MEK 1100 Obligatorisk oppgave 2

Kenneth Ramos Eikrehagen

27. april 2017

Innhold

Oppgave 1

a)

Jeg laget først en kode som henter dataene vi har fått oppgitt. Denne laget jeg for seg selv siden jeg importer dette programmet videre i andre koder slik at jeg slepper å skrive de samme kommandoene flere ganger.
Legger ved koden under:

```
1  import scipy.io as sio
2  from pylab import*
3
4  data = sio.loadmat("data.mat")
5  x = data.get("x")
6  y = data.get("y")
7  u = data.get("u")
8  v = data.get("v")
9  xit = data.get("xit")
10 yit = data.get("yit")
```

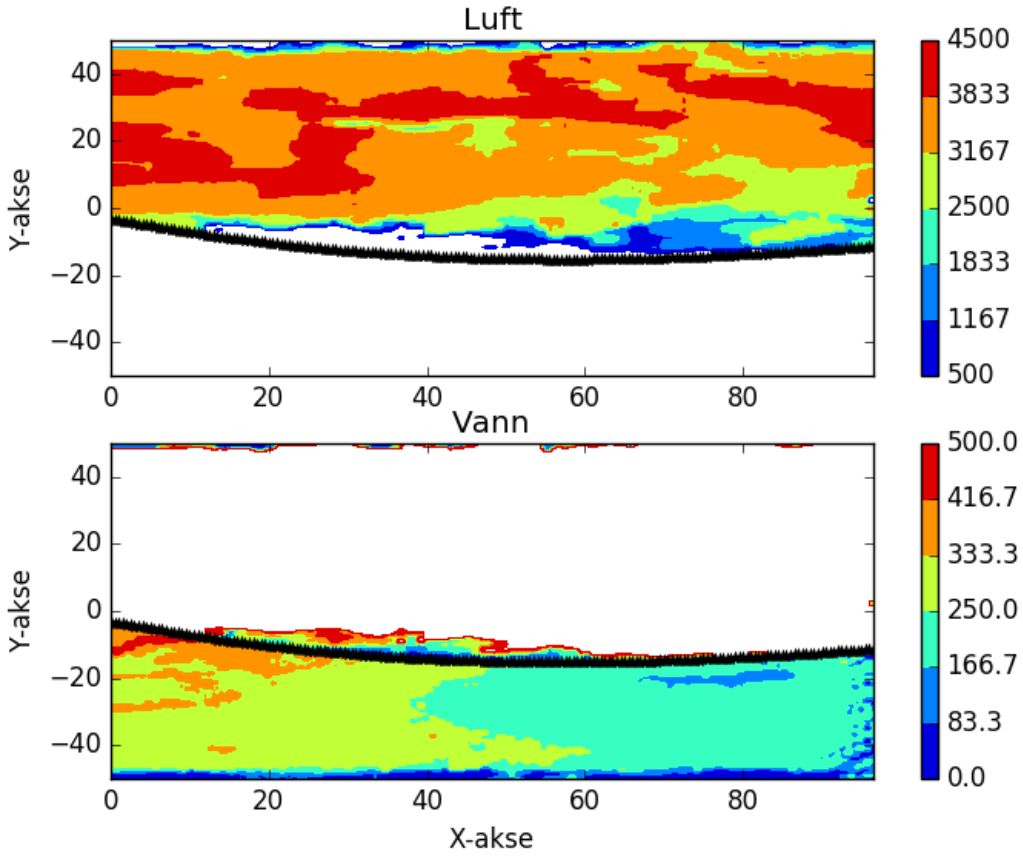
Neste kode sjekker om matrisene x, y, u, v og vektorene xit og yit hadde de oppgitte verdiene. Legger ved koden.

```
1  from oblig2 import*
2  #Legger matriseme og vektorene i liste
3  dat = [x,y,u,v,xit,yit]
4  #Sjekker storrelsen til matrisene og vektorene
5  for j in range(6):
6      value = []
7      indx = ['x','y','u','v','xit','yit']
8      for i in dat:
9          value += [shape(i)]
10     if j < 4:
```

```
11          print 'Matrisen %s har (x,y)' %(indx[j])
12          print value[j]
13      elif j >= 4:
14          print 'Vektoren %s har (x,y)' %(indx[j])
15          print value[j]
16  #test funksjoner.
17  #sjekker om x regulert med et intervall paa 0.5
18  def test_x(q):
19      for i in q:
20          for j in range(194-1):
21              tst = i[j+1]-i[j]
22              sucsess = 0
23              if tst == 0.5:
24                  sucsess
25              else :
26                  print 'something went wrong'
27  #sjekker om y er regulert med et intervall paa 0.5
28  #og om den tar hele diameteren til roret
29  def test_y(q):
30      ykor = []
31      for i in y:
32          ykor += [i[1]]
33          if abs(i[1]) > 50:
34              print 'Overstiger diameteren!'
35          else:
36              ykor
37      for j in range(194):
38          tst = ykor[j+1]-ykor[j]
39          success = 0
40          if tst == 0.5:
41              success
42          else :
43              print 'this is not right'
44  #kaller paa testfunksjonene
45  test_x(x)
46  test_y(y)
```

Koden printer følgende i terminalen
Terminal: python oblig2.py
Matrisen x har (x,y)
(201, 194)
Matrisen y har (x,y)
(201, 194)
Matrisen u har (x,y)
(201, 194)
Matrisen v har (x,y)
(201, 194)
Vektoren xit har (x,y)
(1, 194)
Vektoren yit har (x,y)
(1, 194)

b)



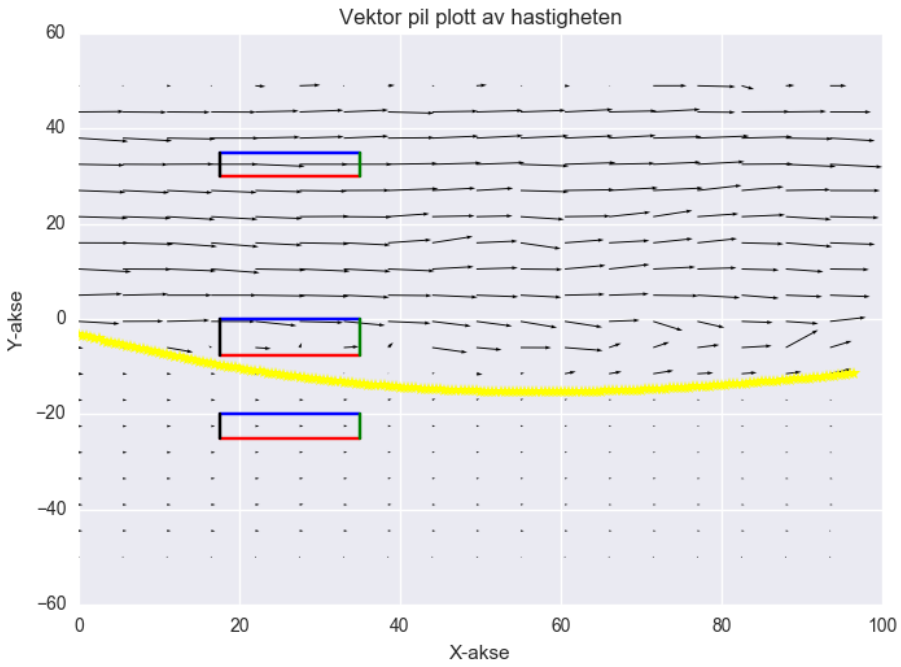
Figur 1: Konturplott av luft og vann. Øverst Luft, nederst vann

For å kunne lage et konturplott av hastighetskomponentene i xy-planet bruker jeg kommandoen countourf(x,y,z) den trenger 3 variabler. Jeg tolker x og y matrisene som punkter i et gridd der x matrisen er x koordinatene og y matrisen er y koordinatene. Jeg lager deretter en variable $z = sqrt(u^2 + v^2)$ der u og v er matrisene som inneholder komponentene til hastighets feltet. På figur 1 ser du konturplottene mine. Jeg har manuelt redigert verdiene slik at man enkelt kan se strukturen i gass- og væskefasen. Den svarte linjen som går gjennom plottene skal demonstrere skilleflaten. Legger ved python koden min under:

```
1 from oblig2 import*
2 #Definerer en variabel til konturplott
```

```
Z = sqrt(u**2+v**2)
v1 = linspace(500,4500,7)
v2 = linspace(0,500,7)
#kontur plott
subplot(2,1,1) #Luft
c=contourf(x,y,Z,v1)
colorbar(c)
plot(xit,yit,'*',color='black')
title('Luft')
ylabel('Y-akse')
subplot(2,1,2) #Vann
cs=contourf(x,y,Z,v2)
colorbar(cs)
plot(xit,yit,'*',color='black')
title('Vann')
xlabel('X-akse')
ylabel('Y-akse')
```

c)



Figur 2: Hastighetsfelt

Jeg brukte kommandoen quiver for å tegne pilene i hastighetsfeltet. Jeg valgte å plotte en pil per 11 punkt, jeg syntes dette ga ett fint inntrykk av hvordan strømmene er i røret.For å skille fluidene har jeg brukt gult for å merke skilleflaten i xy-planet. Jeg har også laget en egen funksjon som tar

inn x og y koordinatene og tegner tilhørende rektangel med farge kodene som ble gitt. Du kan se plottet på figur 2 på forrige side
Legger ved python-koden under.

```
#pilplott
n=11
quiver(
    x[::n, ::n],
    y[::n, ::n],
    u[::n, ::n],
    v[::n, ::n],
    units='width', width = 0.0015)
#Rektanglene
def rektangel(xi,yi,xj,yj):
    x1 = x[yi][xi]; x2 = x[yj][xj]
    y1 = y[yi][xi]; y2 = y[yj][xj]
    plot([x1,x2],[y1,y1], color='red')
    plot([x2,x1],[y2,y2], color='blue')
    plot([x1,x1],[y1,y2], color='black')
    plot([x2,x2],[y2,y1], color='green')
rektangel(35,160,70,170)
rektangel(35,85,70,100)
rektangel(35,50,70,60)
```

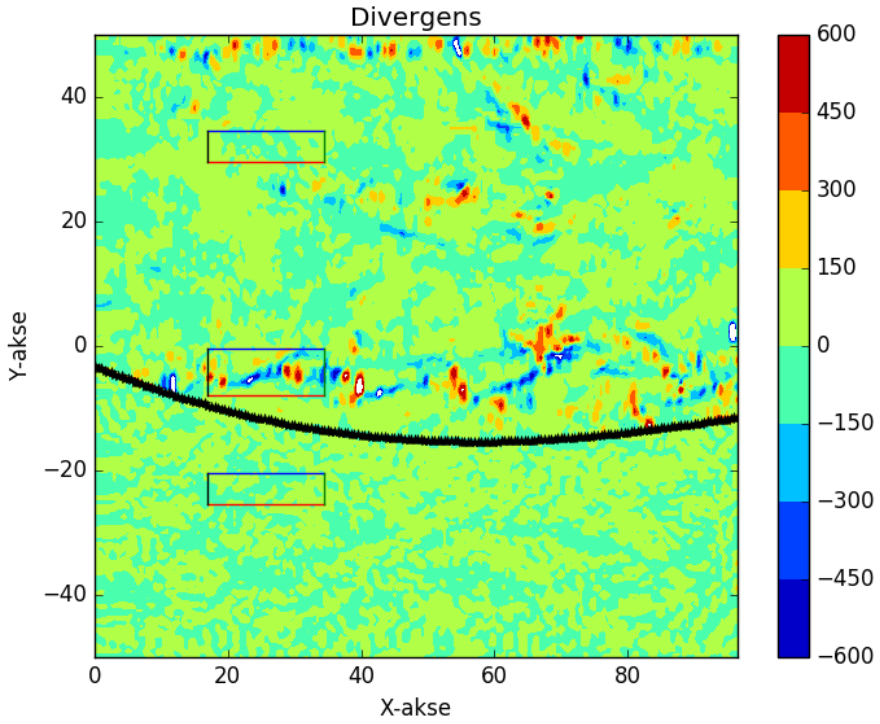
d)

For å regne ut divergensen til $u\mathbf{i}+v\mathbf{j}$ bruker jeg at $div(u\mathbf{i}+v\mathbf{j}) = \nabla\cdot(u\mathbf{i}+v\mathbf{j})$

$$\nabla\cdot(u\mathbf{i}+v\mathbf{j}) = (\mathbf{i}\frac{\partial}{\partial x}+\mathbf{j}\frac{\partial}{\partial y})\cdot(u\mathbf{i}+v\mathbf{j}) = \frac{\partial}{\partial x}u+\frac{\partial}{\partial y}v$$

Jeg har kodet dette i python, der kan jeg bruke kommandoen $gradient()$ som regner ut gradienten til matrisen jeg sender inn. Jeg bruker også kommandoen axis som gjør at jeg kan velge hvilken akse jeg vil ha. ($axis = 1 \Rightarrow x$-akse, $axis = 0 \Rightarrow y$-akse). Du kan se plottet mitt på figur 3 på neste side
Legger ved koden under:

```
#divergens funksjon
def div(F,G):
    return gradient(F,axis=1)+gradient(G,axis=0)
#konturplott
verdier = linspace(-600,600,9)
contourf(x,y,div(u,v),verdier)
colorbar()
#skilleflate
plot(xit,yit,'*',color='black')
#Rektanglene
def rektangel(xi,yi,xj,yj):
    x1 = x[yi][xi]; x2 = x[yj][xj]
    y1 = y[yi][xi]; y2 = y[yj][xj]
    plot([x1,x2],[y1,y1], color='red')
```



Figur 3: Divergens

```
15      plot([x2,x1],[y2,y2], color='blue')
16      plot([x1,x1],[y1,y2], color='black')
17      plot([x2,x2],[y2,y1], color='green')
18  rektangel(34,159,69,169)
19  rektangel(34,84,69,99)
20  rektangel(34,49,69,59)
```

$div(u\mathbf{i} + v\mathbf{j})$ er ikke divergensen til $\mathbf{v}$ fordi vi mangler den siste komponenten til $\mathbf{v}$. Divergensen til $\mathbf{v}$ er:

$$\begin{aligned} div(\mathbf{v}) &= \nabla \cdot \mathbf{v} \\ &= (\mathbf{i}\frac{\partial}{\partial x} + \mathbf{j}\frac{\partial}{\partial y} + \mathbf{k}\frac{\partial}{\partial z}) \cdot (u\mathbf{i} + v\mathbf{j} + w\mathbf{k}) \\ &= \frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v + \frac{\partial}{\partial z}w \end{aligned}$$

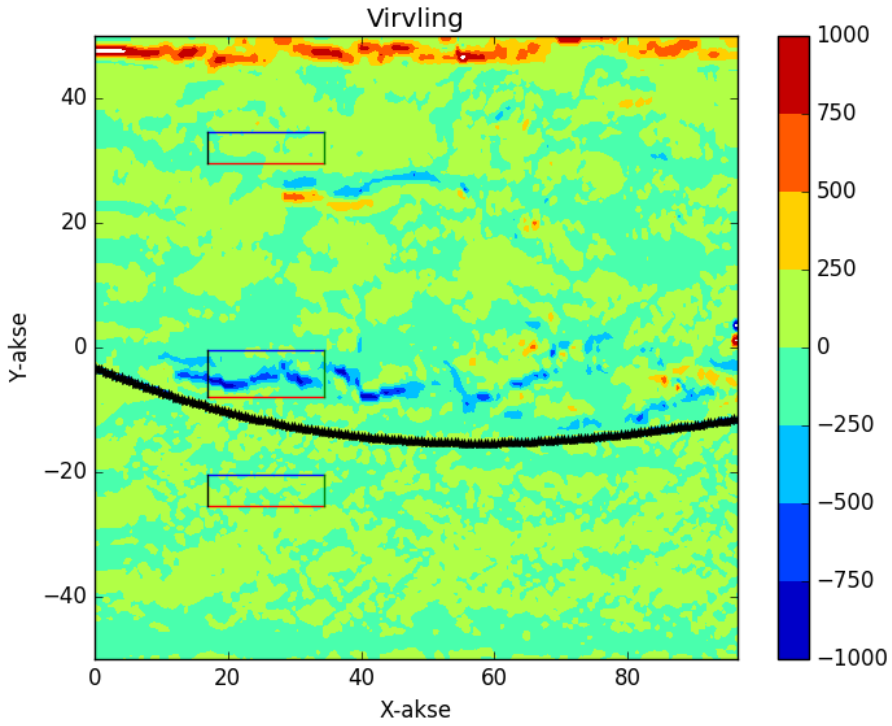
Siden fluidene er inkompressible betyr dette at fluidene har konstant tetthet som medfører at divergensen til hastighetsfeltet er null.

$$div(\mathbf{v}) = \nabla \cdot \mathbf{v} = 0$$

Med dette kan vi nå finne et uttrykk for w.

$$
\begin{aligned}
div(\mathbf{v}) = \nabla \cdot \mathbf{v} &= 0 \\
\frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v + \frac{\partial}{\partial z}w &= 0 \\
\frac{\partial}{\partial z}w &= -(\frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v) \\
w &= -\int (\frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v)dz
\end{aligned}
$$

e)



Figur 4: Virvling normalt på xy-planet

For å finne virvlinga til $\mathbf{v}$ bruker jeg at $curl(\mathbf{v}) = \nabla \times \mathbf{v}$.

$$\begin{aligned} curl(\mathbf{v}) &= \nabla \times \mathbf{v} \\ &= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ u & v & 0 \end{vmatrix} \\ &= -\mathbf{i}\frac{\partial}{\partial z}v + \mathbf{j}\frac{\partial}{\partial z}u + \mathbf{k}(\frac{\partial}{\partial x}v - \frac{\partial}{\partial y}u) \end{aligned}$$

Siden jeg skal plotte virvlinga som står normalt på xy-planet bruker jeg $\mathbf{k}(\frac{\partial}{\partial x}v - \frac{\partial}{\partial y}u)$ fordi $\mathbf{k}$ står normalt på både $\mathbf{i}$ og $\mathbf{j}$ som utspenner hele xy-planet.
Du kan se plottet mitt på figur 4 på forrige side. Legger ved python koden under:

```
#virvling
k = gradient(v,axis=1) - gradient(u,axis=0)
#konturplott
verdi = linspace(-1000,1000,9)
contourf(x,y,k,verdi)
colorbar()
#skilleflate
plot(xit,yit,'*',color='black')
#rektanglene
def rektangel(xi,yi,xj,yj):
    x1 = x[yi][xi]; x2 = x[yj][xj]
    y1 = y[yi][xi]; y2 = y[yj][xj]
    plot([x1,x2],[y1,y1], color='red')
    plot([x2,x1],[y2,y2], color='blue')
    plot([x1,x1],[y1,y2], color='black')
    plot([x2,x2],[y2,y1], color='green')
rektangel(34,159,69,169)
rektangel(34,84,69,99)
rektangel(34,49,69,59)
```

f)

Her skal jeg anvende stokes sats på rektanglene, skriver opp formelen:

Stokes sats

$$\int_{\sigma} (\nabla \times \mathbf{v}) \cdot \mathbf{n} d\sigma = \oint_{\lambda} \mathbf{v} \cdot d\mathbf{r}$$

Siden integral kan tenkes på som en sum,

$$\int f(x)dx \approx \sum_{i} f(x_i)\Delta x$$

programmerer jeg integralene jeg skal ta deretter. Sirkulasjonen til et rektangel kan deles opp i fire integral, ett integral per side av rektangelet, og summen av disse integralene. Sirkulasjonen går rundt rektangelet i én retning hele veien. Jeg har definert at sirkulasjonen går mot klokka. Som du kan se i koden min har jeg satt ett minus tegn på to av sidene, dette er fordi python ikke vil gå mot aksene. Motsatt fortegn fikser dette problemet.
Når jeg skal anvende stokes husker jeg på at det er et flate-integral som betyr dobbelt integral. Jeg må derfor multiplisere summen med både Δx og Δy som begge er 0.5, så jeg ganger derfor summen med 0.25.
Legger ved koden min:

```
#Sirkulasjon
def sirkulasjon(xi,yi,xj,yj):
    dt = 0.5
    side1 = sum(u[yi,xi:xj+1]*dt)
    side2 = sum(v[yi:yj+1,xj]*dt)
    side3 = -sum(u[yj,xi:xj+1]*dt)
    side4 = -sum(v[yi:yj+1,xi]*dt)
    sirk = side1 + side2 + side3 + side4
    print 'Bunn:          %.5f' %(side1)
    print 'Hoyre side:    %.5f' %(side2)
    print 'Topp:          %.5f' %(side3)
    print 'Venstre side: %.5f' %(side4)
    return 'Sirkulasjon = %.5f' %(sirk)
#stokes sats
def stokes(x1,y1,x2,y2):
    dvx = gradient(v,0.5,axis=1)
    duy = gradient(u,0.5,axis=0)
    nXv = dvx - duy
    q = sum(nXv[y1:y2+1,x1:x2+1])*0.25
    return 'Stokes = %.5f'%(q)
#skriver ut informasjonen
print '-------------------------------'
print 'Rektangel 1'
print sirkulasjon(34,159,69,169)
print stokes(34,159,69,169)
print 'Differanse = %.5f'%(2695.51409-2621.55870)
print '-------------------------------'
print 'Rektangel 2'
print sirkulasjon(34,85,69,99)
print stokes(34,85,69,99)
print 'Differanse = %.5f'%(-60635.94012--61095.33233)
print '-------------------------------'
print 'Rektangel 3'
print sirkulasjon(34,49,69,59)
```

Kode printer følgende i terminalen:
Terminal python oblig2f.py

Rektangel 1
Bunn: 70100.52388
Hoyre side: 266.27358

Topp: -68332.85610
Venstre side: 661.57274
Sirkulasjon = 2695.51409
Stokes = 2621.55870

Rektangel 2
Bunn: 652.32920
Hoyre side: 118.49871
Topp: -61243.46478
Venstre side: -163.30325
Sirkulasjon = -60635.94012
Stokes = -61095.33233

Rektangel 3
Bunn: 5133.34785
Hoyre side: 207.91001
Topp: -5410.03972
Venstre side: 78.30288
Sirkulasjon = 9.52102
Stokes = -12.21433

Stokes sats sier at kurve integralet(sirkulasjonen) er lik flate integralet(stokes). Jeg ser her at de ikke er like! Dette kan forklares med at det kan ha kommet vanndråper eller lignende på kameraet som lager "hull"i flaten, dette medfører at flate integralet blir feil. Med det i tankene er de ganske så like uansett, ikke så mye som skiller dem. Dette havner under kategorien måle usikkerhet, vi har ikke fått oppgitt en måleusikkerhet så kan ikke si så mye mer enn det.
Det er størst aktivitet i de to øverste rektanglene, noe som er forventet siden hastigheten i lufta er mye høyere enn den i vannet. Resultatene jeg får når jeg analyserer verdiene som kommer ut fra hver siden av rektanglene er overraskende.
Rektangel 1 og 3 er som forventet, omtrent like mye strømning langs topp og bunn, men det som likevel overrasker er at strømningen inn og ut er så forskjellig. Observasjonene i rektangel 2 er veldig interessante. Verdien langs bunn av rektangelet er veldig mye lavere enn på toppen. Hadde forventet at den skulle være lavere, men ikke lavere enn verdiene som er på topp eller bunn på rektangel 3. Verdiene inn og ut av rektangel 2 er også spennende, siden de nesten nuller hverandre ut, akkurat som topp og bunn på de to andre rektanglene.

g)

Her skal jeg bruke Gauss sats, skriver opp formelen:

Gauss sats

$$\int_\tau \nabla \cdot \mathbf{v} d\tau = \int_\sigma \mathbf{v} \cdot \mathbf{n} d\sigma$$

Jeg vet at divergensen til hastighets feltet er null ($\nabla \cdot \mathbf{v} = 0$). Dette betyr at integralet på venstre siden er lik null, da må følgelig integralet på høyre siden også være lik null.
Jeg har funnet tidligere i oppgaven at hvis $\nabla \cdot \mathbf{v} = 0$, får jeg et utrykk for w, $w = -\int(\frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v)dz$
Jeg skal regne ut den integrerte fluksen av hastighetsvektoren $u\mathbf{i} + v\mathbf{j}$ ut av sidene av rektanglene orientert langs xy-planet. Siden jeg skal bruke Gauss sats kan jeg skrive dette om til $\int_\sigma \mathbf{v} \cdot \mathbf{n} d\sigma$. Jeg deler opp integralet slik at jeg integrer hver flate for seg selv. Da peker $\mathbf{n}$ ut av hver flate, dette gjør at jeg skal prikke hastighets vektoren med hennholdsvis $\pm\mathbf{i}$ og $\pm\mathbf{j}$. Dette gir integralene:

$$\int_\sigma (u\mathbf{i} + v\mathbf{j}) \cdot \mathbf{n} d\sigma = \int_{hyre} udx + \int_{topp} vdy - \int_{venstre} udx - \int_{bunn} vdy$$

Jeg er usikker på hvordan jeg skal gjøre om disse integralene til kurve integraler. Så jeg klarer dessverre ikke løse resten av denne oppgaven. Setter pris på en forklaring på hva som er lurt å gjøre får å løse dette problemet.