python_basics

November 15, 2022

1 Python:

1.1 basic features

https://www.python.org/

[1]:
```
print("Hello, World!")
```

```
Hello, World!
```

[2]:
```
a = 5
b = 2
```

[3]:
```
a + b
```

[3]: 7

[4]:
```
1 + a * b
```

[4]: 11

[5]:
```
a ** b
```

[5]: 25

[6]:
```
# different in python 3: a//b
# for same behaviour run: from __future__ import division
a / b
```

[6]: 2.5

[7]:
```
a / float(b)
```

[7]: 2.5

[8]:
```
a % b
```

[8]: 1

[9]:
```
min(a, b)
```

[9]:
```
2
```

[10]:
```
a == b
```

[10]:
```
False
```

[11]:
```
a != b
```

[11]:
```
True
```

[12]:
```
a += 3
a
```

[12]:
```
8
```

[13]:
```
#  Python Lists
a = [1, "hello", 5.5]
a
```

[13]:
```
[1, 'hello', 5.5]
```

[14]:
```
len(a)
```

[14]:
```
3
```

[15]:
```
a[2]
```

[15]:
```
5.5
```

[16]:
```
a.append("how are you?")
a
```

[16]:
```
[1, 'hello', 5.5, 'how are you?']
```

[17]:
```
for x in a:
    print(x)
```

```
1
hello
5.5
how are you?
```

[18]:
```
for i, x in enumerate(a):
    print("element {}: {}".format(i, x))
```

```
element 0: 1
element 1: hello
element 2: 5.5
element 3: how are you?
```

[19]:
```
a[0] = 10
a
```

[19]:
```
[10, 'hello', 5.5, 'how are you?']
```

[20]:
```
#  Python Tuples:
b = (-1, "bye", 'c')
b
```

[20]:
```
(-1, 'bye', 'c')
```

[21]:
```
b[-1]
```

[21]:
```
'c'
```

[22]:
```
b[0] = 10
b
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In [22], line 1
----> 1 b[0] = 10
      2 b

TypeError: 'tuple' object does not support item assignment
```

[23]:
```
x, y = b
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In [23], line 1
----> 1 x, y = b

ValueError: too many values to unpack (expected 2)
```

[30]:
```
x
```

[30]:
```
'how are you?'
```

[31]:
```
y
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In [31], line 1
----> 1 y
```

```
NameError: name 'y' is not defined
```

[32]:
```
#  Python Dictionaries (Keys, values)
a = {"name":"Mary", "age":23, "sign":"capricorn"}
a
```

[32]:
```
{'name': 'Mary', 'age': 23, 'sign': 'capricorn'}
```

[33]:
```
a[1]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In [33], line 1
----> 1 a[1]

KeyError: 1
```

[34]:
```
a["job"] = "student"
a
```

[34]:
```
{'name': 'Mary', 'age': 23, 'sign': 'capricorn', 'job': 'student'}
```

[35]:
```
#  Python Funtions
def f(a, b=4, c=5):
    if a > 2 and b < 10:
        return a
    elif c == 5:
        return b
    else:
        return a + b + c
```

