# Numerical Analysis
# Assignment 4

Chuan Lu

September 18, 2017

**Problem 1.** Problem 2.2, Page 118

***Solution.*** The code of the three schemes are as follows.

```
function [root, index, iteration, error, count] = bisection(f, a, b, tol)
% Bisection for root finding;
% f: the function to find root;
% [a, b]: the interval which root lies in; f(a)*f(b) < 0;
% tol: tolerance;
if(feval(f, a)*feval(f, b) > 0)
    error('Sign of value on edges of interval shoule be differernt');
end
count = 0; index = (0:1:100)';
iteration = zeros(100, 1);
error = zeros(100, 1);
while(1)
    count = count + 1;
    root = (a+b)/2;
    iteration(count) = root;
%     error(count) = abs(feval(f, root));
    error(count) = abs(b-root);
    if(b-root < tol)
        return;
    elseif(count > 100)
        warning('Count over 100, might not converge');
        return;
    end
    if(feval(f, b)*feval(f, root) <= 0)
        a = root;
    else
        b = root;
    end
end
```

```
function [root, index, iteration, error, count] = newton(f, df, x0, tol)
% Newton's Method for root extraction;
% f, df: function and it's derivative;
% x0: initial point;
% tol: tolerance;
count = 1; index = (0:1:100)';
iteration = zeros(100, 1); iteration(1) = x0;
error = zeros(100, 1); error(1) = abs(x0);
while(1)
    count = count + 1;
    x1 = x0 - feval(f, x0) / feval(df, x0);
    iteration(count) = x1;
    error(count) = abs(x1-x0);
    if(abs(x1-x0) < tol)
        root = x1;
```

```
        return;
    elseif(count > 100)
        warning('Count over 100, may not be convergence');
        root = x1;
        return;
    end
    x0 = x1;
end
```

```
function [root, index, iteration, error, count] = secant(f, x0, x1, tol)
% Secant Method for root extraction;
% f: the function;
% x0, x1: initial points;
% tol: tolerance;
count = 2; index = (0:1:100)';
iteration = zeros(100, 1); iteration(1) = x0; iteration(2) = x1;
error = zeros(100, 1); error(1) = abs(x0); error(2) = abs(x1-x0);
while(1)
    count = count + 1;
    f0 = feval(f, x0);
    f1 = feval(f, x1);
    x2 = x1 - (x1 - x0)/(f1 - f0) * f1;
    iteration(count) = x2;
    error(count) = abs(x2-x1);
    if(abs(x2 - x0) < tol)
        root = x2;
        return;
    elseif(count > 100)
        warning('Count over 100, may not be convergence');
        root = x2;
        return;
    end
    x0 = x1;
    x1 = x2;
end
```

```
f1 = @(x)(exp(x)-3*x^2);
df1 = @(x)(exp(x)-6*x);
f2 = @(x)(x-1-0.3*cos(x));
df2 = @(x)(1+0.3*sin(x));
tol = 1e-6;
format long;

% f1: Prob 2(a);
figure(1);
% FOR THE NEGATIVE ROOT
