# Introduktion til Matplotlib og Datavisualisering i Python

AAMS - DLE

November 23, 2025

# Formål med præsentationen

- Lære grundlæggende datavisualisering i Python.
- Forstå brugen af Matplotlib til plotning.
- Bruge Numpy til at generere advanceret data.
- Forstå grundlæggende Pandas.

# Introduktion til Matplotlib

Matplotlib er et kraftfuldt bibliotek til at skabe visualiseringer i Python.

Installer `matplotlib` i terminalen, hvis det ikke allerede er installeret:

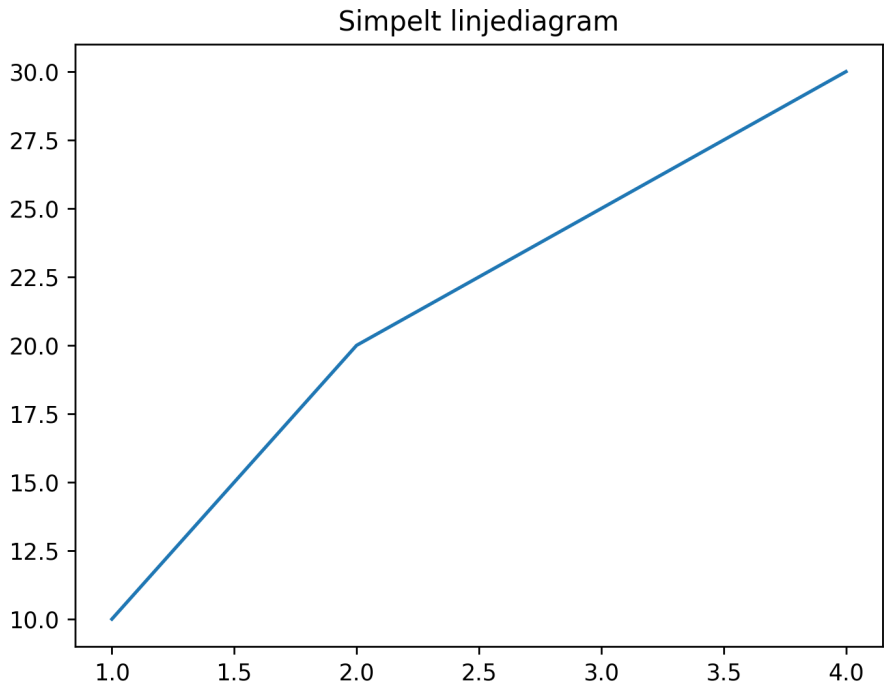
Listing 1: Installation af Matplotlib

```
pip install matplotlib
```

# Grundlæggende Plot

Et simpelt linjediagram:

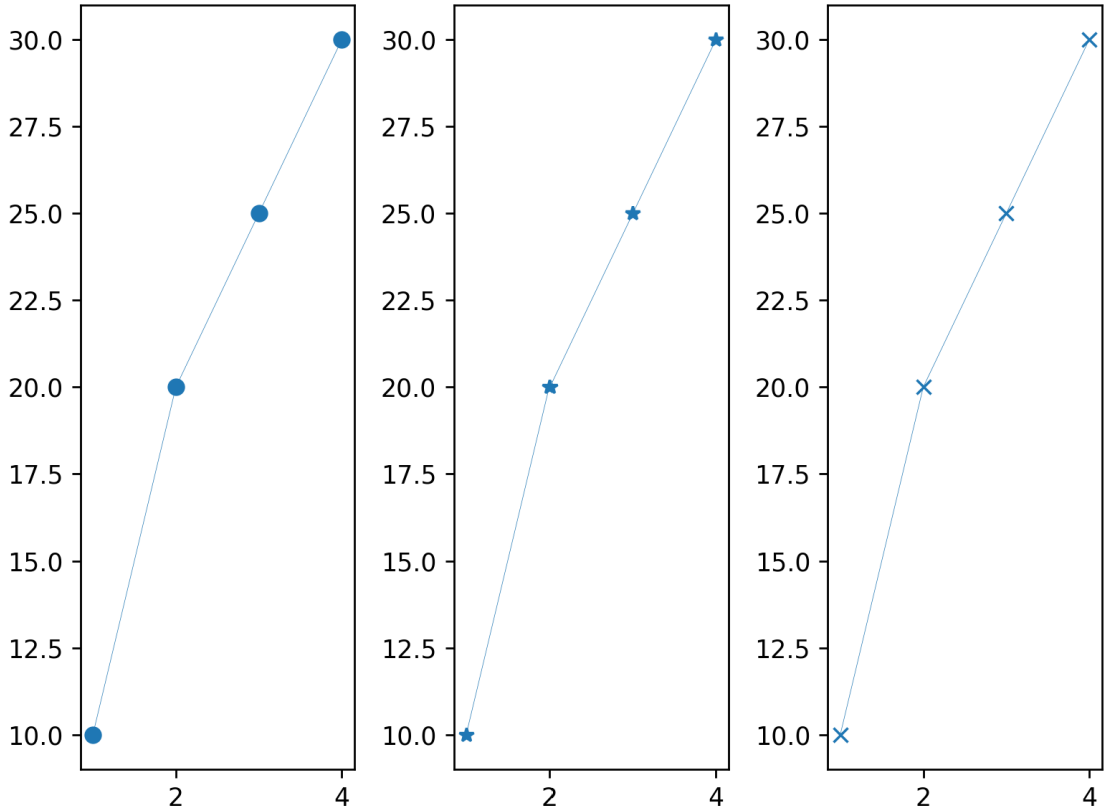
```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]
plt.plot(x, y)
plt.title("Simpelt-Linjediagram")
plt.show()
```



