

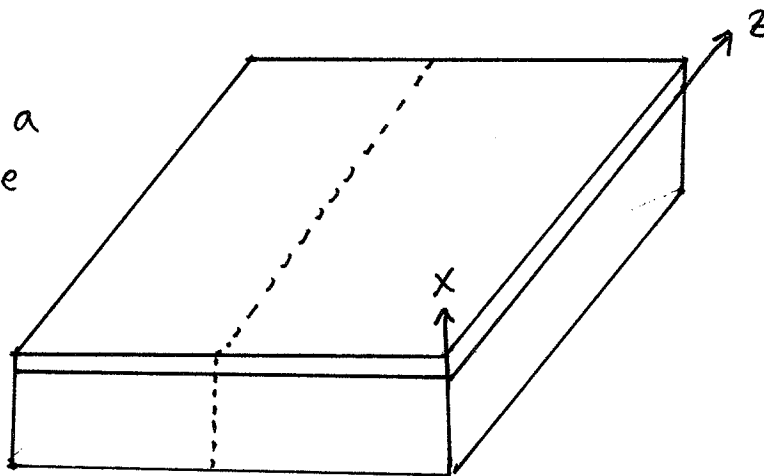
3/21/00

Elen 6440

①
S. Blair

Beam propagation method for integrated optics

We can treat propagation in a slab waveguide as a z-D problem.



Propagation of the TE waveguide mode is exactly described by the scalar Helmholtz equation:

$$\frac{\partial^2 A(x, z)}{\partial x^2} + \frac{\partial^2 A(x, z)}{\partial z^2} + \frac{\omega^2}{c^2} n^2(x, z) A(x, z) = 0$$

$$E = \frac{1}{2} A e^{-i\omega t} + c.c.$$

for propagation along z, we can re-write this equation

$$\frac{\partial A}{\partial z} = \pm i \sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2(x, z)} A$$

$$\text{or } A(x, z) = e^{i \int_0^z \sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2} dz} A(x, 0)$$

if we let $z = \Delta z$, where Δz is a small distance, then we can approximate the integral

$$A(x, \Delta z) \approx e^{i \Delta z \sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2}} A(x, 0)$$

where n^2 must be evaluated at $z = \frac{\Delta z}{2}$

Now, the ^{square-root} exponent of an operator is interpreted by its Taylor expansion, where

$$\sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2} = \frac{\omega}{c} n \left[1 + \frac{c^2}{2n^2 \omega^2} \frac{\partial^2}{\partial x^2} - \dots \right]$$

the refractive index $n(x, z) = n_0 + \Delta n(x, z)$, where $\frac{\Delta n}{n} \ll 1$, and we can approximate

$$\frac{\omega}{c} n [\] \approx \frac{\omega}{c} (n_0 + \Delta n) \left[1 + \frac{c^2}{2n_0^2 \omega^2} \frac{\partial^2}{\partial x^2} - \dots \right]$$

so now,

$$\begin{aligned} A(x, \Delta z) &= e^{i \frac{\omega}{c} \Delta n(\frac{\Delta z}{2}) \Delta z} e^{i \frac{\omega}{c} n_0 \left[1 + \frac{c^2}{2n_0^2 \omega^2} \frac{\partial^2}{\partial x^2} - \dots \right] \Delta z} A(x, 0) \\ &= e^{i \frac{\omega}{c} \Delta n(\frac{\Delta z}{2}) \Delta z} e^{i \left[\frac{\omega}{c} n_0 + \frac{c}{2n_0 \omega} \frac{\partial^2}{\partial x^2} - \dots \right] \Delta z} A(x, 0) \end{aligned}$$

we can evaluate the operator in the Fourier domain

$$A(x, \Delta z) = e^{i \frac{\omega}{c} \Delta n(\frac{\Delta z}{2}) \Delta z} \left\{ e^{i \left[\frac{\omega}{c} n_0 + \frac{c}{2n_0 \omega} \frac{\partial^2}{\partial x^2} - \dots \right] \Delta z} \times \sum_{m=1-N/2}^{N/2} E_m e^{(2\pi i/L) m x} \right\}$$

E_m = ampl. coefficients of Fourier series for $A(x, 0)$
 $k_x = \frac{2\pi m}{L}$ where L = physical length of sampling grid
 N = # of samples

$$A(x, \Delta z) = e^{i \frac{\omega}{c} \Delta n \Delta z} \left\{ \sum_{m=1-N/2}^{N/2} E_m e^{i k_x x} \right\}$$

(3)

$$\begin{aligned}
 \text{where } \mathcal{E}_m' &= \mathcal{E}_m e^{i \left[\frac{\omega}{c} n_0 - \frac{c}{2n_0\omega} k_x^2 + \dots \right] \Delta z} \\
 &= \mathcal{E}_m e^{i \left[k_f n_0 - k_x^2 / 2k_f n_0 + \dots \right] \Delta z} \\
 &= \mathcal{E}_m e^{i k_0 \left[1 - k_x^2 / 2k_0^2 + \dots \right] \Delta z} \\
 &= \mathcal{E}_m e^{i k_0 \sqrt{1 - k_x^2 / k_0^2} \Delta z} = \mathcal{E}_m e^{i \sqrt{k_0^2 - k_x^2} \Delta z}
 \end{aligned}$$

$$A(x, \Delta z) = e^{i \frac{\omega}{c} \Delta n \Delta z} \left\{ \sum \mathcal{E}_m e^{i \sqrt{k_0^2 - k_x^2} \Delta z} e^{i k_x x} \right\}$$

Symbolically, we have

$$\begin{aligned}
 A(x, \Delta z) &= e^{i \frac{\omega}{c} \Delta n \Delta z} \hat{\mathcal{F}}^{-1} \left\{ \hat{\mathcal{F}} \{ A(x, 0) \} e^{i \sqrt{k_0^2 - k_x^2} \Delta z} \right\} \\
 &= e^{i \phi \Delta z} e^{i \hat{Q} \Delta z} A(x, 0)
 \end{aligned}$$

it turns out that accuracy can be improved by using the symmetrized form

$$A(x, \Delta z) = e^{i \frac{\Delta z}{2} \hat{Q}} e^{i \phi \Delta z} e^{i \frac{\Delta z}{2} \hat{Q}} A(x, 0)$$

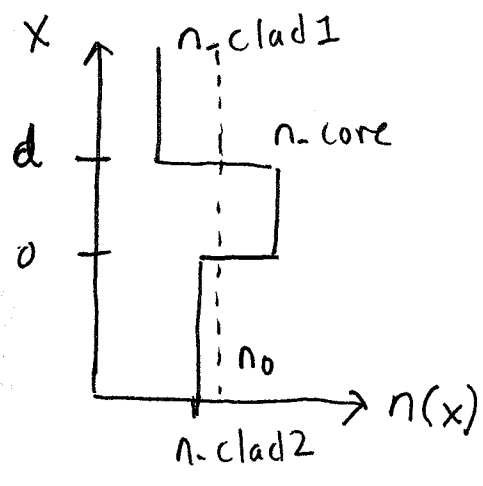
note that we must choose Δz small so that $A(x, \Delta z) \approx A(x, 0)$

for propagation over a distance $2\Delta z$:

$$A(x, 2\Delta z) = e^{i \frac{\Delta z}{2} \hat{Q}} e^{i \phi \Delta z} e^{i \frac{\Delta z}{2} \hat{Q}} e^{i \phi \Delta z} e^{i \frac{\Delta z}{2} \hat{Q}} A(x, 0)$$

and so on for $z = 3\Delta z$, etc.

now, the question ~~arises~~ arises as to how to choose n_0 for a given refractive index distribution. there are many solutions, but it can be shown that the optimal solution results where $n_0 = \frac{n_{max} + n_{min}}{2}$.



actually, you can choose just about any n_0 that satisfies

$$n_{min} < n_0 < n_{max}.$$

this method of beam propagation is known as the split-step Fourier method. there are a bunch of other techniques such as: finite difference, FDTD, finite element, and so on. the split-step method is by far the easiest to understand and implement, and can give fairly accurate results.

Derivation of the Beam Propagation Algorithm

Scalar Helmholtz equation

$$\frac{\partial^2 E(x, z)}{\partial x^2} + \frac{\partial^2 E(x, z)}{\partial z^2} + \frac{\omega^2}{c^2} n^2(x, z) E(x, z) = 0$$

$$\frac{\partial E(x, z)}{\partial z} = \pm i \sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2(x, z)} E(x, z)$$

integrate ...

$$E(x, \Delta z) = e^{i \int_0^{\Delta z} \sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2(x, z)} dz} E(x, 0)$$

approximate ...

$$E(x, \Delta z) \approx e^{i \Delta z \sqrt{\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2(x, 0)}} E(x, 0)$$

- Δz is small so that $n^2(x, 0) \approx n^2(x, \Delta z)$
- $\partial^2/\partial x^2$ independent of z

binomial expand exponential argument ...

$$\left(\frac{\partial^2}{\partial x^2} + \frac{\omega^2}{c^2} n^2(x, 0) \right)^{1/2} \rightarrow n(x, 0) k_o \left(1 + \frac{c^2}{2n_o^2 \omega^2} \frac{\partial^2}{\partial x^2} - \dots \right)$$

$$E(x, \Delta z) = e^{in(x, 0)k_o \Delta z} \left\{ e^{ik \Delta z [(1/2k^2)(\partial^2/\partial x^2) - \dots]} E(x, 0) \right\}$$

- $n(x, 0) = n_o + \Delta n(x, 0)$
- $k_o = \omega/c$ is free-space propagation constant
- $k = \omega n_o/c$ is material propagation constant
- in Fourier domain - $\frac{\partial}{\partial x} \rightarrow ik_x$ and $\frac{\partial^2}{\partial x^2} \rightarrow -k_x^2$

$$E(x, \Delta z) = e^{in(x,0)k_o\Delta z} \left\{ e^{ik\Delta z[(1/2k^2)(\partial^2/\partial x^2)-\dots]} \sum_{m=1-N/2}^{N/2} \mathcal{E}_m e^{(2\pi i/L)mx} \right\}$$

- \mathcal{E}_m is the Fourier series for $E(x, 0)$
- $k_x = 2\pi m/L$ where L is length of sampling grid
- N is number of samples

$$E(x, \Delta z) = e^{in(x,0)k_o\Delta z} \left\{ \sum_{m=1-N/2}^{N/2} \mathcal{E}'_m e^{(2\pi i/L)mx} \right\}$$

$$\begin{aligned} \mathcal{E}'_m &= \mathcal{E}_m e^{ik\Delta z(-k_x^2/2k^2+\dots)} \\ &= \mathcal{E}_m e^{-ik\Delta z} e^{ik\Delta z\sqrt{1-k_x^2/k^2}} \\ &= \mathcal{E}_m e^{-ik\Delta z} e^{i\Delta z\sqrt{k^2-k_x^2}} \end{aligned}$$

$$E(x, \Delta z) = e^{i\Delta n(x,0)k_o\Delta z} \sum_{m=1-N/2}^{N/2} \mathcal{E}_m e^{i\Delta z\sqrt{k^2-k_x^2}}$$

- $\Delta n(x, 0) = n_2 |E(x, 0)|^2$ for SPM
- $\Delta n(x, 0) = n_2 [|E_x(x, 0)|^2 + 2|E_y(x, 0)|^2]$ for SPM + CPM

Full propagation equation

$$E_x[x, (\ell + 1)\Delta z] = e^{i\Delta n(x, \ell\Delta z)k_o\Delta z} \mathcal{F}^{-1} \left\{ \mathcal{F} \{E_x(x, \ell\Delta z)\} e^{i\Delta z\sqrt{k^2-k_x^2}} \right\}$$

- $\Delta n(x, \ell\Delta z) = n_2 [|E_x(x, \ell\Delta z)|^2 + 2|E_y(x, \ell\Delta z)|^2]$

```

% this program takes an input beam and propagates it down a slab
% waveguide.  propagation occurs in 1-D.  you will need to add your
% eigenvalue solver to calculate the exact mode profile of the waveguide.
% you will also modify the code to simulate coupling between two
% 1-d waveguides.

