

MATH 320 Fall 2010, Homework 5, GPS tracking

Due November 17, 2010

1 Motivation

Handheld GPS devices are great fun and very easily available. Many people have them on their phones, and some of these output a running history of where they've been (only on request, one hopes...). Using these GPS tracks, one can do a surprisingly good job of piecing together a story about what the tracked person or object was doing, if good maps are available.

2 Objectives

After this assignment, you should...

- Apply basic numerical differentiation and integration
- Correlate GPS data with appropriate mapping tools

3 Your tasks

What you are given: I'll give you the following files:

- `gpstrack.gpx`: the output of my GPS
- `readGPX.m`: a function that reads GPX files into Matlab

The GPS track was from a walk that I took over the summer, when I had a few errands to run around and nearby Penn's campus. So I dropped the GPS in my pocket and let it record my location as I travelled. I took a few different kinds of transportation during this time. What's amusing is that you can make a pretty good guess at what I was doing based on where (and how) I went.

You may find Google Maps or Google Earth very helpful in this project, though neither are necessary to complete it (just a good map with latitude and longitude marked on it). The latter can load GPX files directly, so that you can see my path on a map. If you want to use Google Maps instead, that's fine – you can enter GPS coordinates into the search box and get markers placed on

a map for you. I use both frequently when I need to match up maps with other kinds of geographic information, simply because they are pretty easy to use.

Problem 1:

Part 1a: Load the track data into your Matlab session. You can do this using the function that I gave you:

```
>> [lats,lons,els,ts]=readGPX('gpstrack.gpx');
```

You will be given back four vectors of data describing the GPS track. For this part of the problem, **please plot these data** using appropriate axes. Perhaps a 3d-plot (using *plot3d*) is a good idea, perhaps using the time samples *ts* as the *z*-coordinate. You might also be interested in showing the elevation, though Penn's campus is pretty flat.

Part 1b: Answer the following questions by marking your plot. (Either print it out, and mark it by hand, or use your favorite graphics program.)

1. At what points did I stop moving?
2. When did I make an (obvious) transition in the form of transportation I chose (you will not be able to identify all such transitions visually)
3. What errands do you think I was running during this track? Can you tell if I failed at any of them? (If you open the GPX file with a text editor, you might be able to figure out *why* I failed!)
4. Do you notice anything else out of the ordinary in the path I selected? (Perhaps such things are ordinary for an absent-minded academic...)

Problem 2: Estimate the derivative from a vector of data using the centered differences formula (page 293 in your textbook). You should write a function which begins

```
function derivative=centerDiff(f,h)
% Compute derivative of a function specified at equally-spaced points
%
% derivative=centerDiff(f,h)
%
% Input: f = vector of function values
%        h = spacing between adjacent datapoints
% Output: derivative = estimate of derivative at datapoints
```

Note carefully that derivative will be shorter than *f*! By how much?

Problem 3: To figure out the length of the track, you'll want to estimate a definite integral. There are several such famous techniques, and we'll discuss more in class (as well as the error bounds on the ones you already know). For this project, Simpson's rule will suffice, so code up the following function:

```

function integral=definiteInt(f,h)
%
% Estimate a definite integral using Simpson's rule
%
% Input: f = vector of function values
%        h = spacing between adjacent datapoints
% Output: integral = estimate of the integral

```

Of course, Simpson's rule requires a specific number of points to work. You're free to decide how to handle the situation where the wrong number of points are entered. Here are a few options:

- Throw an error (perhaps using the Matlab *error* function),
- Silently discard datapoints or zero-pad the data (maybe emit a *warning* message),
- Switch to another integration method (again with a warning message), or
- Figure out how to modify the Simpson's rule formula you learned in Calculus class to handle the wrong number of data points (my favorite option).

Problem 4:

Answer the following questions using the routines you constructed:

1. What was my maximum speed over the course of the track?
2. How far did I travel over the entire track?
3. When did I switch modes of transport? (The most effective way to notate this is to color the tracks by speed.)
4. How far did I travel on foot?

Collect all of your functions into a directory ("Folder" for the Windows users!), zip it up, and submit using Blackboard. If you want, you can put scans or electronic documents in that zip file, or you can give them to me in class on paper, but the functions must be submitted electronically.