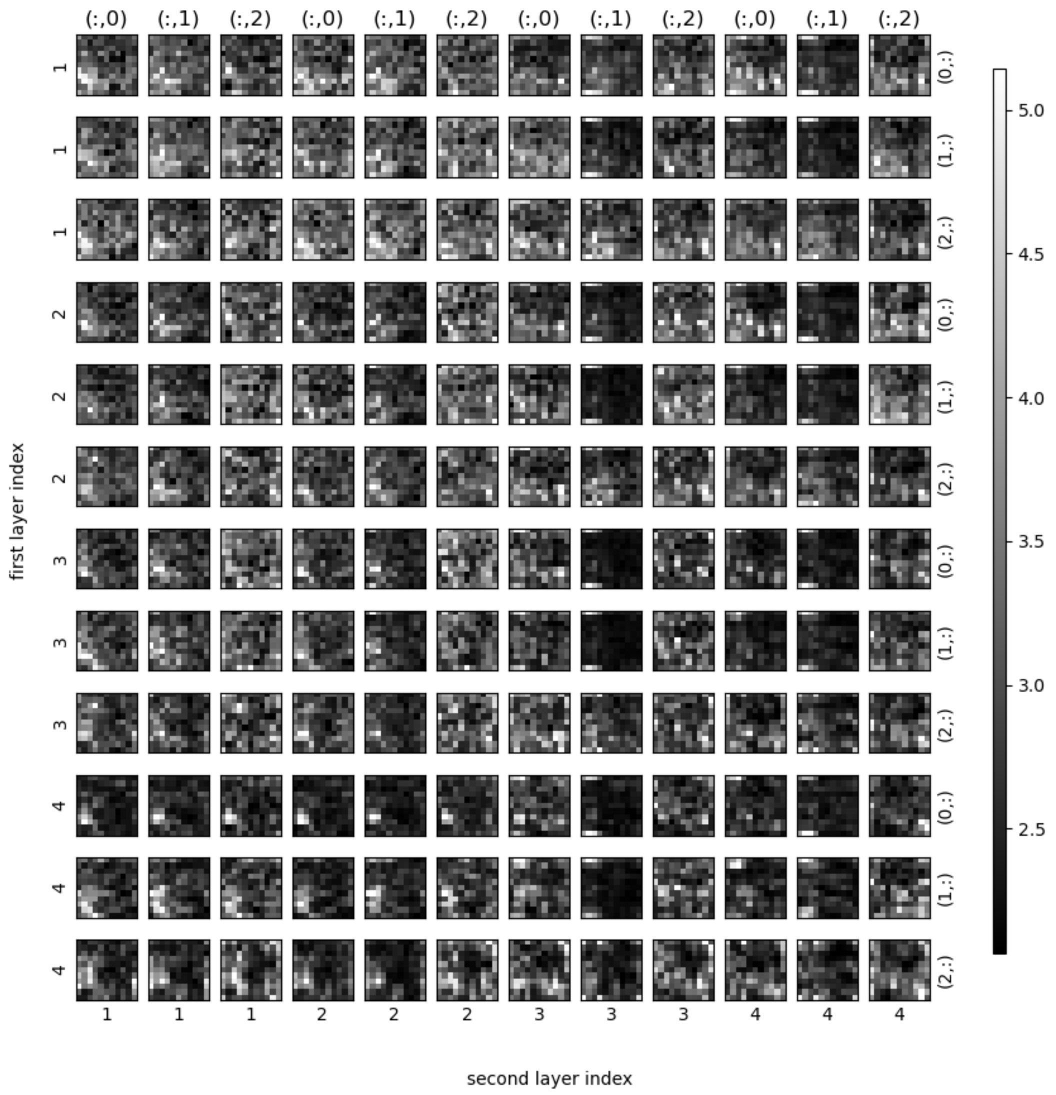


FIGURE 5.6. The MWSN 2nd-layer output of the image shown in Figure 5.5a

along with subsampling and modulus operation. In Figure 5.6, the (i, j)-th block consists of 25×25 MWSN coefficients $\boldsymbol{s}_{j_1,j_2}^{(l_1,l_2)}$ with

(1) $l_1 = (i-1) \mod 3$,

(2) $l_2 = (j-1) \mod 3$,

(3) $j_1 = \lfloor (i-1)/3 \rfloor + 1$,

(4) $j_2 = \lfloor (j-1)/3 \rfloor + 1$,

where $i, j \in \{1, \ldots, 12\}$. For example, the lower right $(i, j) = (12, 12)$-th block represents $s^{(2,2)}_{4,4}$. As we traverse from left to right at each row of Figure 5.6, we capture more intricate texture information. Hence, we can visualize the complex textures in the 2nd-layer.

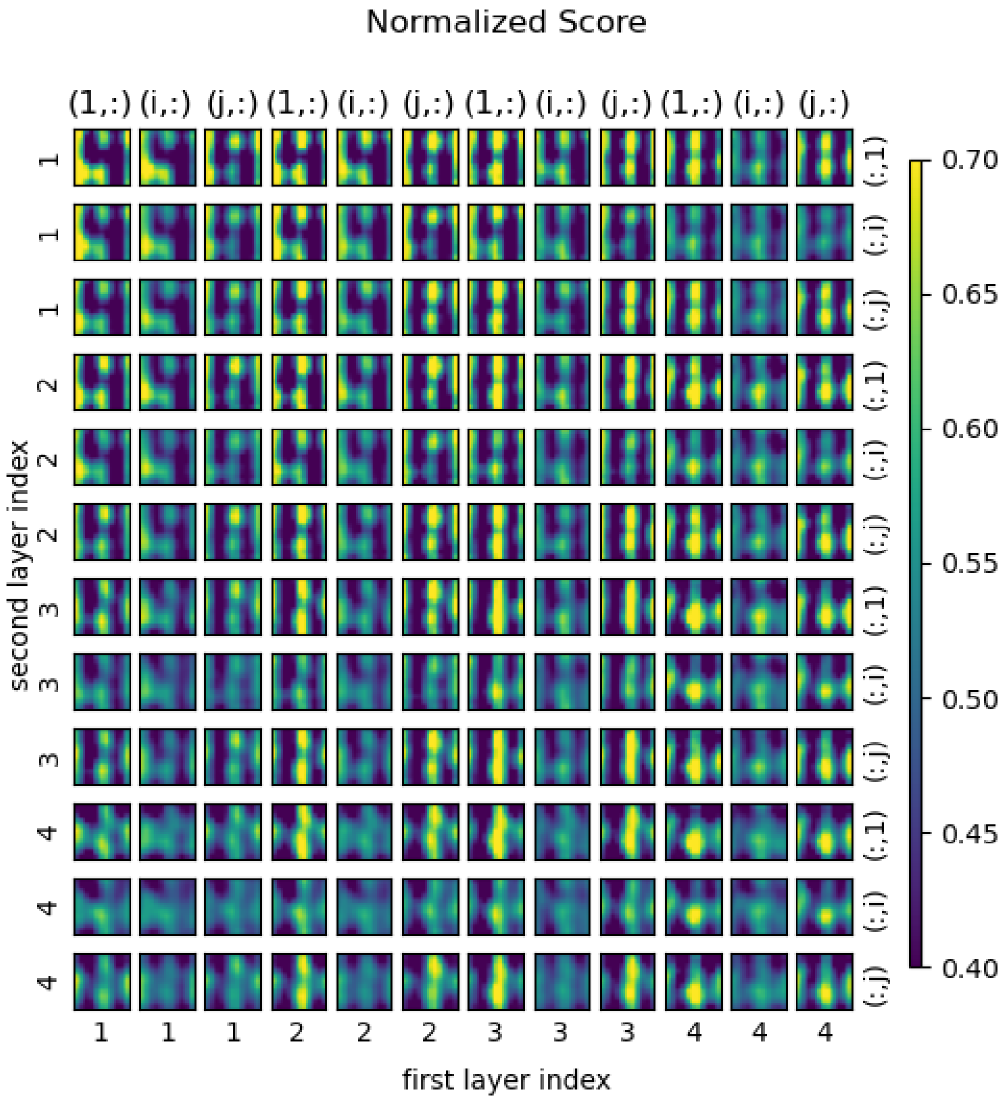


FIGURE 5.7. The normalized score on test dataset to measure the extent of the feature map in contribution to the effective classification

5.3.4. Result Interpretation. Our work goes beyond plain classification. We also explored the possibility to explain the result by determining which MWSN coefficients are significant in

Method	Dimension in PCA	Classification Accuracy
Kymatio-ST-GLMnet	30	91.1190%
MWSN-GLMnet	30	93.2038%
MWSN-GLMnet	100	95.1247%
RMWSN-GLMnet	30	95.0143%
RMWSN-GLMnet	100	95.8446%

TABLE 5.2. Table of average test accuracy over 10 experiments with MWSN-GLMnet

texture classification using GLMNet [**51**, Chap. 3]. Table 5.2 shows the test accuracy by repeating two-fold cross validation 10 times in each case together with PCA-compressed coefficient / feature vectors. We could also draw the same conclusion that the MWSN performed better than Kymatio-ST, and our RMWSN showed a better performance than that of the MWSN. Even though the accuracy was higher with slightly higher PCA dimension (say 100) than PCA dimension of 30, we set the same PCA dimension to 30 for the rest of our experiments using MWSN for consistency.

The normalized scores as demonstrated in Figure 5.7 could be computed by finding the PCA component index that corresponds to the highest standard score of the parameter $|\boldsymbol{\theta}|$, where $|\boldsymbol{\theta}| := \left[|\theta_1|, \ldots, |\theta_n|\right]$, for which the mean loss is minimized in GLMNet. Our result from Figure 5.7 showed that the mixed directional textures $(\mathbb{i}, \mathbb{j})$ did not attribute a higher contribution in the effective classification than the directional textures $(1, \mathbb{i}), (1, \mathbb{j}), (\mathbb{i}, \mathbb{i}), (\mathbb{i}, \mathbb{j})$. Meanwhile, the result could be attributed more to the vertical textures than the horizontal textures. In particular, the vertical textures were significant in all frequency components represented by the scale indices in Figure 5.7.

In summary, our newly proposed MWSN and RWSN are able to extract the texture information that are easily interpretable. Our work addresses the lack of interpretability from the conventional deep learning algorithms. In addition, we made use of the MWT to construct a new STN that performed better than Kymatio-STN under the same setting and allows a straight-forward interpretation on the textures. To improve the interpretability of the MWSN coefficients that are important for classification, we have two plans: 1) convert the second layer coefficients $\boldsymbol{s}_{j_1,j_2}^{(l_1,l_2)}$ into the instantaneous amplitude, phase, and orientation representation via Equation (2.43) before applying the PCA; and 2) replace the PCA by the Local Discriminant Basis (LDB) method [**104, 105**] since the latter can directly extract features that are helpful for classification instead of extracting high variance features by the PCA.

CHAPTER 6

Scattering Transform in Sonar Signal Processing

In this chapter, we present the work which is an extension of Reference [**106**] from Professor Saito's group with the new setting of 2-D STN. The work contributes to the presentation on "Robust features Extraction from Acoustic Wavefields for Object Classification" in the Office of Naval Research (ONR) MCM Virtual Program Review in 2021 and 2022. In particular, the 2-D acoustic wavefields can be represented as a matrix consisting of waveforms collected at different receiver locations relative to an object of interest. Analyzing the 2-D acoustic wavefields should allow us to extract more object information such as the the curvature of hyperbolic events than analyzing the individual 1-D waveforms separately. We believe a 2-D STN is able to extract more robust features from the 2-D acoustic wavefields than applying 1-D STN on 1-D acoustic waveforms separately.

6.1. Problem Overview in Sonar Signal Processing

Mine or underground object detection depends on beamforming of sonar waveforms and interpretation of resulting images generated from those sonar waveforms. In particular, *unexploded ordinance* (UXO), which is any military ammunition that has failed to function as designed, can be detected through deploying *unmanned underwater vehicles* (UUV) supplied with *synthetic aperture sonar* (SAS). See Figure 6.1 for some examples of UXO. As a consequence of many human conflicts, plenty UXO remains are found on the seabed [**70**], hence imposing risks to ships and contaminating the environment. One possible way of detecting and classifying them like dolphins do is to make use of the raw sonar waveforms and the wavefield scattered from an object.

Some preliminary work for direct automated detection of UXOs from the raw sonar signals have been done in Professor Saito's group [**76**, **83**, **84**]. Scattering transform is recently used as a tool to naturally process the sonar data [**106**]. The modeled mechanism for generating the waveforms allows us to characterize the variation by space-frequency deformation due to the properties of the

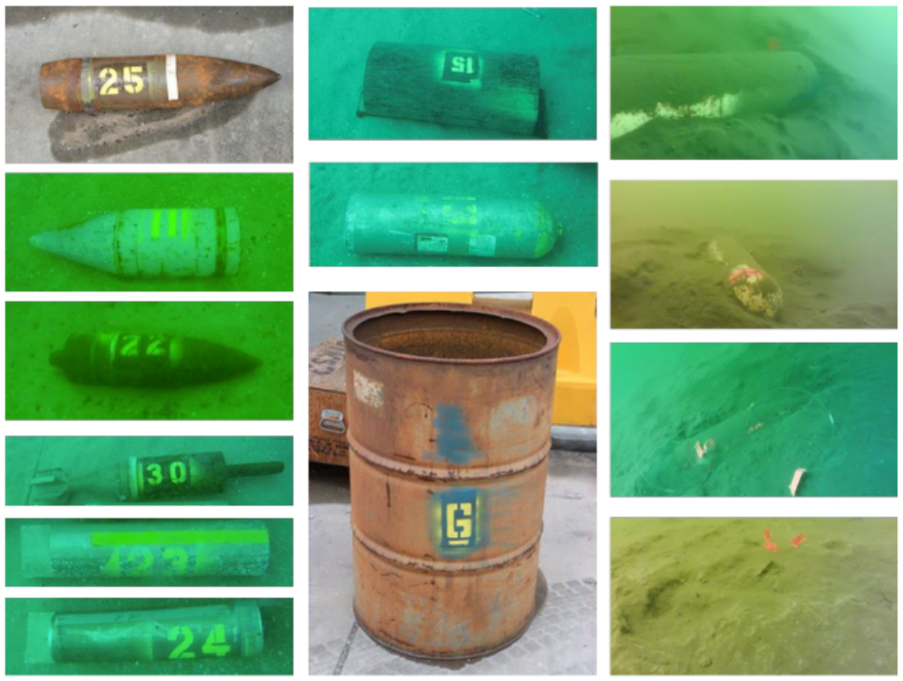
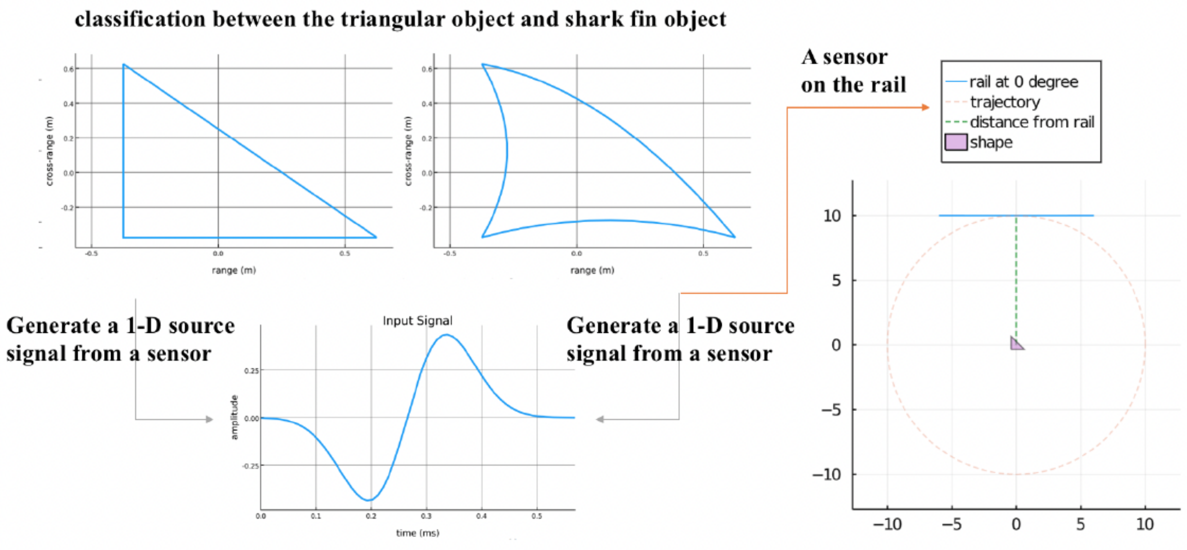


FIGURE 6.1. Example of unexploded ordinance (UXO) in a target environment [**61**].

objects and the measurement conditions. Then the PCA-compressed scattering coefficients are fed into a linear classifier such as the GLMNet [**51**, Chap. 3]. Meanwhile, the result can be naturally interpreted by the scattering coefficients and the fitted coefficients in the linear model as we already discussed in Chapters 4 and 5.

The prior work by Saito and Weber [**106**] applied the scattering transform on the BAYEX14 dataset [**61**] which contains acoustic wavefields scattered from 14 objects partially buried at multiple distances from the location of the transducer/receiver pair with various rotated positions on the top of a sand ocean bed at a depth of about 8 meters in a layer of shallow mud. A sensor on a rail as shown in Figure 6.2 on the right rotates around the object and records at different angles, each of which generating a 2-D wavefield. Multiple sensor locations on a rail act like a multi-detector for fetal peak detection [**63**] which enables an improvement in detecting sonar objects. See Appendix A for more details of the published work [**63**].

The sensor in practice acts like an imaginary grid under water. It sends an acoustic wave and records the response coming from transmissions and reflections at each nod of the grid. On the



classification between the triangular object and shark fin object
A sensor on the rail
rail at 0 degree
trajectory
distance from rail
shape
Generate a 1-D source signal from a sensor
Input Signal
Generate a 1-D source signal from a sensor
range (m)
cross-range (m)
amplitude
time (ms)

FIGURE 6.2. The 1-D signal can be generated from a sensor on an receiver location detecting an underwater object using Gabor signal as a source.

other hand, synthetic waveform by a numerical fast solver using the Helmholtz equation which are integral equations of scattering theory, in the low- and mid-frequency regimes at different sound speed can be produced [**13**,**14**,**15**]. The solver we used in the numerical experiments in this chapter models what happens when an underground transmitter delivers an acoustic wave to track objects characterized by their shape and the wavespeed into them.

