System Identification Tools for Precise Control of Motors

Project Manager: Shreya Jha (Georgia Institute of Technology)
Member: Shaymaa Mahmoud (American University in Cairo)
Member: Michelle Bang (Oregon State University)
Member: Emre Isik (University of Cambridge)
Academic Mentor: Chunyang Liao
Industry Mentor: Srilakshmi Pattabiraman

August 22, 2024

Outline

1. Introduction
2. Vector Fitting Algorithm
   - Overview
   - Numerical Results
3. Neural Networks
   - Overview
   - ReLU Neural Networks
   - Rational Neural Networks
     - Proposed Architecture 1
     - Proposed Architecture 2
4. Summary

Introduction: Analog Devices Inc. (ADI)

- Semiconductor company specializing in signal processing, and power management technologies, headquartered in Wilmington, Massachusetts.
- Solutions to drive advancements in manufacturing, EVs, healthcare, drones, 3D printers and many other industries.

Introduction: The Project Motivation

- Analog Devices, Inc. provides precision motors and advanced control solutions, necessitating precise control of a motor's trajectory.
- Computing an accurate input-output relationship of any given system is vital in achieving this position control
- Transfer Function (TF) describes a component's behavior with respect to any given input.

Introduction: The Project Motivation

Figure: Blackbox System

- Transfer Functions can be computed using physics based principles, but this becomes more difficult with increased complexity.
- We can use data-driven methods to compute the input-output relationship, such as Vector Fitting.
- Vector Fitting works well with clean data, but performs poorly with noisy data.

Introduction: The Project Motivation

Figure: Blackbox System

- Transfer Functions can be computed using physics based principles, but this becomes more difficult with increased complexity.
- We can use data-driven methods to compute the input-output relationship, such as Vector Fitting.
- Vector Fitting works well with clean data, but performs poorly with noisy data.
- **GOAL: Use machine learning techniques in conjunction with traditional data-driven methods to compute a system's input-output relationship.**

Contents

1 Introduction

2 Vector Fitting Algorithm
- Overview
- Numerical Results

3 Neural Networks
- Overview
- ReLU Neural Networks
- Rational Neural Networks
  - Proposed Architecture 1
  - Proposed Architecture 2

4 Summary

- The Vector Fitting algorithm is designed to solve the following problem.

Setting Up the Problem:

For a linear time-invariant system:

$$y(t) = \int_{-\infty}^{+\infty} h(t-\tau)x(\tau)d\tau$$

- $y(t)$ is the output signal in t-domain.
- $x(t)$ is the input signal in t-domain.
- $h(t)$ is the unknown impulse response.

- **NEXT STEP: Convert to s-domain (frequency-domain) by applying the Laplace transform.**

Overview Cont.

Problem Statement: Input/Output Relationship

$$Y(s) = H(s)X(s) \quad \Longrightarrow \quad H(s) = \frac{Y(s)}{X(s)}$$

- $s = j\omega$, where ω is frequency and $j = \sqrt{-1}$.
- $Y(s)$ is the output signal in s-domain.
- $X(s)$ is the input signal in s-domain.
- $H(s)$ is our unknown transfer function.

- **GOAL: We want to approximate H(s).**

Vector Fitting Algorithm

We can approximate $H(s)$ with $\hat{H}(s)$ with VF:

$$\hat{H}(s) = \frac{\sum_{i=0}^{n} a_i s^i}{\sum_{i=0}^{n} b_i s^i} = r_0 + \sum_{i=1}^{n} \frac{r_i}{s - p_i}$$

- $\{p_i\}_{i=1}^{n}$ are poles that are initialized and updated within the algorithm and $\{r_i\}_{i=1}^{n}$ are unknown residues.
- Given K samples $(j\omega_1 \ldots j\omega_K)$ and corresponding $H(j\omega_1) \ldots H(j\omega_K)$ as our inputs, we aim to minimize the mean squared error:

$$e = \frac{1}{K} \sum_{k=1}^{K} |\hat{H}(j\omega_k) - H(j\omega_k)|^2$$

- The algorithm iteratively minimizes the mean squared error through least squares regressions after fixing the unknowns in the denominator (Gustavsen and Semlyen, 1999).

