

Föreläsning 7

CSV-filer, matplotlib.

Genomgång av Lektion 9 (obligatorisk uppgift 4)

csv-data

- Python har modulen csv för att läsa och skriva s k csv-data, *comma separated values*.
Formatet csv är det vanligaste import- och export-formatet för kalkylblad och databaser.

Ex: people.csv:

Filens innehåll:

```
No, Name, country  
1, Alex, USA  
2, Erik, Sweden  
3, Cheng, China
```

Pythonkod läser csv-filen

=>

och skriver ut innehållet:

```
['No', ' Name', ' country']  
['1', ' Alex', ' USA']  
['2', ' Erik', ' Sweden']  
['3', ' Cheng', ' China']
```

```
import csv  
  
with open('people.csv', 'r') as csvFile:  
    reader = csv.reader(csvFile)  
    for row in reader:  
        print(row)
```

läs filen som en csv-fil.
reader-objektet itererar över varje
rad i filen, och returnerar **listor**,
en per rad, där elementen är **strängar**.
Det är **komma** som separerar elementen!

cvs-data, lektion 9, uppgift A

- A. Skriv en funktion `load_csv(filename)` som skapar och returnerar ett lexikon. Nycklarna är landskoder och värdena är listor med CO2-utsläppen (flyttal) för åren 1960 – 2014: {AFG: [414.371,491.378,],AGO:[550.05,454.708,...],ALB:[....]}

1	Country Name, Country Code, Indicator Name, 1960, 1961, 1962, 1963, 1964, 1965
2	Afghanistan, AFG, CO2 emissions (kt), 414.371, 491.378, 689.396, 707.731, 839.743
3	Angola, AGO, CO2 emissions (kt), 550.05, 454.708, 1180.774, 1151.438, 1224.778, 1
4	Albania, ALB, CO2 emissions (kt), 2024.184, 2280.874, 2464.224, 2082.856, 2016.8
5	United Arab Emirates, ARE, CO2 emissions (kt), 11.001, 11.001, 18.335, 22.002, 18
6	Argentina, ARG, CO2 emissions (kt), 48815.104, 51180.319, 53695.881, 50083.886
7	Antigua and Barbuda, ATG, CO2 emissions (kt), 36.67, 47.671, 102.676, 84.341, 91
8	Australia, AUS, CO2 emissions (kt), 88202.351, 90589.568, 94912.961, 101029.517
9	Austria, AUT, CO2 emissions (kt), 30821.135, 31862.563, 33905.082, 36992.696, 38
10	Belgium, BEL, CO2 emissions (kt), 91000.272, 92793.435, 98117.919, 105781.949, 1
11	Benin, BEN, CO2 emissions (kt), 161.348, 128.345, 135.679, 121.011, 143.013, 150.3
12	Burkina Faso, BFA, CO2 emissions (kt), 44.004, 91.675, 84.341, 88.008, 110.01, 102.
13	Bulgaria, BGR, CO2 emissions (kt), 22295.36, 25973.361, 30736.794, 34411.128, 42
14	Bahrain, BHR, CO2 emissions (kt), 575.719, 1771.161, 1591.478, 1195.442, 1598.81
15	Bahamas, The, BHS, CO2 emissions (kt), 410.704, 546.383, 726.066, 707.731, 1085.
16	Belize, BLZ, CO2 emissions (kt), 44.004, 36.67, 60.672, 62.320, 84.341, 84.341, 80.67

FÖRENKLAD ALGORITM (det fattas detaljer):

import csv# FÖRENKLAD ALGORITM (det fattas detaljer):

```
def load_csv(filename): # CO2Emissions_filtered.csv
    lex = {}
    with open(filename, 'r') as csvFile:
        reader = csv.reader(csvFile)
        for row in reader: # row blir en lista
            county_code = #det andra elementet på raden. Alltid?
            data = #gör en lista med all data på raden. Start-kolumn? Alltid?
            #lägg in det nya key-value-paret landskod och data i lex
    return lex
```