% bisection
a = -1; b = 0;
[root, index, iteration, error, count] = bisection(f1, a, b, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
hold on;
% Newton
x0 = -0.5;
[root, index, iteration, error, count] = newton(f1, df1, x0, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
hold on;
% Secant
x0 = -1; x1 = 0;
[root, index, iteration, error, count] = secant(f1, x0, x1, tol);
```

```
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
title('Error estimation for e^x-3x^2, the negative root');
legend('Bisection', 'Newton', 'Secant');
% for the smaller positive root
% bisection
figure(2);
a = 0; b = 1;
[root, index, iteration, error, count] = bisection(f1, a, b, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
hold on;
% Newton
x0 = 0.5;
[root, index, iteration, error, count] = newton(f1, df1, x0, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
hold on;
% Secant
x0 = 0; x1 = 1;
[root, index, iteration, error, count] = secant(f1, x0, x1, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
title('Error estimation for e^x-3x^2, the smaller positive root');
legend('Bisection', 'Newton', 'Secant');
% for the larger positive root
% bisection
figure(3);
a = 1; b = 4;
[root, index, iteration, error, count] = bisection(f1, a, b, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
hold on;
% Newton
x0 = 3.5;
[root, index, iteration, error, count] = newton(f1, df1, x0, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
hold on;
% Secant
x0 = 3; x1 = 4;
[root, index, iteration, error, count] = secant(f1, x0, x1, tol);
disp([index(1:count), iteration(1:count), error(1:count)]);
semilogy(index(1:count), error(1:count));
title('Error estimation for e^x-3x^2, the larger positive root');
legend('Bisection', 'Newton', 'Secant');

%
% % f2: Prob 2(d);
% figure(2);
% % bisection
% a = 0; b = pi/2;
% [root, index, iteration, error, count] = bisection(f2, a, b, tol);
% disp([index(1:count), iteration(1:count), error(1:count)]);
% semilogy(index(1:count), error(1:count));
% hold on;
% % Newton
% x0 = 0.5;
% [root, index, iteration, error, count] = newton(f2, df2, x0, tol);
% disp([index(1:count), iteration(1:count), error(1:count)]);
% semilogy(index(1:count), error(1:count));
% hold on;
```

```
% % Secant
% x0 = 0; x1 = 1;
% [root, index, iteration, error, count] = secant(f2, x0, x1, tol);
% disp([index(1:count), iteration(1:count), error(1:count)]);
% semilogy(index(1:count), error(1:count));
%
% title('Error estimation for x-1-0.3cos(x)');
% legend('Bisection', 'Newton', 'Secant');
```

***Result.*** The result of output are as follows, and in each result there are three columns, index, value, error respectively. The first three results are for the three roots in (a), sorting from the smallest to the largest.

```
>> main
                    0   -0.500000000000000    0.500000000000000
    1.000000000000000   -0.250000000000000    0.250000000000000
    2.000000000000000   -0.375000000000000    0.125000000000000
    3.000000000000000   -0.437500000000000    0.062500000000000
    4.000000000000000   -0.468750000000000    0.031250000000000
    5.000000000000000   -0.453125000000000    0.015625000000000
    6.000000000000000   -0.460937500000000    0.007812500000000
    7.000000000000000   -0.457031250000000    0.003906250000000
    8.000000000000000   -0.458984375000000    0.001953125000000
    9.000000000000000   -0.458007812500000    0.000976562500000
   10.000000000000000   -0.458496093750000    0.000488281250000
   11.000000000000000   -0.458740234375000    0.000244140625000
   12.000000000000000   -0.458862304687500    0.000122070312500
   13.000000000000000   -0.458923339843750    0.000061035156250
   14.000000000000000   -0.458953857421875    0.000030517578125
   15.000000000000000   -0.458969116210938    0.000015258789063
   16.000000000000000   -0.458961486816406    0.000007629394531
   17.000000000000000   -0.458965301513672    0.000003814697266
   18.000000000000000   -0.458963394165039    0.000001907348633
   19.000000000000000   -0.458962440490723    0.000000953674316

                    0   -0.500000000000000    0.500000000000000
    1.000000000000000   -0.460219570045525    0.039780429954475
    2.000000000000000   -0.458963518356852    0.001256051688672
    3.000000000000000   -0.458962267538189    0.000001250818663
    4.000000000000000   -0.458962267536949    0.000000000001240

                    0   -1.000000000000000    1.000000000000000
    1.000000000000000                    0    1.000000000000000
    2.000000000000000   -0.275321257596836    0.275321257596836
    3.000000000000000   -0.588196207258892    0.312874949662057
    4.000000000000000   -0.439364813249275    0.148831394009618
    5.000000000000000   -0.457108107412876    0.017743294163601
    6.000000000000000   -0.458991546782558    0.001883439369682
    7.000000000000000   -0.458962224440445    0.000029322342112
    8.000000000000000   -0.458962267535948    0.000000043095503
    9.000000000000000   -0.458962267536949    0.000000000001000
```

```
                    0    0.500000000000000    0.500000000000000
    1.000000000000000    0.750000000000000    0.250000000000000
    2.000000000000000    0.875000000000000    0.125000000000000
    3.000000000000000    0.937500000000000    0.062500000000000
    4.000000000000000    0.906250000000000    0.031250000000000
    5.000000000000000    0.921875000000000    0.015625000000000
    6.000000000000000    0.914062500000000    0.007812500000000
    7.000000000000000    0.910156250000000    0.003906250000000
    8.000000000000000    0.908203125000000    0.001953125000000
    9.000000000000000    0.909179687500000    0.000976562500000
```

```
  10.000000000000000   0.909667968750000   0.000488281250000
  11.000000000000000   0.909912109375000   0.000244140625000
  12.000000000000000   0.910034179687500   0.000122070312500
  13.000000000000000   0.909973144531250   0.000061035156250
  14.000000000000000   0.910003662109375   0.000030517578125
  15.000000000000000   0.910018920898438   0.000015258789063
  16.000000000000000   0.910011291503906   0.000007629394531
  17.000000000000000   0.910007476806641   0.000003814697266
  18.000000000000000   0.910009384155273   0.000001907348633
  19.000000000000000   0.910008430480957   0.000000953674316

                   0   0.500000000000000   0.500000000000000
   1.000000000000000   1.165089482438443   0.665089482438443
   2.000000000000000   0.936226937560653   0.228862544877789
   3.000000000000000   0.910396664872018   0.025830272688636
   4.000000000000000   0.910007661863127   0.000389003008890
   5.000000000000000   0.910007572488714   0.000000089374413

                   0                   0                   0
   1.000000000000000   1.000000000000000   1.000000000000000
   2.000000000000000   0.780202717105698   0.219797282894302
   3.000000000000000   0.902866735744908   0.122664018639210
   4.000000000000000   0.910623538896086   0.007756803151178
   5.000000000000000   0.910004960093375   0.000618578802711
   6.000000000000000   0.910007571538623   0.000002611445248
   7.000000000000000   0.910007572488710   0.000000000950087
   8.000000000000000   0.910007572488709   0.000000000000001
```

```
                   0   2.500000000000000   1.500000000000000