[36]:
```
f(4)
```

[36]:
```
4
```

[37]:
```
f(4, 11)
```

[37]:
```
11
```

[38]:
```
f(4, c=6, b=11)
```

[38]:
```
21
```

2 NumPy: multi-dimensional arrays and scientific computing

https://www.numpy.org/

[39]:
```
import numpy as np
```

[40]:
```
a = np.array([0, 2, 4, 6, 8, 10, 12, 14, 16])
a
```

[40]:
```
array([ 0,  2,  4,  6,  8, 10, 12, 14, 16])
```

[41]:
```
a.ndim
```

[41]:
```
1
```

[42]:
```
a.shape
```

[42]:
```
(9,)
```

[43]:
```
a[2]
```

[43]:
```
4
```

[44]:
```
a[2:]
```

[44]:
```
array([ 4,  6,  8, 10, 12, 14, 16])
```

[45]:
```
a[:4]
```

[45]:
```
array([0, 2, 4, 6])
```

[46]:
```
a[2:7]
```

[46]:
```
array([ 4,  6,  8, 10, 12])
```

[47]:
```
a[2:7:2]
```

[47]:
```
array([ 4,  8, 12])
```

[48]:
```
a[-1]
```

[48]:
```
16
```

[49]:
```
a[::-1]
```

[49]:
```
array([16, 14, 12, 10,  8,  6,  4,  2,  0])
```

[50]:
```
a[[0, 4, 5]]
```

[50]:
```
array([ 0,  8, 10])
```

[51]:
```
b = a > 3
b
```

[51]:
```
array([False, False,  True,  True,  True,  True,  True,  True,  True])
```

[52]:
```
a[b]
```

[52]:
```
array([ 4,  6,  8, 10, 12, 14, 16])
```

[53]:
```
a = np.array([[0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11]])
a
```

[53]:
```
array([[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]])
```

[54]:
```
a.ndim
```

[54]:
```
2
```

[55]:
```
a.shape
```

[55]:
```
(3, 4)
```

[56]:
```
a[1, 2]
```

[56]:
```
6
```

[57]:
```
a[0]
```

[57]:
```
array([0, 1, 2, 3])
```

[58]:
```
a[:, 1:3]
```

[58]:
```
array([[ 1,  2],
       [ 5,  6],
       [ 9, 10]])
```

[59]:
```
a.T
```

[59]:
```
array([[ 0,  4,  8],
       [ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11]])
```

[60]:
```
a + 10
```

[60]:
```
array([[10, 11, 12, 13],
       [14, 15, 16, 17],
       [18, 19, 20, 21]])
```

[61]:
```
a ** 2
```

[61]:
```
array([[  0,   1,   4,   9],
       [ 16,  25,  36,  49],
       [ 64,  81, 100, 121]])
```

[62]:
```
a * [10, 20, 30, 40]
```

[62]:
```
array([[  0,  20,  60, 120],
       [ 40, 100, 180, 280],
       [ 80, 180, 300, 440]])
```

[63]:
```
np.sin(a)
```

[63]:
```
array([[ 0.        ,  0.84147098,  0.90929743,  0.14112001],
       [-0.7568025 , -0.95892427, -0.2794155 ,  0.6569866 ],
       [ 0.98935825,  0.41211849, -0.54402111, -0.99999021]])
```

[64]:
```
np.mean(a)
```

[64]:
```
5.5
```

[65]:
```
a.mean(axis=1)
```

[65]:
```
array([1.5, 5.5, 9.5])
```

[66]:
```
np.max(a)
```

[66]:
```
11
```

[67]:
```
np.max(a, axis=1)
```

[67]:
```
array([ 3,  7, 11])
```

[68]:
```
np.arange(10)
```

[68]:
```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

[69]:
```
np.linspace(2, 4, 5)
```

[69]:
```
array([2. , 2.5, 3. , 3.5, 4. ])
```

[70]:
```
np.zeros((2, 3))
```

[70]:
```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

[71]:
```
np.full((2, 3), 2.5)
```

[71]:
```
array([[2.5, 2.5, 2.5],
       [2.5, 2.5, 2.5]])
```

3 matplotlib: plotting

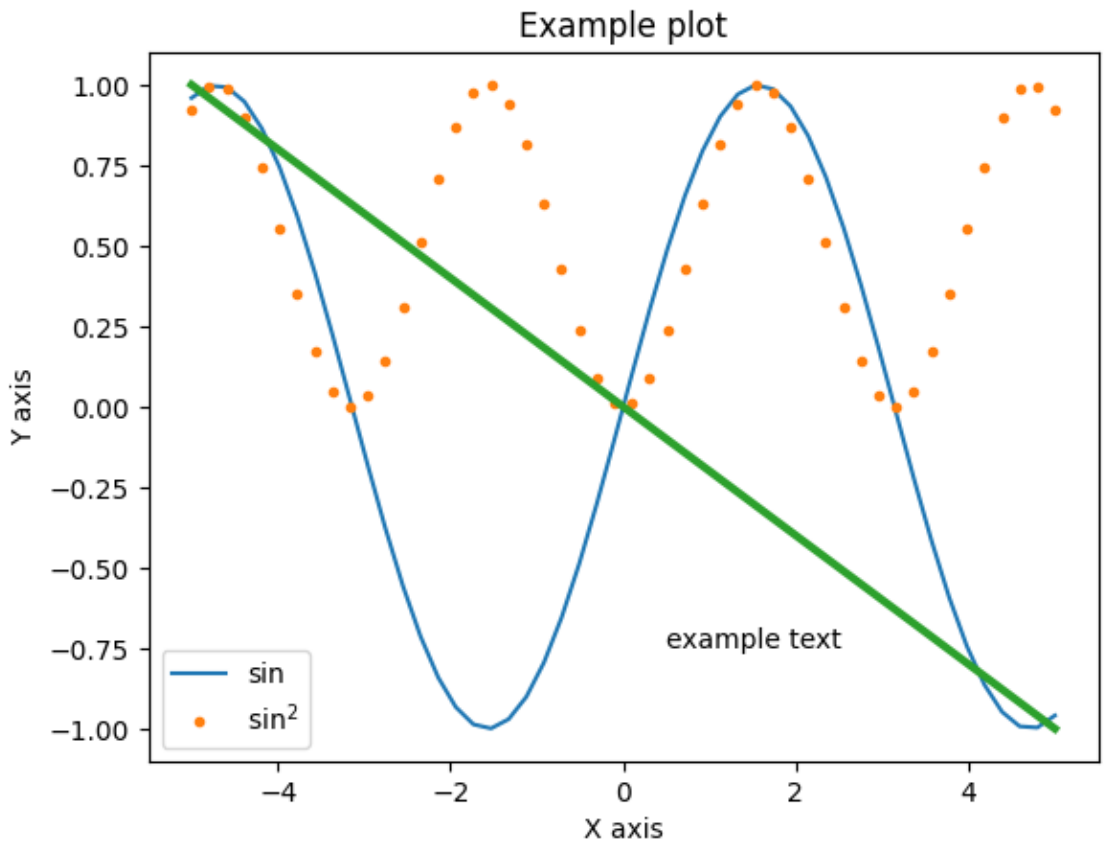
https://matplotlib.org/

[72]:
```
import matplotlib.pyplot as plt
```

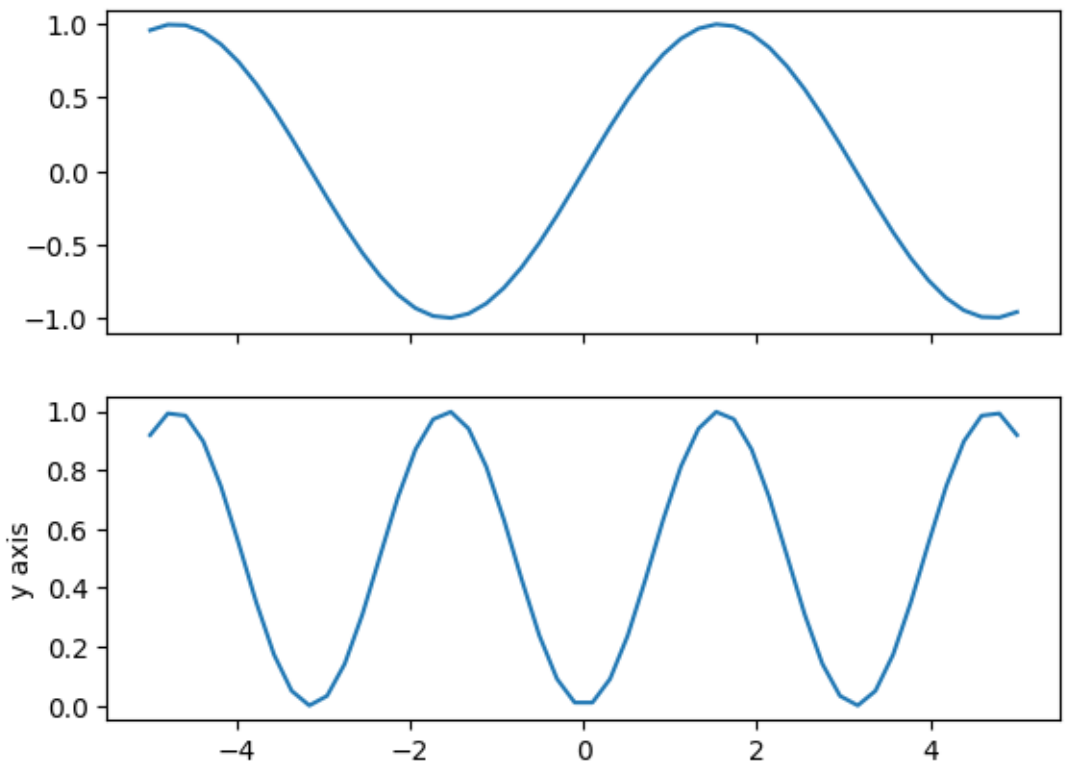
[73]:
```
#%matplotlib notebook
%matplotlib inline
```

[74]:
```
x = np.linspace(-5, 5, 50)
y = np.sin(x)
y2 = y ** 2
y3 = -x / 5
```

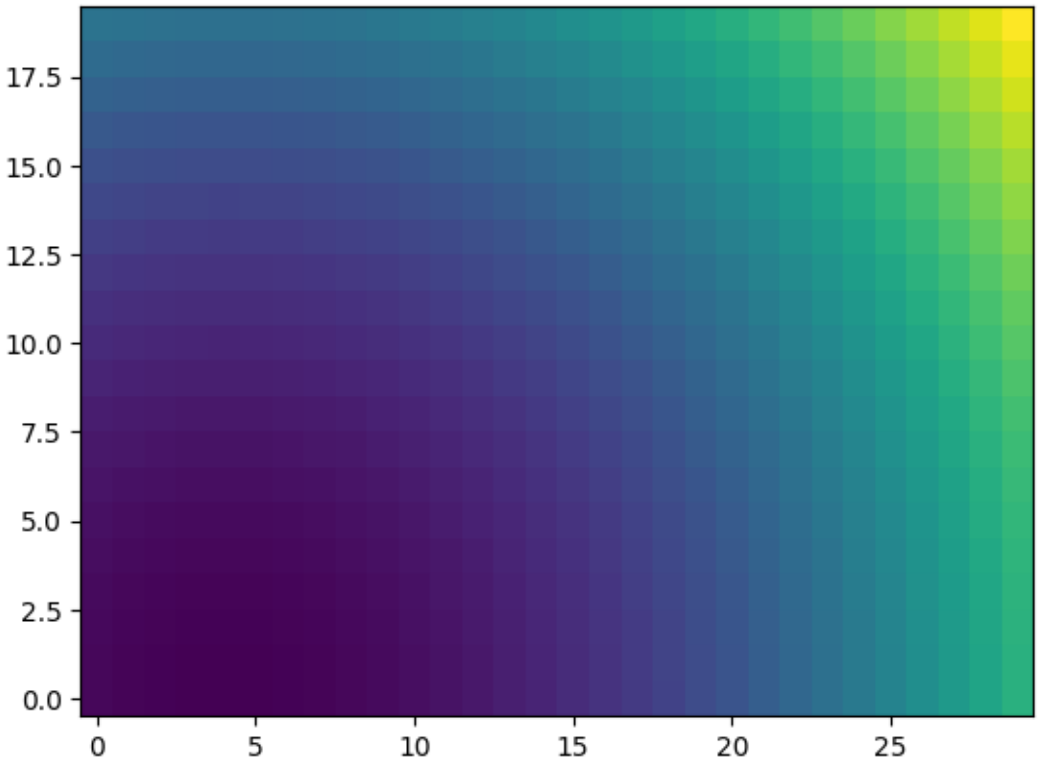
[75]:
```
plt.figure()
plt.plot(x, y, label='sin')
plt.plot(x, y2, '.', label='$\sin^{2}$')
plt.plot(x, y3, linewidth=3)
plt.annotate('example text', xy=(0.5, -0.75))
plt.xlabel("X axis")
plt.ylabel("Y axis")
plt.title("Example plot")
plt.legend()
plt.show()
```



