scipy.optimize.curve_fit

Chih-Keng Hung

February 17, 2025

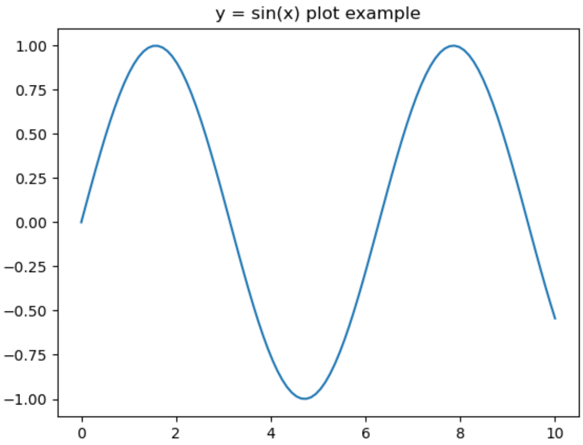
1 Plotting example

Code:

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 10, 100)
y = np.sin(x)
plt.plot(x, y)
plt.title('y = sin(x) plot example')
plt.show()
```

Output:



2 Fitting example

Code:

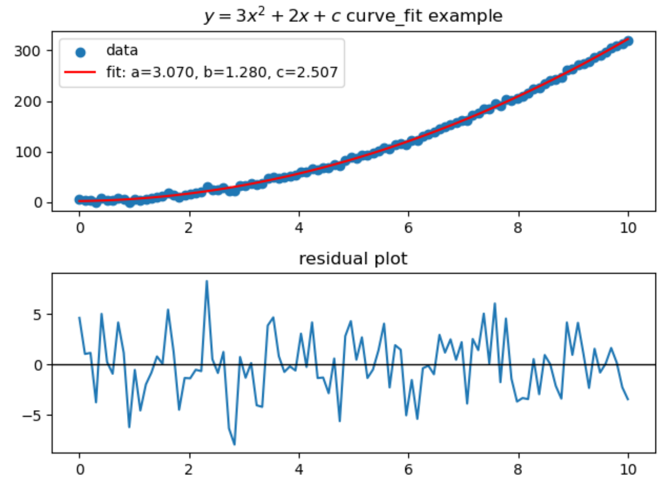
```
from scipy.optimize import curve_fit
import numpy as np
import matplotlib.pyplot as plt

def func(x, a, b, c):
    return a*x**2+b*x+c
```

```

a = 3
b = 2
c = 1
sigma0 = 3
x = np.linspace(0, 10, 100)
y = func(x, a, b, c) + np.random.normal(0, sigma0, len(x))
popt, pcov = curve_fit(func, x, y)
fig, ax = plt.subplots(2, 1)
ax[0].scatter(x, y)
ax[0].plot(x, func(x, *popt), 'r-')
ax[0].legend(['data', 'fit: a=%5.3f, b=%5.3f, c=%5.3f' % tuple(popt)])
ax[0].set_title('$y = 3x^2+2x+c$ curve_fit example')
ax[1].plot(x, y-func(x, *popt))
ax[1].axhline(0, color='black', lw=1)
ax[1].set_title('residual plot')
fig.tight_layout()
plt.show()
```

Output:



3 Practice 1

Purpose:

1. To practice the basic usage of `curve_fit` in `scipy.optimize`.

Task:

1. Generate a set of data points defined by a function $y = a * x + b + noise$ with `x = np.linspace(1, 10, 10)`.
2. The noise is sampled from a normal distribution with mean $= 0$ and standard deviation $= \sigma_0$.(define σ_0 whatever you like)
3. Use the `curve_fit` function in `scipy.optimize` to fit the data points with the function $y = a * x + b$.
4. Plot the data points and the fitting curve.
5. Calculate the residuals and plot the residuals.

Hint:

1. curve_fit example is provided above.

Write your code:

```python
from scipy.optimize import curve_fit
import numpy as np
import matplotlib.pyplot as plt

# Define the function

# Generate the data points

# Fit the data points

# Plot the data points and the fitting curve

# Calculate the residuals and plot the residuals

plt.show()
```

4 Practice 2

Purpose:

1. To read out the data from a csv file.

2. To store the data properly for further analysis.

Task:

1. Read out the data from a csv file.

2. Plot the data points.

3. The following is an example of how to read out the data from a csv file.

4. You can modify the code to read out the data from the file you have.

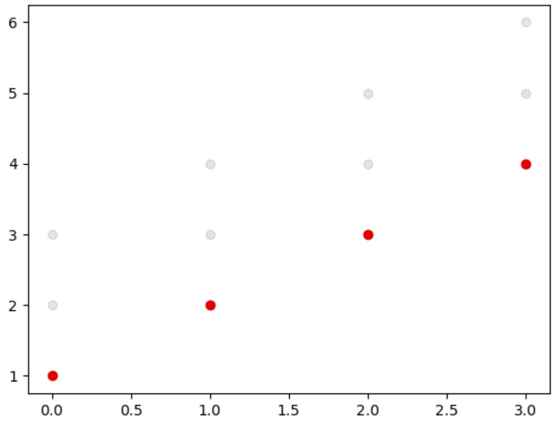
Write your code base on the following example:

```python
import numpy as np
import matplotlib.pyplot as plt

path = r"----path to your data file----"
try:
    data = np.loadtxt(path, skiprows=1, usecols=(0, 1), max_rows=2, delimiter=',')  # setting proper skiprows, usecols, max_rows to read data
except:
    pass
#Rearange the data in a proper way you like.
# ox = data[:, 0]??
#or
# ox = data[0, :]??
#oy = data[:, 1]??
#or
# oy = data[1, :]??
ox = [0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3]   #example original data x
oy = [1, 2, 3, 4, 2, 3, 4, 5, 3, 4, 5, 6]   #example original data y
for i in range(3):  #rearrange the data in a proper way
```

```
    if i == 0:
        x = np.array(ox[i*4:(i+1)*4])
        y = np.array(oy[i*4:(i+1)*4])
    else:
        x = np.vstack((x,ox[i*4:(i+1)*4]))
        y = np.vstack((y,oy[i*4:(i+1)*4]))
#Add code to calculate the information you need from the data.
#Print or plot the information you want to check.
print('x shape:', x.shape)
print('y shape:', y.shape)
print('y mean:', np.mean(y, axis=0))
plt.scatter(x[0, :], y[0, :], c='r')
plt.scatter(x, y, c='k', alpha=0.1)
plt.show()
```

Example output:



```
x shape: (3, 4)
y shape: (3, 4)
y mean: [2.5 3.5 4.5 5.5]
```