Denoising Autoencoder for Multi-level Random Telegraph Signal Analysis

by

Bowen Deng

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2023

© Bowen Deng 2023

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Random Telegraph Signals (RTSs) are discrete random fluctuations between different current levels that usually appear in short-channel Metal-Oxide-Semiconductor Field Effect Transistors (MOSFETs). The statistical analysis of RTSs can facilitate the reliability assessment of short-channel MOSFETs. The accurate extraction of parameters in terms of amplitude, level transition, and durations from RTSs is of great importance to statistical analysis. However, background noise like Gaussian white noise or pink noise usually coexists with RTSs, making statistical analysis challenging. Such RTSs with background noise are noisy RTSs, which may contain level fluctuations in both local areas and general trends. To alleviate the impact of background noise on noisy RTSs, RTS reconstruction is usually applied to reconstruct pure RTS traces. The research on RTS reconstruction mainly focuses on models for digitization and parameter extraction, with comparatively limited emphasis on in-depth explorations of RTS denoising. To study the impact of RTS denoising on RTS reconstruction, the research in this thesis explores various RTS denoising models and proposes two Denoising AutoEncoder (DAE) models to enhance the performance of RTS denoising and further improve RTS reconstruction.

The research in this thesis generates a synthetic training dataset of multi-level noisy RTSs for DAE training. Four classical denoising models are implemented to compare denoising performance with DAE models on a synthetic evaluation dataset. In addition, three two-level and one multi-level RTS reconstruction models are implemented to validate further that RTS denoising cascading RTS reconstruction can enhance the performance of RTS reconstruction. According to the experimental analysis, DAE models not only have state-of-the-art denoising performance of multi-level RTSs as far as we know. Additionally, these models can dramatically enhance the performance of RTS reconstruction, especially when the pink noise level is high.

Acknowledgements

I would like to thank everyone who made this thesis possible.

First of all, I would like to express my sincerest thanks to my supervisor, Professor Na Young Kim. Professor Kim gave me the opportunity to transfer from MEng to MASc for research and provided all the support she could when I needed help. She respected my research and provided me with opportunities to study in different fields.

Secondly, I would like to thank the committee members, Prof. Zhou Wang and Prof. Mark Crowley, for their time. I am truly grateful for their reviews of my thesis.

I would like to thank Dr. HeeBong Yang for his guidance and advice on my research project and thesis writing.

I would also like to thank the other members of QuIN Lab, Zhuoyang He, and Scott Holtshouse. I would like to thank Steve Weiss of IQC Technology Services for his technical support. Thanks to my friend Bowen Li for his advice on my research. Finally, I would like to thank my parents for their support and encouragement during my study and thesis writing. I couldn't have done this without them.

Table of Contents

Author's Declaration ii

Abstract iii

Acknowledgements iv

List of Figures viii

List of Tables xi

List of Abbreviations xii

1 Introduction 1

2 Background Noise in RTS 5

2.1 Stochastic Process 5

2.1.1 Stationarity 5

2.1.2 Probability Density Function 6

2.2 Power Spectral Density 6

2.3 Background Noise coexist with RTSs 7

2.3.1 Gaussian white noise 7

2.3.2 Pink Noise 7

2.3.3 Noisy RTSs in the real world . . . 10
2.4 Generation of synthetic noisy RTSs . . . 10
2.5 Categories of RTSs . . . 12
2.5.1 Normal RTSs . . . 13
2.5.2 Anomalous RTSs . . . 13
2.6 Noise level range for recognizable RTSs . . . 14

3 RTS Denoising Models **18**
3.1 Moving Average Filtering . . . 18
3.2 Fast Fourier Transform Denoising . . . 19
3.3 Discrete Wavelet Transform Denoising . . . 21
3.4 Empirical Mode Decomposition Denoising . . . 21
3.5 Denoising Autoencoder Denoising . . . 23
3.5.1 Convolutional Neural Networks . . . 23
3.5.2 Long Short-Term Memory . . . 27
3.5.3 Denoising Autoencoder . . . 29
3.5.4 Denoising Autoencoder based on Fully Connected Networks . . . 31
3.5.5 Denoising Autoencoder based on U-Net and Bi-LSTM . . . 32

4 RTS Denoising Experiments and Analysis **35**
4.1 Dataset . . . 35
4.2 Experiment Details . . . 38
4.3 Root Mean Squared Error . . . 38
4.4 Signal-to-noise Ratio . . . 39
4.5 Denoising Visualization Analysis . . . 46
4.6 Ablation Study . . . 53
4.6.1 Effect of Bi-LSTM . . . 53
4.6.2 Effect of skip connection . . . 60

5 RTS Reconstruction Models **62**
5.1 Two-level RTS reconstruction 62
5.1.1 Convolutional Filtering Reconstruction 62
5.1.2 Discrete Wavelet Transform Reconstruction 63
5.1.3 Hidden Markov Model Reconstruction 65
5.1.4 Digitization for Two-Level Reconstruction 65
5.2 Multi-level RTS reconstruction 67

6 Enhancement of RTS Reconstruction by DAE Denoising **70**
6.1 Dataset 70
6.2 Evaluation metrics for RTS reconstruction 71
6.2.1 Amplitude Error 71
6.2.2 Digitization Error 71
6.2.3 Tau Error 72
6.3 Experiment Details 72
6.4 Two-level RTS Reconstruction Analysis 74
6.5 Multi-level RTS Reconstruction Analysis 76

7 Conclusion **79**

References **81**

List of Figures

1.1 Parameters of a two-level RTS 2
1.2 Comparison between pure RTSs and noisy RTSs with background noise . 3

2.1 Signal trace, PDF, and PSD of a Gaussian white noise 8
2.2 Signal trace, PDF, and PSD of a pink noise 9
2.3 Real RTS signal traces sampled from MOSFETs 10
2.4 Generation of noisy RTSs with Gaussian white noise 11
2.5 Generation of noisy RTSs with pink noise 12
2.6 Signal traces and PDFs of 1-trap RTSs with different noise levels 14
2.7 Signal traces and PDFs of 2-trap RTSs with different noise levels 15
2.8 Signal traces and PDFs of 3-trap RTSs with different noise levels 15
2.9 Signal traces and PDFs of coupled RTSs with different noise levels 16
2.10 Signal traces and PDFs of metastable RTSs with different noise levels . . . 16
2.11 Signal traces and PDFs of missing-level RTSs with different noise levels . 17

3.1 Fast Fourier Transform denoising flowchart 20
3.2 Discrete Wavelet Transform denoising flowchart 22
3.3 Empirical Mode Decomposition denoising flowchart 25
3.4 Example of one-dimensional convolution 26
3.5 Example of one-dimensional max pooling 27
3.6 Example of one-dimensional transpose convolution 28

3.7 Architecture of an LSTM cell . . . 30
3.8 Architecture of Bi-LSTM . . . 30
3.9 Architecture of general DAE . . . 31
3.10 Architecture of DAE FCN . . . 33
3.11 Architecture of DAE UBL . . . 34

4.1 RMSEs of 1-trap RTSs . . . 40
4.2 RMSEs of 2-trap RTSs . . . 41
4.3 RMSEs of 3-trap RTSs . . . 42
4.4 RMSEs of coupled RTSs . . . 43
4.5 RMSEs of missing-level RTSs . . . 44
4.6 RMSEs of metastable RTSs . . . 45
4.7 SNRs of 1-trap RTSs . . . 47
4.8 SNRs of 2-trap RTSs . . . 48
4.9 SNRs of 3-trap RTSs . . . 49
4.10 SNRs of coupled RTSs . . . 50
4.11 SNRs of missing-level RTSs . . . 51
4.12 SNRs of metastable RTSs . . . 52
4.13 Denoising plot of 1-trap RTS with pink noise . . . 54
4.14 Denoising plot of 1-trap RTS with Gaussian white noise . . . 54
4.15 Denoising plot of 2-trap RTS with pink noise . . . 55
4.16 Denoising plot of 2-trap RTS with Gaussian white noise . . . 55
4.17 Denoising plot of 3-trap RTS with pink noise . . . 56
4.18 Denoising plot of 3-trap RTS with Gaussian white noise . . . 56
4.19 Denoising plot of coupled RTS with pink noise . . . 57
4.20 Denoising plot of coupled RTS with Gaussian white noise . . . 57
4.21 Denoising plot of missing-level RTS with pink noise . . . 58
4.22 Denoising plot of missing-level RTS with Gaussian white noise . . . 58

4.23 Denoising plot of metastable RTS with pink noise 59
4.24 Denoising plot of metastable RTS with Gaussian white noise 59
4.25 Performance comparison between DAE and DAE+Bi-LSTM 60
4.26 Performance comparison between DAE+Bi-LSTM and DAE UBL 61

5.1 Convolutional filtering reconstruction process 64
5.2 Discrete wavelet transform reconstruction process 66
5.3 Hidden Markov model reconstruction process 67
5.4 Two-level RTS digitization process 67
5.5 Three-step analysis protocol architecture 69

6.1 Two-level RTS reconstruction process 73
6.2 Multi-level RTS reconstruction process 74

List of Tables

4.1 Statistics of the synthetic training dataset 36
4.2 Statistics of an RTS category in synthetic training dataset 36
4.3 Statistics of the synthetic evaluation dataset 37
4.4 Statistics of an RTS category in synthetic evaluation dataset 37
4.5 Trainable parameters of DAE denoising 38
4.6 Ablation study of DAE UBL 61

6.1 Statistics of the synthetic evaluation dataset for two-level RTS reconstruction 70
6.2 Statistics of the synthetic evaluation dataset for multi-level RTS reconstruction 71
6.3 Reconstruction errors of CF reconstruction 75
6.4 Reconstruction errors of DWT reconstruction 75
6.5 Reconstruction errors of HMM reconstruction 76
6.6 Successfully processed samples of three-step analysis protocol 77
6.7 Digitization errors of three-step analysis protocol 77
6.8 Tau high errors of three-step analysis protocol 78
6.9 Tau low errors of three-step analysis protocol 78

List of Abbreviations

aRTS Anomalous random telegraph signal 13, 35, 79

Bi-GRU bidirectional gated recurrent units 68, 69, 73

Bi-LSTM bidirectional long short-term memory 29, 32, 34, 53, 60

CF Convolutional filtering 62, 72, 74, 76, 80

CNN Convolutional Neural Network 23, 25–27, 31

DAE Denoising Autoencoder 4, 18, 23, 29, 31, 32, 38, 39, 46, 53, 60, 72, 79, 80

DAE FCN Denoising autoencoder based on fully connected networks 31, 32, 38, 39, 46, 53–59, 79

DAE UBL Denoising autoencoder based on U-Net and Bi-LSTM 32, 34, 38, 39, 46, 53–60, 73–80

DFT Discrete Fourier transform 19, 20

DWT Discrete wavelet transform 21, 38, 63, 72, 75, 76, 79, 80

EMD Empirical mode decomposition 21, 23, 38, 79

FCN fully connected networks 31

FFT fast Fourier transform 20, 38, 79

FT Fourier transform 6, 19

GMM Gaussian mixture model 68

HMM Hidden Markov model 65, 72, 75, 76, 80

IC integrated circuit 1

IDFT inverse discrete Fourier transform 19, 21

IFFT inverse fast Fourier transform 20

IMFs intrinsic mode functions 21, 23

KDE kernel density estimation 19, 68

LSTM long short-term memory 27–29

MA Moving average 18, 19, 38, 46, 53–59, 68, 79

MOSFETs metal-oxide-semiconductor field effect transistors 1, 2, 10, 12

MSE mean squared error 32, 34, 38

nRTS normal random telegraph signal 13, 35, 79

PDF Probability density function 6, 7, 13, 14, 68

PMOS p-channel metal–oxide–semiconductor 10

PSD Power spectral density 6, 7

ReLU rectified linear unit 26, 31, 32

RMSE root mean squared error 38, 39, 46, 53, 60, 79

RNN recurrent neural network 27, 68, 69

RTS random telegraph signal 2, 4, 10–14, 18, 20, 21, 23, 31, 32, 35, 38, 39, 46, 53, 60, 62–65, 67–80

SNR signal-to-noise ratio 38, 39, 46, 53, 79

Chapter 1

Introduction

Electronic devices have become essential to telecommunication, health care, automation, energy management, and many other areas. These devices control the flow of electrical currents for specific functions. Electronic devices encompass a wide range of devices in our daily lives, such as televisions, computers, and headphones. An electronic device comprises many electronic components that are basic discrete devices or physical entities in an electronic system used to affect electrons or their associated fields [1]. Active components, one of the two main categories of electronic components, are electronic components that can actively amplify, switch, or control electrical signals. Active components include transistors, integrated circuit (IC), diodes, and some other components. The invention of IC has enabled the miniaturization of electronic devices by integrating a large number of electronic components into a small chip. Miniaturization can significantly decrease the size, cost, and power consumption of electronic devices while improving their performance.

Although ICs are composed of many different categories of electronic components, metal-oxide-semiconductor field effect transistors (MOSFETs) are one of the most important electronic components in ICs. MOSFETs have served as the main category of transistors in ICs for the past two decades [61]. Four terminal devices are in MOSFETs: source, drain, gate, and body. The source is the terminal device where charge carriers enter the channel between the source and the drain. The drain is the terminal device where charge carriers exit the channel. The gate is a metal electrode whose voltage (gate voltage) controls the current and width of the channel. The body is usually connected to the source in most MOSFETs.

Because of the fabrication process, device dimensions, temperature, and bias, tem-

poral fluctuation (noise) becomes one of the accuracy-limiting factors [50]. The channel current in MOSFETs can be affected by various temporal fluctuations, including random telegraph signal (RTS), thermal noise, Gaussian white noise, pink noise, and other fluctuations. RTSs are discrete random fluctuations or switching events as a function of time [9], which is usually a significant concern in short-channel MOSFETs. These fluctuations are randomly caused by charge carrier trapping and de-trapping in the oxide layer [3][5]. In recent years, the study and observation of RTSs have been emphasized more heavily to assess the reliability of MOSFETs [35][53][58] and generate the true random numbers [7].

The simplest two-level RTSs are frequently encountered as a single carrier stochastic process that switches between two distinct low and high levels [65]. Three parameters in Fig.1.1 are significant in two-level RTS analysis. The transition amplitude Δ_{RTS} refers to the difference between low and high levels of a two-level RTS. τ_{high} and τ_{low} denote individual durations for the high and low levels. By collecting all individual durations, $\bar{\tau}_{high}$ and $\bar{\tau}_{low}$ can be obtianed by fitting Poisson function [56]. Compared to two-level RTSs, multi-level RTSs are more complex signals with multiple discrete levels and traps.

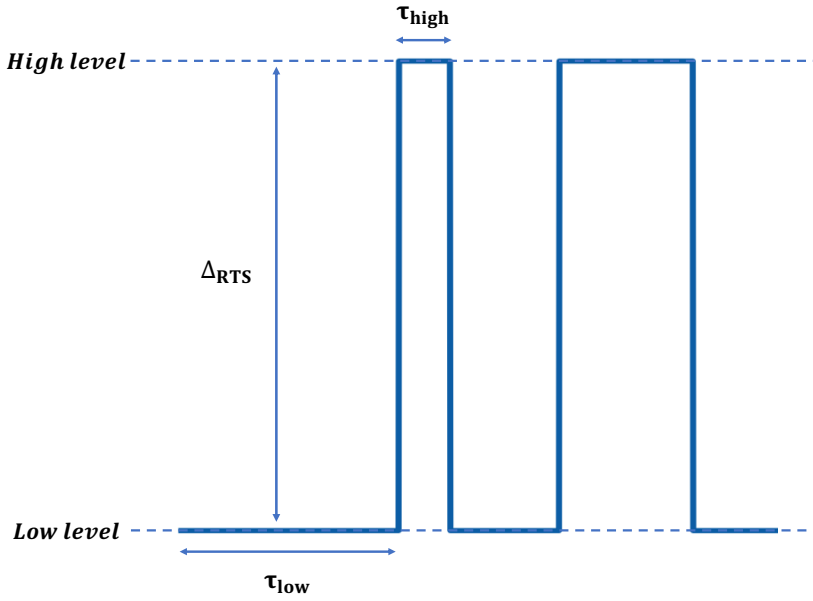
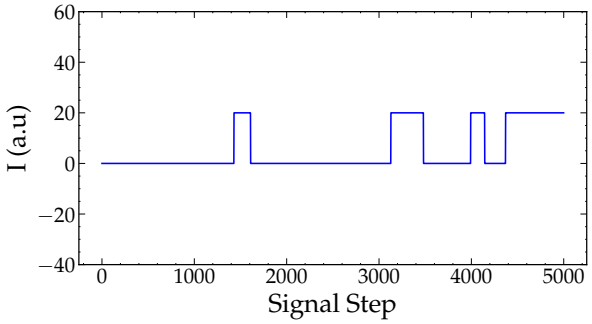
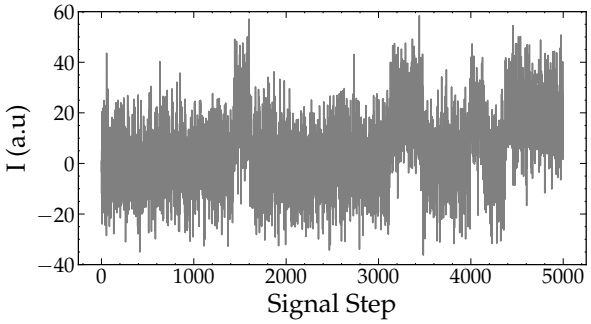


Figure 1.1: Parameters of a two-level RTS

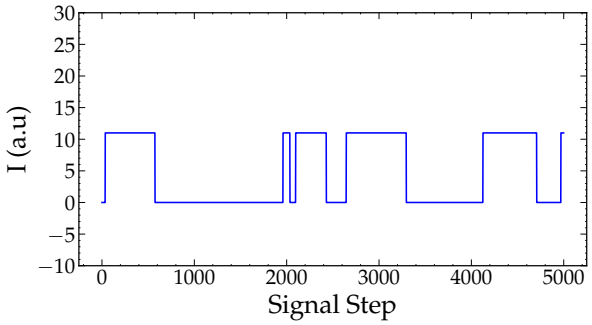
The statistical analysis is straightforward for pure two-level and multi-level RTSs.



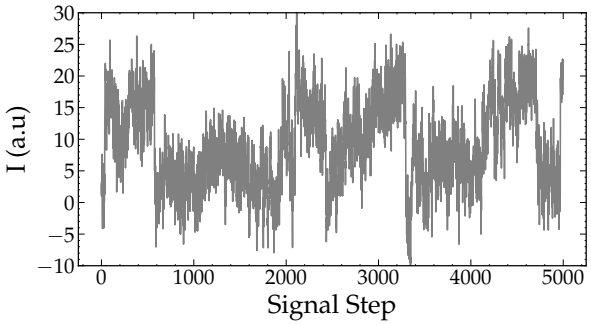
(a) Pure 1-trap RTS with Gaussian white noise



(b) Noisy 1-trap RTS with Gaussian white noise



(c) Pure 1-trap RTS with pink noise



(d) Noisy 1-trap RTS with pink noise

Figure 1.2: Comparison between pure RTSs and noisy RTSs with background noise

We can directly apply statistical analysis on the pure RTS in the time domain to extract parameters easily. However, for noisy RTSs in the real application, unwanted background noise that coexists with two-level and multi-level RTSs make the analysis challenging. Common background noise that coexist with RTS include Gaussian white noise [24] [33] and pink noise [5][47]. The details of these two types of background noise can be found in section 2.3. Fig.1.2 displays the comparison between two pure RTSs (Fig.1.2(a), Fig.1.2(c)) and related noisy RTSs with Gaussian white noise and pink noise (Fig.1.2(b), Fig.1.2(d)). According to the figure, Gaussian white noise and pink noise have covered the signal traces of two pure RTSs and increased the difficulty of identifying level transitions.

To successfully analyze noisy RTSs, it is necessary to implement the RTS reconstruction. Most studies focus on digitization and parameter extraction methods during reconstruction, but few look into RTS denoising. As a preprocessing process, an effective RTS denoising model can greatly contribute to RTS reconstruction and even other types of RTS analysis. With this motivation, we propose two models based on Denoising Autoencoder (DAE) for two-level and multi-level RTS denoising with state-of-the-art performances as far as we know and they can even enhance the performance of RTS reconstructions. In addition, these models are easy to implement with fewer predefined parameters but have better robustness compared to the previous denoising models.

The research in this thesis investigates the denoising model for synthetic noisy RTSs in depth and verifies the enhancement of the denoised signal for RTS analysis. Chapter 2 explores background noise characteristics and generation of noisy RTSs. Chapter 3 introduces four traditional denoising models and proposes two new DAEs for RTS denoising. In Chapter 4, the focus turns to denoising errors for all models. A thorough analysis of the denoising errors is provided to reveal the potential advantages of the two DAE models. Chapter 5 introduces RTS reconstruction models for two-level and multi-level RTS reconstructions. Finally, Chapter 6 analyzes the reconstruction errors of these reconstruction models in detail to validate the enhancement of RTS reconstruction when applying a DAE as a preprocessing technique.

Chapter 2

Background Noise in RTS

2.1 Stochastic Process

In real-world signal processing, it is ubiquitous that some background noises coexist with the pure signal and make signal acquisition and analysis challenging. Stochastic processes can describe and model different background noise. A stochastic process is a collection of random variables in the same probability space [41], where random variables are variables determined by chance or randomness. A stochastic process z_t on domain $\mathcal{T}$ can be stated as.

$$p(z_{t_1}, ..., z_{t_m}) \tag{2.1}$$

for all $t_1, ..., t_m \in \mathcal{T}$.

2.1.1 Stationarity

Stationarity is one of the fundamental properties of stochastic processes. A stochastic process $z(t)$ on a domain $\mathcal{T}$ is strictly stationary if its statistical properties are invariant over time, which can be stated as [22],

$$p(z_{t_1}, ..., z_{t_m}) = p(z_{t_{1+h}}, ..., z_{t_{m+h}}) \tag{2.2}$$

for all $t_1, ...t_m, h \in \mathcal{T}$. Because of the fixed statistical parameters, stationarity allows us to get insights from stochastic processes with limited samples.

2.1.2 Probability Density Function

Probability density function (PDF), denoted as $f(x)$, is a function that describes the probability distributions of a continuous random variable. Given a PDF, the probability of a random variable x can be calculated as [26],

$$P(x_0 \leq x \leq x_1) = \int_{x_0}^{x_1} f(x)dx \tag{2.3}$$

where x is a continuous variable, and $[x_0, x_1]$ is the specific range of x. $f(x)$ must meet the following equations to ensure its validity.

$$f(x) \geq 0, \quad ,\forall x \tag{2.4}$$

$$\int_{-\infty}^{+\infty} f(x)dx = 1 \tag{2.5}$$

2.2 Power Spectral Density

Power spectral density (PSD) is a frequency-domain method for measuring power distributions with a specific range of frequencies. PSD can be calculated by applying Fourier transform (FT) which can convert the signal from the time domain to the frequency domain. It is typically expressed in units such as watts per hertz (W/Hz) or decibels per hertz (dB/Hz). For a signal $x(t)$, PSD can be calculated as [34],

$$S_x(f) = \lim_{T\to\infty} \frac{|X(f)|^2}{T} \tag{2.6}$$

where $S_x(f)$ is the PSD of $x(t)$, T is the duration of the signal, $X(f)$ is the FT of $x(t)$, and f is the frequency of $x(t)$. $X(f)$ can be calculated as.

$$X(f) = \int_{-\frac{T}{2}}^{\frac{T}{2}} x(t)e^{-2\pi ft}dt \tag{2.7}$$

2.3 Background Noise coexist with RTSs

2.3.1 Gaussian white noise

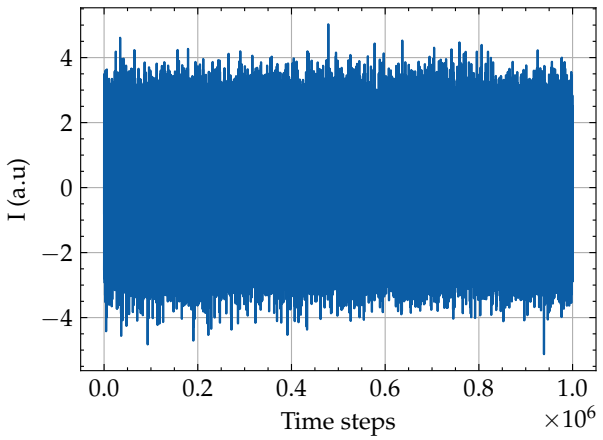
Gaussian white noise is a type of ubiquitous background noise that follows the Gaussian distribution and has the same intensity or power throughout the frequency spectrum. However, the frequency range is typically constrained due to system or measurement limitations. In practice, noise can be considered Gaussian white noise as long as it has the same intensity or power across a specific frequency range. Gaussian white noise is strictly stationary [27] with constant mean μ_{wn} and standard deviation σ_{wn}. Fig.2.1 display the signal trace, PDF, and log scale PSD of Gaussian white noise with 1,000,000 time steps, $\mu_{wn} = 0$, and $\sigma_{wn} = 1$. The PDF of Gaussian white noise can be calculated as,

$$f_G(I_{wn}) = \frac{1}{\sigma_{wn}\sqrt{2\pi}} e^{-\frac{I_{wn}^2}{2\sigma_{wn}^2}} \tag{2.8}$$

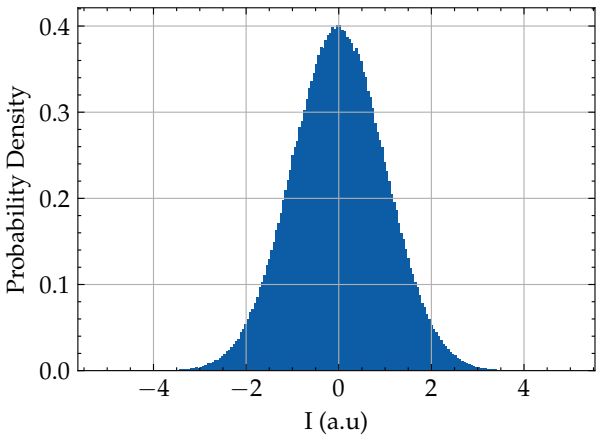
where I_{wn} is the current intensity of the Gaussian white noise. The PDF in Fig.2.1(b) is estimated by a histogram with 200 bins. It can be seen from the figure that PDF of Gaussian white noise obeys the Gaussian distribution. The log scale PSD in Fig.2.1(c) is estimated by Welch's method [62] with a sampling frequency of 10,000 Hz and 1,024 steps for each segment. According to the figure, the density fluctuates around a constant from 0 Hz to 5,000 Hz, the largest undistorted interval obtained with a sampling frequency of 10,000 Hz.

2.3.2 Pink Noise

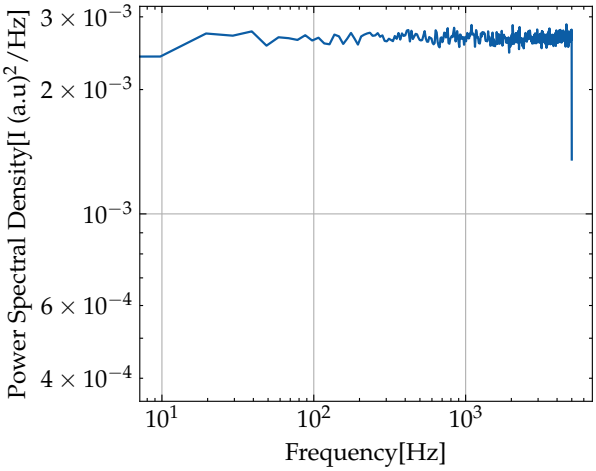
Pink noise, also known as $1/f$ noise, is a random signal whose PSD is inversely proportional to frequency. Pink noise does not strictly follow the Gaussian distribution. However, since pink noise is the sum of many random processes, its probability distribution still tends to approximate the Gaussian distribution. Due to the mean and standard deviation of pink noise are not constant, pink noise is not stationary. Fig.2.2 displays the signal trace, PDF, and log scale PSD of pink noise with 1,000,000 time steps and $\sigma_{pn} = 1$. The PDF in Fig.2.2(b) is estimated by a histogram with 200 bins. It can be seen that PDF is similar to a Gaussian distribution with $\mu_{pn} \neq 0$. The log scale PSD in Fig.2.2(c) is estimated by Welch's method [62] with a sampling frequency of 10,000 Hz and 1,024 steps for each segment. According to the figure, the PSD decreases linearly as the frequency increases.



(a) Signal trace of a Gaussian white noise

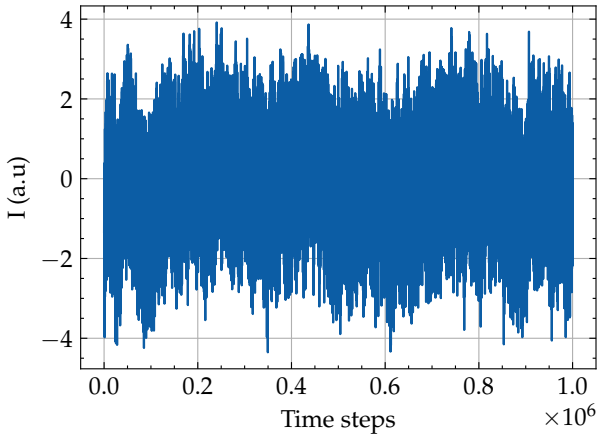


(b) PDF of a Gaussian white noise

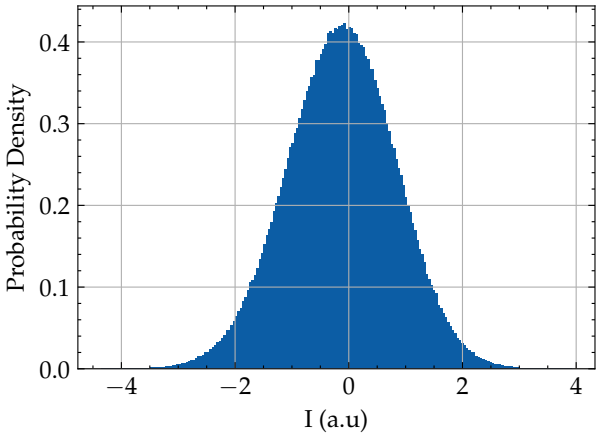


(c) PSD of a Gaussian white noise

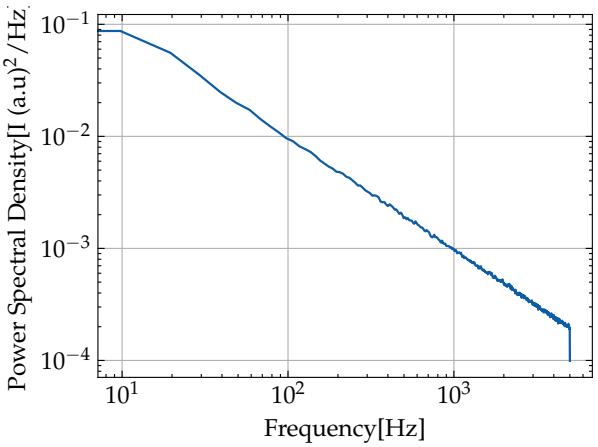
Figure 2.1: Signal trace, PDF, and PSD of a Gaussian white noise



(a) Signal trace of a pink noise



(b) PDF of a pink noise

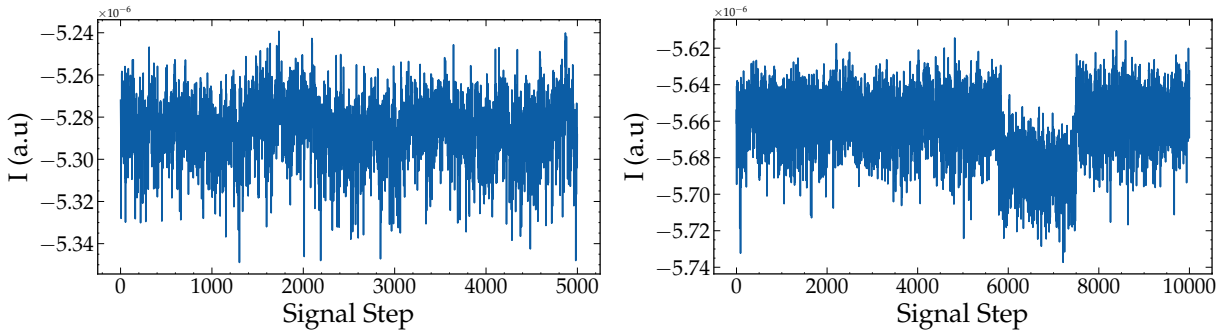


(c) PSD of a pink noise

Figure 2.2: Signal trace, PDF, and PSD of a pink noise

2.3.3 Noisy RTSs in the real world

In order to explore the impact of these background noise on pure RTS trace, Fig.2.3 shows two real noisy RTSs with a signal length of 5000 sampled from a 28nm p-channel metal–oxide–semiconductor (PMOS), a type of MOSFETs. It can be seen from the figure that the signal traces of the two real noisy RTSs are more obviously affected by pink noise than white noise. This is mainly because the signal traces contain obvious general trend fluctuations, which can only be caused by pink noise (explained in 2.4). However, we cannot ignore the local fluctuations caused by Gaussian white noise and pink noise. In order to simulate real RTSs, synthetic noisy RTSs will be generated for RTS denoising and reconstruction in the next section.



(a) Signal trace of a real noisy RTS (Gate voltage=-0.49V, Drain voltage=-0.01V)

(b) Signal trace of a real noisy RTS (Gate voltage=-0.51V, Drain voltage=-0.01V)

Figure 2.3: Real RTS signal traces sampled from MOSFETs

2.4 Generation of synthetic noisy RTSs

A noisy RTS is usually assumed to be the addition of a pure RTS and a background noise. The current intensity $I(t)$ of a synthetic noisy RTS at signal step t can be calculated as,

$$I(t) = I_{RTS}(t) + \sigma_n I_n(t) \tag{2.9}$$

where $I_{RTS}(t)$ is the current intensity of the pure RTS at signal step t, σ_n is the standard deviation of the background noise, $I_n(t)$ is the current intensity of the background noise.

Q_n is defined to represent the background noise level which can be calculated as,

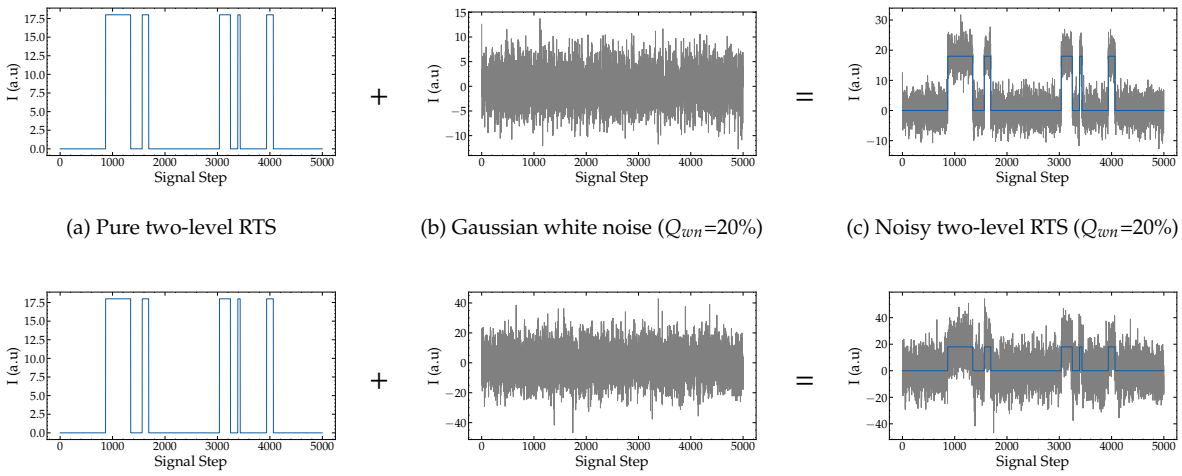
$$Q_n = \frac{\sigma_n}{min_{\{1,2...,N_{traps}\}}(\Delta_{RTS,i})} \tag{2.10}$$

where N_{traps} is the trap number of an RTS, $\Delta_{RTS,i}$ is the transition amplitude of the i^{th} trap. Q_n can be Gaussian white noise level Q_{wn} or pink noise level Q_{pn}, σ_n can be Gaussian white noise standard deviation σ_{wn} or pink noise standard deviation σ_{pn}. Therefore, $I(t)$ can also be calculated as,

$$I(t) = I_{RTS}(t) + min_{\{1,2...,N_{traps}\}}(\Delta_{RTS,i})Q_n I_n(t) \tag{2.11}$$

when N_{traps} = 1, which means for two-level RTSs, $I(t)$ can also be calculated as.

$$I(t) = I_{RTS}(t) + \Delta_{RTS}Q_n I_n(t) \tag{2.12}$$

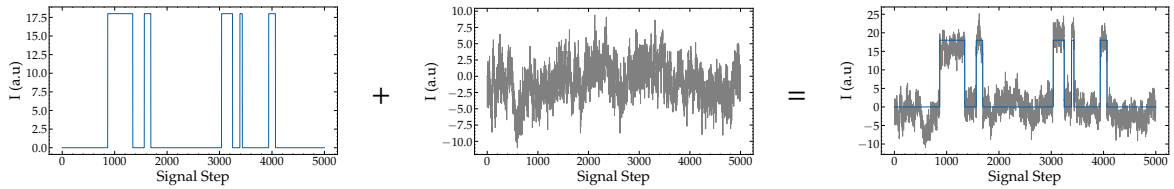


(a) Pure two-level RTS (b) Gaussian white noise (Q_{wn}=20%) (c) Noisy two-level RTS (Q_{wn}=20%)

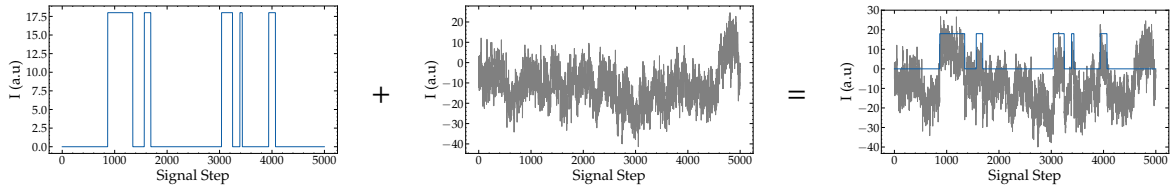
(d) Pure two-level RTS (e) Gaussian white noise (Q_{wn}=60%) (f) Noisy two-level RTS (Q_{wn}=60%)

Figure 2.4: Generation of noisy RTSs with Gaussian white noise

According to Eq.2.11, a synthetic noisy RTS with Gaussian white noise or pink noise can be generated as training and evaluation data for denoising models. Fig.2.4 shows the generations of synthetic noisy RTSs with Q_{wn} = 20% and 60%. Fig.2.4(a) and Fig.2.4(d) are pure two-level RTS traces in the time domain. Fig.2.4(b) and Fig.2.4(e) are the Gaussian white noise traces generated according to Δ_{RTS} and Q_{wn} of two-level RTSs . Fig.2.4(c) and Fig.2.4(f) are the additions of the pure RTS and the Gaussian white noise,



(a) Pure two-level RTS (b) Pink noise (Q_{pn}=20%) (c) Noisy two-level RTS (Q_{pn}=20%)



(d) Pure two-level RTS (e) Pink noise (Q_{pn}=60%) (f) Noisy two-level RTS (Q_{pn}=60%)

Figure 2.5: Generation of noisy RTSs with pink noise

which are synthetic noisy RTSs. According to the figures, as Q_{wn} increases, the larger the level fluctuations appear in the synthetic noisy RTSs. Furthermore, when Q_{wn} = 60%, the amplitude of the Gaussian white noise has already exceeded Δ_{RTS}. According to Fig.2.4(c) and Fig.2.4(f), it is evident that the Gaussian white noise only causes local fluctuations of the RTS.

Fig.2.5 displays the generations of synthetic noisy RTSs when Q_{pn} = 20% and 60%. The level fluctuations RTSs with pink noise are positively correlated with Q_{pn}. Pink noise has a more complex impact on RTSs than Gaussian white noise because pink noise not only creates fluctuations in the local areas of RTSs but also influences the general trends of RTSs. When Q_{pn} is small, the synthetic noisy RTS trace in Fig.2.5(c) still fluctuates near the pure RTS trace. When Q_{pn} is large, the synthetic noisy RTS traces in Fig.2.5(f) deviate from the general trends of the pure RTS traces. Therefore, synthetic noisy RTSs with pink noise are more challenging to analyze than those with Gaussian white noise.

2.5 Categories of RTSs

RTSs that appear in MOSFETs are not only two-level RTSs but also include multi-level RTSs that have multiple levels and traps. The analysis and processing of multi-level

RTSs are challenging due to their complex statistical properties, which are much more intricate than two-level cases. To provide a comprehensive understanding of the various categories of RTSs, a detailed introduction is included in this section to clarify six categories of two-level and multi-level RTSs discussed in this thesis.

2.5.1 Normal RTSs

Three categories of normal random telegraph signal (nRTS) are introduced, including 1-trap, 2-trap, and 3-trap RTSs. 1-trap RTSs are the same as two-level RTSs. Fig.2.6(a) and Fig.2.6(d) display the signal trace and PDF of a pure 1-trap RTS. According to the figure, it can be seen that the 1-trap RTS has two peaks representing the low level and the high level, which are consistent with the properties of a two-level RTS. The theoretical number of PDF peaks for multi-level RTSs can be calculated as,

$$N_{peaks} = 2^{N_{traps}} \tag{2.13}$$

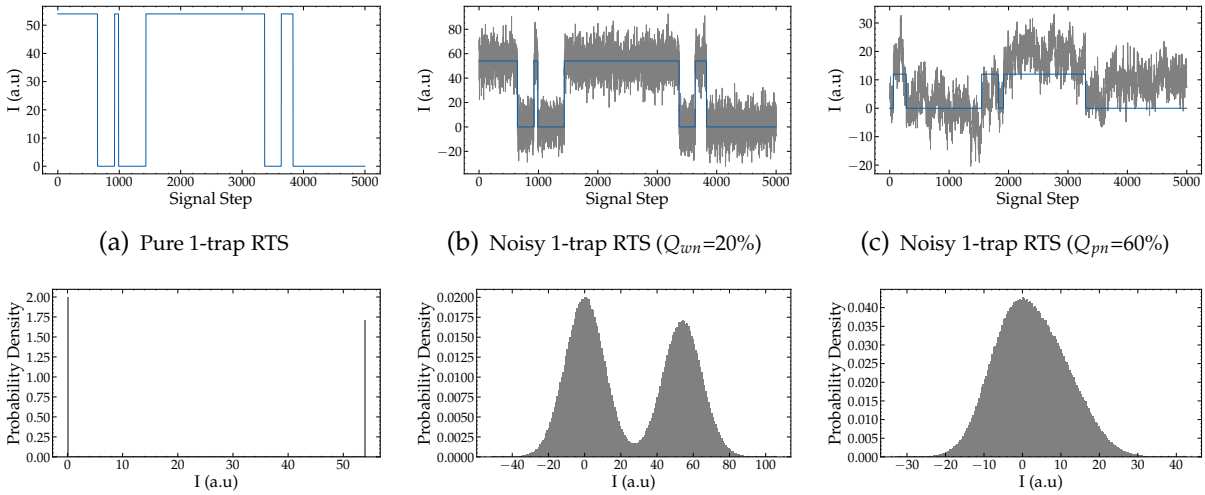
where N_{peaks} is the number of PDF peaks, and N_{traps} is the number of traps. According to Fig.2.7(d) and Fig.2.8(d), N_{peaks} of pure 2-trap and 3-trap RTSs are 4 and 8, respectively. Combining the observations of N_{traps} and N_{peaks}, it can be seen that as N_{traps} increases, the complexity (N_{peaks}) of nRTS increases exponentially.

2.5.2 Anomalous RTSs

Anomalous random telegraph signal (aRTS) occurs when unanticipated anomalies disrupt the consistent current level transition, resulting in RTSs that require closer inspection. In this thesis, three categories of aRTSs are considered according to observable features [60][66]. The first category is metastable RTSs, which can be described as a 1-trap aRTSs with two distinctive zones exhibiting opposite $\bar{\tau}_{high}$ and $\bar{\tau}_{low}$. Fig.2.10(a) shows the signal trace of a metastable RTS. According to the figure, zone A and zone B are dominated by high level ($\bar{\tau}_{high} > \bar{\tau}_{low}$) and low level ($\bar{\tau}_{high} < \bar{\tau}_{low}$), respectively. The other two categories are multi-level aRTSs. Only $N_{traps} = 2$ is considered in the research. The first multi-level aRTSs is coupled RTSs, Δ_{RTS} of each trap is influenced by the state of the other trap (Fig.2.9(a) and Fig.2.9(d)). The second multi-level aRTS, missing-level RTS, is a special case of coupled RTSs. In a 2-trap missing-level RTS, one level is missing, resulting in only three levels in the RTS (Fig.2.11(a) and Fig.2.11(d)).

2.6 Noise level range for recognizable RTSs

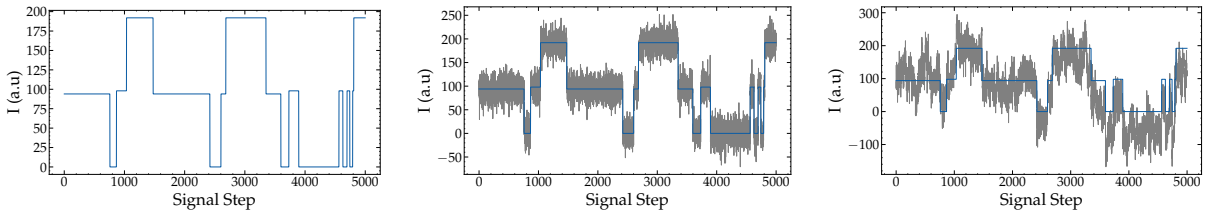
According to the observation in Section 2.4, it has been established that when Q_{wn} or Q_{pn} is equal to 60%, amplitude of background noise are even greater than those of two-level RTSs. This phenomenon also occurs in multi-level RTSs. According to Fig.2.6 to Fig.2.11, the impact of background noise on PDFs for two-level and multi-level RTSs is examined for RTSs when Q_{wn} = 20% and Q_{pn}=60% to provide further insight into how background noise affects PDF peaks of RTSs. When Q_{wn} = 20%, PDFs of RTSs spread from the concentration of several intensities to a specific range. Except for 1-trap cases (1-trap and metastable), which still maintain the correct peaks number, other categories of RTSs have some peaks covered due to level fluctuations. When Q_{pn} = 60%, PDFs of most RTSs have only one prominent peak, which means that the fluctuation of RTS has been covered by background noise. Therefore, only $Q_{wn} \leq 60\%$ and $Q_{pn} \leq 60\%$ are considered to ensure that RTSs still keep some characteristics of their category.



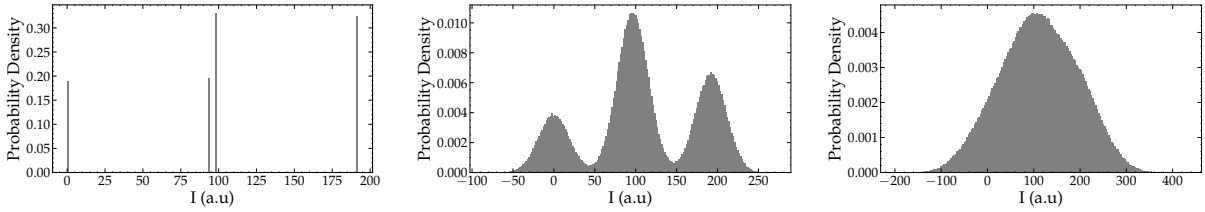
(a) Pure 1-trap RTS (b) Noisy 1-trap RTS (Q_{wn}=20%) (c) Noisy 1-trap RTS (Q_{pn}=60%)

(d) PDF of pure 1-trap RTS (e) PDF of noisy 1-trap RTS (Q_{wn}=20%) (f) PDF of noisy 1-trap RTS (Q_{pn}=60%)

Figure 2.6: Signal traces and PDFs of 1-trap RTSs with different noise levels

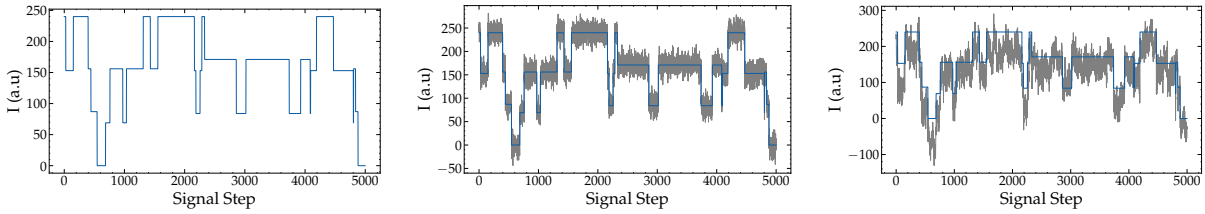


(a) Pure 2-trap RTS (b) Noisy 2-trap RTS (Q_{wn}=20%) (c) Noisy 2-trap RTS (Q_{pn}=60%)

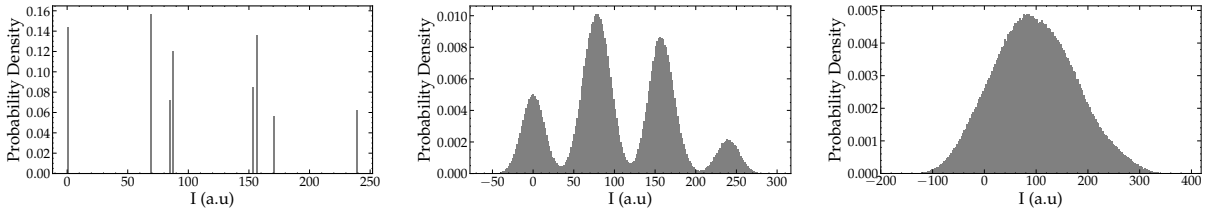


(d) PDF of pure 2-trap RTS (e) PDF of noisy 2-trap RTS (Q_{wn}=20%) (f) PDF of noisy 2-trap RTS (Q_{pn}=60%)

Figure 2.7: Signal traces and PDFs of 2-trap RTSs with different noise levels

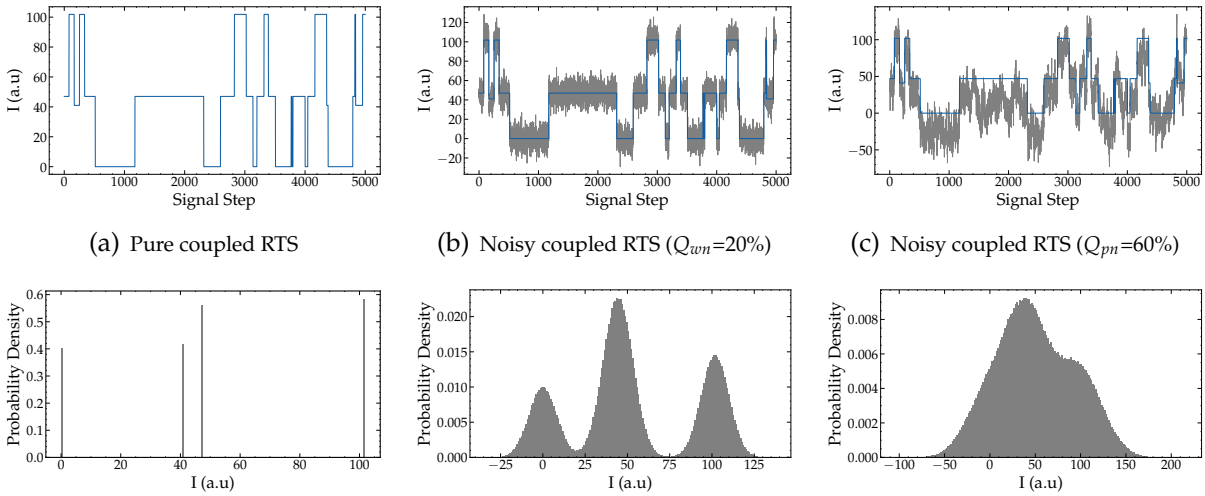


(a) Pure 3-trap RTS (b) Noisy 3-trap RTS (Q_{wn}=20%) (c) Noisy 3-trap RTS (Q_{pn}=60%)



(d) PDF of pure 3-trap RTS (e) PDF of noisy 3-trap RTS (Q_{wn}=20%) (f) PDF of noisy 3-trap RTS (Q_{pn}=60%)

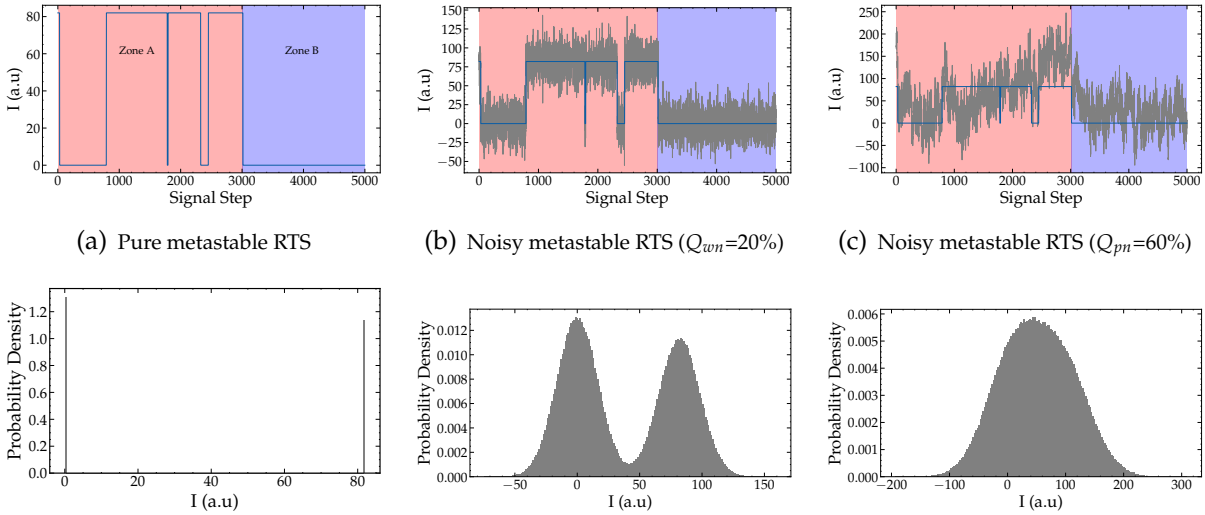
Figure 2.8: Signal traces and PDFs of 3-trap RTSs with different noise levels



(a) Pure coupled RTS (b) Noisy coupled RTS (Q_{wn}=20%) (c) Noisy coupled RTS (Q_{pn}=60%)

(d) PDF of pure coupled RTS (e) PDF of noisy coupled RTS (Q_{wn}=20%) (f) PDF of noisy coupled RTS (Q_{pn}=60%)

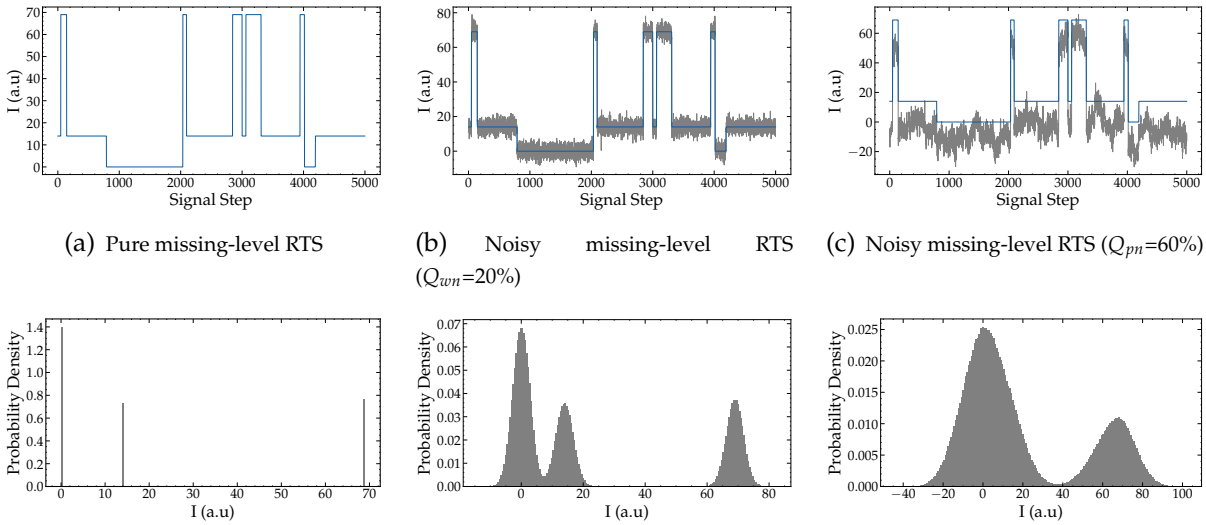
Figure 2.9: Signal traces and PDFs of coupled RTSs with different noise levels



(a) Pure metastable RTS (b) Noisy metastable RTS (Q_{wn}=20%) (c) Noisy metastable RTS (Q_{pn}=60%)

(d) PDF of pure metastable RTS (e) PDF of noisy metastable RTS (Q_{wn}=20%) (f) PDF of noisy metastable RTS (Q_{pn}=60%)

Figure 2.10: Signal traces and PDFs of metastable RTSs with different noise levels



(a) Pure missing-level RTS

(b) Noisy missing-level RTS (Q_{wn}=20%)

(c) Noisy missing-level RTS (Q_{pn}=60%)

(d) PDF of missing-level RTS

(e) PDF of missing-level RTS (Q_{wn}=20%)

(f) PDF of missing-level RTS (Q_{pn}=60%)

Figure 2.11: Signal traces and PDFs of missing-level RTSs with different noise levels

Chapter 3

RTS Denoising Models

To analyze noisy RTSs efficiently and comprehensively, RTS reconstruction is usually applied to uncover features from noisy RTSs. Although there have been many related studies on RTS reconstruction, most of them mainly focus on digitization and statistical analysis [23][38][44][45][46][65]. In fact, RTS denoising, a widely used preprocessing technique, plays an important role in the performance of RTS reconstruction. To delve deeper into RTS denoising, we propose two DAEs to achieve superior denoising performance. Four classical denoising models are also introduced for comparison in this chapter.

3.1 Moving Average Filtering

Moving average (MA) filter, the simplest and easiest digital filter [32], is widely applied for one-dimensional signal denoising [43][52][54]. The algorithm details of MA filtering are shown in Algorithm 1. In general, MA filtering is capable of smoothing the noisy signal by sliding a window onto valid signal points and assigning the average intensities to each point. The window size L can be successfully estimated from the noise level [48],

$$L = \text{floor}[60 \cdot \tan^{-1}(\frac{\sigma_{n_{est}}}{4\pi})] \tag{3.1}$$

where $\sigma_{n_{est}}$ is the standard deviation of the background noise. For RTS denoising, MA filtering was utilized as a time domain filtering technique to increase the Lorentzian spectrum detection rate in [18]. MA filtering was also applied to uncover density peaks

from background noise before performing kernel density estimation (KDE)[48][63][64]. However, MA filtering has limitations in effectively denoising signals with high noise levels and keeping the edge sharpness of the signal.

Algorithm 1 Moving Average Filtering

Input: Noisy RTS x, window size L where $L \leq \text{len}(x)$
Output: Denoised RTS x_{denoised}
procedure MA FILTERING(x, L)
    Initialize an empty 1D array x_{denoised} where $\text{len}(x_{\text{denoised}}) = \text{len}(x)$
    Initialize a sum variable $S = 0$
    for $n = L$ to $\text{len}(x)$ **do**
        for $i = 1$ to L **do**
            $S \leftarrow S + x[i]$
        end for
        $x_{\text{denoised}}[n] \leftarrow S/L$
    end for
    return x_{denoised}
end procedure

3.2 Fast Fourier Transform Denoising

Besides denoising models in the time domain, there are also some denoising models in the frequency domain. FT is one of the most prevalent models for signal denoising and analysis [17][38][42][65]. Discrete Fourier transform (DFT) is the approximation of the continuous Fourier Transform of a function [39]. It can convert a time domain signal to a spectrum for analysis and reconstruct the signal from the frequency domain, called inverse discrete Fourier transform (IDFT). The process of DFT can be defined as,

$$X[k] = \sum_{m=0}^{N-1} x[m] \cdot e^{-j\frac{2\pi}{N}km} \tag{3.2}$$

where $X[k]$ represents the spectrum of the signal at frequency index k, and $x[m]$ is the signal intensities at signal step m. The process of IDFT can be defined as.

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \cdot e^{j\frac{2\pi}{N}kn} \tag{3.3}$$

Although DFT provides precise frequency domain representation for discrete signals, the algorithm is not efficient enough for processing signals with long signal lengths. The time complexity of DFT is $O(N^2)$. To solve this problem, fast Fourier transform (FFT) is proposed by Cooley and Tukey [12] to reduce the time complexity to $O(NlogN)$.

