MAT 1110: Løsningsforslag til Oblig 1, V-08

Løsning: a) Vi har

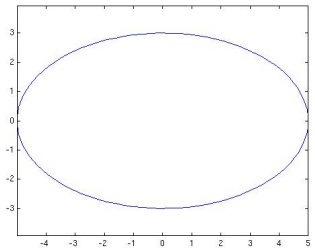
$$\frac{x(t)^2}{5^2} + \frac{y(t)^2}{3^2} = \cos^2 t + \sin^2 t = 1$$

som viser at alle punktene på $\mathcal{C}$ ligger på ellipsen med sentrum i origo og halvakser $a = 5$ og $b = 3$. Det er lett å overbevise seg om at når t går fra 0 til 2π, så gjennomløper $\mathbf{r}(t)$ ellipsen én gang med start og stopp i punktet $(5, 0)$. Siden $c = \sqrt{a^2 - b^2} = \sqrt{5^2 - 3^2} = 4$, er brennpunktene $(4, 0)$ og $(-4, 0)$.

For å tegne ellipsen gir vi kommandoene

```
t=linspace(0,2*pi,100);
x=5*cos(t);
y=3*sin(t);
plot(x,y)
axis('equal')
```

Resultatet blir slik:



.

b) Formelen for buelengde er

$$L = \int_a^b \sqrt{x'(t)^2 + y'(t)^2}\, dt$$

I vårt tilfelle er $x'(t) = -5\sin t$ og $y'(t) = 3\cos t$, så vi får

$$L = \int_0^{2\pi} \sqrt{(-5\sin t)^2 + (3\cos t)^2}\, dt = \int_0^{2\pi} \sqrt{25\sin^2 t + 9\cos^2 t}\, dt =$$

$$= \int_0^{2\pi} \sqrt{16\sin^2 t + 9\sin^2 t + 9\cos^2 t}\, dt = \int_0^{2\pi} \sqrt{16\sin^2 t + 9}\, dt =$$

$$= \int_0^{2\pi} \sqrt{9 + 16\sin^2 t}\, dt$$

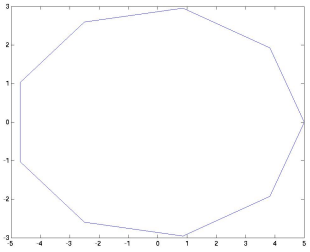
Vi bruker MATLAB til å finne en tilnærmet verdi:

```
quad(@(t)sqrt(9+16*(sin(t)).^2),0,2*pi)
ans =
        25.5270
```

c) Det er flere måter å gjøre dette punktet på. Vi velger den mest primitive der vi skriver ut hele uttrykket. Det tar tid, men med litt "klipp og lim" er det overkommelig. Husk at tre punktumer ... på slutten av en linje forteller MATLAB at kommandoen fortsetter på neste linje:

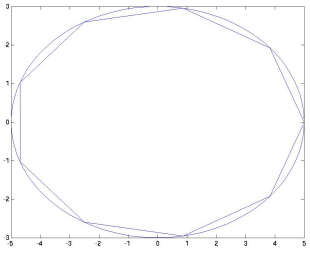
```
l=sqrt((x(1)-x(2))^2+(y(1)-y(2))^2)+sqrt((x(2)-x(3))^2+(y(2)-y(3))^2)+...
sqrt((x(3)-x(4))^2+(y(3)-y(4))^2)+sqrt((x(4)-x(5))^2+(y(4)-y(5))^2)+...
sqrt((x(5)-x(6))^2+(y(5)-y(6))^2)+sqrt((x(6)-x(7))^2+(y(6)-y(7))^2)+...
sqrt((x(7)-x(8))^2+(y(7)-y(8))^2)+sqrt((x(8)-x(9))^2+(y(8)-y(9))^2)+...
sqrt((x(9)-x(10))^2+(y(9)-y(10))^2)

l =
   25.0117
```

Punktene som inngår i denne summen ligger på ellipsen, og summen vi regner ut, gir oss lengden til den brudne stien som forbinder dem. Figuren nedenfor viser hvordan denne stien ser ut:



.

