

编程作业02

Group3: 黄磊 赵士云 郑婧娴

主要负责:

黄磊: Part II、Part III讨论、文档整理;

赵士云: Part III

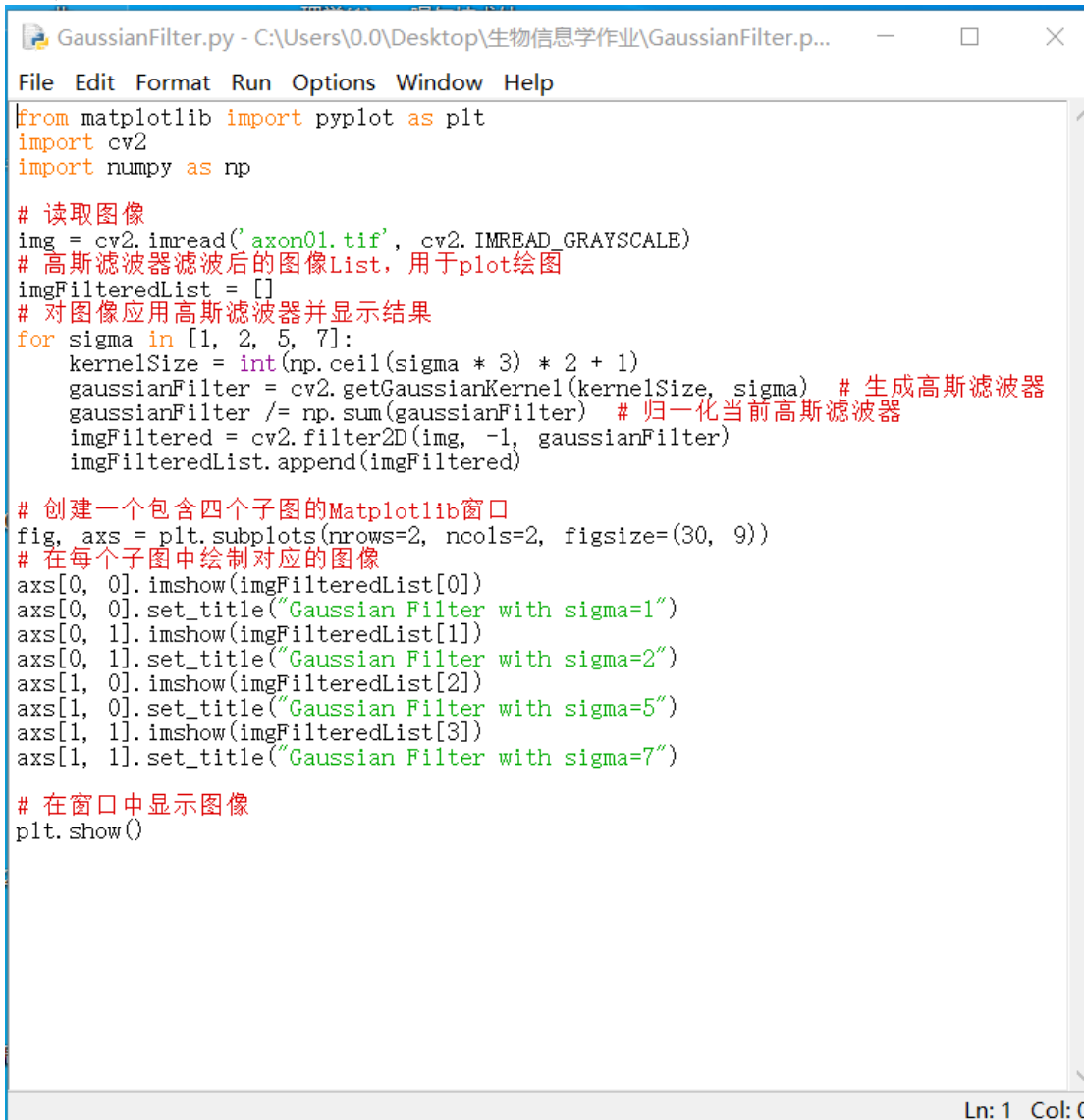
郑婧娴: Part I (该同学代码基础较为薄弱, 需要做一些代码学习, 所以负责比较简单的部分)

PART1: 使用高斯核进行空间滤波和梯度计算

摘要: 第一项编程作业是实现一个高斯滤波器, 代入不同参数到高斯核计算公式中。第二项是根据类中所涵盖的内容, 利用与 $\sigma=1,2,5$ 下的高斯核的一阶导数的卷积来计算水平和垂直方向上的导数显示衍生图像。

代码执行指令:

指令1:



```
from matplotlib import pyplot as plt
import cv2
import numpy as np

# 读取图像
img = cv2.imread('axon01.tif', cv2.IMREAD_GRAYSCALE)
# 高斯滤波器滤波后的图像List, 用于plot绘图
imgFilteredList = []
# 对图像应用高斯滤波器并显示结果
for sigma in [1, 2, 5, 7]:
    kernelSize = int(np.ceil(sigma * 3) * 2 + 1)
    gaussianFilter = cv2.getGaussianKernel(kernelSize, sigma) # 生成高斯滤波器
    gaussianFilter /= np.sum(gaussianFilter) # 归一化当前高斯滤波器
    imgFiltered = cv2.filter2D(img, -1, gaussianFilter)
    imgFilteredList.append(imgFiltered)

# 创建一个包含四个子图的Matplotlib窗口
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(30, 9))
# 在每个子图中绘制对应的图像
axs[0, 0].imshow(imgFilteredList[0])
axs[0, 0].set_title("Gaussian Filter with sigma=1")
axs[0, 1].imshow(imgFilteredList[1])
axs[0, 1].set_title("Gaussian Filter with sigma=2")
axs[1, 0].imshow(imgFilteredList[2])
axs[1, 0].set_title("Gaussian Filter with sigma=5")
axs[1, 1].imshow(imgFilteredList[3])
axs[1, 1].set_title("Gaussian Filter with sigma=7")

# 在窗口中显示图像
plt.show()
```

指令2:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# 读取图像
axon02 = cv2.imread("axon02.tif", cv2.IMREAD_GRAYSCALE)
cell_nucleus = cv2.imread("cell_nucleus.tif", cv2.IMREAD_GRAYSCALE)

# 定义卷积计算函数
def convolve(image, kernel):
    return cv2.filter2D(image, -1, kernel)

# 定义 $\sigma$ 值和卷积核大小
sigmaList = [1, 2, 5]

# 计算水平和垂直方向的导数图像, 并显示
fig, axs = plt.subplots(nrows=len(sigmaList), ncols=6, figsize=(54, 27))

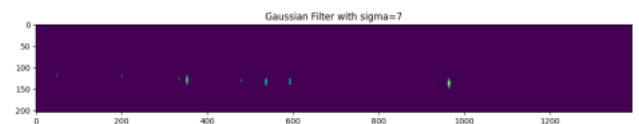
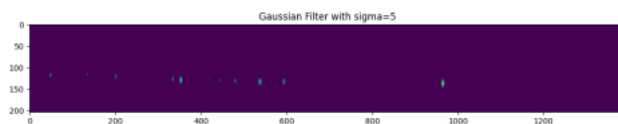
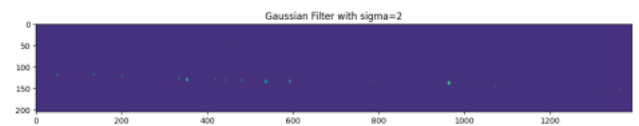
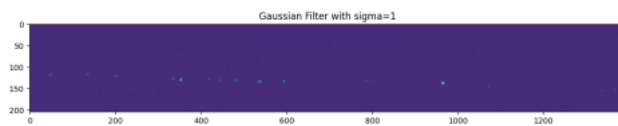
for i, sigma in enumerate(sigmaList):
    ksize = int(np.ceil(sigma * 3) * 2 + 1)
    k = cv2.getDerivKernels(1, 0, ksize)
    k1 = cv2.getDerivKernels(0, 1, ksize)
    kernelDx = cv2.getDerivKernels(1, 0, ksize)[0].T
    kernelDy = cv2.getDerivKernels(0, 1, ksize)[1]
    # 计算水平方向的导数
    axon02Dx = convolve(axon02, kernelDx)
    cellnucleusDx = convolve(cell_nucleus, kernelDx)
    # 计算竖直方向的导数
    axon02Dy = convolve(axon02, kernelDy)
    cellnucleusDy = convolve(cell_nucleus, kernelDy)

