Given Lesson 1's introduction, the program participant should now be able to read in another NetCDF file and plot the results. Complexities may arise owing to dataset format differences. These are addressed below for a specific dataset file.

Extra Practice: Application to sea surface temperature data

Note: the following is not necessary but is helpful for obtaining the COESSING certificate at the end of the week! as it demonstrates to the instructors that you understand the previous material.

Try the following data file. It is sea surface temperature (SST) obtained from RSS:

http://data.remss.com/SST/daily/mw_ir/v05.0/netcdf/2020/20200717120000-REMSS-L4_GHRSST-SSTfnd-MW_IR_OI-GLOB-v02.0-fv05.0.nc (http://data.remss.com/SST/daily/mw_ir/v05.0/netcdf/2020/20200717120000-REMSS-L4_GHRSST-SSTfnd-MW_IR_OI-GLOB-v02.0-fv05.0.nc)

1. Download the file. Place in a location such as `data/sea_surface_temperature`
2. Display the metadata of the file using `xarray`
3. Read in the netCDF file
4. As we did for SSS, try creating a map of SST as a function of latitude (y-axis) and longitude (x-axis) in units of degrees C (you will need to convert from Kelvin to Celcius)
5. **Zoom into the Gulf of Guinea**
6. Answer the following two questions:
   - What similarities are present in the SST data when compared to the SSS data?
   - What differences are present in the SST data when compared to the SSS data?

Since this SST data is valid on the same date as the SSS datafile that we just examined (valid on July 17, 2020), there should be some features common to both.

What you should show to the instructor: You should show one of the following: (1) metadata associated with the above netcdf file and (2) SST as a function of latitude and longitude in degrees C over the Gulf of Guinea region

A few helpful points to get you started

1. The relevant "sst" variable is `analyzed_sst`
2. Time is given to us in units of seconds since 1981/01/00 00:00
3. `analyzed_sst` is not in dimensions of [lat,lon] as expected but in [time,lat,lon]. But there is only one time. So, one way to read in the important data is as follows: `analysed_sst = nc.variables["analysed_sst"][0,:,:]`
4. `analyzed_sst` is temperature in units of Kelvin so you need to subtract 273.15 to convert to degrees Celcius
5. `analyzed_sst` is given to us as -180 degrees to 180 degrees in longitude. Therefore it is simpler to index the data than it was for sea surface salinity. We provide code below to do this.

Code to read in the sea surface temperature for a single time step

In [ ]:
```
# Read in the data from the netcdf file.
nc = Dataset(infile, "r")
etime = nc.variables["time"][:] # time in seconds since 1981/01/00 00:00
lat = nc.variables["lat"][:] # latitude (degrees), values = [-90, 90]
lon = nc.variables["lon"][:] # longitude (degrees), values = [-180, 180]
analysed_sst = nc.variables["analysed_sst"][0,:,:] # sea_surface_temperature, K
elvin
```

Code to subset the sea surface temperature (SST) dataset near the Gulf of Guinea

In [33]:
```
# Subset for the region of interest.
latlim = np.array([-10.0,10.0]) # in degrees
lonlim = np.array([-20.0,15.0]) # in degrees
latlim = np.double(latlim)
lonlim = np.double(lonlim)
ilat1 = (lat >= latlim[0]) & (lat <= latlim[1])
ilon1 = (lon >= lonlim[0]) & (lon < lonlim[1]);
ilat = ilat1;
ilon = ilon1;

lats = lat[ilat1]
lons = lon[ilon1]

index1 = np.array(np.where(ilat))
index2 = np.array(np.where(ilon))
#print(index1)
#print(index2)
sst_block1 = sst[ilat,:]
sst_block1 = sst_block1[:,ilon]

nlats = len(lats)
nlons = len(lons)
sst_block = np.zeros([nlats,nlons])
sst_block[0:nlats,0:nlons] = sst_block1
```

Code to form a mask for the land.

In [34]:
```
# Form a mask for the land.
# This mask uses the bad values to identify land.
mask = np.zeros([nlats,nlons])
igood = (sst_block >= -3) # find good values
mask[igood] = 1
inan = (sst_block < -3) # find bad values
mask[inan] = np.nan # not a number
```

Code to plot the SST dataset near the Gulf of Guinea

In [ ]:
```
# Plot the sea surface temperature.
plt.pcolor(lons,lats,sst_block*mask,cmap="coolwarm") # the colormap changes to
red/blue
plt.xlabel('Longitude (deg)')
plt.ylabel('Latitude (deg)')
plt.title('Sea Surface Temperature: '+fname) # here we need to insert a date in
side the brackets
plt.grid()
plt.colorbar()
#plt.show() # to save the file, we must comment out this line for some reason
(ask Dr. Paige)
outfile = "SST_map.png" # define output filename
plt.savefig(outfile,format='png',dpi=200)
```

In [ ]:

```

```