8. En fase i to dimensjoner

Utvid programmet til også å inkludere strøm i to dimensjoner for en fase i et horisontalt system.

- Bruk implisitt formulering
- Løs ligningssettet med LSOR eller eliminasjon og sammenlign
- Skriv programmet slik at det fremdeles kan simulere strøm i en dimensjon dersom `mx = 1` eller `my = 1`
- Dimensjoner til et 5*5 system
- Testdata er

  `mx = my =5`
  Δx `= 10, i = 1, mx`
  Δy `= 10, j = 1, mx`
  Δz `= 1`
  `kx = 100,` alle blokker
  `ky = 100,` alle blokker
  `qo(1,1) = - 1.0`
  `qo(5,5) = 1.0`
  ϕ `= 0.05,` alle blokker
  `stmax = 5`
  væskeegenskaper som i Øving 7
  `delmin, delmax, dtmult, dpmax` som i øving 4

- Dersom programmet fungerer riktig skal følgende være oppfylt:
  - Materialbalansen er iorden
  - Trykkene er symmetriske og antisymmetriske om diagonalene i rutenettet

Kommentarer

For strøm av en fase i en dimensjon har vi fra før

$$\frac{\partial}{\partial x}\left(\frac{ckb}{\mu}\frac{\partial p}{\partial x}\right)+q=\phi\frac{db}{dp}\frac{\partial p}{\partial t},$$

og utvidet til to dimensjoner får en etter en helt tilsvarende utledning som for en dimensjon

$$\frac{\partial}{\partial x}\left(\frac{ckb}{\mu}\frac{\partial p}{\partial x}\right)+\frac{\partial}{\partial y}\left(\frac{ckb}{\mu}\frac{\partial p}{\partial y}\right)+q=\phi\frac{db}{dp}\frac{\partial p}{\partial t}.$$

Også helt analogt til en dimensjon gir nå diskretiseringen

$$ox^{+}\Delta p_x^{+}+ox^{-}\Delta p_x^{-}+oy^{+}\Delta p_y^{+}+oy^{-}\Delta p_y^{-}+a9=a2(p_{i,j}-p_{i,j}(t))/\Delta t. \qquad . \tag{1}$$

I denne ligningen er $\Delta p_x^{+}=p_{ip,j}-p_{i,j}$, trykkdifferansen mellom to naboblokker i x -retning, ip = i +1 , trykk $p_{i,j}(t)$ er tatt ved gammel tid t, mens de andre trykkene er ved ny tid $t+\Delta t$, (implisitt formulering). Variabelen i angir blokk nummer i x -retning, og j i y -retning:

	(i,jp)	
(im,j)	(i,j)	(ip,j)
	(i,jm)	

Setter en inn i ligning 1 for trykkdifferansene får en

$$\begin{aligned} ox_{i,j}^{+}(p_{ip,j}-p_{i,j})+ox_{i,j}^{-}(p_{im,j}-p_{i,j})+ & \\ oy_{i,j}^{+}(p_{i,jp}-p_{i,j})+oy_{i,j}(p_{i,jm}-p_{i,j})+a9_{i,j} &= a2(p_{i,j}-p_{i,j}(t))/\Delta t. \end{aligned} \tag{2}$$

Samler en nå ledd med like trykk får en

$$\begin{aligned} ox_{i,j}^{-}\ p_{im,j} & \\ +\,oy_{i,j}^{-}\ p_{i,jm} & \\ -\,(ox_{i,j}^{-}+ox_{i,j}^{+}+oy_{i,j}^{-}+oy_{i,j}^{+}+a2_{i,j}/\Delta t)\ p_{i,j} & \\ +\,ox_{i,j}^{+}\ p_{ip,j} & \\ +\,oy_{i,j}^{+}\ p_{i,jp} &= -(a2_{i,j}p_{i,j}t/\Delta t+a9_{i,j}). \end{aligned}$$

Nå settes, for hver (i,j), $bb=ox^{-}+ox^{+}+oy^{-}+oy^{+}+a2/\Delta t$, og en definerer

$$\begin{aligned} \mathtt{a(i,j)} &= ox_{i,j}^{-}/bb, \\ \mathtt{c(i,j)} &= ox_{i,j}^{+}/bb, \\ \mathtt{e(i,j)} &= oy_{i,j}^{-}/bb, \\ \mathtt{f(i,j)} &= oy_{i,j}^{+}/bb, \\ \mathtt{d(i,j)} &= (a2_{i,j}\ p_{i,j}(t))/\Delta t+a9_{i,j})/bb, \end{aligned}$$

og ligning 1 blir

$$\mathtt{a}(\mathtt{i},\mathtt{j})\mathtt{p}(\mathtt{im},\mathtt{j})+\mathtt{c}(\mathtt{i},\mathtt{j})\mathtt{p}(\mathtt{ip},\mathtt{j})-\mathtt{p}(\mathtt{i},\mathtt{j})+\mathtt{e}(\mathtt{i},\mathtt{j})\mathtt{p}(\mathtt{i},\mathtt{jm})+\mathtt{f}(\mathtt{i},\mathtt{j})\mathtt{p}(\mathtt{i},\mathtt{jp})=-\mathtt{d}(\mathtt{i},\mathtt{j}). \tag{3}$$

For å løse ligning 3 ved eliminasjon, må en først velge et nummereringssystem for blokkene. Dette kan gjøres på to hovedmåter,

- Naturlig nummerering,
- Alternerende diagonal nummerering.

La oss først se på naturlig nummerering, og vi bruker som eksempel et 5*4 blokksystem, nummerert som vist under

4	8	12	16	20
3	7	11	15	19
2	6	10	14	18
1	5	9	13	17

Vi antar at alle `a(1,j), c(mx,j), e(i,1), f(i,my)` er lik null ved at de tilsvarende $\mathtt{ox}^{\pm}, \mathtt{oy}^{\pm}$ er satt lik null. Fra ligning 3 får en da det lineaære ligningssystemet

$$\overset{\Rightarrow}{K}\cdot\vec{p}^t = \vec{d}^t,$$

hvor koeffisientmatrisen $\overset{\Rightarrow}{K}$ er gitt ved

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-1	f_1			c_1															
2	e_2	-1	f_2			c_2														
3		e_3	-1	f_3			c_3													
4			e_4	-1				c_4												
5	a_5				-1	f_5			c_5											
6		a_6			e_6	-1	f_6			c_6										
7			a_7			e_7	-1	f_7			c_7									
8				a_8			e_8	-1				c_8								
9					a_9				-1	f_9			c_9							
10						a_{10}			e_{10}	-1	f_{10}			c_{10}						
11							a_{11}			e_{11}	-1	f_{11}			c_{11}					
12								a_{12}			e_{12}	-1				c_{12}				
13									a_{13}				-1	f_{13}			c_{13}			
14										a_{14}			e_{14}	-1	f_{14}			c_{14}		
15											a_{15}			e_{15}	-1	f_{15}			c_{15}	
16												a_{16}			e_{16}	-1				c_{16}
17													a_{17}				-1	f_{17}		
18														a_{18}			e_{18}	-1	f_{18}	
19															a_{19}			e_{19}	-1	f_{19}
20																a_{20}			e_{20}	-1

og $\vec{p}^t$ er søylevektoren av de ukjente trykkene og $\vec{d}^t$ er søylevektoren av leddene på høyre siden av ligning 3. Koeffisientmatrisen $\overset{\Rightarrow}{K}$ er pentadiagonal med nuller utenom de 5 diagonalene. Båndbredden er `2*my + 1 = 9`. En teller hurtigst i den retning som har minst blokker for å få minst båndbredde.

I prinsipp kan dette lineære ligningssettet løses på samme måte som ved *tridia*: en skaffer først nuller under diagonalen mens en arbeider seg nedover, og går så oppover igjen.

Ved *alternerende diagonal nummerering*, med samme 5*4 system, velges følgende nummereringsmønster:

13	5	17	9	20
2	14	6	18	10
11	3	15	7	19
1	12	4	16	8

Koeffisientmatrisen $\overset{\Rightarrow}{K}$ blir da

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-1										f_1	c_1								
2		-1									e_2		f_2	c_2						
3			-1								a_3	e_3		f_3	c_3					
4				-1								a_4			f_4	c_4				
5					-1								X	X			X			
6						-1								X	X		X	X		
7							-1								X	X		X	X	
8								-1								X			X	
9									-1								X	X		X
10										-1								X	X	X
11	X	X	X								$\oplus$	$\oplus$		$\oplus$	$\oplus$					
12	X		X	X							$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$					
13		X			X						$\oplus$	$\oplus$	$\oplus$	$\oplus$			$\oplus$			
14		X	X		X	X					$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$		$\oplus$	$\oplus$		
15			X	X		X	X				$\oplus$	$\oplus$		$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	
16				X			X	X				$\oplus$			$\oplus$	$\oplus$		$\oplus$	$\oplus$	
17					X	X			X				$\oplus$	$\oplus$	$\oplus$		$\oplus$	$\oplus$		$\oplus$
18						X	X		X	X				$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$
19							X	X		X					$\oplus$	$\oplus$		$\oplus$	$\oplus$	$\oplus$
20									X	X							$\oplus$	$\oplus$	$\oplus$	$\oplus$

Matrisen er ikke skrevet helt ut, og X betegner et ledd av typen `a,c,e,f`. I nedre høyre kvadrant er det før eliminering kun tallet 1 på diagonalen som er forskjellig fra null.

