Numerical Analysis Assignment #1

Xinglu Wang Student Number: 3140102282
College of Information Science & Electronic Engineering, Zhejiang University

It is my first time to write an article with LaTeX and in English. I am quite exciting and find myself learned a lot in this wonderful course!
After programming and debugging, I am impressed that Fix-Point method is hard to find a good $g(x)$, and when I choose $g(x) = x - \frac{f(x)}{f'(x)}$, this method will be sensitive to initial value.

Problem 1:

a. This problem satisfies condition of Intermediate Value Theorem, since $f \in C[a, b]$
Meanwhile,

$$\min(f(x_1), f(x_2)) \leq \frac{f(x_1) + f(x_2)}{2} \leq \max(f(x_1), f(x_2))$$

Therefore,

$$\exists \xi \in [x_1, x_2] : \qquad f(\xi) = \frac{f(x_1) + f(x_2)}{2}$$

b. Similarly, because c_1 and c_2 is positive, $f(\xi)$ the weighted average of $f(x_1)$ and $f(x_2)$, which means,

$$\min(f(x_1), f(x_2)) \leq \frac{c_1 f(x_1) + c_2 f(x_2)}{c_1 + c_2} \leq \max(f(x_1), f(x_2))$$

Therefore,

$$\exists \xi \in [x_1, x_2] : \qquad f(\xi) = \frac{c_1 f(x_1) + c_2 f(x_2)}{c_1 + c_2}$$

c. When $f(x) = x$, $\begin{cases} c_1 = -1 \\ c_2 = 2 \end{cases}$ *and* $\begin{cases} x_1 = 1 \\ x_2 = 2 \end{cases}$,

$$\frac{c_1 f(x_1) + c_2 f(x_2)}{c_1 + c_2} = 2 \notin [1, 2]$$

In this condition, therefore

$$!\exists \xi \in [1, 2] : \qquad f(\xi) = \frac{c_1 f(x_1) + c_2 f(x_2)}{c_1 + c_2}$$

Problem 2:

a.

This problem satisfies Mean Value Theorem, since $f \in C[a, b]$ and f is differentiable on (a, b), therefore the absolute error is

$$\left|\tilde{f}(x_0) - f(x_0)\right| = |f'(\xi)|\,|\varepsilon| \approx \varepsilon\,|f'(x_0)| \text{ ,where } x_0 - \varepsilon \leq \xi \leq x_0 + \varepsilon$$

And the relative error is

$$\left|\frac{\tilde{f}(x_0) - f(x_0)}{f(x_o)}\right| = |f'(\xi)| \left|\frac{\varepsilon}{f(x_0)}\right| \approx \varepsilon \frac{|f'(x_0)|}{|f(x_0)|} \text{ , where } x_0 - \varepsilon \leq \xi \leq x_0 + \varepsilon$$

b.

- **i** Submit ε and x_0 into formula above, we know the absolute error is 1.3591×10^{-5} and the relative is 5.0000×10^{-6}
- **ii** Similarly, the absolute error is 2.7015×10^{-6}. and the relative is 3.2105×10^{-6}.

c.

- **i** Bound of absolute error is 1.1013 and the relative is 5.0000×10^{-5}.
- **ii** Bound of absolute error is 4.1954×10^{-5} and the relative is 7.7118×10^{-5}.

We find that ***(ii)*** in **c** is quite special, which shows that for e^x relative error is more than absolute one because of its rapid exponential incasement!

Problem 3:

a: ***(i)***

$$A = \frac{4}{5} + \frac{1}{3} = \frac{17}{15}$$

(ii)

$$A = chop(chop(\frac{4}{5}) + chop(\frac{1}{3})) = chop(0.800 + 0.333) = 1.133$$

