# GEOS397_HW4

## Table of Contents



Josh Enterkine and Javier Colton

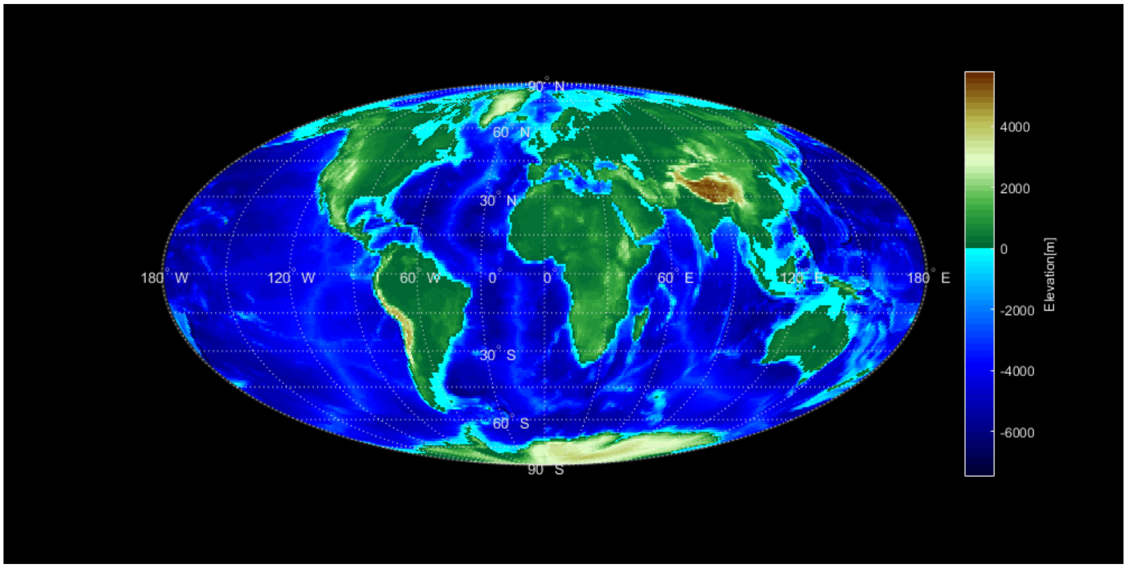
# Part 1: Make a basemap and determine a crude ocean volume

```
clear
close all
clc

global lat;
global lon;
global LAT;
global LON;
global topo;
global ax;
global oceanVol1m;
global h;

% Step 1: Load the topo.mat data set

load( 'topo.mat' ); % loading the MATLAB global topography data into
 memory

% Step 2: Plot a basemap

% setup the figure properties that we want
h = figure; % creating new figure for basemap

hold on;

h.InvertHardcopy = 'off'; % sets the colors of saved figure to be
 those of display
h.Color = 'k'; % sets background of map figure to black
h.Position = [100 100 1000 500]; % sets position of figure window on
 opening
h.PaperPositionMode = 'auto'; % sets size of saved figure to be same
 as display

% setup the map axes
ax = axesm('Mollweid', 'Frame', 'on', 'Grid', 'on'); % setting
 properties of map, display options
```

```
setm(ax,'MLabelLocation',60); % setting interval for labelling
 longitudes
setm(ax,'PLabelLocation',30); % setting interval for labelling
 latitudes
mlabel('MLabelParallel',0); % setting axis to label longitudes (i.e.
 0-degrees latitude)
plabel('PLabelMeridian',-25); % setting axis to label latitudes (i.e.
 -25-degrees longitude)
axis('off'); % sets background of axes to null (figure background
 shows instead)
setm(ax,'FontColor',[0.9 0.9 0.9]); % sets font color to better color
 (white or close to)
setm(ax,'GColor',[0.9 0.9 0.9]); % sets font color to better color
 (white or close to)

LAT = topolatlim(1):topolatlim(2); % setting values for latitude
LON =  topolonlim(1):topolonlim(2); % setting values for longitude

idx180 = find(LON > 180, 1); % find first index of longitude array
 that is larger than 180 deg
LON(idx180:end) = LON(idx180:end) - 360; % subtract 360 degrees to
 make lon interval [-180 180] instead of [0 360]

[lon, lat] = meshgrid(LON,LAT); % compute the lat/lon of every grid
 point in topo
pcolorm(lat,lon,topo); % plot the matrix of elevations on the map
demcmap(topo); % give it a better colormap

c = colorbar; % creating colorbar and handle
c.Label.String = 'Elevation[m]'; % labeling colorbar
c.Color = [0.9 0.9 0.9]; % setting color of colorbar to match


% Part 3: Modify the topo matrix to represent ocean depth
topoOcean = topo; % creating new to preserve topo for Part 3
distdim(topoOcean, 'm', 'km'); % Converting units
topoOcean=ans; %applying this to topoOcean
topoOcean(topoOcean>0)=0; %Filtering out any values above sealevel and
 setting them equal to zero


% Answer: we could compute the volume of the ocean knowing the depth
 by
% multiplying the depth by the area covered by each pixel. Doing so
 above would calculate only the volume of negative height
% values (e.g. the oceans, Death Valley, inland bodies of water with
 depths
% below sea-level).


% Step 4: Compute the area of each pixel


EARTH_RADIUS_KM = 6371; % Earths Radies
```

```
earthCircumfrence = 2*pi*EARTH_RADIUS_KM % Finding the earth's
 circumfrence
pixelWidth = earthCircumfrence/360 %Find the pixel width
pixelHeight = (earthCircumfrence/2)/180 %Find the pixel height
pixelArea = pixelWidth * pixelHeight %Find the pixela area by
 multiplying the width and height

topoOceanVolume = pixelArea * topoOcean; %Finding the volume for each
 cell
negativeTotalVolume = sum(sum(topoOceanVolume)); %The total sum of all
 the cells resulting in negative volume
oceanVolumeTotalPart1 = abs(negativeTotalVolume) %correcting for the
 negative volume.


% Answer: Using this area, the total volume of the oceans on Earth
 (really
% the volume below sealevel) is approximately 1.8226e+09 cubic
 kilometers.
```

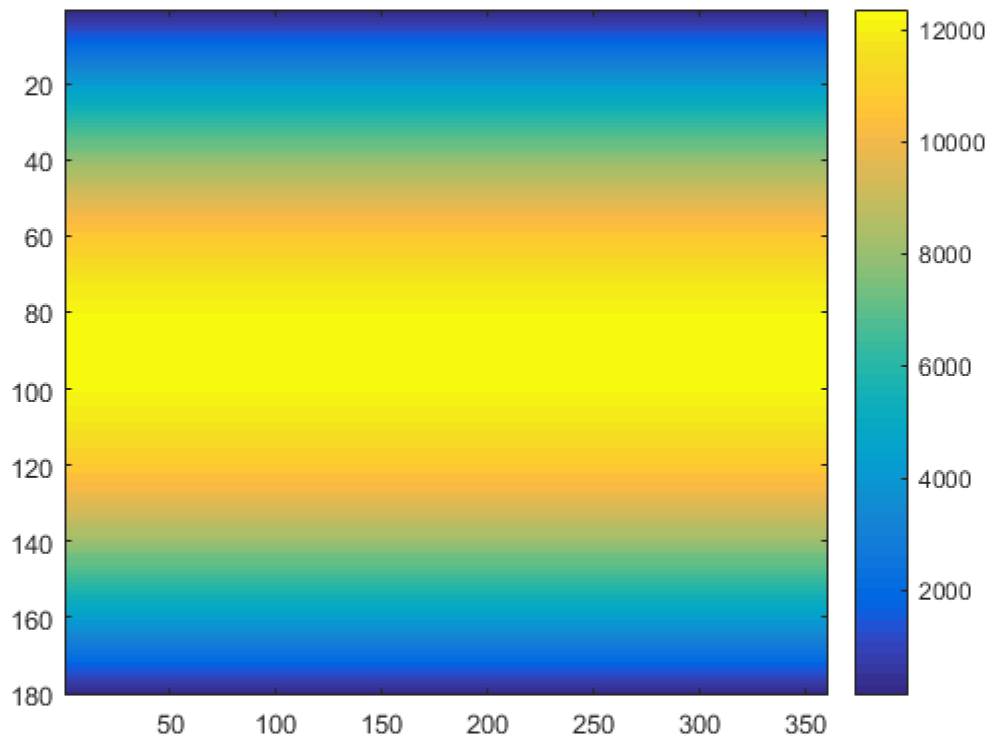
```
earthCircumfrence =

   4.0030e+04


pixelWidth =

  111.1949


pixelHeight =

  111.1949


pixelArea =

   1.2364e+04


oceanVolumeTotalPart1 =

   1.8226e+09
```




# Part 2: A more accurate volume estimate

```
% Answer: This means that our previous estimate of ocean volume treats
 the
% area of a pixel (square degree) as a perfect square, and
 consequently,
% the volume beneath it as a perfect box when in reality the x*y
 profile
% diminishes in size as the depth increases. This means that we would
 be
% overestimating ocean volume.

% Step 1: Compute the area between two lines of latitude

% Answer: The area represented by two lines of latitude one degree
 apart
% with this model would be 1/360 of the area of each ring(A(ring)
 equation).
% This gives an answer of approzimately 1.076 x 10^4 square km.
 Calculation
% below.
areaRing =
 (2*pi*(EARTH_RADIUS_KM*EARTH_RADIUS_KM))*abs(sin((30*pi)/180) -
 sin((29*pi)/180)); % area of ring
pixelArea2 = areaRing/360; % area of one 'cell'

% Step 2: Compute the area of each pixel

% declaring variables and instantiating requisite formulas, etc.
latRad = (LAT*pi)/180; % converting Latitude degrees to radians
numPixelValRow = numel(LAT) - 1; % one row per horizontal area band
numPixelValCol = numel(LON) - 1; % one column per vertical area band
pixelValues = zeros(numPixelValRow , numPixelValCol); % empty table to
 store calculated pixel values
```

```
for jj=1:(numel(LAT)-1) % for each "ring"

    nLatRad1 = latRad(jj); % latitude value at index 1
    nLatRad2 = latRad(jj+1); % latitude value at index 2
    areaRing =
 (2*pi*(EARTH_RADIUS_KM*EARTH_RADIUS_KM))*abs(sin(nLatRad1) -
 sin(nLatRad2)); % area of first ring

    pixelArea3 = areaRing/360; % area of each latitude band within
 ring (i.e. pixel) will have 1/360th of the area
    pixelValues(jj,:) = pixelArea3; % assigning this value to each
 cell in row

end

% Checking graphically that pixelArea3 worked, per instructions

 p = figure; % new figure for plot
 p = imagesc(pixelValues); % using imagesc since scaled values
 colorbar % turning on colorbar for figure
```



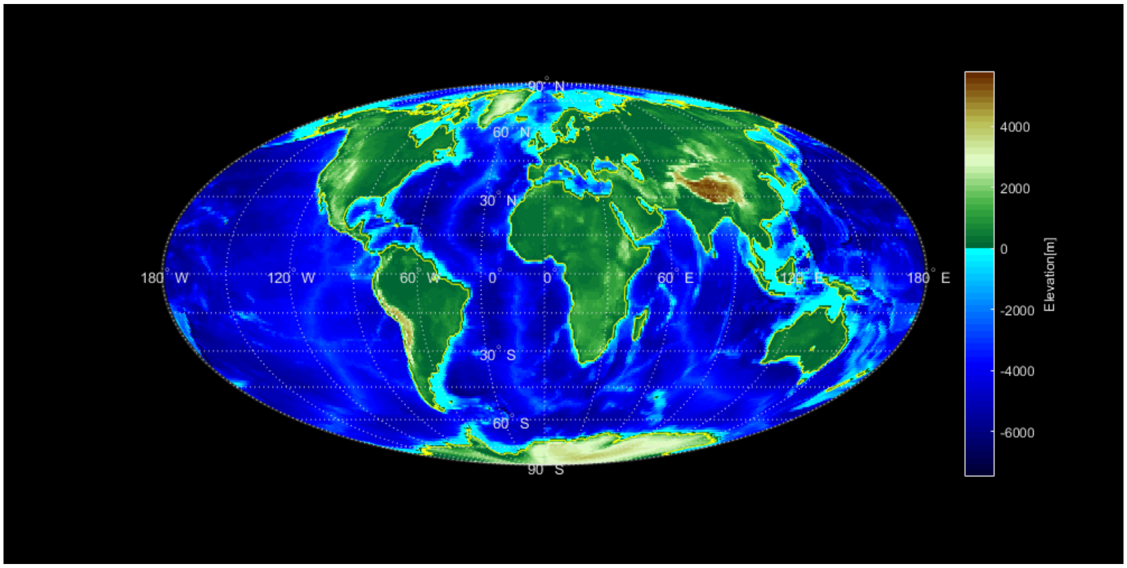

# Part 3 Sea-level rise due to Antarctica

```
% Step 1: Compute the volume of the water contained on Antarctica
```

```
% Answer: The toal volume of mass on Antarctica, assuming all
% land below 60S latitude is Antarctica, and all topographic values
 above 0
% are ice, is 9.5883e+06 km^2. See below for calculation.

% instantiating variables, etc.
topoLand = topo; % creating new to preserve topo
antarcticIceTotal = 0;

% removing all negative values from topo
for kk=1:numel(topoLand) % runs through each value in topoLand

    if topoLand(kk) < 0 % conditional for topo change per
 specifications
        topoLand(kk)= 0; % reassign negative elevations to 0
    end

end

% setting topo values to 0 for everywhere North of 60S (assuming
 Antarctica is all land below 60S latitude)
topoAntarctica = topoLand; % new table for Antarctica
for ll=1:149 % removing all values N of 60S

    topoAntarctica(ll,:) = 0;

end

topoAntarctica = topoAntarctica./1000; % converting altitudes from m
 to km
antarcticIce = topoAntarctica .* pixelValues; % multiplying area by
 height for each cell using pixelArea3 approach (best estimate)

% summing values in antarcicIce, but there must be a better way than a
 loop
for mm=1:numel(antarcticIce);

    antarcticIceTotal = antarcticIceTotal + antarcticIce(mm);

end

% Answer: The toal volume of water held in ice on Antarctica, assuming
 all
% land below 60S latitude is Antarctica, and all topographic values
 above 0
% are ice, is 8.6295e+06 km^2. See below for calculation.
antarcticWater = antarcticIceTotal * 0.9;


% Step 2: Compute the change in total ocean volume for incremental
 changes
% in sea-level height

% calculating starting ocean volume using new area metric
```

```
oceanVolValues2 = topoOcean .* pixelValues; % volume of ocean at each
 square degree
oceanVolPart3 = 0; % initializing total vol

for yy=1:numel(oceanVolValues2)

    oceanVolPart3 = oceanVolPart3 + abs(oceanVolValues2(yy));
end


topoOceanRise = topo; % new topo to preserve data (in km)
topoOceanRise1m = topoOceanRise; % for inital 1m rise, per
 instructions
meltwaterVector = zeros(1,100); % per instructions



% calculating ocean rise for 1m of meltwater added (by subtracting 1m
 from
% all land topo, per instructions)
for zz=1:numel(topoOceanRise1m)

    if topoOceanRise1m(zz)>0 % if it is land
        topoOceanRise1m(zz) = topo(zz) - 1; % subtract 1m

        if topoOceanRise1m(zz) > 0 % if still above 0m high
            topoOceanRise1m(zz) = 0; % reset to 0
        end
    end
    if topoOceanRise1m(zz) < 0 % if below sea-level
        topoOceanRise1m(zz) = 1/1000; % add 1m
    end

end

oceanVol1m = (topoOceanRise1m .* pixelValues); % depth times area =
 volume
oceanVol1mTotal = 0; % initializing total

% calculating total for ocean volume plus 1m
for oo=1:numel(oceanVol1m)
   oceanVol1mTotal = oceanVol1mTotal + abs(oceanVol1m(oo));
end

oceanVolAntarcticEquiv = zeros(180, 360);

% calculating volumes from 1m to 100m added sea-level
volTot = 0; % temporary variable for volume
for pp=1:100 % for 100 one-meter decreases in land height

    for qq=1:numel(topoOceanRise) % for each value in topoOceanRise
        if topoOceanRise(qq)>0 % if it is land
            topoOceanRise(qq) = topoOceanRise(qq) - pp; % subtract in
 1m increments
```

```
%
            if topoOceanRise(qq) > 0 % if still above 0m high
                topoOceanRise(qq) = 0; % set to zero
            end
        end

        if topoOceanRise(qq) < 0
           topoOceanRise(qq) = pp;
           volAtPixel = topoOceanRise(qq) .* pixelValues(qq); %
 calculating volume at pixel
           volTot = volTot + abs(volAtPixel); %
           if pp==7
               oceanVolAntarcticEquiv(qq) = abs(topoOceanRise(qq) +
 oceanVolValues2(qq));
           end
        end

    end
    meltwaterVector(pp) = volTot/1000 ;

end


% Step 3: Match the change in volume with the volume held on
 Antarctica

% A sea-level rise of 7m to 8m is approximately equal to the sea-level
 rise
% expected if all of the ice on Antarctica melted, according to my
% calculations. For the purposes of a single answer, I would say 7m.




% Step 4: Plot the new coastline

figure(h)
LATnew = LAT(1:180);
LONnew = LON(1:360);
V = [7 7];
contourm(LATnew,LONnew,oceanVolAntarcticEquiv, V, 'y')

% Answer: Parts of Florida, Cape Cod
%
% Answer: This modelling approach could be improved if we chose to use
 a
% finer spatial resolution (instead of square degrees)
% We made the asumption that everything over sea level is ice. This is
 not
% true!
% We did not account for the fact that the earth isn't a perfect
 sphere.
```




# Part 4

Answer: We would add in the total water volume of Greenland's ice into line 241. This would add together the antarctic water volume and the greenland water volume before we started the loop for sea level rise.

```
pixelValuesGreenland=pixelValues(:,110:160); %Selecting the columns
 for Greenland.
pixelValuesGreenland=pixelValuesGreenland(6:30,:); %Selecting the rows
 for Greenland.

gtopo = topo(:,110:160); %Selecting the columns for Greenland so that
 it matches pixelValues.
gtopo2 = gtopo(6:30,:); %Selecting rows for Greenland
gtopo2(gtopo2<0)=0; %Filtering out any data points below sea level.

greenlandIceVolume = (gtopo2/1000).*pixelValuesGreenland; %Calculating
 the volume for Greenland's Ice in each cell
greenlandWaterVolume = greenlandIceVolume * 0.9; %Turing that volume
 to water

totalGreenlandWaterVolume = sum(sum(greenlandWaterVolume)) %Summing
 all of the cells to get the total volume.

% The total volume of water contained in greenland's ice would be
% 5.6121e+06 cubic kilometers.
```

```
totalGreenlandWaterVolume =

   5.6121e+06
```

*Published with MATLAB® R2016a*