

## Plotting Stuff

```
In [ ]: import numpy as np
X = np.linspace(0, 10, 40)
print(X)
```

```
In [ ]: type(X)
```

```
In [ ]: C = np.cos(X) # applies function to the vector, heavily optimized
print(C)
```

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

```
In [ ]: # only needed in jupyter notebook
%matplotlib inline
```

```
In [ ]: _ = plt.plot(X, C) # Notice the underscore
```

```
In [ ]: S = np.sin(X)
print(S)
```

```
In [ ]: plt.figure()
plt.plot(X, C)
_ = plt.plot(X, S)
```

```
In [ ]: plt.figure()
plt.plot(X, C, label='cos')
plt.plot(X, S, label='sin')
plt.xlabel('x')
plt.ylabel('sin(x)/cos(x)')
plt.legend()
```

## Case study: Time taken by fib(n)

```
In [6]: def fibslow(n): # our old fib function
        if n <= 1 :
            return n

        else:
            return fibslow(n-1) + fibslow(n-2)

def fibfast(n): # Just another algo for computing fib
    if n <= 1: return n

    a, b = 0, 1

    for i in range(1, n+1):
        a, b = b, a + b

    return a
```

Beyond Basic Programming - Intermediate Python

recluze.net/learn

```
In [7]: def time_function(fn, X):
        from datetime import datetime

        times = []
        for i in X:
            start_time = datetime.now()

            _ = fn(i) # actual call to the function, don't care about value

            end_time = datetime.now()
            time_taken = end_time - start_time # returns a timedelta
            time_taken = time_taken.microseconds

            times.append(time_taken)

        return times
```

```
In [8]: X = range(1, 30)

        fibslow_times = time_function(fibslow, X)
        fibfast_times = time_function(fibfast, X)

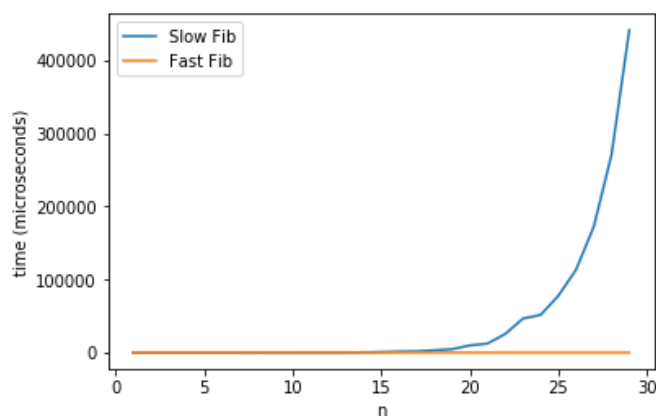
        print("Slow: ", fibslow_times)
        print("Fast: ", fibfast_times)

Slow: [10, 15, 3, 5, 7, 10, 44, 30, 38, 58, 94, 139, 242, 417, 857, 1439, 1810, 3131, 4611, 97
17, 12260, 25439, 46601, 51586, 77910, 113543, 172739, 271235, 442046]
Fast: [4, 4, 3, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 2, 5, 5, 4, 3, 3, 3, 3, 3, 13, 6, 5, 5, 6]
```

```
In [9]: import matplotlib.pyplot as plt
        %matplotlib inline
```

```
In [10]: plt.figure()
         plt.plot(X, fibslow_times, label='Slow Fib')
         plt.plot(X, fibfast_times, label='Fast Fib')
         plt.xlabel('n')
         plt.ylabel('time (microseconds)')
         plt.legend()
```

Out[10]: <matplotlib.legend.Legend at 0x112cdcc88>



In [ ]: