Laser beam Fourier Propagation

David Schmidt

April 27, 2018

Functions

```python
def circ_aperture(x,y,A_dia,In):
    boolarray=(x**2+y**2)<A_dia**2
    return boolarray*In

def wGaus(z,zR,w0):
    return(w0*np.sqrt(1+z/zR)**2)

def plot2DInt(axin,Iin,Lin,title):
    axin.set_title(title)
    Inorm=Iin/Iin.max()
    plt.imshow(Inorm,origin='lower', interpolation='none', extent=[-Lin/2,Lin/2,-Lin/2,Lin/2],cmap='gray')
    axin.set_aspect('equal')

def plot1DInt(axin,xin,Iin,Min,title):
    axin.set_title(title)
    plt.plot(xin,Iin[int(Min/2),:]/max(Iin[int(Min/2),:]))
    axin.grid(color='black', linestyle='-', linewidth=.1)
```

Propagation Functions - TF

```python
def propTF(Ein,L,lam,z):
    """
    propagation - transfer function approach
    assumes same x and y side lengths and uniform sampling
    Ein - source plane field
    L - source and observation plane side length
    lam - wavelength
    z - propagation distance
    u2 - observation plane
    """
    (M,N)=np.shape(Ein)
    dx=L/M

    fx=np.arange(-1/(2*dx),1/(2*dx),1/L)
    FX, FY = np.meshgrid(fx, fx, sparse=True)

    H=np.exp(-1j*np.pi*lam*z*(FX**2+FY**2))
    H=nf.fftshift(H)
    U1=nf.fft2(nf.fftshift(Ein))
    U2=H*U1
    Eout=nf.ifftshift(nf.ifft2(U2))
    return(Eout)
```

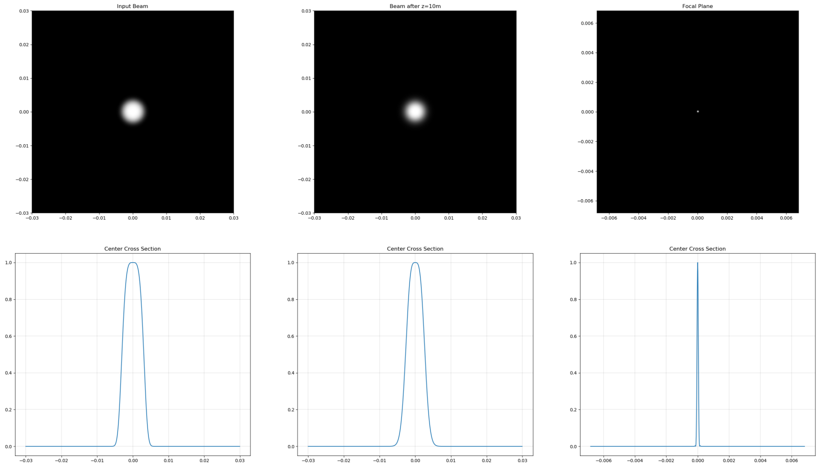
Propagation Functions - FF

```
def propFF(Ein,L,lam,z):
    """
    propagation - transfer function approach
    assumes same x and y side lengths and uniform sampling
    Ein - source plane field
    L - source plane side length
    lam - wavelength
    z - propagation distance
    Eout - observation plane
    L2 - observation plane side length
    """
    (M,N)=np.shape(Ein)
    dx=L/M
    k=2*np.pi/lam

    L2=lam*z/dx
    dx2=lam*z/L

    x2=np.arange(-L2/2,L2/2,dx2)
    X2, Y2 = np.meshgrid(x2, x2, sparse=True)

    c=1/(1j*lam*z)*np.exp(1j*k/(2*z)*(X2**2+Y2**2))
    Eout=c*nf.ifftshift(nf.fft2(nf.fftshift(Ein)))*dx**2
    return([Eout,L2,x2])
```

