MEK1100 - Oblig 2

Hans-Petter Harveg

April 2018

a)

I denne oppgaven har jeg lastet inn datafilen og skrevet testfunksjoner for de gitte oppgavene. Spesifikt har jeg skrevet følgende funksjoner

- *get_matrix_sizes()*: skriver ut dimensjonene på matrisen ved hjelp av funksjonen *shate()*.
- *test_pixel_spread()*: sjekker at bredden mellom pixlene er 0.5. Fordi vi har en matrise med x-verider og en med y-verdier sjekker jeg begge matrisene. Funksjonen returnerer *False* dersom den finner en verdi $\Delta x \neq 0.5$.
- *test_y_range()*: sjekker at dataen spenner høyden på røret.

Koden i sin helhet ligger under kildekode.

b)

I denne oppgaven har jeg laget to plot

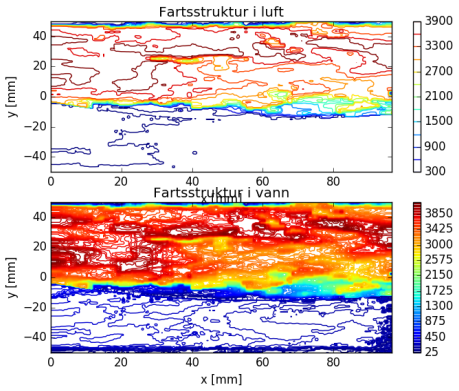

Figure 1: Fartstruktur i hennholdsvis luft og vann.

c)

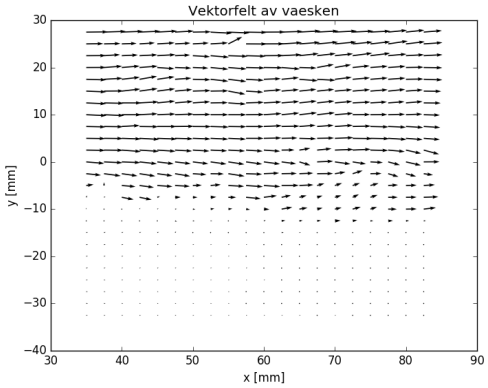

Figure 2: Fordi hastigheten er så liten i vannet i forhold til i lufta er det vanskelig å se vektorpilene.

d)

Divergensen til $\vec{v}$ er gitt ved

$$\nabla \cdot \vec{v} = \frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v + \frac{\partial}{\partial z}w, \tag{1}$$

men måten eksperimentet er satt opp på gir $\frac{\partial}{\partial z}w = 0$. Imkompressibel vil si at når man følger en fluidpartikkel gjennom et hastighetsfelt vil den ha konstant tetthet

$$\frac{D\rho}{dt} = 0. \tag{2}$$

Derav fra kontinuitetsligningen har vi at

$$\frac{D\rho}{dt} + \rho\nabla \cdot \vec{v} = 0, \tag{3}$$

hvor vi har at $\nabla \cdot \vec{v} = 0$ for imkomprissibelt fluid. Dette betyr at $\nabla \cdot \vec{v} = 0$

$$\frac{\partial}{\partial z}w = -\left(\frac{\partial}{\partial x}u + \frac{\partial}{\partial y}v\right) \tag{4}$$

e)

Virvlingen til $\vec{v}$ er gitt ved

$$\nabla \cdot \vec{v} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ \partial_x & \partial_y & \partial_z \\ u & v & w \end{vmatrix}. \tag{5}$$

Som gir

$$\hat{k}\left(\frac{\partial}{\partial_x}v - \frac{\partial}{\partial_y}u\right), \tag{6}$$

altså komponenten *normalt på xy-planet.*

f)

g)

