Population Modeling

Ron Wilson

May 6, 2020

Abstract

Methods of model discovery & model selection are explored in this write-up. Specifically, the models include dynamic mode decomposition (DMD), time-delayed DMD, and regression models such as pseudo-inverse method, lasso regression, & ridge regression.

1 Introduction and Overview

There are two different parts to this write-up. The first part analyzes the Canadian lynx and snowshoe hare populations from 1845 to 1903. In part two, we will look at a snippet from a Belousov-Zhabotinsky chemical oscillator movie.

The goal for part one is to model the population sizes using DMD & regression methods, and determine which model gave the best fit to the original data. For part two, we want to see if we can accurately model the chemical oscillator using dynamic mode decomposition.

2 Theoretical Background

Here I will provide some of the necessary background information needed for the algorithms explored in this write-up.

2.1 Singular Value Decomposition

A significant piece of the DMD algorithm is to compute the singular value decomposition (SVD) of the data matrix A. The SVD of this m-by-n matrix is defined as:

$$A = U\Sigma V^* \tag{1}$$

The SVD is computed in MATLAB using the function `[U,S,V] = svd(A,'econ')` (see Appendix A). Note that this is using the economy-size SVD. This is the same singular value decomposition as in equation 1, except that this reduced SVD removes the extra rows or columns of zeros from the matrix Σ, and the corresponding columns in U & V that they would be multiplied by. The economy SVD is used to improve the execution time of the code without losing the accuracy of the SVD.

2.2 Lotka-Volterra Model

For populations models such as the one in part one, it is common to use empirical predator-prey models. One such model is the Lotka-Volterra model, which describes the following system of equations:

$$\dot{x} = (b - py)x, \;\; \dot{y} = (rx - d)y \tag{2}$$

where x, y are the population sizes of the prey and the predators, respectively. The parameters b, p, r, d describe the interaction between the two species. In the case of the hares & lynxes, we can fit values for the interaction parameters to find a best fit model for their population sizes.

2.3 Model Selection

We will use two different methods to determine the best model; KL divergence and AIC\BIC scores.

2.3.1 Kullback-Leibler Divergence

The model with the lowest Kullback-Leibler (KL) divergence compared to the original data is the most accurate. KL divergence can be calculated by the following equation:

$$\text{KL} = \int_X p \, \log(\frac{p}{q}) \; d\mu \tag{3}$$

where P, Q are probability measures over set X such that $p = \frac{dP}{d\mu}$ and $q = \frac{dQ}{d\mu}$ are continuous functions with respect to μ. Then, equation 3 measures the KL divergence from Q to P. In the context of this problem, we can find probability distributions for a model, Q, and a probability distribution for the original data, P, to find the KL divergence for each model.

2.3.2 AIC & BIC Scores

AIC\BIC scores measure the information lost by a particular model. Because of this, the model with the lowest score retained the most information and is the most optimal fit to the truth model. This is different to KL divergence in that it measures information lost overall rather than just how close together the two models were. The Akaike information criterion (AIC) is computed by the following:

$$\text{AIC} = 2k - 2ln(\hat{L}) \tag{4}$$

where k is the number of estimated parameters of the model and $ln(\hat{L})$ is the log-likelihood function calculated from every data point in matrix A:

$$ln(\hat{L}) = \sum ln(A) \tag{5}$$

Using the same number of estimated parameters and log-likelihood function, we can calculate the Bayesian information criterion (BIC):

$$\text{BIC} = k \; ln(n) - 2ln(\hat{L}) \tag{6}$$

where n is the sample size of the data set A.

3 Algorithm Implementation and Development

In this section I will detail the algorithms used on the two data sets. For the MATLAB codes corresponding to these algorithms, see Appendix B.

3.1 Part 1: Hare & Lynx Populations

Before we can analyze the data set (which contains 30 data points for each population over 58 years), we need to interpolate the data to generate the sample "truth" model. This is done using the MATLAB function `spline(x,y,xq)` (see Appendix A). The interpolated data can be seen in figure 1 where the dots represent the original data points. Now that we have the truth model, we can construct various best fit models and find the most accurate model.

3.1.1 Dynamic Mode Decomposition (DMD)

After combining the interpolated data sets for the hare & lynx populations into a single data matrix X, we can perform dynamic mode decomposition (DMD).

1. First, we split the data set X_1^n into two sets; $X1 = X_1^{n-1}$ and $X2 = X_2^n$.
2. Next, we compute the economy-size SVD of the matrix $X1$ such that $X1 = U\Sigma V^T$.

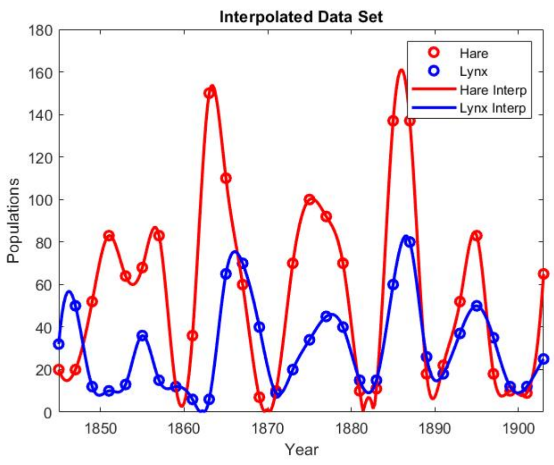

Figure 1: The interpolated data set

   (a) Because the data set X only has two rows (one for hare populations and one for lynx populations), there are only two singular values.
   (b) We can then perform a rank $r = 2$ truncation on the SVD components such that $U = U(:, 1:2)$, $\Sigma = \Sigma(1:2, 1:2)$, and $V^T = V^T(:, 1:2)$.

3. Using the truncated SVD components we can compute a new matrix $\widetilde{X} = U^T * X2 * V^T * \Sigma^{-1}$. Notice that this matrix is computed with data matrix $X2$.
4. Now, we find the eigenvalues and eigenvectors of $\widetilde{X}$ using the MATLAB function `eig(`$\widetilde{X}$`)` (see Appendix A).
5. Using the eigenvalues and eigenvectors we can find the DMD modes for the data set.
   (a) First, using the eigenvectors W, we find $\Phi = X2 * V^T * \Sigma^{-1} * W$, get the initial conditions $u0 = X(:, 1)$, and solve the $\Phi y0 = u0$ problem for $y0$ using the backslash command in MATLAB.
   (b) Then, using the eigenvalues D, we can find $\Omega = \frac{ln(D)}{dx}$ where dx is the same step size that was used to interpolate the data.
   (c) With $y0$ and Ω, we can now find the DMD modes by looping through the length of data set (x_space) and calculating the following for each iteration i:

$$\text{modes}(:, i) = y0 * e^{\Omega * \text{x\_space}(i)} \tag{7}$$

   Multiplying modes by Φ will produce the DMD modes for the two population sizes. We can then split it into the hare population (first row of DMD modes) and the lynx population (second row of DMD modes).
6. Finally, with the DMD modes for each species, we can plot the real parts of the modes versus the truth model to visualize the DMD model.

3.1.2 Time-Delay DMD

The algorithm for time-delay DMD is very similar to the algorithm for DMD. However, we want to delay the data matrix X in time before performing DMD. To do this, we need to construct a Hankel matrix. For this problem I used 25 embeddings such that the Hankel matrix is defined as the following:

$$H = \begin{bmatrix} \text{hare}(1:\text{end}-24) \\ \text{lynx}(1:\text{end}-24) \\ \text{hare}(2:\text{end}-23) \\ \text{lynx}(2:\text{end}-23) \\ \cdots \\ \cdots \\ \text{hare}(24:\text{end}-1) \\ \text{lynx}(24:\text{end}-1) \\ \text{hare}(25:\text{end}) \\ \text{lynx}(25:\text{end}) \end{bmatrix} \tag{8}$$

The Hankel matrix takes initial data measurements for the hare & lynx populations (in the first two rows) and delays it in time over 25 embeddings (the next 48 rows). So, H is a matrix of size 50-by-length of x_space. I can then split H into $X1$ and $X2$ as in the DMD algorithm where $X1 = H(:, 1 : \text{end} - 1)$ and $X2 = H(:, 2 : \text{end})$. From here, the time-delay DMD algorithm is the same as the DMD algorithm.

However, it is important to note that there are now 50 singular values from SVD. In figure 2, I plotted the singular value spectrum for the time-delay DMD method. The first 6 singular values makes up more than 95% of the total variance, so I chose an $r = 6$ rank truncation on the SVD components.

3.1.3 Best Fit Model Discovery

There are three additional models that I constructed using model discovery. These are best fit models using the Lotka-Volterra system of equations from equation 2. Here is the algorithm that I used to find these models:

1. First, I used an initial guess for the Lotka-Volterra interaction parameters b, p, r, & d to solve equation 2 using `ode45` in MATLAB with the data matrix X. I separated the results from `ode45` into 2 different matrices, one representing the hare population (Xh) and one representing the lynx population (Xl).

2. Next, for each population, I calculated the derivative using the central difference formula where x is the population size at each data point and h is the step size of the data set:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} \tag{9}$$

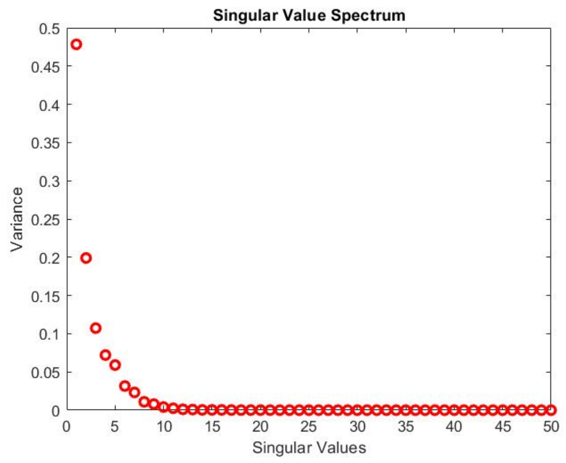

Figure 2: The singular value spectrum from the Hankel matrix

3. I then built a library of functions A to fit the population models with. The matrix A consists of the polynomials Xh, Xl, & Xh*Xl. For sparse regression models I also included polynomials up to degree 3 and the trigonometry functions sin & cos to find which functions are most important in finding the best fit model.

4. Now that we have the matrix A, we can solve the $Ax = b$ problem. Here, b is the matrix found from the central difference formula for each population. I solved this problem three different ways to get three different models. The first method is the pseudo-inverse method where I calculated the inverse of A in MATLAB with the function `pinv(A)` and multiplied by b. For sparse regression, I chose to use lasso & ridge regression with $\lambda = 0.025$. I did this with the MATLAB functions `lasso(A,b,'Lambda',0.025)` & `ridge(b,A,0.025)`. See Appendix A for the MATLAB functions referenced here.

5. Plotting a bar graph of the solutions x shows the coefficients of the individual functions within the library matrix A. Thus, we can see which functions were the most important for sparse regression. Then, we can apply these coefficients to the functions with the original data set and add all of these terms together to get a best fit curve for each population.

6. Finally, I plotted all three of these models and visually observed whether the model fit the data well. I would then change my initial guess for the interaction parameters of the Lotka-Volterra model and re-run the code until I got the best possible fit for each of the three models.

3.1.4 Model Selection

I now have five different models for the hare & lynx populations; DMD, time-delayed DMD, pseudo-inverse (pinv) method, lasso regression, & ridge regression. Next, I need to evaluate the five models to try and find the most accurate model.

1. First, I found the KL divergence of each of the five models evaluated against the truth model. To do this, I computed equation 3 using P as the truth model and Q as the five estimated models. In order to solve this equation, I needed to get a probability density function (PDF) for each of the models, which I found in MATLAB with the function `hist`. Also, I normalized the functions found from `hist` by their integral over x_space in order to get the normalized PDF's. With this, I can compute the integrand of equation 3 with the MATLAB function `trapz`.

2. After finding the KL divergence, I took the three best models (the models with the lowest KL divergence) and computed their AIC/BIC scores. I used equations 4 & 6 to do this. For the log-likelihood function, I calculated equation 5 using the PDF's of each model as the matrix A. Of the three remaining models, the one with the best (lowest) AIC/BIC scores will be selected as the best model to the populations of hares & lynxes.

3.2 Part 2: BZ Chemical Oscillator

For the BZ data set, I only generated two models (DMD and time-delayed DMD). The algorithm for finding these models is very similar to sections 3.1.1 & 3.1.2. The only difference here is that the data set is 3-dimensional, measuring x and y in time. Because of this, I selected four specific frames in time and performed DMD on the individual frames.