Contents

1. Introduction
2. Vector Fitting Algorithm
   - Overview
   - Numerical Results
3. Neural Networks
   - Overview
   - ReLU Neural Networks
   - Rational Neural Networks
     - Proposed Architecture 1
     - Proposed Architecture 2
4. Summary

Synthetic Data Generation

- Synthetic data was generated in Python by defining an arbitrary transfer function of degree 3.
- The data consisted random frequencies that were log-spaced apart with its corresponding H values.
- Two types of synthetic data were tested:
  - Clean, noise-free data.
  - Noisy data.
- Noise was randomly drawn from a standard normal distribution and added separately to the phase and magnitude of H.
- The strength of the noise level can be adjusted depending on the Signal-to-Noise Ratio (SNR).

$$\mathrm{SNR} = 10 \log\left(\frac{\text{signal}}{\text{noise}}\right)$$

Vector Fitting Algorithm trained on Synthetic Clean Data

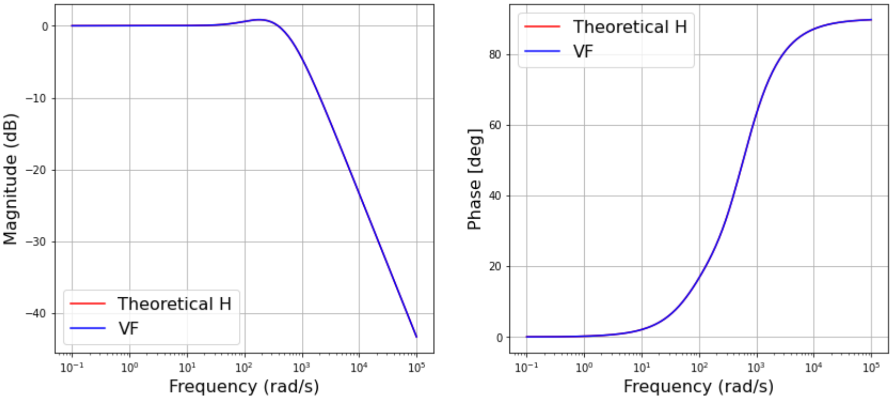


Figure: Magnitude and Phase vs. Frequency

Given $H(s) = a + bj$,

- $\text{Magnitude}(s) = \sqrt{a^2 + b^2}$
- $\text{Phase}(s) = \tan^{-1}(\frac{b}{a})$

Vector Fitting Algorithm trained on Synthetic Noisy Data

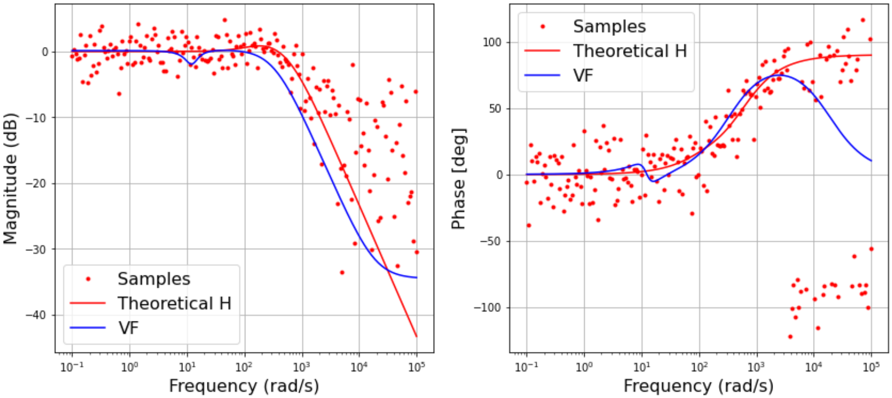


Figure: Magnitude and Phase vs. Frequency

Simulated Data Generation

- Simulated data is generated from the *lsim*() function in MATLAB.
- Both real and simulated data is taken in the time-domain.
  - A Discrete Fourier Transform using the Fast-Fourier Transform function from SciPy is used to transform the data into s-domain.
- Simulated data uses the same transfer functions derived from the Trinamic motors for the following loops:
  1. Current Open Loop, order: 2
  2. Current Closed Loop, order: 3
  3. Velocity Loop, order: 7
- Order represents the degree of the transfer function.
  - ex.

$$\frac{s+2}{s^2+s+3}$$

    is a second order transfer function.

Current Loop

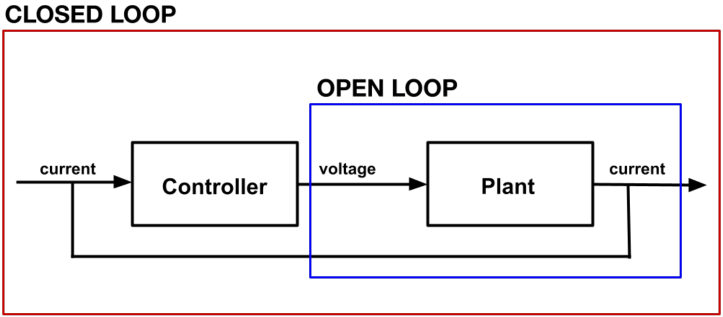


Figure: Control Loop Diagram of Current Loop.

Velocity Loop

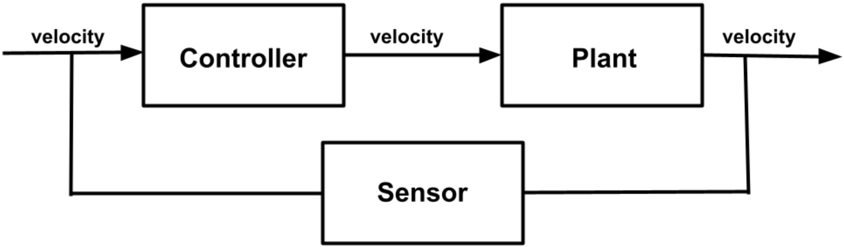


Figure: Control Loop Diagram of Velocity Loop.

Vector Fitting Algorithm trained on Simulated Data

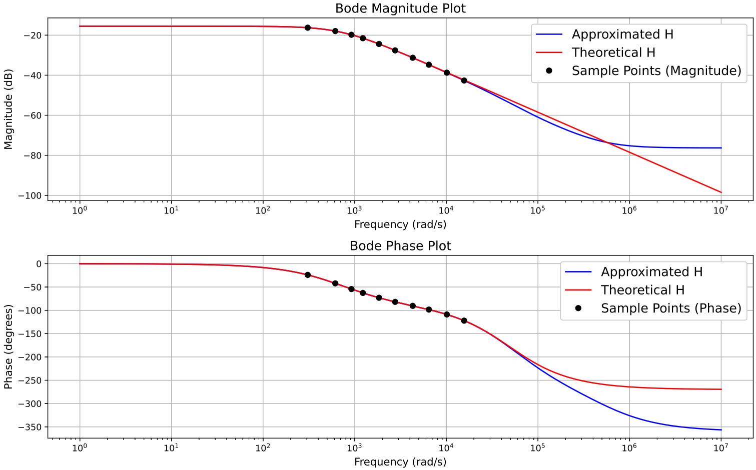


Figure: Open Loop- Magnitude and Phase vs. Frequency.

Real Data Generation

- We generated real data with the stepper motors provided by Analog.
- 50 singular sine waves were fed into the system. Each wave is an input to our algorithm.

Vector Fitting Algorithm trained on Real Data

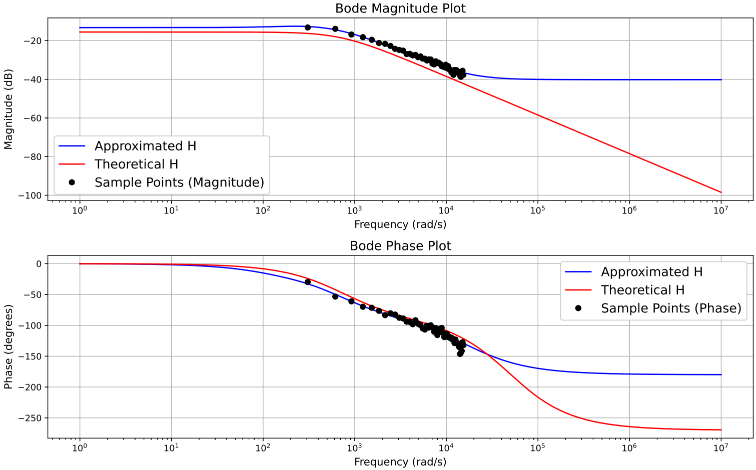


Figure: Open Loop- Magnitude and Phase vs. Frequency.

Vector Fitting Algorithm trained on Real Data

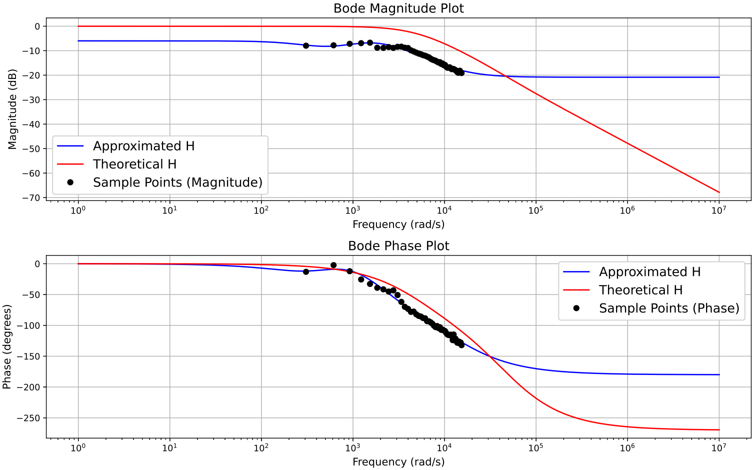


Figure: Closed Loop- Magnitude and Phase vs. Frequency.

Vector Fitting Algorithm trained on Real Data

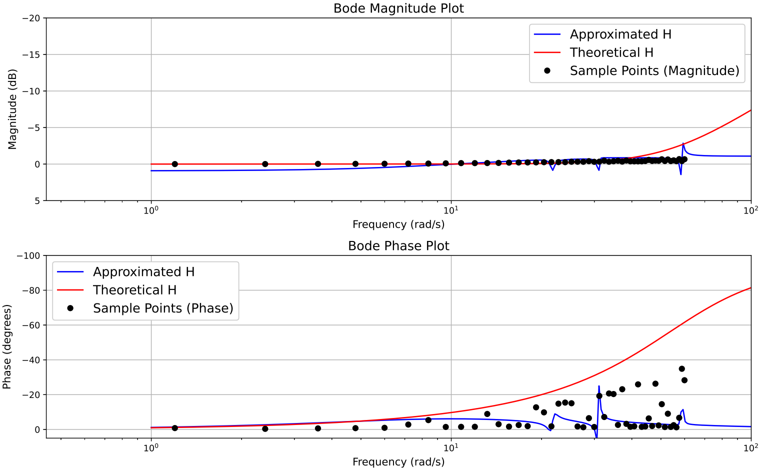


Figure: Velocity Loop- Magnitude and Phase vs. Frequency.

Contents

1. Introduction
2. Vector Fitting Algorithm
   - Overview
   - Numerical Results
3. Neural Networks
   - Overview
   - ReLU Neural Networks
   - Rational Neural Networks
     - Proposed Architecture 1
     - Proposed Architecture 2
4. Summary

Neural Networks Overview

- Neural networks are composed of multiple layers, each designed to perform linear operation and then apply non-linear activation function
- Mathematically, the function can be expressed as $\mathbf{x} \rightarrow f(\mathbf{W}\mathbf{x} + \mathbf{b})$, where $\mathbf{W}$ is the weight matrix, $\mathbf{b}$ is the bias vector, and f is a non-linear activation function.

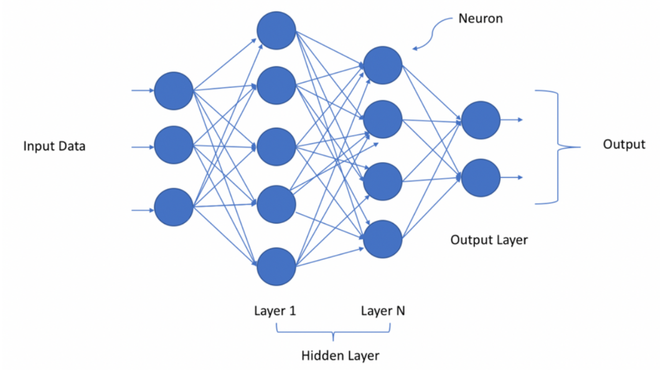


Figure: Structure of deep neural network

Neural Networks Overview

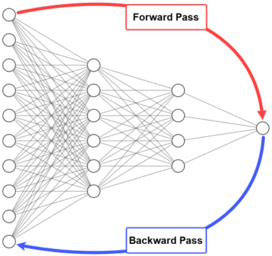


Figure: Forward propagation and backward propagation.

- In the forward pass, input data is passed through the neural network layer by layer, with each layer performing calculations and transformations, to create an output prediction.
- The backward pass involves computing the gradients of the loss function with respect to each parameter using chain rule, which is then used to update the parameters to minimize the loss.

Contents

1. Introduction
2. Vector Fitting Algorithm
   - Overview
   - Numerical Results
3. Neural Networks
   - Overview
   - ReLU Neural Networks
   - Rational Neural Networks
     - Proposed Architecture 1
     - Proposed Architecture 2
4. Summary

ReLU Activation Function

Rectified Linear Unit (ReLU) Equation

$$\text{ReLU}(x) = \max(0, x)$$

- ReLU is widely used because it is computationally efficient and fast to evaluate.
- Many theoretical studies have compared deep and shallow ReLU networks from the perspective of approximation theory (Daubechies et al., 2019).
- Weight matrix $\mathbf{W}$ and bias vector $\mathbf{b}$ are the training parameters for ReLU neural networks.