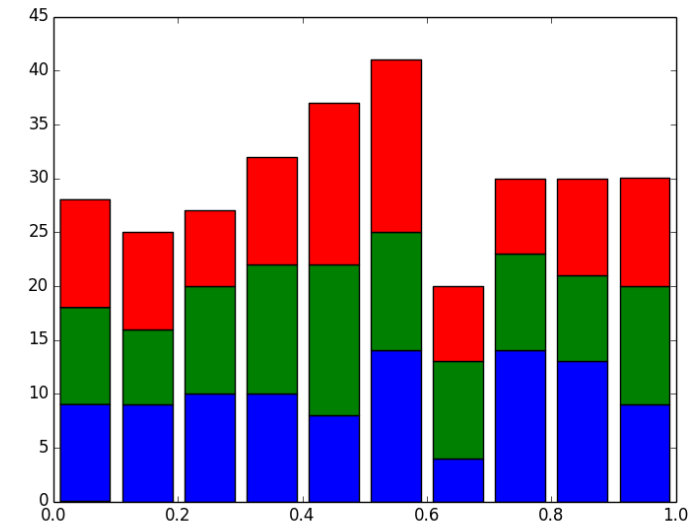
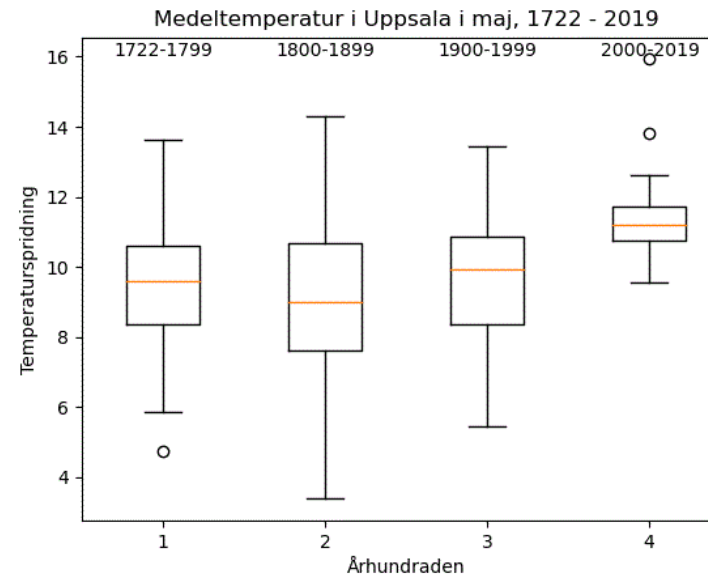
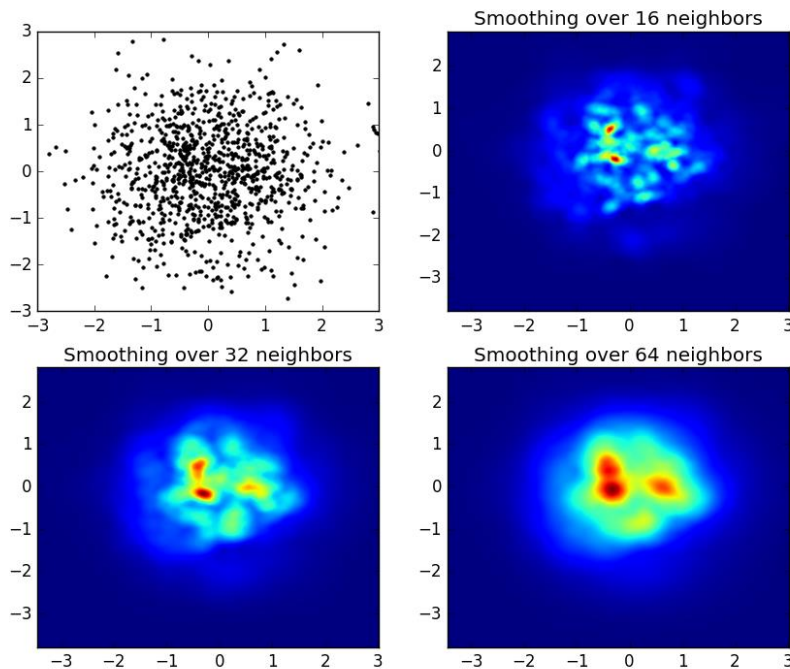
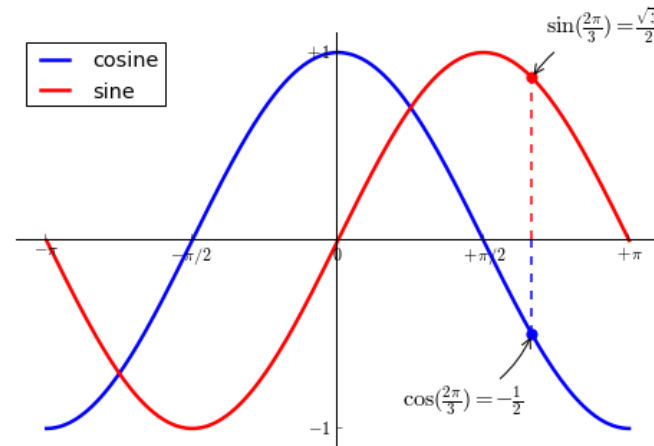
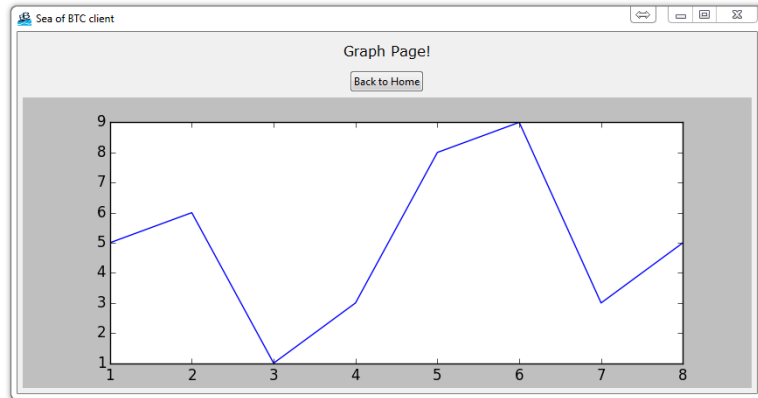
BREAK-OUT ROOM-uppgift:

- Titta på filen CO2Emissions_filtered.csv. Finns det “skräp” i datat? (t ex rad 15, 147)
- Hur kan man hantera att raderna ser olika ut?
- Vilken typ har datat?
- Fattas det något steg i algoritmen?



Datavisualisering med Python: matplotlib

“Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.” <https://matplotlib.org/>



are licensieras enligt [CC BY-SA](#)

ras enligt [CC BY-SA-NC](#)

Datavisualisering med matplotlib

```
import matplotlib.pyplot as plt # pyplot är en modul i paketet matplotlib
```

#Rita linje resp. punkter:

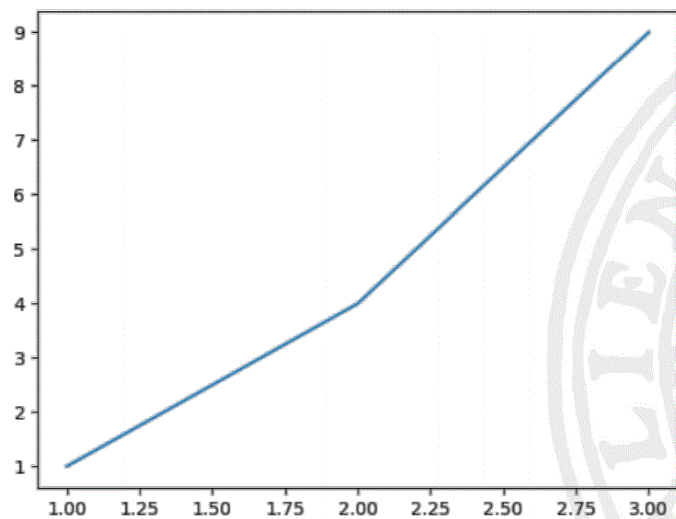
a) `matplotlib.pyplot.plot([1,2,3],[1,4,9],"-")` Onödigt långt!

`plt.plot([1,2,3],[1,4,9],"-")` # a) linje mellan (1,1), (2,4), (3,9)

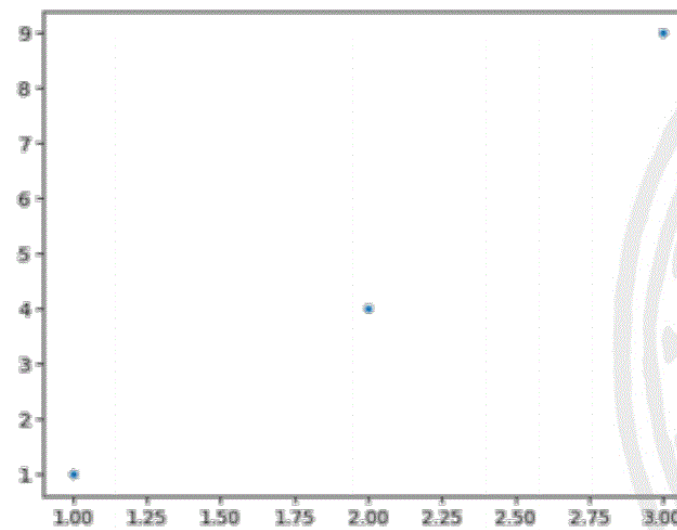
`plt.plot([1,2,3],[1,4,9],".")` # b) ritar punkterna

`plt.show()`

a)

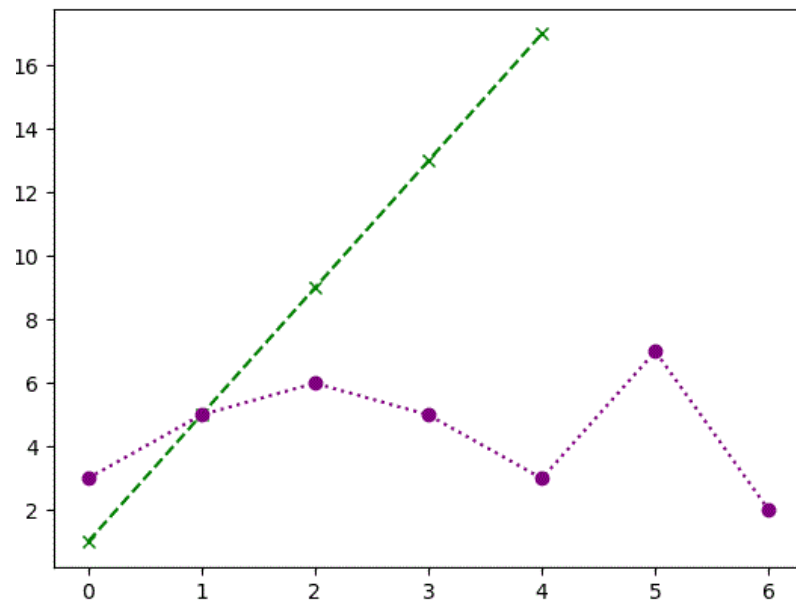
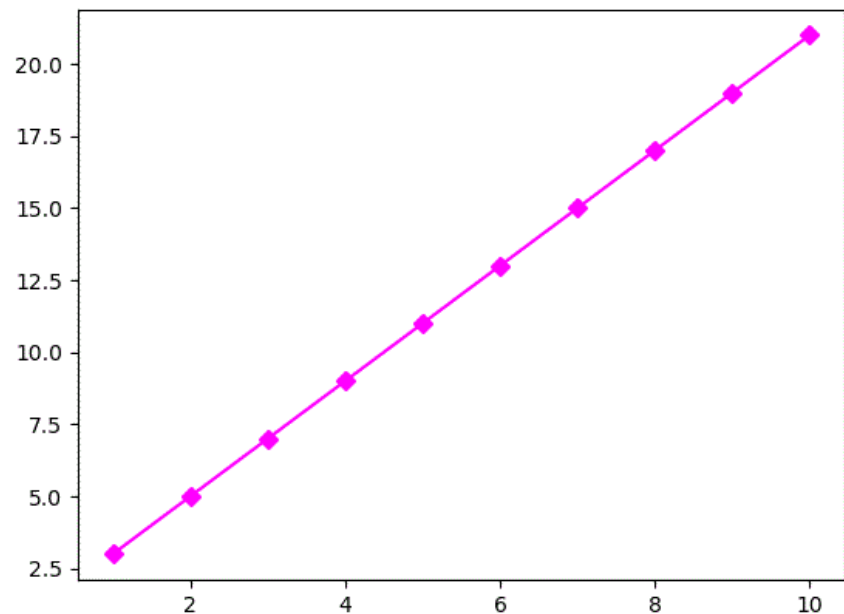
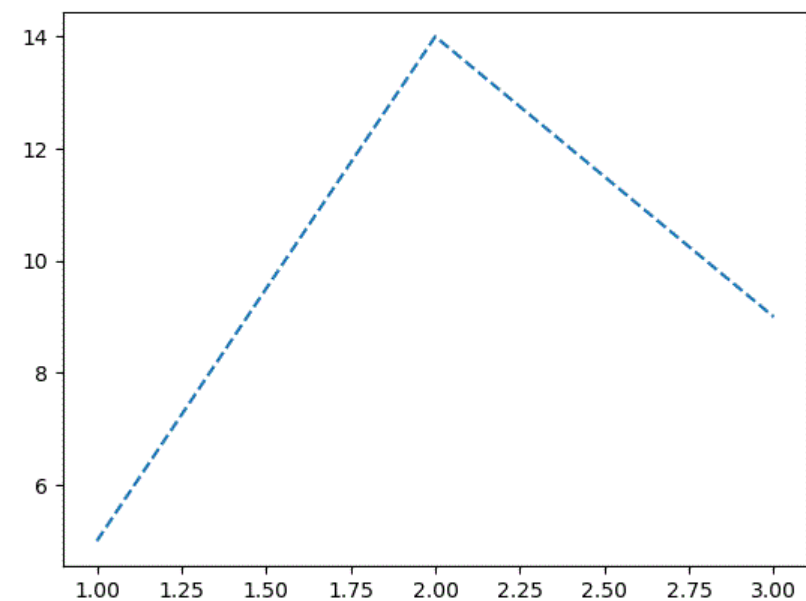


b)



