CP.1.1.6, Sauer3

Use the Bisection Method to calculate the solution of $\cos x = \sin x$ in the interval [0,1] within six correct decimal places. Please derive the number of steps needed for convergence in advance.

CP.1.1.6, Sauer3, solution, Langou

- See
  https://colab.research.google.com/drive/1t_uyY621T6yoBfAOCjZpq-O2sTzV5XFO
- Our `bisect` Python code is at: http://math.ucdenver.edu/~langou/4650/4650.git/bucket/bisect.py.html

Let

$$\begin{array}{rcl} f : [0,1] & \to & \mathbb{R} \\ x & \mapsto & \cos(x) - \sin(x). \end{array}$$

We note that f is continuous on $[0,1]$, And we note that

$$f(0) = \cos(0) - \sin(0) = 1 \text{ so that } f(0) > 0,$$

and

$$f(1) = \cos(1) - \sin(1) \approx -0.30 \text{ so that } f(1) < 0.$$

So that Bisection Method will converge to a root $x_\star$ such that $f(x_\star) = 0$.
We derive the number of steps needed for convergence.

```
from math import log2
from math import ceil
a = 0
b = 1
tol = 1e-6
k = ceil( log2( b - a ) - log2(tol) - 1 )
numevals = k + 2
print( k )
print( numevals )
```

```
19
21
```

We use Bisection Method to get a root in the interval [0,1] within six correct decimal places.

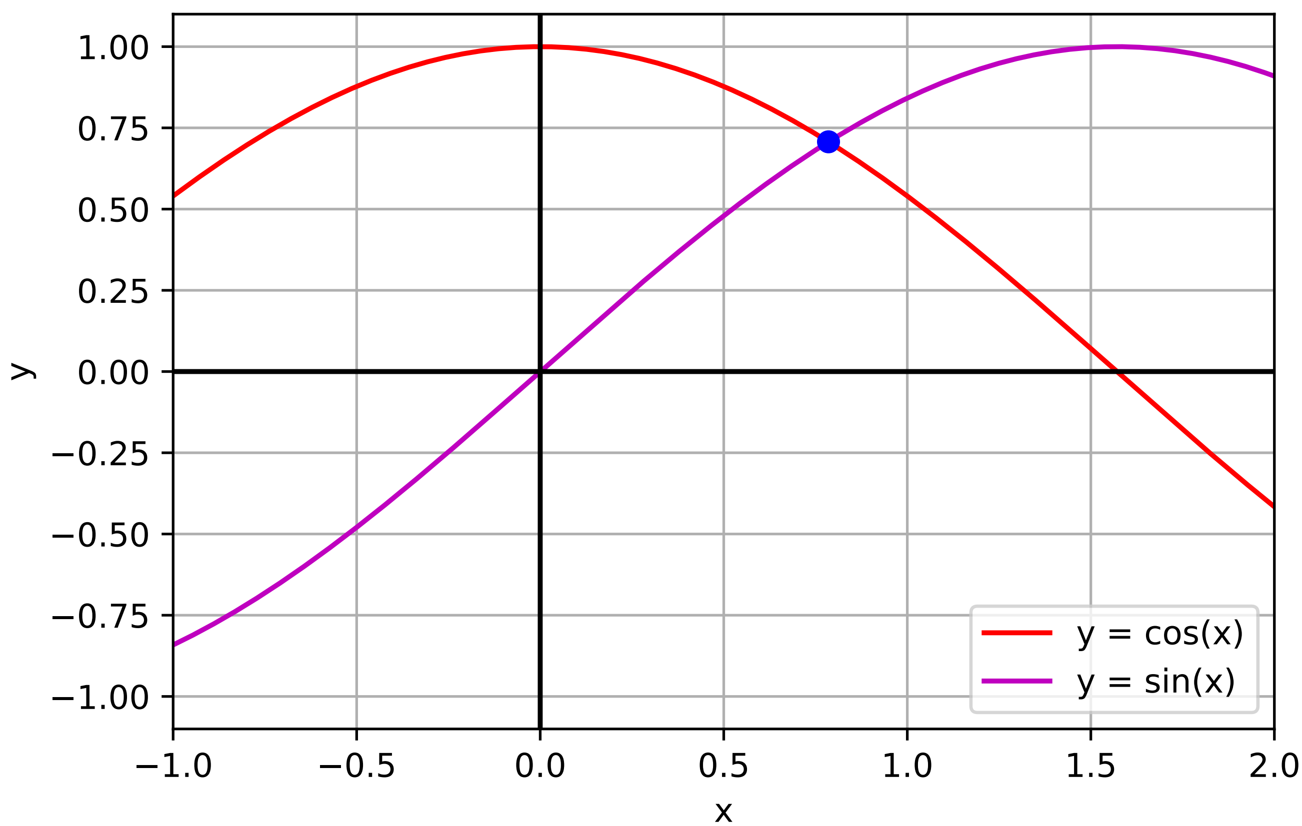
```
from math import sin
from math import cos
import scipy.optimize
import numpy as np
```

```
f = lambda x : cos(x) - sin(x)

x_fsolve = scipy.optimize.fsolve( f, 0. )[0]
print( x_fsolve )

x_bisec, err_bound, numeval = bisect( f, 0., 1., 1e-6, False )
print( x_bisec, err_bound[-1], numeval )
```

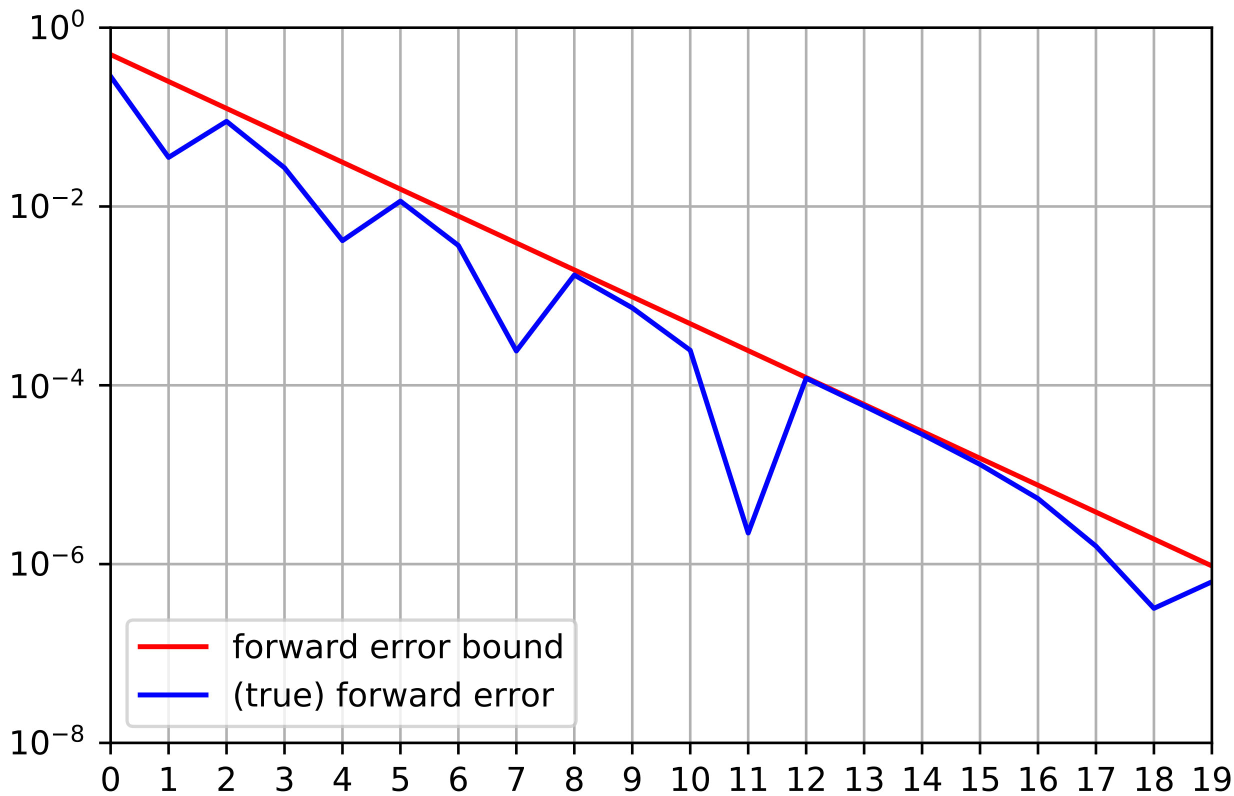
```
0.7853981633974484
0.7853975296020508 9.5367431640625e-07 21
```

Not needed for full credit

We can plot cos and sin and see that indeed they intersect in $[0, 1]$



We can plot the true forward error and the associated forward error bound.



10^0
10^{-2}
10^{-4}
10^{-6}
10^{-8}
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
forward error bound
(true) forward error