Homework 1 solution from group 4

1 Introduction

The following task is about solving the Beam propagation method for a given Gaussian beam profile at the input of a step indexed waveguide and see its propagation along the propagation distance within the waveguide.

This is done by numerical solving the paraxial wave equation with the help of the explicit implicit Crank-Nicholson method. Also, the convergence of this algorithm with the resolution of the step size and computation space of grid size is investigated.

2 Theory

The problem is modelled as follow, the propagation of the beam along the dimension of interest which here is the z-axis is distributed into two parts. One carries the fast oscillations which anyways isn't that important while analysing the power distribution in space as the fast propagating parts carry no information in this regard. Using these equations simplifies the generic wave question into a much more computationally simple one called the paraxial wave equation. This equation also suits our physical system, and we are precisely looking at the propagation of the beam within a narrow divergence regime at the centre of the waveguide, hence the light propagated almost parallel to z-axis.

Hence a general field profile can be represented as :

$$\Phi(x, y, z) = \phi(x, y, z) \exp(-ikn_0 z)$$

where $\Phi(x, y, z)$ represents the slowly varying part. Now a refractive index can be chosen to vary along z-axis as well but weakly to stimulate large index contrast, which here we represent through the step index profile.

Switching further within the scalar regime and operating under the SVEA one can then reduce the Helmholtz's equation into the scalar paraxial wage equation, which in 1D reads as follows

$$\left[i\frac{\partial}{\partial z} + \frac{1}{2\bar{k}}\frac{\partial^2}{\partial x^2} + \frac{k^2(x) - \bar{k}^2}{2\bar{k}}\right] v(x,z) = 0$$

where k_ represents the average refractive index seen along z-axis (This encapsulates the negligible changes of n along z-axis).

Rearranging this equation leads to:

$$\frac{\partial}{\partial z} v(x,z) = \left[\frac{i}{2\bar{k}}\frac{\partial^2}{\partial x^2} + i\frac{k^2(x) - \bar{k}^2}{2\bar{k}}\right] v(x,z) = Lv(x,z)$$

Hence, we have $Lv(x,z) = \frac{i}{2\bar{k}}\frac{\partial^2 v(x,z)}{\partial x^2} + iW(x)v(x,z)$ where $W(x) = \frac{k^2(x)-\bar{k}^2}{2\bar{k}}$ After employing the finite difference method, as seen in the previous task, one can turn the Linear operator L^ into a Matrix representation. This leads to:

$$\mathbf{L} = \frac{i}{2\bar{k}(\Delta x)^2}\begin{pmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & 1 & -2 & 1 \\ 0 & \cdots & 0 & 1 & -2 \end{pmatrix} + i\begin{pmatrix} W_1 & 0 & \cdots & & 0 \\ 0 & \ddots & & & \vdots \\ \vdots & & W_j & & \\ & & & \ddots & 0 \\ 0 & \cdots & & 0 & W_N \end{pmatrix}$$

$$W_j = \frac{k^2\left(x_j\right) - \bar{k}^2}{2\bar{k}}$$

This takes care of the R.H.S. Whereas the L.H.S can be evaluated by using any of the following algorithms: 1)The Forward Difference Method:

$$\frac{\partial}{\partial z} v_j(z)\bigg|_{z^n} \approx \frac{v_j^{n+1} - v_j^n}{\Delta z} \approx \sum_l L_{jl} v_l^n$$

2)The Backward Difference Method

$$\frac{\partial}{\partial z}\bar{v_j}(z)\bigg|_{z^{n+1}} \approx \frac{v_j^{n+1} - v_j^n}{\Delta z} \approx \sum_l L_{jl} v_l^{n+1}$$

3)The central Difference Method

$$\frac{\partial}{\partial z} \nu_j(z)\bigg|_{z^{n+1/2}} \approx \frac{\nu_j^{n+1} - \nu_j^n}{\Delta z} \approx \frac{1}{2}\sum_l \left(L_{jl}\nu_l^n + L_{jl}\nu_l^{n+1}\right)$$

But upon analysing the stability condition one finds that the while forward method is unstable, the backward difference though being stable does not comply with energy conservation. Which leaves us with the central method being the optimal one.

The Crank Nicholson method: After following the central difference method for the L.H.S and after rearranging one reaches with a linear matrix equation which quantifies what is called the Crank-Nicholson Method. It is represented as following:

$$\left(\mathbf{I}-\frac{1}{2}\Delta z\mathbf{L}\right)\mathbf{v}^{n+1}=\left(\mathbf{I}+\frac{1}{2}\Delta z\mathbf{L}\right)\mathbf{v}^{n}$$
$$\mathbf{A}\mathbf{v}^{n+1}=\mathbf{B}\mathbf{v}^{n}$$

3 Implementation

Our Task was to apply Crank-Nicolson (CN) scheme to implement the Beam Propagation Method(BPM) for a 1D Gaussian field distribution. In 1D field, the scalar paraxial wave equation can be written as

$$\frac{\partial}{\partial z}v(x,z)=[\frac{i}{2\bar{k}}\frac{\partial^2}{\partial x^2}+i\frac{k^2(x)-\bar{k}^2}{2\bar{k}}]v(x,z)$$

where $k(x)=\frac{2\pi}{\lambda}n(x)$, and $\bar{k}$ is the average value of k.

So our first step is to create the linear differential operator L:

$$L=\frac{i}{2\bar{k}}\frac{\partial^2}{\partial x^2}+i\frac{k^2(x)-\bar{k}^2}{2\bar{k}}$$

which is shown in the lines of 10-15 of the source code. Due to the very high sparseness of matrix L, `scipy.sparse` package was used to define tridiagonal matrix L. The secondary and the main diagonals of L are defined on lines 11 and 12. Line 4 moves the secondary diagonals 1 position left and right away from the main diagonal. In line 15 `dia-array` was used to assemble the operator matrix L.

The second step to implement the CN scheme to calculate v_{n+1}. As mentioned above, CN is an explicit-implicit method, we obtain the central difference by averaging between for- and backward difference, yields,

$$(I-\frac{1}{2}\Delta zL)v_{n+1}=(I+\frac{1}{2}\Delta zL)v_n$$

where I is an identity matrix with the same size of L. The solution of the above equation are calculated by using `scipy.spares.spsolve` function as shown in lines 22 and 23. `Scipy.sparse` can save computer memory and reduce computation time significantly.

The whole source code of CN method is shown below.

```python
def beamprop_CN(v_in, lam, dx, n, nd, z_end, dz, output_step=1):

    size = dz * output_step
    z = np.arange(0, z_end, size)

    N = len(v_in)
    k = 2 * np.pi / lam * n
    k_mean = 2 * np.pi / lam * nd
    W = (k ** 2 - k_mean ** 2) / (2 * k_mean)
    sec = (1j / (2 * k_mean * dx ** 2)) * np.ones(N)
    main = -2 * sec + 1j * W
    data = np.array([sec, main, sec])
    offsets = np.array([-1, 0, 1])
    L = sps.dia_array((data, offsets), shape=(N, N)).tocsc()
    I = sps.eye(N)
    A = I - 0.5 * size * L
    B = I + 0.5 * size * L

    v = np.zeros((N, len(z))).astype('complex')
    v[:, 0] = v_in
    for i in range(1, len(z)):
        v[:, i] = spsolve(A, B @ v[:, i-1])

    return v, z
```

4 Examples

The Figure 3 shows the example considered to simulate the BPM method. For this a step index profile with respective refractive index for core and cladding are specified, which is ploted in Fig. 1. Then an input beam profile plotted in Fig. 2, which is chosen to be normalized Gaussian is selected as our input optical signal for this example.

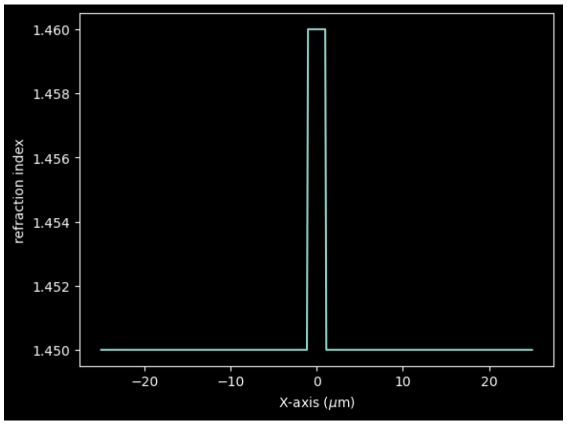

Figure 1: refractive index distribution

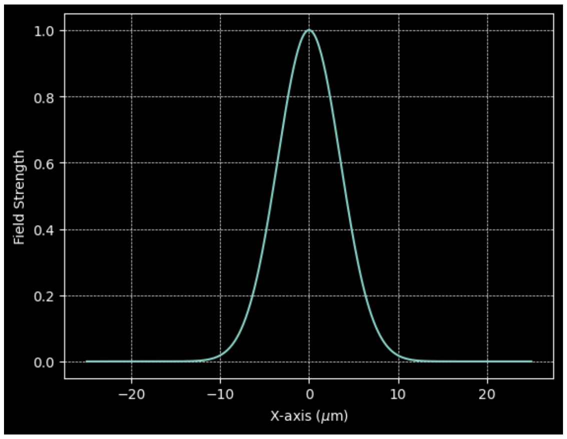

Figure 2: Gaussian field distribution

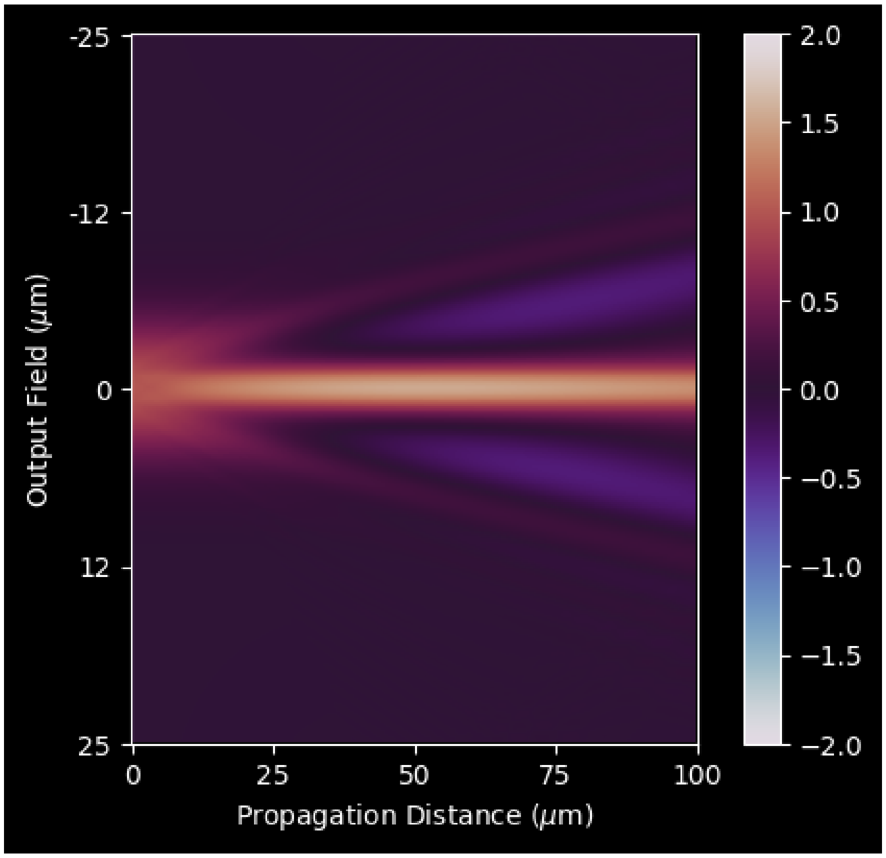

Figure 3: The field distribution with given parameters

5 Convergence Tests

The accuracy and convergence of the implemented algorithm are relative to two parameters, the resolution of the field distribution (dx) determined by the number of grid points (Nx) and the step size of propagation (dz). The convergence of the beam propagation method in dependence of the dx is tested with the following script:

The range of Nx is set to 51 to 1001 in line 9. A `for` loop is used to compute the output field with `beampro_CN` function for each Nx. Because the numbers of grid points are different, the output fields have different shapes. Hence, we unified the shape by picking the values at the integer position alone x-axis (..., -2, -1, 0, 1, 2, ...) in line 15. At last, The root-mean-square error (RMSE) between each output field and the true field

$$RMSE = \sqrt{\frac{\sum (v_{out} - v_{true})^2}{N}}$$

was calculated in line 16. The ground truth field distribution is approximated by the result for a huge value of Nx (5001). The RMSE vs the number of grid points (Nx) is plotted in Fig. 4. With the increase of grid points the RMSE reduce steadily and the downward trend is becoming less pronounced. When the grid points are more than 350, in order words dx is less than 0.14 μm, RMSE drop to less than 10^{-3}.

The step size in propagation is also one of parameter affecting the output fields. The convergence in dependence of the step size is tested with the range of $0.005\mu m$ to $1\mu m$ and the approximate true output field is calculated with dz = $0.001\mu m$. The results are presented in Fig. 5. In general, the RMSE increases with increasing of step size, and there is an oscillation when dz is in the range of $0.6\mu m$ to $0.9\mu m$. However, for all dz values the RMSEs are smaller than 10^{-5}. As conclusion, the step size is not the major factor affecting the result of CN method.

Figure 6 visualizes the change of the result caused by increasing of Nx. The cyclic colormap 'twilight' was used. As discussed above, dz have a limit influence to the result. We decrease dz with increasing of Nx to keep the numbers of points in z and x direction identical and give a better visualization. The initial gaussian field concentrates in the area of the code alone with propagation. In case of Nx = 101, the central amplitude of the gaussian field reduce slightly. In other cases, there is no difference among the result by different Nx.

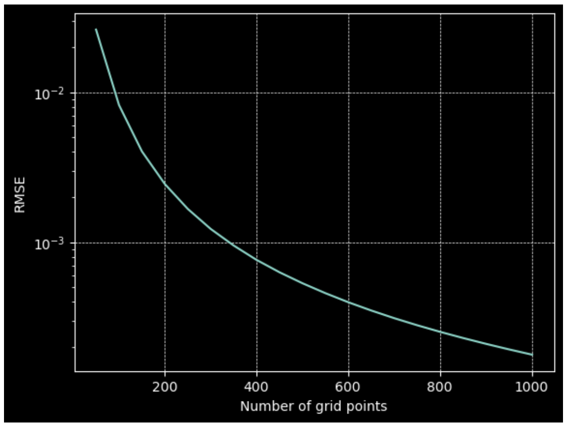

Figure 4: Convergence of the field distribution upon variation of Number of grid points

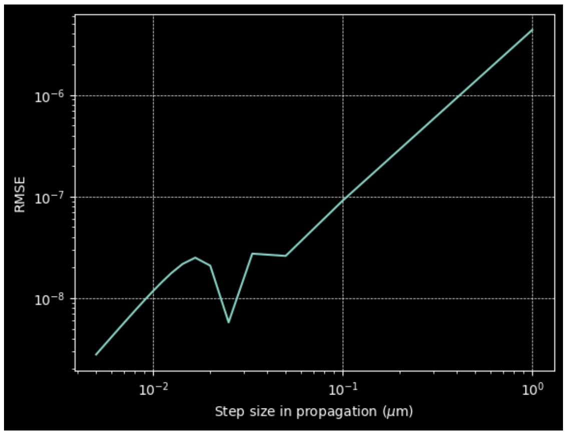

Figure 5: Convergence of the field distribution upon variation of Step size in propagation direction

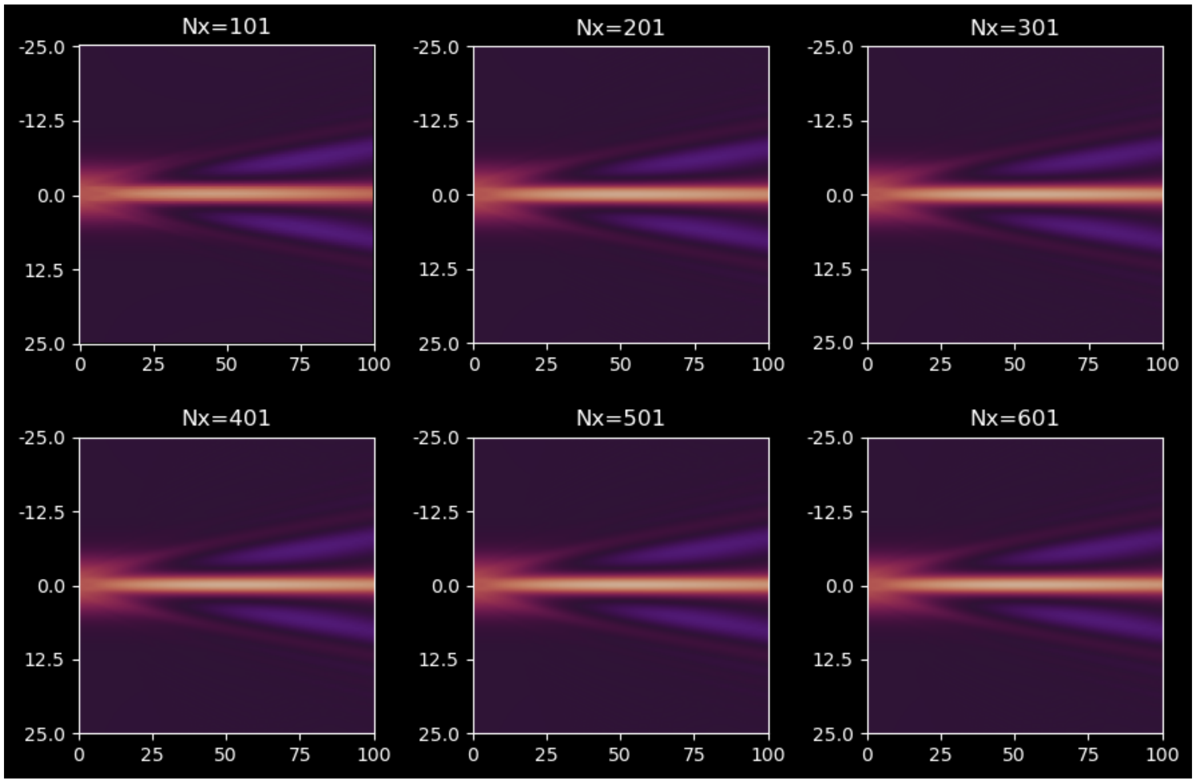

Figure 6: field distributions upon with variation of Nx