REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
08/20/2021	Technical Report		

4. TITLE AND SUBTITLE

Deconvolution of Temporally Under-Resolved Image Sequences for Coupled Dynamical Systems

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER	5e. TASK NUMBER	5f. WORK UNIT NUMBER
Q2DF		Q2DF

6. AUTHOR(S)

Benjamin Maloy (Tufts University), Addie McCurdy (University of St. Thomas), Nagaprasad Rudrapatna (Duke University), Sharadiant Turner (Spelman College)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Air Force Research Laboratory Rocket Propulsion Division In-Space Propulsion Branch Edwards AFB, CA 93524	AFRL/RQRS	AFRL-RQ-ED-TR-2021-228

12. DISTRIBUTION/AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

13. SUPPLEMENTARY NOTES

PA Clearance Number: AFRL-2021-3009, Clearance Date: 08 Sep 2021
This material is based on work supported by the Air Force Research Laboratory.
Institute for Pure and Applied Mathematics (IPAM); an institute of the National Science Foundation, located at UCLA. Research in Industrial Projects for Students (RIPS) NSF Grant (DMS) 1925919

14. ABSTRACT

The Hall-effect thruster, a pervasive in-space propulsion device, can be effectively modelled as a coupled dynamical system using a strange attractor. This fact allows the use of Takens' Embedding Theorem (and therefore convergent cross mapping) to analyze different component variables in the system. Namely, if the system state is given by (X(t); Y (t)) and Y (t) is corrupted by noise while X(t) remains comparatively clear, X(t) can be used to retrieve the denoised Y (t) signal. Previous work by the Air Force Research Laboratory (AFRL) and past RIPS-AFRL research teams has yielded methods to denoise corrupted signals, as well as reconstruct underlying signals from temporally under-resolved time series (deconvolution). Both denoising and deconvolution involve mapping a clean reference signal to phase space and averaging the corrupted signal within partitions of phase space. The original work by the AFRL utilized a grid-based partition, while the 2020 RIPS-AFRL research team used Voronoi diagram-based partitions of phase space. In this work, we primarily extend the capabilities of the Voronoi diagram-based partition to work with deconvolution and substantially decrease the computational runtime of the deconvolution algorithm so that it can be applied to a system where many corrupted signals are coupled to the same clean reference signal. Additionally, we develop optimization strategies that increase the accuracy of our reconstructions of target signals.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT	b. ABSTRACT	C. THIS PAGE		
Unclassified	Unclassified	Unclassified	SAR	78

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Robert Martin	N/A

PREVIOUS EDITION IS OBSOLETE.

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

Research in Industrial Projects for Students

Sponsor

Air Force Research Laboratory

Final Report

Deconvolution of Temporally Under-Resolved Image Sequences for Coupled Dynamical Systems

Student Members

Benjamin Maloy (Project Manager), *Tufts University*,
benjamin.maloy@tufts.edu

Addie McCurdy, *University of St. Thomas*

Nagaprasad Rudrapatna, *Duke University*

Sharadiant Turner, *Spelman College*

Academic Mentor

Abigail Hickok, ahickok@math.ucla.edu

Sponsoring Mentors

Dr. Robert Martin, robert.martin.101@us.af.mil

Dr. Daniel Eckhardt, daniel.eckhardt.3@us.af.mil

External Consultant

Dr. Samuel Araki, samuel.araki.ctr@us.af.mil

August 20, 2021

This project was jointly supported by the Air Force Research Laboratory and NSF Grant ⟨DMS⟩ 1925919.

Abstract

The Hall-effect thruster, a pervasive in-space propulsion device, can be effectively modelled as a coupled dynamical system using a strange attractor. This fact allows the use of Takens' Embedding Theorem (and therefore convergent cross mapping) to analyze different component variables in the system. Namely, if the system state is given by $(X(t), Y(t))$ and $Y(t)$ is corrupted by noise while $X(t)$ remains comparatively clear, $X(t)$ can be used to retrieve the denoised $Y(t)$ signal. Previous work by the Air Force Research Laboratory (AFRL) and past RIPS-AFRL research teams has yielded methods to denoise corrupted signals, as well as reconstruct underlying signals from temporally under-resolved time series (deconvolution). Both denoising and deconvolution involve mapping a clean reference signal to phase space and averaging the corrupted signal within partitions of phase space. The original work by the AFRL utilized a grid-based partition, while the 2020 RIPS-AFRL research team used Voronoi diagram-based partitions of phase space. In this work, we primarily extend the capabilities of the Voronoi diagram-based partition to work with deconvolution and substantially decrease the computational runtime of the deconvolution algorithm so that it can be applied to a system where many corrupted signals are coupled to the same clean reference signal. Additionally, we develop optimization strategies that increase the accuracy of our reconstructions of target signals.

Acknowledgments

We would like to express our sincere gratitude to the following people and organizations who made this work possible. First, we would like to thank our industry mentors, Robert Martin and Daniel Eckhardt, as well as our external consultant, Samuel Araki. Their support and suggestions aided in the success of the project and the accomplishment of our goals. We would also like to thank our academic mentor, Abigail Hickok, who was devoted to our success and proved to be extremely helpful in grasping difficult theoretical aspects of the work. We appreciate the entire IPAM team for allowing us to work in a safe environment in these stressful times, and would especially like to thank the RIPS director, Susana Serna. We would like to thank the National Science Foundation, which offered its support of this program through Grant No. 1925919, and the Air Force Research Laboratory for their financial support and technical expertise.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The In-Space Propulsion Branch of the Air Force Research Laboratory is dedicated to the advancement of chemical and electrical propulsion devices. One propulsion method of interest is the Hall-effect thruster (HET), which relies on the ionization of noble gases to produce plasma that propels a vehicle equipped with the thruster forward. It was shown in [5] that it is fruitful to consider the HET as a coupled dynamical system. There are many different signals from the HET that can be used to define the state of the system, and each of these signals is driven by the dynamics of the HET system (visualized in Figure 1.1).

An attractor in dynamical systems is a state that the system tends towards, while a strange attractor system is one where the state oscillates chaotically around this attractor. The convective and reactive processes in the HET induces nonlinear fluctuations around more stable quasiperiodic behavior, which draws a likeness between the HET and strange attractors. This fact allows us to leverage powerful results from dynamical systems theory, including Taken's Embedding Theorem [10] and convergent cross mapping [9], to analyze the HET system.

This report reviews background information from dynamical system theory and discusses how to apply ideas from convergent cross mapping to denoise and deconvolute poorly resolved signals that are coupled with a clear reference signal. For the HET, the signals measured are electrical current readings through different parts of the thruster; some of these signals can be read with high precision, while others are subject to significant noise. The general idea is that we can find the underlying signal from the noisy signal by considering its relation (via Taken's Embedding Theorem) to the clear signal. Furthermore, even if the noisy signal is temporally under-resolved (that is, if you are only given the noisy signal for some discrete time measurements), the underlying signal can be reconstructed. We can test denoising and deconvolution on well-known strange attractors, like the Lorenz system (see Section 8.1.1).

Takens' Embedding Theorem highlights the connection between a state manifold, like $(X(t), Y(t))$, and the 'shadow manifold' that is constructed by lagging one coordinate of the state and embedding it in phase space. A shadow manifold, which for a two-dimensional embedding is of the form $(X(t), X(t-\tau))$, can be constructed for either time series projection of the state manifold. Takens' crucial insight was that the shadow manifold is diffeomorphic to the state manifold; therefore, since each time series projection's shadow manifold is diffeomorphic to the same state manifold, each

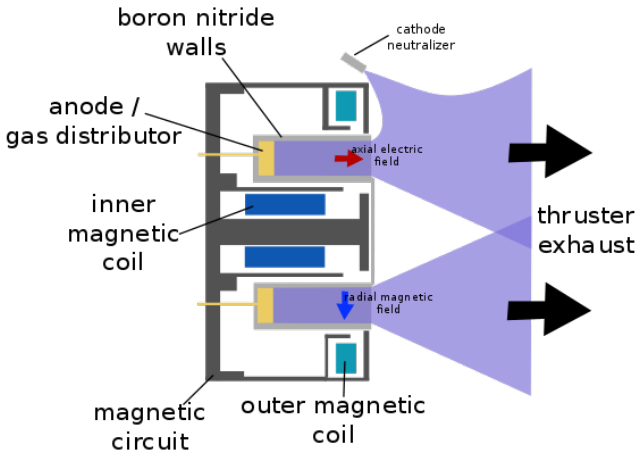


Figure 1.1: A diagram of a Hall-effect thruster. Each of the labeled components could represent a dimension of the dynamical system that models the HET by measuring the electrical current flowing through that component.

shadow manifold is diffeomorphic to each other. Araki and collaborators demonstrated algorithms to denoise and deconvolute that partitioned phase space into a rectilinear grid. Noise was reduced by averaging the noisy signal in grid cells in phase space to recover the underlying signal [1]. The denoising algorithm is presented in Section 2.4.

We replaced the rectilinear grid partition of phase space from [1] with a Voronoi diagram-based partition (nearest neighbor partition) for the purpose of deconvoluting a temporally under-resolved signal; this partition is discussed in Section 2.3. That is, regions of phase space with more data points will have smaller cells than those with more sparse representation. This new way to partition phase space was successful and allowed us to deconvolute temporally under-resolved signals that were constructed from well-defined attractor systems. We also sped up the deconvolution algorithm so that it can be applied to a large dataset with many noisy signals coupled to one clean reference signal.

After discussing background information in Chapter 2, the deconvolution algorithm is detailed in Chapter 3. Chapter 4 covers how to find the optimal parameters for the algorithm, and Chapter 5 shows how well the algorithm did in reconstructing known signals and experimental signals. Finally, Chapter 6 discusses applying the deconvolution algorithm to movies where each pixel is convoluted and coupled to the same reference signal.

Chapter 2

Background

As mentioned in Chapter 1, the Hall-effect thruster can be modelled as a dynamical system with a strange attractor, say $S(t) = \{x_1(t), x_2(t)...\}$. We can model the dimensions of $S(t)$ by attaching probes and gathering various measurements. For example, the anode (Figure 2.1a), cathode (Figure 2.1b), ring3 (Figure 2.1c), and ring4 (Figure 2.1d) current readings are all dimensions of $S(t)$. As in [1], we consider the anode signal to be relatively noiseless and use it as our clean reference signal to denoise the other dimensions of the HET system.

We can use other systems of the same type, such as the Lorenz system, referenced in Section 8.1. For most of our work, we use simulated data from the Lorenz system (Figure 2.2) to test the effectiveness of our algorithms. In particular, we projected the X (Figure 2.3a) and Y (Figure 2.3b) dimensions of the Lorenz system to time series, which we used as our clean reference signal and the signal to be reconstructed, respectively. More details of our data simulation are given in Step 2 of Algorithm 2 in Chapter 3.

Since the Hall-effect thruster can be modelled as a dynamical system with a strange attractor, we can apply results from dynamical systems theory, namely Takens' Theorem and convergent cross mapping.

2.1 Takens' Embedding Theorem

Theorem 2.1 *Let M be a compact manifold of dimension d, and let X be a time series representation of a dimension of M such that the projection onto X is generic. For all $E \geq 2d + 1$, if M_X is an E-dimensional shadow manifold of M formed by plotting E time lags of X (by multiples of τ), that is,*

$$M_X(t) = (X(t), X(t - \tau), \ldots, X(t - (E - 1)\tau))$$

at each time t, then there exists a diffeomorphism between M_X and M.

A shadow manifold refers to plotting a time series against the same time series where each value is shifted in time by τ. The number of lagged copies of the original time series, $X(t)$, is the embedding dimension. See Figure 2.4b for a two-dimensional shadow manifold. To better understand the relevance of Takens' Embedding Theorem, let's examine an application. Though the Lorenz attractor (see

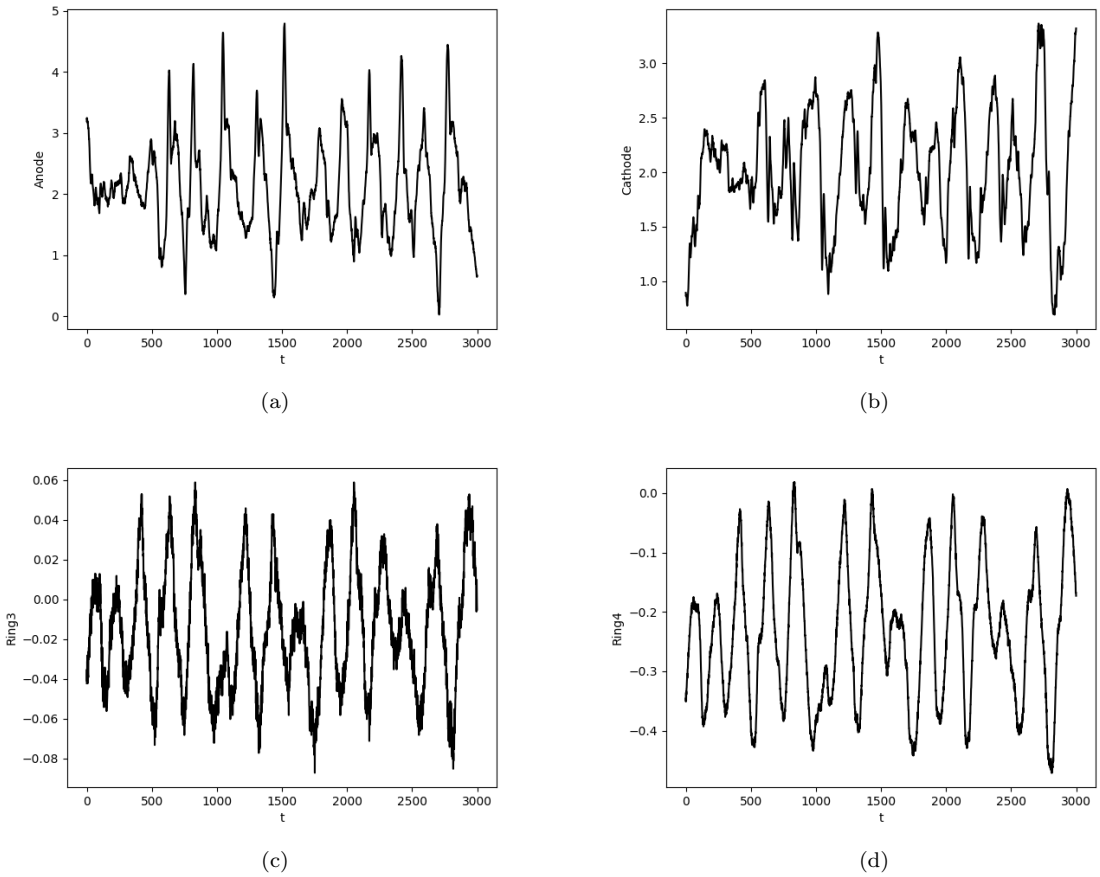


Figure 2.1: Dimensions of the HET system (a) anode current (b) cathode current (c) ring3 current (d) ring4 current

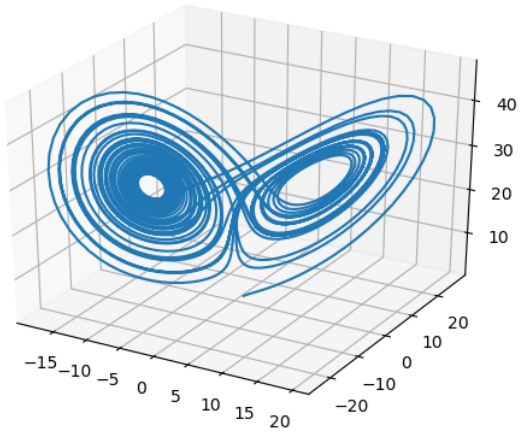
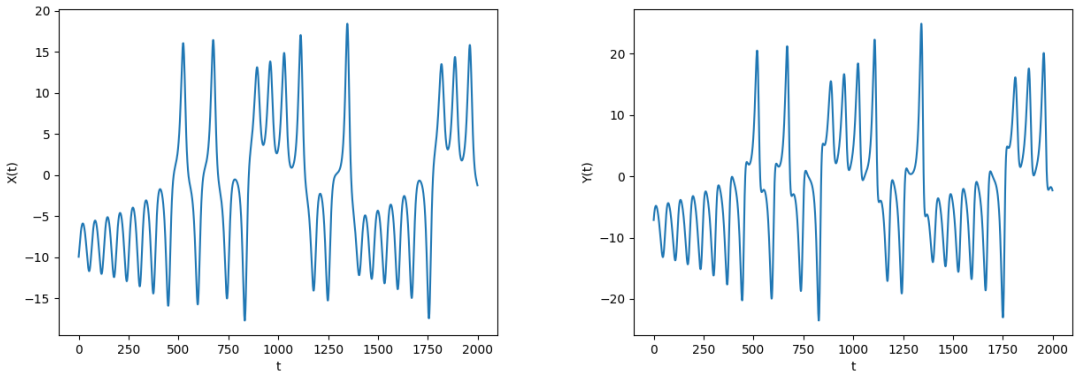
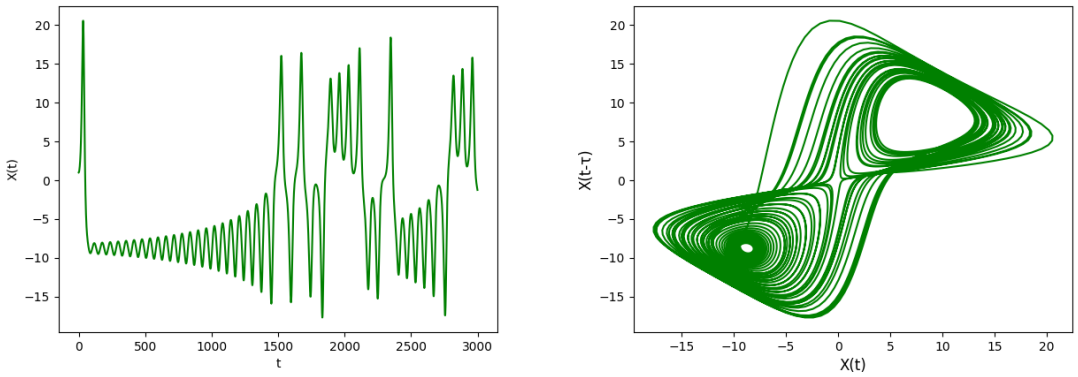


Figure 2.2: The state (butterfly) manifold for the Lorenz system



(a) (b)

Figure 2.3: Time series projections from the Lorenz system (a) X dimension (b) Y dimension



(a) (b)

Figure 2.4: (a)$X(t)$ from the Lorenz system and (b) its corresponding time-delayed shadow manifold, M_X

Section 8.1.1) is not exactly a manifold, there is an extension of Theorem 2.1 that applies to fractal attractors like this one. Let M be the Lorenz attractor system where $M = (X(t), Y(t), Z(t))$. $X(t)$, for instance[1], can be used to construct a two-dimensional shadow manifold

$$M_X(t) = (X(t), X(t - \tau))$$

which is an embedding that maps t to M_X (visualized in Figure 2.4b). Takens' Theorem says that, if M_X is sufficiently high dimensional, there is a diffeomorphism between M_X and the Lorenz state attractor. The consequences of this relationship are expanded upon in later sections, namely Section 2.2 and Algorithms 1 and 2.

[1]Shadow manifolds can be constructed analogously with $Y(t)$

2.2 Convergent Cross Mapping

Convergent cross mapping (CCM) is an algorithm that can help distinguish causality from spurious correlation in time series from dynamical systems [9]. In CCM, the ability (or inability) of one signal to reconstruct another is used to test whether or not the signals come from a shared dynamical system. In this work, we do not test for causality. Rather, our signal reconstruction algorithm builds off of the ideas developed in CCM, leveraging the fact that our signals are known to originate from a coupled dynamical system.

Given a system $M(t) = (X(t), Y(t))$, the CCM algorithm reconstructs the signal $Y(t)$ from $X(t)$ using the diffeomorphisms between M and the shadow manifolds M_X and M_Y, which we call g_X and g_Y respectively.

To generate the cross mapped approximation of the signal $Y(t)$, we find the corresponding time lagged coordinates on M_X. Based on the one-to-one mappings g_X^{-1} and g_Y, we know that nearest neighbors on M_X will also be nearest neighbors on M_Y, and thus we can reconstruct $Y(t)$. The adapted denoising and deconvolution algorithms that build off this method are described in detail in Section 2.4 and Chapter 3 respectively.

2.3 Voronoi Diagrams

In [1], denoising is accomplished using a structured mesh, or a grid. The 2020 RIPS-AFRL team extended this denoising algorithm to use an unstructured mesh, specifically Voronoi diagrams. Voronoi diagrams are constructed using a sample of points from the shadow manifold M_X in phase space, which are called 'cell representatives'. Phase space is then partitioned into regions that are nearest to each cell representative. For example, in the two-dimensional case, cell boundaries are created by intersecting the perpendicular bisectors between each cell, as shown in Figure 2.5.

By definition, any point in a given Voronoi cell will be closer to the cell representative of that cell than any other cell representative. We use these Voronoi cells as replacements of grid cells for several reasons. First, using Voronoi cells guarantees that no cell will be empty, which is an important condition for our denoising and deconvolution algorithms, outlined below. Second, given the random selection of cell representatives, areas more densely populated with points will likely have more cell representatives, leading to smaller cells and a more uniform distribution of points per cell. These two facts are demonstrated in Figure 2.6.

It is critical to partition phase space into the correct number of cells, as with too few cells too many points will be averaged in a cell and the dynamics of the system will not be captured. Conversely, with too many cell representatives, there will be too few points in each cell and the average values in each cell will not reduce the noise. Thus, we must optimize the number of Voronoi cells as a function of the number of time-averaged points in the convoluted signal. This optimization is described in detail in Section 5.1.

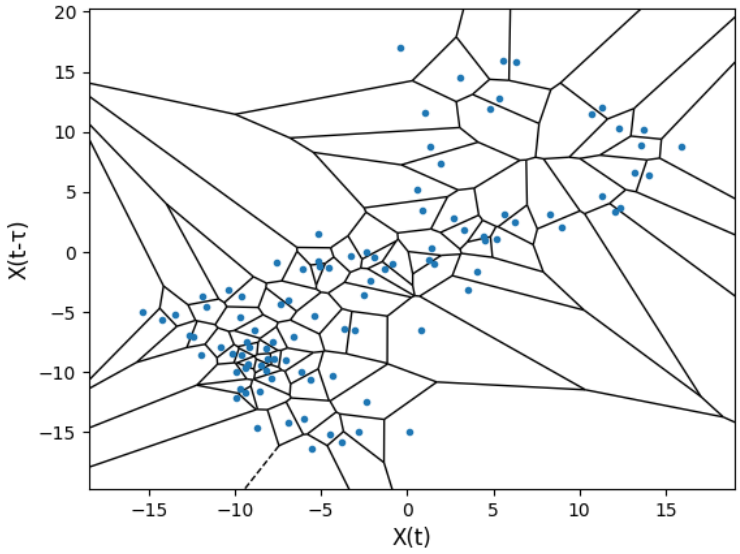


Figure 2.5: A Voronoi diagram shown in phase space. The cell representatives are shown in blue.

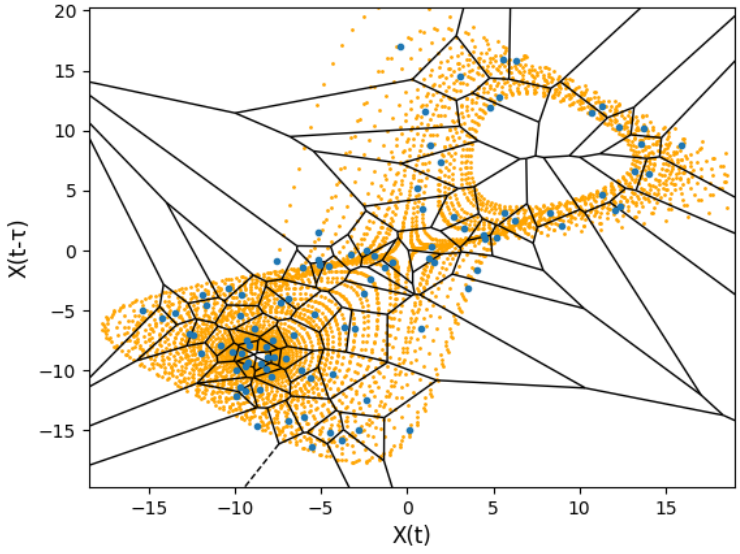


Figure 2.6: A shadow manifold is overlaid on a Voronoi diagram. Cell representatives are shown in blue.

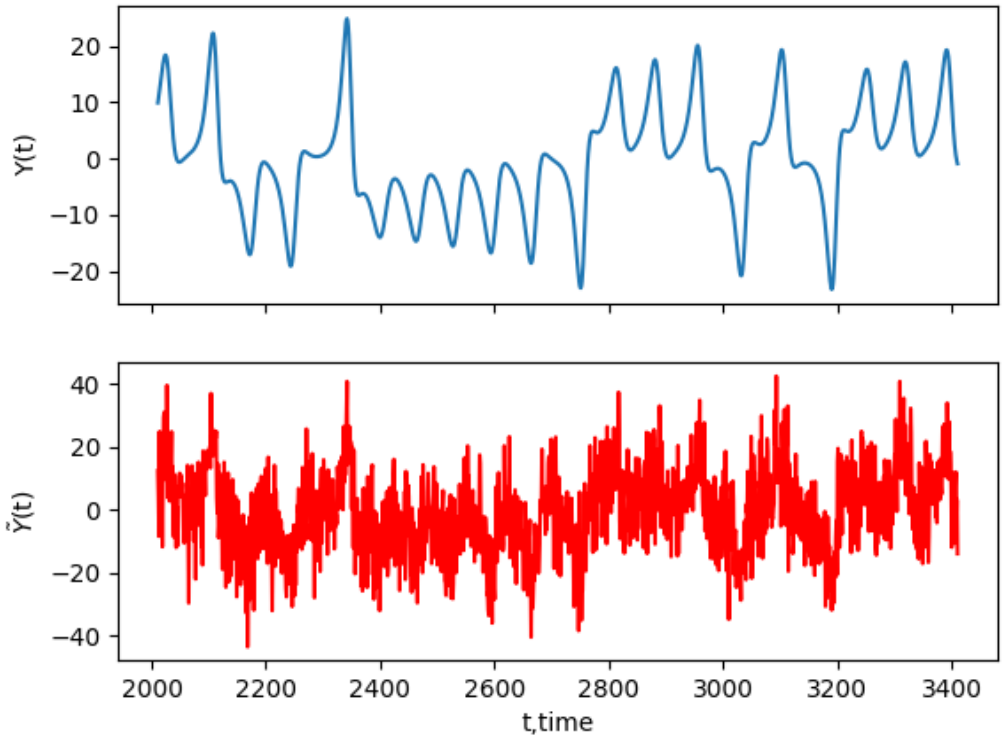


Figure 2.7: (Top)$Y(t)$ signal from the Lorenz system and (Bottom) the same signal after adding Gaussian noise

2.4 Denoising

While investigating various phenomena, experimentalists often collect inherently noisy data. This noise varies based on the type of equipment used and includes unpredictable vibrations. We simulated this noise in using synthetic Gaussian noise.

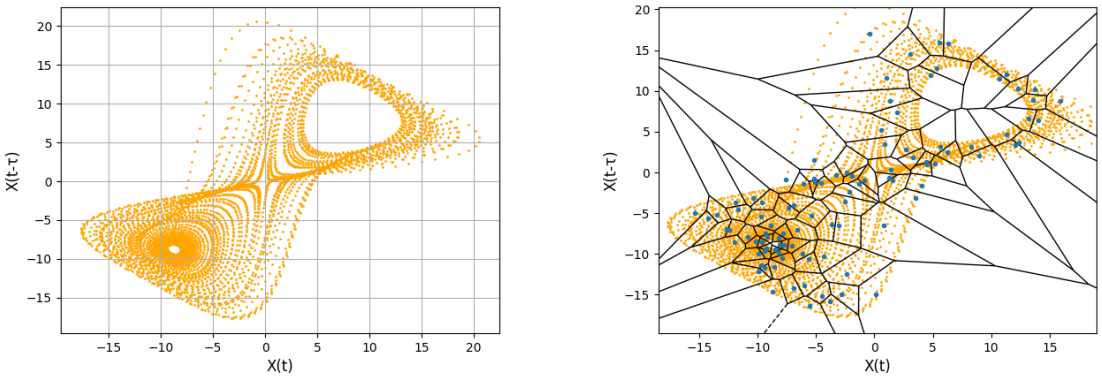
As mentioned above, given a clean signal $X(t)$ and a noisy signal $\tilde{Y}(t)$, $\tilde{Y}(t)$ can be cleaned by applying Theorem 2.1, convergent cross mapping, and a covering that partitions the space into cells. Based on [1] and our thorough literature review, we arrive at the following algorithm:

Algorithm 1 (A1)

0. We are given some state manifold $M(t) = (X(t), Y(t))$. This can be the first two dimensions of the Lorenz attractor system, for example. $\tilde{Y}(t)$ is a noisy signal, whose denoised version is coupled with $X(t)$, the clean reference signal (as shown in Figure 2.7).

1. Construct a shadow manifold, $M_X(t)$, according to Theorem 2.1 and create a mesh, either grid (Figure 2.8a) or Voronoi diagram (Figure 2.8b), on M_X.

2. Map points from $\tilde{Y}$ on the time interval $[t_a, t_b]$ onto M_X.



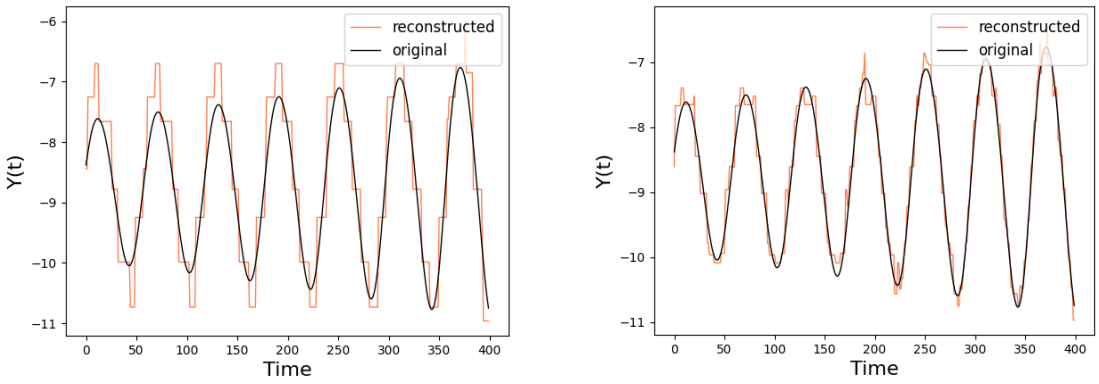
(a) Grid Mesh (b) Voronoi Diagram

Figure 2.8: Partitioning of phase space using (a) Grid mesh and (b) Voronoi diagram

3. Average the $\tilde{Y}$ values within cells of the mesh.

4. Map values on a distinct time interval $[t_c, t_d]$ to M_X.

5. Once the t values have been mapped to M_X, each value will correspond to a cell in the mesh. Each $Y(t_k)$ is assigned the average value in the cell where $M_X(t_k)$ resides. Thus, the clean signal Y has been reconstructed.



(a) (b)

Figure 2.9: Reconstructed signals from the denoising algorithm using (a) Grid mesh (b) Voronoi diagram

We were able to implement A2 in C++ and successfully denoise signals with both the structured (Figure 2.9a) and unstructured (Figure 2.9b) meshes while also improving algorithm speed. As you can see in these images, both methods reconstructed the $Y(t)$ signal from the Lorenz system fairly well, with the unstructured mesh outperforming the grid.

2.5 Error Quantification Metrics

For the remainder of the paper, we will be using two methods of quantifying error. The first is the Pearson Correlation Coefficient, or PCC. PCC is given by

$$PCC = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}}$$

for all $x_i \in X$ and $y_i \in Y$, where X and Y are real-valued signals.

PCC is a measure of linear correlation between two signals. A PCC of 1 means that the signals are perfectly correlated, while a PCC of 0 means that the signals have no relation. We use 1-PCC as an error measurement since an ideal PCC is 1.

The second error measurement is Root Mean Squared Error, or RMSE. RMSE is given by

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{N}}$$

where $\hat{y}$ is the reconstructed signal, y is the target signal, and N is the number of data points being compared in each signal.

RMSE is a measure of deviation from the original signal. We found that PCC is not always an accurate measurement of error, as even perfectly correlated signals may have significantly different amplitudes (particularly for periodic systems). RMSE gives a measure of accuracy that paired with PCC gives a more holistic view of algorithmic performance. To facilitate comparisons of reconstruction error between different systems, we use a normalized version of Root Mean Squared Error. This variant is sometimes referred to as the Coefficient of Variation (called Normalized RMSE henceforth) and is computed as the ratio of the Root Mean Squared Error to the range of the reconstructed signal

$$CV(RMSE) = \frac{RMSE}{\hat{y}_{max} - \hat{y}_{min}}$$

where $\hat{y}_{max}$ and $\hat{y}_{min}$ denote the maximum and minimum values, respectively, of the reconstructed signal.

Chapter 3

Deconvolution

We are interested in reconstructing the underlying signal from a temporally under-resolved time-series. Our algorithm is applicable to any system where the speed of the dynamics, or changes in the system, are faster than the rate at which you can acquire data. This could mean that a camera shutter speed is slower than the speed of dynamics of some system that is being recorded. We mainly simulate convolution by reading in a noisy signal, averaging a time window of length τ_w, discarding the frames used in the average, and repeating until we have the desired number of points in a temporally under-resolved signal, N_V. Explicitly, temporally under-resolved signal, $V(t)$, is obtained from the corrupted signal $\tilde{Y}$ by time-averaging $\tilde{Y}$ within a designated measurement window of length τ_w, according to the following definite integral:

$$V(t) = \frac{1}{\tau_w} \int_{t-\frac{\tau_w}{2}}^{t+\frac{\tau_w}{2}} \tilde{Y}(t)dt.$$

In [1], a deconvolution algorithm was presented. The original algorithm was for a rectilinear grid partition of phase space, but we extend it to the Voronoi diagram partition of phase space. Q denotes the total number of cells in the partition and q denotes the index of the cell. Both the rectilinear grid and the Voronoi diagram can be enumerated in this way.

There are two time domains that we require for our algorithm. One is a training region where we need a temporally resolved reference signal and the convoluted signal from the underlying target signal. This training region will be used to train a model which can then be used to construct a temporally resolved target signal for a disjoint testing time domain where the clean reference signal is still defined, but the convoluted signal is not. Although the convoluted signal is not given on the testing domain, we still call the algorithm deconvolution, because the training model is acquired from the temporally under-resolved signal.

Algorithm 2 (A2)[1]

0. Approximate the integral to convolute points to yield $V(t)$ as the following

[1]The code starting at step 1 for this algorithm is in Chapter 10

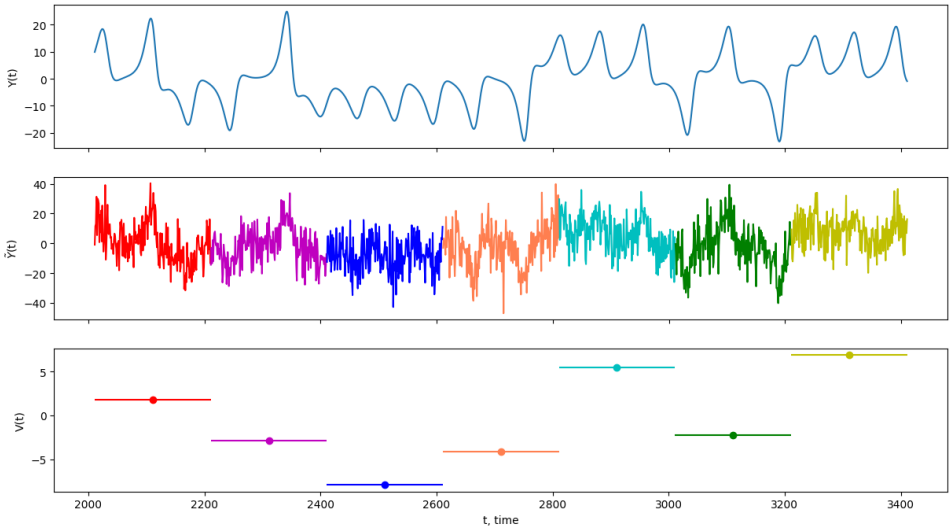


Figure 3.1: (Top) Target signal to be reconstructed, Y (Middle) Target signal with noise added, $\tilde{Y}$ (Bottom) The convoluted target signal, V, time-averaged in a window of 500 timesteps

Riemann sum:

$$V(t_p) \approx \frac{1}{N_w} \sum_{t_k=t_p}^{t_p+N_w} \tilde{Y}(t_k),$$

where N_w is the number of discrete points in the timespan of length τ_w starting at t_p. The t_p values are given by

$$t_p \in \{(E-1)\tau + pN_w : p \in [0, N_V)\}$$

where E is the embedding dimension and τ is the time lag.

1. Construct the shadow manifold for the training region with τ. An offset of $(E-1)\tau$ is added to ensure that the signal is never indexed for negative values when performing the time-delay embedding

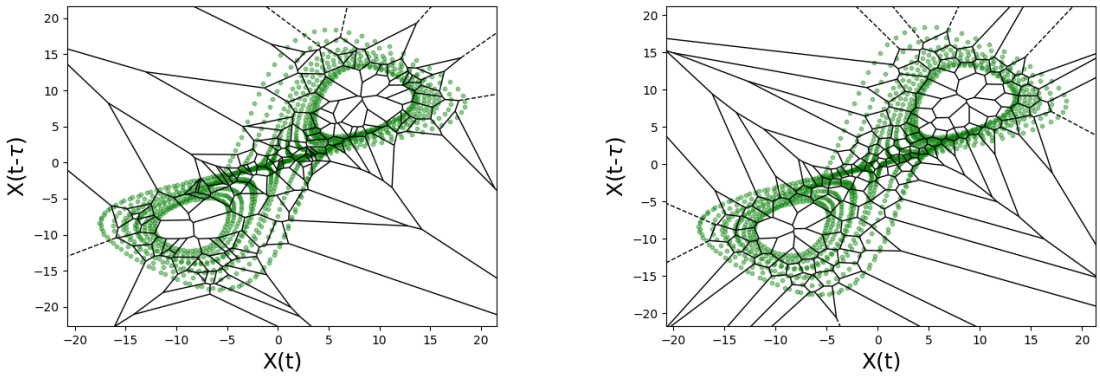
$$M_{X,train} = \{(X(t), X(t-\tau), \ldots, X(t-(E-1)\tau)) : t \in [(E-1)\tau, t_{f,train}]\}$$

where

$$t_{f,train} = (E-1)\tau + (N_V \cdot N_w)$$

2. Select N_C linearly spaced points on $M_{X,train}$ and partition the phase space with a Voronoi diagram with cell representatives located at these N_C points.

3. Find the center of mass of each of these cells and adjust the partition of phase space to make these centers of mass the new Voronoi diagram representatives. This step is k-means clustering; it is optional, and can be repeated as many times as necessary (we call the iteration number k_{it}). Figure 3.2 shows the effects of using k-means clustering to readjust the cell representatives for a more uniform partition of phase space.



(a) (b)

Figure 3.2: The Voronoi partition of phase space with (a) no k-means clustering, where the cells are often narrow and/or very small, and with (b) 5 iterations of k-means clustering, where the cells are noticeably more uniform in shape and size

4. For each $V(t_p)$, estimate how much time the points in $\{M_{X,train}(t_k)\}$ for $t_k \in (t_p, t_p + N_w)$ spend in each Voronoi cell. Let Q be the total number of Voronoi cells and let q be an index for the Voronoi cells in a 1D list; then for each $V(t_p)$, we construct a set, $\{c_{pq}\}$, where

$$c_{pq} = \frac{N_t}{N_w}$$

and N_t is the number of points on M_X in Voronoi cell q for the time window $(t_p, t_p + N_w)$.

5. Recall that

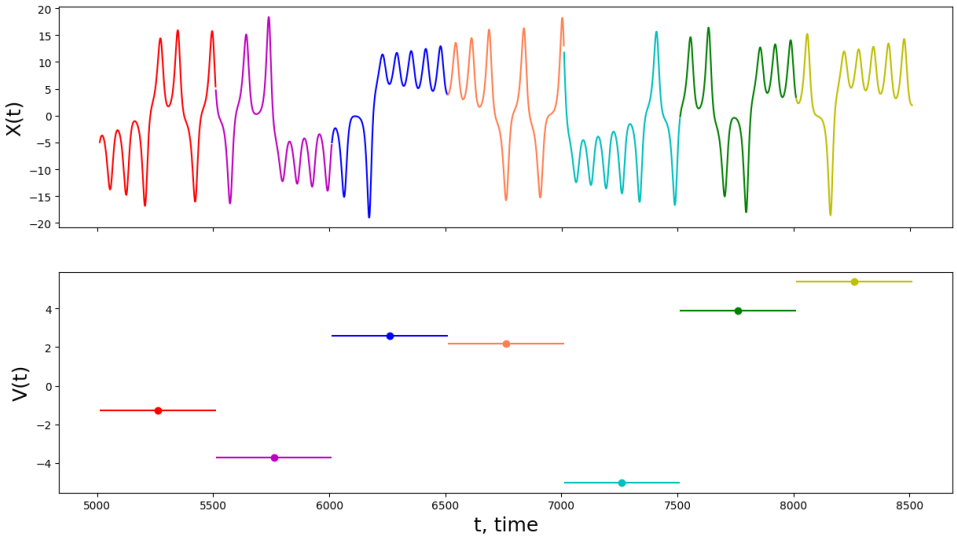
$$V(t_p) \approx \frac{1}{N_w} \sum_{t_k=t_p}^{t_p+N_w} \tilde{Y}(t_k),$$

and because Takens' Theorem guarantees the existence of diffeomorphisms such that the segments of $\{M_{X,train}(t_k)\}$ for $t_k \in (t_p, t_p + N_w)$ for $V(t_p)$ are isomorphic to the corresponding region of the shadow manifold found by time-delay embedding the target signal $Y(t)$, we know that

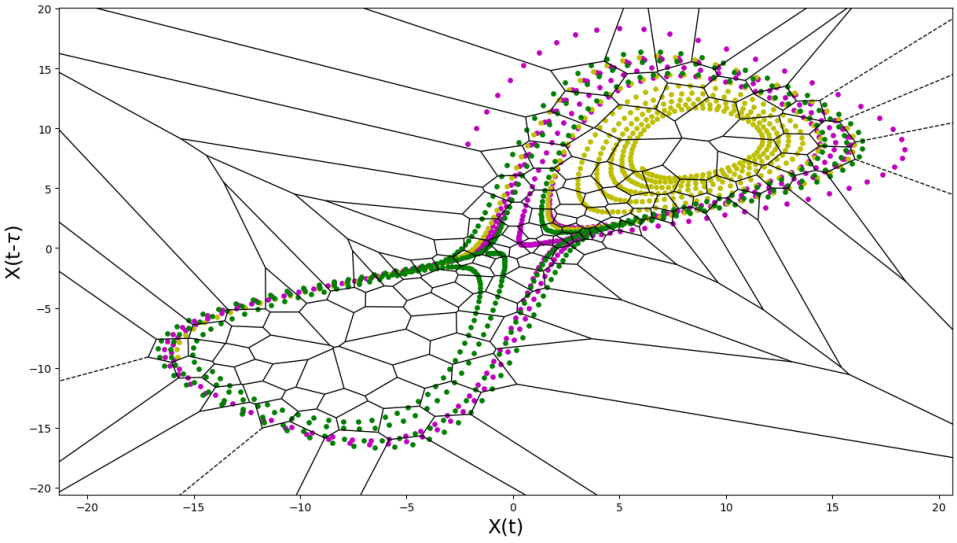
$$V(t_p) \approx \frac{1}{N_w} \sum_{t_k=t_p}^{t_p+N_w} g_Y(g_X^{-1}(M_x(t_k))).$$

Furthermore, the fraction of the curve on the shadow manifold in each cell will be preserved by the diffeomorphisms, and we can approximate $V(t_p)$ by summing over each cell of phase space. Therefore,

$$V(t_p) \approx \sum_{q=0}^{Q-1} c_{pq} Y_q, \tag{3.1}$$



(a)



(b)

Figure 3.3: (a) (Top) The clean reference signal, X, color coordinated with the time window in which $\tilde{Y}$ was averaged to construct V (Bottom) The temporally under-resolved signal, V (b) The X-shadow manifold where every other time window from (a) is shown so that the points on the shadow manifold in Voronoi cells are visible

where Y_q is the average value of Y the Voronoi cell q. Then we can construct an N_V by Q matrix, A, where the (p, q) entry is c_{pq}; then the system $A\vec{x} = \vec{b}$ represents the relation given by Equation 3.1 and can be solved for Y_q for each Voronoi cell. Explicitly, solving for Y_q in each cell involves solving the linear system $A\vec{x} = \vec{b}$ where $\vec{x} = [Y_1, Y_2, \ldots, Y_Q]^T$ and $\vec{b} = [V_1, V_2, \ldots, V_{N_V}]^T$. Y_q is the value that is assigned to the q^{th} cell after solving the system.

6. Once we have solved the linear system, we have found the value of Y_q for each cell, which can then be mapped back to state space to reconstruct the target signal for a disjoint time region after the training region. This is done by constructing another shadow manifold for a disjoint domain of $X(t)$ to use as the testing manifold. Each point in the testing domain, t_k, is mapped to the training shadow manifold, $M_{X,test}(t_k)$, and then $Y(t_k)$ is assigned the value of the Voronoi cell in which $M_{X,test}(t_k)$ resides.

7. Instead of just assigning $Y(t_k)$ the value at the center of the Voronoi cell, it is better to use a weighted sum of the k_{NN} nearest Voronoi cell representatives using the following interpolation scheme, adapted from [7]:

$$Y(t_k) = \sum_{i=0}^{k_{NN}-1} w_i Y_{q,i}$$

$$w_i = \frac{u_i}{\sum u_i}$$

$$u_i = \exp\left(- \frac{||M_X(t) - M_X(t_{q,i})||}{||M_X(t) - M_X(t_{q,0})||} \right)$$

where $Y_{q,i}$ is the i-th nearest Voronoi cell value, $t_{q,i}$ is the time index of that point in phase space with $t_{q,0}$ representing the nearest Voronoi cell, and $||.||$ represents the Euclidean distance between points on the shadow manifold.

3.1 Deconvolution of Lorenz System Data

The deconvolution algorithm in [1] required 500000 sample points from $V(t)$ to reconstruct the underlying signal from a temporally under-resolved signal for the Lorenz system by using a grid partition of phase space. Partitioning into a grid left many grid cells with none or very few data points, but using Voronoi diagrams decreases the number of required sample points to get a comparable reconstruction because there are not any empty cells. The noisy signal is constructed by introducing noise with mean zero and variance 100 to Y. The time window included 500 timesteps, and Figure 3.1 shows that this time window captures several local extrema. In Figure 3.4, the results of the deconvolution algorithm applied to the Lorenz system are visualized.

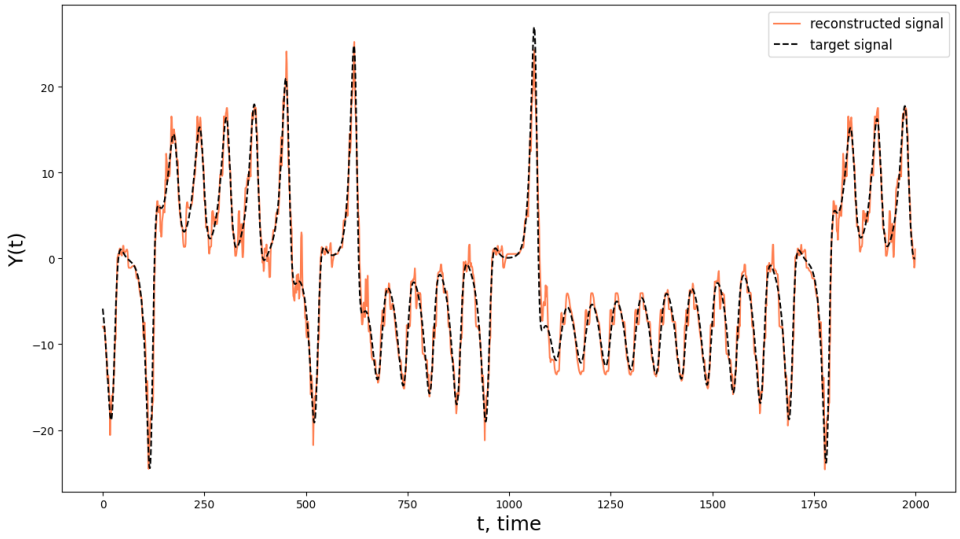


Figure 3.4: A reconstruction of $Y(t)$ from the Lorenz system using $N_V = 100000$, $N_w = 500$ timesteps, $E = 2$, $\tau = 13$, and $k_{it} = 5$. The PCC is 0.9866 and the Normalized RMSE is 0.02916.

3.2 Deconvolution of Sine Data

Araki and collaborators also deconvoluted the Sine system, given in Section 8.1.2, using their original algorithm in [1]. The period of the system is about 600 timesteps, so we selected a time window of 1500 timesteps to capture several periods. Figure 3.5 shows a subset of the two input signals for the deconvolution algorithm, $X(t)$ and $V(t)$, for the Sine system, and Figure 3.6 shows the results of the reconstruction.

3.3 Deconvolution of Hall-effect thruster Data

We also deconvoluted Hall-effect thruster system data, where the reference signal, $X(t)$, is the anode current and the $Y(t)$ target signal that we deconvolute is the cathode current. These two signals are strongly coupled in the HET and are responsible for inducing the magnetic field that ionizes the noble gas, which in turn propels the thruster. The signal readings are showing current values in Amperes. The input signals $X(t)$ and $V(t)$ are shown in Figure 3.7, and the time window is 600 timesteps so that several extrema are included. The results of the reconstruction are shown in Figure 3.8. Notice that the Normalized RMSE associated with this HET reconstruction is around five times larger than the Normalized RMSE for the Lorenz and Sine reconstructions (which have relatively similar Normalized RMSEs).

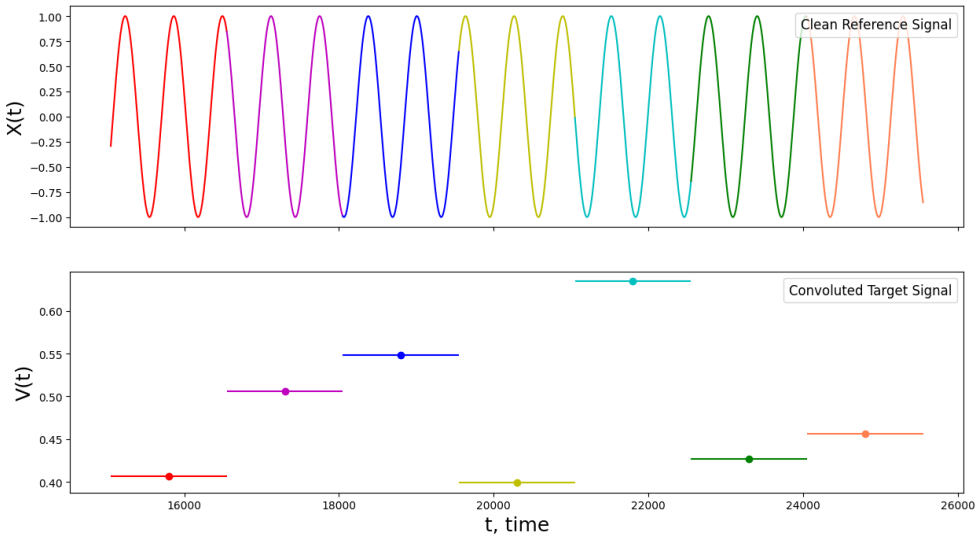


Figure 3.5: (Top) The clean reference signal, $\sin(t)$ (Bottom) The convoluted signal in a time window of 1500 timesteps

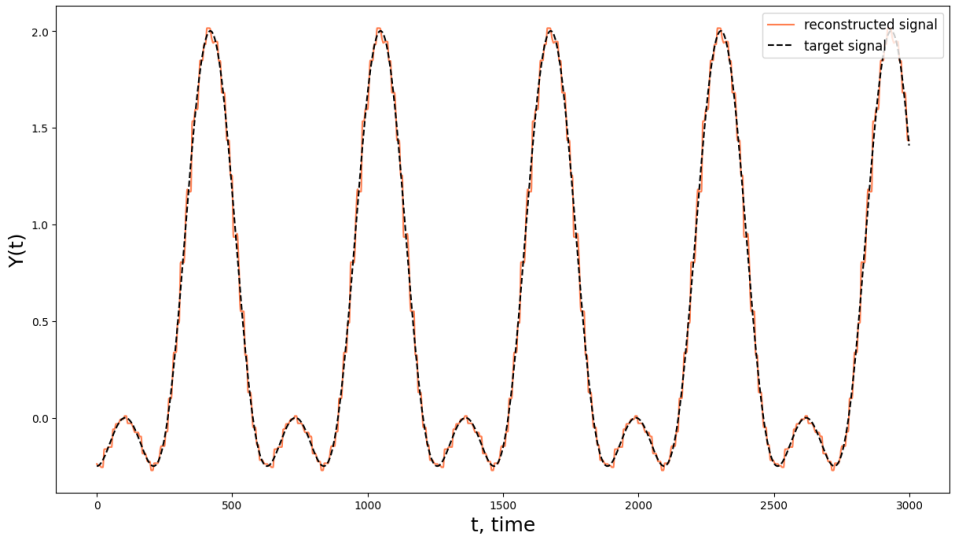


Figure 3.6: A reconstruction of $Y(t)$ from the Sine system using $N_V = 50000$, $N_w = 1500$ timesteps, $E = 2$, $\tau = 10$, and $k_{it} = 3$. The PCC is 0.9969 and the Normalized RMSE is 0.0266

