Assignment Procedures

The assignment is to be submitted in two parts. Part 1 is due 6pm Tuesday, Oct 27 (BEFORE class) and is worth 20% of the grade. The rest of the assignment (Part 2) is to be submitted by 6pm on Tuesday, Nov 10 (BEFORE lab).

- Assignments should be handed in as **pdfs** with your name(s) on it containing

  1. Figures and the text answering the questions given
  2. Listings of the scripts and functions you have created.

  If you are running MATLAB 2016a or later, you may find it easy to use the matlab LIVE EDITOR to combine text and figures, but this is not required.

- ASSIGNMENTS SHOULD BE DONE IN PAIRS. Alone or in groups of 3 is OK with instructor permission. Either way, you MUST document the degree of collaboration. To do this EVERYONE must submit a file `partner.m` via Canvas. The file will contain the following lines:

```
partner.name='YYYYYY';  % Last name or 'none'
partner.collab='AAAA';
Time_spent= XX;         % Hours of time
```

  where the `collab` string is a phrase like 'checked answers only', 'worked whole assignment together', or some other brief statement that describes the nature of your collaboration (or 'none' if there was none). This is just for our information – we'd like to keep track of how much collaboration people feel is needed / helpful: it doesn't affect your grade (although it will help us make sure grading is done correctly). However, if we find evidence of collaboration that is NOT reported...then that WILL affect your grade.

  Hand this in for BOTH part 1 and part 2.

- Your assignment grade will be based on correct work, but also on good code, good coding practice, and aesthetically pleasing output.

  1. All figures must be labeled with titles, axis labels, units where appropriate, legends if needed and text as requested.
  2. Code must be appropriately commented - enough explanation to see what happens but not verbose comments that distract from the actual code itself. Use whitespace wisely!

Introduction

The ocean in the Salish Sea is affected by tidal forces, wind, and spatial changes in density, all of which drive currents. What do these currents do? If we release a floating object in the Strait of Georgia (the part of the Salish Sea right beside Vancouver), where will it end up? Simple questions like turn out to be surprisingly difficult to answer.

At present, we do know that there is an "estuarine" circulation in the Salish Sea. Fresh water flows in from the Fraser River just south of Richmond. This fresh water eventually ends up in the Pacific[*]. So, there is a mean flow of surface water out of the Fraser, south past Victoria and then west out to the Pacific[†]. We also know that it takes a few weeks to get there, and so objects floating in the Strait probably also take about that long to leave our waters. But we would like to get a better idea of what the mean speed of their drift is, and how this mean might change from time to time.

[*] if it didn't the whole area would eventually be full of fresh water only, but it isn't

[†] we think that very little goes around the northern tip of Vancouver Island because there is only a narrow channel separating it from the mainland there

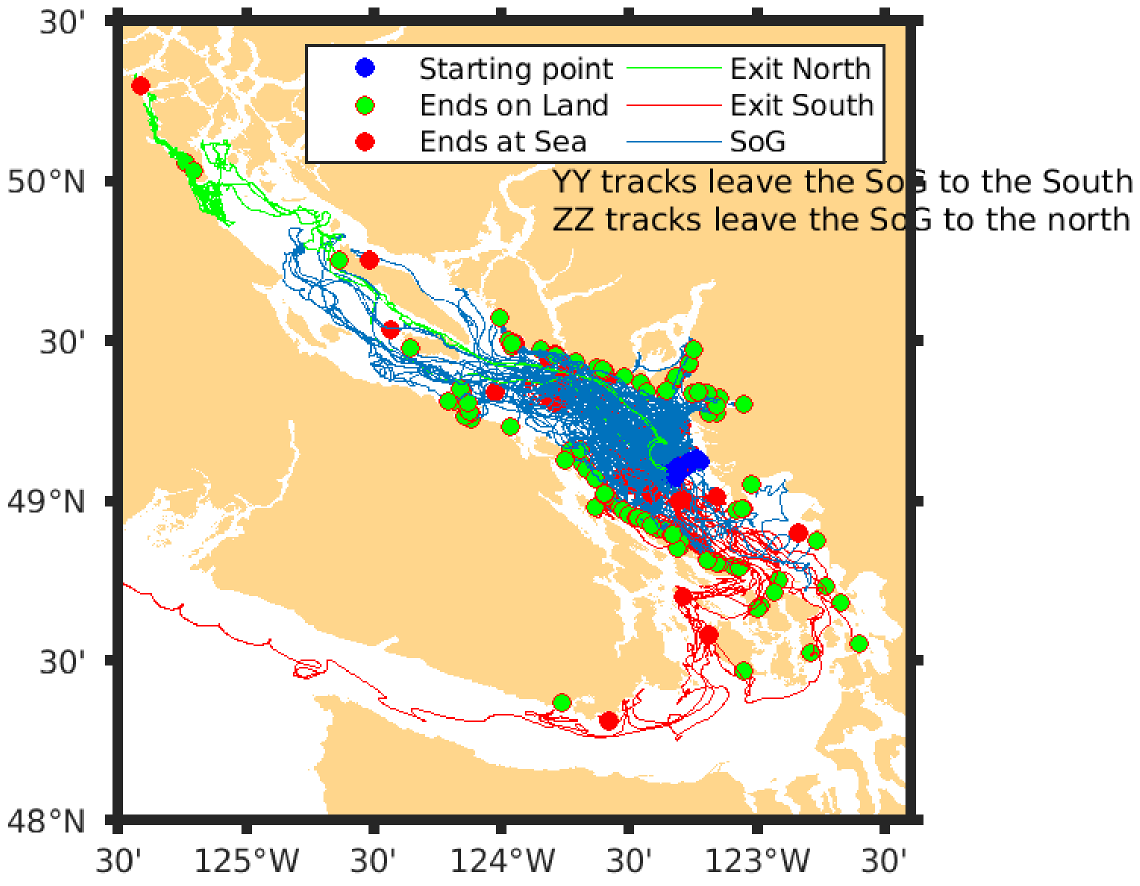

Figure 1: "A Summary Plot": You need to hand in something like for Part 1 (pick a land colour that you like).

Over the past few years, the ODL drifters project (`drifters.eoas.ubc.ca`) has been releasing GPS-tracked drifters into the Strait of Georgia near the mouth of the Fraser River, in order to better understand how the surface water flows out to the Pacific. In the assignment, you will analyze some of the data from this project.

Part 1: A Summary Plot

The `Drifter_dataset.mat` file contains a structure array `D` with 153 drift tracks that all start in the vicinity of Sand Heads, at the mouth of the Fraser River. These include tracks where the drifter never touches land and eventually dies at sea (can be identified by `firstLifeTime==0`), and tracks where the drifter grounds (in this case `firstLifeTime>0` gives the time interval at which this occurs). For times between `launchDate` and `firstGroundDate` (or `endDate` if the drifter doesn't ground), points with `atSea==1` are valid points; if any `atSea~=1` then the corresponding `mtime,lat` and `lon` points are not valid for various reasons. Sometimes grounded drifters refloat after some time ashore and drift further, but we will ignore any data that were acquired after a grounding.

First, write code to show all the drift tracks up until either the point of first grounding, or the end of the track if the drifter does not ground. The track lines should be coloured to indicate the location of the track endpoint:

- Tracks that exit the northern Strait (i.e. with end point west of $125.19°$W, north of $50.0°$N) should be green.
- Tracks that leave the southern Strait (roughly, with end points south of about $48.78°$N latitude, but note that one track that completely leaves the Strait this way ends up north of this latitude and you should account for this) should be red.
- Tracks that end within the Strait of Georgia should be light blue.

As well as showing all the tracks, you should also label the starting and end points:

- Label the track starting points with dark blue markers (these should all be near the mouth of the Fraser River).
- Label track end points at time of first grounding (if they ground) with green markers.
- Label track end points if the drifter never grounds with red markers.

You should use `if` statements and logical operators to determine into which category a given track belongs.

Finally, add to the plot a `text` line that states how many tracks fall into each category - something like

```
          100 tracks ground
          21 tracks leave the SoG to the south
          2 tracks leave the SoG to the north
```

(or whatever the numbers are)

To hand in Part 1, provide the code and the plot. It should look something like Figure 1 (with the added text). Also hand in the `partner.m` file on Connect.

Note - READ THE REST OF THE ASSIGNMENT so you can see what you have to do for the next part while answering this part. Also, the **Handy Tips** at the end will be helpful.

Part 2: Statistics

Now we want to get some summary statistics.

1. First, make a `histogram` plot showing the time to grounding, for all drifters that ground. Calculate the mean and standard deviation of this time, and add this to the plot.
2. Second, plot the drifter latitude (a proxy for the distance up-Strait or down-Strait of the deployment location) as a function of time. Use the same line colours and end point markers for grounded and at-Sea end points that you used in Part 1.

   Now, add a thick line that gives the MEDIAN drifter latitude, every half-day, from the deployment time to 15 days after deployment. That is, at 0.5 days after deployment, find the latitude of all drifters still afloat and alive, and take the median. Repeat for 1 day after deployment, 1.5 days after, and so on.

   Also, find the MEDIAN ABSOLUTE DEVIATION (or MAD, see `mad` to calculate this) of the latitudes, i.e.

   $$MAD(X) \equiv median(|X - median(X)|) \tag{1}$$

   and plot the $median \pm MAD$ as well.

   Note that the median and the MAD are somewhat similar to the mean and standard deviation, but are less sensitive to outliers.
3. Finally, answer the questions:

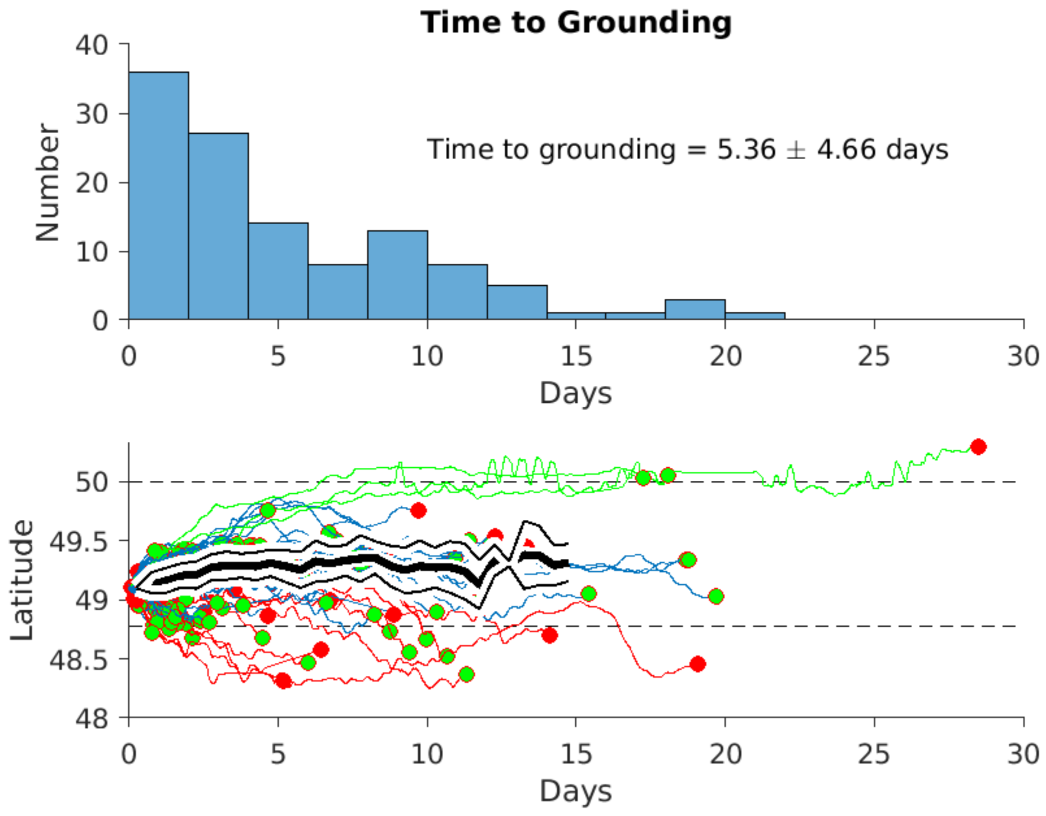

Figure 2: “Statistics”: You need to hand in something like for Part 2

(a) Are the drifters moving seaward (on average)?

