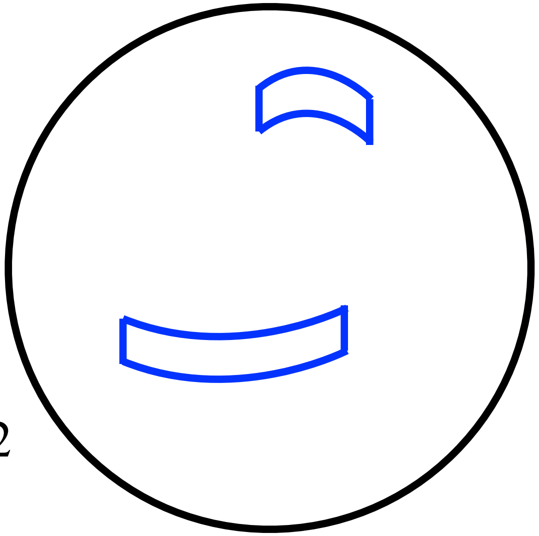
Areas on the Sphere and HEALPix

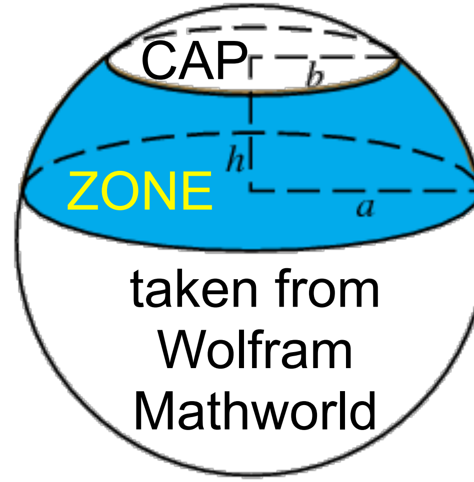
Areas on the sphere



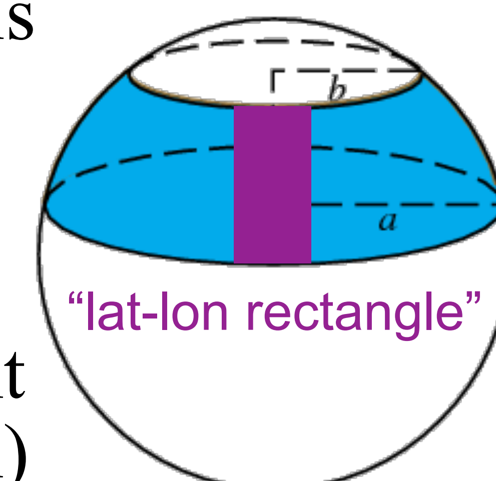
- I've provided you with a lot of options for determining distances on the sphere...but what about areas?
- The area of the entire (unit) sphere is 4π steradians or about 41252.96 deg^2
- One way to keep track of the area of regions of the sphere is to just subdivide it
  - half the sphere has an area of 2π steradians (41252.96/2 deg^2), a quarter of the sphere has an area of π steradians (41252.96/4 deg^2), etc.
- Or, spherical calculus tells us the area of a zone (the surface area of a spherical segment)

Areas on the sphere

- The area of a zone (on the unit sphere) is *$2\pi h$* in *steradians* (see the link to Wolfram MathWorld on the syllabus)



- The area of a *cap* is then *$2\pi(1-h)$*.
  - The spherical cap will come in very useful in the next lecture
- The area of a "*rectangle drawn on the sphere,*" which is a fraction of a zone, is *$f2\pi h$* where *f* is the fraction in this "*lat-lon rectangle*"



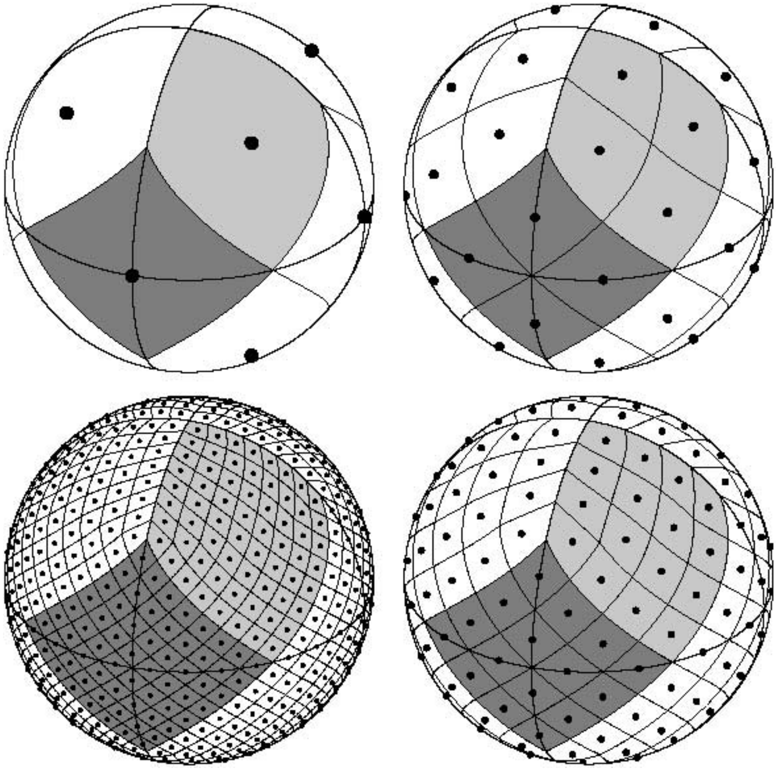
- A "*lat-lon rectangle*" as I'll call it (it doesn't have an "official" name) is bounded by lines of longitude (or Right Ascension) and latitude (or declination)

Hierarchical, Equal Area, iso-Latitude Pixelization

- Areas on the sphere become yet more complex if they are not simple astronomical fields bounded by lines of Right Ascension and declination
- So, a number of tricks have been developed to keep track of areas in large surveys of the sky
- One such trick, *HEALPix*, relies on the idea from a few slides ago (1/2 the sphere is 2π steradians, 1/4 is π steradians, etc.) and is a genuine quad-tree scheme
- Go to the syllabus' JPL HEALPix primer link
  - read *Discretization of Functions on the Sphere* (pay particular attention to Figure 2)
  - also read *Geometric and Algebraic Properties*...

Hierarchical, Equal Area, iso-Latitude Pixelization

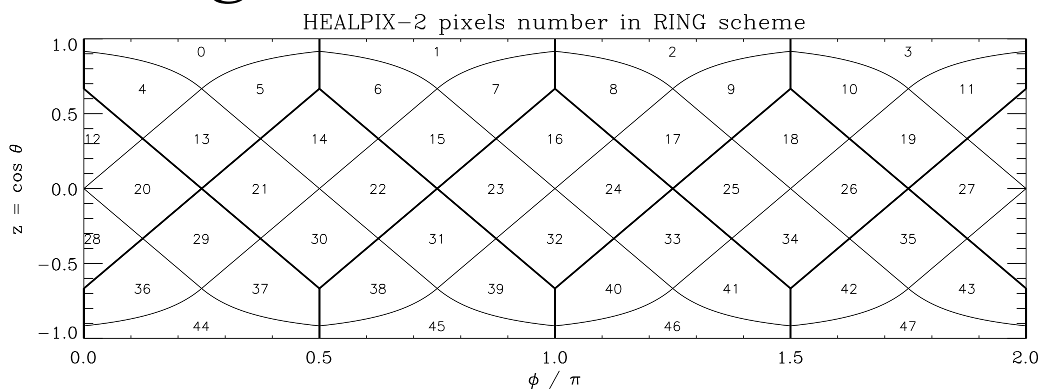
- *Nside* expresses the resolution of the grid. For *Nside* resolution you get 4**Nside*-1 isolatitude rings and 12*$Nside^2$ pixels.
- The base-resolution has 12 pixels in 3 rings around the poles and equator. *Nside* is the number of divisions on a base-resolution pixel to reach the desired resolution.

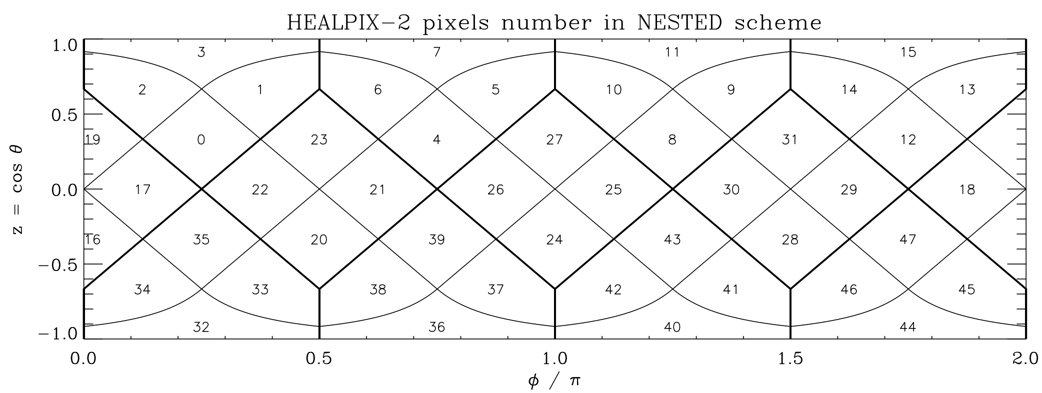


Nside = 1, 2, 4, 8

Hierarchical, Equal Area, iso-Latitude Pixelization

- Pixels have equal area and are centered on lines of constant latitude.
- There are two indexing schemes for pixels, *ring* and *nested*. The *ring* scheme is the default.





Hierarchical, Equal Area, iso-Latitude Pixelization

- Install the Python version of *HEALPix* in *astroconda*
  - *conda config --add channels conda-forge*
    conda install healpy
- It can be called and used, e.g., as follows:
  - *import healpy; healpy.ang2pix(nside,theta,phi)*
  - *theta* and *phi* are in radians, *phi* = RA and *theta* = [π/2 (radians) - dec] i.e. *theta = 0* is the north pole (dec = 90^{o})..see the wikipedia definition linked from the syllabus
- The most useful commands for our purposes are linked from the syllabus under *HEALPix Pixelisation related functions*

Python tasks

1. Generate a random set of 1000000 points on the surface of the sphere with coordinates ra,dec (α,δ) degrees that correctly populate the sphere equally in area, recall:
   - *ra = 360.*(random(1000000))* and
   - *dec=(180/np.pi)*np.arcsin(1.-random(1000000)*2.)*
   - plot your points, note density near the poles and equator.

2. Use *ang2pix* with *nside=1* to determine which pixels each of your *ra, dec* points lie within at the *nside=1* level of the *HEALpix* hierarchy
   - convert *ra, dec* to radians and take 90^{o} (π/2 radians) - *dec* so that *ra* becomes *phi* and *dec* becomes *theta*
   - What is the area of an *nside=1 HEALpix* pixel?

Python tasks

3. Use the *numpy.histogram* command to print out how many of your points lie in each *HEALpix* pixel
   - Is the answer consistent with pixels being equal-area?
4. *numpy.where* will return the indices that obey a logic command. So, if you've called your array of pixels "pix" then *w = np.where(pix == 2)* will make *w* a list of indices for which *phi*, *theta* (or *ra*, *dec*) lie in pixel 2
   - Plot *ra, dec* using matplotlib marker 'k.' and over-plot *ra[w], dec[w]* for those points in pixel 2, using a different color. Repeat for pixel 5 and pixel 8
5. Use *ang2pix* with *nside=2* to map your *ra,dec* points to HEALpix at the next level of the hierarchy
   - Which *nside=2* pixels lie inside pixel 5 at *nside=1*?