MATH 561: Numerical Analysis I

Instructor: Prof. Wolfgang Bangerth
bangerth@colostate.edu

Homework assignment 5 – due 4/10/2017

Problem 1 (Convergence order for sequences). We often care about how quickly a sequence $\{x_n\}_{n=1}^{\infty}$ converges, for example to characterize how quickly the error goes to zero. For many methods that produce sequences of iterates x_n, we can write convergence in the form

$$\|x_n - \bar{x}\| \approx C\|x_{n-1} - \bar{x}\|^r,$$

where $\bar{x} = \lim_{n\to\infty} x_n$ is the limit to which the sequence converges, and we call C, r the *asymptotic constant* and the *rate of convergences*, respectively. In the limit $n \to \infty$, the approximate equality usually becomes an exact equality, i.e., the difference between left and right hand side is of higher order. In practice, one typically wants C to be small and r to be large because this corresponds to rapid convergence.

In the particular case where the sequence converges to zero (for example, if the x_n denote the *error* of an iteration, the form above simplifies to

$$\|x_n\| \approx C\|x_{n-1}\|^r.$$

For many methods, we can determine r analytically, and maybe even C. However, it is often instructive in practice to verify that the order one gets from an implementation indeed matches what one expects from theory. For the following four cases of sequences, each of which has $\bar{x} = 0$, determine the order of convergence and the asymptotic constant:

(a)

$$a_n = 5.0625, 2.25, 1, \frac{4}{9}, \frac{16}{81}$$

(b)

$$b_n = 2.718, 2.175, 1.740, 1.392, 1.113, 0.8907$$

(c)

$$c_n = 0.318, 0.180, 0.0761, 0.021, 3.04 \cdot 10^{-3}, 1.68 \cdot 10^{-4}, 2.17 \cdot 10^{-6}$$

(c)

$$\begin{aligned} d_n = 0.9, 0.899473, 0.898894, 0.898257, 0.897557, 0.896787, 0.895942,\\ 0.895013, 0.893992, 0.89287, 0.891638, 0.890284, 0.888798. \end{aligned}$$

All of these sequences converge to zero, though at different speeds. For each sequence, determine how many terms are necessary to get the value below 10^{-8}, i.e., determine the smallest n so that $a_n \leq 10^{-8}$ (and similarly for b_n, c_n, d_n).

(20 points)

Problem 2 (Secant method). This problem is an example of finding the root of a function f that is only given in form of a procedure, a likely case in applications, instead of as a closed form expression.

In order to define the function $g(x)$, consider the following iteration: set $a_0 = 1$ and compute the values a_i by the following iteration:

$$a_i = a_{i-1} + \frac{x \cos a_{i-1} + x}{10}.$$

Clearly, we can compute a_1 from $a_0 = 1$ for each value of x. Similarly, we can compute a_2 from a_1, and so on. Now, let $g(x)$ be the function whose value equals a_{10} for any given value of x.

a) Write a function that given a value x returns $g(x) = a_{10}$ by computing the iteration above. Use your program to plot $g(x)$ in the range $-10 \leq x \leq 10$.

b) Assume we want to solve the equation $f(x) = 0$ where $f(x) = g(x) - 2$. State why Newton's method may be ill-suited for this task.

c) Write a program that finds a root of $f(x) = g(x) - 2$ up to 6 digits accuracy using the secant method. Use $x_0 = 0, x_1 = 1$ as starting points. State how many iterations you needed to get the desired accuracy. (Hint: Because the problem – probably – doesn't have an analytic solution, the exact location x^* of the root of $f(x)$ is unknown. So how do you know when your iterate x_k is accurate to six digits, i.e. that $x_k - x^* < 10^{-6}$? In order to test algorithms, one often lets them run for a quite significant number of iterations, for example so that between x_k and x_{k+1} the 10th or 12th digit doesn't change any more. If that is the case, then one can be virtually assured that the first 9 or 11 digits of x_{k+1} are correct. You can then use this as a pretty good approximation of x^* and compare the first few iterates against it, to count how many it takes so that the first 6 digits coincide.)

d) Write a program that solves the same problem using the bisection method instead of the secant method, using $a_0 = 0, b_0 = 1$ as the initial interval. State how many iterations it takes this time to get 6 correct digits. How do the secant and the bisection method compare in terms of function evaluations for this problem? **(30 points)**

Problem 3 (Root finding methods). Compare, in words, the bisection method, Newton's method, and the secant method with respect to the following criteria: reliability of finding a root of a function, speed of convergence, complexity (i.e., a method is better if it needs fewer evaluations of $f(x)$ per iteration, or if it only needs function values rather than derivatives). **(10 points)**

Problem 4 (Multidimensional Newton). Apply Newton's method to the problem of finding a solution $x^* = (x_1^*, x_2^*)$ so that

$$f_1(x_1, x_2) = e^{x_2 - \sin(x_1)} - 1 = 0,$$
$$f_2(x_1, x_2) = e^{x_2 + \sin(x_1)} - 1 = 0.$$

It has an obvious solution at $x^* = (0, 0)$.

a) Write a program that finds this minimum by starting at a point in the vicinity of the solution. Show how the iterates x_k converge to x^* and determine the convergence rate and asymptotic constant.

b) We know that Newton's method only converges when we start close enough to the solution. Try what happens if you start from points farther and farther away from $x^* = (0, 0)$. Describe and explain in pictures and words what you observe.

(20 points)

Problem 5 (Finding minima). Using a program that implements Newton's method optimization, find minima of the function

$$f(x_1, x_2, \ldots, x_{10}) = \sin(x_1)\sin(x_2)\cdots\sin(x_{10}).$$

Demonstrate that the point you converge to is indeed a minimum. Can you find starting points for which the method converges to something other than a minimum? **(20 points)**