Eksamensoppgave 2 - INF5620

Henrik Andersen Sveinsson

December 7, 2013

1 Oppgavetekst

a

Set up a wave equation problem in 2D with zero normal derivative as boundary condition. Assume a variable wave velocity.

Mention a physical problem where this mathematical model arises. Explain the physical interpretation of the unknown function.

b

Present a finite difference discretization. Explain in particular how the boundary conditions and the initial conditions are incorporated in the scheme.

c

Explain (in princple) how the 2D discretization can be extended to 3D.

d

Set up the stability condition in 3D. Also quote results on about accuracy of the method in 3D and define the accuracy measure(s) precisely.

e

Explain how you can verify the implementation of the method.

f

The scheme for the wave equation is perfect for parallel computing. Why? What are the principal ideas behind a parallel version of the scheme?

2 Bølgelikningen på grunt vann i 2D

$$\frac{\partial^2}{\partial t^2}u(t,x,y) = \frac{\partial}{\partial x}\left(q(x,y)\frac{\partial}{\partial x}u\right) + \frac{\partial}{\partial y}\left(q(x,y)\frac{\partial}{\partial y}u\right) \tag{1}$$

Denne likningen oppstår ved modellering av tsunamier. Jeg har antatt en konstant tetthet, slik at den kan inngå i funksjonen q. Den ukjente funksjonen

u er vannivået relativt til gjennomsnittlig vannivå. $q(x,y) = v(x,y)^2$ der v er bølgehastigheten. Grensebetingelsen som vi vil sette opp er $\frac{\partial u}{\partial n} = 0$, som er en Neumannbetingelse.

3 Finite difference diskretisering

Her velger jeg å bruke tilfellet at q er konstant, for å få litt mindre å skrive. Jeg tenker det er fornuftig når det kun er 15 minutter. (Spørre HPL om det er greit å anta dette her) Da ser likningen slik ut:

$$u_{tt} = c^2(u_{xx} + u_{yy}) \tag{2}$$

Vi velger en sentert differanse i alle ledd, med $\Delta x = \Delta y = h$

$$\frac{u_{i,j}^{n+1} - 2u_{i,j}^{n} + u_{i,j}^{n-1}}{\Delta t^2} = c^2 \frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta h^2} \tag{3}$$

$$u_{i,j}^{n+1} = 2u_{i,j}^{n} - u_{i,j}^{n-1} + \frac{c^2 \Delta t^2}{h^2}\left(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n} - 4u_{i,j}^{n}\right) \tag{4}$$

Dette git en metode for å finne tilstanden i et tidssteg, gitt alle funksjonsverdiene i de to forrige tidsstegene. Men vi ser at man kun finner funksjonsverdier for punkter (i, j), men for å finne disse trenger vi punktene rett utenfor randen. Her kommer randbetingelsene inn i bildet. Vi skal se på to typer grensebetingelser. For grunt vann-likninger er nok Neumannbetingelsen den mest relevante, men vi skal også se på Dirichletbetinglesen.

3.1 Dirichletbetingelsen

Den enkleste randbetingelsen man kan tenke seg:

$$u(\mathbf{x}) = f(\mathbf{x}) \forall \mathbf{x} \in \partial\Omega \tag{5}$$

Dette løser problemet vi hadde på randen. Dersom vi har en rand med kjente verdier trenger vi bare å regne ut verdiene strengt innenfor denne, og vi har dermed alltid tilgang til verdier som ligger ett hakk "utenfor" de verdiene vi regner ut.

3.2 Neumannbetingelsen

Neumannbetingelsen går på den deriverte på randen, og vi setter $\frac{\partial u}{\partial n} = f(\mathbf{x})$ på $\partial\Omega$. Dersom vi ser på områder som kun har normal i koordinatretningene, er det greit å takle disse randbetingelsene også, ved å gjøre en endelig differansetilnærming til randbetingelsen:

$$\frac{u_{i+1} - u_{i-1}}{2h} = f(u_i) \tag{6}$$

Dette gir

$$u_{i+1} = u_{i-1} + 2hf(u_i) \tag{7}$$

For høyre og øvre grenseflate, og

$$u_{i-1} = u_{i+1} - 2hf(u_i) \tag{8}$$

For venste og nedre grenseflate. Her evaluerer man i prinsippet punkter som ligger utenfor domenet man jobber på, så dette legges inn før man regner funksjonsverdier på det indre av domenet.

3.3 Initialbetingelse

For å løse bølgelinkningen trenger man to initialbetingelser i hvert punkt: Funksjonsverdien og den første deriverte. Da kan man lage et "Ghost timestep" for tidssteg -1.

Vi kjører en differansemetode på den deriverte av initialbetingelsen:

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n-1}}{2\Delta t} = V_{i,j} \tag{9}$$

$$u_{i,j}^{-1} = u_{i,j}^{1} - 2\Delta t V_{i,j} \tag{10}$$

Vi setter dette inn i det generelle skjemaet, og får for u^1

$$u_{i,j}^{1} = 2u_{i,j}^{0} - (u_{i,j}^{1} - 2\Delta t V_{i,j}) + \frac{c^2 \Delta t^2}{h^2}\left(u_{i+1,j}^{0} + u_{i-1,j}^{0} + u_{i,j+1}^{0} + u_{i,j-1}^{0} - 4u_{i,j}^{0}\right) \tag{11}$$

Innsatt for initialbetingelse og snudd litt på, får vi:

$$u_{i,j}^{1} = I_{i,j} + 2\Delta t V_{i,j} + \frac{c^2 \Delta t}{h^2}(I_{i+1,j} + I_{i-1,j} + I_{i,j+1} + I_{i,j-1} - 4I_{i,j}) \tag{12}$$

Her har vi eliminert dette "Ghost timestep" -1, så nå kan man regne ut u^1, og deretter kjøre algoritmen vanlig.

4 Utvidelse til 3D

Utvidelsen til 3D er nærmest triviell. Man legger på hele del-operatoren i likningen:

$$u_{tt} = c^2 \nabla^2 u \tag{13}$$

5 Stabilitet i 3D

6 Verifikasjon

7 Parallellisering

Skjemaet er perfekt for parallellisering, fordi man vet alt man trenger for neste tidssteg direkte i et tidssteg, og fordi man kun trenger informasjon for noder i nærheten for å regne oppførsel i neste tidssteg.