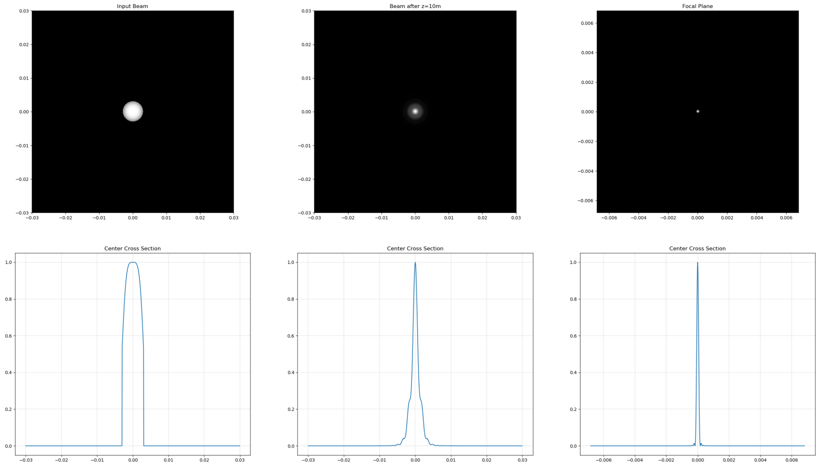
z-scan function

```
def ScanTF2D(Ein,L,lam,zf,dz,M):
    """
    Uses TF propagation to get a scan from z=0 to z=zf in steps of dz and returns the typical 2D plot of x and z
    """
    z=np.arange(0,zf+dz,dz)
    array=np.zeros((len(z),M))
    for ii in range(len(z)):
        ufull=propTF(Ein,L,lam,z[ii])
        Ifull=abs(ufull)**2
        Icut=Ifull[int(M/2),:]
        array[ii]=Icut
    f=plt.figure()
    axf = f.add_subplot(111)
    axf.set_title("Cross-section propagation along z. (m=2 with clipping)")
    Inorm=array/array.max()
    plt.imshow(Inorm,origin='lower')# interpolation='none', extent=[-Lin/2,Lin/2,-Lin/2,Lin/2],cmap='gray')
    axf.set_aspect('equal')
```

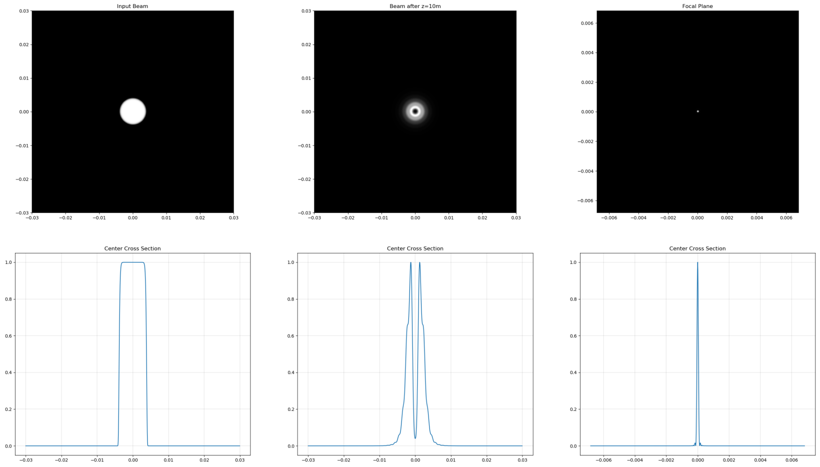
Examples


Input Beam
Beam after z=10m
Focal Plane
Center Cross Section
Center Cross Section
Center Cross Section

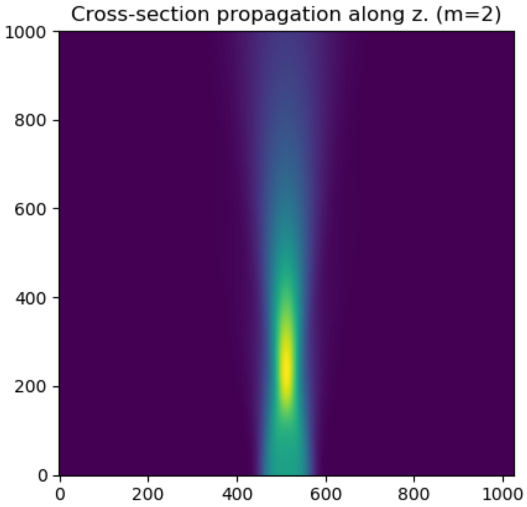
Examples


Input Beam
Beam after z=10m
Focal Plane
Center Cross Section
Center Cross Section
Center Cross Section

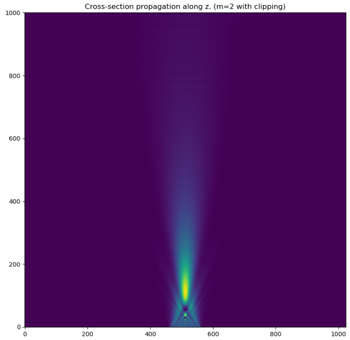
Examples

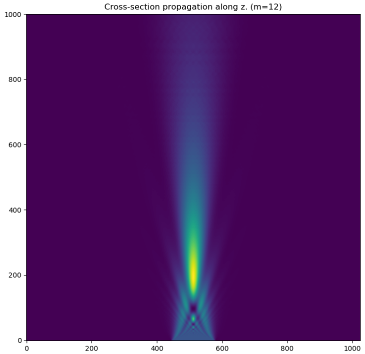

Input Beam
Beam after z=10m
Focal Plane
Center Cross Section
Center Cross Section
Center Cross Section

Examples


Cross-section propagation along z. (m=2)
1000
800
600
400
200
0
0
200
400
600
800
1000

Examples


Cross-section propagation along z. (m=2 with clipping)
0
200
400
600
800
1000


Cross-section propagation along z. (m=12)
0
200
400
600
800
1000