[76]:
```
fig, ax = plt.subplots(2, sharex=True)
ax[0].plot(x, y)
ax[1].plot(x, y2)
ax[1].set_ylabel('y axis')
plt.show()
```



[77]:
```
y, x = np.mgrid[0:20, 0:30]
z = (x - 4)**2+ y**2
plt.figure()
plt.pcolormesh(x, y, z, shading='auto')
plt.show()
```



4 SciPy: extra modules for scientific computation

https://www.scipy.org/

[78]:
```
from scipy.optimize import curve_fit
import numpy as np
import matplotlib.pyplot as plt
```

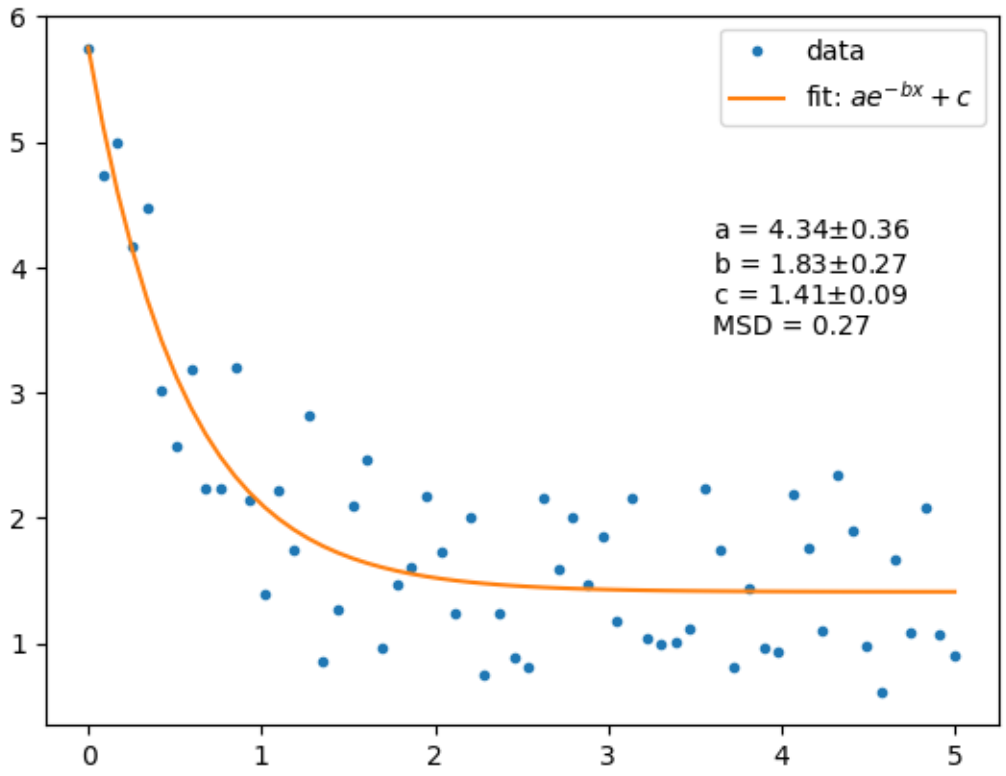
[79]:
```
def f(x, a, b, c):
    return a * np.exp(-b * x) + c
```

