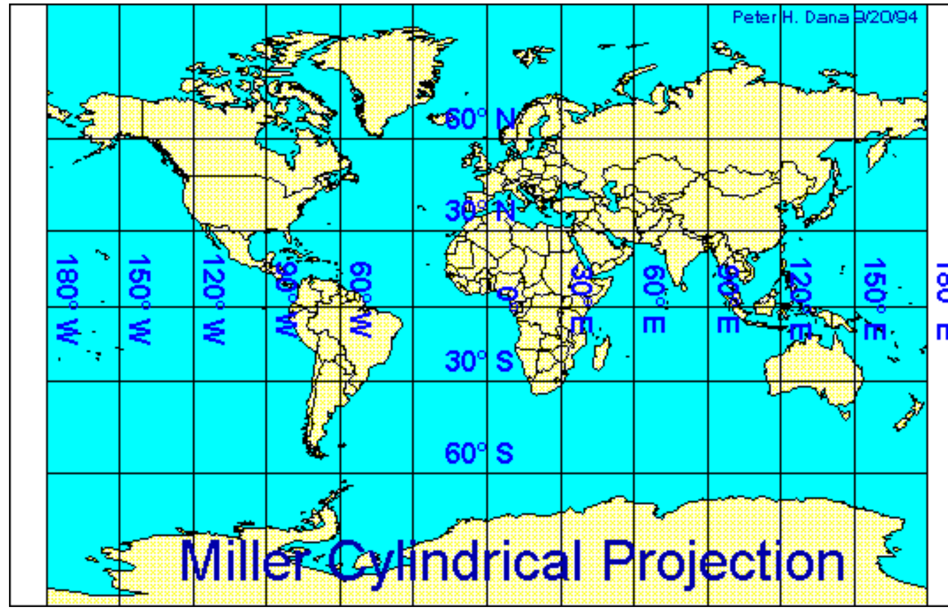
Map Projections

Map Projections

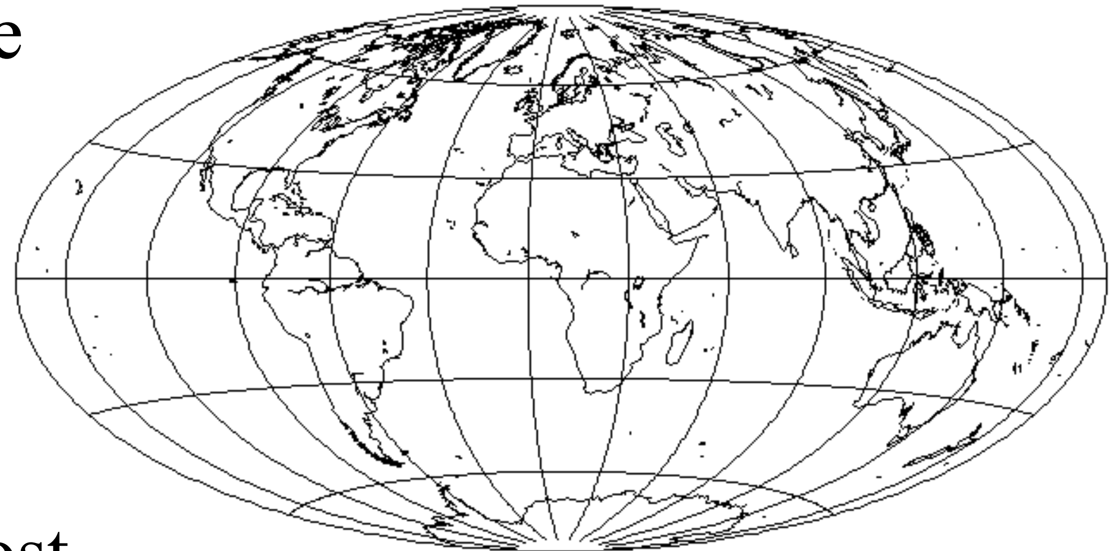
- The problem of forcing a spherical surface onto a map is an old problem in cartography
- Spherical areas do not appear equal when projected into Cartesian coordinates



- Consider how the cylindrical projection in the image greatly inflates the area of the poles (e.g., Antarctica)
- An extensive discussion of solutions to this problem is linked to from the syllabus under *Map Projections*

Equal Area Projections

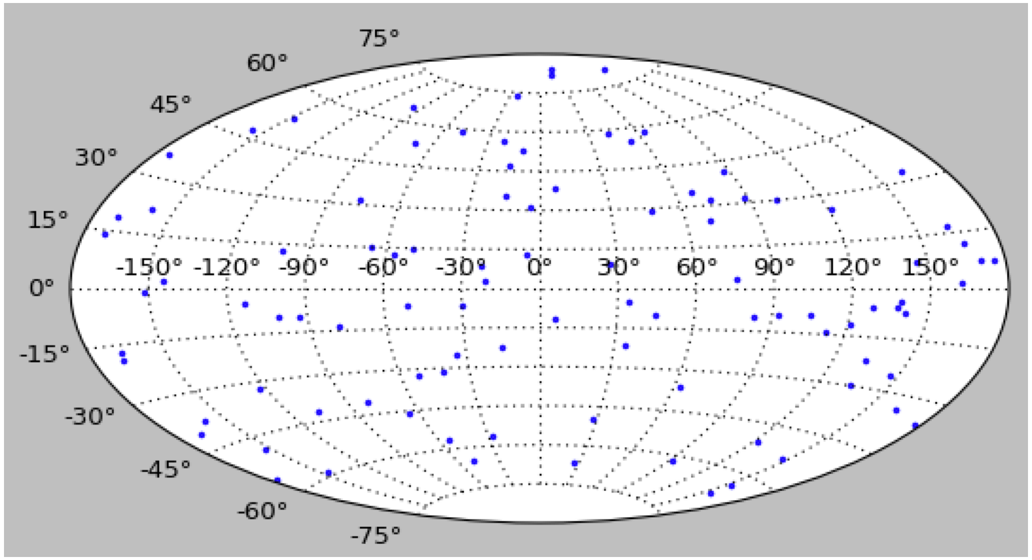
- It is possible to create mappings in which spherical surfaces are equal-area when projected onto a flat surface



- The solution used most often to display sky areas in astronomy is the Hammer-Aitoff projection (depicted for the Earth in the image)
- Thie Hammer-Aitoff projection *is* equal area (note how much smaller Antarctica is in this depiction)
- More information, including the equations behind the Hammer-Aitoff projection are at the syllabus links

Map Projections in Python

- Projections such as the Hammer-Aitoff are available in matplotlib



- The general set of commands is
  - *import matplotlib.pyplot as plt*
  - *fig = plt.figure()*
  - *ax = fig.add_subplot(111, projection="aitoff")*
    - here 111 means "subplot 1 of a 1x1 grid of plots"
  - *ax.scatter(ra, dec); fig.show()*
    - ra, dec here must be in radians with -π < ra < π

Map Projections in Python

- Other useful commands and keywords include
  - *ax.scatter(ra,dec,marker=‘o’,color=‘b’,s=0.7,alpha=0.5)*
    - here I supplied the points a shape, color, size and opacity
    - the points will be small blue, half-transparent circles
  - *xlab = [‘14h','16h','18h','20h','22h','0h','2h','4h','6h','8h','10h']*
  - *ax.set_xticklabels(xlab, weight=800)*
    - here I supplied x-axis labels and made them **bold**
    - the point of xlab is to label in <u>hours</u> instead of degrees
  - *ax.grid(color='k', linestyle='solid', linewidth=0.5)*
    - here I drew a grid of axes of a given style and thickness
    - the grid will be black, solid, and not too thick

Python tasks

1. Write a function to generate a random set of 10000 points populated in equal area on the surface of the sphere with coordinates ra,dec (α,δ) in *radians*
   - *from numpy.random import random*
   - *ra = 2*np.pi*(random(10000)-0.5)*
   - *dec = np.arcsin(1.-random(10000)*2.)*
   - plot (ra,dec) on a standard (x,y) grid…are there more points near the poles or near the equator of the sphere?
2. Now plot your points in an *Aitoff* projection
   - Change the x-labels to hours instead of degrees
   - Add a thick, blue, dashed axis grid using *grid*
   - Change your plot to a *Lambert* projection

Python tasks

3. Make a plot (binned at 1^o) to map Galactic dust at $0^o < \alpha < 360^o$ and $-90^o < \delta < 90^o$ in *Aitoff* projection
   - Generate a grid in (α,δ) in *degrees with* $0.5 < \alpha < 359.5^o$ and $-89.5^o < \delta < 89.5^o$ use a 1^o x 1^o binning
     - *see np.meshgrid from the dust maps lecture*
     - *make RA increase to the left/East*
   - Determine the values of the reddening (what we called *ebmv* in the *dust maps* lecture) at each (α,δ)

Python tasks

- Convert (α,δ) to (*x*,*y*) in Aitoff projection:
  - *from astropy import wcs*
  - *w = wcs.WCS(naxis=2)*
  - *w.wcs.cdelt=[1,1]*
  - *w.wcs.crval=[0.5,-89.5]*
  - *w.wcs.crpix=[1,1]*
  - *w.wcs.ctype = ["RA---AIT", "DEC--AIT"]*
  - *xmap,ymap = w.wcs_world2pix(ramap, decmap, 0)*
- Plot (α,δ) and (x,y)
  - *lev = np.arange(50)*0.03*
  - *plt.contourf(xmap,ymap,ebv_map,levels=lev)*
  - *plt.contourf(ramap,decmap,ebv_map,levels=lev)*

Python tasks

- Use *contour* (see the *dust maps* lecture) to create and plot contours for *ebmv* at each (*x*,*y*)...these should now be correctly projected
  - *note that your plot won't be* ***quite*** *correct as we didn't bin in* ***equal area*** *in the first step*
- Plot the Galactic Plane. Does the dust follow the Galaxy?