histogram

March 9, 2017

1 Histogram vs Histogram_lut

```
In [1]: import numpy
        from silx.math.histogram import Histogramnd, HistogramndLut
        from silx.gui.plot import Plot1D, Plot2D
        %gui qt
```

```
WARNING:fabioimage:PIL is not installed ... trying to do without
WARNING:tifimage:PIL is not installed ... trying to do without
WARNING:bruker100image:PIL is not installed ... trying to do without
WARNING:xsdimage:lxml library is probably not part of your python installation: dis
```

This function create some data with noise.

```
In [2]: def createDataSet():
            shape = (400, 400)
            xcenter = shape[0]/2
            ycenter = shape[1]/2
            t = numpy.zeros(shape)
            y, x=numpy.ogrid[:t.shape[0], :t.shape[1]]
            r=1.0+numpy.sin(numpy.sqrt((x-xcenter)**2+(y-ycenter)**2)/8.0)
            return r + numpy.random.rand(shape[0], shape[1])

        data = createDataSet()
```

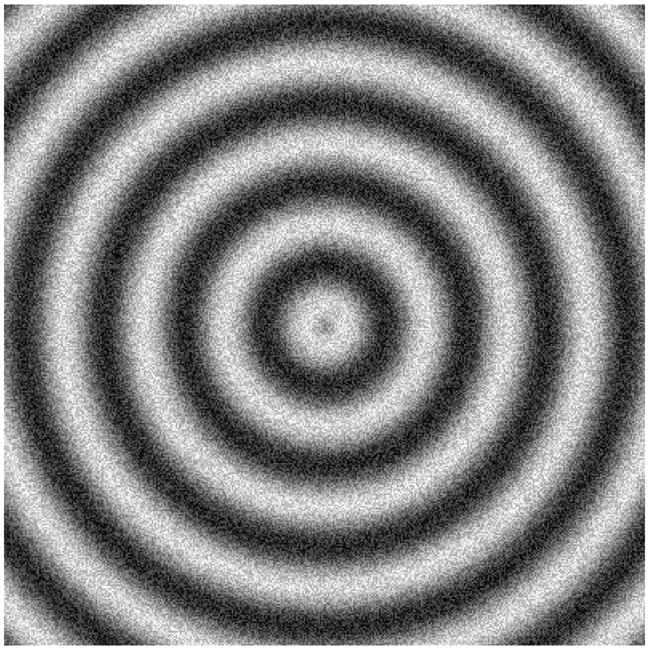
Simple display of the fist element of the list

```
In [3]: p = Plot2D()
        p.addImage(legend='dataExample', data=data)
        p.show()
```

1.1 Exercise : use Histogramnd to compute azimutal integration

1.1.1 we compute raddi to center for each pixel

```
In [4]: def computeradius(data):
            xcenter=data.shape[0]/2
```



input data

```
        ycenter=data.shape[1]/2
        y, x=numpy.ogrid[:data.shape[0], :data.shape[1]]
        r=numpy.sqrt((x-xcenter)**2+(y-ycenter)**2)
        return r
```

```
In [5]: radii = computeradius(data)
        plotRadii = Plot2D()
        plotRadii.addImage(radii)
        plotRadii.show()
```

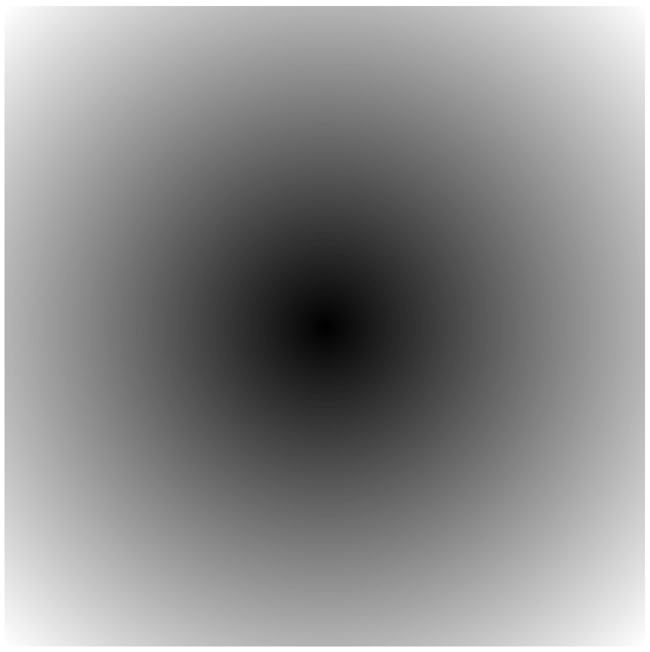
1.1.2 plot the histogram of the radii

documentation :

