Chapter 3

Getting Started with Matlab

In this lab, you will practice reading in some data from a text file and making line graphs from it. You will also learn how to read in two dimensional or three dimensional data from a netCDF file, and make color-shaded maps from it, with coast outlines added. All the data we use in this course will be either in text or netCDF form. netCDF is widely used for storing and disseminating earth science data and atmospheric or oceanic simulations. It's very worthwhile to learn how to deal with it.

First log into `hemulen` in an xterm. Now start up matlab on hemulenby typing `matlab`.

3.1 Reading data from text files

First we'll plot up some data from a weather balloon. This data is in the directory `/home1/geo232/data/radiosondes` so first type `cd /home1/geo232/data/radiosondes`. to set the directory. If you type `ls` you'll get a list of what's there. Note that Matlab allows you to execute standard Unix commands like `ls` from the Matlab prompt.

Each file with a `.dat` extension has data from one ascent. The corresponding `.apa` file has information about where and when the sonde was launched, if you are curious. You can look at the text within matlab by typing `type 93032700.apa`, for example. If you do `type 93032700.dat` you'll see a bunch of columns of numbers To see what the various columns mean, do `type readme.1st` to look at the description file. From this you should see that column 3 has the altitude of the data point,whereas column 5 has the temperature in degrees C.

To load the data from 93032700.dat into a matrix called `sonde`, type

```
sonde = load('93032700.dat');
```

The argument is the filename. The semicolon tells matlab not to echo all the data to the screen. `load` will read in either space-delimited or tab-delimited text data, which will handle most of the cases you are likely to encounter.

Now you have the data in an array called `sonde`. Type `whos` to look at its dimensions. We can make some new variables that are easier to work with. To store the height in a one-dimensional array called "z", type:

```
z = squeeze(sonde(:,3));
```

Note matlab's convention for which index indicates the column! Now, following the example, load the temperature into an array called `T`.

You can check your results by just typing `T` or `z`.

3.2 Making a line graph

Now, to make a graph of $T(z)$, just do `plot(T,z)` and the plot should appear on your screen. z will be on the vertical axis, and T on horizontal, as is the usual meteorological convention. Do `help plot`, `help title`, `help xlabel` and `help ylabel` to see how to pretty up the graph.

Exercise: Do a plot of $T(z)$ where T is given in degrees Kelvin instead of degrees C.

Exercise: Plot $T(p)$, where p is the pressure. Plot T as a function of $-log(p)$. Plot relative humidity as a function of height.

3.3 Reading netcdf data and making contour plots

Note that the netcdf reading package is not part of the standard Matlab installation, so if you are ever using Matlab on a computer other than hemulen, you may need to download and install the mexCDF package.

To prepare your matlab session for using netCDF, type `ncstartup`. The only data reading command you will need for this course is `ncload`. If Matlab complains when you try to execute `ncload`, it means you haven't set the `MATLABPATH` environment variable properly.

To get to the directory with our netcdf files, type `cd /ray1/geo232/data`. You can do an `ls` to get a listing of what's in the directory. There is a file there called `Jones_climatology.nc`. This file contains a climatology of long-term mean surface temperature for each month of the year, on a latitude-longitude grid. A useful netcdf utility is `ncdump`. This command gives you a summary description of what's in the file. To check out the climatology file, type `ncdump Jones_climatology.nc`. You can also use ncdump from the Unix command line when you're not in Matlab, but then you need to remember to type `ncdump -h` instead of just `ncdump`. The "-h" flag tells `ncdump` to just give you the summary or "header" information from the file, rather than the entire contents data and all.

To load in all the data from the file, type

```
ncload Jones_climatology.nc;
```

If you leave off the semicolon, Matlab will try to type all the data to the screen as it's loading it in. This is a general feature of all Matlab commands. Now type `whos` to see what you have gotten. There is a longitude scale called `X`, a latitude scale called `Y` and a set of twelve monthly mean surface temperatures in the array called `temp`. This file is small enough that you can just load in every variable. `ncload` also allows you to read in just selected variables, which can be necessary for larger data sets. For example

```
ncload Jones_climatology.nc,X,Y;
```

just loads in the latitude and longitude scales. It is also possible to load in just selected parts of an array (e.g. just one month of data), but this requires much more knowledge of the netCDF calls in Matlab. You can find out how to do this by typing `help netcdf`, but in this course we will be using Python instead for such data manipulation, when necessary.

Now, to make a color shaded map of the data for January, use the command:

```
contourf(X,Y,squeeze(temp(1,:,:)))
```

By default, you get 6 contour bands. If you wanted 12, you'd do:

```
contourf(X,Y,squeeze(temp(1,:,:)),12)
```

Note that, by default Matlab array indices are "1-based", so that the first element, corresponding to January, has index "1" rather than zero. Many other software packages and computer languages start their indexing with zero instead, and this is

an issue you always need to keep in mind. It is easy to introduce bugs by mixing up one-based and zero-based indexing. Historically speaking "Fortran culture" uses one-based indexing and "c-culture" uses zero-based indexing.

Now suppose you want to add coastlines, so it's easier to relate the data to geographical location. You can load in the coastline data by typing `load coast`. If you do a `whos`, you'll see you've just loaded in one-dimensional arrays called `lat` and `long`. *Warning: if you are ever analyzing a data set which uses one or both of these names for its scales,you will need to copy something to a renamed variable to avoid problems.* Now look at the data for the longitude to see whether it is based [-180,180] or [0,360]. Oops! Looks like its the former, whereas our data we've just plotted on the contour plot uses [0,360]. We'll have to take this into account soon.

Now you can overlay the coast outlines on your plot. To keep the previous plot from going away, first type `hold on`. Then use

```
plot(long,lat)
```

Because the coastlines are on a (-180,180) longitude scale, the preceding plot command only gives you the right half of the coastlines; the rest are clipped off. To get the coastlines on the left half of the plot, just do:

```
plot(long+360.,lat)
```

When you're done plotting the coasts, type `hold off`.

You can add a color bar telling you what the colors mean by typing `colorbar`.

An alternate way to tell what data values are associated with the colors is to add labels to your contours. To do this, you need to save some extra information when generating the plot:

```
[c,h] = contourf(X,Y,squeeze(temp(1,:,:)));
clabel(c,h)
```

You can replace `c` and `h` with any variable names of your choice. They are just places to store information about the contour lines. As long as you have stored these values, you can add contour labels at any point in the process, e.g. either before or after putting in continent outlines.

Now let's do some simple data manipulation. Compute an array giving the annual average temperature at each point on the latitude-longitude grid,i.e. the average of the 12 fields stored in `temp`. You can certainly do this by explicitly adding up all 12 elements,but can you also write a small loop that does it? Can you write a few statements that do the average in a loop-free way?

Now, take any one month (say, January), and compute the zonal mean, i.e. the average along latitude circles. This will be a one-dimensional array. Plot it vs. latitude to see what it looks like.

Exercise. Make a graph of the temperature vs. time for a point near the Equator, and for some points near the North and South poles. The time variable in this file is called `T`, and corresponds to the month of year.

3.4 On your own

In the data directory, you'll find a much bigger dataset of observed surface temperature, called `ncep_monthly_surfT.nc`. This has 484 months of monthly mean surface temperature, starting with January, 1958. Play around with it, to see what you can find. How much does one January differ from another?

Chapter 4

The Earth's Radiation Budget, Part I

In this lab you will analyze observations of the Earth's radiation budget. The observations are from the Earth Radiation Budget Experiment (ERBE). The ERBE dataset consists of observations of the infrared and solar energy budget of the Earth taken by instruments aboard a constellation of satellites observing the Earth from outside the atmosphere. The dataset you will analyze contains monthly means of the observations. The data for each year of ERBE is contained in a file named `ERBEyy.nc` in the data directory, where `yy` is the two-digit year code. For example, `ERBE86.nc` contains twelve monthly fields for the year 1986, for each of the ERBE data fields. You can do an `ncdump -h` on this file to check what is in it and to find the dimensions. Each data field is in a three-dimensional array with dimensions `(month,lat,lon)`. You can read the entire file directly into Matlab using `ncload`.

ERBE compiled radiation budget data for both the total sky and for clear-sky subregions separately. This makes it possible to identify the effects of clouds. Since difficulties in quantitatively representing cloud effects stand in the way of many climate calculations one would like to do, improving the understanding of clouds and providing a basis for validation of theories and models was one of the main objectives of ERBE. In Part I you will deal with total-sky data. You will examine cloud effects in Part II.

The all-sky fields are:

- `NET` : The net radiation balance (Solar absorption minus longwave emission), with the convention that positive numbers correspond to a net gain of energy by the Earth.
- `LW` : The longwave emission from the Earth. Add this to `NET` to get the

absorbed solar radiation.

- `SW` : The solar energy exiting from (reflected by) the Earth.
- `alb` : The albedo of the Earth.

All radiation fields have units of W/m^2, and the albedo is dimensionless. The corresponding clear-sky fields you will look at in Part II have the same names but with `CS` appended (e.g. `albCS`).

As in just about any dataset derived from observations, there are places and times where the data is missing. For example, it is impossible to measure albedo in the Arctic or Antarctic night, because there is no sunlight to reflect. At other times, a satellite may have been missing, or the instrument may have had a failure. Missing values are indicated in the arrays by the value `-1e32`. Missing data is an unfortunate fact of life, which will complicate the calculation of many of the statistics you would want to get from the data.

Start up Matlab and load in an ERBE dataset, proceed with the analysis. In Matlab, the index `1` represents January while `12` represents December.

4.1 Browsing the data

Use `contourf` to take a look at the pattern of net radiation and of OLR and albedo for one month in each season. You need not save the figures to include in your lab report, but please describe in general what you find, along with some typical numbers characterizing the fields.

Note that because of the large value of the missing-data code, the automatic intervals picked out by `contourf` are pretty useless. To get around this, you need to specify contour levels by hand.

For example, a reasonable range for the net radiation field is -200 to 200 W/m^2. To set up an evenly spaced vector of 30 contour levels covering this interval, and then do a contour plot for January execute

```
clevels = linspace(-200.,200.,30);
contourf(lon,lat,squeeze(NET(1,:,:)),clevels);
```

You can use any name you want for the vector in which the levels are stored. You can also add continent outlines, but note you will have to copy `lat` to another vector with a different name first, to avoid it being over-written. Also, the ERBE data is on a (0,360) longitude scale, so you'll have to remember to deal with that as well.

4.2 Radiating temperature and greenhouse effect

For March, make a map of the effective radiating temperature of the planet. This temperature is the temperature an ideal blackbody would have to have in order to produce the same longwave emission as observed. It is defined by $T_b = (OLR/\sigma)^{.25}$. Make a map of the actual surface temperature for March of the same year, from the NCEP surface temperature dataset. Compare the two and discuss the differences in terms of what you know about the greenhouse effect.

There's another way to examine the influence of the greenhouse effect on surface temperature. Compute the March solar absorption S_{abs} and then compute the temperature a blackbody surface would have in equilibrium with this absorbed solar radiation, if it were not affected by the overlying atmosphere. Discuss how this temperature compares to the actual surface temperature. To interpret this calculation, you will need to think somewhat about the effect of horizontal heat transports – i.e. the fact that the atmosphere and ocean can redistribute heat horizontally. Why is it better to do this kind of comparison for March (equinox conditions) rather than January (solstice conditions)? It would be even better to do the comparison for annual mean conditions.

These maps, and selected others from subsequent sections, should be included in your lab report. You can use the `File` menu on the Matlab graphics window to export the graphics to `.gif` or `.eps` format for inclusion in your favorite word processor document. If you are writing in TeX, you should use the `.eps` format, which can be imported with the `epsfig` package. Note that the files you save will be in your directory on hemulen, so you will need to download them using `ftp` or `scp` to whatever machine you are using to write up your lab report.

4.3 Some mean statistics

Compute the mean albedo of the Earth for a year of your choice, averaging over all months and all latitudes and longitudes. Note that the mean over latitudes needs to be area-weighted, since a lat-lon box has different area at the Equator than at the pole. A bit of sample Matlab code that does the average over latitude for a given month and longitude follows:

```
norm = 0.
avg = 0.
for i = 1:length(lat)
  norm = norm + cos(pi*lat(i)/180.);
  avg = avg + alb(imonth,i,ilon)*cos(pi*lat(i)/180.);
```

```
end
avg = avg/norm;
```

Note theconversion from degrees to radians in the cosine. Now, this would work fine if there were no missing data, but if even one missing data code is included, it ruins your whole computation. A modification of the above code that takes care of the problem is:

```
norm = 0.
avg = 0.
for i = 1:length(lat)
 if alb(imonth,i,ilon) > -1.e30
   norm = norm + cos(pi*lat(i)/180.);
   avg = avg + alb(imonth,i,ilon);
 end
end
avg = avg/norm;
```

You can generalize this in the obvious way to the average over all months and longitudes. The simplest implementation is to add up the values in the arrays one element at a time, checking for whether the value is invalid before adding it in. This method runs much more slowly than more clever methods making use of Matlab's array syntax, but the latter are a little tricky to get used to. The brute-force simple approach works adequately fast. Compute the time series of albedo as a function of month, and make a plot.

Now we will estimate the net meridional heat transport by the atmosphere-ocean system. First compute the annual mean zonal mean top-of-atmosphere radiative imbalance. You do this by averaging the field `NET` over longitude and time, to obtain a function we'll call $F(\phi)$, where ϕ is the latitude. Remember to eliminate missing data codes from your average. The value of F at latitudes where there is no valid data will come out infinite. Make a line plot of $F(\phi)$ and discuss it. You may need to re-set the vertical axis scale if there are infinite values, or if you didn't properly eliminate missing-value codes. Type `help axis` to see how to reset the axis scales on your graph.

Finally, we will turn $F(\phi)$ into an energy flux in Watts. The numbers get big, so the usual units used for measuring such planetary fluxes are Petawatts (1 PW = 10^{15} W). Let $H(\phi)$ be the meridional flux and a be the radius of the Earth. Then H is obtained by solving the equation:

$$\frac{dH}{d\phi} = 2\pi a^2 cos(\phi) F(\phi) \tag{4.1}$$

This is written assuming the latitude ϕ is in radians. The factors multiplying $F(\phi)$ take into account the element of area on a sphere, between closely separated latitude circles.

At (actually near) one pole, you apply the boundary condition that $H = 0$ since the heat has nowhere to go. Then you integrate up toward the other pole. If the planet as a whole is in balance, the heat flux will also be zero there (why?).

Equation 4.1 can be solved in Matlab using a numerical integration, but as a crude approximation, you can simply start with $H = 0$ near the South pole, and then proceed to sum up the right hand side multiplied by $d\phi$, which for ERBE is $2.5\pi/180.$ radians. It may work better to start at the Equator (where there is less missing data) and then work outwards to both poles; then you can add in an appropriate constant to make the flux nearly vanish at the poles.

Compute $H(\phi)$ and plot it. At what rate is energy exported from the Tropics (i.e. across 30N and 30S)?

Chapter 5

The Earth's Radiation Budget, Part II

The net effect of clouds on the Earth's radiation balance results from the sum of two strong but opposing effects: the albedo effect, which reduces the absorbed solar radiation and exerts a cooling influence, and the cloud-greenhouse effect, which reduces OLR and exerts a warming. For this reason, accurate representation of cloud effects is one of the major challenges in climate science. It comes as no surprise, then, that the ERBE experiment put great emphasis on observing the cloud effects. It did this by subsampling each "scene" and classifying it into cloudy and clear-sky subscenes. By looking at radiation in the clear sky scenes, it estimates what the radiation (visible or longwave) would be if the whole scene were clear. The difference between the clear-sky and all-sky radiation values is the cloud effect. For example, in the netcdf file, the variable NET is the all-sky top of atmosphere net radiation budget, whereas NETCS is the clear-sky value. Similarly, LWCS is the clear-sky outgoing longwave radiation, whereas LW is the all-sky value. There is a fair amount of missing data in the "..CS" fields, because sometimes there aren't enough clear-sky pixels to accurately estimate clear-sky radiation. Sometimes also the satellite data processing algorithm can't accurately identify cloudy pixels. This happens particularly over ice-covered regions, because both clouds and ice are very reflective.

In this lab, you will examine the cloud effects, using the ERBE dataset.

5.1 Effect of clouds on the net radiation budget

Get the net radiation budget for March, with and without clouds. Subtract the clear-sky budget from the full budget, to get the effect of clouds. With this definition of cloud effect, negative values would mean that clouds reduce the net radiation driving the Earth's climate. Make an image and look at the map of cloud effects. Where are clouds acting to warm the climate? Where are they acting to cool the climate? Where does the effect add up to nearly zero?

Do the same for the other months, and look at selected months to get an idea of the seasonal cycle of the cloud effects. Average all the months to obtain the annual average cloud effect. As in the previous lab, you will need to take into account missing data when you do this. Take the zonal mean of the cloud effect and make a line graph to show how the net influence of clouds depends on latitude.

5.2 Albedo vs. greenhouse effect of tropical clouds

If everything went well in the previous section, you will have noted that clouds have relatively little net effect in the tropics. Is this because there aren't any clouds there, or because the cloud albedo and greenhouse effects nearly cancel? To see this, let's look at the breakdown of the effect of clouds more closely. To make things simpler, we'll deal only with the January data, though you are encouraged to look at other months as well.

First get the cloud greenhouse effect, defined as LW(total) - LW(clear). When this is negative, clouds reduce OLR; i.e. they have a warming effect on the climate. Compute this quantity and discuss.

Now compute the cloud albedo effect i.e. the solar, or shortwave, cooling that results from clouds reflecting sunlight back to space. Recall that SW is the shortwave radiation flux that has been reflected by the planet; SWCS, similarly, is the clear-sky reflected flux. You could use the albedo to get the corresponding absorbed radiation fields, but since you are only interested in the impact of clouds here, it suffices to simply subtract SW from SWCS to get the cloud-induced increase in reflection – which is also the decrease in solar absorption. Carry out the subtraction. Where does this tend to cancel the cloud warming effect? *Warning:* You have to be careful about missing data when interpreting your results. If data is missing from both SW and SWCS at some gridpoint, then the difference in values will be the difference of the two missing data codes, i.e. zero. You have to be aware of this when you see regions where the "cloud effect" comes out exactly zero.

Based on these calculations, answer the question raised in the first paragraph

of this section.

5.3 How much of the Earth's albedo is due to clouds?

As a function of latitude, estimate what part of the Earth's albedo is due to clouds. You can do this most easily by examining the ALB and ALBCS fields.

5.4 How much warmer or colder would the Earth be without clouds?

WQ2002: This section is optional, for extra credit. Do it if you have time and interest.

If the Earth had no atmosphere, then sensitivity to a change in the radiation budget could be computed as follows

$$F = \sigma T^4, F + \delta F = \sigma(T + \delta T)^4, \delta F \approx (4\sigma T^3)\delta T \tag{5.1}$$

whence

$$\delta T = \frac{1}{4\sigma T^3}\delta F \tag{5.2}$$

This yields a sensitivity coefficient of about $.2K/(W/m2)$ for $T = 285K$; that is, it takes about a $5W/m^2$ change in net radiation to cause a $1K$ change in the surface temperature. Extensive studies with moist atmospheres and general circulation models indicate that a more correct value would be around $.5K/(W/m^2)$.

Based on the latter sensitivity coefficient, and the annual mean results of Part 1, discuss how the temperature of the earth would change if clouds were eliminated. How would the change be distributed geographically? Which areas would experience the greatest change? Which the smallest? Which seasons do you expect to experience the greatest change?

Now suppose that the albedo of tropical clouds changes by 10% relative to its present value (e.g. going from 30% albedo to 33% albedo), while the cloud greenhouse effect remains fixed. How much would tropical temperature change, assuming lateral heat fluxes in the atmosphere remain fixed?