In [1]:
```
fname = "/Users/mender/PycharmProjects/diff_cor/p3deg_p8s_1_00001.cbf" # an image of silver nanopart
icles
```

In [2]:
```
import fabio,re
```

```
WARNING:xsdimage:lxml library is probably not part of your python installation: disabling xsdimage f
ormat
```

In [3]:
```
cbf = fabio.open(fname)
```

In [4]:
```
info = cbf.getheader()['_array_data.header_contents'].split('#')
```

In [5]:
```
detDist = next( x for x in info if re.search('Detector_distance', x) )
detDist = float( detDist.split()[1]) # meters
```

In [6]:
```
wavelen = next( x for x in info if re.search('Wavelength', x) )
wavelen = float( wavelen.split()[1] )  # angstroms
```

In [7]:
```
pixSize = 0.000172 # meters, should be fixed for pilatus
```

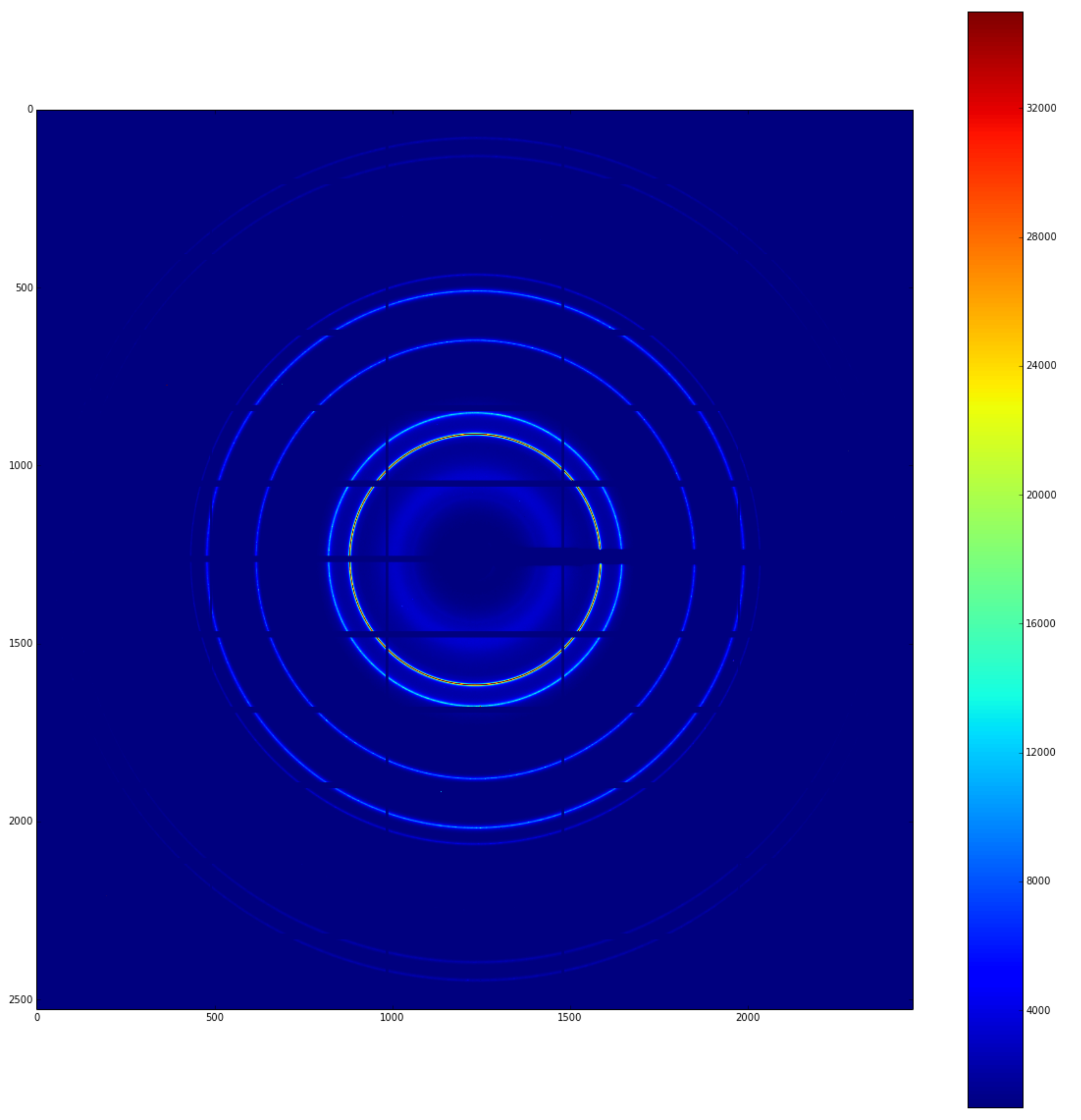
In [8]:
```
print detDist, wavelen
```

```
0.18801 0.7293
```

In [10]:
```
img = cbf.data
```

In [11]:
```
figure(1,figsize=(20,20));imshow( img, vmin=1000, vmax = 35000);colorbar()
```

Out[11]:
```
<matplotlib.colorbar.Colorbar instance at 0x10d29cbd8>
```



In [16]:
```
# we can guess the center and pixel radius of the 111 Bragg ring use popi to optimize:
qi = 352. # pixel unit, 111 reflection
beamX = 1231.5 # pixel unit ,default for pilatus 6M
beamY = 1263.5 # pixel unit, default for pilatus 6M
```

In [14]:
```
from popi.popi import polar
```

In [17]:
```
#Parameters
#----------#

#d       : numpy 2D numpy image array
#pixsize : pixel size in meters
#detdist : samplt -to -detector distance in meters
#wavelen : wavelength of xrays in angstroms

#OPTIONAL PARAMETERS
#------------------
#a : float, x center on detector (defaults to Xdim/2)
#b : float, y center on detector (defaults to Ydim/2)

pp = polar( img, pixSize, detDist, wavelen, beamX,beamY )
```

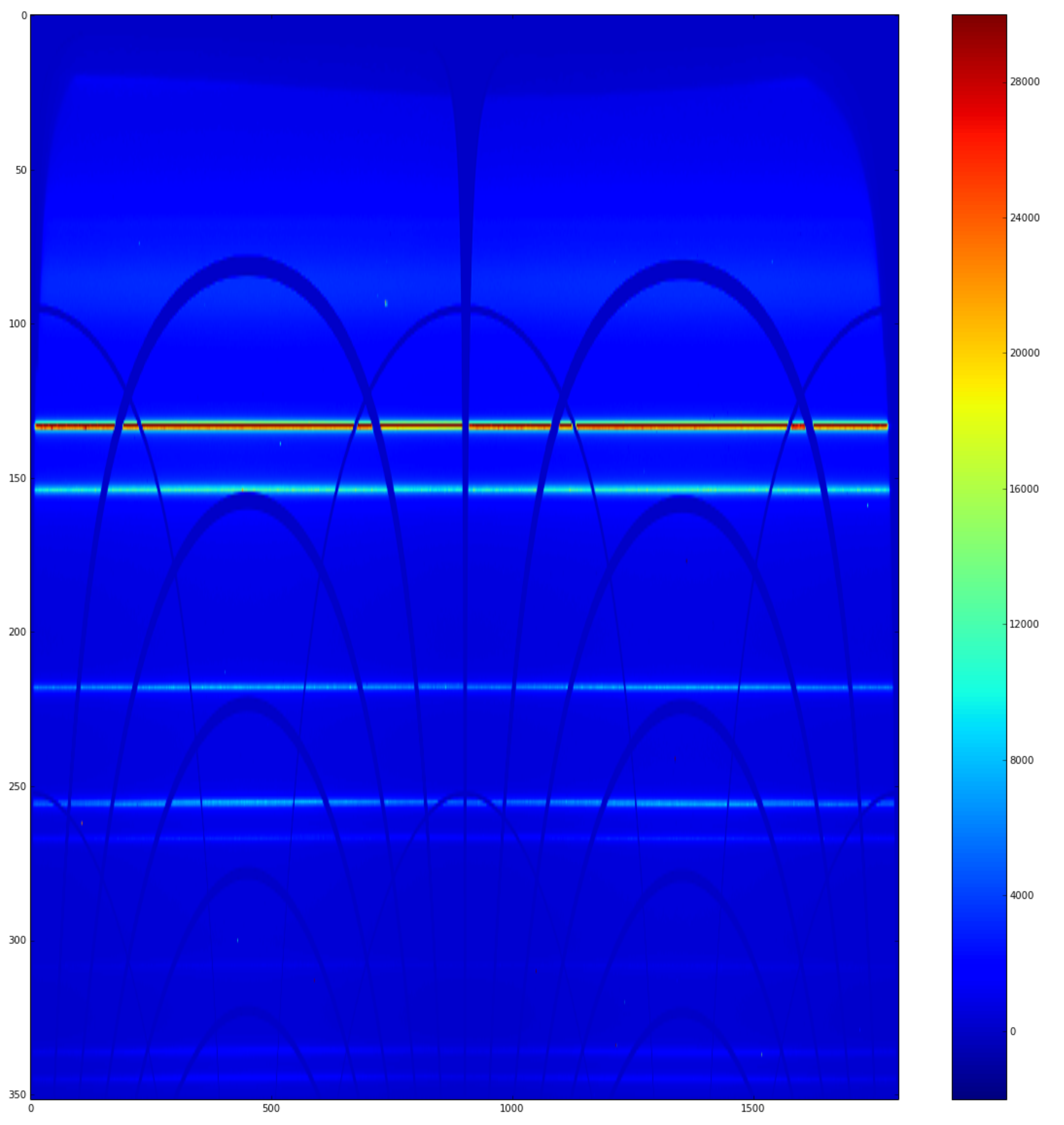
In [25]:
```
# Here we define a parameter space (center, radius in pixels) for a Bragg ring and then
# We then scan the space and choose the set of parameters which gives the Bragg ring of
# maximum angular average.

pp.center(qMin = qi-7, qMax = qi+7, center_res = 1, Nphi = 1000, size=10., dq = 1 )
qi = pp.q_center
pp.center(qMin = qi-1, qMax = qi+1, center_res = 0.1, Nphi = 3000, size=2., dq = 0.1)
#Finds the center of pilatus image:
#  polarpilatus.center(qMin,qMax,center_res=0.5,Nphi=50,size=20.)

#-------------------
#PARAMS
#-------------------
#qMin,qMax  : min and max ring position in pixel units
#center_res : resolution of desired center in pixel units
#Nphi       : number of phi bins when maximizing angular average
#size       : defines a box around the center of the detector (pixel units)
#dq         : resolution of the radius
```

In [39]:
```
# optimized parameters, pixel units
q = pp.q_center # Bragg ring pixel radius
beamX = pp.x_center
beamY = pp.y_center
```

In [56]:
```
qres = 0.02 # inverse angstroms
Nphi = 1800
polar_intensity = pp.Interpolate_to_polar(qres=qres, Nphi = Nphi)
Nq = polar_intensity.shape[0]
```

In [41]:
```
figure(2,figsize=(20,20)); imshow( polar_intensity, vmax = 30000,aspect='auto');colorbar()
```

Out[41]:
```
<matplotlib.colorbar.Colorbar instance at 0x10b7a25a8>
```



In [60]:
```
# Now we can mask all bad pixels:
polar_intensity[polar_intensity < 0] = -1
masked_pol_inten = numpy.ma.masked_equal( polar_intensity, -1 )
```

In [61]:
```
ang_ave = masked_pol_inten.mean(1)
q_range = [ x*qres for x in xrange( Nq )]
```

In [68]:
```
figure(3,figsize=(20,20))
plot( q_range, ang_ave,lw = 2)
xlabel(r'$q$ $[\AA^{-1}]$',fontsize=24)
ylabel(r'$I(q)$',fontsize=24)
```

Out[68]:
```
<matplotlib.text.Text at 0x12bf6d810>
```