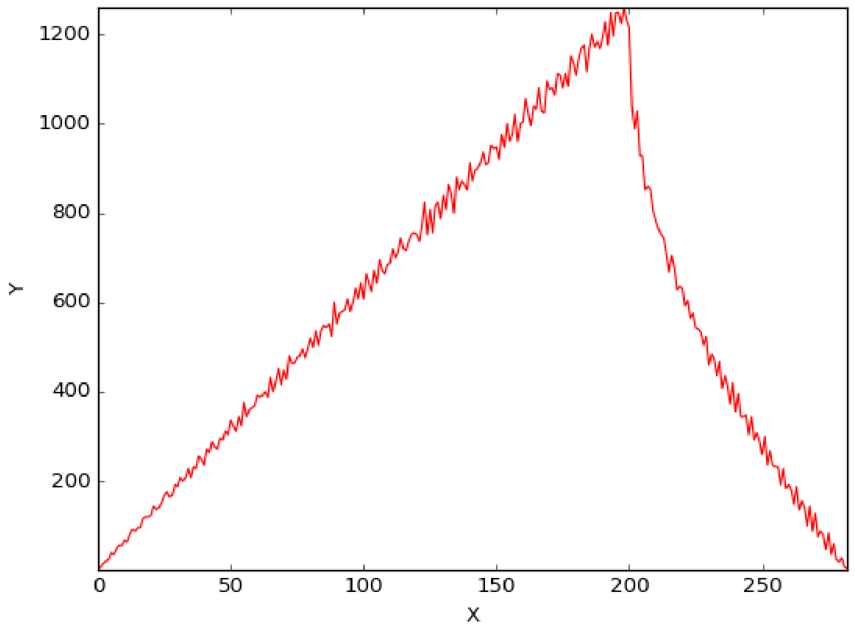
- http://pythonhosted.org/silx/modules/math/histogram.html

```
In [6]: nb_bins = int(numpy.ceil(radii.max()))
        histo_range = [0, nb_bins]
        histogram=Histogramnd(sample=radii.ravel(),
                              n_bins=nb_bins,
                              histo_range=histo_range)
```

```
In [7]: plotHisto = Plot1D()
        plotHisto.addCurve(x=range(nb_bins), y=histogram.histo, color='red')
        plotHisto.show()
```



