Homework 3

Lea Hibbard

February 17, 2024

Problem 1

Consider the equation $2x - 1 = sin(x)$

Solution

(a) Find a closed interval $[a, b]$ on which the equation has a root r, and use the Intermediate Value Theorm to prove that r exists.
First, $f(x)$ must be continuous on some interval $[-\pi, \pi]$.

$$\lim_{x \to r} f(x) = f(r)$$

$$\lim_{x \to r} 2x - sin(x) - 1 = 2r - sin(r) - 1$$

This is true for all $r \in [-\pi, \pi]$ so $f(x)$ is continuous on $[-\pi, \pi]$.
Next we will evaluate $f(-\pi)$ and $f(\pi)$.

$$\begin{aligned} f(-\pi) &= -2\pi - sin(-\pi) - 1 \\ &= -2\pi - 1 < 0 \end{aligned}$$

$$\begin{aligned} f(\pi) &= 2\pi - sin(\pi) - 1 \\ &= 2\pi - 1 > 0 \end{aligned}$$

Note $f(-\pi) < 0, f(\pi) > 0$, so $f(-\pi)f(\pi) < 0$. By the Intermediate Value Theorem, $f(-\pi) < u < f(\pi)$, where $u = f(r) = 0$. Therefore a root, r, must exist.

(b) Prove that r is the only root of the equation (on all of $\mathbb{R}$.

$$\begin{aligned} f'(x) &= 2 - cos(x) \\ -1 \leq &\ cos(x) \leq 1 \\ 1 \leq &\ f'(x) \leq 3 \end{aligned}$$

Therefore, $f(x)$ is always increasing so r must be the only root of the equation.

(c) The following is from a modified bisection method (modified from class example) that counts the number of iterations.

Calling Script

```
import numpy as np
import matplotlib.pyplot as plt
from bisection_example import bisection

f = lambda x: 2*x-1-np.sin(x)
[astar,ier, count]  = bisection(f, -np.pi, np.pi, 1e-8)
print('the approximate root is',astar)
print('the error message reads:',ier)
print('f(astar) =', f(astar))
print('this method took', count, 'iterations')
```

Approximation and Iterations

```
The approximate root is 0.8878622154822129
f(astar) = 5.354353072029028e-09
This method took 29 iterations.
```

Problem 2

The function $f(x) = (x-5)^9$ has a root (with multiplicity 9) at $x = 5$ and is monotonically increasing (decreasing) for $x > 5$ ($x < 5$) and should thus be a suitable candidate for your function above. Use $a = 4.82$ and $b = 5.2$ and $tol = 1e-4$ and use bisection with:

Solution

(a) $f(x) = (x-5)^9$
From bisection method:

```
The approximate root is 5.000073242187501
The error message reads: 0
f(astar) = 6.065292655789404e-38
This method took 11 iterations
```

Code

```
f2 = lambda x: (x-5)**9
a = 4.82
b = 5.2
tol = 1e-4
[astar, ier, count] = bisection(f2,a,b,tol)
print('the approximate root is',astar)
print('the error message reads:',ier)
print('f(astar) =', f2(astar))
print('this method took', count, 'iterations')
```

(b) $f(x) = x^9 - 45x^8 + \ldots - 1953125$
From bisection method:

```
the approximate root is 5.12875
the error message reads: 0
f(astar) = 0.0
this method took 3 iterations
```

Code

```
f2expanded = lambda x: x**9 - 45*x**8 + 900*x**7 - 10500*x**6 + 78750*x**5 - 39375
[astar, ier, count] = bisection(f2expanded,a,b,tol)
print('the approximate root is',astar)
print('the error message reads:',ier)
print('f(astar) =', f2expanded(astar))
print('this method took', count, 'iterations')
```

(c) Here we can see that the expanded function performs worse, giving a relative error of

$$\text{Relative error (expanded)} = \frac{|5.12875 - 5|}{|5|} = 0.02575$$

as compared to part a, with a relative error of

$$\text{Relative error} = \frac{|5.000073242187501 - 5|}{|5|} = 1.4648437500142108 \times 10^{-5}$$

This is because when we use the expanded function, we are multiplying, adding, and subtracting very large numbers, which results in loss of precision, making our evaluation of the expanded function significantly more inaccurate than the condensed form.

Problem 3

(a) Use a theorem from class (Theorem 2.1 from text) to find an upper bound on the number of iterations in the bisection needed to approximate the solution of $x^3 + x - 4 = 0$ lying in the interval $[1, 4]$ with an accuracy of 10^{-3}.

(b) Find an approximation of the root using the bisection code from class to this degree of accuracy. How does the number of iterations compare with the upper bound you found in part (a)?

Solution

(a) **Theorem 2.1** Suppose that $f \in C[a, b]$ and $f(a)f(b) < 0$. The Bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approximating a zero p of f with

$$|p_n - p| \leq \frac{b-a}{2^n} \text{ when } n \geq 1.$$

$$a = 1 \; , \; b = 4 \; , \; |p_n - p| = 10^{-3}$$

Plugging these values in, we get

$$10^{-3} \leq \frac{4-1}{2^n}$$
$$n \leq 11.55074679$$

Therefore the maximum number of iterations needed is 12.

(b)
```
The approximate root is 1.378662109375
The error message reads: 0
f(astar) = -0.0009021193400258198
This method took 11 iterations
```

The actual number of iterations required was less than the upper bound but was still close.

Problem 4

For each of the following iterations figure out if they converge to x^*. If so, what is the order of convergance? If linear, what is the rate?

Solution

We will approximate each iteration as a Taylor series, with $x_{n+1} = g(x)$ where generally:

$$g(x) = x_{n+1} = g(x^*) + g'(x^*)(x_n - x^*) + \frac{g''(x^*)}{2}(x_n - x^*)^2 + \ldots$$

Where $g(x^*) = x^*$ by definition, since x^* is a fixed point.
Rearranging:

$$\frac{x_{n+1} - x^*}{x_n - x^*} = g'(x^*) + \frac{g''(x^*)}{2}(x_n - x^*) + \ldots$$

- If $\left|g'(x^*)\right| > 1, |x_{n+1} - x^*| > |x_n - x^*|$, so as $n \to \infty$, x_n **diverges** away from x^*.
- If $|g'x| < 1$, $|x_{n+1} - x^*| < |x_n - x^*|$ so as $n \to \infty$, $x_n \to x^*$. Thus the iteration **converges**
- For $g^n(x^*) = 0$ for all $n \in [1, N]$ where N is an integer, and $g^{N+1}(x^*) \neq 0$, $\alpha = N + 1$ (where α is the order of convergence, as per the definition given in the original problem statement).
  - As an example, if $\left|g'(x^*)\right| = 0$ and $\left|g''(x^*)\right| \neq 0$, rearranging the Taylor series results in
  $$\frac{|x_{n+1} - x^*|}{|x_n - x^*|^2} = \frac{g''(x^*)}{2}$$
  so as $n \to \infty$, the iteration will converge quadratically ($\alpha = 2$)

(a) $x_{n+1} = -16 + 6x_n + \frac{12}{x_n}, x^2$

$$g(x) = x_{n+1}$$

$$g'(x) = 6 - \frac{12}{{x_n}^2}$$

$$g'(x^*) = g'(2) = 6 - \frac{12}{2^2} = 3$$

$$g(x^*) > 1$$

Since $g(x^*) > 1$, the iteration will **diverge**.

(b) $x_{n+1} = \frac{2}{3}x_n + \frac{1}{x_n^2}, x^* = 3^{\frac{1}{3}}$

$$g(x) = x_{n+1}$$

$$g'(x) = \frac{2}{3} - \frac{2}{x^3}$$

$$g'(x^*) = g'(3^{\frac{1}{3}}) = \frac{2}{3} - \frac{2}{((3^{\frac{1}{3}})^3} = 0$$

Since $g(x^*) = 0$, the iteration will **converge**. But how fast? We will look at $g''(x^*)$

$$g''(x) = \frac{6}{x^4}$$

$$g''(x^*) = g''(3^{\frac{1}{3}})^3) = \frac{6}{(3^{\frac{1}{3}})^{-4}} \neq 0$$

Therefore, the iteration will **converge quadratically**

(c) $x_{n+1} = \frac{12}{1+x_n}, x^* = 3$

$$g(x) = x_{n+1}$$

$$g'(x) = \frac{-12}{(1+x)^2}$$

$$g'(x^*) = g'(3) = \frac{-12}{(1+3)^2} = \frac{-3}{4}$$

Since $\left|g(x^*)\right| < 1$, the iteration will **converge linearly** at a rate of $\frac{3}{4}$.

Problem 5

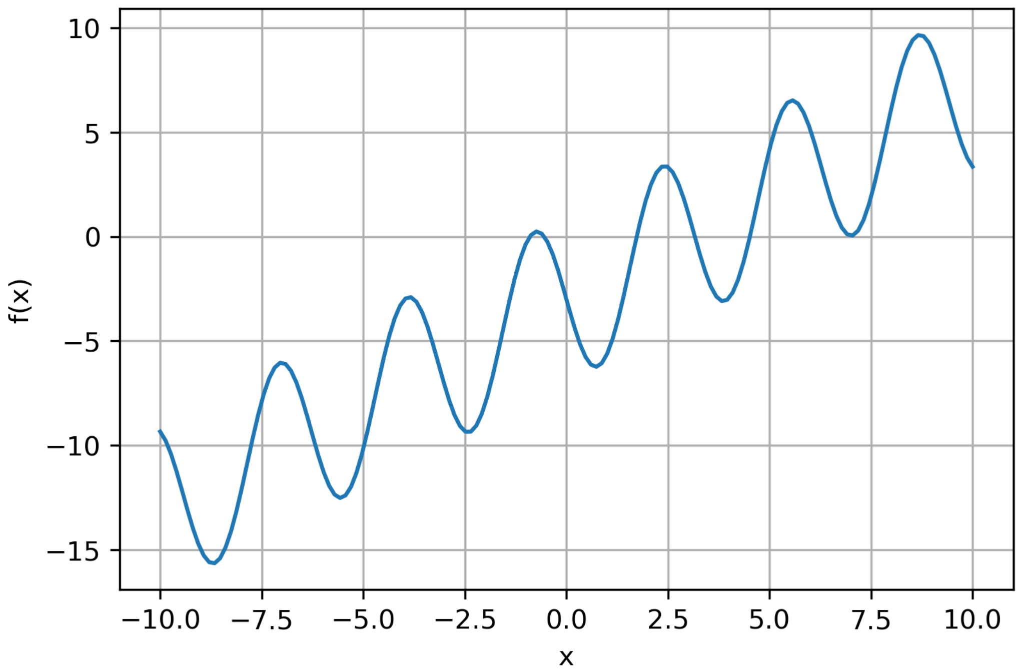
Determine the roots of the scalar equation

$$x - 4sin(2x) - 3 = 0$$

with at least 10 accurate digits (relative error smaller than 0.5×10^{-n})

Solution

(a) Plot $f(x) = x - 4sin(2x) - 3$ using Python, including all zero crossings. How many are there?



There are 5 zero crossings (Near $x = 7$ is not actually a root. Despite coming close to a value of $f(x) = 0$, it does not reach it).

(b) Write a program/use code from class to compute the roots using the fixed point iteration

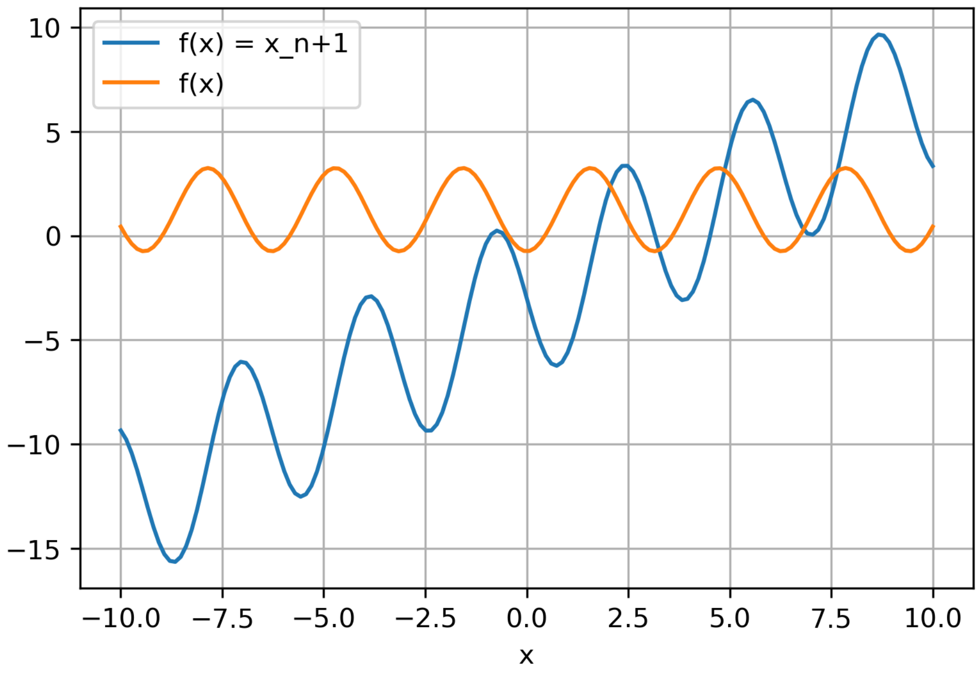
$$x_{n+1} = -sin(2x_n) + 5x_n/4 - 3/4$$

Use a stopping criterium that gives an answer with ten correct digits. Find, empirically which of the roots can be found with the above iteration. Give a theoretical explanation.

Using fixed point iteration with the above iteration, the following roots were found:

$$x^* = 3.161826487, -0.5444424007 \tag{1}$$

The other 3 roots ($x^* \approx -0.898, 1.732, 4.518$) are unable to be found and diverge because $f'(x^*) > 1$ for those values, as can be seen in the below graph:


f(x) = x_n+1
f(x)
10
5
0
−5
−10
−15
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
x