Computational Photonics

Seminar 05, 17.05.2024

Homework 2:
Implementation of the Beam Propagation Method

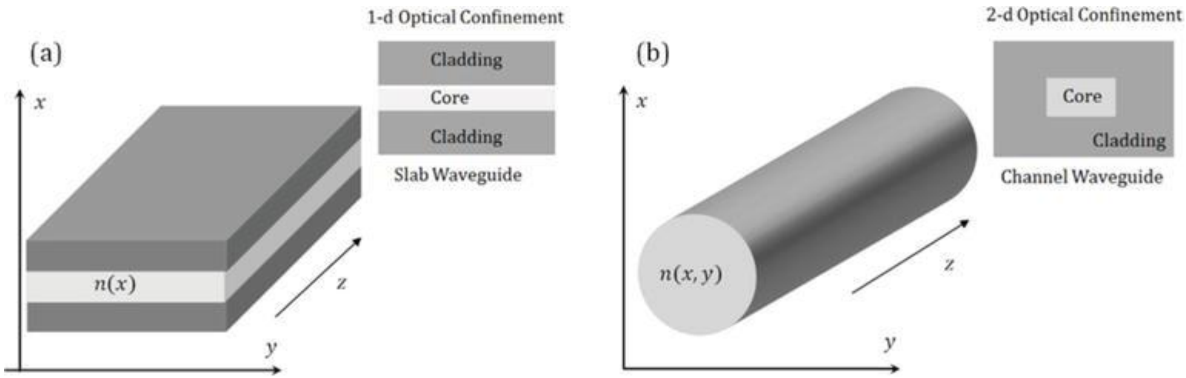
- Implementation of the explicit-implicit Crank-Nicolson scheme

- Test by propagating a Gaussian beam through an inhomogeneous medium

Abbe School of Photonics | JENA
Friedrich-Schiller-Universität

Beam Propagation Method

- Usual waveguiding geometry (***z***-axis as explicit propagation direction)

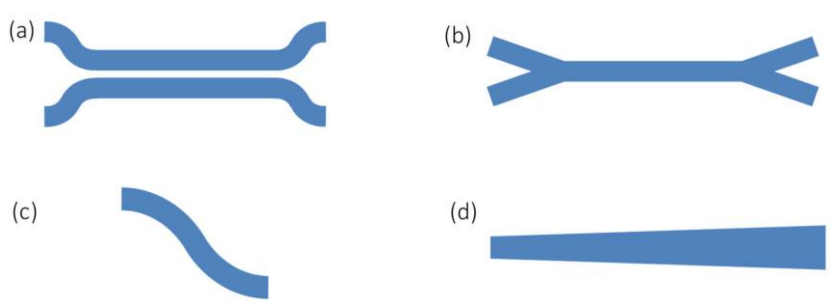


Selvaraja, S. K. , & Sethi, P. (2018). Review on Optical Waveguides. In (Ed.), Emerging Waveguide Technology. IntechOpen. https://doi.org/10.5772/intechopen.77150

So far: invariance in propagation direction

→ Stationary modes

- Now: allow for slow variation along propagation direction



Propagation of <u>input light field</u> through inhomogeneous medium

→ Initial value problem

Abbe School of Photonics JENA
Friedrich-Schiller-Universität

Overview of BPM

- Assumptions:
  - Slowly Varying Envelope Approximation (SVEA)

    $$\Phi(x,y,z)=\phi(x,y,z)\exp(-ikn_0 z) \quad \text{with } k=2\pi/\lambda$$

  - Without reflection
  - For scalar BPM: negligible coupling between transverse field components

- Validity of assumption:
  - Require slow variation of refractive index along z
  - Require good choice of n_0
  - For scalar BPM: low index contrast in transverse direction

Overview of BPM

- Inputs:
  - Incident wavelength λ
  - Incident field profile $\mathbf{E_0}(x, y) = \mathbf{E}(x, y, z = 0)$
  - Refractive index $n(x, y, z)$ (*can vary slightly along z)

