

# Homework 2 of Computational Mathematics

AM15 黃琦翔 111652028

March 27, 2024

1.  $x^3 = x + 1 \implies x^2 = 1 + \frac{1}{x} \implies x = \sqrt{1 + \frac{1}{x}} = g(x)$ .  $p_1 = g(p_0) = \sqrt{1+1} = \sqrt{2} \approx 1.414$ .  $p_2 = g(p_1) = \sqrt{1 + \frac{1}{\sqrt{2}}} \approx 1.3065$ .  $p_3 = g(p_2) \approx 1.3172$ .  $p_4 = g(p_3) \approx 1.326$ .  $p_5 = g(p_4) \approx 1.324$ . Then,  $p_4$  is the answer that we want to find.
2. Let  $f(x) = x^3 + x - 4$ ,  $f'(x) = 3x^2 + 1 < 49$  for all  $x \in [1, 4]$ . Thus, for  $|x - y| < \frac{10^{-3}}{49} \approx 2.0409e - 5$ ,  $|f(x) - f(y)| < 10^3$ . Find  $n$  s.t.  $3 \cdot 2^{-n} < 2.0409e - 5$ ,  $n > -\log_2\left(\frac{2.0409e - 5}{3}\right) \approx 17.1653$ . Thus, the bound of the number of iteration is 18. Then, by python code below, the root is about 1.3787.

```
HW2 > HW2.py > ...
1  a = [1]
2  b = [4]
3
4  def f(x):
5      return x**3 + x - 4
6
7  while 1:
8      mid = (a[-1] + b[-1])/2
9      val = f(mid)
10     print(mid, val)
11
12     if abs(val) < 0.0001:
13         break
14     elif val > 0:
15         b.append(mid)
16     else:
17         a.append(mid)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數> &
2.5 14.125
1.75 3.109375
1.375 -0.025390625
1.5625 1.377197265625
1.46875 0.637176513671875
1.421875 0.2965202331542969
1.3984375 0.13326025009155273
1.38671875 0.05336350202560425
1.380859375 0.013844214379787445
1.3779296875 -0.005808685906231403
1.37939453125 0.004008884658105671
1.378662109375 -0.0009021193400258198
1.3790283203125 0.0015528278327110456
1.37884521484375 0.0003252155581776057
1.378753662109375 -0.00028848656066315925
1.3787994384765625 1.8355831034710945e-05
```

3. In this question, we let  $p_{n+1} = g(p_n)$  and  $p = \sqrt[3]{21}$ . And by  $\lim_{n \rightarrow \infty} \frac{|p_{n+1} - p|}{|p_n - p|^\alpha} = \lambda$ ,

$$\frac{|p_{n+1} - p|}{|p_n - p|} \approx \frac{\lambda |p_n - p|^\alpha}{\lambda |p_{n-1} - p|^\alpha} \approx \left| \frac{p_n - p}{p_{n-1} - p} \right|^\alpha, \text{ then } \alpha \approx \frac{\ln |(p_{n+1} - p)/(p_n - p)|}{\ln |(p_n - p)/(p_{n-1} - p)|}.$$

Then, by observing the function, we get b is quadratic convergence (Newton's method). In the meanwhile, though  $\sqrt[3]{21}$  is a fixed point of c, but it will diverges or converges to 0 for any open interval contains  $\sqrt[3]{21}$ .

Thus, by result of iteration with python below, the order of speed of convergence is  $b > d > a$ .

```

20
21 def problem_3():
22     import math
23     p = 21**(1.0/3)
24     def a(x):
25         return (20*(x**3)+21)/(21*(x**2))
26
27     def b(x):
28         return x - (x**3 - 21)/(3*(x**2))
29
30     def c(x):
31         return x - (x**4 - 21*x)/(x**2 - 21)
32
33     def d(x):
34         return math.sqrt(21/x)
35
36     def Alpha(f, x):
37         return math.log(abs((f(x) - p)/(x - p)))
38
39     functions = {"a": a, "b": b, "c": c, "d": d}
40
41     for i in functions.keys():
42         f = functions[i]
43         p_now = 1
44         for k in range(20):
45             p_now = f(p_now)
46             # print(k+1, "err:", abs(p_now - p))
47         try:
48             alpha = Alpha(f, f(p_now))/Alpha(f, p_now)
49             print(i, alpha)
50         except:
51             print(i, "Cannot compute alpha by this method.")
52
53     print("---")
54
55 def problem_4():
56     def a(x):

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數> & C:/Users/9ryan/AppD
a 0.9982564193669963
---
b Cannot compute alpha by this method.
---
c Cannot compute alpha by this method.
---
d 1.0000015723442375
---
PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數>

```

4.  $|g'(x)| = |-2^{-x} \ln(2)|$  is continuous and decreasing on  $\mathbb{R}$ . Thus,  $g'(\frac{1}{3}) \approx 0.55015$ . And since  $g(\frac{1}{3}) \approx 0.79370 > \frac{1}{3}$  and  $g(1) = \frac{1}{2} < 1$ . By Theorem 2.3,  $g(x)$  has unique fixed point on  $[\frac{1}{3}, 1]$ .

```

55 def problem_4():
56     def g(x):
57         return 2**(-x)
58
59     x = 0.3334
60     step = 0
61     while abs(x-g(x)) >= 10**(-4):
62         x = g(x)
63         step += 1
64         print("step:", step, ",value:", x)
65
66
67 if __name__ == "__main__":
68     problem_4()

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

```

step: 1 ,value: 0.79366385007938
step: 2 ,value: 0.5768772000497738
step: 3 ,value: 0.6704133579003061
step: 4 ,value: 0.6283266346632733
step: 5 ,value: 0.6469263427213542
step: 6 ,value: 0.6386394847182052
step: 7 ,value: 0.6423183934819865
step: 8 ,value: 0.6406825519733249
step: 9 ,value: 0.6414094204320899
step: 10 ,value: 0.6410863425561439
step: 11 ,value: 0.6412299238405336

```

PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數>

By the result from python above, we need 12 steps to let error less than  $10^{-4}$ . Since  $|g'(x)| \leq \ln(2)2^{-x} \leq 0.551$ , we take  $k = 0.551$  and  $p_0 = 1$ .  $|p_n - p| \leq \frac{k^n}{1-k}|p_1 - p| \approx k^n \cdot 1.113585746 \leq 10^{-4}$ . Then,  $n \geq 15.633566$ , that is, the bound is 16 times of iteration.

5. (a) Let  $x_*$  be nonzero fixed point of  $g$ ,  $x_* = 2x_* - Ax_*^2 \implies x_* = Ax_*^2 \implies x_*^*A = 1$ . Thus,  $x_* = \frac{1}{A}$ .  
 (b) Since  $g'(x) = 2 - 2Ax$  and we need  $|g'(x)| < k < 1$ ,

$$\begin{aligned}
 |2 - 2Ax| < k &\iff -k < 2 - 2Ax < k \\
 &\iff \frac{k-2}{2A} < x < \frac{k+2}{2A} \\
 &\stackrel{k=1-\varepsilon}{\iff} \frac{1}{2A} + \frac{\varepsilon}{2A} < x < \frac{3}{2A} - \frac{\varepsilon}{2A}
 \end{aligned}$$

Thus,  $x \in \left(\frac{1}{2A} + \frac{3}{2A}\right)$ .

6. Let  $g(x) = \frac{x}{2} + \frac{A}{2x}$ , and  $g'(x) = \frac{1}{2} - \frac{A}{2x^2}$ . For  $x \in (\sqrt{A}, \infty)$ ,  $|g'(x)| \leq \frac{1}{2}$ . And since  $x > \sqrt{A}$ ,  $\frac{x}{2} + \frac{A}{2x} - \sqrt{A} = \frac{x^2 + A - 2Ax}{2x} = \frac{(x - \sqrt{A})^2}{2x} > 0$ . Thus, if  $x > \sqrt{A}$ ,  $g(x) > \sqrt{A}$ . By Corollary 2.5,  $g(x)$  converges to  $\sqrt{A}$  for all  $(\sqrt{A}, \infty)$ .

Then, for  $x \in (0, \sqrt{A})$ ,  $\frac{x}{2} + \frac{A}{2x} - \sqrt{A} = \frac{(A-x)^2}{2x} > 0$ . Thus, for  $0 < x < \sqrt{A}$ ,  $g(x) > \sqrt{A}$ , and then by the argument above, it will converge to  $\sqrt{A}$ , too.

Therefore, for  $x_0 > 0$ ,  $x_n$  will converge to  $\sqrt{A}$ .

7. (a)  $p_2 = p_1 - \frac{f(p_1)[p_0 - p_1]}{f(p_0) - f(p_1)} = -\frac{-1}{-1 + \cos(-1) - 1} = \frac{1}{-2 + \cos(1)} \approx -0.6850733573260451$ .
- $$p_3 = p_2 - \frac{f(p_2)[p_1 - p_2]}{f(p_1) - f(p_2)} \approx -1.252076488909229.$$
- (b)  $p_2 = p_1 - f(p_1) \cdot \frac{p_0 - p_1}{f(p_0) - f(p_1)} \approx -0.6850733573260451$ . Since  $f(p_2) < 0$ ,  $f(p_2) \cdot f(p_0) < 0$ .
- Then,  $p_3 = p_2 - f(p_2) \cdot \frac{p_0 - p_2}{f(p_0) - f(p_2)} \approx -0.8413551256656522$ .
8. From the question, we can rewrite it as  $f(i) = 1000(1 - (1 + i)^{-30 \cdot 12}) - 135,000i = 0$ . After testing, we know that  $f(0.002) > 0$  and  $f(0.01) < 0$ .

```

92
93 def problem_8():
94     def secant_method(f, x_1, x_2):
95         return x_1 - (f(x_1)*(x_2 - x_1))/(f(x_2) - f(x_1))
96
97     f = lambda x: 1000*(1-(1+x)**(-360))-135000*x
98     x = [0.002, 0.01]
99     for _ in range(8):
100         p = secant_method(f, x[-1], x[-2])
101         print(p, f(p))
102         x.append(p)
103
104 if __name__ == "__main__":
105     problem_8()

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數> & C:/Users/9ryan/AppData/Local/Programs/Python/Python39-64/Python.exe C:/Users/9ryan/AppData/Local/Programs/Python/Python39-64/Python.exe
0.005130556113250995 148.91891659914575
0.006507247379340629 24.713056780832744
0.006781165610560291 -3.2322577565320216
0.006749483221502495 0.04480741496536211
0.0067499164158026734 7.675006418139674e-05
0.006749917159088972 -1.8349055608268827e-09
0.006749917159071203 1.1368683772161603e-13
0.0067499171590712035 0.0
PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數>

```

Then, by secant method and python,  $i \approx 0.0067499171590712035$ . Thus, the interest rate the borrower can afford is about  $0.0067499171590712035 \cdot 12 = 0.080999005908854442 = 8.0999005908854442\%$ .

9. (a)  $\lim_{n \rightarrow \infty} \frac{|1/(n+1)^k|}{|1/n^k|} = \lim_{n \rightarrow \infty} \left( \frac{n}{n+1} \right)^k = 1$  for all  $k$ . Thus,  $p_n$  converges linearly.
- (b)  $\lim_{n \rightarrow \infty} \frac{10^{-2^{n+1}}}{|10^{-2^n}|^2} = \lim_{n \rightarrow \infty} 10^{-2^{n+1} + 2^n \cdot 2} = 1$ . Thus,  $p_n$  converges linearly.
10. (a)

i	$p_i$	j	$\hat{p}_j$
0	0.5		
1	0.8775825618		
2	0.6390124941	0	0.7313851863
3	0.8026851006	1	0.7360866917
4	0.6947780267	2	0.7376528713
5	0.7681958312	3	0.7384692208
6	0.7191654459	4	0.7387980651

(b) From 1., we let  $g(x) = \sqrt{1 + \frac{1}{x}}$  and do iteration.

i	$p_0$	$p_1$	$p_2$	$\hat{p}_i$
0	2	1.22474487139	1.34777467735	1.33092441907
1	1.3309244190	1.32338863029	1.32500412497	1.32471893841

The answer above is given by python.