[80]:
```
n = 60
x = np.linspace(0, 5, n)
y = f(x, 5, 2, 0.5) + 2 * np.random.rand(n)
```

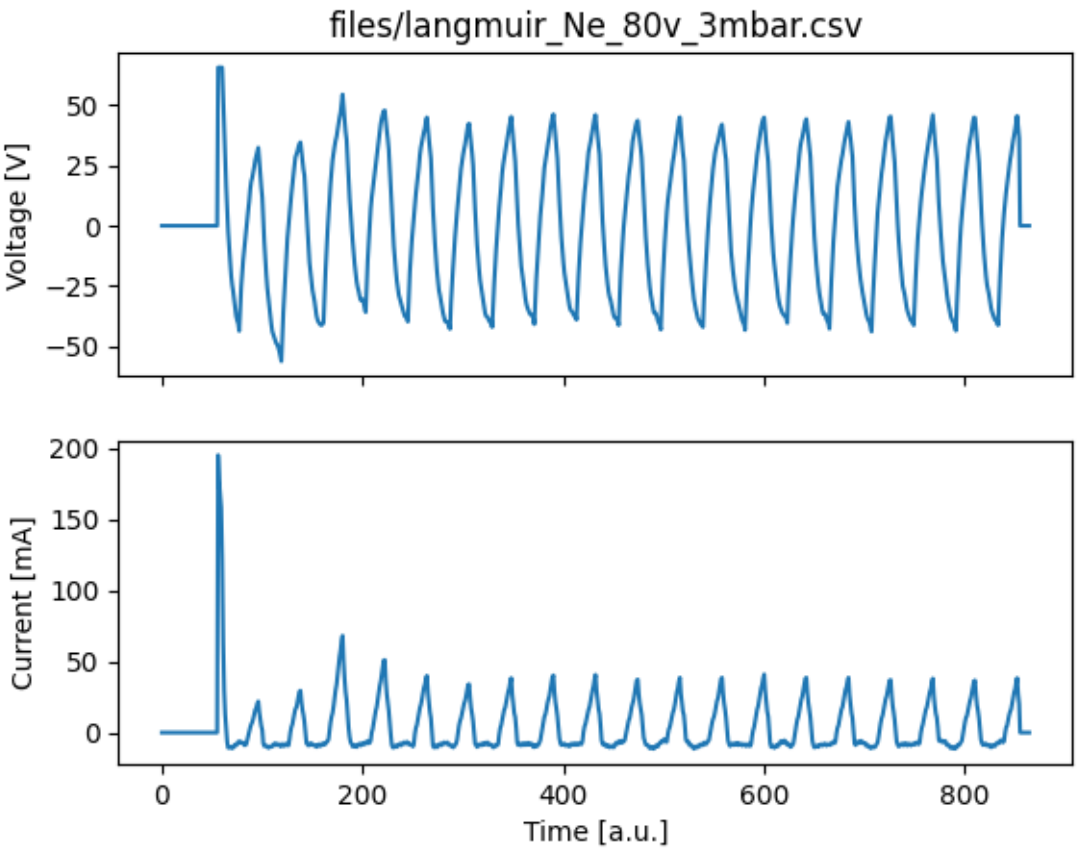
[81]:
```
popt, pcov = curve_fit(f, x, y)
perr = np.sqrt(np.diag(pcov))
y_fit = f(x, *popt)
msd = np.sum((y - y_fit) ** 2) / n
```

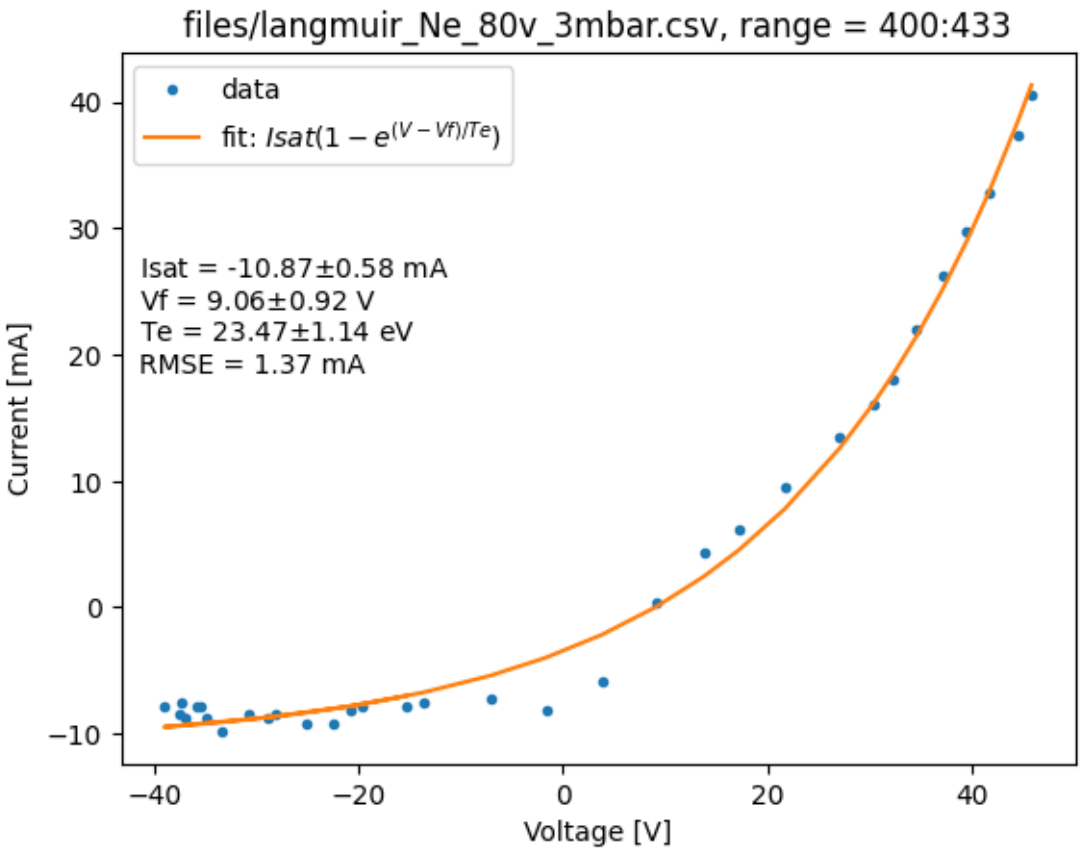
[82]:
```
pnames = ['a', 'b', 'c']
results = ''
for name, value, error in zip(pnames, popt, perr):
    results += '{} = {:.2f}$\pm${:.2f}\n'.format(name, value, error)
results += 'MSD = {:.2f}'.format(msd)

plt.plot(x, y, '.', label='data')
plt.plot(x, y_fit, label='fit: $ae^{-bx} + c$')
plt.annotate(results, xy=(0.7, 0.55), xycoords='axes fraction')
plt.legend()
plt.show()
```



[84]:
```
%run langmuir_fit.py
```


files/langmuir_Ne_80v_3mbar.csv
Voltage [V]
50
25
0
−25
−50
Current [mA]
200
150
100
50
0
0
200
400
600
800
Time [a.u.]




```
<Figure size 640x480 with 0 Axes>
```

[ ]:

[ ]: