MATLAB: FINAL PROJECT

TEAM NAME: DAAB

Team 22
6/1/2021
Angelina Licos, Derek Tang, Brady Muramoto, Aditya Honap

3. Final Project Report

3.1 Abstract

The purpose of this project was to create an analytical program that provided insight about football punts, using five different figures. Throughout this project, we learned many different MATLAB functions and tools in order to help us create a successful code.

3.2 Nomenclature

data	The inputted data should have three columns with distance (yards), height (yards), and the horizontal angle (degrees) of the punts respectively. There should be enough data for 100 punt trials.
predata	Predata is the transposed matrix of the inputted data. Distance (yards), height (yards), and the horizontal angle (degrees) of the punts should be represented by the first, second, and third rows
i	This is the indexing variable for all of the for-loops in the code
r1	The number of rows in data array, which should be 100 for the 100 punt trials
c1	The number of columns in the data array, which should be 3 for the 3 columns of distance, height and angle.
d	Creates a vector of all the distances in data
h	Creates a vector of all the heights in data
a	This is a matrix of the calculation (h/(d/2)^2)
b	This is a matrix of the calculation (d/2)
c	This is an equivalent variable to h
para	Creates an array that inputs all the generated functions of the parabolas into each row in the form ‘@(x) [function]’. It incorporates the variables a,b,c.
length	Determines the max distance of the inputted punt trials and adds 5. This is for plotting purposes to ensure that all the parabola trajectories are completely plotted
paravalx	Uses the functions of the parabolas described in para and creates an array of x-values associated with each function. Each cell contains its separate vector. (An array inside an array)
paravalz	Uses the functions of the parabolas described in para and creates an

	array of y-values associated with each function. Each cell contains its separate vector. (An array inside an array)
valuecount	An array that determines the dimensions of each of the sets of x and y described in paravalx and paravalz (for plotting purposes)
funcs	An array that plots each of the parabola x,y,z variables (Note: the y-value are an array of 0’s with the size outlined by valuecount)
direction	A vector of [0,0,1] which is meant to allow the parabolas to be rotated along the z direction.
curvelength	This creates a vector that incorporates all the distances outlined in data
curveangle	This creates a vector that incorporates all the angles outlined in data
tanangle	Calculates all the tangent values of the angles outlined in curveangle and the distance outlined in curvelength
widthlength	Multiples the curvelength and the tanangle in order to finish the trigonometric calculation
widthplot	Plots all of the landing positions of the punts on one axis along from the perspective of the end zone of the football field.
category1	Number of times there was a distance between 30 and 40
category2	Number of times there was a distance between 40 and 50
category3	Number of times there was a distance between 50 and 60
category4	Number of times there was a distance between 60 and 70
val	Taking each distance value from the data one by one for each iteration of the for loop.
catVec	Taking all of the values of the categories (category1 - category4) and putting them into vector form.
ang	The vertical angle of the parabola in degrees
v	The initial velocity of a kick given height and distance
t	Time of flight of the ball given height and distance

3.2 Introduction

The ultimate goal of this project was to design a code that could generate at least five figures using one hundred data points.

This project was conducted to generate analytics about football punts for competitive purposes. Using data such as the height, distance, and angle of the punt, the overall trajectory of each of the punts can be determined and modeled in 3-D plots. Information such as the landing position and predicted distance can be used to give significant advantages to the offensive team. Our final code creates five distinct figures that provide insight on the punts, given the correct inputs.

3.3 Background of the project

In American football, one of the most important aspects of the game is special teams. Special teams relates to kicking and kick-receiving the ball after a fourth down or after scoring. The program we made is designed to improve NFL analytics as it analyzes the punts of kickers. Teams can use this to predict where kickers normally punt and how long the ball stays in the air. Strategies can be developed in order to properly counter the trends of specific kickers. Teams can also use the information to improve their own punting.

3.4 Brief Overview of the Solution Using MATLAB

Our team created a program in MATLAB that takes the heights, distances, and angles (horizontally) of punts and analyzes various aspects about them. With these three points, we can map out the approximate trajectory of the football in 2D and 3D space, show how often a kicker kicks to the left or right, analyze how often a kicker kicks between certain distances, and determine the optimal kick to maximize time of flight.

This program will help show a kicker’s tendencies when the user inputs these data points for a given kicker. Once they input these stats, the program will return useful information that will help teams plan how to play punts and how they should be positioned to improve their chances of a better kick return. When the user inputs information about one of their own kickers, they can use the information to show their kicker the optimal kick, which is generally the kick with the most time of flight so the kicking team has enough time to catch up to the kick. This will reduce the amount of time and space the kick returner has to make a move, and therefore, reduce how far the kick returner runs the ball.

As we worked through our project, we made a few adjustments to make our data more realistic, we generated graphs with different vertical angles, and we removed and added different figures in order to better suit our intended user. Our intended users are NFL coaches (or any coaches for that matter with access to data on opposing kicker’s statistics).

3.5 Data Collection/Acquisition

We created a set of data based off of real data from the top American punters. On average, the distance that punters punt is around 40-50 yards, whether it is through a field goal or across the field to the opposing team. In order to emulate this, our distances range from 30 to 70, and were randomly generated. Punters' kicks reach heights of around 28.33 yards. Our heights in the data table are random values that range from 23 to 33 yards. Our angles range from -20 to 20.

3.6 Data Analysis

We used randomized datasets while being in a reasonable range, and used up to 100 different punts of height, distance, and angle to analyze our data. The large sample size also helped in the stable analysis. These are important in the long term because a larger set of data to analyze will yield better and more accurate results, so that is why it is important and useful.

When we determined the important figures, we took into consideration what was important to football coaches and football stat analysts. The most important ones are the trajectory/location of the kicks and time of flight, and each of our five figures analyzes these valuable statistics.

3.7 Solution and Demonstration

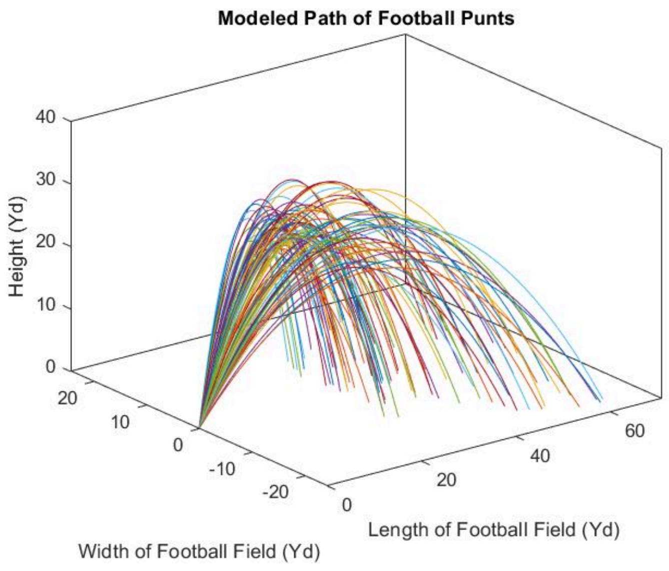

Figure 1: Modeled Path of Football Punts

Using the vertex formula of parabolas, the trajectory of each of the football punts was graphed on a 3D plot. All of these data sets include the correct height, distance and angle inputted in the code. This figure allows the user to fully visualize all of the inputted punts of a specific kicker or football team.

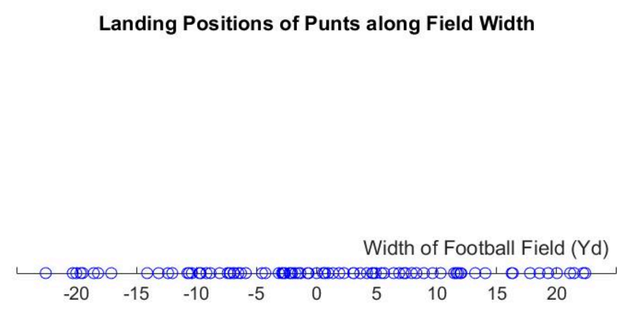

Figure 2: Landing Positions of Punts along Field Width

The landing position of the punts was plotted against the width of the football field as blue circles. This could prove beneficial in determining which side the punt will land on. The specific landing positions were calculated using trigonometric functions from the given distance inputs.

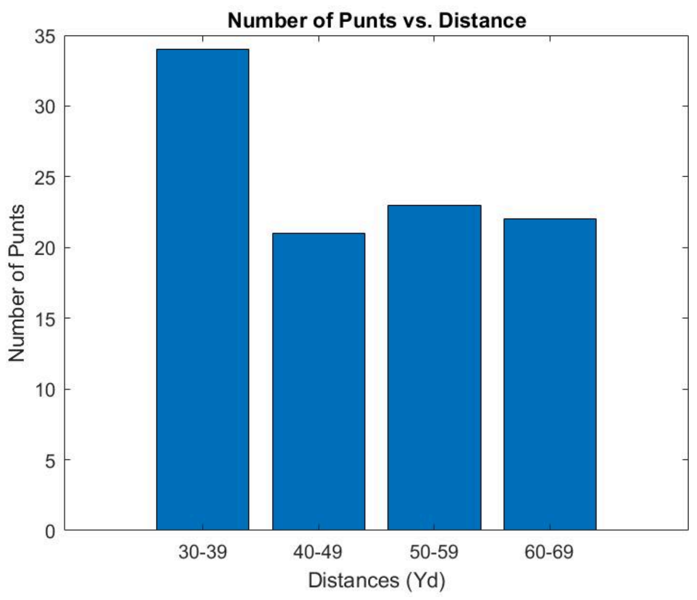

Figure 3: Number of Punts vs. Distance

Using a histogram, each of the punts were categorized into specific ranges based on distances. Similar to Figure 2, distance of the punts can provide insight to the receiving team for offensive strategies. This was completed using a ‘for loop’.

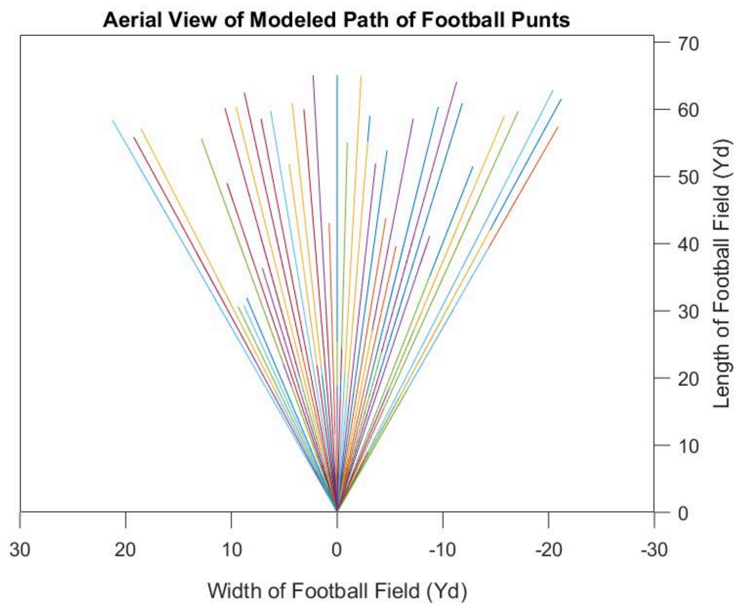

Figure 4: Aerial View of the Modeled Path of Football Punts

This figure is very similar to Figure 1. However, this displays an aerial view of all the punts. This might influence the receiving team to change the layout of their players. The plot was generated using the campos function to fix the perspective of the user.

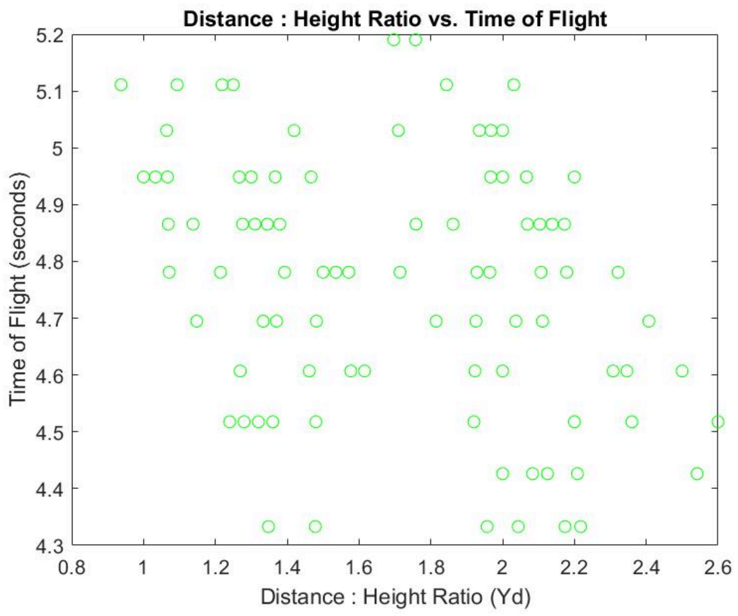

Figure 5: Distance:Height Ratio vs. Time of Flight

This figure shows how long the ball stays in the air for a given distance to height ratio. When the ratio is higher (when x is higher), the general trend is that time of flight decreases. This scatter plot indicated the optimal kick for certain kickers.

3.8 Technical Lessons Learned

We learned the importance and ease of plotting and creating figures. The MATLAB style of figures and charts help the analysis and visualization in real -life situations as well. In this project, we worked a lot with the plotting function in MATLAB. We made trajectory graphs, scatter plots, line plots, and bar graphs for our audience to better visualize our data. ‘campos’ was another really important function that reoriented the view of our 3D graph in order to give the user a birds eye view of the ball’s distance. In order to print the individual graphs in the 3D trajectory graph, we use ‘sprintf’, and the function ‘zeros’ made a matrix of zeros that we later filled with our own data. For loops helped condense and simplify our code for a specified number of runs.

3.9 Professional Lessons Learned

Our team worked very well together from the beginning. We had a fluid work strategy and divided the work evenly amongst each other. The most effective form of delegation was through dividing the figures. Since the entire code was based off of the data, we each made respective figures that matched our skill levels. If we were to do this project again, we would have finished a lot more of the work in the earlier weeks. That way, we would have much more time to debug in the later weeks. Matlab is a professional tool in itself, so learning how to operate the main functions of the application was beneficial for all of our professional careers. Many companies use matlab for analysis of big data, so having a grasp on the tool can set us apart from other applicants.

3.10 Summary and Conclusion

Overall, each of us learned the basics of the matlab programming language. By dividing up the work, we were able to learn from each other’s code and enhance our understanding of the application. Much of our code was made up of iterations and for loops, so our team holistically has a deeper comprehension mainly of this function. We learned how to communicate effectively with the team in order to finish the project in a timely manner. Our project takes 300 values (100 distance values, 100 height values, 100 angles), and analyzes them in five separate, unique figures. Each figure has an intended visual purpose for football players and coaches alike to understand the statistics of punters. By creating our code, our team was able to learn the foundational concepts behind matlab.

Appendix A

```
%% Figure 1
figure(1)
predata = data.';       %transpose the data mat
                    %data with first row=distance, second row=height, third row=angle

[r1,c1] = size(data);   %determining the size of data mat
para = cell(r1,1);      %creating an empty matrix for housing the functions

for i = 1:r1            %This is for generating the functions of parabolas
```

```
    d = predata(1,i);   %first row, distance
    h = predata(2,i);   %second row, height
    a = (h/(d/2)^2);    %coeff of x^2 in parabola function
    b = d/2;
    c = h;

    para{i} = eval(sprintf('@(x) -%d*(x-%d)^2+%d',a,b,c)); %converts the data into a parabola formula in the form of an
array
end

length = max(predata(1,:))+5;                %uses the given distances of puts and expands the range (for graphing)
paravalx = cell(r1,1);                       %creates empty arrays to generate the values of y from the parabola formula
paravalz = cell(r1,1);

for i = 1:r1
    [paravalx{i},paravalz{i}]=fplot(para(i,1),[0,length]); %calculates the values of the parabola on the x and z axis, note:
z represents height
end

valuecount=cell(r1,1);

for i = 1:r1
    valuecount{i}=size(paravalx{i},2);           %determines the number of values from the paraval matrix
end

funcs = cell(r1,1);                          %empty array for all the graphs functions
direction = [0,0,1];                         %direction in the z

                    %figure 1: creates a plot of the trajectory of the punt with the correct angle, height and distance
for i = 1:r1
funcs{i}=plot3(paravalx{i},zeros(1,valuecount{i}),paravalz{i});           %plots the function in 3d. Note: the arrays have
to opened using the {} notation
rotate(funcs{i},direction,data(i,3),[0,0,0]);                   %rotates the angle
hold on
xlim([0,max(predata(1,:))+5]);            %this is meant to properly graph the length of the football field
zlim([0,40]);                        %this is meant to properly graph the correct height
xlabel('Length of Football Field (Yd)')       %proper labels
ylabel('Width of Football Field (Yd)')
zlabel('Height (Yd)')
box on
title('Modeled Path of Football Punts');
end
hold off

%% Figure 2
figure(2)                        %making line plot
curvelength = cell(r1,1);            %This is for finding the distance of each punt
curveangle = cell(r1,1);             %This is for finding the angle of each punt
tanangle = cell(r1,1);              %This is for calculating the tangent of each punt
widthlength = cell(r1,1);            %This is for calculating the left or right shift of each punt
widthplot = cell(r1,1);

for i = 1:r1
curvelength{i}=predata(1,i);
```

```
curveangle{i}=predata(3,i);
tanangle{i}=tand(curveangle{i});                     %finds the tangent
widthlength{i}=curvelength{i}*tanangle{i};            %trigonometric calculation
widthplot{i}=plot(-widthlength{i},zeros(r1,1),'bo');   %plots all the landing spots on xaxis
hold on
xlim([-25,25]);                        %changes the limits to width of football field
set(gca,'YColor','none');              %removes the y axis
set(gca,'XAxisLocation','origin')
ylim([-1,1]);
xlabel('Width of Football Field (Yd)')
title('Landing Positions of Punts along Field Width');
box off
end
hold off

%% Figure 3
figure(3)          % creating figure 3
category1 = 0;      % creating category variables (these will change later)
category2 = 0;
category3 = 0;
category4 = 0;
for i = 1:r1                  % repeats 100 times for the 100 points
   val = predata(1,i);         % gets distance value from predata by chronological order at coordinate (1,n)

   if val >= 30 && val < 40            % if the distance value is between 30 and 40, category1 increases by 1
      category1 = category1 +1;
   end
   if val >= 40 && val < 50            % if the distance value is between 40 and 50, category2 increases by 1
      category2 = category2 +1;
   end
   if val >= 50 && val < 60            % if the distance value is between 50 and 60, category3 increases by 1
      category3 = category3 +1;
   end
   if val >= 60 && val < 70            % if the distance value is between 60 and 70, category4 increases by 1
      category4 = category4 +1;
   end
end

catVec = [category1 category2 category3 category4];  % puts categories into vector form

bar(catVec(1,:))                       % bar graph of categories
xlabel('Distances (Yd)')
ylabel('Number of Punts')
xticklabels({'30-39','40-49','50-59','60-69'});
title('Number of Punts vs. Distance');

%% Figure 4
figure(4)             % remaking figure 1, but changing the view so that there is an aerial view of the trajectories
for i = 1:r1

funcs{i}=plot3(paravalx{i},zeros(1,valuecount{i}),paravalz{i});              %plots the function in 3d. Note: the arrays have
to opened using the {} notation
rotate(funcs{i},direction,data(i,3),[0,0,0]);                         %rotates the angle
hold on
```

```
xlim([0,max(predata(1,:))+5]);                      %this is meant to properly graph the length of the football field
zlim([0,40]);                                       %this is meant to properly graph the correct height
xlabel('Length of Football Field (Yd)')             %proper labels
ylabel('Width of Football Field (Yd)')
zlabel('Height (Yd)')
box on
title('Aerial View of Modeled Path of Football Punts');
end
campos([0,0,15000]) %aerial view

%% Figure 5
figure(5)
for i = 1:r1
    d = predata(1,i);                       %first row, distance
    h = predata(2,i);                       %second row, height
    ang = atand((4*h)/d);                   %take the height and distance of ith column and find ang in degrees
    v = sqrt((2*h*9.8)/sind(ang)^2);        %velocity given height and angle
    t = d/(v*cosd(ang));                    %time of flight given distance and velocity

    plot(d/h,t,'go')
    xlabel('Distance : Height Ratio (Yd)')       %x-axis is ratio of distance:height of punt
    ylabel('Time of Flight (seconds)')           %y-axis is time of flight of football
    title('Distance : Height Ratio vs. Time of Flight')
    hold on
end
```

Appendix B

We generated our own data:

Distance	Heights	Angle
65	28	0
60	26	-11
45	23	13
33	29	-11
32	30	-20
40	27	4
63	29	8
35	32	-5
44	30	-6
61	31	9
31	29	-15

57	27	13
39	28	-2
31	29	-1
65	25	-19
50	26	20
58	33	-3
30	32	0
48	28	-14
37	25	18
50	24	12
53	24	-14
34	23	-9
61	29	4
65	27	2
55	28	-1
66	30	-18
59	30	19
33	31	15
61	24	-20
56	33	-19
42	26	13
44	28	-15
36	27	-3
44	31	2
39	29	0
33	25	5
40	32	-7
41	30	19
30	28	13
37	27	10
42	28	-10
38	29	-1
43	28	1
48	25	-5
65	26	-18

62	31	-16
32	30	16
39	30	3
54	29	-5
55	27	-20
52	26	5
59	28	-7
62	30	-16
34	28	20
40	29	-1
31	30	4
40	32	-8
60	29	18
42	26	-12
32	25	17
50	23	-9
61	31	10
62	30	-11
48	24	0
31	27	-1
37	29	11
38	30	0
53	31	-20
59	32	7
61	28	-9
42	28	8
41	26	3
52	27	-4
39	29	-10
54	28	20
50	24	-18
51	23	20
34	25	3
38	26	1
39	32	2

30	30	-19
62	29	20
60	30	3
42	28	10
49	27	-16
61	26	-15
31	25	5
51	24	9
47	23	20
51	29	-20
59	25	-3
31	23	-9
65	32	-2
65	28	-10
39	28	10
60	31	6
30	32	5
55	25	20
33	26	0