   1.000000000000000   3.250000000000000   0.750000000000000
   2.000000000000000   3.625000000000000   0.375000000000000
   3.000000000000000   3.812500000000000   0.187500000000000
   4.000000000000000   3.718750000000000   0.093750000000000
   5.000000000000000   3.765625000000000   0.046875000000000
   6.000000000000000   3.742187500000000   0.023437500000000
   7.000000000000000   3.730468750000000   0.011718750000000
   8.000000000000000   3.736328125000000   0.005859375000000
   9.000000000000000   3.733398437500000   0.002929687500000
  10.000000000000000   3.731933593750000   0.001464843750000
  11.000000000000000   3.732666015625000   0.000732421875000
  12.000000000000000   3.733032226562500   0.000366210937500
  13.000000000000000   3.733215332031250   0.000183105468750
  14.000000000000000   3.733123779296875   0.000091552734375
  15.000000000000000   3.733078002929688   0.000045776367188
  16.000000000000000   3.733100891113281   0.000022888183594
  17.000000000000000   3.733089447021484   0.000011444091797
  18.000000000000000   3.733083724975586   0.000005722045898
  19.000000000000000   3.733080863952637   0.000002861022949
  20.000000000000000   3.733079433441162   0.000001430511475
  21.000000000000000   3.733078718185425   0.000000715255737

                   0   2.500000000000000   2.500000000000000
   1.000000000000000   0.169035683438571   2.330964316561429
   2.000000000000000  -6.294360859189478   6.463396542628049
   3.000000000000000  -3.147383208427374   3.146977650762104
   4.000000000000000  -1.579533818925226   1.567849389502149
   5.000000000000000  -0.827855195441739   0.751678623483487
   6.000000000000000  -0.528260047463372   0.299595147978367
   7.000000000000000  -0.462409157189051   0.065850890274321
   8.000000000000000  -0.458971637587928   0.003437519601124
   9.000000000000000  -0.458962267606549   0.000009369981378
  10.000000000000000  -0.458962267536949   0.000000000069601
```

```
                    0    1.000000000000000    1.000000000000000
    1.000000000000000    4.000000000000000    3.000000000000000
    2.000000000000000    1.122844579209717    2.877155420790283
    3.000000000000000    1.401922836216871    0.279078257007154
    4.000000000000000    0.946930674683539    0.454992161533332
    5.000000000000000    0.917253469504324    0.029677205179215
    6.000000000000000    0.910160014702522    0.007093454801802
    7.000000000000000    0.910008221025807    0.000151793676715
    8.000000000000000    0.910007572547104    0.000000648478703
    9.000000000000000    0.910007572488709    0.000000000058394
```

```
                    0    0.785398163397448    0.785398163397448
    1.000000000000000    1.178097245096172    0.392699081698724
    2.000000000000000    0.981747704246810    0.196349540849362
    3.000000000000000    1.079922474671491    0.098174770424681
    4.000000000000000    1.129009859883832    0.049087385212341
    5.000000000000000    1.104466167277661    0.024543692606170
    6.000000000000000    1.116738013580747    0.012271846303085
    7.000000000000000    1.122873936732289    0.006135923151543
    8.000000000000000    1.125941898308061    0.003067961575771
    9.000000000000000    1.127475879095946    0.001533980787886
   10.000000000000000    1.128242869489889    0.000766990393943
   11.000000000000000    1.128626364686860    0.000383495196971
   12.000000000000000    1.128434617088375    0.000191747598486
   13.000000000000000    1.128338743289132    0.000095873799243
   14.000000000000000    1.128386680188753    0.000047936899622
   15.000000000000000    1.128410648638564    0.000023968449811
   16.000000000000000    1.128422632863469    0.000011984224906
   17.000000000000000    1.128428624975922    0.000005992112453
   18.000000000000000    1.128425628919696    0.000002996056226
   19.000000000000000    1.128424130891582    0.000001498028113
   20.000000000000000    1.128424879905639    0.000000749014057

                    0    0.500000000000000    0.500000000000000
    1.000000000000000    1.167298749806364    0.667298749806364
    2.000000000000000    1.128496956086359    0.038801793720005
    3.000000000000000    1.128425093253079    0.000071862833280
    4.000000000000000    1.128425092992225    0.000000000260854

                    0                    0                    0
    1.000000000000000    1.000000000000000    1.000000000000000
    2.000000000000000    1.142446054871640    0.142446054871640
    3.000000000000000    1.128326304321001    0.014119750550639
    4.000000000000000    1.128425023756146    0.000098719435145
    5.000000000000000    1.128425092992570    0.000000069236424
    6.000000000000000    1.128425092992225    0.000000000000346
```

***Graph.*** And the error estimation is shown as 1.

## **Problem 2.** Problem 2.12, Page 119

***Solution.*** Denote

$$f(x) = x^2 - a,$$

we know the Newton's scheme for this function is

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = \frac{1}{2}(x_n + \frac{a}{x_n}).$$

If we square both side of the equation, we can get

$$x_{n+1}^2 - a = \left(\frac{x_n^2 + a}{2x_n}\right)^2 - a = \left(\frac{x_n^2 - a}{2x_n}\right)^2.$$

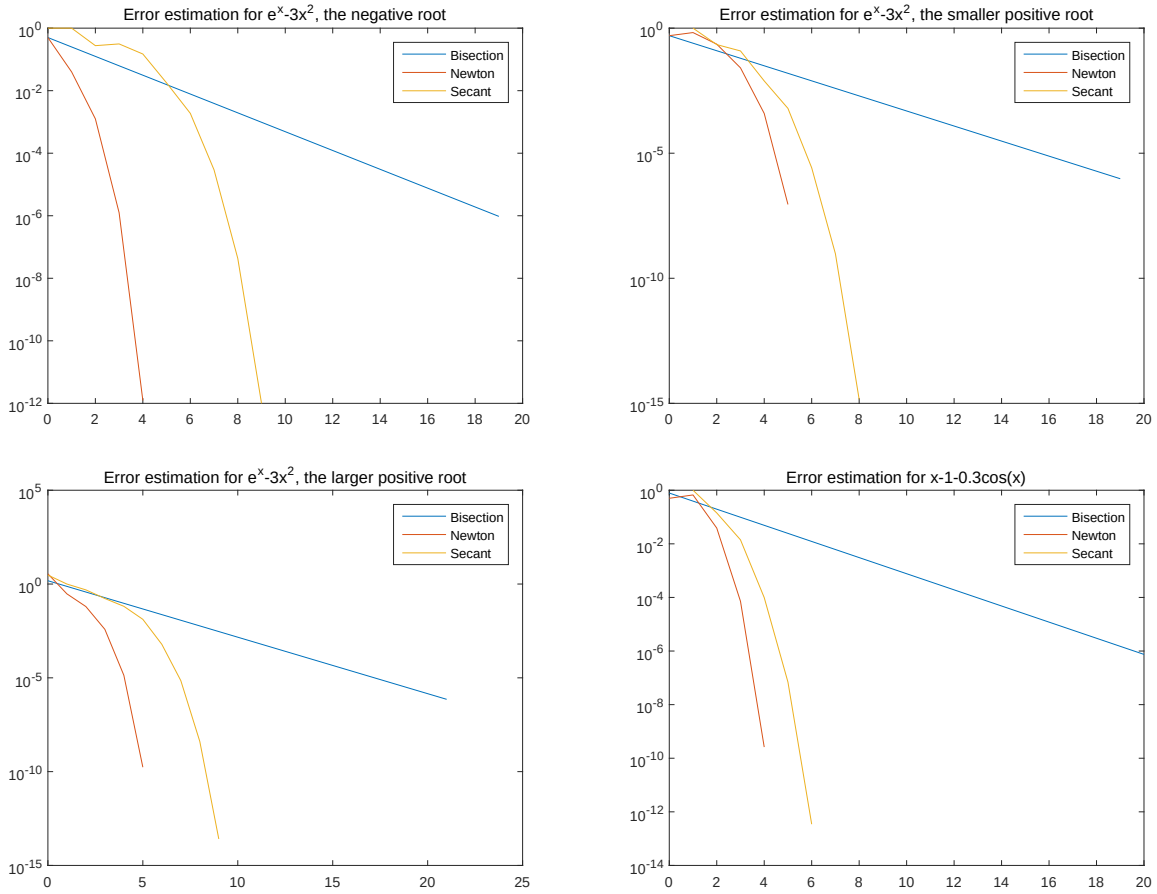


Figure 1: Error estimation

Then $x_n > \sqrt{a}$. Thus

$$x_{n+1} - x_n = \frac{a - x_n^2}{2x_n} < 0,$$

which means $\{x_n\}$ is a strictly decreasing sequence. If we consider the error, we can know

$$e_{n+1} = \sqrt{a} - x_{n+1} = -\frac{(x_n - \sqrt{a})^2}{2x_n} = -\frac{e_n^2}{2x_n},$$

and the relative error

$$\mathrm{Rel}(x_{n+1}) = \frac{e_{n+1}}{\sqrt{a}} = -\frac{e_n^2}{2\sqrt{a}x_n} = -\frac{\sqrt{a}}{2x_n}\left(\mathrm{Rel}(x_n)\right)^2.$$

Then

$$|\mathrm{Rel}(x_n)| = \left(\frac{\sqrt{a}}{2}\right)^n \frac{\mathrm{Rel}(x_0)^{2^n}}{\prod_{i=0}^{n-1} x_i} < \left(\frac{\sqrt{a}}{2}\right)^n \frac{\mathrm{Rel}(x_0)^{2^n}}{(\sqrt{a})^n} = \frac{\mathrm{Rel}(x_0)^{2^n}}{2^n}.$$

Thus

$$|\mathrm{Rel}(x_4)| < \frac{10^{-16}}{16}.$$