ENG06 PROJECT #2

COLLABORATION POLICY:

You need to form a team of two to complete the project. You are only allowed to talk and collaborate with members within your team. Team members are expected to equally participate, and collaboratively work towards the completion of the project. ***Other than contacting the teaching assistants for clarification, you may not seek the assistance of other persons.***

1.1 LOGISTICS:

1. No later than noon, on August 24 each team must declare the members of the team. ONE electronic message should be sent to Stanley with the subject line: **Engineering 6 Project Team**. The body of the message should contain i) three suggested team names (from which one will be assigned to you), and ii) the names of the team members.
2. By 12 noon on Wednesday, September 7 the projects are due (no late submissions will be accepted).
3. During the lab period on September 7, each team will be scheduled to demonstrate their effort. The times will be assigned no later than end of September 5.

1.2 GRADING CRITERIA:

The projects are open ended. As long as your program can perform the assigned tasks, there will be no correct or wrong approaches. Certainly there will be more acceptable and attractive solutions, and that will be judged in comparison with competing solutions.

The deliverables are: i) a report, ii) an electronic submission of all programs and supporting files (zipped into a folder with the name of your team), and iii) a demonstration.

Bottom line will be, if we cannot get your program to execute, it will not be graded. In all cases it will be essential that you submit a complete set of files to test your program. It will also be important to give clear instructions of how to run your program. This could be done in various ways. One good way to document how your program executes is to give an example case and the results.

The breakdown of grading among project will vary. The grades to each team member will be adjusted according to how the project tasks were delegated and who was responsible for what aspects of the project. How this will be determined is the following. In addition to interviews and information gained during the demonstration, each project will allocate at least 10% of the grade to a section that must be included as an Appendix A. Appendix A must contain:

- A table with the breakdown of the tasks to complete the project, and who was responsible for what part of the project. The intent here is to determine who did what to implement the project. While it is perfectly reasonable that some tasks can be completed jointly, it is

unrealistic to claim that that everyone work together *on all aspects* of the project equally. If all tasks are attributed equally, it will be viewed as unrealistic and come under specific scrutiny.

- The expectation is that each team member must take responsibility for a specific aspect of the project and being able to explain what their contributions to the project have been. This does not mean that person is solely responsible for working on it – it means that the person takes the lead on its execution.
- In Appendix A each member must provide a brief personal summary of what the person's involvement and contributions in the project. Before the project is submitted, the summaries must be provided to all members for review and comment. Appendix A must conclude with the following statement:

  "All team members have read the task summaries contained in this report and have been given an opportunity to comment. "

1.3 SUBMISSION REQUIREMENTS:

Each project submission must have a project report that contains any relevant material deemed essential, and it must contain an Appendix A as described previously. The first page of the report must contain the team name and the names of all members. The front page MUST contain the following statement:

> *"The project was completed exclusively by the members of the team. All critical external resources have been cited. We have given credit any pieces of information that are not common knowledge which include another person's idea, opinion, or theory, any facts, statistics, graphs, drawings or programming code."*

For the purposes of this project, any information provided in this problem statement or in course notes and lectures should be consider common knowledge.

If it is discovered that you have borrowed or used material from a source that is not credited, it will be consider plagiarism and the case will be turned over to Student Judicial Affairs.

The report must be saved a PDF document with the name of the team and report (e.g. if the team names is Foxes, then the report name should be Foxes_Report.pdf).

Each team will submit files in one zipped folder (*do not use any other type of file compression technique or format*) to SmartSite. With the exception of the project report that must be a PDF, all other files must be compatible with your Matlab program. Be sure to include .m files, .fig.wav and any images your program needs to run. If your program doesn't run, you won't receive credit!

All material that must be reviewed for the project must be submitted by the project deadline. No other supplement information or submissions will be accepted after the deadline.

1.4 DEMONSTRATION FORMAT

1) Each team will interactively present their solutions to the project by demonstrating how the code is executed.
2) Only programs submitted by the deadline must be used.

3) You have the option of using a laptop provided by your team or use a workstation in the lab. If you choose the latter option, bring your programs on a memory stick.
4) All members must participate in the presentation. How this will be done will be up to you.
5) One person must not dominate the presentation.
6) The assumption is that all members have a reasonable familiarity with the project, even if they have not been the lead person on that specific topic. During presentation any member of the team can be specifically asked to answer a question

2 BACKGROUND FOR PROJECT:

The first concept to clearly understand is how a point on the earth is represented by latitude and a longitude coordinate pair.

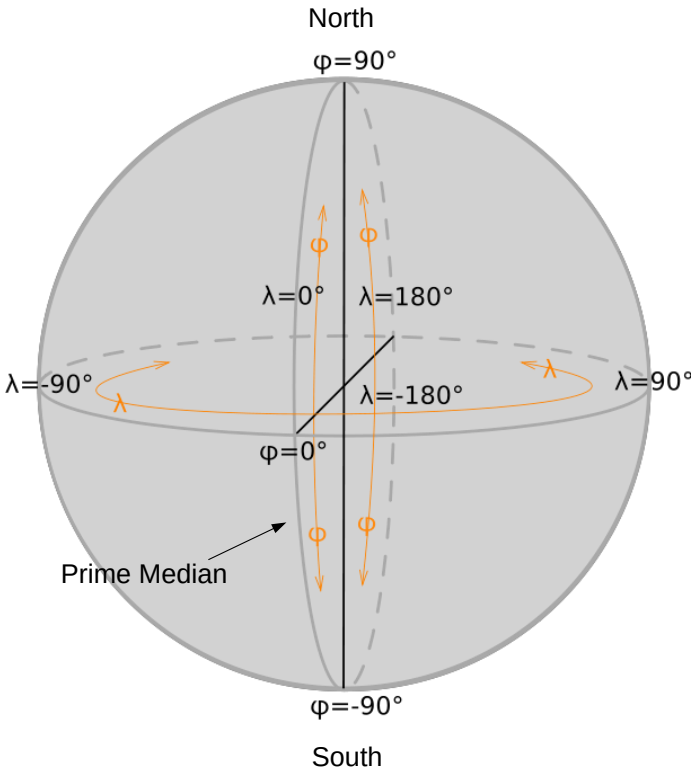
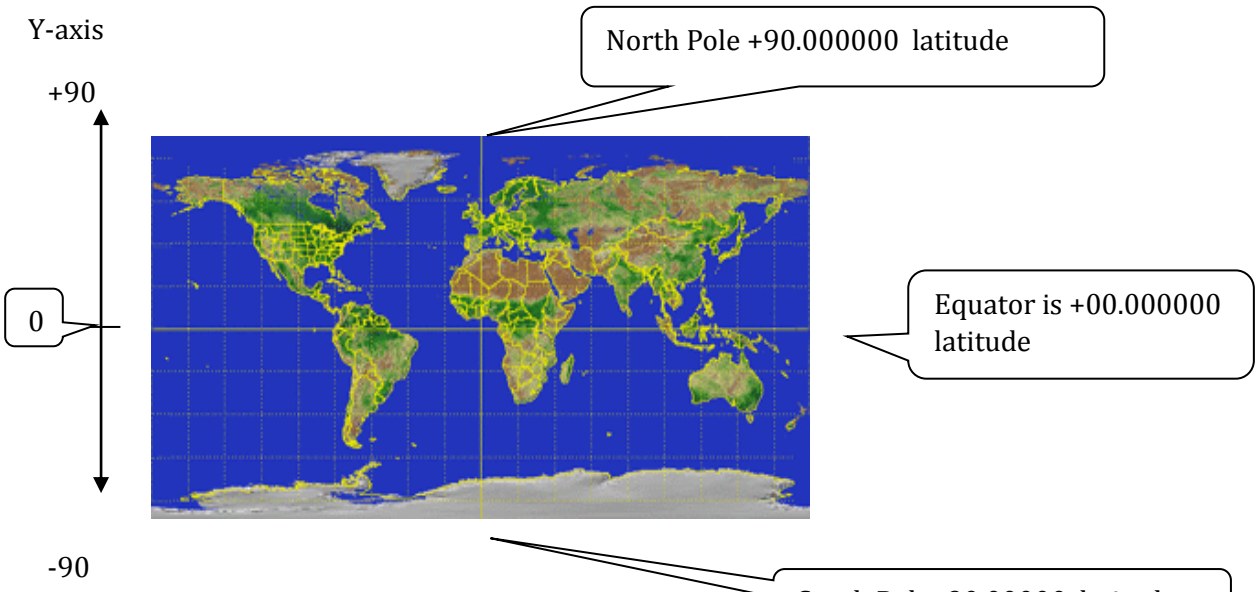


FIGURE 1. THE LATITUDE IS REPRESENTED BY THE ANGLE φ. THE LONGITUDE IS REPRESENTED BY THE ANGLE λ.

WHAT IS LATITUDE?[1] Latitude, or the Y coordinate, is the distance from the equator. Latitude lines run east and west (horizontal) along the surface of the earth. Every point above the equator is in the "+" (positive) range and anything below the equator is in the "-" (negative) range. Coordinate values are preceded by a + or - sign. Think of the standard X/Y chart and think of latitude as being a point along the vertical Y axis. Every area in the United States will have a positive (+) latitude or Y coordinate. Any point on the equator has a latitude of zero while the north

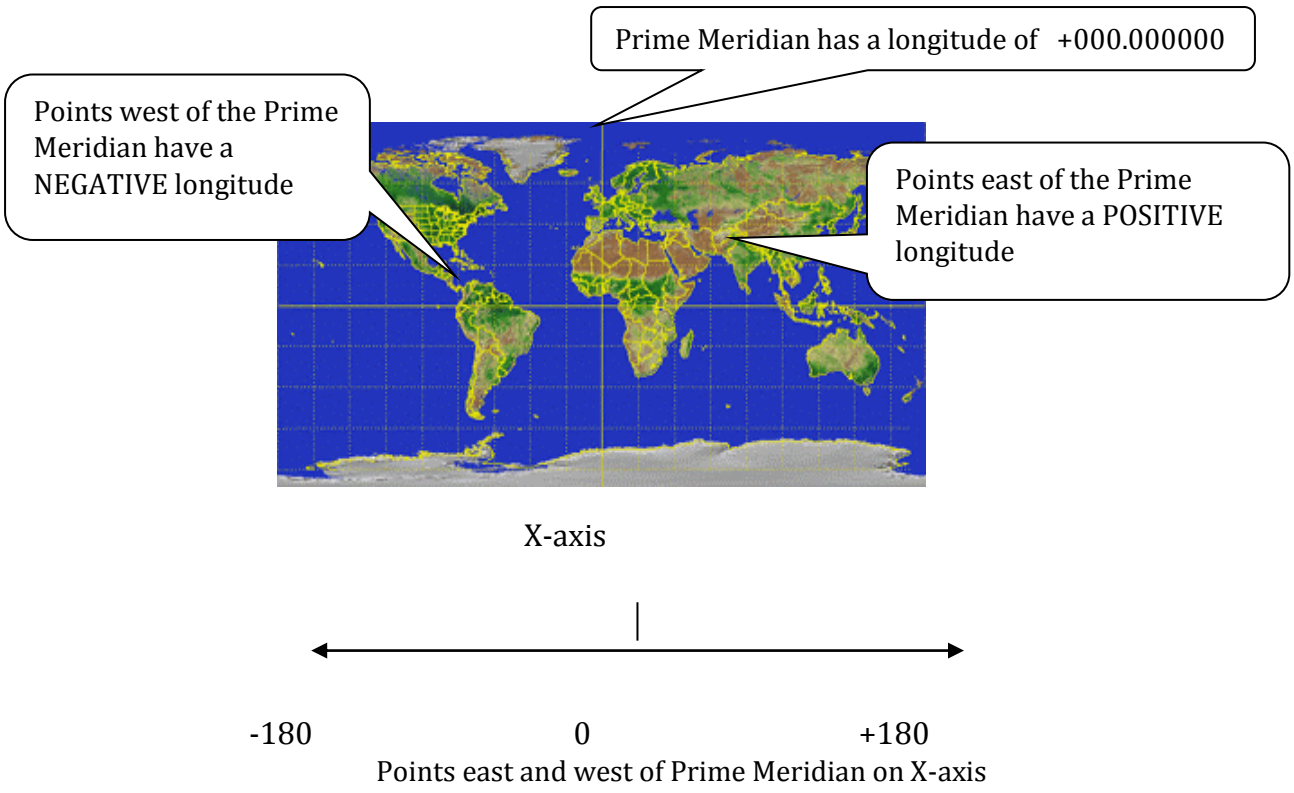
[1] Adapted from *www.al911.org/wireless/XYTUTORP.DOC*

pole is +90 and the south pole is -90. Each degree of latitude is divided into 60 equal parts called minutes and each minute can be further divided into 60 seconds. On the surface of the earth, one degree of latitude is about 69 miles (110 kilometers). Because the earth is not quite a perfect sphere, the distances get slightly greater toward the poles where there is a slight flattening of the earth.



Points above and below equator on Y-axis

WHAT IS LONGITUDE? Longitude, or the X coordinate, is the distance from the Prime Meridian which is in Greenwich, England a borough of London. The earth is divided into two parts, or hemispheres, of east and west longitude. Think of the earth as a globe that is divided into 360 equal slices (180 west and 180 east of Greenwich). The lines between the slices on the globe are called meridians, thus the term "prime meridian" for the meridian that is the starting point, represented by +000.000000 longitude. All points along the prime meridian have a longitude of +000.000000. Longitude lines run north and south along the surface of the earth. Anything east of the Prime Meridian is in the "+" (positive) range and anything west of the Prime Meridian is in the "-" (negative) range. Think of the standard X/Y chart and the longitude being a point on the horizontal X axis. All areas within the United States will have a negative (-) longitude or X coordinate. The space between two meridans is greatest at the equator, about 69 miles (111 kilometers). This space narrows as the meridians approach the north and south poles, so the distance between meridians is not constant. For example, a degree of longitude at New Orleans, LA is about 60 miles (97 kilometers) while at Winnipeg, Canada, a degree of longitude is less than 45 miles (72 kilometers).



2.1.1 MAP PROJECTIONS:

Mapping longitudinal and latitude coordinates onto a plane causes a distortion. This occurs for the same underlying reason that the time it takes to walk "around the world" near a pole is significantly shorter that if the person decides to walk at the equator. For example, a Cartesian plot of a landmass based on longitudinal and latitude distorts the area. If an area is plotted close to a pole, the area will look significantly larger than if the area was plotted at the equator (for the same reason,). For this reason, map projections are used to display longitudinal and latitude information (read the Wikipedia article on map projections). There are a large variety of projections, because it depends on what information the user wants to display. In this project you will use the equirectangular and the Mercator projections.

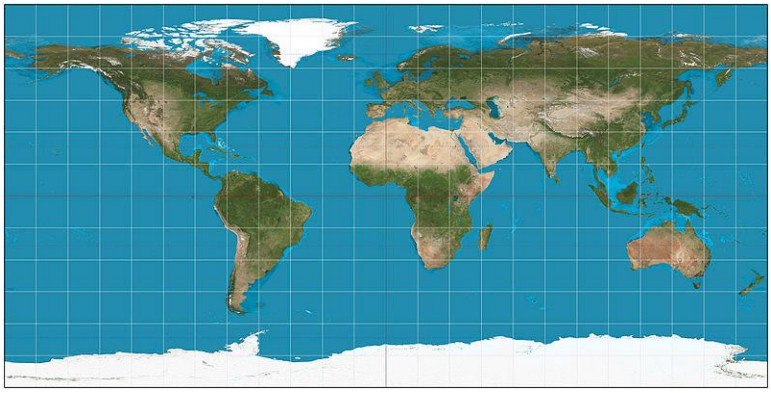


FIGURE 2. EQUIRECTANGULAR PROJECTION OF THE WORLD. SOURCE.

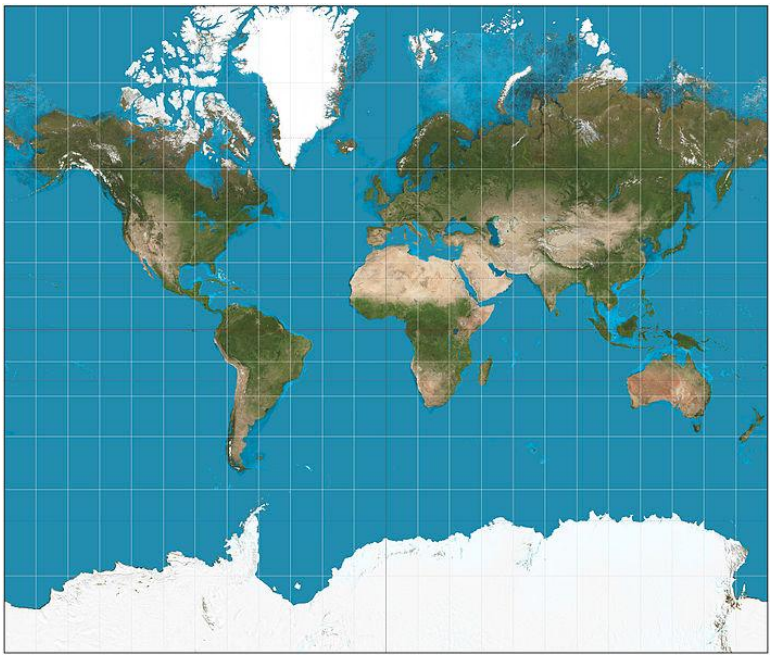


FIGURE 3. MERCATOR PROJECTION OF THE WORLD BETWEEN 82°S AND 82°N. SOURCE .

2.1.2 MAP INFORMATION AS AN ARRAY OF STRUCTURES:

The information of the states will be provided in an array of structures. Load this structure in with

```
>> load states
```

The format of the array of structures is the following:

Field Name	Data Type	Description	Comments
Geometry	String	One of the following shape types: 'Point', 'MultiPoint', 'Line', or 'Polygon'.For a 'PolyLine', the value of the Geometry field is simply 'Line'.	For a 'PolyLine', the value of the Geometry field is simply 'Line'.
BoundingBox	2-by-2 numerical array	Specifies the minimum and maximum feature coordinate values in each dimension in the following form: $\begin{pmatrix} \min(Long) & \min(Lat) \\ \max(Long) & \max(Lat) \end{pmatrix}$	Omitted for shape type 'Point'.
X, Y, Lon, or Lat	1-by-N array of class double	Coordinate vector.	
Attr	String or scalar number	Attribute name, type, and value.	Optional. There are usually multiple attributes.

The fields of the structure can be seen by typing:

```
>> states
states =

51x1 struct array with fields:
    Geometry
    BoundingBox
    Lon
    Lat
    Name
    LabelLat
    LabelLon
    PopDens2000
```

Typing

```
>> AllGeometry={states(:).Geometry}'
```

will indicate that the first position of the structure always contains 'Polygon'. If one only need to inspect the entries, the easiest is to use the Variable Editor. More often, one want to use the variables in a program. Let us review how to extract entries from the array of structures . For example, focusing on the BoundingBox of the second:

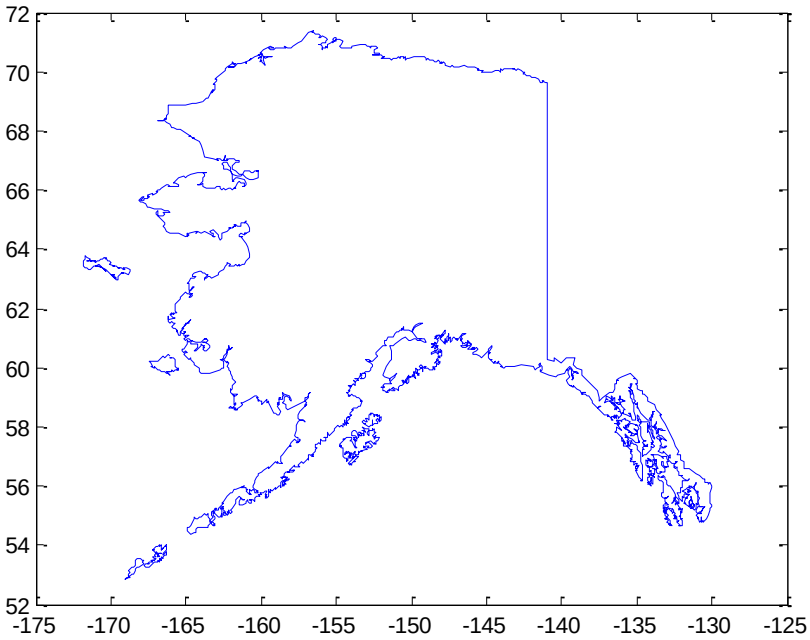
```
>> entry=2; BoundingBoxExtract={states(entry).BoundingBox};
BoundingBoxExtract

BoundingBoxExtract =

    [2x2 double]
```

This shows that Bounding Box is a *cell array*, so the elements are extracted using**:**

```
>> minLong= BoundingBoxExtract{1}(1,1)
maxLong= BoundingBoxExtract{1}(2,1)
minLat= BoundingBoxExtract{1}(1,2)
maxLat= BoundingBoxExtract{1}(2,2)

minLong =
 -171.8500

maxLong =
 -129.9800

minLat =
   52.8200

maxLat =

   71.4000
```

And this indicates that the limits of the longitude and the latitude of the state described in the second record. The next two entries contain the Longitude and Latitude coordinates of a polygon describing the outline of the state. Plot the boundary associated with the second entry:

```
plot(states(2).Lon, states(2).Lat)
```



Inspect the values in states(2).Lon, states(2).Lat and you will note that they contain embedded NaNs to delimiting polygon parts. Which state is this? The Name entry contains the name of the state

```
>> states(2).Name

ans =

Alaska
```

It is useful to extract the indices programmatically:

```
names = {states.Name};
indexHawaii = strmatch('Hawaii',names);
indexAlaska = strmatch('Alaska',names);
```

2.1.3 CALCULATION OF AREA OF A POLYGON ON A SPHERE:

You will need to calculate the area of a polygon on a sphere that is described by a polygon described by a set points of latitude φ and longitude λ. For the purpose of this project, we will assume that earth is a perfect sphere that has a radius of 6367.5 km.[2] Given an arbitrary set of latitude and longitude pairs representing points on a closed polygon, how is the surface area on the sphere enclosed by the polygon calculated? First you need to understand fully how spherical coordinates are used to calculate surface area (take the time to become fully versed in the topic by watching this video - if you do not "get" the underlying concepts, you will become lost in the rest of the project.)

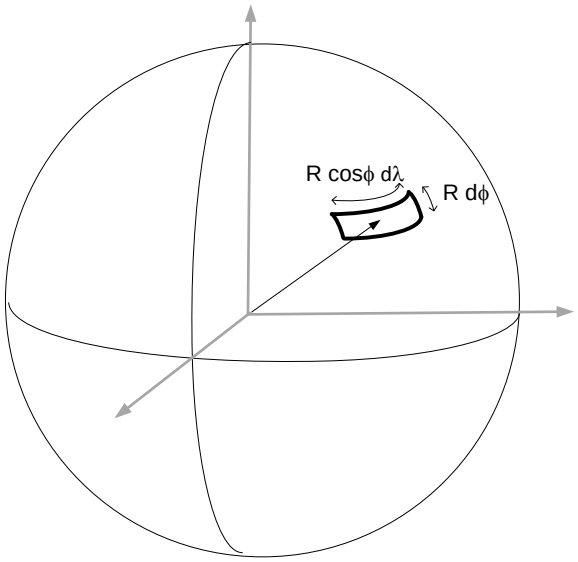


FIGURE 4. SURFACE AREA INTEGRATION BY LATIDUNAL STRIPS.

EQUAL LATITUDINAL APPROACH: This can be done by integrating the contributions from equal latitudinal strips. From this knowledge, it follows that the area of each latitudinal strip will be $(\mathrm{R}\cos(\phi)(\lambda_1\text{-}\lambda_0))(\mathrm{R}d\phi)$, where ϕ is the latitude, λ_1 and λ_0 are the starting and ending longitudes, and all angles are in radians. To obtain the total surface area, all the areas of latitudinal strips are summed.

To see how this works, let us use this approach to calculate the surface area described by one eighth of a sphere, or in terms of latitude and longitude, the polygon described by the vertices $(\phi_0 = 0, \lambda_o = 0)$, $\left(\phi_1 = 0, \lambda_1 = \frac{\pi}{2}\right)$, $(\phi_2 = \frac{\pi}{2}, \lambda_2 = \frac{\pi}{2})$, and $(\phi_3 = 0, \lambda_3 = 0)$. One eighth of a sphere,has a surface area of so the answer should be $4\pi R^2/8 = \pi R^2/2$. Let us first calculate the area by performing the surface area calculation in close form:

$$A = \int_{\phi_0=0}^{\phi_2=\frac{\pi}{2}} \int_{\lambda_o=0}^{\lambda_2=\frac{\pi}{2}} R^2 \cos\phi \; d\phi \, d\lambda = R^2 \left(\frac{\pi}{2} - 0\right)\left(\sin\frac{\pi}{2} - \sin 0\right) = \pi R^2/2.$$

[2] Wolfram Alpha

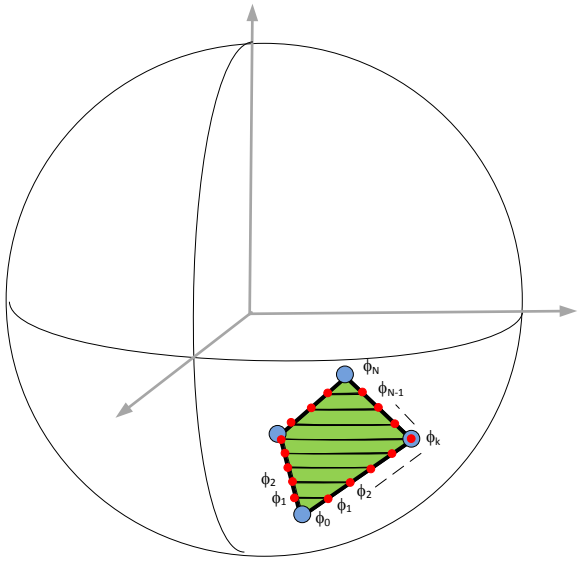
The earth radius is R= 6367.5 km, and so the surface area is:

```
>> pi*R^2/2

ans =  6.3688e+007
```

Now, let us do the integration numerically for $d\phi = 0.0001$, then:

```
>> R=6367.5 ; lambda0=0; lambda1=pi/2; phi0=0; phi1=pi/2;
>> inc=0.00001; del=[ phi0:inc: phi1];
>> cosdel=cos(del)*inc; sum((R^2*( lambda1- lambda0))* cosdel)

ans =  6.3688e+007
```

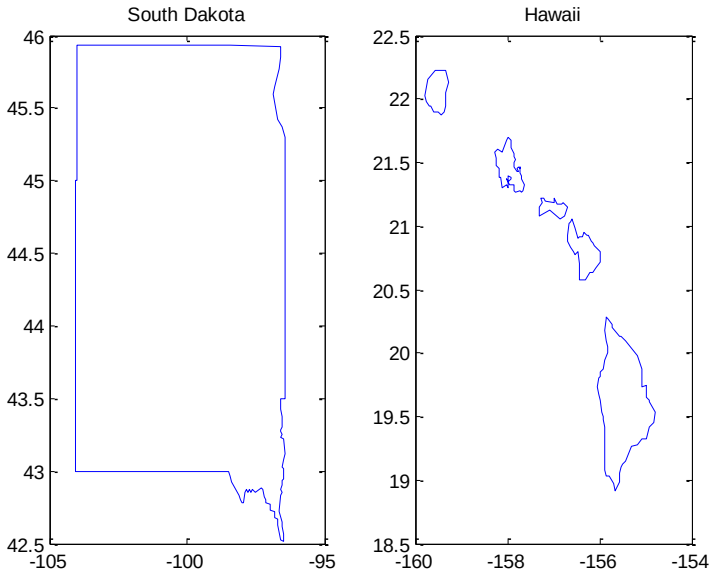
The answer is correct. Let us extend the latitude-strip approach to a general polygonal region. Let us focus on a 4 sided polygon. If two sides of the four sided polygon are described by two latitudes and two longitudes, then using the latitude-strip approach, the area calculation is straightforward extension of the previous idea. If the polygon is more complicated, as the example shown below, then before you can apply the latitude-strip approach, the constant latitude strips need to be generated. From the given four points (the blue dots), a more finely polygon is created[3] described by constant latitude coordinates (represented by the red dots), and then surface area can be calculated.



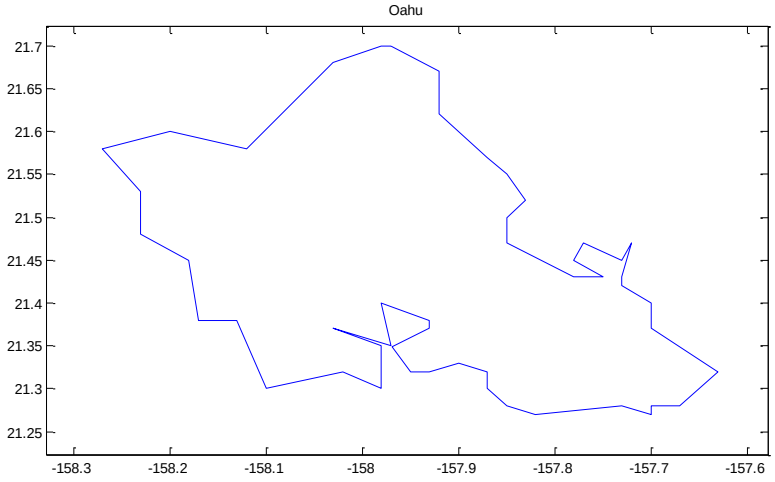
Let us look at possible complications in calculating of the area of a state. Run the code:

[3] Piecewise linear interpolation could play a role here.

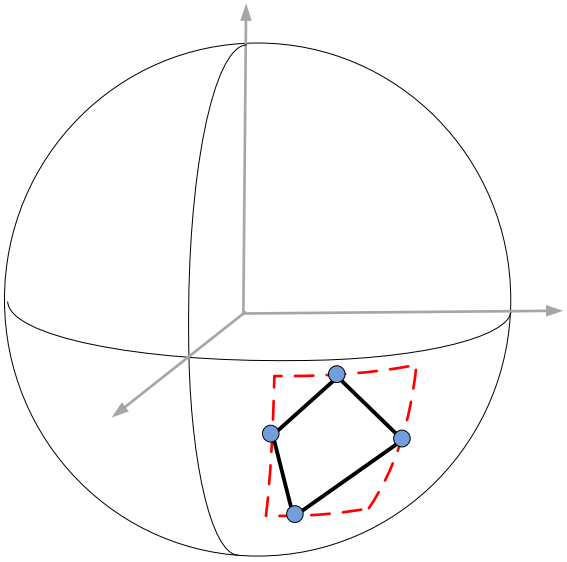
```
>> load states;  subplot(1,2,2);
>> num=11; plot(states(num).Lon, states(num).Lat);title('Hawaii');
>> subplot(1,2,1);
>> num=41; plot(states(num).Lon, states(num).Lat);title('South Dakota');
```



If the state has a simple outline, such as South Dakota, calculating the area will be accurate even if the finer detail in the south west corner is not accurately captured. South Dakota is also is a continuous region. In contrast, Hawaii, is more challenging. Hawaii consists of separate regions, so this needs to be handled. The small details of each of island will need to be captured. If the latitude-strip approach is used, one will need to handle an additional complication seen in the island of Oahu. At the south side, the outline folds back onto itself. Handling such complications, using the latitude-strip, becomes non trivial.



MONTE CARLO APPROACH: An alternate approach to calculate the surface area is to use the Monte Carlo surface integration approach that was discussed in class. In this approach the polygon is placed within a reference region that has a known surface area. Within this reference region, a large number of points are randomly placed, and from the number of points that fall within the polygon the area can be calculated. Matlab has built-in routine **inpolygon** to determine if a point falls with a polygonal region. For a polygonal region described by latitude and longitude, a reference region could be a bounding box polygon described by two latitudes and two longitudes (similar to the bounding box in the mapping structure described in the previous section), for which the surface area can be easily calculated using the latitude-strip. The general idea is shown below.



CONSTANT AREA APPROACH: In the next section you will learn how to perform a surface integral over a two dimensional function. The ideas in that section could be applied to devise a technique to calculate the surface area from a large number of small patches ϕ_{inc}=λ_{inc}= δ. See the discussion at the end of the Surface Integral section for more details.

2.1.4 SURFACE INTEGRALS

In this project, you will write a program to calculate the solar power incident in a specific month onto a selected area on the earth. This requires calculating the surface integral

$$P = \iint S(\phi,\lambda)\, dA = \iint S(\phi,\lambda)\, dA = R^2 \int_{\phi_0}^{\phi_1} \int_{\lambda_o}^{\lambda_1} S(\phi,\lambda) \cos\phi \; d\phi \; d\lambda$$

where $S(\phi,\lambda)$ is the solar radiation in Watts/m2 onto a region. The information on $S(\phi,\lambda)$ is provided in a two dimensional table $S_{\phi_k \lambda_l}$ where the ϕ_k points and λ_l points are in 1 degree increments. Only if the $S(\phi,\lambda)$ does not vary rapidly over the area of integration, then the term $S(\phi,\lambda)$ is a constant and can be pulled outside the integral, and then

$$P = R^2 \int_{\phi_0}^{\phi_1} \int_{\lambda_o}^{\lambda_1} S(\phi,\lambda) \cos\phi \; d\phi \; d\lambda \approx SR^2 \int_{\phi_0}^{\phi_1} \int_{\lambda_o}^{\lambda_1} \cos\phi \; d\phi \; d\lambda$$

Note: The approximation is only valid if $S(\phi,\lambda)$ is a constant with respect to ϕ,λ. Typically, over the area of interest, $S(\phi,\lambda)$ will not be a constant, but if the area is divided in small enough patches $\Delta_{ij} = R^2 \cos\phi_i \, d\phi_i \, d\lambda_j$ such that $S_{\phi_i\lambda_j} = S_{ij}$ is constant, then

$$P = R^2 \int_{\phi_0}^{\phi_1} \int_{\lambda_o}^{\lambda_1} S(\phi,\lambda) \cos\phi \; d\phi \, d\lambda \approx \sum_{j=1}^{N} \sum_{i=1}^{M} S_{ij} \, \Delta_{ij}$$

While the solar radiation is not constant over a surface area, it is expected to vary slowly over typical areas of interest. This can be verified by plotting the solar radiation as a function of longitude and latitude. For example, using the data in the two dimensional table $S_{\phi_k\lambda_l}$ plot the data that fall within the bounding box $24° < \phi < 38°$ and $-108° < \lambda < -92°$ will span Texas.

Let us assume you need to calculate the total power incident onto Texas for a specific month. Looking at the map of Texas below, and considering that the $S_{\phi_k\lambda_l}$ data is provided is on a "coarse" 1 degree grid, how do one deal with the parts of Texas that do not completely fall onto the 1 degree grid? And, how can we get a numerical approximation of the incident power?

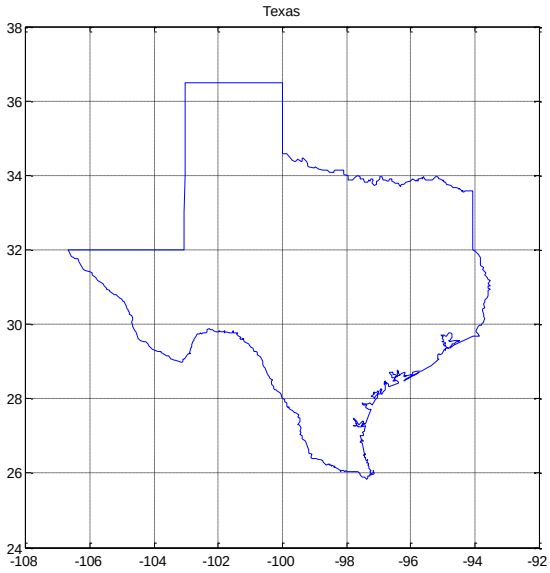
The approach is that we will regrid the data to a grid that is finer that the 1 degree grid. But, first let us generate a two dimensional finely spaced grid and identify only those points in the grid that fall within Texas. We make use of the function **inpolygon** to determine if a given point is inside a given polygon:

> "IN = **inpolygon**(X,Y,xv,yv) returns a matrix IN the same size as X and Y. Each element of IN is assigned the value 1 or 0 depending on whether the point (X(p,q),Y(p,q)) is inside the polygonal region whose vertices are specified by the vectors xv and yv."[4]

First we calculate a spatial array $[X_0]$ and $[Y_0]$ that spans the bounding box $24° < \phi < 38°$ and $-108° < \lambda < -92°$ with increments much smaller than 1 degree. Let say ϕ_{inc}=λ_{inc}= δ degrees. Then, using the polygon that defines Texas, we ask which points in $[X_0]$ and $[Y_0]$ fall inside Texas, and we set those points to 1 to create a two dimensional array $[A_1]$. The next step is to make a correction for the curvature of the earth, since a square defined by ϕ_{inc}=λ_{inc}= 0.1 in North Texas will have a different area than a square defined by ϕ_{inc}=λ_{inc}= 0.1 in South Texas. After this correction is figured out, we change the unity entries of $[A_1]$ with the appropriate scaling factor and creating a new matrix $[A_2]$.

Next we turn our attention to the $S_{\phi_k\lambda_l}$ data which is on a too coarse a grid. We use two dimensional interpolation, using for example the function **interp2**, to regrid the $S_{\phi_k\lambda_l}$ data to the same grid as spatial array $[A_1]$ and place the result in a two dimensional array $[S_1]$. Now you set all the points in $[S_1]$ that fall outside of Texas to zero, by using matrix manipulation and the information in $[A_1]$, and create a matrix $[R]$ that contain the entries $S_{ij}\,\Delta_{ij}$ for only the points that fall within Texas. With this result, we simply sum all the entries in $[R]$ to obtain the final result.

[4] Matlab Documentation



How does one use the above approach to calculate the area of a state? Let us, revisit a derivation of the surface area of a sphere. The surface area of a sphere can be calculated by solving the integral

$$A = \int_{\phi_0=-\frac{\pi}{2}}^{\phi_1=\frac{\pi}{2}} \int_{\lambda_o=0}^{\lambda_2=2\pi} R^2 \cos\phi \;\; d\phi \, d\lambda = 2\pi R^2 \left(\sin\frac{\pi}{2} - \sin\left(-\frac{\pi}{2}\right)\right) = 4\pi R^2.$$

The important observation is that it solves a surface integral like

$$P = \iint S(\phi,\lambda)\, dA = \iint S(\phi,\lambda)\, dA = R^2 \int_{\phi_0}^{\phi_1} \int_{\lambda_o}^{\lambda_1} S(\phi,\lambda) \cos\phi \;\; d\phi \, d\lambda$$

with $S(\phi,\lambda) = 1$.

3 PROBLEM STATEMENT:

3.1.1 GENERAL COMMENTS

1. While specific instructions are given about the assumptions that should be made, in the event that additional assumptions are needed, make sure the assumptions are clearly documented in the material that you hand in.
2. Throughout the project, all results must be displayed in metric units.

3.1.2 INPUTS

You are given four files:

1. An array of structures with the name *states*, which is loaded in by typing "load states", that has the same structure as described in the section "Map Information as an Array of Structures".
2. An Excel file StateInfo.xlsx, which contains the following information:

State	Land Area miles sq	Water Area miles sq	Total miles sq	Capital	Lat Cap	Long Cap	Cost of Solar Electricity cent/kWh
Alabama	50744	1675.01	52419.02	Montgomery	32.36154	-86.2791	9.19
Alaska	571951.3	91316	663267.3	Juneau	58.30194	-134.42	15.93
Arizona	113634.6	363.73	113998.3	Phoenix	33.44846	-112.074	10.08
Arkansas	52068.17	1110.45	53178.62	Little Rock	34.73601	-92.3311	7.25
:	:	:	:	:	:	:	:
Wyoming	97100.4	713.16	97813.56	Cheyenne	41.14555	-104.802	6.49

3. An Excel file Cities.xlsx, which contains the following information:

State	Lat	Long	City
Alabama	30.63	88.07	MobileAeros
Alabama	30.68	88.25	Mobile
Alabama	31.28	85.72	FortRucker
Alabama	31.32	85.45	Dothan
Alabama	31.87	86.02	Troy
Alabama	32.3	86.4	Montgomery
:	:	:	:
Wyoming	44.55	110.42	Yellowstone
Wyoming	44.77	106.97	Sheridan

4. Insolation is a measure of solar radiation energy received on a given surface area in a given time. Insolation is also known as total or global solar radiation. The average insolation, as a function of latitude and longitude, calculated as the 22-year Monthly & Annual Average of the amount of electromagnetic energy (solar radiation) incident on the surface of the earth, in units of kWh/m^2/day is found here (you are given a text file with this information). The conversion into other units are given in the Table 1.

5. TABLE 1. CONVERSION FACTOR (MULTIPLY TOP ROW BY FACTOR TO OBTAIN SIDE COLUMN).

	W/m^2	$kW{\cdot}h/(m^2{\cdot}day)$	sun hours/day	$kWh/(m^2{\cdot}y)$
W/m^2	1	41.66666	41.66666	0.1140796
$kW{\cdot}h/(m^2{\cdot}day)$	0.024	1	1	0.0027379
sun hours/day	0.024	1	1	0.0027379
$kWh/(m^2{\cdot}y)$	8.765813	365.2422	365.2422	1

For use in this project, the information in the two dimensional table has been reformatted into a textfile SolarRadioation.txt

CAUTION: You must assume that the format of the array structure *states* , StateInfo.xlsx , Cities.xlsx and SolarRadioation.txt will stay the same, yet the content can change. The array structure in *states* and StateInfo.xlsx contain information for each state in the United States and the District of Columbia. While one reasonably could assume the number of states will not change, your program should work even if the number of states changes. Also, the information contained in the fields can be modified by user at any time. The number of entries in the file Cities.xlsx can definitively change, as more cities are added. This means that the manner in how you obtain, extract and process the information from these sources must be independent of the number and content of the entries, but should be format specific.

3.1.3 TASK #1

From the information in the *states*, calculate the land area of all of the states and compare your calculations with the information contained StateInfo.xlsx.

1. First you should develop at least two approaches to calculate the surface area from a polygon described by longitude and latitude points on a sphere. For example, the first approach could combine the Equal Latitudinal and the Monte Carlo approaches. The second approach could be the Constant Area Approach. Make sure that you test your area calculations against known results first (show these tests in the final report) – and then apply it to state geometries. In your report describe your test examples, and then use Alaska and California as an example of representative output.
2. Comment on the numerical area calculations for each state in the United States and the District of Columbia. Display your result graphically in a manner that is clear. From the graphical representation it should be clear if your area calculations, using the given boundaries correlate to the area of the total state or just the land area. Show your results in the final report.
3. Create a simple GUI from which the user selects a state, and then the program calculates the area using the two approaches in Task #1.1, display the results, as well as the area information in StateInfo.xlsx. All area results should be displayed in in km^2.

3.1.4 TASK #2

The information in the array of structures states must be combined with the information in StateInfo.xlsx and Cities.xlsx, to create a new array of structures call *newstates*. The program needs to read in the information and process it using a constructor function to create the array of structures *newstates* that must have the following cell fields:

N x1 struct array with fields:
- Geometry
- BoundingBox
- Lon
- Lat
- Name
- LandArea
- WaterArea
- Capital
- CapLat
- CapLon
- CostElec
- Cities
- CitLat
- CitLon

3.1.5 TASK #3

In this task you will display map information using the *newstates* structure defined in Task #2.

1. Create program that uses the information contained in newstates to
   1.1. draw a equirectangular projected map of a region specified by a four sided polygon described by $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \phi_1, \phi_2, \phi_3, \phi_4\}$
   1.2. display the location and the names of all state capitals in the region, and
   1.3. indicate the position of all the cities in the region (do not display the city names).
   In your report use as a bounding box that include California as an example of representative output.

2. Extend the program written in Task 3 Part 1 to create a GUI that will
   2.1. display a map of the US using a Mercator projection, including Hawaii and Alaska (displays as insets)
   2.2. let the user interactively select a region to display. In some manner the user needs to define a polygon. You need to decide what the most convenient manner doing this will be.
   2.3. The selected region is displayed in a same manner as described in Task 3 Part 1.
3. Now, modify the program written in Task 3 Part 2 to allow the user to hover over the map using the mouse and then the name of the city closest to the mouse position is displayed (if you cannot get the "hovering" to work, as a backup approach, the user can click on the map).

3.1.6 TASK #4

In this task, you will write a program to calculate the solar power incident in a specified month onto a selected area on the earth.

$$P = \iint S(\phi, \lambda)\, dA$$

1. Create a program that uses the information contained in *newstates* and the insolation database to display the solar radiation in W/m^2 in a selected state for a specified month. The information must be displayed as a color plot with a specified ϕ_{inc}=λ_{inc}= δ degrees spacing. You should perform two dimensional interpolation, using for example the function **interp2**, to regrid the $S_{\phi_k \lambda_l}$ data to the desired grid spacing δ. The plot should be generated using the Matlab function **pcolor** with interpolated shading. The scale of the plot should be displayed so that it is clear what color correlates with what numerical value. In your report use Alaska and Texas in March as an example of representative output.
2. Based on the program developed in Task 4.1, calculate the total power in Watts incident onto a selected state.
   2.1. First write a test programs to calculate surface integrals.
      2.1.1. Step 1: Calculates a surface integral defined by a polygon on a planar surface, using similar input parameters as the surface integral that must eventually calculated on a sphere. As a test case, you can a polygon that describes a square, and evaluate
      $$\int_{y_0}^{y_1} \int_{x_0}^{x_1} f(x, y)\, dxdy$$
      where $f(x, y)$ is function that can be easily integrated in close form (Hint: you can check your close form answers using Wolfram Alpha) . Show these tests in the final report.
      2.1.2. Extend the 2.1.1. approach to a spherical surface. Select a suitable test case and report on the tests in the final report.
   2.2. With the programs developed in 2.1 in place, you can proceed. Assuming that the whole state was covered by solar panels, how much will the state make in a given month if it sells the electricity, at the rates given in StateInfo.xlsx assuming that the efficiency of conversion of the solar panels is a given percentage. In your report use Alaska and Texas in March as an example of representative output.
3. Generalize the approach in Tasks 4.1-4.2, and write a program that accepts a four sided polygon described by $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \phi_1, \phi_2, \phi_3, \phi_4\}$ from a user, which could span multiple states in the United States, and calculate the same information as in Task 4.2 is calculated. In your report use a bounding box that includes California and some surrounding states. Use March as an example of representative output.

3.1.7 TASK #5

This task combines the approaches used in the previous tasks to generate an interactive GUI which

1. Displays a map of the US
2. The user selects a region on the map.
3. The selected region is displayed along with the capital of the states, and the cities in the state is identified by location and name as described previously)

4. The user then selects a four sided polygon within the region as well as a month. The program displays the information as described in Task 4.3.

You are in complete control in designing the GUI. The GUI could have a single or multiple pages. It should be possible for the user to seamlessly navigate the interface to, for example, reset the parameters and start all over without leaving the program.