```

103
104 def problem_10():
105     def Aitkens_method(f, x_0, step):
106         x = [x_0, f(x_0), f(f(x_0))]
107         for i in range(step):
108             p = x[-3] - ((x[-2]-x[-3])**2)/(x[-1] - 2*x[-2] + x[-3])
109             print(f"p_{i}: {p}")
110             x.append(f(x[-1]))
111
112     def Steffensens_method(f, x_0):
113         x = [x_0]
114         step = 0
115         while abs(x[-1]-f(x[-1])) >= 10**(-4):
116             p_0 = x[-1]
117             p_1 = f(p_0)
118             p_2 = f(p_1)
119             p = p_0 - ((p_1 - p_0)**2)/(p_2 - 2*p_1 + p_0)
120             print(f"i: {step}, p_0: {p_0}, p_1: {p_1}, p_2: {p_2}, p: {p}")
121             x.append(p)
122             step += 1
123
124     from math import cos, sqrt
125     f = lambda x: cos(x)
126     x_0 = 0.5
127     print("Aitken's method: ")
128     Aitkens_method(f, x_0, 5)
129     print("Steffensens's method: ")
130     g = lambda x: sqrt(1 + 1/x)
131     y_0 = 2
132     Steffensens_method(g, y_0)
133
134 if __name__ == "__main__":
135     problem_10()

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數> & C:/Users/9ryan/AppData/Local/Programs/Python/Py  
Aitken's method:  
p\_0: 0.7313851863825818  
p\_1: 0.7360866917130169  
p\_2: 0.7376528713963997  
p\_3: 0.7384692208762632  
p\_4: 0.7387980651735903  
Steffensens's method:  
i: 0, p\_0: 2, p\_1: 1.224744871391589, p\_2: 1.3477746773580983, p: 1.330924419078614  
i: 1, p\_0: 1.330924419078614, p\_1: 1.3233886302912288, p\_2: 1.3250041249776752, p: 1.3247189384145663  
○ PS C:\Users\9ryan\OneDrive - 國立陽明交通大學\HW\計數>

$P(x) = x^3 - 5x^2 + 8x - 6, x_0 = 2.$		
Step	y	z
0	$a_0 = 1$	$a_0 = 1$
1	$1 \cdot 2 - 5 = -3$	$1 \cdot 2 + -3 = -1$
2	$-3 \cdot 2 + 8 = 2$	$-1 \cdot 2 - 2 = 0$
3	$2 \cdot 2 - 6 = -2$	

Then,  $P(2) = -2$  and  $P'(2) = 0$ .

$P(x) = x^3 - 5x^2 + 8x - 6, x_0 = 4.$		
Step	y	z
0	$a_0 = 1$	$a_0 = 1$
1	$1 \cdot 4 - 5 = -1$	$1 \cdot 4 - 1 = 3$
2	$-1 \cdot 4 + 8 = 4$	$3 \cdot 4 + 4 = 16$
3	$4 \cdot 4 - 6 = 10$	

Then,  $P(4) = 10$  and  $P'(4) = 16$ .

- (b)  $P(x) = x^3 - 5x^2 + 8x - 6$  and  $P'(x) = 3x^2 - 10x + 8$ . And since  $P'(2) = 0$ , Newton's method may not be usefully in  $x_0 = 2$ . Then, for  $x_0 = 4$ ,

Newton's method for $P(x)$ with $x_0 = 4$ .				
i	$x_i$	$f(x_i)$	$f'(x_i)$	$x_{i+1}$
0	4	10	16	3.375
1	3.375	2.490234375	8.421875	3.07931354359925
2	3.079313543599258	0.42222920359638394	5.65338026338887	3.00462739408783
3	3.004627394087836	0.02322272062870922	5.03708339103081	3.00001704344919
4	3.000017043449198	8.52184079072060e-05		

- (c)  $P(x) = (x - 3)(x^2 - 2x + 2) = 0$ . Then,  $x = \frac{2 \pm 2i}{2}$ .

(d)