Introduktion til matplotlib - en grafikpakke til Python Af Henrik Sterner (hst@nextkbh.dk)

Matplotlib er en grafikpakke til Python, som er inspireret af MATLAB. Den er særdeles velegnet til at visualisere data og resultater af beregninger.

Matplotlib har en række forskellige moduler, som kan bruges til forskellige formål. Vi vil her gennemgå de mest almindelige moduler.

Om brugen af disse slides

- Disse slides forsøger at eksemplifere en lang række af de vigtige begreber i Python
- De må på ingen måde kopieres uden tilladelse fra Henrik Sterner
- De er lavet i markdown og kan derfor nemt konverteres til andre formater
- Ved brug af Visual Studio Code kan de konverteres til HTML, PDF, PowerPoint, Word, LaTeX og mange andre formater
- Slides er tilgængelige på github.com/henriksterner/IntelligenteSystemer/

Konvertere slides til andre formater

- Fra markdown til pdf ved brug af pandoc:

```
pandoc -t beamer -o slidesmatplotlib.pdf slidesmatplotlib.m
```

- Fra markdown til word ved brug af pandoc:

```
pandoc -t docx -o slidespython.docx slidesmatplotlib.md
```

Installation af matplotlib

Til at starte med skal vi installere matplotlib. Det gøres nemmest ved at åbne en terminal og skrive:

```
pip install matplotlib
```

Hvis du bruger Anaconda, så er matplotlib allerede installeret. Hvis du bruger Google Colab, så er matplotlib også allerede installeret.

Import af matplotlib

Når matplotlib er installeret, så skal vi importere det i vores program. Det gøres ved at skrive:

```
import matplotlib.pyplot as plt
```

Her importeres matplotlibs pyplot-modul, som vi vil bruge til at lave vores plots. Test at det virker ved at skrive:

```
plt.plot([1,2,3,4])
plt.ylabel('noget')
plt.show()
```

Hvis du bruger Anaconda, så kan du også skrive:

```
%matplotlib inline
```

Det gør at plots vises direkte i notebooken.

Plot af punkter

Vi kan plotte punkter ved at bruge funktionen plot. Den tager to lister som argumenter, en liste med x-koordinater og en liste med y-koordinater. Hvis vi kun giver en liste, så bruger den den som y-koordinater og laver selv en liste med x-koordinater, som er 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Herunder ses en generisk udgave af funktionen plot:

```
plt.plot(x, y)
```

Her er x og y lister med x- og y-koordinater.

Eksempler på plot af punkter

Herunder ses et eksempel på et plot af punkter:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16])
plt.show()
```

Kun plot af y-koordinater

Herunder ses et eksempel på et plot af punkter, hvor vi kun angiver y-koordinater:

```
import matplotlib.pyplot as plt
plt.plot([1,4,9,16])
plt.show()
```

Plot af punkter med forskellige farver og typer

Vi kan også plotte punkter med forskellige farver og typer. Det gøres ved at angive et ekstra argument til funktionen plot. Det ekstra argument er en streng, som angiver farven og typen. Herunder ses en generisk udgave af funktionen plot med farve og type:

```
plt.plot(x, y, 'farve+type')
```

Her er x og y lister med x- og y-koordinater. Farve+type er en streng, som angiver farven og typen. Herunder ses en tabel med de forskellige farver og typer:

Eksempel på plot af punkter med forskellige farver og typer

Herunder ses et eksempel på et plot af punkter med forskellige farver og typer:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.show()
```

I denne kode er der angivet farven r (rød) og typen o (cirkel).

Forskellige farver i plot

Farve	Type
b	blå
g	grøn
r	rød
c	cyan
m	magenta
y	gul
k	sort
w	hvid

Forskellige typer i plot

Type	Type
.	prikker
,	pixel
o	cirkel
v	trekant ned
^	trekant op
<	trekant venstre
>	trekant højre
s	firkant
p	pentagon

Plot af flere punkter i samme figur

Vi kan plotte flere punkter i samme figur. Det gøres ved at kalde funktionen plot flere gange. Herunder ses en generisk udgave af funktionen plot med flere plot i samme figur:

```
plt.plot(x1, y1, 'farve+type', x2, y2, 'farve+type', x3, y3
```

Her er x1, y1, x2, y2, x3 og y3 lister med x- og y-koordinater. Farve+type er en streng, som angiver farven og typen.

Eksempel på plot af flere punkter i samme figur

Herunder ses et eksempel på et plot af flere punkter i samme figur:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16], 'ro', [1,2,3,4], [1,2,3,4]
plt.show()
```

I denne kode er der angivet farven r (rød) og typen o (cirkel) for det første plot og farven b (blå) og typen + (plus) for det andet plot.

Plot af punkter med akser

Vi kan også plotte punkter med akser. Det gøres ved at kalde funktionen axis. Den tager fire tal som argumenter, xmin, xmax, ymin og ymax. Herunder ses en generisk udgave af funktionen axis:

```
plt.axis([xmin, xmax, ymin, ymax])
```

Her er xmin, xmax, ymin og ymax tal, som angiver minimum og maximum for x- og y-aksen.

Eksempel på plot af punkter med akser

Herunder ses et eksempel på et plot af punkter med akser:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.axis([0, 6, 0, 20])
plt.show()
```

Plot af punkter med titel og akser

Vi kan også plotte punkter med titel og akser. Det gøres ved at kalde funktionerne title, xlabel og ylabel. Herunder ses en generisk udgave af funktionerne title, xlabel og ylabel:

```
plt.title('titel')
plt.xlabel('x-akse')
plt.ylabel('y-akse')
```

Hvor titel, x-akse og y-akse er streng-argumenter.

Plot af punkter med titel og akser

Herunder ses et eksempel på et plot af punkter med titel og akser:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.title('titel')
plt.xlabel('x-akse')
plt.ylabel('y-akse')
plt.show()
```

I denne kode er der angivet titel, x-akse og y-akse.

Plot af punkter med grid

Vi kan også plotte punkter med grid. Det gøres ved at kalde funktionen grid. Den tager et boolsk argument som angiver om der skal være grid eller ej. Herunder ses en generisk udgave af funktionen grid:

```
plt.grid(boolsk)
```

Her er boolsk et boolsk argument, som angiver om der skal være grid eller ej.

Eksempel på plot af punkter med grid

Herunder ses et eksempel på et plot af punkter med grid:

```
import matplotlib.pyplot as plt
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.grid(True)
plt.show()
```

Plot af punkter med flere figurer

Vi kan også plotte punkter med flere figurer. Det gøres ved at kalde funktionen figure. Den tager et tal som argument, som angiver nummeret på figuren. Herunder ses en generisk udgave af funktionen figure:

```
plt.figure(nummer)
```

Her er nummer et tal, som angiver nummeret på figuren.

Eksempel på plot af punkter med flere figurer

Herunder ses et eksempel på et plot af punkter med flere figurer:

```
import matplotlib.pyplot as plt
plt.figure(1)
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.figure(2)
plt.plot([1,2,3,4], [1,2,3,4], 'b+')
plt.show()
```

Plot af punkter med flere figurer i samme figur

Vi kan også plotte punkter med flere figurer i samme figur. Det gøres ved at kalde funktionen subplot. Den tager tre tal som argumenter, som angiver antallet af rækker, antallet af kolonner og nummeret på figuren. Herunder ses en generisk udgave af funktionen subplot:

```
plt.subplot(rækker, kolonner, nummer)
```

Her er rækker, kolonner og nummer tal, som angiver antallet af rækker, antallet af kolonner og nummeret på figuren.

Eksempel på plot af punkter med flere figurer i samme figur

Herunder ses et eksempel på et plot af punkter med flere figurer i samme figur:

```
import matplotlib.pyplot as plt
plt.subplot(1,2,1)
plt.plot([1,2,3,4], [1,4,9,16], 'ro')
plt.subplot(1,2,2)
plt.plot([1,2,3,4], [1,2,3,4], 'b+')
plt.show()
```

Her angiver plt.subplot(1,2,1) at der skal være 1 række, 2 kolonner og at figuren skal være nummer 1. Tilsvarende angiver plt.subplot(1,2,2) at der skal være 1 række, 2 kolonner og at figuren skal være nummer 2.

Plot af funktioner

Vi kan også plotte funktioner. Det gøres ved at kalde funktionen plot. Den tager to lister som argumenter, en liste med x-koordinater og en liste med y-koordinater. Herunder ses en generisk udgave af funktionen plot:

```
plt.plot(x, y)
```

Her er x og y lister med x- og y-koordinater.

Eksempel på plot af funktioner

Herunder ses et eksempel på et plot af funktioner:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 2*np.pi, 0.1)
y = np.sin(x)
plt.plot(x, y)
plt.show()
```

I denne kode er der angivet x-koordinaterne som en liste med værdier fra 0 til 2 med intervaller på 0.1 og y-koordinaterne som en liste med sinus tilsvarende x-koordinaterne.

Plot af funktioner med titel og akser

I det følgende eksempel er der angivet titel, x-akse og y-akse:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 2*np.pi, 0.1)
y = np.sin(x)
plt.plot(x, y)
plt.title('sinus')
plt.xlabel('x-akse')
plt.ylabel('y-akse')
plt.show()
```

Scatter plot

Et scatter plot er et plot af punkter. Det gøres ved at kalde funktionen scatter. Den tager to lister som argumenter, en liste med x-koordinater og en liste med y-koordinater. Herunder ses en generisk udgave af funktionen scatter:

```
plt.scatter(x, y)
```

Her er x og y lister med x- og y-koordinater.

Eksempel på scatter plot

Herunder ses et eksempel på et scatter plot:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.random.rand(100)
y = np.random.rand(100)
plt.scatter(x, y)
plt.show()
```

Her er der genereret 100 tilfældige tal mellem 0 og 1 til x- og y-koordinaterne. Disse tal er gemt i to lister, som er brugt som argumenter til funktionen scatter.

Scatter plot med farver

Vi kan også lave et scatter plot med farver. Det gøres ved at kalde funktionen scatter. Den tager tre lister som argumenter, en liste med x-koordinater, en liste med y-koordinater og en liste med farver. Herunder ses en generisk udgave af funktionen scatter med farver:

```
plt.scatter(x, y, c=farver)
```

Her er x og y lister med x- og y-koordinater og farver en liste med farver.

Histogram i matplotlib

Vi kan også lave et histogram i matplotlib. Det gøres ved at kalde funktionen hist. Den tager en liste som argument, som angiver værdierne. Herunder ses en generisk udgave af funktionen hist:

```
plt.hist(værdier)
```

Her er værdier en liste med værdier.

Eksempel på histogram i matplotlib

Herunder ses et eksempel på et histogram i matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.random.normal(170, 10, 250)
plt.hist(x)
plt.show()
```

Her er der genereret 250 tilfældige tal med normalfordeling med middelværdi 170 og standardafvigelse 10. Disse tal er gemt i en liste, som er brugt som argument til funktionen hist.

Plot af punkter med farvebar

Vi kan også plotte punkter med farvebar. Det gøres ved at kalde funktionen colorbar. Herunder ses en generisk udgave af funktionen colorbar:

```
plt.colorbar()
```

Eksempel på plot af punkter med farvebar

Herunder ses et eksempel på et plot af punkter med farvebar:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.random.rand(100)
y = np.random.rand(100)
colors = np.random.rand(100)
plt.scatter(x, y, c=colors)
plt.colorbar()
plt.show()
```

Her er der genereret 100 tilfældige tal mellem 0 og 1 til x- og y-koordinaterne og farverne. Disse tal er gemt i tre lister, som er brugt som argumenter til funktionen scatter. Funktionen scatter bruges til at plotte punkter med farver. Til sidst er der kaldt funktionen colorbar, som viser farvebaren.

Cirkeldiaagram i matplotlib

Vi kan også lave et cirkeldiagram i matplotlib. Det gøres ved at kalde funktionen pie. Den tager en liste som argument, som angiver størrelserne. Herunder ses en generisk udgave af funktionen pie:

```
plt.pie(størrelser)
```

Her er størrelser en liste med størrelser.

Eksempel på cirkeldiagram i matplotlib

Herunder ses et eksempel på et cirkeldiagram i matplotlib:

```
import matplotlib.pyplot as plt
størrelser = [20, 40, 60, 80]
plt.pie(størrelser)
plt.show()
```

3d plot i matplotlib

Vi kan også lave et 3d plot i matplotlib. Det gøres ved at kalde funktionen plot_surface. Den tager to lister som argumenter, en liste med x-koordinater, en liste med y-koordinater og en liste med z-koordinater. Herunder ses en generisk udgave af funktionen plot_surface:

```
plt.plot_surface(x, y, z)
```

Her er x, y og z lister med x-, y- og z-koordinater.

Eksempel på 3d plot i matplotlib

Herunder ses et eksempel på et 3d plot i matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
x = np.arange(-5, 5, 0.25)
y = np.arange(-5, 5, 0.25)
x, y = np.meshgrid(x, y)
r = np.sqrt(x**2 + y**2)
z = np.sin(r)
fig = plt.figure()
ax = Axes3D(fig)
ax.plot_surface(x, y, z)
plt.show()
```

Logaritmisk plot i matplotlib

Vi kan også lave et logaritmisk plot i matplotlib. Det gøres ved at kalde funktionen semilogx, semilogy eller loglog. Herunder ses en generisk udgave af funktionerne semilogx, semilogy og loglog:

```
plt.semilogx(x, y)
plt.semilogy(x, y)
plt.loglog(x, y)
```

Her er x og y lister med x- og y-koordinater.

Eksempel på logaritmisk plot i matplotlib

Herunder ses et eksempel på et logaritmisk plot i matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 10, 0.1)
y = np.exp(x)
plt.semilogy(x, y)
plt.show()
```

Her er der genereret 100 tal mellem 0 og 10 til x-koordinaterne og y-koordinaterne er beregnet som eksponentialfunktionen tilsvarende x-koordinaterne. Disse tal er gemt i to lister, som er brugt som argumenter til funktionen semilogy.

Bar plot i matplotlib

Vi kan også lave et bar plot i matplotlib. Det gøres ved at kalde funktionen bar. Den tager to lister som argumenter, en liste med x-koordinater og en liste med højder. Herunder ses en generisk udgave af funktionen bar:

```
plt.bar(x, højder)
```

Her er x en liste med x-koordinater og højder en liste med højder.

Eksempel på bar plot i matplotlib

Herunder ses et eksempel på et bar plot i matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 10, 0.1)
y = np.exp(x)
plt.bar(x, y)
plt.show()
```

Stem plot i matplotlib

Vi kan også lave et stem plot i matplotlib. Det gøres ved at kalde funktionen stem. Den tager to lister som argumenter, en liste med x-koordinater og en liste med y-koordinater. Herunder ses en generisk udgave af funktionen stem:

```
plt.stem(x, y)
```

Her er x og y lister med x- og y-koordinater.

Eksempel på stem plot i matplotlib

Herunder ses et eksempel på et stem plot i matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 10, 0.1)
y = np.exp(x)
plt.stem(x, y)
plt.show()
```

Gemme plot i matplotlib

Vi kan også gemme et plot i matplotlib. Det gøres ved at kalde funktionen savefig. Den tager en streng som argument, som angiver filnavnet. Herunder ses en generisk udgave af funktionen savefig:

```
plt.savefig(filnavn)
```

Her er filnavn en streng, som angiver filnavnet.

Eksempel på at gemme plot i matplotlib

Herunder ses et eksempel på at gemme et plot i matplotlib:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 10, 0.1)
y = np.exp(x)
plt.plot(x, y)
plt.savefig('plot.png')
```