AMATH 482 Homework 3:

Principle Component Analysis

Rose Ju

February 21, 2020

Abstract

The goal of the report is analysis the given data of the movie files using principal component analysis. The videos are composed of a hanging mass which was taken by cameras in three different directions. Four cases of the movie are introduced, including, ideal case, noisy case, horizontal displacement and horizontal displacement and rotation. I will be using singular value decomposition (SVD) and plotting the principal component along with the energy percentage to determine the dominant component.

I. Introduction and Overview

We are given data of videos from three angles of four different cases. The four cases include ideal case, noisy case, horizontal displacement and horizontal displacement and rotation. Ideal cases represent a small displacement of the mass in the z direction and ensuing oscillations. The entire motion is in the z direction with simple harmonic motion. Noisy case repeats the ideal case experiment but introduces camera shake into the recording which makes is difficult to extract the simple harmonic motion. Horizontal displacement case involves the mass being released off

center which produces motion in $x - y$ plane in additional to the z direction. Thus, there is both a pendulum motion and the harmonic oscillations. Lastly, the horizontal displacement and rotation case contains an off-center released mass along with rotations. Therefore, it has motion in the $x - y$ plane, motion in the z direction and rotation.

For all cases described above, I will be finding the displacements of the can from each camera angle and store the data into x and y coordinates. Then I will be constructing them into matrices in order to apply a singular value decomposition (SVD) which factorizes the matrix into constitutive components. Principle component analysis (PCA) will be used in this study and it is an application of SVD. From the PCA, I will be able to extract the expected results with sensible accuracies.

II. Theoretical Background

i. Singular Value Decomposition (SVD)

Singular Value decomposition splits a matrix into produces of three matrices, defined as

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

The columns of U and V are orthonormal and the matrix Σ is a diagonal matrix with positive real entries sorted from largest to smallest. By performing an SVD on matrix A, it first applied a rotation with V^T then stretches by the diagonal matrix Σ. Lastly, it rotates back using the orthonormal matrix U. The resulted matrix from SVD would reduce redundancy and find the maximal variance signal. By the properties of SVD, every matrix A has an SVD if the proper basis of the range and domain, U and V, are chosen. The resulted matrix will then be useful for simplified calculations.

ii. Principal Component Analysis (PCA)

Principal Component Analysis is an application of SVD. It produces an eigenvalue decomposition and reduces the dimensionality of a dataset while preserving the variabilities of the dataset. It helps scientists to increase the interpretabilities of a given data meanwhile minimizing information loss. PCA creates uncorrelated variables from the maximal variances. By finding these variables or principal components, the problem reduces to a simple eigenvalue problem. Therefore, PCA is a powerful method for researchers and analysts to explore a set of data and examine it with minimal efforts.

III. Algorithm Implementation and Development

- Load the video data files and for each frame, transforms the colored image into a grayscale so the size of the data is reduced which simplifies the calculation.
- Find the location interval of the moving object with `pcolor()`. Set the values outside of the interval pixels to be 0. Thus, all locations outside of the interval will be black.
- Search for maximum value within the location interval and save maximum point index
- Construct the corresponding x and y vectors from the indices for each frame by using `ind2sub()` and trim all vectors such that they have the same length
- Put all vectors into a new single matrix and make each row to be centered at 0.
- Performs SVD to save its U, Σ and V matrices
- Plot the diagonals of the Σ matrix divided by its diagonal sums, which represent the total energy percentage.
- Find the principle component, by multiplying Σ and V. Plots all columns in this matrix

IV. Computational Results

i. Ideal Case

Figure 1(a) shows the hanging object oscillating from three camera. From this ideal case with a stable camera, we are able to see that the can is moving up and down in a simple harmonic motion. Figure 1(b) shows a graph of the energy percentage for each principle component. We can see that the dominant component occurs in the first mode. The sum of the first two modes is about 90% of the energy in the motion.

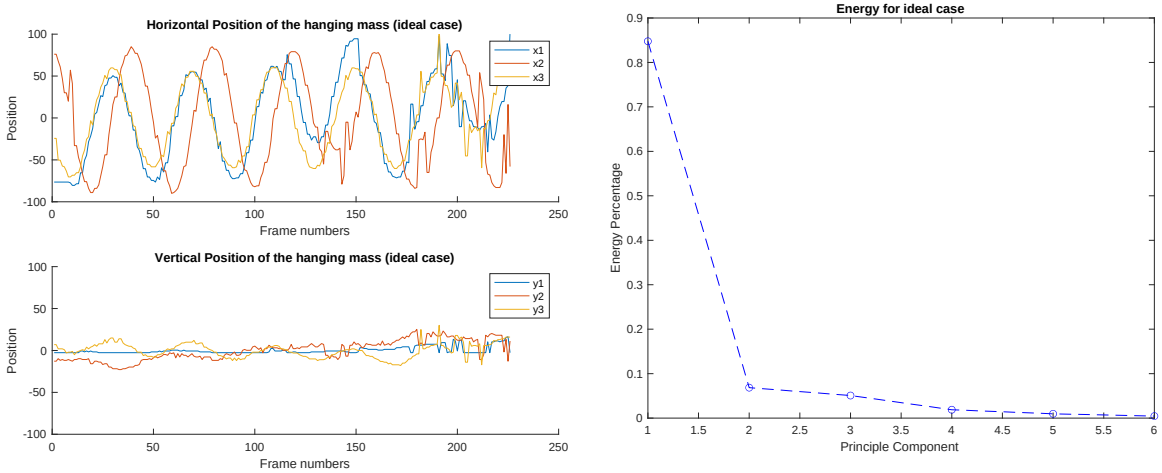

Figure 1: (a) Horizontal and Vertical Position per frame for Ideal Case (right)
(b) Percentage of energy for Ideal Case (left)

ii. Noisy Case

The data from the noisy case is a result of the shaking camera. Therefore, we can see from figure 2 (a) that there is a lot of noises in both horizontal and vertical position. Due to the shaking camera, it is difficult to extract a clear reading of object positions. However, we can see from the energy percentage graph in figure 2(b), the first two modes still makes up for more than 80% of the total energy in this case.

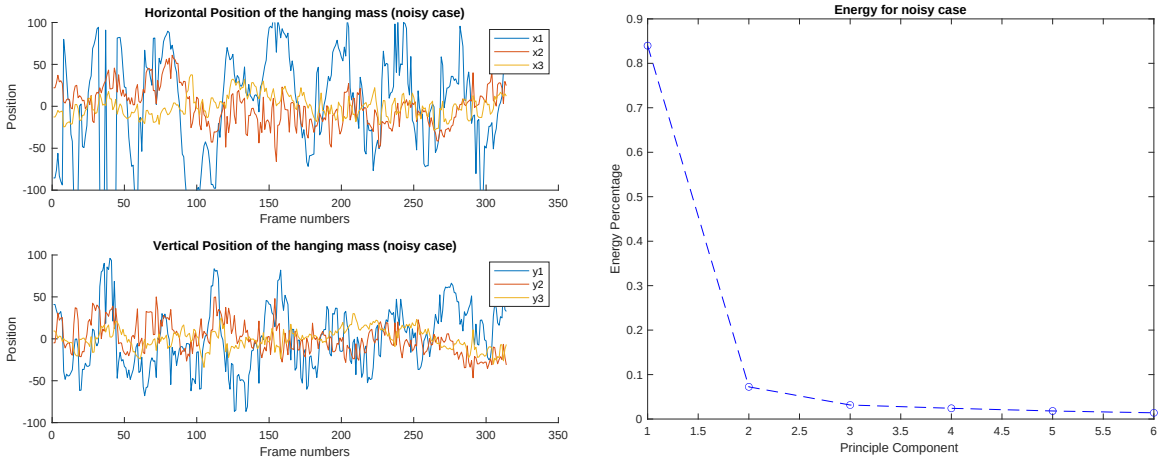

Figure 2: (a) Horizontal and Vertical Position per frame for Noisy Case (right)

(b) Percentage of energy for Noisy Case (left)

iii. Horizontal Case

In the horizontal case, the mass is released off-centered therefore, in additional to the horizontal and vertical motion, it also produced motion in the z-direction. In other words, the mass is producing a sample harmonic motion along with a pendulum motion. From Figure 3(a), we can see that $x1$ and $y1$ has a much bigger oscillation than the rest which are the dominating

components in the horizontal case. Supporting evidence from figure 3(b) also shows that the first two principal components of the horizonal case makes up for the majority of the data.

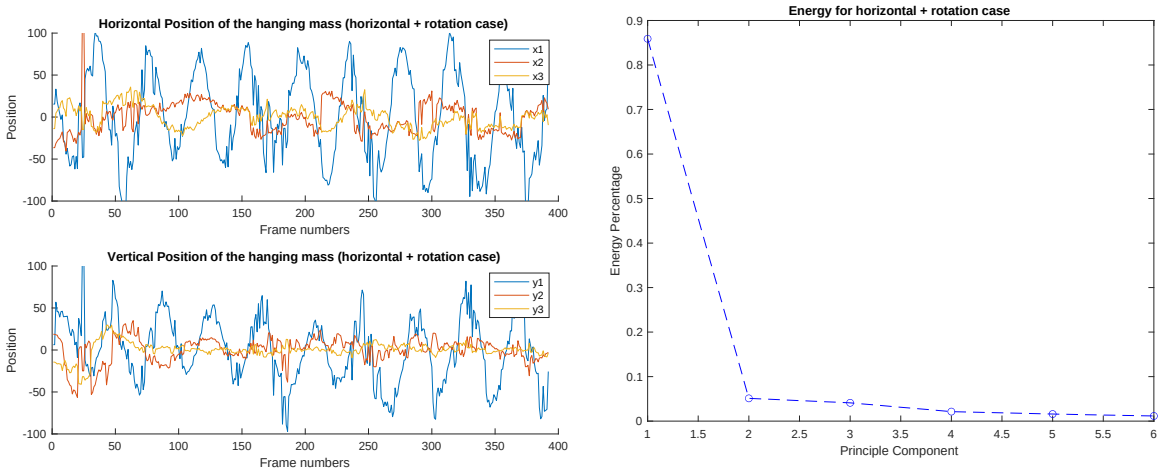

Figure 3: (a) Horizontal and Vertical Position per frame for Noisy Case (right)

(b) Percentage of energy for Noisy Case (left)

iv. Horizontal and Rotational Case

Figure 4 demonstrates a horizontal and rotational case of the hanging can. Similar to the previous case, the mass is released off-center. In additional, the mass is also set to be rotating in this study. Therefore, it produces a simple harmonic motion in the vertical and horizontal direction and rotational motion in the z-direction. From figure 4(a) we can see that the dominating components are still coming from the first two components ($x1$ and $y1$). They sum up to be almost 90% of the total energy.

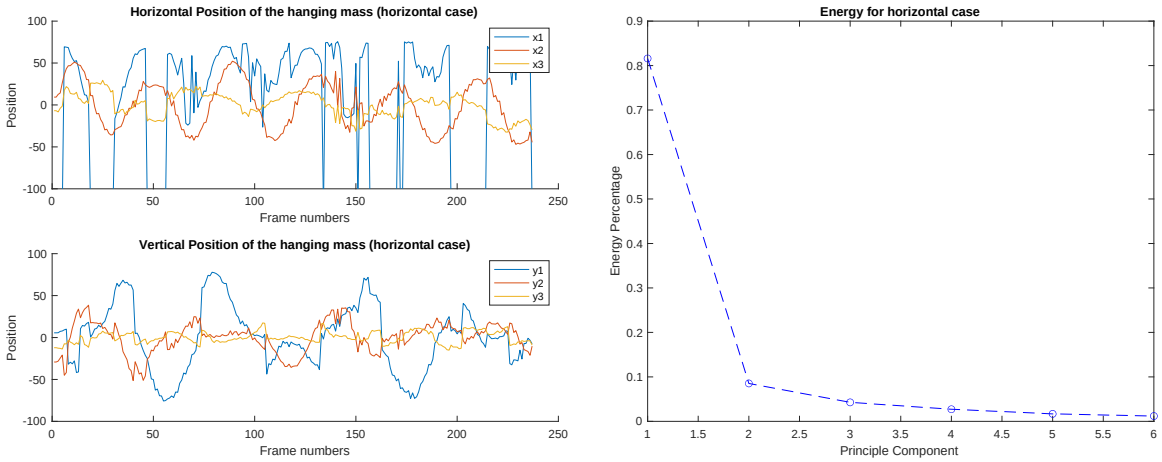

Figure 4: (a) Horizontal and Vertical Position per frame for Noisy Case (right)

(b) Percentage of energy for Noisy Case (left)

V. Summary and Conclusions

In this report, we have explored the principle component analysis method for analyzing data. We can see from the results that the majority of the original data fits within the expectation of going through PCA. We can see that top two principle components of the 6 extracted vector values seems to produce the matching image of our original data set. In additional to applying PCA, we were able to remove excessive data and therefore obtains a reduced data with similar accuracy. However, the data after performing PCA does lose some degrees of accuracy which could be critical for some cases. In the further, analysts should be deciding carefully the tradeoff for storage space with loss of accuracy.

VI. Appendix A – MATLAB functions

`ind2sub([],i)` – given index and transforms the index into indices of the given matrix. This was used to determine the $x\ y$ coordinates after finding the maximum.

`load` – load data from MAT-file into the tensor. We used this to load the dataset into our workspace.

`max(X)` – calculates the maximum value of the input element. The function was used to find the maximum of the dataset

`svd(A)` - performs a singular value decomposition to the $x\ y$ matrix.

`plot()` - plot the dataset of displacements of each cases and their energy percentages.

VII. Appendix B – MATLAB codes

```
close all; clear all; clc;

%% load data and basic setup
for i = 1:3
    for j = 1:4
        load(['cam' num2str(i) '_' num2str(j) '.mat']);
    end
end

n11 = size(vidFrames1_1,4);
n21 = size(vidFrames2_1,4);
n31 = size(vidFrames3_1,4);
n12 = size(vidFrames1_2,4);
n22 = size(vidFrames2_2,4);
n32 = size(vidFrames3_2,4);
n13 = size(vidFrames1_3,4);
n23 = size(vidFrames2_3,4);
n33 = size(vidFrames3_3,4);
n14 = size(vidFrames1_4,4);
n24 = size(vidFrames2_4,4);
n34 = size(vidFrames3_4,4);

%% ideal
close all;
x1 = [];
y1 = [];
for i = 1:n11
    X = rgb2gray(vidFrames1_1(:,:,:,i));
    X(1:200,:) = 0;
    X(:,1:320) = 0;
    X(:,480:end) = 0;
    [~, I] = max(X(:));
    [x,y] = ind2sub(size(X),I);
    x1 = [x1 x];
    y1 = [y1 y];
end

x2 = [];
y2 = [];
for i = 1:n21
    X = rgb2gray(vidFrames2_1(:,:,:,i));
    X(1:50,:) = 0;
    X(300:end,:) = 0;
    X(:,1:260) = 0;
    X(:,330:end) = 0;
    [~, I] = max(X(:));
    [x,y] = ind2sub(size(X),I);
    x2 = [x2 x];
    y2 = [y2 y];
end

x3 = [];
y3 = [];
for i = 1:n31
    X = rgb2gray(vidFrames3_1(:,:,:,i));
    X(1:200,:) = 0;
    X(350:end,:) = 0;
    X(:,1:200) = 0;
    X(:,480:end) = 0;
```

```
    [~, I] = max(X(:));
    [y,x] = ind2sub(size(X),I);
    x3 = [x3 x];
    y3 = [y3 y];
end

n = min([length(y1), length(y2), length(y3)]);
x1 = x1(:,1:n);
y1 = y1(:,1:n);
x2 = x2(:,1:n);
y2 = y2(:,1:n);
x3 = x3(:,1:n);
y3 = y3(:,1:n);
mat = [x1; y1; x2; y2; x3; y3];
[M,N] = size(mat);
[U,S,V] = svd(mat);
figure(1)
plot(diag(S)./sum(diag(S)),'bo--')
xlabel('Principle Component')
ylabel('Energy Percentage');
title('Energy for ideal case')

[M,N] = size(mat);
mat = mat - repmat(mean(mat,2),1,N);
[U,S,V] = svd(mat);
svdmat = U*S*V';
x1 = svdmat(1,:);
y1 = svdmat(2,:);
x2 = svdmat(3,:);
y2 = svdmat(4,:);
x3 = svdmat(5,:);
y3 = svdmat(6,:);

figure(2)
subplot(2,1,1)
hold on
plot(x1);
plot(x2);
plot(x3);
hold off
ylim([-100, 100])
xlabel('Frame numbers')
ylabel('Position')
title('Horizontal Position of the hanging mass (ideal case)')
legend('x1','x2','x3');

subplot(2,1,2)
hold on
plot(y1);
plot(y2);
plot(y3);
hold off
ylim([-100, 100])
xlabel('Frame numbers')
ylabel('Position')
title('Vertical Position of the hanging mass (ideal case)')
legend('y1','y2','y3');
```