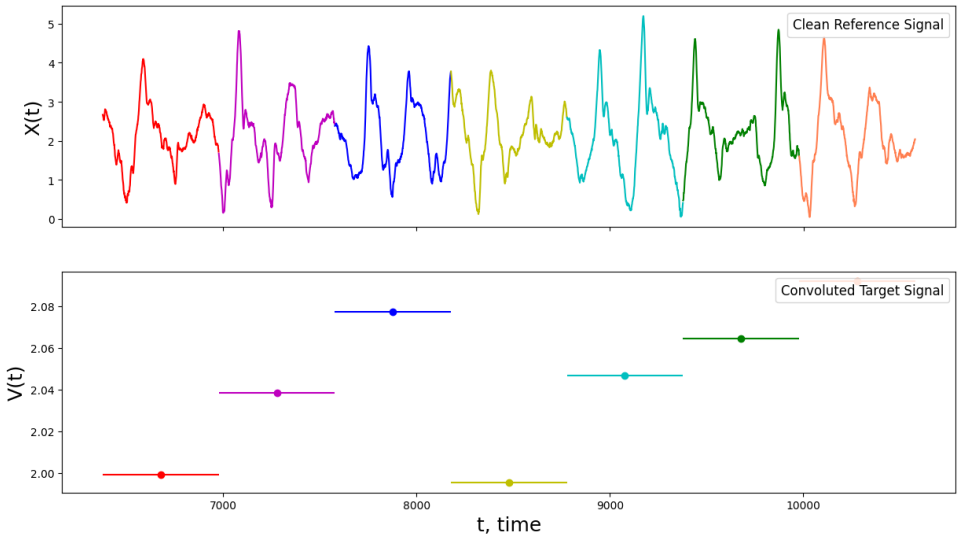


Figure 3.7: (Top) The clean reference signal, which is the HET anode current reading (Bottom) The convoluted signal found from the HET cathode current over a time window of 600 timesteps

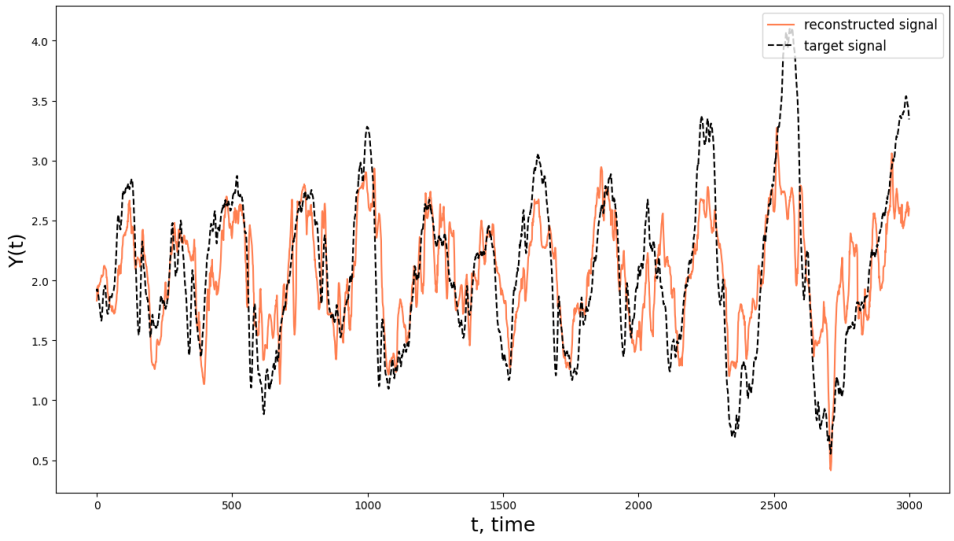


Figure 3.8: A reconstruction of the cathode signal in the HET system using $N_V = 100000$, $N_w = 600$ timesteps, $E = 8$, $\tau = 64$, and $k_{it} = 3$. The PCC is 0.7677 and the Normalized RMSE is 0.1523.

Programming Language	(k_{it}, k_{NN})	N_V	PCC	Wall-clock Time
Octave	Smoothing from [1]	500000	0.881	28m 12.7s
C++	(0, 0)	4500	0.8983	0.9s
C++	(0, 7)	10000	0.965	6.5s
C++	(0, 7)	100000	0.986	1m 2.6s
C++	(0, 7)	300000	0.990	3m 28.6s

Table 3.1: Timing data comparing PCC and Wall-clock Time on an Intel Xeon Gold 6240 machine @ 2.60 GHz; we decided not to use k-means clustering in the interest of time; the time window

3.4 Runtime Data

Below in Table 3.1, we list information regarding the relationship between PCC and Wall-clock Time for our algorithm. We also compare our version of deconvolution to the original implementation of the algorithm from [1], which was written in Octave. This Octave algorithm did not separate into training and testing regions, so we made the testing region the same size as the training region to make the runtime comparisons more fair.

3.5 Grid-Based Partition

We also implemented the grid-based partition of phase space to deconvolute the Lorenz system. Using a cell width of 3 timesteps, we reconstructed the temporally under-resolved signal shown in Figure 3.4. The grid-based reconstruction has a lower PCC and a higher normalized RMSE, so there is work to be done for it to be a suitable replacement to the Voronoi diagram method. This grid version of the deconvolution algorithm ran slightly faster (with the same input parameters) than the Voronoi diagram version (18 seconds compared to 28 seconds). This discrepancy in speed may be explained by the fact that the grid method does not require a nearest neighbor search. So, if we further optimized this algorithm, it is reasonable to assume that it would run extremely fast and could potentially be as accurate as the Voronoi diagram based method. The reconstruction can be found in Figure 3.9.

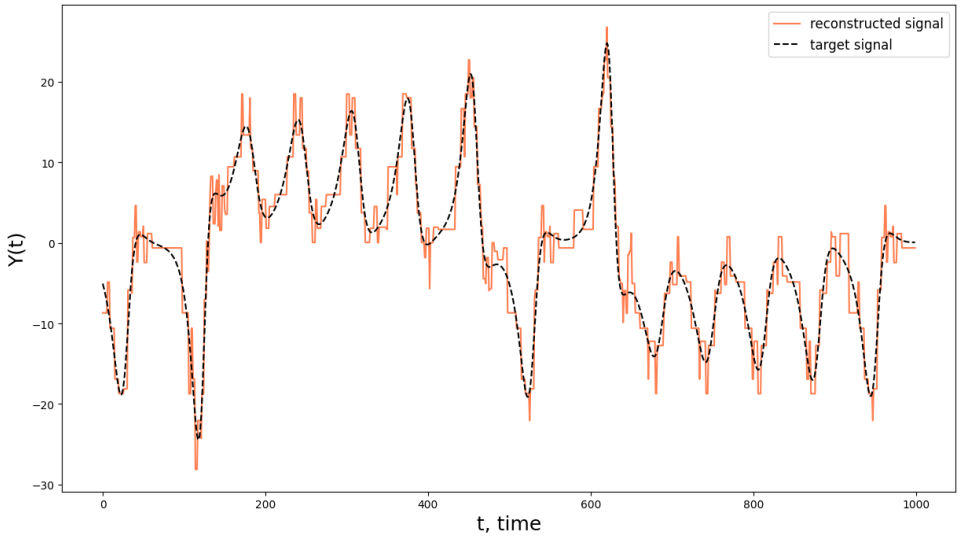


Figure 3.9: Reconstruction of $Y(t)$ from the Lorenz system using a grid mesh with $N_V = 100000$, $N_w = 500$ timesteps, $E = 2$, and $\tau = 13$. The PCC is 0.971 and the Normalized RMSE is 0.041.

Chapter 4

Parameter Optimization

In this section, we discuss how to construct the proposed parameter optimization algorithm and evaluate its performance according to PCC and Normalized RMSE in comparison to heuristic-based (baseline) methods. For a given number of samples from the temporally under-resolved signal V (obtained by time-averaging the corrupted Y signal in the training region over a time window of length τ_w), we are interested in optimizing five input parameters for the deconvolution algorithm, namely the time lag (τ), the embedding dimension (E), the number of Voronoi cells used in the unstructured mesh (N_C), the number of iterations of the k-means clustering algorithm applied to the training manifold (k_{it}), and the number of near-neighboring cell representatives used in interpolation applied to the testing manifold (k_{NN}). We show that inputting the optimal parameters into the deconvolution algorithm leads to accuracy improvements in the reconstructed target signal.

4.1 Time Lag Optimization: Fraser's Criterion

We begin by presenting a baseline method to obtain the time lag parameter, τ. It is helpful to consider the upper and lower thresholds on τ.[1] If τ is too short, then $X(t)$ will be approximately equal to $X(t-\tau)$, and the resulting shadow manifold will resemble a diagonal line (Figure 4.1a). This is called redundancy because little to no additional information is gained from the time-delay embedding [8]. On the other hand, if τ is too long, $X(t)$ and $X(t-\tau)$ will appear to be only stochastically related (Figure 4.1b). This is called irrelevance [8]. Note that the shadow manifolds in Figures 4.1a and 4.1b do not resemble the butterfly manifold described earlier for the Lorenz system (see Figure 2.2). Assuming that two dimensions are sufficient for the embedding, we would expect the two manifolds to look relatively similar since there exists a diffeomorphism between M_X and M by Theorem 2.1. It is worth mentioning that, although the shadow manifolds included in this chapter are associated with the Lorenz system, the phenomena—redundancy and irrelevance—are troublesome in other attractor systems (see Section 8.1). Based on this dissimilarity alone, it

[1] The following discussion is a simplification of the problem in which only two lagged time series are considered. Redundancy and irrelevance continue to pose issues in higher dimensional time-delay embeddings.

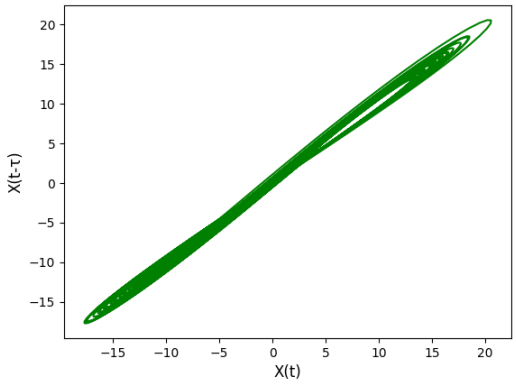
should be apparent that the optimal time lag parameter lies in between these two extremes; in other words, τ must be chosen so that both redundancy and irrelevance are avoided. The objective then is to maximize the amount of information garnered from the time-delay embedding, i.e. to minimize the dependence of $X(t)$ and $X(t-\tau)$ while preserving the non-stochastic relationship between the two lagged time series.

A naïve way to accomplish this goal would be to find the first zero of the linear autocorrelation function for the two lagged time series. However, this autocorrelation function is only a measure of linear dependence. A quantity which describes a more generalized dependence between $X(t)$ and $X(t-\tau)$ is preferred since it better explains the relationship between the lagged time series. In [6], a heuristic called Fraser's "first minimum of mutual information" criterion was initially presented. Fraser and Swinney leveraged mutual information (MI) to propose an alternative to the "first zero of the linear autocorrelation function" criterion [6]. Broadly speaking, the MI of two random variables describes the amount of information obtained about one random variable by observing the other. In this case, the mutual information of the lagged time series, $X(t)$ and $X(t-\tau)$, is of interest as it measures both linear and nonlinear dependencies. According to Fraser's criterion, the time lag corresponding to the first local minimum of MI is a good guess for the optimal τ. To evaluate the performance of this optimization technique, we can examine the shadow manifold generated with the lag identified by Fraser's criterion. From Figure 4.2a, it should be clear that the first local minimum of MI corresponds to a time lag between 0 and 50, namely 16. Now, looking at the shadow manifold constructed using this lag parameter in Figure 4.1c, there is, at minimum, visual confirmation that $\tau = 16$ is a reasonable choice.[2] Intuitively, this is because the shadow manifold resembles the state manifold for the Lorenz system. Again, we expect this behavior due to the diffeomorphism between the state and shadow manifolds (guaranteed by Theorem 2.1).

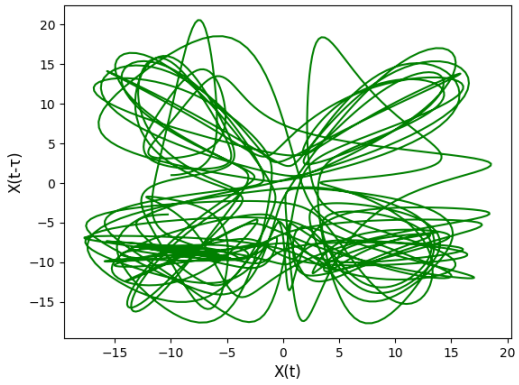
We similarly optimized the time lag for the HET anode current[3] data using Fraser's criterion and determined that $\tau = 54$ corresponds to the first local minimum of MI, as shown in Figure 4.2b. However, time lag optimization according to Fraser's criterion in the case of the Sine system (see Section 8.1.2) proved more challenging. Unlike in Figures 4.2a and 4.2b, the mutual information shared among the lagged time series of the Sine system fluctuates rapidly as a function of the time lag (visualized in Figure 4.2c). This hinders our ability to select the first 'true' local minimum of MI. The first time lag value that corresponds to the typical definition of a local minimum—the preceding and subsequent points have higher MI—may simply be the product of these small fluctuations. In the case of the Sine system, $\tau = 8$ corresponds to the first local minimum of MI. But it is not at all clear why this choice of lag should be better than, for instance, the time lag associated with the second local minimum of MI. This phenomenon highlights a significant deficiency in Fraser's criterion, specifically when analyzing perfectly periodic systems, and motivates the development of new

[2]For further discussion regarding baseline methods for time lag optimization, see [8]. In [8], a different heuristic—the time lag corresponding to the global maximum of MI in the orthogonal embedding, which is constructed using discrete Legendre coordinates—is presented.

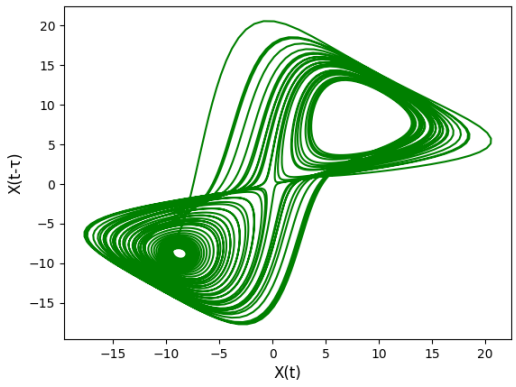
[3]As the time lag parameter only relates to lagged versions of $X(t)$, it is sufficient to only know the origin of the X data. In this example, we specify that the X signal is the discharge current from the anode component of the Hall-effect thruster.



(a)



(b)



(c)

Figure 4.1: Various shadow manifolds for the Lorenz attractor system: (a) Redundant: $\tau = 1$ (b) Irrelevant: $\tau = 1000$ (c) Optimal—Fraser: $\tau = 16$

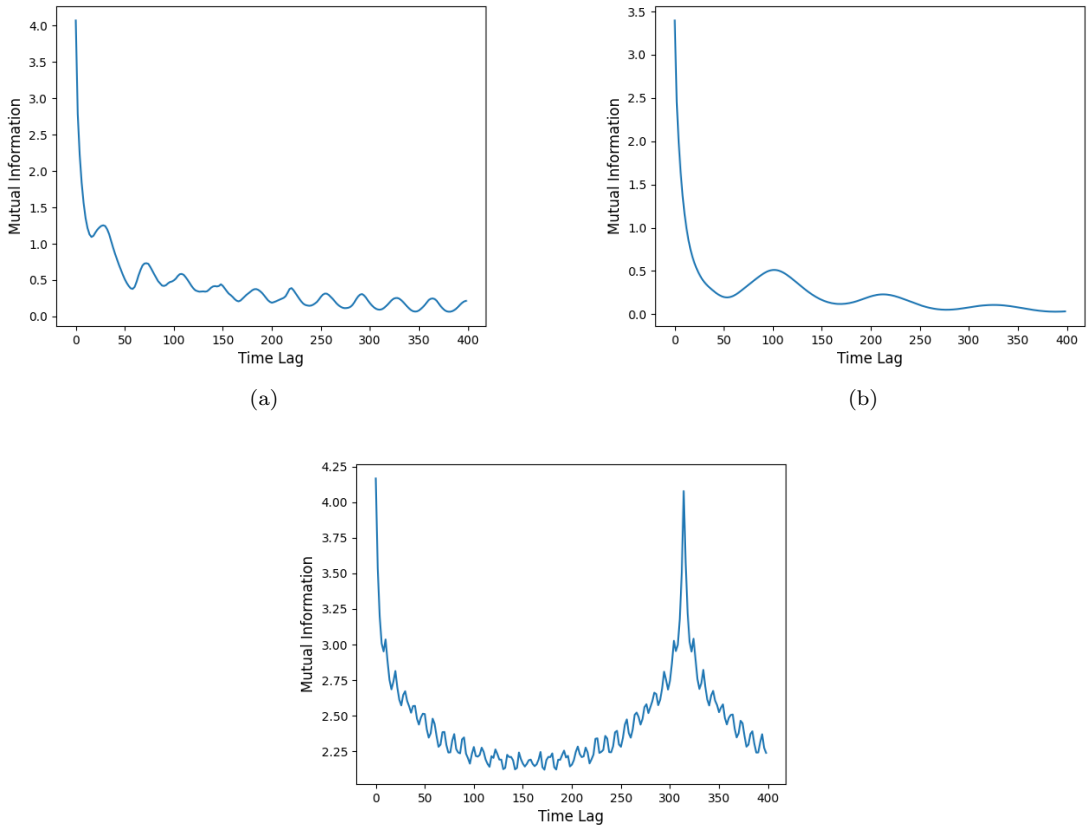


Figure 4.2: Mutual information plotted as a function of τ for (a) Lorenz (b) HET anode current (c) Sine

parameter optimization approaches. The optimal lag values, according to Fraser's original criterion, for each system of interest are listed in Table 4.1. While exploring this flaw in Fraser's original heuristic, we developed an alternate criterion that attempts to control for the fluctuations seen in periodic systems. According to this novel criterion, we locate the first point that satisfies the usual requirements for a local minimum and also marks the beginning of a region of increasing mutual information.[4] By extending the typical definition of the first local minimum, we were able to minimize the impact of fluctuations in time lag optimization for the Sine system. Using this variant of Fraser's criterion, we identified $\tau = 64$ as the best lag for the Sine system. We found reconstruction accuracy (measured in terms of PCC and Normalized RMSE) was higher when we used this lag parameter instead of $\tau = 8$.

