

0.0.1 Gaussian beam solution propagating at large angles

Purpose: The purpose of this exercise is twofold. First, it illustrates how the FFT-BPM formula derived in the class can be used to obtain the well-know Gaussian beam solution. Second, and more importantly, we obtain an implementation of an exact solution which will be utilized for testing our BPM algorithms not only in this section but also in a number of other situations discussed later in the course.

Task: The importance of testing in numerical simulation can not be overstated. To test implementations of BPM methods throughout this course, we will use exactly known solutions whenever possible. Since this Section deals with the propagation in homogeneous media, exact paraxial Gaussian beam solutions are suitable for these testing purposes.

A) Starting from the general formula developed in the lecture notes, derive an analytic expression for the propagated Gaussian beam in one transverse dimension. Specify the initial beam by its waist, wavelength, and an angle that defines the direction of propagation of the beam. Since later in the course we will need beams that propagate off-axis at an angle which large, i.e. not paraxial, it will be useful to have an initial condition function that is valid for arbitrary angle between the beam axis and the BPM propagation axis.

B) Write a function to implement the beam formula derived.

C) Generalize the result to two transverse dimension.

Solution:

A) First, let us derive the solution for an initially collimated Gaussian beam characterized by its waist w . We choose the frame of reference such that the beam waist occurs at the origin, and the propagation is along the positive z -axis. Later we will transform the result into the coordinate system of the computational domain.

The beam profile at $z = 0$ is a simple Gaussian function characterized by w :

$$\exp[-x^2/w^2] .$$

The first step is to obtain its Fourier transform:

$$\int_{-\infty}^{+\infty} dx \exp[-ikx] \exp[-x^2/w^2] . \quad (1)$$

This can be evaluated using the Gaussian-integral formula

$$\int_{-\infty}^{+\infty} dx \exp[-ax^2 + bx] = \sqrt{\frac{\pi}{a}} e^{\frac{b^2}{4a}} . \quad (2)$$

In this case we have $a = 1/w^2$ and $b = -ik$, and obtain

$$\int_{-\infty}^{+\infty} dx \exp[-ikx] \exp[-x^2/w^2] = \sqrt{\pi w} \exp[-k^2 w^2/4] \quad (3)$$

The next FFT-BPM step is the application of the *paraxial* propagator,

$$\sqrt{\pi w} \exp[-k^2 w^2/4] \exp[-izk^2/(2\beta)] . \quad (4)$$

The second Fourier transform takes the solution back to real space:

$$(2\pi)^{-1} \int_{-\infty}^{+\infty} dk \exp[+ikx] \sqrt{\pi w} \exp[-k^2 w^2/4] \exp[-izk^2/(2\beta)] . \quad (5)$$

Here one collects like terms in the argument of the exponential in order to identify parameters a, b for yet another application of the above Gaussian-integral formula, and the result becomes:

$$\frac{1}{\sqrt{1 + \frac{2iz}{\beta w^2}}} \exp \left[-\frac{x^2}{w^2(1 + \frac{2iz}{\beta w^2})} \right] \quad (6)$$

This is a paraxial solution with its waist at $z = 0$. However, this in general will not be suitable for initial conditions in our BPM simulations. Should we need an initial condition such that it would focus at a distance d from where we initiate the BPM integration, we simply replace $z \rightarrow z - d$.

The solution we have is written in terms of the complex beam parameter with the minor difference (from the standard formulas) that we have imposed the normalization to unity at $z = 0$ and $x = 0$.

As usual, the dependence on the propagation coordinate is only through its ratio to the characteristic length-scale of the beam which is the Rayleigh range

$$\frac{2z}{\beta w^2} = \frac{z}{z_R} \quad z_R = \frac{\pi w^2}{\lambda} = \frac{\beta w^2}{2} .$$

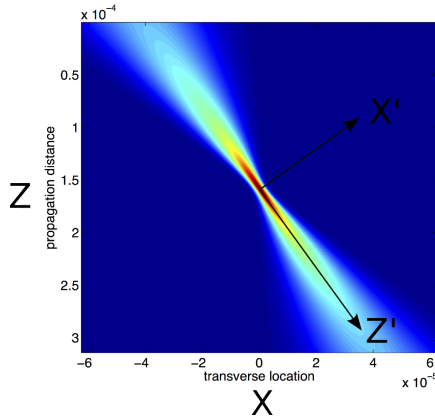
A Gaussian beam propagating at an angle, “placed” in the computational domain can be obtained by transforming the above result into a rotated coordinate system:

$$z' = z \cos \alpha + x \sin \alpha \quad , \quad x' = -z \sin \alpha + x \cos \alpha . \quad (7)$$

However, one must remember that the whole solution also contains the carrier wave, so the transformed (rotated) solution is

$$\frac{1}{\sqrt{1 + \frac{2iz'}{\beta w^2}}} \exp \left[-\frac{x'^2}{w^2(1 + \frac{2iz'}{\beta w^2})} \right] e^{+i\beta z'} \quad (8)$$

B) This function is implemented in the instructor’s solution *GaussianBeam1DRotated.m*. The coordinate system used in this functions are illustrated in the following figure.



Gaussian beam coordinate systems utilized in the derivation of the solution. The intensity of a beam propagating at an angle is shown in the coordinate system (x, z) of the computational domain. The primed coordinates $(x'$ and $z')$ span the frame of reference which originates in the waist point of the beam and is oriented along its propagation direction.

A note is in order concerning the way beams propagating at an angle are realized in some beam propagation simulations. If the angle of propagation is not too large, the primed variables can be approximated by the original ones, except the z' that appears in the carrier-wave exponential (why?). The first-order approximation results in a phase-factor

$$e^{+i\beta z'} \approx \exp[i\beta z] \exp[i\beta \alpha x] \approx \exp[i\beta z] \exp[ik_{\perp} x] . \quad (9)$$

Thus, adding a phase-front tilt to the normal Gaussian will make it propagate at an angle. It is important to remember that this approximation is only good for small angles.

C) The generalization for two transverse dimensions is based on the fact that for any initial condition in the form a direct product

$$F(x)G(y) ,$$

the *paraxial* beam propagation formula factorizes into two one-dimensional contributions from each x and y . This is rather straightforward, and left to the Reader to show that if the beam waist is the same in both transverse direction, the solution becomes

$$\frac{1}{1 + \frac{2iz'}{\beta w^2}} \exp \left[-\frac{x'^2 + y'^2}{w^2(1 + \frac{2iz'}{\beta w^2})} \right] e^{+i\beta z'} . \quad (10)$$

0.0.2 Implementation of FFT-BPM, paraxial and non-paraxial regimes

Task 1: Implement a one-dimensional beam propagation method based on expansion into plane waves, using FFT in the paraxial approximation. Set up the simulation such that it starts from the initial condition given by the function in the previous exercise. Let the algorithm propagate the solution through multiple propagation steps, and compare the resulting numerical solution with the exact one, given again by the same formula as the initial condition, this time evaluated for the appropriate propagation distance. Near-perfect agreement should be obtained for propagation along axis (zero beam angle).

Solution:

The implementation of the Fourier transform based beam propagation method is rather straightforward. There are three central portions of the program:

- Coordinate system. It is practical to precalculate coordinate arrays for both real-space and spectral-space representation. The later means to calculate the grid of spatial wavenumbers, for example as in the following listing:

Listing 1: 1D-Maxwell integration step implementation

```
1 % LX stands for the computational domain dimension
2 LX = ...
3
4 % dk represents the ‘‘wave-number quantum’’
5 dk = 2*pi/LX
6
7 % NX is the number of grid points in the spatial domain
8 NX = ...
```

```

9
10 % define transverse wavenumbers = spectral coordinates
11 kx = zeros(1,NX);
12 for k=0:NX/2
13     kx(1+k) = dk*k;
14 end
15 for k=NX/2+1:NX-1
16     kx(1+k) = dk*(k - NX);
17 end

```

This method honors the layout of frequencies normally used in implementations of FFTs. The first half of the arrays contains positive spatial frequencies, while the second half carries their negative counterparts.

- Spectral propagator. Having calculated the array of spatial wavenumbers, the propagator can be precalculated for the given integration step size dz :

Listing 2: 1D-Maxwell integration step implementation

```

1 % k0 stands for the reference wavenumber
2 k0 = 2.0*pi/lambda
3
4 % two versions of the propagator array below:
5
6 % propagator: paraxial version
7 px = exp(-1i*(kx.*kx)/(2*k0)*dz +1i*k0*dz);
8
9 % nonparaxial propagator
10 px = exp(+1i*dz*( sqrt(k0*k0 - kx.*kx) ) ) );

```

Note that this implementation preserves the absolute phase of the solution as it includes also the phase change of the carrier wave. This is not strictly necessary, but needed if we want to compare not only the intensity but also the real part of the complex amplitude (which represents the electric field) between the exact and numerical solutions.

- Propagation step. Here we show the example in Matlab, where we can define a “one-liner” function to perform the whole update which returns the array holding the new solution amplitude:

Listing 3: 1D-Maxwell integration step implementation

```

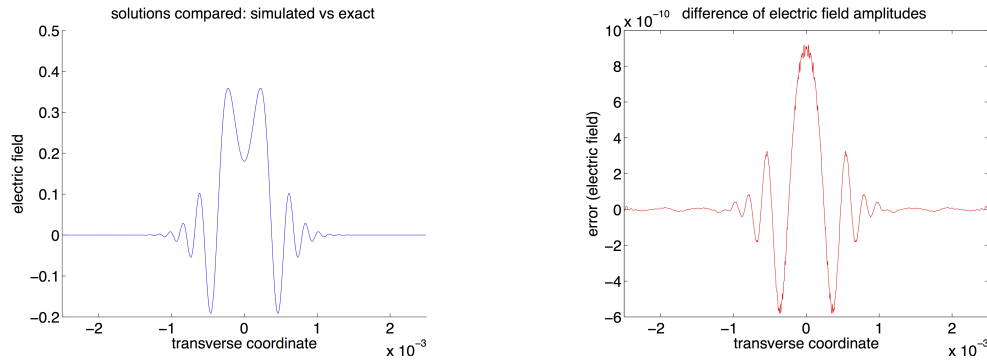
1 % define one linear step
2 LinearStep = @(amplitudein , propagator) ifft (propagator.*fft (amplitudein));

```

Note that in this case the forward and inverse transforms are properly normalized. With some other FFT implementation, the user must remember to ensure that the norm of the solutions remains preserved.

The rest of the simulation code is not specific to the method discussed, and implements the test based on comparison with the exactly known solution. To make this test non-trivial, the spatial profile

of the beam must undergo substantial changes. One possibility is to start with a focused beam and let it evolve through its focal point. This is illustrated in the instructor's solution stored in the file named *pParaxialTest.m*.



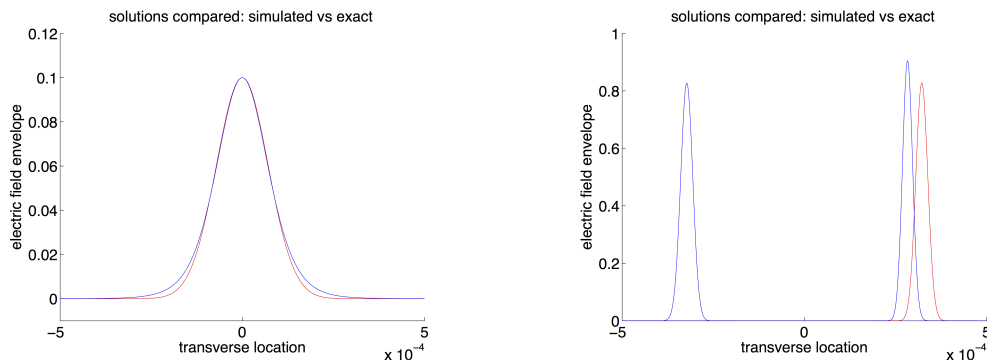
Test of 1D FFT-BPM implementation: Propagation of a Gaussian beam ($\lambda = 800$ nm) beyond its focus in which it attains the waist of 100 micron. The total propagation length was ten Rayleigh ranges. The whole simulation was executed in fifty steps. The left panel shows comparison of the numerical and analytic solution, with two curves that are practically indistinguishable. The panel on the right shows the deviation between the simulated and analytic electric field amplitudes. This error increases with the number of integration steps taken, as the error accumulate. However, as expected the gap between the analytic and simulated results is very small.

Task 2: Implement in your script also the possibility to use the exact, non-paraxial propagator. Execute the simulation, and observe the deviations between the exact paraxial and numerical solution. Try to find a regime in which these deviations manifest as clearly as possible.

Solution:

a) One obvious possibility to induce non-paraxial effects is a tightly focused beam. Instructor's solution *pNonparaxialWaist.m* illustrates such a regime. As the name suggests, the divergence of the beam is controlled by choosing its waist to be of the order of the wavelength. The reader should experiment with the simulation parameters in order to verify that unless the beam approaches the wavelength scale, the difference between paraxial and exact solution is insignificant.

b) A more stringent test of the FFT-BPM program in the non-paraxial regime is made possible simulating a beam propagating at a steep angle with respect to the axis of the computational domain. This arrangement is also more relevant to situations one often encounters in practical beam-propagation simulations. Instructor's solution *pNonparaxialAngle.m* contains an example.



Left: Non-paraxial effects in a tightly focused ($w = 1$ micron) Gaussian beam ($\lambda = 800$ nm). The non-paraxial solution (blue) exhibits slightly wider non-Gaussian beam pedestal.

Right: Non-paraxial effects in the Gaussian beam propagating at an angle of 0.35 radians (≈ 20 degrees) over a distance of 40 Rayleigh ranges. Focus with $w = 15$ micron attained roughly in the middle of the simulated propagation distance. The initial beams is the peak in the left-hand portion of the figure, the propagated beam profiles are on the right. The paraxial solution, shown in blue, propagates at a shallower angle. Obviously, propagation at an angle is more sensitive to non-paraxiality.

Puzzler: Script *pPuzzler.m* is intended to simulate at-an-angle propagation. It uses the same method as the one tested previously, yet a simulation run reveals a clearly incorrect behavior. The beam seems to propagate in a wrong direction, which may be a bit unexpected for what is an “exact” algorithm that passed a test. It is the reader’s task to identify a remedy to this problem, and explain its origin.