(iii)

$$A = round(0.800 + 0.333) = 1.133$$

(iv) Relative error is 0.029% and 0.029%

b: ***(i)***

$$A = \frac{20}{33} - \frac{3}{20} = \frac{301}{660}$$

(ii)

$$A = chop(chop(0.605) - chop(\frac{3}{20})) = chop(0.605 - 0.150) = 0.455$$

(iii)

$$A = round(0.606 - 0.150) = 0.456$$

(iv) Relative error is 0.2331% and 0.0133%

Problem 4:

a. When $\gamma < \beta$ and $\gamma < \alpha$

$$\lim_{x \to 0} \frac{F(x) - c_1 L_1 - c_2 L_2}{x^\gamma} = \lim_{x \to 0} \frac{c_1 O(x^\alpha) + c_2 O(x^\beta)}{x^\gamma} = 0$$

which is equal to

$$F(x) = c_1 L_1 + c_2 L_2 + O(x^\gamma)$$

b. When $\gamma < \beta$ and $\gamma < \alpha$

$$\lim_{x \to 0} \frac{G(x) - L_1 - L_2}{x^\gamma} = \lim_{x \to 0} \frac{O((c_1 x)^\alpha) + O((c_2 x)^\beta)}{x^\gamma} = 0$$

Problem 5:

Implement the Bisection method in C or matlab and find solutions accurate to within for the following problems. (List the midpoints in each iteration as well) .

a. $e^x - x^2 + 3x - 2 = 0 \quad$ for $0 \le x \le 1$
After applying the Bisection method , the value of x is 0.257530212402344, the midpoint is listed and ploted below:

Midpoint List	
0.5	0.256836
0.25	0.257324
0.375	0.257568
0.3125	0.257446
0.28125	0.257507
0.265625	0.257538
0.257813	0.257523
0.253906	0.257530
0.255859	

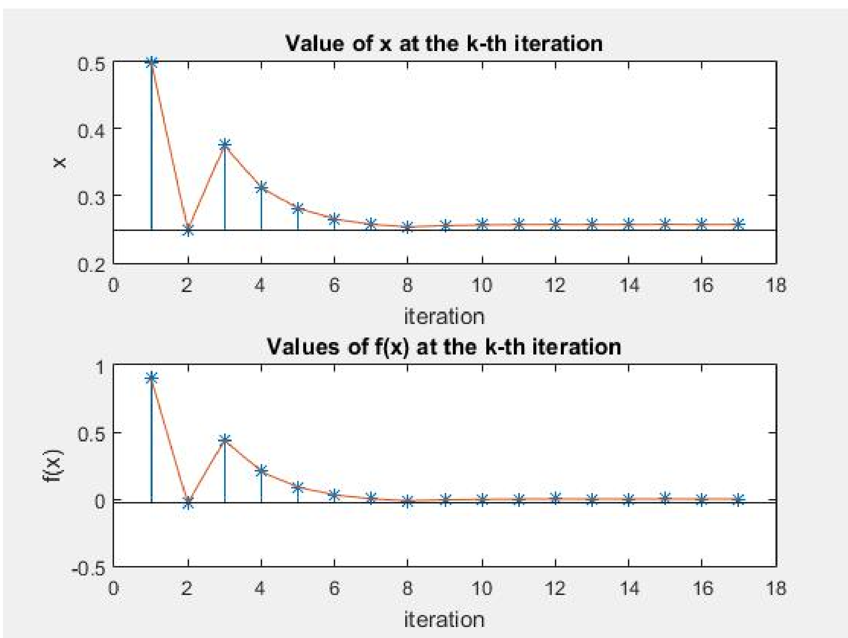


Fig.5.1 Plot iteration points for Bisection Method

b. $x \cos x - 2x^2 + 3x - 1 = 0 \quad$ for $0.2 \le x \le 0.3$ and $1.2 \le x \le 1.3$
The root of $x \cos x - 2x^2 + 3x - 1 = 0$ with the initial condition of $0.2 \le x \le 0.3$ is 0.297528076171875, the midpoint is listed and ploted below:

Midpoint List	
0.25	0.297266
0.275	0.297461
0.2875	0.297559
0.29375	0.297510
0.296875	0.297534
0.298438	0.297522
0.297656	0.297528

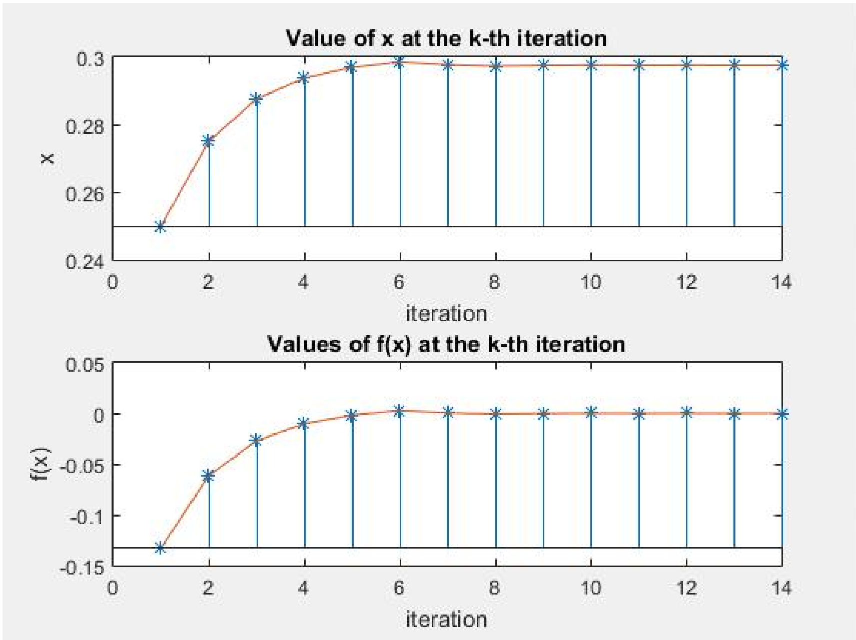


Fig.5.a.1 Plot iteration points for Bisection Method

The root of $x\cos x - 2x^2 + 3x - 1 = 0$ with the initial condition of $1.2 \leq x \leq 1.3$ is 1.256622314453125, the midpoint is listed and ploted below:

Midpoint List	
1.25	1.25664
1.275	1.25645
1.2625	1.25654
1.25625	1.25659
1.25937	1.25662
1.25781	1.25663
1.25703	1.25662

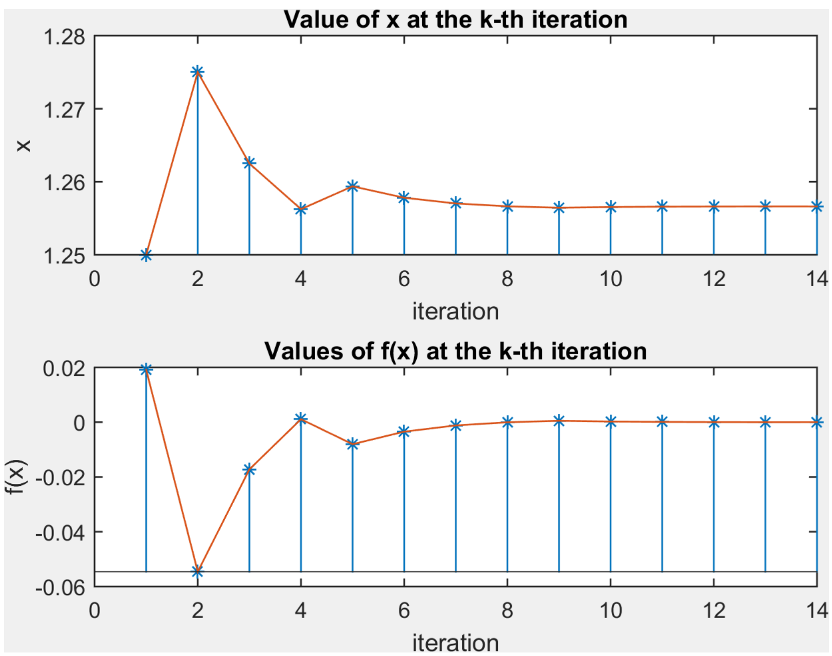


Fig.5.a.2 Plot iteration points for Bisection Method

Usage of Bisection Method and process to solve problem is shown below:

```
func1=@(x) exp(x)-x^2+3*x-2; %for problem a
func2=@(x) x*cos(x)-2*x^2+3*x-1; %for problem b
res1=Bisection(func1,[0,1],10^-5,1000)
%specify function_handle,left and right initial valuer,TOL and max iteration times ...
    correspondingly
res2=Bisection(func2,[0.2,0.3],10^-5,1000)
res3=Bisection(func2,[1.2,1.3],10^-5,1000)
```

The implement of Bisection Method is shown below:

```
function res=Bisection(func,lrMat,TOL,MaxIter,EasyForm)
if nargin==4
    EasyForm=false; %which means do not plot.
elseif size(lrMat,1)≠1 || size(lrMat,2)≠2 || nargin <4
    disp('Check you input!')
end
left=lrMat(1); right=lrMat(2);
%% Avoid fun(left) and func(right) is all positive or all negtive, for the stablility ...
    of solution, I choose not to use randi, although these behaviour may cause some ...
    soltion missing.
while func(left)*func(right)>0
    mid=(left+right)/2;
%     switch randi([0,1])
%         case 1
            right=mid;
%         case 0
%             left=mid;
%     end
end
%% State Bisection
mid=(left+right)/2;
k=1;
sol=zeros(100,1); sol(1)=mid;
fun=zeros(100,1); fun(1)=func(mid);
while (k≤MaxIter) && (abs(func(mid))>TOL)
    if func(left)*func(mid)>0
        left=mid;
    else
        right=mid;
    end
    mid=(left+right)/2;
    k=k+1;
    sol(k)=mid;
    fun(k)=func(mid);
end
res=sol(k);
disp(sol(1:k))
%% Plot And Disp(res)
if EasyForm==false
    figure1=figure;
    subplot1 = subplot(2,1,1,'Parent',figure1);
    box(subplot1,'on');
    hold(subplot1,'on');
    stem(sol(1:k),'Parent',subplot1,'Marker','*','BaseValue',min(sol(1:k)));
    plot(sol(1:k));
    xlabel('iteration');
    ylabel('x');
    title('Value of x at the k-th iteration');

    subplot2 = subplot(2,1,2,'Parent',figure1);
    box(subplot2,'on');
    hold(subplot2,'on');
    stem(fun(1:k),'Parent',subplot2,'Marker','*','BaseValue',min(fun(1:k)));
    plot(fun(1:k))
```

```
    xlabel('iteration');
    ylabel('f(x)');
    title('Values of f(x) at the k-th iteration');
    needSave=sol(1:k);
    save xMat needSave

end
```

Problem 6:

PROBLEM: Implement the fixed-point iteration method in C or matlab and find solutions accurate to within 10^{-2} for the following problems. (List pn in each iteration as well).

a. $2\sin\pi x + x = 0$ on [1, 2], use $p_0 = 1$
SOLUTION For this problem we should be careful the initial guess result from Bisection method, A crude estimate reduce to divergence while an accurate one make Fix-Point method not function.

- Estimate the initial value by plot or Monte Carlo method. Then we can use Bisection method to reduce TOL to suitable value. Here I just estimate the x coordination of intersection point.
- Then we construct a suitable $g(x) = x$ from $f(x) = 0$, and my choice is $g(x) = x - \frac{f(x)}{f'(x)}$, which is exactly another form of Newton method. Apparently, Newton method is a kind of Fix-Point method, with a perfect $g(x)$ satisfies $|f'(p)| < 1$ automatically. I love this merit.
- Now that the g(x) we choose has satisfies $|f'(p)| < 1$ automatically, as Fix-Point method show, we just need to generate x via g(x) step by step.

For problem a, the initial step for estimation is shown below, which contain $f_1(x) = x$ and $f_2(x) = -2sin(\pi x)$. Apparently, to choose $g(x) = -2sin(\pi x)$ is not proper.

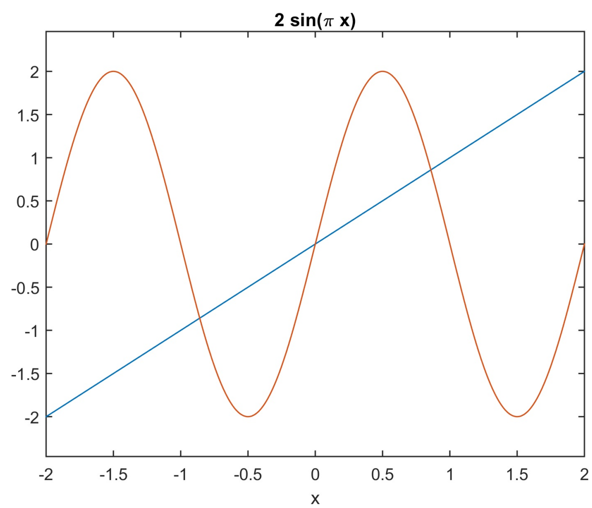


Fig.5.a.1 Figure for estimation.

My first mistake is shown below, because choose $g(x) = x - \frac{f'(x)}{f(x)}$ instead of $g(x) = x - \frac{f(x)}{f'(x)}$, the root divergent! I think I will never forget this formula, and I am impressed that computer will be correct if I choose a correct algorithm.

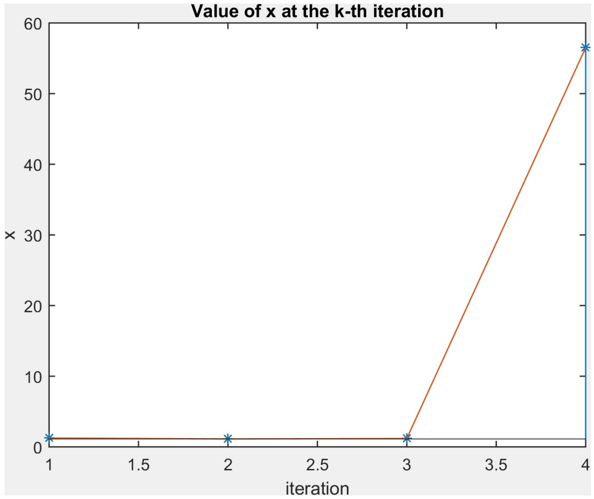


Fig.5.a.2 the result of my first time mistake.

The root for $2\sin\pi x + x = 0$ on [1, 2], use $p_0 = 1$ is 1.206034907345460, for the initial point is 1 which excludes the root near 1.6. Iterations points and graph are shown below

1.000000000000000
1.189279751107925
1.205656458413502
1.206034907345460

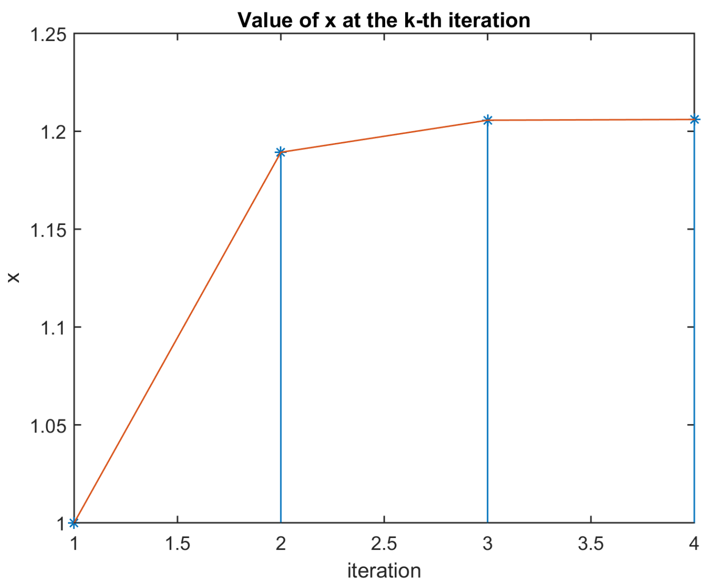


Fig.5.a.3 Plot iteration points for Bisection Method

b. $3x^2 - e^x = 0$
First we must plot the function to estimate.There are three roots. After running programme, we find they are -0.458962274194841, 0.910017665783406 and 3.733079065494898.

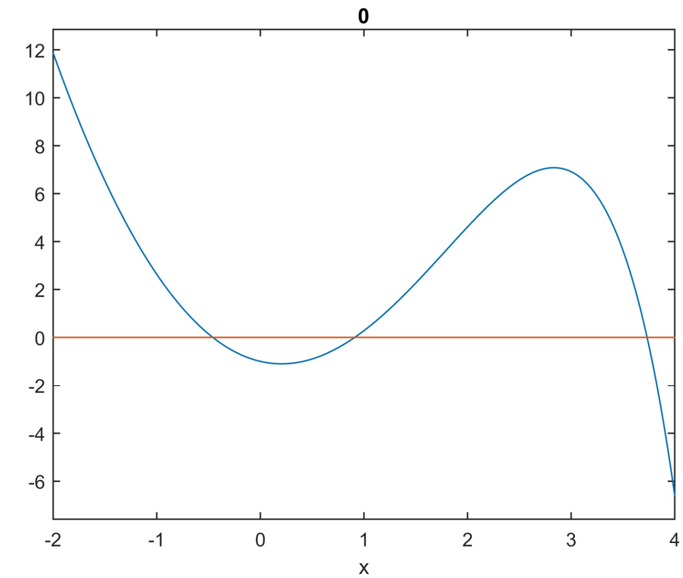


Fig.6.b For estimate

-1.000000000000000
-0.586656659702033
-0.469801907724523
-0.459053916955023
-0.458962274194841
-0.458962274194841

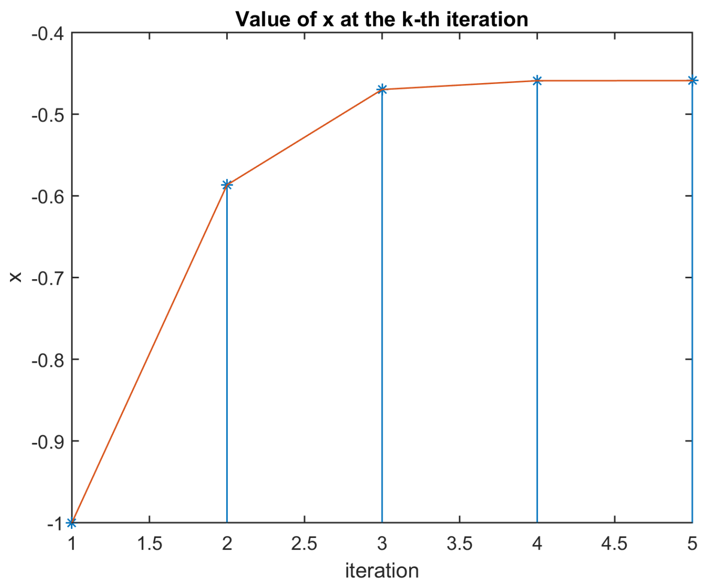


Fig.6.b.1 x in every iteration of Fix-Point method.

1.000000000000000
0.914155281832543
0.910017665783406

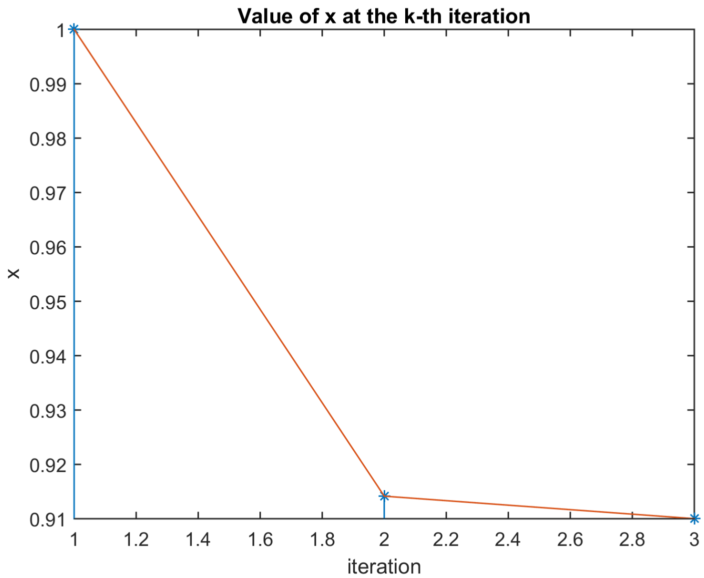
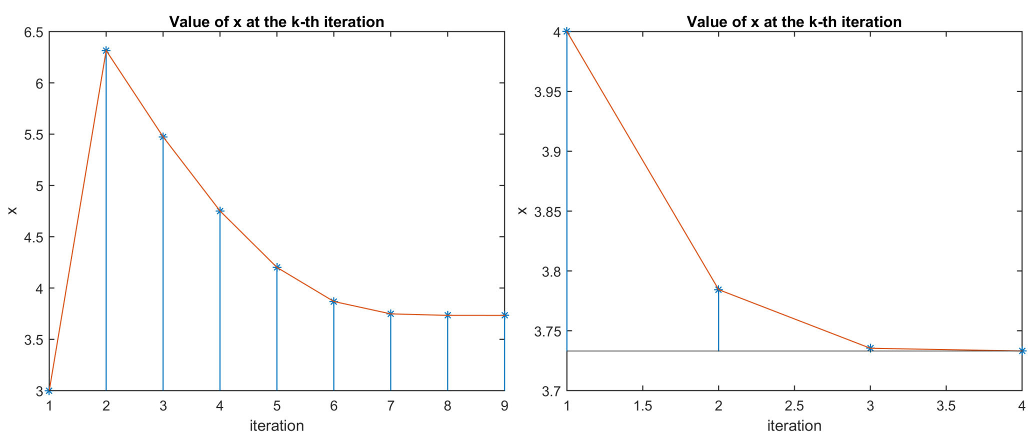


