THE NUMPY ARANGE FUNCTION

SHARP SIGHT

WHAT YOU'LL LEARN

- What the NumPy arange function is

- The syntax of `np.arange()`

- How to create arrays that contain "ranges" of numbers
  - simple sequences from "start" to "stop"
  - more complex sequences

A quick introduction to numpy arange

The numpy arange function creates arrays with "ranges" of numbers

Sequences that start at 0 and increase up to a stopping point

0	1	2	3

Sequences that start at a specific number and increase

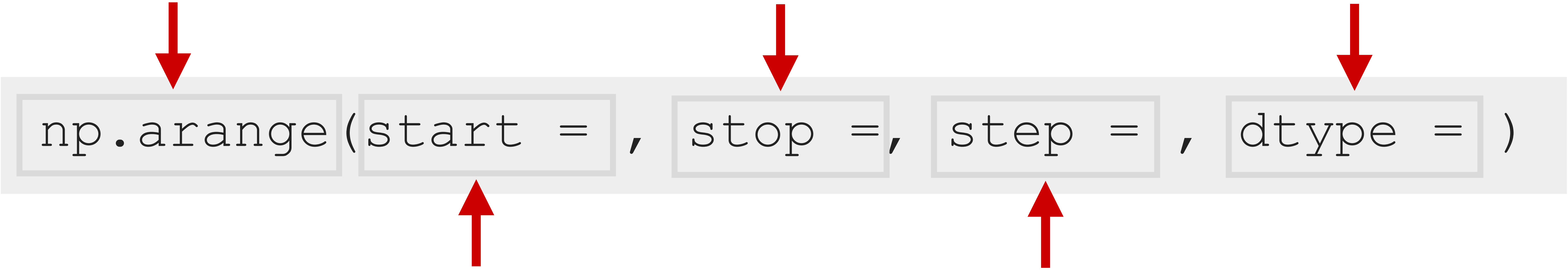
5	6	7	8

Sequences that start at a specific number and increase by a specific step value

10	20	30	40

The syntax of np.arange

THE SYNTAX OF NUMPY ARANGE



The end of the interval

(This number is *excluded* from the output)

The function name

The data type (optional)

```
np.arange(start = , stop =, step = , dtype = )
```

The start of the interval (optional)

The "step" between values (optional)

THE PARAMETERS OF NUMPY ARANGE

Parameter	What it controls	Default	Required?
`start`	The starting number of the sequence	0	No
`stop`	The end point of the sequence (the sequence will *not* include this number)		Yes
`step`	The size of the "step" between the values	`1`	No
`dtype`	The data type of the output	Inferred	No

EXAMPLES OF NUMPY ARANGE

Create a sequence of numbers from 0 to 5

```
np.arange(start = 0,stop = 6)
array([0, 1, 2, 3, 4, 5])


np.arange(6)
array([0, 1, 2, 3, 4, 5])
```

You can remove the explicit parameters and just use the stop number

This is faster to type, but harder to read …

SPECIFY A SPECIFIC START AND STOP POSITION

Here, we're using the `start` parameter and the `stop` parameter to specify the start and stop point of the sequence

```
np.arange(start = 5, stop = 11)
array([ 5,  6,  7,  8,  9, 10])
```

SPECIFY A SPECIFIC STEP VALUE FOR THE SEQUENCE

Remember: the stop number is *not* included … if we want our sequence to go to 40, the stop number must be at least 1 greater

The step parameter indicates that we will increment in steps of 10

```
np.arange(start = 10, stop = 41, step = 10)
array([10, 20, 30, 40])
```

Specify a specific output data type

The dtype parameter enables us to specify the data type of the output

```
np.arange(start = 1, stop = 6, dtype = float)
array([1., 2., 3., 4., 5.])
```

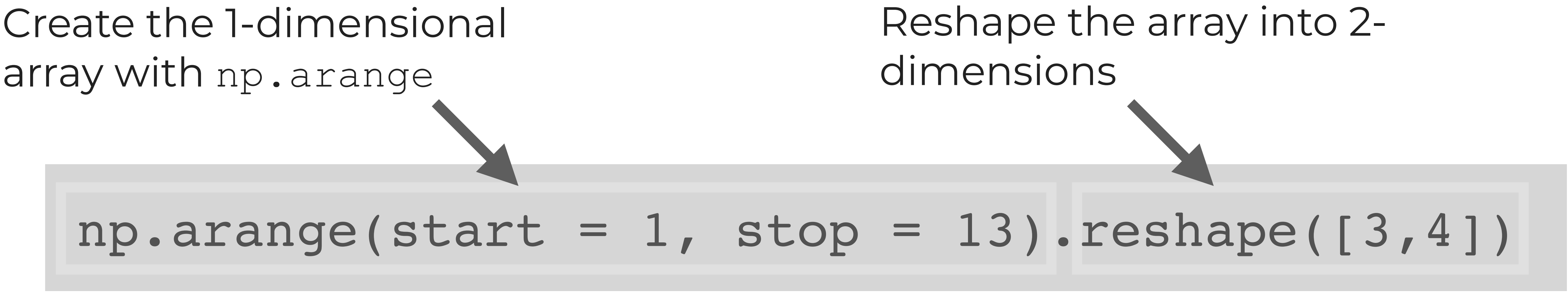
How to create 2-dimensional arrays with numpy arange

How to create a 2-dimensional array with np.arange

- Numpy arange does not have a way to create 2-dimensional arrays

- You need to combine multiple NumPy tools
  - `np.arange`
  - the NumPy reshape method

To create 2D arrays, combine np.arange with numpy reshape



Create the 1-dimensional array with `np.arange`

Reshape the array into 2-dimensions

```
np.arange(start = 1, stop = 13).reshape([3,4])
```

OUT:

1	2	3	4
5	6	7	8
9	10	11	12

Recap

Recap of what we learned

- The `np.arange` function creates sequences of numbers

- We can control the sequence with the parameters
  - `start`
  - `stop`
  - `step`

- Create 2D sequences by combining `np.arange` and the `reshape` method