# Marker i pyplot

**Marker:** bruges til at give hvert punkt i plottet et karakteristika.

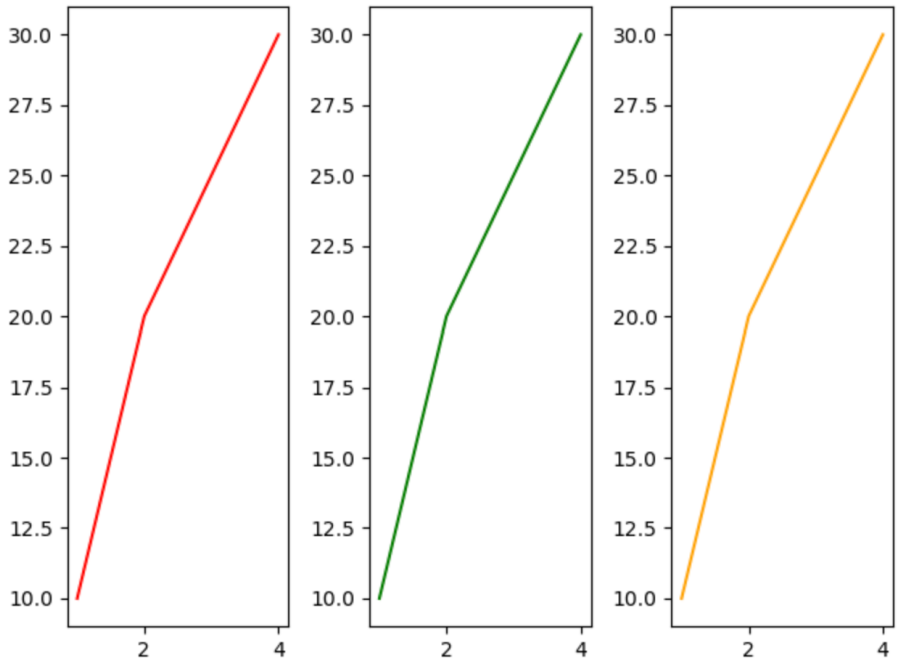
| | |
|---|---|
| 'o' | Circle |
| '*' | Star |
| '.' | Point |
| ',' | Pixel |
| 'x' | X |
| 'X' | X (filled) |



# Colors i pyplot

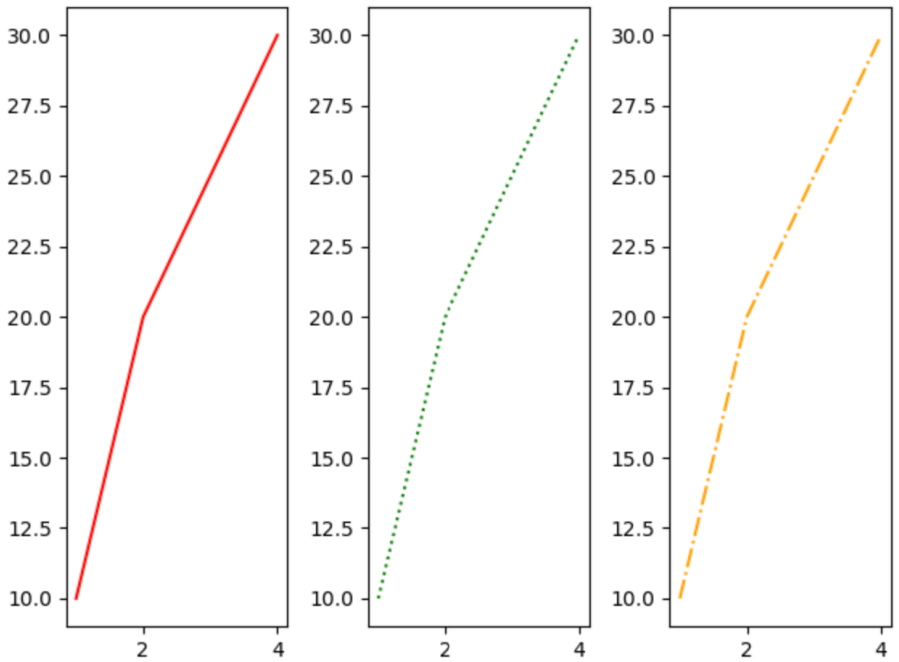
Disse plots kan have forskellige farver ved at indsætte følgende color references:

