Optimization

This lesson discusses how to find the minima of a curve.

WE'LL COVER THE FOLLOWING

Introduction

Optimization, or finding the minima or maxima of a function, is an important field in mathematics. Common applications of optimization are the minimization of entities such as cost, risk, and error, or the maximization of productivity, efficiency, and profit.

The minimization function `fmin` finds the next local minimum starting from a user-provided initial position. `fmin` is part of the `scipy.optimize` module.

Finding local minima

There are two minimum input arguments for `fmin`:

1. The function to minimize.
2. The initial value where we start our search for the minimum.

`fmin` returns the value of the independent variable for which the function is locally minimized.

```
fmin(func, x0)
```

💡 The search for the minimum is local, since the algorithm follows the

local gradient.

Let's find the minima of this function:

$$f(x) = sin(x) - 2e^{-(x-0.4)^2}$$

We will be using a plot of the function's data to best represent it:

```
from scipy import *
from scipy import optimize
import matplotlib.pyplot as plt

def f(x):
    return sin(x) -  (2 * exp(-(x - 0.4)**2))

x = arange(-7, 7, 0.1)
y = f(x)

point1 = -1.0
point2 = 3.0

min1 = optimize.fmin(f, point1)
min2 = optimize.fmin(f, point2)

fig, ax = plt.subplots(figsize=(12, 8))
ax.plot(x, y, label = '$\sin(x)-2e^{-(x-0.4)^2}$')  # plotting curve

ax.plot(min1, f(min1), 's', label='min 1', marker='v', color='r')    # plotting min1
ax.plot(point1, f(point1), 's', label='start 1', marker='o', color='r')     # plotting starti

ax.plot(min2, f(min2), 's', label='min 2', marker='v', color='g')    # plotting min2
ax.plot(point2, f(point2), 's', label='start 2', marker='o', color='g')     # plotting starti

ax.set_xlabel('x')
ax.grid()
ax.set_ylabel('y')
ax.legend(loc='best')
```

We repeat the search for the minimum starting from these two values:

1. `point1` = -1.0
2. `point2` = 3.0

This is to demonstrate that, depending on the starting value, we may find different minima of the function `f()`.

> 💡 The `fmin()` function returns a NumPy array which can only contain

> one number. In general, `fmin()` can be used to find the minimum in a
>
> higher-dimensional function as well. In that case, the NumPy array would contain those coordinates that minimize the objective function.

Minima inside a bound

We can also use the `fminbound()` function. It finds the *minimum* of the function in the given range. `fminbound()` has a different syntax and uses different underlying algorithms. Let's use the example above and find the minima of the given function in the range:

$$-2.0 < x < 6.0$$

```
from scipy import *
from scipy import optimize
import matplotlib.pyplot as plt

def f(x):
    return sin(x) -  (2 * exp(-(x - 0.4)**2))

x = arange(-10, 10, 0.1)
y = f(x)

point1 = -2.0
point2 = 6.0

minima = optimize.fminbound(f, point1, point2)
print(minima)

fig, ax = plt.subplots(figsize=(12, 8))
ax.plot(x, y, label = '$\sin(x)-2e^{-(x-0.4)^2}$')  # plotting curve

ax.plot(minima, f(minima), 's', label='minima', marker='v', color='r')    # plotting min1

ax.set_xlabel('x')
ax.grid()
ax.set_ylabel('y')
ax.legend(loc='best')
```

In line 14, we have called the function `fminbound` and the range given is -2 to 6. In line 20, we have plotted the function as well as the minima point.

In the next lesson, we will learn about computing Fourier transforms.