Homework 2 of Computational Mathematics

Chang, Yung-Hsuan
111652004
Department of Applied Mathematics

March 22, 2024

Problem 1. Use a fixed-point iteration method to determine a solution accurate to within 10^{-2} for $x^3 - x - 1 = 0$ on $[1, 2]$. Use $p_0 = 1$.

Solution. We want to find the solution to $f(x) = x$ with $f(x) = \sqrt{1 + \frac{1}{x}}$. Then, by calculator,

$$p_1 = \sqrt{2} = 1.414,$$

$$p_2 = 1.307,$$

$$p_3 = 1.329,$$

$$p_4 = 1.324,$$

which is accurate to within 10^{-2} for $x^3 - x - 1 = 0$ on $[1, 2]$. □

Problem 2. Use Theorem 2.1 to find a bound for the number of iterations needed to achieve an approximation with accuracy 10^{-3} to the solution of $x^3 + x - 4 = 0$ lying in the interval $[1, 4]$. Find an approximation to the root with this degree of accuracy.

Solution. Let $f(x) = x^3 + x - 4$. Since $f \in C[1,4]$ and $f(1) \cdot f(4) = (-2) \cdot 64 < 0$. The Bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approaches to a zero p of f with

$$|p_n - p| \leq \frac{4-1}{2^n}, \quad \text{when } n \geq 1.$$

Then,

$$\frac{3}{2^n} \leq 10^{-3} \implies n \geq \log_2(3000) > 11.$$

111652004_CM_HW2.tex U   111652004_CM_HW2_P2.py U ×

HW2 > 111652004_CM_HW2_P2.py > ...

```python
end_point_1 = 1
end_point_2 = 4
tolerance = 0.001
maximum_number_of_iterations = 50
p_0 = 2.5
i = 1

def f(x):
    return x**3 + x - 4

FA = f(end_point_1)

while i <= maximum_number_of_iterations:
    p = end_point_1 + (end_point_2 - end_point_1) / 2
    FP = f(p)
    if (p == 0 or (end_point_2 - end_point_1) / 2 < tolerance):
        print(f"p = {p} with {i} iterations.")
        exit(0)
    i = i + 1
    if FA * FP > 0:
        end_point_1 = p
        FA = FP
    else:
        end_point_2 = p

print(f"Method failed after {maximum_number_of_iterations}.")
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS   COMMENTS

```
PS E:\Eiken\Visual Studio Code Git Sync\CM_HW> & C:/Users/yungh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "
p = 1.378662109375 with 12 iterations.
```

Using the Bisection method, by Python, $p = 1.37866$. $\square$

Problem 3. The following four methods are preposed to compute $21^{1/3}$. Rank them in order, based on their apparent speed of convergence, assuming $p_0 = 1$.

a. $p_n = \dfrac{20p_{n-1} + \frac{21}{{p_{n-1}}^2}}{21}$

b. $p_n = p_{n-1} - \dfrac{{p_{n-1}}^3 - 21}{3{p_{n-1}}^2}$

c. $p_n = p_{n-1} - \dfrac{{p_{n-1}}^4 - 21p_{n-1}}{{p_{n-1}}^2 - 21}$

d. $p_n = \sqrt{\dfrac{21}{p_{n-1}}}$

Solution.

Problem 4. Use Theorem 2.3 to show that $g(x) = 2^{-x}$ has a unique fixed point on $\left[\frac{1}{3}, 1\right]$. Use fixed-point iteration to find an approximation to the fixed point accurate to within 10^{-4}. Use corollary 2.5 to estimate the number of iterations required to achieve 10^{-4} accuracy, and compare this theoratical estimate the the number actually needed.

Solution. We know that $g \in C\left[\frac{1}{3}, 1\right]$ and $g(x) \in [0.5, 0.9637] \subseteq \left[\frac{1}{3}, 1\right]$. Then g has at least a fixed point in $[a, b]$. Moreover, $g'(x)$ exists on $\left(\frac{1}{3}, 1\right)$. Choose $k = 0.7$. Then

$$\begin{aligned}\left|\frac{\mathrm{d}}{\mathrm{d}x}2^{-x}\right| &= \ln 2 \cdot 2^{-x} \\ &< \ln 2 \cdot 2^{-0} \\ &= \ln 2 \\ &< k\end{aligned}$$

for all $x \in (0, \infty)$. Hence, $g'(x)$ exists on $\left(\frac{1}{3}, 1\right)$ and a positive $0 < k < 1$ exsits with $|g'(x)| \leq k$ for all $x \in \left(\frac{1}{3}, 1\right)$. Then there exists exactly one fixed point in $\left[\frac{1}{3}, 1\right]$. It is known the assumption of Theorem 2.4 holds, i.e., $g'(x)$ exists on $\left(\frac{1}{3}, 1\right)$ and a positive $0 < k < 1$ exsits with $|g'(x)| \leq k$ for all $x \in \left(\frac{1}{3}, 1\right)$. By Corollary 2.5,

$$|p_n - p| \leq 0.7^n \max\{0.6 - \frac{1}{3}, 1 - 0.6\}.$$

Then,

$$0.7^n \max\{0.6 - \frac{1}{3}, 1 - 0.6\} \leq 10^{-4} \implies n > 23.$$

111652004_CM_HW2.tex U   111652004_CM_HW2_P4.py U

HW2 > 111652004_CM_HW2_P4.py > ...

```python
p_0 = 0.6
tolerance = 0.0001
maximum_number_of_iterations = 50
i = 1

def g(x):
    return 0.5**x

while i <= maximum_number_of_iterations:
    p = g(p_0)
    if abs(p - p_0) < tolerance:
        print(f"p = {p} with {i} iterations.")
        exit(0)
    i = i + 1
    p_0 = p

print(f"Method failed after {maximum_number_of_iterations}.")
```

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS   COMMENTS

```
PS E:\Eiken\Visual Studio Code Git Sync\CM_HW> & C:/Users/yungh/AppData/Local/Microsoft/WindowsApps/python3.11.exe "e
p = 0.6412138835623649 with 9 iterations.
```

By Python, the number of iteration actually needed is 9, which is smaller due to my overestimation for error. □

Problem 5. Let A be a given positive constant and $g(x) = 2x - Ax^2$.

a. Show that if fixed-point iteration converges to a nonzero limit, then the limit is $p = \dfrac{1}{A}$, so the inverse of a number can be found using only multiplications and subtractions.

b. Find an interval about $\dfrac{1}{A}$ for which fixed-point iteration converges, provided p_0 is in that interval.

Solution.

Problem 6. Show that if A is any positive number, then the sequence defined by

$$x_n = \frac{1}{2}x_{n-1} + \frac{A}{2x_{n-1}}, \quad \text{for } n \geq 1,$$

converges to $\sqrt{A}$ whenever $x_0 > 0$.

Solution.

Problem 7. Let $f(x) = -x^3 - \cos x$. With $p_0 = -1$ and $p_1 = 0$, find p_3.

a. Use the Secant method.

b. Use the method of False Position.

Solution.

Problem 8. Problems involving the amount of money required to pay off a mortgage over a fixed period of time involve the formula

$$A = \frac{P}{i}\left(1 - (1+i)^{-n}\right),$$

known as an *ordinary annuity equation.* In this equation, A is the amount of the mortgage, P is the amount of each payment, and i is the interest rate per period for the n payment periods. Suppose that a 30-year home mortgage in the amount of \$135,000 is needed and that the borrower can afford house payments of at most \$1000 per month. What is the maximal interest rate the borrower can afford to pay?

Solution.

Problem 9.

a. Show that for any positive integer k, the sequence defined by $p_n = \dfrac{1}{n^k}$ converges linearly to $p = 0$.

b. Show that the sequence $p_n = 10^{-2^n}$ converges quafratically to 0.

Solution.

Problem 10.

a. The following sequences are linearly convergent. Generate the first five terms of the sequence $\{\hat{p_n}\}$ using Aitken's Δ^2 method.

$$p_0 = 0.5, \quad p_n = \cos(p_{n-1}), \quad n \geq 1$$

b. Use Steffensen's method to find, to an accuracy of 10^{-4}, the root of $x^3 - x - 1 = 0$ that lies in $[1, 2]$.

Solution.

Problem 11. Given a polynomial $P(x) = x^3 - 5x^2 + 8x - 6$, do the following:

a. Evaluate $P(2)$, $P'(2)$, $P(4)$, and $P'(4)$ by Horner's method.

b. Find the root of $P(x)$ with error less than 0.00001 between $[2, 4]$ by using the Newton method with initial point $x_0 = 2$ and $x_0 = 4$. Determin which initial point may lead to the root.

c. Deflate $P(x)$ into a quadartic ppolynomial by using the results in (b) and find the complex roots of $P(x)$.

d. Perform one step of Muller's Method starting from initial $(0, P(0))$, $(1, P(1))$ and $(2, P(2))$.

e. Implement a MATLAB code of Muller's Method to find the complex root within error less than 0.00001 and compare with the answer you find in (c).

Solution.