DOING PHYSICS WITH PYTHON

COMPUTATIONAL OPTICS

RAYLEIGH-SOMMERFELD 1

DIFFRACTION INTEGRAL

BEAM PROPAGATION FROM AN APERTURE: MATHEMATICAL CONCEPTS

Ian Cooper

Please email me any corrections, comments, suggestions or additions: **matlabvisualphysics@gmail.com**

DOWNLOAD DIRECTORIES FOR PYTHON CODE

Google drive

GitHub

emRS01.py

RAYLEIGH-SOMMERFELD DIFFRACTION INTEGRAL OF THE FIRST KIND

The **Rayleigh-Sommerfeld region** includes the entire space to the right of the aperture. It is assumed that the Rayleigh-Sommerfeld diffraction integral of the first kind is valid throughout this space, right down to the aperture. There are no limitations on the maximum size of either the aperture or observation region, relative to the observation distance, because **no approximations have been made**.

The Rayleigh-Sommerfeld diffraction integral of the first kind (RS1) can be expressed as

(1) $$E_P = \frac{1}{2\pi}\iint_{S_A} E_Q \frac{e^{j k r_{PQ}}}{{r_{PQ}}^3} z_p\,(j k r_{PQ} - 1)\,\mathrm{d}S$$

where $E_P(x_P, y_P, z_P)$ is the electric field at the observation point $P(x_P, y_P, z_P)$, $E_Q(x_Q, y_Q, 0)$ is the electric field within the aperture and r_{PQ} is the distance from an aperture point Q to the observation point P. The double integral is over the area of the aperture S_A. The wavelength of the light and the propagation constant are

$$\lambda = \frac{2\pi}{k} \quad k = \frac{2\pi}{\lambda}$$

The irradiance I is proportional to the square of the magnitude of the electric field, hence the irradiance in the space beyond the aperture can be calculated by

(2) $I = I_0 \left| E^* E \right|$ I_0 is a normalizing constant

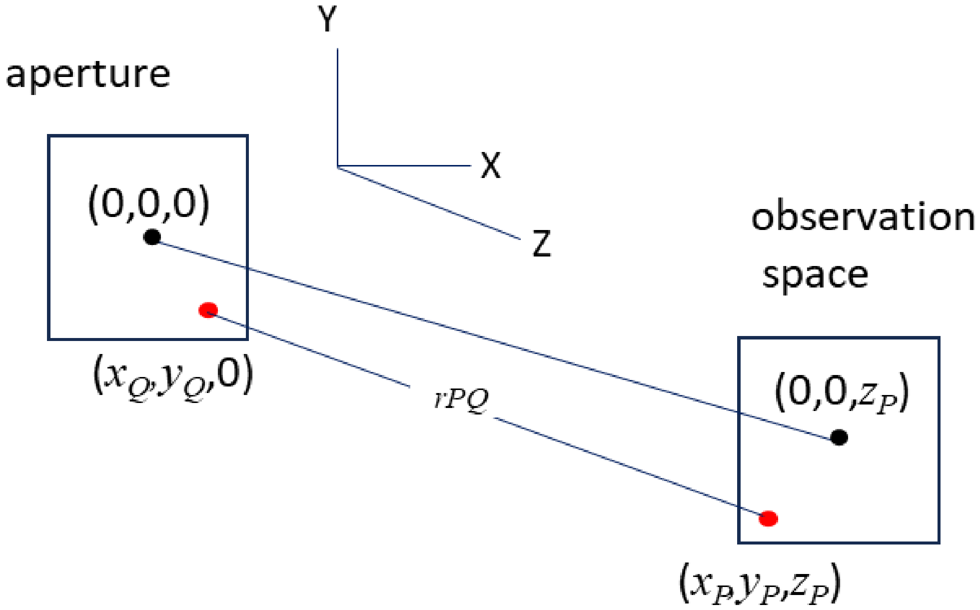


Fig. 1. Geometry of the aperture and observation spaces.

The electric field $E_P\left(x_P, y_P, z_P\right)$ at the point P$\left(x_P, y_P, z_P\right)$ can be computed by evaluating the double integral (equation 1) over the aperture space Q numerically using a two-dimensional form of Simpson's 1/3 rule as given by equation 3 when arbitrary units are used for the electric field and irradiance

(3)

$$E_P(x_P,y_P,z_P) = z_P \sum_{m=1}^{n_Q}\sum_{n=1}^{n_Q}\left(\left(\frac{e^{jk\,r_{PQmn}}}{r_{PQmn}^{\;3}}\right)\left(jk\,r_{PQmn}-1\right)\left(E_{Qmn}\,S_{mn}\right)\right)$$

where S_{mn} are the Simpson's two-dimensional coefficients. Each term in equation (2) can be expressed by a square matrix and the matrices can be manipulated very easily in Python to give the estimate of the integral as indicated in figure 2.

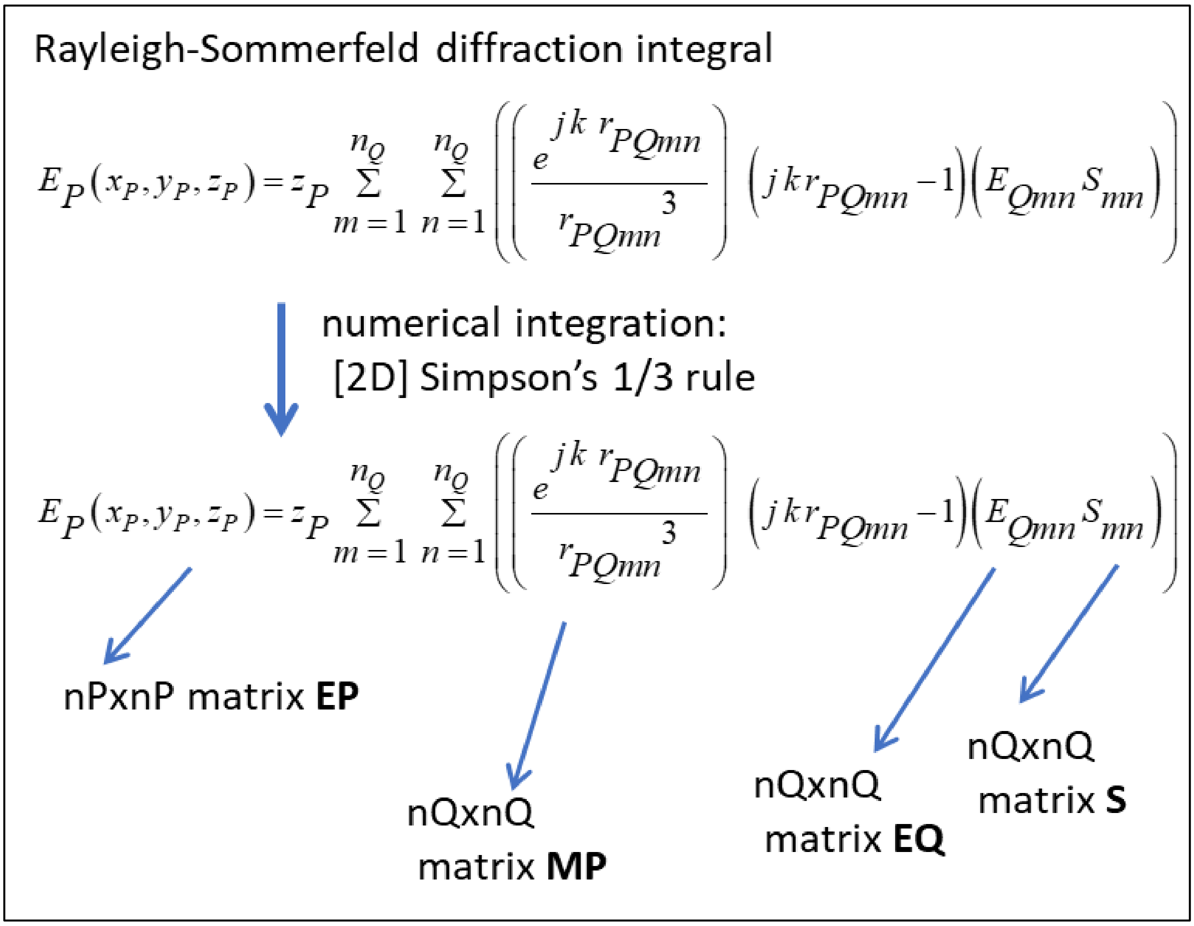


Fig. 2. Matrices used in computed the diffraction integral.

