Part MATH-UA 252 — Numerical Analysis

Based on lectures by Jason Kaye

Homework done by Aaron Ma

Fall 2024

These notes are not endorsed by the lecturers, and I have modified them (often significantly) after lectures. They are nowhere near accurate representations of what was actually lectured, and in particular, all errors are almost surely mine.

1 Problem 1

1.1 a

Plot the function. Then use the MATLAB function fzero to find the root; consider this to be the exact solution x^* of $f(x) = 0$

Solution:

Set the function as:

$$f(x) = e^x - x^2 - 0.5$$

We can plot the function and find the solution with MATLAB

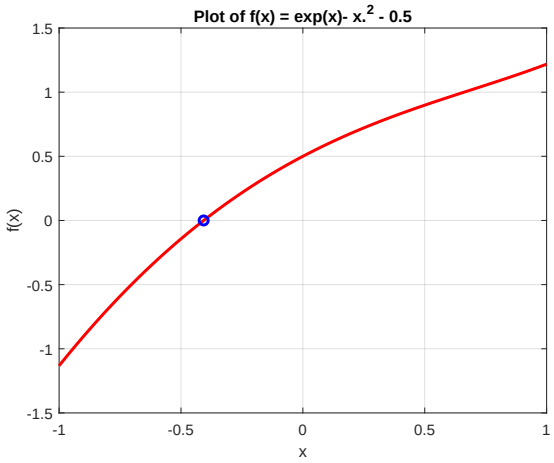

Figure 1: $f(x) = e^x - x^2 - 0.5$

we can also find that $x^* = -0.406997$

1.2 b

Solution:

We can plot the graph of error with the number of trials for 1-31

In the picture, the blue line represents the log of the actual error of each trial and the red line represents the log of the error bound. While there is fluctuation in the error, it is clear to see that the error never exceeds the error bound.

Figure 2: Error of each trial and the error bound

1.3 c

As e_k is the error between the xiter value and the actual x value, and bisection always eliminates the potential searching section in half, it is clear that $e_k < E_k = \frac{b-a}{2^k}$. This means that the bisection method converges linearly and $\frac{E_k}{E_{k+1}}$ is a constant. This means that $\log(e_k) < \log(\frac{b-a}{2^k}) = \log(b-a) - k\log(2)$, which indicates a linear relationship on the grpah

2 Problem 2

2.1 b

The code is submitted with other codes and here is a copy the code

```
% set up the parameters
a = -1; % the beginning of the interval
b = 1; % end of the interval
x0 = 0; % initial guess for newton's method and secant method
x1 = 1; % the second parameter for secant method
f = @(x)cos(exp(x)); % function
fp = @(x) -sin(exp(x)) .* exp(x); % derivative of the function
tol = 10^(-10); % tolerance
x_star = fzero(f,0); % x_star, the actual solution

% Newton's Method
[x,niter,xiter] = newton_tol(f,fp,x0,50,tol); % get the solution and
% all the iterative trials info by Newton's method

% show the solution
fprintf("x*: %.12f\n", x_star); % show x_star

% Newton's method
fprintf("Newton's method\n");
fprintf("x: %.12f\n", x); % show final solution by Newton's method with 12
% digits
fprintf("niter: %d\n", niter); % the number of iteration
fprintf("xiter: %.12f\n", abs(xiter));
fprintf("error: %.12f\n", abs(xiter - fzero(f,0))); % the error in 12 digit

% Bisection method
[x,niter,xiter] = bisection_tol(f,a, b, 50, tol); % get the solution and
% all the iterative trials info with bisection method

% show the solution of Bisection method
fprintf("Bisection method\n");
fprintf("x: %.12f\n", x); % show final solution by Bisection method with 12
% digits
fprintf("niter: %d\n", niter); % the number of iteration
fprintf("error: %.12f\n", abs(xiter - fzero(f,0))); % the error in 12 digit
for i = 2:length(xiter)
    fprintf("difference: %.12f\n", abs(xiter(i) - xiter(i-1)));
end

% Secant Method
```

```
fprintf('Secant Method\n');
[x, niter, xiter] = secant_tol(f,x0,x1,50,10^-10);
fprintf('x: %.12f\n', x);% the final solution by Bisection method

fprintf('niter: %d\n', niter); % the number of iteration
fprintf('error: %.12f\n', abs(xiter - fzero(f,0))); % the error in 12 digit
```

The final error in 12 digit is:

(i) Newton's Method: error: 0.000000000000

(ii) Secant's Method: error: 0.000000000000

(iii) Bisection Method: error: 0.000000000022

2.2 c

For bisection method, as we always limit the section into half, and that the error must be smaller than the length of the entire section, we can calculate the maximum number of iterations we need for a certain error. As a result, the algorithm can be devised into robust.

$$\text{As the bisection method limit the error by:} \varepsilon < \frac{b-a}{2^k}$$

$$\text{If we want an error of } \varepsilon\text{, we can turn itermax into } k = \log_2 \frac{b-a}{\varepsilon} + 1$$

and we can assure that we will converge the solution into an error smaller than tolerance
However, for Newton's method and Secant method, it is a different story. As the iteration may oscillate between two points and even diverge, two of the great example on the book is

$$f(x) = x^{\frac{1}{3}}$$
$$f(x) = x^3 - x - 3$$

where the result never converge. We can first use bisection method to find the section that is small enough, and then use the two strategy for that.

2.3 d

It can be shown from the graph that the number of iteration needed for bisection converges almost linearly, while the other two are so efficient that the number of trials they use barely changes.

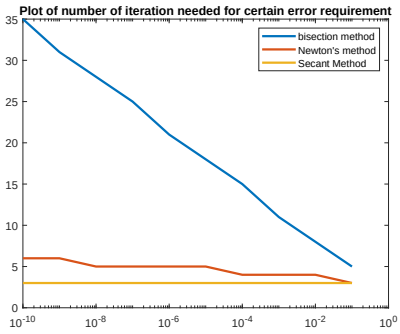

Figure 3: Number of iteration needed for different error requirement

3 Problem 3

3.1 a

As the graph shows, the convergence order for bisection method is 1 as the plot shows an almost linear relationship between the log of error and the number of trials, while the rest two are bigger than 1 as they converge much more faster

3.2 b

From the definition of p, we know that

$$\lim_{k\to\infty} \frac{\varepsilon_k}{(\varepsilon_{k-1})^p} = C$$

for k sufficiently large

$$\begin{aligned} C(\varepsilon_{k-1})^p &\approx \varepsilon_k \\ \log C + p\log\varepsilon_{k-1} &\approx \log\varepsilon_k \\ \log\varepsilon_k &\approx c + p\log\varepsilon_{k-1} \end{aligned}$$

3.3 c

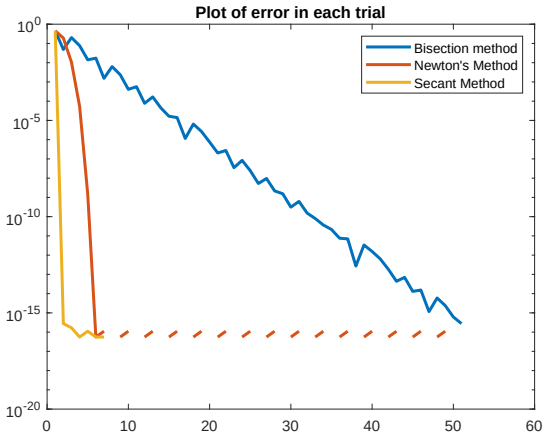

Figure 4: Error of different methods in each trial

4 Problem 4

4.1 b

According to Matlab, the amount of iteration needed for this iteration is 6 and the amount of iteration needed for bisection method only is 39.

5 Problem 5

5.1 a

$$
\begin{aligned}
x_{k+1} &= x_k - \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} f(x_k) \\
x_{k+1} &= \frac{x_k(f(x_k) - f(x_{k-1})) - f(x_k)(x_k - x_{k-1}))}{f(x_k) - f(x_{k-1})} \\
x_{k+1} &= \frac{-x_k f(x_{k-1} + x_{k-1} f(x_k)}{f(x_k) - f(x_{k-1})} \\
x_{k+1} &= \frac{x_k f(x_{k-1}) - x_{k-1} f(x_k)}{f(x_{k-1}) - f(x_k)}
\end{aligned}
$$

5.2 b

$$
\begin{aligned}
\varphi(x_k, x_{k-1}) &= \frac{x_{k+1} - x^*}{(x_k - x^*)(x_{k-1} - x^*)} \\
&= \frac{\frac{x_k f(x_{k-1}) - x_{k-1} f(x_k)}{f(x_{k-1}) - f(x_k)} - x^*}{(x_k - x^*)(x_{k-1} - x^*)} \\
&= \frac{f(x_{k-1})(x_k - x^*) - f(x_k)(x_{k-1} - x^*)}{(x_k - x^*)(x_{k-1} - x^*)} \cdot \frac{1}{f(x_{k-1}) - f(x_k)} \\
&= \left(\frac{f(x_{k-1})}{x_{k-1} - x^*} - \frac{f(x_k)}{x_k - x^*}\right) \cdot \frac{1}{f(x_{k-1}) - f(x_k)} \\
&= \left(\frac{f(x_{k-1}) - f(x^*)}{x_{k-1} - x^*} - \frac{f(x_k) - f(x^*)}{x_k - x^*}\right) \cdot \frac{1}{f(x_{k-1}) - f(x_k)}
\end{aligned}
$$

As $f(x^*) = 0$
In addition, as $x_k \to x^*$,

$$
\begin{aligned}
\frac{f(x_k) - f(x^*)}{x_k - x^*} &= f'(x^*) \\
f(x_k) &= f(x^*) \\
\therefore \varphi(x_k, x_{k-1}) &= \left(f'(x^*) - \frac{f(x_{k-1}) - f(x^*)}{x_{k-1} - x^*}\right) \cdot \frac{1}{f(x_{k-1}) - f(x_k)}
\end{aligned}
$$

5.3 c

Use L'Hopital law from (b)

$$
\begin{aligned}
\because f(x^*) &= 0 \\
\varepsilon(x_k, x_{k-1}) &= \frac{(x^* - x_{k-1})f'(x^*) + f(x_{k-1}) - f(x^*)}{(f(x^*) - f(x_{k-1}))(x^* - x_{k-1})} \\
&= \frac{-f'(x^*) + f'(x_{k-1})}{-f'(x_{k-1}(x^* - x_{k-1}) + f(x_{k-1})} \\
&= \frac{f''(x_{k-1})}{f'(x_{k-1}) - f''(x_{k-1})(x^* - x_{k-1}) + f'(x_{k-1})} \\
&= \frac{f''(x_{k-1}}{2f'(x_{k-1})} \\
&= \frac{f''(x_{k-1}}{2f'(x_{k-1})}
\end{aligned}
$$

if $x_{k-1}, x_k \to x^*$

5.4 d