MAT3440

Oblig1

He Gu

heg@student.matnat.uio.no

Problem 1

(a)

We know that $f(x)$ is continous, which means that $x(t)$ and $y(t)$ must also be continous. Therefore $H(x(t), y(t))$ is continous, which means that $H(t)$ is also continous., where $H(t)$ is the same function as $H(x, y)$ but instead, we focus on t.

And since $x(t)$ and $y(t)$ are solutions of the equation, then we have that

$$\begin{aligned} H'(t) &= \frac{d\left(\frac{1}{2}y^2 + F(x)\right)}{dt} \\ &= y(t) \cdot y'(t) + f(x) \cdot x'(t) \\ &= x'(t) \cdot \big(-f(x)\big) + f(x) \cdot x'(t) \\ &= 0 \end{aligned}$$

which means that $H(t)$ is a constant with focusing on t.

Thus, $H(0) = H(t)$ for all t, i.e. $H\big(x(0), y(0)\big) = H(x(t), y(t))$.

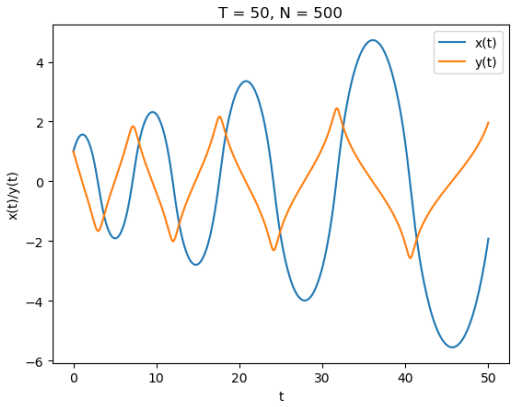
(b)

CODE(PYTHON):

```
import numpy as np
import matplotlib.pyplot as plt
def forward_eul(T, N, x0, y0):
    h = T/N
    t = np.linspace(0, T, N)
    x = np.zeros(N)
    y = np.zeros(N)
    x[0] = x0
    y[0] = y0
```

```
    for i in range(N-1):
        x[i+1] = x[i] + y[i]*h
        y[i+1] = y[i] - (2*x[i]/(1+x[i]**2))*h
    return t, x, y
t, x, y = forward_eul(50, 500, 1, 1)
plt.xlabel("t")
plt.ylabel("x(t)/y(t)")
plt.plot(t, x, t, y)
plt.legend(["x(t)","y(t)"])
plt.title("T = 50, N = 500")
```

RESULTS & FIGURES:



(c)

Since $f(x) = \frac{2x}{1+x^2}$, we shall have $F(x) = \ln(1+x^2) + C$, where C is a constant.

And thus it is easy to check that

$$\left|H\big(x(T),y(T)\big) - H\big(x(0),y(0)\big)\right| = \left|\frac{1}{2}y(T)^2 - \frac{1}{2}y(0)^2 + F(x(T) - F(x(0))\right|$$
$$= \left|\frac{1}{2}y(T)^2 - \frac{1}{2} + \ln(1+x(T)^2) - \ln 2\right|$$

Now we could use scripts to calculate the error.

CODE(PYTHON):

```
import numpy as np
import matplotlib.pyplot as plt
def forward_eul(T, N, x0, y0):
    h = T/N
    t = np.linspace(0, T, N)
    x = np.zeros(N)
    y = np.zeros(N)
    x[0] = x0
```

```
    y[0] = y0
    for i in range(N-1):
        x[i+1] = x[i] + y[i]*h
        y[i+1] = y[i] - (2*x[i]/(1+x[i]**2))*h
    return t, x, y
T = 50
n = 6
N = [500, 1000, 2000, 4000, 8000, 16000]
error = np.zeros(n)
print("   N       error      1/error")
for i in range(n):
    t, x, y = forward_eul(T, N[i], 1, 1)
    h = 0.5*(y**2)+np.log(1+x**2)
    error[i] = h[-1] - h[0]
    print("%5g    %.6f    %.6f"%(N[i],error[i],1/error[i]))
```

RESULT

```
    N       error      1/error
  500    2.285809    0.437482
 1000    1.497736    0.667674
 2000    0.869036    1.150701
 4000    0.461163    2.168431
 8000    0.240337    4.160823
16000    0.119727    8.352331
```

It is easy to see that when N becomes **twice larger**, $(\frac{1}{error})$ will also be approximately **twice larger**. Thus it is reasonable to have that

$$err_N \sim \frac{1}{0.002N}$$

And this indicates that $\boldsymbol{\alpha = 1}$

(d) & (e):

Som beskjeden sier, disse to delene skal ignoreres siden det finnes feil.

(f)

CODE(PYTHON):

```
import numpy as np
import matplotlib.pyplot as plt
def forward_eul(T, N, x0, y0):
    h = T/N
```

```
    x = np.zeros(N)
    y = np.zeros(N)
    x[0] = x0
    y[0] = y0
    for i in range(N-1):
        x[i+1] = x[i] + y[i]*h
        y[i+1] = y[i] - (2*x[i+1]/(1+x[i+1]**2))*h
    return x, y
T = 50
n = 6
N = [500, 1000, 2000, 4000, 8000, 16000]
error = np.zeros(n)
print("   N       abs(error)")
for i in range(n):
    x, y = forward_eul(T, N[i], 1, 1)
    h = 0.5*(y**2)+np.log(1+x**2)
    error[i] = abs(h[-1] - h[0])
    print("%5g    %.6f"%(N[i],error[i]))
```

RESULT

```
    N       abs(error)
  500    0.097358
 1000    0.049393
 2000    0.008965
 4000    0.000866
 8000    0.000119
16000    0.000153
```

Errors of this system is apparently much smaller than the former system. But as for finding the α, it could be a little more complicated.

Although we know that some of the errors are in fact negative, the result is still a little wired than we expected, but it is reasonable, since T will also influence the result. And at some special point, T will show more influence than N. Thus, the errors will be easily influenced if they are already too small, i.e. when N is large enough to make errors "too precise" towards 0.

Therefore we shall choose smaller N to estimate our "α", and obviously, by the same reason, I prefer to choose $\alpha = 1$.

$$\text{i.e.} \quad err_N \sim \frac{1}{0.02N}$$

which is 10 times smaller than errors of the former system.

(g)

While $x = 0$ and $y = 0$, it is easy to check that both x' and y' is zero, which means that the origin$(0, 0)$ is indeed a fix point.

Then, we are going to check its stability. We could start with calculating the corresponding Jacobian matrix $\begin{pmatrix} f_x & f_y \\ g_x & g_y \end{pmatrix}$, which should be

$$\begin{pmatrix} 0 & 1 \\ -f'(x) & -\mu \end{pmatrix}$$

And thus we could find the characteristic equation:

$$\lambda^2 + \mu\lambda + f'(x) = 0$$

Since μ and $f'(x)$ are positive, then the real parts of both eigenvalues of the matrix above must be **negative**. And therefore the fixed point is **stable**.

Now we could conclude that the origin is a linearly stable fixed point.

Since the real parts of the two eigenvalues must be negative, then we could conclude that the fixed point $(0, 0)$ must be a **stable spiral point**, which means that no matter what initial condition is, the orbit will point at $(0, 0)$, and thus we shall obviously have

$$\lim_{t \to \infty} \begin{pmatrix} \boldsymbol{x(t)} \\ \boldsymbol{y(t)} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}$$

for any $x(0)$, $y(0)$.

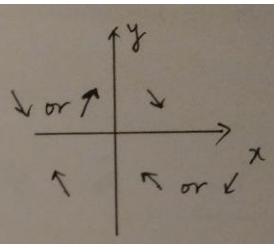
(h)

Since $f'(x)$ is equal to zero if and only if $x = 0$, then it is easy to check that the system above has only ONE fixed point, i.e. $(0, 0)$. Thus, in order to sketch a phase portrait of different μ, we need to figure out the stability with different μ.

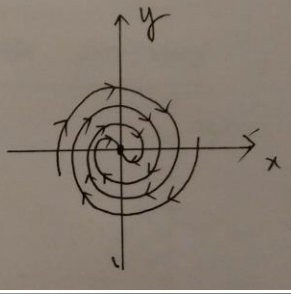
While $\mu > 0$, we already have known that the fixed point $(0, 0)$ is stable. And it is not difficult to find out that while $\mu < 0$, the point $(0, 0)$ is unstable. In addition, while $\mu = 0$, the fixed point is “alone”, i.e. “center”.

According to our results above and referring part of the **trace-determinant plane**, we could conclude that

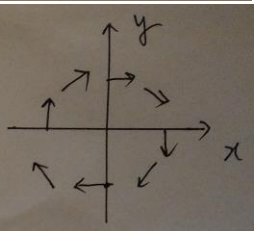
While $\mu > 0$, the origin is stable &



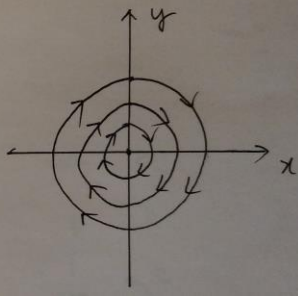
Thus we have the phase portrait:



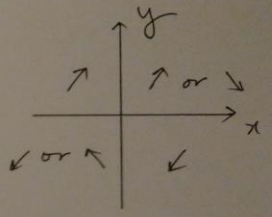
While $\mu = 0$, the origin is "center" &



Thus we have the phase portrait:



While $\mu < 0$, the origin is unstable &



Thus we have the phase portrait: