Math/CS 466/666 Programming Project/Homework 3

Newton's Method: Solution Key

1. Consider Newton's method for solving $f(x) = 0$ where $f(x) = x^3 - 3$ using the starting point $x_0 = 1$.

(i) Let $e_n = x_n - 3^{1/3}$ and create a table with three columns showing n, x_n and e_n for $n = 0, 1, \ldots, 8$.

The program is

```c
#include <stdio.h>
#include <math.h>

double f(double x){
    return x*x*x-3;
}
double df(double x){
    return 3*x*x;
}
double g(double x){
    return x-f(x)/df(x);
}
int main(){
    printf("#p1i.c\n");
    double x=1,xinf=pow(3.0,1.0/3);
    printf("#%2s %22s %22s\n","n","xn","en");
    for(int i=0;;i++){
        printf("%3d %22.14e %22.14e\n",i,x,x-xinf);
        if(i>=8) break;
        x=g(x);
    }
    return 0;
}
```

The output from running the program was

```
#p1i.c
# n                     xn                     en
  0   1.00000000000000e+00  -4.42249570307408e-01
  1   1.66666666666667e+00   2.24417096359258e-01
  2   1.47111111111111e+00   2.88615408037028e-02
  3   1.44281209824934e+00   5.62527941934950e-04
  4   1.44224978959900e+00   2.19291591328644e-07
  5   1.44224957030744e+00   3.34234593127314e-14
  6   1.44224957030741e+00   8.05562214156730e-17
  7   1.44224957030741e+00   8.05562214156730e-17
  8   1.44224957030741e+00   8.05562214156730e-17
```

- **(ii)** A sign of quadratic convergence is that the number of significant digits double at each iteration. Does that happen in this case?

The number of significant digits in the approximation x_n is defined as the largest nonnegative integer k such that

$$\frac{|x_n - 3^{1/3}|}{|3^{1/3}|} \le 5 \times 10^{-k}.$$

We modify the program to report number of significant digits in each iteration. The program is

```c
#include <stdio.h>
#include <math.h>

double f(double x){
    return x*x*x-3;
}
double df(double x){
    return 3*x*x;
}
double g(double x){
    return x-f(x)/df(x);
}
unsigned sigdig(double ps,double p){
    double relerror=fabs((p-ps)/p);
    return (int)ceil(-log(relerror/5)/log(10));
}
int main(){
    printf("#p1ii.c\n");
    double x=1,xinf=pow(3.0,1.0/3);
    printf("#%2s %22s %22s %12s\n","n","xn","en","sigdig");
    for(int i=0;;i++){
        printf("%3d %22.14e %22.14e %12u\n",i,x,x-xinf,sigdig(x,xinf));
        if(i>=8) break;
        x=g(x);
    }
    return 0;
}
```

The output from running the program was

```
#p1ii.c
# n                     xn                     en       sigdig
  0   1.00000000000000e+00  -4.42249570307408e-01            2
  1   1.66666666666667e+00   2.24417096359258e-01            2
  2   1.47111111111111e+00   2.88615408037027e-02            3
  3   1.44281209824934e+00   5.62527941935009e-04            5
  4   1.44224978959900e+00   2.19291591330162e-07            8
  5   1.44224957030744e+00   3.35287353436797e-14           15
  6   1.44224957030741e+00   0.00000000000000e+00   2147483648
  7   1.44224957030741e+00   0.00000000000000e+00   2147483648
  8   1.44224957030741e+00   0.00000000000000e+00   2147483648
```

For the range $n = 1, 2, 3, 4$ the number of significant digits approximately doubles from iteration to iteration. This can also be seen from the fact that the exponents

`e-02, e-04, e-07, e-14`

appearing in the exponential notation representation of e_n approximately double from iteration to iteration.

Note that the number of significant digits don't double between the $n = 0$ and $n = 1$ iterations, because x_0 isn't close enough to $3^{1/3}$ that the asymptotic regime is in effect. Moreover, for iterations $n = 5$ and greater, further quadratic convergence is prevented by rounding error present in the double precision arithmetic used for the computation.

(iii) Comment on how rounding error effects the numerical convergence of Newton's method.

In general Newton's method is resistant to rounding errors, because any errors made in earlier iterations are corrected by subsequent iterations. Thus, there is no accumulation of rounding errors.

(iv) Write $|e_{n+1}| = M_n|e_n|^2$ and compute M_n for $n = 1, 2, 3$, and 4. In this case is M_n bigger or less than 1?

The program to compute M_n is

```c
#include <stdio.h>
#include <math.h>

double f(double x){
    return x*x*x-3;
}
double df(double x){
    return 3*x*x;
}
double ddf(double x){
    return 2;
}
double g(double x){
    return x-f(x)/df(x);
}
int main(){
    printf("#p1iii.c\n");
    double x=1,xinf=pow(3.0,1.0/3);
    printf("#%2s %22s %22s %22s\n","n","xn","en","Mn");
    for(int i=0;i<=4;i++){
        double eold=x-xinf;
        double y=g(x);
        double enew=y-xinf;
        double M=fabs(enew/eold/eold);
        printf("%3d %22.14e %22.14e %22.14e\n",i,x,eold,M);
        x=y;
    }
    return 0;
}
```

and it produces the output

```
#p1iii.c
# n                   xn                      en                      Mn
  0   1.00000000000000e+00  -4.42249570307408e-01   1.14741652343580e+00
  1   1.66666666666667e+00   2.24417096359258e-01   5.73069948436890e-01
  2   1.47111111111111e+00   2.88615408037027e-02   6.75312944377540e-01
  3   1.44281209824934e+00   5.62527941935009e-04   6.93000870018061e-01
  4   1.44224978959900e+00   2.19291591330162e-07   6.95036222636418e-01
```

Therefore, the values of M_n for $n = 1, 2, 3, 4$ are less than 1.

The fact that M_n is not extremely large helps explains why the number of significant digits nearly doubles between each iteration. Suppose the approximation x_n was good to k significant digits. Then

$$|e_n/3^{1/3}| \le 5 \times 10^{-k}.$$

Since

$$\begin{aligned}|e_{n+1}/3^{1/3}| = M_n|e_n|^2/3^{1/3} &\le (\texttt{6.95036222636418e-01})(3^{1/3})\big(5 \times 10^{-k}\big)^2\\ &\le 5 \times 5.01207846722729 \times 10^{-2k} \le 5 \times 10^{0.8-2k},\end{aligned}$$

it follows that x_n is good to at least $2k - 0.8$ significant digits. As 0.8 is small compared to even a few significant digits, then in this case, quadratic convergence is observed from nearly the first iteration.

(v) Find the limit of M_n when $n \to \infty$ analytically. What is the exact value of the limit?

Let $a = 3^{1/3}$. By Taylor's theorem for each n there exists ξ_n between a and x_n such that

$$0 = f(a) = f(x_n) + f'(x_n)(a - x_n) + \frac{1}{2} f''(\xi_n)(a - x_n)^2.$$

By definition

$$\begin{aligned} M_n &= \frac{|e_{n+1}|}{|e_n|^2} = \frac{|x_{n+1} - a|}{|x_n - a|^2} = \frac{|g(x_n) - a|}{|x_n - a|^2} = \frac{|x_n - f(x_n)/f'(x_n) - a|}{|x_n - a|^2} \\ &= \frac{|f'(x_n)(x_n - a) - f(x_n)|}{|f'(x_n)||x_n - a|^2} = \frac{|f(x_n) + f'(x_n)(a - x_n)|}{|f'(x_n)||x_n - a|^2} = \frac{|f''(\xi_n)|}{2|f'(x_n)|}. \end{aligned}$$

Therefore

$$\lim_{n\to\infty} M_n = \lim_{n\to\infty} \frac{|f''(\xi_n)|}{2|f'(x_n)|} = \frac{|f''(a)|}{2|f'(a)|}.$$

In our case

$$f'(x) = 3x^2 \qquad \text{and} \qquad f''(x) = 6x.$$

Consequently

$$\lim_{n\to\infty} M_n = \frac{6a}{2 \cdot 3a^2} = \frac{1}{a} = 3^{-1/3} \approx 0.6933612743506346.$$

Note that the first two digits of the limit agrees with the values of M_3 and M_4 computed in the previous questions.