Plotting with Matplotlib

Pacchetto `matplotlib`

The main plotting package in Python is matplotlib

It can be used to build all sorts of scientific plot

- `matplotlib` is an extensive package

- ...And we'll use just a small part of its capabilities

If you are interested, the online documentation is very well done

Plots are built in `matplotlib` by calling functions

- A function is used to construct an (empty) figure

- ...Then other functions populate the figure with graphical elements

- ...And yet other functions can modify the figure properties

We'll see how to use a few key functions step by step

Preparing Plot

First, we need to import the `ptplot` module

```
In [14]:   from matplotlib import pyplot as plt
```

- The alias `plt` is usually employed for `pyplot` when importing

Then, we build a new figure using `figure`

```
In [15]:   plt.figure(figsize=(20, 3))
```

```
Out[15]:   <Figure size 2000x300 with 0 Axes>

           <Figure size 2000x300 with 0 Axes>
```

- We can specify the size witht the (optional) `figsize` parameter
- Size is specify as a tuple in the `(width, height)` form

The `plot` Function

We can draw curves using the `plot` function

In this example, we'll use it to draw the function: $f(x) = \sin(x) + 0.1x$

- First, we build arrays with the x and y coordinates of points on the curve

```python
In [16]: import numpy as np

x = np.linspace(0, 10, 12)
y = np.sin(x) + 0.1 * x
n = 4
print(f'First {n} elements of x: {x[:n]}')
print(f'First {n} elements of y: {y[:n]}')

First 4 elements of x: [0.         0.90909091 1.81818182 2.72727273]
First 4 elements of y: [0.         0.87985455 1.15137413 0.67529476]
```

- We use `linspace` to define a sequence of x values

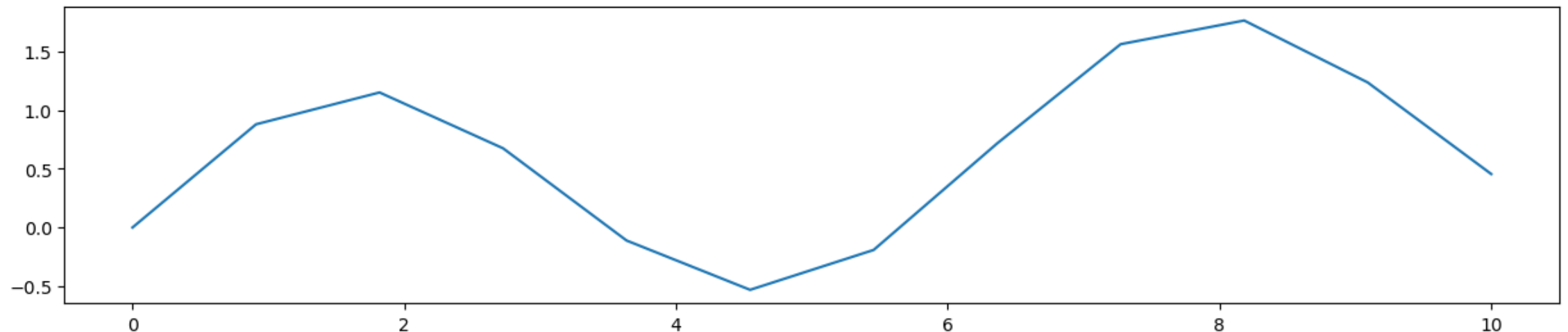- ...Then we use `numpy` operations to evaluate the function for those points

The `plot` Function

Now we can call `plot(x, y, ...)` to draw the curve

- The x and y parameters are iterables with x and y coordinates

- The function draws a curve by connecting adjacent points

```
In [17]:  plt.figure(figsize=(15, 3))
          plt.plot(x, y)
          plt.show()
```
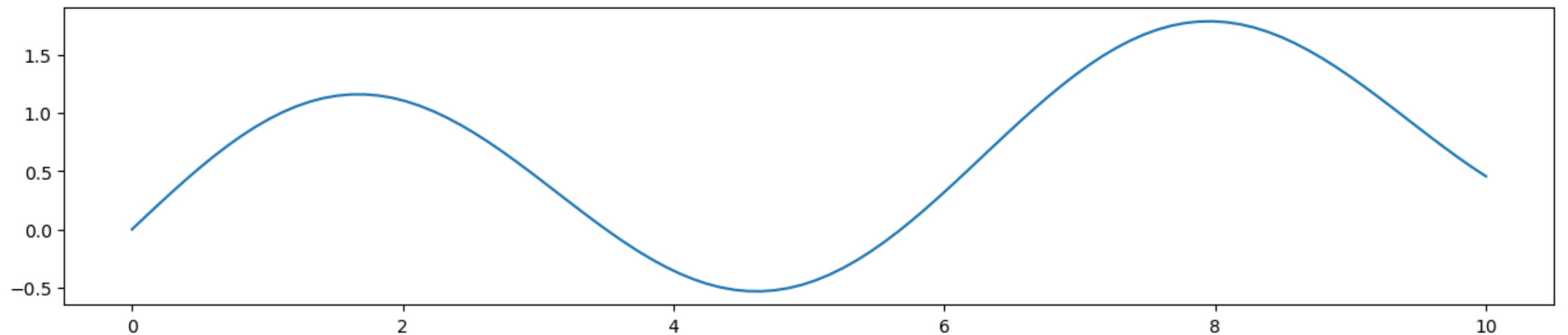
The `plot` Function

We can improve the plot quality by just using more points

```
In [19]: x = np.linspace(0, 10, 100)
         y = np.sin(x) + 0.1 * x

         plt.figure(figsize=(15, 3))
         plt.plot(x, y)
         plt.show()
```
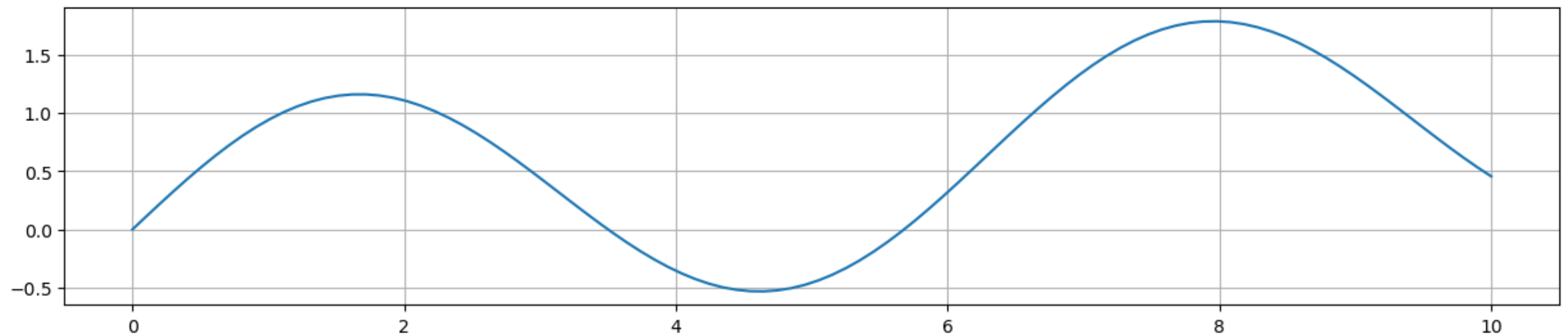
The `plot` Function

We can add a reference grid by calling `grid`

```
In [20]: x = np.linspace(0, 10, 100)
         y = np.sin(x) + 0.1 * x

         plt.figure(figsize=(15, 3))
         plt.plot(x, y)
         plt.grid()
         plt.show()
```
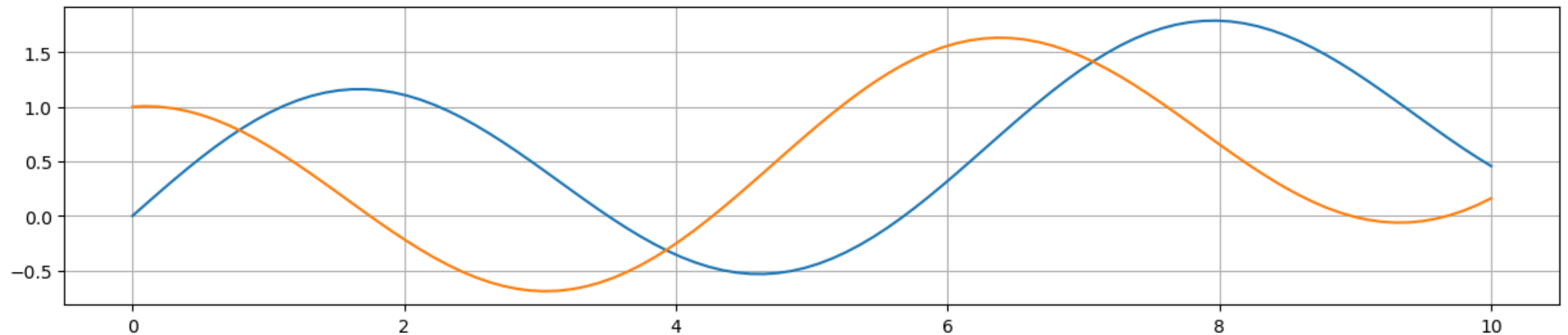
The `plot` Function

We can also draw multiple curves on the same plot

```python
In [22]: x = np.linspace(0, 10, 100)
         y = np.sin(x) + 0.1 * x
         y2 = np.cos(x) + 0.1 * x

         plt.figure(figsize=(15, 3))
         plt.plot(x, y)
         plt.plot(x, y2)  # Disegno la seconda curva
         plt.grid()
         plt.show()
```
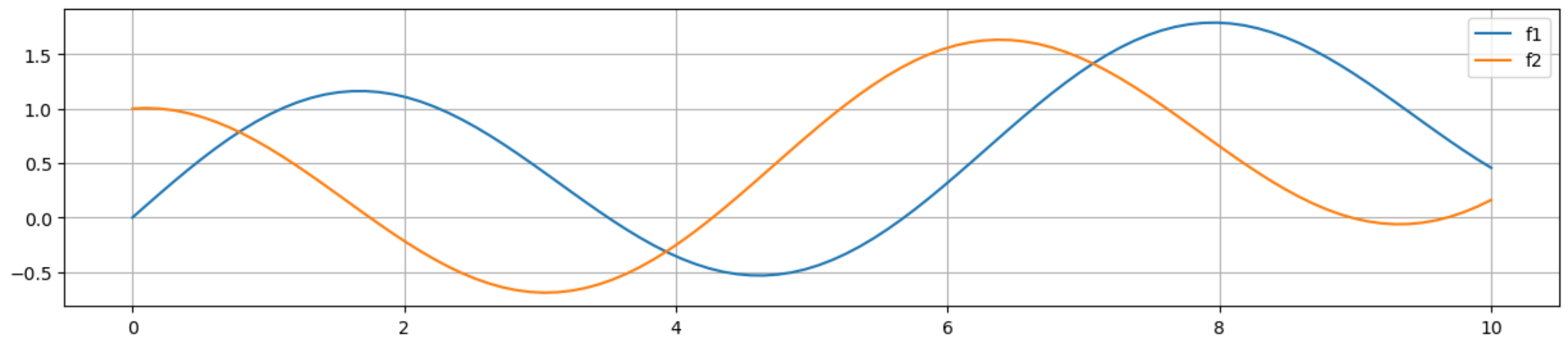
The `plot` Function

We can assign a label to each curve, the call draw a legend with `legend`

```python
In [24]: x = np.linspace(0, 10, 100)
         y = np.sin(x) + 0.1 * x
         y2 = np.cos(x) + 0.1 * x

         plt.figure(figsize=(15, 3))
         plt.plot(x, y, label='f1')
         plt.plot(x, y2, label='f2')
         plt.grid()
         plt.legend()
         plt.show()
```

Exercise

Consider the function $1.5x^3 - 1.5x^2 - x - 0.4$

- Draw it on the interval $[-1, 1.5]$, by using a different number of points

In [ ]:

Exercise

Compare on the $[0, 10]$ interval

...The functions x^4 and $e^x - 1$

```
In [ ]:
```

Exercise

Visualli determione for which x values the following function is 0

$$\sin(x) - \frac{1}{2}(e^x - 1)$$

■ Proceed by drawing the curve and checking where it crosses the $x - axis$

```
In [ ]:
```