Python. Dopasowywanie wykresu do danych

B.Kowal

29 listopada 2021

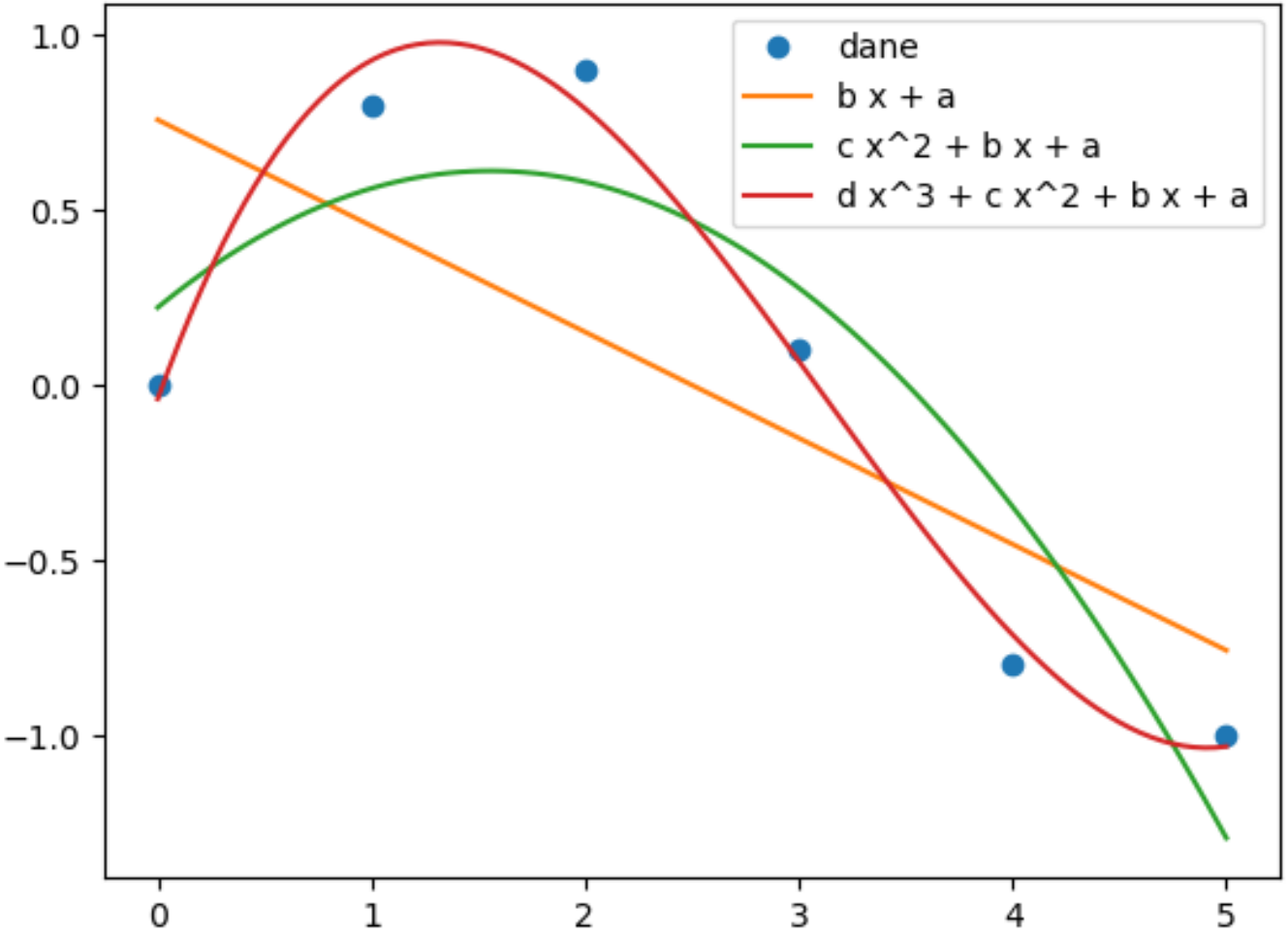
numpy.polyfit - dopasowanie wielomianu metodą najmniejszych kwadratów

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

x = np.array([0.0, 1.0, 2.0, 3.0,  4.0,  5.0])
y = np.array([0.0, 0.8, 0.9, 0.1, -0.8, -1.0])
xp = np.linspace(0, 5, 100)

z1, z2, z3 = np.polyfit(x, y, 1), np.polyfit(x, y, 2), np.polyfit(x, y, 3)
f1, f2, f3 = np.poly1d(z1), np.poly1d(z2), np.poly1d(z3)
plt.plot(x, y, 'o', xp, f1(xp), '-', xp, f2(xp), '-', xp, f3(xp), '-')
plt.legend(['dane',  'b x + a',  'c x^2 + b x + a',  'd x^3 + c x^2 + b x + a'])
plt.show()
```

numpy.polyfit - dopasowanie wielomianu metodą najmniejszych kwadratów



numpy.polyfit, błąd dopasowania

```python
import numpy as np

x = np.array([0.0, 1.0, 2.0, 3.0,  4.0,  5.0])
y = np.array([0.0, 0.8, 0.9, 0.1, -0.8, -1.0])

z = np.polyfit(x, y, 3, full=True)
f = np.poly1d(z[0])
print("Wielomian: \n"+ str(f))
print("Wspolczynniki wielomianu: \n"+ str(z[0]))
print("Suma kwadratow odchylen od wartosci danych: \n"+ str(z[1]))
```

```
Wielomian:
         3          2
0.08704 x - 0.8135 x + 1.693 x - 0.03968
Wspolczynniki wielomianu:
[ 0.08703704 -0.81349206  1.69312169 -0.03968254]
Suma kwadratow odchylen od wartosci danych:
[0.03968254]
```

numpy.poly1d - wielomian jednej zmiennej

```python
import numpy as np

f = np.poly1d([1,3,2,1])
print("Wielomian: \n"+ str(f))
print("Wspolczynniki: \n"+ str(f.c))
print("Stopien: \n"+ str(f.o))
print("Miejsca zerowe: \n"+ str(f.r))
print("Pochodna wielomianu: \n"+ str(f.deriv()))
```

```
Wielomian:
   3     2
1 x + 3 x + 2 x + 1
Wspolczynniki:
[1 3 2 1]
Stopien:
3
Miejsca zerowe:
[-2.32471796+0.j         -0.33764102+0.56227951j -0.33764102-0.56227951j]
Pochodna wielomianu:
   2
3 x + 6 x + 2
```

scipy.interpolate.interp1d - interpolacja funkcjami sklejanymi

```
import numpy as np
from scipy.interpolate import interp1d

x = np.array([0.0, 1.0, 2.0, 3.0,  4.0,  5.0])
y = np.array([0.0, 0.8, 0.9, 0.1, -0.8, -1.0])
xp = np.linspace(0, 5, 100)

f0 = interp1d(x, y, kind='zero')
f1 = interp1d(x, y, kind='slinear')
f2 = interp1d(x, y, kind='quadratic')
f3 = interp1d(x, y, kind='cubic')

plt.plot(x, y, 'o', xp, f1(xp), '-', xp, f2(xp), '-', xp, f3(xp),
↪  '-', xp, f4(xp), '-')
plt.legend(['dane',  'zero',  'slinear', 'quadratic', 'cubic'])
```

scipy.interpolate.interp1d - interpolacja funkcjami sklejanymi