Υπολογιστικός Ηλεκτρομαγνητισμός
Άσκηση 2

Σεϊτανίδου Δήμητρα

21 Απριλίου 2020

Αριθμητική διασπορά στις 2 διαστάσεις

Σε αυτή την άσκηση θα μελετήσουμε την διασπορά στην ταχύτητα φάσης που παρατηρούμε όταν λύνουμε αριθμητικά τις εξισώσεις του Maxwell. Στην περίπτωση των δύο διαστάσεων οι κεντρικές διαφορές των εξισώσεων Maxwell για την περίπτωση TMz (transverse-magnetic mode with respect to z) είναι οι εξής:

$$\frac{H_{x_{i,j+1/2}}^{n+1/2} - H_{x_{i,j+1/2}}^{n-1/2}}{\Delta t} = \frac{1}{\mu_{i,j+1/2}} \left(\frac{E_{z_{i,j+1}}^{n} - E_{z_{i,j}}^{n}}{\Delta y} \right)$$

$$\frac{H_{y_{i+1/2,j}}^{n+1/2} - H_{y_{i+1/2,j}}^{n-1/2}}{\Delta t} = \frac{1}{\mu_{i+1/2,j}} \left(\frac{E_{z_{i+1,j}}^{n} - E_{z_{i,j}}^{n}}{\Delta x} \right)$$

$$\frac{E_{z_{i,j}}^{n+1} - E_{z_{i,j}}^{n}}{\Delta t} = \frac{1}{\epsilon_{i,j}} \left(\frac{H_{y_{i+1/2,j}}^{n+1/2} - H_{y_{i-1/2,j}}^{n+1/2}}{\Delta x} - \frac{H_{x_{i,j+1/2}}^{n+1/2} - H_{x_{i,j-1/2}}^{n+1/2}}{\Delta y} \right)$$

Επίσης θεωρούμε ότι η λύση της κυματικής εξίσωσης δίνεται από τις παρακάτω εξισώσεις:

$$E_{z_{i,j}}^{n} = E_{z_0} e^{j(\omega n \Delta t - k_x i \Delta x - k_y j \Delta y)}$$

$$H_{x_{i,j}}^{n} = H_{x_0} e^{j(\omega n \Delta t - k_x i \Delta x - k_y j \Delta y)}$$

$$H_{y_{i,j}}^{n} = H_{y_0} e^{j(\omega n \Delta t - k_x i \Delta x - k_y j \Delta y)}$$

Από τα δύο παραπάνω σετ σχέσεων προκύπτει η σχέση για την αριθμητική διασπορά:

$$\left[\frac{1}{c\Delta t} \sin\left(\frac{\omega \Delta t}{2} \right) \right]^2 = \left[\frac{1}{\Delta x} \sin\left(\frac{k_x \Delta x}{2} \right) \right]^2 + \left[\frac{1}{\Delta y} \sin\left(\frac{k_y \Delta y}{2} \right) \right]^2 \tag{1}$$

Αν ορίσουμε $S = c\Delta t \left(\frac{1}{\Delta x} + \frac{1}{\Delta y} \right)$ και λ/N το βήμα που παίρνουμε τότε η (1) γράφεται:

$$\frac{1}{S^2} \sin^2\left(\frac{\pi S}{N} \right) = \sin^2\left(\frac{\Delta x k \cos\phi}{2} \right) + \sin^2\left(\frac{\Delta y k \sin\phi}{2} \right) \tag{2}$$

Με ϕ συμβολίζεται η διεύθυνση διάδοσης του κύματος στον χώρο xy και η διεύθυνση προς τα θετικά του άξονα x ορίζεται με $\phi = 0$. Μια εναλλακτική μορφή της παραπάνω εξίσωσης, που εφαρμόζει την επαναληπτική μέθοδο του Newton, και είναι η σχέση η οποία θα χρησιμοποιήσουμε στον κώδικα, είναι η:

$$k_{i+1} = k_i - \frac{\sin^2(Ak_i) + \sin^2(Bk_i) - C}{A\sin(2Ak_i) + B\sin(2Bk_i)} \tag{3}$$

όπου

$$A = \frac{\Delta x \cos\phi}{2}, \qquad B = \frac{\Delta y \sin\phi}{2}, \qquad C = \frac{1}{S^2}\sin^2\left(\frac{\pi S}{N}\right)$$

Τέλος η ταχύτητα φάσης υπολογίζεται από την σχέση:

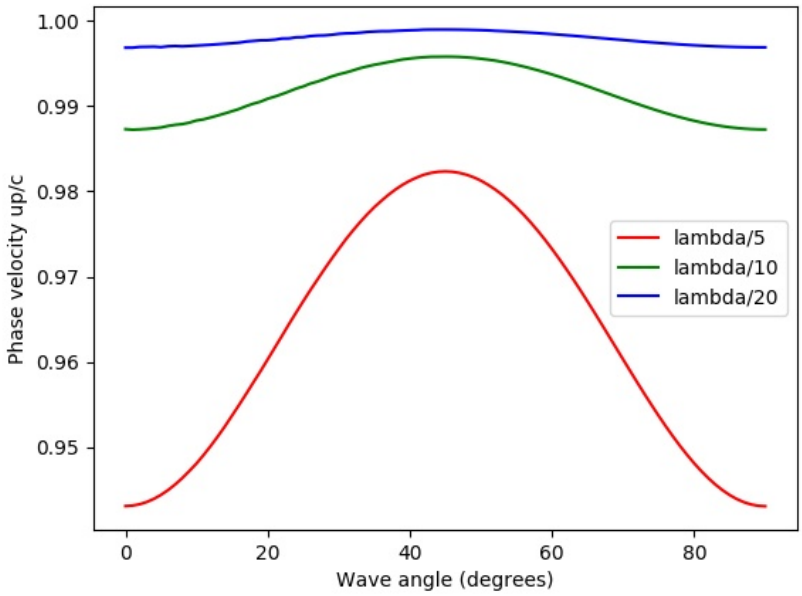
$$\frac{u_p}{c} = \frac{2\pi}{k} \tag{4}$$

Αποτελέσματα

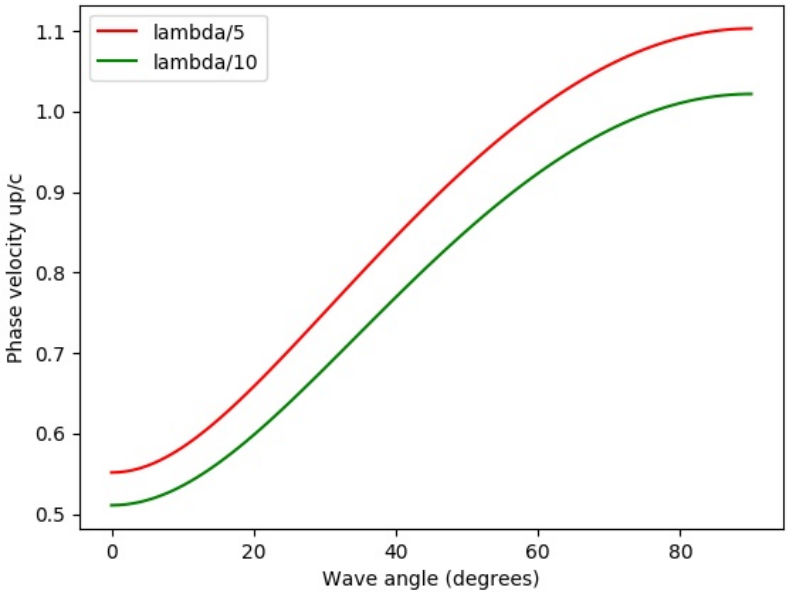
Θα εφαρμόσουμε τις σχέσεις (3) και (4) για δύο περιπτώσεις, μια για $\Delta x = \Delta y$ και μια για $\Delta x = 2\Delta y$. Θεωρούμε ότι $\Delta x = 1/N$, $\Delta t = 0.5\Delta x$, $c = 1$ η ταχύτητα του φωτός και $\lambda = 1$ το μήκος κύματος. Για την περίπτωση $\Delta x = \Delta y$ παίρνουμε $N = 5, 10, 20$ ενώ για την περίπτωση $\Delta x = 2\Delta y$ παίρνουμε $N = 5, 10$. Αρχική τιμή για τον κυματάριθμο επιλέγουμε την $k_{i=0} = 2\pi$.

Τα διαγράμματα που δίνουν τη μεταβολή της αριθμητικής ταχύτητα φάσης με το ϕ δίνονται παρακάτω.

Σχήμα 1: Αριθμητική ταχύτητα φάσης με $\Delta x = \Delta y$



Σχήμα 2: Αριθμητική ταχύτητα φάσης με $\Delta x = 2\Delta y$



Παρατηρούμε λοιπόν ότι η αριθμητική ταχύτητα φάσης είναι μικρότερη από την αναμενόμενη τιμή της (ταχύτητα φωτός) και εξαρτάται από την διεύθυνση διάδοσης, παρουσιάζει δηλαδή ανισοτροπική συμπεριφορά σε συνάρτηση με τον χώρο. Επιπλέον παρατηρούμε ότι όσο πιο μικρό βήμα παίρνουμε, τόσο πλησιάζουμε την ταχύτητα του φωτός. Τέλος στην περίπτωση $\Delta x = 2\Delta y$ παρατηρούμε ότι η ακρίβεια αυξάνεται κατά τον άξονα y και μειώνεται κατά τον άξονα x.

Ο κώδικας που χρησιμοποιήσαμε για να πάρουμε τα παραπάνω αποτελέσματα δίνεται στην επόμενη ενότητα.

Κώδικας

```
import numpy as np
import matplotlib.pyplot as plt

# coefficient definition
def A(theta,dx):
    return dx*np.sin(theta)/2

def B(theta,dy):
    return dy*np.cos(theta)/2

N = np.array([5,10,20])
dx = 1/N
dy = 1/N
dt = 0.5*dx
c = 1 #light speed
S1 = c*dt/dx  # Courant factor for dx=dy
C1 = (np.sin(np.pi*S1/N))**2/S1**2
l = 1 # wavelength
up = np.zeros((len(N),91)) # phase velocity

```

```
# numerical phase velocity for dx=dy
for i in range(len(N)):
    for theta in range(0,91):
        thetar = theta*np.pi/180
        k0 = 2*np.pi/l
        e = 1;
        while (abs(e)>0.0005):
            k = k0 - (np.sin(A(thetar,dx[i])*k0)**2 + np.sin(B(thetar,dy[i])*k0)**2-C1[i])/(A(thetar,dx[i])*np.sin(2*A(thetar,dx[i])*k0)+B(thetar,dy[i])*np.sin(B(thetar,dy[i])*k0))
            e = k-k0
            k0 = k
        up[i,theta] = (2*np.pi/k)

x = np.arange(0,91,1)
plt.clf()
plt.plot(x,up[0,:],'r',label='lambda/5')
plt.plot(x,up[1,:],'g',label='lambda/10')
plt.plot(x,up[2,:],'b',label='lambda/20')
plt.xlabel('Wave angle (degrees)')
plt.ylabel('Phase velocity up/c')
plt.legend()
plt.savefig('fig1.jpeg')

# numerical phase velocity for dx!=dy
N = np.array([5,10])
dx = 1/N
dy = dx/2
dt = 0.5*dx
S2 = c*dt/dx + c*dt/dy # Courant factor for dx!=dy
C2 = (np.sin(np.pi*S2/N))**2/S2**2
up = np.zeros((len(N),91))

for i in range(len(N)):
    for theta in range(0,91):
        thetar = theta*np.pi/180
        k0 = 2*np.pi/l
        e = 1;
        while (abs(e)>0.0005):
            k = k0 - (np.sin(A(thetar,dx[i])*k0)**2 + np.sin(B(thetar,dy[i])*k0)**2-C2[i])/(A(thetar,dx[i])*np.sin(2*A(thetar,dx[i])*k0)+B(thetar,dy[i])*np.sin(B(thetar,dy[i])*k0))
            e = k-k0
            k0 = k
        up[i,theta] = (2*np.pi/k)

plt.clf()
plt.plot(x,up[0,:],'r',label='lambda/5')
plt.plot(x,up[1,:],'g',label='lambda/10')
plt.xlabel('Wave angle (degrees)')
plt.ylabel('Phase velocity up/c')
plt.legend()
plt.savefig('fig2.jpeg')
```