Level 5 Laboratory: Computational Physics — Exercise 2

The deadline for this exercise is **Friday 2nd February 2018** at 12:30 p.m. Your report and program (*.py) files should be uploaded into Blackboard at the appropriate point in the Second Year Laboratory (PHY2DLM_2017) course.
S. Hanna

Objectives of the exercise

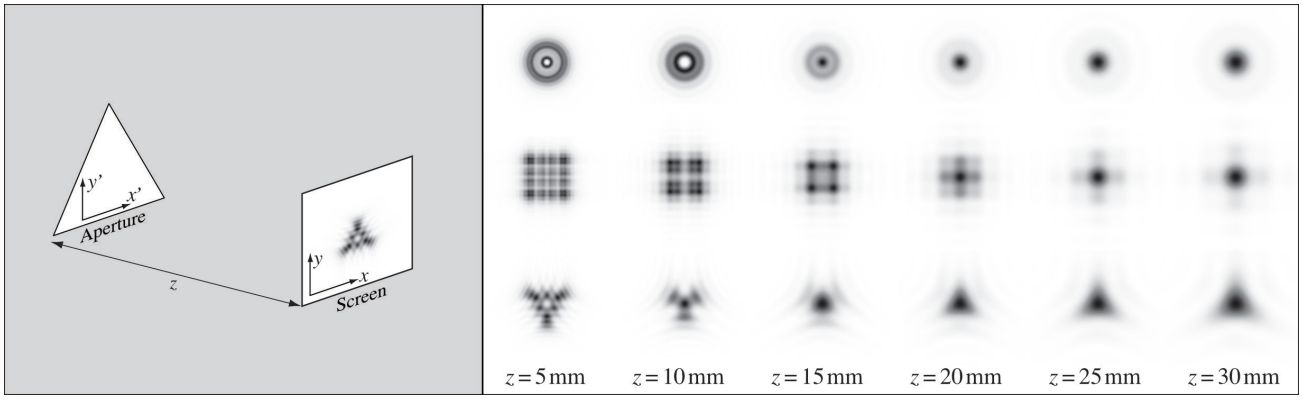
- To experiment with Simpson's rule for numerical integration.
- To gain experience of plotting data.

We expect that you will need help as you develop your programs. Please consult the demonstrators as frequently as you need to during the drop-in sessions. They are there to help.

Problem: Fresnel diffraction from an aperture – Simpson's rule (10 marks)

In the computing lectures, you have been briefed about some basic types of numerical integration. Here we will apply one of them, Simpson's rule, to an interesting diffraction problem.

The general set-up for the diffraction experiment is shown below. Incident waves, travelling parallel to the z-axis, are diffracted by an aperture, and the scattering pattern observed on a screen. In the Fresnel approximation, the screen is near compared with the size of the aperture, and the diffraction pattern varies strongly with the separation, z. Examples of this "near-field" diffraction are shown in the figure for circular, square and triangular apertures.



A general formula for Fresnel diffraction is given by:

$$E(x,y,z) = \frac{e^{ikz}}{i\lambda z}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} E(x',y')\exp\left\{\frac{ik}{2z}\left[(x-x')^2+(y-y')^2\right]\right\}\mathrm{d}x'\,\mathrm{d}y' \tag{1}$$

where the origin ($z = 0$) is taken at the aperture and $k = 2\pi/\lambda$. The integral is performed over the $x' - y'$ plane, but the electric field of the incident light is zero everywhere *except* in the aperture, where it is a constant, i.e.:

$$E(x',y') = \begin{cases} E_0 & x', y' \text{ in aperture} \\ 0 & \text{otherwise} \end{cases}$$

This is most simply achieved by choosing the limits of integration to fit the aperture and ignoring the constant phase factor of e^{ikz}/i:

$$E(x,y,z) = \frac{kE_0}{2\pi z}\iint_{Aperture}\exp\left\{\frac{ik}{2z}\left[(x-x')^2+(y-y')^2\right]\right\}\mathrm{d}x'\,\mathrm{d}y' \tag{2}$$

$E(x,y,z)$ is interpreted as the electric field of the diffracted light at coordinates (x,y) on a screen distance z from the aperture. The observed diffraction intensity will be proportional to $|E^2|$:

$$I(x,y,z) = \epsilon_0 c E(x,y,z)E^*(x,y,z) \tag{3}$$

where E^* is the complex conjugate of E, c is the speed of light and ϵ_0 is the permittivity of free space.

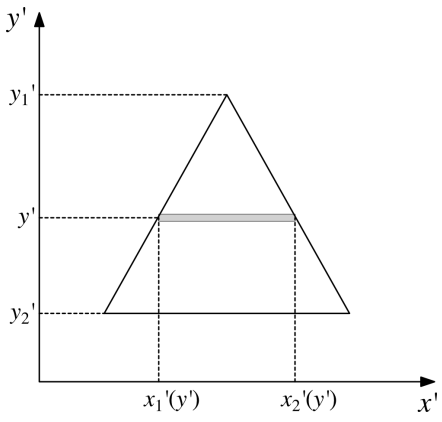
Solution of the integral in Eq. (2) can be achieved by separating the x' and y' integrals as follows, to allow the 1-dimensional Simpson's rule to be applied:

$$E(x, y, z) = \frac{kE_0}{2\pi z} \int_{y'_1}^{y'_2} X(x, y', z) \exp\left[\frac{ik}{2z}(y - y')^2\right] \mathrm{d}y' \qquad (4)$$

with:

$$X(x, y', z) = \int_{x'_1(y')}^{x'_2(y')} \exp\left[\frac{ik}{2z}(x - x')^2\right] \mathrm{d}x' \qquad (5)$$

This is equivalent to dividing the aperture into strips parallel to the x'-axis; the function $X(x, y', z)$ is value of the integral over one such strip at height y', between limits $x'_1(y')$ and $x'_2(y')$, as illustrated in the figure.



We will build up the solution to Eq. (2) in stages, first doing a 1-d integration, and then using that within a 2-d context. In carrying out the following stages, you should write a single Python program, based around the menu code used in Exercise 1. Write a separate piece of code for each of the sections, but re-use code and functions as much as possible, for efficiency.

1-d integration

a) Write a function to perform Simpson's rule integration. The function should take as arguments, the lower and upper limits of integration, the name of the function to be integrated and the number, N, of intervals for the integration. Remember, for N intervals there will be $N + 1$ points and, for the Simpson formula to work properly, you must have N even. Test your function by using it to integrate $\sin x$ between 0 and π, as in Lecture 3.

HINT: Passing a function to another function is allowed in Python, as shown in the following example:

```
import math
def myfunc (func, x):
    return func(x)**2
x = math.pi / 6.0
print ("x: ", x, "sin(x)**2: ", myfunc(math.sin,x), "cos(x)**2: ", myfunc(math.cos,x))
```

b) When you are happy that your Simpson's rule code is working, make a new version to evaluate Eq. 5 (you will need to pass in more parameters than in the previous part). Choose a fixed value of z, fixed limits, x'_1 and x'_2, and ignore y'. Plot a graph of $|X(x)|^2$ *vs.* x. Does the graph have the appearance you expect i.e. single slit diffraction? What happens when you vary N and when you vary z?

HINT: To generate the plot, you will need to create two arrays, one for the $|X(x)|^2$ values and the other for the x values. These can either be Python lists or Numpy arrays. This example demonstrates both methods:

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

NumPoints = 200
xmin = 0.0
xmax = np.pi
dx = (xmax - xmin) / (NumPoints - 1)
xvals = [0.0] * NumPoints
yvals = np.zeros(NumPoints)

for i in range(NumPoints):
    xvals[i] = xmin + i * dx
    yvals[i] = np.sin(xvals[i])

plt.plot(xvals,yvals)
plt.show() # May not be needed
```

2-d integration

c) Write another Simpson's rule function to integrate the Fresnel integral shown in Eq. 4. This will be another 1-d integration, but the function integrated will include the integral from part (b). To keep the problem simple, perform the integral over a square or rectangle, so that x'_1, x'_2, y'_1 and y'_2 are all constants.

To test your function, choose a fixed value of z for the distance to your screen, and calculate the diffraction intensity for a grid of (x, y) values on the screen, typically in the range $-1.0 \leq x, y \leq 1.0$ mm. Store these values in a 2-dimensional array and plot the array as an image; it should appear similar to those in the figure above. Here is an example of the use of a 2-d array to generate an image:

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

NumPoints = 200
delta = 4.0*np.pi / (NumPoints - 1)
intensity = np.zeros( (NumPoints,NumPoints) )

for i in range(NumPoints):
    x = i * delta
    for j in range(NumPoints):
        y = j * delta
        intensity[i,j] = np.sin(x) * np.cos(y)

plt.imshow(intensity)
plt.show()
```

Make sure you use sensible values for the wavelength of light ($0.4 \leq \lambda \leq 0.8\,\mu$m) and the aperture size ($0.001 \leq \rho \leq 0.25$ mm). The 2-d Simpson calculation can be very slow, so avoid taking N too large. $N = 50$ is a reasonable choice, but you may observe artefacts for small z or large apertures. Some experimentation will be needed. Repeat your calculation for different values of z and comment on any qualitative differences in your report.

d) Finally, adapt your code to allow for variable limits i.e. pass in functions for the limits $x'_1(y')$ and $x'_2(y')$, to allow you to calculate the diffraction from circular or triangular apertures, or any other simple shapes of your choice.

Report (10 marks)

As previously, you should prepare a concise report outlining your methods and highlighting your findings with suitable graphs or tables. Credit will be given for a reasoned discussion of your findings.

Submitting your work

You should submit the following to Blackboard:

- A concise report, in MS Word or pdf format;
- Your program for numerical integration using Simpson's rule;

Please note the following points:

- Please upload your "program.py" files.
- However, Turnitin won't accept a file ending ".py" so please rename your file with a ".txt" extension.
- Please also give your programs sensible distinguishing names, including your name or userid e.g. "my_userid_ex2.txt".

If you have any problems submitting your work, please contact Dr. Hanna (s.hanna@bristol.ac.uk) or ask a demonstrator.