Kildekode

```
import scipy.io as sio
import numpy as np
import matplotlib.pyplot as plt

class DataSet():
    def __init__(self, filename):
        filedata = sio.loadmat(filename)
        self.data = {"x":filedata.get("x"), "y":filedata.get("y"),
    "u":filedata.get("u"), "v":filedata.get("v"), "xit":filedata.
    get("xit"), "yit":filedata.get("yit") }


    def get_data(self):
        return (self.data["x"], self.data["y"], self.data["u"],
    self.data["v"], self.data["xit"], self.data["yit"])


    def get_matrices_sizes(self):
        for key, value in self.data.items():
            [a, b] = np.shape(value)
            print "Size of [", key, "]:", b, a

        return True


    def test_pixel_spread(self):
        [x, y, u, v, xit, yit] = self.get_data()

        for iy in range(201):
            prev_x = 0
            prev_y =0
            for ix in range(1,192):
                if x[iy][ix]-prev_x != 0.5:
                    return False
                prev_x = x[iy][ix]

                if y[iy][ix]-prev_y != 0.5:
                    return False
                prev_y = y[iy][ix]

        return True
```

```
39
40
41      def test_y_range(self):
42          yit = 0
43
44
45      def plot_contours(self):
46          [x, y, u, v, xit, yit] = self.get_data()
47
48          c = np.sqrt(u**2 + v**2)
49
50          plt.subplot(2,1,1)
51          plt.contour(x, y, c, 15)
52          plt.colorbar()
53          dataset.render_plot("Fartsstruktur i luft", "x [mm]", "y [
    mm]")
54
55          plt.subplot(2,1,2)
56          plt.contour(x, y, c, 200)
57          plt.colorbar()
58          dataset.render_plot("Fartsstruktur i vann", "x [mm]", "y [
    mm]")
59
60          plt.show()
61
62
63      def plot_quiver(self, x_0, x_1, y_0, y_1, step):
64          [x, y, u, v, xit, yit] = self.get_data()
65
66          #plt.quiver(x[x_0:x_1:step, y_0:y_1:step], y[x_0:x_1:step,
    y_0:y_1:step], u[x_0:x_1:step, y_0:y_1:step], v[x_0:x_1:step,
    y_0:y_1:step])
67          plt.quiver(x, y, u, v)
68          dataset.render_plot("Vektorfelt av vaesken", "x [mm]", "y [
    mm]")
69
70          self.plot_square([35, 160],[70, 170])
71          self.plot_square([35, 85],[70, 100])
72          self.plot_square([35, 50],[70, 60])
73
74          plt.show()
75
76
77      def plot_square(self, top_left, bottom_right):
78          x_0 = top_left[0]
79          x_1 = top_left[1]
80          y_0 = bottom_right[0]
81          y_1 = bottom_right[1]
82
83          self.plot_line(x_0, x_1, y_0, y_0, 'b') # top
84          self.plot_line(x_1, x_1, y_0, y_1, 'g') # right
85          self.plot_line(x_0, x_1, y_1, y_1, 'r') # bottom
86          self.plot_line(x_0, x_0, y_0, y_1, 'c') # left
87
88
89      def plot_line(self, x_0, x_1, y_0, y_1, col):
90          plt.plot([x_0, x_1], [y_0, y_1], col)
```

```
91
92
93      def render_plot(self, title, xlabel, ylabel):
94          plt.title(title)
95          plt.xlabel(xlabel)
96          plt.ylabel(ylabel)
97
98
99  # Problem a)
100 dataset = DataSet("data.mat")
101 dataset.get_matrices_sizes()
102 dataset.test_pixel_spread()
103 dataset.test_y_range()
104
105 # Problem b)
106 #dataset.plot_contours()
107
108 # Problem c)
109 dataset.plot_quiver(0, 194, 0, 201, 5)
110 #dataset.plot_quiver(35, 160, 70, 170, 5)
111 #dataset.plot_quiver(35, 85, 70, 100, 5)
112 #dataset.plot_quiver(35, 50, 70, 60, 5)
113
114
115 # Problem d)
116
117 # Problem e)
118
119 # Problem f)
120
121 # Problem g)
```