(b) How long does it take for a drifter to move from the Fraser River to the Pacific

To hand in Part 2, we need the code and the resulting figure, it should look something like Figure 2. Write the answers to the questions somewhere. NOTE - if you had errors in Part 1, we recommend handing in a corrected version with Part 2.

Handy tips! (You will want to read this)

- The drifter data is contained in a structure `D` in the file `Drifter_dataset.mat`. Drifter data is complicated, as drifters can ground and refloat. The data format is:

```
%    id  - integer: a unique identifier (e.g., labelled # on drifter body)
%    design  - integer: drifter design code (1-6)
%    tzone - string: Time zone
%
%    mtime - float vector: matlab times (increasing)
%    lon   - float vector: longitudes (degrees E)
%    lat   - float vector: latitudes  (degrees N)
%
%    comment - string: descriptive information (deployment #, fate,
%              other info that might be useful.
```

```
%
%    atSea - integer vector: flag for each point, classifying it as:
%                    1 - good - at sea, freely floating (valid)
%                    2 - bad - at sea but trapped in rocky intertidal
%                              (floating but not free)
%                    3 - bad - on land (grounded, test data, etc.)
%                    4 - bad - at sea (large GPS error, on ship, etc.)
%    endsOnLand  - logical: flag
%                     1 - grounded at or just after last atSea==1 point
%                     0 - track ends at sea
%    foundOnLand - logical: flag
%                     1 - Drifter found by human
%                     0 - lost
%
%    launchDate  - float: matlab time for first valid point (atSea==1)
%                         (time of first point if no atSea==1)
%    launchDateS - string: first valid date (human readable).
%    endDate    - float: matlab time for last valid point  (atSea==1)
%                        (zero if no atSea==1)
%    lifeTime   - float: decimal days from launchDate to endDate
%
%    refloated  -  logical: flag
%                     1 - refloated after grounding (atSea==1 points after
%                          one or more atSea==4).
%                     0 - no grounding occured before last atSea==1 point
%
%    firstGrndDate - float: matlab time for first grounding
%                   - matlab time of first of a string of atSea~=1, unless
%                    the last point in the record has atSea==1 and
%                    endsOnLand==1 in which case it is the time of the last
%                    point.
%                   - 0 if (endsOnLand==0 & refloated==0)
%    firstLifeTime - float: decimal days from launch to first grounding
%                   - 0 if endsOnLand==0 & refloated==0
```

- MATLAB date-format is a number which represents decimal days since Jan/1/0000. Useful functions that can be used to manipulate this include `datestr`, `datenum`, and `datevec`.
- You will need a map. A very simple one can be drawn using

```
C=load('BCcstlne.mat');
plot(C.ncst(:,1),C.ncst(:,2),'color',[0 .5 0]);
axis([-124.4 -122.2 48 50]);
```

A nicer one (that takes longer to run) can be drawn using

```
C=load('BCcstlne.mat');
for k=1:length(C.k)-1,
 ii=C.k(k)+1:C.k(k+1)-1;
 patch(C.ncst(ii,1),C.ncst(ii,2),[0 .5 0],'edgecolor','none');
end;
```

Bonus points for making the map more ‘map-like’ (labels not in ugly decimals with minus signs, titles, proper data aspect ratio). Add comments around the code that draws the map telling us what you have done to make it more ‘map-like’ to get assessed for the bonus.

- I always like using `set(gca,'box','on')` and `set(gca,'tickdir','out')` to make plots nicer.