Fig.6.b.2 x in every iteration of Fix-Point method

3.000000000000000
6.315435464093259
5.474149482730207
4.751646063932222
4.201134675674261
3.868723025906899
3.747916887626595
3.733278953493994
3.733079065494898

4
3.000000000000000
6.315435464093259
4.000000000000000
3.784361145167370
3.735379375079544
3.733083897874097

What is shown below is the x of each iteration, using different initial point, 3 and 4 correspondingly. We find that Fix-Point method(Newton method here) is sensitive to initial value. A good initial estimate means very fast convergency.



x in every iteration of Fix-Point method using different initial value of x.

Usage of Fix-Point method and the process to solve the problem is shown below:

```
func3=@(x) 2*sin(pi*x)+x; %for problem a
func4=@(x) 3*x^2-exp(x); %for problem b
%% for estimate:
 figure('color',[1,1,1]);
 ezplot('x',[-2,2]);hold on;
```

```
 ezplot('2*sin(pi*x)',[-2,2]);
 figure('color',[1,1,1]);
 ezplot(func4,[-2,4]);hold on;
 ezplot('0',[-2,4]);

res4=FixPoint(func3,1); %TOL=10^-2, MaxIter=1000 is default, by using nargin.
res5=FixPoint(func4,-1);
res6=FixPoint(func4,0);
res7=FixPoint(func4,1);
res8=FixPoint(func4,3);
res9=FixPoint(func4,4);
```

The implement of Fix-Point method is shown below:

```
function res=FixPoint(func,IniGuess,TOL,MaxIter)
%global time
if nargin==3
    MaxIter=1000;
elseif nargin==2
    TOL=10^-2;
    MaxIter=1000;
else
    disp('Check You Input!');
end
%%
res=IniGuess;sol=res;
syms f gsym symx
f=func(symx);
gsym=symx-f/diff(f);
g=matlabFunction(gsym);
%%
k=1;
while(abs(g(res)-res)>TOL && k<MaxIter)
    res=g(res);
    sol(end+1)=res;
    k=k+1;
end
sol(end+1)=g(res);
disp('The Iteration Point of X is ' );
disp(sol');
figure('color',[1,1,1]); box on ; hold on;
stem(sol,'Marker','*','BaseValue',min(sol));
plot(sol);
xlabel('iteration');
ylabel('x');
title('Value of x at the k-th iteration');
%need=num2str(time);
%time=time+1;
%export_fig(gcf,need,'-m3');
```

Problem 7:

PROBLEM: $g \in C^1[a,b]$ and p be in (a,b) with $g(p)=p$ and $|g'(p)|>1$. Show that there exists a $\delta>0$ such that if $0<|p_0-p|<\delta$, then $|p_0-p|<|p_1-p|$. Thus, no matter how close the initial approximation p_0 is to p, the next iterate p_1 is farther away, so the fixed-point iteration does not converge if $p_0 \neq p$.
SOLUTION: since $g \in C^1[a,b]$, which means derivation of g is continuous and $|g'(p)|>1$, which means it is bounded.Then we get:

$$\exists \delta > 0, \forall x \in [p-\delta, p+\delta]: \qquad |g'(x)| > 1$$

Here we can get a δ, then we prove δ is what the we quests for. According to the Differential Mean Value Theorem

$$\exists \xi \in [\min(p_0, p), \max(p_0, p)] : \qquad p_1 - p = g(p_1) - g(p) = g'(\xi)(p_0 - p)$$

As we know, $0 < |p_0 - p| < \delta \Rightarrow \xi \in [p - \delta, p + \delta] \Rightarrow |g'(\xi)| > 1$, then we prove that

$$|p_1 - p| = |g(p_1) - g(p)| = |g'(\xi)(p_0 - p)| > |p_0 - p|$$

No matter how close to p p_0 is, it can never turn back, thus the fixed-point iteration does not converge if $p_0 \neq p$.