FY1008_M06

November 11, 2025

1 Matplotlib

matplotlib er hovedpakken i python for plotting. Du har allerede laget enkle plot via denne modulen.

Submodulen *matplotlib.pyplot* er ofte grei og bruke da den forsøker å tilby funksjoner ala MATLAB. For mange av de enklere plot-typen er funksjonene fra denne modulen ofte tilstrekkelig.

Her vil vi bare gi en veldig kort introduksjon….

Noen relevante online resurser er:

- matplotlib.pyplot tutorials
- matplotlib eksempler

Samt videoene: * matplotlib Intro Video 1 * matplotlib Intro Video 2 * matplotlib Intro Video 3

1.0.1 Inport

[145]:
```
# Numpy
import numpy as np

# Matplotlib
import matplotlib.pyplot as plt
```

1.0.2 Standard xy-plot

Dette har du trolig brukt allerede.

[ ]:
```
# Function
Sinc = lambda x: np.sinc(x/np.pi)


# Data
x  = np.linspace(-20,20,401)
y  = Sinc(x)
yy = np.sinc(x)


# Plot y
plt.plot(x,y,label='y')
```

```
plt.title(r"$\sin(x)/(x)$")
plt.xlabel('x')
plt.xlabel('y')


# Plot yy
#plt.plot(x,yy,label='yy')
#plt.legend()
```

Man kan kontrollere mange ulike parametre knyttet til hvordan plottet skal se ut.

[ ]:
```
# Create a figure and an axes object
fig, ax = plt.subplots()

# Plot the data
ax.plot(x, y)

# Set ticks to point inwards
#ax.tick_params(direction='in')

# Optionally, you can also set ticks on the top and right sides to be visible
# and point inwards
#ax.tick_params(top=True, right=True, direction='in')

# Add labels and title
ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
ax.set_title("Plot with Inward Ticks")

# Display the plot
plt.show()
```

Mange ulike parametre kan kontrolleres via ax-variabelen!

Hva nå om man ønsker seg å plotte de to funksjonene $y(x)$ og $yy(x)$ med delt x-akse? Dvs vi ønsker oss figurene overfor hverandre.

[Se også matplotlib.pyplot.subplots].

[ ]:
```
plt.figure()

# Øverset plot
plt.subplot(211)
plt.plot(x, y, 'b--')

# Nederste plot
plt.subplot(212)
plt.plot(x,yy,'r-')
```

Tilsvarende for 2×2 figurer:

[ ]:
```
plt.figure()

# top left
plt.subplot(221)
plt.plot(x, y, 'b--')

# top right
plt.subplot(222)
plt.plot(x,yy,'r-')

# bottom left
plt.subplot(223)
plt.plot(x, -y, 'g--')

# bottom right
plt.subplot(224)
plt.plot(x,-yy,'y-')
```

Men, hvordan kontrollerer man område mellom plottene…..

[ ]:
```
plt.figure()

# Øverset plot
plt.subplot(211)
plt.plot(x, y, 'b--')

# Nederste plot
plt.subplot(212)
plt.plot(x,yy,'r-')

# Modifiser oppsettet
#plt.subplots_adjust(hspace=0)
```

Hva om vi ønsker ticks på alle aksene.

[ ]:
```
# 1. Create a figure and axes
fig, ax = plt.subplots()

# Plot the data
ax.plot(x, y)

# 2. Enable ticks on the top and right sides
ax.tick_params(top=True, right=True)

# 3. Inward ticks
ax.tick_params(direction='in')

# Punktene 2 og 3 kan ev. kombineres
```

```
#ax.tick_params(top=True, right=True, direction='in' )

# Add titles and labels for context
ax.set_title("Ticks on all four sides")
ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")

plt.show()
```

1.1 Plotting in 3D

Mange eksempler på 3D plotting finner du her.

[161]:
```
# Data
x = np.linspace(-3.0, 3.0, 256)
y = np.linspace(-3.0, 3.0, 256)

X, Y = np.meshgrid(x, y)

Z1 = np.exp(-X**2 - Y**2)
Z2 = np.exp(-(X - 1)**2 - (Y - 1)**2)
Z = (Z1 - Z2) * 2
```

[ ]:
```
plt.close()
plt.figure()

#fig, ax = plt.subplots()
#ax = fig.add_subplot(projection='3d')    # Må ha denne!

ax = plt.figure().add_subplot(projection='3d')

ax.plot_surface(X, Y, Z)

ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')
```

1.1.1 Contour plots

[ ]:
```
def f(x, y):
    return np.sin(x) ** 10 + np.cos(10 + y * x)
# end function

x = np.linspace(0, 5, 50)
y = np.linspace(0, 5, 40)

X, Y = np.meshgrid(x, y)
```

```
Z = f(X, Y) # temperature

Nlevels=50
plt.contourf(X, Y, Z, Nlevels, cmap='RdGy')

# Lag color bar
#cbar = plt.colorbar()
```

Det er mange color maps å velge i fra.

[ ]:
```
# En annen colormap
plt.contourf(X, Y, Z, Nlevels, cmap='jet')


# Lag color bar
cbar = plt.colorbar()
cbar.set_label('Temperature [C]')
# Ev. plt.colorbar().set_label('Temperature [C]')

plt.xlabel('x [mm]')
plt.ylabel('y [mm]')
plt.title('Temperatur fordeling')
```

1.1.2 Endring av default setup for matplotlib

Endringer som skal gjelde for alle plot, kan gjøres ved å endre parametrene i *plt.rcParams*

[ ]:
```
plt.rcParams['lines.linewidth'] = 3      # endrer linje tykkelsen

# Ticks på alle akser
plt.rcParams['xtick.top'] = True
plt.rcParams['xtick.bottom'] = True
plt.rcParams['ytick.left'] = True
plt.rcParams['ytick.right'] = True

# Tick directions
plt.rcParams['xtick.direction'] = 'in'
plt.rcParams['ytick.direction'] = 'in'
```

Eventuelt kan mange parametre i plt.rcParams bli oppdatert samtidig ved bruk av *plt.rcParams.update*:

[ ]:
```
# Dictionary med nye tick settings
new_rc_params = {
    'xtick.top': True,
    'ytick.right': True,
    'xtick.direction': 'in',
    'ytick.direction': 'in',
```

```
    'xtick.labelsize': 12,
    'ytick.labelsize': 12,
    'xtick.major.size': 8,
    'ytick.major.size': 8
}

# Oppdaterer tick parameterene
#plt.rcParams.update( new_rc_params )
```

Disse endringene vil gjelde for alle plot (inntil parametrene blir redefinert).

[ ]:
```
# Data
x  = np.linspace(-20,20,401)
y  = Sinc(x)
yy = np.sinc(x)

# Produser grafen
plt.plot(x,y)
```

[ ]:
```
plt.figure()

# Øverset plot
plt.subplot(211)
plt.plot(x, y, 'b--')

# Nederste plot
plt.subplot(212)
plt.plot(x,yy,'r-')
```

1.1.3 Mer informasjon

Det er `uendelig` mye mer informasjon og tricks som man kan bruke for å lage illustrative of spennende plots. Det er umulig å dekke alle mulighetene her, men matplotlib eksempel sidene er et bar startpunkt for å utforske dette videre.