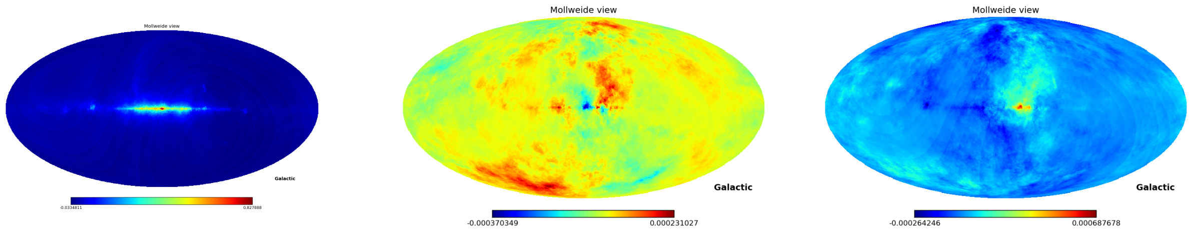
INTENSITY MAPPING TECHNIQUES

T. ANSAH - NARH

August 14, 2014

RESAMPLING IMAGE DATA

```
In [178]: import numpy as np

In [178]: import healpy as hp

In [178]: import pyfits

In [178]: from scipy import signal

In [178]: from scipy.interpolate import griddata

In [170]: n1= hp.pixelfunc.remove_dipole(c[68][2], nest=True)

monopole: -1.07764e-06  dipole: lon: 157.221, lat: -71.3636, amp: 1.31231e-05

In [186]: n1= hp.pixelfunc.remove_dipole(c[68][0], nest=True)
monopole: 0.0343881  dipole: lon: -177.802, lat: -73.4226, amp: 0.00927786

In [188]: n1= hp.pixelfunc.remove_dipole(c[68][1], nest=True)
monopole: -1.2806e-06  dipole: lon: 110.132, lat: -44.9694, amp: 6.75658e-06

In [190]: n1= hp.pixelfunc.remove_dipole(c[68][3], nest=True)
monopole: 0  dipole: lon: 0, lat: nan, amp: 0
```



(a) Intensity I **(b) Intensity Q** **(c) Intensity U**

Figure 1: Mollview Projection for I, Q and U

SINC FUNCTION INTERPOLATION

```
 In [12]: u = np.array(dt, dtype=np.floa)
np.float     np.float16   np.float64   np.floating
np.float128  np.float32   np.float_

In [12]: u = np.array(dt, dtype=np.float)

In [13]: z = signal.resample(u[0][0],30)

In [14]: w,h = signal.freqz(z)

In [15]: p.plot(w, np.sinc(h), 'b')
Casting complex values to real discards the imaginary part
Out[15]: [<matplotlib.lines.Line2D at 0x4880210>]
In [19]: p.subplot(1,2,1)
Out[19]: <matplotlib.axes.AxesSubplot at 0x5567d10>

In [20]: p.plot(w, np.sinc(h), 'b')
Out[20]: [<matplotlib.lines.Line2D at 0x5636950>]

In [21]: p.subplot(221)
Out[21]: <matplotlib.axes.AxesSubplot at 0x5644d90>

In [22]: p.plot(w, np.sinc(h), 'b')
Out[22]: [<matplotlib.lines.Line2D at 0x55a5c50>]
In [23]: p.title('SINC FUNCTION')
Out[23]: <matplotlib.text.Text at 0x565b090>

In [24]: p.xlabel('SINC')
Out[24]: <matplotlib.text.Text at 0x5647950>

In [25]: p.ylabel('SINC')
Out[25]: <matplotlib.text.Text at 0x564d290>

In [26]: p.xlabel('NORMALIZED FREQUENCY (radians/sample)')
Out[26]: <matplotlib.text.Text at 0x5647950>

In [27]: p.title('SINC RESPONSE FUNCTION')
Out[27]: <matplotlib.text.Text at 0x565b090>

In [28]: p.legend('I')
Out[28]: <matplotlib.legend.Legend at 0x55b2710>

In [29]: z = signal.resample(u[0][1],30)

In [30]: w,h = signal.freqz(z)

In [31]: p.subplot(222)
Out[31]: <matplotlib.axes.AxesSubplot at 0x55b6790>
```

```
In [32]: p.plot(w, np.sinc(h), 'b')
Out[32]: [<matplotlib.lines.Line2D at 0x5699790>]

In [33]: p.subplot(2,2,2)
Out[33]: <matplotlib.axes.AxesSubplot at 0x55b6790>

In [34]: p.plot(w, np.sinc(h), 'b')
Out[34]: [<matplotlib.lines.Line2D at 0x56a30d0>]

In [35]: p.xlabel('NORMALIZED FREQUENCY (radians/sample)')
Out[35]: <matplotlib.text.Text at 0x55ae190>

In [36]: p.ylabel('SINC')
Out[36]: <matplotlib.text.Text at 0x55b7650>

In [37]: p.title('SINC RESPONSE FUNCTION')
Out[37]: <matplotlib.text.Text at 0x558a450>

In [38]: p.legend('Q')
Out[38]: <matplotlib.legend.Legend at 0x569eb90>

In [39]: z = signal.resample(u[0][2],30)

In [40]: w,h = signal.freqz(z)

In [41]: p.subplot(2,2,3)
Out[41]: <matplotlib.axes.AxesSubplot at 0x5654790>

In [42]: p.plot(w, np.sinc(h), 'b')
Out[42]: [<matplotlib.lines.Line2D at 0x56d8d10>]

In [43]: p.xlabel('NORMALIZED FREQUENCY (radians/sample)')
Out[43]: <matplotlib.text.Text at 0x56722d0>

In [44]: p.ylabel('SINC')
Out[44]: <matplotlib.text.Text at 0x5674a50>

In [45]: p.title('SINC RESPONSE FUNCTION')
Out[45]: <matplotlib.text.Text at 0x56838d0>

In [46]: p.legend('U')
Out[46]: <matplotlib.legend.Legend at 0x56d9850>

In [47]: p.title('')
Out[47]: <matplotlib.text.Text at 0x56838d0>

In [48]: z = signal.resample(u[0][3],30)

In [49]: w,h = signal.freqz(z)
```

```
In [50]: p.subplot(2,2,4)
Out[50]: <matplotlib.axes.AxesSubplot at 0x48ebdd0>

In [51]: p.plot(w, np.sinc(h), 'b')
Out[51]: [<matplotlib.lines.Line2D at 0x50a9f10>]

In [52]: p.legend('V')
Out[52]: <matplotlib.legend.Legend at 0x56d8550>

In [53]: p.xlabel('NORMALIZED FREQUENCY (radians/sample)')
Out[53]: <matplotlib.text.Text at 0x56e16d0>

In [54]: p.ylabel('SINC')
Out[54]: <matplotlib.text.Text at 0x56f2f10>

In [55]:
```

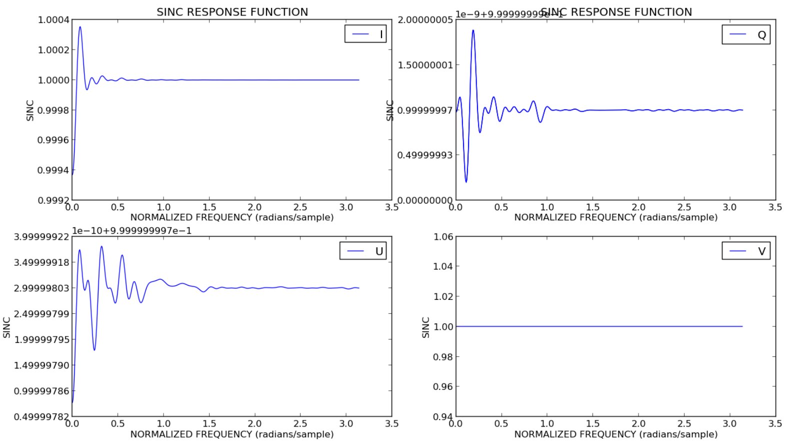


Figure 2: SINC RESPONSE FUNCTION

BILINEAR INTERPOLATION

```
In [78]: grid_x, grid_y = np.mgrid[0:1:30j,0:1:20j]

In [80]: s1=hp.get_interp_val(dt[0][0], grid_x,grid_y, nest=False)

In [95]: p.subplot(221)
Out[95]: <matplotlib.axes.AxesSubplot at 0x7951ad0>

In [96]: p.imshow(s1,extent=(0,1,0,1),origin='lower')
Out[96]: <matplotlib.image.AxesImage at 0x7684710>

In [97]: p.title('I')
Out[97]: <matplotlib.text.Text at 0x7653290>

In [98]: s1=hp.get_interp_val(dt[0][1], grid_x,grid_y, nest=False)

In [99]: p.subplot(222)
Out[99]: <matplotlib.axes.AxesSubplot at 0x76849d0>

In [100]: p.imshow(s1,extent=(0,1,0,1),origin='lower')
Out[100]: <matplotlib.image.AxesImage at 0x959cb50>

In [101]: p.title('Q')
Out[101]: <matplotlib.text.Text at 0x9586ed0>

In [102]: s1=hp.get_interp_val(dt[0][2], grid_x,grid_y, nest=False)

In [103]: p.subplot(223)
Out[103]: <matplotlib.axes.AxesSubplot at 0x959c5d0>

In [104]: p.imshow(s1,extent=(0,1,0,1),origin='lower')
Out[104]: <matplotlib.image.AxesImage at 0x8d7a6d0>

In [105]: p.title('U')
Out[105]: <matplotlib.text.Text at 0x95aff10>

In [106]: s1=hp.get_interp_val(dt[0][3], grid_x,grid_y, nest=False)

In [107]: p.subplot(224)
Out[107]: <matplotlib.axes.AxesSubplot at 0x8d7ab10>

In [108]: p.imshow(s1,extent=(0,1,0,1),origin='lower')
Out[108]: <matplotlib.image.AxesImage at 0x9ce8e90>

In [109]: p.title('V')
Out[109]: <matplotlib.text.Text at 0x95ac650>
```