```

```

lambda_f = 0.6328; % free-space wavelength, in um
xsize    = 256;    % transverse x grid size
dx       = lambda_f/5.0; % x sampling in um
distance = 500;    % propagation distance in um
dz       = 1.0;    % propagation step size in um
skip     = 20;     % # of propagation steps to skip when storing field

```

```

n_core   = 1.537; % core refractive index
n_clad1  = 1.53; % cladding 1 refractive index
n_clad2  = 1.53; % cladding 2 refractive index
thick    = 2.0;  % core thickness in um

```

```

%
% insert eigenvalue solver here
%
e_index = 1.534313; % effective mode index, fund. TE mode

```

```

k_eff    = 2.*pi*e_index/lambda_f; % calculate propagation constant
k_core   = 2.*pi*n_core/lambda_f;
k_clad1  = 2.*pi*n_clad1/lambda_f;
k_clad2  = 2.*pi*n_clad2/lambda_f;

```

```

k_x_core = sqrt(k_core*k_core-k_eff*k_eff);
alpha1_x = sqrt(k_eff*k_eff-k_clad1*k_clad1);
alpha2_x = sqrt(k_eff*k_eff-k_clad2*k_clad2);

```

```

xaxis    = dx*((1:xsize) - xsize/2); % set-up x axis scale
kx       = 2.0*pi*((1:xsize) - xsize/2)/(xsize*dx); % set-up kx scale
kz       = sqrt(k_eff*k_eff - kx.*kx); % propagation transfer

```

```

field    = 1:xsize;
index    = 1:xsize;

```

```

for k = 1:xsize,
    rx = xaxis(k);
    if abs(rx) < 0.5*thick % field inside core
        field(k) = cos(k_x_core*rx);
        index(k) = n_core-e_index;
    elseif rx <= -0.5*thick % field inside cladding 1
        field(k) = cos(0.5*k_x_core*thick)*exp(alpha1_x*(rx+0.5*thick));
        index(k) = n_clad1-e_index;
    else % field inside cladding 2
        field(k) = cos(0.5*k_x_core*thick)*exp(-alpha2_x*(rx-0.5*thick));
        index(k) = n_clad2-e_index;
    end
end

```

```

transfer1 = exp(i*0.5*kz*dz); % phase due to propagation, half-step
transfer1c = exp(-i*0.5*kz*dz); % phase due to propagation, half-step
transfer2 = exp(i*kz*dz); % phase due to propagation, full-step
refract   = exp(i*2.0*pi*index*dz/lambda_f); % waveguide refraction

```

```

steps = round(distance/dz); % number of propagation steps
zaxis = skip*dz*(1:steps/skip+1); % set-up z axis scale

```

```

store = zeros(steps/skip+1,xsize);

store(1,1:xsize) = abs(field).^2;

fft_field = fft(field); % take fourier transform of field
fft_field = fftshift(fft_field); % shift DC to center of grid
fft_field = fft_field.*transfer1; % prop. half step

for k = 1:steps,
    fft_field = fftshift(fft_field); % shift DC back to corner
    field = ifft(fft_field); % convert back to real space
    field = field.*refract; % propagate through waveguide

    if mod(k,skip) == 0
        store(k/skip+1,1:xsize) = abs(field).^2;
    end

    fft_field = fft(field); % take fourier transform of field
    fft_field = fftshift(fft_field); % shift DC to center of grid
    fft_field = fft_field.*transfer2; % diffract full step
end
fft_field = fft_field.*transfer1c; % correct for half step
fft_field = fftshift(fft_field); % shift DC back to corner
field = ifft(fft_field); % convert back to real space
store(steps/skip+1,1:xsize) = abs(field).^2;

set(gcf,'DefaultAxesFontSize',14);
set(gcf,'DefaultAxesFontName','Times');
set(gcf,'DefaultSurfaceMeshStyle','row');
hold off

mesh(xaxis,zaxis,store);
axis([min(xaxis) max(xaxis) min(zaxis) max(zaxis) 0 2]);

```