CSC336: Assignment 4

Due on Wednesday, Dec 5, 2018

Zhongtian Ouyang
1002341012

Problem 1

Find root using Newton's Method and Secant Method

a)
Code:

```
x0 = 1;
xprev = x0;
fprintf("%1s %20s %20s\n", "n", "x(n)", "x(n)-sqrt(2)");
fprintf("%1d %20.15f %20.15f\n", 0, x0, x0-sqrt(2));
for n = 1:5
    xn = xprev - (xprev^2 - 2)/(2*xprev);
    xprev = xn;
    fprintf("%1d %20.15f %20.15f\n", n, xn, xn-sqrt(2));
end
```

Output:

```
n                 x(n)         x(n)-sqrt(2)
0    1.000000000000000   -0.414213562373095
1    1.500000000000000    0.085786437626905
2    1.416666666666667    0.002453104293572
3    1.414215686274510    0.000002123901415
4    1.414213562374690    0.000000000001595
5    1.414213562373095    0.000000000000000
```

b)
Code:

```
x0 = 1;
x1 = 2;
xprev = x0;
xcur = x1;
f = @(x)x^2 - 2;
fprintf("%1s %20s %20s\n", "n", "x(n)", "x(n)-sqrt(2)");
fprintf("%1d %20.15f %20.15f\n", 0, x0, x0-sqrt(2));
fprintf("%1d %20.15f %20.15f\n", 1, x1, x1-sqrt(2));
for n = 2:7
    xn = xcur - f(xcur)*(xcur-xprev)/(f(xcur) - f(xprev));
    xprev = xcur;
    xcur = xn;
    fprintf("%1d %20.15f %20.15f\n", n, xn, xn-sqrt(2));
end
```

Output:

```
n              x(n)          x(n)-sqrt(2)
0     1.000000000000000    -0.414213562373095
1     2.000000000000000     0.585786437626905
2     1.333333333333333    -0.080880229039762
3     1.400000000000000    -0.014213562373095
4     1.414634146341463     0.000420583968368
5     1.414211438474870    -0.000002123898225
6     1.414213562057320    -0.000000000315775
7     1.414213562373095     0.000000000000000
```

Problem 2

Convergence Properties based on g'(x)*

(a)
$g_1(x) : g_1'(x) = \frac{2}{3}x, g_1'(2) = \frac{4}{3}$. Since $|g_1'(2)| > 1$, fixed-point iteration will diverge.
$g_2(x) : g_2'(x) = \frac{3}{2\sqrt{3x-2}}, g_2'(2) = \frac{3}{2\sqrt{3\times 2-2}} = \frac{3}{4}$. Since $|g_2'(2)| < 1$ and $g_2'(2) > 0$, the fixed-point iteration converges linearly with C = 3/4 and the iterates approach the fixed-point from one side.
$g_3(x) : g_3'(x) = \frac{2}{x^2}, g_3'(2) = \frac{1}{2}$. Since $|g_3'(2)| < 1$ and $g_3'(2) > 0$, the fixed-point iteration converges linearly with C = 1/2 and the iterates approach the fixed-point from one side.
$g_4(x) : g_4'(x) = \frac{2(x^2-3x+2)}{(2x-3)^2}, g_4'(2) = 0$. Since $|g_4'(2)| = 0$ and $g_4''(2) \neq 0$, the fixed-point iteration converges quadratically.

b)
Code:

```
g = {@(x) (x^2+2)/3;
        @(x) sqrt(3*x - 2);
        @(x) 3-2/x;
        @(x) (x^2-2)/(2*x-3);};
x0 = 2.05;
for i = 1:length(g)
    fprintf("function g%d:\n",i);
    fprintf("%1s %12s %12s %12s\n","n", "x", "Error", "C value");
    x = x0;
    prev_error = x - 2;
    for j=1:5
        x = g{i}(x);
        error = x - 2;
        fprintf("%1d %12f %12e %12e\n", j, x, error, error/prev_error);
        prev_error = error;
    end
end
```

```
```

Output:

```
function g1:
n            x        Error      C value
1     2.067500 6.750000e-02 1.350000e+00
2     2.091519 9.151875e-02 1.355833e+00
3     2.124817 1.248169e-01 1.363840e+00
4     2.171616 1.716156e-01 1.374939e+00
5     2.238638 2.386381e-01 1.390539e+00
function g2:
n            x        Error      C value
1     2.037155 3.715488e-02 7.430976e-01
2     2.027675 2.767469e-02 7.448467e-01
3     2.020649 2.064942e-02 7.461481e-01
4     2.015428 1.542756e-02 7.471184e-01
5     2.011537 1.153739e-02 7.478430e-01
function g3:
n            x        Error      C value
1     2.024390 2.439024e-02 4.878049e-01
2     2.012048 1.204819e-02 4.939759e-01
3     2.005988 5.988024e-03 4.970060e-01
4     2.002985 2.985075e-03 4.985075e-01
5     2.001490 1.490313e-03 4.992548e-01
function g4:
n            x        Error      C value
1     2.002273 2.272727e-03 4.545455e-02
2     2.000005 5.141917e-06 2.262443e-03
3     2.000000 2.643885e-11 5.141828e-06
4     2.000000 0.000000e+00 0.000000e+00
5     2.000000 0.000000e+00          NaN
```

From the output we can see our analysis in part (a) are correct.
g1: The error increases, showing the iteration diverges as we expected. Also, the C is greater than 1 and keeps increasing.
g2: The error decrease linearly. The convergence rate is linear. The C value is around and approach 0.75. The errors have the same sign.
g3: The error decrease linearly. The convergence rate is linear. The C value is around and approach 0.5. The errors have the same sign.
g4: The error decrease about quadratically. The convergence rate is qudratic. The C value squared every iteration.

Problem 3

Use matlab fzero

fun(x) is the function we are finding the root of. With some testing, I found out the fun(v0), where v0 is the volume calculated using ideal gas law, is always positive in our case, so if we can find a value v1 such that fun(v1) is negative, we get our interval. I choose that value to be 0.04266. Since $(p + a/v^2) \geq 0$ and $-(R * T) \leq 0$, if $(v - b) < 0, fun(v) < 0$. When $v = 0.04, v - b = 0.04 - b = 0.04 - 0.04267 = -0.00267 < 0$ Code:

```
R = 0.082054;
a = 3.592;
b = 0.04267;
T = 300;
Ps = [1 10 100];

fprintf("%8s %22s %22s\n", "Pressure", "Van Der Waals Volume", "Ideal Gas Law
    Volume")
for i = 1:3
    p = Ps(i);
    fun = @(v)(p + a/v^2)*(v - b) - R * T;
    v0 = R * T / p;
    v = fzero(fun, [0.04 v0]);
    fprintf("%8d %22e %22e\n", p, v, v0)
end
```

Output:

```
Pressure   Van Der Waals Volume   Ideal Gas Law Volume
       1           2.451259e+01           2.461620e+01
      10           2.354496e+00           2.461620e+00
     100           7.951083e-02           2.461620e-01
```

From the output, we can see that as pressure increases, the difference between the volume calculated using Van Der Waals equation and the volume calculated using Ideal Gas Law increases. Also, the volume from ideal gas law is always greater than the volume from Van Der Waals.

Problem 4

Newton's method

a)
To approximate r, which is the root for function $f(x) = 1/x - b$, using newtons method, we do one iteration using $x_{k+1} = x_k - f(x_k)/f'(x_k)$. In our case, $r_1 = r_0 - f(r_0)/f'(r_0)$. It is possible to avoid divisions by rearranging the $f(x)/f'(x)$ part of the formula

$$\frac{f(x)}{f'(x)} = \frac{1/x - b}{-(1/x^2)} = (\frac{1}{x} - b)(-x^2) = -\frac{x^2}{x} + bx^2 = -x + bx^2, x \neq 0$$

So with the rearranged formula, $r_1 = r_0 - (-r_0 + br_0^2) = 2r_0 - br_0^2$

b)
From part (a), we get $r_1 = 2r_0 - br_0^2$. $r = 1/b \to b = 1/r$ since $b, r \neq 0$

$$\frac{r - r_1}{r} = \frac{r - 2r_0 + br_0^2}{r} = \frac{r - 2r_0 + \frac{1}{r}r_0^2}{r} = \frac{r(r - 2r_0 + \frac{1}{r}r_0^2)}{r \times r} = \frac{r^2 - 2rr_0 + r_0^2}{r^2} = \frac{(r - r_0)^2}{r^2} = (\frac{r - r_0}{r})^2$$

c)
Since relative error is squared after one iteration. Suppose the error for r_0 can be expressed as $a \times 10^b$, then error for r_1 is $a^2 \times 10^{2b}$ where $1 \leq a^2 < 100$. So $a^2 \times 10^{2b} \in [1 \times 10^2 b, 9.999... \times 10^{2b+1}]$. From week 1 lecture notes, we know that if two number's relative error $\in [10^{-p-1}, 10{-}p + 1]$, the two numbers agree to p digits. In our case, r_0 and r agree to about b digits, r_1 and r agree to about 2b digits. r_1 has roughly twice as many correct digits as r_0 has.