Neural Network Structure

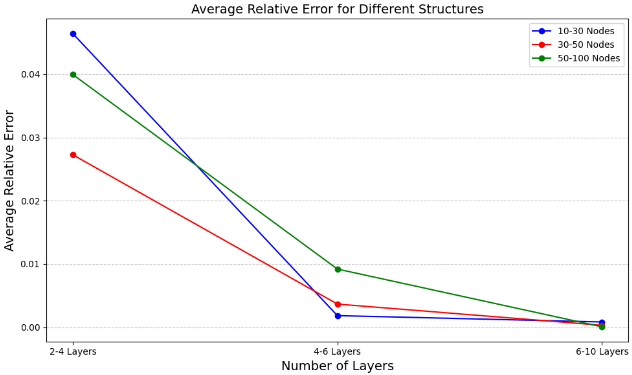


Figure: Accuracy score analysis for different structures

- Highly sensitive to both number of layers and number of nodes
- Requires many layers and nodes, especially with noisy data - high computational complexity

Data Requirements

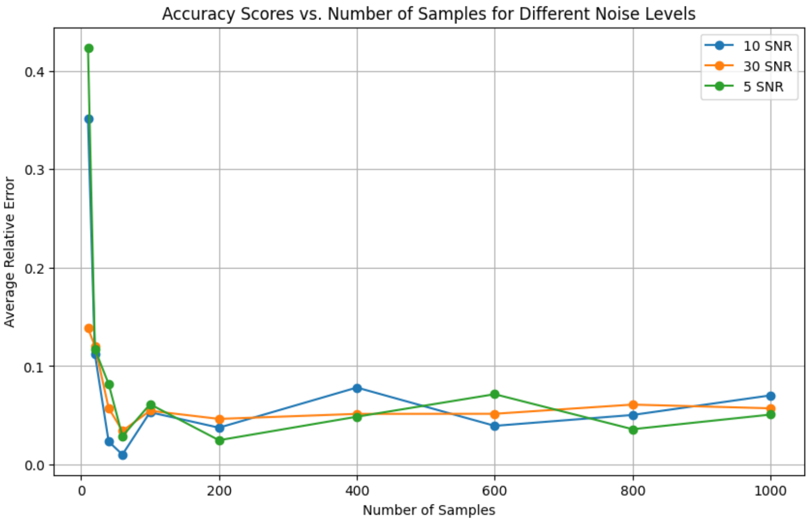


Figure: Data requirements for training at different noise levels

Data Requirements

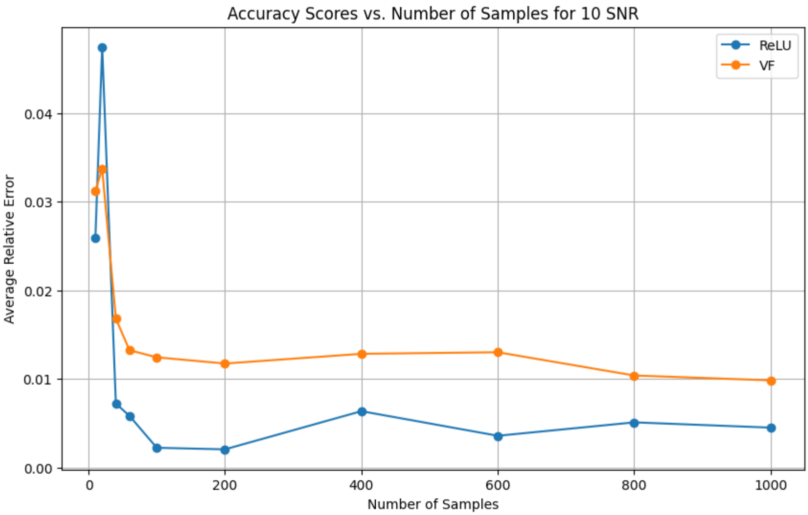


Figure: 10 SNR Deep NN and VF comparison

Main Takeaways

- For number of samples below 50, noise distorts accuracy significantly
- ReLU neural network is robust to noise given that there is sufficient amount of training data
- Nevertheless, it requires 200+ samples to achieve reasonable accuracy - data hungry
- Impossible to analyse the transfer function approximated by the neural network

Contents

1 Introduction

2 Vector Fitting Algorithm
- Overview
- Numerical Results

3 Neural Networks
- Overview
- ReLU Neural Networks
- Rational Neural Networks
  - Proposed Architecture 1
  - Proposed Architecture 2

4 Summary

Rational Activation Function

Rational Activation Function Equation

$$f(x) = \frac{\sum_{i=1}^{n} a_i x^i}{\sum_{i=1}^{n} b_i x^i}$$

- The coefficients $\{a_i\}_{i=1}^n$ and $\{b_i\}_{i=1}^n$ in the rational activation function are trainable parameters.
- Rational activation functions mirror the mathematical properties of the target transfer function $H(s) = \frac{Y(s)}{X(s)}$, making the entire neural network a rational function.

Contents

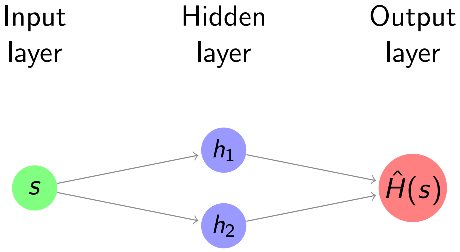
1 Introduction

2 Vector Fitting Algorithm
- Overview
- Numerical Results

3 Neural Networks
- Overview
- ReLU Neural Networks
- Rational Neural Networks
  - Proposed Architecture 1
  - Proposed Architecture 2

4 Summary

Rational NN Architecture 1 - Explanation



- The number of nodes in the hidden layer depends on the order of the system.
- For a second-order system, use two nodes:

$$h_1 = \frac{a_1 s + b_1}{c_1 s + d_1}, \quad \text{and} \quad h_2 = \frac{a_2 s + b_2}{c_2 s + d_2}$$

- The summation of these nodes approximates the system's transfer function:

$$\hat{H}(s) = \sum_{i=1}^{n} h_i$$

- Coefficients are initialized as:

$$a_i, b_i, c_i, d_i \sim \mathcal{N}(0, 1)$$

Rational NN Architecture 1 - Synthetic Data

We use the second-order transfer function to generate synthetic data.

Training Samples	**Noise Level (SNR)**	**Rational Model Error**	**VF Model Error**
10	Noise-free	0.08279	0.00000
50	Noise-free	0.01154	0.00000
100	Noise-free	0.000246	0.00000
10	10 dB	0.30738	0.40425
50	10 dB	0.13575	0.53133
100	10 dB	0.03900	0.70630
10	5 dB	0.53178	1.07197
50	5 dB	0.16444	0.93528
100	5 dB	0.02870	1.21592

Table: Average test relative error comparison between rational and VF algorithm for 1000 test samples

- The VF model shows insignificant error in noise-free conditions, but its performance degrades significantly in high noise.
- As training samples increase, rational networks have lower test error .
- Rational model performs better than VF in noisy scenarios.

Rational NN Architecture 1 - Simulated Data

We use the open loop transfer function for simulating the training data

Training Samples	Noise Level (SNR)	Rational Model Error	VF Model Error
10	Noise-free	1.1759	0.0002
50	Noise-free	0.7107	0.00011
10	10 dB	1.8889	0.1070
50	10 dB	1.1093	0.7093
10	5 dB	1.9220	1.8763
50	5 dB	1.0846	3.8102

Table: Average test relative error comparison between rational and VF models for 1000 test samples

- In noisy conditions, the performance of the Rational model improves as the number of samples increases.
- VF model struggles with higher noise levels, especially with a higher number of training samples.

Rational NN Architecture 1 - Real Data

Loop Type (50 samples)	Rational Model Error	VF Model Error
Open Loop	0.328912	0.3889
Closed Loop	0.292844	0.4025

Table: Average test relative error comparison between rational and VF model for 1000 test samples

- VF model performs better in open loop than in closed loop.
- Performance differences highlight the models' sensitivity to loop type in real-world data.

