AMATH 563 Homework 2

Christopher Liu

May 6, 2020

Abstract

We explore 3 methods, dynamical mode decomposition, time-delay embedding and sparse regression to model 2 data sets. Using the KL divergence, the Akaike Information Criterion and the Bayesian Information criterion, we are then able to determine the accuracy of these models in modelling the data sets. Due to the poor quality and small size of the data sets, the produced models were highly inaccurate.

1 Introduction and Overview

The study of dynamical systems is motivated by their ability to provide a mathematical framework to describe real-world phenomena. Dynamical systems not only allow us to characterize behavior but also predict future behavior of a system [1]. In this assignment, we will be using data-driven techniques to attempt to identify, characterize and analyze dynamical systems. The methods used in this assignment are dynamic mode decomposition (DMD), time-delay DMD and sparse regression. We will apply these methods to 2 data sets, a historical data set concerning Canadian lynx and snowshoe hare populations and a movie of a Belousov-Zhabotinsky chemical oscillator. Finally, we will apply methods in model selection such as KL divergence, the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) to compare the models created by the different methods.

2 Theoretical Background

2.1 Dynamic Mode Decomposition

Dynamic mode decomposition (DMD) was deve;oped by Schmid to identify spatio-temporal coherent structures from high dimensional data and is based on principal orthogonal decomposition [1]. To carry out the DMD, we first need to arrange our measurements of our system, $x(t_i)$ into two matrices ,

$$X = \begin{bmatrix} | & | & \dots & | \\ x(t_0) & x(t_1) & \dots & x(t_{m-1}) \\ | & | & \dots & | \end{bmatrix}, \; X' = \begin{bmatrix} | & | & \dots & | \\ x(t_1) & x(t_2) & \dots & x(t_m) \\ | & | & \dots & | \end{bmatrix} \tag{1}$$

where $t_i = t_{i-1} + \Delta t$.The DMD algorithm will then, using the SVD, discover the leading spectral decomposition of the best-fit linear operator, A such that [1]

$$X' \approx AX$$

One of the main applications of DMD is the ability to expand the system state in terms of a data-drive spectral decomposition [1].

$$x(t_i) = \Phi \Lambda^t b \tag{2}$$

where Φ are the DMD modes or eigenvectors of A, Λ are the eigenvalues of A and b is the mode amplitudes given by $b = \Phi^\dagger x(t_0)$ where $\Phi^\dagger$ is the pseudo-inverse of Φ. This expansion of the system state allows us to create a forecast future behavior of the system.

2.2 Time-Delay Embedding

Given a time series of a single measurement $x(t)$ of a system, it is not always clear how many variables are present. Time-delay embedding gives us a way to recover latent or hidden variables in the system by computing the SVD of the Hankel matrix, H.

$$H = \begin{bmatrix} x(t_0) & x(t_1) & \dots & x(t_{m_c}) \\ x(t_1) & x(t_2) & \dots & x(t_{m_c}+1) \\ \vdots & \vdots & \ddots & \vdots \\ x(t_{m_o}) & x(t_{m_o}+1) & \dots & x(t_m) \end{bmatrix} = U\Sigma V^* \tag{3}$$

By analyzing the singular values of H, we can estimate the intrinsic dynamical rank of the system. Furthermore, we can treat the columns of our Hankel matrix as "observations of the system" and use that to create X and X' for the DMD and create a future forecast of the system.

2.3 Sparse Regression

Sparse regression, which we utilized in a previous assignment, can also be employed to discover dynamical systems from data [1]. By leveraging the fact that most dynamical systems are of the form,

$$\frac{dx}{dt} = f(x) \tag{4}$$

and the dynamics, f, only have a few dominant terms, we can approximate the system with a linear model,

$$G\Xi = \begin{bmatrix} | & | & \dots & | \\ g_1(x(t)) & g_2(x(t)) & \dots & g_m(x(t)) \\ | & | & \dots & | \end{bmatrix} \begin{bmatrix} | \\ \Xi \\ | \end{bmatrix} \approx \dot{x}(t) \tag{5}$$

where G is our library of nonlinear functions g_m and Ξ is a vector of coefficients determining the active terms in G and contains the fewest non-zero terms as possible. We can then use sparse regression such as Lasso to solve for Ξ and determine the active terms of our system. This system is typically over-determined as we have many more measurements than candidate functions.

2.4 KL Divergence, AIC, BIC

To determine how good our models are, we will use the KL diveregence, the Akaike Information Criterion (AIC) and the Bayesian Iformation Criterion (BIC). Due to small size of our data set, cross validation will not be employed. The KL divergence measures the information lost when a model, g, is used to represent, f, the truth or measurements. The KL divergence of two probability distributions is given as,

$$\mathrm{KL}(f, g) = \sum_{i=1}^{k} P \cdot \log\left(\frac{P_i}{Q_i}\right) \tag{6}$$

where P is the probability distribution of the "truth" and Q is the probability distribution of the approximating model [2]. Note that when $P = Q$, $\log(1) = 0$ and $I(f, g) = 0$ so no information is lost [1].

The AIC and BIC are defined as the following,

$$AIC = 2K - 2\log[\mathcal{L}(\hat{\mu}|x)] \tag{7}$$

$$BIC = \log(n)K - 2\log[\mathcal{L}(\hat{\mu}|x)] \tag{8}$$

where K is the number of parameters used in the model, $\hat{\mu}$ is an estimate of the best parameters used, n is the number of data points and x are the measurements. $\log \mathcal{L}$ is the log-likelihood function at its maximum point given by,

$$\log \mathcal{L} \approx \frac{-n}{2} \log\left(\frac{\sum_i (f(x_i) - g(x_i))^2}{n}\right)$$

Both the AIC and BIC penalize models for having a greater number of parameters. BIC differs from AIC by also penalizing models that incorporate more data [1].

3 Algorithm Implementation and Development

3.1 DMD and Time-Delay DMD

For the Hare and Lynx dataset, only 30 data points were provided. The DMD and time-delay DMD struggled to produce good forecasts when such a small data set was used. Spline interpolation was used to create a data set with a time-step of 0.2 years as opposed to 2. The matrices X and X' were made from stacking the hare and lynx population data into 1 matrix. The forecasting was carried out by performing the DMD on half of the data and using the spectral expansion to predict future states. This could then be compared to the measurements.The Hankel matrix was constructed also by combining both the hare and lynx data with 15 time steps. The forecast, similar to regular DMD, was carried out by creating a Hankel matrix with half the hare and lynx data. For the BZ data, to minimize computational time, the data set was truncated to a single row.

3.2 Sparse Regression

Before carrying out sparse regression, we first fit our data to the Lotka-Volterra (LV) equations using the psuedo-inverse as it is commonly used to model such phenomenon. The LV equations are

$$\dot{x} = (b - py)x, \;\; \dot{y} = (rx - d)y$$

where b, p, r, d are positive coefficients. For sparse regression, our library functions were chosen to be polynomials of degree 1 to 3, the product of the 2 data sets and sine. Sparse regression was carried out using Lasso with $\alpha = 0.3$ and the active terms were determined by inspection. The derivative of our population data was calculated using 2nd-order accurate centered finite difference. Once our active terms were determined, we numerically integrated the resulting ODEs to create our model. Non-interpolated data gave the best result for these methods.

3.3 KL Divergence, AIC, BIC

To compute the KL divergence, we determined the corresponding probability density functions of the models by creating histograms with the same uniform bins and were normalized. In order to avoid dividing by 0, all of the probability density functions were shifted upwards by 0.01. For all 3 metrics, the DMD and Time-Delay DMD models were sub-sampled as interpolated data was used. AIC and BIC were calculated for the 3 models with the lowest KL divergence

4 Computational Results

4.1 Canadian Lynx and Snowshoe Hare Populations

Upon inspection of the data set, we observe that the measurements are very sparse in time and there are not many data points. Furtheremore, it is very jagged and there does not seem to be a discernable pattern. Even with interpolation the DMD, as shown in Figure 1, has great difficulty determining the correct modes. It does not even do a good job at fitting the inputted data (the left half). Nonetheless, it is still able to determine that there is some oscillatory behavior in this system.

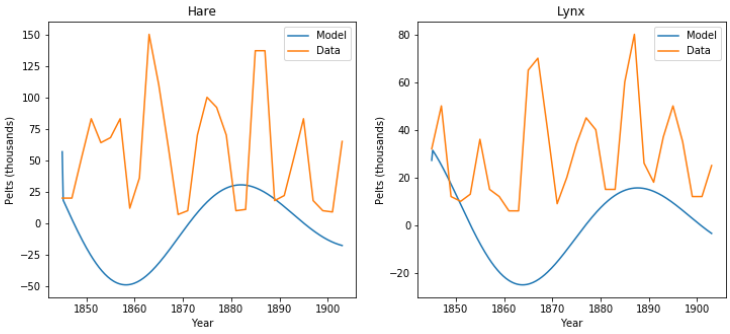


Figure 1: DMD of the Interpolated Hare and Lynx Data

Using time-delay embedding (Figure 2), we can estimate that there are 5-7 latent variables in this system. There is one dominant value that accounts for almost 50% of the system with a handful of a increasingly smaller non-zero values. The poor quality of the data makes it difficult to determine exactly how many variables there are in this system.

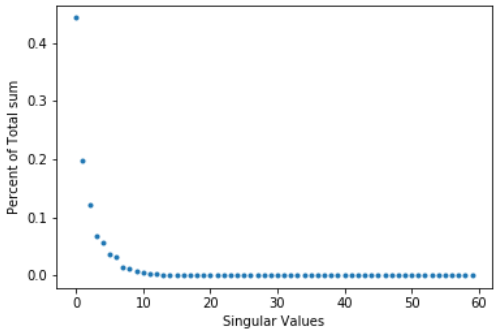


Figure 2: Singular values of the Hankel matrix

Figure 3 shows the models created by the time-delay embedding. This does a significantly better in not only fitting the inputted data points (points before 1874) but also makes reasonable short-term predictions up until around 1876. This is likely due to the poor quality of the data and the exponential term, Λ^t, in the model. This highlights one of the pitfalls of the DMD in that it can struggle to produce accurate forecasts in the distant future as the model can explode as t increases due to the exponential term. Two plots for each model are shown as the first point in each model has a large error despite the rest of the model doing an okay job at fitting the data.

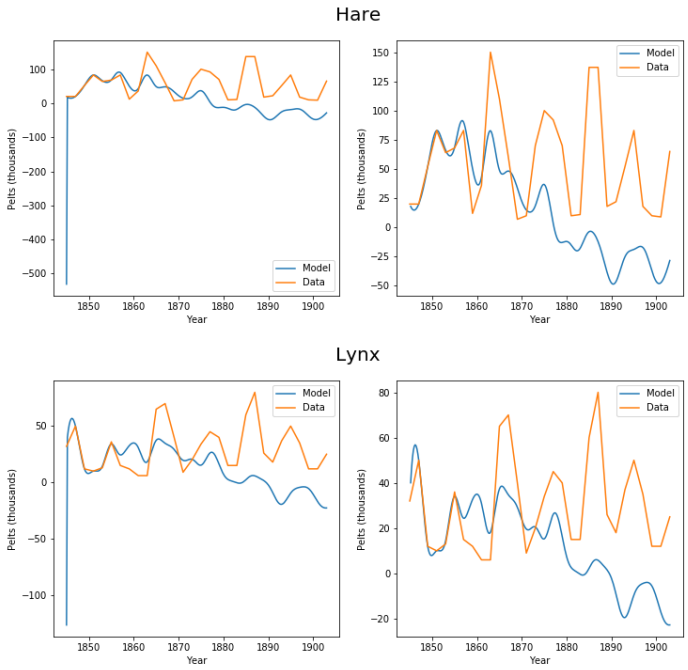


Figure 3: Time-Delay DMD, the plots on the right omit the first point

Next, we fitted the data to the Lotka-Volterra system of equations. The coefficients were found to be $b = 0.156, p = 0.00385, d = 0.205, r = 0.00237$. Regression also seems to have a difficult time capturing the dynamics of the system. This is most likely due to the poor quality of the data. It does, however, appear to be more realistic than the DMD models despite having a larger maximum error, as the model is always positive. Similar to regular DMD, the LV equations also seem to extract the oscillatory behavior of the system with the peak of the model somewhat lining up with a peak in the data.

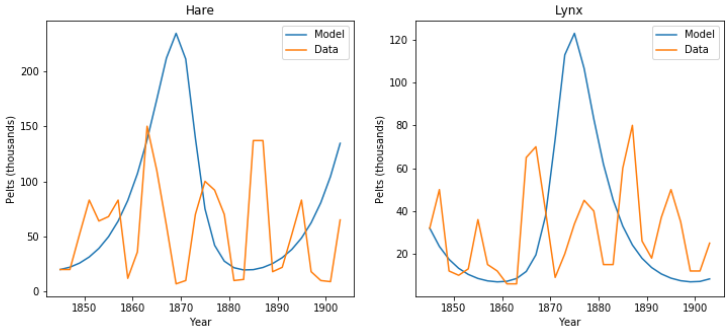


Figure 4: Lotka-Volterra Models

The sparse regression also has difficulty capturing the dynamics of the system. The hare model has linear terms and a sine term while the lynx model has linear and quadratic terms. The hare model does not capture any oscillatory behavior while the lynx model seems to capture the oscillatory behavior well but is shifted upwards. In terms of actually capturing the behavior and not necessarily accurately predicting the population, the lynx model seems to have done the best job so far. We can also note that both models seem to have large errors for some of the earlier points.

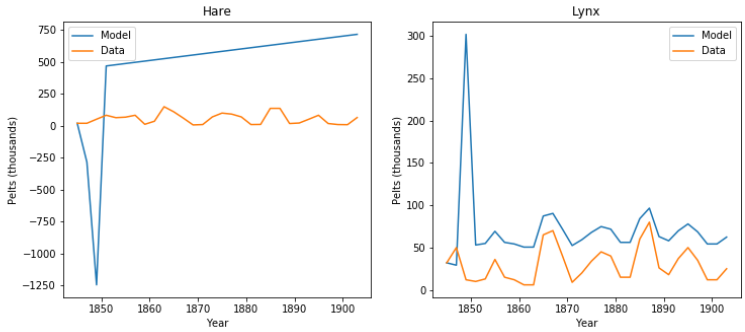


Figure 5: Model from Sparse Regression

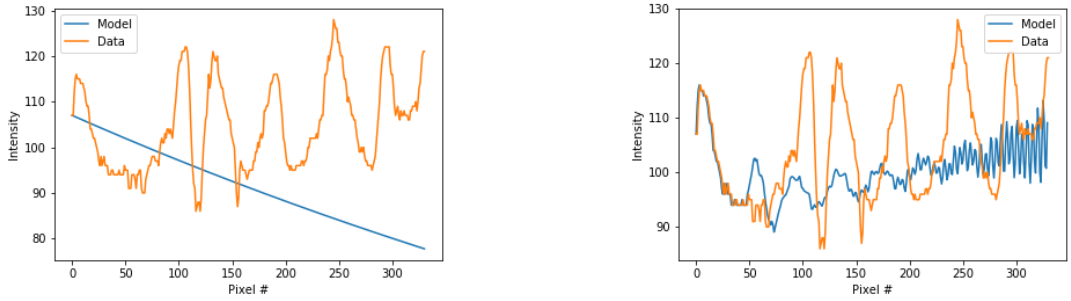
The metrics used to determine which model is best are shown in Table 1. The time-delay model performed the best as expected, followed by L-V and regular DMD. Despite the sparse regression of the lynx data having consistent peaks and dips with the data, it was the most inaccurate model and had the worst KL divergence. The regular DMD had the worst AIC and BIC scores as it used significantly more parameters than the other models. On the other hand, LV had the best AIC and BIC scores as it only had 2 parameters.

Method	KL (Hare)	KL (Lynx)	AIC (Hare)	AIC (Lynx)	BIC (Hare)	BIC(Lynx)
DMD	2.32	1.86	842.8	798.3	1249.	1205.
T-DMD	1.18	0.78	406.3	343.9	490.4	428.0
L-V	1.75	3.84	267.1	222.6	270.0	225.4
Sparse	2.79	3.46	-	-	-	-

Table 1: KL Divergence, AIC, BIC of the different models

4.2 Belousov-Zhabotinsky Chemical Oscillator

The DMD and time-delay DMD was also applied to the Belousov-Zhabotinsky Chemical Oscillator (BZ oscillator) data set. The inputted data was selected from a row of a section of the data that had the cleanest oscillations. Even then, the data is incredibly noisy and the oscillations are extremely non-uniform (Figure 6). Similar to the previous data set, the regular DMD produces a highly inaccurate model. It also fails to capture any oscillatory behavior. The time-delay model is able to capture oscillatory behavior and does a decent job at capturing the behavior of the oscillations but once again fails for more distant forecasts. The behavior towards the end of the model suggests that it has difficulty discerning the noise from the actual dynamics of the system.



(a) Regular DMD

(b) Time-Delay DMD

Figure 6: DMD and Time-Delay DMD Forecast Models for the BZ Oscillator

5 Summary and Conclusions

We have attempted to apply a handful of methods to characterize the behavior of two data sets. In doing so, we have shown how these methods can fail when the data sets are small and of poor quality and thus have highlighted the difficulty in creating and analyzing dynamical systems from real-world data. Despite the poor accuracy of the models we were still able to produce oscillatory models that, in the case of sparse regression, were able to characterize the behavior of the system relatively well. Ultimately, we have shown that these methods, despite having the potential to be very powerful, are only as good as the data provided.

References

[1] Jose Nathan Kutz and Steven Brunton. *Data-driven science & engineering machine learning, dynamical systems, and control.* Cambridge University Press, 2019.

[2] Kenneth P. Burnham and David R. Anderson. *Model selection and multimodel inference: a practical information-theoretic approach.* Springer, 2010.

Appendix A Python Functions

- `numpy.linspace(start,stop,number)` Returns evenly spaced numbers of a specified interval
- `interp1d(x,y, kind="cubic")` Spline interpolation of a a 1-d function of third order
- `numpy.linalg.svd(A)` Computes the SVD of a given matrix
- `numpy.linalg.eig(A)` Computes the eigenvalue decomposition of a given matrix
- `numpy.linalg.diag(x)` Returns a diagonal matrix with entries equal to vector x
- `numpy.linalg.solve(A,b)` Solves the system $Ax = B$ for x
- `numpy.stack` Stacks two matrices or vectors along a given axis
- `numpy.linalg.pinv(A)` Returns the Moore-Penrose pseudoinverse of A
- `integrate.odeint(ode,X0,time)` Integrates a system of ordinary differential equations for given initial conditions over a specified range.
- `numpy.histogram(x,bins=bin_edges)` Creates a histogram of x with the specified bin edges
- `numpy.trapz(x,t)` Integrates along the given axis using the composite trapezoidal rule.

Appendix B Python Code

https://github.com/chrismhl/AMATH563/blob/master/Homework2/AMATH563$_{H}omework2_{C}ode.ipynb$