Working with Scientific Data Sets in MATLAB: An Exploration of Ocean Color and Sea Surface Temperature

|  |
| --- |
| **Learning Goals**Through this exercise students will be introduced to the concepts of remote sensing, ocean color and using MATLAB to import and manipulate and visualize scientific datasets. Students are guided through a command lines and script development in the MATLAB environment to 1) download ocean color data, 2) import data, 3) sub-scene the global data set into an area of interest and 4) plot annual averages within the identified “sub-scened” area. This activity is meant to serve as an introduction to using MATLAB with ocean color data. Some synthesis of ideas and critical thinking is required when students are requested to apply the scripted lesson to another dataset of sea surface temperature.**Context for Use**This is a computer lab activity that will take approximately 2 hours or longer, depending on year specific efficiency and individual capabilities of the students. Students are then challenged to repeat the process they have just completed, utilizing a different data set. The exercise was designed for students that do not have any prior experience in MATLAB. Additional resources (e.g. Attaway text) are provided to students so that they can become more familiar with MATLAB before undertaking this activity. Students should be familiar with the concept of a matrix, though they are not required to have had Linear Algebra. This exercise is given in a course called Ecosystem Assessment. We go through a series of exercises and case studies, where students gather environmental data about a particular marine ecosystem and align data collection and analysis with an ecosystem-based management plan.**Description and Teaching Materials**A full description of this activity is included in the attached file, "Working with Scientific Data Sets in MATLAB: An Exploration of Ocean Color and Sea Surface Temperature." In this activity, students are guided through a series of steps including (1) accessing online ocean color data, (2) reading and georeferencing the data with MATLAB, (3) sub-setting the data, (4) plotting, and (5) creating a batch script to repeat the above steps on multiple files. With each step, students are asked specific questions to help them understand the data sets they are working with and the purpose of the MATLAB commands and functions they are implementing. At the end of the guided activity, students are presented with an assessed challenge task. This tasks involves performing the same set of steps using a sea surface temperature data set. The activity assumes that the College or University has a MATLAB license that students can access. Data for this activity are freely available and easy to download from the supplied websites. This exercise is not posted anywhere else and has not yet been thoroughly evaluated and tested. **Teaching Notes and Tips**In the challenge task most students have problems orienting the matrix in the appropriate geographic space, the color scale values (very different for temperature) and pointing MATLAB to the correct directory for loading the appropriate files. To overcome many of the above difficulties the activity is run for small groups (<15) allowing the instructor to give the students individual attention. Students are also encouraged to work with one another but are required to submit their own work.**Assessment**Upon completion of the guided and challenge tasks students provide plots of annual averaged ocean color, or chlorophyll (2002-2015) and sea surface temperature (2002-2015) for inclusion into an “ecosystem assessment” report. They use this data to describe chlorophyll and temperature patterns in their chosen region and try to determine what is causing this inter annual change. Both the guided and challenge tasks are assessed with a marking rubric.**References and Resources**1. **Background information on ocean color data**

http://oceancolor.gsfc.nasa.gov/1. **Ocean Color Data**

http://oceandata.sci.gsfc.nasa.gov/MODIS-Aqua/Mapped/Annual/4km/chlor\_a1. **Description of NetCDF Data format**

http://www.unidata.ucar.edu/software/netcdf/1. **Sea Surface Temperature Data**

http://oceandata.sci.gsfc.nasa.gov/MODIS-Aqua/Mapped/Annual/4km/sst4ftp://podaac-ftp.jpl.nasa.gov/OceanTemperature/avhrr/L3/pathfinder\_v5/annual/day/04km/1. **User Manual for Use of Pathfinder Sea Surface Temperature Data**

ftp://podaac-ftp.jpl.nasa.gov/OceanTemperature/avhrr/L3/pathfinder\_v5/docs/usr\_gde4\_0.html#data set details |

1. Introduction to Remote Sensing

****

Figure 1: A simple model of radiative transfer

Earth remote sensing is defined as the use of electromagnetic radiation to acquire information about the ocean. Land and atmosphere without being in physical contact with the object, surface of phenomena under investigation (Martin, 2014). Because the satellite instrument is not in contact with the object or surface, the properties must be inferred by amount and radiation that is reflected across the full spectrum of visible light, or the visible portion of the electromagnetic spectrum.

As electromagnetic radiation reaches the earth’s surface, molecules and particles of the land, water and atmosphere environments interact with solar energy in the 400-2500nm spectral region through absorption, reflection, and scattering. Some materials will reflect certain wavelengths of light, while other materials will absorb certain wavelengths. Many materials have unique patterns of reflectance and absorption across the electromagnetic spectrum. Before radiances are recorded at the sensor, the electromagnetic radiation has already passed through the atmosphere twice (sun to target and target to sensor). During this process (Figure 1), EMR has been modified by the processes of scattering by air molecules and aerosols and by absorption. The main contributors to gaseous absorption are atmospheric water vapor, carbon dioxide, ozone, nitrous oxide, carbon monoxide, methane and oxygen. Major atmospheric bands, such as those of water vapor centered at approximately 940, 1140, 1380 and 1880 nm, the oxygen band at 760 nm, and the CO2 band near 2006 nm can influence the relative brightness over an image so that it is different to that in the ground scene.

If the EMR interacts with the surface of the water, the reflection scattering and transmission of light across the air-water interface are governed by the Fresnel coefficient of reflection, which is modified by surface roughness, a function of wind speed. Photons can also penetrate the water column and can be absorbed by water itself, photosynthetic plant material, or dissolved organic substances. Multiple scattering effects by both photosynthetic and non-photosynthetic particles can lead to an increased probability of eventual absorption, or photons can be eventually ejected back through the surface. Upwelled light emerging from the surface carries information on all these processes. Because nearly all absorption takes place in the upper two to three attenuation lengths (a few to tens of meters), this information is confined to that region, and quantities derived the water-leaving spectral radiance are generally referred to as surface values.

Remote sensing is a cost-effective way to gather data about environmental conditions and can be used to understand variability and trends. In this exercise will start to gather environmental data for the Tasman Sea, particularly chlorophyll concentrations and temperature of the surface of the ocean. By visualizing these data we can view annual changes to important parameters of the environment and begin to make more informed assessment management decisions.

1. Ocean Color Science

A series of videos, produced by Universite du Littoral - Cote d'Opale has been provided. These videos discuss several topic related to the science behind the acquisition of remote sensing data for studying the ocean. The topics include optical processes, measurements at sea and ocean color in space. Watch the four videos and complete the quiz in the quizzes section of MyLO.

1. Ocean Color and Scientific Datasets

The OceanColor Web was developed and is maintained by the Ocean Biology Processing Group at the Goddard Space and Flight Center of the National Aeronautical and Space Administration in the United States. OBPG serves as the Distributed Active Archive Center (DAAC) for all Ocean Biology (OB) data produced or collected under NASA's Earth Observing System Data and Information System (EOSDIS). This website thus serves as the primary data access portal to the NASA OB.DAAC. The website provides access to several data sets that describe the abundance and distribution of chlorophyll (i.e. phytoplankton) in the global ocean along with several other products such as sea surface temperature, which we will look at later. You can go to the L3 Visual Browser and see all the types of data that are available (http://oceancolor.gsfc.nasa.gov/cgi/l3). To get to the actual data you will need to go to http://oceandata.sci.gsfc.nasa.gov/MODIS-Aqua/Mapped/Annual/4km/chlor\_a. Download this data and put them in an appropriate working directory. You will need the data for this tutorial. The data is compressed is provided in netCDF format.

NetCDF stands for Network Common Data Form. NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. NetCDF was developed and is maintained by the Unidata program at the University Corporation for Atmospheric Research (UCAR). Data in netCDF format is (UniData, 2014):

* “Self-Describing. A netCDF file includes information about the data it contains.
* Portable. A netCDF file can be accessed by computers with different ways of storing integers, characters, and floating-point numbers.
* Scalable. Small subsets of large datasets in various formats may be accessed efficiently through netCDF interfaces, even from remote servers.
* Appendable. Data may be appended to a properly structured netCDF file without copying the dataset or redefining its structure.
* Sharable. One writer and multiple readers may simultaneously access the same netCDF file.
* Archivable. Access to all earlier forms of netCDF data will be supported by current and future versions of the software.”
1. Opening Scientific Datasets with MATLAB

MATLAB provides a series of functions which allow you to interact with NetCDF files. The MATLAB functions enable reading data from and writing data to NetCDF files (known as data sets in NetCDF terminology). The two main functions that you will use to access ocean color data are:

* ncdisp - display contents of NetCDF data source in Command Window. ncdisp(source) displays all the groups, dimensions, variable definitions, and all attributes in the NetCDF data source, source, as text in the Command Window.
* ncread – Read data from variable in NetCDF data source. vardata = ncread(source,varname) reads data from the variable varname in source, which can be either the name of a NetCDF file or an OPeNDAP NetCDF data source.

To read and open the ocean color datasets in Net CDF formats by following the steps below.

1. Open MATLAB.
2. Change your current folder to location where you data files are located.
3. In the Command Window type:

>> ncdisp('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc')

On the screen you will see a printout of the table of contents of the NetCDF file. The output show the content of the NetCDF file broken down into the categories of 1) “Global Attributes,” 2) “Dimensions,” 3) “Variables” and 4) “Groups.”

|  |
| --- |
| **Saving Content Directly from a Website**An alternate methods of accessing the ocean color data is to use the *websave* function in MATLAB. outfilename = websave(filename,url) saves content from the web service specified by url and writes it to filename. The *websave* function returns the full filename path as outfilename. You can obtain the ocean color by executing the following commands. You can copy the url for each file by right-clicking the filename on the website. The following command will save the file ‘A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc’ at ‘http://oceandata.sci.gsfc.nasa.gov/cgi/getfile/’ to a file named ‘A2002.nc.’ These scripts can be incorporated into your looping script for batch processing of these files. However, the process of downloading tends to take a long time, so we will stick with working with the files directly from the computer.url='http://oceandata.sci.gsfc.nasa.gov/cgi/getfile/A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc'filename='A2002.nc'outfilename=websave(filename,url) |

*Q1. What are the x and y dimensions of the chlor\_a variable?*

1. There are three variables that you will need to process and subscene the data. The variables are *chlor\_a* (chlorophyll concentration), *lat* (latitude) and *lon* (longitude).
2. To read the variables into the MATLAB workspace we will use the *ncread* command.

chlor\_a=ncread('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc', 'chlor\_a');

lon=ncread('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc', 'lon');

lat=ncread('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc', 'lat');

*Q2. What are the dimensions (rows x columns) of the chlor\_a variable?*

*Q3. What are the dimensions of the lon variable?*

*Q4. What are the dimensions of the lat variable?*

The dimensions of the variables should appear in the Workspace window of MATLAB. Alternatively you can type

size(chlor\_a);

and the dimensions of the variable will be displayed in the Command Window. Make sure that you have answered Q2-Q4 correctly. This will be important later.

1. We will now visualize the data by using the imagesc command. Read about imagesc in the MATLAB help. Type the following command into the Command Window.

imagesc(chlor\_a);

*Q5. What do you see?*

1. Correct, the Figure 1 Window with a big blue square in the middle and a faint outline of the coastlines of the world. This has happened because the coloring scale of the window is being driven by a set of anomalous outliers. We will need to set the limits of the colorscale within the imagesc commands. First create a variable clims in the command window by typing

clims=[0 0.3];

and, then reissue the imagesc commend by typing the following.

imagesc(chlor\_a, clims);

*Q3. What do you see, now?*

1. What you should see is a figure window that shows the distribution and abundance of chlorophyll as collected by the MODIS Aqua sensor averaged over the year 2002. However, the orientation of the map is not correct. It is rotated 90 degrees to the left. To correct for this we can embed the *rot90* function within the *imagesc* function to orient the image correctly.

>> imagesc(rot90(chlor\_a), clims);

1. Now that the map appears to be oriented correctly we can add the longitude and latitude values to the figure axes.

>> imagesc(lon, lat, rot90(chlor\_a), clims);

1. Check the axes values closely, Even though the longitude and latitude have been added to the axis labels, the values for latitude appear to be backwards (negative latitude in the northern hemisphere, as opposed to the southern hemisphere). To fix this problem we can embed the *flipud* function to flip the chlor\_a matrix up-down. We will then have to set the orientation of the y-axes to normal using the *set* function.

>> imagesc(lon, lat, flipud(rot90(chlor\_a)), clims);

>> set(gca,'YDir', 'normal');

1. The units of the chlorophyll concentration values in the image are mg m-3. You can readjust the clim values and see how the map changes. Type colorbar into the command window and a color bar with values indicating the concentration of chlorophyll. Type…

>> h=colorbar

Make sure that you type the “h=” We will use the h variable that you create later. What you are doing is creating a “handle” so that MATLAB knows which graphic you would like to issue commands to. You could have assigned another letter for the handle under the *imagesc* function and then used this number to refer to that particular graphic in the figure.

1. You can create a title for you figure by using the title function

>> title(‘2002’)

1. Add a label to the colorbar so that we know what the units are in the mg m-3.

>> ylabel(h, 'mg m^{-3}')

**What you will need to do next ……**

1. Using the indexing capabilities of MATLAB and the referencing techniques above, create a subscene of the Tasman Sea using the bounding coordinates of…

subAX=[145 175 -51 -30]; or 145° to 175° east longitude and 31° to 51° south latitude.

1. Then you will create a script that will allows you to process all the images to give us an overall mean chlorophyll concentration and annual concentrations maps for the sub-scened region of the Tasman Sea.
2. Sub-Scene Ocean Color Data

Now, we will continue working with the same imagery, but will extract a sub-scene and use a portion of the imagery for viewing and save it for later analysis. We will create a subscene of the Tasman Sea using the bounding coordinates of…

>> subAX=[145 175 -51 -30]; %or 145° to 175° east longitude and 31° to 51° south latitude.

We will now go through the steps of creating this sub-scene and then develop a script with which we can automate the processing of multiple images through the execution of one script.

***Sub-scene and Image***

1. Load the image, 'A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc’, into MATLAB.
2. Prepare your image as you did last week so that it is georeferenced properly. Follow the procedures above through step 13. You should be able to execute the commands from you Command History window

The commands from above are reproduced below.

>> chlor\_a=ncread('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc', 'chlor\_a');

>> lon=ncread('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc', 'lon');

>> lat=ncread('A20020012002365.L3m\_YR\_CHL\_chlor\_a\_4km.nc', 'lat');

>> imagesc(lon, lat, flipud(rot90(chlor\_a)), clims)

>> set(gca,'YDir', 'normal')

>> title('2002');

>> h=colorbar

>> ylabel(h, 'mg m^{-3}')

1. Create the variable subAX by typing

>> subAX=[145 175 -51 -31];

into the command window.

1. Now use the indexing abilities of MATLAB to find only those areas that fall within the latitude and longitude coordinates of the subAX variable. Type the following commands into the Command window.

xidx=find(lon>=subAX(1) & lon<=subAX(2))

yidx=find(lat>=subAX(3) & lat<=subAX(4))

1. Now extract the data from the global matrix of chlorophyll values for 2002 using your defined indices (xidx and yidx).

>> sub\_chl=chlor\_a(xidx,yidx,:);

1. Now display the sub-scened area. You will need to fix the orientation of the image to view it properly.

>> imagesc(lon(xidx),lat(yidx), flipud(rot90(sub\_chl)), clims);

>> set(gca,'YDir', 'normal');

***Creating a Script and Batch Processing***

You now have 13 more images to process. It would be cumbersome to go through the above procedure 13 more times. In some cases you would want to process daily imagery and at 365 images per year for 13 years. At that rate you’d probably click your finger off before you were finished. MATLAB allows you to develop a script which will repeat your commands across a pre-determined set of files. Please refer to the video you watch earlier in the semester regarding scripts and loops.

[http://au.mathworks.com/videos/writing-a-MATLAB-program-69023.html?s\_tid=srch](http://au.mathworks.com/videos/writing-a-matlab-program-69023.html?s_tid=srch)...

We will simply recycle many of the commands we have already issued by pasting them in a script editor and create a loop to tell MATLAB to run these commands as many times as files we need processing.

1. Press the “New Script” button under the File menu. This will open a blank page in the MATLAB script editor. This is where we will develop our script and then run it by pressing the button with the right facing green arrow on it, or the “Save and Run” button.
2. I always like to start a script by clearing all existing variables and closing all open figures. This usually avoids confusion and reduces the probability of error generation. Type the following command in the new script

clear; close all;

1. Now we need to do is to call the directory we are working in and ask it how many files are there that need processing. The *dir* function is useful for doing that. We can create an flist, or a list of file names in a variable and MATLAB will then read that the list and apply commands to each name in that list.

flist=dir(‘C:\Users\afischer\Documents\MATLAB\chl\\*.nc’)

You need to make sure that you have the appropriate directory path so that MATLAB knows to go to the folder where you are keeping your files. Also note the application of the wildcard (\*). This tells MATLAB to find all files that end with the file extension of .nc.

1. When you issue this command MATLAB creates a *struct* array. Type flist into the command window. Now type flist.name to make sure that the file names you called are showing up in the flist variable. You should see all 14 file names in this list.
2. Let’s now create a variable to tell us how many files we have. Type following command and examine the results.

nfiles=length(flist);

1. We will also need to create a directory path variable so that MATLAB knows what directory to look into. We can also use this variable later to assign a location to save our subscenes.

dirpath=['C:\Users\afischer\Documents\MATLAB\chl\']

1. Ok. Time to start the loop. A loop must start with “for” and end with “end.” Type for and end into your script. You should see the color of your text change. MATLAB knows these words are reserved for loops and colors them accordingly.
2. We now need to tell MATLAB how many times to run the loop. Right next to for type

for N=1:nfiles

1. Before you used the HDF Import tool to import your HDF file. Here we will simply write out the script that the Import Data tool uses to import these files. You can go back to the HDF Import Tool and see the command in the import dialogue box. It is located at the bottom of the dialogue in the “Dataset Import Command” box. However, we will need to replace the specific name with our generic loop file name as follows.

>> filename=[dirpath flist(N).name];

>> chlor\_a=ncread(filename, 'chlor\_a');

>> lon=ncread(filename, 'lon');

>> lat=ncread(filename, 'lat');

1. You can display the file name in the command window as things are processing to make sure that MATLAB is reading the files correctly. Add the following command to your script.

disp(filename);

We can now add the other commands that we created earlier to our script. When you are finished your script should look something like this.

clear all; close all;

flist=dir('C:\Users\afischer\Documents\MATLAB\chl\\*.nc');

nfiles=length(flist);

dirpath=['C:\Users\afischer\Documents\MATLAB\chl\'];

for N=1:nfiles

 filename=[dirpath flist(N).name];

 chlor\_a=ncread(filename, 'chlor\_a');

 lon=ncread(filename, 'lon');

 lat=ncread(filename, 'lat');

 subAX=[145 175 -51 -31];

 clims=[0 0.3];

 xidx=find(lon>=subAX(1) & lon<=subAX(2));

 yidx=find(lat>=subAX(3) & lat<=subAX(4));

 sub\_chl=chlor\_a(xidx,yidx,:);

end

1. Save your script with a good name that will help you remember what it does. You can also provide comments within the script that will help you remember and describe to other users what it does. Simple enter the % sign before the text and MATLAB will know that this is a comment and will not try to read these words as commands. Here is an example

% Subscenes global MODIS chlorophyll images to the area defined by subAX.

1. For later processing we can we can save the variable which contains the sub-scened data and the lonx and laty coordinates that we will use to plot the data.
2. First, to reduce confusion and simplify the things let’s extract the characters from the file name which represent the year. Add the following line to your script.

year=flist(N).name(2:5)

Type this in the command line so that you see what it does.

1. Now we need to create a variable which contains the subset of longitude and latitude values. Add this to your script in the loop section.

sub\_lonx=lon(xidx);

sub\_laty=lat(yidx);

1. This next part is a little tricky. We have to use the eval command so that we can save the variables with the appropriate files names. You can look up eval in help to see what it does. Simply, you should enter the below commands into your script.

year=char(year)

outname=['X', year]

eval(['save ' outname ' sub\_chl']);

1. In the end, you should have a script like this.

% Subscenes global MODIS chlorophyll images to the area defined by subAX.

clear all; close all;

flist=dir('C:\Users\afischer\Documents\MATLAB\chl\\*.nc');

nfiles=length(flist);

dirpath=['C:\Users\afischer\Documents\MATLAB\chl\'];

for N=1:nfiles

 filename=[dirpath flist(N).name];

 chlor\_a=ncread(filename, 'chlor\_a');

 lon=ncread(filename, 'lon');

 lat=ncread(filename, 'lat');

 subAX=[145 175 -51 -31];

 clims=[0 0.3];

 xidx=find(lon>=subAX(1) & lon<=subAX(2));

 yidx=find(lat>=subAX(3) & lat<=subAX(4));

 sub\_chl=chlor\_a(xidx,yidx,:);

 year=flist(N).name(2:5);

 sub\_lonx=lon(xidx);

 sub\_laty=lat(yidx);

 year=char(year);

 outname=['C', year];

 eval(['save ' outname ' sub\_chl ' ' sub\_lonx ' ' sub\_laty']);

end

1. Run your script by pressing the button with the right facing green arrow. Watch the “Current Folder Window” to see that your data is being saved there. You should have 14 .mat files that begin with the letter “C” and contain a number for the year.
2. The last thing we will do it to create a small script to plot the data for all years for the sub-scened area. For this subplot command. Look this up in MATLAB help to understand what it does. We will also be using the load command. Since you previously saved your data out of MATLAB we can use the load command to bring that data back into your work space.
3. Create a new script that load all of your data and plots all of the images in a row from left to right.
4. First start with the clear all; and close all; commands. As before, define your file lists and directory path. As before create a loop, except after you read your file within the loop add the subplot command, for 2 rows and 7 columns of plots. Use your counter N to indicate that the Nth file should be plotted in the Nth window.

subplot(2,7,N)

1. Define your file name and load the data associated with your file name.

filename=[dirpath flist(N).name];

load(filename);

1. Now image your data in the plot area provided.

imagesc(sub\_lonx, sub\_laty, sub\_chl, clims);

1. Make sure that you set your color limits, or clims and adjust your y-axis direction so that your map is drawn right side up.
2. In the end you should have a script that looks like this.

clear all; close all;

flist=dir('C:\Users\afischer\Documents\MATLAB\chl\C\*');

nfiles=length(flist);

dirpath=['C:\Users\afischer\Documents\MATLAB\chl\'];

clims=[0 0.3];

for N=1:nfiles

 filename=[dirpath flist(N).name];

 subplot(2,7,N)

 load(filename);

 imagesc(sub\_lonx, sub\_laty, sub\_chl, clims);

 set(gca, 'YDir', 'normal');

end

***Producing Better Plots***

Now we will work on fixing up the plot to make it look a little better for presentation and begin to learn how to examine the temporal and spatial variability in this time series of data. Some things you might want to do include removing the numbers from the x and y axis and only including them on your last plot, adding title to each of the plots and adding a colorbar.

1. You can modify your script to eliminate the X and Y labels from the axes of your plot. Modify the set command in your plotting script so that the line looks like this. This tells MATLAB to leave the XTick and YTick, or the X and Y labels blank.

set(gca, 'YDir', 'normal', 'XTick', [], 'YTick', []);

1. We will also want to add X and Y labels only in the last plot window (N=14). To do that we will add an *if* statement that executes statements if condition is true. *if* expression, statements, *end* evaluates an expression, and executes a group of statements when the expression is true. An expression is true when its result is nonempty and contains only nonzero elements (logical or real numeric). Otherwise, the expression is false. In this case we will had specific ‘Xlims’ and ‘YLims” only on the last plot or when N=14.

if N==14

 set(gca,'YDir','normal','XTick',[145 160 175],'YTick',[-50 -42 -32]);

 end

1. Now let’s add a title to each of your plots. First we will have to extract the year from the file name. You have already done this in the last script. Add the following lines to your plot.

year=flist(N).name(2:5);

 title(year);

1. Now add a colorbar so that it does not squish the subplot of your figure window. For this we will use handle graphics. Type the command colorbar but set it equal to h and then use the set command to manually set the location of your colorbar. Do this outside of your loop.

h=colorbar

set(h, 'Position', [.02 .11 .015 .8150])

1. Lastly, add a label to identify the appropriate units of the chlorophyll measurements.

ylabel(h,'Mean Chlorophyll (mg m^{-3})');

1. Ultimately, your script should look like this…

clear all; close all;

flist=dir('C:\Users\afischer\Documents\MATLAB\chl\C\*');

nfiles=length(flist);

dirpath=['C:\Users\afischer\Documents\MATLAB\chl\'];

clims=[0 0.45];

for N=1:nfiles

 filename=[dirpath flist(N).name];

 subplot(2,7,N)

 load(filename);

 axis equal

 imagesc(sub\_lonx, sub\_laty, flipud(rot90(sub\_chl)), clims);

 set(gca, 'YDir', 'normal', 'XTick', [], 'YTick', []);

 if N==14

 set(gca,'YDir','normal','XTick',[145 160 175],'YTick',[-50 -42 -32]);

 end

 year=flist(N).name(2:5);

 title(year);

end

h=colorbar;

set(h, 'Position', [.06 .11 .015 .8150])

ylabel(h,'Mean Chlorophyll (mg m^{-3})');

**Challenge Task - Remote Sensing Sea Surface Temperature**

Now it is your turn to process imagery on sea surface temperature for the same area using a data set from the same sensor, MODIS Aqua.

http://oceandata.sci.gsfc.nasa.gov/MODIS-Aqua/Mapped/Annual/4km/sst4

Your task is to extract and plot annual sea surface temperature following the methods above.

**You final plots of annual chlorophyll values and annual SST values will be included in your first assessment.**

1. References

Attaway, S., 2013. MATLAB: a practical introduction to programming and problem solving. Butterworth-Heinemann.

Martin, S., 2014. An introduction to ocean remote sensing. Cambridge University Press.