Fig.3.1 and Algorithm 2 show the complete denoising procedure of FFT denoising model. Given a noisy RTS (Fig.3.1, left), the algorithm computes the spectrum (Fig.3.1, middle) through FFT. Since background noise tends to exhibit a lower intensity than pure RTS, filtering frequency components with lower intensities by hard thresholding is applied for denoising (rows 5 and 6 in Algorithm 2). The scale parameter s for the threshold T is 1,000. Finally, the denoised RTS is reconstructed through inverse fast Fourier transform (IFFT) (Fig.3.1, right).

Algorithm 2 FFT Denoising

1: **Input:** Noisy RTS x with signal length N, scale parameter s
2: **Output:** Denoised RTS x_{denoised}
3: **procedure** FFT DENOISING(x, s)
4: Compute the spectrum of the noisy RTS through FFT: $X \leftarrow \text{FFT}(x)$
5: Set the threshold T = max(x) - min(x) / s
6: Remove the intensity is lower than T: $X_{\text{denoised}} \leftarrow \{0 \text{ if } |X[k]| \leq T, \text{ else } X[k]\}$
7: Reconstruct the denoised RTS through IFFT: $x_{\text{denoised}} \leftarrow \text{IFFT}(X_{\text{denoised}})$
8: **return** denoised RTS x_{denoised}
9: **end procedure**

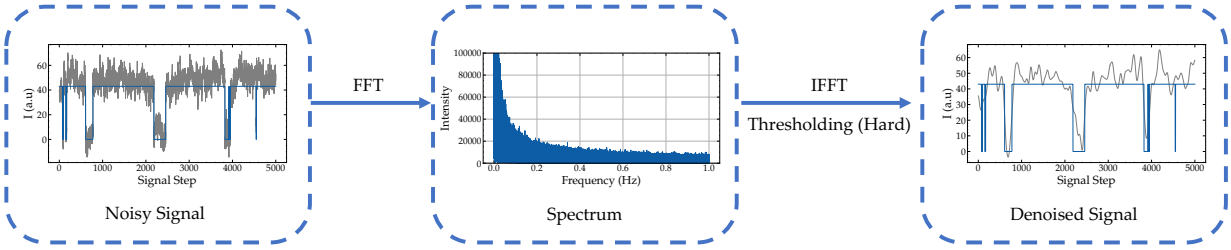


Figure 3.1: Fast Fourier Transform denoising flowchart

3.3 Discrete Wavelet Transform Denoising

Discrete wavelet transform (DWT) is an advanced frequency domain method that can capture localized features in both time and frequency domains. DWT is capable of decomposing the signal into a set of fixed wavelets, which are small, localized oscillations in the time domain. DWT denoising is widely applied in signal processing [2][13][14][23] due to its ability to analyze signals with different resolutions. Haar DWT is applied here for RTS denoising. The process of DWT with Haar wavelets can be considered as the decomposition of noisy RTS into approximation coefficients $a[k]$ and detail coefficients $d[k]$,

$$a[k] = \frac{x_{2k-1} + x_{2k}}{\sqrt{2}} \qquad 1 < k \leq \frac{N}{2} \tag{3.4}$$

$$d[k] = \frac{x_{2k-1} - x_{2k}}{\sqrt{2}} \qquad 1 < k \leq \frac{N}{2} \tag{3.5}$$

where k is the signal step, x_{2k-1} is the RTS intensity of the signal step $2k-1$, and N is the signal length of RTS. By iteratively applying Eq.3.4 and 3.5 to the approximation coefficients, multi-level DWT can be achieved, which implements wavelet decompositions on multiple scales and resolutions.

Fig.3.2 and Algorithm 3 display the complete denoising procedure of the DWT denoising model. First, DWT decomposition is iteratively performed to obtain the approximation and detail coefficients of all levels (row 4 in Algorithm 3). Then the denoising threshold is calculated based on the detail coefficient of level j (row 5 and 6 in Algorithm 3). Soft thresholding is applied to remove noise components for all detail coefficients. Finally, IDFT is used to transform the components into denoised RTS x_{denoised}. For RTS denoising, the level of decomposition is 5.

3.4 Empirical Mode Decomposition Denoising

Empirical mode decomposition (EMD) is a data-driven method that does not require predefined functions and parameters. EMD can adaptively decompose the signal into intrinsic mode functions (IMFs), representing different intrinsic harmonics of the signal. Therefore, EMD performs very well in decomposing non-stationary signals. The algorithm details of EMD are shown as Algorithm 4. The algorithm returns a set of IMFs

Algorithm 3 DWT Denoising

Input: Noisy RTS x with signal length N, the level of decomposition j
Output: Denoised RTS x_{denoised}
procedure DWT DENOISING(x, j)
    Perform multi-level DWT decomposition to obtain approximation and detail coefficients A, D in all levels $A_i, D_i \leftarrow \text{DWT}(A_{i-1})$
    Calculate $\sigma = \frac{1}{0.6745}\text{median}(|D_j - \bar{D}_j|)$
    Set the threshold $T = \sigma\sqrt{2\log L}$
    Soft thresholding all detail coefficients $D_i[t] \leftarrow \{0 \text{ if } D_i[t] \leq T, \text{else } sign(D_i[t]) * (|D_i[t]| - T)\}$
    Reconstruct the denoised RTS through IDWT $x_{\text{denoised}} \leftarrow \text{IDWT}(A, D)$
    return denoised RTS x_{denoised}
end procedure

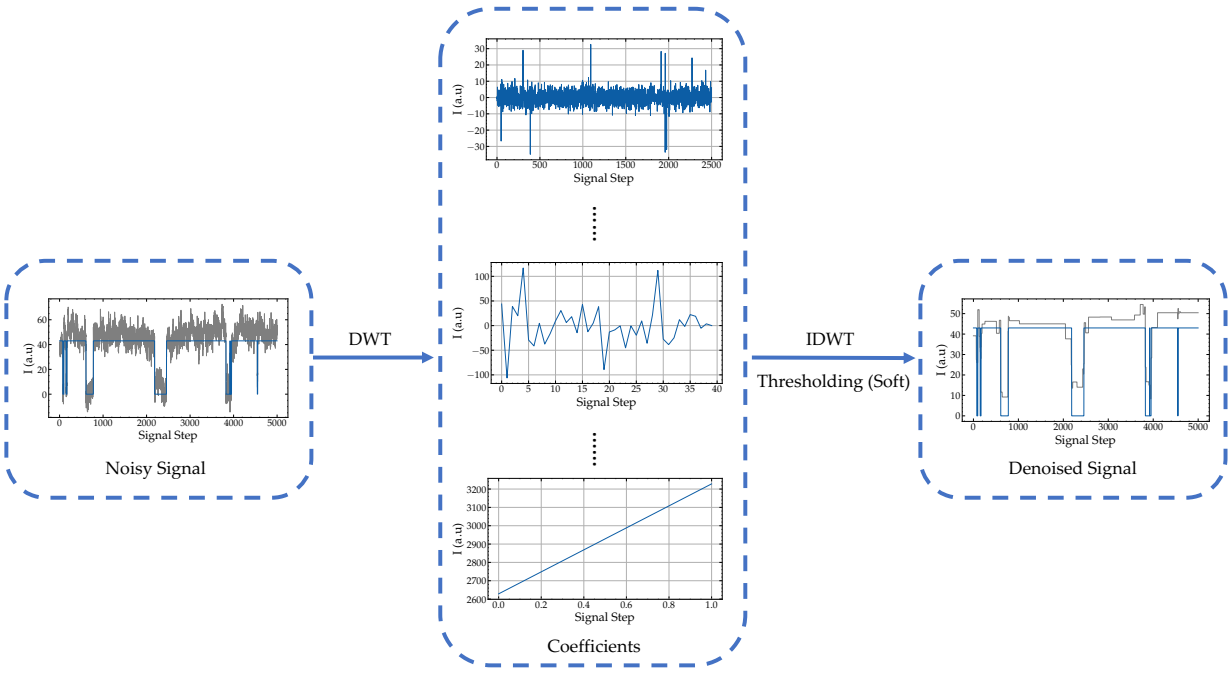


Figure 3.2: Discrete Wavelet Transform denoising flowchart

and a residue by iteratively decomposing the input signal. Row 7 to 19 of Algorithm 5 shows how IMFs are calculated. First, the local extrema in the input signal are found and divided into local maxima and local minima. The local maxima and local minima are interpolated with the curve-fitting method [28] to obtain the upper and lower envelopes, respectively. Then the IMFs candidate can be obtained by calculating the difference between the input signal and the mean value of the envelope. IMFs have to satisfy two characteristics:
1. The difference between extrema (maximum and minimum) and zero crossings must be no more than 1.
2. The mean value of the envelope defined by the local maxima and minima must be zero.
The characteristics of these two IMFs are also very important criteria to calculate IMFs. If the result of candidates does not meet these two characteristics, it will not be considered as one of the IMFs (row 12 to 19 in Algorithm 4). Row 15 to 20 are mainly to collect the results of decomposition based on different conditions. The algorithm will repeat row 7 to 20 as long as $r(t)$ is not a monotonic function and $r(t)$ and $m(t)$ have a high amplitude.

Fig.3.3 and Algorithm 5 show the complete denoising procedure of the EMD denoising adapted from [37]. After performing the above EMD processing on RTS, only IMFs that are greater than or equal to the level threshold are retained and sum them with the final residue $r(t)$ to obtain the denoised RTS. The level threshold ϵ is 5. This approach can effectively remove relatively high-frequency background noise components and retain relatively low-frequency RTS components.

3.5 Denoising Autoencoder Denoising

3.5.1 Convolutional Neural Networks

Convolutional Neural Network (CNN) is a neural network with convolution layers as the core module. CNN can very efficiently extract features of one-dimensional or even multi-dimensional signals and complete low-level and high-level tasks such as classification, detection, segmentation, and denoising. There are several typical layers that are widely applied in CNN to extract data. Convolutional layers, pooling layers, and transpose convolutional layers that are applied by DAE models will be introduced below.

Algorithm 4 Empirical Mode Decomposition

Input: Noisy signal $x(t)$
Output: Residue $r(t)$, IMFs = $\{\text{IMF}_1, \text{IMF}_2 \text{ ... } \text{IMF}_n\}$
procedure EMD($x(t)$)
    Initialize the residue $r(t)$ as an empty signal
    Initialize the composition level n = 1
    while r(t) is not a monotonic function and r(t) and m(t) have high amplitude **do**
        Find all local extrema values in $x(t)$ including local maxima and local minima
        Interpolate local maxima to obtain upper envelopes $u(t)$
        Interpolate local minima to obtain lower envelopes $l(t)$
        Calculate average of envelopes $m(t) = \frac{u(t)+l(t)}{2}$
        Calculate the difference between signal/residue and average of envelopes $d(t) = x(t) - m(t)$
        if $d(t)$ satisfies two IMF characteristics **then**
            $\text{IMF}_n \leftarrow d(t)$
            Add IMF_n to IMFs
            $n \leftarrow n + 1$
            $x(t) \leftarrow d(t)$
        else
            $x(t) \leftarrow d(t)$
        end if
        $r(t) \leftarrow r(t) + d(t)$
    end while
    return IMFs, $r(t)$
end procedure

Algorithm 5 EMD Denoising

Input Noisy RTS $x(t)$, Level threshold ϵ
Output Denoised RTS $x(t)_{denoised}$
procedure EMD DENOISING($x(t)$, ϵ)
    $IMFs, r(t) = EMD(x(t))$
    $IMFs = \{IMF_\epsilon, IMF_{\epsilon+1}, \text{ ... }, IMF_n\}$
    $x_{denoised} = sum(IMFs) + r(t)$
    return $x_{denoised}$
end procedure

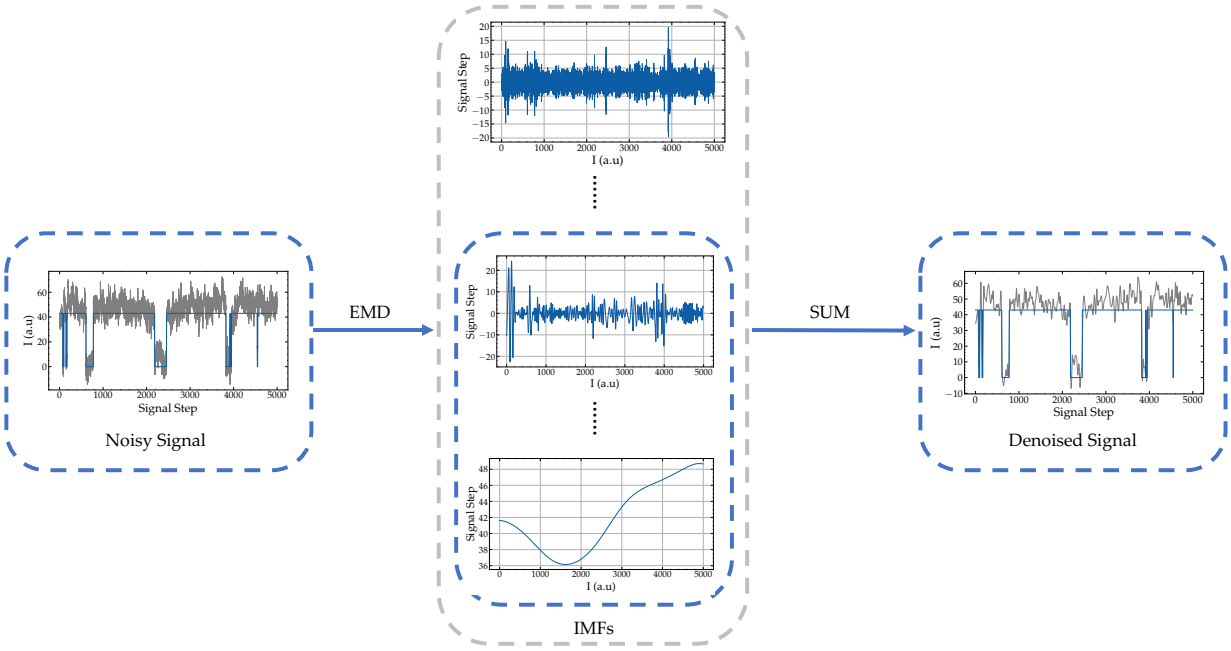


Figure 3.3: Empirical Mode Decomposition denoising flowchart

Convolutional Layers

Convolutional layers are the most important building blocks of CNNs, accounting for most of the parameters in CNNs. A convolutional layer contains several parameters, including input data, kernels, and feature maps. In signal processing, the input of the layer is a one-dimensional signal. In addition, the convolution layer also contains kernels, also known as filters. The kernel is a one-dimensional weight array. The length and width of kernels are usually the same odd numbers, and the kernel size also determines the size of the receptive field. Fig.3.4 shows an example of one-dimensional convolution. During convolution, the kernel slides at each input signal position according to a certain stride. In this example, the stride equals to 1. In step 1, the dot product of the input slice $[a, b, c]$ and the kernel $[\omega_1, \omega_2, \omega_3]$ is calculated as the first element of the feature map. Then the same operation is applied to the input slice $[b, c, d]$ in step 2 and obtains the second element of the feature map.

According to the above details of the convolutional layer, a convolutional layer can extract features from the input data. When multiple convolutional layers are applied to extract features from the input signal, different levels of features can be obtained for analysis. Additionally, convolutional layers are more efficient than fully connected

layers due to localized features and weight sharing. After each convolution operation, an activation function is applied to the feature maps. An activation function is a function that decides which neuron can be activated. It is usually a nonlinear function because it can introduce nonlinear components to the neural network to enhance its representation of complex features. The activation function applied in our model is rectified linear unit (ReLU). ReLU sets all negative values to 0 for output and outputs positive values directly to introduce nonlinearity effectively and discriminatively.

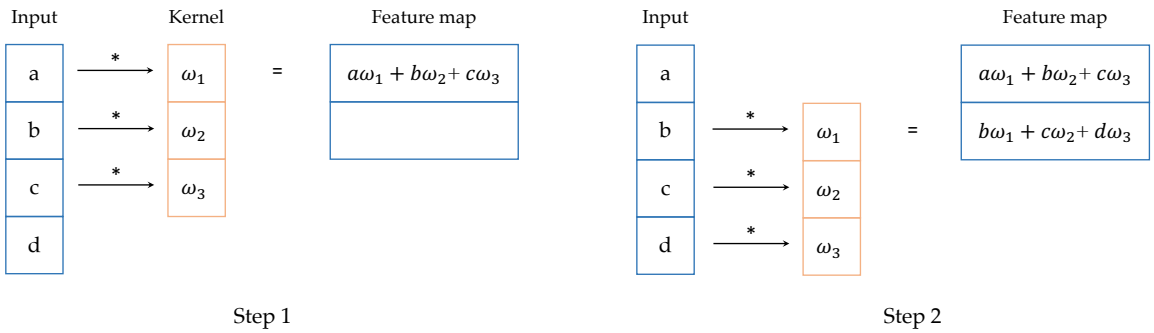


Figure 3.4: Example of one-dimensional convolution

Pooling Layers

Pooling layers are the layers responsible for dimensionality reduction in a CNN. Although the convolution layer has downsampled the input data to a certain extent, the convolution process still requires a lot of calculations for signal or image data, resulting in low network efficiency. Pooling layers apply sliding windows to move on the input data at a certain stride and perform geometric operations on the data in each window position to obtain results of dimensionality reduction. Therefore, pooling layers are insensitive to fluctuations and noise in the input data, which can improve the robustness of CNN.

The pooling layer applied in our models is the max pooling layer. Fig.3.5 shows an example of one-dimensional max pooling. The window size of the max pooling is 2. When the one-dimensional window slides to a certain position, the maximum value of the overlapped areas is the element at the corresponding position in the output.

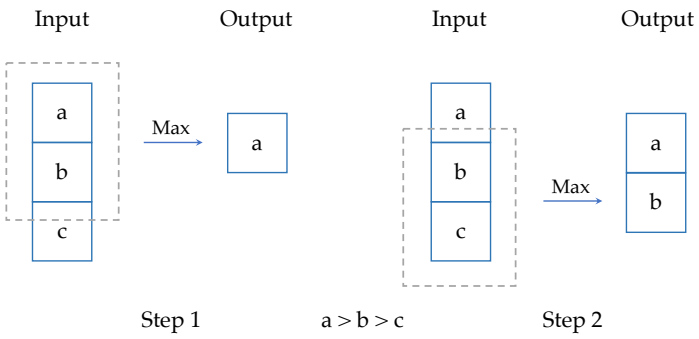


Figure 3.5: Example of one-dimensional max pooling

Transpose Convolutional Layers

In some tasks, such as denoising and segmentation, we want to upsample the feature map to an output of the same size as the input. Transpose convolutional layers are a kind of upsampling layer in CNN that can learn automatically. In signal processing, the input of the transposed convolutional layers is a one-dimensional feature map. Fig.3.6 shows an example of one-dimensional transpose convolution. In step 1, the dot products are calculated of the first element of the input a are all the kernel weights to obtain the output $[a\omega'_1, a\omega'_2, a\omega'_3]$. The same calculation is applied to element b and obtain $[b\omega'_1, b\omega'_2, b\omega'_3]$ in step 2. The two outputs are added with an offset of 1 to obtain a one-dimensional output $[a\omega'_1, a\omega'_2 + b\omega'_1, a\omega'_3 + b\omega'_2, b\omega'_3]$. The transpose convolution performs the reverse operation relative to the convolution. However, transpose convolution can only recover the size of the original input data and cannot really restore the input of the convolution operation because the intensities of the elements are different. Compared to other upsamplings with a fixed pattern, the transposed convolution can learn the optimal pattern for a specific task based on the training data.

3.5.2 Long Short-Term Memory

Long Short-Term Memory cell

Traditional recurrent neural network (RNN) often suffers from gradient vanishing after many recurrent steps, making them unable to capture long-term dependencies in the sequence. long short-term memory (LSTM) [25], an advanced RNN architecture, solves

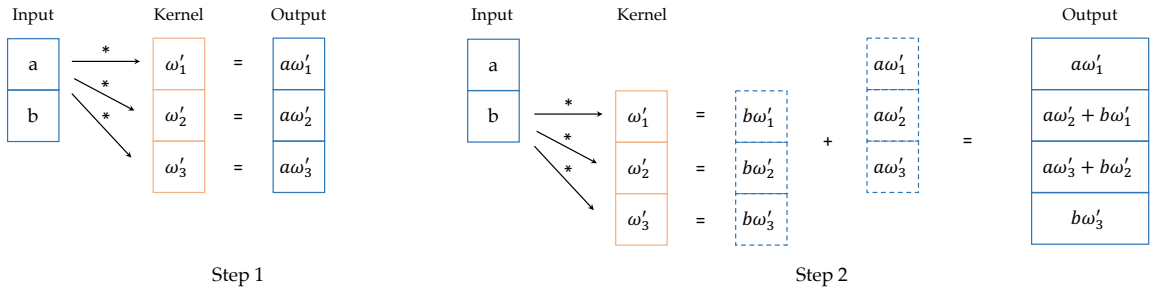


Figure 3.6: Example of one-dimensional transpose convolution

this problem by introducing gating mechanisms. As shown in Fig.3.7, an LSTM cell consists of three gates [40].

Forget Gate

The function of the forget gate is to determine how much information to forget from long-term memory $c(t-1)$ of the previous cell state. Forget Gate Activation $f(t)$ can be calculated as.

$$f_t = \sigma(W_f \cdot [x_t, h_{t-1}] + b_f) \tag{3.6}$$

where σ is the sigmoid activation function, $x(t)$ is the input sequence at signal step t, W_f is the weight of the forget gate, h_{t-1} is the hidden state at signal step $t-1$ and b_f is the bias of the forget gate.

Input Gate

The function of the input gate is to determine how much information to store from the current input $x(t)$ and the hidden state h_{t-1} of the previous cell state. Input gate activation $i(t)$ and the cell candidate activation $\tilde{c}(t)$ can be calculated as,

$$i_t = \sigma(W_i \cdot [x_t, h_{t-1}] + b_i) \tag{3.7}$$

$$\tilde{c}(t) = tanh(W_c \cdot [x_t, h_{t-1}] + b_c) \tag{3.8}$$

where *tanh* is the hyperbolic tangent activation function. After completing the above calculations, update cell state c(t) can be calculated as.

$$c(t) = f(t) * c(t-1) + i(t) * \tilde{c}(t) \tag{3.9}$$

Output Gate

The function of the output gate is to determine how much information to store from the update cell state $c(t)$ to generate the output $o(t)$ and the hidden state $h(t)$,

$$o(t) = \sigma(W_o \cdot [x_t, h_{t-1}] + b_o) \tag{3.10}$$

$$h(t) = o(t) + tanh(c(t)) \tag{3.11}$$

Through these three gates, LSTM cells can selectively forget past cell states and selectively memorize new inputs to better control information flow and capture long-term dependencies.

Bidirectional Long Short-Term Memory

Although LSTM has a good performance in time series prediction, it can only predict the output of the current signal step based on information from previous signal steps. In practical time series analysis, the output at the current signal step may still be related to the information at the future signal steps. To enable LSTM to capture the time dependencies bidirectionally, bidirectional long short-term memory (Bi-LSTM) was proposed based on the architecture of [51]. Fig.3.8 shows the architecture of Bi-LSTM. Bi-LSTM consists of a forward layer and a backward layer. The forward layer takes the starting input of the sequence from the left, and the backward layer takes the starting input of the sequence from the right. The output of Bi-LSTM is the concatenation of the output of the backward layer and the output of the forward layer. For example, when the signal step is equal to t, $y(t) = concat(h'(0), h(t))$. In summary, Bi-LSTM can obtain more comprehensive and accurate sequence analysis results by processing sequences bidirectionally. Therefore, one of our models also introduces Bi-LSTM to improve the performance of the DAE model.

3.5.3 Denoising Autoencoder

DAE, a type of unsupervised artificial neural network, was proposed for signal and image denoising [11][15][59]. Fig.3.9 shows the architecture of the general DAE. A DAE comprises an encoder and a decoder. During the training process, DAE first encodes the noisy signal through the encoder into a low-dimensional latent feature vector and then decodes the vector into a denoised signal through the decoder. Hendrickson et al.

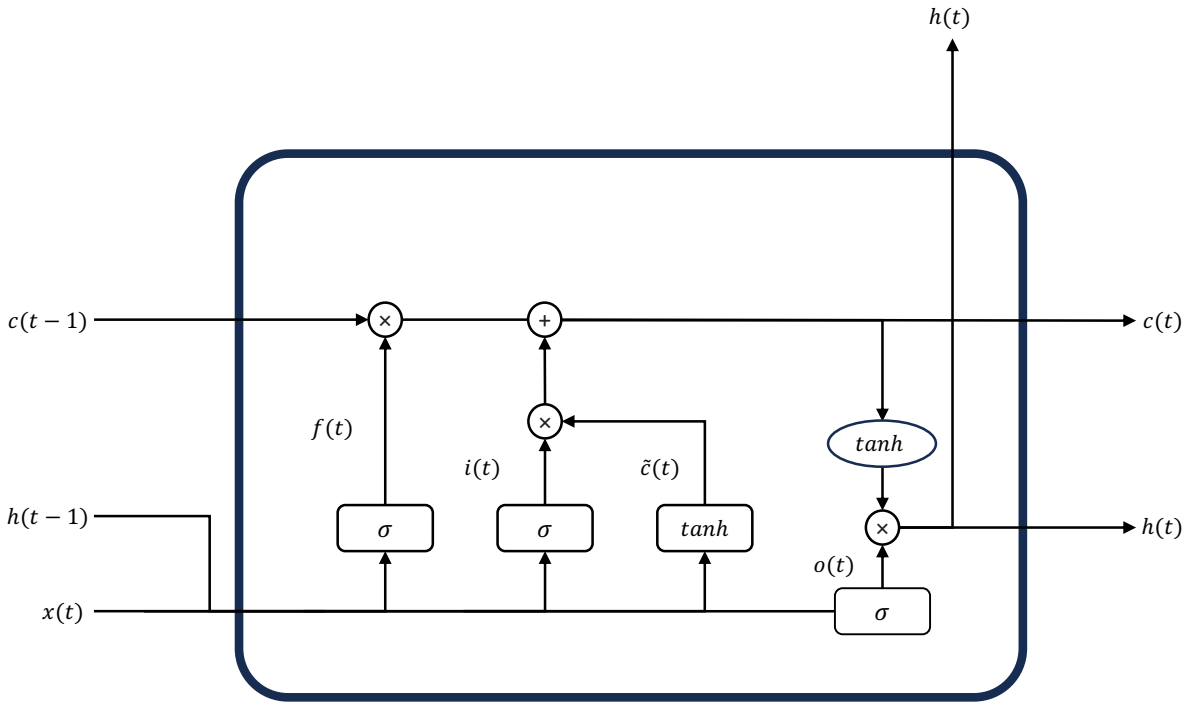


Figure 3.7: Architecture of an LSTM cell

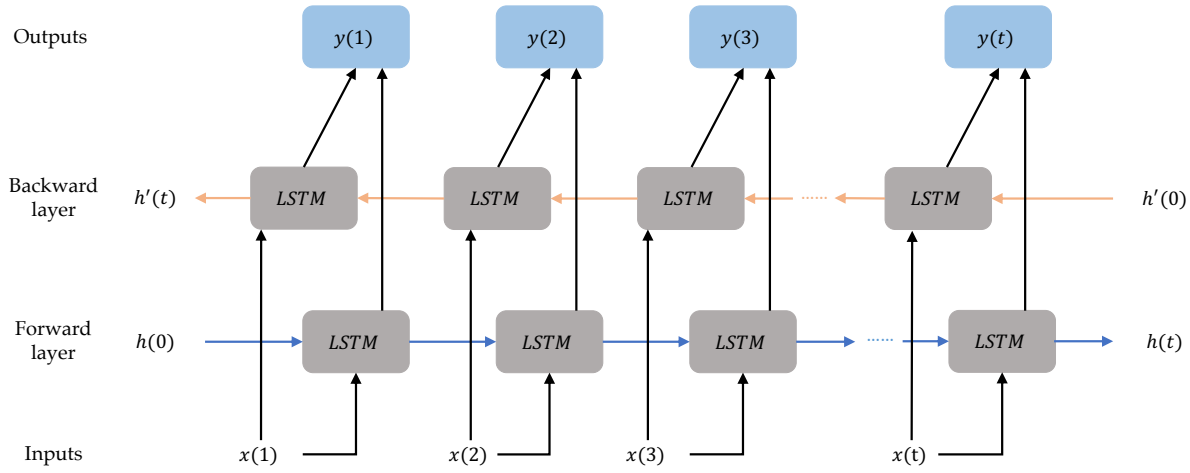


Figure 3.8: Architecture of Bi-LSTM

proposed a DAE model to denoise dark current RTS (DC-RTS) [24]. After training, the model can automatically denoise RTSs with different transition amplitude and duration. However, this model has only been proven to be capable of performing extraction for two-level RTSs with low noise levels. In this thesis, two DAE models are proposed to enable DAE to denoise multi-level RTSs.

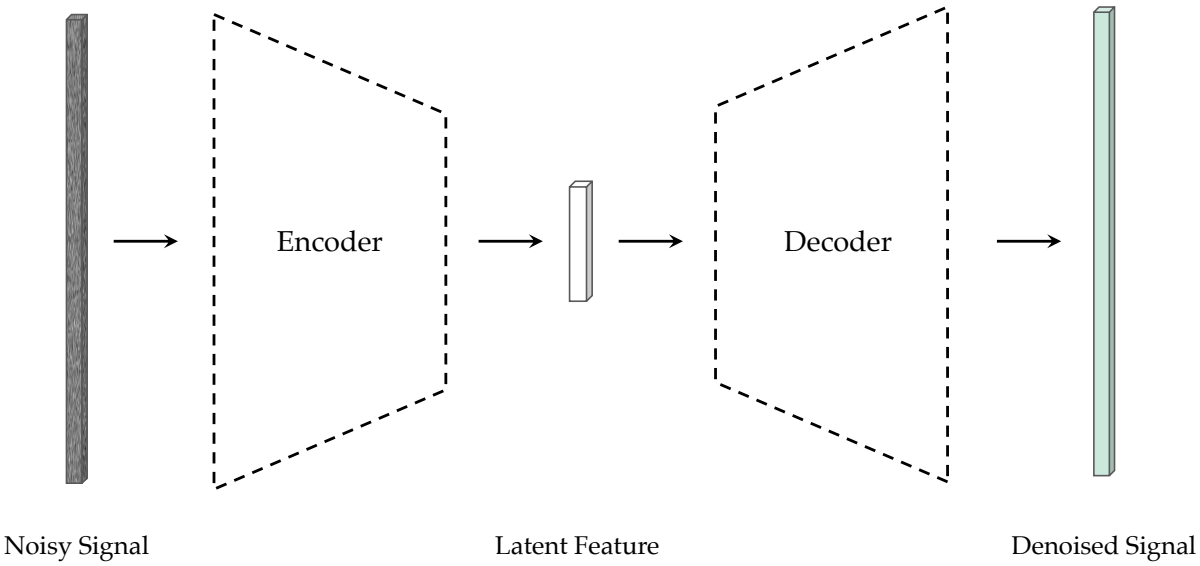


Figure 3.9: Architecture of general DAE

3.5.4 Denoising Autoencoder based on Fully Connected Networks

Denoising autoencoder based on fully connected networks (DAE FCN) is the model adapted from [24], which follows the basic framework of DAE and is also based on fully connected networks (FCN). A fully convolutional network is a type of CNN. Different from traditional CNN, which will connect several fully connected layers after a series of convolutional layers, the fully convolutional network uses a locally connected convolutional layer instead of a fully connected layer with a large number of parameters. Therefore, DAE based on fully convolutional networks are more efficient and lightweight than DAE with fully connected layers.