| | |
|---|---|
| 'r' | red |
| 'g' | green |
| 'b' | blue |
| 'c' | cyan |
| 'm' | magenta |
| 'y' | yellow |
| 'k' | black |
| 'w' | white |



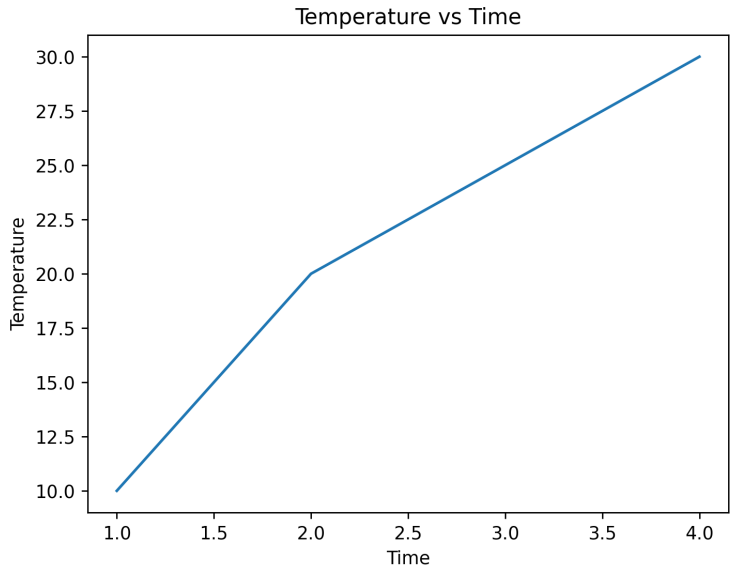
# linestyle i pyplot

| | |
|---|---|
| '-' | solid |
| ':' | dotted |
| '–' | dashed |
| '-.' | dashdot |
| ' ' | none |



# Labels og title

```
import matplotlib.pyplot as plt
x = [1,2,3,4]
y = [10,20,25,30]
plt.plot(x, y)

plt.title("Temperature vs Time")
plt.xlabel("Time")
plt.ylabel("Temperature")
plt.legend
plt.plot(x,y)
plt.show()
```

