R/Phi Image Integration

This script lives in /sdf/group/lcls/ds/ana/tutorials/psana1_examples/radInteg.py and demonstrates how to do an angular integration of a 2D area-detector image. If you don't have a 2D image, but instead have pixel values and positions, consider using the more general BinnedStatistic1D/BinnedStatistic2D /BinnedStatisticDD classes in skbeam.core.accumulators.binned_statistic (not documented here, but documentation is available within IPython).

This method for doing radial integrations is part of the world-reusable scikit-beam project hosted by BNL. This goal of this python library is to provide low-level building-blocks that can be easily installed around the world enabling scientists to reuse familiar tools at different laboratories.

This example shows three different integrations, one one-dimensional (projecting onto the radial axis) and two two-dimensional (projecting onto the radial and phi axes) with and without masks. This code is based on the scipy "binned_statistic" code which allows for very flexible binning and many different operations on the pixels within a bin (sum/mean/count/user-defined)

```
from skbeam.core.accumulators.binned_statistic import RadialBinnedStatistic, RPhiBinnedStatistic
import numpy as np
img = np.reshape(np.arange(9),(3,3))
print('Image:\n',img)
mask = np.ones_like(img)
mask[1][1]=0
print(f'\nMask:\n{mask}')
radbinstat = RadialBinnedStatistic(img.shape, bins=3,
                                   statistic='sum',
                                   origin=(0,0),
                                   range = (0,2),
                                   mask=mask)
rphibinstat = RPhiBinnedStatistic(img.shape, bins=(3,1),
                                  statistic='sum',
                                  origin=(0,0),
                                  range = ((0,2),(0,np.pi/3)))
rphibinstat_mask = RPhiBinnedStatistic(img.shape, bins=(3,1),
                                       statistic='sum',
                                       origin=(0,0),
                                       range = ((0,2),(0,np.pi/3)),
                                       mask=mask)
print('\nAngular integration with mask:')
print(radbinstat(img))
print('\nBin edges and centers:')
print(radbinstat.bin_edges)
print(radbinstat.bin_centers)
print('\n2D R/Phi Angular integration (1 phi bin) with phi range and mask:')
print(rphibinstat_mask(img))
print('\n2D R/Phi Angular integration (1 phi bin) with phi range and no mask:')
print(rphibinstat(img))
print('\nR/Phi bin edges:')
print(rphibinstat.bin_edges[0])
print(rphibinstat.bin_edges[1])
```

The output of running this script is below. It attempts to demonstrate:

- the origin location (row=0, col=0)
- the range and mask argument: first integration results are 0, 4(=3+1), 8(=6+2) where the 4 value has been ignored because mask[1][1] is 0. Image values 5,7,8 are ignore because of the range limits.
- the units of R (pixels) and phi (radians) where phi is defined at arctan2(row/col) (i.e. x is array-column (second index) and y is array-row (first index))

Read the IPython help documentation using the "?" operator in IPython for more details, as well as the documentation for scipiy binned_statistic.

```
Image:
[[0 1 2]
 [3 4 5]
 [6 7 8]]
Mask:
[[1 1 1]
 [1 0 1]
 [1 1 1]]
Angular integration with mask:
[ 0.  4.  8.]
Bin edges and centers:
[ 0.          0.66666667  1.33333333  2.        ]
[ 0.33333333  1.          1.66666667]
2D R/Phi Angular integration (1 phi bin) with phi range and mask:
[[ 0.]
 [ 1.]
 [ 2.]]
2D R/Phi Angular integration (1 phi bin) with phi range and no mask:
[[ 0.]
 [ 1.]
 [ 6.]]
R/Phi bin edges:
[ 0.          0.66666667  1.33333333  2.        ]
[ 0.          1.04719755]
```