An Introduction to State Space Reconstruction

Applications and Mathematical Prerequisites

J. Robert Buchanan

October 16, 2003

Millersville University of Pennsylvania
email: Bob.Buchanan@millersville.edu

Introduction

"What is past is prologue."

William Shakespeare

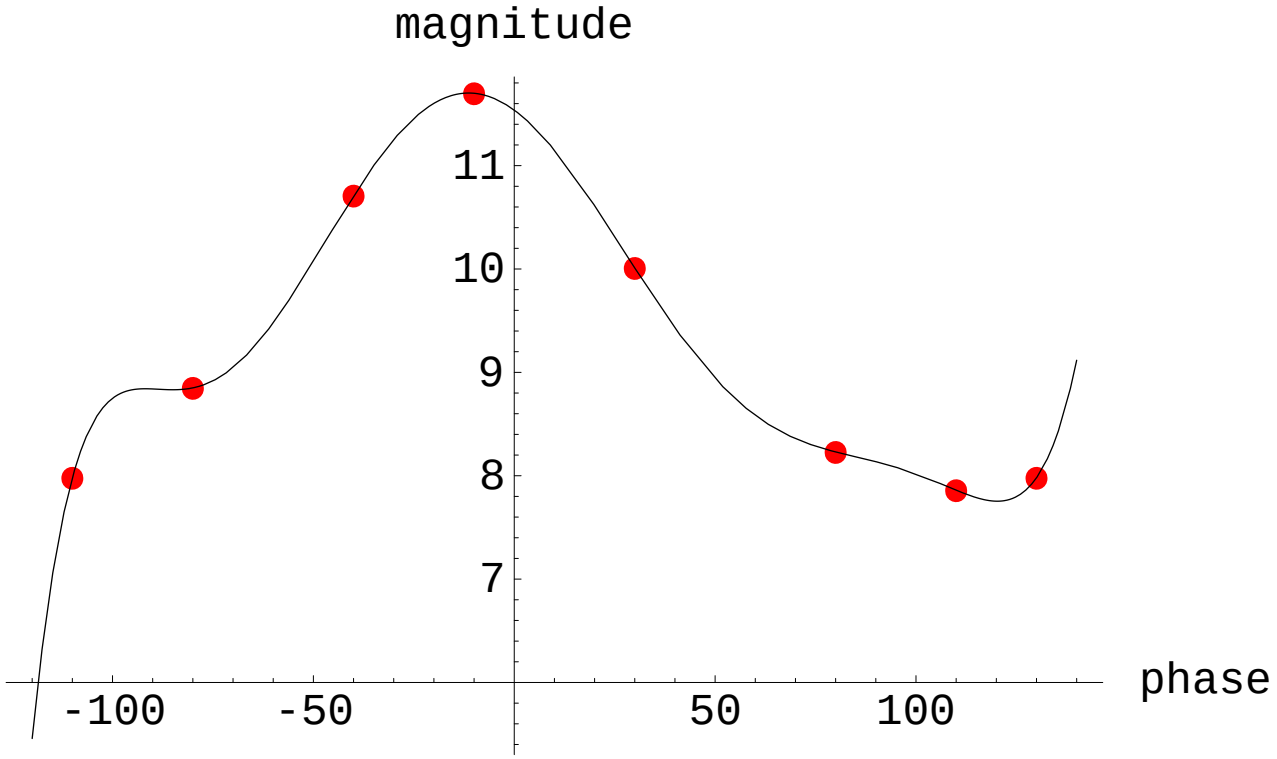
- Prediction based on a time series of observations.
  - Fluid flow
  - Sunspot Activity
  - Mechanical vibrations
  - Ice age
  - Epidemics
  - Human speech

Techniques

- Polynomial and rational interpolation
- Fourier series and wavelets
- Neural networks
- Modeling

Polynomial Interpolation

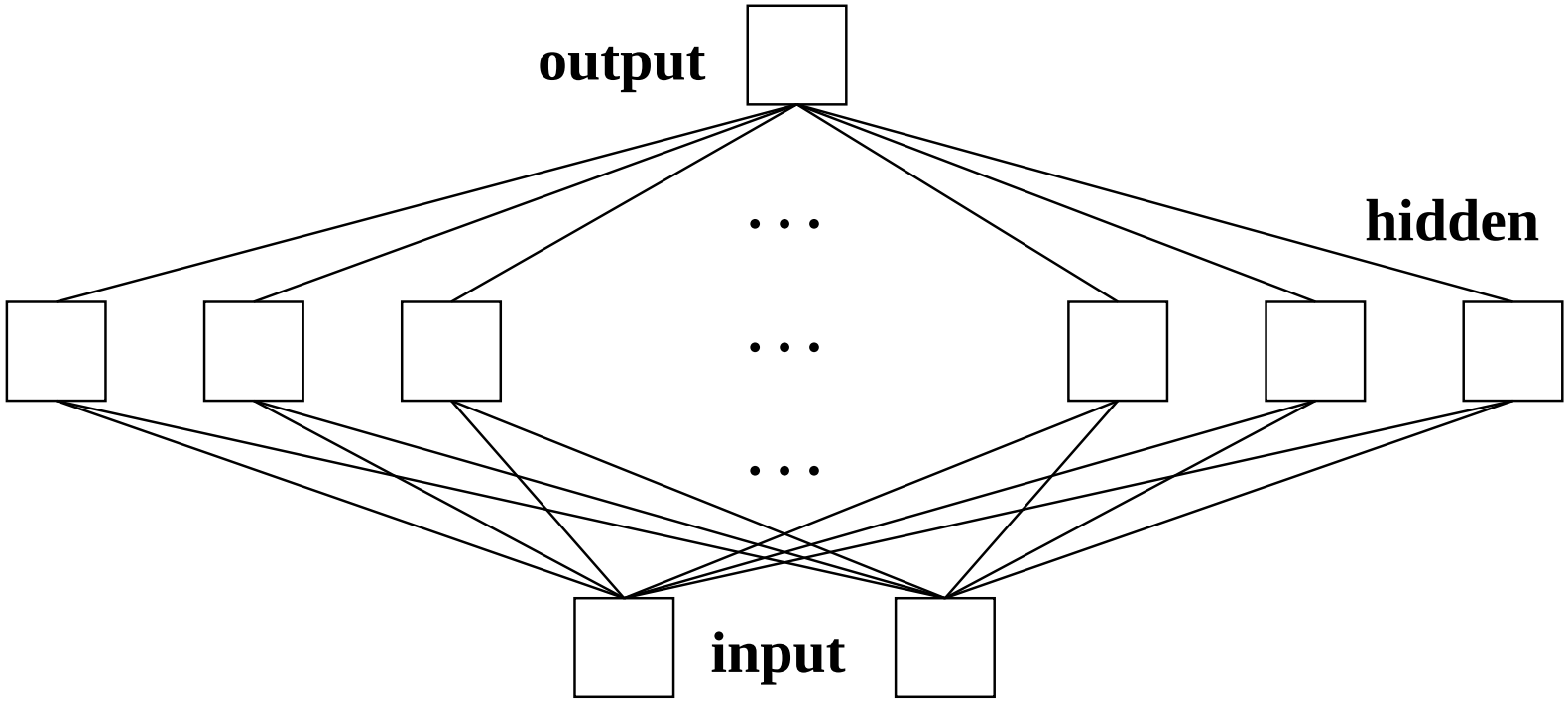
- Blow up as $t \to \pm\infty$
- Poor extrapolation outside of data set



Star S in Big Dipper

Neural Networks

- Non-linear function approximation
- Layers, thresholds, weights, sigmoid functions



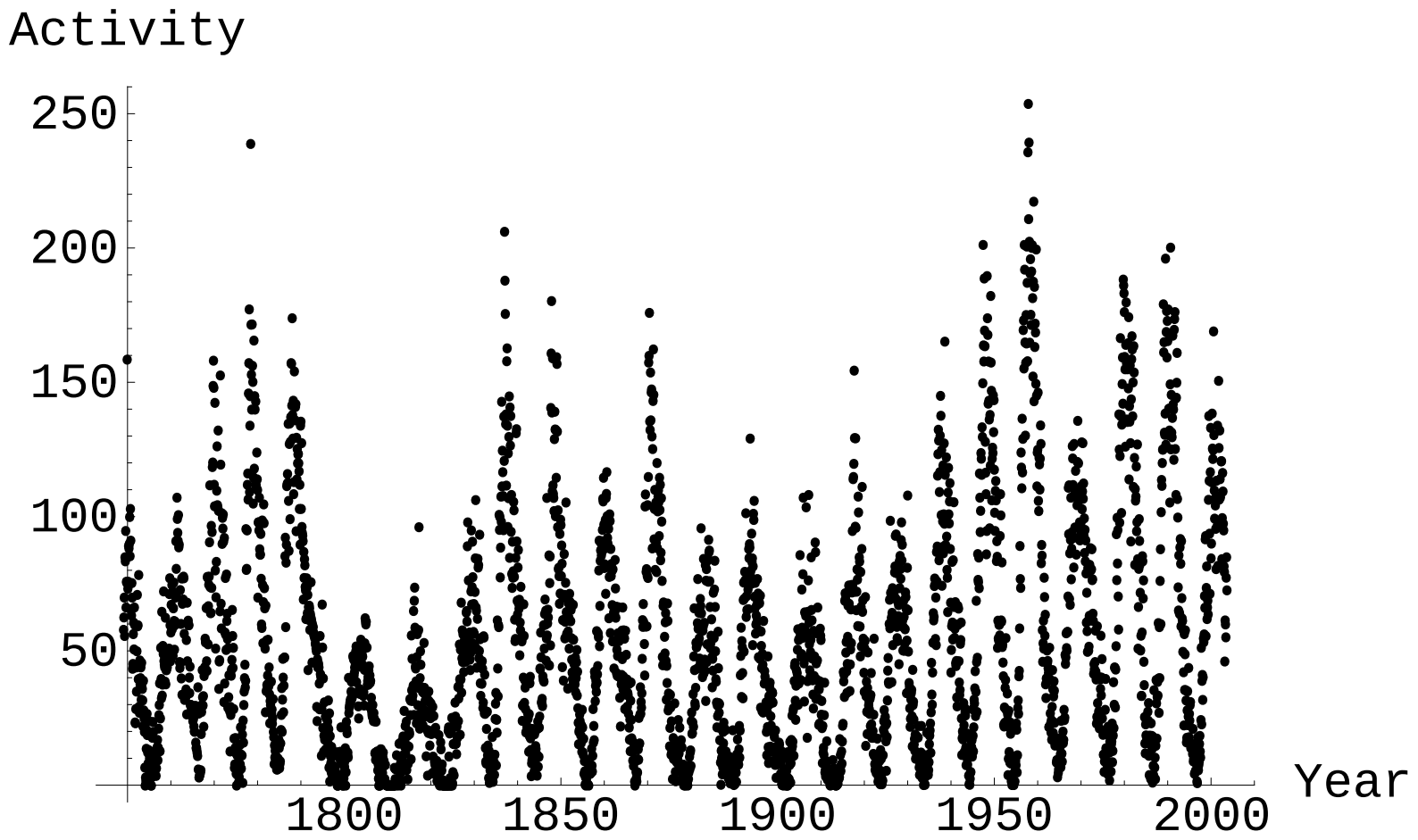
Global *vs.* Local Modeling

- **Global**: derive equations of motion from first principles, measure initial conditions, integrate solution forward in t.

$$\begin{aligned} mu'' + \gamma u' + ku &= F(t) \\ u(t_0) &= u_0 \\ u'(t_0) &= u_0' \end{aligned}$$

- **Local**: build a model directly from data
  - State space reconstruction
  - Nonlinear function approximation

Sunspot Activity Data



Source: www.nasa.gov

Brief Introduction to Local Modeling

Difference equation model:

$$P_t = F(P_{t-1}, P_{t-2}, \ldots, P_{t-j})$$

Sequence of observations over time:

$$\{P_0, P_1, \ldots, P_N\}$$

Time delay embedding with dimension m, Takens (1981):

$$\begin{aligned} \mathbf{x}_{m-1} &= \langle P_0, P_1, \ldots, P_{m-1} \rangle \\ \mathbf{x}_m &= \langle P_1, P_2, \ldots, P_m \rangle \\ &\vdots \\ \mathbf{x}_N &= \langle P_{N-m+1}, P_{N-m+2}, \ldots, P_N \rangle \end{aligned}$$

Takens's Theorem

From Casdagli, *et al.* (1991):

Dynamical system: $x(t) = f^t(x(0)), \quad x \in \mathbb{R}^n$

Observable: $y(t) = g(x(t)), \quad y \in \mathbb{R}^d$

Delay construction map:

$$\Phi(x(t)) = \langle g(f^{-\tau m_p}(x(t))), \ldots, g(x(t)), \ldots, g(f^{\tau m_f}(x(t))) \rangle$$

If $m = m_f + m_p + 1 \geq 2n + 1$ then Φ is a smooth, one-to-one coordinate transformation with a smooth inverse.

Local Modeling 2

Prediction:

$$\hat{P}_{N+1} = G(\mathbf{x}_N)$$

Multi-step prediction:

$$\begin{aligned}
\hat{P}_{N+2} &= G(\langle P_{N-m+2}, P_{N-m+3}, \ldots, \hat{P}_{N+1} \rangle) \\
\hat{P}_{N+3} &= G(\langle P_{N-m+3}, P_{N-m+4}, \ldots, \hat{P}_{N+2} \rangle) \\
&\vdots
\end{aligned}$$

Discrete Fourier Transform

Def'n: For a sequence of observations, $\langle u_1, u_2, \ldots, u_n \rangle$, the *discrete Fourier transform* (DFT) is the sequence $\langle f_1, f_2, \ldots, f_n \rangle$, where

$$f_s = \frac{1}{\sqrt{n}} \sum_{r=1}^{n} u_r e^{i2\pi(r-1)(s-1)/n}.$$

The *inverse discrete Fourier transform* (IDFT) can be used to recover $\mathbf{u}$.

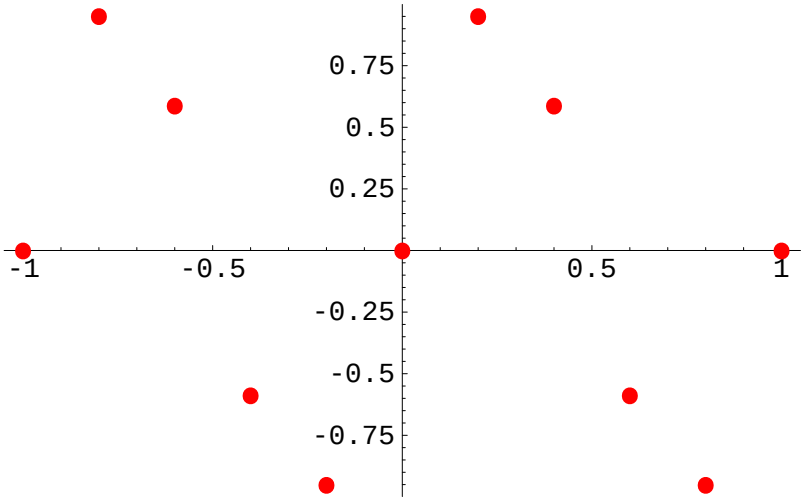
$$u_r = \frac{1}{\sqrt{n}} \sum_{s=1}^{n} f_s e^{-i2\pi(r-1)(s-1)/n}.$$

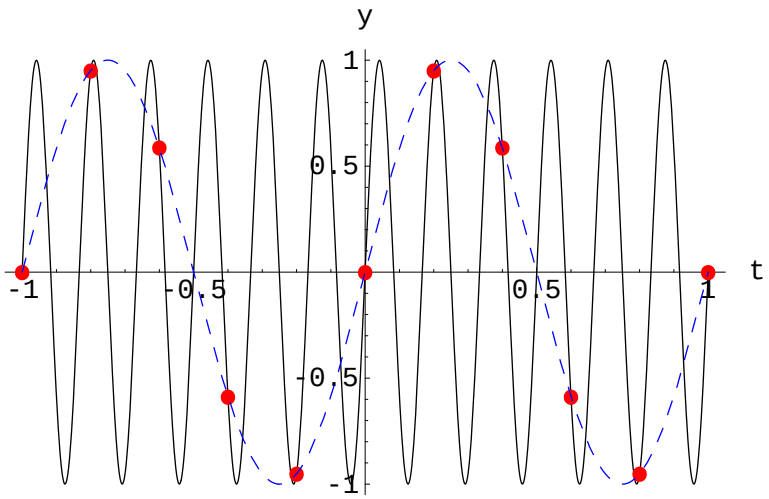
Properties of DFT and IDFT

- IDFT(DFT($\mathbf{u}$)) = $I_{n\times n}$
- If $\mathbf{u} \in \mathbb{R}^n$ then $f_1 \in \mathbb{R}$.
- If $\mathbf{u} \in \mathbb{R}^{2m}$ then $f_{m+1} \in \mathbb{R}$.
- If $\mathbf{u} \in \mathbb{R}^{2m}$ then $f_{m+1-j} = \overline{f_{m+1+j}}$ for $j = 1, \ldots, m-1$.
- If $\mathbf{u} \in \mathbb{R}^{2m-1}$ then $f_{2m-j} = \overline{f_{j+1}}$ for $j = 1, \ldots, m-1$.
- Almost the converses of the last two.

Sampling Frequency

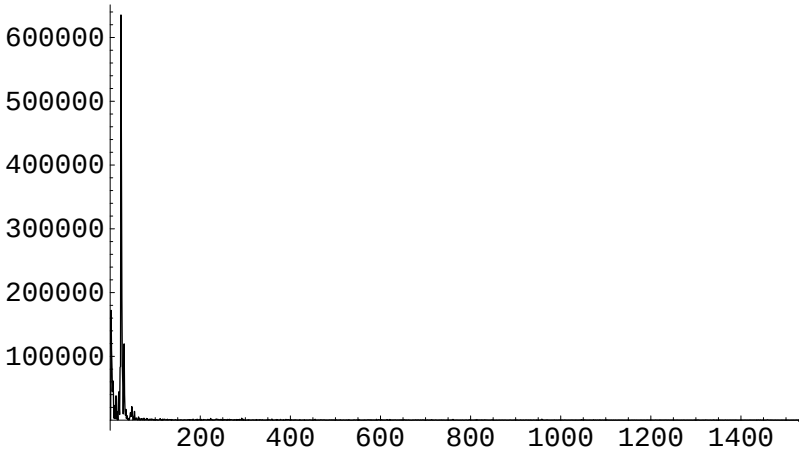
- Aliasing
- Nyquist frequency
- Two samples per period necessary to completely reconstruct signal from Fourier transform.

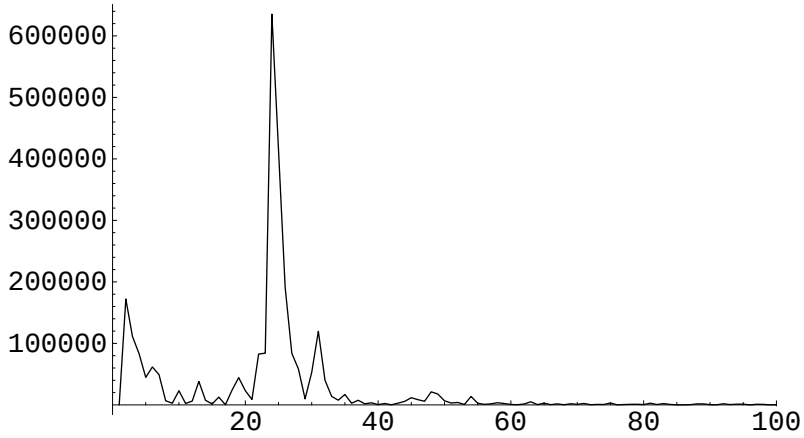




Power Spectrum

Let $\mathbf{f} = \mathrm{DFT}(\mathbf{u} - \overline{\mathbf{u}})$, then $\mathbf{p} = \langle p_1, p_2, \ldots p_n \rangle$ where $p_k = |f_k|^2$.
Spectrum of sunspot activity:





Interpolation

- Zero padding in the time domain interpolates in the frequency domain.
- Zero padding in the frequency domain interpolates in the time domain.

$$\langle f_1, \ldots, f_m, f_{m+1}, f_{m+2}, \ldots, f_{2m} \rangle \mapsto$$

$$\sqrt{k+1}\langle f_1, \ldots, f_m, \frac{f_{m+1}}{2}, \underbrace{0, \ldots, 0}_{2km-1 \text{ terms}}, \frac{f_{m+1}}{2}, f_{m+2}, \ldots, f_{2m} \rangle$$

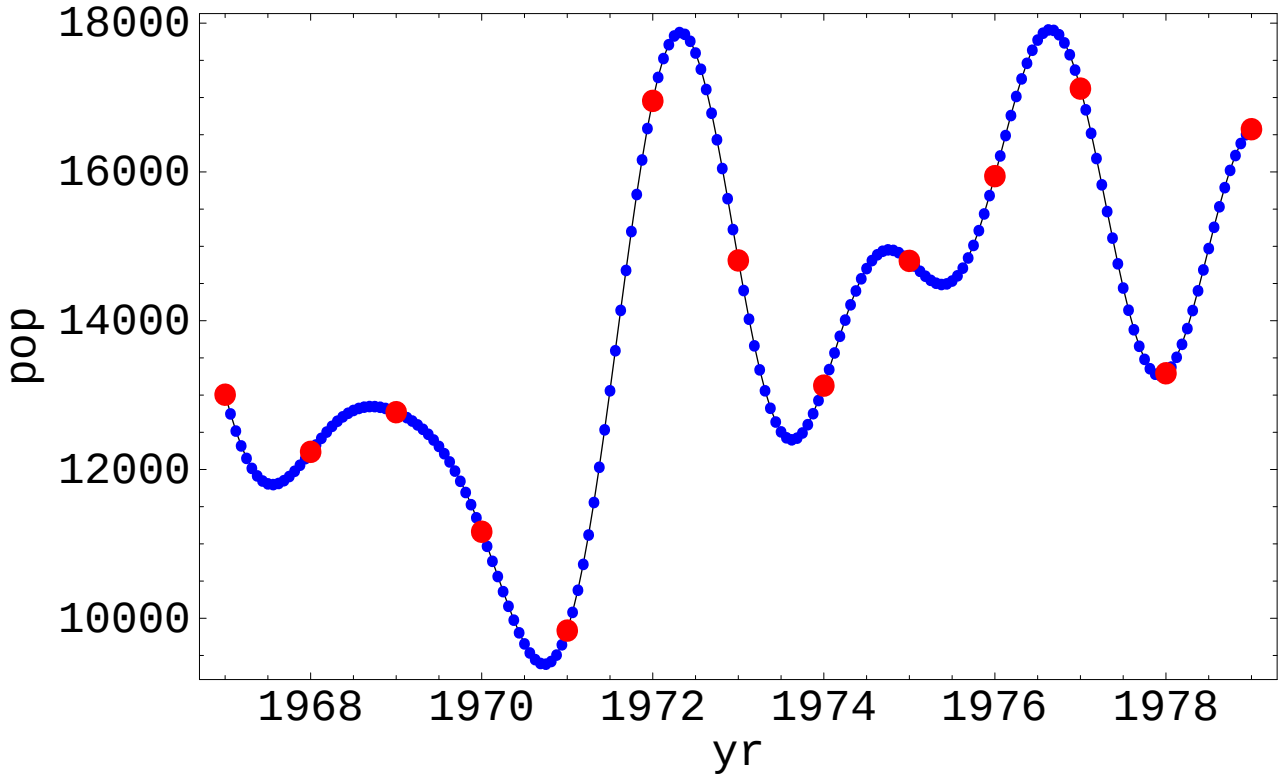
$$\langle f_1, \ldots, f_m, f_{m+1}, \ldots, f_{2m-1} \rangle \mapsto$$

$$\sqrt{k+1}\langle f_1, \ldots, f_m, \underbrace{0, \ldots, 0}_{(2m-1)k \text{ terms}}, f_{m+1}, \ldots, f_{2m-1} \rangle$$

Interpolation 2

If $\mathbf{p}$ is one of the zero-padded forms of $\mathbf{f}$ then

$$v_{(k+1)(r-1)+1} = \mathrm{IDFT}(\mathbf{p})_{(k+1)(r-1)+1} = u_r.$$



Filtering

- Remove high frequency noise
- Reduce data storage requirements
- Perform in parallel with interpolation

$$\mathbf{x}_i = L_3 \circ L_2 \circ L_1(\langle P_{i-w+1}, P_{i-w+2}, \ldots, P_i \rangle)$$

L_1 Fourier transform

L_2 Low pass filter $m/2$ frequencies

L_3 Inverse Fourier transform

Low Pass Filtering

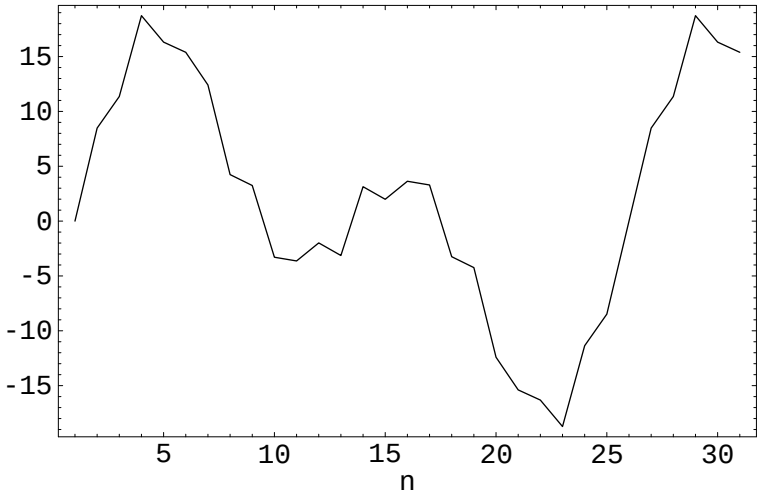
$$\langle f_1, f_2, \ldots, f_k, f_{k+1}, \ldots, f_{n-k+1}, f_{n-k+2}, \ldots, f_n \rangle *$$

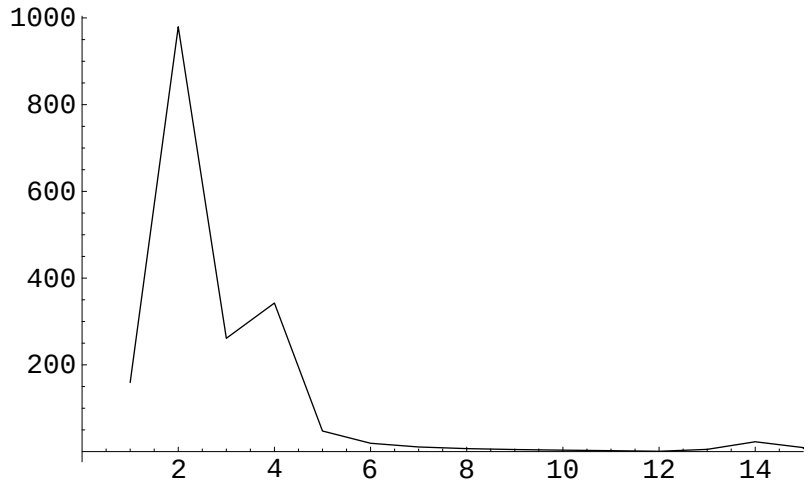
$$\langle 1, \underbrace{1, \ldots, 1, \frac{1}{2}}_{k \text{ terms}}, 0, \ldots, 0, \underbrace{\frac{1}{2}, 1, \ldots, 1}_{k \text{ terms}} \rangle$$

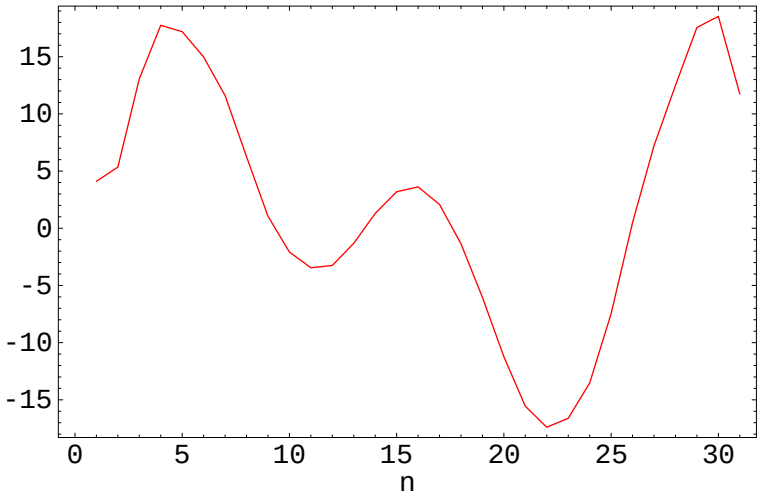
$$= \langle f_1, f_2, \ldots, f_k, \frac{f_{k+1}}{2}, 0, \ldots, 0, \frac{f_{n-k+1}}{2}, f_{n-k+2}, \ldots, f_n \rangle$$

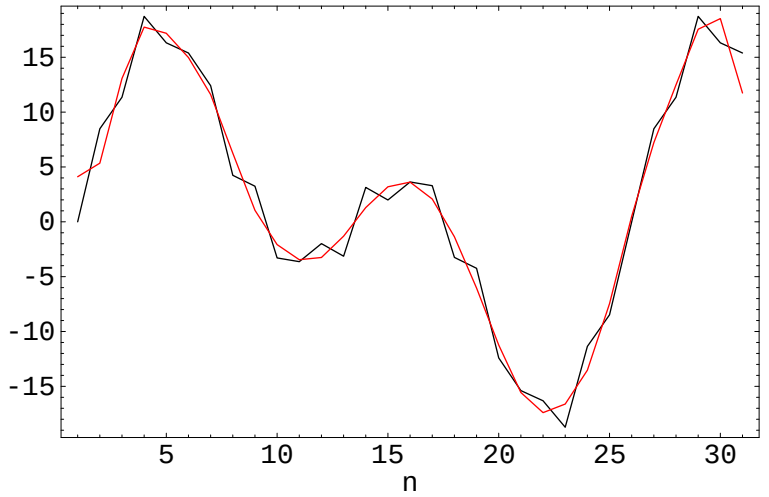
- Vector factors are of the same length and multiplied term-by-term.
- Conjugate symmetry is preserved.
- Can remove pairs of 0's in product to reduce data storage requirements.

Filtering Example









Prediction Algorithm

1. Create filtered embedding of sequence of observations
2. Find $k \geq 1$ nearest neighbors of $\mathbf{x}_N$
3. For neighbors $\{\mathbf{x}_{n_1}, \ldots, \mathbf{x}_{n_k}\}$ find $\{P_{n_1+1}, \ldots, P_{n_k+1}\}$
4. Approximate the map $\mathbf{x}_\alpha \stackrel{G}{\longmapsto} P_{\alpha+1}$
5. Evaluate $G(\mathbf{x}_N) = \hat{P}_{N+1}$

Types of Maps: Averaging

Direct:

$$\hat{P}_{N+1} = \frac{\sum_{i=1}^{k} w_i P_{n_i+1}}{\sum_{i=1}^{k} w_i}$$

Integrated:

$$\hat{P}_{N+1} = P_N + \frac{\sum_{i=1}^{k} w_i (P_{n_i+1} - P_{n_i})}{\sum_{i=1}^{k} w_i}$$

Weights depend on the distance between the neighboring vector and the query vector.

$$w_i = \left[1 - \left(\frac{d(\mathbf{x}_N, \mathbf{x}_{n_i})}{d(\mathbf{x}_N, \mathbf{x}_{n_k})} \right)^2 \right]^2$$

Another Type of Map: Linear

From Sauer (1994):

1. Let $\mathbf{c}$ be the center of mass of $\{\mathbf{x}_{n_1}, \ldots \mathbf{x}_{n_k}\}$
2. For some $l \leq m$ find the l-dimensional subspace of $\mathbb{R}^m$ containing $\mathbf{c}$ closest to the span of $\{\mathbf{x}_{n_1}, \ldots \mathbf{x}_{n_k}\}$

$$A = \begin{bmatrix} \mathbf{x}_{n_1} - \mathbf{c} \\ \vdots \\ \mathbf{x}_{n_k} - \mathbf{c} \end{bmatrix} = U^t D V$$

3. Project $\{\mathbf{x}_{n_1} - \mathbf{c}, \ldots, \mathbf{x}_{n_k} - \mathbf{c}\}$ onto $\mathbb{R}^l$
4. Find the affine map $G : \mathbb{R}^l \longrightarrow \mathbb{R}$ which best fits the data $\{(\Pi(\mathbf{x}_{n_1} - \mathbf{c}), P_{n_1+1}), \ldots, (\Pi(\mathbf{x}_{n_k} - \mathbf{c}), P_{n_k+1})\}$

$$\hat{P}_{N+1} \approx G(\Pi(\mathbf{x}_N - \mathbf{c}))$$

Singular Value Decomposition

If $A \in \mathbb{R}^{k \times m}$, then the *singular value decomposition* of A is

$$A = U^t D V,$$

where

- $U \in \mathbb{R}^{k \times k}$
- $V \in \mathbb{R}^{m \times m}$
- $D \in \mathbb{R}^{k \times m}$

$$D = \begin{pmatrix} \Sigma & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}.$$

- $\Sigma \in \mathbb{R}^{r \times r}$ is diagonal with $r < \min\{k, m\}$.

The first r rows of V are a basis for the rowspace of A.

Possible Estimators/Causality Window

- Map G predicts one step ahead.
- Define G_j which predicts j steps ahead.
- Define $w \in \mathbb{N}$ as the length of the causality window.

$$\hat{P}_{N+1} = \frac{1}{w} \sum_{j=1}^{w} G_j(\mathbf{x}_{N-j+1})$$

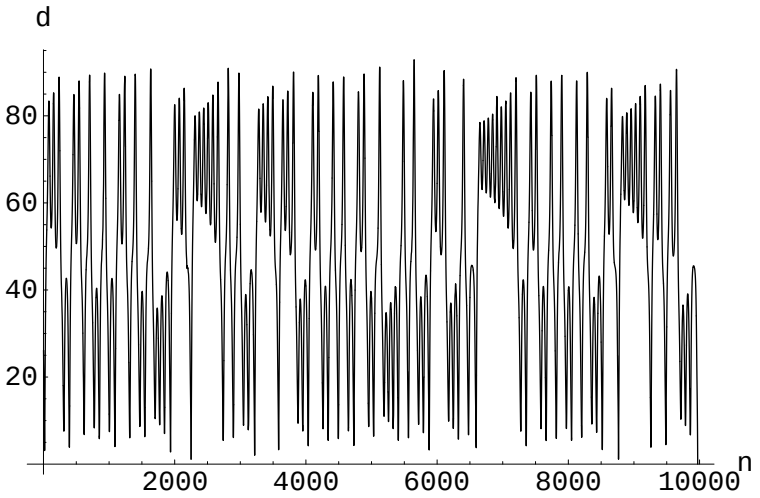
Mixes *direct* (single step) and *iterated* (multi-step) prediction.

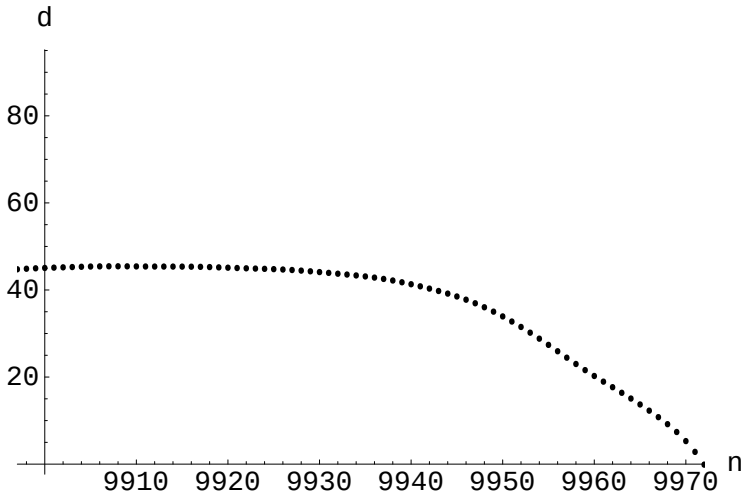
Nearest Neighbors

Metric:

$$d^2(\mathbf{x}_a, \mathbf{x}_b) = \sum_{i=0}^{m-1} \lambda^i (\mathbf{x}_{a,m-i} - \mathbf{x}_{b,m-i})^2$$

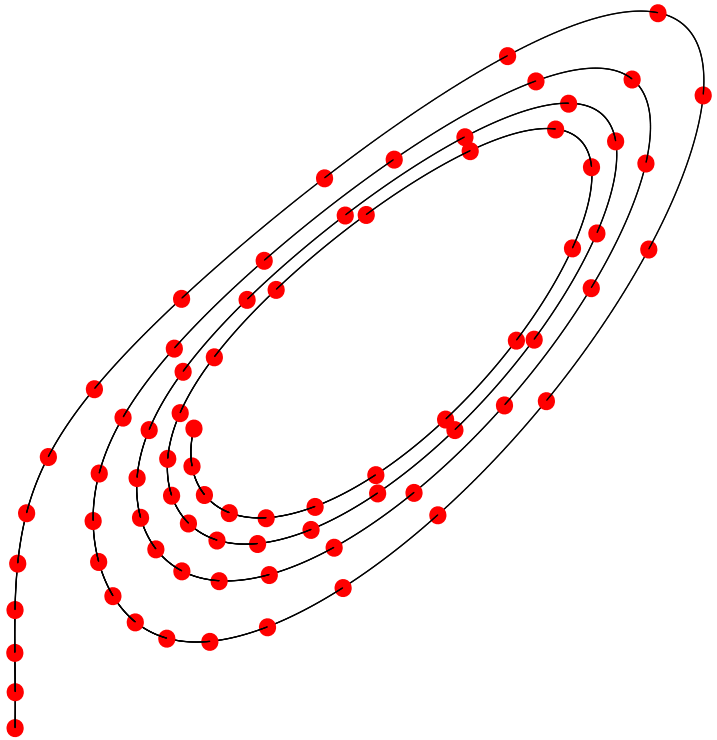
with $0 < \lambda \leq 1$.





Nearest Trajectories

High frequency sampling may produce nearest neighbors which lie on same trajectory.



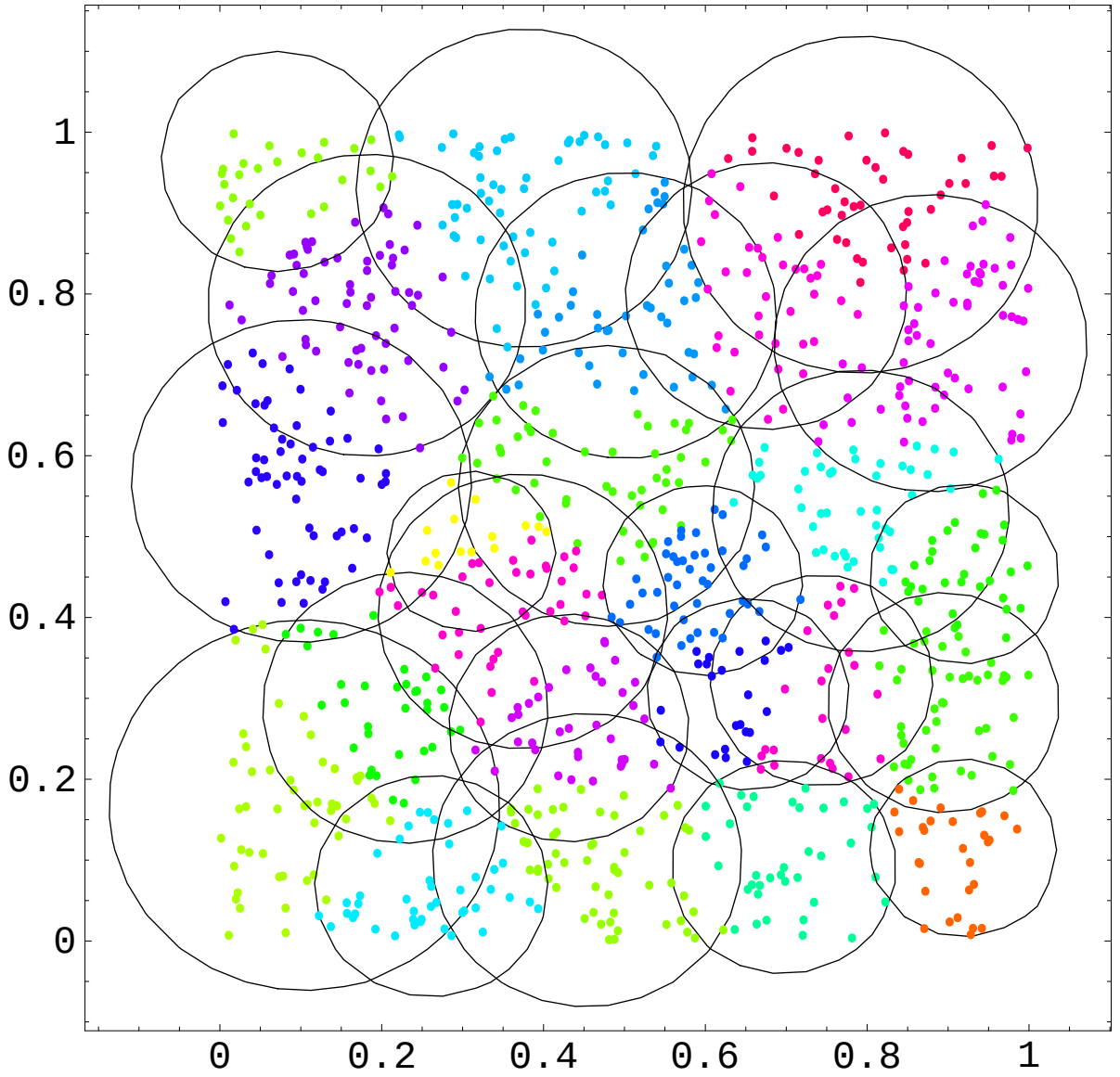
Tune-able Parameters

- Weight(s) used in metric
- Interpolation factor
- Embedding dimension
- Critical frequency of filter
- Length of causality window
- Number of nearest trajectory neighbors used
- If linear model used, number of dominant singular vectors

Nearest Neighbor Searching

- Brute force search $O(N)$, trivial complexity, requires calculating N distances.
- Binary search $O(\log N)$, requires preprocessing step to produce binary tree, nontrivial complexity.
  - Root cluster contains all embedding vectors
  - Find two cluster centers and divide into two sub-clusters
  - Recurse until terminal clusters have manageable number of embedding vectors

Terminal Clusters after Preprocessing



Data Structure for Cluster

Each cluster is represented by a data structure with attributes:

start: Index of first delay coordinate vector in cluster

end: Index of last delay coordinate vector in cluster

center: Index of center of cluster closest to the center of mass

radius: Maximum distance from center to any other vector in cluster

left: Left child sub-cluster (null for leaf clusters)

right: Right child sub-cluster (null for leaf clusters)

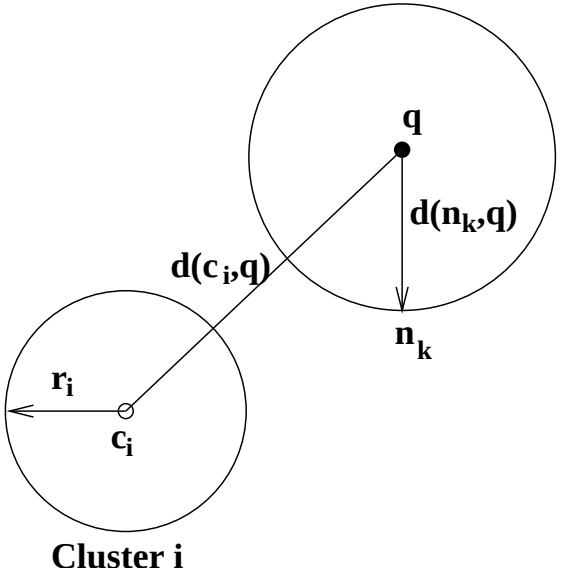
Searching for k Nearest Neighbors

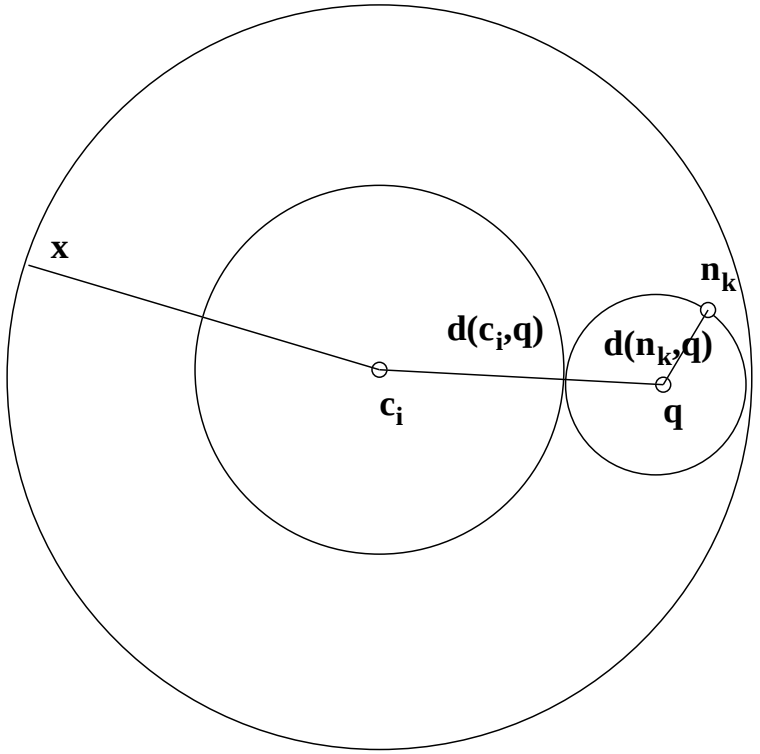
Given query vector $\mathbf{q}$ and root cluster

1. Process clusters according to priority set by lower bound on distance from cluster to $\mathbf{q}$.
2. Skip clusters for which $d(\mathbf{q}, \mathbf{c}) - r > d_k$.
3. For terminal clusters, skip delay coordinate vectors for which $|d(\mathbf{q}, \mathbf{c}) - d(\mathbf{x}, \mathbf{c})| > d_k$
4. Terminate search when lower bound for all remaining clusters exceeds d_k.

Implemented in C language and called from *Mathematica*.

Exclusion Rules





Speed Comparisons

Data: 49977 delay coordinate vectors of dimension 25.

Time to pre-process: 3.893465 seconds

Time to brute force search: 1.925136 seconds

Time to binary search: 0.001003 seconds

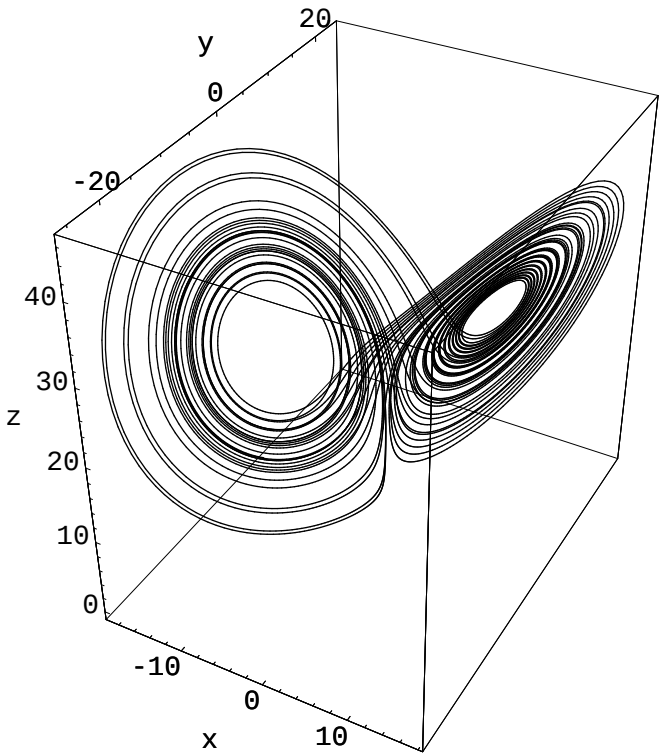
Trial Run

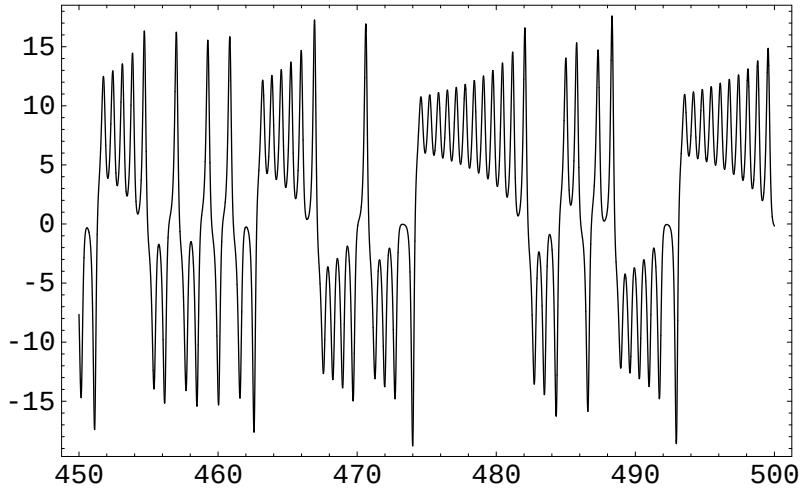
- Data set: sequence of x values from the Lorenz ODE.

$$\begin{aligned}
\frac{dx}{dt} &= 10(y-x) \\
\frac{dy}{dt} &= 28x - y - xz \\
\frac{dz}{dt} &= -\frac{8}{3}z + xy \\
x(0) &= 1/10 \\
y(0) &= -1/5 \\
z(0) &= 3/10
\end{aligned}$$

- 10000 data points generated with $\Delta t = 0.05$.

Lorenz Attractor

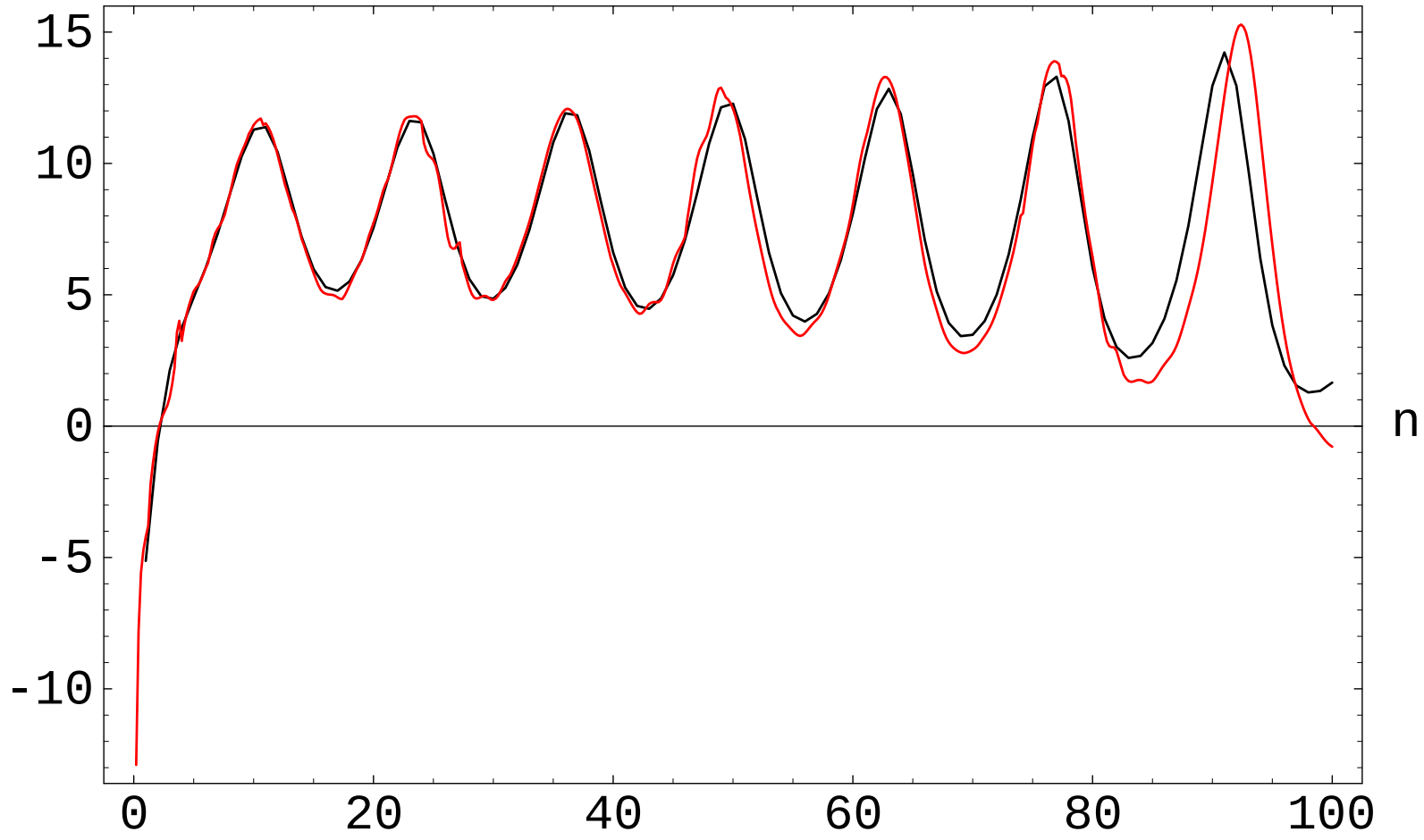




Algorithm Parameters

- Model uses only $\{x_{500}, x_{501}, \ldots, x_{2499}\}$.
- No noise in data, thus no filtering necessary.
- Interpolation factor of 5.
- Input window of length 32, output delay coordinate vector of length 16.
- Linear model with 2 dominant singular vectors.
- Causality window of length 16.
- Predict next 500 interpolated points in the orbit of the x coordinate of the Lorenz system (next 100 un-interpolated points).

Results



Legend: true Lorenz output, predicted Lorenz output

Future

- Have algorithm optimize model parameters
- Apply to different data sets