Hongyu Cao

Department of Mathematics

University of Washington

**ABSTRACT**

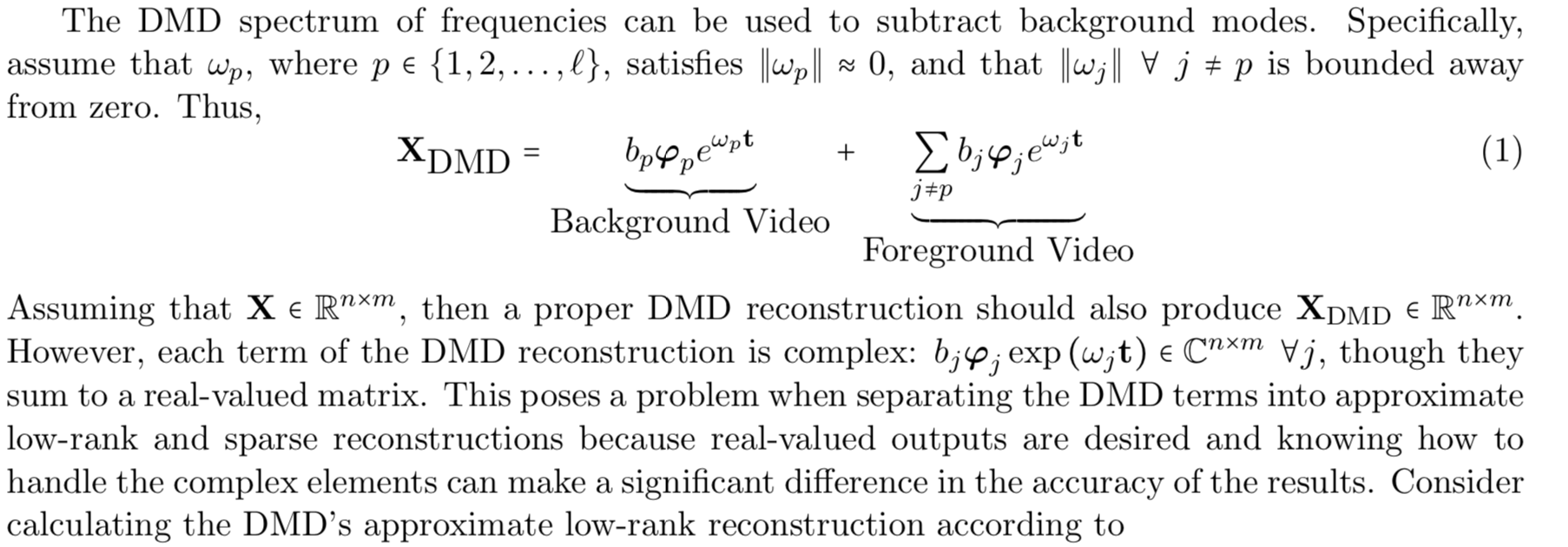
This project is to separate the foreground and the background of a video by using Dynamic Mode Decomposition (DMD) We will compare the its result with the original video to observe the accuracy of this algorithm.

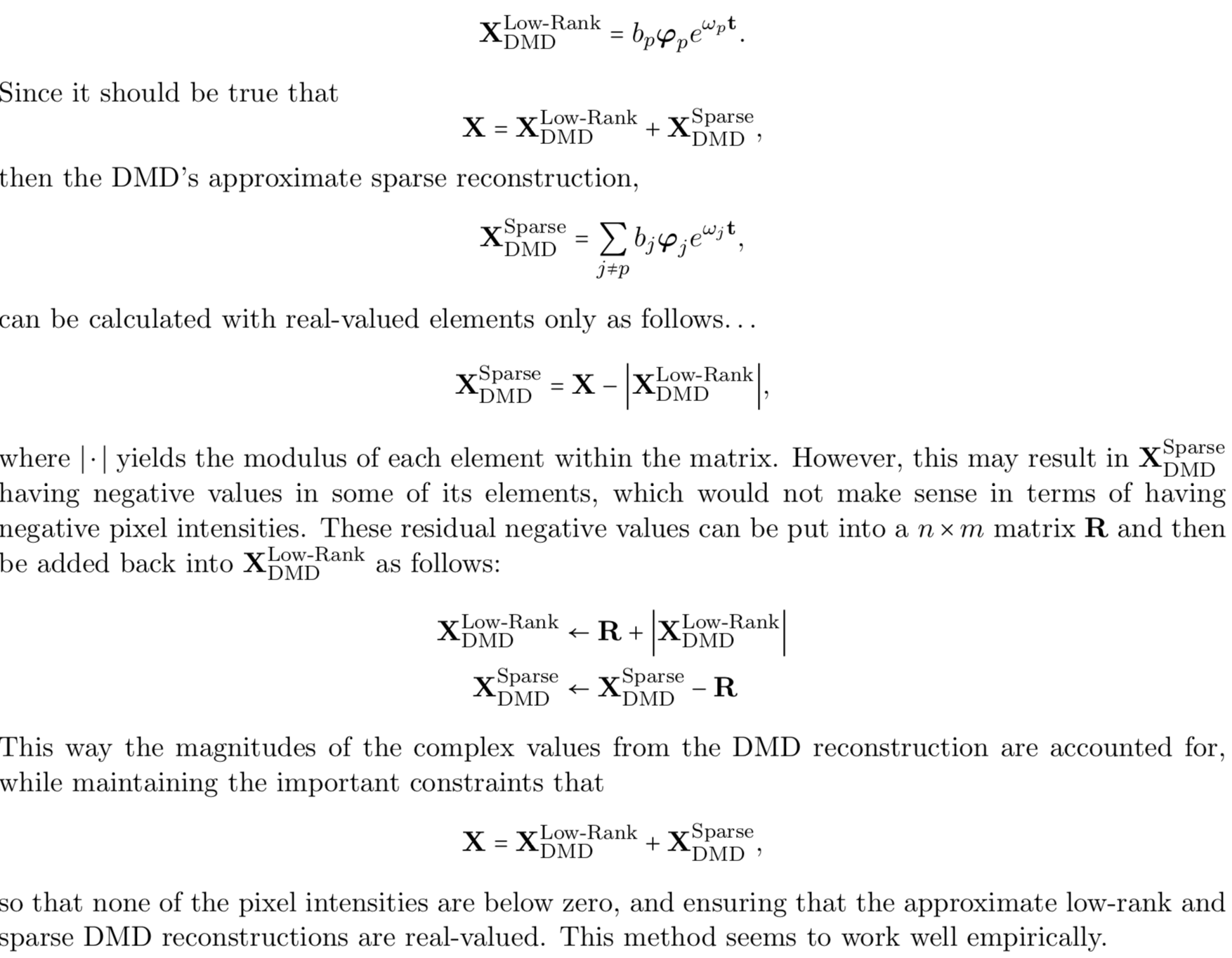
**I. Introduction**

Dynamic Mode Decomposition (DMD) is a powerful tool for analyzing a dynamical systems measured by high-dimensional data. The dynamic mode decomposition is the featured method of this book. It’s like a combination of spatial dimensionality-reduction such as POD and Fourier transforms in time. One of the advantage of this method is it totally depends on the data and it’s easy to execute. It simply relies on collecting snapshots of data at a number of times where k = 1, 2, 3 …,m. In this homework, we can apply this method perfectly to the video since video is made of lots of frames and each frame is a bunch of data of the value of each pixel.

**II. Theoretical Background**

Below I just attached the explanation from this homework, which talks about how we can use DMD to separate the foreground and background of a video.

****

****

**III. Algorithm Implementation and Development**

I only use one video and it’s from viratdata. First I read the video file and get each frame of it and put all of the frames in single column form into a matrix. For simplicity and efficiency, I didn’t use all the video but only take frame from 1 to 200 and resize it into ½ of the original one. Each column of the matrix is a frame. Then we do a SVD to subtract important information and use DMD to separate foreground and background based on different threshold of Omega. In this case, I choose the minimum value of Omega. And we can get low dimension matrix as the foreground moving object and sparse matrix as background. Finally, I reconstruct the background and foreground frame at frame 50, 100, 150, 200 and compare them with the original frame.

**Sec. IV. Computational Results**

The video is about a parking lot and on the road there are cars passing. The environment is complicated, consisting of building, trees and sky etc.

