M348-53610: Scientific Computing
Homework # 02
Handout: 01/26/2016, Tuessday
Due: 02/02/2016, Tuesday

Submission. Please make your homework neat and stapled. You have to submit your homework in ECJ 1.204 before **3:00 PM** on the due date. Note that *no late homework will be accepted without compelling reasons.*

---

1 To be Graded

Problem 1. Let $f : \mathbb{R} \to \mathbb{R}$ be defined as $f(x) = \sqrt{x} - \cos x$ on $[0, 1]$. Perform 3 steps of the Bisection method to find the first three approximations, c_3, to the root of $f(x) = 0$. Write down your calculation procedure.

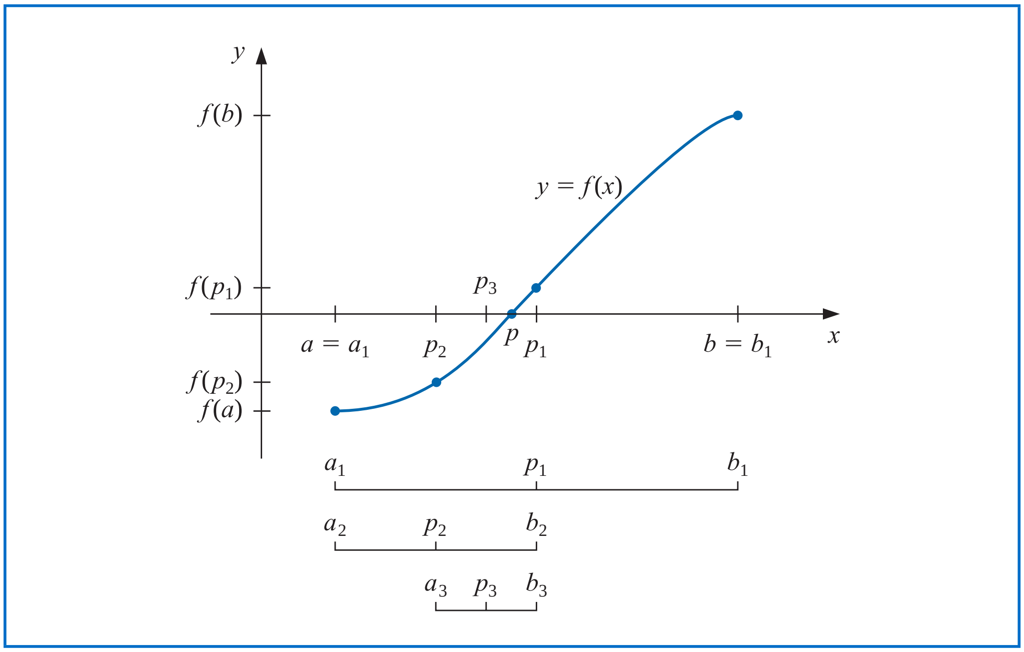
Problem 2. Consider the Bisection method for finding a root of $f(x) = 0$ on $[a, b]$. We showed in class that the error, i.e. accuracy, of the k-th approximation by the Bisection method is bounded by $\varepsilon = \dfrac{|b-a|}{2^k}$. Use the Bisection method to find the solution accurate to within $\varepsilon = 10^{-2}$ for $x^4 - 2x^3 - 4x^2 + 4x + 4 = 0$ on the interval $[0, 2]$.

Problem 3 (Programming Assignment)**.** A full description of the Bisection Algorithm is provided in Algorithm 2.1 of Burden and Faires (also attached here in case you do not have the textbook).
(a) Implement the Bisection algorithm in Matlab, C++ or any other program language that you prefer. Modify the algorithm such that your program output the iteration number, the approximate root, and the function value at the current iteration for each iteration of the algorithm. Include the full program in your submission.
(b) Use the program to find root of the function $f(x) = e^x - x^2 + 3x - 2$ on $[0, 1]$ within accuracy of 10^{-8}. Include output of all iterations in your submission.

2 Reading Assignments

- Review Sections 2.1 and 2.2 of Burden & Faires or 3.1 and 3.9 of Epperson.

Figure 2.1



ALGORITHM **2.1**

Bisection

To find a solution to $f(x) = 0$ given the continuous function f on the interval $[a, b]$, where $f(a)$ and $f(b)$ have opposite signs:

INPUT endpoints a, b; tolerance TOL; maximum number of iterations N_0.

OUTPUT approximate solution p or message of failure.

Step 1 Set $i = 1$;
$FA = f(a)$.

Step 2 While $i \leq N_0$ do Steps 3–6.

Step 3 Set $p = a + (b - a)/2$; (*Compute* p_i.)
$FP = f(p)$.

Step 4 If $FP = 0$ or $(b - a)/2 < TOL$ then
OUTPUT (p); (*Procedure completed successfully*.)
STOP.

Step 5 Set $i = i + 1$.

Step 6 If $FA \cdot FP > 0$ then set $a = p$; (*Compute* a_i, b_i.)
$FA = FP$
else set $b = p$. (*FA is unchanged.*)

Step 7 OUTPUT ('Method failed after N_0 iterations, N_0 =', N_0);
(*The procedure was unsuccessful.*)
STOP. ■

Other stopping procedures can be applied in Step 4 of Algorithm 2.1 or in any of the iterative techniques in this chapter. For example, we can select a tolerance $\varepsilon > 0$ and generate $p_1, \ldots, p_N$ until one of the following conditions is met: