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 $4\pi$ 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 $2\pi$ steradians (41252.96/2 deg$^2$), a quarter of the sphere has an area of $\pi$ 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).
Areas on the sphere

- From the coordinate discussion of a few lectures ago, we can easily find the $h$ in $f2\pi h$
  
  \[ h = z_2 - z_1 = \sin \delta_2 - \sin \delta_1 \]

- $2\pi f$ depends on the fraction of the full circle covered by the $\alpha$ range of interest (in radians $2\pi f$ is just the difference in $\alpha$):
  
  \[ 2\pi f = (\alpha_2^{\text{radians}} - \alpha_1^{\text{radians}}) \]

- From $f2\pi h$, the area of a lat-lon rectangle bounded by $\alpha$ and $\delta$ is...
  
  \[ (\alpha_2^{\text{radians}} - \alpha_1^{\text{radians}})(\sin \delta_2 - \sin \delta_1) \]
Areas on the sphere

- So, in steradians, the area of a lat-lon rectangle bounded by Right Ascension $\alpha$ and declination $\delta$ is
  
  $$(\alpha_2^{\text{radians}} - \alpha_1^{\text{radians}})(\sin\delta_2 - \sin\delta_1)$$

- Then, the area of a lat-lon rectangle bounded by $\alpha$ and $\delta$ is given by…
  
  $$(180/\pi)(180/\pi)(\alpha_2^{\text{radians}} - \alpha_1^{\text{radians}})(\sin\delta_2 - \sin\delta_1)$$

  …in square degrees

- Or, in a more compact form useful when working with astronomical coordinates (for which $\alpha$ is usually expressed in degrees)
  
  $$(180/\pi)(\alpha_2^{\text{degrees}} - \alpha_1^{\text{degrees}})(\sin\delta_2 - \sin\delta_1)$$
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 $2\pi$ steradians, 1/4 is $\pi$ 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

- We have a Python version of HEALPix in astroconda
- It can be called and used, e.g., as follows
  - `import healpy`
  - `healpy.ang2pix(nside, theta, phi)`
  - Note that `theta` and `phi` are in radians, and that `phi` corresponds to Right Ascension and \([\pi/2\) (radians) - declination] corresponds to `theta`
  - i.e. `theta = 0` is the north pole (90°), 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 \( \alpha, \delta \) degrees that correctly populate the sphere equally in area

   - \( ra = 360. \times \text{random}(1000000) \) and
   - \( \text{dec} = (180/\pi) \times \text{np.arcsin}(1.-\text{random}(1000000)*2.) \)

   - plot your points...is there a higher density of points near the poles or the equator of the sphere?

2. Use \texttt{ang2pix} with \texttt{nside=1} to determine which pixels each of your \( ra, \text{dec} \) points lie within at the \texttt{nside=1} level of the \texttt{HEALpix} hierarchy

   - convert \( ra, \text{dec} \) to radians and take \( 90^\circ \) (\( \pi/2 \) radians) - \( \text{dec} \) so that \( ra \) becomes \textit{phi} and \( \text{dec} \) becomes \textit{theta}
   - What is the area of an \texttt{nside=1 \textit{HEALpix} pixel}?
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_ring` 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`?