AMATH 582 Homework 5

Junzhao Liu

March 11, 2021

Abstract

The goal of this homework is Use the Dynamic Mode Decomposition method on the video clips which contains a fireground and background object and separate the video stream to both the foreground and the background.

1 Introduction and Overview

As we learned in class the aim of the dynamic Mode Decomposition is to take advantage of low dimensionality in experimental data without having to rely on a given set of governing equations and it allows us to predict what might be in the future, build the matrix contains all information for the video

1.1 Build the matrix X_DMD

Use video Reader to read the video we need to work with and store all matrix that represent each snapshot and turn those to a m*n matrix for further analysis

2.Theoretical Background

Let N equals to the number of spatial points saved per unit time snapshot and M equals to the number of snapshots taken.

The snapshot matrix is denoted as

$$U(\mathbf{x}, t_m) = \begin{bmatrix} U(x_1, t_m) \\ U(x_2, t_m) \\ \vdots \\ U(x_n, t_m) \end{bmatrix}$$

Then we can use it to from columns of data matrices

$$\mathbf{X} = \begin{bmatrix} U(\mathbf{x}, t_1) & U(\mathbf{x}, t_2) & \cdots & U(\mathbf{x}, t_M) \end{bmatrix},$$

Find Koopman operator A which is a time-independent operator such as

$$\mathbf{x}_{j+1} = \mathbf{A}\mathbf{x}_j$$

Form the data matrix X compute SVD of $\mathbf{X}_1^{M-1}$ then find the eigenvalues and eigenvectors of the matrix

$$\tilde{\mathbf{S}} = \mathbf{U}^* \mathbf{X}_2^M \mathbf{V} \Sigma^{-1}$$

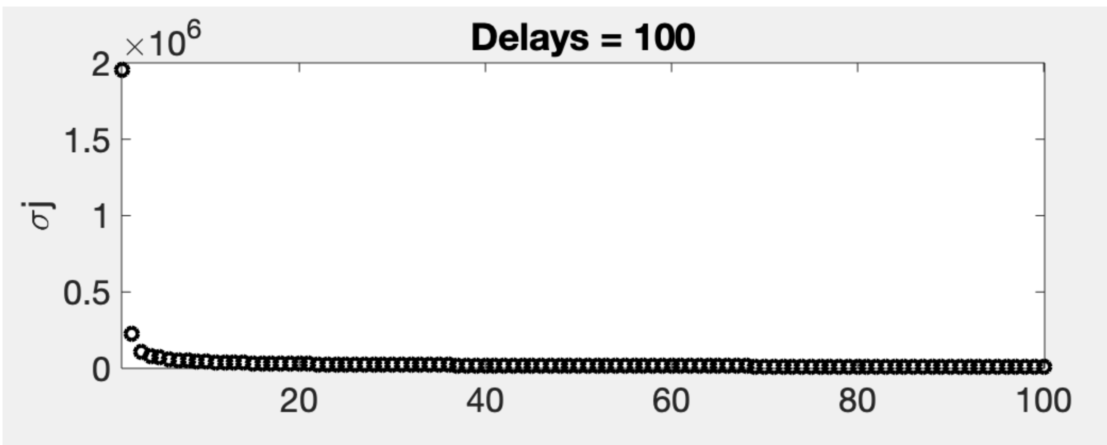
Finally, find the coefficients.

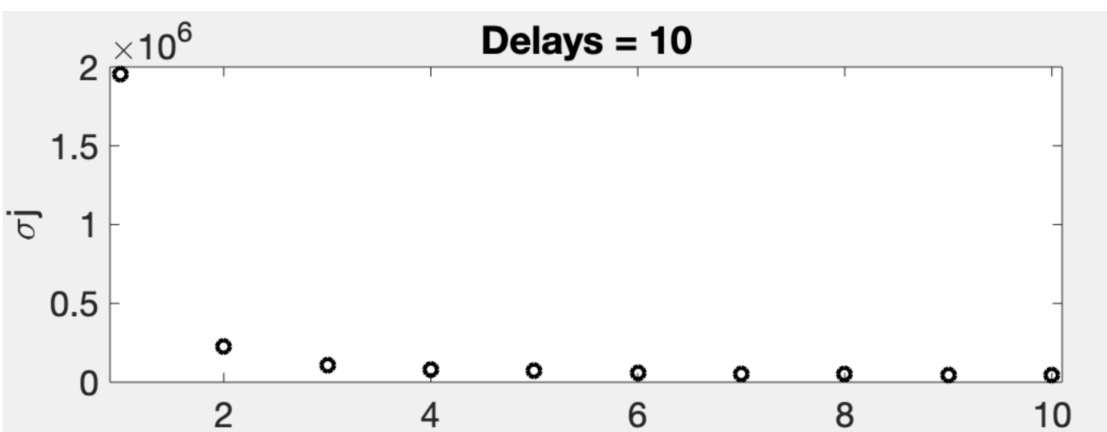
3 Algorithm Implementation and Development

1. Use `function viedioReader` to read video to Matlab
2. Store all snapshot matrix into a vector
3. Turn it to a n*m matrix and apply SVD decomposition to find the lower rank DMD
4. then use the original matrix subtract the lower rank to obtain the result

2 Computational Results

The plot of SVD results





Summary and Conclusions

I think I am in the right approach to find the matrix of DMD, but unfortunately I didn't plot out the PDE and DMD solutions, therefore I can't do any further analysis to obtain the final result.

Appendix A MATLAB Functions

[U,S,V] = svd(A) performs a singular value decomposition of matrix A, such that A = U*S*V'.

B = reshape(A,sz1,...,szN) reshapes A into a sz1-by-...-by-szN array where sz1,...,szN indicates the size of each dimension. You can specify a single dimension size of [] to have the dimension size automatically calculated, such that the number of elements in B matches the number of elements in A. For example, if A is a 10-by-10 matrix, then reshape(A,2,2,[]) reshapes the 100 elements of A into a 2-by-2-by-25 array.

waterfall(X,Y,Z) creates a waterfall plot, which is a mesh plot with a partial curtain along the *y*dimension. This results in a "waterfall" effect. The function plots the values in matrix Z as heights above a grid in the *xy*-plane defined by X and Y. The edge colors vary according to the heights specified by Z.

Appendix B MATLAB code

```
clear all; clc
v = VideoReader('monte_carlo_low.mp4')
numFrames = 0;
current = cell([],1) ;
currentcolumn = zeros(540*960,379);
while hasFrame(v)
    F = readFrame(v);
    numFrames = numFrames + 1;
    current{numFrames} = F(:,:,3);
    currentcolumn(:,numFrames)=reshape(current{numFrames},540*960,1);
    %imagesc(F)
    %drawnow
end
 usol = currentcolumn;
L = 40; n = 379;
x2 = linspace(-L/2,L/2,n+1); x = x2(1:n);
k = (2*pi/L)*[0:n/2-1 -n/2:-1]';

Time
slices = 318;
t = linspace(0,6.323,slices+1); dt = t(2) - t(1);

X = usol';
X1 = X(:,1:end-1);
X2 = X(:,2:end);


[U, Sigma, V] = svd(X1,'econ');
S = U'*X2*V*diag(1./diag(Sigma));
[eV, D] = eig(S); % compute eigenvalues + eigenvectors
mu = diag(D); % extract eigenvalues
omega = log(mu)/dt;
Phi = U*eV;


y0 = Phi\X1(:,1); % pseudoinverse to get initial conditions
```

```
u_modes = zeros(length(y0),length(t));
for iter = 1:length(t)
   u_modes(:,iter) = y0.*exp(omega*t(iter));
end
u_dmd = Phi*u_modes;


% [U, S, V] = svd(X1,'econ');
subplot(2,1,1)
t = linspace(0,6.323,518399);
% Singularvalues
delays=100;
plot(diag(S),'ko','Linewidth',2)
ylabel('\sigmaj')
set(gca,'Fontsize',16,'Xlim',[0.9 delays+0.1])
title(['Delays = ', num2str(delays)])
subplot(2,1,2)
% % Right-singular vectors
% plot(t(1:end-delays+1),V(:,1),'r','Linewidth',2)
% hold on
% plot(t(1:end-delays+1),V(:,2),'b--','Linewidth',2)
% xlabel('t')
% ylabel('vj(t)')
% set(gca,'Fontsize',16)



subplot(2,1,1), waterfall(x,t,abs(usol)), colormap([0 0 0])
xlabel('x')
ylabel('t')
zlabel('|u|')
title('PDE Solution')
set(gca,'FontSize',16)
```