Den neste figuren viser hvordan den brudne stien passer inn i ellipsen:



.

Vi ser at det ikke er så stor forskjell mellom lengden til den brudne stien og buelengden til ellipsen, men at den brudne stien alltid vil være kortere siden den følger rette linjer mellom delepunktene. Bruker vi en finere oppdeling

av intervallet $[0, 2\pi]$, vil vi få en bedre tilnærming til buelengden.

d) Vi lager m-filen

```
function l=pathlength(x,y)
N=length(x);
l=0;
for k=1:N-1
    l=l+sqrt((x(k+1)-x(k))^2+(y(k+1)-y(k))^2);
end
```

Filen sjekker først hvor lange input-vektorene er, og adderer så opp lengdene av linjestykkene som forbinder punktene. Vi gjør nå disse kjøringene (der vi etterhvert går over til `format long` for å kunne se endringene i desimalutviklingen):

```
t=linspace(0,2*pi,100);
x=5*cos(t);
y=3*sin(t);
l=pathlength(x,y)

l =
   25.5227

t=linspace(0,2*pi,1000);
x=5*cos(t);
y=3*sin(t);
l=pathlength(x,y)

l =
   25.5270

t=linspace(0,2*pi,10000);
x=5*cos(t);
y=3*sin(t);
l=pathlength(x,y)

l =
   25.5270

format long
l

l =
  25.526998443411909
```

```
t=linspace(0,2*pi,10^6);
x=5*cos(t);
y=3*sin(t);
l=pathlength(x,y)

l =
  25.526998863356983
```

e) Vi kan bruke det samme programmet som ovenfor, men må passe litt ekstra på det som skjer i $t = 0$. Dette kan gjøres på flere måter, men siden kjøringene er få, velger vi å korrigere for hånd i hvert enkelt tilfelle. Legg merke til at siden $x(t) = t$, kan vi legge inn x direkte:

```
x=linspace(0,1,10^2);
y=x.*sin(1./x);
```

Den siste kommandoen kan få oss i trøbbel siden uttrykket ikke gir mening når $x = 0$. Kontrollerer vi hva som ligger i førstekomponenten til `y`, får vi

```
y(1)

ans =
   NaN
```

der `NaN` er en forkortelse for "not a number". Vi ønsker at førstekomponenten (som representerer $f(0)$) skal være 0, så vi endrer verdien ved å skrive

```
   y(1)=0;
```

Nå har vi riktige input-vektorer `x` og `y` og kan kjøre programmet:

```
l=pathlength(x,y)
l =
    2.5050
```

Vi bruker samme fremgangsmåte for flere delepunkter:

```
x=linspace(0,1,10^3);
y=x.*sin(1./x);
y(1)=0;
l=pathlength(x,y)

l =
```

```
   3.0462

x=linspace(0,1,10^4);
y=x.*sin(1./x);
y(1)=0;
l=pathlength(x,y)

l =
   3.8222

x=linspace(0,1,10^6);
y=x.*sin(1./x);
y(1)=0;
l=pathlength(x,y)

l =
  5.3138
```

Vi tar med en "bonuskjøring" i forhold til oppgaven for å undersøke enda bedre:

```
x=linspace(0,1,10^7);
y=x.*sin(1./x);
y(1)=0;
l=pathlength(x,y);

l =
    6.0334
```

I motsetning til i ellipsetilfellet ser vi ikke ut til å ha noen klar konvergens i dette tilfellet — verdiene fortsetter å vokse med omtrent samme hastighet når vi putter inn flere delepunkter. Det skyldes at kurven vi nå ser på, svinger så mye opp og ned at den er uendelig lang (lag en figur). Skriver du opp formelen for buelengden, får du et integral med t i nevneren, og det viser seg at dette integralet divergerer (riktignok svært langsomt).