6.2. Sonar Wavefield Generation using Gabor Function as the Source

6.2.1. 2-D Helmholtz equation. Each receiver point on the rail sends an acoustic wave which is harmonic in time and propagates in two different fields. Firstly the wave propagates into a material until reaching the object. Then a part of the wave under transmission to the object propagates again until reaching the next edges of the object, and other part of the wave is reflected. Mathematically, we define the acoustic impedance Z_i into the field indexed by i as the sound pressure divided by the speed of particle and the surface area in which an acoustic wave propagates. The transmission and reflection coefficients are given by

$$\begin{aligned} \eta_{\text{reflection}} &= \left(\frac{Z_1 - Z_2}{Z_1 + Z_2}\right)^2 \\ \eta_{\text{transmission}} &= \left(\frac{4Z_1Z_2}{Z_1 + Z_2}\right)^2. \end{aligned} \tag{6.1}$$

In our simulation model, there are two components: one inside the object where the wave frequency does not alter, one outside the object with change in wavenumber. Boundary conditions like reflection and transmission are also crucial in the interface. The 2-D Helmholtz equation, which models scattering from an interface between two fluids, synthesizes the SAS dataset as described in Reference [**13**] as follows.

Set Ω^c to be the surrounding 2-D space and Ω to be the object. Let u be the response to a sinusoidal signal in the domain Ω with frequency ω. Note that the wavenumber $k_i = \omega/c_i$, where c_i is the speed of sound in the material. For example, c_1 is the speed of sound of the object and c_2 is the speed of the surrounding medium, i.e., water. Let v be the response outside the domain Ω. Then the partial differential equation (PDE) is

$$\begin{gathered} \Delta u + k_1^2 u = 0 \text{ in } \Omega \\ \Delta v + k_2^2 v = 0 \text{ in } \Omega^c \\ u - v = g \text{ on } \partial\Omega \\ \partial_\nu u - \partial_\nu v = \partial_\nu g \text{ on } \partial\Omega \\ \sqrt{|x|}(\partial_{|x|} - ik_2)v(x) \to 0 \text{ as } |x| \to \infty, \end{gathered} \tag{6.2}$$

where the function g can be set to 0 and ∂_ν is the normal derivative. In particular, if the last condition in Equation (6.2) is dropped, then the solution can be written as

$$w(\boldsymbol{x}, t) = u(\boldsymbol{x})\mathrm{e}^{\mathrm{i}\omega t}.$$

Note that the ranges of the speed of sound c_i are from 343 m/s in air, 1503 m/s in water to 5100 m/s in aluminum. A fast solver developed by Ian Sammis and James Bremer [**13, 15**] can be employed to synthesize our sonar dataset. The initial data generation method was proposed by Vincent Bodin [**12**], who is a summer intern formerly under the supervision by Professor Saito. We model the material as a fluid in an 2-D interface with just one layer, and establish the following settings:

(1) The sensor follows a straight-line trajectory line called rail.
(2) The surrounding space where the acoustic waves propagate is sent is a 2-D space.
(3) The borders in the surrounding space are not taken into account in the entire 2-D space.

The observation point $\boldsymbol{x}_0(r) = (x, r)$ on a rail can be visualized in Figure 6.2 on the right, where $x = 10$m is the distance between the center on a rail and the underwater object, and r is the displacement of a receiver on the rail ranged from -6m to 6m. In the simulation process, the 481 receiver points (indexed from -240 to 240) on the rail are spaced out evenly to emit acoustic waves and record the responses. Define $\boldsymbol{x}_\theta(r) = R_\theta \boldsymbol{x}_0(r)$ as the point of the rotated rail by the rotation matrix R_θ.

Let $s \in L^1(\mathbb{R}) \cap L^2(\mathbb{R})$ be a multi-frequency input signal. An example of input signal is a Gabor function which is displayed on the left in Figure 6.2. The response from the input signal s can be approximated by integrating across different frequencies. Suppose v is the solution to the 2-D Helmholtz equation outside the domain Ω as described in Equation (6.2), and P is the phase information from v. For a transmitter located at $\boldsymbol{x} \in \Omega^c$, the ideal reconstruction of the response f to the input signal s is given by

$$f(t, \boldsymbol{x}) = \int_{\mathbb{R}} \mathcal{F}[s](\omega) v(\omega, \boldsymbol{x}) \mathrm{e}^{\mathrm{i}\omega\left(t - P(\omega, \boldsymbol{x})\right)} \, \mathrm{d}\omega. \tag{6.3}$$

Zero padding on the input signal s is crucial to generate the sonar data by the Helmholtz equation in order that the periodic version computed through FFT is similar to the version with finite support. Note that in practice the response f is computed using the associated frequencies range used to generate the input source signal.

6.2.2. 2-D Wavefield Generation. Our first SAS synthesis data setup is the target shape discrimination between triangular object and shark-fin object as shown in Figure 6.2. These objects can be designed before we proceed to the PDE solver on the 2-D Helmholtz Equation as described in Equation (6.2).

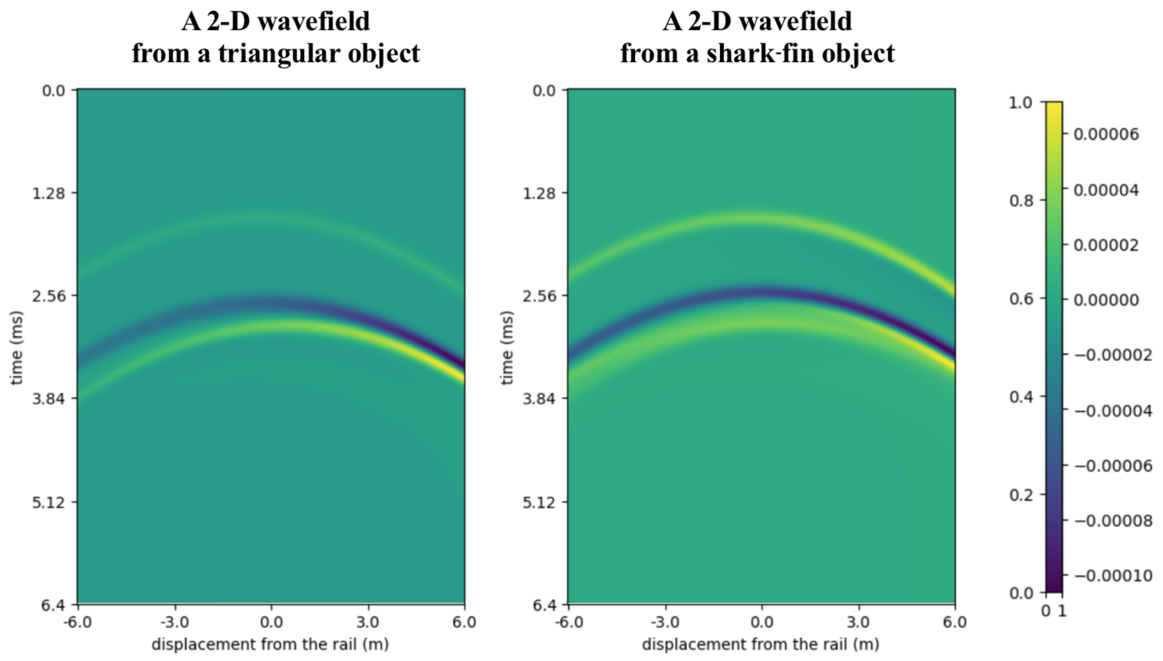


FIGURE 6.3. Example of 2-D wavefields in the SAS dataset at angle 0.

(1) The vertices of a triangular object centered at 0 are located at

$$(-0.625, -0.375), (0.375, -0.375), (0.375, 0.625).$$

(2) The vertices $p_1 = \left(p_1^{(0)}, p_1^{(1)}\right), p_2 = \left(p_2^{(0)}, p_2^{(1)}\right), p_3 = \left(p_3^{(0)}, p_3^{(1)}\right)$ of a shark-fin object are the same as those of the triangular object. We can parametrize edges of the shark-fin object by a a quadratic curve across the vertices. The intermediate points to fit the polynomial of degree 2 are

$$\left(\frac{p_1^{(0)} + p_2^{(0)}}{2}, \frac{9p_2^{(1)} + p_3^{(1)}}{10}\right), \left(\frac{p_1^{(0)} + 9p_2^{(0)}}{10}, \frac{p_2^{(1)} + p_3^{(1)}}{2}\right), \left(\frac{p_1^{(0)}, +p_2^{(0)}}{2}, \frac{p_1^{(1)} + 3p_3^{(1)}}{4}\right).$$

We can generate the full 2D wavefields for both triangular object and shark-fin object as shown in Figure 6.3 for object characterization.

For each object, we synthetically generate sonar signals with the following scenarios:

(1) 3 different acoustic velocities (2000 m/s, 2250 m/s, 2500 m/s).

(2) 5 different distances (5.2m, 10.2m, 15.2m, 20.2m, 25.2m) from the center of rail to the center of the object.

We also rotate each object with 36 different angles, uniformly from 0 degree to 350 degree. Therefore, there are 540 wavefields generated for each object. We normalize the signal by the matrix norm $\|\cdot\|_2$, and add the white Gaussian noise with signal-to-noise ratio (SNR) = 5dB.

As mentioned in Subsection 6.1 of this chapter, we need to create a database for a set of monochromatic signals that retrieve the response as the wavefield. The set of 320 frequencies covered in the database is

$$\Big\{ \quad 1 * 156.25 \text{ Hz}, \quad 2 * 156.25 \text{ Hz}, \quad \cdots, \quad 320 * 156.25 = 50000 \text{ Hz} \quad \Big\}.$$

In each signal, we have 641 time samples and 481 1-D waveforms. We split the data evenly and randomly into training and testing set to perform two-fold cross validation by 10 times.

6.3. Sonar Detection by Monogenic Wavelet Scattering Network

Let $g(\boldsymbol{x})$ be a 2-D shark-fin wavefield. We can extract additional information based on the monogenic representation of the sonar signal. For instance, Figure 6.4 displays the Riesz-x and Riesz-y components of the monogenic signal of a shark-fin signal shown in Figure 6.3. We write the monogenic signal as

$$g^{+}(\boldsymbol{x}) = g(\boldsymbol{x}) + \mathbb{i} g^{(1)}(\boldsymbol{x}) + \mathbb{j} g^{(2)}(\boldsymbol{x}),$$

where $g^{(1)}(\boldsymbol{x})$ and $g^{(2)}(\boldsymbol{x})$ corresponds to the Riesz-x and Riesz-y components respectively. See Subsection 2.4 for more details on the mathematical background on monogenicity.

Based on Equation (2.43), the monogenic signal also has its own monogenic amplitude, local phase and local phase direction. While the monogenic amplitude measures the phase-invariant magnitude of the monogenic signal, the local phase is an angular quantity by comparing the ratio of the signal $g(\boldsymbol{x})$ as the adjacent side, and the amplitude of the monogenic signal $\|g^{+}(\boldsymbol{x})\|$ as the hypotenuse. Based on the local phase in Figure 6.4, the contextual information with strong amplitude (i.e., the sensible curves in the middle) is dominated by the arches bent from the horizontal line, while the background is dominated by the vertical lines.

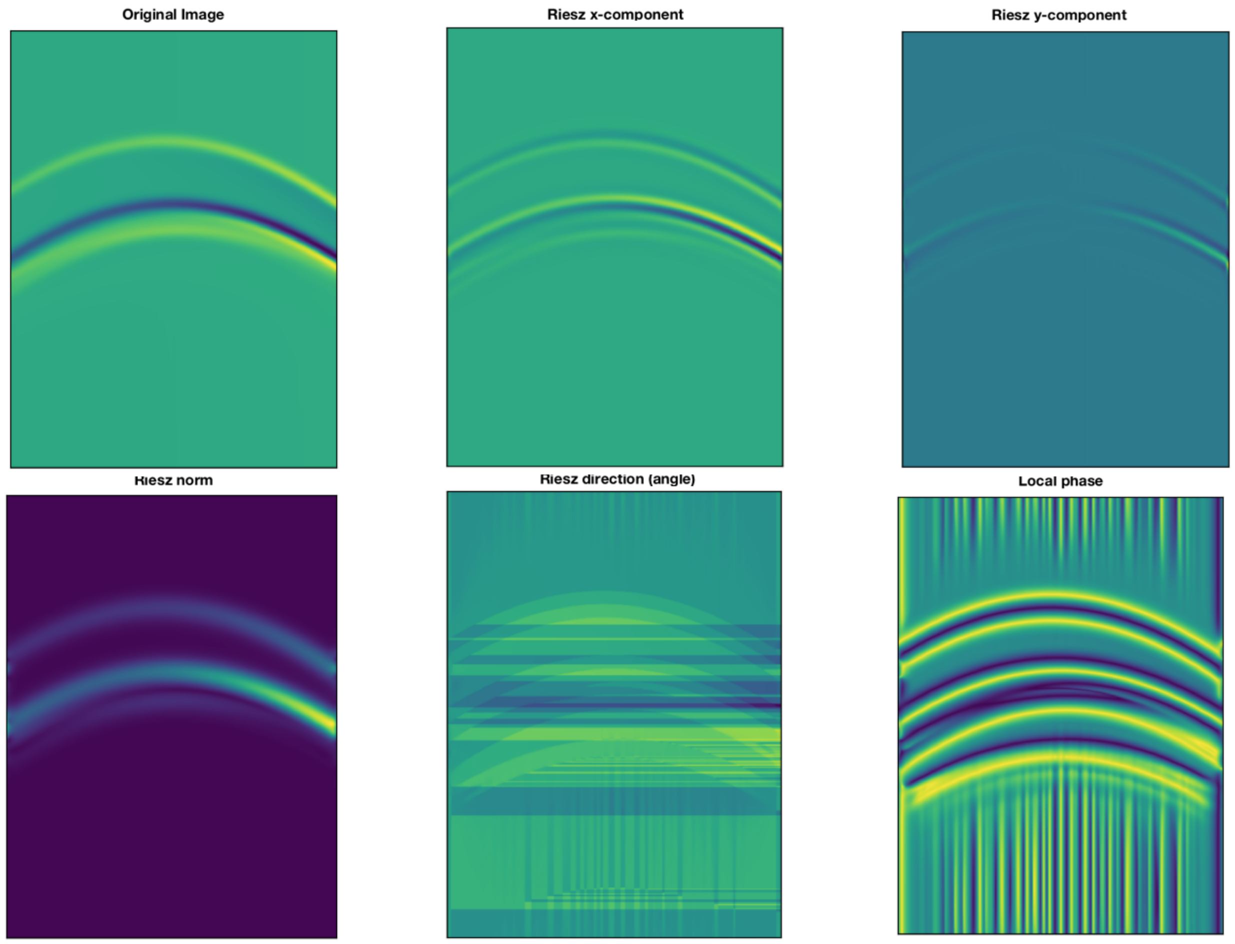


FIGURE 6.4. The monogenic signal of a shark-fin object.

Furthermore, Figure 6.5 shows that the monogenic signal can be transformed with different scales. The subfigures in the first three columns have the same notation as the 1st-layer scattering outputs represented by the yellow boxes in the MWSN in Figure 5.1, except that the monogenic representations in Figure 6.5 do not go through the subsampling process. For every scale j, the monogenic signal at that scale can be expressed as

$$g^{+}_{1,j}(\boldsymbol{x}) = g_{1,j}(\boldsymbol{x}) + \mathbb{i}g^{(1)}_{1,j}(\boldsymbol{x}) + \mathbb{j}g^{(2)}_{1,j}(\boldsymbol{x}),$$

where $g_{1,j}(\boldsymbol{x})$, $g^{(1)}_{1,j}(\boldsymbol{x})$ and $g^{(2)}_{1,j}(\boldsymbol{x})$ are the 2-D signal, the Riesz-x component, Riesz-y component respectively at the scale j as displayed in Figure 6.5. The subscript 1 refers to the 1st-layer of the MWSN. For each monogenic signal $g^{+}_{1,j}$, we can find the monogenic amplitude, local phase and

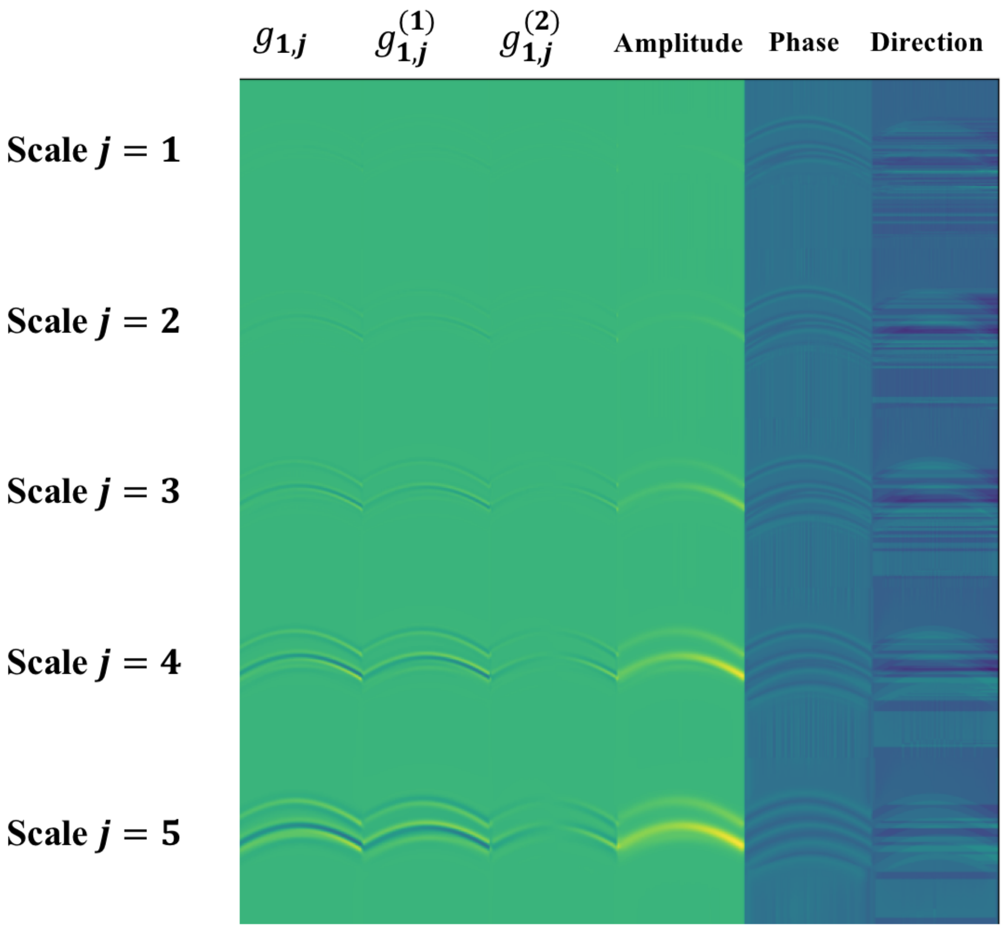


FIGURE 6.5. The multi-resolutional monogenic signal of a shark-fin object.

local phase direction in reference to Equation (2.43). Again, for every scale in the 5th and 6th columns of Figure 6.5, the contextual information corresponding to the large monogenic amplitude is dominated by the arches bent from horizontal, and the background is dominated by the vertical lines.

6.3.1. The MWSN features extractor and classification performance in SAS. As mentioned in Subsection 6.1, we used the SAS sonar dataset [**12**, **106**, **129**] for evaluating the classification performance for different objects. The dataset consists of raw sonar waveforms scattered from certain objects for the SAS processing. In this subsection, we employed the synthetic examples from two objects (shark-fin and triangular objects) as described in Subsection 6.2.1. We

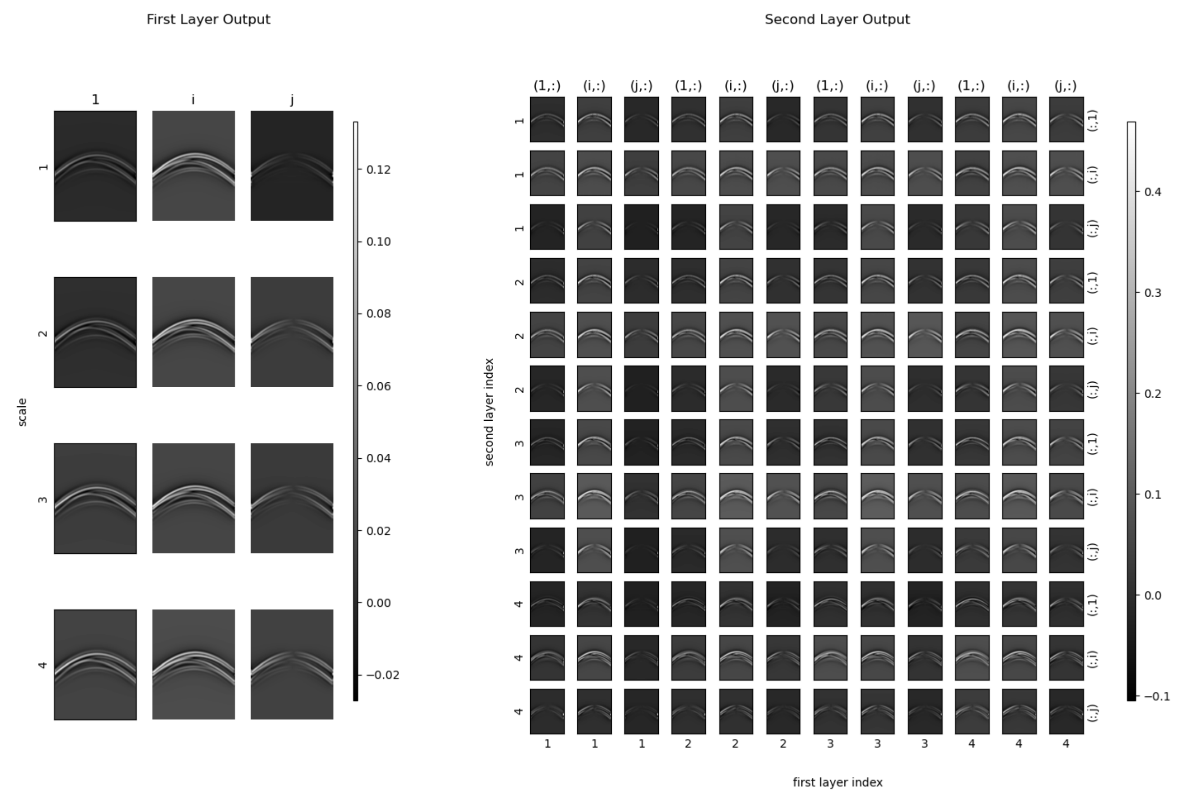


FIGURE 6.6. The 1st-layer MWSN (left) and 2nd-layer MWSN (right) representation of a 2-D signal from the shark-fin object in the SAS dataset.

compared the classification performance among the original 2-D wavefields, the 1st- and 2nd-layer MWSN features, and the corresponding PCA hierarchical features with no more than 1000 features dimensions.

Figures 6.6 and 6.7 show the MWSN features from a shark-fin object and a triangular object respectively with the maximum scale $J = 4$. In particular, Figure 6.6 display the 1st-layer MWSN features on the left and the 2nd-layer MWSN features on the right from a shark-fin object. The 1st-layer MWSN features have a very similar structure with the multiscale monogenic signal represented in Figure 6.5, except that 1st-layer MWSN features were extracted through the additional subsampling operations and averaging with a low-pass filter. To be specific, we set the subsampling rate $r_m = r'_m = 2$ in the MWSN. The 2nd-layer MWSN features on the right has more contextual details than the 1st-layer features. Each row index s in the right subfigure has the corresponding scale $\lceil s/3 \rceil$ and the corresponding contextual emphasis (isotropic, Riesz-x, Riesz-y) on the $1, \mathbb{i}$ or

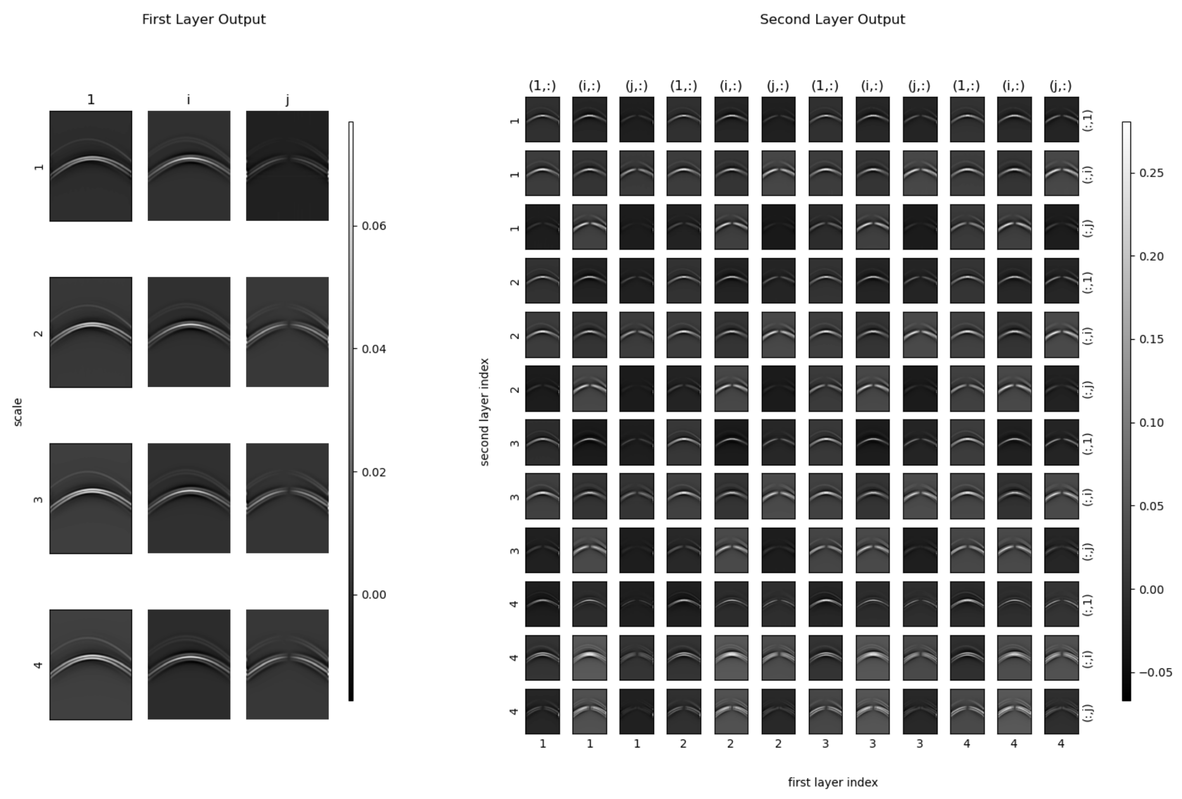


FIGURE 6.7. The 1st-layer MWSN (left) and 2nd-layer MWSN (right) representations of a 2-D signal from the triangular object in the SAS Dataset.

𝕛 components respectively. The column index represents the 1st-layer information from top left to bottom right order in the left subfigure, which is used to compute the 2nd-layer information. Hence, we can look at the resulting features with different directional information with different scales from the 1st-layer, and visualize the contextual details further microscopically with different resolutions and paired directions in the 2nd-layer. The similar patterns of the 1st-layer MWSN features on the left and the 2nd-layer MWSN features from the triangular object can be observed in Figure 6.7. To discriminate these features, we need a classifier and performance metric to quantify the classification performance.

The *area under the curve* (AUC) of the *receiver operating characteristics* (ROC), which is a performance metric with graphical visualization for binary classification at multiple threshold settings [**29**]. The ROC curve is a probability curve plotted with the *true positive rate* (TPR) on the y-axis against the *false positive rate* (FPR) on the x-axis. The AUC, which measures the degree of separability, can be computed by finding the area under the ROC curve. The range of the AUC is from 0.5 for random guessing, to 1 for a perfect classifier.

We computed the AUC from our experiments by repeating two-fold cross validation 10 times together. In our experiments, the Julia package for GLMNet [**66**] was used as the classifier. Dimension reduction is necessary before feeding the features into the GLMNet classifier because the initial features dimension is too large for our classifier, and we used the PCA as implemented in [**59**]. We set the PCA dimension to be 1000 at maximum for our MWT and MWSN features. The precise PCA dimension was determined by the PCA algorithm with the ratio of variance preserved in the principal subspace (pratio) set to be 0.99. We compared the classification performance with various features fed into the the classifier:

(1) original wavefields
(2) Absolute value of the 2-D Fourier transform (AVFT)
(3) Monogenic wavelet transform (MWT)
(4) the 1st-layer MWSN (L1-MWSN)
(5) the 2nd-layer MWSN (L2-MWSN)
(6) AVFT of the 1st-layer MWSN (L1-AVFT-MWSN)
(7) AVFT of the 2nd-layer MWSN (L2-AVFT-MWSN).

The ROC curves of the following features are shown in Figure 6.8. In particular, the AUC was around 0.9802 for the L2-AVFT-MWSN features , i.e., the absolute value of the Fourier transform (AVFT) of the MWSN in the 2nd-layer. The L2-AVFT-MWSN features outperformed the other features in our experiments. The AVFT together with the hierarchical features extracted in the MWSN capture more intrinsic geometric information in the 2-D wavefields. It suggests that our L2-AVFT-MWSN features together with a classifier is capable of discriminating the two objects from their 2-D wavefields under various scenarios.

We further explored the possibility to interpret the results from the 2nd-layer MWSN features. The features shown in Figure 6.9 is the projection of the L2-MWSN space from the single top principal components. Each row index has the corresponding scale and the corresponding contextual direction on the 1 (isotropic), $\mathbb{i}$ (time), and $\mathbb{j}$ (offset) components. We found the PCA component index corresponding to the highest standard score of the parameter $|\boldsymbol{\theta}|$ from the L2-AVFT-MWSN

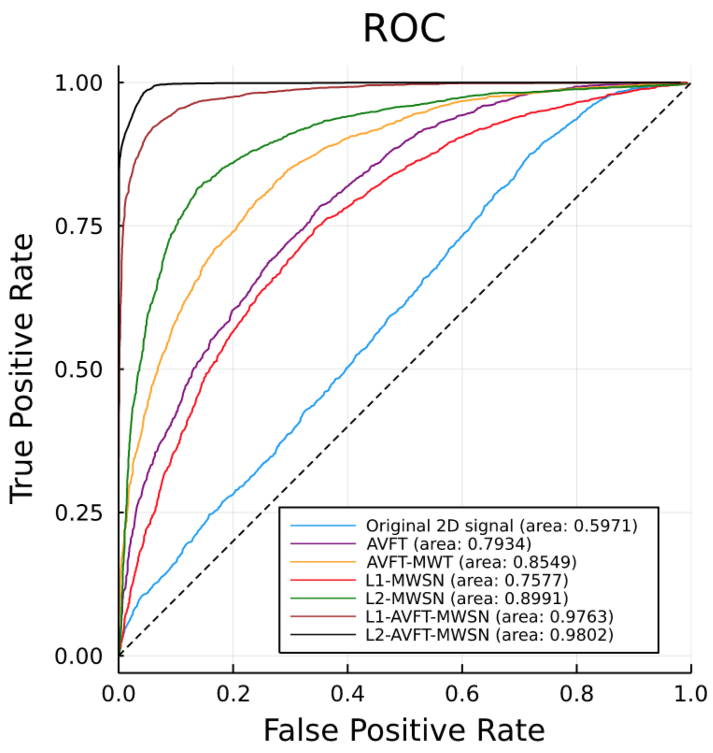


FIGURE 6.8. The AUC performance on the SAS dataset under the ROC curve.

features for which the mean loss is minimized in GLMNet, and then projected it back to obtain a set of significant coefficients corresponding to the L2-MWSN. We indicated the most significant components by red boxes, which correspond to the isotropic and the Riesz-x components (corresponding to the time) of larger scales. Taking the absolute value of the Fourier transform of the L2-MWSN coefficients has improved the classification rates thanks to the additional translation invariance.

6.4. Object Classification using the dolphin's Clicks

6.4.1. Dolphin's clicks as the source for synthesizing 2-D acoustic wavefields. Obstacle avoidance technology is crucial for the US Navy to conduct shallow water and very shallow water (SW and VSW) mine counter-measure (MCM). In particular, UXO is a concern as mentioned in Subsection 6.1. To address the problem of mine detection and hunting, the dolphin-based marine mammal systems (MMS) are adopted in the US Navy [**86**]. The trained bottlenose dolphins naturally possess a SW and VSW-adapted biological sonar. With the native echolocation abilities

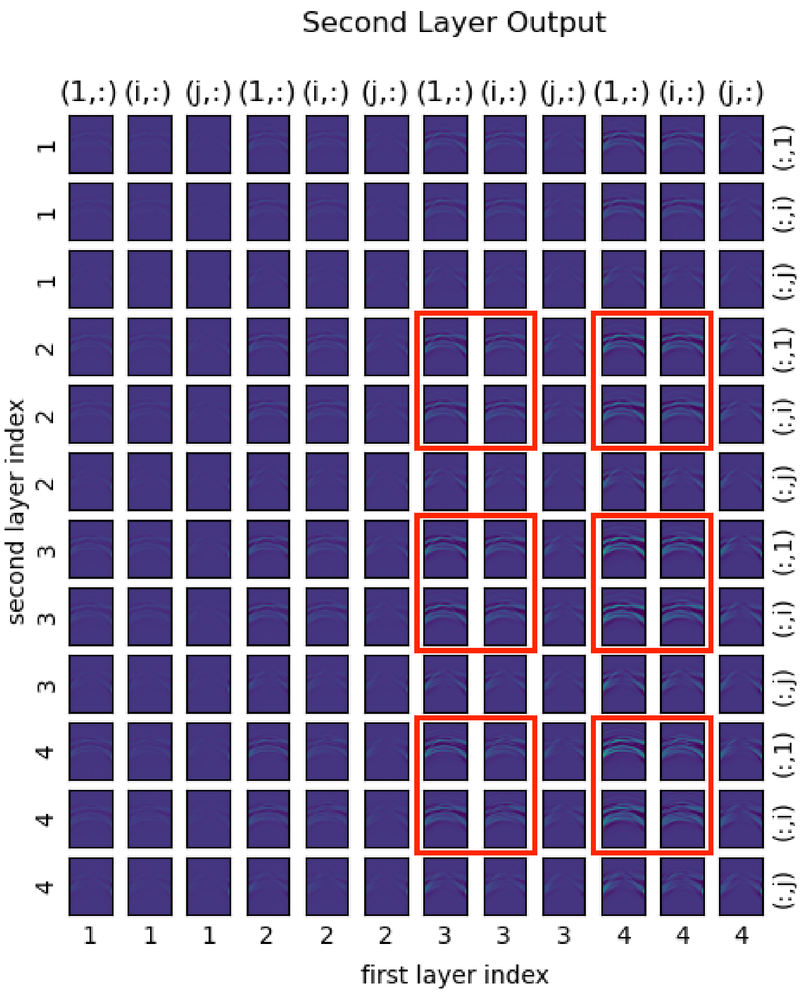


FIGURE 6.9. The normalized score in the 2nd-layer MWSN.

of dolphins adapted for their habitats, it has been demonstrated that the dolphins' biological sonar outperforms hardware systems available for mine detection and classification in various difficult environments such as SW and VSW and with buried mines [**86**].

As described by Au [**8**], dolphins have naturally evolved high-frequency echolocation which is similar to the biosonar possessed by bats. Being a predator and prey like bats, dolphins coevolves prey hearing and predator signaling through increasing frequencies of echolocation signals to extend outside their preys and predators [**119**]. Herzing [**53**] suggested that dolphins use echolocation to distinguish different preys such as fish and squid. The echolocation signals can be produced by dolphins with bimodal frequency spectra of a relatively low-frequency peak from 40 to 50 kHz, and

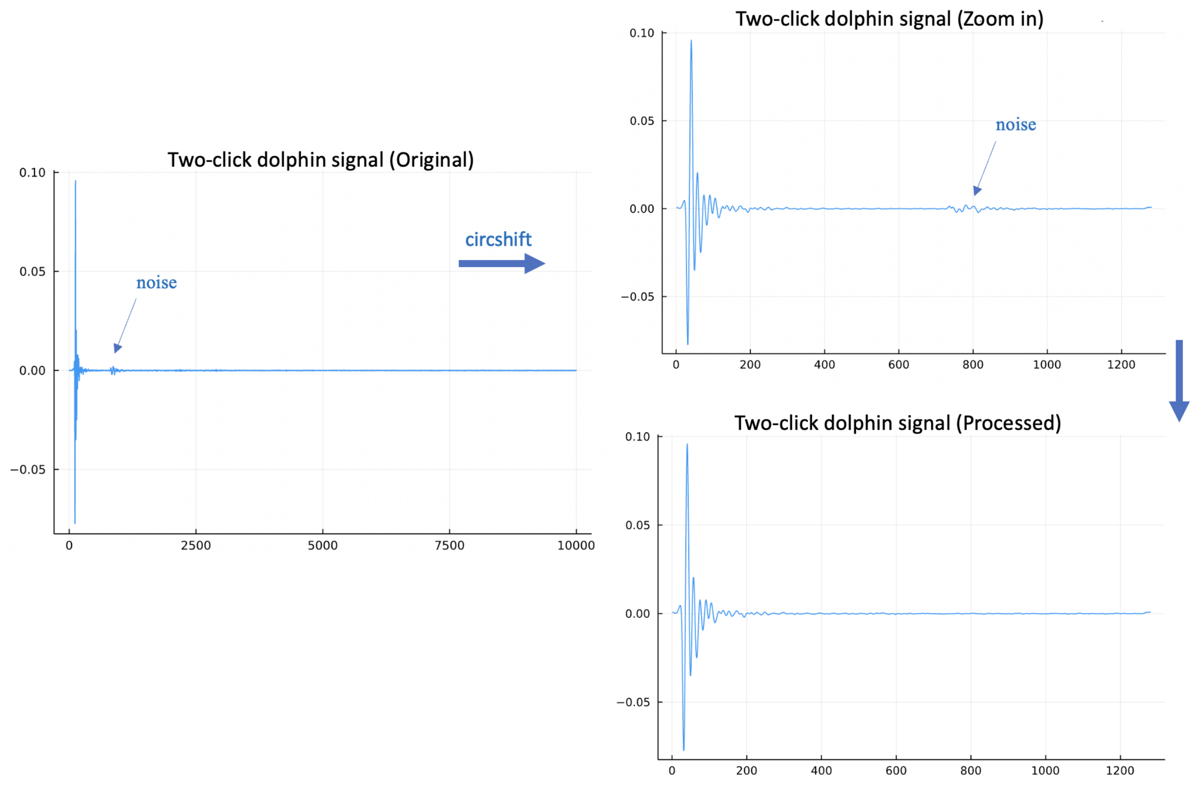


FIGURE 6.10. The processed dolphin signal.

a very high-frequency peak from 110 to 130 kHz [9]. Some underwater hearing experiments have revealed that bottlenose dolphins can detect low frequency sound between 50 and 150 Hz, with similar response patterns to the test signal at higher frequencies [121].

Because of the advantage of dolphin's echolocation capabilities, we used dolphins' clicks as a source instead of the Gabor function in the previous subsections, to generate the responses from the triangle and shark-fin objects. We obtain the recordings of clicks of a dolphin from Dr. Dorian Hauser of National Marine Mammal Foundation, as shown in Figure 6.10 on the left. A dolphin typically generates many clicks successively, resulting in more complicated patterns. In our signal, there are 2 dolphin clicks, and the first click possesses the most information. The sampling rate for the dolphin signal is 2 MHz, i.e., the sampling rate is

$$\Delta t = 1/(2 \times 10^6) = 0.5\mu \text{ second.}$$

Initially, we circularly shifted (circshift) the dolphin signal in Figure 6.10 so that the support of the signal is closer to the origin, just like the Gabor-function signal in Figure 6.2. We cropped the signal with length of 1281 as indicated on the right of Figure 6.10. To further discard the noise in the cropped dolphin signal, we replace the second half portion (i.e., $t = 641, 642, \cdots, 1281$) of the cropped signal by the portion (i.e., $t > 1281$) after the cropped portion. The response generated from the dolphin signal is mainly dependent on the first half portion which is similar to the Gabor-function signal.

6.4.2. Wavefield generation using 2-D Helmholtz equation. We can generate the wavefields from the 2-D Helmholtz equation (6.2) using the dolphin signal as the source. As described in Subsection 6.2.1, a sensor acts like an imaginary grid under water, sends an acoustic wave and records the response coming from transmissions and reflections at each node of the grid. Subsection 6.2.2 outlines the detailed procedure to generate 2-D wavefields using the 2-D Helmholtz equation, except that we use the dolphin signal as a main source here. In addition, we alter the database for a set of monochromatic signals that retrieve the response from the dolphin signal. The set of 640 frequencies covered in the new database is

$$\Big\{ \quad 1 * 80 \text{ Hz}, \quad 2 * 80 \text{ Hz}, \quad \cdots, \quad 640 * 80 = 51200 \text{ Hz} \quad \Big\}.$$

Figure 6.11 display the 2-D wavefields generated from triangular object and shark-fin object using the dolphin signal as a source.

6.4.3. The multiscale MWT of wavefields. Figure 6.12 displays the monogenic signal of a shark-fin object with different scales using the dolphin signal as the source for wavefield generation. In the first three columns, the subfigures have the same notation as the scattering outputs in the first layer represented similarly by the yellow boxes in the MWSN in Figure 5.1. The major difference is the omission of the subsampling procedure for the multiscale monogenic signal representation in Figure 6.12. For every scale j, the monogenic signal at that scale can be written as

$$g^{+}_{1,j}(\boldsymbol{x}) = g_{1,j}(\boldsymbol{x}) + \mathbb{i} g^{(1)}_{1,j}(\boldsymbol{x}) + \mathbb{j} g^{(2)}_{1,j}(\boldsymbol{x}),$$

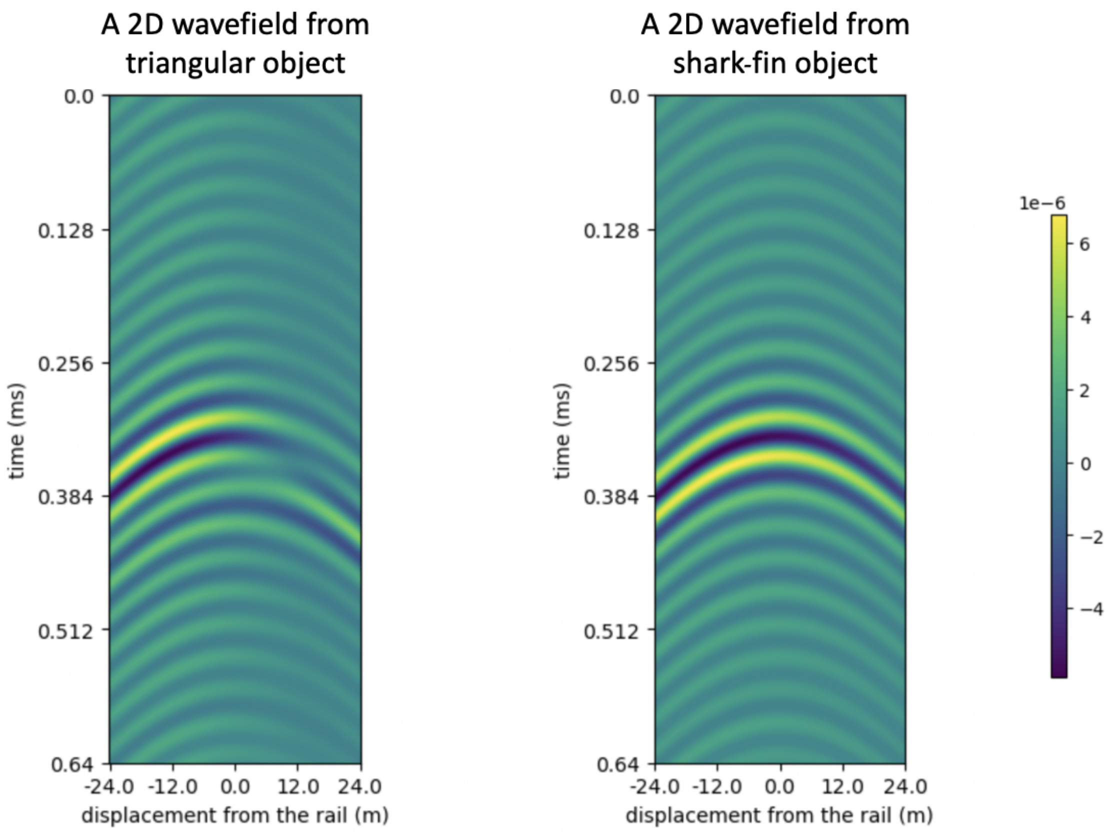


FIGURE 6.11. The generated wavefields of the triangular and shark-fin objects using the dolphin signal as a source.

where $g_{1,j}(\boldsymbol{x})$, $g^{(1)}_{1,j}(\boldsymbol{x})$ and $g^{(2)}_{1,j}(\boldsymbol{x})$ are the isotropic, the Riesz-x and Riesz-y components respectively, as shown in Figure 6.12. For each monogenic signal $g^{+}_{1,j}$, we can compute the monogenic amplitude, local phase, and local phase direction according to Equation (2.43). The contextual information is dominated by the arches bent from the horizontal line to a large extent based on the local phase and the phase direction.

6.4.4. Classification performance. We used AUC as the classification performance metric and repeated the experiments 10 times with two-fold cross validation. Again, the Julia package for GLMNet [**66**] was used as the classifier, with PCA [**59**] as the dimension reduction method. We set the PCA dimension to be 1000 at maximum for our MWT and MWSN features. The PCA dimension was determined by the PCA algorithm with the ratio of variances preserved in

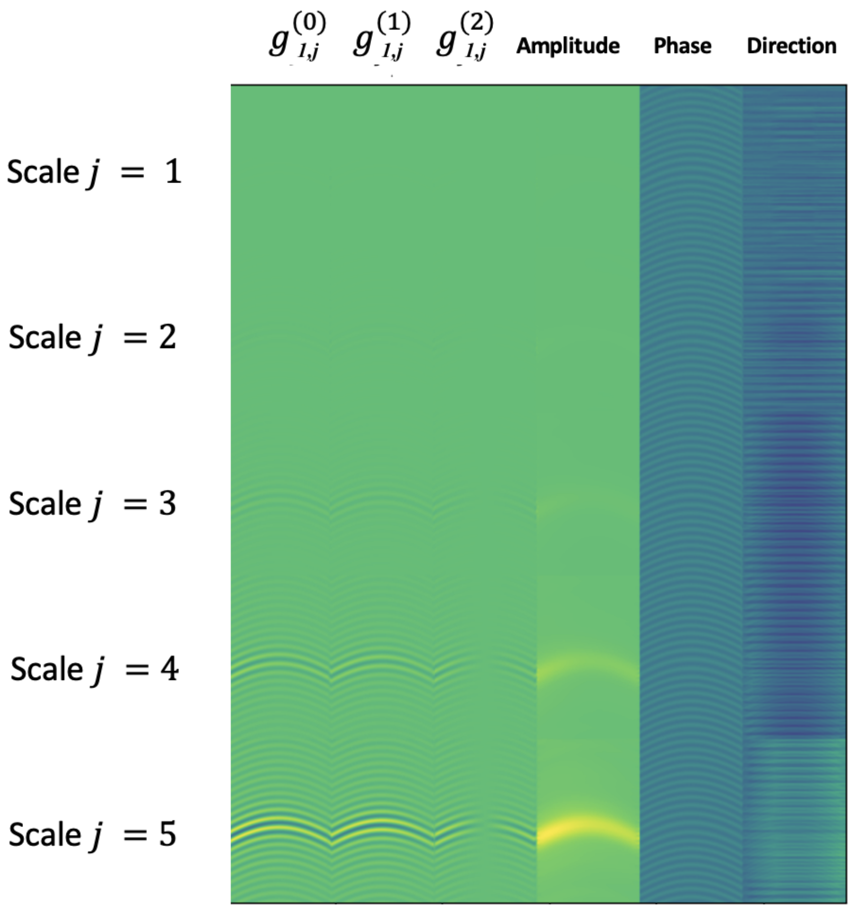


FIGURE 6.12. The multi-resolutional monogenic signal of a shark-fin object using the dolphin signal as the source.

the principal subspace (pratio) set to be 0.99. We compared the classification performance with various features fed into the the classifier:

(1) original wavefields
(2) Absolute value of the 2-D Fourier transform (AVFT)
(3) Monogenic wavelet transform (MWT)
(4) the 1st-layer MWSN (L1-MWSN)
(5) the 2nd-layer MWSN (L2-MWSN)
(6) AVFT of the 1st-layer MWSN (L1-AVFT-MWSN)
(7) AVFT of the 2nd-layer MWSN (L2-AVFT-MWSN).

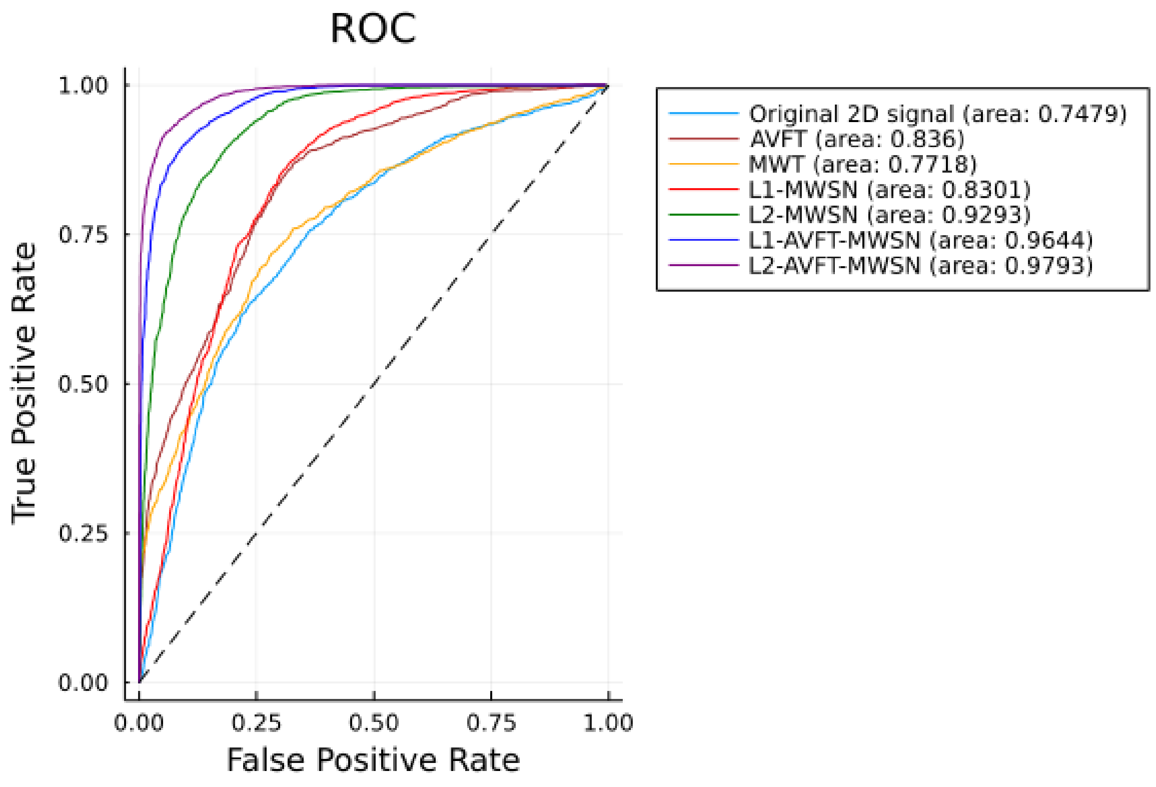


FIGURE 6.13. The AUC performance on the SAS dataset using the dolphin signal as the source.

Note that the MWSN features were generated under the subsampling rate $r_m = 3$ and $r'_m = 2$. Therefore, before applying PCA on the vectorized MWSN outputs, the length of the 1st-layer MWSN is round$(1281/3/2) = 214$. Then the length of the 2nd-layer MWSN is round$(1281/3/3/2) = 71$. Table 6.1 shows the output dimension of MWSN for each 2-D wavefield.

TABLE 6.1. Output dimension of MWSN for each wavefield.

Layer index	**Size / Length**
Input	1281
0	641
1	(214, 80, 12)
2	(71, 26, 12, 12)

Figure 6.13 demonstrates the ROC curves of the these features. The original 2-D wavefield had an AUC of 0.7479, which shows the discriminative power of using the dolphin signal as a source without additional features extraction. Also, the AUC of the 1st and 2nd-layer MWSN were

around 0.8301 and 0.9293 respectively, which were higher than using Gabor signal as a source. The AUC was around 0.9793 for L2-AVFT-MWSN, which was extracted by taking the absolute value of the Fourier transform (AVFT) of the MWSN in the 2nd-layer. Again, the L2-AVFT-MWSN features outperformed the other features in our experiments, with comparable performance using the dolphin and Gabor signals as sources. It suggests that our L2-AVFT-MWSN features together with the GLMNet classifier is capable of discriminating the two objects from their 2-D wavefields under various scenarios.

We then interpreted the performance purely from the 2nd-layer MWSN features. The features displayed in Figure 6.14 is the projection of the L2-MWSN space from the top principal components based on the GLMNet parameter $|\boldsymbol{\theta}|$. Each row index has the corresponding scale and the corresponding contextual direction with respect to the 1 (isotropic), $\mathbb{i}$ (time), and $\mathbb{j}$ (offset) components. We determined the PCA component index corresponding to the highest standard score of the parameter $|\boldsymbol{\theta}|$ from the L2-MWSN features for which the mean loss is minimized in GLMNet. Then we projected the corresponding PC component back to obtain a set of significant coefficients corresponding to the L2-MWSN. The most significant components correspond to the isotropic and the Riesz-x components (with respect to time) of larger scales. Taking the absolute value of the Fourier transform of L2-MWSN coefficients has improved the classification rates thanks to the additional translation invariance.

On the other hand, the L2-AVFT-MWSN features is hard to be visualized. After the 2-D Fourier transform, the most prominent Fourier coefficients are condensed into the left hand corner for each block in the ST features space. In order to visualize the score for L2-AVFT-MWSN, we again found the PCA component index corresponding to the highest standard score of the parameter $|\boldsymbol{\theta}|$ from the L2-AVFT-MWSN features for which the mean loss is minimized in GLMNet. Then we projected the corresponding PC component to obtain a set of significant coefficients corresponding to the L2-AVFT-MWSN. We took take the average of these significant coefficients in each block and then normalized over all blocks to generate in Figure 6.15. Based on Figure 6.15, we indicated the most significant components by the red box in Figure 6.16, which correspond to the isotropic and the Riesz-x components (corresponding to the time) and Riesz-y components (corresponding to the offset) of the largest scale from the 1st-layer of the MWSN.

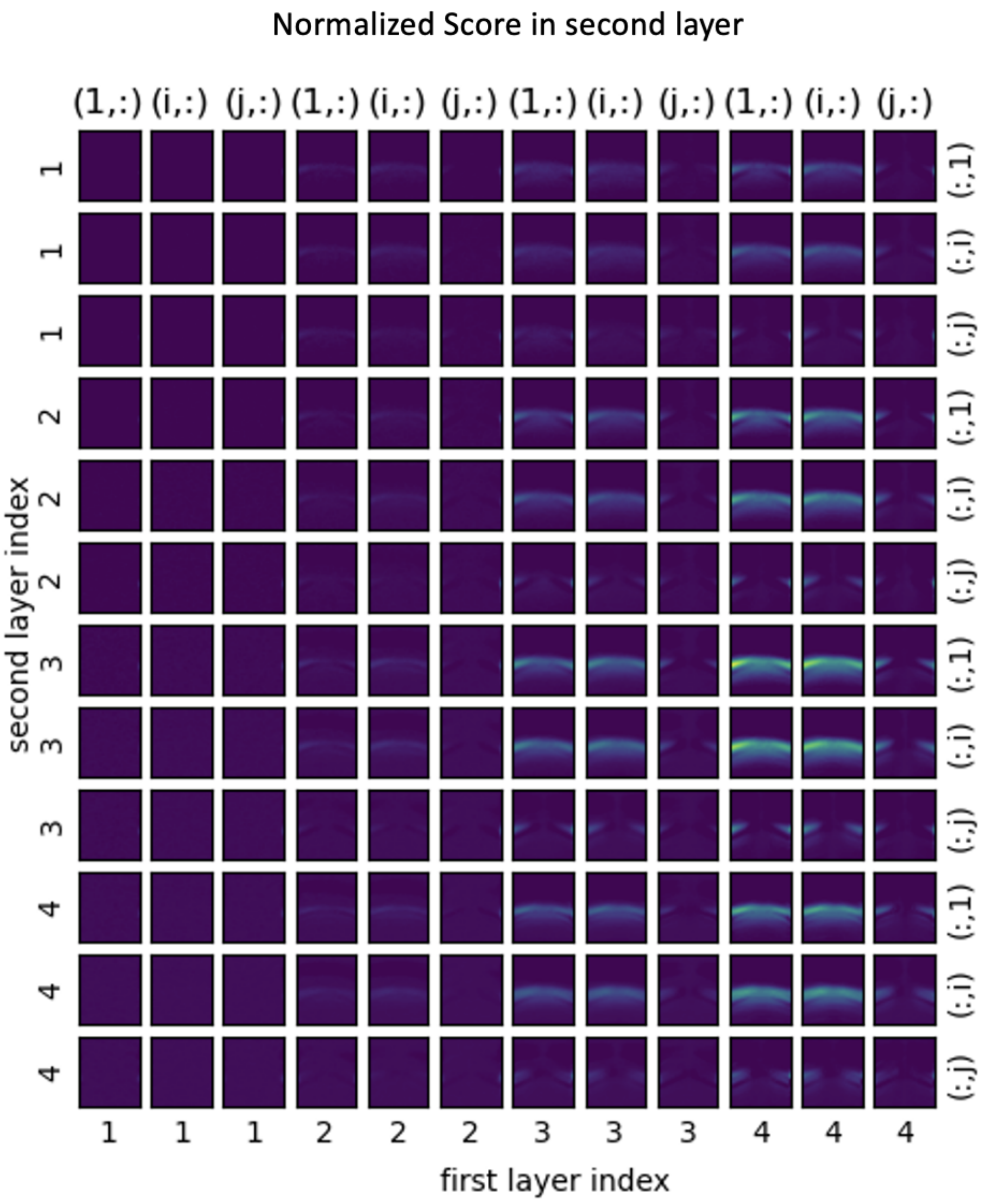


FIGURE 6.14. The normalized score in the 2nd-layer MWSN.

6.5. Object Material Classification Using the Dolphin Signal

In this subsection we seek to examine the effect of the change in the object property, in particular, the speed of sound. Besides the ability to discriminate object shapes, a good classifier should be also sensitive to the other object properties. Reflection coefficients as shown in Equation (6.1) and

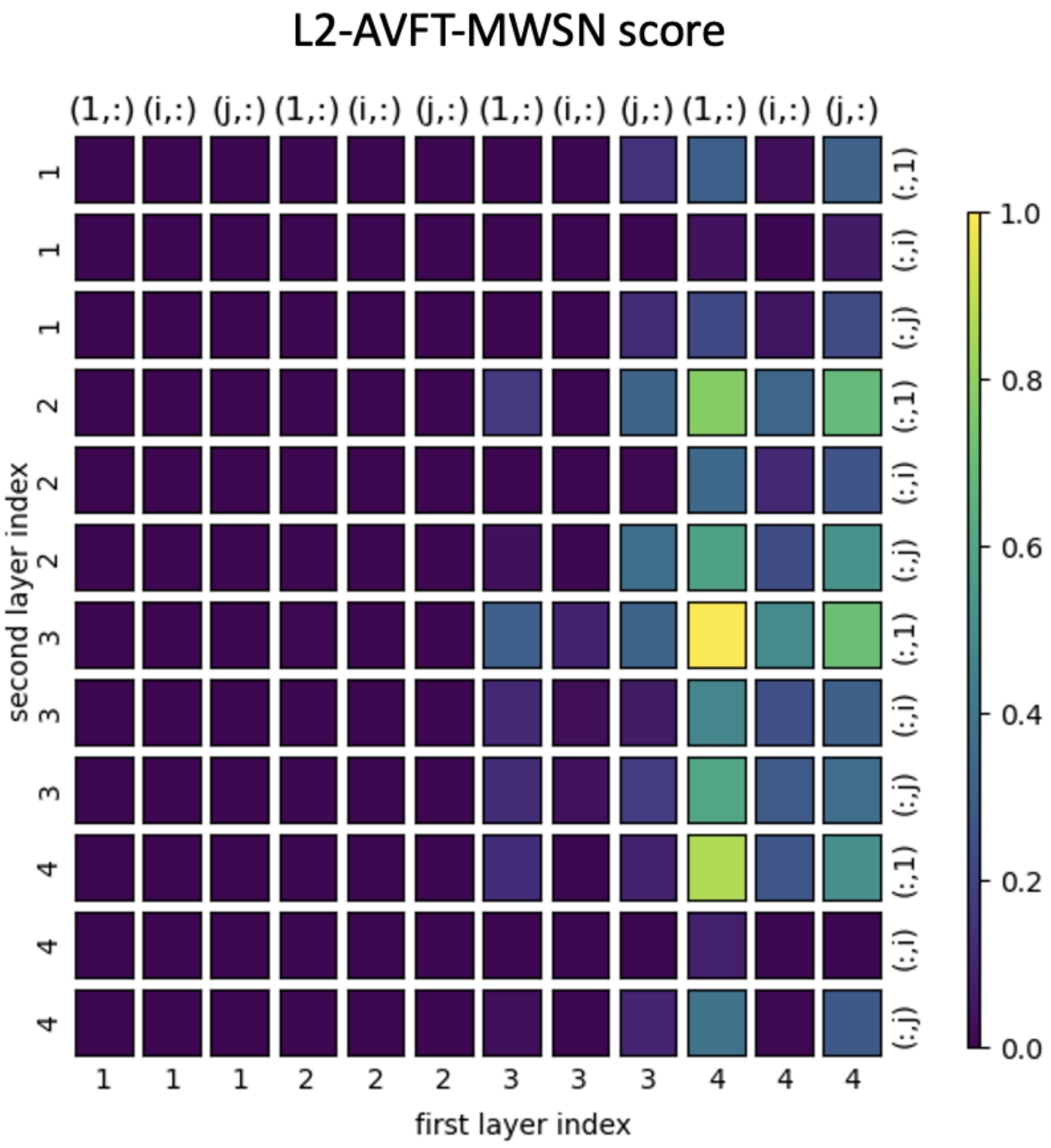


FIGURE 6.15. The normalized score in the L2-AVFT-MWSN with averaging.

the Snell's law [**106**, Section 3.3] can be used to determine the effect on the detected response from the change of the speed of sound. As stated in Reference [**106**, Section 3.3], the scale information should be a strong indicator of the effect of the speed of sound in the object material. As the absolute value of the 2-D Fourier transform (AVFT) has the access of the frequency information,

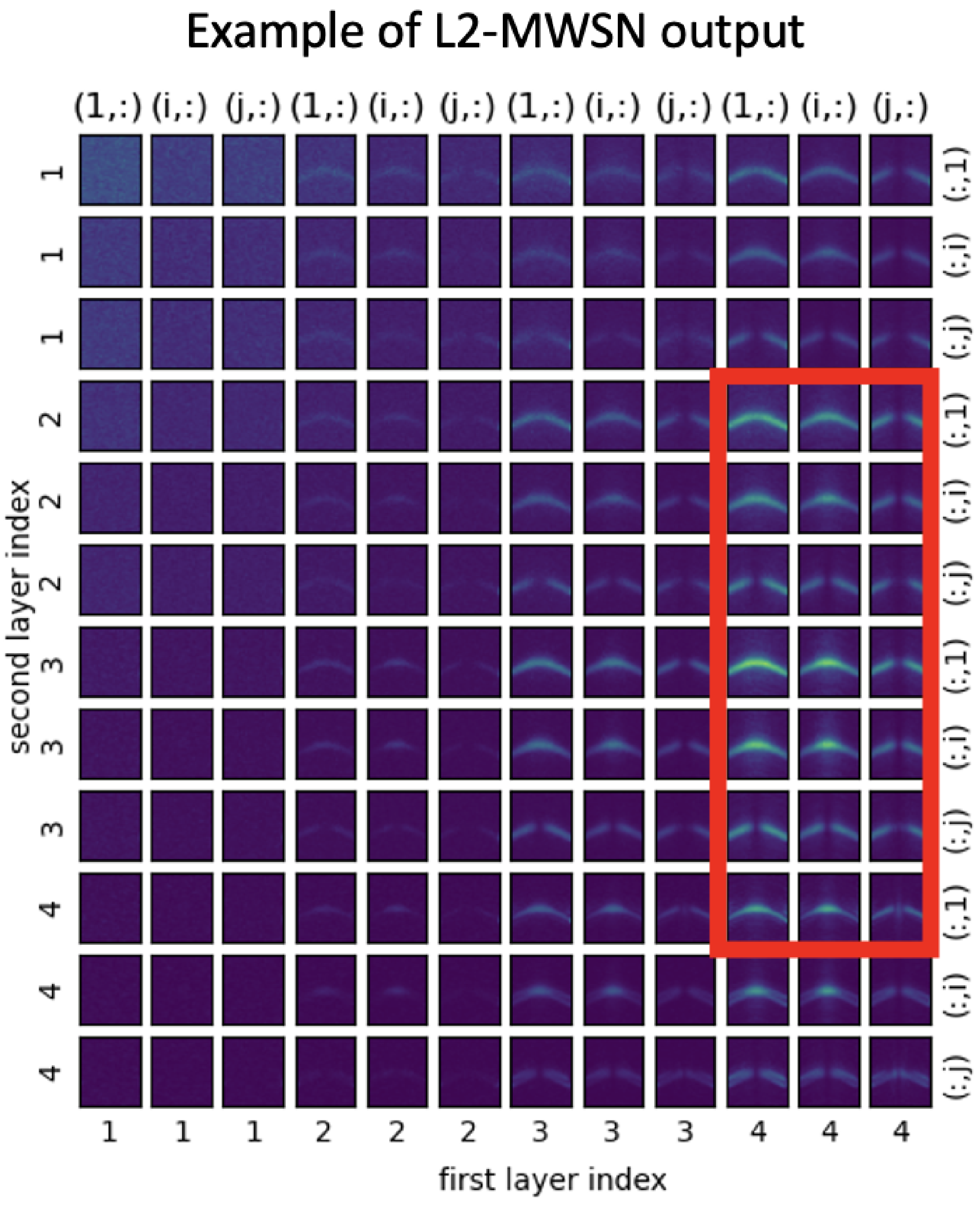


FIGURE 6.16. The L2-MWSN features, with the red box indicating the significant blocks. Each significant blocks are determined by the normalized score in the L2-AVFT-MWSN as shown in Figure 6.15.

we will show that it is a good indicator on the material differences. Below, we will also display the performance using the MWSN coefficients with different scales.

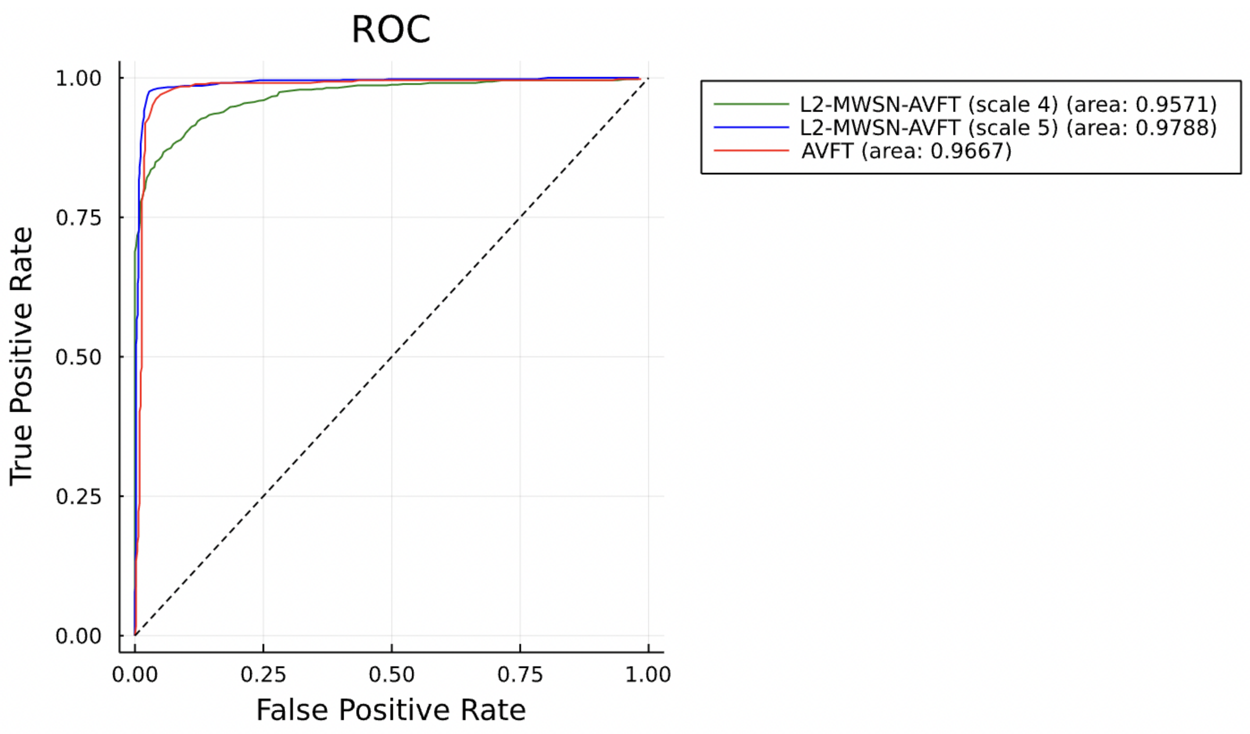


FIGURE 6.17. The AUC performance on the material discrimination in the triangular object on the SAS dataset using the dolphin signal as the source under the ROC curve.

We compared two different speed of sound, $c_1^{(1)} = 2000$ m/s and $c_1^{(2)} = 2500$ m/s, in the triangular object, using the dolphin signal as a source to generate the 2-D wavefields without the Hanning window. We again used the AUC metric to measure the classification performance between different materials (i.e., $c_1^{(1)}$ and $c_1^{(2)}$). We performed two-fold cross validation in 10 different experiments and computed the average AUC for three individual cases:

(1) Absolute value of the 2-D Fourier transform (AVFT)

(2) AVFT of the 2nd-layer MWSN (L2-AVFT-MWSN) with a maximum scale of 4

(3) AVFT of the 2nd-layer MWSN (L2-AVFT-MWSN) with a maximum scale of 5.

The L2-AVFT-MWSN features were selected as we showed that they could perform better than the L2-MWSN features. The PCA-compressed features of pratio = 0.99 and of dimension 1000 in each case are fed into the GLMNet classifier [**51**, Chap. 3]. The ROC curve which visualizes the classification result in each case is shown in Figure 6.17. The AVFT of the input 2-D wavefields performed very well with an AUC of 0.9667. The L2-AVFT-MWSN with a finer scale of 5 has an

AUC of 0.9788 which is better than its coarser version of $J = 4$. The frequency resolution is not sufficient enough with a scale of 4 in these experiments. However, the L2-AVFT-MWSN with a maximum scale of 5 is the most discriminative on the different speed of sound among all cases. The result using the MWSN coefficients aligns with the result in Reference [**106**].

CHAPTER 7

Conclusion

Throughout the course of the preceding chapters in this dissertation, we develop new scattering transform frameworks in reference to the mathematical concept of analyticity and monogenicity, and solve image and signal processing problems with machine learning tools. These new frameworks allow us to achieve better classification performance in various numerical experiments. Since the scattering coefficients and linear models such as GLMNet are interpretable, we can explain these experimental results, whereas interpretation from the conventional convolutional neural networks is extremely challenging.

We now summarize our contributions of this dissertation. In Chapter 3, the *Turbulence Removal Network* (TRN) was proposed for the first time to restore images from videos distorted by atmospheric turbulence using deep neural network in the face of data scarcity. To synthesize sufficient videos for data-hungry deep neural network, we introduced a new data augmentation algorithm to simultaneously distort the image with blurs and geometric distortion. The approach has shown to generalize atmospheric turbulence well by illustrating the performance in both synthetic and real cases. Considering that deep neural networks lack interpretability and there are some scenarios when training deep convolutional neural network (CNN) is an overkill, we can use the scattering transform as an alternative to CNN.

In Chapter 4, we created a new framework to incorporate the generalized Morse wavelet in the scattering transform network (GMW-STN) for the analysis of 1-D nonstationary signals. We demonstrated a better performance of the GMW-STN than Morlet-STN for music genres classification. The result can be explained by the analyticity of the wavelet as reviewed in Chapter 2. In addition, we showed that the accuracy in music genres classification is higher when we increase the number of layers in the scattering transform network. Above all, we provided the interpretation of the scattering transform coefficients computed from the music signals, whereas the conventional deep learning methods cannot.

In Chapter 5 we proposed and developed the novel framework of the 2-D scattering transform network. Motivated by the ideas of monogenicity as a natural extension of analyticity in higher dimension, as examined in Chapter 2, we developed the *Monogenic Wavelet Scattering Network* (MWSN). The new network together with the support vector machine (SVM) classifier exhibited a better accuracy in 2D texture image classification than the standard 2-D scattering transform network using Morlet wavelet with the SVM classifier. Our MWSN extracts interpretable coefficients, which can aid the explanation of these experimental results along with a linear machine learning model.

Lastly, the research on the scattering transform in object classification using sonar signals in Chapter 6, which is supported by the Office of Naval Research (ONR), is a promising direction. The dissertation extends the research [**106**] from Professor Saito's group by evaluating the performance in object classification using sonar signals and the new MWSN as proposed in Chapter 5. We have promising results on the synthetic aperture sonar (SAS) dataset using both Gabor functions and dolphin's clicks as source signals.

We conclude by exploring a number of open future research directions. First, the comparison between the MWSN applied to the spectrograms of the input music tracks and the GMW-STN applied to the input music tracks in Chapter as shown 4 raises new and interesting research. Second, we can enhance the interpretability of the MWSN coefficients by further leveraging the advantage of monogenic wavelet transform (MWT). We can convert the 2nd-layer coefficients into the instantaneous monogenic amplitude, local phase, and orientation as stated in Equation (2.43). We can explore different dimension reduction methods such as he Local Discriminant Basis (LDB) method [**104**, **105**] as the latter can extract discriminant features directly instead of gathering features of high variance by the PCA.

APPENDIX A

Multi-Detector Signal Extraction for Transabdominal Fetal Pulse Oximetry

This appendix highlights the work on multi-detector fetal signal extraction [**63**]. Existing signal extraction algorithms such as *REpeating Pattern Extraction Technique* (REPET) [**98**] and Adaptive REPET [**97**] identify the audio segments that are periodically repeating through spectrograms. Reference [**63**] used the *recursive least squares* (RLS) *adaptive noise cancellation* (ANC) filtering for fetal signal extraction. The work relates to the dissertation theme by the linear model (i.e., the RLS ANC filtering) used in the research and the advantage of multi-detector in detecting objects at difference distances from the emitters. See Subsection 3.3 and 6.1 for these respective references in the main content. Some materials in this appendix are extracted from Reference [**63**] with the dissertation author as a contributing author of the publication.

A.1. Problem overview

Through the use of ultrasound Doppler signal, we can track the fetal heart rate (FHR) throughout pregnancy for the assessment of antenatal fetal well-being [**37**]. To detect fetal hypoxia and reduce fetal mortality, uterine contraction followed by fetal bradycardia were suggested as protocol [**108**], even though electronic FHR monitoring adversely leads to an increased rate of emergency cesarean deliveries [**89**]. Some studies further indicated that there was no significant reduction of hazards to fetus due to fetal hypoxia in face of the rising number of cesarean deliveries [**2**, **85**]. Some challenges such as the interference of the maternal and fetal breathing reduces FHR detection accuracy using Doppler ultrasound [**62**]. Normal physiological responses can sometimes interrupt non-reassuring FHR monitoring [**2**]. A high false positive rate is expected by simply relying on FHR monitoring, hence causing inefficacies in surgical intervention by obstetricians. [**7**, **40**].

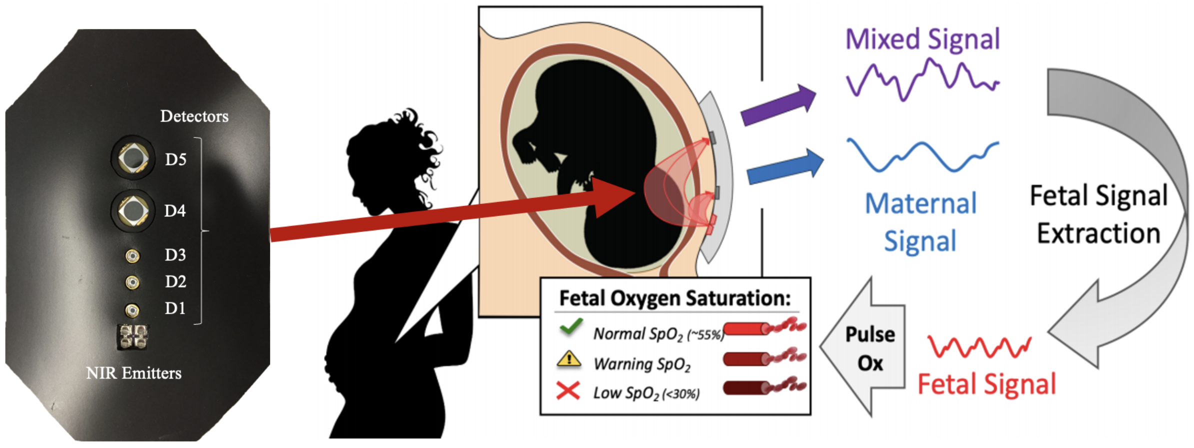


FIGURE A.1. The overview of transabdominal fetal pulse oximetry, with the optical probe for the emitters and photodetectors on the left.

Transabdominal fetal pulse oximetry (TFO) was recently developed as a non-invasive device to measure fetal oxygen saturation [**35**, **41**]. Figure A.1 shows the overview of the TFO [**34**]. In the reflectance-based TFO, the LED light sources in near-infrared (NIR) region emit photons that propagate through maternal abdomen and scatter through maternal tissue before reaching the fetus [**137**]. As fetal tissue is a few centimeters away from the maternal abdomen's surface, higher optical penetration depth are selected in the emitters from NIR region [**32**, **34**]. Then a portion of the photons traveling through fetal tissue are reflected back to the surface and captured by photodetectors [**32**]. See Figure A.1 for the optical probe on the left for the NIR emitters and five photodetectors (D1-D5) at various distances from the NIR emitters. As the distance between photodetector and NIR emitter increases, the detected light signal travels through a longer distance and hence penetrates deeper into the tissue.

When the detected signal travels through both the maternal and fetal medium, it carries a mixture of maternal and fetal information. In particular, the maternal heart rate (MHR), maternal respiration rate (MRR) and Mayer waves are included in maternal information [**36**]. The main challenge however is to extract the fetal signal from the mixed signal carrying most maternal information, so as to evaluate fetal oxygen saturation [**33**]. The closest detector (D1) to the emitter in

Figure A.1 captures maternal information only. Far photodetectors capture more fetal information as opposed to near photodetectors, but the captured signal is weaker due to the attenuation of the detected light penetrated through a longer distance. In Section 6.1, multiple sonar sensor locations on a rail also collect more object information than a single sensor.

A.2. Fetal signal extraction in multi-photodetector

We present a multi-detector fetal signal extraction method to recover weak fetal information gathered by TFO from the mixed signal which has both maternal and fetal information. Then we evaluate the proposed method on data gathered from a hypoxic fetal lamb in utero. Details on data acquisition from hypoxic lamb model can be found in Reference [**63**].

A.2.1. ANC in RLS.

RLS is an algorithm to recursively find coefficients that minimize a weighted linear least squares function and match the desired signal given a reference signal. We applied RLS for ANC. In particular, we treat the noise-only (maternal-only) signal as the reference signal, which is captured by the nearest photodetector D1 with 1.5 cm away from the emitter. The reference signal contains maternal heart rate (MHR) (1 Hz - 1.7 Hz / 60-100 breaths per minute (bpm)), maternal respiration rate (MRR) ($\sim$ 0.2 Hz - 0.5 Hz / 12-20 bpm) and Mayer waves ($\sim$ 0.1 Hz) [**36**]. On the other hand, the FHR is our desired signal which was then extracted from photodetectors D2 to D5 by ANC, together with signal preprocessing.

A finite impulse response (FIR) filter is a filter with a finite durational response to any input of finite length. To be precise, let $\boldsymbol{b} = (b_0, b_1, \cdots, b_N)$ be the N-th order FIR filter. Let $\boldsymbol{x}, \boldsymbol{y}$ be the input and output signals respectively. The output $\boldsymbol{y}$ can be expressed as a convolution of the input $\boldsymbol{x}$ with the most recent values and the vector $\boldsymbol{b}$ as the weight given by the FIR filter:

$$y_n = \sum_{k=0}^{N} b_k x_{n-k}, \tag{A.1}$$

where x_{n-k} is commonly known a the k-th tap of the input $\boldsymbol{x}$.

Suppose that $\boldsymbol{x}$ is the reference signal (i.e., maternal-only signal) and $\boldsymbol{d}$ is the desired signal (i.e., fetal-only signal). The goal of the algorithm is to recover the desired signal $\boldsymbol{d}$ from the reference input $\boldsymbol{x}$ using the FIR filter $\boldsymbol{w}$ with $p+1$ taps:

$$\begin{aligned} d_n &\approx \sum_{i=0}^{p} w_i x_{n-i} \\ &= \boldsymbol{w}^\intercal \boldsymbol{x}^{(n)}, \end{aligned} \tag{A.2}$$

where $\boldsymbol{x}^{(n)} = (x_n, x_{n-1}, \cdots, x_{n-p})$ is a vector containing the most recent values of $\boldsymbol{x}$. Denote the approximation of the recovered desired signal as

$$\begin{aligned} \tilde{d}_n &= \sum_{i=0}^{p} w_i^{(n)} x_{n-i} \\ &= \boldsymbol{w}^{(n)^\intercal} \boldsymbol{x}^{(n)}, \end{aligned} \tag{A.3}$$

where $\boldsymbol{w}^{(n)}$ is the current estimate of the filter $\boldsymbol{w}$.

A.2.2. Derivation of the RLS algorithm in algebraic equation. Let $\lambda \in (0,1]$ be the forgetting factor which weighs the importance of the most recent samples. Define the error $\boldsymbol{e}$ by

$$e_n = d_n - \tilde{d}_n \tag{A.4}$$

We then introduce a cost function C such that

$$\begin{aligned} C(\boldsymbol{w}^{(n)}) &= \sum_{k=0}^{n} \lambda^{n-k} e_k^2 \\ &= \sum_{k=0}^{n} \lambda^{n-k} |d_k - \tilde{d}_k|^2 \\ &= \sum_{k=0}^{n} \lambda^{n-k} |\boldsymbol{w}^\intercal \boldsymbol{x}^{(k)} - \boldsymbol{w}^{(k)^\intercal} \boldsymbol{x}^{(k)}|^2. \end{aligned} \tag{A.5}$$

By taking the partial derivative with respective to the recent coefficient vector $\boldsymbol{w}^{(n)}$ and setting it to be zero, one has, for $j = 0, 1, \cdots, p$,

$$
\begin{aligned}
0 = \frac{\partial C(w^{(n)})}{\partial w_j^{(n)}} &= \sum_{k=0}^{n} \lambda^{n-k} 2 e_k \frac{\partial e_k}{\partial w_j^{(n)}} \\
&= 2 \sum_{k=0}^{n} \lambda^{n-k} e_k x_{k-j} \\
&= 2 \sum_{k=0}^{n} \lambda^{n-k} \left(d_k - \sum_{i=0}^{p} w_i^{(k)} x_{k-i} \right) x_{k-j}.
\end{aligned}
\tag{A.6}
$$

Therefore,

$$
\sum_{k=0}^{n} \lambda^{n-k} d_k x_{k-j} = \sum_{k=0}^{n} \lambda^{n-k} \sum_{i=0}^{p} w_i^{(k)} x_{k-i} x_{k-j}. \tag{A.7}
$$

We denote $\boldsymbol{R}_x(n)$ be the weighted sample covariance matrix for $\boldsymbol{x}$. Also, we denote $\boldsymbol{r}_{dx}(n)$ as the equivalent estimate for the cross-covariance between $\boldsymbol{d}$ and $\boldsymbol{x}$. Then we can express the above equation by

$$
\begin{aligned}
\boldsymbol{R}_x(n) \boldsymbol{w}^{(n)} &= \boldsymbol{r}_{dx}(n) \\
\boldsymbol{w}^{(n)} &= \boldsymbol{R}_x^{-1}(n) \boldsymbol{r}_{dx}(n).
\end{aligned}
\tag{A.8}
$$

A.2.3. Derivation of the RLS Algorithm in Recursive Form. Let $\Delta \boldsymbol{w}^{(n-1)}$ is the correction factor at the time $n-1$. We will derive a solution of the form

$$
\boldsymbol{w}^{(n)} = \boldsymbol{w}^{(n-1)} + \Delta \boldsymbol{w}^{(n-1)}. \tag{A.9}
$$

Let $\boldsymbol{x}^{(j)} = (x_j, x_{j-1}, \cdots, x_{j-p})$. We write the cross covariance in terms of the recursive form:

$$
\begin{aligned}
\boldsymbol{r}_{dx}(n) &= \sum_{k=0}^{n} \lambda^{n-k} d_k \boldsymbol{x}^{(k)} \\
&= \sum_{k=0}^{n-1} \lambda^{n-k} d_k \boldsymbol{x}^{(k)} + d_n \boldsymbol{x}^{(n)} \\
&= \lambda \boldsymbol{r}_{dx}(n-1) + d_n \boldsymbol{x}^{(n)}.
\end{aligned}
\tag{A.10}
$$

Likewise, the sample covariance matrix can be written in terms of recursive form:

$$\begin{aligned}\boldsymbol{R}_x(n) &= \sum_{k=0}^{n} \lambda^{n-k} \boldsymbol{x}^{(k)} \boldsymbol{x}^{(k)^\mathsf{T}} \\ &= \lambda \boldsymbol{R}_x(n-1) + \boldsymbol{x}^{(n)} \boldsymbol{x}^{(n)^\mathsf{T}}.\end{aligned} \tag{A.11}$$

To find $\boldsymbol{R}_x^{-1}$ in Equation (A.8), we apply the Woodbury matrix identity [**47**]:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}. \tag{A.12}$$

Let $A := \lambda \boldsymbol{R}_x(n-1),\ U := \boldsymbol{x}^{(n)},\ C := I,\ V := \boldsymbol{x}^{(n)^\mathsf{T}}$. Then

$$\begin{aligned}\boldsymbol{R}_x^{-1}(n) &= \left(\lambda \boldsymbol{R}_x(n-1) + \boldsymbol{x}^{(n)} \boldsymbol{x}^{(n)^\mathsf{T}}\right)^{-1} \\ &= \lambda^{-1}\boldsymbol{R}_x(n-1)^{-1} - \lambda^{-1}\boldsymbol{R}_x(n-1)^{-1}\boldsymbol{x}^{(n)}(I + \boldsymbol{x}^{(n)^\mathsf{T}}\lambda^{-1}\boldsymbol{R}_x(n-1)^{-1}\boldsymbol{x}^{(n)})^{-1}\boldsymbol{x}^{(n)^\mathsf{T}}\lambda^{-1}\boldsymbol{R}_x(n-1)^{-1}.\end{aligned} \tag{A.13}$$

Define $\boldsymbol{P}(n)$ and $\boldsymbol{g}(n)$ by

$$\begin{aligned}\boldsymbol{P}(n) &:= \boldsymbol{R}_x^{-1}(n) \\ &= \lambda^{-1}\boldsymbol{P}(n-1) - \boldsymbol{g}(n)\boldsymbol{x}^{(n)^\mathsf{T}}\lambda^{-1}\boldsymbol{P}(n-1), \\ \boldsymbol{g}(n) &:= \lambda^{-1}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)}(I + \boldsymbol{x}^{(n)^\mathsf{T}}\lambda^{-1}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)})^{-1} \\ &= \boldsymbol{P}(n-1)\boldsymbol{x}^{(n)}(\lambda + \boldsymbol{x}^{(n)^\mathsf{T}}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)})^{-1}.\end{aligned} \tag{A.14}$$

We can further derive

$$\begin{aligned}\boldsymbol{g}(n) &= \boldsymbol{P}(n-1)\boldsymbol{x}^{(n)}(\lambda + \boldsymbol{x}^{(n)^\mathsf{T}}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)})^{-1} \\ \boldsymbol{g}(n)(\lambda + \boldsymbol{x}^{(n)^\mathsf{T}}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)}) &= \boldsymbol{P}(n-1)\boldsymbol{x}^{(n)} \\ \boldsymbol{g}(n)\lambda + \boldsymbol{g}(n)\boldsymbol{x}^{(n)^\mathsf{T}}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)} &= \boldsymbol{P}(n-1)\boldsymbol{x}^{(n)} \\ \boldsymbol{g}(n)\lambda &= \boldsymbol{P}(n-1)\boldsymbol{x}^{(n)} - \boldsymbol{g}(n)\boldsymbol{x}^{(n)^\mathsf{T}}\boldsymbol{P}(n-1)\boldsymbol{x}^{(n)} \\ \boldsymbol{g}(n) &= \lambda^{-1}[\boldsymbol{P}(n-1) - \boldsymbol{g}(n)\boldsymbol{x}^{(n)^\mathsf{T}}\boldsymbol{P}(n-1)]\boldsymbol{x}^{(n)} \\ &= \boldsymbol{P}(n)\boldsymbol{x}^{(n)}.\end{aligned} \tag{A.15}$$

Then,

$$
\begin{aligned}
\boldsymbol{w}^{(n)} &= \boldsymbol{P}(n)\boldsymbol{r}_{dx}(n) \\
&= \lambda \boldsymbol{P}(n)\boldsymbol{r}_{dx}(n-1) + d_n \boldsymbol{P}(n)\boldsymbol{x}^{(n)} \\
&= \lambda[\lambda^{-1}\boldsymbol{P}(n-1) - \boldsymbol{g}(n)\boldsymbol{x}^{(n)^\top}\lambda^{-1}\boldsymbol{P}(n-1)]\boldsymbol{r}_{dx}(n-1) + d_n\boldsymbol{g}(n) \qquad \text{(A.16)} \\
&= \boldsymbol{P}(n-1)\boldsymbol{r}_{dx}(n-1) - \boldsymbol{g}(n)\boldsymbol{x}^{(n)^\top}\boldsymbol{P}(n-1)\boldsymbol{r}_{dx}(n-1) + d_n\boldsymbol{g}(n) \\
&= \boldsymbol{P}(n-1)\boldsymbol{r}_{dx}(n-1) + \boldsymbol{g}(n)[d_n - \boldsymbol{x}^{(n)^\top}\boldsymbol{P}(n-1)\boldsymbol{r}_{dx}(n-1)].
\end{aligned}
$$

Therefore,

$$\boldsymbol{w}^{(n)} = \boldsymbol{w}^{(n-1)} + \boldsymbol{g}(n)\Big[d_n - \boldsymbol{x}^{(n)^\top}\boldsymbol{w}^{(n-1)}\Big]. \qquad \text{(A.17)}$$

Let $\alpha_n = d_n - \boldsymbol{x}^{(n)^\top}\boldsymbol{w}^{(n-1)}$ be a prior error. Then

$$\boldsymbol{w}^{(n)} = \boldsymbol{w}^{(n-1)} + \boldsymbol{g}(n)\alpha_n. \qquad \text{(A.18)}$$

A.2.4. Summary of the RLS Algorithm. The details of the RLS algorithm for the p-th order RLS filter are shown as follows:

1. Parameters:
   - (i) p: the order of the RLS filter
   - (ii) λ: the forgetting factor
   - (iii) δ: the value for initialization of $\boldsymbol{P}(0)$
2. Initialization:
   - (i) $\boldsymbol{w}(n) = 0$
   - (ii) $\boldsymbol{x}(k) = 0$ for $k = -p, \cdots, -1$
   - (iii) $\boldsymbol{d}(k) = 0$ for $k = -p, \cdots, -1$
   - (iv) $\boldsymbol{P}(0) = \delta I_{p+1}$

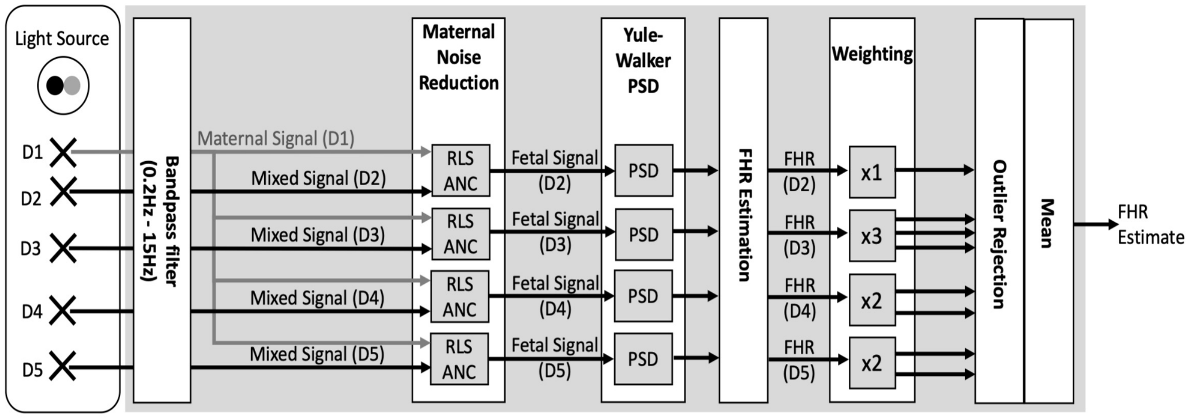


FIGURE A.2. The Multi-Photodetector Fetal Heart Rate Extractor.

3. Procedure:

   For $n = 1, 2, \cdots$, we define

$$
\begin{aligned}
\boldsymbol{x}(n) &= (x_n, x_{n-1}, \cdots, x_{n-p}) \\
\alpha(n) &= d(n) - \boldsymbol{x}^{\mathsf{T}}(n)\boldsymbol{w}(n-1) \\
\boldsymbol{g}(n) &= \frac{\boldsymbol{P}(n-1)\boldsymbol{x}(n)}{\lambda + \boldsymbol{x}^{\mathsf{T}}(n)\boldsymbol{P}(n-1)\boldsymbol{x}(n)} \\
\boldsymbol{P}(n) &= \lambda^{-1}\Big(\boldsymbol{P}(n-1) - \boldsymbol{g}(n)\boldsymbol{x}^{\mathsf{T}}(n)\boldsymbol{P}(n-1)\Big) \\
\boldsymbol{w}(n) &= \boldsymbol{w}(n-1) + \alpha(n)\boldsymbol{g}(n).
\end{aligned}
\tag{A.19}
$$

A.2.5. ANC in multi-photodetector system. As described in the problem overview A.1, we have a multi-photodetector system where photodetectors D2 to D5 capture fetal signal. Detectors D2 to D5 are 3, 4.5, 7, and 10 centimeters respectively away from the NIR emitters [**35**], generating 4 different mixed signals which are then processed by the ANC algorithm (with $\lambda = 0.99,\ p = 100$) for fetal signal extraction. As opposed to the existing single-detector FHR extraction approach [**39**], we propose the more robust multi-photodetector FHR extraction approach that ultilizes the extracted FHRs from four photodetectors. A diagram of the multi-photodetector FHR extractor implemented in MATLAB is presented in Figure A.2.

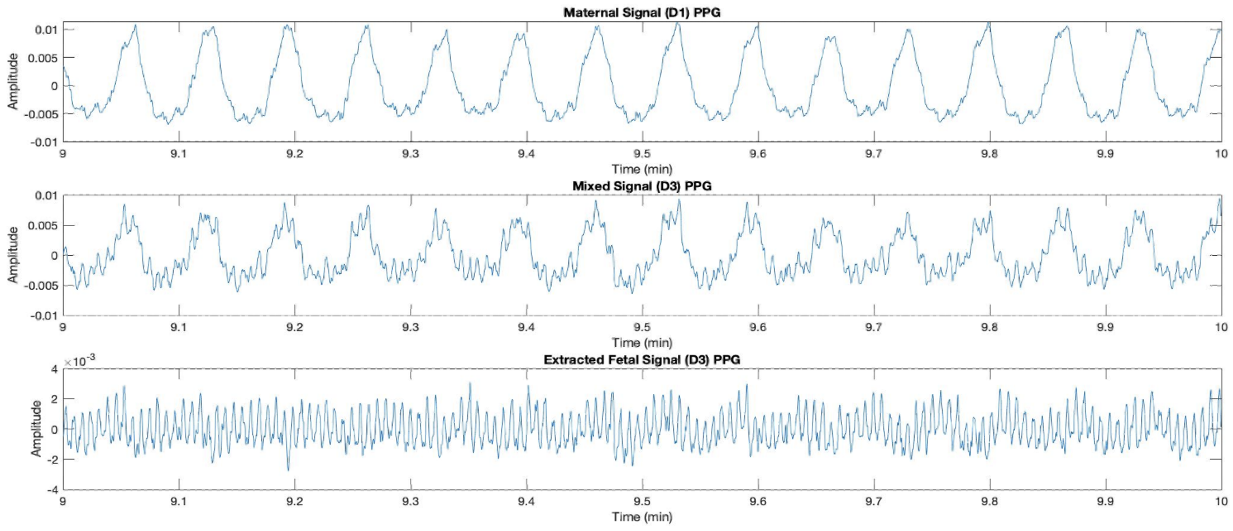


FIGURE A.3. Example of material, mixed and extracted fetal signals in the time domain from round 1.

To remove noise components from high frequency and very low frequency before applying ANC, we used bandpass filters between 0.2 Hz - 15 Hz on the the maternal signal captured by photodetector D1, and mixed signals captured by photodetectors D2 to D5. We remove MRR and MHR from the mixed signal by the RLS adaptive filtering algorithm so that the resulting signal mainly contains the fetal information. The power spectral density (PSD) of the resulting signal was then calculated by Yule-Walker autoregressive method [**64**] of order 100. Then the PSD passed to FHR estimation block which recorded the frequency with highest PSD within a pre-defined search span of 110 bpm - 270 bpm (1.83 Hz - 4.5 Hz) on FHR in reference to the FHRs found in the literature [**125**]. We assigned weight on each photodetector based on the source-detector distance and the prior information on the photodetectors as described in the overview A.1. The weights of D2, D3, D4, D5 are 1, 3, 2, 2 respectively. We reject the FHR estimations which are 3 Median Absolute Deviation (MAD) away from the weighted median. Eventually, the mean FHR is computed from the remaining weighted FHR estimates.

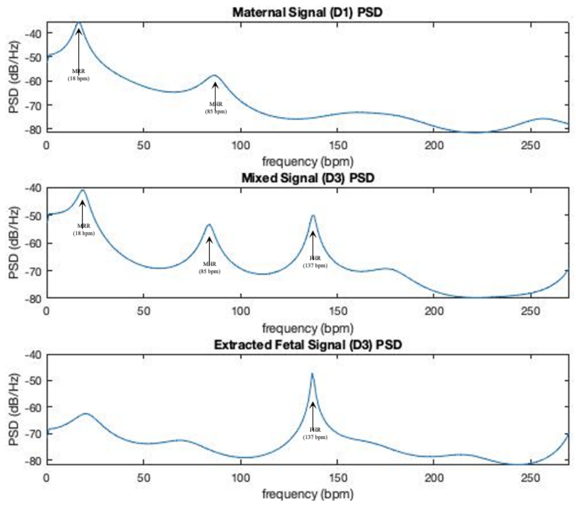


Figure A.4. Power-Spectral Densities of respective signals in Figure A.3.

A.3. Results of fetal heart rate estimation

In the hypoxic lamb model, we recorded for 53 and 31 minutes in round 1 and 2 respectively. We split the dataset into 1-minute long windows with 30-second window for overlapping datapoints. Hence, we computed a new FHR for every 30 seconds, and estimated the FHR for a 1-minute timeframe. Figure A.3 displays a maternal signal at photodetector D1, a mixed-signal captured at photodetector D3 and the extracted fetal signal after ANC from round 1 in the time-domain. Figure A.4 shows the corresponding PSD. The three peaks (from left to right) in Figure A.4 are maternal respiration rate (MRR), maternal heart rate (MHR) and fetal heart rate (FHR). The figures show effective cancellation of MRR and MHR to give a significant FHR peak. The mixed signal has similar patterns with the maternal-only measurement which leads to better ANC performance.

Figure A.5 shows the FHR estimations from all four photodetectors D2-D5 in the whole duration from the 1st round, with the mean FHR computed in Table A.1. The soar in FHR towards the end in Figure A.5 is due to fetal hypoxia. Root-mean-square error (RMSE) was the performance metric between estimated FHRs and reference FHRs measured by hemodynamics. The performance in photodetector D3 was the best among all photodetectors in Table A.1.

We display he FHR estimations with noisier data collected in 2nd round in Figure A.6, with a performance summary in Table A.2. We see that the performance is consistently better measured

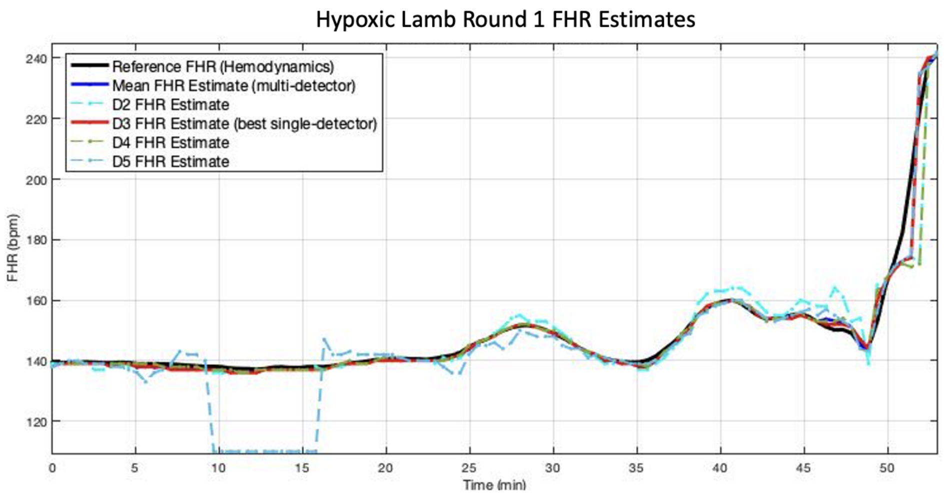


Figure A.5. Hypoxic lamb FHR estimates in the 1st round.

	RMSE (bpm)	Minimum Absolute Error (bpm)	Maximum Absolute Error (bpm)
D2 FHR	6.4381	0.0079	49.6693
D3 FHR	**3.2584**	**0.0035**	**27.5689**
D4 FHR	6.0331	0.0035	50.8693
D5 FHR	10.4637	0.0039	28.0960
mean FHR	**3.2457**	**0.0035**	**27.2355**

Table A.1. Summary of hypoxic lamb FHR estimates in the 1st round.

	RMSE (bpm)	Minimum Absolute Error (bpm)	Maximum Absolute Error (bpm)
D2 FHR	43.4845	4.5531	100.8720
D3 FHR	**4.7262**	**0.0427**	**24.2303**
D4 FHR	23.5761	0.0324	104.8720
D5 FHR	10.2379	0.2765	33.4131
mean FHR	**3.8492**	**0.0221**	**10.2075**

Table A.2. Summary of hypoxic lamb FHR estimates in the 2nd round.

by mean FHR as we can gather more information from all four photodetectors. These results illustrate that multi-detector FHR extraction approach provides more robust and more accurate fetal signal information as opposed to single-detector. Our method is superior to the latter one because the multi-detector can compensate for data loss and noise found in a single detector.

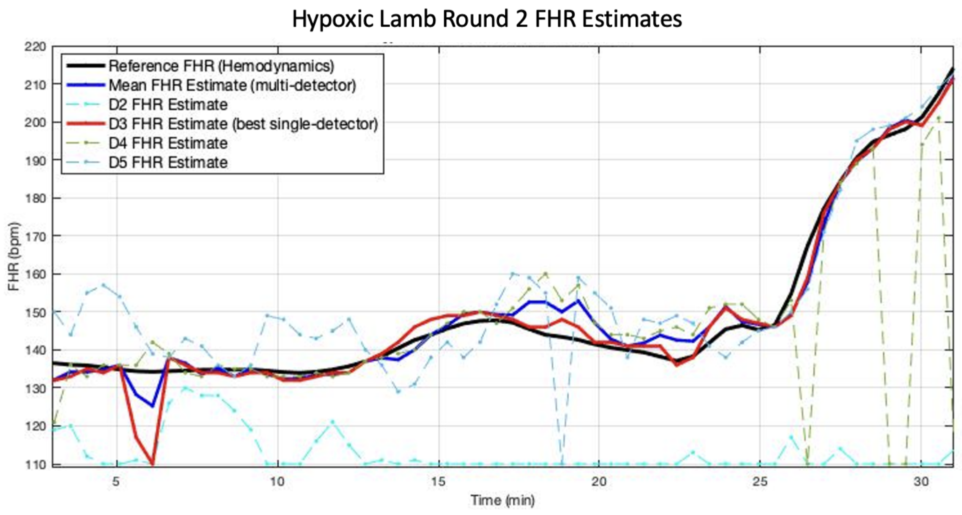


FIGURE A.6. Hypoxic lamb FHR estimates in the 2nd round.

A.4. 1.5-D scattering transform and classification performance in SAS

We compared the classification performance between 1.5-D scattering transform and the 1-D scattering transform using the Morlet Wavelet. Here 1.5D scattering transform refers to the incorporation of the 2-D monogenic feature into the 1-D signals which were then fed into a 1-D scattering transform network.

We generated the 2-D wavefield using dolphin signal as a source. See Section 6.4.2 for more details. In Section , we showed the monogenic signal from the 2-D wavefield. We set the scale j of MWT to be 6. Then a subset of the MWT components were stacked to be ST features:

(1) Isotropic component of scale 4
(2) Isotropic component of scale 5
(3) Riesz component (space) of scale 5
(4) Isotropic component of scale 6
(5) Riesz component (space) of scale 6

As we seen from the score analysis in Section 6.4.4, features with higher scales contributed to a higher success in classification. In particular, the isotropic texture and the texture along space axis of higher scales were more significant in sonar classification. Then we processed these features by 1-D scattering transform network using Morlet wavelet.

We used AUC as the classification performance metric and repeated the experiments 10 times with two-fold cross validation. Again, the Julia package for GLMNet [**66**] was used as the classifier, with PCA [**59**] as the dimension reduction method. We set the PCA dimension to be 1500 at maximum for our MWT and MWSN features. The PCA dimension was determined by the PCA algorithm with the ratio of variances preserved in the principal subspace (pratio) set to be 0.99. We compared the classification performance with various features fed into the the classifier:

(1) the 2nd layer MWSN (L2-MWSN) using all MWT features

(2) the 2nd layer MWSN (L2-MWSN) using a subset of MWT features

The reported AUC using all MWT features was 0.9314, while the AUC using a subset of MWT features was 0.9512. Hence, we concluded that selecting a subset of MWT features further improved the classification performance in 1-D scattering transform.

Bibliography

[1] A. F. Agarap, *Deep learning using rectified linear units (RELU)*, arXiv preprint arXiv:1803.08375, (2018).

[2] Z. Alfirevic, G. M. Gyte, A. Cuthbert, and D. Devane, *Continuous cardiotocography (CTG) as a form of electronic fetal monitoring (EFM) for fetal assessment during labour*, Cochrane Database of Systematic Reviews, (2017).

[3] S. Allamy and A. L. Koerich, *1D CNN Architectures for Music Genre Classification*, in 2021 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2021, pp. 01–07.

[4] S.-i. Amari, *Backpropagation and stochastic gradient descent method*, Neurocomputing, 5 (1993), pp. 185–196.

[5] M. Andreux, T. Angles, G. Exarchakis, R. Leonarduzzi, G. Rochette, L. Thiry, J. Zarka, S. Mallat, J. Andén, E. Belilovsky, et al., *Kymatio: Scattering transforms in Python.*, J. Mach. Learn. Res., 21 (2020), pp. 1–6.

[6] Arjovsky, Martin and Chintala, Soumith and Bottou, Léon, *Wasserstein generative adversarial networks*, International Conference on Machine Learning, (2017), pp. 214–223.

[7] K. C. Arnold and C. J. Flint, *Intrapartum fetal heart rate monitoring: nomenclature, interpretation, and general management principles*, in Obstetrics Essentials, Springer, 2017, pp. 101–107.

[8] W. W. Au, *Hearing in whales and dolphins: An overview*, Hearing by whales and dolphins, (2000), pp. 1–42.

[9] W. W. Au and D. L. Herzing, *Echolocation signals of wild atlantic spotted dolphin (Stenella frontalis)*, Journal of the Acoustical Society of America, 113 (2003), pp. 598–604.

[10] R. Bellman, *Dynamic Programming*, Princeton University Press, (1957).

[11] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, *Julia: A fresh approach to numerical computing*, SIAM Review, 59 (2017), pp. 65–98.

[12] V. Bodin, *Waveform creation and analysis*, rapport de stage de recherche, UC Davis & Ecole Polytechnique, 2013.

[13] J. Bremer, *A fast direct solver for the integral equations of scattering theory on planar curves with corners*, Journal of Computational Physics, 231 (2012), pp. 1879–1899.

[14] J. Bremer and Z. Gimbutas, *On the numerical evaluation of the singular integrals of scattering theory*, Journal of Computational Physics, 251 (2013), pp. 327–343.

[15] J. Bremer, V. Rokhlin, and I. Sammis, *Universal quadratures for boundary integral equations on two-dimensional domains with corners*, Journal of Computational Physics, 229 (2010), pp. 8259–8280.

[16] J.-S. Brittain, D. M. Halliday, B. A. Conway, and J. B. Nielsen, *Single-trial multiwavelet coherence in application to neurophysiological time series*, IEEE Transactions on Biomedical Engineering, 54 (2007), pp. 854–862.

[17] J. Bruna and S. Mallat, *Classification with scattering operators*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2011, IEEE, 2011, pp. 1561–1566.

[18] ———, *Invariant scattering convolution networks*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (2013), pp. 1872–1886.

[19] P. Cerejeiras and U. Kähler, *Monogenic Signal Theory*, In: Alpay D.(eds) Operator Theory. Springer, Basel., 2014.

[20] W. H. Chak, C. P. Lau, and L. M. Lui, *Subsampled turbulence removal network*, Mathematics, Computation and Geometry of Data, 1 (2021), pp. 1–33.

[21] W. H. Chak and N. Saito, *Monogenic wavelet scattering network for texture image classification*, JSIAM Letters, 15 (2023), pp. 21–24.

[22] W. H. Chak, N. Saito, and D. Weber, *The Scattering Transform Network with Generalized Morse Wavelets and its Application to Music Genre Classification*, in 2022 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR), IEEE, 2022, pp. 25–30.

[23] C.-C. Chang and C.-J. Lin, *LIBSVM: A library for support vector machines*, ACM Transactions on Intelligent Systems and Technology, 2 (2011), pp. 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[24] K. Choi, G. Fazekas, M. Sandler, and K. Cho, *Convolutional recurrent neural networks for music classification*, in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2017, pp. 2392–2396.

[25] Y. M. Costa, L. S. Oliveira, and C. N. Silla Jr, *An evaluation of convolutional neural networks for music classification using spectrograms*, Applied Soft Computing, 52 (2017), pp. 28–38.

[26] K. J. Dana, B. Van Ginneken, S. K. Nayar, and J. J. Koenderink, *Reflectance and texture of real-world surfaces*, ACM Transactions On Graphics (TOG), 18 (1999), pp. 1–34.