Rational NN Architecture 1 - Real Data

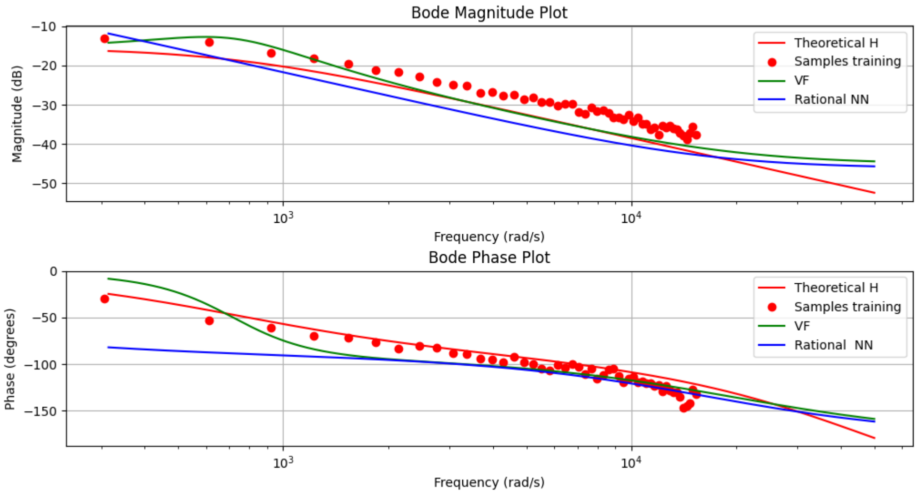


Figure: Performance with 50 real samples for open loop

Rational NN Architecture 1 - Real Data

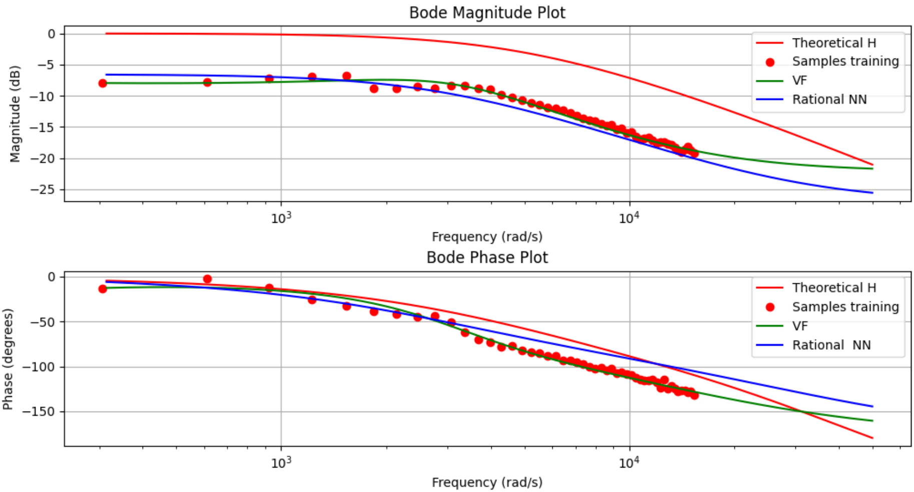


Figure: Performance with 50 real samples for closed loop

Main Insights for Proposed Architecture 1

- **Noise Sensitivity:** While the Rational NN is more resilient to noise, the VF performance degrades significantly with increasing noise levels, especially in synthetic data.
- **Scalability:** The Rational NN model scales effectively with an increase in training data, reducing the test error substantially as the number of samples grows, consistent across both synthetic and simulated data.

Contents

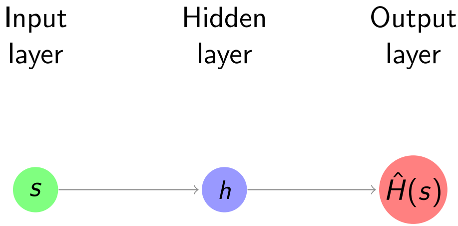
1 Introduction

2 Vector Fitting Algorithm
- Overview
- Numerical Results

3 Neural Networks
- Overview
- ReLU Neural Networks
- Rational Neural Networks
  - Proposed Architecture 1
  - Proposed Architecture 2

4 Summary

Rational NN Architecture 2 - Explanation



- $h(s) = \frac{\sum_{i=1}^{n} a_i s^i}{\sum_{i=1}^{n} b_i s^i}$, $\{a_i\}_{i=1}^{n}$ and $\{b_i\}_{i=1}^{n}$ are the parameters tuned in training.
- In this architecture, n is chosen depending on the order of the system being modeled.
- There is only one node in the hidden layer regardless of the order of system.
- This architecture uses gradient descent whereas Vector Fitting uses a least squares regression.

Rational NN Architecture 2 - Initialization Methods

- **Random Initialization** chooses the values of $\{a_i\}_{i=1}^n$ and $\{b_i\}_{i=1}^n$ by drawing from a normal distribution.
- **Vector Fitting Initialization** sets the values of $\{a_i\}_{i=1}^n$ and $\{b_i\}_{i=1}^n$ to be the coefficients that the Vector Fitting algorithm returns.

Rational NN Architecture 2 - Synthetic Data

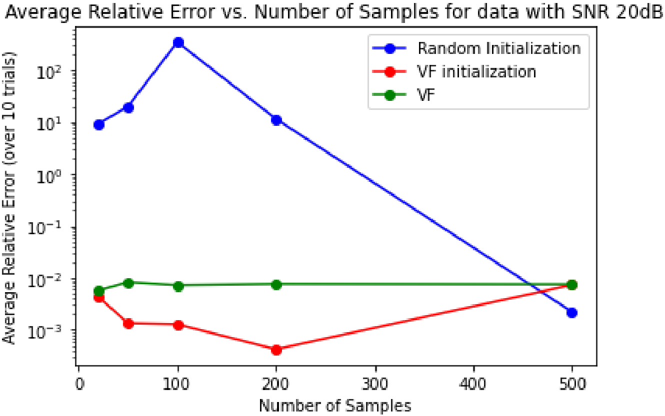


Figure: Average relative error by number of training samples with SNR 20 dB noisy data.

Rational NN Architecture 2 - Synthetic Data

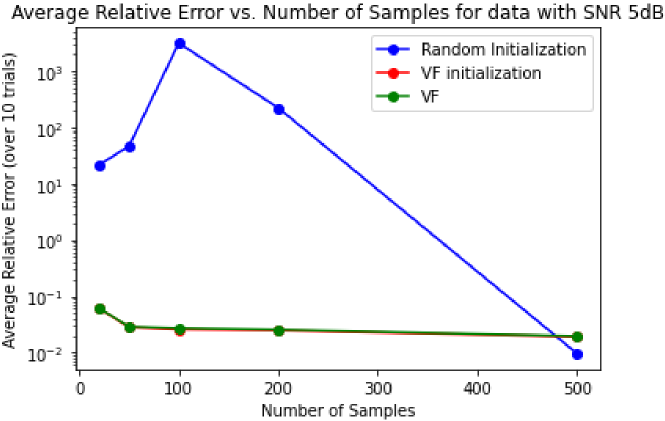


Figure: Average relative error by number of training samples with SNR 5 dB noisy data.

Rational NN Architecture 2 - Real Data

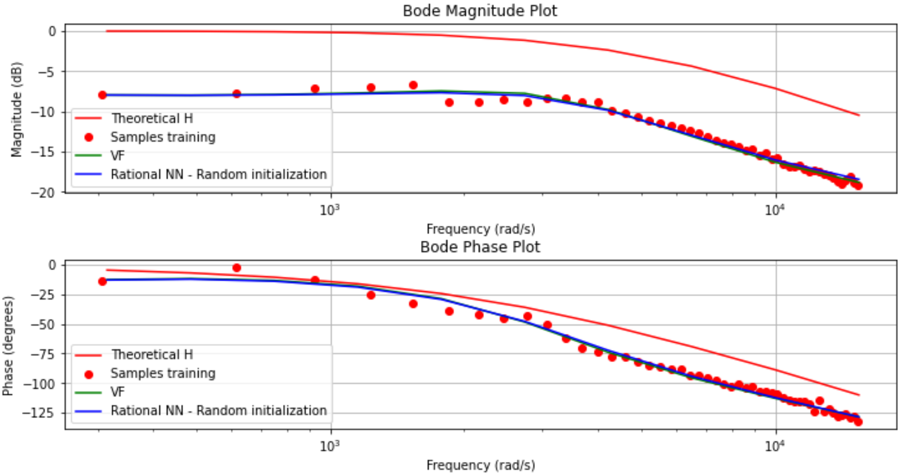


Figure: Vector fitting and complex rational neural net with vector fitting initialization trained on 50 real data samples (closed loop).

Rational NN Architecture 2 - Real Data

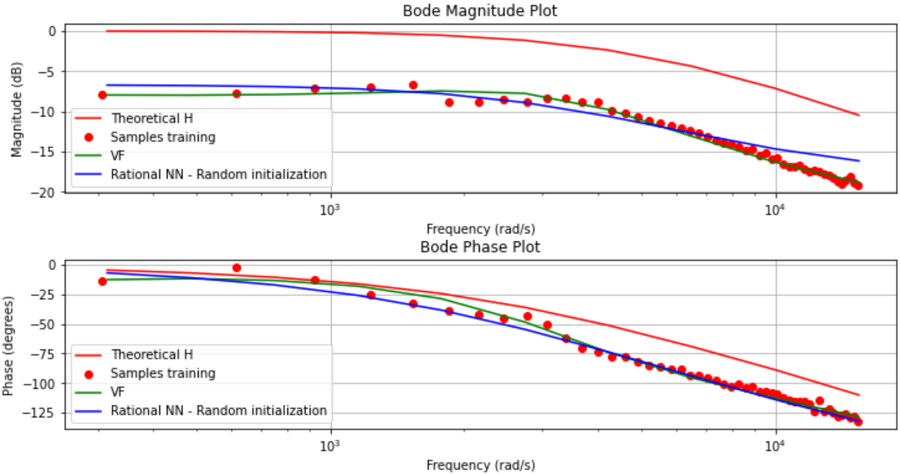


Figure: Vector fitting and complex rational neural net with random initialization trained on 50 real data samples (closed loop).

Contents

1. Introduction
2. Vector Fitting Algorithm
   - Overview
   - Numerical Results
3. Neural Networks
   - Overview
   - ReLU Neural Networks
   - Rational Neural Networks
     - Proposed Architecture 1
     - Proposed Architecture 2
4. Summary

Summary

- Implemented the Vector Fitting algorithm to compute the input/output relationship between frequency and responses, and assessed its performance with clean and noisy data
- Developed a deep learning neural net to directly compute the input/output relationship which performs better than Vector Fitting with noisy data
- Developed a rational neural net that has less trainable parameters than the deep neural net and performs better than Vector Fitting with noisy data
- Developed a rational neural net that is initialized with the Vector Fitting outputs and tunes the results to achieve better performance

Acknowledgements

- Thank you to Analog Devices, Inc. for proposing this project and sponsoring us!
- Thank you to IPAM for also sponsoring us!
- Special thanks to our industry mentor Srilakshmi Pattabiraman and our academic mentor Chunyang Liao for helping us throughout the program.

Thanks for listening!
Questions?