En skaffer seg nå nuller under diagonalen ved å eliminere leddene i nedre venstre kvadrant. Dette medfører nye ledd i nedre høyre kvadrant —her er endrete ledd angitt med symbolet $\oplus$. Disse nye leddene kan, med litt omtanke, settes opp direkte. Deretter løser en for trykkene til blokkene 11-20 ved å skaffe nuller under diagonalen i nedre høyre kvadrant etc. Til slutt løses for trykkene 1-10 ved å bruke opprinnelig ligning. Båndbredden er `2*my +1 = 9`, som før.

Løsning med naturlig nummerering

```
      subrutine solve(c,b,ibdwth,n)
      implicit double precision(a-h,o-z)
c-----------------------------
c*** n: antall ligninger, n = my*mx
c*** ibdwth: båndbredden, enten lik 2*my + 1 eller
c... lik 2*mx + 1, avhengig av hva som er minst
c... For et 5*5 system er ibdwth = 11
c*** c: inneholder høyre siden av ligningen, konstantleddet
c... For et 5*5 system: c(25)
c*** b: koeffisientmatrisen, kun båndet er lagret
c... For et 5*5 system: b(25,11)
c-------------------------------------
      dimension c(n), b(n,ibdwth)
      ihalf = (ibdwth + 1)/2
      ihafm2 = ihalf - 2
      ihafm1 = ihalf - 1
      nm1 = n - 1
      index = 0
c
c-- Her begynner eliminasjonen under diagonalen
c
```

```
   do 60 k = 1, nm1
      if(dabs(b(k,ihalf)).lt.1.d-19) go to 25
      limit = k + ihafm2
      if (limit.gt.nm1) limit = nm1
      l = ihalf + index
      jlim = ibdwth + index
      do 50 i = k, limit
         lmi = l - i
         ip1 = i + 1
         if (b(ip1,lmi).eq.0.) go to 50
         factor = - b(ip1,lmi)/b(k,ihalf)
         m = ihafm1 + k
         do 30 j = m, jlim
            if (dabs(factor).lt.1.d-30) go to 30
            jmndx = j - index
            if (dabs(b(k,jmndx)).lt.1.d-30) go to 30
            jmi = j - i
            b(ip1,jmi) = factor * b(k,jmndx) + b(ip1,jmi)
30          continue
         b(ip1,lmi) = factor
50       continue
      index = index + 1
60  continue
c
c-Ferdig med å gjøre koeffisientmatrisen øvre triangulær
c-Justerer nå høyre siden av ligningen på tilsvarende måte
c
```

```
    index = 0
    do 61 K = 1, nm1
       limit = k + ihafm2
       l = ihalf + index
       if (limit.gt.nm1)limit = nm1
          do 51 i = k, limit
          ip1 = i + 1
          c(ip1) = c(ip1) + b(ip1,l-i)*c(k)
51        continue
       index = index + 1
61  continue
c
c-Regner ut trykket i siste blokken, og går så oppover
c-langs diagonal igjen.
c
    c(n) = c(n)/b(n,ihalf)
c

   do 70 i = 1, nm1
      temp = 0.d0
      isub = n-i
      do 80 j = 1, ihafm1
         if (j.gt.i) go to 70
80       temp = temp + b(isub,ihalf+j)*c(isub+j)
70  c(isub) = (c(isub) - temp)/b(isub,ihalf)
   go to 26
25  write(6,27)
27  format(1h0,10x,' **** sorry, your pivot element is too small ***')
   stop
26  return
   end
```

Før en kan kalle opp SOLVE og løse ligningssystemet, må en sette opp matrisene `bb(n,ibdwth)` og `cc(n)`. Vi bruker her `bb` og `cc` for ikke å komme i konflikt med andre symboler i hovedprogrammet. Vi antar at $\mathtt{my} \leq \mathtt{mx}$. Merk at `ih` er posisjonen til diagonalleddet i det båndet som er lagret i `bb`-matrisen. Først nullstiller vi båndmatrisen:

```
      ibdwth = 2*my + 1
      ih = (ibdwth +1)/2
      ic = 0
      do 10 i = 1, mx
         do 20 j = 1, my
            ic = ic + 1
            do 30 ii = 1, ibdwth
               bb(ic,ii) = 0.d0
 30         continue
 20      continue
 10   continue
```

```
      ic = 0
      do 40 i = 1, mx
         do 50 j = 1, my
            ic = ic + 1
            bb(ic,1) = a(i,j)
            bb(ic,ih-1) = e(i,j)
            bb(ic,ih) = -1.d0
            bb(ic,ih+1) = f(i,j)
            bb(ic,ibdwth) = cc(i,j)
            cc(ic) = - d(i,j)
 50      continue
 40   continue
      n = mx*my
      call solve(cc,bb,ibdwth,n)
```

Løsningen returneres fra solve i matrisen `cc` og vi må fiske ut de todimensjonale trykkene fra denne endimensjonale matrisen:

```
      ic = 0
      do 60 i = 1, mx
         do 70 j = 1, my
            ic = ic + 1
            p(i,j) = cc(ic)
 70      continue
 60   continue
```

Subrutinen `solve` bruker samme framgangsmåte som en har i `tridia`.

I boken til Aziz og Settari er det reprodusert kode til en annen løsningsrutine som en også kan bruke. Denne er basert på endimensjonale matriser.

Løsning med LSOR

I denne metoden, *Line SOR*, brukes SOR linjevis med kall til *tridia* for hver linje. Vi kan velge mellom å sveipe i x - eller y -retning, eller en blanding.

Denne metoden er mye brukt, spesielt i tredimensjonale modeller. Dens anvendelighet avhenger blant annet av et godt estimat på ω, og at reservoaret har liten anisotropi.

Vi ser på ligningen for blokk (i,j) og sløyfer disse indeksene på koeffisientene:

$$\mathtt{a}\,\mathtt{p}_{\mathtt{im},\mathtt{j}} + \mathtt{c}\,\mathtt{p}_{\mathtt{ip},\mathtt{j}} - \mathtt{p}_{\mathtt{i},\mathtt{j}} + \mathtt{e}\,\mathtt{p}_{\mathtt{i},\mathtt{jm}} + \mathtt{f}\,\mathtt{p}_{\mathtt{i},\mathtt{jp}} = -\mathtt{d}.$$

Dersom vi nå sveiper i x -retning, så skriver vi denne ligningen på formen

$$\mathtt{a}\,\mathtt{p}_{\mathtt{im},\mathtt{j}} - \mathtt{p}_{\mathtt{i},\mathtt{j}} + \mathtt{c}\,\mathtt{p}_{\mathtt{ip},\mathtt{j}} = -\mathtt{e}\,\mathtt{p}_{\mathtt{i},\mathtt{jm}} - \mathtt{f}\,\mathtt{p}_{\mathtt{i},\mathtt{jp}} - \mathtt{d}.$$

For $\mathtt{p}_{\mathtt{i},\mathtt{jm}}$ og $\mathtt{p}_{\mathtt{i},\mathtt{jp}}$ på høyre side av ligningen bruker vi de sist itererte verdiene. Dermed kan vi uttrykke høyresiden med et tall, og bruker *tridia* til å løse for de ukjente trykkene på venstre siden.

```
c
c- start iterasjonssløyfe for LSOR
c
   do 10 kkk = 1, 20
      isw = 0
      jm = 1
      do 20, j = 1, my
         if (j.gt.1) jm = j - 1
         if (j.lt.my) jp = j + 1
         im = 1
         do 30 i = 1, mx
            if(i.gt.1) im = i - 1
            if(i.lt.1) ip = i + 1
            aa(i) = a(i,j)
            cc(i) = c(i,j)
            dd(i) = d(i,j) + e(i,j)*p(i,jm) + f(i,j)*p(i,jp)
30       continue
         call tridia(mx, ···)
c
c -- Her overføres aa, cc og dd til tridia, og løsningen returneres
c -- i pp. Må derfor endre common block i tridia
c
         do 40 i = 1, mx
            p(i,j) = p(i,j) + om*(pp(i) - p(i,j))
            if (dabs(p(i,j) - pp(i)).gt. eps) isw = 1
40       continue
20    continue
   kkkcum = kkk
   if (isw.eq.0) hopp ut og skriv kkkcum
10 continue
```

Den teoretiske bakgrunnen for *LSOR* kan en finne i boken til Aziz og Settari.

Noen påminnelser

- De fleste "gamle" størrelsene får nå doble indekser og må dimensjoneres. Dette gjelder f.eks. `kx(5,5), phi(5,5), p(5,5), dex(5), ckx(5,5), etc.`
- `kx, ky, phi, q,` etc. leses og skrives for hver blokk
- Nye størrelser deklareres, som f.eks. `oyplus(5,5), oymin(5,5), dey(5), cky(5,5), ky(5,5), bb(25,11), cc(25), aa(5), cc(5), dd(5), pp(5)`

- De fleste “gamle” sløyfer med telleren `i` blir nå doble med både `i` og `j`.