# Assignment 3

Math Fundamentals for Robotics 16-811, Fall 2018
DUE: Thursday, October 11, 2018

[redacted] (Andrew ID: [redacted])

1. Consider the function $f(x) = 0.5 + \sin x$ over the interval $[-\pi/2, \pi/2]$.

   (a) What is the Taylor series expansion for $f(x)$ around $x = 0$ ?

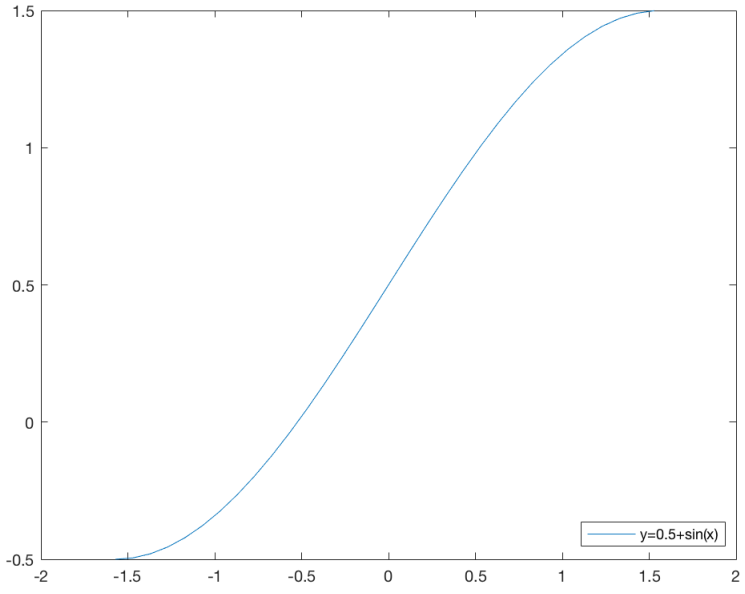
   **SOLUTION:**

$$f'(x) = \cos x, \quad f''(x) = -\sin x, \quad f'''(x) = -\cos x, \quad f^{(4)} = \sin x, \quad ...$$

$$\begin{aligned} f(x) &= f(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2 + \frac{f'''(0)}{3!}x^3 + \frac{f^{(4)}(0)}{4!}x^4 + ... \\ &= 0.5 + x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \frac{1}{7!}x^7 + ... \\ &= 0.5 + \sum_{n=0}^{\infty}(-1)^n \frac{1}{(2n+1)!}x^{(2n+1)} \end{aligned}$$

   (b) Graph $f(x)$ over the interval $[-\pi/2, \pi/2]$.

   **SOLUTION:**



(c) Determine the best uniform approximation by a quadratic to the function $f(x)$ on the interval $[-\pi/2, \pi/2]$. What are the $L_\infty$ and $L_2$ errors for this approximation?

**SOLUTION:**

Assume that the best uniform approximation by a quadratic to the function $f(x)$ on the interval $[-\pi/2, \pi/2]$ is:

$$p(x) = ax^2 + bx + c$$

According to the theorem, there are 4 points $x_0 \le x_1 \le x_2 \le x_3$ such that the max error occurs at the $x_i$s with alternating sign. Because $\sin x$ is odd symmetric, if $a \neq 0$, there would not be 3 intersection points between $f(x)$ and $p(x)$ on the interval $[-\pi/2, \pi/2]$ to have 4 points satisfying the theorem. So $a = 0$, $p(x) = bx + c$. Thus,

$$e(x) = f(x) - p(x) = 0.5 + \sin x - bx - c$$

As the theorem mentioned, we have $x_0 = -\pi/2$, $x_3 = \pi/2$, and

$$e(x_0) = -e(x_1) = e(x_2) = -e(x_3)$$

$$\begin{aligned} e(x_0) &= -0.5 + \frac{\pi}{2}b - c \\ e(x_1) &= 0.5 + \sin x_1 - bx_1 - c \\ e(x_2) &= 0.5 + \sin x_2 - bx_2 - c \\ e(x_3) &= 1.5 - \frac{\pi}{2}b - c \end{aligned}$$

So we can get that:

$$\begin{aligned} c &= 0.5 \\ b &= \frac{1 - \sin x_1}{\pi/2 - x_1} = \frac{1 + \sin x_2}{\pi/2 + x_2} \end{aligned}$$

Since we get maximum error at $x_1$ and $x_2$, then we have $e'(x_1) = e'(x_2) = 0$.

$$e'(x) = \cos x - b$$

$$\begin{aligned} e'(x_1) &= \cos x_1 - b = 0 \\ e'(x_2) &= \cos x_2 - b = 0 \end{aligned}$$

With the above equations, we now have:

$$(\pi/2 - x_1)\cos x_1 + \sin x_1 - 1 = 0, \quad x_2 = -x_1$$

By using the Muller's method (implemented in `code/q1.m`), get the solution for $x_1$ is $x_1 = -0.7603$. So, $b = \cos x_1 = 0.7246$.

Thus, the best uniform approximation by a quadratic to the function $f(x)$ is:

$$p(x) = 0.7246x + 0.5$$

So, the $L_\infty$ and $L_2$ errors for this approximation are:

$$\begin{aligned} L_\infty &= ||e(x)||_\infty = |e(x_1)| = 0.1382 \\ L_2 &= ||e(x)||_2 \\ &= \sqrt{\int_{-\pi/2}^{\pi/2} (\sin x - 0.7246x)^2 dx} \\ &= 0.1704 \end{aligned}$$

(d) Determine the best least-squares approximation by a quadratic to the function $f(x)$ on the interval $[-\pi/2, \pi/2]$. What are the $L_\infty$ and $L_2$ errors for this approximation?

**SOLUTION:**

To get the best least-squares approximation by a quadratic to the function $f(x)$, we use the basis functions:

$$\phi_0(x) = 1, \quad \phi_1(x) = x, \quad \phi_2(x) = x^2$$

Convert them into orthogonal basis:

$$\begin{aligned} < \phi_0, \phi_0 > &= \int_{-\pi/2}^{\pi/2} 1 dx = \pi \\ < \phi_1, \phi_1 > &= \int_{-\pi/2}^{\pi/2} x^2 dx = \frac{\pi^3}{12} \\ < x\phi_0, \phi_0 > &= \int_{-\pi/2}^{\pi/2} x dx = 0 \\ < x\phi_1, \phi_1 > &= \int_{-\pi/2}^{\pi/2} x^3 dx = 0 \end{aligned}$$

$$\begin{aligned} b_0 &= 1 \\ b_1 &= \left[ x - \frac{< x\phi_0, \phi_0 >}{< \phi_0, \phi_0 >} \right] \phi_0 = x \\ b_2 &= \left[ x - \frac{< x\phi_1, \phi_1 >}{< \phi_1, \phi_1 >} \right] \phi_1 - \frac{< \phi_1, \phi_1 >}{< \phi_0, \phi_0 >} \phi_0 = x^2 - \frac{\pi^2}{12} \end{aligned}$$

Thus, the least-square approximation is a linear combination of basis $b_0, b_1, b_2$, where the coefficients are:

$$a_0 = \frac{< f(x), b_0 >}{< b_0, b_0 >} = \frac{\int_{-\pi/2}^{\pi/2}(0.5 + \sin x)dx}{\int_{-\pi/2}^{\pi/2} 1dx} = \frac{0.5\pi}{\pi} = 0.5$$

$$a_1 = \frac{< f(x), b_1 >}{< b_1, b_1 >} = \frac{\int_{-\pi/2}^{\pi/2}(0.5x + x\sin x)dx}{\int_{-\pi/2}^{\pi/2} x^2 dx} = \frac{2}{\frac{1}{12}\pi^3} = \frac{24}{\pi^3}$$

$$a_2 = \frac{< f(x), x_2 >}{< b_1, b_1 >} = \frac{\int_{-\pi/2}^{\pi/2}(0.5 + \sin x)(x^2 - \frac{\pi^2}{12})dx}{< b_1, b_1 >} = 0$$

So, the best least-squares approximation by a quadratic to the function $f(x)$ is:

$$p(x) = \frac{24}{\pi^3}x + 0.5 = 0.7740x + 0.5$$

So, the error function is:

$$e(x) = f(x) - p(x) = \sin x - 0.7740x$$

when $x = \frac{\pi}{2}$ or $x = -\frac{\pi}{2}$ we have the maximum error.

So, the $L_\infty$ and $L_2$ errors for this approximation are:

$$\begin{aligned} L_\infty &= ||e(x)||_\infty = |e(\frac{\pi}{2})| = 0.2158 \\ L_2 &= ||e(x)||_2 \\ &= \sqrt{\int_{-\pi/2}^{\pi/2} (\sin x - 0.7740x)^2 dx} \\ &= 0.1507 \end{aligned}$$

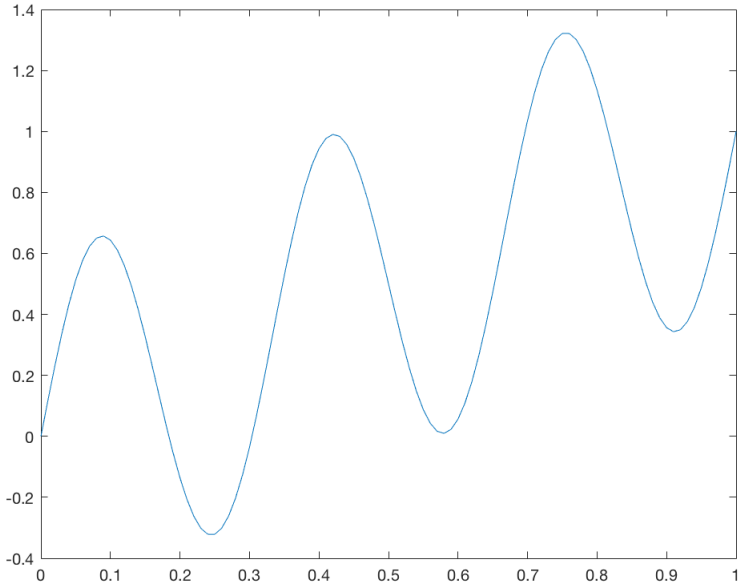
2. Suppose very accurate values of some function $f(x)$ are given at the points $0 = x_0, x_1, ..., x_{100} = 1$, with the $x_i$ uniformly distributed over the interval [0, 1]. (So $x_i = i/100, i = 0, ..., 100.$) The values $f(x_i)$ are given in the file 'problem2.txt' in sequential order (so, for example, $f(0.12) = f(x_{12}) = 0.560293281586165$).
What is the function $f(x)$?

   [Provide a succinct description using one or more analytic expressions.]

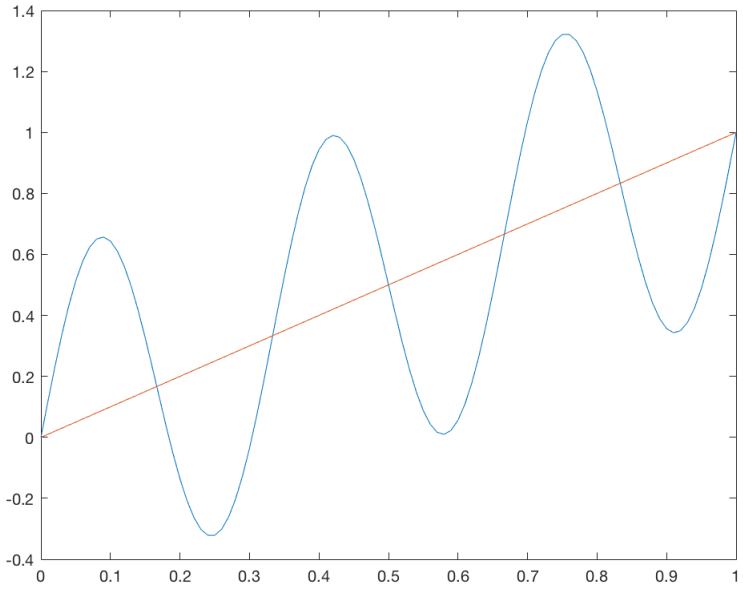
   [Hint: Try "basis" functions, $1, x, x^2, ..., \cos(\pi x), \sin(\pi x), \cos(2\pi x), \sin(2\pi x), ...,$ etc. This is one of those over- and under-constrained systems. Try to find a relatively simple description of the function $f(x)$, by determining which function coefficients may be set to zero. There may be multiple candidate answers. Find the one with the fewest nonzero coefficients.]

   **SOLUTION:**

For this problem, we first plot the function $f(x)$ with these accurate values read from file 'problem2.txt' at points $0 = x_0, x_1, ..., x_{100} = 1$. The plotted function is shown as below:

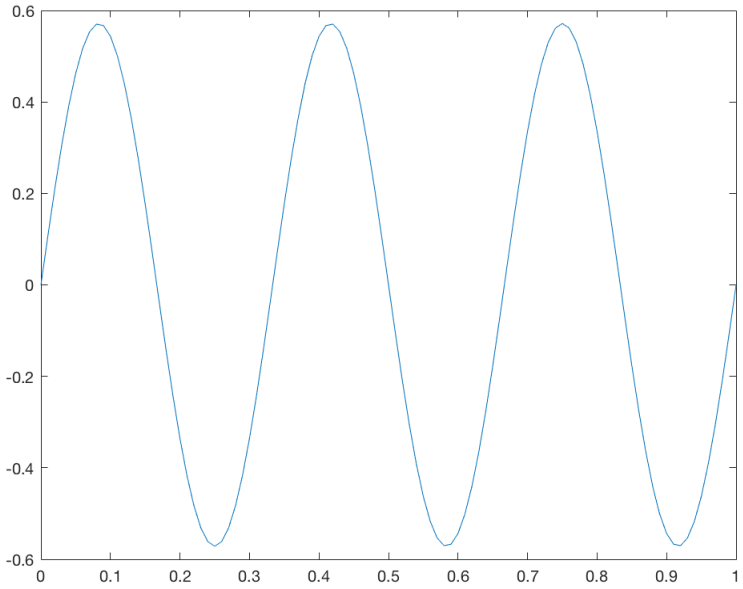



We can see from the figure that the values of $f(x)$ in interval $[0, 1]$ is almost periodical around the line $g(x) = (f(x_{100}) - f(x_0))\, x + f(x_0) = x$ shown as below:




So we try to subtract the values of $g(x)$ from the values of $f(x)$, namely, let $t(x) = f(x) - g(x)$ over interval $[0, 1]$, and then we plot the function $t(x)$. The figure is shown

as below:




So with the observation of the figures, function $t(x)$ is a periodical function and the period is $\frac{1}{3}$. We can get the approximation function of $t(x)$ is constructed with basis $\sin 6\pi x$.

With the analysis and observations above, we now know that the simplest way to describe the function $f(x)$ is to use the basis functions $x$ and $\sin 6\pi x$. The code is implemented in `code/q2.m`. The function `get_coefficient` is used for estimating the coefficients of these two basis functions to describe $f(x)$. The inputs of the function are: `y` (matrix with all values from file 'problem2.txt'), `x` (matrix with all $x$ values) and `base` (matrix containing basis values). The returned values are `A` (matrix containing the calculated coefficients) and `error` (the error with the presentation. )

By calling the function `get_coefficient`, the estimated coefficients of these basis functions are returned as below:

$$A = \begin{bmatrix} 1 \\ 0.5714 \end{bmatrix}$$

So the presentation of function $f(x)$ is:

$$f(x) = x + 0.5714 \sin 6\pi x$$

The returned value `error` indicates that with this presentation, the error is $1.08e{-}14 \approx 0$.

3. The Chebyshev polynomials of the first kind are defined according to:

$$T_n(\cos\theta) = \cos(n\theta), \quad for\ n \geq 0.$$

(a) Derive $T_6$ and $T_7$ and show that they are orthogonal polynomials relative to the inner product.

$$< g, h >= \int_{-1}^{1} (1-x^2)^{-1/2} g(x)h(x)dx$$

**SOLUTION:**

$T_0(x) = 1$
$T_1(x) = x$
$T_2(x) = 2xT_1(x) - T_0(x) = 2x^2 - 1$
$T_3(x) = 2xT_2(x) - T_1(x) = 4x^3 - 3x$
$T_4(x) = 2xT_3(x) - T_2(x) = 8x^4 - 8x^2 + 1$
$T_5(x) = 2xT_4(x) - T_3(x) = 16x^5 - 20x^3 + 5x$
$T_6(x) = 2xT_5(x) - T_4(x) = 32x^6 - 48x^4 + 18x^2 - 1$
$T_7(x) = 2xT_6(x) - T_5(x) = 64x^7 - 112x^5 + 56x^3 - 7x$

$$\begin{aligned} < T_6(x), T_7(x) > &= \int_{-1}^{1} (1-x^2)^{-1/2} T_6(x)T_7(x)dx \\ &= \int_{-1}^{1} (1-x^2)^{-1/2}(32x^6 - 48x^4 + 18x^2 - 1)(64x^7 - 112x^5 + 56x^3 - 7x)dx \\ &= \int_{-1}^{1} (1-x^2)^{-1/2} \sum_{i=0}^{6} a_i x^{2i+1} dx \end{aligned}$$

Let $P(x) = \sum_{i=0}^{6} a_i x^{2i+1}$, where $a_i$ is the coefficient of $x^{2i+1}$ calculated from $T_6(x)T_7(x)$. Then the equation above can be represented as:

$$\begin{aligned} < T_6(x), T_7(x) > &= \int_{-1}^{1} (1-x^2)^{-1/2} P(x)dx \\ &= -(1-x^2)^{1/2} \frac{P(x)}{x}\bigg|_{-1}^{1} + \int_{-1}^{1} (1-x^2)^{1/2} \frac{P'(x)x - P(x)}{x^2} dx \\ &= \int_{-1}^{1} (1-x^2)^{1/2} \left( \frac{P'(x)x - P(x)}{x^2} \right) dx \\ &= \int_{-1}^{1} (1-x^2)^{1/2} \left( \frac{\sum_{i=0}^{6} a_i x^{2i+1} - \sum_{i=0}^{6} a_i x^{2i+1}}{x^2} \right) dx \\ &= 0 \end{aligned}$$

So $T_6$ and $T_7$ are orthogonal polynomials relative to the given inner product.

(b) Relative to this inner product, all the $T_n$, with $n > 0$, have the same length. Establish the fact by computing the length of $T_n$ (with $n$ left symbolic, while assuming $n > 0$).

**SOLUTION:**

$T_n(\cos\theta) = \cos(n\theta), \quad for\ n \geq 0$

Let $\cos\theta = x$, then $x^2 = \cos^2\theta$

$$\begin{aligned}
< T_n, T_n > &= \int_{-1}^{1} (1-x^2)^{-1/2} T_n^2(x) dx \\
&= \int_{-1}^{1} (\sin^2\theta)^{-1/2} \cos^2(n\theta) d(\cos\theta) \\
&= \int_{-\pi}^{0} (\sin\theta)^{-1} \cos^2(n\theta)(-\sin\theta) d\theta \\
&= \int_{-\pi}^{0} -\cos^2(n\theta) d\theta \\
&= \int_{0}^{\pi} \cos^2(n\theta) d\theta \\
&= \int_{0}^{\pi} \frac{1}{2}(\cos 2n\theta + 1) d\theta \\
&= \frac{\pi}{2}
\end{aligned}$$

So, relative to this inner product, all the $T_n$, with $n > 0$, have the same length.

4. After weeks of work you have finally completed construction of a gecko robot. It is a quadruped robot with suctioning feet that allow it to walk on walls. It is also equipped with a Kinect-like sensor, providing a 3D point cloud observation of the world. You want to use these point clouds to reason about the environment and aid in navigation.

**SOLUTION:** For task (a) and (b), the code is implemented in `code/q4ab.m`; For task (c) and (d), the code is implemented in `code/q4cd.m`.

(a) You boot up the robot and place it on a table, taking an initial observation. The observation is saved in the provided `clear_table.txt`, and lists $(x, y, z)$ locations in the following format:

$$\begin{matrix} x_1 & y_1 & z_1 \\ & \vdots & \\ x_n & y_n & z_n \end{matrix}$$

Points are in units of meters and the positive x-direction is right, positive y-direction is down, and positive z-direction is forward. Find the least-squares approximation plane that fits the data. Visualize your fitted plane along with the data. What is the average distance of a point in our data set to the fitted plane? (i.e., how accurate is our sensor?)

**SOLUTION:**

For a plane in the 3D space, it can be represented in the format: $z = ax + by + c$. So given the 3D points in `clear_table.txt`, use the least-squares approximation method to approximate the plane.

Function `get_coefficient` is used to estimating the coefficients of the basis. The inputs include `x`, `y` and `z`, which are vectors of $x$, $y$ and $z$ coordinates of $n$ points respectively. The outputs are `t` (the vector of estimated coefficients $a, b, c$) and `error` (the error of the estimation respect to value $z$ given $x$ and $y$). Function `get_avg_dist` is used for calculating the average distance of the points to the fitted plane. The inputs are `x`, `y`, `z` and `t` (the coordinates of the points and the coefficients of the fitted plane.) The output is `avg_dist`.

By calling function `get_coefficient`, we can get that the returned coefficients are as below:

$$t = [-1.7931, -18.7452, 9.4391]^T$$

So the fitted plane is:

$$z = -1.7931x - 18.7452y + 9.4391$$

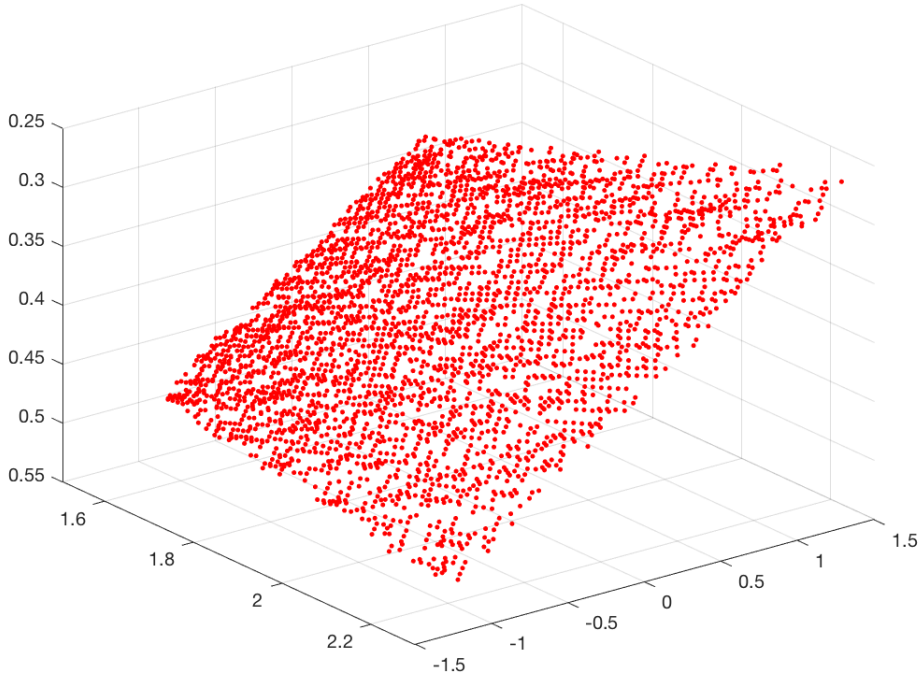
namely,

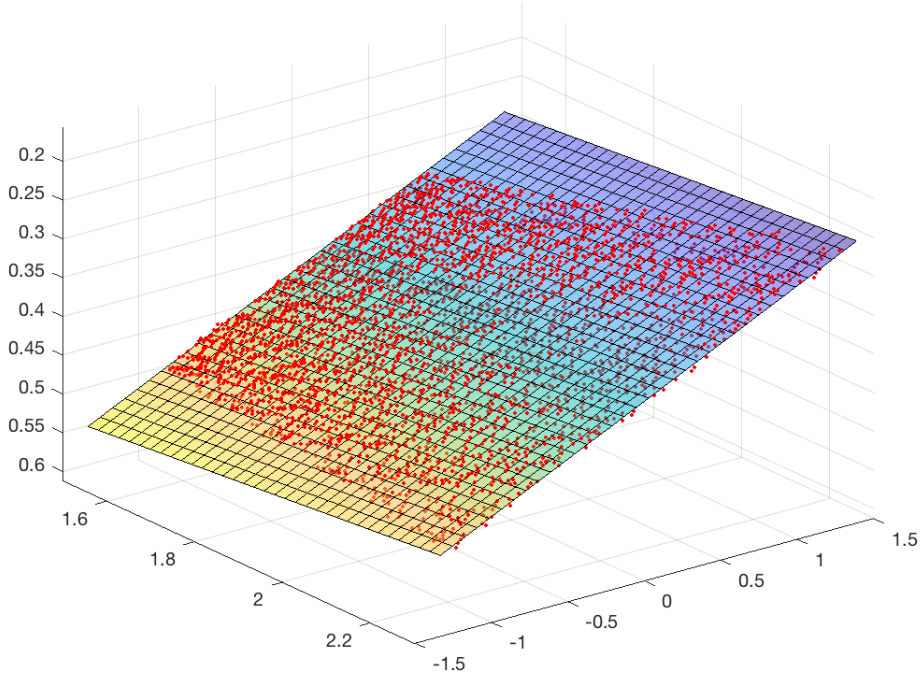
$$1.7931x + 18.7452y + z - 9.4391 = 0$$

By calling function `get_avg_dist`, we can get the average distance of the points in the dataset to the fitted plane is 0.0029.

In script `code/q4ab.m`, calculating the coefficients of plane and the average distance to the plane, as well as the plotting, are all included in function `q4_a`. The input is just the data file, and the output are `t`, `error` and `avg_dist`.

The figures of plotting 3D points and showing the fitted plane along with the data points are shown as below:


0.25
0.3
0.35
0.4
0.45
0.5
0.55
1.6
1.8
2
2.2
-1.5
-1
-0.5
0
0.5
1
1.5


0.2
0.25
0.3
0.35
0.4
0.45
0.5
0.55
0.6
1.6
1.8
2
2.2
-1.5
-1
-0.5
0
0.5
1
1.5

(b) Interested in your gecko robot, your cat jumps up on the table. You take a second observation, saved as the provided `cluttered_table.txt`. Using the same method as above, find the least-squares fit to the new data. How does it look? Why?

**SOLUTION:**

Using the same method as above, the least-squares fit to the new data is:

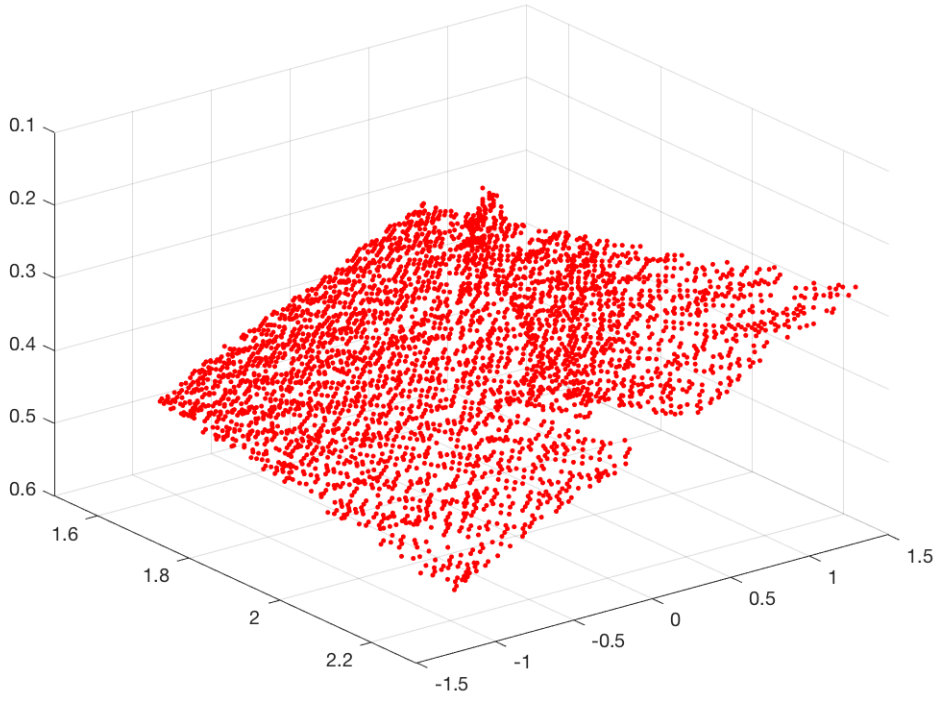
$$z = -0.2001x - 2.0224y + 2.6567$$

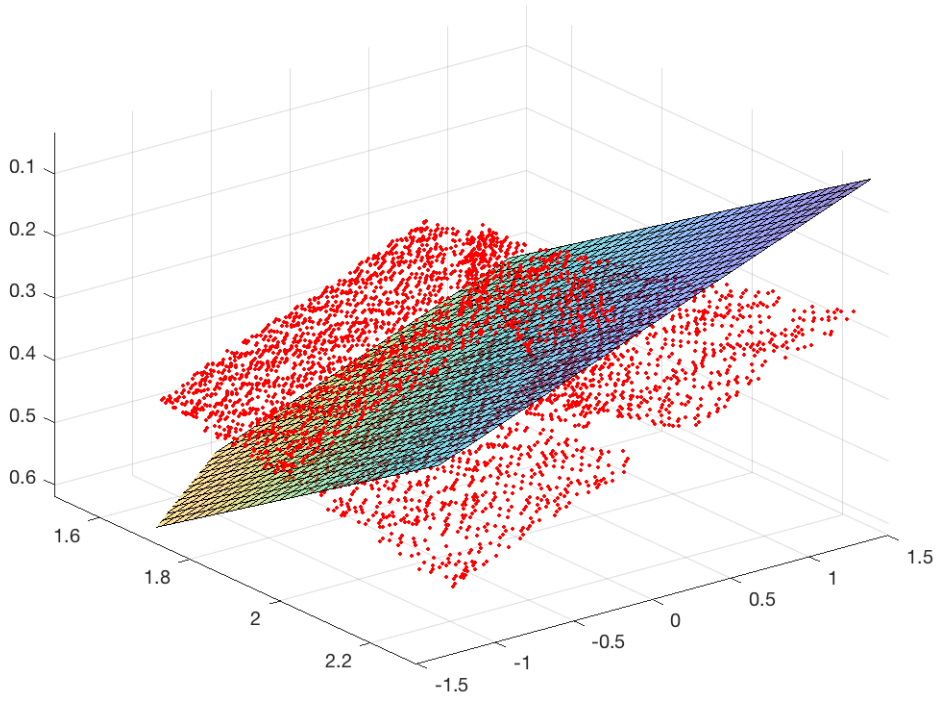
namely,

$$0.2001x + 2.0224y + z - 2.6567 = 0$$

The average distance of the points in the dataset to the fitted plane is 0.0771.

The figures of plotting the given 3D points and showing the fitted plane along with the data points are shown as below:






The plane does not fit the data well. The reason should be that the cap jumps up on the table, and there are many 3D points detected from the cat in the new observation. These points from the cat are mostly not on the plane, instead they are above the plane since the cat is standing. So with these points, it will cause the least-squares approximation deviates from the real plane.

(c) Can you suggest a way to still find a fit to the plane on the table regardless of clutter? Verify your idea by writing a program that can successfully find the dominant plane in a list of points regardless of outliers. [Hint: You may assume that the number of points on the plane is much larger than the number of points not on the plane.] Visualize `cluttered_table.txt` with your new plane.

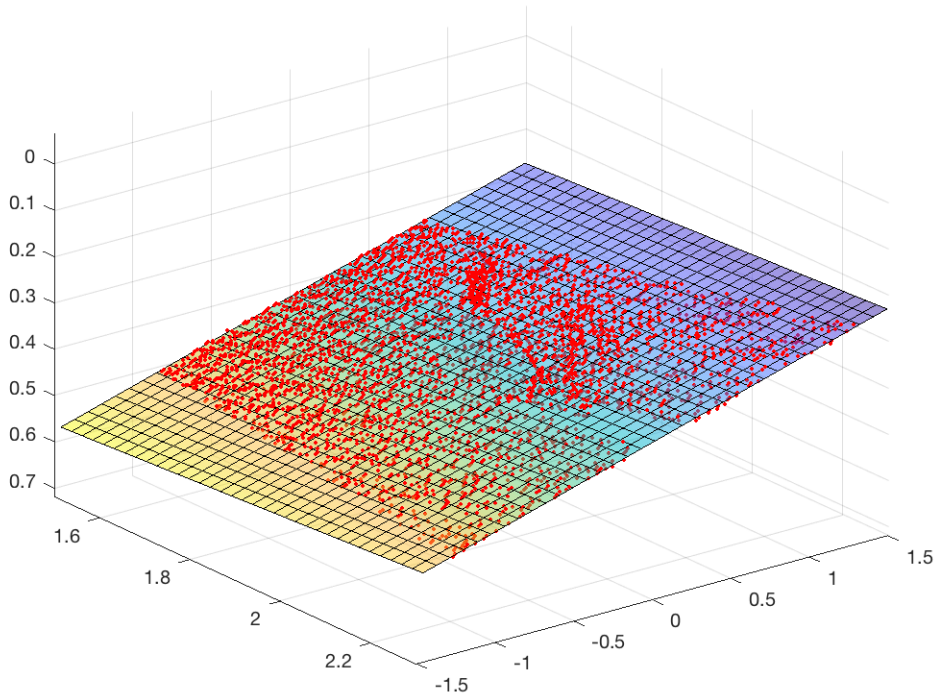
**SOLUTION:**

This part is implemented in `code/q4cd.m`. The main idea to deal with clutter is that: Since we actually just need three points to determine a plane in 3D pace, then in this task, we just randomly select three points from the dataset and then get the plane. Calculate the distance to the plane for each data point in the dataset. Then we get those points that have distance no larger than 0.05 meter. If such points take up 2/3 of the total number of points in dataset, then these points are all near the plane and can be used for approximating the plane (since the number of points on the plane is much larger than the number of points not

on the plane). If the percentage of such points less than 2/3, then randomly select three points again until find points meet the requirement. Finally, use all the points near the plane (distance no larger than 0.05 meter) to approximate the plane using the least-squares approximation method.

In this script, function `get_coefficient`, `get_avg_dist` and `plot_on_plane` are exactly the same as in `code/q4ab.m`. Function `find_main_plane` is implemented based on the idea described above. It would find all the points near the plane and get the coefficients of the plane with the least-square approximation. The inputs are `x`, `y`, `z` and `ratio`, where `x`, `y`, `z` are $x$, $y$, $z$ coordinates of the points, and `ratio` is the threshold of the points ratio to determine whether the current plane is a main plane. All of these are integrated in function `q4_c`, taking input `data_file` and returning `t`, `error`, `avg_dist`.

The plotting result of the data points and the fitted plane is shown as below. The average distance of the points in the dataset to this plane is 0.0154.




(d) Encouraged by your results when testing on a table, you move your geckobot into the hallway and take an observation saved as the provided `clean_hallway.txt`. Describe an extension to your solution to part (c) that finds the four dominant planes shown in the scene, then implement it and visualize the data and the four planes. You may assume that there are roughly the same amount of points in

each plane.

**SOLUTION:**

This part is also implemented in `code/q4cd.m`. Based on the idea in part (c), in this part, the near points ratio is set to 1/5 since there are roughly the same amount of points in each plane and there are 4 planes in total. Each time a plane is determined, remove all the points that are near this plane from the dataset and then use the rest points to find the next plane until all the planes are found. This process is integrated in function `q4_d`. The input is `data_file` and the outputs include `t`, `error`, `avg_dist`.

The four dominant planes are described as:

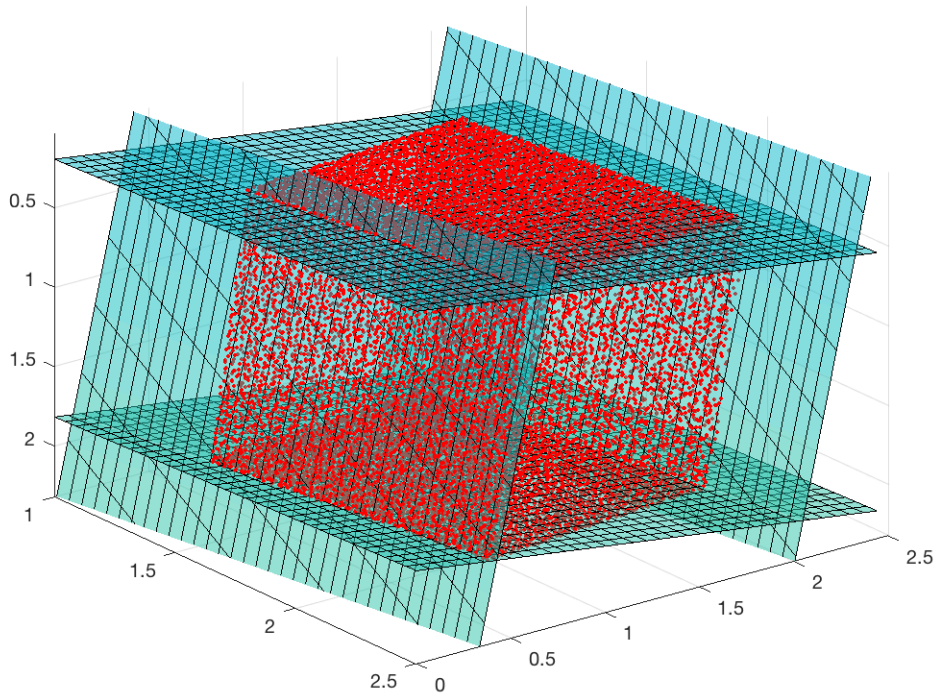
$$z = 4.6078x + 0.8337y - 0.9744$$

$$z = 3.1692x - 18.1807y + 4.3653$$

$$z = 4.5717x + 0.8621y - 8.6163$$

$$z = 3.9833x - 23.2422y + 42.9423$$

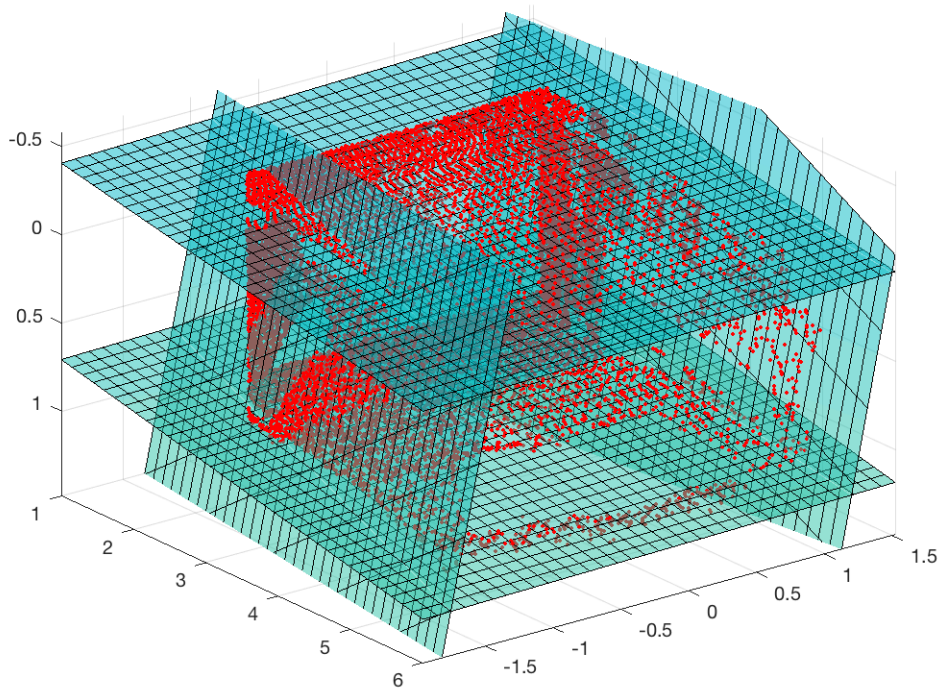
The plotting result of the data points and the fitted plane is shown as below. of the points in the dataset to this plane is 0.0081.



(e) You decide its time to test your gecko robots suction feet and move it to a different hallway. The feet are strong enough to ignore the force of gravity, allowing the robot to walk on the floor, walls, or ceiling. However, the locomotion of the legs works best on smooth surfaces with few obstacles. Using your solution from part (d), describe how you can mathematically characterize the smoothness of each surface. Load the provided scan `cluttered_hallway.txt`, find and plot the four wall planes, describe which surface is safest for your robot to traverse, and provide the smoothness scores from your mathematical characterization. Note that you can no longer assume that there are roughly the same amount of points in each plane.

**SOLUTION:** For this part, the code implemented in `code/q4e.m`

Using the similar method in part (d), the obtained plotting result is shown below:




The four dominant planes are described as:

$$z = 0.1427x + 8.6578y - 4.7394$$

$$z = 0.2637x + 10.4954y + 5.6817$$

$$z = 15.0128x + 3.7486y - 16.6503$$

$$z = -9.6646x - 2.3641y - 8.3336$$

The difference between method (e) and method (d) is that, when determining the last two main planes, I first cluster the remain data points into two separate parts

and then find the planes separately. In such a way, we can reduce the impact caused by the noise data.

The smoothness scores are set as the average distances from the each group of data points to the according plane (the data points groups are generated when finding the planes). The returned four average distances are:

$$[0.0279,\ 0.0257,\ 0.1663,\ 0.0613]$$

So the smoothest plane is the second one, that is, the top plane in the above figure.