    # 绘制图像
    axs[i][0].imshow(axon02, cmap="gray")
    axs[i][0].set_title("axon02")
    axs[i][1].imshow(axon02Dx, cmap="gray")
    axs[i][1].set_title(f"dx_axon02 ( $\sigma={sigma}$ )")
    axs[i][2].imshow(axon02Dy, cmap="gray")
    axs[i][2].set_title(f"dy_axon02 ( $\sigma={sigma}$ )")
    axs[i][3].imshow(cell_nucleus, cmap="gray")
    axs[i][3].set_title("cell_nucleus")
    axs[i][4].imshow(cellnucleusDx, cmap="gray")
    axs[i][4].set_title(f"dx_cell_nucleus ( $\sigma={sigma}$ )")
    axs[i][5].imshow(cellnucleusDy, cmap="gray")
    axs[i][5].set_title(f"dy_cell_nucleus ( $\sigma={sigma}$ )")

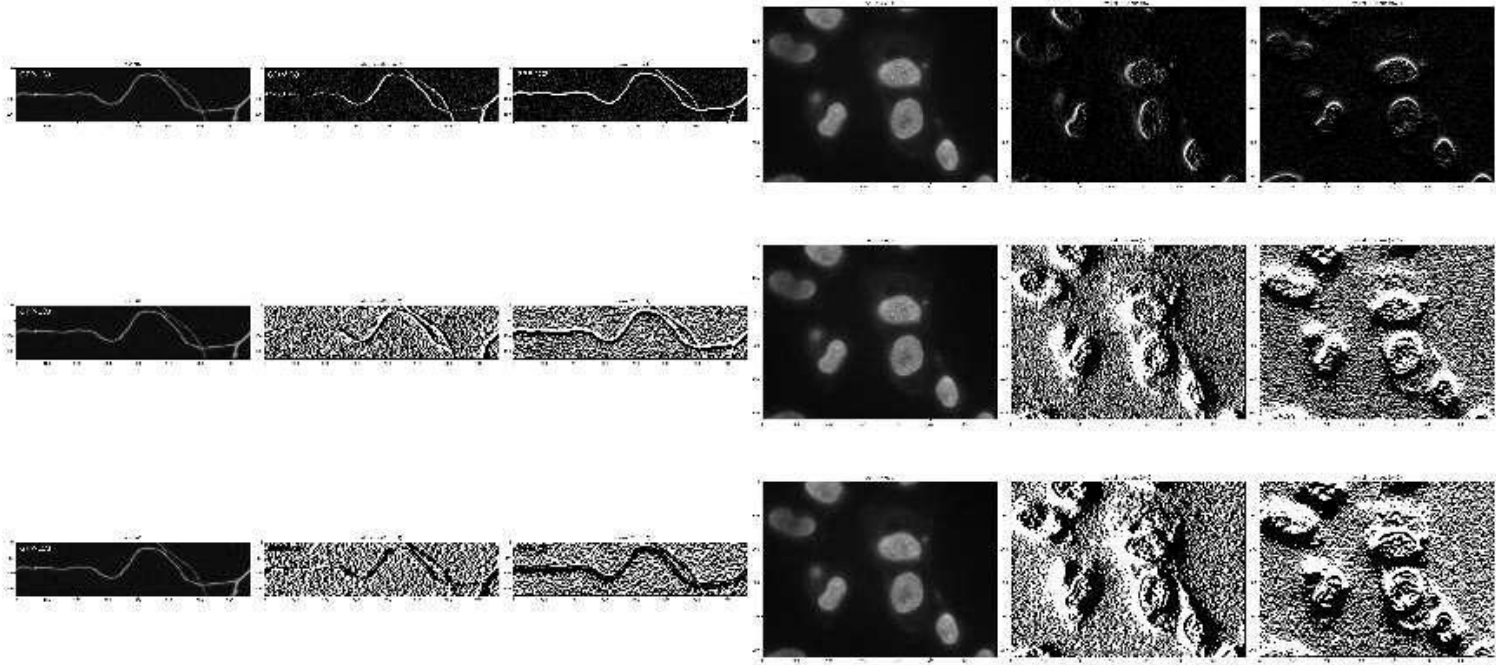
plt.tight_layout()
plt.show()
```

结果部分:

Result1:



Result2:



在指令1中学习了Python中函数的写法以及高斯滤波器的计算，先对图像进行读取，代入参数生成高斯滤波器，创建一个包含四个子图的 matplotlib 窗口，分别绘制图像。
在指令2中学到了如何用高斯核的一阶导数的卷积来计算水平和垂直方向上的导数显示衍生图像。

PART2: 计算和拟合 Airy Disk

使用 Python库scipy中自带的贝塞尔函数进行绘制，采用的是第一类贝塞尔函数，调用方式为:

