AMATH 482 Homework 3

Rishabh Verma

February 24th, 2021

Abstract

An object undergoing simple harmonic motion is filmed by three cameras in subpar lab conditions. The method of principal component analysis is used to isolate the simple harmonic motion from other factors, particularly noise and independent movement in extraneous dimensions. Principal component analysis is found to be an effective technique to identify patterns of variance in multi-dimensional data and isolate the most interesting feature.

1 Introduction and Overview

Consider a simple harmonic oscillator formed by a mass fixed to a spring. Under ideal circumstances, subjecting the mass to a non-zero initial velocity or non-zero initial displacement will cause the mass-spring system to undergo simple harmonic motion along the dimension given by the spring's orientation.

We can capture video data of the oscillator to study it, presenting the task of analyzing 1-D motion from 2-D videos in 3-D space. Sources of error may include:

- The camera may be oriented in space such that the spring's dimension of motion does not perfectly match the camera's x-dimension or y-dimension.
- An imperfectly aligned initial setting causing extraneous motion along other dimensions. To make things more complicated, this dimension would be invisible to the camera if it is aligned with the camera's normal, suggesting the need for multiple cameras.

In order to develop a method to examine the ideal case, it is necessary to test this method and see how well it performs under extreme non-ideal conditions.

Data is captured from 4 experiments. In each experiment, an oscillating mass-spring system is videoed by three observers holding video cameras in radially different locations. Two of the observers hold their cameras vertically, and one observer holds their camera at a rotated angle. The resulting video data is manually synced to the frame and converted to greyscale for analysis.

The four experiments are respectively subject to the following circumstances

1. Control case, with no difficulties added to the ideal conditions
2. Shaky camerawork, to examine how well the developed procedure performs with significant noise.
3. Horizontal displacement, to examine how well the developed procedure performs with significant extraneous motion.
4. Horizontal displacement and rotation of the spring mass, to examine how well the developed procedure deals with an object that may be difficult to track from frame to frame.

2 Theoretical Background

2.1 PCA and the SVD

The Singular Value Decomposition (SVD) is a remarkable result from linear algebra that characterizes linear transformations. It is well understood that a linear transformation between two vector spaces can be encapsulated in a matrix A. The SVD tells us that under the transformation $T(x) = Ax$, the unit sphere in the domain vector space is transformed (i.e. rotated and stretched) to a hyperellipse in the codomain vector space. The SVD provides that for all matrices $A \in \mathbb{C}^{n\times m}$, there exists a decomposition

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

where Σ is a diagonal $n \times m$ matrix and U and V are unitary, so their columns can form an orthogonal basis. Left-multiplication of a vector x by V^* gives a re-expression in an orthogonal basis, $[x]_V$. Left-multiplication of this by Σ scales each orthogonal basis vector by a constant σ, causing the sphere to stretch into a hyperellipse. Left-multiplication by U then rotates the hyperellipse into the proper orientation.

In Principal Component Analysis, we seek to analyze the relationship between joint random variables in a set of observations. Each observation is inserted as a row in a matrix A. Suppose we have n observations of k real random variables. If $n > k$ (as would be practical), then A is a tall skinny $n \times k$ matrix. The corresponding linear transformation is from $\mathbb{R}^k \to \mathbb{R}^n$. Assuming that the data is not perfectly linear and A has rank k, the SVD identifies an orthogonal basis in $\mathbb{R}^k$ which is sent to a k-d hyperellipse in $\mathbb{R}^n$. The k basis vectors for $\mathbb{R}^k$ form what are called the principal components of the data, and the radius of the hyperellipse along one of its axes is proportional to the variance along that principal component. The principal components corresponding to the greatest variance characterize changes in the data; independent relationships between the random variables.

3 Algorithm Implementation and Development

There are two independent parts. The first algorithm gets the (x, y) coordinates of the oscillating mass in each video and stores them in a vector. The second algorithm is simply Principal Component Analysis.

Let n be the number of frames in the video. Let c be the number of cameras recording.

Algorithm 1 requires the initial location (x_1, y_1) of the bucket from the perspective of each camera.

Once the location of the oscillating mass has been recorded for each frame, it can be passed into Algorithm 2 to isolate simple harmonic motion. The resulting matrix $V^\top X$ contains in the ij-th entry the result of projecting the measured location in frame j onto the i-th basis vector. Since the basis vectors have norm 1, this entry can be regarded as a projection coefficient by which to multiply the basis vector.

Algorithm 1: Retrieve mass locations from video data

```
for all j=2:n do
  for all k=1:c do
    Take the $j$-th frame and apply a 2-D Gaussian Filter centered at $(x_{j-1_k}, y_{j-1_k})$
    Store $(x_{j_k}, y_{j_k}) \leftarrow$ the coordinates of the max value in the filtered frame
  end for
end for
return the resulting matrix containing $(x_j, y_j)$ for each camera at all $j = 1 : n$
```

Algorithm 2: Principal Component Analysis and projection

```
Update each row in $X$ so that each row is centered about its mean
Compute the Singular Value Decomposition $U\Sigma V^\top$ for $\frac{1}{\sqrt{n-1}}X^\top$
return $V^\top X$
```

4 Computational Results

It has been experimentally verified that the i-th singular value is indeed equal to the variance of the i-th set of projection coefficients.

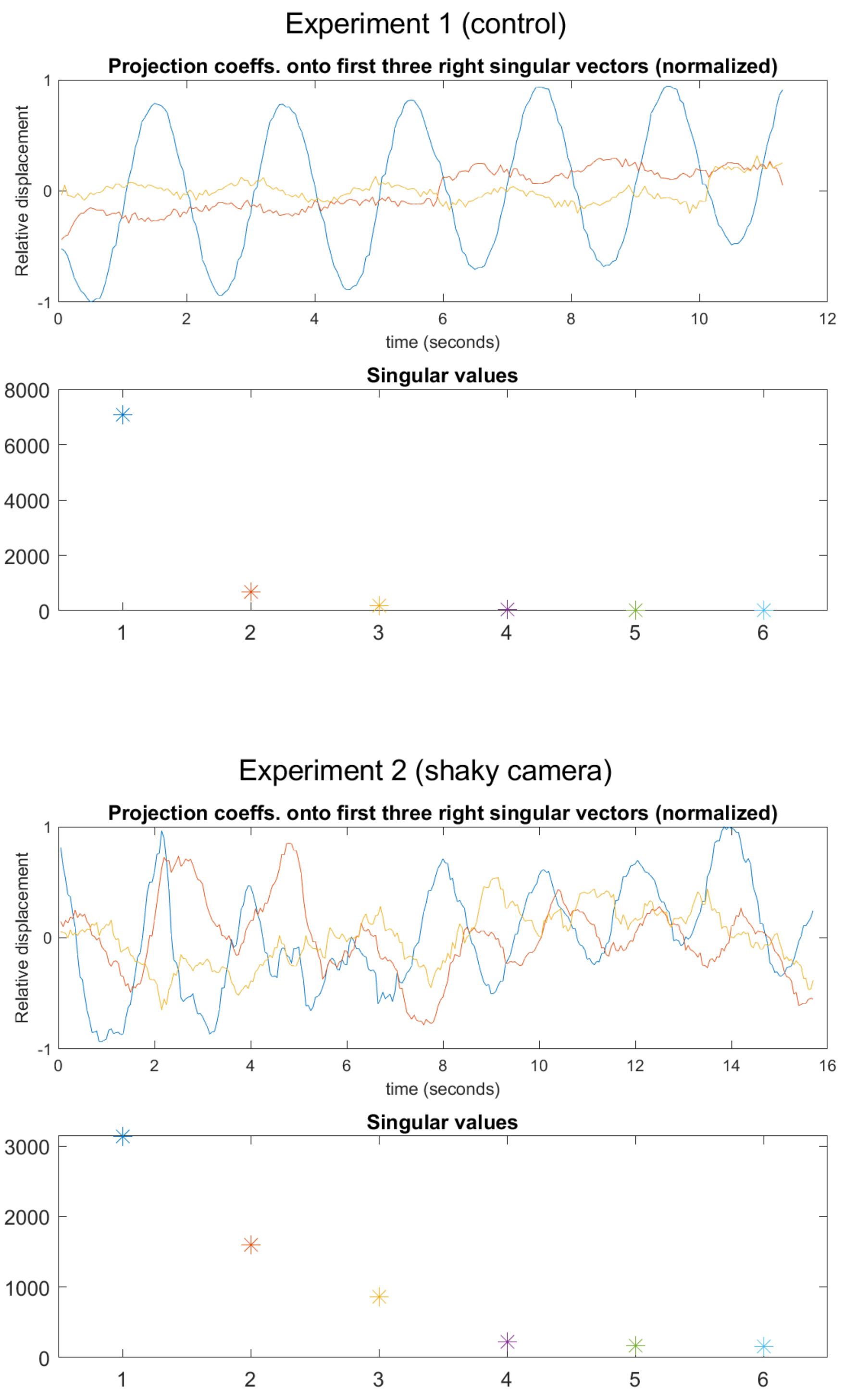
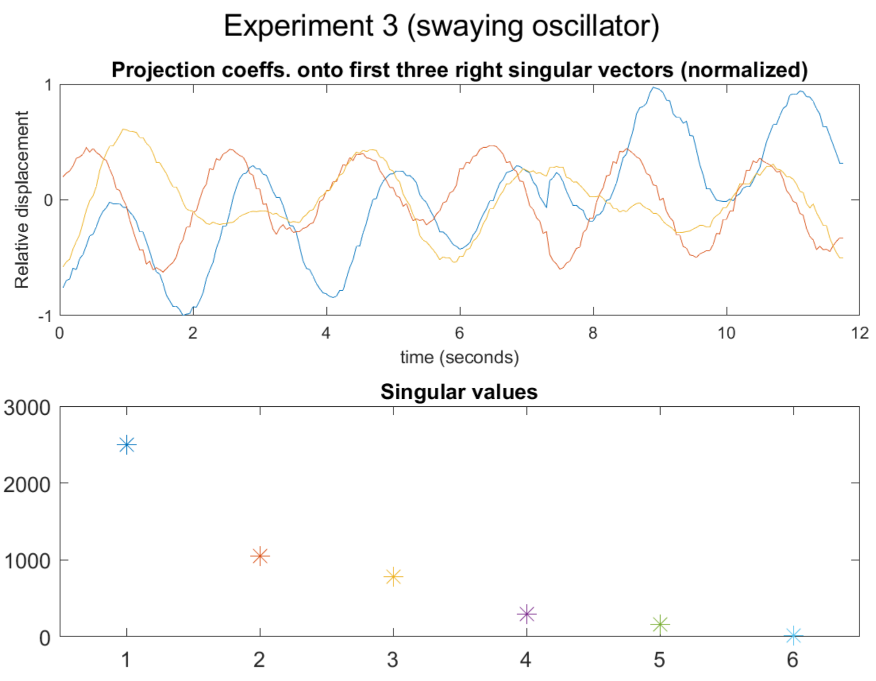


Figure 1: Results of experiments 1 and 2



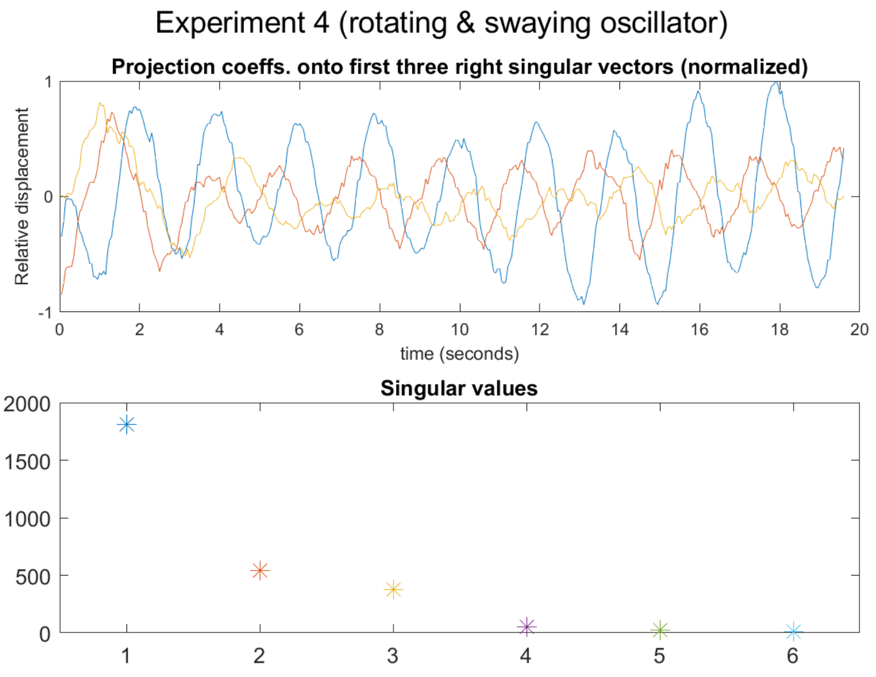


Figure 2: Results of experiments 3 and 4

The first right singular vectors for each of the four experiments are

$$\begin{bmatrix}0.0008\\0.5600\\0.0068\\0.6739\\0.4788\\0.0547\end{bmatrix}, \begin{bmatrix}0.0176\\-0.3494\\0.2758\\-0.8822\\-0.1516\\0.0146\end{bmatrix}, \begin{bmatrix}0.0710\\0.4762\\0.2341\\0.8352\\0.0151\\-0.1250\end{bmatrix}, \begin{bmatrix}-0.03335\\0.4645\\-0.0030\\0.8848\\0.0150\\-0.0046\end{bmatrix} \tag{2}$$

4.1 Results and interpretation of each experiment

For all graphs, the result of projecting the data onto the first three right singular vectors has been displayed. The last three singular vectors correspond to very small singular values in each experiment, and the resulting projection would clutter the graph.

Experiment 1 is the control experiment performed under ideal circumstances. The first singular value is much larger than the others, suggesting that the data varies in only one dimension. Though this projection appears sinusoidal, its average displacement seems to wander from the mean of 0. One possible explanation is that the location-tracking algorithm latched on to a different part of the bucket. With data captured over a longer window, Algorithm 1 might capture more reliable data.

Experiment 2 involves shaky cameras. The second and third singular values are more significant than in any other experiment, suggesting that a high degree of variance was captured. The method of PCA identifies the components with the greatest variance, and it is possible that the noise presented too much noise for PCA to deal with. It is also possible that Algorithm 1 was unable to perform with so much noise. This experiment is particularly susceptible to experimental error; the initial frames were synchronized manually, and this may have been done improperly since it is difficult to synchronize noisy video data without a clock recorded.

Experiment 3 involves a oscillator with additional sway in the other two dimensions, and these dimensions appear to have been captured well. The second and third singular values are of comparable magnitude, and the projection coefficients appear to form independent sinusoids. This corresponds well with the actual situation of an object oscillating in the vertical dimension with spring-mass behavior, and swaying in the other two horizontal dimensions with pendular behavior.

Experiment 4 is very much like experiment 3 except the mass is also rotating (and the data is captured for a bit longer). Since the angular dynamics in a spring-mass system or a pendulum are virtually independent of translational motion (looking at you, Coriolis force), the oscillation in each dimension should be unaffected. If the location-tracking algorithm is ideal, we should expect similar results to experiment 3, which is fortunately what we see.

5 Summary and Conclusions

In all experiments, the first singular value is dominant, the second and third singular values are lesser yet still significant, and the last three singular values are negligible. The question arises, "are the three significant singular values an artifact of using three cameras?" I think the dominant singular vector from experiment 1 in result 2 provides evidence against this question since the SVD was able to identify the main sources of variance occurred in y_1, y_2, x_3. This makes sense because cameras 1 and 2 recorded the motion vertically, and camera 3 recorded the motion horizontally.

The orientation of the cameras is the same for all four experiments, yet the dominant singular vectors of experiments do not all capture this. I should expect y_1, y_2, x_3 to be dominant in all singular vectors. This is true for those resulting from experiments 1 and 2. In experiment 3, only y_1 and y_2 seem to dominate the variance, and so the SVD was unable to discern much of use from the third camera. It is possible that the data from the third camera was simply not useful for capturing the mass-spring oscillation. It is also possible that the third camera was particularly susceptible to error arising from Algorithm 1.

The fact that experiment 2 performs so poorly suggests that either Algorithm 1 does not deal well with noisy videos, or that the SVD does not deal well with noisy data. Because the SVD is a statistical method–it identifies independent axes of variance–it makes sense that noise manifests as high variance. It is possible that with larger samples, the noise will be averaged out, so further experiments may want to test how well the SVD performs when dealing with a larger sample of noisy data.

Appendix A MATLAB Functions

- `BW = rgbvid2grayvid(RGB)` inputs an RGB video represented as a matrix of dimensions `height, width, 3, num_of_frames` and outputs the result of calling `rgb2gray` on each frame.
- `[U,S,V] = svd(X)` returns the Singular Value Decomposition such that `U*S*V'=X`

Appendix B MATLAB Code

B.1 main.m

```matlab
%% Read experiment 1 data
clear all;

experiment = 1;
a = (150)^2;  % controls width of gaussian tunnel vision while tracking

% Load the data
for cam = 1:3
    filename = strcat('cam', string(cam), '_', string(experiment), '.mat');
    load(filename)
end

% Set the starting parameters
switch experiment
    case 1
        start1=1; start2=10; start3=1;
        init = [228 324 179 274 270 320];
        v1 = vidFrames1_1; v2 = vidFrames2_1; v3 = vidFrames3_1;
        exp_title = "Experiment 1 (control)";
    case 2
        start1=1; start2=18; start3=5;
        init = [306 324 125 369 245 363];
        v1 = vidFrames1_2; v2 = vidFrames2_2; v3 = vidFrames3_2;
        exp_title = "Experiment 2 (shaky camera)";
    case 3
        start1=5; start2=27; start3=1;
        init = [283 320 190 336 233 354];
        v1 = vidFrames1_3; v2 = vidFrames2_3; v3 = vidFrames3_3;
        exp_title = "Experiment 3 (swaying oscillator)";
    case 4
        start1=1; start2=2; start3=3;
        init = [273 377 247 241 352 222];
        v1 = vidFrames1_4; v2 = vidFrames2_4; v3 = vidFrames3_4;
        exp_title = "Experiment 4 (rotating & swaying oscillator)";
    otherwise
        disp("Invalid experiment id");
        return;
end

% Trim the data, convert to bw
v1 = v1(:,:,:,start1:end);
v2 = v2(:,:,:,start2:end);
v3 = v3(:,:,:,start3:end);
```

```
num_frames = min([size(v1,4) size(v2,4) size(v3,4)]);

v1 = v1(:,:,:,1:num_frames);
v2 = v2(:,:,:,1:num_frames);
v3 = v3(:,:,:,1:num_frames);

% Now we can convert to bw
bw1 = rgbvid2grayvid(v1);
bw2 = rgbvid2grayvid(v2);
bw3 = rgbvid2grayvid(v3);
bw = cat(4, bw1, bw2, bw3);

clear filename;
clear vidFrames* v1 v2 v3;
clear bw1 bw2 bw3;
clear start*;
%%

frame_size = [size(bw, 1) size(bw, 2)];
big_gauss = ones(frame_size*2);
gauss_f = @(x) exp(-1/a*x^2);
xmax = frame_size(1);
ymax = frame_size(2);
for i = 1:2*xmax
    for j = 1:2*ymax
        big_gauss(i,j) = gauss_f(i-xmax)*gauss_f(j-ymax);
    end
end

location = zeros(6, num_frames);
location(:,1) = init;

for j = 2:num_frames
    for cam = 1:3
        frame = double(bw(:,:,j,cam));
        x0 = location(cam*2-1, j-1);
        y0 = location(cam*2, j-1);
        cone = big_gauss((481-x0):(960-x0), (641-y0):(1280-y0));
        frame = frame .* cone;

        [M, I] = max(frame(:));
        [x, y] = ind2sub(frame_size, I);
        location(cam*2-1, j) = x;
        location(cam*2, j) = y;
    end
end

clear i j big_gauss cam cone frame M I;
clear x0 y0 x y xmax ymax gauss_f;
%%
% location is [y1 x1 y2 x2 y3 x3]'
% let's make X [x1 y1 x2 y2 x3 y3]'
X = [location(2,:)
```

```
    location(1,:)
    location(4,:)
    location(3,:)
    location(6,:)
    location(5,:)];

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

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

% coefficients of projection onto first component
figure(1);
Y = v'*X;
Y = Y / max(abs(Y), [], 'all');
figure(1)
subplot(2,1,1);
plot((1:n)/20, Y(1,:), ...
     (1:n)/20, Y(2,:), ...
     (1:n)/20, Y(3,:) );%, ...
     %(1:n)/20, Y(4,:), ...
     %(1:n)/20, Y(5,:), ...
     %(1:n)/20, Y(6,:));
xlabel("time (seconds)");
ylabel("Relative displacement");
set(gca, 'ytick', -1:1);
title("Projection coeffs. onto first three right singular vectors (normalized)", "FontSize", 13);
subplot(2,1,2);
plot(1, lambda(1), '*', ...
        2, lambda(2), '*', ...
        3, lambda(3), '*', ...
        4, lambda(4), '*', ...
        5, lambda(5), '*', ...
        6, lambda(6), '*', 'MarkerSize', 12);
set(gca, 'xtick',1:6, 'FontSize', 13);
xlim([0.5 6.5]);
title("Singular values", 'FontSize', 13);

sgtitle(exp_title, 'FontSize', 18);

%saveas(gcf, strcat('fig', string(experiment), '.png'));
```

B.2 rgbvid2grayvid.m

```
function [I] = rgbvid2grayvid(RGB)
%   Convert an RGB video (4-D array) to a grayscale video frame-by-frame
    sz = size(RGB);
    bw_sz = [sz(1) sz(2) sz(4)];
    I = uint8(zeros(bw_sz));
    for j = 1:sz(4)
        I(:,:,j) = rgb2gray(RGB(:,:,:,j));
    end
```

```
end
```