Analyzing a Mass on a Spring

Zhiyang Cheng

February 24, 2021

1 Introduction and Overview

This report we are attempted to find the different displacement of a mass on a spring using the Principle Component Analysis. The specific spring-mass system is given in the video within four cases. The cases are as follows:
1. Test 1: Ideal Case- the entire motion is in the z direction with simple harmonic motion being observed.
2. Test 2: Noisy Case. This one is more difficult to extract the simple harmonic motion. But the dynamics will still be extracted with the PCA algorithms.
3. Test 3: Horizontal Displacement. In this case the mass is released off-centre so as to produce motion in the x y plane as well as the z direction.
4. Test 4: Horizontal Displacement and Rotation. In this case the mass is released off-centre and rotates so as to produce rotation in the x y plane and motion in the z direction.

2 Theoretical Background

2.1 The Singular Value Decomposition - SVD

The Singular Value Decomposition (SVD) of a matrix is a factorization of that matrix into three matrices.

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

The singular value decomposition of an m$\times$ n complex matrix M is a factorization of the form $U\Sigma V^*$.
U is an $m \times m$ complex unitary matrix, Σ is an $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal, and V is an $n \times n$ complex unitary matrix.

2.2 Principal Component Analysis - PCA

The principal component analysis is special way to use the singular value decomposition (SVD). Starting with a vector and we can easily calculate its mean and variances. The six rows of the measurement matrix X contain the x and y-coordinates. For each X we construct the covariance matrix C_x defined by

$$C_x = \frac{1}{n-1} X X^T \tag{2}$$

PCA is to find a basis in which Cx is diagonal and we use SVD to achieve the goad. SVD can help us captures as much energy as possible. Suppose we have X have the SVD and let the result be Y. Then, the covariance of Y is:

$$C_y = \frac{1}{n-1} \Sigma^2 \tag{3}$$

3 Algorithm Implementation and Development

3.1 Setting up

To start to analyze the displacement of the mass on spring provided by the videos, we first load camera mat files using Matlab command load. The data are 4D datasets.

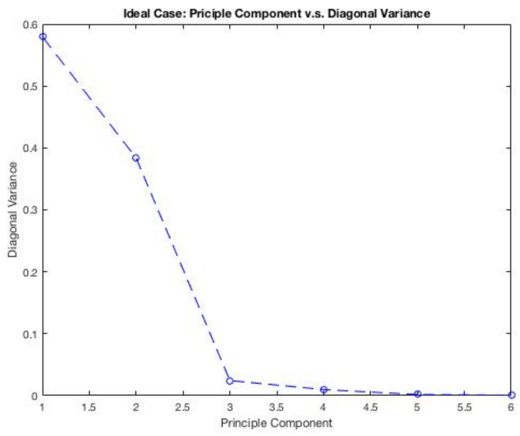
3.2 Analyze

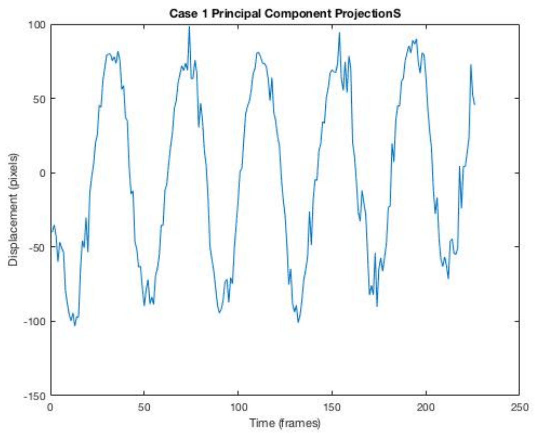
We want to focus on the displacing mass. In the foor loop we can locate the coordinates of the displacing mass and averaged them out. We want to store it in to a array. We may lot out the trajectory for each cases to understand what is going on. Since each camera has different number of image, we trim them to be the same length. We then perform SVD.
Lastly we want to plot out the vraince and then compared the energy of the singular values. We plot out the projections on the selected principle component. We repeated the steps above for all four cases.

4 Computational Results

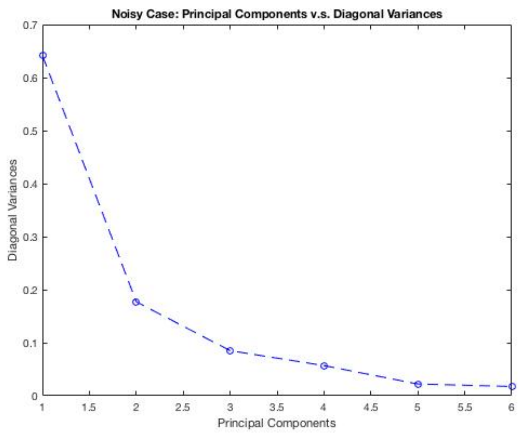
4.1 Case 1

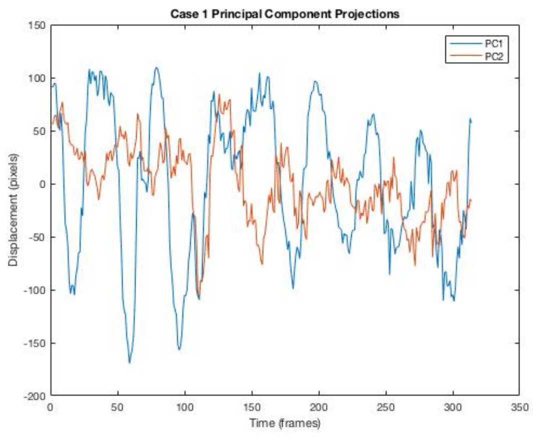
Based on figure below, there is only one principal component. We compared all of the resulting component and this is one capturing most of the energy. This make sense since the ideal case suggests that the entire motion is in the z direction.



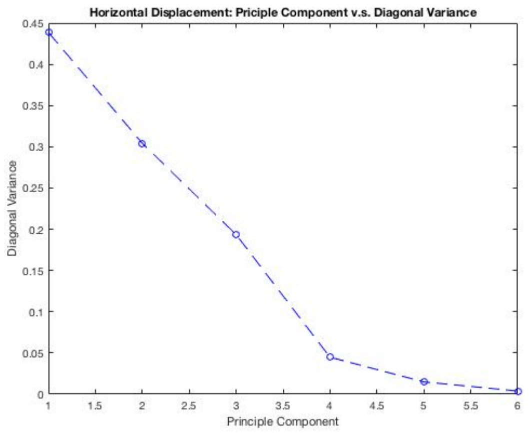


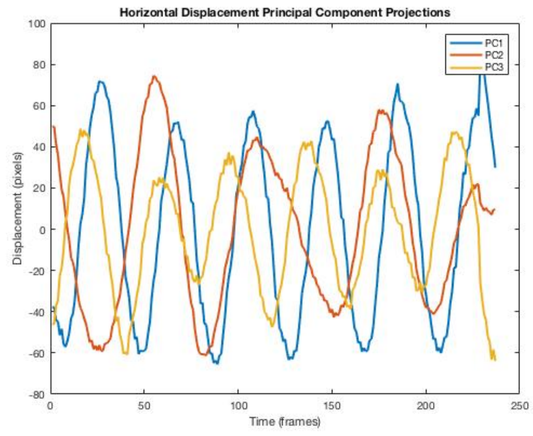
Based on figure below, there are two principal component. We compared all of the resulting component and the second variances seem to be higher. This make sense becuase of the niose in the data.