distance pixel-image center



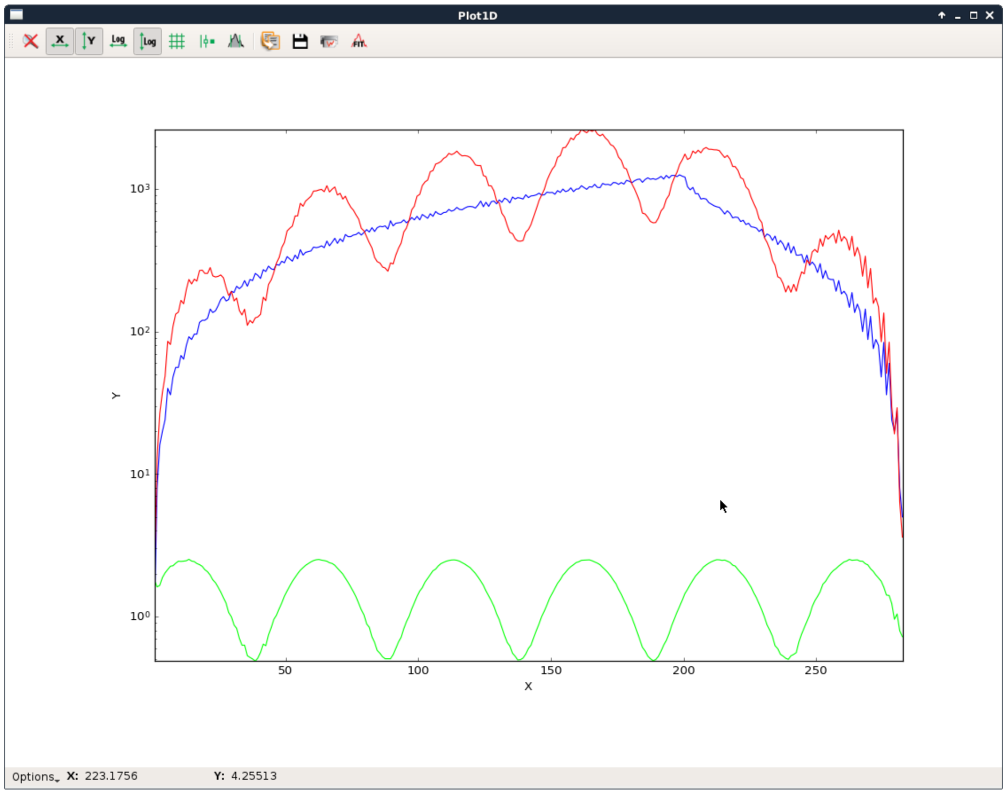
distance pixel-image center

1.1.3 compute azimutal integration

goal : get the mean contribution of each pixels for each radius
step 1 : get the contribution of each pixels for each radius

```
In [8]: nb_bins = int(numpy.ceil(radii.max()))
        histo_range = [0, nb_bins]
        histogram=Histogramnd(sample=radii.ravel(),
                              n_bins=nb_bins,
                              histo_range=histo_range,
                              weights=data.ravel())
```

step 2 : get the mean and plot it



integration

```
In [9]: plotHisto = Plot1D()
        binscenter=(histogram.edges[0][1:] + histogram.edges[0][0:-1]) / 2.0
        plotHisto.addCurve(x=binscenter, y=histogram.histo, ledend='h unweighted')
        plotHisto.addCurve(x=binscenter, y=histogram.weighted_histo, legend='h weig
        normalization=histogram.weighted_histo/histogram.histo
        plotHisto.addCurve(x=binscenter, y=normalization, legend='integration')
        plotHisto.show()
```

```
WARNING:silx.gui.plot.Plot:addCurve: deprecated extra arguments
```

2 Exercice : compute the azimutal integration over n images

we want to reproduced the same action but over a stack of image : - pixel distance two the center is not evolving - only pixel values are

```
In [10]: dataset = [ createDataSet() for i in range(10) ]
```

2.1 First way : using Histogramnd

```
In [11]: def computeDataSetHisto():
             histogram=None
             for d in dataset:
                 if histogram is None:
                     histogram=Histogramnd(radii.ravel(),
                                           n_bins=nb_bins,
                                           histo_range=histo_range,
                                           weights=d.ravel())
                 else:
                     histogram.accumulate(radii.ravel(), weights=d.ravel())

             return histogram
```

```
In [12]: # plot It
         plotDataSetHistoNd = Plot1D()
         histogramDS = computeDataSetHisto()
         binscenter=(histogramDS.edges[0][1:] + histogramDS.edges[0][0:-1]) / 2.0
         normalization=histogramDS.weighted_histo/histogramDS.histo
         plotDataSetHistoNd.addCurve(x=binscenter, y=normalization, color='red')
         plotDataSetHistoNd.show()
```

2.2 second way : using HistogramndLut

```
In [13]: def computeDataSetHistoLut():
             histogram=HistogramndLut(radii.ravel(),
                                      n_bins=nb_bins,
                                      histo_range=histo_range)
             for d in dataset:
                 histogram.accumulate(d.ravel())

             return histogram
```

```
In [14]: # plot It
         plotDataSetHistoLut = Plot1D()
         histogramLut = computeDataSetHistoLut()
         normalization=histogramLut.weighted_histo()/histogramDS.histo
         plotDataSetHistoLut.addCurve(binscenter, y=normalization, color='red')
         plotDataSetHistoLut.show()
```

2.3 Compare results

```
In [15]: numpy.array_equal(histogramLut.weighted_histo(), histogramDS.weighted_hist
```

```
Out[15]: True
```

2.4 Compare execution time

```
In [ ]: %timeit computeDataSetHisto()
```

```
In [ ]: %timeit computeDataSetHistoLut()
```