```
import scipy.special as ss
J1 = ss.jv(1, x)
```

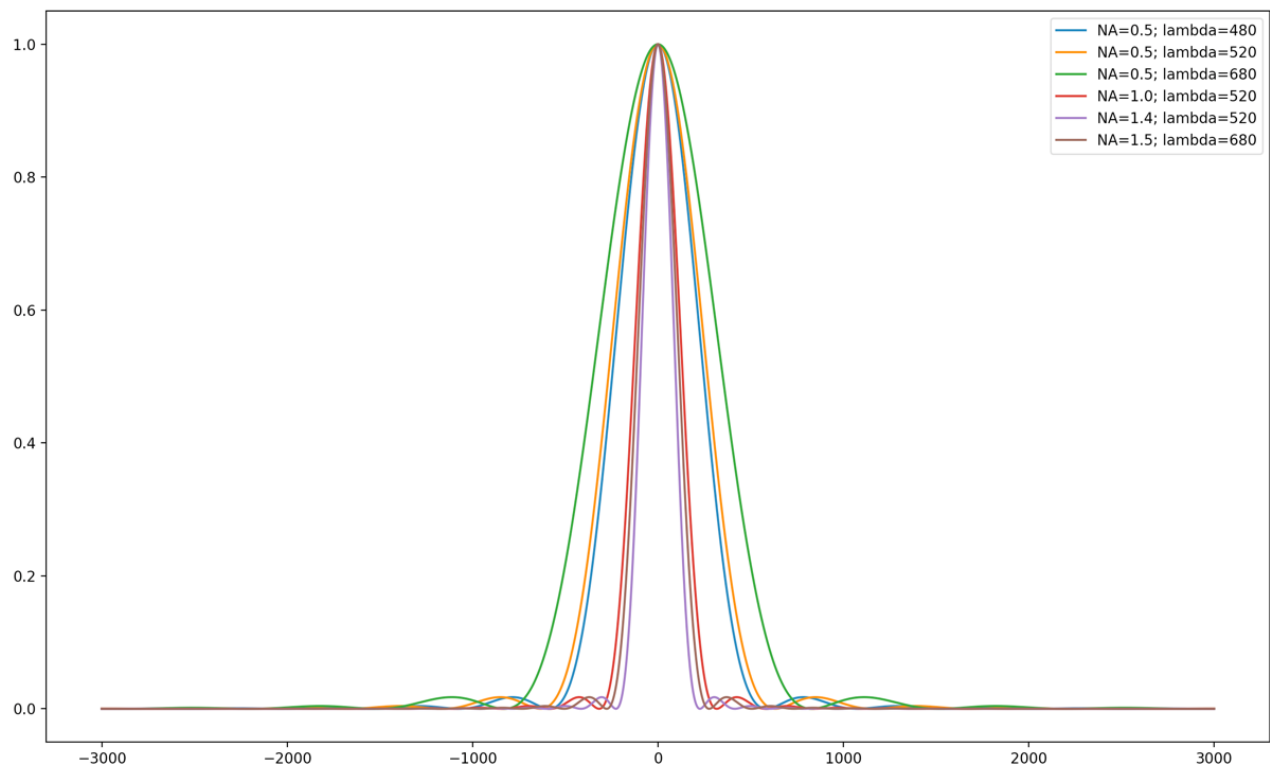
Airt Disk对应为理想显微镜透镜具有圆形的点扩散函数 $h(r)$ ，其形式为:

$$h(r) = \left(\frac{2J_1(ar)}{ar}\right)^2, with \ a = \frac{2\pi \cdot NA}{\lambda}$$

使用python，编写对应函数，得到各个测例的艾里斑半径如下所示:

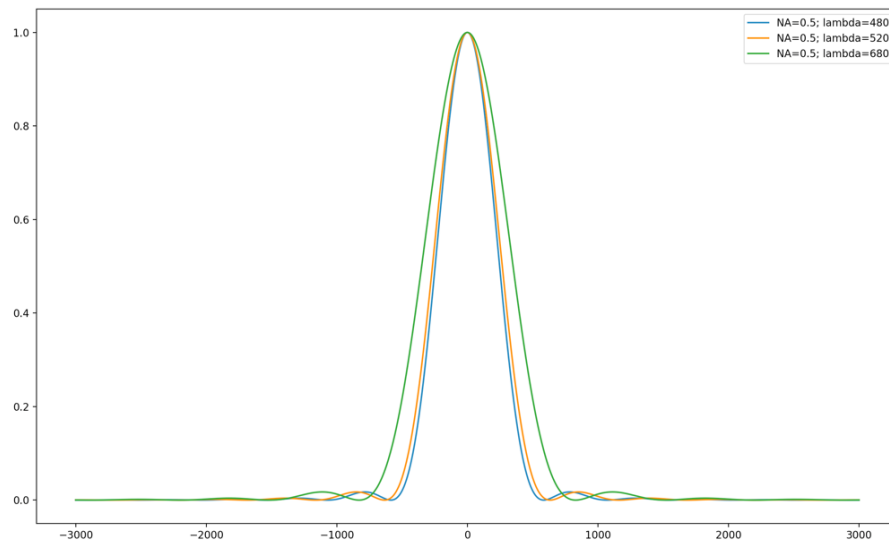
```
NA=0.5; lambda=480 , and the radius is : 585.6
NA=0.5; lambda=520 , and the radius is : 634.4
NA=0.5; lambda=680 , and the radius is : 829.6
NA=1.0; lambda=520 , and the radius is : 317.2
NA=1.4; lambda=520 , and the radius is : 226.57142857142858
NA=1.5; lambda=680 , and the radius is : 276.53333333333336
```

绘制对应图像如下图所示:



为了使得对比更加清晰，我们采取控制变量法，研究各个量对整体艾里斑的影响：

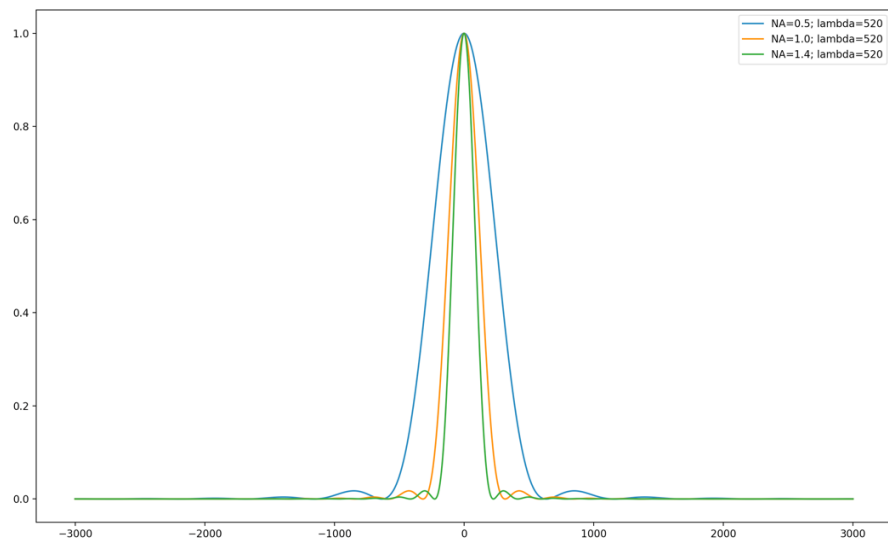
a) 固定 $NA = 0.5$ ，而 $\lambda = [480, 520, 680]$ ；



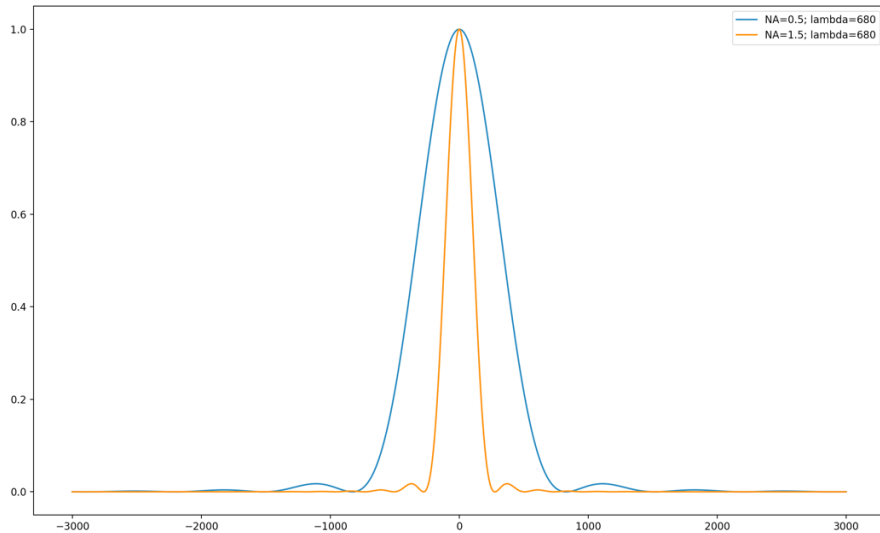
可以看出，随着 λ 的增加，艾里斑半径逐渐增大，图像逐渐变胖。

b) 固定 $\lambda = 520$ ，而 $NA = [0.5, 1.0, 1.4]$ ；

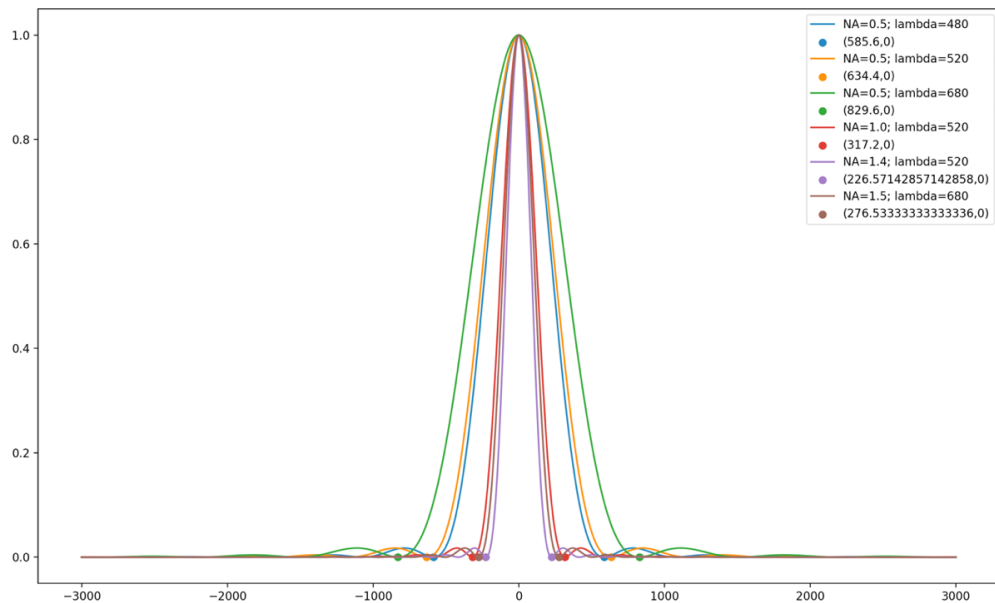
随着 NA 的逐渐增大，艾里斑半径逐渐减小，图像变的细长瘦高。



- c) 固定 $\lambda = 680$ ，而 $NA = [0.5, 1.5]$;
 随着 NA 增大，图像变得细长瘦高，艾里斑半径减小。在该图中非常明显。



因此可以看出，艾里斑半径与 λ 正相关，与 NA 负相关。绘制带有零点的图，可知零点符合半径公式：



2.2 使用 `astropy` 库进行一维高斯拟合，可以得到拟合曲线的均值和方差：

```
from astropy.modeling import models, fitting
g_init = models.Gaussian1D(amplitude=1., mean=0, stddev=1.)
fit_g = fitting.LevMarLSQFitter()
```

这样就建立了一个高斯拟合模型，之后带入艾里函数曲线上的点进行拟合即可：

```
g = fit_g(g_init, x, hr)
mean, sigma = g.mean.value, g.stddev.value
```

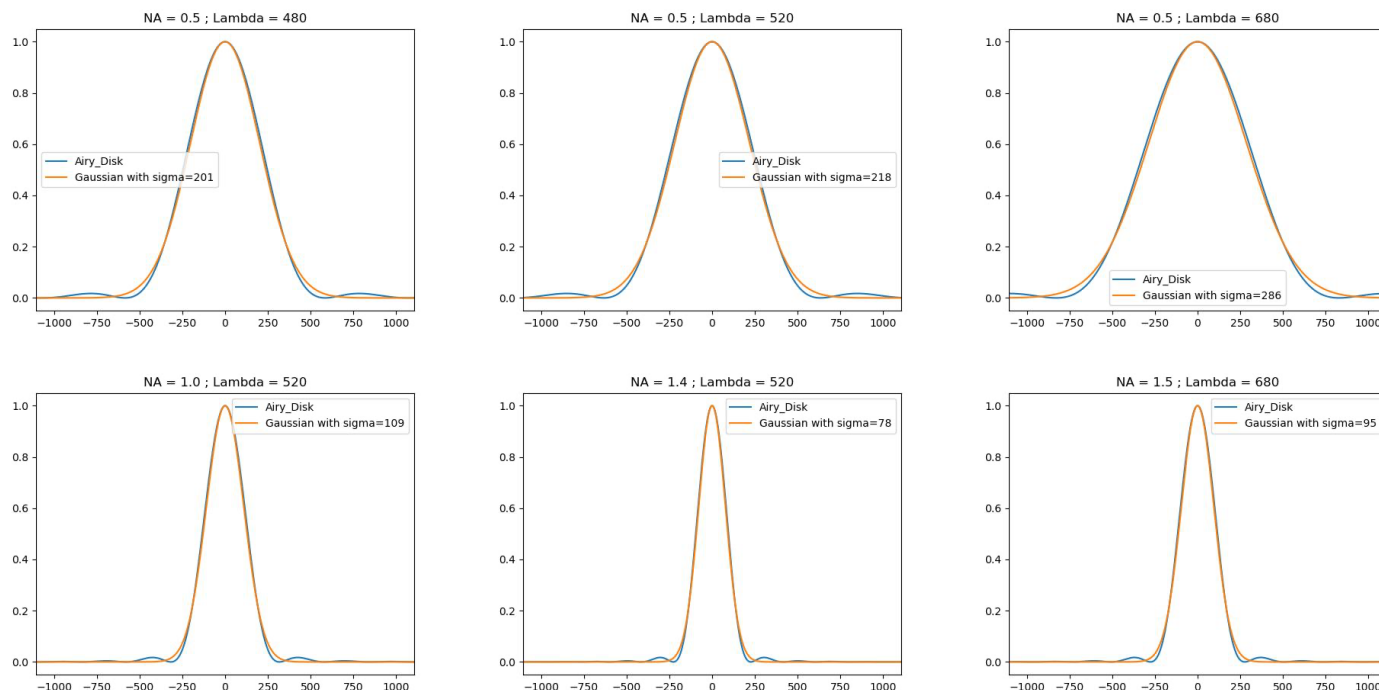
因此，对六个测例分别进行高斯拟合，得到各自的拟合均值、方差，以及方差与半径的比值：

```
===== NA: 0.5 Lambda: 480 =====
Gaussian: mean = -1.831861989995287e-13 sigma = 201.93388116423068
r / sigma = 2.89995911841925
===== NA: 0.5 Lambda: 520 =====
Gaussian: mean = -9.427724578142955e-13 sigma = 218.76169627266415
r / sigma = 2.8999592287366665
===== NA: 0.5 Lambda: 680 =====
Gaussian: mean = -2.761244367131191e-09 sigma = 286.0530891017808
r / sigma = 2.900160954754868
===== NA: 1.0 Lambda: 520 =====
Gaussian: mean = -5.261804060727754e-14 sigma = 109.3808500917339
r / sigma = 2.899959176894085
===== NA: 1.4 Lambda: 520 =====
Gaussian: mean = -9.349957243419902e-16 sigma = 78.12917039141149
r / sigma = 2.8999594829479323
===== NA: 1.5 Lambda: 680 =====
Gaussian: mean = -3.9743597670970306e-14 sigma = 95.35766865522852
r / sigma = 2.8999590408733305
```

从上述数值可以看出，拟合均值约为0，方差大小不一，但是艾里斑半径与方差的比值约为3，因此可以认为，使用高斯函数拟合艾里斑函数有规律：

$$\frac{r}{\sigma} = 3, \quad \text{with } r = 0.61 \cdot \frac{\lambda}{NA}$$

下图展示了各自拟合曲线与原艾里斑曲线的关系。



PART3: 在像素分辨率和亚像素分辨率上进行粒子检测

3.1 检测背景噪声

取图片左上角10*100尺寸的部分用于检测背景噪声，检测得到的结果为：

File name	mean	Std
001_a5_002_t001	338.080	25.392
001_a5_002_t002	335.321	27.071
001_a5_002_t003	335.233	25.907
001_a5_002_t004	337.252	25.535
001_a5_002_t005	337.339	25.693

3.2 检测局部最大和最小值

3.2.1 计算 σ

$$D = 0.61 * \frac{\lambda}{NA} = 0.61 * 515 / 1.4 \text{ nm} = 224.39 \text{ nm} = (224.39 / 65) \text{ pixels} = 3.45 \text{ pixels}$$
$$\sigma = D / 3 = 1.15$$

3.2.2 用 3×3 核检测局部最大和最小

3.2.3 比较 3×3 和 5×5 的区别

结果保存在 diff.mat 文件中，可以使用 `importdata('diff.mat')` 打开。

3.3/3.4 数据比较多，使用 Delaunay triangulation 最终结果输出在名为BIP_mat文件夹中，与原图片文件同名的mat文件。