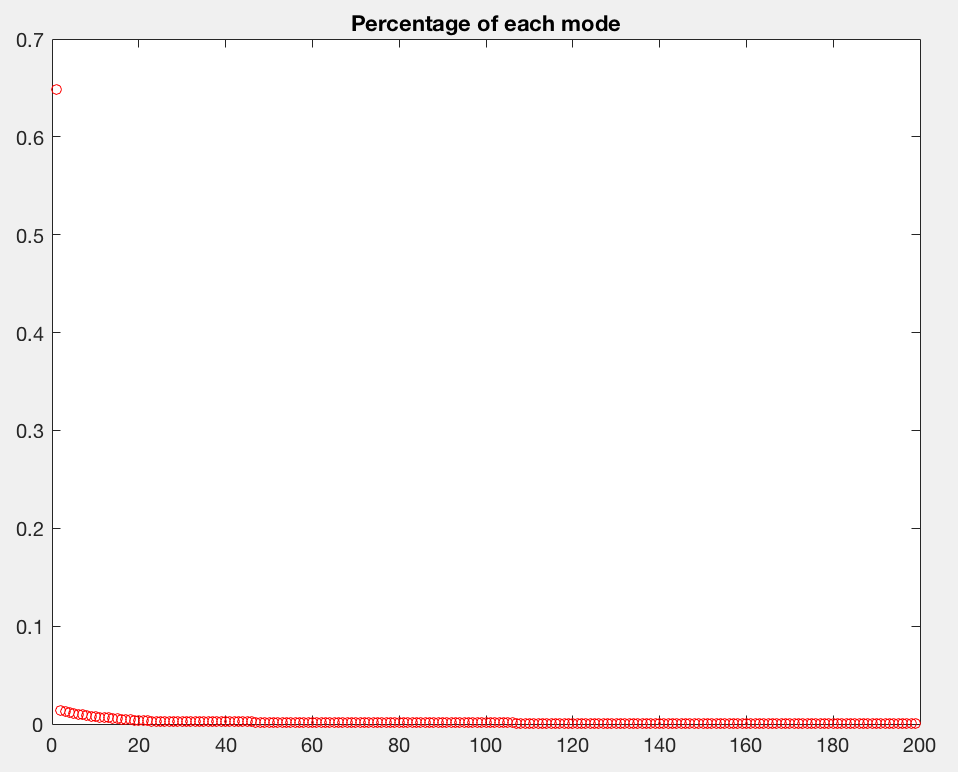
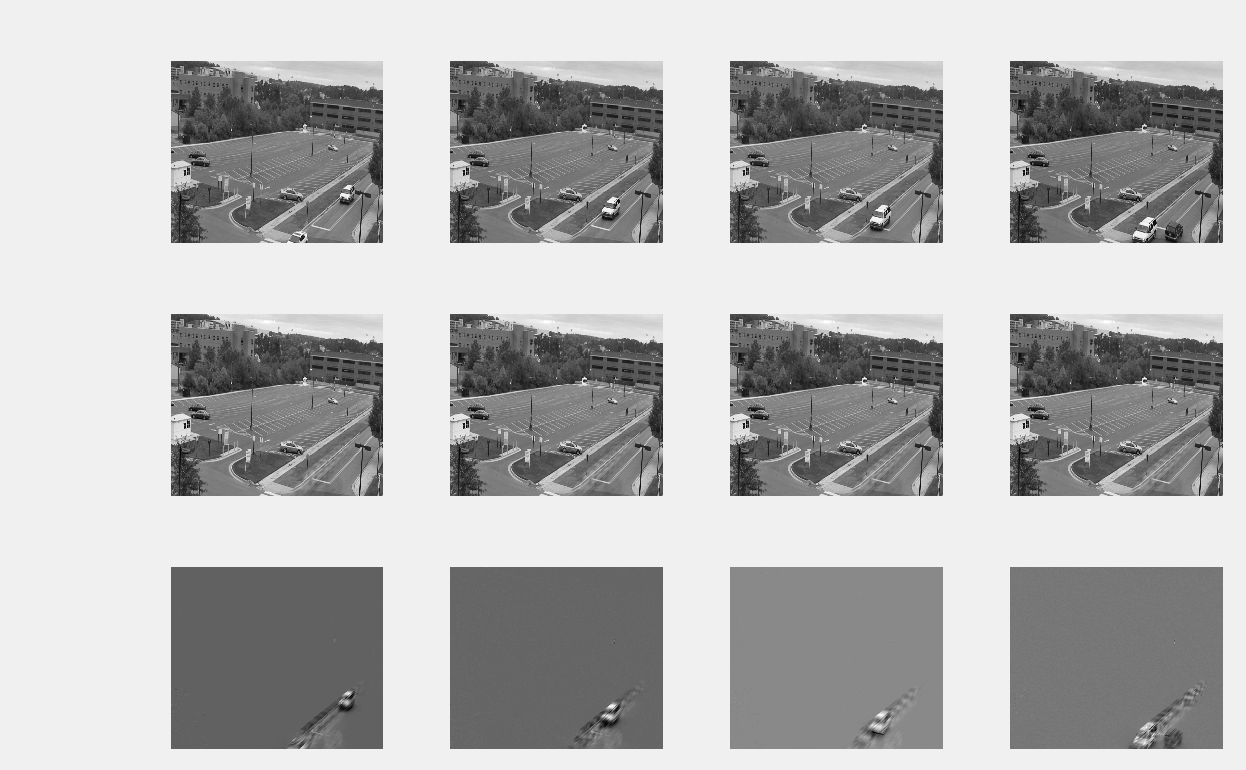


Figure 1: Percentage of the mode of the frames and it’s almost 0 after 20.

So we can choose r = 20.

Comparison of the DMD result and original video

Original Frame



Foreground

Background

Figure 2: The result of using DMD to separate background and moving object

**Sec. V. Summary and Conclusions**

we can say DMD did a well on this video though the moving object is still with some background of its moving track.

**Appendix A MATLAB functions used and brief implementation explanation**

v = VideoReader(filename) creates object v to read video data from the file named filename.

video = readFrame(v) reads the next available video frame from the file associated with v.

[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.

**Appendix B MATLAB codes**

%% Read video

video=[];

v = VideoReader('mov1.mp4');

while hasFrame(v)

frame = readFrame(v);

frame = rgb2gray(frame);

frame=imresize(frame,0.5);

frame = reshape(frame ,[] ,1);

video = [video, frame];

end

%% Original Video

[m,n] = size(video); n=200;

video = video(:,1:n);

%% SVD

video = double(video);

v1 = video(:,1:n-1);

v2 = video(:,2:n);

[U,S,V] = svd(v1,'econ');

%% Plot mode

plot(diag(S)/sum(diag(S)),'ro','Linewidth',[0.5],'MarkerSize', 5), title('Percentage of each mode')

%%

r=20; U=U(:,1:r);

S = S(1:r, 1:r);

L =n-1;

slices = n-1;

t = linspace(0, 1, slices+1);

dt = t(2) -t(1);

nf = 1;

slicesf=slices \* nf ;

tf = linspace(0, nf, slicesf+1);

%% DMD

V=V(:, 1:r);

Atilde =U'\*v2\*V/S;

[W,D] = eig(Atilde);

Phi = v2\*V/S\*W;

lambda = diag(D);

omega = log(lambda)/dt;

y = Phi\video(:,1);

%% Seperate Omega

small = find(abs(omega) == min(abs(omega)));

big = find(abs(omega) ~=min(abs(omega)) );

omega1 = omega(small );

omega2 = omega(big);

y1 = y(small);

y2 =y(big);

%% Seperate foreground and background

r1 = length(small);

r2 = length(big);

u\_bak = zeros(r1, length(t)); u\_for = zeros(r2, length(t));

for iter = 1:length(tf)

u\_bak(:,iter) = (y1.\*exp(omega1\*(tf(iter))));

end

for iter = 1:length(tf)

u\_for (: , iter ) = (y2.\*exp(omega2\*( tf ( iter ))));

end

%%

X\_low = Phi(:,small)\*u\_bak;

X\_sparse = Phi(:,big)\*u\_for;

%%

X = X\_low+X\_sparse;

X\_sparse = real(X - abs(X\_low));

X\_sparse = X\_sparse - min(min(X\_sparse)); X\_low = abs(X\_low) + min(min(X\_sparse));

%% Reconstruction of frame

figure (1) ,

a=[50,100,150,200];

for i = 1:4

subplot (3 ,4 ,i);

temp = video(:,a(i));

temp = reshape (temp,360,640);

imagesc(temp); colormap ( gray ) ;

axis off ;

end

for i = 5:8

subplot (3 ,4 ,i) ,

temp = X\_low(:,a(i-4));

temp = reshape (temp,360,640);

imagesc(temp); colormap ( gray ) ;

axis off;

end

for i = 9:12

subplot (3 ,4 ,i) ,

temp = X\_sparse(:,a(i-8));

temp = reshape (temp,360,640);

imagesc(temp); colormap ( gray ) ;

axis off;

end