AMATH 482 HW3: Music Genre Identification

Ken Lo

March 10, 2018

Abstract

Using Dynamic Mode Decomposition, separates a video's foreground and background object.

1 Introduction and Overview

In this assignment, I applied Dynamic Mode Decomposition onto videos each consisting of a foreground object and a background object(specifically, a speaker in front of a background). Through Dynamic Mode Decomposition, I attempted to recreate the foreground and background objects and showed the results.

2 Theoretical Background

2.1 Sigular Value Decomposition (SVD)

A Singular Value Decomposition (SVD) takes the form

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$$

where

$$\mathbf{U} \in \mathbb{C}^{m\times m} \text{is unitary}$$
$$\mathbf{V} \in \mathbb{C}^{n\times n} \text{is unitary}$$
$$\mathbf{\Sigma} \in \mathbb{R}^{m\times n} \text{is diagonal}$$

Here the matrix $\mathbf{\Sigma}$ is diagonal with entries $\sigma_1 < \sigma_2 < \cdots < \sigma_n$.

By theorem 14.1.77 from class notes, any matrix $\mathbf{A}$ always has an SVD. Through SVD, we can decompose our data matrix and reduce the redundancy of our data, thus creating a lower-dimensional representation of our data.

2.2 Dynamic Mode Decomposition

Given a dynamical system

$$\frac{dx}{dt} = f(x, t; \mu)$$

the Dynamic Mode Decomposition procedure constructs the proxy, approximate locally linear dynamical system

$$\frac{dx}{dt} = \mathbf{A}\mathbf{x}$$

. With initial condition $\mathbf{x}(\mathbf{0})$, we have the solution to the above system

$$x(t) = \sum_{k=1}^{n} \phi_k \exp(\omega_k t) b_k = \Phi \exp(\Omega t) b$$

, where ϕ_k and ω_k are the eigenvectors and eigenvalues of $\mathbf{A}$, respectively.

The DMD computes the leading eigendecomposition of the best-fit linear operator $\mathbf{A}$ relating the data $\mathbf{X}' \approx \mathbf{A}\mathbf{X}$:

$$\mathbf{A} = \mathbf{X}'\mathbf{X}^{\dagger}$$

The DMD modes, are the eigenvectors of $\mathbf{A}$, and each mode corresponds to a particular eigenvalue of $\mathbf{A}$.

3 Algorithm Implementation and Development

3.1 Gathering Data

The data in this assignment are videos consisting of a foreground object and a background object. Here I used various speakers' monologues' videos as the test subject. These videos are downloaded from YouTube and only 10 seconds of each of these videos are used to test our DMD algorithm.

3.2 Preparing video data

Read in the videos into MATLAB and convert them into image frames matrices. Then we turn each of these frames' matrices into a column vector and stack them horizontally into a big data matrix `A`. In addition, we also created two data matrices with time lags, with A_1 taking only the first $n-1$ of the columns and A_2 taking the last $n=1$ of the columns in A.

3.3 DMD Algorithm

First, we take the SVD of the data matrix A, and have a 2-rank truncation. Then we construct $\tilde{A}$, which is the 2 by 2 projection of the full matrix. Then we compute the eigendecomposition of $\tilde{A}$, obtaining the eigenvectors and eigenvalues. And finally we reconstruct the eigendecompositino of A from the eigenvalues and eigenvectors. The eigenvectors of A are given by columns of

$$\Phi = X'V\Sigma^{-1}W$$

.

Then we compute the initial coefficient values b_k, which can be obtained by multipling the pseudo-inverse of Φ by the initial snapshot x_1 at $t_1 = 0$.

Having done the above steps, we can approximate $x(t)$ as mentioned in section 2.

4 Computational Results

4.1 Test Video 1: Prof. Kutz Lecturing

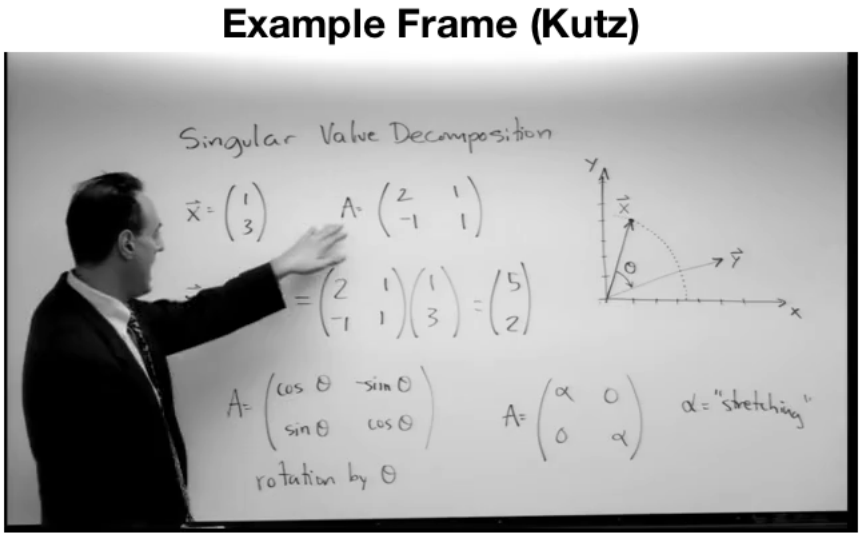

Figure 1: Example frame from Kutz Video

Figure 1 shows an example from Kutz's lecture video, with which we will apply DMD in order to separate the foreground and background object.

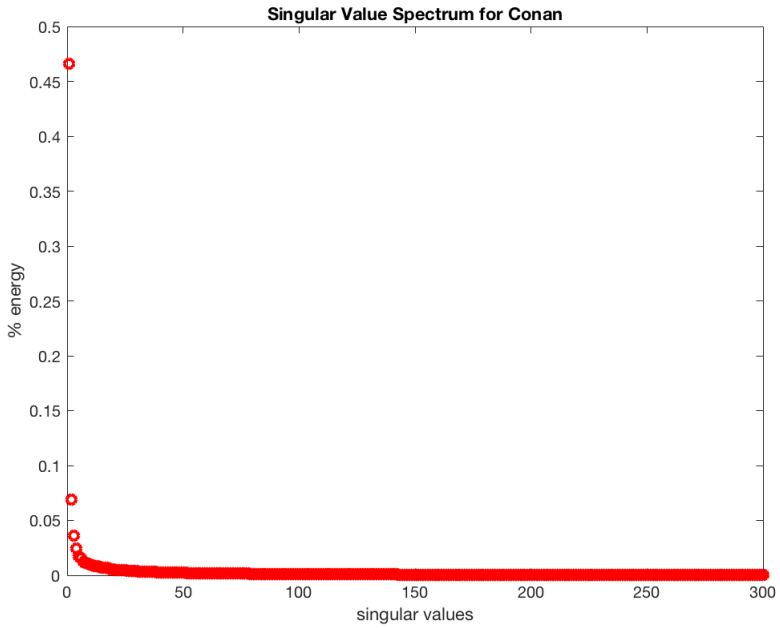

Figure 2: Singular Value Spectrum

Figure 2 shows the singular value spectrum of the video, we can clearly see that there is only a few dominant modes, and the singular values getting very close to 0 at around 20. Thus, I chose the first 20 modes to perform DMD on.

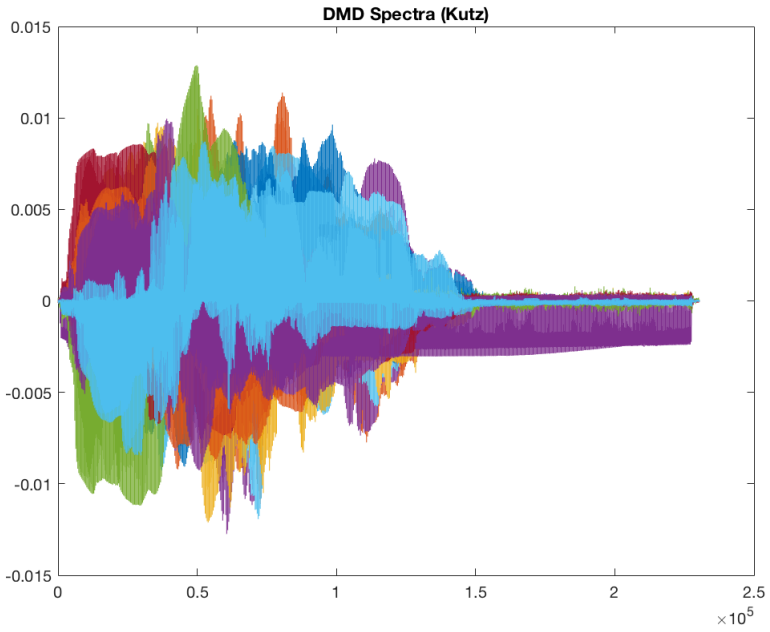

Figure 3: DMD Spectra of 20 modes

In figure 3, we can see that the DMD were able to separate some different signals from the videos.

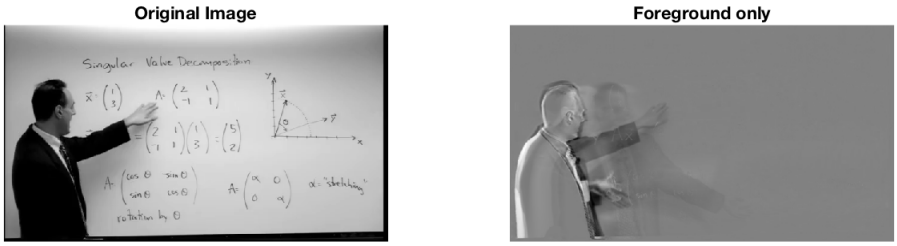

Figure 4: Comparing the original image and only the foreground (Prof. Kutz)

In figure 4, the side by side comparison shows that the DMD did a decent job separating the white board (background) from Prof. Kutz (foreground)

4.2 Test 2: Conan Monologue

The video to test in this case is a short video the late night talk show Conan's monologue.

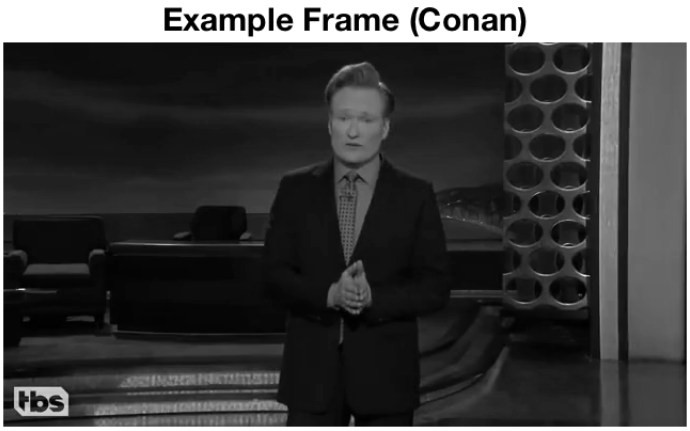

Figure 5: Example image

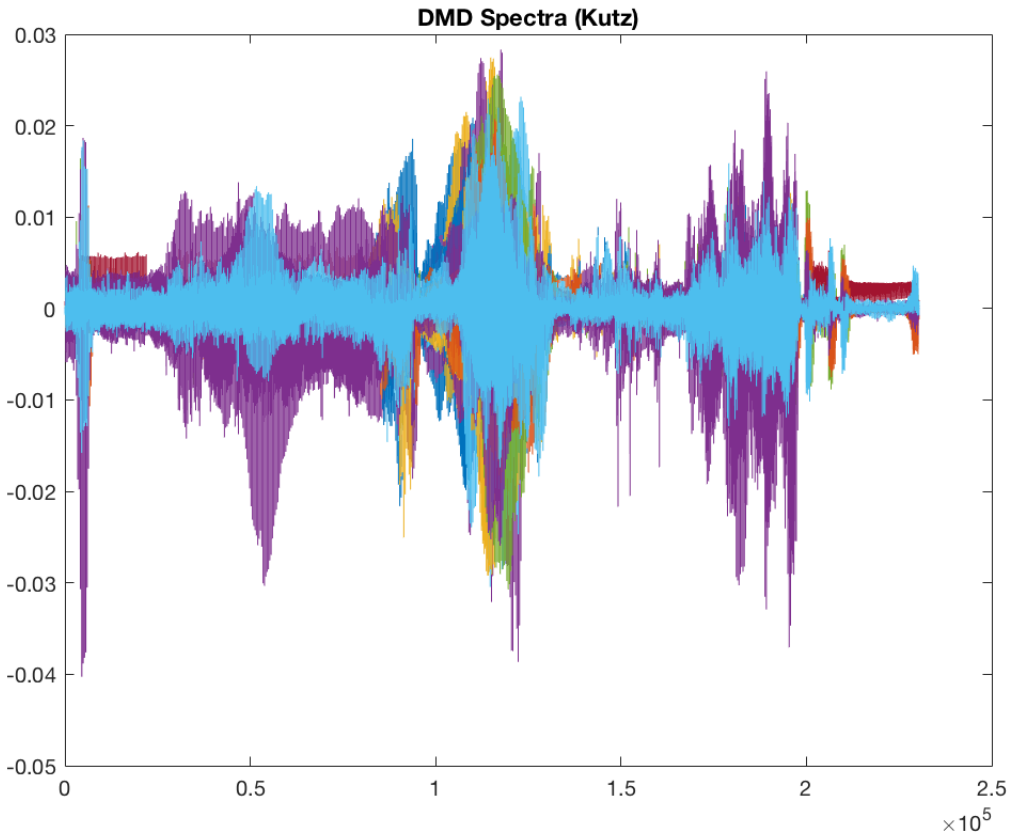

Figure 6: DMD Spectra of 20 modes

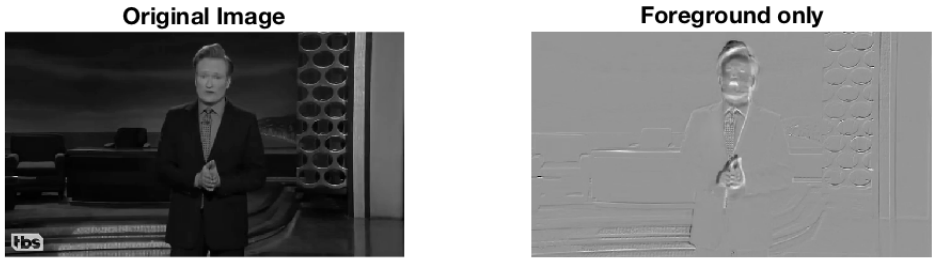

Figure 7: Comparing original image and only the foreground (Conan)

Similar to test 1, we have one speaker in front with a still background. In figure 7, we can see that the DMD still managed to outline the shape of Conan in the sparse representation of the image.

4.3 test 3: Backpack Kid

The third video to be tested is the backpack kid, this video might pose a challenge to our DMD algorithm since there are multiple objects moving in the video.

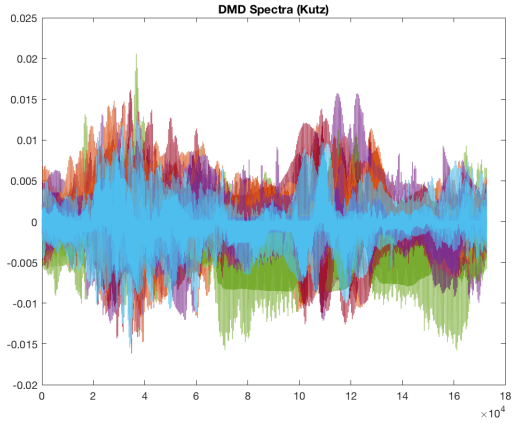

Figure 8: DMD Spectra of 20 modes

Figure 9: Comparing original image and only the foreground (the kid)

From figure 9, the DMD surprisingly were able to not only isolate the backpack kid, but also the other actors on the stage, including the singer (Katy Perry).

5 Summary and Conclusions

In this assignment, we have experimented with and witnessed the power of DMD, especially in separating foreground and background objects in a video. However, our methods would require a completely still background in order to give us a perfect isolation on the object of interest.

DMD is an incredible tool for data reduction, data mining and when analyzing time series data.

A MATLAB functions used and brief implementation explanation

- readFrames - read a frame from a video
- videoReader =-read a video file, storing the frames as well as some metadata
- svd - compute SVD
- eig- compute eigenvalues and eigenvectors
- reshape - reshape matrices into and from vectors

B MATLAB codes

B.1 hw4.m

```
clear all; close all; clc
vr = VideoReader('data/bpkid_short.mp4');
frames = vid2frames(vr);
nFrames = round(vr.Duration * vr.FrameRate);
fHeight = vr.Height;
fWidth = vr.Width;

%% example frame
imshow(frames(:,:, 1), [])
title("Example Frame (Conan)", 'fontsize', 15)
%% vectorize images
A = zeros(size(frames, 1) * size(frames, 2), nFrames);
A1 = zeros(size(frames, 1) * size(frames, 2), nFrames - 1);
A2 = zeros(size(frames, 1) * size(frames, 2), nFrames - 1);

for i=1:nFrames
   f = frames(:,:,i);
   vec = f(:);
   A(:,i) = vec;
   if i < nFrames % frame 1-299
       A1(:,i) = vec;
   end
   if i > 1 % frame 2-300
       A2(:,i - 1) = vec;
   end
end

%% SVD
[u,s,v] = svd(A, 'econ');


%% Sigular Value Spectrum
figure(3)
plot(diag(s)/sum(diag(s)), 'ro', 'LineWidth', 2)
title('Singular Value Spectrum for Conan')
xlabel('singular values')
ylabel('% energy')
%% DMD
```

```
r=20;
[u,s,v]=svd(A1,'econ');

Ur=u(:,1:r);
Sr=s(1:r,1:r);
Vr=v(:,1:r);

dt=0.033;
Atilde=Ur'*A2*Vr/Sr;
[W,D]=eig(Atilde);
Phi=A2*Vr/Sr*W;

lamda=diag(D);
omega=log(lamda)/dt;
figure(4)
plot(real(Phi))
title('DMD Spectra (Kutz)')

a = A(:,100); %% Low rank reconstruction
b=Phi\a;
tdynamics = zeros(r,nFrames);
for i = 1:length(tdynamics)
    tdynamics(:,i) = (b.*exp(omega*i));
end

A_lowrank = Phi*tdynamics;
A_Sparse = A_lowrank - abs(A(:,100));

%%
subplot(1,2,1)
imshow(frames(:,:,1), [])
title('Original Image')
subplot(1,2,2)
imshow(reshape(A_Sparse(:,1), [fHeight, fWidth]), [])
title('Foreground only')
%% svd reconstruction
% j=1;
%  for i=[1,10,50,100,200,nFrames - 1]
%     subplot(2,3,j)
%     test = u(:,1:i) * s(1:i, 1:i) * v(:,1:i).';
%     test = uint8(test(:,100));
%     imshow(reshape(test,[fHeight,fWidth]))
%     title("SVD Reconstruction (first " + num2str(i) + " POMs)")
%     set(gca, 'fontsize', 10)
%     j=j+1;
%  end
%  %% dmd reconstruction
%  figure(2)
%  j=1;
%  for i=[1,10,50,100,200,nFrames]
%     subplot(2,3,j)
%     test = uint8(A_Sparse(:,1));
%     imshow(reshape(test,[fHeight,fWidth]))
```

```
%     title("Sparse: " + num2str(i))
%     j=j+1;
%  end
%
%  figure(4)
%  j=1;
%  for i=[1,10,50,100,200,nFrames]
%     subplot(2,3,j)
%     test = uint8(A_lowrank(:,i));
%     imshow(reshape(test,[fHeight,fWidth]))
%     title("Low-Rank: " + num2str(i))
%     j=j+1;
%  end

 %%

 sparseFrames = frames;
 for i=1:nFrames
     sparseFrames(:,:,i) = reshape(A_Sparse(:,i), fHeight, fWidth);
 end

 %%
 for i = 1:nFrames
     imshow(real(sparseFrames(:,:,i)), [])
 end
```

B.2 vid2frames.m

```
function frames = vid2frames(vid)
nFrames = round(vid.FrameRate * vid.Duration);

height = vid.height;
width = vid.width;

frames = zeros(height, width, nFrames);
for i=1:nFrames
    frame = rgb2gray(readFrame(vid));
    frames(:,:,i) = frame(1:vid.height, 1:vid.width);
end
```