1 Learning Goals

- To understand and use different interpolation algorithms
- To gain more familiarity with with loops and vectorization
- To use MATLAB built-in functions (including `interp2`)
- To gain more experience in handling complex data sets
- To learn how to scavenge code from other code

2 Introduction

During coastal upwelling, dense deep water is brought up over the shelf-break onto the shallow shelf. In theory, we then see horizontal gradients of density - at the same depth, water is heavier inshore. Heavier water is usually more saline, so we also see saltier water inshore. In this lab we are going to examine some salinity data from sections south-west of Vancouver Island. Ultimately, we want to answer the question: Is water more saline on the shelf at this time? There are two steps to this process:

- Interpolate data along oceanographic sections to a common distance from the shelf edge.
- Compare the mean profiles at a distance of 0 and 20 km from the shelf edge to see if they are statistically different.

3 The lab

3.1 Preliminaries

Download the folder `lab11`. You may have to double click on it to unzip it.

Run the script `main_script.m`. It calls several functions including `makeScatterPlot.m`. These are all in the folder that you've downloaded and unzipped. Make sure you can follow the main script. You should also be able to follow all the code in the functions. As in many real problems it is not necessary to know the details of how each function works but you must know how to use them properly.

The script plots the original salinity-depth casts (profiles), which were taken along a series of tracks back and forth across the shelf edge, at their distance from the shelf break (the 200 m contour line). You can see that each track has data taken over a distinct distance range, different from that of other tracks, and has data gaps at various distances. The script will produce a file called `lab11_data.mat` that contains the data needed for your part of this lab.

Note: The function `makeScatterPlot` draws the points in the last figure with a point size of `8`. See the `scatter` command in the function and type `help scatter` to see how it it works (`scatter` is a built-in MATLAB function). You may need to change the point size (to make the points large enough to see easily but not so large that they overplot each other), depending on your version of MATLAB.

Now, write a script `lab11_init.m` which begins by loading the file `lab11_data.mat`. Check that all is well by cutting and pasting code from `main_script.m` to reproduce the final figure made in that script. This should be `figure(1)` in your `lab11_init.m` script. Make sure the x-axis of the bottom two sub-panels is always labeled. Do not draw the map that is in the original script. You can use much of the code structure from `main_script` for parts 3 and 4 of the lab. Units of salinity are 'ppt' (parts-per-thousand or g/kg); average seawater is 34.7.

3.2 Interpolation (2D interpolation)

Before you do this part make sure you have looked at the `week11_tues_notes_interpolation.pdf` slides, and run and look at the `demoweek11_tues.m` script.

Interpolate the salinity-depth-distance data from each track (use a loop) onto depth profiles that are evenly spaced in distance from X = -20 km to X = +40 km in 0.5 km increments. The depth data are already evenly spaced. Use `interp2` and type `help interp2` or `doc interp2` to see how it works. It is very similar to `interp1` in the demo script. Your code will be cleanest if you use the built-in function `meshgrid` (we saw this earlier in the course) to set up new `X` and `Y` matrices for your interpolated data. This is useful because you'll need (dist,depth) pairs for every measurement when plotting your results. Investigate the results of using different interpolation methods as follows:

1. Use `nearest neighbor` interpolation. Plot your interpolated data in Figure 2 using `makeScatterPlot`. Figure 2 should have a similar layout to Figure 1. You'll see that the MATLAB scatter function colors points with `NaN`. See if you can figure out how to plot only the points that are NOT `NaN`. *Hint:* `isnan` is helpful.
2. Change the interpolation method to `linear`. See `help interp2`.
3. Repeat one more time, changing the interpolation method to `spline`. (MATLAB will produce a warning for each track that NaNs are being ignored: don't worry about this.). You should see that in this case the spline option results in clearly erroneous extrapolations.

3.3 Taking a depth profile at a common distance from the shelf edge for all tracks

Now we will obtain an interpolated salinity versus depth profile at a single perpendicular distance, X = 0 km, from the 200 m contour. Set up a variable `sal00_pro` that has 10 columns: each column should contain the interpolated salinities for a given track at X = 0 km, computed using the `linear` option in `interp2`.

Now repeat, getting the depth profiles at a distance of 20 km from the shelf break. Put in `sal20_pro`.

Set up Figure 3 with three subplots. Plot the interpolated salinities on the x-axis and depth on the y-axis for all 0-km profiles in the left subplot. Plot the 20-km profiles in the middle plot. You can plot the profiles for all the tracks using either a loop or a single plot command. Provide informative labels / title. Note that `axis ij` will plot 0 m depth at the top of the y-axis and have depths increasing downwards which makes a nice-looking plot.

Now compute the mean salinity and the standard deviation in salinity as a function of depth at each distance. Plot the mean 0-km salinity on the x-axis and depth on the y-axis in the right subplot as a thick black solid line and plot the mean $\pm$ 1 standard deviation as thin solid lines on either side of your black curve. Plot the mean 20-km salinity as a thick red line and plot the mean $\pm$ 1 standard deviation as thin red dashed lines on either side of your black curve.

You can use the `mean` and `std` functions, or even better the `nanmean` and `nanstd` functions. Be sure to read the `nanstd` carefully – if you use the option where you specify the dimension of the input array to which the `std` should be applied, you should set the `flag` argument to "0".

The question you should be able to answer from this plot is - given the scatter in the data, are the mean profiles from the two locations different by more than the standard deviation in the data? Write a brief message to the screen using `fprintf` that says whether the two profiles are different or not.

4 To Hand In

A script called exactly `lab11.m` that loads `lab11_data.mat` and produces only:

1. Figure 2 above, (i.e., the section plot from Section 3.3, using linear interpolation), and
2. Figure 3 above (i.e., the profiles with mean and standard deviation, from Section 3.3).

You can assume we have working versions of all other functions needed for this lab...and remember to add your partner name and time spent as usual.