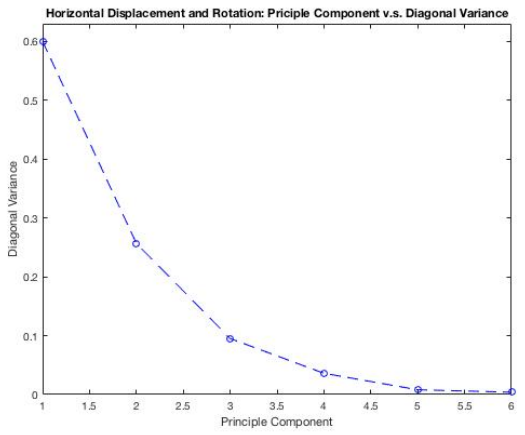


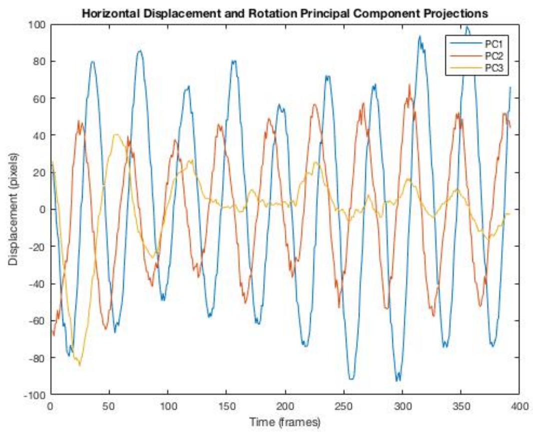
Based on figure below, there are three principal component. We compared all of the resulting component and those three seem be stronger than the other.





Based on figure below, there are three principal component. We compared all of the resulting component and those three seem be stronger than the other. In this case the mass is released off-centre and rotates so as to produce rotation in the x y plane and motion in the z direction which in a way is similar to case 1.





5 Summary and Conclusions

Based on the computational results above, we can see that Principal Component Analysis in capturing the movement. The ideal case and the last case- Horizontal Displacement and Rotation Case are the two most interesting result to me. The result of ideal case directly shows that it is moving in one direction and only one principal component captures the strongest energy.

6 References

https://en.wikipedia.org/wiki/Linear$_discriminant_analysis$
$https://en.wikipedia.org/wiki/Singular_value_decomposition$

Appendix A MATLAB Functions

- `[x,y,k] = ind2sub(siz,IND)` returns the x, y, and k indeces, which can then be translated into spacial or frequency domain coordinates.
- `diag(x)` returns a column vector of the main diagonal elements of x
- `[U,S,V] = svd(x)` performs a singular value decomposition of matrix x

Appendix B MATLAB Code

```
%% Test 1
clear; close all; clc;

load('cam1_1.mat')
load('cam2_1.mat')
load('cam3_1.mat')

frames11 = vidFrames1_1(:,270:400,:,:);
num1 = size(frames11,4);
cam1 = zeros(num1,2);
for i = 1:num1
    t1 = frames11(:,:,1,i);
    [Max,Ind] = max(t1(:));
    [x1,y1] = ind2sub([size(t1,1), size(t1,2)], Ind);
    cam1(i,:) = [x1,y1];
end

frames21 = vidFrames2_1(:,200:370,:,:);
num2 = size(frames21,4);
cam2 = zeros(num2,2);
for i = 1:num2
    t2 = frames21(:,:,1,i);
    [Max,Ind] = max(t2(:));
    [x1,y1] = ind2sub([size(t2,1), size(t2,2)], Ind);
    cam2(i,:) = [x1,y1];
end

frames31 = vidFrames3_1(213:350,:,:,:);
num3 = size(frames31,4);
cam3 = zeros(num3,2);
for i = 1:num3
    t3 = frames31(:,:,1,i);
    [Max,Ind] = max(t3(:));
    [x1,y1] = ind2sub([size(t3,1), size(t3,2)], Ind);
    cam3(i,:) = [x1,y1];
end

cam1x = cam1(:,1) - mean(cam1(:,1));
cam1y = cam1(:,2) - mean(cam1(:,2));
cam2x = cam2(:,1) - mean(cam2(:,1));
cam2y = cam2(:,2) - mean(cam2(:,2));
cam3x = cam3(:,1) - mean(cam3(:,1));
cam3y = cam3(:,2) - mean(cam3(:,2));

trimLen = min([length(cam1x), length(cam2x), length(cam3x)]);
trimCam1x = cam1x(1:trimLen);
trimCam1y = cam1y(1:trimLen);
trimCam2x = cam2x(1:trimLen);
trimCam2y = cam2y(1:trimLen);
trimCam3x = cam3x(1:trimLen);
trimCam3y = cam3y(1:trimLen);
projects = [trimCam1x trimCam1y trimCam2x trimCam2y trimCam3x trimCam3y]';
```

```
[U,S,V] = svd(projects, 'econ');
sig = diag(S);

figure()
plot(sig.^2/sum(sig.^2),'b--o','Linewidth',1)
xlabel('Principle Component'); ylabel('Diagonal Variance')
title('Ideal Case: Priciple Component v.s. Diagonal Variance')

comp = U' * projects;
comp = comp(2,:)
plot(1:trimLen, comp(:,:))
xlabel('Time (frames)'); ylabel('Displacement (pixels)')
title('Case 1 Principal Component Projections')

%% Test2
close all; clear all; clc
load('cam1_2.mat');
load('cam2_2.mat');
load('cam3_2.mat');

[a1, b1 , c1 , d1 ] = size(vidFrames12) ;
r1 = [];
for i = 1:d1
   graph = rgb2gray(vidFrames12(:,:,:,i));
   graph(:,1:250) = 0;
   graph(:,450:end) = 0;
   graph(1:200,:) = 0;
   graph(440:end,:) = 0;

   thre = graph(:) > 250;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r1 = [r1; mean(x), mean(y)];
end

[a2, b2 , c2 , d2 ] = size(vidFrames22);
r2 = [];
for i = 1:d2

   graph = rgb2gray(vidFrames22(:,:,:,i));

   graph(:,1:220) = 0;
   graph(:,400:end) = 0;
   graph(380:end,:) = 0;

   thre = graph(:) > 247;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r2 = [r2; mean(x), mean(y)];
end

[a3, b3 , c3 , d3] = size(vidFrames32);
r3 = [];
```

```
for i = 1:d3

   graph = rgb2gray(vidFrames32(:,:,:,i));

   graph(:,1:200) = 0;
   graph(:,500:end) = 0;
   graph(1:200,:) = 0;
   graph(340:end,:) = 0;

   thre = graph(:) > 250;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r3 = [r3; mean(x), mean(y)];
end

min_len = min([length(r1(:,1)), length(r2(:,1)), length(r3(:,1))]);

r1 = r1(1:min_len,:);
r2 = r2(1:min_len,:);
r3 = r3(1:min_len,:);

projects = [r1';r2';r3'];


[m,n]=size(projects);
mn=mean(projects,2);
projects=projects-repmat(mn,1,n);

[u,s,v]=svd(projects'/sqrt(n-1));
lam=diag(s).^2;

Y= projects' * v;
sig=diag(s);


figure()
plot(1:6, lam/sum(lam), 'bo--', 'Linewidth', 1);
title('Noisy Case: Principal Components v.s. Diagonal Variances');
xlabel('Principal Components'); ylabel('Diagonal Variances');

figure()
plot(1:min_len, Y(:,1),1:min_len, Y(:,2),'Linewidth', 1)
ylabel('Displacement (pixels)'); xlabel('Time (frames)');
title('Case 1 Principal Component Projections');
legend('PC1', 'PC2')
ylim([-200 150])

%% Test3
close all; clear all; clc
load('cam1_3.mat');
load('cam2_3.mat');
load('cam3_3.mat');

[a1, b1 , c1 , d1 ] = size(vidFrames13) ;
```

```
r1 = [];
for i = 1:d1
   graph = rgb2gray(vidFrames13(:,:,:,i));

   graph(:,1:250) = 0;
   graph(:,460:end) = 0;
   graph(1:212,:) = 0;
   graph(440:end,:) = 0;

   thre = graph(:) > 245;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r1 = [r1; mean(x), mean(y)];
end

[a2, b2 , c2 , c2 ] = size(vidFrames23);
r2 = [];
for i = 1:c2
   graph = rgb2gray(vidFrames23(:,:,:,i));

   graph(:,1:220) = 0;
   graph(:,450:end) = 0;
   graph(360:end,:) = 0;

   thre = graph(:) > 260;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r2 = [r2; mean(x), mean(y)];
end

[a3, b3 , c3 , d3 ] = size(vidFrames33);
r3 = [];
for i = 1:d3
   graph = rgb2gray(vidFrames33(:,:,:,i));

   graph(:,1:250) = 0;
   graph(:,500:end) = 0;
   graph(1:200,:) = 0;
   graph(338:end,:) = 0;

   thre = graph(:) > 248;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r3 = [r3; mean(x), mean(y)];
end


trimLen = min([length(r1(:,1)), length(r2(:,1)), length(r3(:,1))]);
r1 = r1(1:trimLen,:);
r2 = r2(1:trimLen,:);
r3 = r3(1:trimLen,:);
projects = [r1';r2';r3'];

[m,n]=size(projects);
```

```
mn=mean(projects,2);
projects=projects-repmat(mn,1,n);

[u,s,v]=svd(projects'/sqrt(n-1));
lam=diag(s).^2;

Y= projects' * v;
sig=diag(s);

figure()
plot(1:6, lam/sum(lam), 'bo--', 'Linewidth', 1);
title('Horizontal Displacement: Priciple Component v.s. Diagonal Variance');
xlabel('Principle Component'); ylabel('Diagonal Variance');

figure()

plot(1:237, Y(:,1), 1:237, Y(:,2), 1:237, Y(:,3), 'Linewidth', 2)
ylabel('Displacement (pixels)'); xlabel('Time (frames)');
title('Horizontal Displacement Principal Component Projections');
legend('PC1', 'PC2', 'PC3')


%% Test4
close all; clear all; clc
load('cam1_4.mat');
load('cam2_4.mat');
load('cam3_4.mat');

[a1, b1 , c1 , d1 ] = size(vidFrames14) ;
r1 = [];
for i = 1:d1
   graph = rgb2gray(vidFrames14(:,:,:,i));

   graph(:,1:330) = 0;
   graph(:,450:end) = 0;
   graph(1:200,:) = 0;
   graph(440:end,:) = 0;

   thre = graph(:) > 243;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r1 = [r1; mean(x), mean(y)];
end

[a2, b2 , c2 , d2 ] = size(vidFrames24);
r2 = [];
for i = 1:d2

   graph = rgb2gray(vidFrames24(:,:,:,i));

   graph(:,1:236) = 0;
   graph(:,420:end) = 0;
   graph(1:79,:) = 0;
   graph(380:end,:) = 0;
   thre = graph(:) > 247;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r2 = [r2; mean(x), mean(y)];
end

[a3, b3 , c3 , d3 ] = size(vidFrames34);
r3 = [];
for i = 1:d3
   graph = rgb2gray(vidFrames34(:,:,:,i));

   graph(:,1:260) = 0;
   graph(:,500:end) = 0;
   graph(1:150,:) = 0;
   graph(302:end,:) = 0;

   thre = graph(:) > 236;
   index = find(thre);
   [y, x] = ind2sub(size(graph),index);
   r3 = [r3; mean(x), mean(y)];
end

min_len = min([length(r1(:,1)), length(r2(:,1)), length(r3(:,1))]);

r1 = r1(1:min_len,:);
r2 = r2(1:min_len,:);
r3 = r3(1:min_len,:);
projects = [r1';r2';r3'];

[m,n]=size(projects);
mn=mean(projects,2);
projects=projects-repmat(mn,1,n);

[u,s,v]=svd(projects'/sqrt(n-1));
lam=diag(s).^2;

Y= projects' * v;
sig=diag(s);

figure()
plot(1:6, lam/sum(lam), 'bo--', 'Linewidth', 1);
title('Horizontal Displacement and Rotation: Priciple Component v.s. Diagonal Variance');
xlabel('Principle Component'); ylabel('Diagonal Variance');
ylim([0,0.63])

figure()

plot(1:min_len, Y(:,1), 1:min_len, Y(:,2), 1:min_len, Y(:,3), 'Linewidth', 1)
ylabel('Displacement (pixels)'); xlabel('Time (frames)');
title('Horizontal Displacement and Rotation Principal Component Projections');
legend('PC1', 'PC2', 'PC3')
```