[27] N. Delprat, B. Escudié, P. Guillemain, R. Kronland-Martinet, P. Tchamitchian, and B. Torresani, *Asymptotic wavelet and gabor analysis: Extraction of instantaneous frequencies*, IEEE Transactions on Information Theory, 38 (1992), pp. 644–664.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, *ImageNet: A large-scale hierarchical image database*, in 2009 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2009, pp. 248–255.

[29] T. Fawcett, *An introduction to ROC analysis*, Pattern Recognition Letter, 27 (2006), pp. 861–874.

[30] M. Felsberg, *Low-level image processing with the structure multivector*, vol. 203, Inst. für Informatik und Praktische Mathematik, 2002.

[31] M. Felsberg and G. Sommer, *The monogenic signal*, IEEE Transactions on Signal Processing, 49 (2001), pp. 3136–3144.

[32] D. Fong, A. Knoesen, and S. Ghiasi, *Transabdominal fetal pulse oximetry: The case of fetal signal optimization*, in 2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom), IEEE, 2017, pp. 1–6.

[33] D. D. Fong, A. Knoesen, M. Motamedi, T. O'Neill, and S. Ghiasi, *Recovering the fetal signal in transabdominal fetal pulse oximetry*, Smart Health, 9 (2018), pp. 23–36.

[34] D. D. Fong, V. J. Srinivasan, K. Vali, and S. Ghiasi, *Optode design space exploration for clinically-robust non-invasive fetal oximetry*, ACM Transactions on Embedded Computing Systems (TECS), 18 (2019), pp. 1–22.

[35] D. D. Fong, K. Vali, and S. Ghiasi, *Contextually-aware fetal sensing in transabdominal fetal pulse oximetry*, in 2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS), IEEE, 2020, pp. 119–128.

[36] D. D. Fong, K. J. Yamashiro, K. Vali, L. A. Galganski, J. Thies, R. Moeinzadeh, C. Pivetti, A. Knoesen, V. J. Srinivasan, H. L. Hedriana, et al., *Design and In Vivo Evaluation of a Non-Invasive Transabdominal Fetal Pulse Oximeter*, IEEE Transactions on Biomedical Engineering, 68 (2020), pp. 256–266.

[37] R. K. Freeman, T. J. Garite, M. P. Nageotte, and L. A. Miller, *Fetal heart rate monitoring*, Lippincott Williams & Wilkins, 2012.

[38] C. Frye, D. de Mijolla, T. Begley, L. Cowton, M. Stanley, and I. Feige, *Shapley explainability on the data manifold*, arXiv preprint arXiv:2006.01272, (2020).

[39] K. B. Gan, E. Zahedi, and M. A. M. Ali, *Transabdominal fetal heart rate detection using NIR photopleythysmography: instrumentation and clinical results*, IEEE transactions on Biomedical Engineering, 56 (2009), pp. 2075–2082.

[40] T. J. Garite, *The search for an adequate back-up test for intrapartum fetal heart rate monitoring*, American Journal of Obstetrics & Gynecology, 208 (2013), pp. 163–164.

[41] S. Ghiasi and D. Fong, *Robust, clinical-grade transabdominal fetal pulse oximetry*, Sept. 14 2021. US Patent 11,116,412.

[42] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[43] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.

[44] B. Goodman and S. Flaxman, *European union regulations on algorithmic decision-making and a "right to explanation"*, AI Magazine, 38 (2017), pp. 50–57.

[45] P. Grohs, T. Wiatowski, and H. Bölcskei, *Deep convolutional neural networks on cartoon functions*, in 2016 IEEE International Symposium on Information Theory (ISIT), IEEE, 2016, pp. 1163–1167.

[46] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, *Improved training of wasserstein GANs*, in Advances in Neural Information Processing Systems, 2017, pp. 5769–5779.

[47] W. W. Hager, *Updating the inverse of a matrix*, SIAM Review, 31 (1989), pp. 221–239.

[48] S. L. Hahn, *Multidimensional complex signals with single-orthant spectra*, Proceedings of the IEEE, 80 (1992), pp. 1287–1300.

[49] S. L. Hahn, *Hilbert Transforms in Signal Processing*, Artech House Inc, Boston, London, 1996.

[50] F. J. Harris, *On the use of windows for harmonic analysis with the discrete Fourier transform*, Proceedings of the IEEE, 66 (1978), pp. 51–83.

[51] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations*, vol. 143 of Monographs on Statistics and Applied Probability, CRC Press, Boca Raton, FL, 2015.

[52] S. Hegyi, *A powerful generalization of the NBD suggested by Peter Carruthers*, in Proceedings of the 8th International Workshop on Correlations and Fluctuations' 98. From QCD to Particle Interferometry, 1999.

[53] D. L. Herzing, *Acoustics and social behavior of wild dolphins: Implications for a sound society*, in Hearing by Whales and Dolphins, Springer, 2000, pp. 225–272.

[54] J. Howlett, *Handbook of Mathematical Functions. edited by milton abramowitz and irene a. stegun. constable (dover publications inc.) paperback edition 1965.*, The Mathematical Gazette, 50 (1966), pp. 358–359.

[55] F. J. Huang and Y. LeCun, *Large-scale learning with svm and convolutional for generic object categorization*, in 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 1, IEEE, 2006, pp. 284–291.

[56] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in International Conference on Machine Learning, PMLR, 2015, pp. 448–456.

[57] N. A. Jaleel and P. V. Kumar, *Implementation of an efficient FPGA architecture for capsule endoscopy processor core using hyper analytic wavelet-based image compression technique*, International Journal of Data Analysis Techniques and Strategies, 12 (2020), pp. 262–286.

[58] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, *What is the best multi-stage architecture for object recognition?*, in 2009 IEEE 12th International Conference on Computer Vision, IEEE, 2009, pp. 2146–2153.

[59] Julia, *MultivariateStats.jl: A Julia package for multivariate statistics and data analysis.* `https://github.com/JuliaStats/MultivariateStats.jl`, 2022.

[60] G. Kaiser and L. H. Hudgins, *A Friendly Guide to Wavelets*, vol. 300, Springer, 1994.

[61] S. Kargl, *Acoustic response of underwater munitions near a sediment interface: Measurement model comparisons and classification schemes*, Final Report MR-2231, Applied Physics Laboratory, Univ. of Washington, Seattle, 2015. SERDP Project.

[62] B. Karlsson, M. Berson, T. Helgason, R. Geirsson, and L. Pourcelot, *Effects of fetal and maternal breathing on the ultrasonic Doppler signal due to fetal heart movement*, European Journal of Ultrasound, 11 (2000), pp. 47–52.

[63] B. Kasap, K. Vali, W. Qian, W. H. Chak, A. Vafi, N. Saito, and S. Ghiasi, *Multi-detector heart rate extraction method for transabdominal fetal pulse oximetry*, in 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE, 2021, pp. 1072–1075.

[64] S. M. Kay, *Modern Spectral Estimation: Theory and Application*, Pearson Education India, 1988.

[65] D. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, in International Conference on Learning Representations (ICLR), San Diega, CA, USA, 2015.

[66] S. Kornblith, *Julia wrapper for fitting Lasso/ElasticNet GLM models using glmnet*, 2013. Julia package.

[67] S. Kornblith and M. Pastell, *LIBSVM.jl: A Julia interface for LIBSVM.* `https://github.com/JuliaML/LIBSVM.jl`, 2021.

[68] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet classification with deep convolutional neural networks*, Advances in Neural Information Processing Systems, 25 (2012), pp. 1097–1105.

[69] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, *DeblurGan: Blind Motion Deblurring Using Conditional Adversarial Networks*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[70] J. KÖLBEL and F. SEUBRING, *Dealing with UXO (Unexploded Ordnance): Detection, Identification, Disposal and Awareness*, Terra et Aqua, 141 (2015), p. 5.

[71] C. P. Lau, Y. H. Lai, and L. M. Lui, *Variational models for joint subsampling and reconstruction of turbulence-degraded images*, Journal of Scientific Computing, (2017), pp. 1–38.

[72] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444.

[73] Y. LeCun, C. Cortes, and C. Burges, *MNIST handwritten digit database*, ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist, (2010).

[74] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., *Photo-realistic single image super-resolution using a generative adversarial network*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[75] J. H. Lienhard and P. L. Meyer, *A physical basis for the generalized gamma distribution*, Quarterly of Applied Mathematics, 25 (1967), pp. 330–334.

[76] L. Lieu and N. Saito, *Signal ensemble classification using low-dimensional embeddings and earth mover's distance*, in Wavelets and Multiscale Analysis, Springer, 2011, pp. 227–256.

[77] J. Lilly, *jlab: A data analysis package for Matlab, v. 1.6.5*, See http://www. jmlilly. net/jmlsoft. html, (2017).

[78] J. M. Lilly and S. C. Olhede, *Higher-order properties of analytic wavelets*, IEEE Transactions on Signal Processing, 57(1) (2009), pp. 146–160.

[79] ———, *On the analytic wavelet transform*, IEEE Transactions on Information Theory, 56(8) (2010), pp. 4135–4156.

[80] ———, *Generalized Morse wavelets as a superfamily of analytic wavelets*, IEEE Transactions on Signal Processing, 60 (2012), pp. 6036–6041.

[81] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, Burlington, MA, third ed., 2009.

[82] S. Mallat, *Group invariant scattering*, Communications on Pure and Applied Mathematics, 65 (2012), pp. 1331–1398.

[83] B. Marchand, *Local Signal Analysis for Classification*, PhD thesis, Department of Mathematics, University of California, Davis, 2010.

[84] B. Marchand, N. Saito, and H. Xiao, *Classification of objects in synthetic aperture sonar images*, in Proceedings of the 14th Workshop on Statistical Signal Processing, IEEE, 2007, pp. 433–437.

[85] J. A. Martin, B. E. Hamilton, M. J. Osterman, and A. K. Driscoll, *Births: final data for 2018*, (2019).

[86] P. W. Moore, *Mine-hunting dolphins of the Navy*, in Detection and Remediation Technologies for Mines and Minelike Targets II, Proceedings of SPIE 3079, 1997, pp. 2–6.

[87] P. M. Morse, *Diatomic molecules according to the wave mechanics. II. vibrational levels*, Physical Review, 34 (1929), p. 57.

[88] V. Nair and G. E. Hinton, *Rectified linear units improve restricted Boltzmann machines*, in Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814.

[89] K. B. Nelson, T. P. Sartwelle, and D. J. Rouse, *Electronic fetal monitoring, cerebral palsy, and caesarean section: Assumptions versus evidence*, British Medical Journal Publishing Group, 355 (2016).

[90] S. Olhede and A. Walden, *Polarization phase relationships via multiple Morse wavelets. II. Data analysis*, Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, 459 (2003), pp. 641–657.

[91] S. C. Olhede and G. Metikas, *The monogenic wavelet transform*, IEEE Transactions on Signal Processing, 57 (2009), pp. 3426–3441.

[92] S. C. Olhede and A. T. Walden, *Generalized Morse wavelets*, IEEE Transactions on Signal Processing, 50 (2002), pp. 2661–2670.

[93] S. C. Olhede and A. T. Walden, *Noise reduction in directional signals using multiple Morse wavelets illustrated on quadrature doppler ultrasound*, IEEE Transactions on Biomedical Engineering, 50 (2003), pp. 51–57.

[94] R. Pascanu, T. Mikolov, and Y. Bengio, *On the difficulty of training recurrent neural networks*, in International Conference on Machine Learning, PMLR, 2013, pp. 1310–1318.

[95] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, *Automatic differentiation in PyTorch*, in NIPS-W, 2017.

[96] K. Pearson, *LIII. on lines and planes of closest fit to systems of points in space*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2 (1901), pp. 559–572.

[97] Z. Rafii, A. Liutkus, and B. Pardo, *REPET for background/foreground separation in audio*, in Blind Source Separation: Advances in Theory, Algorithms and Applications, Springer, 2014, pp. 395–411.

[98] Z. Rafii and B. Pardo, *Repeating pattern extraction technique (REPET): A simple method for music/voice separation*, IEEE Transactions on Audio, Speech, and Language Processing, 21 (2012), pp. 73–84.

[99] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun, *Unsupervised learning of invariant feature hierarchies with applications to object recognition*, in 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–8.

[100] I. Y. Ren, A. Volk, W. Swierstra, and R. Veltkamp, *Analysis by Classification: A Comparative Study of Annotated and Algorithmically Extracted Patterns in Symbolic Music Data*, in Proceedings of the 19th International Society for Music Information Retrieval Conference, Paris, France, Sept. 2018, ISMIR, pp. 539–546.

[101] O. Ronneberger, P. Fischer, and T. Brox, *U-net: Convolutional networks for biomedical image segmentation*, in International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, 2015, pp. 234–241.

[102] W. Rudin, *Functional Analysis 2nd ed., Mcgwaw-Hill*, 1991.

[103] N. C. Sagias, G. K. Karagiannidis, P. T. Mathiopoulos, and T. A. Tsiftsis, *On the performance analysis of equal-gain diversity receivers over generalized Gamma fading channels*, IEEE Transactions on Wireless Communications, 5 (2006), pp. 2967–2975.

[104] N. Saito and R. R. Coifman, *Local discriminant bases and their applications*, J. Mathematical Imaging and Vision, 5 (1995), pp. 337–358. Invited paper.

[105] N. Saito, R. R. Coifman, F. B. Geshwind, and F. Warner, *Discriminant feature extraction using empirical probability density estimation and a local basis library*, Pattern Recognition, 35 (2002), pp. 2841–2852.

[106] N. Saito and D. S. Weber, *Underwater object classification using scattering transform of sonar signals*, in Wavelets and Sparsity XVII, Proceedings of SPIE 10394, International Society for Optics and Photonics, 2017.

[107] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, *Improved techniques for training GANs*, in Advances in Neural Information Processing Systems, 2016, pp. 2234–2242.

[108] B. S. Schifrin, *Fetal heart rate monitoring during labor*, The Journal of the American Medical Association, 222 (1972), pp. 196–202.

[109] I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury, *The dual-tree complex wavelet transform*, IEEE Signal Processing Magazine, 22 (2005), pp. 123–151.

[110] J. O. Smith, *Mathematics of the Discrete Fourier Transform (DFT): With Audio Applications*, 2nd Ed., W3K Publishing, 2007.

[111] R. Soulard and P. Carré, *Characterization of color images with multiscale monogenic maxima*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 40 (2018), pp. 2289–2302.

[112] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting*, Journal of Machine Learning Research, 15 (2014), pp. 1929–1958.

[113] E. W. Stacy, *A generalization of the gamma distribution*, Annals of Mathematical Statistics, (1962), pp. 1187–1192.

[114] E. M. Stein and G. Weiss, *Introduction to Fourier Analysis on Euclidean Spaces (PMS-32), Volume 32*, Princeton University Press, 1971.

[115] C. Stolojescu-Crisan, *A hyperanalytic wavelet based denoising technique for ultrasound images*, in International Conference on Bioinformatics and Biomedical Engineering, Springer, 2015, pp. 193–200.

[116] Y. Sun, X. Wang, and X. Tang, *Deep convolutional network cascade for facial point detection*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3476–3483.

[117] M. Sundararajan and A. Najmi, *The many Shapley values for model explanation*, in International Conference on Machine Learning, PMLR, 2020, pp. 9269–9278.

[118] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, *On the importance of initialization and momentum in deep learning*, in International Conference on Machine Learning, PMLR, 2013, pp. 1139–1147.

[119] J. A. Thomas, C. F. Moss, and M. Vater, *Echolocation in Bats and Dolphins*, University of Chicago Press, 2004.

[120] C.-L. Tu, W.-L. Hwang, and J. Ho, *Analysis of singularities from modulus maxima of complex wavelets*, IEEE Transactions on Information Theory, 51 (2005), pp. 1049–1062.

[121] C. W. Turl, *Low-frequency sound detection by a bottlenose dolphin*, Journal of the Acoustical Society of America, 94 (1993), pp. 3006–3008.

[122] G. Tzanetakis and P. Cook, *Musical genre classification of audio signals*, IEEE Transactions on Speech and Audio Processing, 10 (2002), pp. 293–302.

[123] D. Ulyanov, A. Vedaldi, and V. Lempitsky, *Instance normalization: The missing ingredient for fast stylization*, arXiv preprint arXiv:1607.08022, (2016).

[124] G. Velarde, T. Weyde, C. E. C. Chacón, D. M. 0001, and M. Grachten, *Composer Recognition Based on 2D-Filtered Piano-Rolls.*, in Proceedings of the 17th International Society for Music Information Retrieval Conference, New York City, United States, Aug. 2016, ISMIR, pp. 115–121.

[125] S. P. Von Steinburg, A.-L. Boulesteix, C. Lederer, S. Grunow, S. Schiermeier, W. Hatzmann, K.-T. M. Schneider, and M. Daumer, *What is the "normal" fetal heart rate?*, PeerJ, 1 (2013), p. e82.

[126] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, *Image quality assessment: from error visibility to structural similarity*, IEEE Transactions on Image Processing, 13 (2004), pp. 600–612.

[127] D. Weber, *ContinuousWavelets.jl*, 2020. Julia package.

[128] ———, *ScatteringTransform.jl: An implementation of the generalized Scattering Transform*, 2021. Julia package.

[129] D. S. Weber, *On Interpreting Sonar Waveforms via the Scattering Transform*, PhD dissertation, Appl. Math., Univ. California, Davis, Dec. 2021.

[130] J. Weston and C. Watkins, *Multi-class support vector machines*, tech. rep., Citeseer, 1998.

[131] T. Wiatowski and H. Bölcskei, *Deep convolutional neural networks based on semi-discrete frames*, in 2015 IEEE International Symposium on Information Theory (ISIT), IEEE, 2015, pp. 1212–1216.

[132] T. Wiatowski and H. Bölcskei, *A mathematical theory of deep convolutional neural networks for feature extraction*, 64 (2018), pp. 1845–1866.

[133] R. Wong, *Error bounds for asymptotic expansions of integrals*, SIAM Review, 22 (1980), pp. 401–435.

[134] B. Xu, N. Wang, T. Chen, and M. Li, *Empirical evaluation of rectified activations in convolutional network*, in ICML Deep Learning Workshop, Lille, France, 2015.

[135] M. D. Yacoub, *The α-μ Distribution: A Physical Fading Model for the Stacy Distribution*, IEEE Transactions on Vehicular Technology, 56 (2007), pp. 27–34.

[136] M. Y. Zhang, J. R. Russell, and R. S. Tsay, *A nonlinear autoregressive conditional duration model with applications to financial transaction data*, Journal of Econometrics, 104 (2001), pp. 179–207.

[137] A. Zourabian, A. M. Siegel, B. Chance, N. Ramanujam, M. E. Rode, and D. A. Boas, *Trans-abdominal monitoring of fetal arterial blood oxygenation using pulse oximetry*, Journal of Biomedical Optics, 5 (2000), pp. 391–405.

[138] R. Zuidwijk, *M. holschneider, Wavelets: An Analysis Tool*, 1995.