**Lab 2: Estimating Velocity and Acceleration using Numerical Differentiation**

One application for numerical differentiation is to estimate the velocity and acceleration of a moving object based off of a set of position measurements. This is how your GPS works: a GPS triangulates your position based on the signal it receives from the GPS satellites in orbit around the earth. However, most GPS systems are also able to tell you the speed at which you are moving as well as your acceleration. How is it able to do this when it is only capable of measuring your position? Numerical differentiation!

From basic physics, we know that the velocity is the first derivative of the position and the acceleration is the first derivative of the velocity or the second derivative of the position. You’ve probably done this in physics when studying basic kinematics where you take the derivative of the x and y position equations to find the equations for velocity and acceleration. Unfortunately, the motion of an object may not be able to be modeled by a mathematical equation. However, as long as we have measurements of an objects position, we can estimate the velocity and acceleration using numerical estimates for the derivative.

In this lab, you will be using real GPS data collected by Microsoft employees in China. The data you will be using is from an individual walking around Beijing, China. Additionally, the equation shown below for the distance is called the Great Circle Equation (GCE), which is how distances are calculated from one location on Earth to another. More information on GPS can be found [here](http://electronics.howstuffworks.com/gadgets/travel/gps.htm) and information on the GCE can be found [here](http://en.wikipedia.org/wiki/Great-circle_distance).

**A. Practice Exercise**

The table below shows position measurements taken every 2 seconds. You will need to calculate the distance (length moved between measurements) and position (overall length moved) before computing the estimates. *These values are actually the GPS coordinates walking from the entrance to Baldwin to TUC.*

***Suggestion: Do this in a script file with Latitude and Longitude stored as vectors. It will make the next part much easier! Check answers with T.A. before moving on.***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Time** | **0** | **2** | **4** | **6** | **8** | **10** |
| **Latitude (deg)** | **39.132844** | **39.132662** | **39.132353** | **39.132076** | **39.132001** | **39.131964** |
| **Longitude (deg)** | **-84.516888** | **-84.516955** | **-84.516878** | **-84.516945** | **-84.517047** | **-84.517095** |
| **Distance (m)** | **0** | 21.0322 | 34.9716 | 31.3173 | 12.1142 | 5.8328 |
| **Position (m)** | **0** | 21.0322 | 56.0039 | 87.3212 | 99.4354 | 105.2682 |
| **Velocity 2PT (m/sec)** | **0** | 10.5161 | 17.4858 | 15.6587 | 6.0571 | 2.9164 |
| **Velocity 3PT (m/sec)** | **0** | 14.0010 | 16.5722 | 10.8579 | 4.4867 |  |
| **Acceleration (m/sec2)** | **0** | 3.4849 | -.9136 | -4.8008 | -1.5703 |  |

**B. Estimating Velocity and Acceleration from GPS Data**

On Blackboard, there are two files called **GPS\_Data1.mat** and **GPS\_Data2.mat**. Download these files and save them in your current MATLAB directory.

1. Create a script file in MATLAB. Add a comment with your name, section number, and Lab. Also add the commands: clear; close all;
2. Add a statement to your code to load GPS\_Data1.mat. This file contains 7 vectors:
   1. latitude: latitude position measurement
   2. longitude: longitude position measurement
   3. hour: hour at which the data point was collected
   4. minute: minute at which the data point was collected
   5. second: second at which the data point was collected
   6. actual\_velocity: vector of actual velocity values
   7. actual\_acceleration: vector of actual acceleration values
3. Add code to your script to compute vectors for the distance (remember the units of your angle!) and position based off of the latitude and longitude measurements.
4. Add code to your script to compute the Δt value. ***To calculate the Δt value you need all three time vectors (hour, minute, and second). Δt is constant so you can use any set of consecutive measurements.*** Don’t hard-code a value because you will be looking at a second data set with a different Δt.
5. Within your script, create a vector of times using the Δt and the number of measurements taken.
6. Before proceeding any further, compare your values for steps 3-5 with the values below. If you did not get these values, check with your TA.

|  |  |  |  |
| --- | --- | --- | --- |
| **Index** | **Distance** | **Position** | **t** |
| 1 | 0 | 0 | 0 |
| 2 | 15.0058 | 15.0058 | 20 |
| 3 | 22.0964 | 37.1022 | 40 |

1. Once you have your Position and time vectors correct, add code to your script to compute the velocity using the 2 point estimate.
2. Compute the absolute error (not percent error) between your 2 point estimated velocity and the actual velocity vector.
3. Create a new figure window with two graphs and plot the actual and estimated velocity on the top graph and the error on the bottom graph. Your result should look similar to the graph below:



1. Add code to your script to compute the velocity using the 3 point estimate.
2. Compute the absolute error between your 3 point estimated velocity and the actual velocity vector   
   ***Be careful of the sizes of your two vectors!***
3. Modify your plot commands from step 9 to include the 3 point estimate for the velocity on the top graph and the 3 point error on the bottom graph.
4. Add code to your script to compute the acceleration using the 2nd derivative estimate.
5. Compute the absolute error between your 2nd derivative estimated acceleration and the actual acceleration vector.
6. Create a new figure window with two graphs and plot the actual and estimated acceleration on the top graph and the error on the bottom graph.
7. Run your script for both data sets (GPS\_Data1.mat and GPS\_Data2.mat) and paste the resulting graphs below. You should have a total of four figures included (velocity and acceleration for GPS\_Data1 and velocity and acceleration for GPS\_Data2).

**PASTE FIGURES HERE (GPS\_Data1):**

 

**PASTE FIGURES HERE (GPS\_Data2):**

 

1. Answer the following questions:

|  |
| --- |
| * 1. What is the main difference you see between GPS\_Data1 and GPS\_Data2? |
| The second set of data has more data points. |
| * 1. What effect does this difference have on the results of your script? |
| There was much less error in the second set. |
| * 1. What are the possible reasons for the error between the estimated velocities and acceleration and the actual velocity and acceleration? |
| The GPS Satellite may use interpolation before find the estimate so that it can get a smaller delta t value when doing the estimation. |

**PASTE SCRIPT HERE:**

%Creator: Jonathan Kenney

%Section 010

%Lab 2: Estimating Velocity and Acceleration Using GPS Data

clear; clc; close all;

load GPS\_Data2.mat;

dist = zeros(1,7);

pos = zeros(1,7);

vel\_2pt = zeros(1,7);

vel2pt\_error = zeros(1,7);

vel\_3pt = zeros(1,6);

vel3pt\_error = zeros(1,6);

acc\_est = zeros(1,6);

acc\_error = zeros(1,6);

dt = (60\*60\*hour(2)+60\*minute(2)+second(2))-(60\*60\*hour(1)+60\*minute(1)+second(1));

t = 0:dt:dt\*26;

t\_less = 0:dt:dt\*25;

for k = 2:length(actual\_velocity)

dist(k) = 222240\*asind(sqrt(sind((latitude(k)-latitude(k-1))/2)^2+cosd(latitude(k-1))\*cosd(latitude(k))\*sind((longitude(k)-longitude(k-1))/2)^2));

pos(k) = pos(k-1)+dist(k);

vel\_2pt(k) = (pos(k)-pos(k-1))/dt;

vel2pt\_error(k) = abs(vel\_2pt(k)-actual\_velocity(k));

end

for k = 2:length(actual\_acceleration)

vel\_3pt(k) = (pos(k+1)-pos(k-1))/(2\*dt);

vel3pt\_error(k) = abs(vel\_3pt(k)-actual\_velocity(k));

acc\_est(k) = (pos(k+1)-2\*pos(k)+pos(k-1))/dt^2;

acc\_error(k) = abs(acc\_est(k)-actual\_acceleration(k));

end

figure(1)

subplot(2,1,1)

plot(t\_less,vel\_3pt,'r\*',t,actual\_velocity,'k-');

xlabel('time (sec)');

ylabel('Velocity (m/s)');

title('Estimated Velocity');

legend('Estimated','Actual');

subplot(2,1,2)

plot(t\_less,vel3pt\_error,'r\*');

xlabel('time (sec)');

ylabel('Abs Error (m/s)');

title('Estimation Error');

figure(2)

subplot(2,1,1)

plot(t\_less,acc\_est,'r\*',t\_less,actual\_acceleration,'k-');

xlabel('time (sec)');

ylabel('Acceleration (m/s^2)');

title('Estimated Acceleration');

legend('Estimated','Actual');

subplot(2,1,2)

plot(t\_less,acc\_error,'r\*');

xlabel('time (sec)');

ylabel('Abs Error (m/s)');

title('Estimation Error');

**Turn in Next Week:** This document with table results, plots, and answers to questions and your script file.