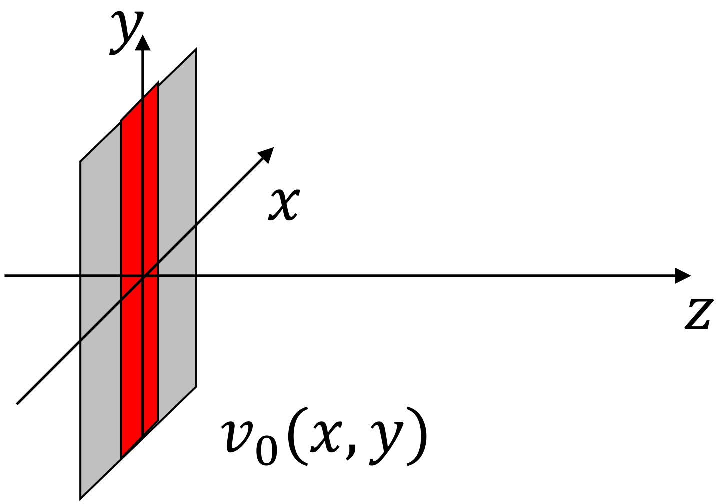
- Outputs:
  - Stationary field distribution of whole space $\mathbf{E}(x, y, z)$

The paraxial wave equation is an initial value problem!

For this homework: scalar BPM

- Scalar field
- invariant along y: 1+1 Dimension

given: $v_0\,(x) = v(x, z=0)$
wanted: $v(x, z)$



<u>Scalar paraxial wave equation in 1+1D</u>

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

with $v(x, z)$ - one vector component of electric field and

$$k(x) = \frac{2\pi}{\lambda} n(x), \qquad \bar{k} = \frac{2\pi}{\lambda}\bar{n}$$

Initial-value problem

- **Rearranging** the paraxial wave equation yields

$$\underbrace{\boldsymbol{\frac{\partial}{\partial z}} v(x,z)}_{\text{discretization of longitudinal coordinate } (z)} = \underbrace{\left[\frac{i}{2\bar{k}}\frac{\partial^2}{\partial x^2} + i\,\frac{k^2(x)-\bar{k}^2}{2\bar{k}}\right]}_{\text{discretization of transverse coordinate } (x)} v(x,z) = \boldsymbol{L} v(x,z)$$

Discretization of transverse coordinate

$$\boldsymbol{L}v(x,z) = \frac{i}{2\bar{k}}\frac{\partial^2 v(x,z)}{\partial x^2} + iW(x)v(x,z), \qquad W(x) = \frac{k^2(x) - \bar{k}^2}{2\bar{k}}$$

- **Discretizing** along x ($x_j = j\Delta x$) …

$$v(j\Delta x, z) = v_j(z)$$

$$W(j\Delta x) = W_j$$

$$\left.\frac{\partial^2 v(x,z)}{\partial x^2}\right|_{x_j} \approx \frac{v_{j+1}(z) - 2v_j(z) + v_{j-1}(z)}{(\Delta x)^2}$$

Discretization of transverse coordinate

- The matrix $\mathbf{L}$ is tridiagonal and symmetric

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

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

- On the first and on the last row of $\mathbf{L}$ one boundary value is missing (implicitly assumed to be zero) → **perfect electric conducting boundary conditions**

Initial-value problem

- **Rearranging** the paraxial wave equation yields

$$\underbrace{\frac{\partial}{\partial z}v(x,z)}_{\text{discretization of longitudinal coordinate }(z)} = \underbrace{\left[\frac{i}{2\bar{k}}\frac{\partial^2}{\partial x^2} + i\frac{k^2(x)-\bar{k}^2}{2\bar{k}}\right]}_{\text{discretization of transverse coordinate }(x)} v(x,z) = \boldsymbol{L}v(x,z)$$

discretization of longitudinal coordinate (*z*)

discretization of transverse coordinate (*x*) ✓

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

Abbe School of Photonics | JENA
Friedrich-Schiller-Universität

Discretization of longitudinal coordinate

Discretizing along z ($z_n = \mathrm{n}\Delta z$) gives second index

$$v(j\Delta x, n\Delta z) = v_j^n$$

<u>Three options:</u>

(how to find an iterative scheme to advance in z direction)

- forward difference

$$\left.\frac{\partial}{\partial z} v_j(z)\right|_{z^n} \approx \frac{v_j^{n+1} - v_j^n}{\Delta z} \approx \sum_l L_{jl} v_l^n = \underline{\mathbf{L}} v^n$$

- backward difference

$$\left.\frac{\partial}{\partial z} v_j(z)\right|_{z^{n+1}} \approx \frac{v_j^{n+1} - v_j^n}{\Delta z} \approx \sum_l L_{jl} v_l^{n+1} = \underline{\mathbf{L}} v^{n+1}$$

- central difference

$$\left.\frac{\partial}{\partial z} v_j(z)\right|_{z^{n+1/2}} \approx \frac{v_j^{n+1} - v_j^n}{\Delta z} \approx \frac{1}{2}\sum_l \left(L_{jl} v_l^n + L_{jl} v_l^{n+1}\right) = \frac{1}{2}\underline{\mathbf{L}}\left(v^n + v^{n+1}\right)$$

Contains the discretization in x

Von Neumann stability analysis

- Let's take the iteration equation of **forward differencing (explicit iteration)**:

$$\frac{\mathbf{A}^{n+1} - \mathbf{A}^{n}}{\Delta z} = \mathbf{L}\mathbf{A}^{n}$$

- Consider now, that the computer will solve that equation with some numerical error.

$$A_j^n = A_{\mathrm{exact},j}^n + \delta_j^n$$

- Since the equation is linear, superposition holds:

$$\Rightarrow \frac{\boldsymbol{\delta}^{n+1} - \boldsymbol{\delta}^{n}}{\Delta z} = \mathbf{L}\boldsymbol{\delta}^{n}$$

Von Neumann stability analysis

- Let us make a Fourier analysis of that error in the transverse spatial domain (x):

$$\delta(x,z) = \sum_{\kappa} c_{\kappa}(z)\exp(i\kappa x), \qquad \kappa = n \cdot \frac{2\pi}{N \cdot \Delta x}, n \in \mathbb{Z}$$

- The $c_{\kappa}(z)$ describe the evolution of the Fourier amplitude with z iteration

- Fourier sum is also a linear operation, so it is sufficient to look at one component

Von Neumann stability analysis

- applying transverse paraxial beam operator **L** to one Fourier component of the error.

- For $W_j = 0$:

$$\tilde{\mathcal{L}}\{c_\kappa(z)\exp(i\kappa x)\} = \frac{i}{2\bar{k}}\frac{\exp(i\kappa\Delta x) - 2 + \exp(-i\kappa\Delta x)}{(\Delta x)^2}\exp(i\kappa x)\,c_\kappa(z)$$

$$= i\frac{\cos(\kappa\Delta x) - 1}{\bar{k}(\Delta x)^2}\exp(i\kappa x)\,c_\kappa(z)$$

Von Neumann stability analysis

- The LHS of the iteration is determined by the chosen differencing scheme

- variant for forward differencing:

$$\frac{c_\kappa(z+\Delta z) - c_\kappa(z)}{\Delta z} \exp(i\kappa x)$$

- whole scheme then reads:

$$c_\kappa(z+\Delta z) = \left[1 + i\Delta z \frac{\cos(\kappa \Delta x) - 1}{\bar{k}(\Delta x)^2}\right] c_\kappa(z)$$

Von Neumann stability analysis

- we have found the expression how a Fourier component of the error iterates forward:

$$c_\kappa(z+\Delta z) = \left[1 + i\Delta z \frac{\cos(\kappa\Delta x) - 1}{\bar{k}(\Delta x)^2}\right] c_\kappa(z)$$

- in general:

$$c_\kappa(z+\Delta z) = g(\kappa) c_\kappa(z)$$

(κ dependent complex error gain factor g)

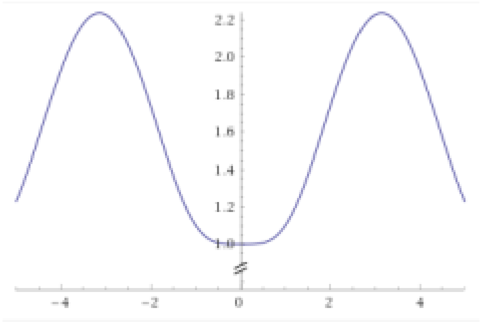
- stability requires:

$$\boxed{|g(\kappa)| \leq 1 \qquad \text{(for all } \kappa!)}$$

Von Neumann stability analysis

- However, in the forward differencing variant, we have:

$$|g(\kappa)| = \left[1 + (\Delta z)^2 \frac{[\cos(\kappa \Delta x) - 1]^2}{\bar{k}^2 (\Delta x)^4}\right]^{\frac{1}{2}} \geq 1$$



- This means that the error will always grow exponentially while iterating in this way (even independent of κ)!

→ **Explicit BPM from forward differencing is**
unconditionally unstable!!!

Von Neumann stability analysis

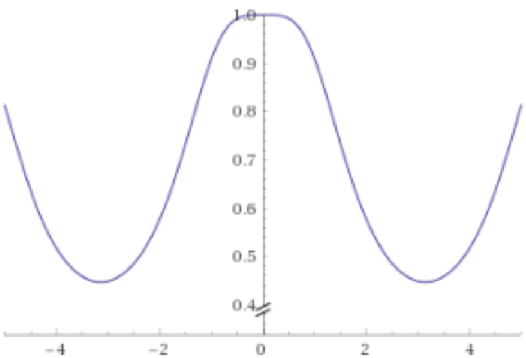
- How about backward differencing?
  - LHS of iteration equation:

$$\frac{c_\kappa(z) - c_\kappa(z-\Delta z)}{\Delta z} \exp(i\kappa x)$$

  - This means we have for the whole iteration equation:

$$c_\kappa(z) = \frac{1}{1 - i\Delta z \frac{\cos(\kappa \Delta x) - 1}{\bar{k}(\Delta x)^2}} c_\kappa(z-\Delta z)$$

$$\Rightarrow |g(\kappa)| \leq 1$$



Von Neumann stability analysis

- How about backward differencing?
  - This means, that implicit iteration BPM based on backward differencing is **unconditionally stable**
  - However, the error levels change during iteration
  - That means, that this scheme cannot be energy conserving!

Von Neumann stability analysis

- What is the ideal iteration scheme?
  - Ideal iterator preserves the numerical error level (pure phase transformation of rounding errors)
  - This requires: $$|g(\kappa)| = 1$$
  - Does it exist?

Von Neumann stability analysis

- **Crank-Nicolson Scheme:**
  - balances explicit & implicit by averaging between for- and backward differencing (results in central differencing)
  - iteration equation:

$$\left(1 - \frac{1}{2}\Delta z\tilde{L}\right) c_\kappa(z+\Delta z) = \left(1 + \frac{1}{2}\Delta z\tilde{L}\right) c_\kappa(z)$$

  - error gain factor:

$$g(\kappa) = \frac{1 + i\Delta z \frac{\cos(\kappa\Delta x)-1}{2\bar{k}(\Delta x)^2}}{1 - i\Delta z \frac{\cos(\kappa\Delta x)-1}{2\bar{k}(\Delta x)^2}} \quad \Rightarrow |g(\kappa)| = 1$$

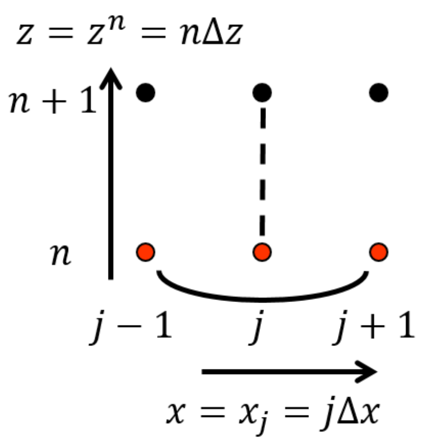
Discretization of longitudinal coordinate

Discretization of the z coordinate

$$z^n = n\Delta z$$
$$v_j^n = v_j(n\Delta z)$$

Forward difference (**Explicit** scheme)

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



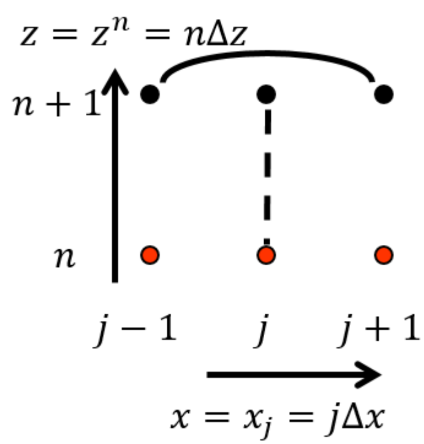
always unstable

error grows exponentially

$$\boxed{\begin{gathered}\mathbf{v}^{n+1} = (\mathbf{I} + \Delta z\mathbf{L})\mathbf{v}^n \\ \mathbf{v}^{n+1} = \mathbf{A}\mathbf{v}^n\end{gathered}}$$

Backward difference (**Implicit** scheme)

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



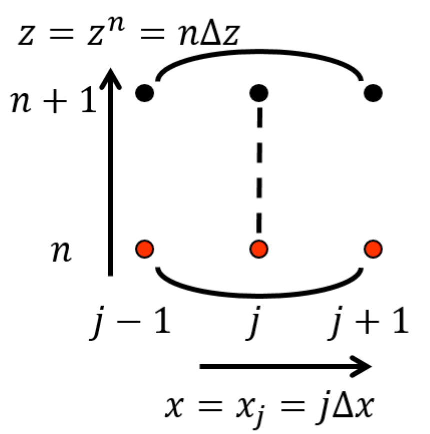
stable but in general **not energy conserving**

$$\boxed{\begin{gathered}\mathbf{v}^{n+1} = (\mathbf{I} - \Delta z\mathbf{L})^{-1}\mathbf{v}^n \\ \mathbf{v}^{n+1} = \mathbf{A}^{-1}\mathbf{v}^n\end{gathered}}$$

Discretization of longitudinal coordinate

Central difference
(Explicit-Implicit scheme)

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



stable and **energy conserving**

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

$$\mathbf{A}\mathbf{v}^{n+1} = \mathbf{B}\mathbf{v}^{n}$$

Tasks

1. Implement the functions **`waveguide`** and **`gauss`** to set up the problem:

   Slab waveguide with step index profile

   excited by a Gaussian initial field distribution

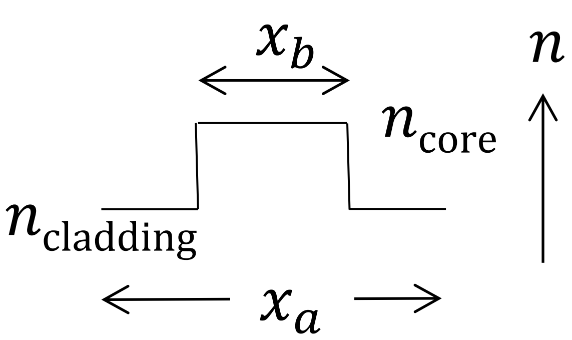
2. Implement the BPM using only the explicit-implicit scheme (function **`beamprop_CN`**)

3. Test the convergence and accuracy of obtained results vs. parameters ***dz*** and ***Nx***

4. *(Voluntary) Implement the BPM using the explicit (forward) and implicit (backward) scheme and compare the outcome to the results obtained with the Crank-Nicolson-scheme.*

Function *waveguide*
index distribution of step profile waveguide

```
def waveguide(xa, xb, Nx, n_cladding, n_core):
    '''Generates the refractive index distribution of a slab waveguide with step
    profile centered around the origin of the coordinate system with a refractive
    index of n_core in the waveguide region and n_cladding in the surrounding
    cladding area. All lengths have to be specified in µm.
    Parameters
    ----------
        xa : float
            Width of calculation window
        xb : float
            Width of waveguide
        Nx : int
            Number of grid points
        n_cladding : float
            Refractive index of cladding
        n_core : float
            Refractive index of core
    Returns
    -------
        n : 1d-array
            Generated refractive index distribution
        x : 1d-array
            Generated coordinate vector
    '''
    pass
```

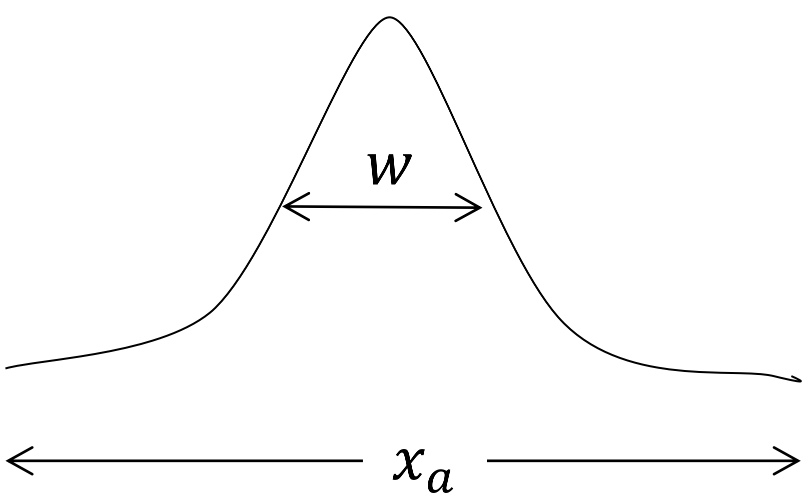


Function *gauss*
Gaussian initial field distribution

```
def gauss(xa, Nx, w):
    '''Generates a Gaussian field distribution v = exp(-x^2/w^2)  centered
    around the origin of the coordinate system and having a width of w.
    All lengths have to be specified in µm.

    Parameters
    ----------
        xa : float
            Width of calculation window
        Nx : int
            Number of grid points
        w  : float
            Width of Gaussian field

    Returns
    -------
        v : 1d-array
            Generated field distribution
        x : 1d-array
            Generated coordinate vector
    '''
    pass
```



BPM using Crank-Nicolson scheme

```python
def beamprop_CN(v_in, lam, dx, n, nd,  z_end, dz, output_step):
    '''Propagates an initial field over a given distance based on the solution of the
    paraxial wave equation in an inhomogeneous refractive index distribution using the
    explicit-implicit Crank-Nicolson scheme. All lengths have to be specified in µm.
    Parameters
    ----------
        v_in : 1d-array
            Initial field
        lam : float
            Wavelength
        dx : float
            Transverse step size
        n : 1d-array
            Refractive index distribution
        nd : float
            Reference refractive index
        z_end : float
            Propagation distance
        dz : float
            Step size in propagation direction
        output_step : int
            Number of steps between field outputs
    Returns
    -------
        v_out : 2d-array
            Propagated field
        z : 1d-array
            z-coordinates of field output
    '''
    pass
```

__Hint__: make use of the routines for sparse matrices from the respective package in SciPy (spsolve).

Test parameters

- Test your implementation with the following parameters:
  - Waveguide: `xa = 50µm`, `xb = 2µm`, `Nx = 251`, `n_cladding = 1.45`, `n_core = 1.46`
  - Initial field: `w = 5.0µm`
  - Solver: `z_end = 100µm`, `dz = 0.5µm`, `nd = 1.455`, `lambda = 1µm`
- The solution will vary slowly along z, you can choose `output_step` larger than 1 (e.g. `output_step*dz = 1.0µm`)

Summary Homework 2

- Solve tasks I, II & III.
- For each task we require that each group implements a program that solves the problem and documents the code and its result (e.g. with an iPython Notebook). This includes a graphical display of the result!
- Submission via email to: teaching-nanooptics@uni-jena.de
- **Due: 06.06.2024, 03:00 sharp**
- The subject line of the email should have the following format:
  - CPho24 - solution to the homework 1: group [Number]; [family_name1, family_name2, family_name3]:
  - gather all your files in a single zip archive (no rar, tar, 7z, gz or any other compression format)