The architecture of DAE FCN is shown in Fig.3.10. All convolutional and transpose convolutional layers apply 1*15 kernels and ReLU as activation functions. Zero padding

is applied in all convolutional and transposed convolutional layers to keep input and output the same size.

The structure of the encoder is $Conv1D(64) \rightarrow MaxPooling1D(2) \rightarrow Conv1D(32) \rightarrow MaxPooling1D(2)$. The channel numbers of $Conv1D$ are 32 and 64. The kernel size of $MaxPooling1D$ is 2. Passing through two encoding modules consisting of a $Conv1D$ layer and a $MaxPooling1D$ layer, a 1*2000*1 noisy signal is encoded into a 1*500*32 latent feature.

The structure of the decoder is $TransposeConv1D(32) \rightarrow Conv1D(32) \rightarrow TransposeConv1D(64) \rightarrow Conv1D(64) \rightarrow Conv1D(1)$. The channel numbers of $Conv1D$ and $TransposeConv1D$ are 32 and 64. The stride of $TransposeConv1D$ is 2. Passing through two decoding modules consisting of a $TransposeConv1D$ layer and a $Conv1D$ layer, the 1*500*32 latent feature is decoded into a 1*2000*1 denoised signal.

The loss function that DAE FCN applies is mean squared error (MSE). MSE can be calculated as.

$$MSE = \frac{1}{N}\sum_{n=1}^{N}(x_n - \hat{x}_n)^2 \tag{3.12}$$

where N is the signal length, x_i is pure signal value in signal step n and $\hat{x}_i$ is denoised signal value in signal step n. DAE FCN learns weights through gradient descent and backpropagation to make MSE as small as possible.

3.5.5 Denoising Autoencoder based on U-Net and Bi-LSTM

Denoising autoencoder based on U-Net and Bi-LSTM (DAE UBL) is an innovative DAE for RTS denoising. The general architecture of DAE UBL is based on U-Net [49]. U-Net is a classic fully convolutional network that is mainly used for semantic segmentation, signal denoising, and image denoising. DAE UBL applies the skip connection of U-Net. These skip connections help preserve low-level features of the encoder and concatenate them with the upsampled features of the decoder to enhance training. In addition, DAE UBL adds two layers of Bi-LSTM between the encoder and the decoder. These Bi-LSTMs play a significant role in effectively capturing bidirectional time dependencies from the signal so that DAE UBL can apply past and future information during denoising.

The architecture of DAE UBL is shown in Fig.3.11. All convolutional and transpose convolutional layers apply 1*3 kernels and ReLU as activation functions. Zero padding is applied in all convolutional and transposed convolutional layers to keep input and output the same size.

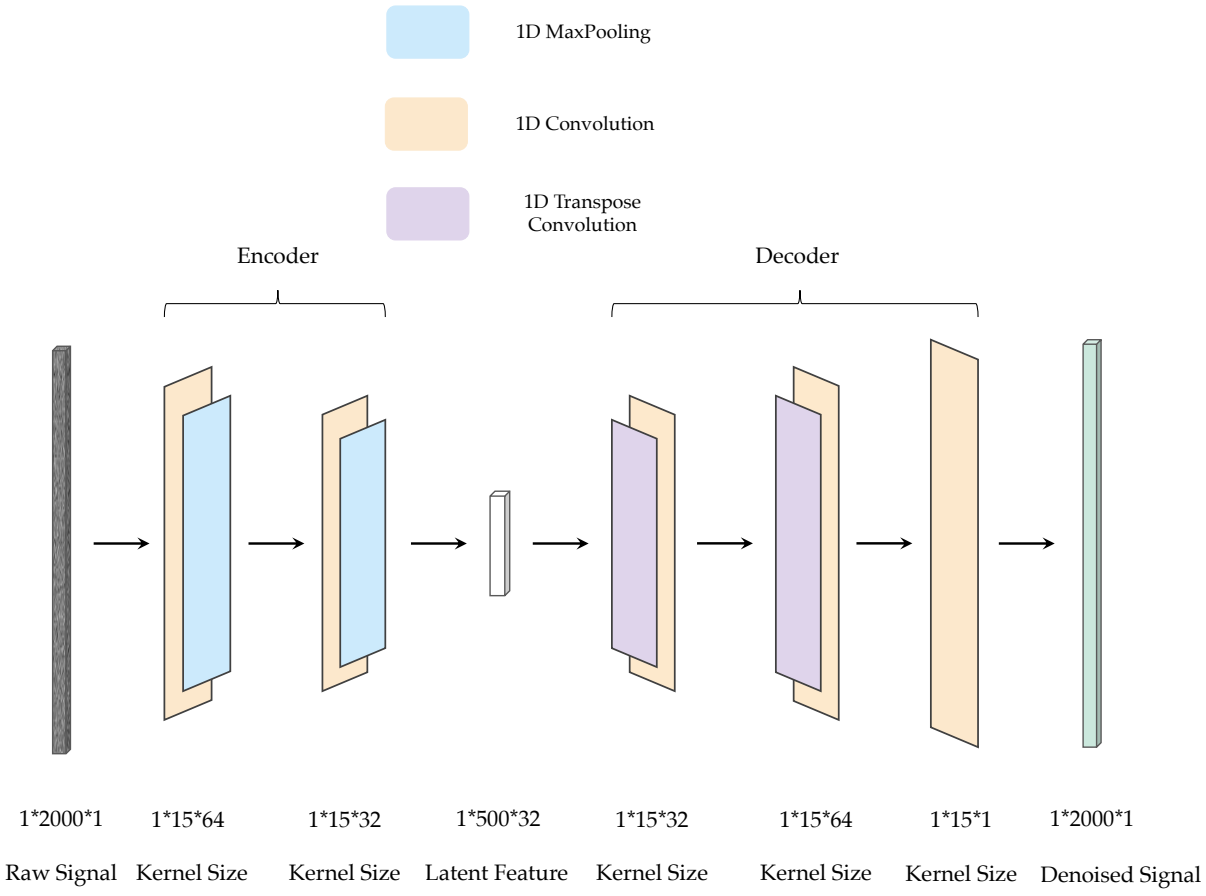


Figure 3.10: Architecture of DAE FCN

The structure of the encoder is $Conv1D(16) \rightarrow MaxPooling1D(2) \rightarrow Conv1D(32) \rightarrow MaxPooling1D(2) \rightarrow Conv1D(64) \rightarrow MaxPooling1D(2)$. The channel numbers of $Conv1D$ are 16, 32, and 64. The kernel size of $MaxPooling1D$ is 2. Passing through the encoder, a 1*2000*1 noisy signal is encoded into a 1*250*64 latent feature. Two layers of Bi-LSTMs process the latent feature and generate a 1*250*256 hidden state. Consider three convolutional layers output in the encoder as $conv1 = Conv1D(16)$, $conv2 = Conv1D(32)$, and $conv3 = Conv1D(64)$. The structure of the decoder is $TransposeConv1D(64) \rightarrow concat(conv3, Conv1D(64)) \rightarrow TransposeConv1D(32) \rightarrow concat(conv2, Conv1D(32)) \rightarrow TransposeConv1D(16) \rightarrow concat(conv1, Conv1D(16)) \rightarrow Conv1D(1)$. $concat$ in the structure means the concatenation with convolutional layers output of the encoder and the decoder. The channel numbers of $Conv1D$ and $TransposeConv1D$ are 64, 32, and 16. The stride of $TransposeConv1D$ is 2. Passing through the decoder, the 1*250*256 hidden state is decoded into a 1*2000*1 denoised signal. The loss function that DAE UBL applies is also MSE.

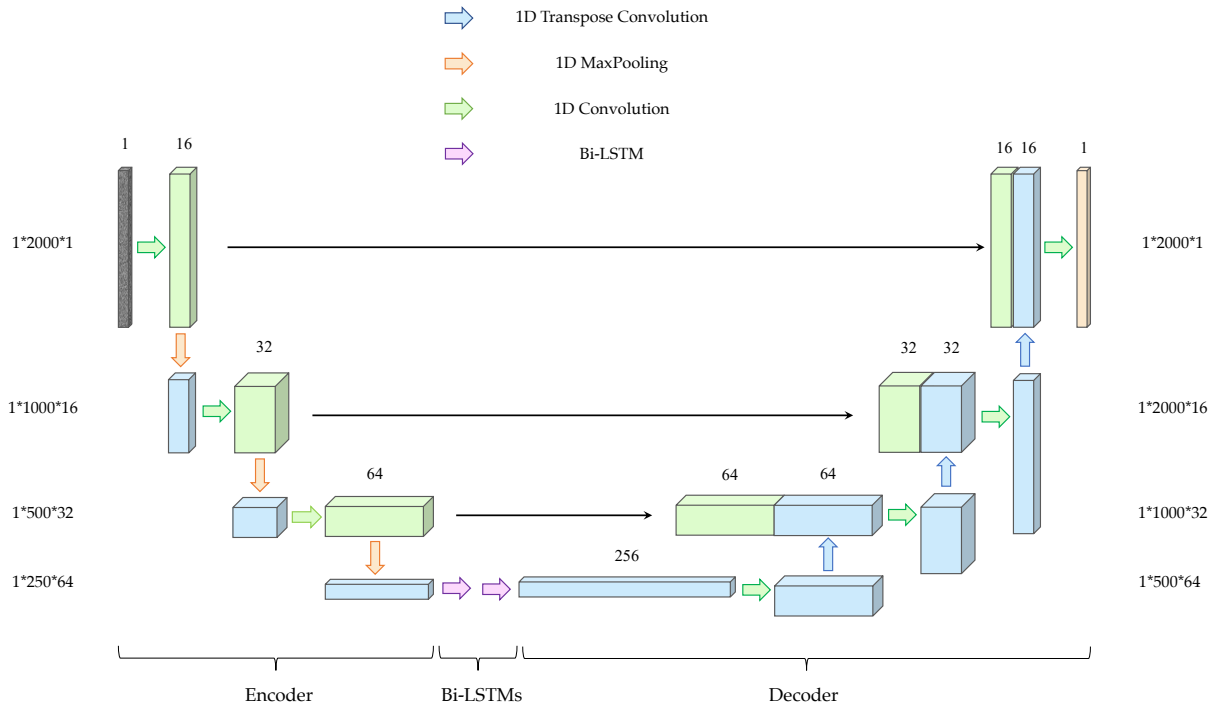


Figure 3.11: Architecture of DAE UBL

Chapter 4

RTS Denoising Experiments and Analysis

4.1 Dataset

Synthetic datasets for RTS denoising are generated according to Eq.2.11. Statistics of the synthetic training dataset for RTS denoising are shown in Table 4.1. The training dataset includes three types of nRTSs and three types of aRTSs with Gaussian white noise or pink noise. All RTSs in the dataset have a fixed signal length of 2,000 signal steps. Each RTS category in the dataset has 12,000 samples. The distribution of synthetic RTSs is Gaussian distribution. The statistics of a RTS category in the synthetic training dataset are shown in Table 4.2. Each noise level range in {1-10%, 11-20%, 21-30%, 31-40%, 41-50%, 51-60%} has 2000 samples.

Statistics of the synthetic evaluation dataset for RTS denoising are shown in Table 4.3. The evaluation dataset includes three categories of nRTSs and three categories of aRTSs with Gaussian white noise or pink noise. All RTSs in the dataset have a fixed signal length of 1,000,000 signal steps. Each RTS category in the dataset has 60 samples. The distribution of synthetic RTSs is Gaussian distribution. Statistics of a RTS category in the synthetic evaluation dataset are shown in Table 4.4. Each noise level range from {1-10%, 11-20%, 21-30%, 31-40%, 41-50%, 51-60%} has ten samples.

RTS category	Noise type	Signal length	Samples number
1-trap	white noise	2,000	12,000
1-trap	pink noise	2,000	12,000
2-trap	white noise	2,000	12,000
2-trap	pink noise	2,000	12,000
3-trap	white noise	2,000	12,000
3-trap	pink noise	2,000	12,000
3-trap	white noise	2,000	12,000
coupled	pink noise	2,000	12,000
coupled	white noise	2,000	12,000
missing-level	pink noise	2,000	12,000
missing-level	white noise	2,000	12,000
metastable	pink noise	2,000	12,000
metastable	white noise	2,000	12,000

Table 4.1: Statistics of the synthetic training dataset

Noise level	Signal length	Samples number
1-10%	2,000	2,000
11-20%	2,000	2,000
21-30%	2,000	2,000
31-40%	2,000	2,000
41-50%	2,000	2,000
51-60%	2,000	2,000

Table 4.2: Statistics of an RTS category in synthetic training dataset

RTS category	Noise type	Signal length	Samples number
1-trap	white noise	1,000,000	60
1-trap	pink noise	1,000,000	60
2-trap	white noise	1,000,000	60
2-trap	pink noise	1,000,000	60
3-trap	white noise	1,000,000	60
3-trap	pink noise	1,000,000	60
3-trap	white noise	1,000,000	60
coupled	pink noise	1,000,000	60
coupled	white noise	1,000,000	60
missing-level	pink noise	1,000,000	60
missing-level	white noise	1,000,000	60
metastable	pink noise	1,000,000	60
metastable	white noise	1,000,000	60

Table 4.3: Statistics of the synthetic evaluation dataset

Noise level	Signal length	Samples number
1-10%	1,000,000	10
11-20%	1,000,000	10
21-30%	1,000,000	10
31-40%	1,000,000	10
41-50%	1,000,000	10
51-60%	1,000,000	10

Table 4.4: Statistics of an RTS category in synthetic evaluation dataset

4.2 Experiment Details

In the experiment, the synthetic training dataset is applied to train DAE models. Then trained DAE models and classical denoising models are evaluated in the synthetic evaluation dataset. The metrics for evaluation include root mean squared error (RMSE), signal-to-noise ratio (SNR), and visual analysis. The experiment aims to demonstrate the effectiveness of the DAE models for RTS denoising.

There are no trainable parameters for classical denoising models, including MA filtering, FFT denoising, DWT denoising, and EMD denoising. The trainable parameters of DAE models (DAE FCN and DAE UBL) are shown in Table 4.5. According to the table, it is obvious that DAE FCN has fewer trainable parameters than DAE UBL.

Model	Trainable parameters
DAE UBL	680,609
DAE FCN	155,809

Table 4.5: Trainable parameters of DAE denoising

The DAE models were trained on servers equipped with Titan Xp GPU and 24 to 28 CPU cores.

For training details, the training and validation split is 80% and 20%. The batch size of the training is 1000. The optimizer of the models is Adam with learning rate = 0.001, β_1=0.9, β_2=0.999, and ϵ=10^{-8} [29].

4.3 Root Mean Squared Error

To measure the difference between the denoised signal and the pure signal, the metric called RMSE is applied for evaluation. RMSE can be calculated as.

$$RMSE = \sqrt{MSE} \tag{4.1}$$

The calculation of MSE can be found in Eq. 3.12.

Boxplots (Fig. 4.1 to Fig. 4.6) are applied to display the distributions of RMSEs for six different denoising models with Q_{wn} or Q_{pn}. The distributions of RMSEs in the synthetic evaluation dataset are represented by quartiles.

For noise level Q_{wn} or Q_{pn}, as the noise level increases, RMSEs for denoising models increase generally. This is because as the noise level increases, the amplitude of the background noise becomes larger compared to that of the pure signal.

For background noise types, RMSEs of all denoising models in RTSs with pink noise have higher quartiles and variance than those in RTSs with Gaussian white noise in general. This trend is reflected explicitly in the boxplots of pink noise RTSs, where the boxes are placed higher, and the lengths of the boxes are longer when Q_{pn} is higher.

For model comparisons, two DAE models significantly improve denoising performance when processing RTSs with Gaussian white noise than pink noise. For white noise cases, RMSEs of the DAE models are usually below 5, while those of the classical models generally fluctuate between 0 and 40. For pink noise cases, RMSEs of the DAE models generally fluctuate between 0 and 40, and those of the classical models generally fluctuate between 0 and 50. It is worth noting that in DAE models, RMSEs and their variance for DAE UBL are lower than those of DAE FCN, especially when denoising RTSs with pink noise. The fluctuation range of RMSEs for DAE UBL is 0 to 30, 25% less than those of DAE FCN.

4.4 Signal-to-noise Ratio

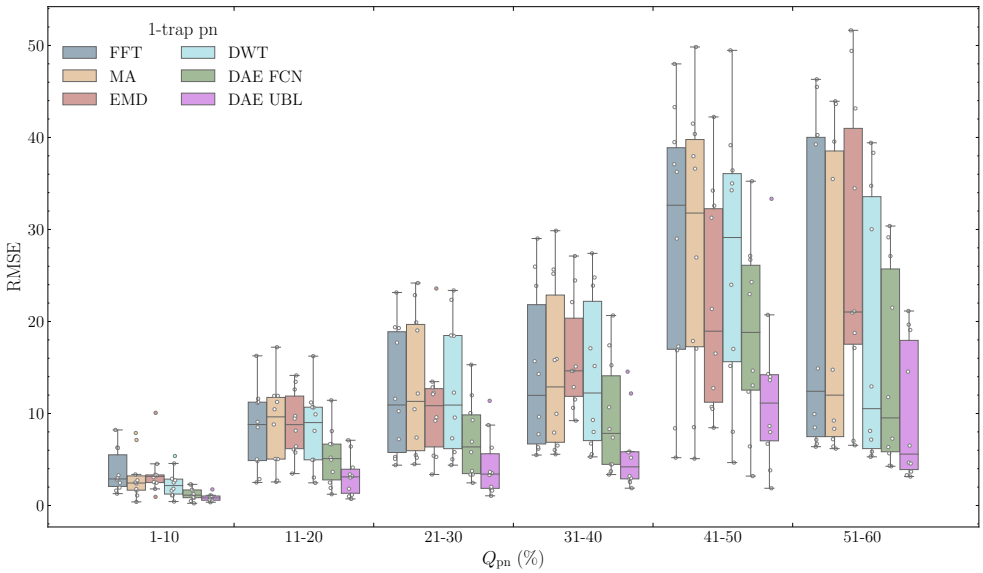
SNR is a metric for measuring the ratio between pure signal power and background noise power. It can be calculated as,

$$SNR = 10 \cdot \log_{10}\left(\frac{\sum_{n=1}^{N} x_n^2}{\sum_{n=1}^{N}(\hat{x}_n - x_n)^2}\right) \tag{4.2}$$

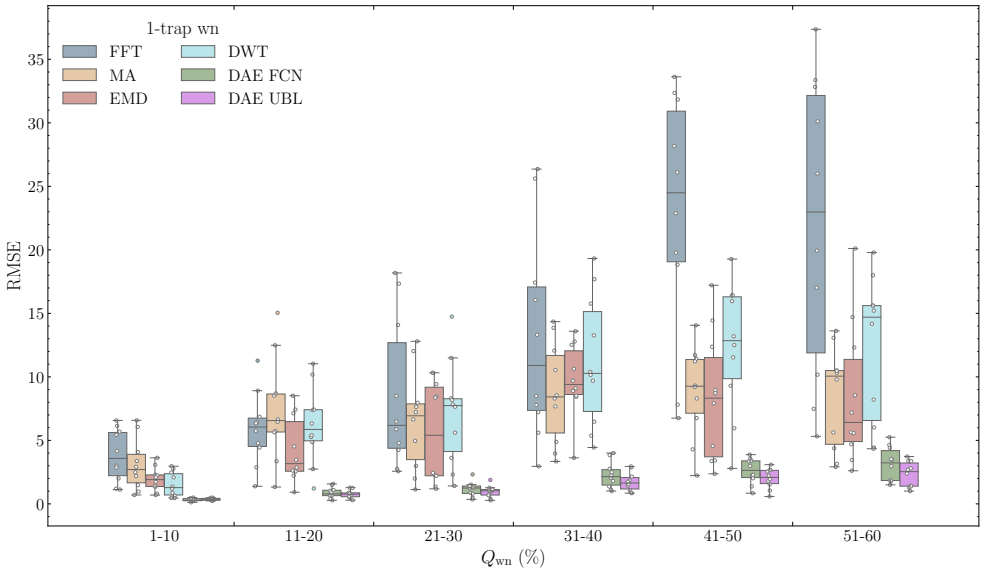
where x_n is the pure signal intensity of the signal step n, $\hat{x}_n$ is the denoised signal intensity of the signal step n, and N is the signal length. In signal quality assessment, we hope that the power of the pure signal can be much greater than the noise power. Therefore, the larger the SNR, the better the denoising effect of the signal.

Boxplots (Fig. 4.7 to Fig. 4.12) are applied to present the distributions of SNRs from six different denoising models on Q_{wn} or Q_{pn}. The distributions of SNRs in the synthetic evaluation dataset are represented by quartiles.

For noise level Q_{wn} or Q_{pn}, as the noise level increases, SNRs for denoising models decrease generally. This is because as the noise level increases, the amplitude of the

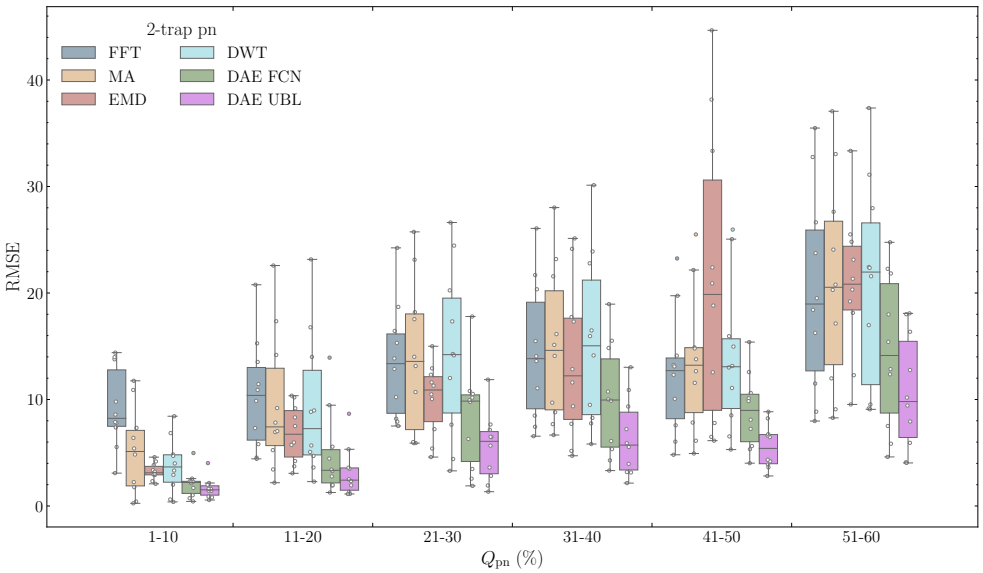


(a) RMSEs of 1-trap RTSs with pink noise

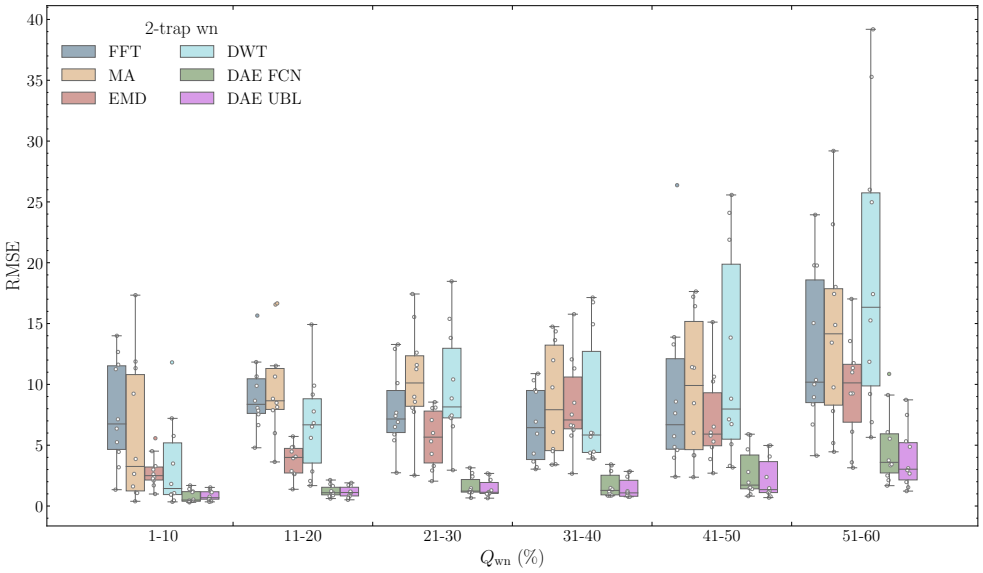


(b) RMSEs of 1-trap RTSs with Gaussian white noise

Figure 4.1: RMSEs of 1-trap RTSs

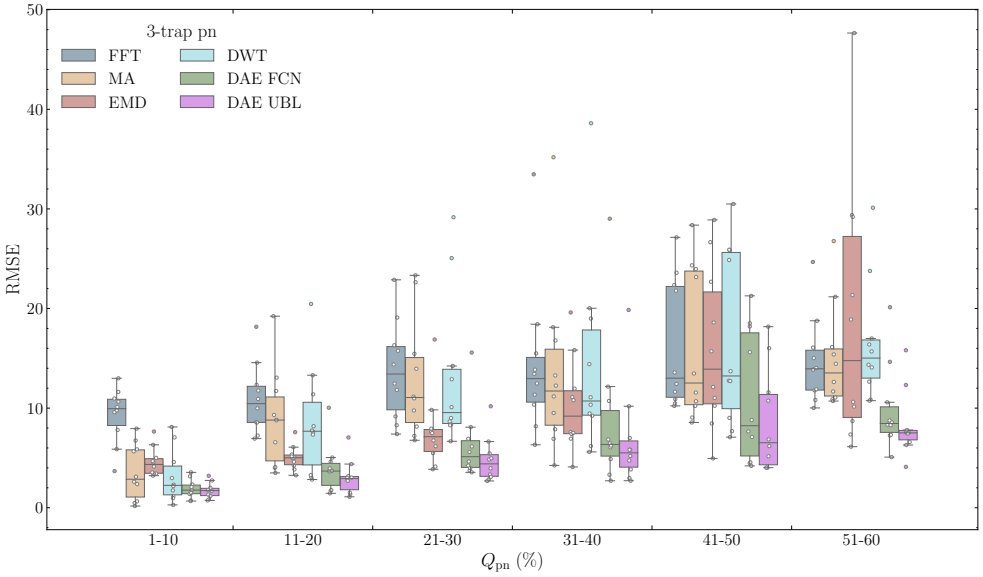


(a) RMSEs of 2-trap RTSs with pink noise

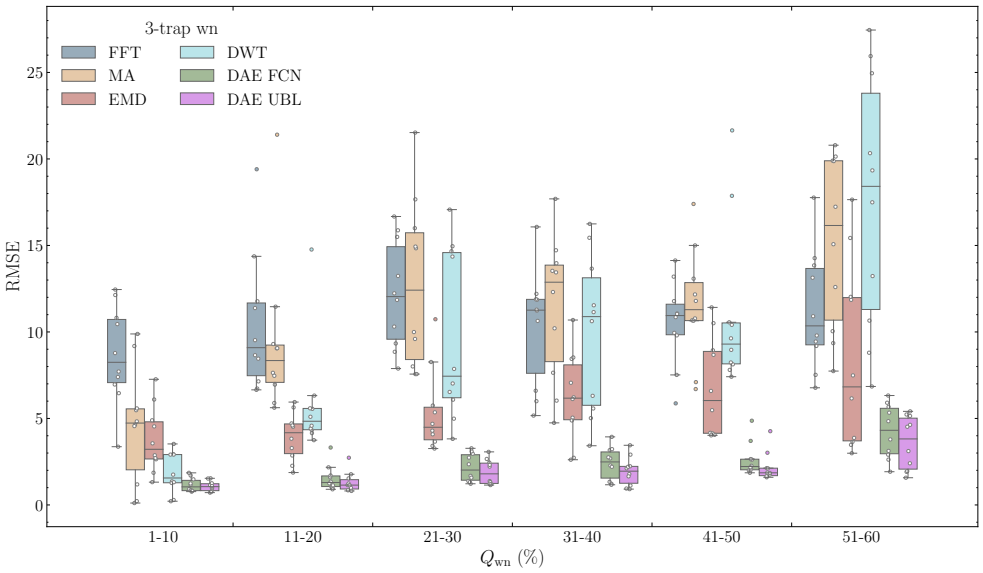


(b) RMSEs of 2-trap RTSs with Gaussian white noise

Figure 4.2: RMSEs of 2-trap RTSs

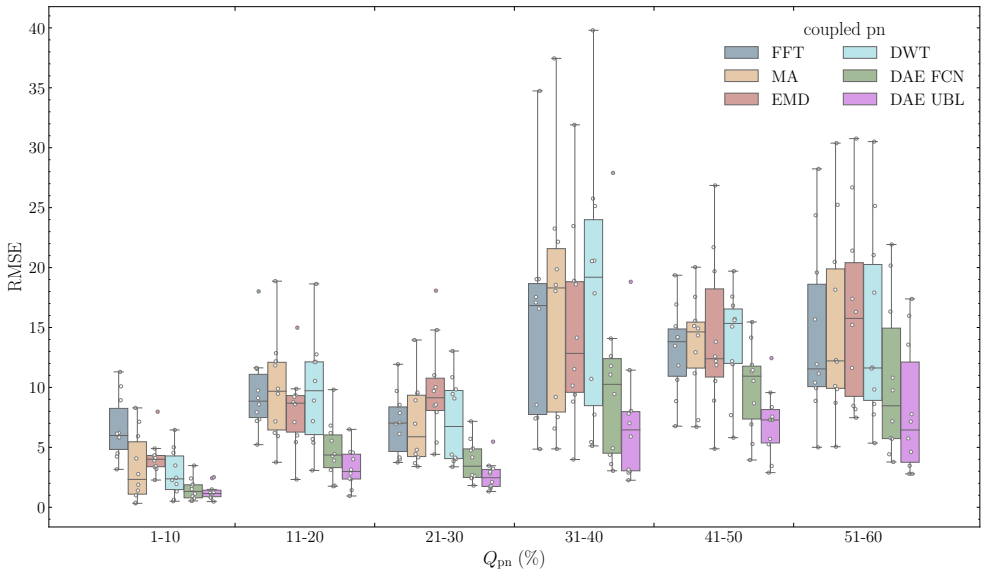


(a) RMSEs of 3-trap RTSs with pink noise

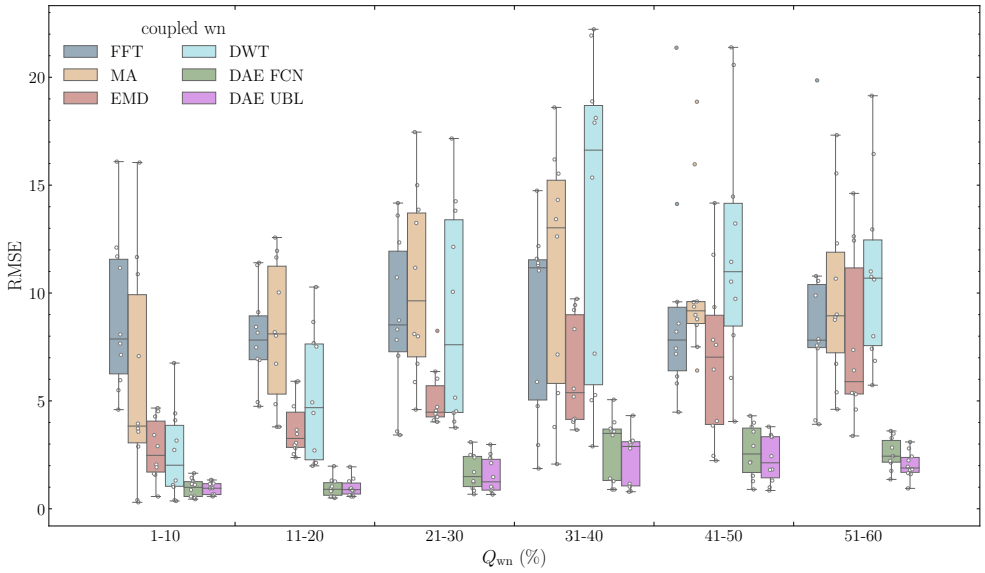


(b) RMSEs of 3-trap RTSs with Gaussian white noise

Figure 4.3: RMSEs of 3-trap RTSs

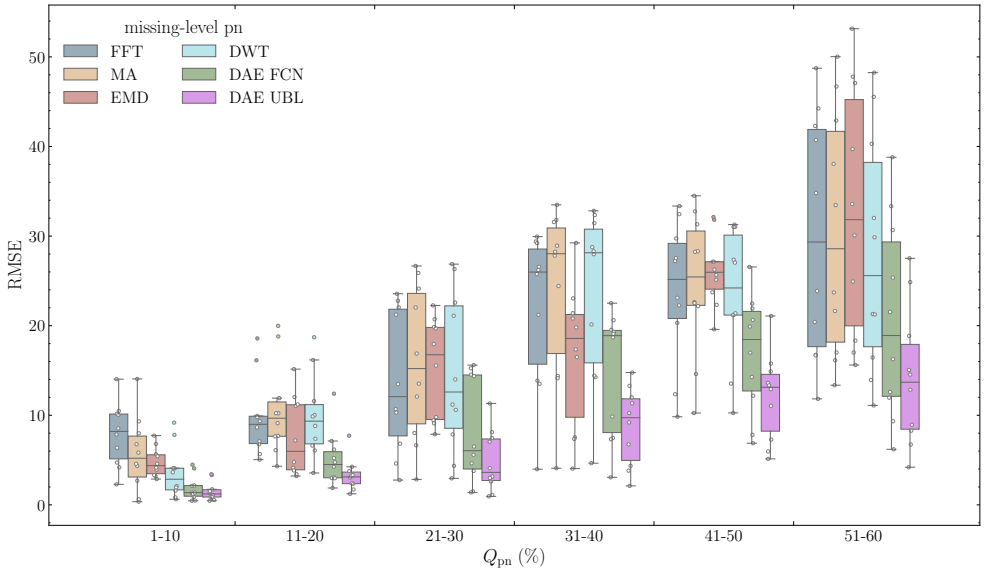


(a) RMSEs of coupled RTSs with pink noise

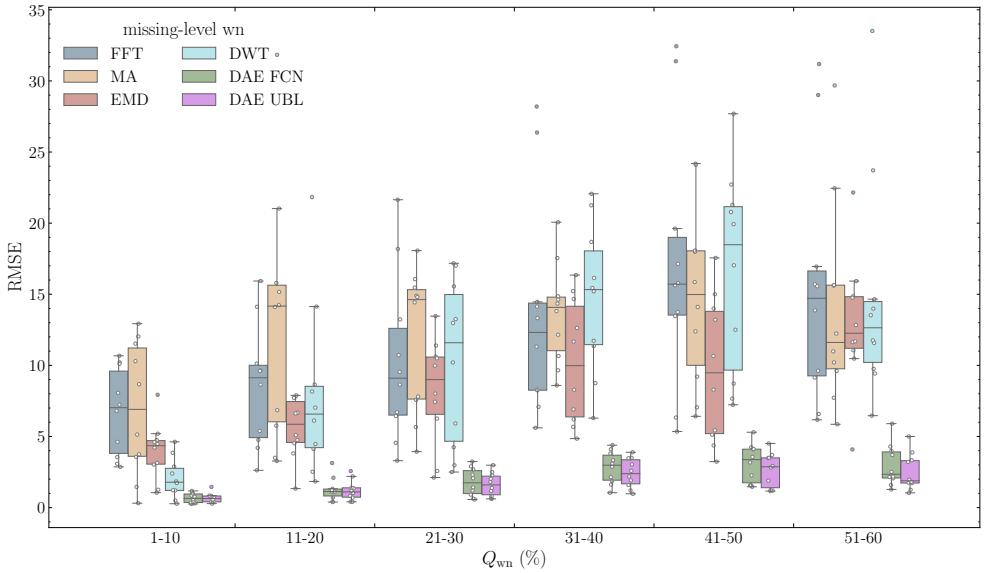


(b) RMSEs of coupled RTSs with Gaussian white noise

Figure 4.4: RMSEs of coupled RTSs

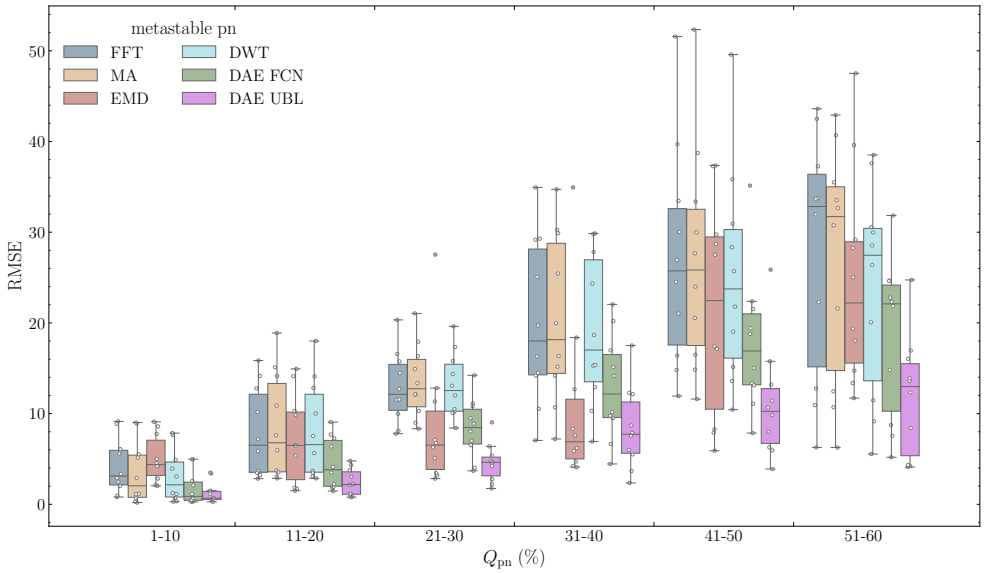


(a) RMSEs of missing-level RTSs with pink noise

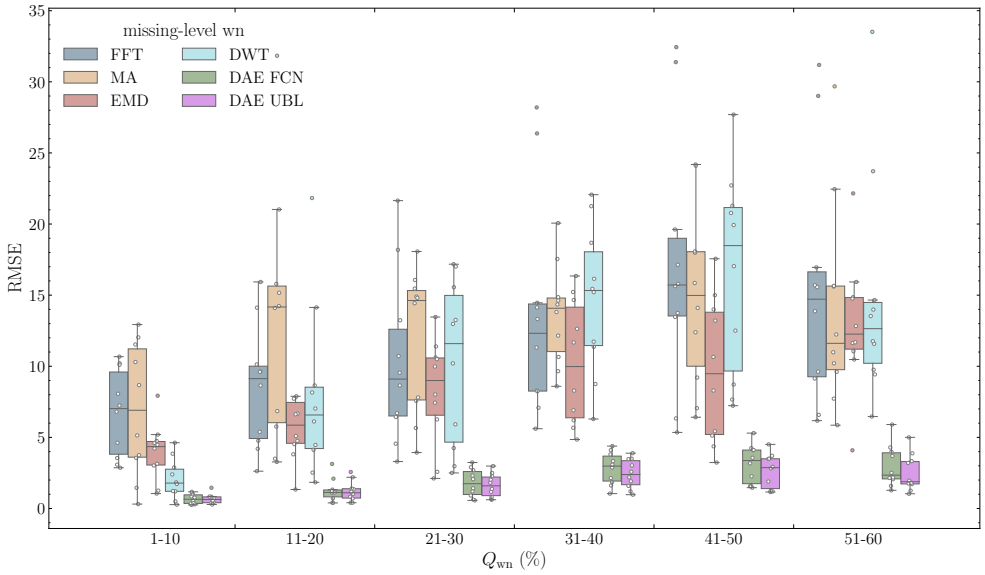


(b) RMSEs of missing-level RTSs with Gaussian white noise

Figure 4.5: RMSEs of missing-level RTSs



(a) RMSEs of metastable RTSs with pink noise



(b) RMSEs of metastable RTSs with Gaussian white noise

Figure 4.6: RMSEs of metastable RTSs

background noise becomes larger compared to that of the pure signal, which leads to increasing SNRs.

For background noise types, SNRs of the denoising models in RTSs with pink noise have lower quartiles and higher variance than those in RTSs with Gaussian white noise in general. Reflected explicitly in the boxplots of pink noise RTSs, the position of the boxes is lower, and the length of the boxes is longer. Additionally, Q_{pn} has a greater impact on SNRs than Q_{wn}, which is specifically manifested in the decrease of SNRs with the increase of Q_{pn} is significantly more obvious than those with the increase of Q_{wn}.

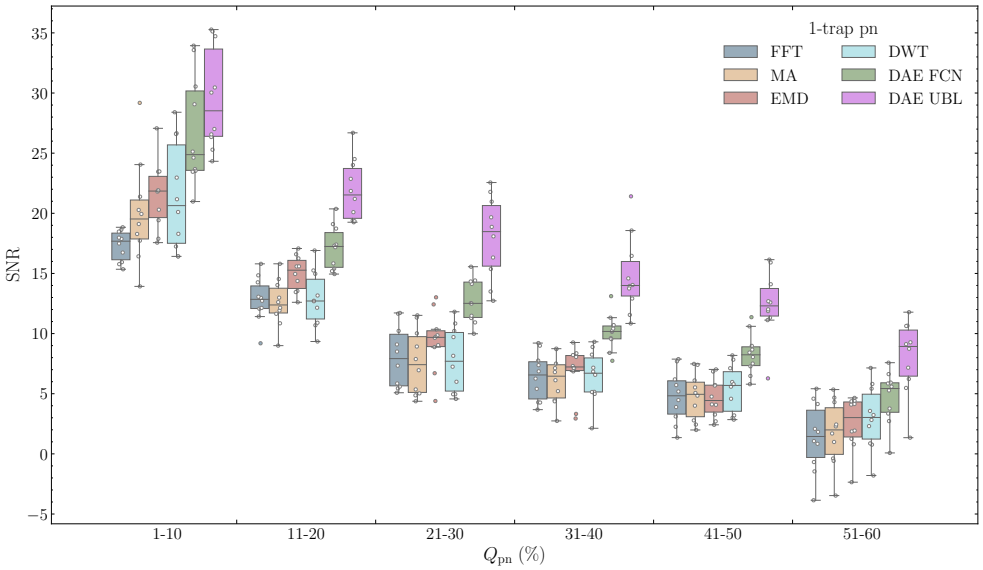
For model comparisons, two DAE models significantly improve denoising performance when processing RTSs with Gaussian white noise than pink noise. For Gaussian white noise cases, SNRs of the DAE models generally fluctuate between 20 and 40, while those of the classical models generally fluctuate between 0 and 50. For pink noise cases, SNRs of the DAE models generally fluctuate between 0 and 45, and those of the classical models fluctuate between -5 and 50. Additionally, DAE models, SNRs and their variance for DAE UBL are lower than those of DAE FCN, especially when denoising RTSs with pink noise. The fluctuation range of RMSEs for DAE UBL is at least 4% less than those of DAE FCN.

4.5 Denoising Visualization Analysis

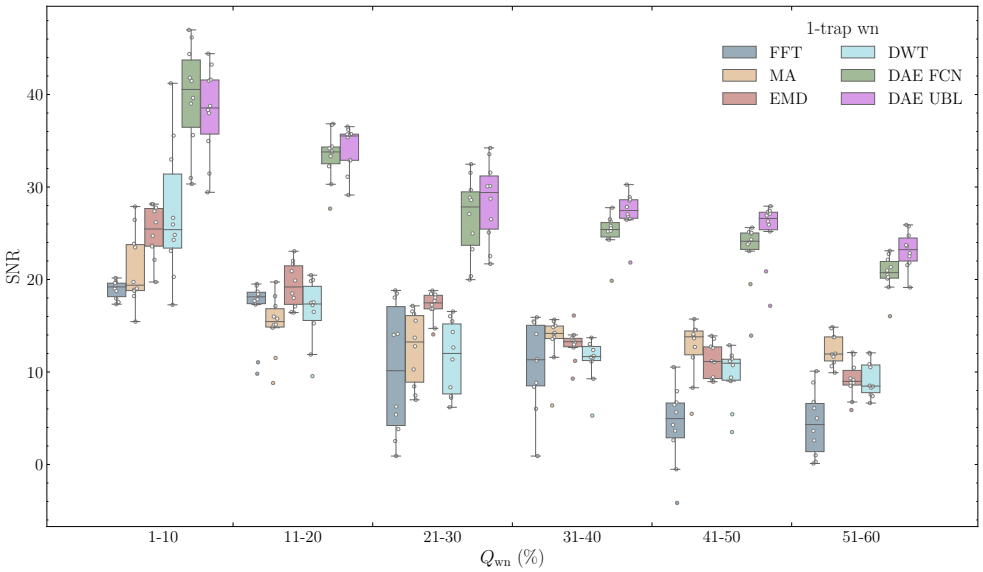
To further demonstrate the superiority of DAE models, the denoising visualization results are provided in Fig. 4.13 to Fig. 4.24. Each figure displays the denoising results of three denoising models (MA filtering, DAE FCN, and DAE UBL) dealing with a specific RTS category when Q_{wn} equals to 20% or Q_{pn} equals to 60%. Each plot includes a raw signal (gray), a denoised signal (green), and a pure signal (blue) for comparison.

For background noise types, for the denoising visualization results of RTSs with Gaussian white noise

1. Gaussian white noise does not cause level fluctuations in the general trend of RTSs (according to 2.4), so the three denoising results do not contain fluctuations in the general trend.
2. The local fluctuations caused by Gaussian white noise are suppressed very well by three denoising models, so the denoising results are also very close to pure signals. Therefore, the denoising visualization results of RTSs with Gaussian white noise may not reflect some of the patterns that will be analyzed next.

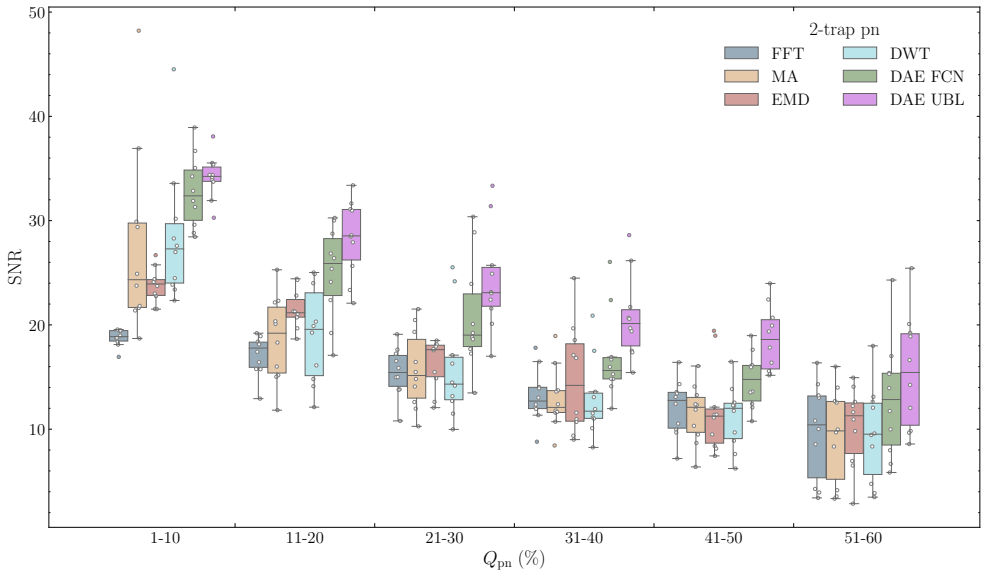


(a) SNRs of 1-trap RTSs with pink noise

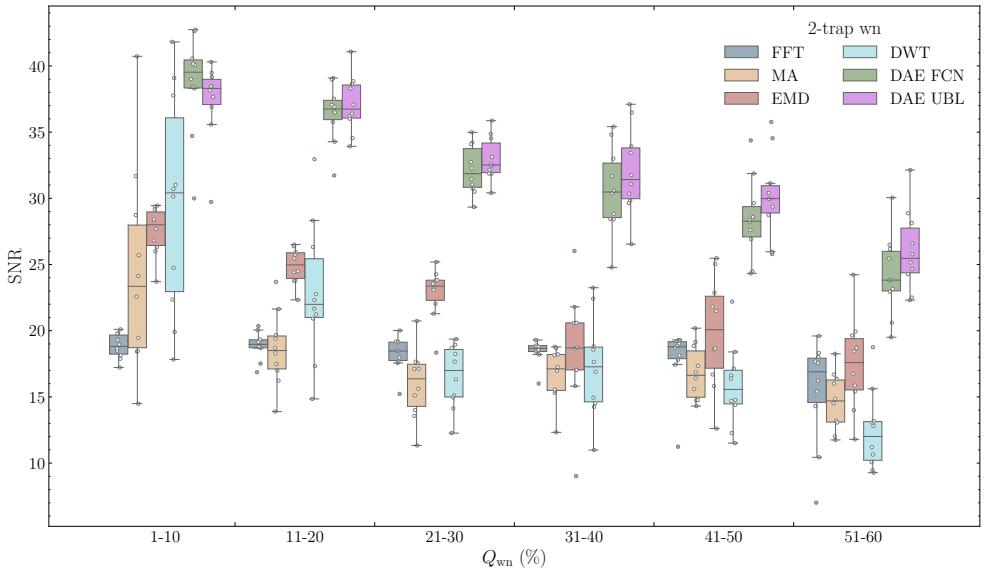


(b) SNRs of 1-trap RTSs with Gaussian white noise

Figure 4.7: SNRs of 1-trap RTSs

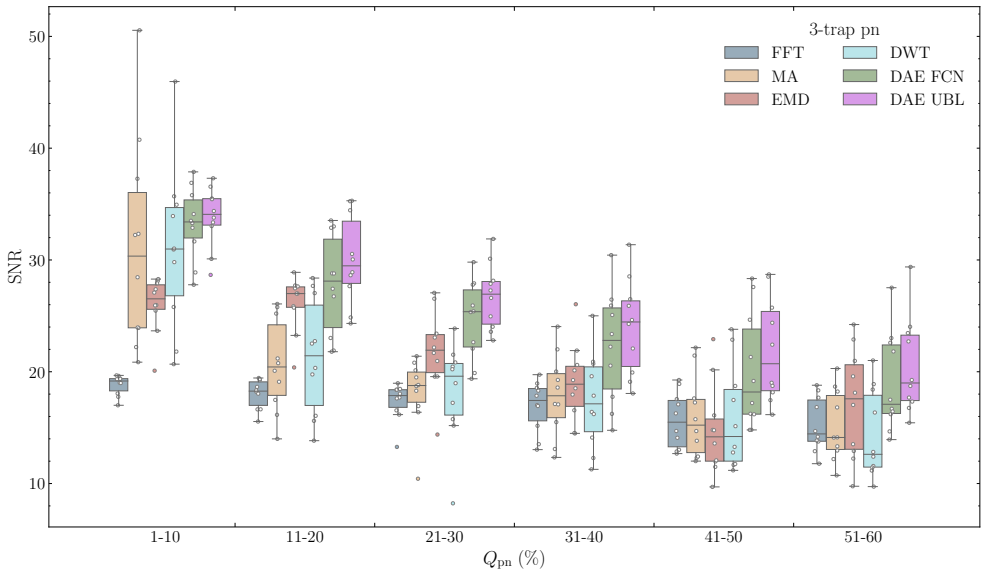


(a) SNRs of 2-trap RTSs with pink noise

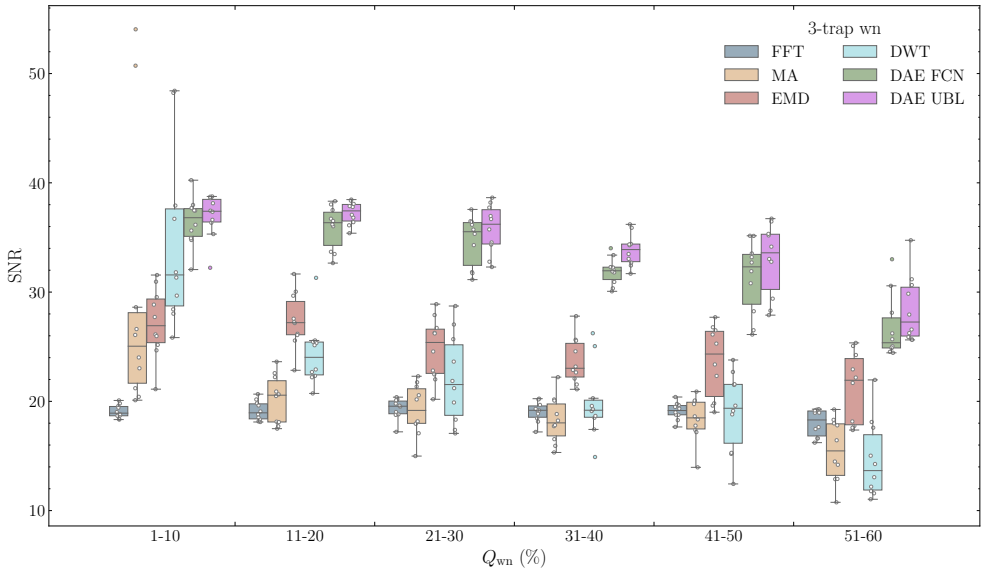


(b) SNRs of 2-trap RTSs with Gaussian white noise

Figure 4.8: SNRs of 2-trap RTSs

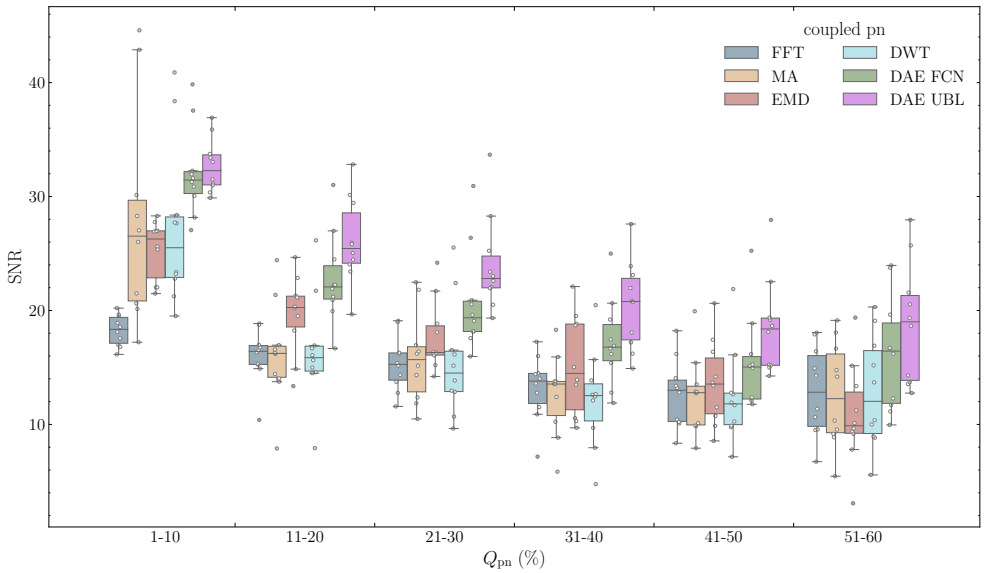


(a) SNRs of 3-trap RTSs with pink noise

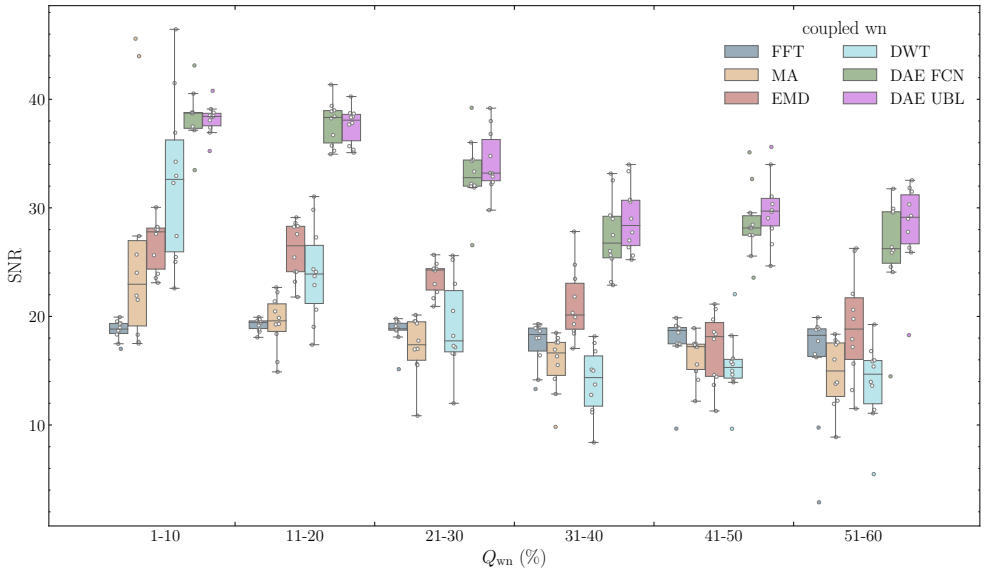


(b) SNRs of 3-trap RTSs with Gaussian white noise

Figure 4.9: SNRs of 3-trap RTSs

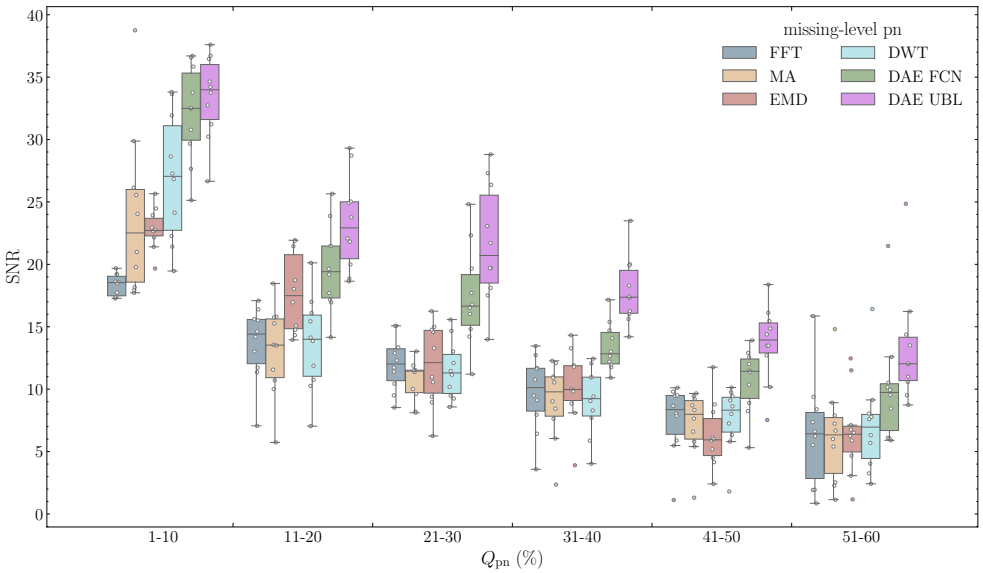


(a) SNRs of coupled RTSs with pink noise

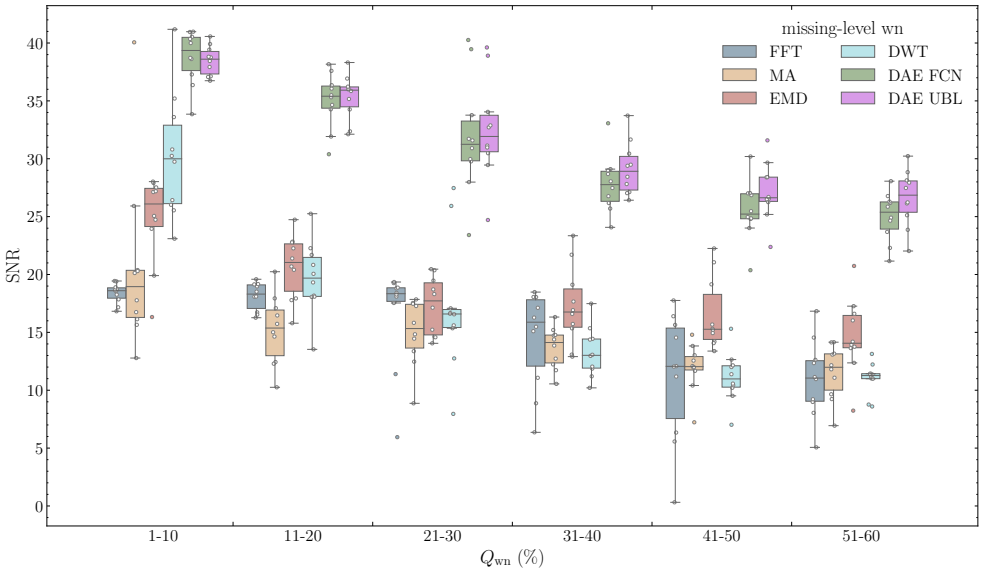


(b) SNRs of coupled RTSs with Gaussian white noise

Figure 4.10: SNRs of coupled RTSs

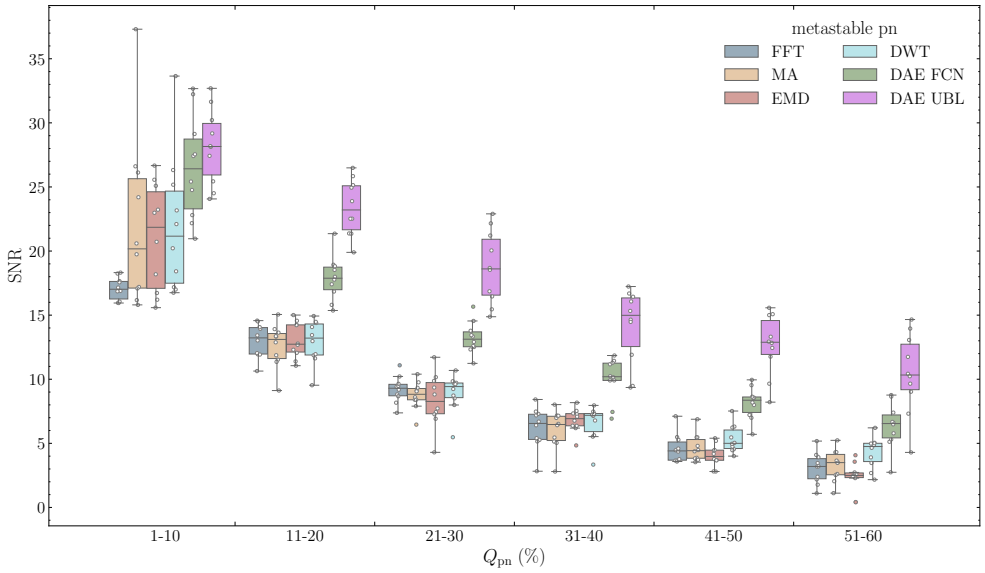


(a) SNRs of missing-level RTSs with pink noise

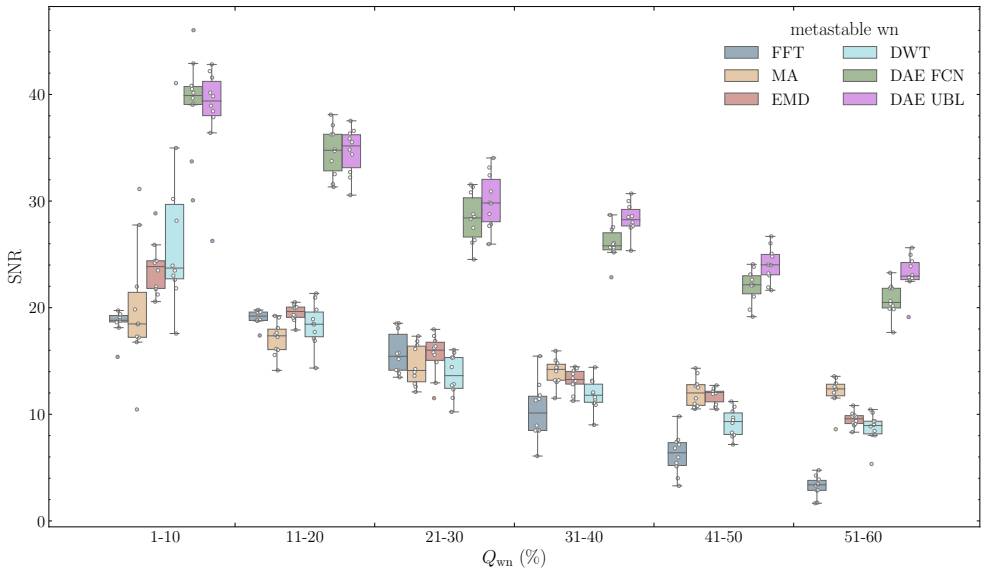


(b) SNRs of missing-level RTSs with Gaussian white noise

Figure 4.11: SNRs of missing-level RTSs



(a) SNRs of metastable RTSs with pink noise



(b) SNRs of metastable RTSs with Gaussian white noise

Figure 4.12: SNRs of metastable RTSs

For the denoising visualization results of RTSs with pink noise, the denoising visualization results still maintain fluctuations in local areas and general trends to a certain extent, which proves that RTSs with pink noise is more challenging to denoise than those with Gaussian white noise.

For the noise level Q_{wn} or Q_{pn}, it is clear that for all RTS categories, the denoising results of RTSs with high noise level ((a), (b), (c)) are worse than those with low noise level ((d), (e), (f)).

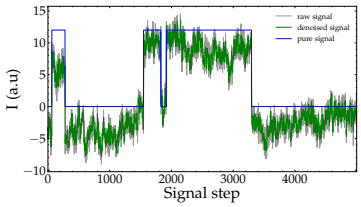
For model comparisons, DAE models have better denoising performance than MA filtering. Although MA filtering is capable of smoothing level fluctuations in local areas to a certain extent, it cannot suppress the level fluctuations in local areas and general trends well. In contrast to the two DAE models, fluctuations in local areas are effectively suppressed. Especially at high noise levels, two DAE models perform better than MA filtering on suppressing level fluctuations in local areas. Among DAE models, DAE UBL has better performance than DAE FCN. DAE FCN cannot control the level fluctuations in general trends in RTSs containing pink noise, which leads to the shift of transition amplitude in the denoised signal. With Bi-LSTM and a better encoder-decoder structure, DAE UBL can remove level fluctuations in general trends and output a flat signal trace by analyzing past and future current intensities. Although the characteristics of this forced flat signal trace may cause DAE UBL to make wrong levels judgments with dramatic intensity shifts when predicting some RTSs with large level fluctuations in general trends, DAE UBL still has the best denoising performance of all denoising models most of the time.

4.6 Ablation Study

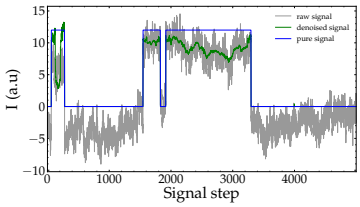
In order to verify that the features from Bi-LSTM and U-Net all enhance the denoising performance of DAE UBL, Table 4.6 is presented to display the denoising errors of the ablation study with different noise types.

4.6.1 Effect of Bi-LSTM

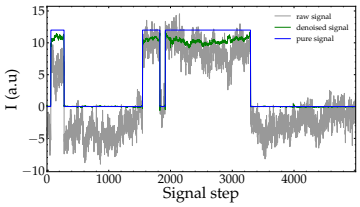
According to the Table 4.6, the mean value and variance of RMSEs significantly decrease after introducing Bi-LSTM to DAE for both Gaussian white noise and pink noise cases. Especially for pink noise cases, the variance of SNR decreases more than 55% compared to that of DAE. For SNRs, DAE+Bi-LSTM has a dramatically higher mean value and



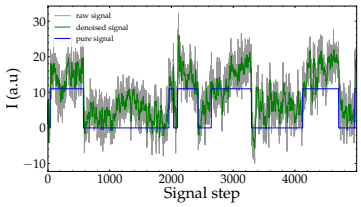
(a) Denoising plot of MA (Q_{pn} = 20%)



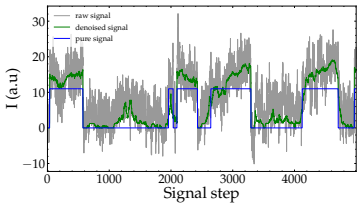
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



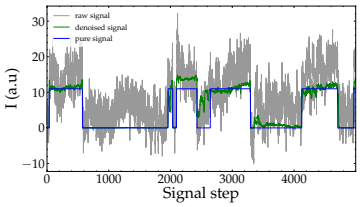
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{pn} = 60%)

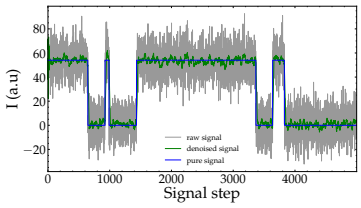


(e) Denoising plot of DAE FCN (Q_{pn} = 60%)

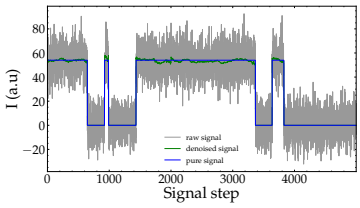


(f) Denoising plot of DAE UBL (Q_{pn} = 60%)

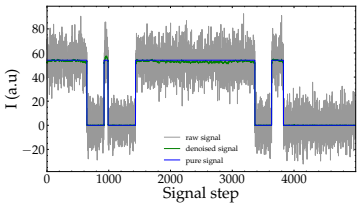
Figure 4.13: Denoising plot of 1-trap RTS with pink noise



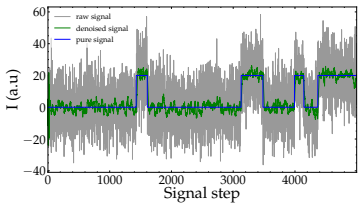
(a) Denoising plot of MA (Q_{wn} = 20%)



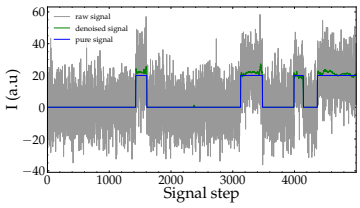
(b) Denoising plot of DAE FCN (Q_{wn} = 20%)



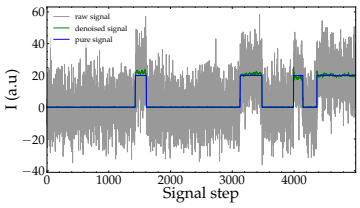
(c) Denoising plot of DAE UBL (Q_{wn} = 20%)



(d) Denoising plot of MA (Q_{wn} = 60%)

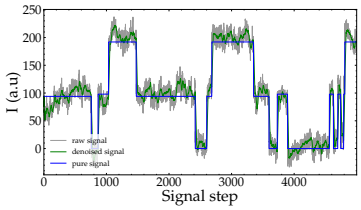


(e) Denoising plot of DAE FCN (Q_{wn} = 60%)

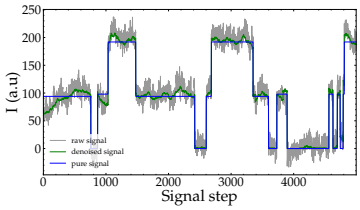


(f) Denoising plot of DAE UBL (Q_{wn} = 60%)

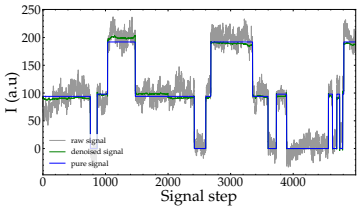
Figure 4.14: Denoising plot of 1-trap RTS with Gaussian white noise



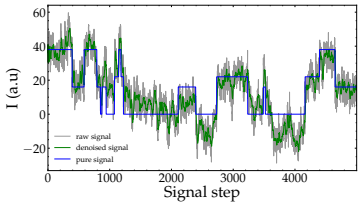
(a) Denoising plot of MA (Q_{pn} = 20%)



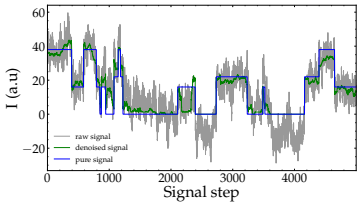
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



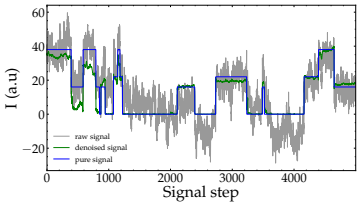
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{pn} = 60%)

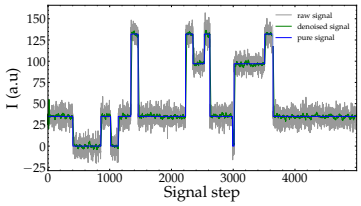


(e) Denoising plot of DAE FCN (Q_{pn} = 60%)

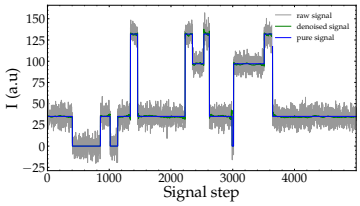


(f) Denoising plot of DAE UBL (Q_{pn} = 60%)

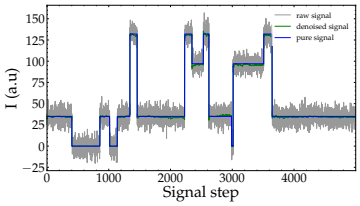
Figure 4.15: Denoising plot of 2-trap RTS with pink noise



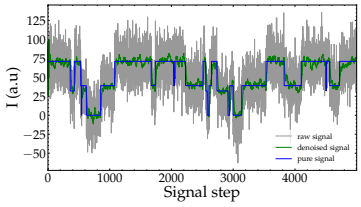
(a) Denoising plot of MA (Q_{wn} = 20%)



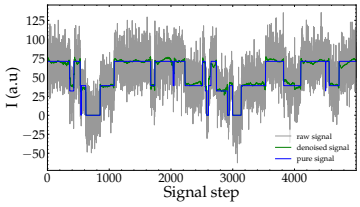
(b) Denoising plot of DAE FCN (Q_{wn} = 20%)



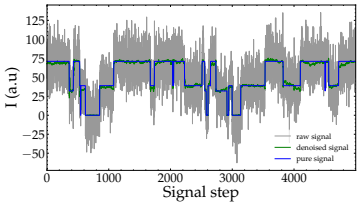
(c) Denoising plot of DAE UBL (Q_{wn} = 20%)



(d) Denoising plot of MA (Q_{wn} = 60%)

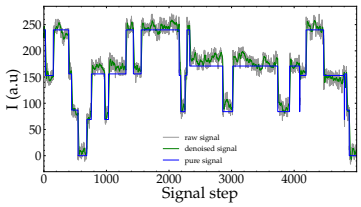


(e) Denoising plot of DAE FCN (Q_{wn} = 60%)

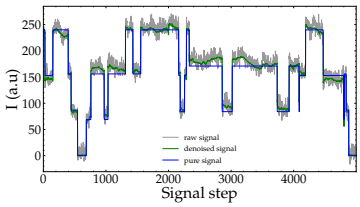


(f) Denoising plot of DAE UBL (Q_{wn} = 60%)

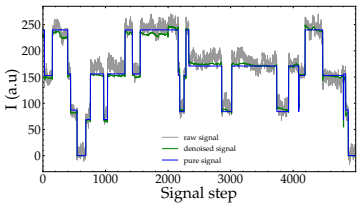
Figure 4.16: Denoising plot of 2-trap RTS with Gaussian white noise



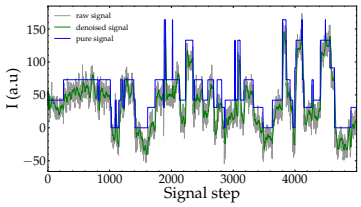
(a) Denoising plot of MA (Q_{pn} = 20%)



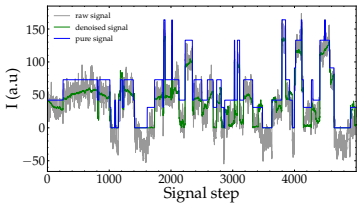
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



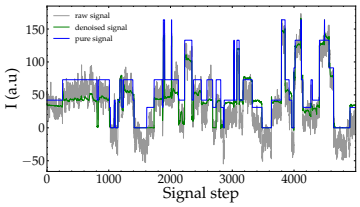
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{pn} = 60%)

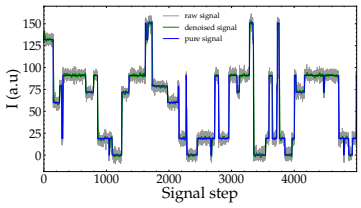


(e) Denoising plot of DAE FCN (Q_{pn} = 60%)

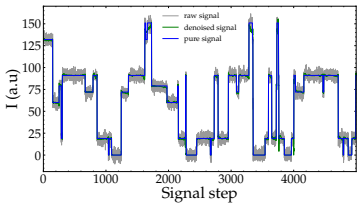


(f) Denoising plot of DAE UBL (Q_{pn} = 60%)

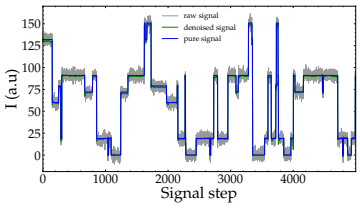
Figure 4.17: Denoising plot of 3-trap RTS with pink noise



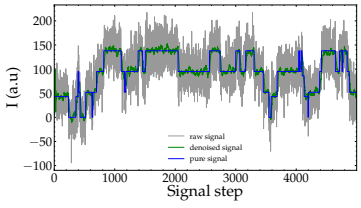
(a) Denoising plot of MA (Q_{wn} = 20%)



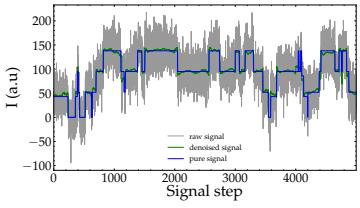
(b) Denoising plot of DAE FCN (Q_{wn} = 20%)



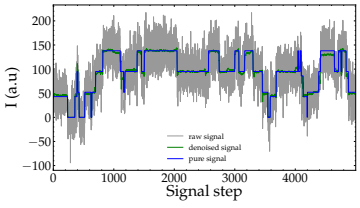
(c) Denoising plot of DAE UBL (Q_{wn} = 20%)



(d) Denoising plot of MA (Q_{wn} = 60%)

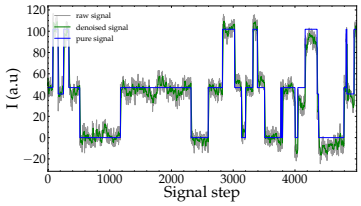


(e) Denoising plot of DAE FCN (Q_{wn} = 60%)

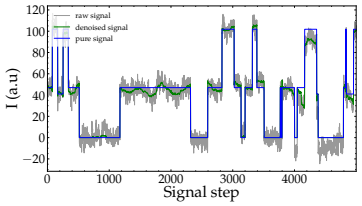


(f) Denoising plot of DAE UBL (Q_{wn} = 60%)

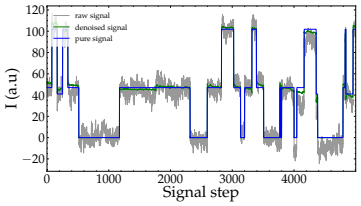
Figure 4.18: Denoising plot of 3-trap RTS with Gaussian white noise



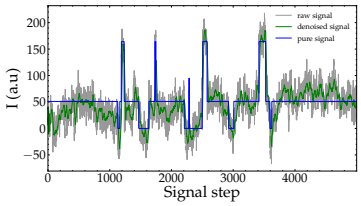
(a) Denoising plot of MA (Q_{pn} = 20%)



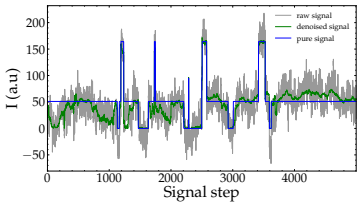
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



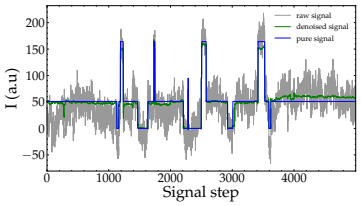
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{pn} = 60%)

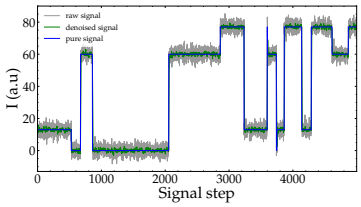


(e) Denoising plot of DAE FCN (Q_{pn} = 60%)

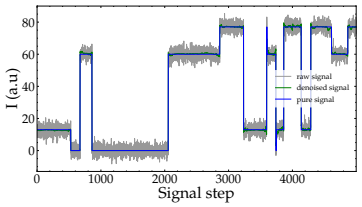


(f) Denoising plot of DAE UBL (Q_{pn} = 60%)

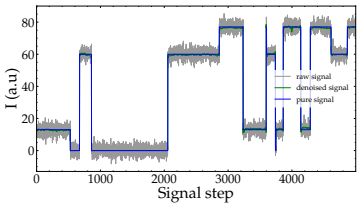
Figure 4.19: Denoising plot of coupled RTS with pink noise



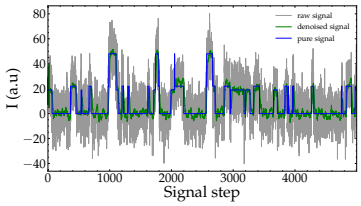
(a) Denoising plot of MA (Q_{pn} = 20%)



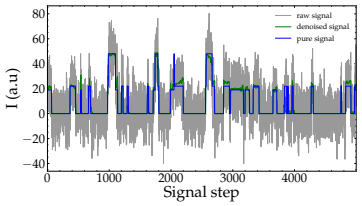
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



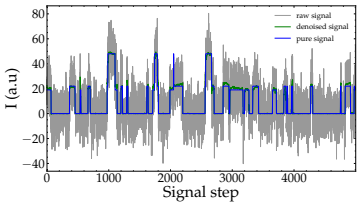
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{wn} = 60%)

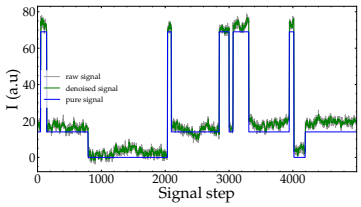


(e) Denoising plot of DAE FCN (Q_{wn} = 60%)

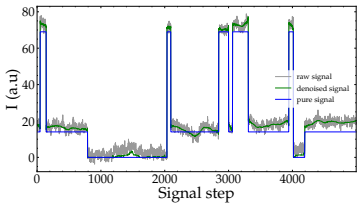


(f) Denoising plot of DAE UBL (Q_{wn} = 60%)

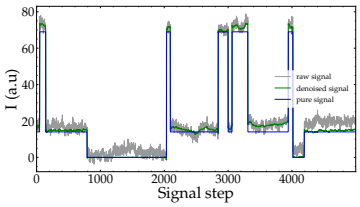
Figure 4.20: Denoising plot of coupled RTS with Gaussian white noise



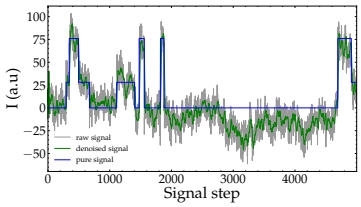
(a) Denoising plot of MA (Q_{pn} = 20%)



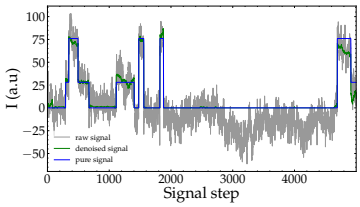
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



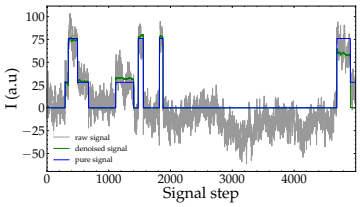
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{pn} = 60%)

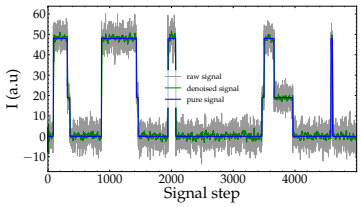


(e) Denoising plot of DAE FCN (Q_{pn} = 60%)

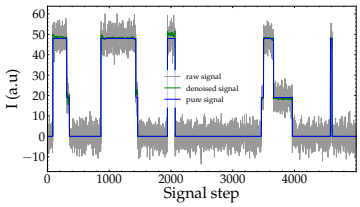


(f) Denoising plot of DAE UBL (Q_{pn} = 60%)

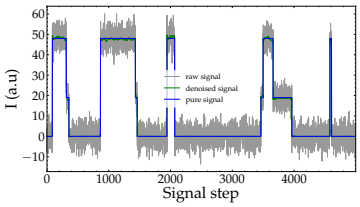
Figure 4.21: Denoising plot of missing-level RTS with pink noise



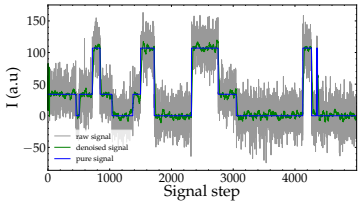
(a) Denoising plot of MA (Q_{wn} = 20%)



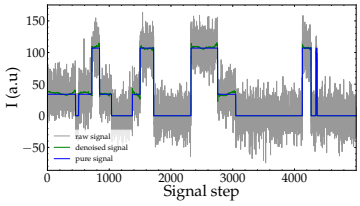
(b) Denoising plot of DAE FCN (Q_{wn} = 20%)



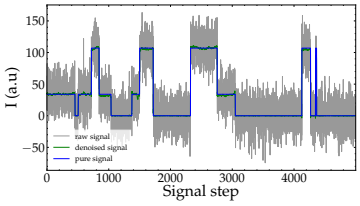
(c) Denoising plot of DAE UBL (Q_{wn} = 20%)



(d) Denoising plot of MA (Q_{wn} = 60%)

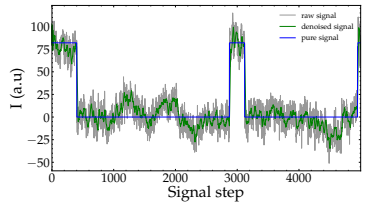


(e) Denoising plot of DAE FCN (Q_{wn} = 60%)

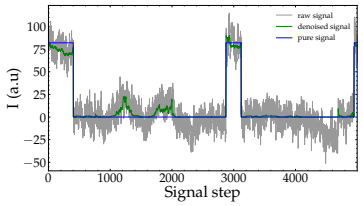


(f) Denoising plot of DAE UBL (Q_{wn} = 60%)

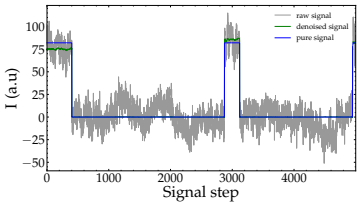
Figure 4.22: Denoising plot of missing-level RTS with Gaussian white noise



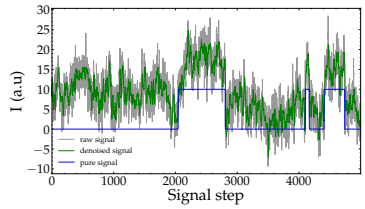
(a) Denoising plot of MA (Q_{pn} = 20%)



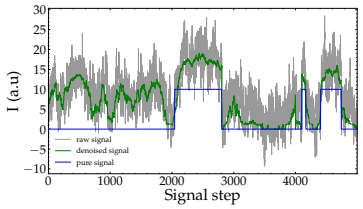
(b) Denoising plot of DAE FCN (Q_{pn} = 20%)



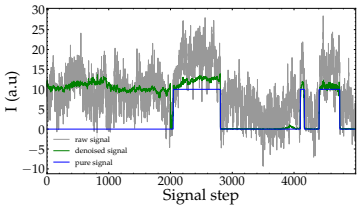
(c) Denoising plot of DAE UBL (Q_{pn} = 20%)



(d) Denoising plot of MA (Q_{pn} = 60%)

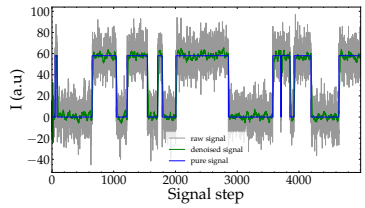


(e) Denoising plot of DAE FCN (Q_{pn} = 60%)

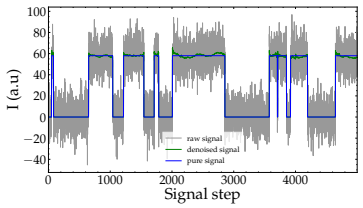


(f) Denoising plot of DAE UBL (Q_{pn} = 60%)

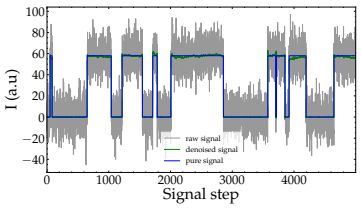
Figure 4.23: Denoising plot of metastable RTS with pink noise



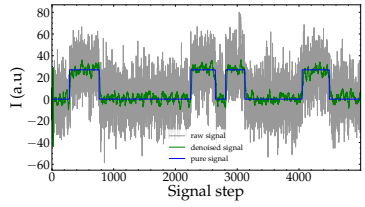
(a) Denoising plot of MA (Q_{wn} = 20%)



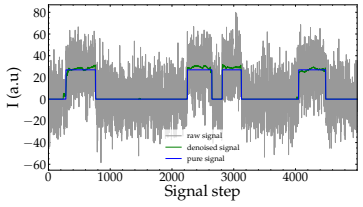
(b) Denoising plot of DAE FCN (Q_{wn} = 20%)



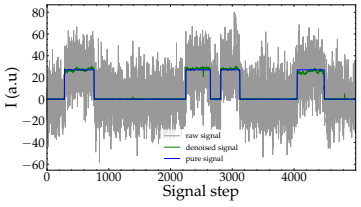
(c) Denoising plot of DAE UBL (Q_{wn} = 20%)



(d) Denoising plot of MA (Q_{wn} = 60%)



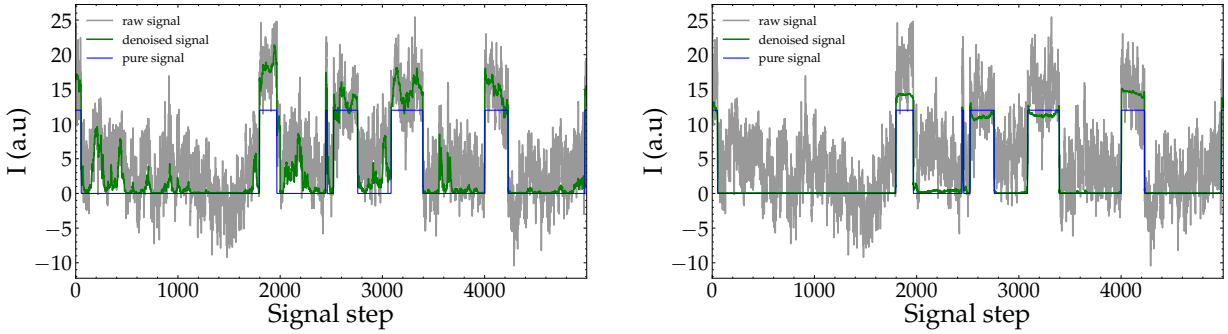
(e) Denoising plot of DAE FCN (Q_{wn} = 60%)



(f) Denoising plot of DAE UBL (Q_{wn} = 60%)

Figure 4.24: Denoising plot of metastable RTS with Gaussian white noise

a smaller variance than those of DAE. According to Fig.4.25, the reason why DAE+Bi-LSTM has better denoising performance than DAE is that it is capable of suppressing level fluctuations in general trends. Thanks to the ability of Bi-LSTM for capturing contexts in both directions, DAE+Bi-LSTM can keep the denoised signal trace flat.



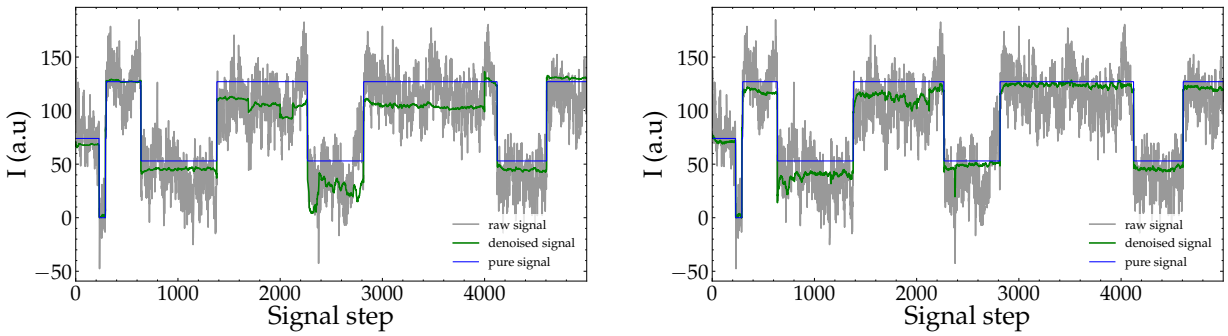
(a) 1-trap RTS denoised by DAE

(b) 1-trap RTS denoised by DAE+Bi-LSTM

Figure 4.25: Performance comparison between DAE and DAE+Bi-LSTM

4.6.2 Effect of skip connection

According to Table 4.6, the mean value and variance of RMSEs decrease after introducing skip connection from U-Net in general. For DAE UBL has a higher mean value and a dramatically smaller variance than those of DAE+Bi-LSTM. This is because denoised signals predicted by DAE+Bi-LSTM lose some low-level features due to the increasing number of feature extractions. As shown in Fig.4.26(a), there are some level shifts in the denoised 2-trap RTS. After introducing the skip connection feature (Fig.3.11) of U-Net, DAE UBL is capable of reusing low-level features from the encoder [4][8], which suppresses the level shifts shown in Fig.4.26(b). In general, DAE UBL improve the robustness of the denoising model compared to DAE+Bi-LSTM.



(a) 2-trap RTS denoised by DAE+Bi-LSTM

(b) 2-trap RTS denoised by DAE UBL

Figure 4.26: Performance comparison between DAE+Bi-LSTM and DAE UBL

Denoising error	Noise type	DAE	DAE+Bi-LSTM	DAE UBL
RMSE (mean)	white noise	2.63	**1.68**	1.71
RMSE (variance)	white noise	3.54	1.56	**1.43**
SNR (mean)	white noise	28.53	**32.12**	31.73
SNR (variance)	white noise	41.57	33.72	**26.46**
RMSE (mean)	pink noise	9.58	6.10	**6.02**
RMSE (variance)	pink noise	63.60	28.21	**27.96**
SNR (mean)	pink noise	17.85	21.41	**21.52**
SNR (variance)	pink noise	60.49	50.04	**48.47**

Table 4.6: Ablation study of DAE UBL

Chapter 5

RTS Reconstruction Models

To further verify the enhancement effect of the denoising model on the RTS analysis. We apply RTS reconstruction models as a reference. In previous research, most RTS reconstruction models are only designed for two-level RTSs, such as convolutional filtering, discrete wavelet transform, and hidden Markov model reconstruction. To reconstruct multi-level RTSs systematically, our previous work proposed a three-step analysis protocol to achieve the reconstruction of multi-level RTSs. To cover all types of reconstruction models, three two-level RTS reconstruction models and one multi-level RTS reconstruction model are introduced in this chapter and implemented to compare the reconstruction errors of the denoised RTSs and those of noisy RTSs in the next chapter.

5.1 Two-level RTS reconstruction

5.1.1 Convolutional Filtering Reconstruction

Convolutional filtering (CF) reconstruction is adapted from a popular reconstruction model proposed in 2009 [20]. The reconstruction process involves the following steps: noisy signal $\rightarrow$ convolutional filtering $\rightarrow$ final reconstruction $\rightarrow$ reconstructed signal (Fig. 5.1).

Convolutional Filtering

Convolutional filtering is capable of detecting fast-changing points in the signal by convolving the noisy two-level RTS and a step-shaped filter to locate level transitions shown in Fig.5.1(b). The step-shaped filter can be written as.

$$H(z) = \frac{2}{L}(-\sum_{i=0}^{\frac{L}{2}-1} z^{-i} + \sum_{i=\frac{L}{2}}^{L-1} z^{-i}) \tag{5.1}$$

where L is the width of the filter, and L is set to be 50 in RTS reconstruction.

Final Reconstruction

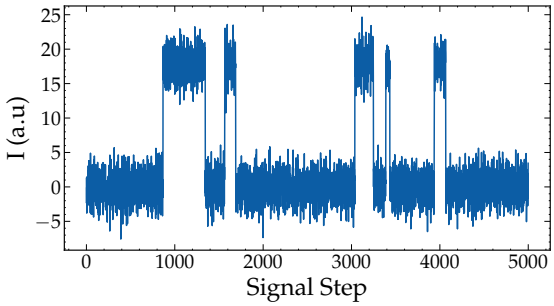
The noisy signal is divided into N slices in the final reconstruction according to fast-changing points. The mean current intensity of each slice is calculated as the current intensity of all signal steps of this slice. The final reconstruction result is shown in Fig.5.1(c).

5.1.2 Discrete Wavelet Transform Reconstruction

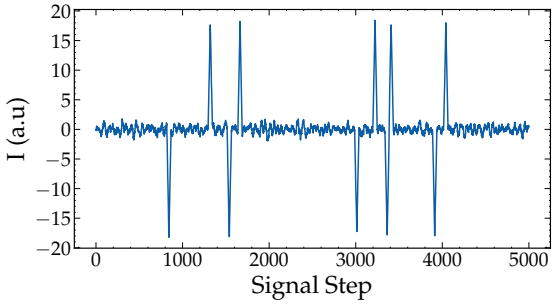
Discrete Wavelet Transform (DWT) reconstruction is based on the RTS reconstruction of DWT denoising. First, the noisy signal is denoised by DWT denoising. The reconstruction process involves the following steps: noisy signal → DWT denoising → temporal screen → thresholding → final reconstruction → reconstructed signal (Fig. 5.2). The details of DWT denoising, final reconstruction, can be found in 3.3, 5.1.1, respectively.

Temporal Screen

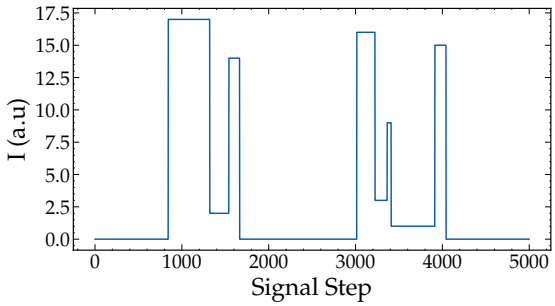
After passing DWT denoising, there are still some noticeable and short amplitude shifts in denoised RTSs. To eliminate these shifts, temporal screens are applied to smooth RTS traces. If the current intensity of the signal step t is different from the current intensity of the signal step $t-1$, calculate all the intensities in the following temporal screen of width $l = 10$, which contains the current intensity of the signal step t. If the intensity of the signal step t is the same as the next l time steps, keep that intensity. If not, the intensity is set to the value of the time step $t-1$. The result of the temporal screen is shown in Fig.5.2(c).



(a) Noisy 1-trap RTS



(b) Convolutional filtering result



(c) Reconstruction result

Figure 5.1: Convolutional filtering reconstruction process

Thresholding

To locate fast-changing points of the signal processed by the temporal screen. Thresholding is applied to find the position of the transition. A difference sequence is derived by computing the intensity of the signal step t minus the intensity of the signal step $t-1$. A threshold of 0.6745 times the maximum difference is applied for thresholding [21]. Any signal step whose difference is greater than the threshold is regarded as a fast-changing point. The intensities of other signal steps are all set to 0.

5.1.3 Hidden Markov Model Reconstruction

Hidden Markov model (HMM) [46] is a statistical probability model for modeling time series. It is used to describe a Markov process with hidden states. The related concepts and implementations of HMM were initially introduced by L.E. Baum and T. Petrie [6] for statistical applications. The model was soon implemented to signal and natural language processing fields.

The process of HMM reconstruction in Fig.5.3 involves the following steps: noisy signal $\rightarrow$ HMM reconstruction $\rightarrow$ reconstructed signal.

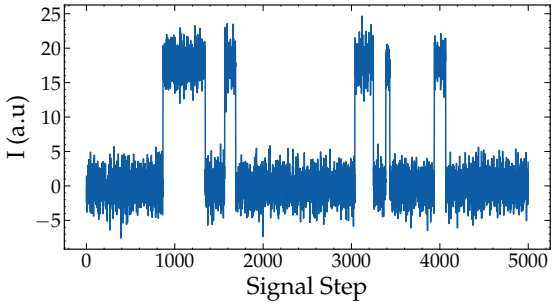
HMM Reconstruction

The process of HMM reconstruction is shown below.
1. Initial a HMM with two hidden states.
2. Train the HMM by employing Expectation Maximization [36] to obtain state transition probabilities, a two-by-two matrix.
3. Perform inference with the trained HMM by Viterbi algorithm [16].

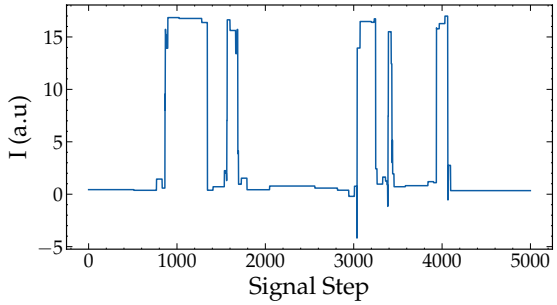
5.1.4 Digitization for Two-Level Reconstruction

Digitization is a very important step for RTS reconstruction to make sure the specific level in RTS has the same current intensities (Fig.5.4(b)). The digitization process for two-level RTSs is shown below.
1. Set a threshold to divide the high and low levels of reconstructed RTSs. The threshold T_d can be calculated as,

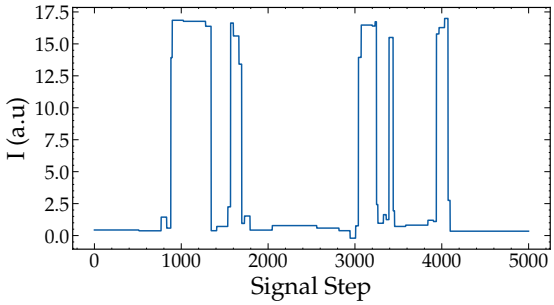
$$T_d = \frac{\max(I_r) - \min(I_r)}{\lambda} \tag{5.2}$$



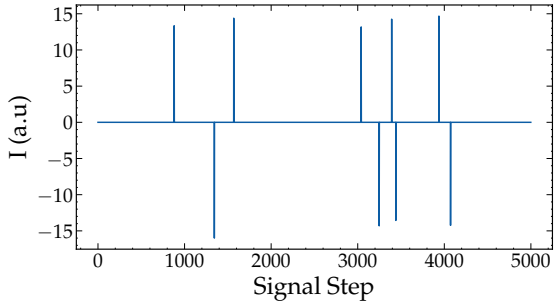
(a) Noisy 1-trap RTS



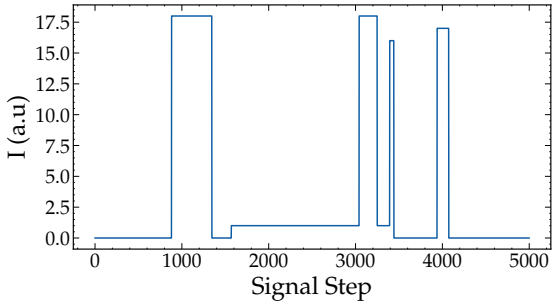
(b) Discrete wavelet transform denoising result



(c) Temporal screen result

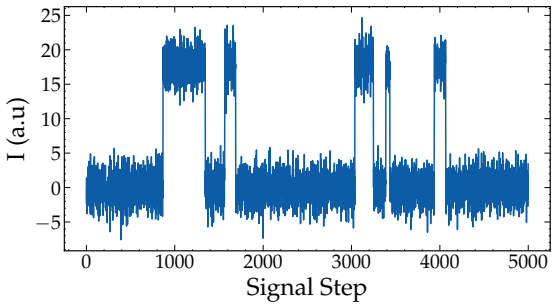


(d) Thresholding result

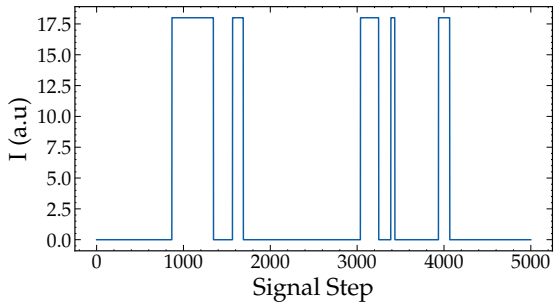


(e) Reconstruction result

Figure 5.2: Discrete wavelet transform reconstruction process



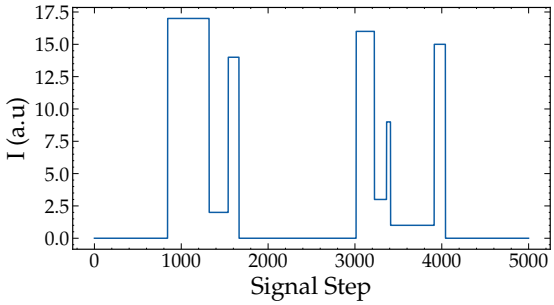
(a) Noisy 1-trap RTS



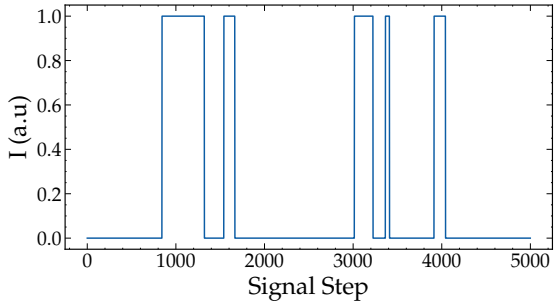
(b) Reconstruction result

Figure 5.3: Hidden Markov model reconstruction process

where I_r is the magnitude sequence of reconstructed RTS, and λ is the scale parameter to decide the value of T_d.
2. Set greater than or equal to T_d to high level, denoted as 1.
3. Set less than T_d to low level, denoted as 0.
λ for the reconstruction models in 5.1.1, 5.1.2, and 5.1.3 are all equal to 2.5.



(a) Reconstruction result of 1-trap RTS



(b) Digitization result

Figure 5.4: Two-level RTS digitization process

5.2 Multi-level RTS reconstruction

There are many models for reconstructing two-level RTSs, while few models are designed for multi-level RTS reconstruction. This is because applying two-level RTSs digitization is easy by directly adopting simple thresholding to separate low and high lev-

els. But digitization for multi-level RTSs can become very challenging when N_{traps} and δ_{RTS} are unknown. To establish a systematic approach for multi-level RTS reconstruction, our previous research [48] proposed a three-step analysis protocol to analyze and reconstruct complex multi-level RTSs. The process of the three-step analysis protocol is shown in Fig. 5.5.

Step 1: Peak Estimation

First, MA filtering is performed on noisy RTSs to uncover the peaks masked by background noise. Filtering details can be found in 3.1. And then KDE with Gaussian kernel is applied to estimate the PDF. The number of traps N_{traps} of RTS can be determined by the number of density peaks of PDF.

Step 2: Amplitude Extraction

Gaussian mixture model (GMM) is applied for modeling PDFs to extract the amplitudes of the RTS. For each RTS component, two Gaussian functions are defined to represent low and high levels,

$$G_{high} = q \cdot G(\mu = I_0 + \Delta_{RTS}, \sigma) \tag{5.3}$$

$$G_{low} = p \cdot G(\mu = I_0, \sigma) \tag{5.4}$$

where $p = \frac{\bar{\tau}_{low}}{\bar{\tau}_{high}+\bar{\tau}_{low}}$, $q = (1-p)$, and G is the Gaussian function with mean μ and standard deviation σ. Non-linear least-squares curve-fitting is applied to fit the probability distribution of a multi-level RTS G_T, which is the convolution of G_{high} and G_{low} pairs. Δ_{RTS} for each trap can be extracted from the fitted parameters of G_T.

Step 3: Pure RTS Reconstruction

An RNN model is applied to convert pure RTS reconstruction to a multi-output classification task [48]. The architecture of the RNN model is based on three bidirectional gated recurrent units (Bi-GRU) layers, and the hidden size of each layer is 256. The input of the model is a normalized noisy RTS between 0 and 1 generated according to the parameters obtained in Step 1 and Step 2, 0 represents the low level, and 1 represents the high level. The time step of the RNN model input is 64, which means that the Bi-GRU layers analyze the input slice with a width of 64 and predict the output of the last signal step of the slice in the last layer. To generate the predicted time series of RTSs

for each trap, the hidden state processed by Bi-GRU is transformed into the predicted probability of high level and low level using the Softmax layer. The prediction with the highest probability is taken level as the prediction result of each signal step. The output channel number of reconstructed RTSs equals the number of traps N_{traps}.

The training process of the RNN model is shown in Fig. 5.5. According to the N_{traps}, Δ_{RTS}, and Q_n obtained in the first two steps, 20 synthetic noisy signals with 50,000 signal length are generated to train the RNN model. After shuffling the synthetic signals, 80% of the dataset is used for training purposes, while the remaining 20% is used for validation. Once the RNN model is trained, the trained model is applied to predict the input noisy signal to obtain the reconstructed signal.

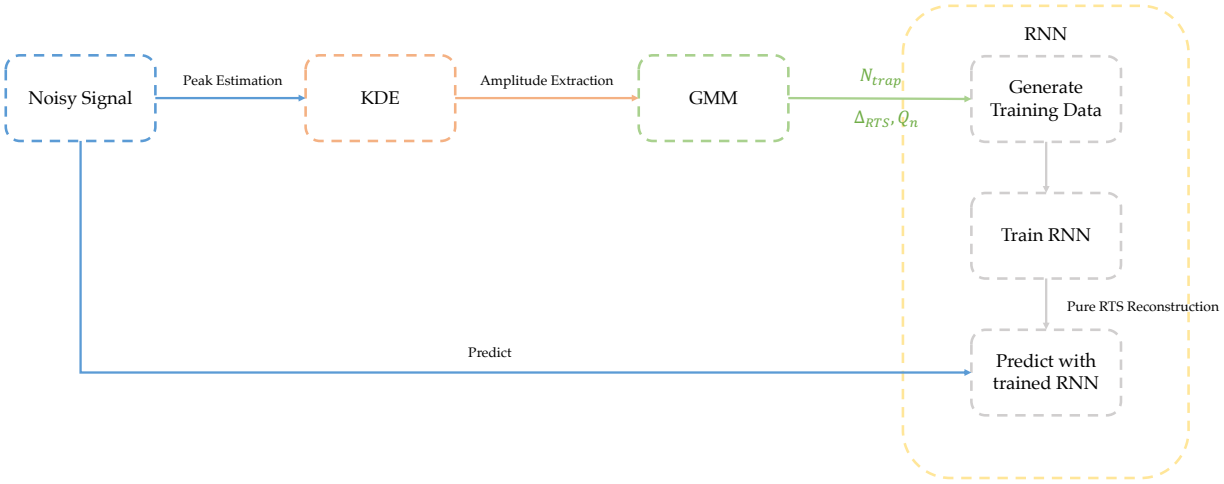


Figure 5.5: Three-step analysis protocol architecture

Chapter 6

Enhancement of RTS Reconstruction by DAE Denoising

6.1 Dataset

The synthetic evaluation dataset for RTS reconstruction is the same as the evaluation dataset for RTS denoising (Table 4.3) but divided into two separate synthetic evaluation datasets: one for two-level RTS reconstruction (as shown in Table 6.1) and the other for multi-level RTS reconstruction (as shown in Table 6.2). In these two datasets, all RTSs have a fixed signal length of 1,000,000 signal steps. Each RTS category in the dataset has 60 samples. Statistics for a specific RTS category can be found in Table 4.4. Each noise level range in {1%-10%, 11%-20%, 21%-30%, 31%-40%, 41%-50%, 51%-60%} has ten samples.

RTS category	Noise type	Signal length	Samples number
1-trap	white noise	1,000,000	60
1-trap	pink noise	1,000,000	60
metastable	pink noise	1,000,000	60
metastable	white noise	1,000,000	60

Table 6.1: Statistics of the synthetic evaluation dataset for two-level RTS reconstruction

RTS category	Noise type	Signal length	Samples number
2-trap	white noise	1,000,000	60
2-trap	pink noise	1,000,000	60
3-trap	white noise	1,000,000	60
3-trap	pink noise	1,000,000	60
3-trap	white noise	1,000,000	60
coupled	pink noise	1,000,000	60
coupled	white noise	1,000,000	60
missing-level	pink noise	1,000,000	60
missing-level	white noise	1000,000	60

Table 6.2: Statistics of the synthetic evaluation dataset for multi-level RTS reconstruction

6.2 Evaluation metrics for RTS reconstruction

6.2.1 Amplitude Error

Amplitude error is introduced to evaluate the error rate of the transition amplitude in RTS reconstruction. For each trap of RTS, amplitude error ϵ_{amp} can be calculated as.

$$\epsilon_{amp} = \frac{\Delta_{RTS} - \Delta_{RTS_{est}}}{\Delta_{RTS}} \tag{6.1}$$

where Δ_{RTS} denotes the actual transition amplitude of the RTS trap, and $\Delta_{RTS_{est}}$ denotes the estimated transition amplitude of the same RTS trap.

6.2.2 Digitization Error

Digitization error is introduced to evaluate the error rate of the predicted levels of the RTS reconstruction model. For each trap of an RTS, digitization error ϵ_{dig} can be calculated as.

$$\epsilon_{dig} = \frac{\sum_{n=1}^{L} I_{RTS}}{L} \tag{6.2}$$

$$I_{RTS} = \begin{cases} 1, & \text{if } i^l_{pred} = i^l_{ori} \\ 0, & \text{if } i^l_{pred} \neq i^l_{ori} \end{cases} \tag{6.3}$$

where L denotes the signal length of the RTS, i^l_{pred} denotes the predicted current intensity of the RTS trap at signal step l, and i^l_{ori} denotes the original current intensity of the RTS trap at signal step l.

In order to evaluate the digitization error for the whole RTS, the general digitization error $\bar{\epsilon}_{dig}$ is also introduced for the evaluation.

$$\bar{\epsilon}_{dig} = \frac{\sum_{n=1}^{L} S}{L} \tag{6.4}$$

$$S = \begin{cases} 1, & \text{if } s^l_{pred} = s^l_{ori} \\ 0, & \text{if } s^l_{pred} \neq s^l_{ori} \end{cases} \tag{6.5}$$

where s^l_{pred} denotes the predicted sum of traps at signal step l, and s^l_{ori} denotes the original sum of traps at signal step l.

6.2.3 Tau Error

Tau error is introduced to evaluate the duration of low and high levels $\bar{\tau}_{low}$, $\bar{\tau}_{high}$ in RTS reconstruction. The process of calculation is shown below.
1. Generate a histogram for all individual durations τ_{low} or τ_{high} in an RTS. The bin number of the histogram N equals 50.
2. Perform a fitting procedure using the Poisson function with bin centers to obtain the fitted duration $\bar{\tau}$ as τ_{low} or τ_{high}. The Poisson function is shown below,

$$f(t) = \bar{\tau}^{-1} \exp(\frac{-t}{\bar{\tau}}) \tag{6.6}$$

Since the first bin contains many outliers, the first bin center is sometimes discarded during the fitting process. Fit start index λ is applied to start fitting from the λ^{th} bin center. In CF reconstruction and DWT reconstruction, λ is equal to 0. In HMM reconstruction, λ is equal to 1.

6.3 Experiment Details

In this experiment, two types of RTS reconstructions with different inputs were adopted to demonstrate the enhancement of RTS reconstruction after preprocessing inputs by DAE denoising model.

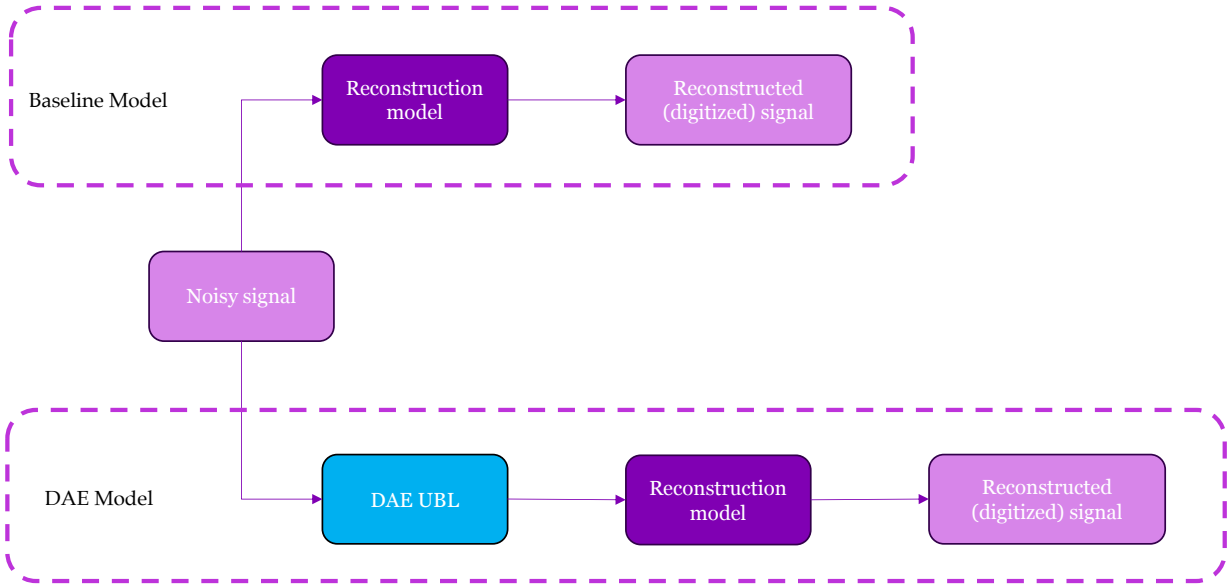


Figure 6.1: Two-level RTS reconstruction process

The experiment process of two-level RTS reconstruction is shown in Fig.6.1, the baseline model directly feeds noisy signals from the synthetic evaluation dataset to the reconstruction model. The DAE UBL model denoise noisy signals from the synthetic evaluation dataset via DAE UBL before feeding them into the model.

The experiment process of multi-level RTS reconstruction is shown in Fig.6.2, the baseline model directly feeds the noisy signal from the synthetic evaluation dataset to the reconstruction model. The DAE UBL model performs denoising via DAE UBL on both generated noisy signals for training and noisy signals to be predicted in Step 3 (pure RTS reconstruction) and then feeds them to the Bi-GRU for training and prediction.

There are no trainable parameters for two-level reconstruction models. The trainable parameter amount of Bi-GRU in the three-step analysis protocol is 759,810. The model was trained on servers equipped with Titan Xp GPU and 24 to 28 CPU cores.

For training details of Bi-GRU, the batch size is 1000. The optimizer of the models is Adam with learning rate = 0.00001, β_1=0.9, β_2=0.999, and ϵ=10^{-8}. The training epoch of the Bi-GRU model is 20.

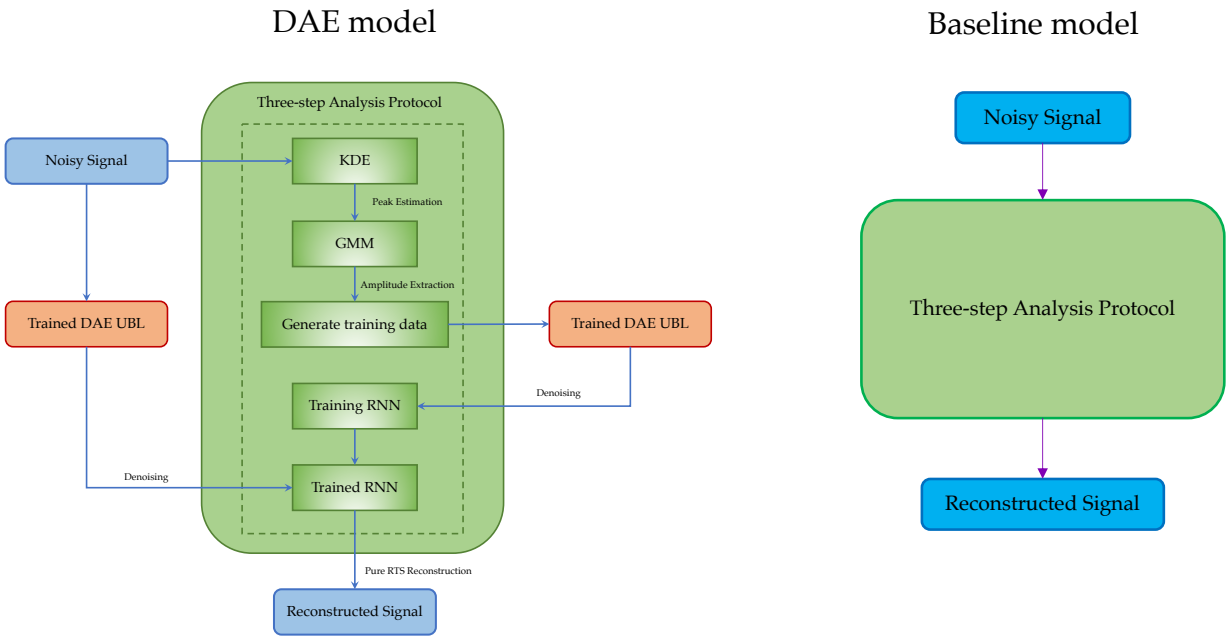


Figure 6.2: Multi-level RTS reconstruction process

6.4 Two-level RTS Reconstruction Analysis

Table 6.3, 6.4, and 6.5 exhibit the reconstruction errors for three two-level RTS reconstruction models in different noise level ranges. The reconstruction errors include digitization errors, amplitude errors, tau high errors, and rau low errors.

For CF reconstruction, the DAE UBL model usually has dramatically lower digitization errors than the baseline model. The disparity in digitization errors between the two models is small when reconstructing RTSs with Gaussian white noise. However, for RTSs with pink noise, the digitization errors of the DAE UBL model increase at a dramatically slower rate compared to the baseline model as Q_{wn} rises. The digitization error of the baseline model in $Q_{pn} \in [31\%, 60\%]$ has exceeded 30%. However, digitization errors of the DAE UBL model are always less than 16% when $Q_{pn} \in [1\%, 60\%]$. Amplitude errors for the two types of models exhibit a similar trend as digitization errors. The DAE UBL model maintains amplitude errors less than 8% and has much lower amplitude errors than the baseline model under high Q_{pn} conditions. Furthermore, the DAE UBL model dramatically outperforms the baseline model in terms of tau high errors and tau low errors.

Q_{wn} / Q_{pn}	Noise type	Digitization Error		Amplitude Error		Tau High Error		Tau Low Error	
		Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL
1-10%	white noise	5.11%	5.11%	**5.52%**	6.52%	**31.55%**	31.77%	**35.48%**	35.82%
11-20%	white noise	**5.00%**	5.01%	**5.05%**	6.24%	20.18%	**18.99%**	**26.52%**	32.66%
21-30%	white noise	**4.45%**	4.46%	**5.02%**	6.25%	26.32%	**22.11%**	**28.25%**	32.51%
31-40%	white noise	5.92%	**5.89%**	**6.19%**	7.80%	43.13%	**37.26%**	**35.85%**	44.00%
41-50%	white noise	6.01%	**5.97%**	**5.65%**	7.52%	59.44%	**35.50%**	**33.71%**	47.95%
51-60%	white noise	4.09%	**4.02%**	**3.94%**	5.04%	51.21%	**15.45%**	21.70%	**20.02%**
1-10%	pink noise	4.41%	**4.38%**	4.49%	**4.68%**	30.50%	**25.84%**	34.81%	**28.53%**
11-20%	pink noise	5.35%	**4.39%**	**3.96%**	4.51%	66.01%	**32.39%**	43.85%	**29.28%**
21-30%	pink noise	13.89%	**4.12%**	**4.98%**	5.16%	84.30%	**23.30%**	101.58%	**24.05%**
31-40%	pink noise	34.35%	**5.86%**	23.79%	**3.59%**	78.09%	**56.86%**	105.51%	**27.02%**
41-50%	pink noise	44.75%	**11.82%**	47.56%	**5.67%**	**82.84%**	156.44%	192.64%	**59.13%**
51-60%	pink noise	45.43%	**15.99%**	66.20%	**7.66%**	**53.90%**	130.41%	105.14%	**54.97%**

Table 6.3: Reconstruction errors of CF reconstruction

For DWT reconstruction, it can be seen from Table 6.4 that reconstruction errors of the baseline model are not satisfactory, particularly in terms of tau high and tau low errors, most of which exceed 100%. However, after the denoising preprocessing via DAE UBL, the performance of the reconstruction model has dramatically improved. The enhancement has reduced all reconstruction errors in almost all noise level ranges, effectively maintaining them within a reasonable range.

Q_{wn} / Q_{pn}	Noise type	Digitization Error		Amplitude Error		Tau High Error		Tau Low Error	
		Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL
1-10%	white noise	**2.70%**	4.79%	**3.11%**	4.51%	**20.49%**	34.06%	**20.38%**	52.79%
11-20%	white noise	23.41%	**2.73%**	27.14%	**3.19%**	270.53%	**16.47%**	195.64%	**26.26%**
21-30%	white noise	26.87%	**3.01%**	29.91%	**3.46%**	744.00%	**12.43%**	474.72%	**36.77%**
31-40%	white noise	29.59%	**3.29%**	33.52%	**4.01%**	1196.24%	**19.19%**	1024.15%	**31.18%**
41-50%	white noise	28.7%	**2.99%**	27.18%	**4.01%**	907.52%	**21.34%**	1306.35%	**23.01%**
51-60%	white noise	31.06%	**2.26%**	32.51%	**2.64%**	1077.25%	**8.19%**	1708.20%	**14.00%**
1-10%	pink noise	25.77%	**2.39%**	3.53%	**2.12%**	31.56%	**13.17%**	43.2%	**33.80%**
11-20%	pink noise	5.35%	**4.39%**	15.76%	**2.42%**	683.78%	**19.04%**	647.94%	**16.11%**
21-30%	pink noise	35.13%	**8.99%**	14.85%	**7.93%**	306.06%	**60.41%**	506.83%	**52.24%**
31-40%	pink noise	44.14%	**20.41%**	12.68%	**7.11%**	859.80%	**44.14%**	2922.68%	**164.40%**
41-50%	pink noise	47.06%	**34.21%**	**14.77%**	15.26%	2543247.52%	**621.51%**	**390.20%**	1069.93%
51-60%	pink noise	45.56%	**35.39%**	29.05%	**15.96%**	703.51%	**42.16%**	**48.15%**	138.23%

Table 6.4: Reconstruction errors of DWT reconstruction

For HMM reconstruction, digitization and amplitude errors of the DAE UBL model are similar to the baseline model when reconstructing all RTS with Gaussian white noise and low Q_{pn} RTSs. But when $Q_{pn} \in [41\%, 60\%]$, errors of the DAE UBL model are reduced by more than 60% compared to the baseline model, which is very impressive.

The tau errors of the two models seem to be scattered or randomly distributed with no obvious trend, but overall the baseline model has lower tau errors when reconstructing RTSs with Gaussian white noise, the DAE UBL model has lower tau errors when dealing with RTSs with pink noise. HMM reconstruction is one of the most accurate RTS reconstruction models. So even if DAE UBL denoising does not improve the model as much as CF and DWT reconstruction, the improvement is still meaningful.

Q_{wn}/Q_{pn}	Noise type	Digitization Error		Amplitude Error		Tau High Error		Tau Low Error	
		Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL
1-10%	white noise	**15.95%**	21.39%	**15.96%**	22.18%	32.99%	**31.96%**	**31.82%**	33.10%
11-20%	white noise	5.60%	**5.21%**	**5.61%**	6.24%	10.77%	**8.48%**	12.57%	**8.27%**
21-30%	white noise	10.19%	**9.35%**	**10.21%**	10.65%	20.36%	**16.27%**	26.75%	**23.47%**
31-40%	white noise	22.27%	**20.95%**	22.22%	**21.96%**	**21.81%**	43.86%	**23.64%**	27.11%
41-50%	white noise	27.97%	**5.39%**	27.91%	**7.02%**	52.33%	**27.23%**	44.47%	**23.64%**
51-60%	white noise	17.39%	**15.79%**	17.36%	**16.64%**	**26.35%**	32.27%	**29.16%**	37.22%
1-10%	pink noise	**5.11%**	16.00%	**5.64%**	16.68%	**59.03%**	63.90%	43.2%	**33.80%**
11-20%	pink noise	11.61%	**7.25%**	13.45%	**8.52%**	**25.18%**	33.87%	**12.84%**	61.03%
21-30%	pink noise	**24.05%**	27.46%	**25.98%**	28.59%	**34.24%**	55.86%	35.18%	**28.23%**
31-40%	pink noise	14.14%	**13.05%**	**11.79%**	13.02%	44.32%	**28.01%**	45.58%	**18.90%**
41-50%	pink noise	25.46%	**5.23%**	22.36%	**4.43%**	**55.04%**	74.77%	71.02%	**33.93%**
51-60%	pink noise	27.62%	**6.51%**	23.70%	**7.36%**	55.37%	**39.37%**	60.45%	**35.35%**

Table 6.5: Reconstruction errors of HMM reconstruction

In summary, the DAE UBL model can significantly and comprehensively improve the performance of three RTS reconstructions.

6.5 Multi-level RTS Reconstruction Analysis

Table 6.6 to 6.9 show the reconstruction errors for the three-step analysis protocol. Not all the RTSs in the synthetic evaluation dataset can be successfully processed by the first two steps of the three-step analysis protocol. Therefore, Table 6.6 is presented to show the statistics of successfully processed samples of different RTS categories. It can be seen from the table that less RTSs with pink noise can be successfully processed by the three-step analysis protocol than RTSs with Gaussian white noise. And as the N_{traps} of RTS increases, the number of successfully processed samples also decreases.

Because the DAE UBL denoising is only applied to Step 3 of the three-step analysis protocol, and the amplitude estimation is finished in Step 2, the amplitude errors of the two models remain identical.

RTS category	Noise type	Success number	Samples number
missing-level	white noise	60	60
coupled	white noise	46	60
2-trap	white noise	48	60
3-trap	white noise	28	60
missing-level	pink noise	35	60
coupled	pink noise	26	60
2-trap	pink noise	27	60
3-trap	pink noise	12	60

Table 6.6: Successfully processed samples of three-step analysis protocol

Digitization errors of the three-step analysis protocol are shown in Table 6.7. Since RTSs when $Q_{pn} \in [51\%, 60\%]$ are not successfully processed by the three-step analysis protocol, digitization errors cannot be calculated. For RTSs with Gaussian white noise, the baseline model has relatively lower digitization errors than those of the DAE UBL model. However, the difference between these two models is slight. For RTSs with pink noise, the DAE UBL model has much lower digitization errors than those of the baseline model. When we focus on general digitization errors, with the increase of Q_{pn}, the DAE UBL model is more effective in reducing general digitization errors than the baseline model.

Q_{wn} / Q_{pn}	Noise type	Digitization Error (Trap 0)		Digitization Error (Trap 1)		Digitization Error (Trap 2)		General Digitization Error	
		Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL
1-10%	white noise	**17.82%**	18.56%	17.14%	**16.92%**	**45.04%**	45.77%	**30.87%**	31.15%
11-20%	white noise	**13.96%**	14.05%	19.86%	**18.99%**	**39.80%**	40.26%	30.87%	**30.08%**
21-30%	white noise	**13.51%**	13.70%	**17.06%**	18.30%	**39.54%**	42.12%	**28.11%**	29.64%
31-40%	white noise	**11.26%**	11.52%	16.24%	**14.04%**	**19.28%**	20.33%	24.87%	**23.05%**
41-50%	white noise	13.63%	**12.64%**	**15.60%**	17.05%	39.53%	**25.51%**	**25.72%**	26.16%
51-60%	white noise	**16.26%**	18.09%	**17.91%**	19.83%	**31.37%**	31.43%	**28.73%**	30.94%
1-10%	pink noise	**16.21%**	16.38%	16.34%	**15.92%**	37.70%	**37.69%**	30.20%	**29.67%**
11-20%	pink noise	16.78%	**16.10%**	17.01%	**16.60%**	**50.71%**	53.55%	30.16%	**29.70%**
21-30%	pink noise	21.89%	**19.01%**	18.60%	**17.79%**	53.06%	53.06%	32.99%	**30.54%**
31-40%	pink noise	32.97%	**29.98%**	31.41%	**26.62%**	**45.11%**	49.31%	51.64%	**47.30%**
41-50%	pink noise	35.30%	**30.91%**	41.60%	**26.35%**	**48.08%**	48.94%	62.13%	**48.93%**
51-60%	pink noise	–	–	–	–	–	–	–	–

Table 6.7: Digitization errors of three-step analysis protocol

Tau errors of the multi-level RTS reconstruction is shown in Table 6.8 and 6.9. The tau low error and tau high error of each trap of the reconstructed model after DAE

UBL denoising preprocessing are generally smaller than the baseline model. Similar to the trend of digitization errors, with the increase of Q_{pn}, the DAE UBL model is more effective in reducing tau high errors and tau low errors than the baseline model.

Q_{wn}/Q_{pn}	Noise type	Tau High Error (Trap 0)		Tau High Error (Trap 1)		Tau High Error (Trap 2)	
		Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL
1-10%	white noise	**31.55%**	33.19%	**24.50%**	24.67%	**25.93%**	37.79%
11-20%	white noise	19.95%	**14.83%**	30.64%	**18.82%**	59.20%	**47.71%**
21-30%	white noise	33.46%	**19.56%**	32.27%	**23.98%**	69.48%	**54.84%**
31-40%	white noise	24.33%	**8.91%**	47.53%	**18.32%**	66.64%	**53.08%**
40-50%	white noise	35.42%	**18.54%**	43.50%	**34.83%**	66.11%	**31.12%**
51-60%	white noise	**40.67%**	51.94%	54.16%	**46.58%**	41.46%	**26.89%**
1-10%	pink noise	24.66%	**17.21%**	25.79%	**18.56%**	77.98%	**74.81%**
11-20%	pink noise	34.46%	**9.85%**	46.37%	**14.81%**	**22.72%**	101.77%
21-30%	pink noise	64.94%	**31.26%**	47.99%	**18.25%**	20.04%	**13.12%**
31-40%	pink noise	71.89%	**44.49%**	67.32%	**37.92%**	53.11%	**41.32%**
41-50%	pink noise	61.72%	**7.24%**	59.23%	**43.42%**	37.77%	**32.05%**
51-60%	pink noise	–	–	–	–	–	–

Table 6.8: Tau high errors of three-step analysis protocol

Q_{wn}/Q_{pn}	Noise type	Tau Low Error (Trap 0)		Tau Low Error (Trap 1)		Tau Low Error (Trap 2)	
		Baseline	DAE UBL	Baseline	DAE UBL	Baseline	DAE UBL
1-10%	white noise	37.95%	**36.38%**	30.64%	**28.95%**	**65.30%**	79.22%
11-20%	white noise	15.94%	**9.85%**	32.72%	**21.81%**	**96.69%**	98.38%
21-30%	white noise	49.94%	**41.46%**	**23.62%**	28.91%	**44.25%**	63.52%
31-40%	white noise	12.27%	**9.01%**	38.91%	**23.90%**	51.12%	**50.16%**
41-50%	white noise	17.73%	**11.71%**	30.94%	**26.49%**	61.01%	**33.80%**
51-60%	white noise	**17.69%**	22.41%	23.34%	**22.81%**	**49.94%**	66.27%
1-10%	pink noise	16.17%	**15.71%**	28.77%	**23.90%**	48.44%	**46.09%**
11-20%	pink noise	24.53%	**10.70%**	**35.99%**	49.15%	31.65%	**22.88%**
21-30%	pink noise	54.21%	**20.87%**	34.15%	**23.14%**	**570.54%**	594.11%
31-40%	pink noise	59.36%	**29.82%**	51.02%	**19.34%**	75.27%	**40.59%**
41-50%	pink noise	77.64%	**56.48%**	77.73%	**33.47%**	71.95%	**63.31%**
51%-60%	pink noise	–	–	–	–	–	–

Table 6.9: Tau low errors of three-step analysis protocol

In summary, after applying the denoising preprocessing via DAE UBL, the reconstruction digitization errors and tau errors of the three-step analysis protocol are reduced, especially for RTSs with pink noise. Furthermore, this enhancement is consistent across all traps within an RTS.

Chapter 7

Conclusion

Noisy random telegraph signals (RTSs) are discrete random fluctuations that coexist with background noise. Analyzing noise RTSs is challenging since background noise covers the level transition and amplitude of the signal to a certain extent. To better understand the relationships between pure RTSs and background noise, we introduced the generation of synthetic noisy RTSs with Gaussian white noise or pink noise. Additionally, the signal traces of synthetic noisy RTSs are analyzed to clarify the impact of Gaussian white noise and pink noise on RTSs, respectively.

In order to enhance the analysis of synthetic noisy RTSs, we proposed two RTS denoising models based on DAE: DAE FCN and DAE UBL. A synthetic training dataset and a synthetic evaluation dataset were built for DAE training. The datasets involve three categories of nRTSs and three categories of aRTSs with Gaussian white noise or pink noise. Four classic denoising models, including MA filtering, DWT denoising, FFT denoising, and EMD denoising were introduced and compared with DAE models for synthetic evaluation dataset. The evaluation metrics for the comparison are RMSE, SNR, and denoising visualization analysis. In the comparison of RTS denoising models, DAE models demonstrate superior performance than classical denoising models. For RTSs with Gaussian white noise, the RMSE fluctuation range of DAE models is reduced by approximately 87.5% than traditional denoising models. The SNR fluctuation range of DAE models is reduced by approximately 60% than traditional denoising models. For RTSs with pink noise, the RMSE fluctuation range of DAE models is reduced by approximately 20% than traditional denoising models. The SNR fluctuation range of DAE models is reduced by approximately 18% than traditional denoising models.

In order to verify that DAE models not only display excellent results on evaluation

metrics but also can enhance the analysis of RTSs. Three two-level RTS reconstruction models in terms of CF, DWT, and HMM reconstruction, and a multi-level RTS reconstruction model called three-step analysis protocol were incorporated to verify the enhancement of DAE UBL for RTS reconstruction performance. We computed digitization, amplitude, and tau errors to characterize the performance of the baseline model and the DAE UBL model. In general, the DAE UBL model has lower digitization, amplitude, and tau errors than those of the baseline model, especially for RTSs when Q_{pn} is high. For CF reconstruction, digitization, amplitude, and tau low errors of the DAE UBL model decrease by more than 80% when $Q_{pn} \in [31\%, 60\%]$ than those of the baseline model. For DWT reconstruction, digitization and tau high errors of the DAE UBL model decrease by more than 20% when $Q_{pn} \in [11\%, 60\%]$. digitization and amplitude errors of the DAE UBL model decrease by more than 20% when $Q_{pn} \in [41\%, 60\%]$. For the three-step analysis protocol, tau high and tau low errors of the DAE UBL model decrease by more than 15% when $Q_{pn} \in [31\%, 50\%]$. And the general digitization error of the DAE UBL model decreases by more than 8% when $Q_{pn} \in [31\%, 50\%]$. The above results prove that the DAE UBL can effectively promote the performance of different RTS reconstruction models. In conclusion, DAE models have comparatively low denoising errors in conventional denoising metrics and can enhance the performance of RTS reconstruction models.

In this thesis, DAE models have been able to perfectly denoise RTSs with Gaussian white noise, but there is still room for improvement in the denoising performance of RTSs with pink noise. Therefore, the next step in this research is to improve the denoising performance of existing models for RTSs with pink noise. Additionally, we only conduct denoising and reconstruction analysis for synthetic noisy RTSs. To verify that the DAE model can also process real RTSs well, we will evaluate the performance of the denoising model on RTSs acquired by experiments.

References

[1] Electronic component — Wikipedia, the free encyclopedia, 2023. [Online; accessed 15-July-2023].

[2] Zaid Abdi Alkareem Alyasseri, Ahamad Tajudin Khader, Mohammed Azmi Al-Betar, Ammar Kamal Abasi, and Sharif Naser Makhadmeh. Eeg signal denoising using hybridizing method between wavelet transform with genetic algorithm. In *Proceedings of the 11th National Technical Seminar on Unmanned System Technology 2019: NUSYS'19*, pages 449–469. Springer, 2021.

[3] Nuditha Vibhavie Amarasinghe, Zeynep Çelik-Butler, Anna Zlotnicka, and Fang Wang. Model for random telegraph signals in sub-micron mosfets. *Solid-State Electronics*, 47(9):1443–1449, 2003.

[4] Reza Azad, Maryam Asadi-Aghbolaghi, Mahmood Fathy, and Sergio Escalera. Bi-directional convlstm u-net with densley connected convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.

[5] Sung-Ho Bae, Jeong-Hyun Lee, Hyuck-In Kwon, Jung-Ryul Ahn, Jae-Chul Om, Chan Hyeong Park, and Jong-Ho Lee. The $1/f$ noise and random telegraph noise characteristics in floating-gate nand flash memories. *IEEE Transactions on Electron Devices*, 56(8):1624–1630, 2009.

[6] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

[7] James Brown, Jian Fu Zhang, Bo Zhou, Mehzabeen Mehedi, Pedro Freitas, John Marsland, and Zhigang Ji. Random-telegraph-noise-enabled true random number generator for hardware security. *Scientific reports*, 10(1):17210, 2020.

[8] Sijing Cai, Yunxian Tian, Harvey Lui, Haishan Zeng, Yi Wu, and Guannan Chen. Dense-unet: a novel multiphoton in vivo cellular image segmentation model based on a convolutional neural network. *Quantitative Imaging in Medicine and Surgery*, 10(6), 2020.

[9] J.P. Campbell, J. Qin, K.P. Cheungl, L. Yu, J.S. Suehlel, A. Oates, and K. Sheng. The origins of random telegraph noise in highly scaled sion nmosfets. In *2008 IEEE International Integrated Reliability Workshop Final Report*, pages 105–109, 2008.

[10] C. M. Chang, Steve S. Chung, Y. S. Hsieh, L. W. Cheng, C. T. Tsai, G. H. Ma, S. C. Chien, and S. W. Sun. The observation of trapping and detrapping effects in high-k gate dielectric mosfets by a new gate current random telegraph noise (ig-rtn) approach. In *2008 IEEE International Electron Devices Meeting*, pages 1–4, 2008.

[11] Hsin-Tien Chiang, Yi-Yen Hsieh, Szu-Wei Fu, Kuo-Hsuan Hung, Yu Tsao, and Shao-Yi Chien. Noise reduction in ecg signals using fully convolutional denoising autoencoders. *IEEE Access*, 7:60806–60813, 2019.

[12] James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

[13] Çiğdem Polat Dautov and Mehmet Siraç Özerdem. Wavelet transform and signal denoising using wavelet method. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4, 2018.

[14] Naveen Kumar Dewangan, Manoj K Kowar, and Kiran Dewangan. A comparative study of denoising of ecg signals using wavelet transform. *Global Journal of Modern Biology and Technology*, 1(3):1–3, 2011.

[15] Shu-Kai S Fan, Chia-Yu Hsu, Chih-Hung Jen, Kuan-Lung Chen, and Li-Ting Juan. Defective wafer detection using a denoising autoencoder for semiconductor manufacturing processes. *Advanced Engineering Informatics*, 46:101166, 2020.

[16] G.D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[17] Ligang Gao, Fabien Alibart, and Dmitri B. Strukov. Analog-input analog-weight dot-product operation with ag/a-si/pt memristive devices. In *2012 IEEE/IFIP 20th International Conference on VLSI and System-on-Chip (VLSI-SoC)*, pages 88–93, 2012.

[18] Owen Gauthier, Sébastien Haendler, Patrick Scheer, Alexandre Vernhet, Quentin Rafhay, and Christoforos Theodorou. Enhanced statistical detection of random

telegraph noise in frequency and time domain. *Solid-State Electronics*, 194:108320, 2022.

[19] Charles J. Geyer. Stationary Stochastic Processes. Lecture Notes, 2012.

[20] V. Goiffon, G. R. Hopkinson, P. Magnan, F. Bernard, G. Rolland, and O. Saint-Pe. Multilevel rts in proton irradiated cmos image sensors manufactured in a deep submicron technology. *IEEE Transactions on Nuclear Science*, 56(4):2132–2141, 2009.

[21] Jürgen Groß. *Linear regression*, volume 175. Springer Science & Business Media, 2003.

[22] Chong Gu. Stationary Stochastic Processes. Lecture Notes, 2021.

[23] Ben Hendrickson, Ralf Widenhorn, Morley Blouke, Denis Heidtmann, and Erik Bodegom. Wavelet analysis of rts noise in cmos image sensors irradiated with high-energy photons. *IEEE Transactions on Nuclear Science*, 67(7):1732–1737, 2020.

[24] Benjamin William Hendrickson. *DC-RTS Noise: Observation and Analysis*. PhD thesis, Portland State University, 2019.

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[26] Robert V Hogg, Joseph W McKean, Allen T Craig, et al. *Introduction to mathematical statistics*. Pearson Education India, 2013.

[27] Ling Hu. Stationary Time Series. Lecture Notes, 2002.

[28] Norden E. Huang, Zheng Shen, Steven R. Long, Manli C. Wu, Hsing H. Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H. Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings: Mathematical, Physical and Engineering Sciences*, 454(1971):903–995, 1998.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Mark A Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37(2):233–243, 1991.

[31] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[32] Hina Magsi, Ali Hassan Sodhro, Faheem Akhtar Chachar, and Saeed A. Khan Abro. Analysis of signal noise reduction by using filters. In *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–6, 2018.

[33] Sumit Majumder. *Random Telegraph Signal Noise in CMOS Image Sensor (CIS) and Use of a CIS in a Low-Cost Digital Microscope*. PhD thesis, 2011.

[34] AJ Malcolm. Multi-level random telegraph noise analysis using machine learning techniques. Master's thesis, University of Waterloo, 2020.

[35] J. Martin-Martinez, J. Diaz, R. Rodriguez, M. Nafria, X. Aymerich, E. Roca, F. V. Fernandez, and A. Rubio. Characterization of random telegraph noise and its impact on reliability of sram sense amplifiers. In *2014 5th European Workshop on CMOS Variability (VARI)*, pages 1–6, 2014.

[36] T.K. Moon. The expectation-maximization algorithm. *IEEE Signal Processing Magazine*, 13(6):47–60, 1996.

[37] A Moshrefi, H Aghababa, and O Shoaei. Employing the empirical mode decomposition to denoise the random telegraph noise. *International Journal of Engineering*, 34(1):90–96, 2021.

[38] T. Nagumo, K. Takeuchi, Shinji Yokogawa, K. Imai, and Yutaro Hayashi. New analysis methods for comprehensive understanding of random telegraph noise. pages 1 – 4, 01 2010.

[39] Henri J. Nussbaumer. *The Fast Fourier Transform*, pages 80–111. Springer Berlin Heidelberg, Berlin, Heidelberg, 1981.

[40] Christopher Olah. Understanding lstm networks, 2015.

[41] Grigorios A. Pavliotis. *Stochastic Processes and Applications: Diffusion Processes, the Fokker-Planck and Langevin Equations*. Springer, 2014.

[42] M Petrychuk, I Zadorozhnyi, Y Kutovyi, S Karg, H Riel, and S Vitusevich. Noise spectroscopy to study the 1d electron transport properties in inas nanowires. *Nanotechnology*, 30(30):305001, may 2019.

[43] Nenad Popovic, Nadica Miljkovic, Olivera Djordjevic, and Tomislav Šekara. Artifact cancellation using median filter, moving average filter, and fractional derivatives in biomedical signals. 2016.

[44] Francesco Maria Puglisi and Paolo Pavan. Rtn analysis with fhmm as a tool for multi-trap characterization in hfox rram. *2013 IEEE International Conference of Electron Devices and Solid-state Circuits*, pages 1–2, 2013.

[45] Francesco Maria Puglisi and Paolo Pavan. Factorial hidden markov model analysis of random telegraph noise in resistive random access memories. *ECTI Transactions on Electrical Engineering, Electronics, and Communications*, 12(1):24–29, 2014.

[46] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[47] K. S. Ralls, W. J. Skocpol, L. D. Jackel, R. E. Howard, L. A. Fetter, R. W. Epworth, and D. M. Tennant. Discrete resistance switching in submicrometer silicon inversion layers: Individual interface traps and low-frequency ($\frac{1}{f}$?) noise. *Phys. Rev. Lett.*, 52:228–231, 1984.

[48] Marcel Robitaille, HeeBong Yang, Lu Wang, Bowen Deng, and Na Young Kim. Deep neural network analysis models for complex random telegraph signals. *Scientific Reports*, 13(1):10403, 2023.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[50] Márcio Cherem Schneider and Carlos Galup-Montoro. *Temporal and spatial fluctuations in MOSFETs*, page 134–176. Cambridge University Press, 2010.

[51] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[52] Zhen Shan, Jianhua Yang, Miguel AF Sanjuán, Chengjin Wu, and Houguang Liu. A novel adaptive moving average method for signal denoising in strong noise background. *The European Physical Journal Plus*, 137(1):50, 2022.

[53] Eddy Simoen and Cor Claeys. *Random Telegraph Signals in Semiconductor Devices*. 2053-2563. IOP Publishing, 2016.

[54] Sonali, Omkar Singh, and Ramesh Kumar Sunkaria. Ecg signal denoising based on empirical mode decomposition and moving average filter. In *2013 IEEE International Conference on Signal Processing, Computing and Control (ISPCC)*, pages 1–6, 2013.

[55] William J Stewart. *Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling*. Princeton university press, 2009.

[56] A Theodorsen, O E Garcia, and M Rypdal. Statistical properties of a filtered poisson process with additive random noise: distributions, correlations and moment estimation. *Physica Scripta*, 92(5):054002, apr 2017.

[57] M. J. Uren, D. J. Day, and M. J. Kirton. 1/f and random telegraph noise in silicon metal-oxide-semiconductor field-effect transistors. *Applied Physics Letters*, 47(11):1195–1197, 1985.

[58] Sara Vecchi, Paolo Pavan, and Francesco Maria Puglisi. A unified framework to explain random telegraph noise complexity in mosfets and rrams. In *2023 IEEE International Reliability Physics Symposium (IRPS)*, pages 1–6, 2023.

[59] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[60] Runsheng Wang, Shaofeng Guo, Zexuan Zhang, Jibin Zou, Dongyuan Mao, and Ru Huang. Complex random telegraph noise (rtn): What do we understand? In *2018 IEEE International Symposium on the Physical and Failure Analysis of Integrated Circuits (IPFA)*, pages 1–7, 2018.

[61] C.H. Wann, K. Noda, T. Tanaka, M. Yoshida, and Chenming Hu. A comparative study of advanced mosfet concepts. *IEEE Transactions on Electron Devices*, 43(10):1742–1753, 1996.

[62] P. Welch. The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2):70–73, 1967.

[63] HeeBong Yang and Na Young Kim. Material-inherent noise sources in quantum information architecture. *Materials*, 16(7), 2023.

[64] HeeBong Yang, Marcel Robitaille, Xuesong Chen, Hazem Elgabra, Lan Wei, and Na Young Kim. Random telegraph noise of a 28-nm cryogenic mosfet in the coulomb blockade regime. *IEEE Electron Device Letters*, 43(1):5–8, 2021.

[65] Y. Yuzhelevski, M. Yuzhelevski, and G. Jung. Random telegraph noise analysis in time domain. *Review of Scientific Instruments*, 71(4):1681–1688, 2000.

[66] Jiayang Zhang, Zhe Zhang, Rusheng Wang, Zixuan Sun, Zuodong Zhang, Shaofeng Guo, and Ru Huang. Comprehensive study on the "anomalous" complex rtn in advanced multi-fin bulk finfet technology. In *2018 IEEE International Electron Devices Meeting (IEDM)*, pages 17.3.1–17.3.4, 2018.