EXAMPLE emRS01.py

Diffraction from a rectangular aperture

wavelength = 6.328e-07 m

Aperature space

Grid point nQ = 159

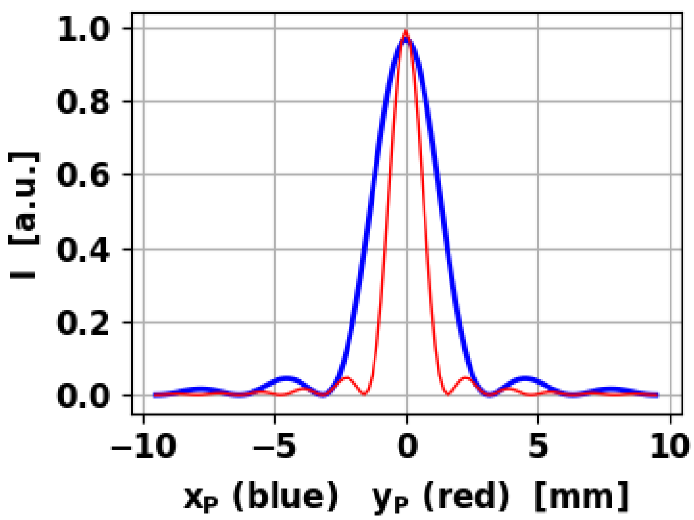
X width = 4.00e-04 m

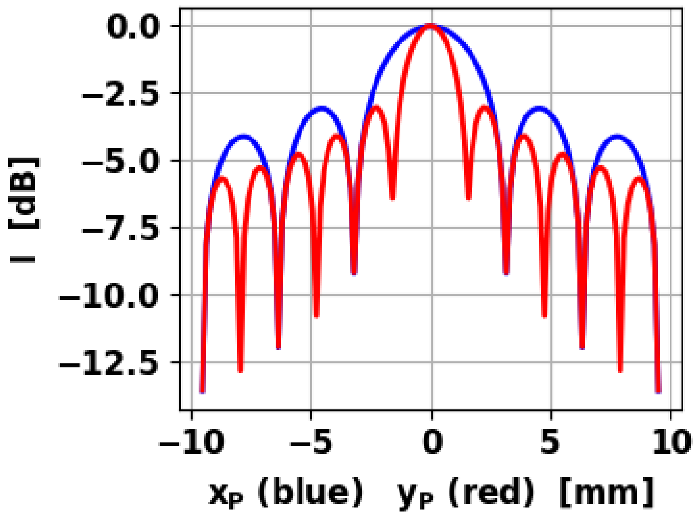
Y width = 8.00e-04 m

Observation

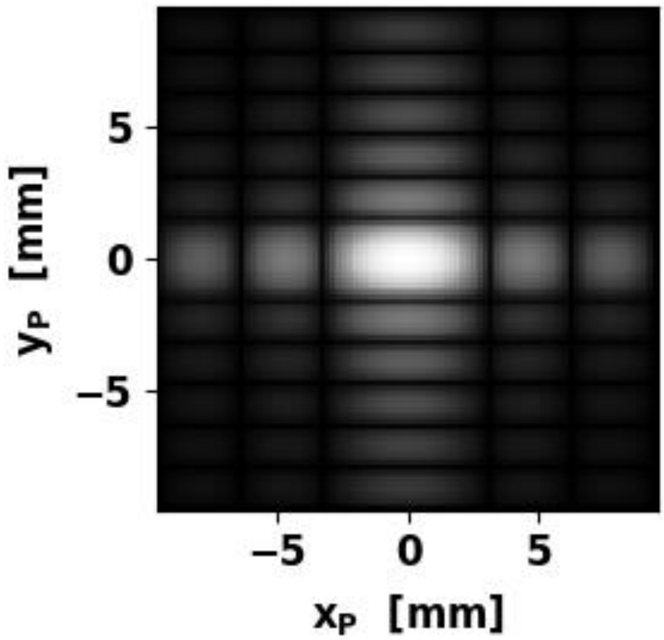
Grid point nP = 121

zP = 1.000e+00





0.0
−2.5
−5.0
−7.5
−10.0
−12.5
I [dB]
−10
−5
0
5
10
xP (blue) yP (red) [mm]



5
0
−5
yP [mm]
−5
0
5
xP [mm]

emRS01.py

```
# INPUT PARAMETERS
num = 30          # number for observation space
nP = num*4+1      # observation points for P:  format integer * 4 + 1
nQ = 159

 # aperture points for Q  must be ODD
wL = 632.8e-9     # wavelength [m]
# Aperautre space: full-width [m]
aQx = 2e-4; aQy = 4e-4

#Observation spacehalf: half-width % zP   [m]
xPmax = 10*1500*wL; yPmax = 10*1500*wL; zP = 1 #55000*wL

# SETUP
k = 2*pi/wL         # propagation constant
ik = k*1j           # jk

# Initialise matrices
unit = np.ones([nQ,nQ])   # unit matrix
rPQ = np.zeros([nQ,nQ]); rPQ3 = np.zeros([nQ,nQ])
MP1 = np.zeros([nQ,nQ]); MP2 = np.zeros([nQ,nQ]); kk =
np.zeros([nQ,nQ])
MP = np.zeros([nQ,nQ])
EQ = np.ones([nQ,nQ])
```

```
# Aperture space
xQmin = -aQx/2;  xQmax = aQx/2
yQmin = -aQy/2;  yQmax = aQy/2
xQ1 = linspace(xQmin,xQmax,nQ)
yQ1 = linspace(yQmin,yQmax,nQ)
xQ, yQ = np.meshgrid(xQ1,yQ1)

RQ = (xQ**2 + yQ**2)**0.5
#EQ[RQ > xQmax] = 0

# Observation space
xPmin = -xPmax; yPmin =  -yPmax
EP = np.zeros([nP,nP])+np.zeros([nP,nP])*1j

xP1 = linspace(xPmin,xPmax,nP)
yP1 = linspace(yPmin,yPmax,nP)
xP, yP = np.meshgrid(xP1,yP1)

# Simpson [2D] coefficients
S = np.ones(nQ)
R = np.arange(1,nQ,2);   S[R] = 4;
R = np.arange(2,nQ-1,2); S[R] = 2
scx, scy = np.meshgrid(S,S)
S = scx*scy
```

```
# % COMPUTATION OF DIFFRACTION INTEGRAL FOR ELECTRIC
FIELD % IRRADIANCE-

for c1 in range(nP):
    for c2 in range(nP):
        rPQ = np.sqrt((xP[c1,c2] - xQ)**2 + (yP[c1,c2] - yQ)**2 + zP**2)
        rPQ3 = rPQ*rPQ*rPQ
        kk = ik * rPQ
        MP1 = exp(kk)
        MP1 = MP1 / rPQ3
        MP2 = zP * (ik * rPQ - unit)
        MP = MP1 * MP2
        EP[c1,c2] = sum(sum(EQ*MP*S))

Irr = np.real(EP*np.conj(EP))
Irr = Irr/amax(amax(Irr))
indexXY = num*2+1
IY = Irr[:,indexXY]
IX = Irr[indexXY,:]

# Position of first zero in irradiance
x0 = wL*zP/(1*aQx)*1e3    # [mm]
y0 = wL*zP/(1*aQy)*1e3    # [m]
```