Table of Contents

Setup

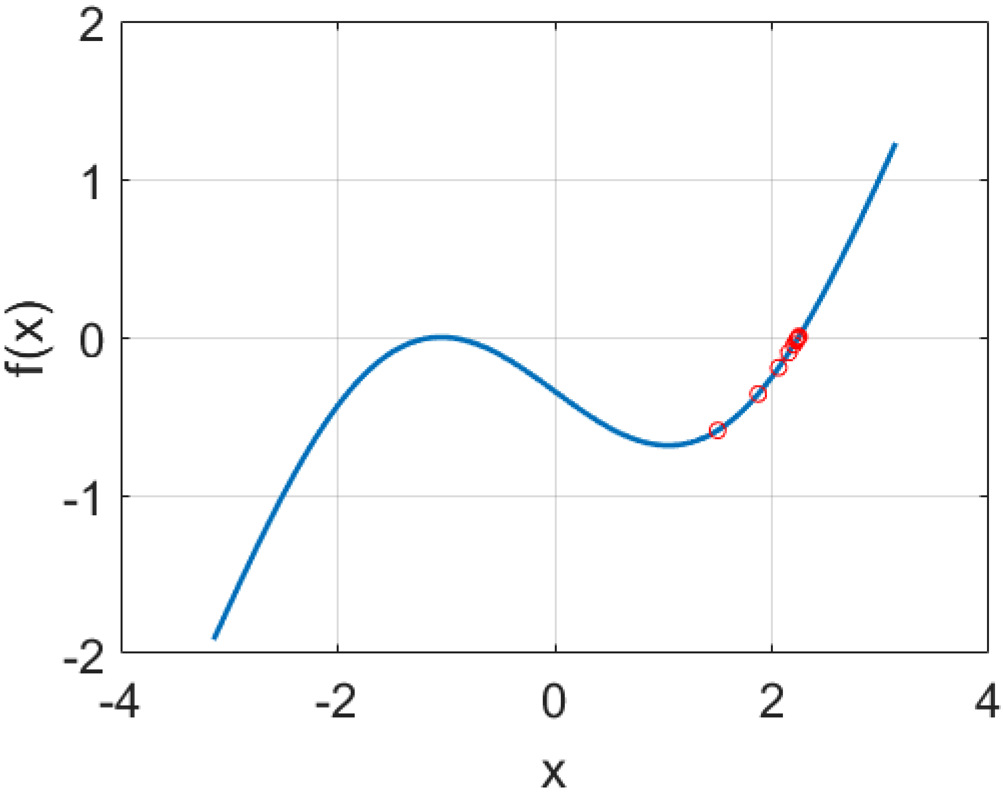
```
close all, clear all, clc
format long, format compact     % Display all digits and condense output in
 the command window
tol=1e-10;                             % Tolerance for stopping criteria
nmax=50;                             % Maximum number of iterations (in case
 stopping criterion is not achieved)
fs=18;                                 % Setting font size in plots (the default
 is a bit small)
f=@(x) (x/2)-sin(x)+(pi/6)-(sqrt(3)/2);                 % Define the function
 f
set(groot,'defaulttextfontsize',fs);
set(groot,'defaultaxesfontsize',fs);
set(groot,'defaultLineLineWidth',2) % Set line width in plots (the default is
 a bit thin)
set(groot,'defaultContourLineWidth',2) % Set line width in contour plots (the
 default is a bit thin)
set(0,'DefaultLegendAutoUpdate','off') % Stops Matlab adding extra legend
 entries automatically
pauses = 0;  % If 1 then pause between plots, if 0 then don't
```

(a)

Bisection

```
x=linspace(-pi,pi,101);                 % Define a set of x values for plotting

figure                                 % Create a new figure
plot(x,f(x))           % Plot f
grid on
xlabel('x')
ylabel('f(x)')
hold on                                % Hold allows you to overlay things on top
 of the current plot

[zero,res,niter,itersb]=bisection(f,0,3,tol,nmax)    % Run the bisection
 method with a=-1, b=1, and tol and nmax as specified above

for i=1:10
    scatter(itersb(i,1),f(itersb(i,1)),'r')   % Plot the first 10 iterates
 from bisection
```

```
    if pauses, pause, end      % pausing after each one if pauses=1
end

zero =
   2.246005589287961
res =
    -1.126554405317393e-11
niter =
    34
itersb =
   1.500000000000000
   2.250000000000000
   1.875000000000000
   2.062500000000000
   2.156250000000000
   2.203125000000000
   2.226562500000000
   2.238281250000000
   2.244140625000000
   2.247070312500000
   2.245605468750000
   2.246337890625000
   2.245971679687500
   2.246154785156250
   2.246063232421875
   2.246017456054688
   2.245994567871094
   2.246006011962891
   2.246000289916992
   2.246003150939941
   2.246004581451416
   2.246005296707153
   2.246005654335022
   2.246005475521088
   2.246005564928055
   2.246005609631538
   2.246005587279797
   2.246005598455667
   2.246005592867732
   2.246005590073764
   2.246005588676780
   2.246005589375272
   2.246005589026026
   2.246005589200649
   2.246005589287961
```



On the above, I chose interval [0,3] for the intial data.

(b)

Newton method

```
df = @(x) (1/2)-cos(x);            % Define the derivative df/dx

x_a=pi;   % Initial guess of \alpha
x_b=-pi/2;   % Initial guess \beta

x=linspace(-pi,pi,101);               % Define a set of x values for plotting
figure
plot(x,f(x),'DisplayName','f')          % Plot f
grid on
xlabel('x')
ylabel('f(x)')
hold on

[zero_a,res_a,niter_a,itersn_a]=newton(f,df,x_a,tol,nmax)    % Run Newton
[zero_b,res_b,niter_b,itersn_b]=newton(f,df,x_b,tol,nmax)    % Run Newton

scatter(itersn_a(1,1),f(itersn_a(1,1)),100,'pk','DisplayName','x_\alpha')
scatter(itersn_b(i,1),f(itersn_b(i,1)), 100,'DisplayName','x_\beta')
legend('Location','best')
```

```
for i=2:5
    scatter(itersn_a(i,1),f(itersn_a(i,1)),100,'pk')
    if pauses, pause, end     % pausing after each one if pauses=1
end

for i=2:10
    scatter(itersn_b(i,1),f(itersn_b(i,1)), 100,'^r')
    if pauses, pause, end     % pausing after each one if pauses=1
end

df_a = df(zero_a)
df_b = df(zero_b)
```

```
zero_a =
   2.246005589297974
res_a =
     0
niter_a =
     5
itersn_a =
   3.141592653589793
   2.322679521183955
   2.247862901050224
   2.246006783255555
   2.246005589298468
   2.246005589297974
zero_b =
  -1.047197557006185
res_b =
     0
niter_b =
    27
itersn_b =
  -1.570796326794897
  -1.315146743627720
  -1.183631832053614
  -1.116174561731184
  -1.081897101040887
  -1.064602959203221
  -1.055914539391675
  -1.051559664465138
  -1.049379518712337
  -1.048288763440561
  -1.047743214536057
  -1.047470397182659
  -1.047333977769972
  -1.047265765378337
  -1.047231658511738
  -1.047214604910861
  -1.047206078066725
  -1.047201814629384
  -1.047199682914469
  -1.047198617027026
  -1.047198084068886
  -1.047197817508164
  -1.047197684165446
  -1.047197617641383
  -1.047197584841854
  -1.047197565790503
  -1.047197557006185
  -1.047197557006185
df_a =
   1.125060675734896
df_b =
     5.031250416287492e-09
```

For $\alpha$, it used 5 iterations to converge, but $\beta$ used 27 iterations to converge.
The reason is that: According to the Theorem 3.4.1., we can know that if $f$ be twice continuously differentiable, and $f'(x) \neq 0$, then the Newton iteration converges to x quadratically. The df_a is (approximatively) equal to 0, so $\alpha$ converges faster.
However, according to the Remark 3.4.2., we can know that if $f'(x) = 0$, the convergence of Newton's method is only linear, that is why $\beta$ converges slower.

(c)

Fixed point iterations

```
phi=@(x) x-2*((f(x))/(df(x)));

x=linspace(-pi,pi,101);               % Define a set of x values for plotting
figure
plot(x,f(x),'DisplayName','f')          % Plot f
grid on
xlabel('x')
ylabel('f(x)')
hold on

[fixp,res,niter,itersfp1]=fixpoint(phi,x_b,tol,nmax)

for i=1:4
    scatter(itersfp1(i,1),f(itersfp1(i,1)),500,'.r')
    if pauses, pause, end     % pausing after each one if pauses=1
end
```

This modified Newton method used 4 iterations to make $\beta$ convergence.

Published with MATLAB® R2021b