Newton's Method

April 20, 2022

1 Newton's Method.

In numerical analysis, Newton's method, also known as the Newton–Raphson method, named after Isaac Newton and Joseph Raphson, is a root-finding algorithm which produces successively better approximations to the roots of a real-valued function.

The most basic version starts with a single-variable function f defined for a real variable x, the function's derivative f , and an initial guess x_0 for a root of f. If the function satisfies sufficient assumptions and the initial guess is close. The process is the repeating of :

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

1.0.1 Implementation

We define `newton_method` function

[1]:
```
def newton_method(f, guess = 0, number_of_iterations = 10):

    for i in range(number_of_iterations):
        guess = guess - (f(guess) / f.derivative()(guess))

    return guess
```

1.1 Test with an example

Use Newton's method to find a solution to the equation

$$x^2 + ln(x) = 0$$

using 5 iterations and initial value $x_\circ = 0.5$.

Find maximum error bound ε, error measured to exact solution and plot $f(x)$

[2]:
```
# Define f(x)

f(x) = x^2 + ln(x)
```

[3]:
```
our_solution = newton_method(f, guess = 0.5, number_of_iterations = 5)
our_solution
```

[3]: 0.652918640419205

[4]:
```
exact_solution = find_root(f, 0.5, 1)
exact_solution
```

[4]: 0.6529186404192052

Find error and maximum error bound ε

[5]:
```
error = abs(our_solution - exact_solution)
error
```

[5]: 4.44089209850063e-16

[6]:
```
maximum_error_bound = (1 - 0.5) / 2^5
maximum_error_bound
```

[6]: 0.0156250000000000

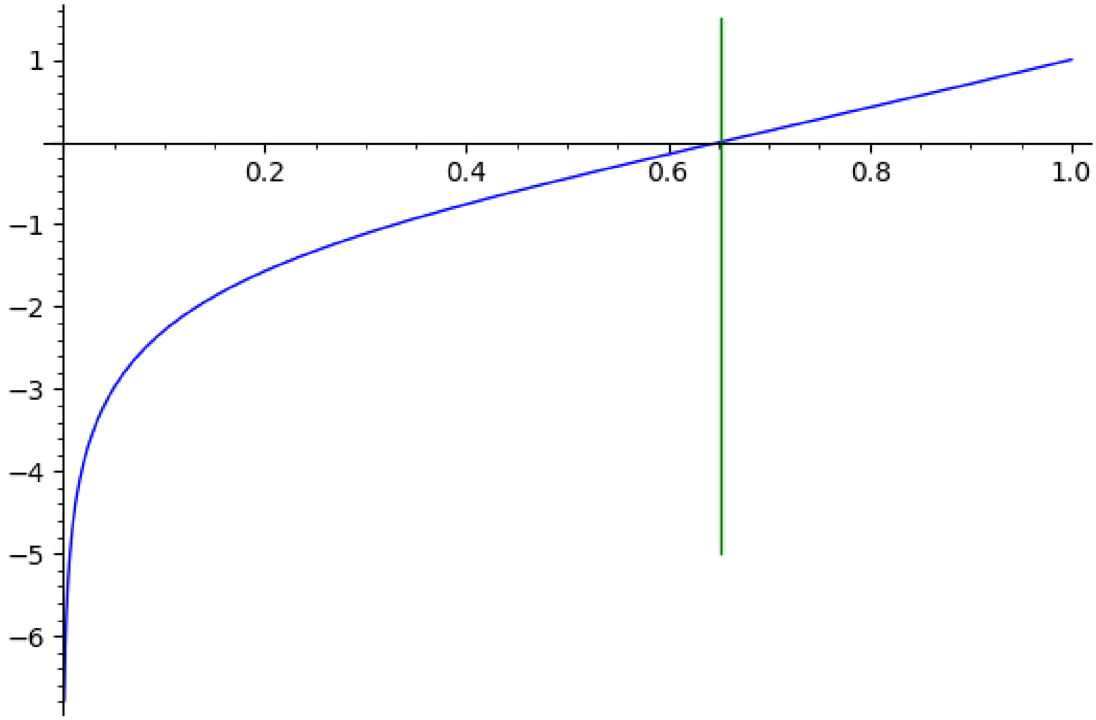
As seen, error is much less than maximum error bound

plot the function and the solution

[7]:
```
p1 =  plot (f, x)

p2 = line([(our_solution, -5), (our_solution, 1.5)], color = "green")

my_plot = p1 + p2

my_plot
```

```
verbose 0 (3835: plot.py, generate_plot_points) WARNING: When plotting, failed
to evaluate function at 100 points.
verbose 0 (3835: plot.py, generate_plot_points) Last error message: 'can't
convert complex to float'
```

[7]:



1
0.2
0.4
0.6
0.8
1.0
−1
−2
−3
−4
−5
−6