4 Computational Results

In this section I will discuss the results of the code. The code can be found in Appendix B (part 1 used `Hare_and_Lynx.m` with `rhs_dyn.m` as the Lotka-Volterra system of equations, and part 2 used `BZ.m`).

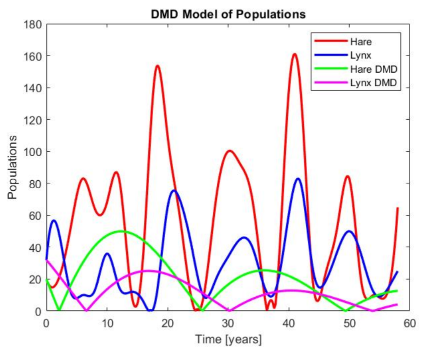

Figure 3: DMD model

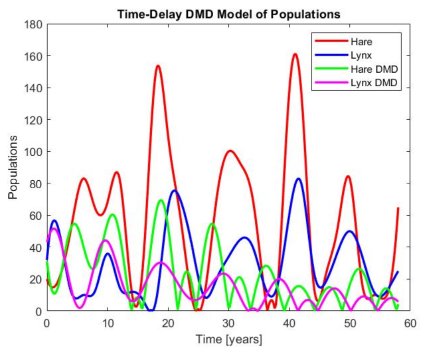

Figure 4: Time-delayed DMD model

4.1 Part 1: Hare & Lynx Populations

The models for DMD & time-delayed DMD can be found in figures 3 & 4, respectively. For the DMD model, the two populations start at the same point as the truth model, but then the model becomes fairly inaccurate. The time-delay DMD model matches the truth model quite nicely at first with damped oscillations until the oscillations become small and approach zero.

The best fit models can be seen in figures 5, 6, & 7 for the pinv method, lasso regression, & ridge regression models, respectively. The pinv model matched the truth model very well with the peaks in the populations occurring at the same time. The lasso regression model matched the lynx populations well, but

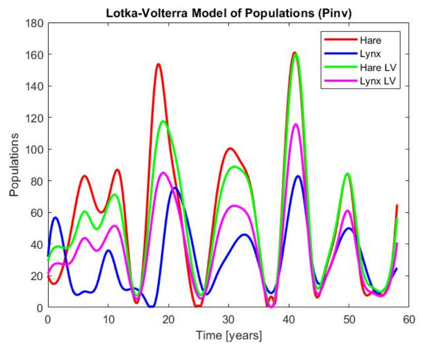

Figure 5: Pseudo-inverse model

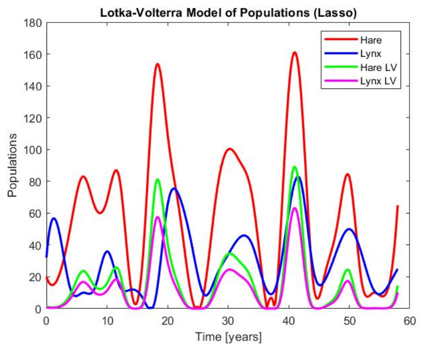

Figure 6: Lasso regression model

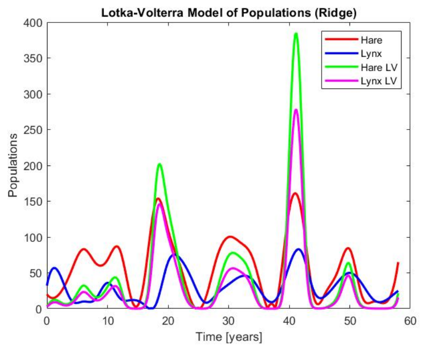

Figure 7: Ridge regression model

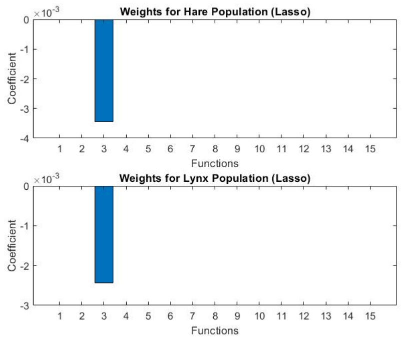

Figure 8: Coefficients for Lasso regression

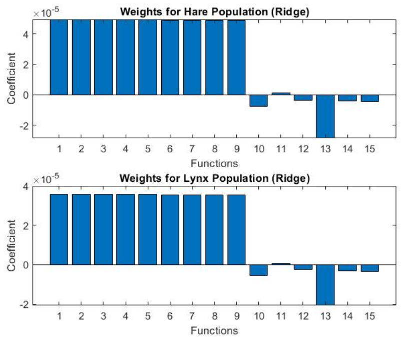

Figure 9: Coefficients for Ridge regression

had much smaller estimations for the hare populations. In contrast, the ridge regression model matched the hare populations, but had lynx population estimates much larger than the truth model. All three best fit models did not do a good job of accurately estimating the populations in the first ten years, but matched the peaks of the population nicely.

The contrast between the two sparse regression models may be because of the most important functions from the library A that were used to get the best fit. In figures 8 & 9 we can see the coefficients on the functions for the lasso model & the ridge model, respectively. Notice that many of the functions were important for the ridge model, but the lasso model was made up of only one important function. That function was Xl^2, which explains why the lynx estimate was so accurate, but the hare estimate wasn't.

It is also important to note that all three of the best fit models used different values for the Lotka-Volterra interaction parameters. The pinv model had $b = 1$, $p = 0$, $r = -0.001$, & $d = 1$. The lasso model used $b = 1$, $p = 0$, $r = -0.005$, & $d = 0$ while the ridge model used $b = 1$, $p = 0$, $r = 0$, & $d = -0.001$. The different parameter values on the sparse regression models can also explain their differences in accuracy. Lasso & ridge both had the same values for b & p, but their values for r & d were almost reversed.

Looking at all five models, it appears that the three best models were the time-delayed DMD model, the pinv model, and the ridge regression model. The KL divergence of the five models proves that these three are indeed the models most similar to the truth model. We can see the KL divergence of the models in figure 10. For the hare population, the KL divergences are; 2.155 for DMD, 1.132 for time-delayed DMD, 0.7232 for pinv, 1.137 for lasso, & 0.852 for ridge. For the lynx population, we have; 1.15 for DMD, 1.121 for time-delayed DMD, 0.915 for pinv, 1.165 for lasso, & 1.074 for ridge. The three smallest KL divergences for the hare population are those for the pinv, ridge, & time-delay DMD models, while the smallest values for the lynx population correspond to the pinv, ridge, & time-delay DMD models as well.

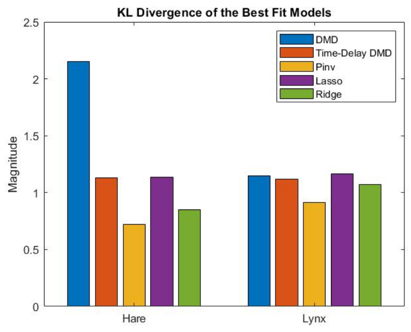

Figure 10: KL divergence of the 5 models

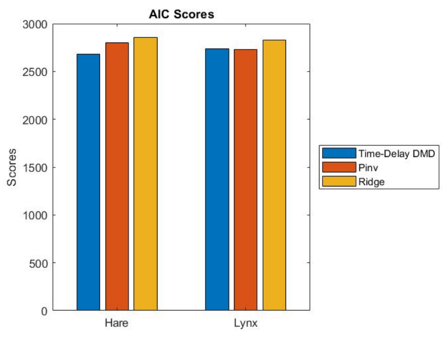

Figure 11: AIC scores of the three best models

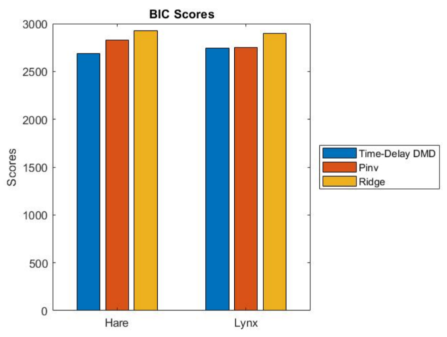

Figure 12: BIC scores of the three best models

Of the three best models, we can look at their AIC/BIC scores (seen in figures 11 & 12, respectively) to determine which model retained the most information from the truth model. The AIC scores are; 2683 (hare) & 2739 (lynx) for time-delay DMD, 2799 (hare) & 2727 (lynx) for pinv, and 2855 (hare) & 2831 (lynx) for ridge. The hare populations had the lowest AIC score with the time-delay DMD model, while the lowest AIC score for the lynx population was with the pinv model. The BIC scores are; 2690 (hare) & 2747 (lynx) for time-delay DMD, 2825 (hare) & 2753 (lynx) for pinv, and 2924 (hare) & 2901 (lynx) for ridge. The model with the lowest BIC score was the time-delay DMD model for both the hare & lynx populations.

4.2 Part 2: BZ Chemical Oscillator

The models for the BZ chemical oscillator were DMD & time-delay DMD, which were found using the same algorithm as the hare & lynx populations. Because of the time data, the algorithm needs to run through each x, y frame in time. However, this is too computationally expensive. Instead, I looked at only four of the frames (frames 1, 500, 1000, & 1200) to get an idea of the models. To be able to understand the models, see figures 13 through 16 for the original frames from the chemical oscillator movie. For the DMD model, see figures 17 through 20, while figures 21 through 24 show the time-delayed DMD model.

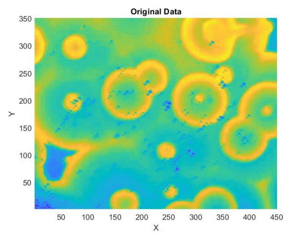

Figure 13: Original data (frame 1)

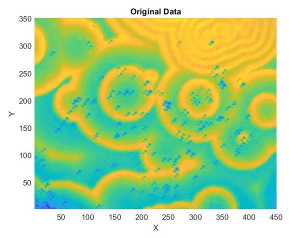

Figure 14: Original data (frame 500)

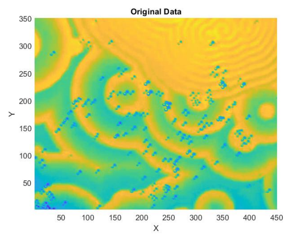

Figure 15: Original data (frame 1000)

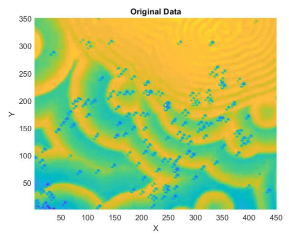

Figure 16: Original data (frame 1200)

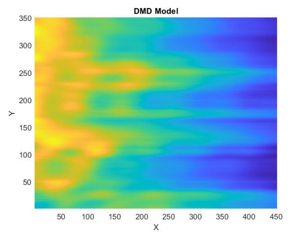

Figure 17: DMD model (frame 1)

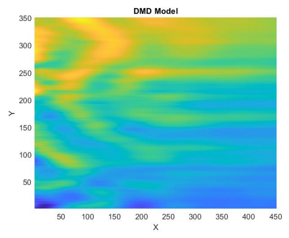

Figure 18: DMD model (frame 500)

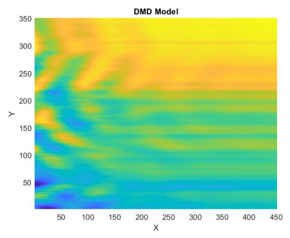

Figure 19: DMD model (frame 1000)

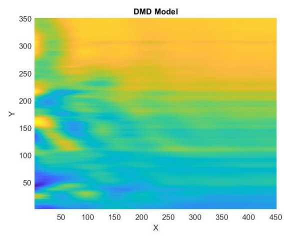

Figure 20: DMD model (frame 1200)

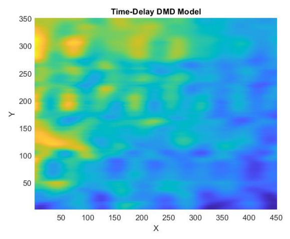

Figure 21: Time-delay DMD model (frame 1)

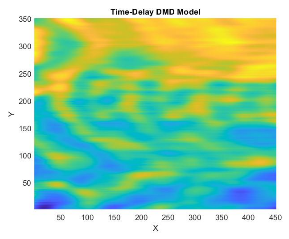

Figure 22: Time-delay DMD model (frame 500)

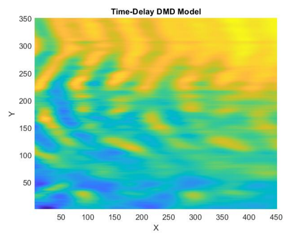

Figure 23: Time-delay DMD model (frame 1000)

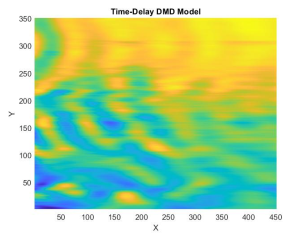

Figure 24: Time-delay DMD model (frame 1200)

From part 1, we expect the time-delay DMD model to be more accurate than the DMD model. Both models appear to show a noisy version of the original frame. However, the time-delay DMD model does look more accurate than the DMD model as we can more clearly see the yellow rings in frame 1200, and all of the frames of the time-delay DMD model show more of the subtle details from the original frames.

5 Summary and Conclusions

From the time-delay DMD model in part 1 (figure 4), it is likely that there are latent variables. We can see that the individuals populations are related to each other through some unknown interaction parameters, which we tried to estimate with the best fit models. Despite this, however, the time-delay DMD model was still the model that retained the most information from the truth model. Similarly, in part 2, the time-delay DMD model was also the most accurate.

It is clear that we can use methods of model discovery & model selection to develop a model that retains most of the information from a given truth model.

Appendix A MATLAB Functions

The following is a list of the important MATLAB functions that were used:

- `s = spline(x,y,xq)` returns a vector of interpolated values `s` corresponding to the query points in `xq`. The values of `s` are determined by cubic spline interpolation of `x` and `y`.
- `plot(X1,Y1,LineSpec1,...,Xn,Yn,LineSpecn,Name,Value)` plots multiple `X`, `Y` pairs using the same axes for all lines, sets the line style, marker type, and color for each line, & specifies line properties using one or more `Name,Value` pair arguments.
- `[U,S,V] = svd(A,'econ')` produces an economy-size singular value decomposition of matrix `A`, such that `A = U*S*V'`.
- `x = diag(A)` returns a column vector of the main diagonal elements of `A`.
- `S = sum(A)` returns the sum of the elements of `A` along the first array dimension whose size does not equal 1.
- `[V,D] = eig(A)` returns diagonal matrix `D` of eigenvalues and matrix `V` whose columns are the corresponding right eigenvectors, so that `A*V = V*D`.
- `X = real(Z)` returns the real part of each element in array `Z`.
- `[t,y] = ode45(odefun,tspan,y0)` where `tspan = [t0 tf]`, integrates the system of differential equations `y'=f(t,y)` from `t0` to `tf` with initial conditions `y0`. Each row in the solution array `y` corresponds to a value returned in column vector `t`.
- `B = pinv(A)` returns the Moore-Penrose Pseudoinverse of matrix `A`.
- `bar(x,y,color,Name,Value)` creates a bar graph with one bar for each element in `y` with bars drawn at the locations specified by `x`, sets the color for all the bars, and specifies properties of the bar graph using one or more name-value pair arguments.
- `B = lasso(X,y,Name,Value)` returns fitted least-squares regression coefficients for linear models of the predictor data `X` and the response `y` with additional options specified by one or more name-value pair arguments.
- `B = ridge(y,X,k)` returns coefficient estimates for ridge regression models of the predictor data `X` and the response `y`. Each column of `B` corresponds to a particular ridge parameter `k`.
- `hist(x,xbins)` creates a histogram bar chart of the elements in vector `x` sorted with intervals or categories determined by the vector `xbins`.
- `Q = trapz(X,Y)` computes the approximate integral of `Y` via the trapezoidal method with respect to the coordinates or scalar spacing specified by `X`.
- `S = load(filename)` loads variables in the MAT-File into the MATLAB workspace.
- `pcolor(C)` creates a pseudocolor plot using the values in matrix `C`.

Appendix B MATLAB Code

The MATLAB codes used to generate the results of this write-up (`Hare_and_Lynx.m`, `rhs_dyn.m`, & `BZ.m`) can be found at the following GitHub repository: `https://github.com/ronwilson016/Data-Science-Projects/tree/master/Population%20Modeling/MATLAB%20Code`