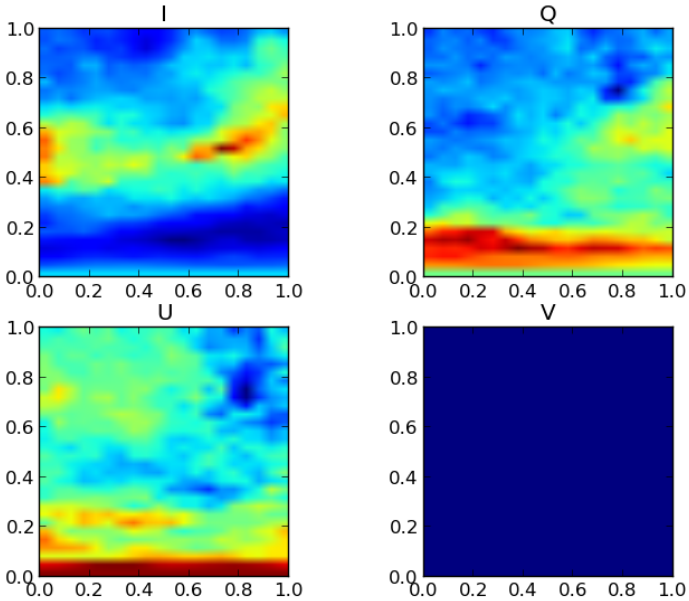


Figure 3: BILINEAR FUNCTION

ANGULAR POWER SPECTRUM

```
In [144]: n1= hp.pixelfunc.remove_dipole(dt[60][0], nest=True)
monopole: 0.0361069  dipole: lon: -177.053, lat: -73.2383, amp: 0.0100991

In [145]: p.loglog(hp.anafast(n1))
Out[145]: [<matplotlib.lines.Line2D at 0xa4e33d0>]

In [146]: n1= hp.pixelfunc.remove_dipole(dt[60][1], nest=True)
monopole: -2.45033e-05  dipole: lon: 61.4464, lat: 24.6892, amp: 3.28706e-05

In [147]: p.loglog(hp.anafast(n1))
Out[147]: [<matplotlib.lines.Line2D at 0xa4f5410>]

In [148]: p.loglog(hp.anafast(n1))
Out[148]: [<matplotlib.lines.Line2D at 0x9725690>]

In [149]: n1= hp.pixelfunc.remove_dipole(dt[60][2], nest=True)
monopole: -5.59359e-05  dipole: lon: 55.6266, lat: 73.3203, amp: 0.000214312

In [150]: p.loglog(hp.anafast(n1))
Out[150]: [<matplotlib.lines.Line2D at 0x13d4a650>]

In [151]: import matplotlib.pyplot as plt

In [152]: plt.legend(['I', 'Q', 'U'])
Out[152]: <matplotlib.legend.Legend at 0xd7c3610>
```

```
In [153]: p.xlabel('l')
Out[153]: <matplotlib.text.Text at 0x1745a0d0>

In [154]: p.ylabel('Cl')
Out[154]: <matplotlib.text.Text at 0xb3f7790>

In [155]: p.title('ANGULAR POWER SPECTRUM')
Out[155]: <matplotlib.text.Text at 0xa4e3610>
```

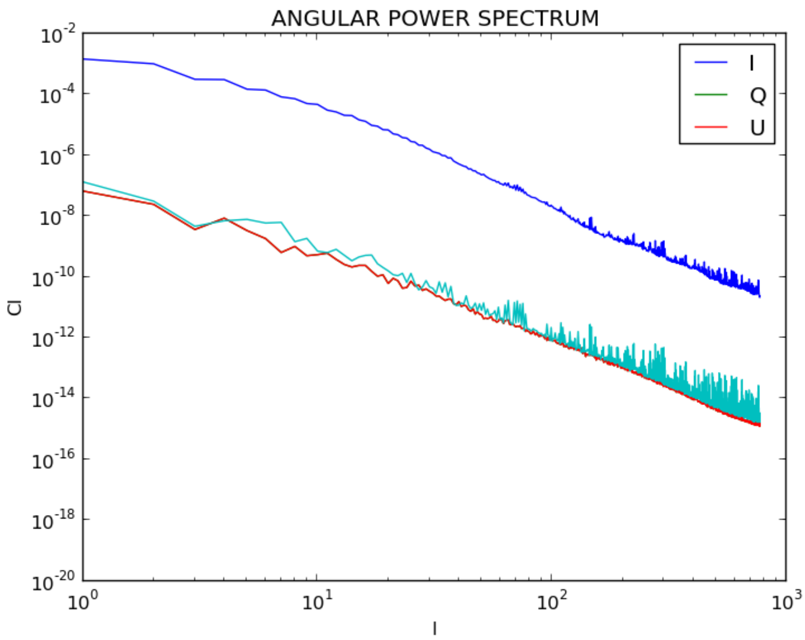


Figure 4: ANGULAR SPECTRUM PROJECTION