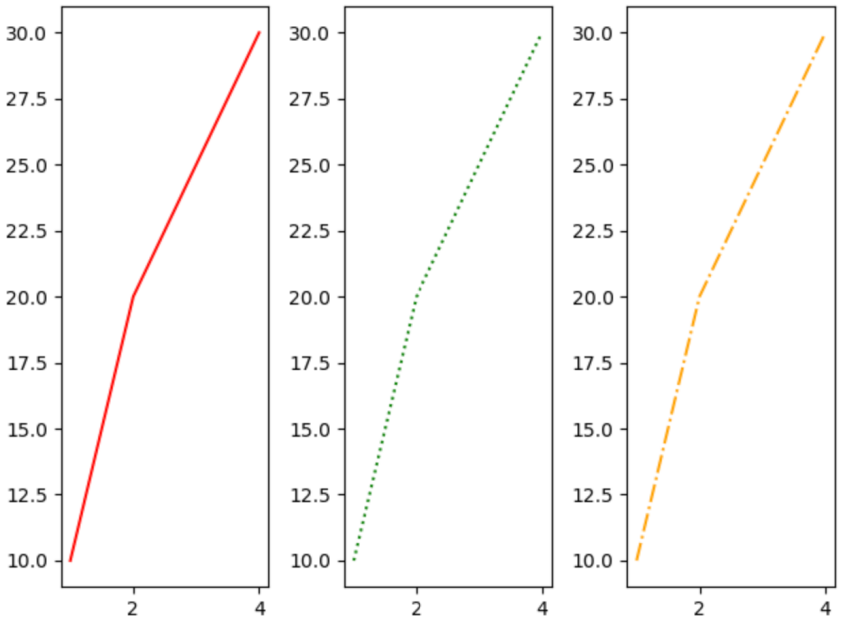


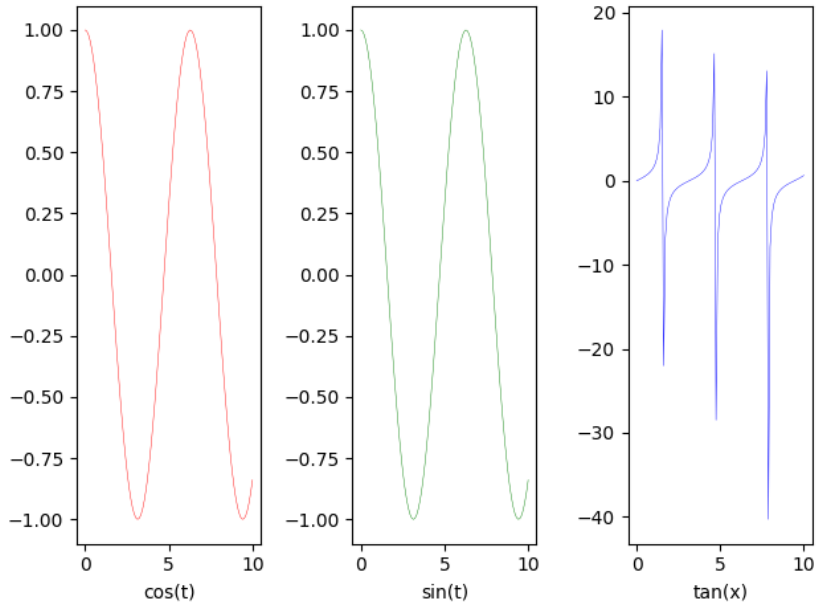
# Subplot

I matplotlib er det muligt at lave subplots

```
subplot(row,column,number)
```

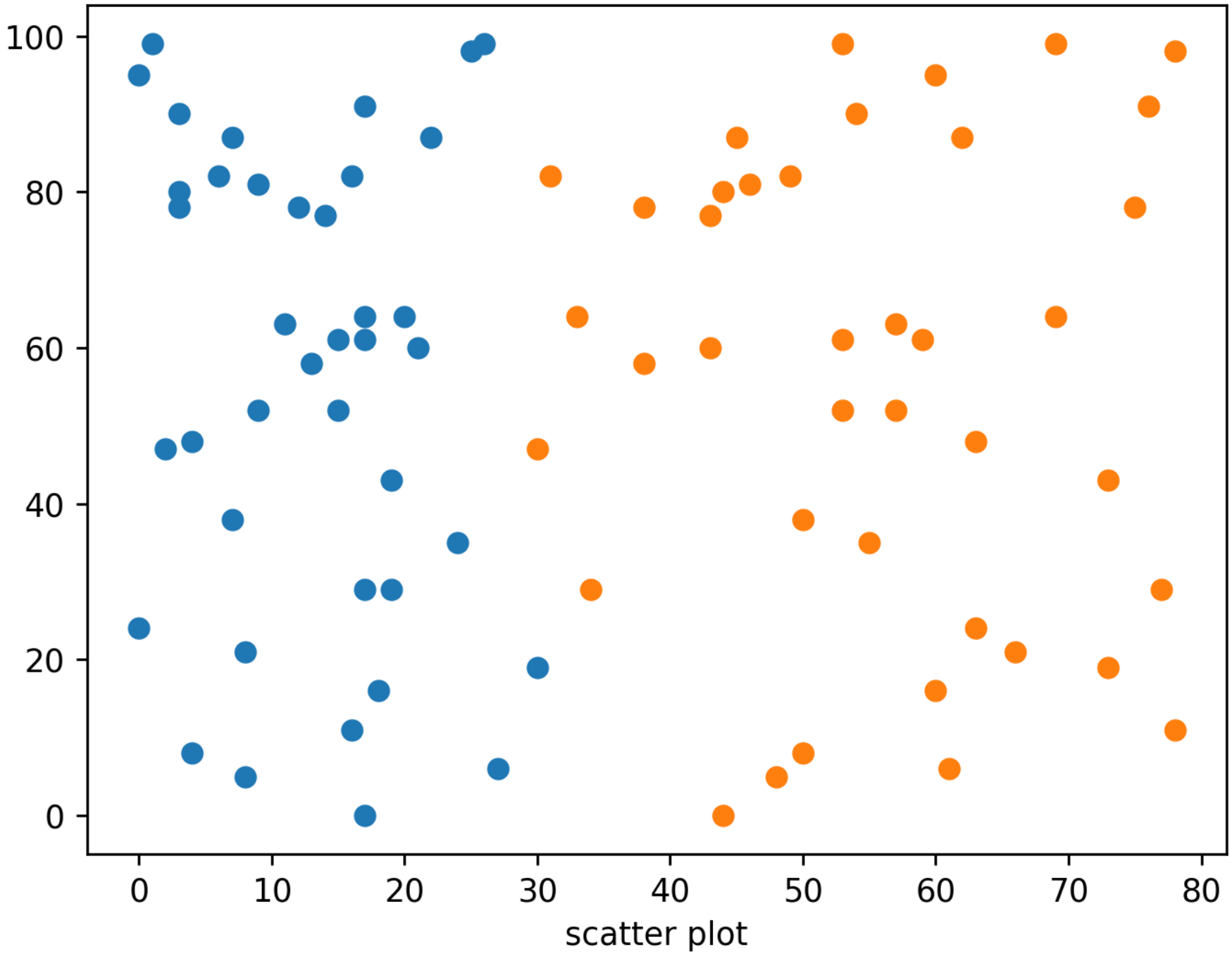
```
#plot 1:
x = [1, 2, 3, 4]
y = [10, 20, 25, 30]

#plot 1
plt.subplot(1, 3, 1)
plt.plot(x,y, color="r", linestyle="-")

#plot 2
plt.subplot(1, 3, 2)
plt.plot(x,y, color="g", linestyle=":")

#plot 3
plt.subplot(1, 3, 3)
plt.plot(x,y, color="yellow", linestyle="-.")

plt.show()
```


30.0
27.5
25.0
22.5
20.0
17.5
15.0
12.5
10.0
2
4


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
5
10
cos(t)
sin(t)
20
10
0
−10
−20
−30
−40
tan(x)

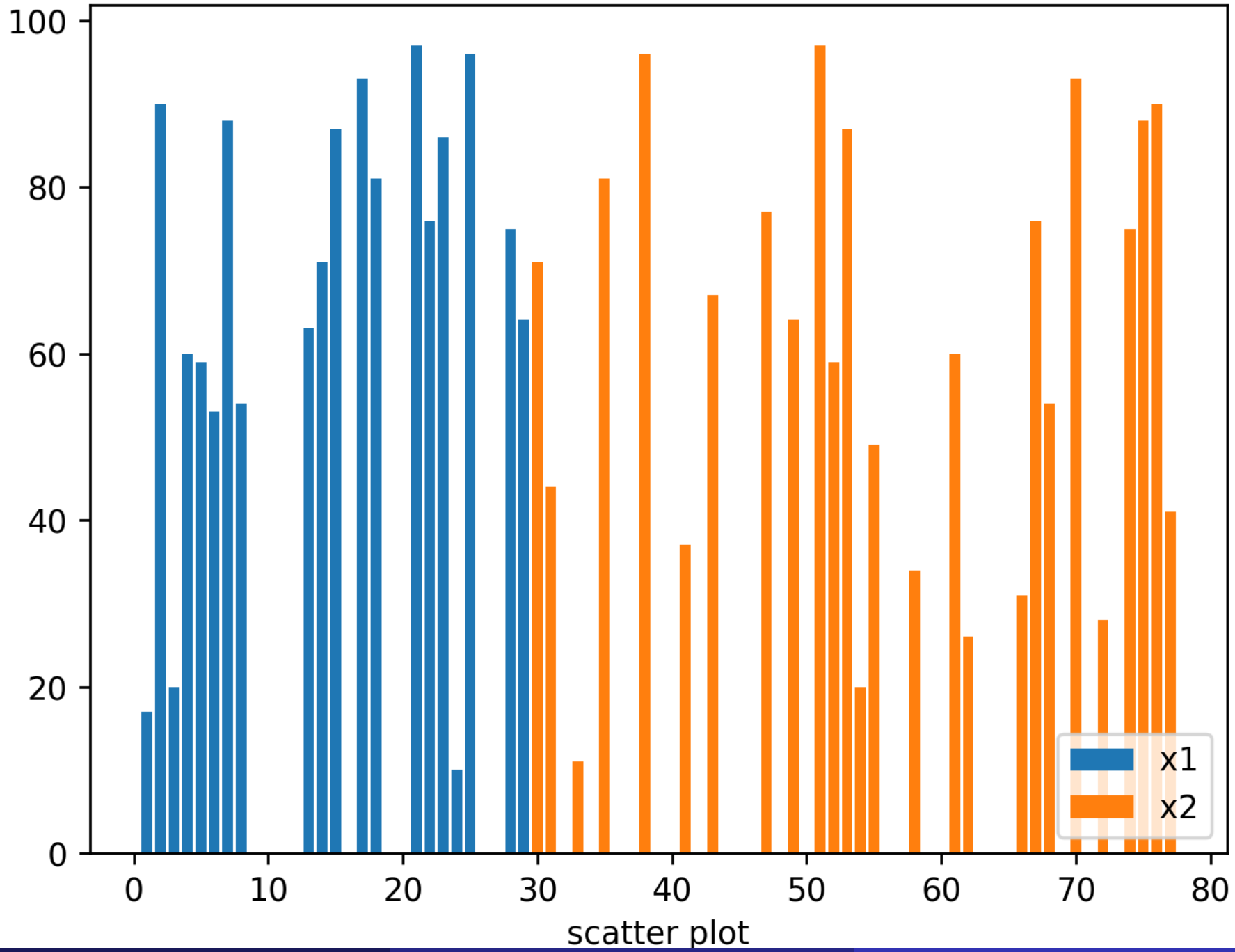
# Scatter plot

```
import matplotlib.pyplot as plt
from random import randint
x1 = []
x2 = []
y = []
for c in range(0,40):
        random1 = randint(0,30)
        random2 = randint(0,100)
        random3 = randint(30,80)
        x1.append(random1)
        y.append(random2)
        x2.append(random3)
plt.xlabel("scatter plot")
plt.scatter(x1,y)
plt.scatter(x2,y)
plt.show()
```


100
80
60
40
20
0
0
10
20
30
40
50
60
70
80
scatter plot

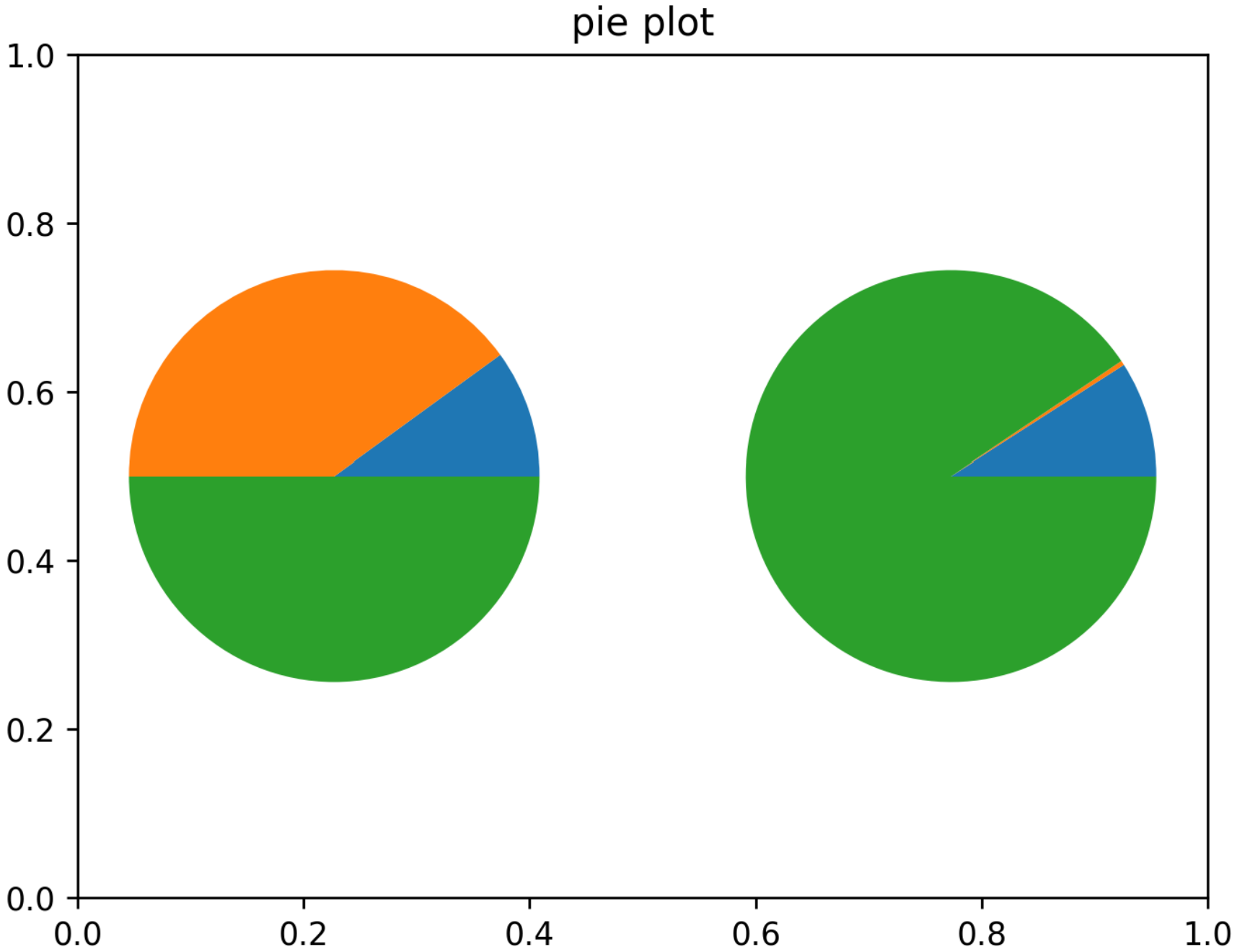
# Bar plot

```
from random import randint
x1 = []
x2 = []
y = []
for c in range(0,40):
        random1 = randint(0,30)
        random2 = randint(0,100)
        random3 = randint(30,80)
        x1.append(random1)
        y.append(random2)
        x2.append(random3)
plt.xlabel("scatter plot")
plt.bar(x1,y, label='x1')
plt.bar(x2,y, label='x2')
plt.legend(["x1","x2"],loc="lower right")
plt.show()
```


100
80
60
40
20
0
0
10
20
30
40
50
60
70
80
scatter plot
x1
x2

# Pie

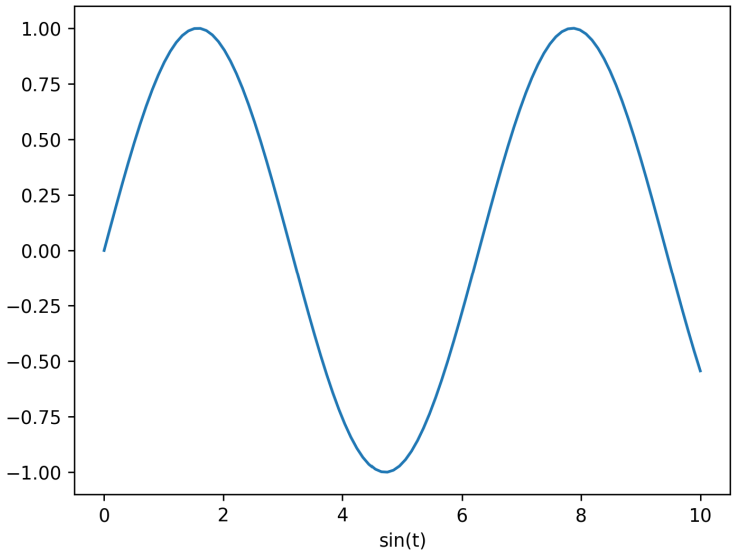
```
x = [10,40,50]
y = [100,4,1000]
plt.title("pie␣plot")
plt.subplot(1,2,1)
plt.pie(x)
plt.subplot(1,2,2)
plt.pie(y)
plt.show()
```


pie plot
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

# Introduktion til Numpy

Numpy bruges til numeriske beregninger og datastrukturer.

```
import matplotlib.pyplot as plt
import numpy as np
t = np.linspace(0, 10, 100)
y = np.sin(t)
plt.xlabel("sin(t)")
plt.plot(x, y)
plt.show()
```



# Append and delete i numpy

```
import numpy as np
x = np.array([1,2,3,4])
x = np.append(x,[5])
print(x)
```

```
import numpy as np
x = np.array([1,2,3,4])
x = np.delete(x,[2]) # delet ved indexing
print(x)
```

# Pandas

Hvis Pandas ikke er installeret

```
pip install pandas
```

# Create CSV file

```
import pandas as pd

# Simuleret sensor data
data = {
        'Tidspunkt': ['2024-02-24 08:00', '2024-02-24 08:10', '2024-02-24 08:20'],
        'Temperatur (°C)': [22.5, 22.8, 23.0],
        'Fugtighed (%)': [45, 47, 46],
        'Gasniveau (ppm)': [120, 130, 128],
        'Bevaegelse registreret': [False, True, False]
}

df = pd.DataFrame(data)

# Gemmer DataFrame som CSV
df.to_csv('c:/users/aso/desktop/c:/users/aso/desktop/Eksamen/SensorData.csv', index=False
    )
print("CSV-filen 'c:/users/aso/desktop/Eksamen/SensorData.csv' er oprettet.")
```

# Read CSV file

```
import pandas as pd

# Laes sensor data
df = pd.read_csv('c:/users/aso/desktop/SensorData.csv'
    )
print(df)
```

# Append to CSV file

```
# Ny maaling
ny_maaling = pd.DataFrame({
        'Tidspunkt': ['2024-02-24 08:30'],
        'Temperatur ( C)': [23.2],
        'Fugtighed (%)': [48],
        'Gasniveau (ppm)': [125],
        'Bevaegelse registreret': [True]
})

# Tilfoej maalingen til CSV-filen
ny_maaling.to_csv('c:/users/aso/desktop/Eksamen/
    SensorData.csv', mode='a', header=False, index=
    False)
print("Ny sensor maaling tilfoejet.")
```

# Delete from CSV file

Når en række skal slettes, så bruges en filtrering.

```
# Laes data
df = pd.read_csv('c:/users/aso/desktop/Eksamen/
    SensorData.csv')

# Slet maalinger med gasniveau under 125 ppm
df = df[df['Gasniveau (ppm)'] >= 125]

# Gem opdateret CSV
df.to_csv('c:/users/aso/desktop/Eksamen/SensorData.csv
    ', index=False)
print("Filtreret sensor data:\n", df)
```

- `df['Gasniveau (ppm)'] >= 125` fungere som et filter og returnere True/False (se næste slide)
- `df[df['Gasniveau (ppm)'] >= 125]` vil returnere kun de som i den indre df er True

## df['Gasniveau (ppm)'] >= 125

```
>>> df['Gasniveau (ppm)'] <= 125
1    False
2    False
3     True
4     True
5     True
6     True
7     True
Name: Gasniveau (ppm), dtype: bool
```

## df[df['Gasniveau (ppm)'] >= 125]

```
>>> df[df['Gasniveau (ppm)'] <= 125]
           Tidspunkt  Temperatur (°C)  Fugtighed (%)  Gasniveau (ppm)  \
3  2024-02-24 08:30            23.2             48              125
4  2024-02-24 08:30            23.2             48              125
5  2024-02-24 08:30            23.2             48              125
6  2024-02-24 08:30            23.2             48              125
7  2024-02-24 08:30            23.2             48              125

   Bevægelse registreret
3                   True
4                   True
5                   True
6                   True
7                   True
```

```
# Laes data
df = pd.read_csv('c:/users/aso/desktop/Eksamen/
    SensorData.csv')

# Slet maalinger med et specifikt gasniveau -> 125 ppm
df = df[df['Gasniveau (ppm)'] != 125]

# Gem opdateret CSV
df.to_csv('c:/users/aso/desktop/Eksamen/SensorData.csv
    ', index=False)
print("Filtreret sensor data:\n", df)
```

# Opdateringer

```
# Laes data
df = pd.read_csv('c:/users/aso/desktop/Eksamen/
    SensorData.csv')

# Juster temperaturen med -0.5 C (kalibrering)
df['Temperatur ( C)'] = df['Temperatur ( C)'] - 0.5

# Gem aendringer
df.to_csv('c:/users/aso/desktop/Eksamen/SensorData.csv
    ', index=False)
print("Temperaturdata er blevet opdateret:\n", df)
```

# Simple plot med Pandas I

```python
# Importer noedvendige biblioteker
import pandas as pd                 # Til databehandling (pandas DataFrame)
import matplotlib.pyplot as plt     # Til at plotte grafer

# 1 Laes CSV-filen (tilpas stien hvis noedvendigt)
df = pd.read_csv('c:/users/aso/desktop/Eksamen/SensorData.csv')

# 2 Tjek kolonnenavne for at sikre korrekt stavning og fjern ekstra mellemrum
print(f"Kolonnenavne i filen: {df.columns.tolist()}")
df.columns = df.columns.str.strip()  # Fjern eventuelle whitespace fra kolonnenavne

# 3 Konverter 'Tidspunkt' til datetime-format
# errors='coerce' sikrer, at ugyldige datoer konverteres til NaT (Not a Time)
df['Tidspunkt'] = pd.to_datetime(df['Tidspunkt'], errors='coerce')

# 4 Tjek datatype for 'Tidspunkt' og 'Temperatur ( C)'
print("\nDatatyper i DataFrame:")
print(df.dtypes)

# 5 Konverter 'Temperatur ( C)' til numerisk (hvis noedvendigt) og fjern ugyldige raekker
df['Temperatur (C)'] = pd.to_numeric(df['Temperatur (C)'], errors='coerce')
df = df.dropna(subset=['Tidspunkt', 'Temperatur (C)'])  # Fjern raekker med NaN-vaerdier
```

# Simple plot med Pandas II

```
# 6 Plot data (linjegraf)
plt.figure(figsize=(10, 6))  # Justere stoerelsen paa grafen (valgfrit)
df.plot(x='Tidspunkt', y='Temperatur-(-C)', kind='line', marker='o', legend=True)

# 7  Tilfoej labels og titel
plt.title('Temperatur-over-tid')
plt.ylabel('Temperatur-(-C)')
plt.xlabel('Tidspunkt')
plt.grid(True)

# 8 Vis grafen
plt.show()
```

# Afslutning

- Vi har introduceret Matplotlib og Numpy.
- Vi har lært at lave grundlæggende plots.
- Vi har lært grundlæggende Pandas