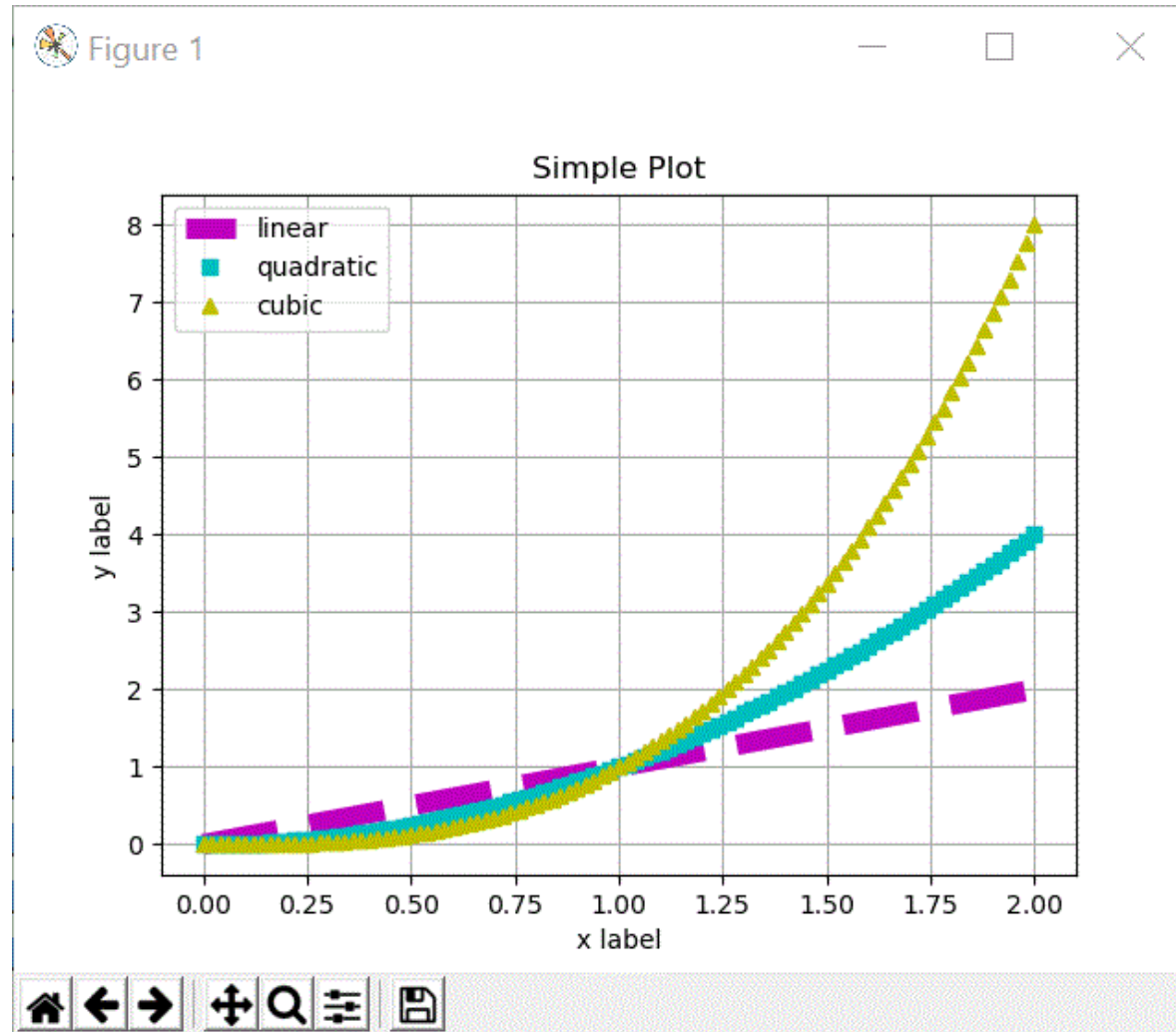
Datavisualisering med matplotlib

Ex1:



Datavisualisering med matplotlib

Ex2:



Datavisualisering med matplotlib

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

Plotta linje:

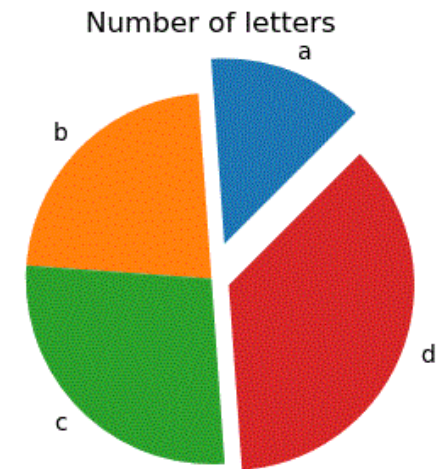
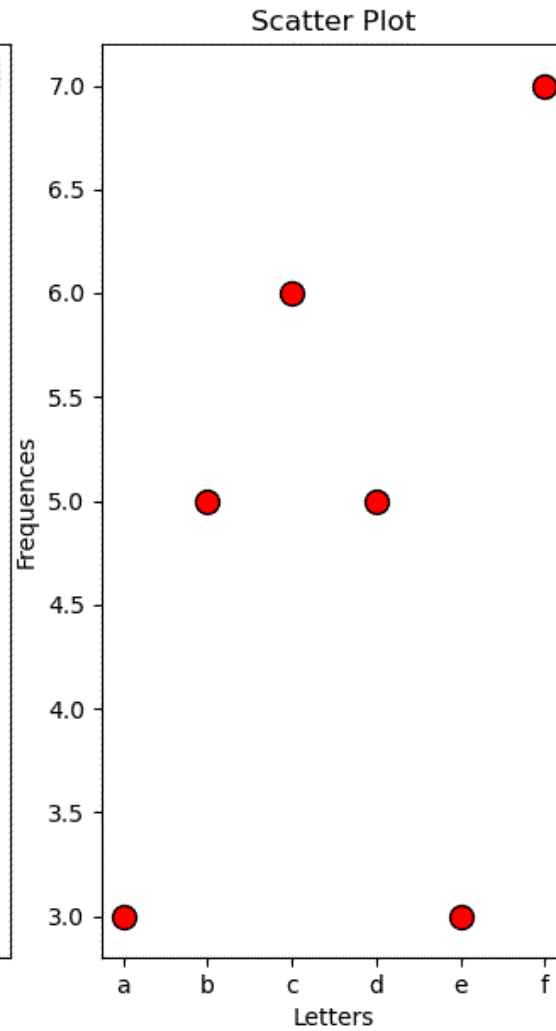
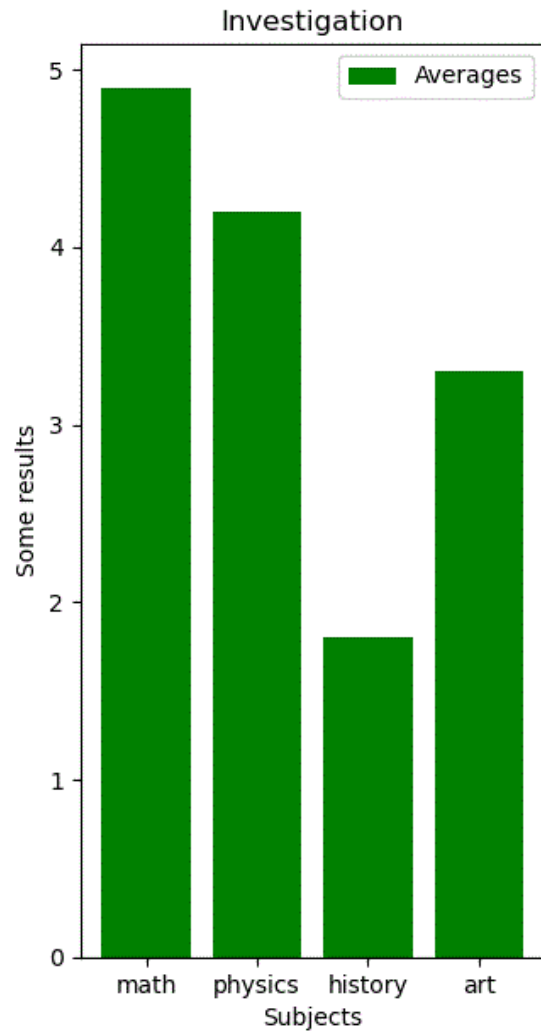
- antal x-värden == antal y-värde
- Linestyle (ls) t ex '-', '--', '-.', ':', 'None', 'solid', 'dashed', 'dashdot', 'dotted', 'D',...
- Marker t ex ^ (triangle), o (circle), x, s (square), p (pentagon)
- Color t ex 'y', 'm', 'c', 'r', 'g', 'b', 'w' and 'k'
- Linewidth
- ...

Exempel Ex3 visar hur man kan plotta flera diagram i samma fönster

Se <https://matplotlib.org/>

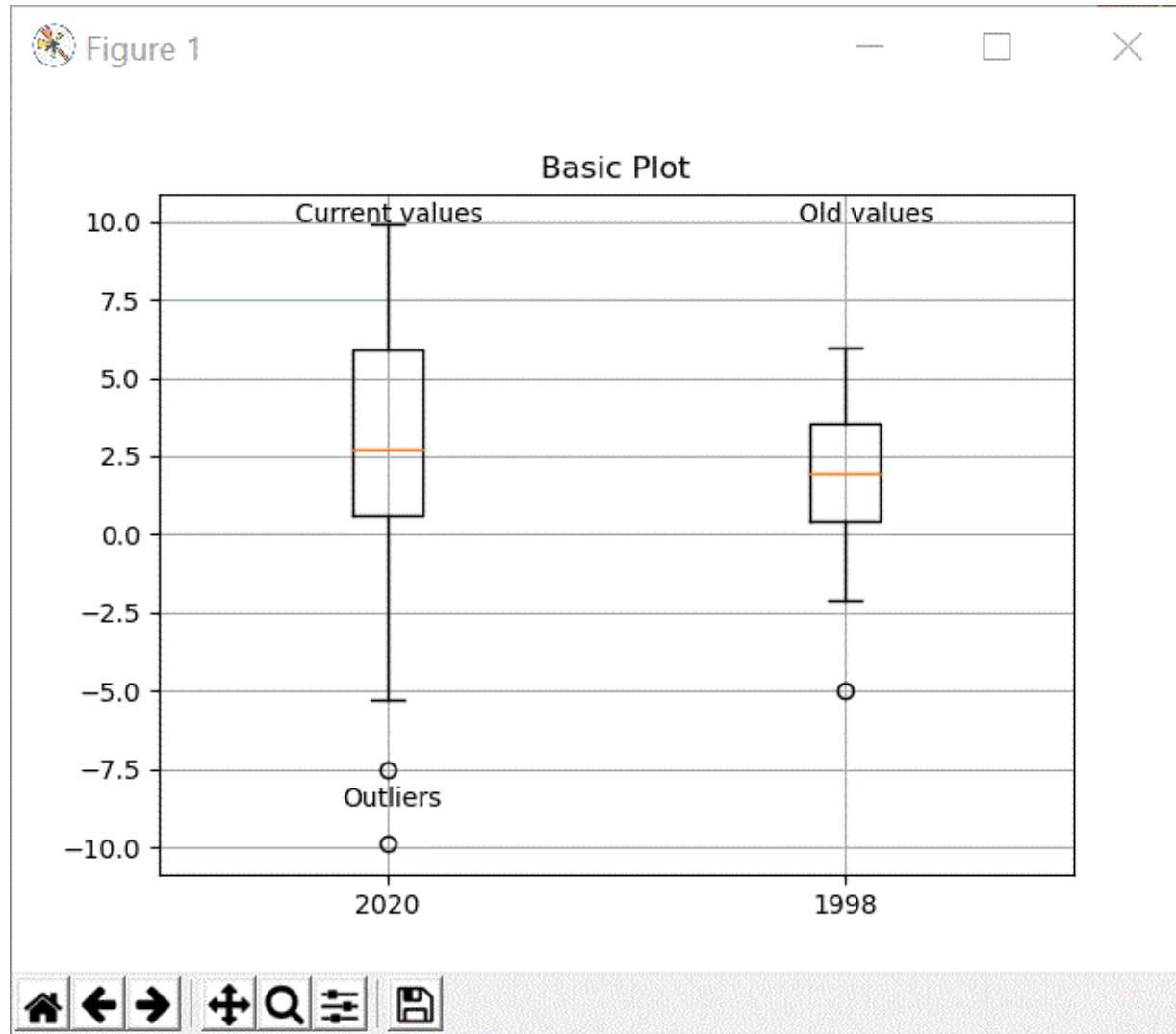
Datavisualisering med matplotlib

Ex3:



Datavisualisering med matplotlib

Ex4:



Datavisualisering med matplotlib

Några viktiga funktioner:

- `plot()`: Används för att plotta punkter eller 'linje'
- `label()`: Används för att ge namn till graferna
- `xlabel()/ylabel()`: Används för att namnge axlarna (x och y) i grafen.
- `title()`: Sätter rubrik på grafen.
- `annotate()`: Lägg in text på angivna koordinater
- `legend()`: Används för att de label som satts ska visas.
- `grid()`: Ger rutnät
- `show()`: Används för att visa upp hela grafen.

Datavisualisering med matplotlib

Exempel på olika diagram:

`plt.plot(...)` linjediagram

`plt.bar(...)` stapeldiagram

`plt.scatter(...)` punktdiagram/spridningsdiagram

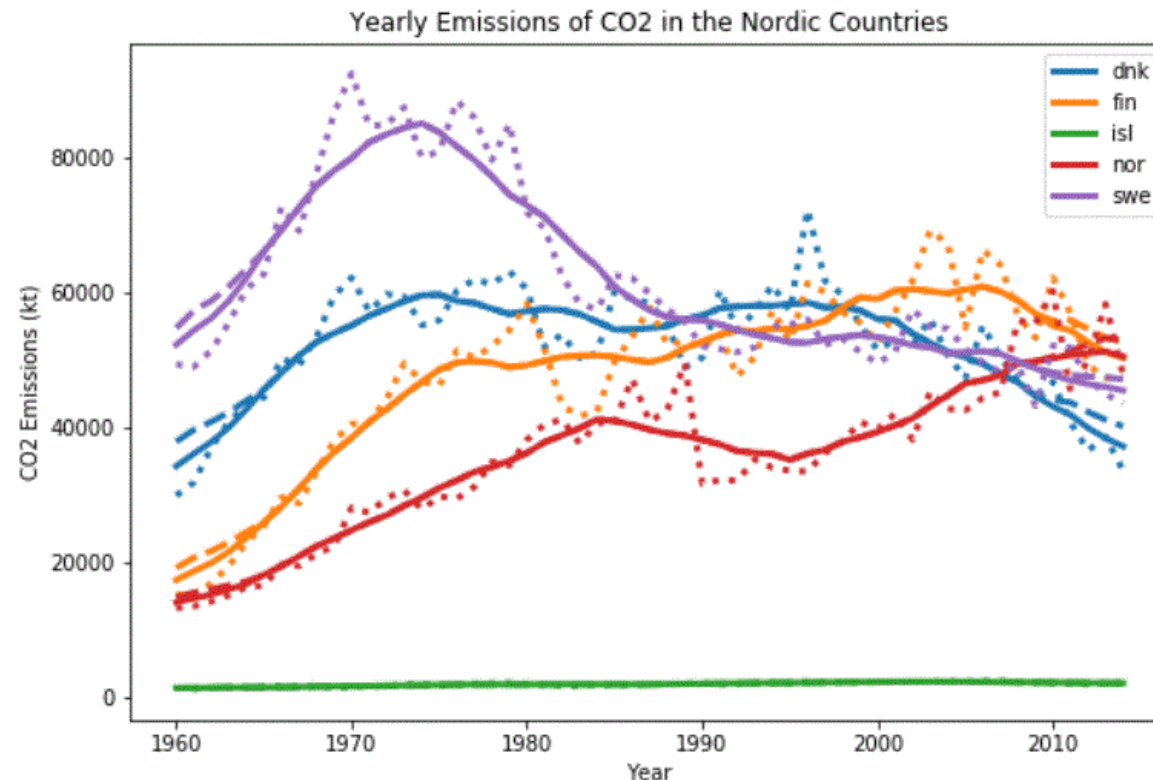
`plt.pie(...)` cirkeldiagram

`plt.hist(...)` histogram

`plt.boxplot()` lådagram: max, min, median, 1:a och 3:a kvartilen

matplotlib, lektion 9, uppgift B

1. Använd funktionen `load_csv` från uppgift A och rita med plot-funktionen CO2-utsläppen för Danmark, Finland, Island, Norge och Sverige för åren 1960 till och med 2014.
2. Minska bruset genom att använda medelvärdesfunktionerna `smooth_a` och `smooth_b` från Lektion 6.



matplotlib, lektion 9, uppgift B

FÖRENKLAD ALGORITM, förslag (det fattas detaljer):

diverse kod, t ex from *filnamnet* import load_csv

Gör lista med landskoderna 'dnk', 'fin', 'isl', 'nor', 'swe'

Anropa load_csv -> {AFG: [414.371,491.378,],AGO:[550.05,454.708,...],ALB:[....]}

Loopa igenom lexikonet från funktionen load_csv:

är det landskoden för ett nordiskt land?

spara i så fall datat i ett lexikon

time = list(range(1960, 2015))

fig, ax = plt.subplots()

Loopa igenom nya lexikonet. För varje värdepar:

ax.plot(#vad ska vara på x-axeln och vad på y-axeln? Se figuren

ax.plot(# använd smooth_a

...

matplotlib, lektion 9, uppgift C

Rita upp lådagram (boxplot) för maj månads medeltemperaturer för Uppsala uppdelat på 1700-, 1800-, 1900- och 2000-talen. Hämta data från textfilen uppsala_tm_1722-2019.dat (**OBS! Inte csv-fil**). Ex:

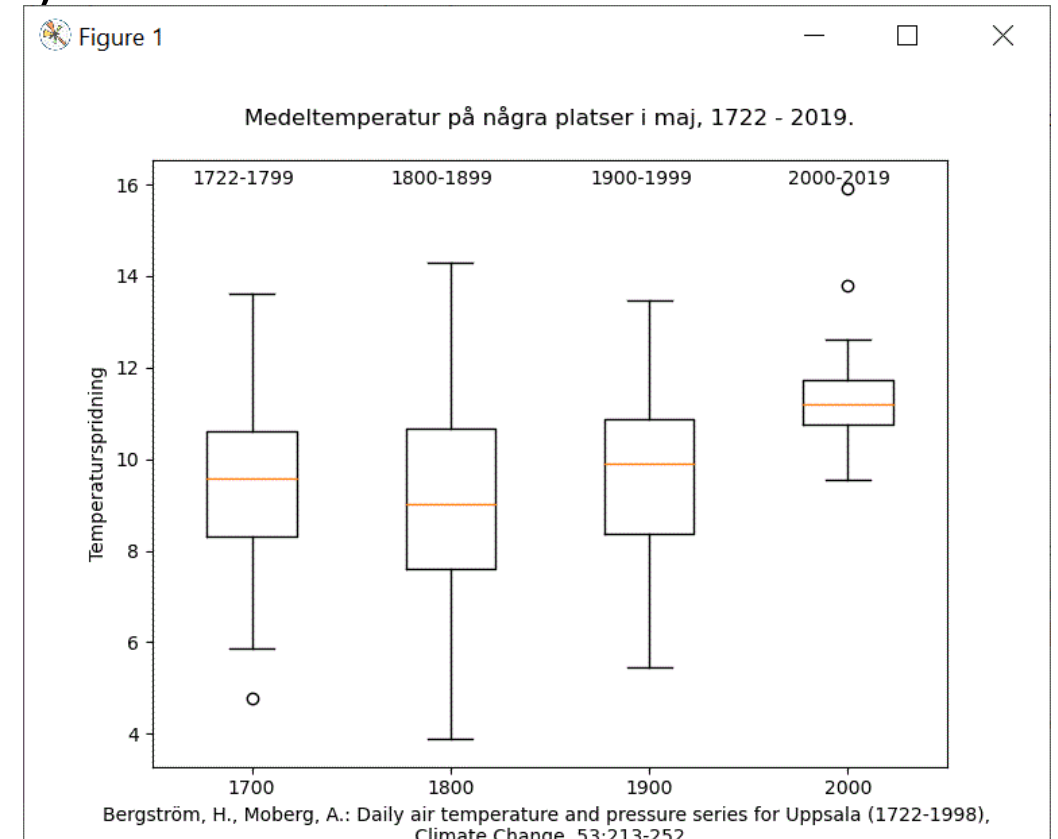
[medeltemp maj 1722, medeltemp maj 1723, ..., medeltemp maj 1799]:

`list_avarage_per_year_1700=[9.8, 8.7, ... , 10.1] # för hela 1700-talet`

`list_century.append(list_avarage_per_year_1700)`

Filen uppsala_tm_1722-2019.dat:

1722	4	27	4.8	4.4	1
1722	4	28	4.8	4.4	1
1722	4	29	5.7	5.3	1
1722	4	30	5.7	5.3	1
1722	5	1	4.6	4.2	1
1722	5	2	3.9	3.5	1
1722	5	3	5.0	4.6	1
1722	5	4	5.6	5.2	1
1722	5	5	6.1	5.7	1
1722	5	6	9.0	8.6	1
1722	5	7	9.8	9.5	1
1722	5	8	11.7	11.4	1
1722	5	9	12.4	12.1	1
1722	5	10	12.3	12.0	1



matplotlib, lektion 9, uppgift C

Öppna filen uppsala_tm_1722-2019.dat för läsning

För varje rad i filen:

- ta bort mellanslag, dvs ta bara med decimaltal. Reguljära uttryck.

- lägg raden i en lista

För varje rad i listan med all data:

- summera temperaturerna i kolumn 4 för maj månad

Beräkna medeltemperaturen.

Spara i en lista

Rita upp



Det fattas saker i algoritmen....

matplotlib, lektion 9, uppgift C

FÖRENKLAD 'ALGORITM', ett förslag (det fattas detaljer):

```
import matplotlib.pyplot as plt, re
```

```
infil = open('uppsala_tm_1722-2019.dat', 'r')  
all_text = [] #Filens innehåll ska sparas i en lista
```

```
for line in infil: # Gör varje rad till en lista utan extra mellanslag:  
    line_text = re.findall(# ta ut alla decimaltal)  
    # Lägg in line_text i all_text
```

```
this_year = int(all_text[0][0]) # börjar år 1722  
#Skapa listor för medeltemp under ett århundrande och de 4 århundranden  
#loopa över all_text:
```

```
    #om det är maj:
```

```
        sum_temp += float(row[3]) # summera temperaturen
```

```
    #Nytt år?
```

```
        Lägg in (sum_temp/31) i lista
```

```
        #Nytt århundrade?
```

```
            Spara listan i fyra_århundradelistan
```

```
#Glöm inte 2000-talet
```

```
#Rita upp de 4 listorna som ligger i århundradelistan som lådagram
```