[4]We specifically searched for the first τ whose preceding neighbor and three subsequent neighbors had higher MI.

4.2 Embedding Dimension Optimization: Cao's Method

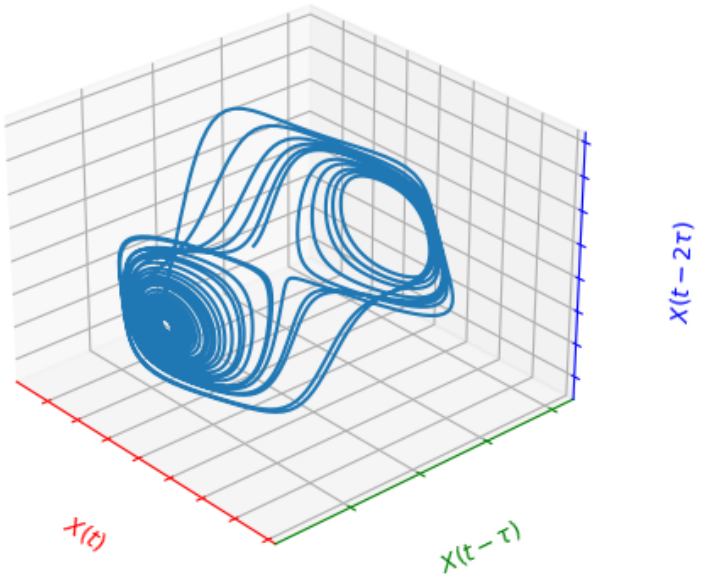


Figure 4.3: Example of a 3D Shadow Manifold

Recall that the embedding dimension, E, equals the number of lagged time series that are used to construct the shadow manifold. In Figure 4.3, we see a shadow manifold, which is constructed with three lagged time series—$X(t)$, $X(t-\tau)$, and $X(t-2\tau)$. In this case, the embedding dimension is three. It is worth mentioning that, if an insufficient embedding dimension is used, then the diffeomorphism between the state manifold and M_X is no longer guaranteed to exist. For instance, if an embedding dimension smaller than three is used for the Lorenz system (whose Hausdorff dimension is slightly greater than two), the diffeomorphism between M and M_X will no longer exist for all[5] regions of the manifolds.

In this section, we present a baseline method to obtain the minimum embedding dimension. This is critical because the lower the embedding dimension the faster deconvolution can occur. Like the 2020 RIPS-AFRL research group, we initially used Cao's method to obtain the minimum embedding dimension [4]. Recall that Theorem 2.1 states that $E \geq 2d+1$ is sufficient, but not necessary, to guarantee a diffeomorphism between the E-dimensional shadow manifold and the state manifold. This implies that there are systems for which this diffeomorphism is preserved even if $E < 2d+1$. Cao's method provides a procedure to select the minimum embedding dimension (for which the diffeomorphism holds) in practice [3, 12]. In principle, this method should be independent of the time lag τ; however, in practice, Cao's method depends heavily on the choice of lag. For instance, when running Cao's method with the time lag obtained via Fraser's criterion, the outputted minimum embedding

[5]Since the crossing of the two-dimensional Lorenz shadow manifold affects a relatively small number of points, reconstructions of the target signal can still be very good. So, many researchers choose to use $E = 2$ in the interest of computational efficiency.

dimension required for phase space reconstruction may be lower for certain systems.

Algorithm 3 (A3)

0. We are given a clean reference signal $X(t)$, which is essentially a time series $x(1), x(2), \ldots, x(N)$, along with an optimal time lag (from Section 4.1) and a range of potential embedding dimensions to consider, $d \in \{1, \ldots, d_{max}\}$.

1. For $d \in \{1, 2, \ldots, d_{max}\}$, reconstruct the i-th time-delayed vector

$$r_i(d) = (x(i), x(i+\tau), \ldots, x(i+(d-1)\tau)),$$

where τ represents the time lag and $i \in \{1, 2, \ldots, N-(d-1)\tau\}$.

2. For $d \in \{1, 2, \ldots, d_{max}\}$, construct a k-dimensional tree with $N-(d-1)\tau$ points, where N represents the number of points in the X signal.

3. Locate the nearest neighbor (excluding self-matches)[6] of $r_i(d)$ in d-dimensional reconstructed phase space, $r_{n(i,d)}(d)$, using the maximum norm[7] as the distance metric.

4. Compute

$$E(d) = \frac{1}{N-d\tau} \sum_{i=1}^{N-d\tau} \frac{\left\| r_i(d+1) - r_{n(i,d)}(d+1) \right\|}{\left\| r_i(d) - r_{n(i,d)}(d) \right\|}$$

for $i \in \{1, 2, \ldots, N-d\tau\}$. As before, $\|.\|$ is the maximum norm. The idea is, if d is a sufficiently high embedding dimension, then near neighbors in the d-dimensional embedding will also be near neighbors in the state manifold. We are looking for the smallest d at which we stop seeing behavior like the crossing of the two-dimensional Lorenz shadow manifold (Figure 4.1c), where points from two different parts of the state manifold are mapped to the same patch of the lower-dimensional shadow manifold. If we were to project the manifold to three dimensions, some of the nearest neighbors at the crossing in the two-dimensional shadow manifold would become separated (as they should be) and no longer be nearest neighbors.

5. Compute

$$E_1(d) = \frac{E(d+1)}{E(d)}.$$

6. Determine for which embedding dimension, d_0, the quantity $E_1(d)$ stops changing significantly. Then, d_0 is the minimum embedding dimension. Intuitively, when constructing a time-delay embedding of dimension d_0, the likelihood of locating false nearest neighbors of points in X (see Step 4 of A3 for an explanation of false nearest neighbors) in Step 3 will be minimized.

[6] We elected to use an approximate nearest neighbor method.

[7] See [3] for an explicit definition of the maximum norm in this context.

If the time series data is obtained from a random set of numbers, an additional step of the algorithm in [3] is relevant. In this step, a quantity denoted $E_2(d)$, that is useful in distinguishing deterministic signals from stochastic signals, is computed. Explicitly,

$$E^*(d) = \frac{1}{N - d\tau} \sum_{i=1}^{N-d\tau} \mid x(i + d\tau) - x(n(i, d) + d\tau) \mid$$

$$E_2(d) = \frac{E^*(d+1)}{E^*(d)}$$

where $x(n(i, d))$ refers to the nearest neighbor of $x(i)$ in d-dimensional reconstructed phase space.

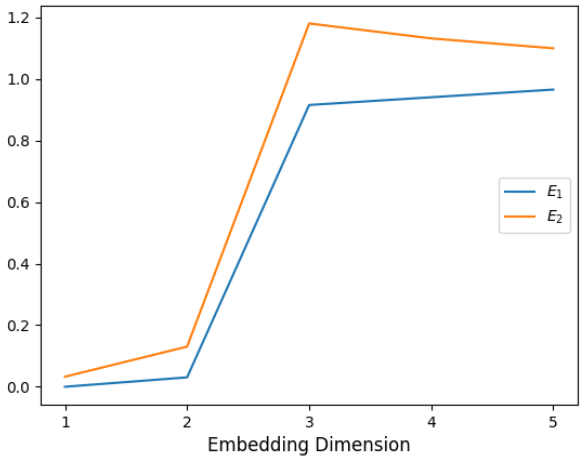
For our purposes, this quantity can be used to confirm that the trend in E_1 (displayed in blue in Figure 4.4) is not simply stochastic behavior. In other words, we compute E_2 to verify that our conclusions based on E_1 are indeed valid, i.e. a significance test. In Figures 4.4a, 4.4b, and 4.4c, we see examples of A3 in action. Notice that, in each figure, E_2 (displayed in orange) is not constantly equal to 1; this confirms our previous assertion that the data we are considering is indeed deterministic and our results are significant. Looking now at the trend in E_1 in the case of the Lorenz attractor (Figure 4.4a), we see that E_1 levels off beginning at $d = 3$. This suggests that the minimum embedding dimension equals three according to Cao's method.

However, A3 does not explicitly define the notion of 'significant change'. While $d_0 = 3$ may seem plausible based on Figure 4.4a since the change in E_1 between $d = 2$ and $d = 3$ is large relative to future changes in E_1 over sequential dimensions[8], this heuristic is too vague to hope for consistent standards of significance among various systems. Applying this baseline method to the HET anode current data highlights this issue. While the changes in E_1 until $d = 8$ are enormous relative to the future changes in E_1 over sequential dimensions[9], E_1 continues to experience non-negligible increases until $d = 9$. Based on our specified level of precision[10], Cao's method identifies $d = 9$ as the minimum embedding dimension. Yet, with a slightly lower standard of precision, $d = 8$ is returned by the algorithm (and $d = 10$ is returned with a higher standard of precision). This instability in the baseline method motivates the development of new parameter optimization approaches. The optimal embedding dimension values, according to Cao's method, for each system of interest are listed in Table 4.1.

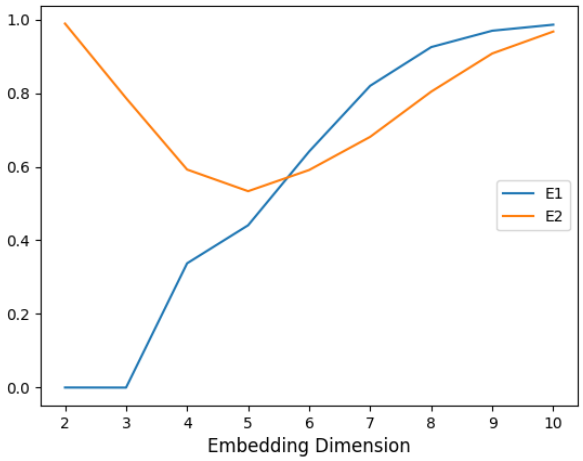
[8]For the Lorenz data, we only considered embedding dimensions less than or equal to five since Theorem 2.1 guarantees that an embedding dimension of $2d_H + 1$ is sufficient. Here, the Hausdorff dimension, d_H is slightly greater than two and, since the embedding dimension must be an integer, we choose to truncate the result of Theorem 2.1 to 5.

[9]For the HET anode current data, we only considered embedding dimensions less than or equal to ten since we assume that the Hall-effect thruster can be modelled as a low-dimensional system. We chose to denote dimensions less than or equal to 10 as 'low'.

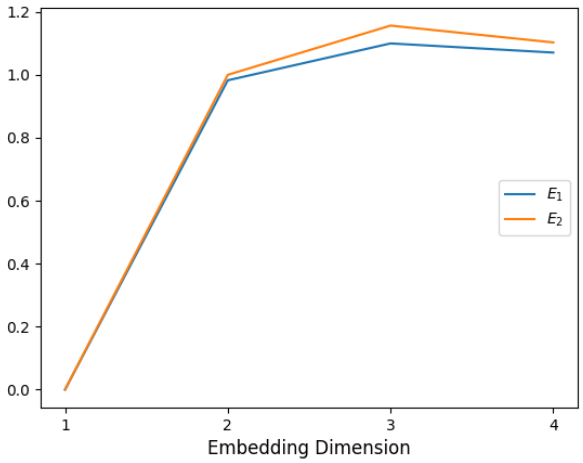
[10]The levels of precision that we used varied based on the system due to scaling discrepancies in E_1. For the Lorenz system, we determined the pair of sequential dimensions for which the absolute difference in E_1 was at least 15 times as large as the absolute difference in E_1 over the neighboring sequential pair. For the HET anode current and Sine system, the only difference was the factor by which the absolute differences in E_1 differed. For the HET anode current, the factor was 2; for the Sine system, the factor was 5.



(a)



(b)



(c)

Figure 4.4: Applying Cao's method to (a) Lorenz (b) HET anode current (c) Sine

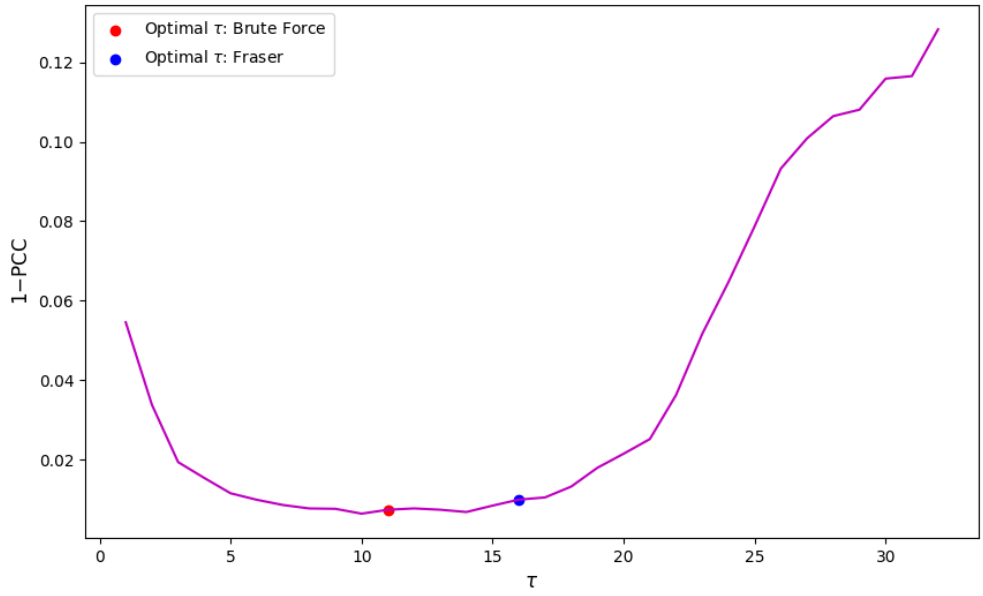


Figure 4.5: Reconstruction error plotted as a function of time lag for the Lorenz system. Red: τ per the proposed method; Blue: τ per Fraser's criterion

Attractor System	τ_{base}	E_{base}
Lorenz	16	3
HET anode	54	9
Sine	8	2

Table 4.1: The optimal parameters selected according to Fraser's criterion and Cao's method

4.3 Single Parameter Optimization

The aforementioned issues with the baseline methods motivates the development of a new technique which returns the optimal value of a given parameter (with all other input parameters fixed[11]) based on reconstruction error. Furthermore, unlike in the case of τ and E, baseline methods for k-means clustering (k_{it}) and interpolation (k_{NN}) optimization are difficult to locate. For this reason, single parameter optimization is used as the heuristic-based method for optimization of k_{it} and k_{NN}. In this section, we present the visual results of single parameter optimization, particularly for the Lorenz attractor system.

Figure 4.5 displays the result of single parameter optimization in the case of time lag.[12] Since the objective of this technique is to test the effectiveness of the baseline method, it makes sense to center the interval of interest at the τ_{base} given by Fraser's

[11] In each of the following tests, $N_V = 100000$, $N_w = 200$ timesteps, and $N_C = 175$.

[12] For this search, the remaining input parameters were fixed as the following: $E = 2$, $k_{it} = 6$, and $k_{NN} = 6$.

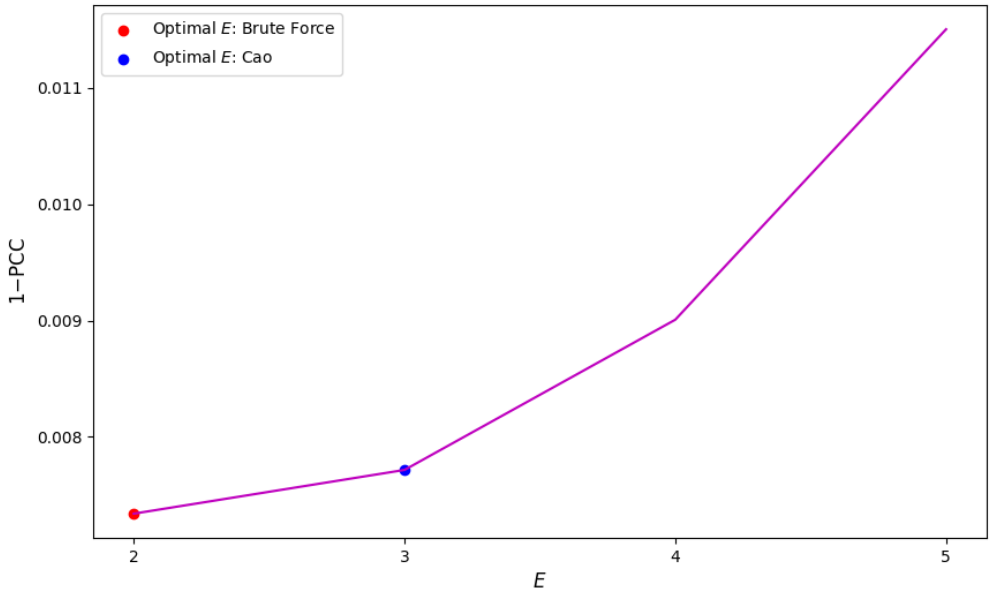


Figure 4.6: Reconstruction error plotted as a function of embedding dimension for the Lorenz system. Red: E per the proposed method; Blue: E per Cao's method

criterion and then to check the neighboring lag values. Notice that $\tau = 10$ appears to maximize reconstruction accuracy. Although Fraser's criterion returns a sub-optimal time lag, the difference in reconstruction error as a result of varying the lag parameter from 10 to 16 is not tremendously significant (slightly greater than 10^{-3}). In fact, this window between $\tau = 10$ and $\tau = 16$ seems to capture the best lag values.

Figure 4.6 displays the results of this approach in the case of embedding dimension.[13] In this case, determining the interval of interest is trivial since we are given an effective upper bound on embedding dimension by Theorem 2.1 (see Footnote 8 of Chapter 4). Notice that an embedding dimension of two minimizes reconstruction error, which is quantified by 1-PCC. Thus, the minimum embedding dimension returned by Cao's method ($E = 3$) is, in fact, sub-optimal based on reconstruction accuracy. The difference in 1-PCC between $E = 2$ and $E = 3$ is roughly 10^{-3}, which may be effectively irrelevant depending on the application.

Figure 4.7 displays the results of this approach in the case of k-means clustering iterations.[14] In this case, determining the interval of interest is difficult as we have not explored a heuristic-based criterion. Unfortunately, as the number of k-means clustering iterations increases, the runtime of the deconvolution algorithm increases. Explicitly, we found that running k-means clustering for more than five iterations poses a significant additional computational cost to our deconvolution algorithm. For

[13]For this search, the remaining input parameters were fixed as the following: $\tau = 11$, $k_{it} = 6$, and $k_{NN} = 6$.

[14]For this search, the remaining input parameters were fixed as the following: $\tau = 11$, $E = 2$, and $k_{NN} = 6$.

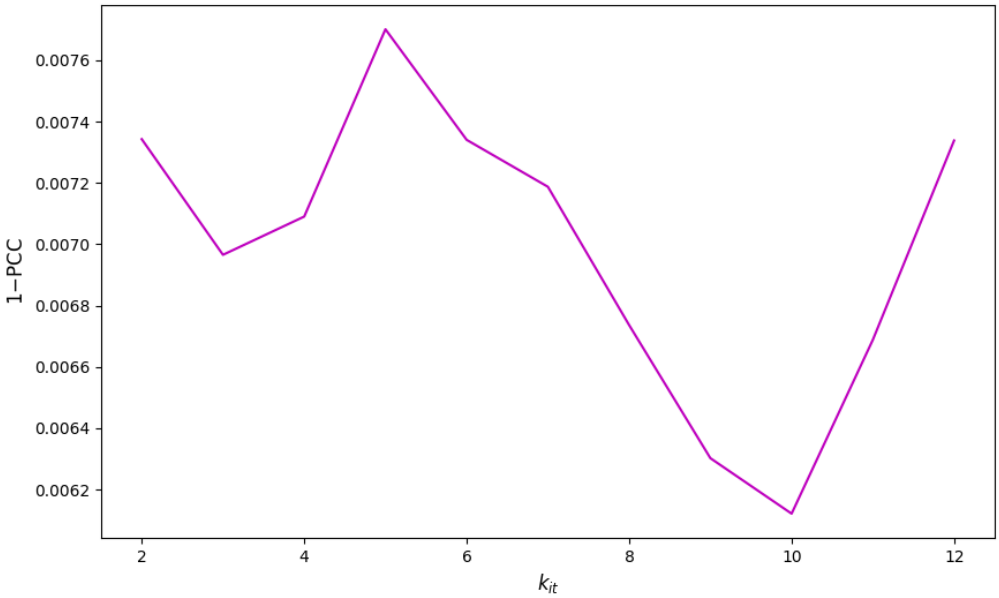


Figure 4.7: Reconstruction error plotted as a function of the number of k-means iterations for the Lorenz system

this reason, we decided to restrict[15] k_{it} to at most 12. According to Figure 4.7, $k_{it} = 10$ minimizes reconstruction error, as measured by 1-PCC. However, a more practical (in terms of computational runtime) option may be $k_{it} = 3$ since the difference in 1-PCC between $k_{it} = 3$ and $k_{it} = 10$ is roughly 10^{-3}.

Figure 4.8 displays the results of this approach in the case of near-neighboring Voronoi cell representatives used in interpolation.[16] Similar to k-means clustering iterations, determining the interval of interest is difficult as we have not explored a heuristic-based criterion to optimize k_{NN}. Using similar rationale as before, we decided to restrict k_{NN} to at most 12. Based on Figure 4.8, $k_{NN} = 7$ minimizes reconstruction error; however, the difference in 1-PCC between $k_{NN} = 6$ and $k_{NN} = 7$ is effectively negligible (roughly 10^{-5}). Thus, $k_{NN} = 6$ may be preferred in the interest of computational efficiency.

4.4 Voronoi Cell Count Optimization

Now, we can briefly describe a baseline method to optimize the number of Voronoi cells used in the unstructured mesh, N_C. This particular procedure is derived from the work of the 2020 RIPS-AFRL research team. In short, this team found a linear relationship between the logarithms of the optimal number of Voronoi cells and the

[15]For $k_{it} > 12$, the runtime of the deconvolution algorithm is too severely impacted with large N_V.

[16]For this search, the remaining input parameters were fixed as the following: $\tau = 11$, $E = 2$, and $k_{it} = 6$.

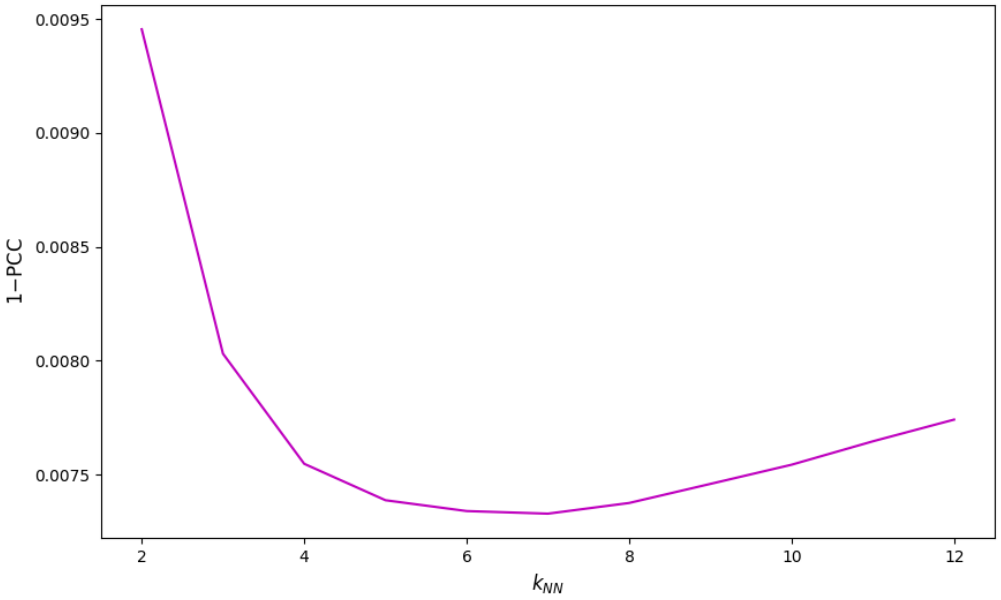


Figure 4.8: Reconstruction error plotted as a function of the number of near-neighboring Voronoi cell representatives for the Lorenz system

amount of training data points from X,

$$\log_2(N_C) = 1.72 \cdot \log_{10}(X_{train}) + 1.08$$

where X_{train} is the amount of available training data [4].

We guessed that this linear relationship would remain intact (up to a constant) if we replaced the amount of training data with N_V,

$$\log_2(N_C) = 1.72 \cdot \log_{10}(N_V) + 1.08$$

since our training time domain equals $N_V \cdot N_w + (E-1)\tau$. In the case of the Lorenz attractor, this modified linear relationship yields $N_C = 439$, and the reconstruction accuracy associated with this choice of Voronoi cell count is moderate (see Table 4.2). We explore this relationship in more detail in Chapter 5 and obtain different equations describing the influence of input size on Voronoi cell count, which return more refined estimates of N_C.

4.5 Proposed Method

Although single parameter optimization provides input parameters which are superior to those returned by heuristic-based methods, the technique fails to fully capture the dependencies among the parameters of interest. We hypothesized that a brute force algorithm which locates the set of parameters that maximize PCC or minimize Normalized RMSE would perform better than single parameter optimization since

every input parameter would be varied across an interval of interest (as opposed to leaving all but one constant). It turns out that selecting this interval of interest for each parameter is a non-trivial exercise.[17] We choose to center our ranges at the value given by the appropriate baseline method for each parameter because we assume that the window of values which includes the optimal choice also includes the parameter value returned by the heuristic-based method. Based on Figures 4.5 and 4.6, it seems that this assumption is reasonable as the optimal parameter values given by Fraser's criterion and Cao's method[18], respectively, are indeed very similar to those returned by the proposed brute force algorithm (as indicated by Tables 4.3 and 4.4). For k_{it} and k_{NN}, we center the intervals of interest based on the output of single parameter optimization. After evaluating the trade-off between additional improvements in reconstruction accuracy and increased runtime, we realized that using $k_{it} = 3$ as the basis for the interval of interest was simply the most feasible option. Similarly, we decided to use $k_{NN} = 6$ as the basis for the corresponding interval of interest. Finally, we used $N_C = 170$ as the basis for the interval of interest when optimizing Voronoi cell count due to the success of our convergence analysis (see Chapter 5). Based on these intervals, our brute force algorithm returned the optimal parameter values listed in Tables 4.3 and 4.4 according to PCC and Normalized RMSE, respectively, for the Lorenz attractor system.[19]

PCC	Normalized RMSE	τ	E	k_{it}	k_{NN}	N_C
0.97440	0.03806	10	2	10	7	439

Table 4.2: The optimal parameters selected for Lorenz data according to baseline methods

PCC	τ	E	k_{it}	k_{NN}	N_C
0.99081	11	2	6	6	175

Table 4.3: The optimal parameters selected for Lorenz data according to the brute force method with PCC maximization

Normalized RMSE	τ	E	k_{it}	k_{NN}	N_C
0.02279	12	2	6	8	180

Table 4.4: The optimal parameters selected for Lorenz data according to the proposed method with Normalized RMSE minimization

Notice that the parameters selected by the proposed method are relatively consistent, regardless of the metric of reconstruction accuracy. Moreover, the parameters

[17]This is another reason to justify why single parameter optimization is an important step in developing the proposed method.

[18]The same holds true for the optimal parameter values returned from single parameter optimization.

[19]In the interest of computational efficiency, we ran the exhaustive brute force search for $N_V = 30000$ instead of $N_V = 100000$. Nevertheless, single parameter optimization proved fruitful in determining search radii.

returned by the baseline[20] methods (listed in Table 4.2) appear to align closely with those deemed optimal by the brute force algorithm. The exception, of course, is the number of Voronoi cells. The fact that the baseline method returns such a large number of Voronoi cells further motivates the analysis in Chapter 5, as the optimization strategy presented in Section 4.4 is clearly flawed. It should be apparent that the parameters returned by the proposed method outperform those returned by baseline methods. However, we are confident that the differences in reconstruction accuracy—both in terms of PCC and Normalized RMSE—are largely as a result of a sub-optimal choice of N_C; for instance, the difference in Normalized RMSE between the sets of parameters listed in Tables 4.2 (except with $N_C = 170$, which we obtain by plugging $N_V = 30000$ into Equation 5.1) and 4.4 is marginally greater than 10^{-3}. This means, if one is willing to sacrifice a slight improvement in reconstruction accuracy (i.e. higher PCC or lower Normalized RMSE), then using heuristic-based criteria (with the updated equations relating N_V and N_C from Chapter 5) and single parameter optimization may be more practical than performing a computationally-expensive brute force search. One can imagine however that, in many applications, maximizing absolute reconstruction accuracy is critical and, in these cases, the proposed method is an attractive option.

[20]Including single parameter optimization

Chapter 5

Convergence Analysis

5.1 Convergence to a Known Signal

For our test systems, we have access to the original Y signal and thus can directly evaluate the accuracy of our reconstructions. For example, with the Lorenz system, we convolute the Y signal (which we obtain by solving the system of differential equations in Equation 8.1), reconstruct it, and calculate the Normalized RMSE and PCC between the reconstructed signal and the target signal (according to Section 2.5). As we increase the input size N_V, which is the number of time-averaged data points in the V signal, we expect the reconstructed signal to statistically converge to the original signal. We can check for this convergence by showing that Normalized RMSE and 1-PCC both approach zero as N_V goes to infinity. We hypothesize that the number of Voronoi cells, N_C must also change as we increase N_V, to maintain the quality of the reconstructions.

Below, we optimize the number of Voronoi cells for the Lorenz system using a similar procedure as the 2020 RIPS-AFRL research team [4]. We examined the effect of varying both N_V and N_C on reconstruction error. For each test, we used the following parameters: $N_w = 200$ timesteps, $\tau = 11$, $E = 2$ with no k-means clustering or interpolation. Figures 5.1 and 5.2 display the results of these tests. Note that each colored line represents a different number of Voronoi cells.

Using these results, we find the optimal number of Voronoi cells for each input size, which are denoted in Figures 5.1 and 5.2 as the red points. These particular points were selected as they minimize Normalized RMSE or 1-PCC, respectively. In Figures 5.3 and 5.4, we plot the optimal number of Voronoi cells as a function of N_V. Interestingly, the optimal number of Voronoi cells was not consistent between the two error metrics for a given input size; that is, the red dots in Figures 5.1 and 5.2 lie on different colored lines. We then used least-squares regression to obtain the best-fit lines in Figures 5.3 and 5.4. The equation of the line in Figure 5.3 provides an estimate for the required number of Voronoi cells to maximize PCC based on a given input size[1]:

[1]We truncate the results given by Equations 5.1 and 5.2 to obtain an integer number of Voronoi cells.

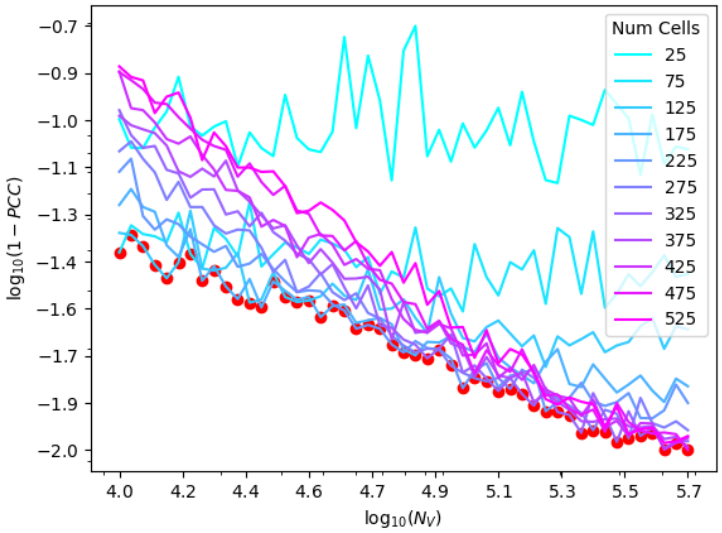


Figure 5.1: Error (measured as 1-PCC) for deconvolution measured as a function of increasing input sizes and variable number of Voronoi cells. Each colored line corresponds to a different number of Voronoi cells. The optimal N_C value for each input size is depicted in red.

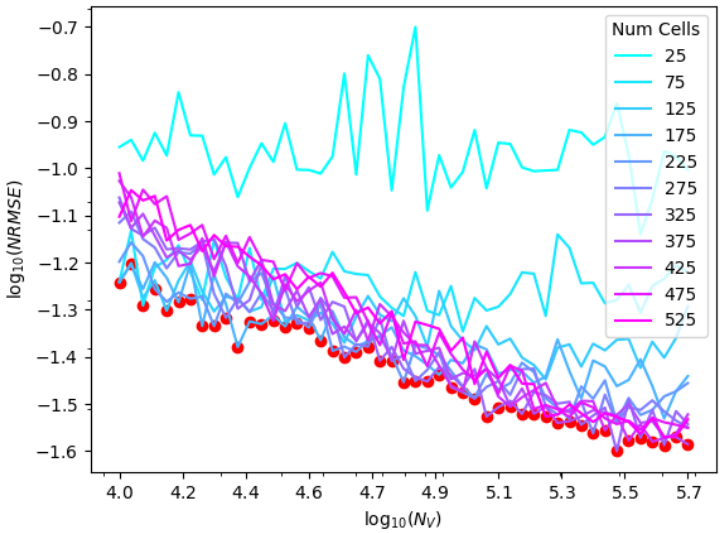


Figure 5.2: Error (measured as Normalized RMSE) for deconvolution measured as a function of increasing input sizes and variable number of Voronoi cells. Each colored line corresponds to a different number of Voronoi cells. The optimal N_C value for each input size is depicted in red.

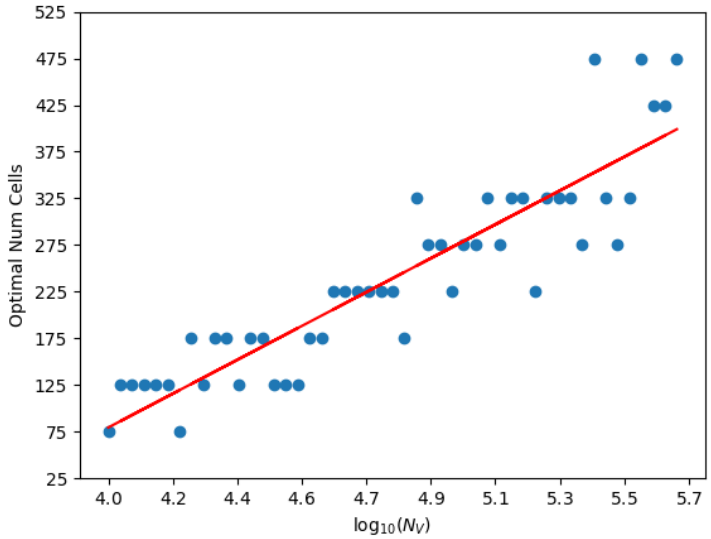


Figure 5.3: Number of Voronoi cells that maximize PCC plotted as a function of input size. The line of best fit is shown in red.

$$N_{C,PCC} = 190.23 \cdot \log_{10}(N_V) - 681.44. \tag{5.1}$$

The equation of the line in Figure 5.4 provides an estimate for the required number of Voronoi cells to minimize Normalized RMSE based on a given input size:

$$N_{C,NRMSE} = 144.02 \cdot \log_{10}(N_V) - 416.97. \tag{5.2}$$

Using these equations to determine the optimal number of Voronoi cells, we plot the error in the reconstructions of $Y(t)$ from the Lorenz system with increasing input sizes (see Figure 5.5). We used Equation 5.1 to generate Figure 5.5a and Equation 5.2 to generate Figure 5.5b. We see that the error in the reconstructed signal (using either 1-PCC or Normalized RMSE) decreases linearly as the input size increases. However, since Figure 5.5a has an R^2 value of 0.975 and Figure 5.5b has an R^2 value of 0.946, the variation in the green points in each plot is described moderately well by the lines of best fit. So, although the green lines in Figures 5.5a and 5.5b suggest that reconstruction error linearly converges to zero at rates of 0.422 and 0.528, respectively, we cannot claim statistical convergence. Rather, we can only say that the green dots follow a general linear pattern and thus will converge at some rate between 0 and 1. A discussion of the limitations of this convergence analysis can be found in Section 7.2.

5.2 Convergence to an Unknown Signal

For experimental data, there is no 'true' signal to compare our reconstructions to. As a substitute, we compare our deconvoluted signal to the reconstruction with $N_w = 1$ timestep. This is because we lose no information from the signal by convoluting to

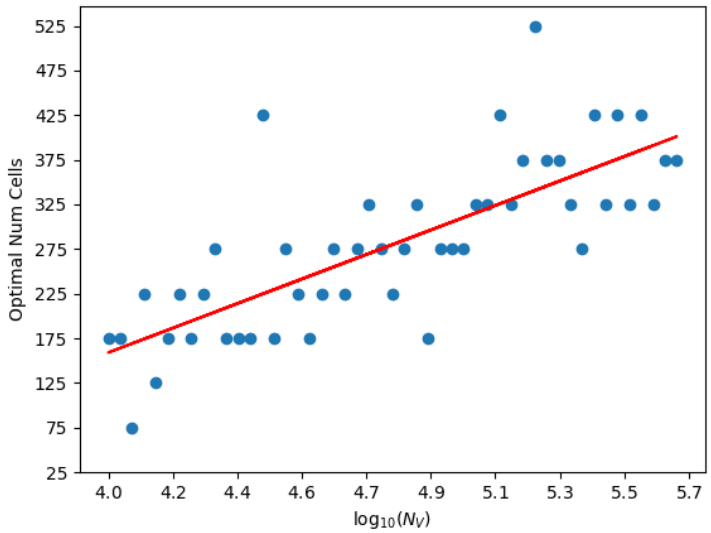
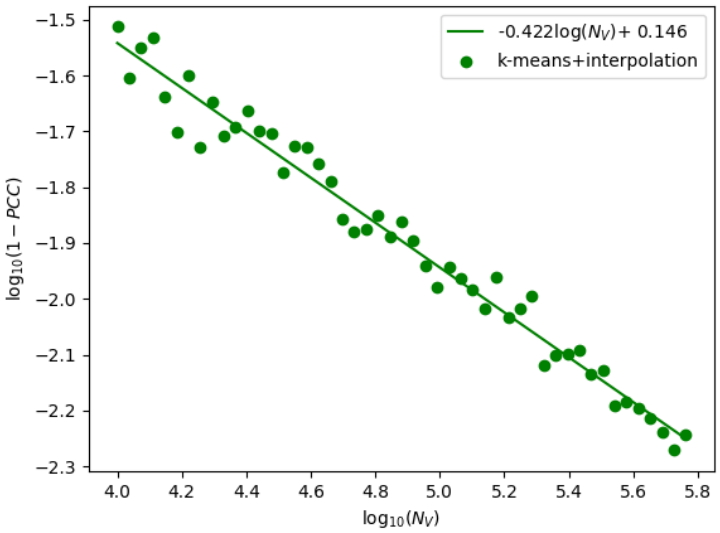


Figure 5.4: Number of Voronoi cells that minimize Normalized RMSE plotted as a function of input size. The line of best fit is shown in red.

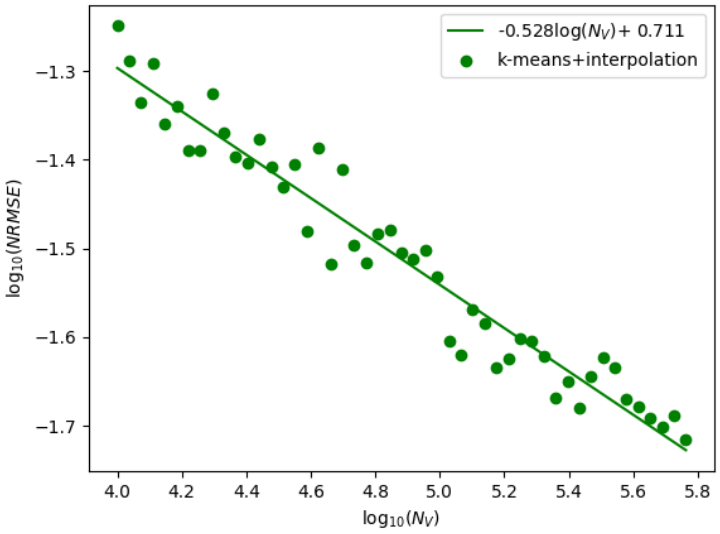
construct $V(t)$, so the reconstruction should be as close to the true signal as possible. Figure 5.6 shows a reconstruction of the HET cathode current using $N_w = 1$ timestep, $N_V = 100000$, $E = 10$, $\tau = 50$, $k_{it} = 5$, and $k_{NN} = 8$. We found that, contrary to our expectations, the error in the reconstruction does not decrease consistently with increasing input size (visualized in Figure 5.7). This could be due to a variety of reasons, including the fact that our reference signal is not the actual 'true' signal. This discrepancy might explain the strange behavior seen in Figure 5.7. Another reason could be the fact that we were unable to obtain the optimal input parameters for HET data (using the proposed method) due to time constraints. In general, we expect the HET reconstructions to be worse since the data is experimental and there are limits to how much data we can actually use. Nevertheless, we are optimistic about achieving better convergence results in the future (see Section 7.2).

5.3 Method Refinement

To refine our methods, we implemented k-means clustering and interpolation. The effectiveness of these methods can be determined by comparing the convergence plots of each method. Figures 5.8 and 5.9 show how reconstruction accuracy is impacted with no additional refinement, with k-means clustering (see Step 3 of A2), with interpolation (see Step 8 of A2), and with both k-means clustering and interpolation for Lorenz data. Interpolation increased the accuracy of our results, while k-means clustering produced mixed results, i.e. sometimes improving accuracy but decreasing accuracy on other occasions. In the future, it would be fruitful to consider additional iterations of k-means clustering (e.g. 10 iterations) to achieve consistent improvements in accuracy.



(a)



(b)

Figure 5.5: Convergence plots for Lorenz data using (a) 1-PCC and (b) Normalized RMSE

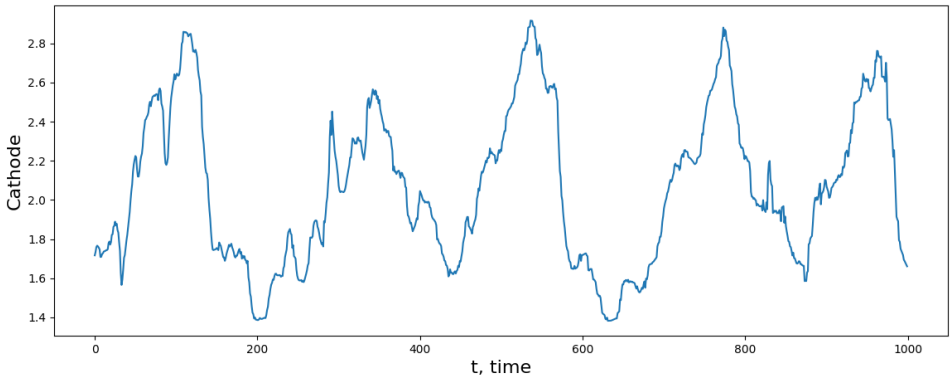


Figure 5.6: A reconstruction of the HET cathode current from the HET anode current using $N_w = 1$. This is the baseline signal for comparisons throughout this section.

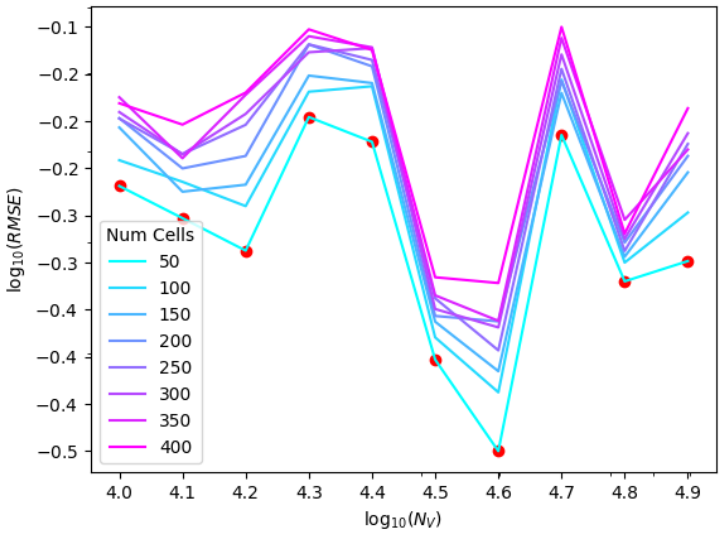


Figure 5.7: Error (measured by RMSE) in the HET cathode current deconvoluted using HET anode current as the clean reference signal for increasing N_V and variable number of Voronoi cells

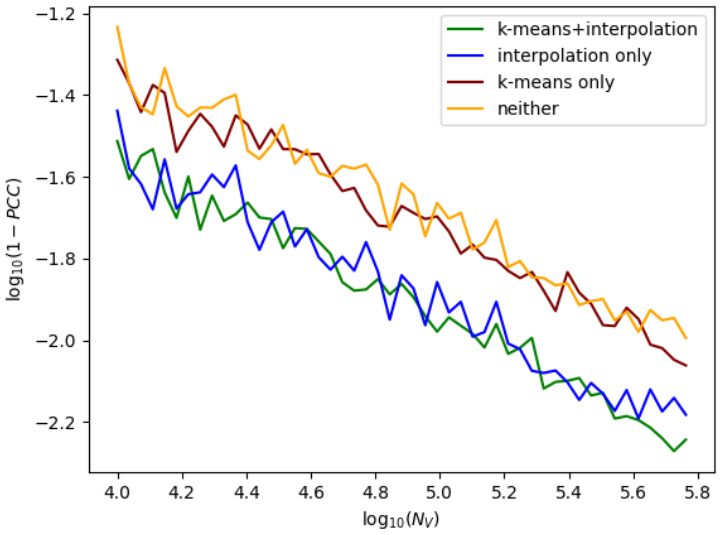


Figure 5.8: Error (measured as 1-PCC) for deconvolution of $Y(t)$ from the Lorenz system measured as a function of increasing input sizes and different types of method refinement

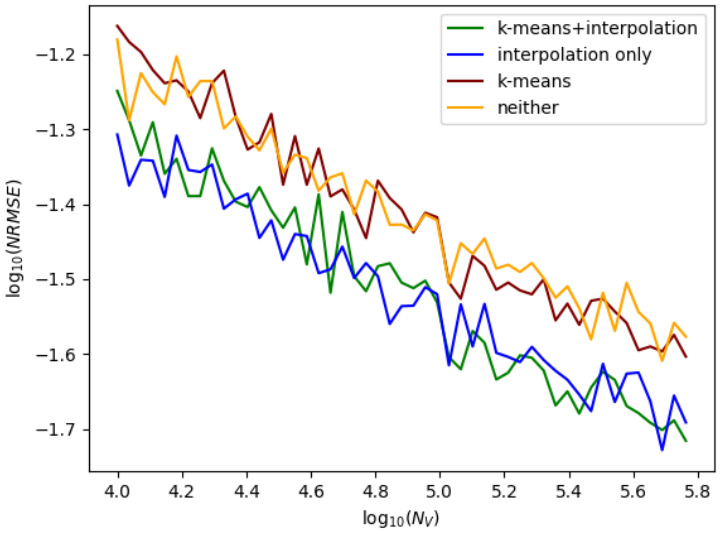


Figure 5.9: Error (measured as Normalized RMSE) for deconvolution of $Y(t)$ from the Lorenz system measured as a function of increasing input sizes and different types of method refinement

Chapter 6

Multiple Pixel Systems

Since we were able to improve runtime, it became feasible to deconvolute multiple signals at once that are all coupled to the same reference signal. We considered a scenario where we have many signals coupled to the same reference signal. In this case, there is one clean reference signal, $X(t)$, and several target signals, $\{Y_1(t), Y_2(t), \ldots, Y_P(t)\}$. We say that each of these target signals represents one pixel and that P is the total number of pixels. Each of these target signals can be found from corresponding convoluted signals, $\{V_1(t), V_2(t), \ldots, V_P(t)\}$, by solving the linear system $A\vec{x}_j = \vec{b}_j$ for the same matrix A given by Step 5 of A2 where $j \in \{1, 2, \ldots, P\}$. Right now, solving each of these linear systems happens in parallel, so the runtime for deconvoluting all of the pixels is comparable to deconvoluting one pixel.

6.1 Movies

Our proof-of-concept example was adapting the Sine system (see Section 8.1.2) to adjust slightly for each $j \in \{1, 2, \ldots, P\}$ so that each pixel is still coupled to the same reference signal but has a different value. The system of interest is given by:

$$\begin{cases} X(t) = \sin(t) \\ Y_j(t) = X(t)(-j + X(t)). \end{cases}$$

The results of deconvoluting this system, where $V_j(t)$ was found by averaging in a time window of 700 timesteps, are shown in Movie 1.[1] By averaging the reconstruction PCC and NRMSE for each signal in the movie, we find that the movie has PCC of 0.9998 and an NRMSE 0.0161, which is an excellent reconstruction.

We also generated a radial Sine movie, where the signal for the pixel in the j^{th} row and the k^{th} column is given by:

$$\left(\sqrt{(1 + (j-6)^2 + (k-6)^2)/12}\right)((-1 + \sin(t)) \cdot \sin(t)).$$

This system can be viewed in Movie 2.[2] A single frame of Movie 2 is shown in Figure 6.2.

[1] The link to Movie 1 is https://ibb.co/XtR1tWD.

[2] The link to Movie 2 is https://ibb.co/VqtBPcZ.

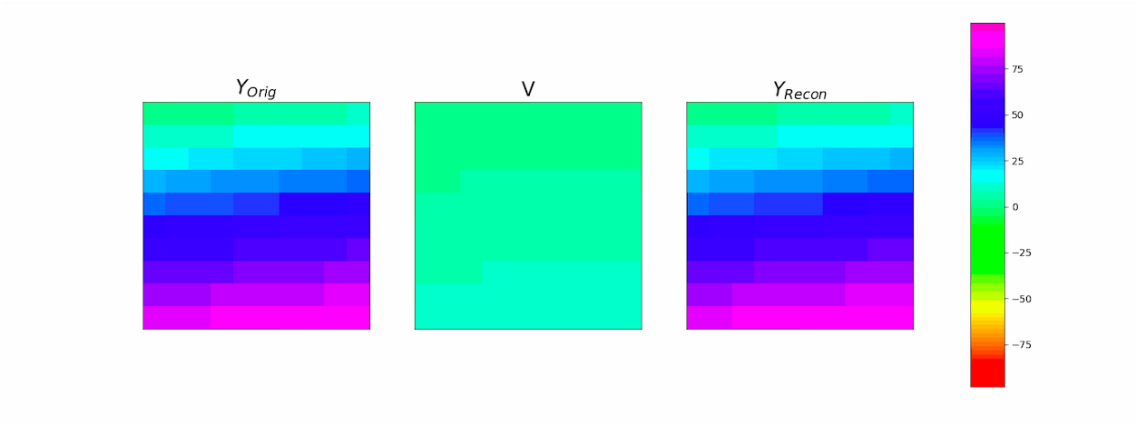


Figure 6.1: A single frame from the Sine system movie

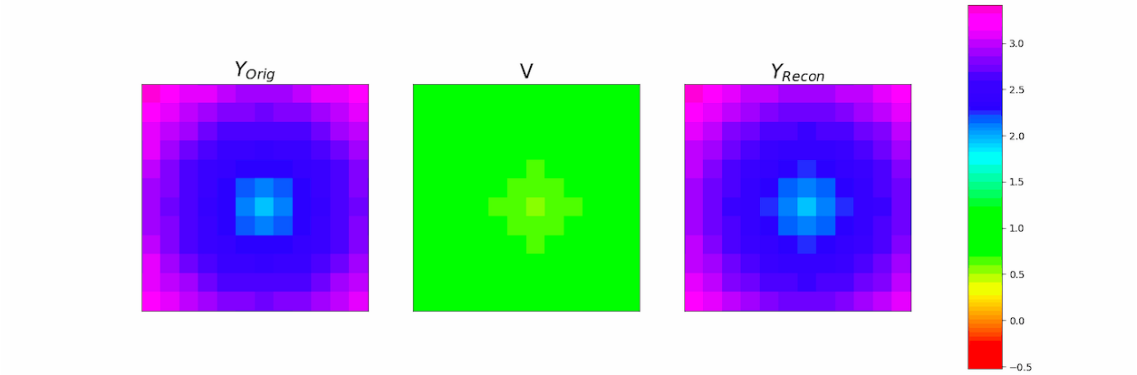


Figure 6.2: A single frame from the radial Sine movie

The average Normalized RMSE and PCC is calculated by finding the error of each individual pixel, adding together all of these values, and dividing by the total amount of pixels. For this radial system, we again found the average Normalized RMSE and PCC over all pixels. The average Normalized RMSE was 0.02281 and the average PCC was 0.99785. This indicates that, on average, each pixel mirrored the true target signal.

Chapter 7

Conclusions

7.1 Summary

By combining the grid deconvolution and Voronoi diagram denoising algorithms, we successfully deconvoluted corrupted, temporally under-resolved signals. We modified the deconvolution algorithm to include Voronoi diagram-based partitioning of phase space and successfully shortened its runtime. A critical step in accomplishing these speed-ups was converting the existing code in Octave to C++. Optimizing the deconvolution algorithm allowed us to quickly recover multiple signals coupled to the same clean reference signal, and this result was critical to the development of our movies. Furthermore, we developed new strategies for optimizing the input parameters to the deconvolution algorithm. Optimizing these parameters substantially increased reconstruction accuracy, which was quantified by the Pearson Correlation Coefficient and Normalized Root Mean Squared Error.

7.2 Suggestions for Future Research

Our work requires more rigorous testing and standardization of testing methods. That is, when we find the convergence of a reconstruction as the training size increases, we should be sampling the error over several training regions and testing regions for the same size. This is equivalent to shifting the training manifold and then averaging the PCC and RMSE for these training manifolds to get a more accurate understanding of the influence of training size on reconstruction error. We are also interested in exploring the relationship between PCC and RMSE. For the HET system (and the Lorenz system, to a lesser extent), the rate of convergence of the reconstruction to the target signal (i.e. reconstruction error decreasing to zero) depends on the error quantification metric. We do not know exactly why this is the case, but it might be fruitful to develop this idea further.

In addition to the systems mentioned previously, our methods could also be applied to other dynamical systems. As discussed in Chapter 2, our algorithm can be applied to systems where the changes in the system occur faster than the rate of data acquisition. For instance, our algorithm could improve our understanding of fluid dynamics by studying the chaotic phenomenon of vortex shedding. With the ability

to now deconvolute many pixel signals, we could use an upstream point in the system as the reference signal for more chaotic behaviour downstream.

Exploring alternate interpolation methods might also be of interest. One substitute for our interpolation scheme is Delaunay triangulation. Delaunay triangulation is a technique for creating a mesh of triangles with a set of data points. Another option is radial basis interpolation. Radial basis functions help to interpolate scattered data points of high dimension.[1] Successful implementation of new interpolation methods may improve reconstruction accuracy, as well as the computational runtime of our deconvolution algorithm.

[1]For more details about radial basis functions, see [11].

Chapter 8

Appendices

8.1 Descriptions of Test Systems

8.1.1 Lorenz System

The Lorenz attractor is described by the following system of ordinary differential equations:

$$\begin{cases} \frac{dX}{dt} = \sigma(Y - X) \\ \frac{dY}{dt} = X(\rho - Z) - Y \\ \frac{dZ}{dt} = XY - \beta Z. \end{cases} \tag{8.1}$$

Throughout this report, we analyzed Equation 8.1 with $\sigma = 10$, $\rho = 30$, $\beta = \frac{8}{3}$, and the initial conditions $(X(0), Y(0), Z(0)) = (1, 1, 1)$.

8.1.2 Sine System

The Sine system is described by the following system of equations:

$$\begin{cases} X(t) = \sin(t) \\ Y(t) = \sin(t)(-1 + \sin(t)). \end{cases} \tag{8.2}$$

Chapter 9

Glossary

9.1 Technical Definitions

Anode: Electrons where oxidation occurs

Cathode: Electrode where reduction occurs

Cell Representative: A time-delay embedded data point randomly selected from the shadow manifold which represents a single cell of the unstructured mesh

Diffeomorphism: Invertible function that maps one manifold to another such that the function and mapping are smooth

Interpolation: Estimation of unknown values based on known values; here, we assign the average of the values associated with the k_{NN} nearest Voronoi cell representatives to each point in phase space

***k*-Means clustering**: A type of cell-adaptation in which cell representatives become the centers of mass of the cells

Nearest neighbor: For a subset of data A, the point $p \in A$ is j's nearest neighbor if $d(j,p) \leq d(j,q)$ for all $q \in A$, where d is the Euclidean distance (unless otherwise specified)

Phase Space: Space in which all possible states of a system are represented, with each possible state corresponding to one unique point in the phase space

Shadow manifold: The manifold generated by plotting entries of the vector $(X(t), X(t-\tau), X(t-2\tau), \ldots, X(t-(E-1)\tau))$

Temporally Resolved: Discrete resolution of a measurement with respect to time

Testing data: The data used from a clean signal in the form of time-delay embedded points used for signal reconstruction in the testing phase of the unstructured mesh reconstruction algorithm

Training data: The data used from both a clean signal and noisy signal to assign average values to a Voronoi diagram in the training phase of the unstructured mesh reconstruction algorithm

Uniform mesh: A grouping of time-delay embedded data points constructed using a grid of uniform spacing on the shadow manifold

Unstructured mesh: A grouping of time-delay embedded data points constructed using the Voronoi diagram on the shadow manifold

Voronoi diagram: Diagram which groups points into cells based on nearest-neighbor identification; often used as a synonym for "unstructured mesh"

9.2 Notation

M_X:	An E-dimensional shadow manifold of M formed by plotting E time lags of X
$M_{X,train}$:	The subset of the shadow manifold corresponding to the training time domain
$M_{X,test}$:	The subset of the shadow manifold corresponding to the testing time domain
E:	The embedding dimension
τ:	The time lag parameter
$\tilde{Y}(t)$:	A time series signal corrupted by noise, where the true, noiseless signal is $Y(t)$
$V(t)$:	Convoluted version of the signal $\tilde{Y}(t)$
N_V:	Number of points in the temporally under-resolved signal, $V(t)$
τ_w:	Time window length
N_w:	The number of timesteps in a temporally resolved signal in a time window
N_t:	The amount of timesteps in a Voronoi cell in phase space
c_{pq}:	$\frac{N_t}{N_w}$ for one point in $V(t)$ and one cell in phase space
Q:	The total number of cells in the Voronoi diagram-based partition of phase space
q:	The index of a cell in the Voronoi diagram-based partition of phase space
Y_q:	The average value of Y in Voronoi cell q and the value that is assigned to the q^{th} cell after solving the linear system

N_C: The number of Voronoi cells used in the unstructured mesh

k_{it}: The number of iterations of the k-means clustering algorithm applied to the training manifold

k_{NN}: The number of near-neighboring cell representatives used in interpolation applied to the testing manifold

9.3 Abbreviations

AFRL: Air Force Research Laboratory. The industrial sponsor of this research.

CCM: Convergent cross mapping. A method developed to detect causality between signals.

HET: Hall-effect thruster. A type of ion thruster studied by AFRL. This system outputs the experimental data we analyze in this work.

IPAM: Institute for Pure and Applied Mathematics. An institute of the National Science Foundation, located at UCLA.

MI: Mutual Information. A measure of general dependence of two signals.

PCC: Pearson Correlation Coefficient. A method used in measuring error.

RIPS: Research in Industrial Projects for Students. A regular summer program at IPAM, in which teams of undergraduate (or incoming graduate) students participate in sponsored research projects.

RMSE: Root Mean Squared Error. A method used in measuring error.

UCLA: The University of California at Los Angeles.

Selected Bibliography Including Cited Works

[1] S. J. Araki, J. W. Koo, R. S. Martin, and B. Dankongkakul, *A grid-based nonlinear approach to noise reduction and deconvolution for coupled systems*, Physica D: Nonlinear Phenomena, 417 (2021), p. 1–12.

[2] B. E. Beal, *The air force research laboratory's in-space propulsion branch*. Preprint found from https://apps.dtic.mil/sti/citations/ADA622305.

[3] L. Cao, *Practical method for determining the minimum embedding dimension of a scalar time series*, Physica D: Nonlinear Phenomena, 110 (1997), p. 43–50.

[4] M. DeBrito, J. Botvinick-Greenhouse, A. Kirtland, M. Osborne, C. Johnson, R. Martin, D. Eckhardt, and S. Araki, *An unstructured mesh approach to nonlinear noise reduction*, Research in Industrial Projects for Students, (2020). Report available in the IPAM repository.

[5] D. Eckhardt, J. Koo, R. Martin, M. Holmes, and K. Hara, *Spatiotemporal data fusion and manifold reconstruction in hall thrusters*, Plasma Sources Science and Technology, 28 (2019), p. 1–14.

[6] A. M. Fraser and H. L. Swinney, *Independent coordinates for strange attractors from mutual information*, Physical Review A, 33 (1986), p. 1134–1140.

[7] C. E. Huerta, R. S. Martin, D. Q. Eckhardt, and J. W. Koo, *Determining causality in hall effect thrusters using extended convergent cross mapping*, Plasma Sources Science and Technology, (2021), p. 1–11.

[8] R. Martin, J. Koo, and D. Eckhardt, *Impact of embedding view on cross mapping convergence*. Preprint obtained from industry representatives from AFRL, 2021.

[9] G. Sugihara, R. May, H. Ye, C.-h. Hsieh, E. Deyle, M. Fogarty, and S. Munch, *Detecting causality in complex ecosystems*, Science, 338 (2012), p. 496–500.

[10] F. Takens, *Detecting strange attractors in turbulence*, Lecture Notes in Mathematics, (1981), p. 366–381.

[11] C. E. Torres and L. Barba, *Fast radial basis function interpolation with gaussians by localization and iteration*, Journal of Computational Physics, (2009), pp. 4976–4999.

[12] X. Xu, X. Liu, and X. Chen, *The cao method for determining the minimum embedding dimension of sea clutter*, 2006 CIE International Conference on Radar, (2006), p. 1–4.

Chapter 10

Code

10.1 Deconvolution Function

The code can be found at https://github.com/BenBait/AFRL_2021. This is the main deconvolution function and is an implementation of Algorithm A2.

```
typedef Eigen::SparseMatrix<double> SpMat; // declares a column-major sparse matrix type of
typedef Eigen::Triplet<double> Tr;
typedef Eigen::VectorXd d_vect;

/*
 * Inputs:
   --> the struct of global parameters for the algorithm
   --> array of doubles representing the clean reference signal, X
   --> array of eigen library vectors representing the temporally
       under-resolved signal, V, for each pixel
   --> the array of time indices for the start of the time windows used to
       construct V from the temporally resolved signal
 * Returns:
   --> nothing
 * Purpose:
   --> The purpose of the deconvolute function is the reconstruct the target
       signal, Y, for the testing region specified by the global parameters
       from the clean reference signal, X, and the temporally under-resolved
       signal, V
 */
void deconvolute(glbls_struct glbls,
                 double *X,
                 d_vect *V,
                 long *time_indices)
{
    // going to use this outstream to write out at various checkpoints
    std::ofstream outfile;

    // load globals to local variables
    int G_id  = glbls.G_id;
```

```
int G_pix = glbls.G_pix;
int G_N_V = glbls.G_N_V;
int G_test_size = glbls.G_test_size;
int G_gap = glbls.G_gap;
double G_t_w = glbls.G_t_w;
int G_lag = glbls.G_lag;
int G_d = glbls.G_d;
int G_num_samples = glbls.G_num_samples;
int G_k = glbls.G_k;
int G_iter = glbls.G_iter;

int t_offset = (G_d-1)*G_lag;

// i and j are utilitily counters, t_k is time counter
int i, j, t_k;

int train_size_X = G_N_V*(int)G_t_w + t_offset;
// allocate the shadow manifold for training
ANNpointArray M_X_train;
M_X_train = annAllocPts(train_size_X-t_offset, G_d);
// construct the shadow manifold for training
for (t_k = t_offset; t_k < train_size_X; t_k++) {
    for (i = 0; i < G_d; i++) {
        M_X_train[t_k-t_offset][i] = X[t_k - i*G_lag];
    }
}

// these nearest neighbor arrays are so that we don't have to call
// the KD tree when we populate the rows of the matrix A
int **nn_arr = new int*[train_size_X - t_offset];
double **nndist = new double*[train_size_X - t_offset];
for (i = 0; i < train_size_X-t_offset; i++) {
    nn_arr[i] = new int[G_k]();
    nndist[i] =  new double[G_k]();
}

/*
 * Construct the Voronoi partition of phase space
 */
std::vector<int> samples;
// step by (|X| / num_samples) through X for voronoi samples
int step_by = (train_size_X / G_num_samples) - 1;

for (i = 0; i < G_num_samples; i++) {
    samples.push_back(t_offset + i*step_by);
}

// add the INITIAL sample points (in phase space) to build the tree
ANNpointArray points;  // cell representative points
```

```
points = annAllocPts(G_num_samples, G_d);
for (i = 0; i < G_num_samples; i++) {
    for (j = 0; j < G_d; j++) {
        points[i][j] = X[samples[i] - j*G_lag];
    }
}

ANNpointArray new_points;  // data points
new_points = annAllocPts(G_num_samples, G_d);

// Create a KD tree to quickly find nearest neighbors
ANNkd_tree*   tree;    // search structure
tree = new ANNkd_tree(points, G_num_samples, G_d);
ANNidxArray   nnIdx;   // near neighbor indices
ANNdistArray  dists;   // near neighbor distances

ANNpoint curr;

// Group neighbors together and find the center of mass of Voronoi cells
int iter;
for (iter = 0; iter < G_iter; iter++) {
    if (iter != 0) {
        for (i = 0; i < G_num_samples; i++) {
            for (j = 0; j < G_d; j++) {
                // ajust cell centers to centers of mass
                // stored in newpoints
                points[i][j] = new_points[i][j];
                new_points[i][j] = 0;
            }
        }
    }

    // count is the number of points in every cell, which we use to find
    // the center of the mass of each cell
    int *count = new int[G_num_samples]();

    nnIdx = new ANNidx[G_k];  // allocate near neigh indices
    dists = new ANNdist[G_k];

    // go to each point on the shadow manifold and assign it to its nearest
    //cell representative
    for (t_k = t_offset; t_k < train_size_X; t_k++) {

        curr = M_X_train[t_k-t_offset];

        tree -> annkSearch(curr,     // query point
                           G_k,      // number of near neighbors
                           nnIdx,    // nearest neighbors (returned)
                           dists,    // distance (returned)
```

```
                        0.0);

        // build the average coordinate in each dimension
        for (i = 0; i < G_d; i++) {
            new_points[nnIdx[0]][i] += curr[i];
        }

        // count the number of points in the cell
        count[nnIdx[0]]++;

        for (i = 0 ; i < G_k; i++) {
            nn_arr[t_k - t_offset][i] = nnIdx[i];
            // THIS ONLY HAPPENS WITH SINE SYSTEM
            // but sometimes the distance returned by the tree is -nan
            // so we just set those points equal to the cell center point
            if (isnan(dists[i])) {
                nndist[t_k-t_offset][0] = 0;
            }
            nndist[t_k - t_offset][i] = dists[i];
        }
    }
    // adjust cell representative points to the center of mass of each cell
    for (i = 0; i < G_num_samples; i++) {
        for (j = 0; j < G_d; j++) {
            new_points[i][j] /= count[i];
        }
    }
    delete [] nnIdx;
    delete [] dists;
    delete [] count;
}

/*
 * Population of the rows of A
 */
std::vector<Tr> sparse_row;

int row, col;
for (row = 0; row < G_N_V; row++) {
    int curr_col;
    double *curr_row = new double[G_num_samples]();

    for (t_k = time_indices[row]; t_k < time_indices[row] + G_t_w; t_k++) {
        // increment the row with the first nearest neighbor of t_k
        curr_col = nn_arr[t_k-t_offset][0];
        curr_row[curr_col] += (1/G_t_w);
    }

    for (col = 0; col < G_num_samples; col++) {
```

```
            if (curr_row[col] != 0) {
                sparse_row.push_back(Tr(row, col, curr_row[col]));
            }
        }

        delete [] curr_row;
    }

    //N_V is number of rows
    SpMat A(G_N_V, G_num_samples);
    // FILL A
    A.setFromTriplets(sparse_row.begin(), sparse_row.end());

    d_vect *M_Y = new d_vect[G_pix];
    for (i = 0; i < G_pix; i++)
        M_Y[i] = d_vect(G_num_samples);

    /*
     * Solve for x in Ax=b
     */
// we can deconvolute all the V signals coupled to X at once, this won't
// do anything if you don't have access to several cores
#pragma omp parallel
#pragma omp for
    for (i = 0; i < G_pix; i++) {
        // LSQR(A, b, x, gamma) (it's thread safe)
        // gamma tests for convergence in the iterative least squares
        LSQR lsqr = LSQR(A, V[i], M_Y[i], 1.0e-8);
        M_Y[i] = lsqr.SolutionX();
    }

    int tot_size_X = train_size_X + G_gap + G_test_size;

    // shadow manifold for testing
    ANNpointArray M_X_test;
    M_X_test = annAllocPts(G_test_size, G_d);
    // construct the training shadow manifold
    int init_test = train_size_X + G_gap;
    for (t_k = init_test; t_k < tot_size_X; t_k++) {
        for (i = 0; i < G_d; i++) {
            M_X_test[t_k - init_test][i] = X[t_k - i*G_lag];
        }
    }

    // change t_offset to be at the end of the training data
    int c;
    double curr_y, u_sum;
    for (i = 0; i < G_pix; i++) {
```

```
outfile.open("rec"+std::to_string(i)+"_"+std::to_string(G_id)+".dat",
             std::ios::out | std::ios::binary);

for (t_k = init_test; t_k < tot_size_X; t_k++){

    curr = M_X_test[t_k - init_test];

    nnIdx = new ANNidx[G_k];  // allocate near neigh indices
    dists = new ANNdist[G_k];

    tree -> annkSearch(curr,     // query point
                       G_k,      // number of near neighbors
                       nnIdx,    // nearest neighbors (returned)
                       dists,    // distance (returned)
                       0.0);

    // THIS ONLY HAPPENS WITH SINE SYSTEM
    // but sometimes the distance returned by the tree is -nan
    // so we just set those points equal to the cell center point
    for (j = 0; j < G_k; j++) {
        if (isnan(dists[j])) {
            dists[0] = 0;
        }
    }
    u_sum = 0;
    curr_y = 0;

    if (dists[0] == 0){
        // this is if the current t_k is for a voronoi cell representative
        // or if we have overlapping values
        curr_y = M_Y[i](nnIdx[0]);
    } else {
        // if we are not at a center cell, we do a sort of radial weighting
        // to interpolate the reconstruction
        for (c = 0; c < G_k; c++) {
            u_sum += exp(-1*(dists[c] / dists[0]));
        }

        // each Y val is a weighted average of the G_k nearest neighbors
        double u_c, w_c;
        for (c = 0; c < G_k; c++) {
            u_c = exp(-1*(dists[c] / dists[0]));
            w_c = u_c / u_sum;
            curr_y += w_c * M_Y[i](nnIdx[c]);
        }
    }
    delete [] nnIdx;
    delete [] dists;
    outfile.write(reinterpret_cast<char*>(&curr_y), sizeof(curr_y));
```

```
        }
        outfile.close();
    }
}
```