

## 0.1 Root Finding in One Dimension

This is an optional, introductory, non-examinable project. Unlike the other projects **there are no marks awarded for it**. Also, unlike the other projects, you may collaborate as much as you like, and (if your College is willing) have a supervision on the project. A model answer will be provided a few weeks into the Michaelmas Term.

### The Methods

The aim of this project is to study three numerical methods for finding roots of an algebraic or transcendental equation  $F(x) = 0$  by:

- (a) binary search (or bisection or interval halving);
- (b) fixed-point (or Picard) iteration after rewriting the equation in the form or, more precisely, one of the *non-unique* forms,  $x = f(x)$ , and then using the iteration scheme

$$x_N = f(x_{N-1}), \quad (1)$$

with a suitable initial guess  $x_0$ ;

- (c) Newton-Raphson iteration (which is a special case of fixed-point iteration), using the scheme

$$x_N = x_{N-1} - \frac{F(x_{N-1})}{F'(x_{N-1})}. \quad (2)$$

The theoretical background to these methods is covered in most textbooks on *Numerical Analysis* (a few of which are listed at the end of this project).\*

### Convergence

If the exact root being approximated is  $x_*$ , we define the *truncation error* in the  $N^{\text{th}}$  iterate as  $\epsilon_N = x_N - x_*$ . The method is said to have  $p^{\text{th}}$ -order convergence if

$$|\epsilon_{N-1}| < \eta \quad \Rightarrow \quad |\epsilon_N| \leq C|\epsilon_{N-1}|^p \quad (3)$$

for some positive constants  $\eta$  and  $C$  (first-order convergence,  $p = 1$ , requires  $C < 1$ ). It can be shown that

- (a) binary search is first-order convergent,
- (b) fixed-point iteration, when convergent, is *in general* first-order convergent for a simple root, i.e. one with  $F'(x_*) \neq 0$ ,
- (c) Newton-Raphson iteration, when convergent, is second-order convergent for a simple root, but only first-order convergent for a multiple root.

---

\* There are also articles in *Wikipedia*, but that on fixed-point iteration is in need of some improvement as of July 2011.

## Examples

The cases to be studied as examples are

$$F(x) \equiv 2x - 3 \sin x + 5 = 0, \quad (4)$$

and

$$F(x) \equiv x^3 - 8.5x^2 + 20x - 8 = 0. \quad (5a)$$

Note that equation (5a) can be factorised and rewritten as

$$F(x) \equiv \left(x - \frac{1}{2}\right)(x - 4)^2 = 0. \quad (5b)$$

**Question 1** Show graphically that equation (4) has exactly one root (which is in fact  $-2.88323687\dots$ ).

## Binary Search

**Programming Task:** write a program to solve equation (4) by binary search.<sup>†</sup> Provide for termination of the iteration as soon as the truncation error is guaranteed to be less than  $0.5 \times 10^{-5}$ , and print out the number of iterations,  $N$ , as well as the estimate of the root. Run the program for a number of suitable starting values to check that it is working; include some of these results in your report.

**Question 2** Suppose that the rounding error in evaluating  $F(x)$  in equation (4) is at most  $\delta$  for  $|x| < \pi$ . By considering a Taylor expansion of  $F(x)$  near  $x_*$ , or otherwise, estimate the accuracy that may be expected for the calculated value of the root.

*Hint:* note that  $|F'(x)| > 4$  for  $-5\pi/4 < x < -3\pi/4$ .

## Fixed-Point Iteration

There are many possible choices of  $f$ , e.g.

$$f(x) = x - h(F(x)), \quad (6)$$

for some function<sup>‡</sup>  $h(F)$  such that  $h(0) = 0$ .

**Programming Task:** write a program to implement the iteration scheme in equation (1) for general  $f$ . Provide for termination of the process as soon as  $|x_N - x_{N-1}| < \epsilon$  or when  $N = N_{max}$ , whichever occurs first. Print out the values of  $N$  and  $x_N$  for each  $N$ , so that you can watch the progress of the iteration.

**Question 3** Use the program to solve (4) by fixed-point iteration by taking

$$h(F) = \frac{F}{2+k} \quad (7a)$$

in (6), so that

$$f(x) = \frac{3 \sin x + kx - 5}{2+k}, \quad (7b)$$

for some constant  $k$ .

---

<sup>†</sup> You may like to consider using a recursive function.

<sup>‡</sup> Or *functional*.

- (i) First run the program with  $k = 0$ ,  $\epsilon = 10^{-5}$ ,  $x_0 = -2$ ,  $N_{max} = 10$ . Plot  $y = f(x)$  and  $y = x$  on the same graph, and use these plots to show why convergence should not occur. Explain the divergence by identifying a theoretical criterion that has been violated.<sup>§</sup>
- (ii) Determine values of  $k$  for which convergence is guaranteed if  $x_N$  remains in the range  $(-\pi, -\pi/2)$ .
- (iii) Choose, giving reasons, a value of  $k$  for which *monotonic* convergence should occur near the root, and also a value for which *oscillatory* convergence should occur near the root. Verify that these two values of  $k$  give the expected behaviour, by running the program with  $N_{max} = 20$ .
- (iv) Also run the case  $k = 16$ . This should converge only slowly, so set  $N_{max} = 50$ . Discuss whether the truncation error is expected to be less than  $10^{-5}$  in this case?
- (v) Discuss whether your results are consistent with first-order convergence.

**Question 4** Now use your program to find the *double root* of equation (5a) by fixed-point iteration by taking

$$h(F) = \frac{1}{20} F, \quad (8a)$$

in (6), so that

$$f(x) = \frac{1}{20}(-x^3 + 8.5x^2 + 8). \quad (8b)$$

By considering  $f'(x_*)$  explain why convergence will be slow at a multiple root for any choice of differentiable function  $h$  in (6).

In your calculations some care may be needed over the choice of  $x_0$ . Also,

- (a) since convergence will be slow, take  $N_{max} = 1000$ ;
- (b) suppress the printing of each iterate, but print out the *final* values of  $N$  and  $x_N$ .

Is this an example of first-order convergence? Does the termination criterion ensure a truncation error of less than  $10^{-5}$ ?

*Note:* it can be shown that the truncation error  $\epsilon_N$  is asymptotic to  $40/(7N)$  as  $N \rightarrow \infty$ .

## Newton-Raphson Iteration

A refinement of (6) is to let  $h$  depend on the derivatives of  $F$ , i.e.

$$f(x) = x - h(F, F', F'', \dots). \quad (9a)$$

In Newton-Raphson iteration

$$h = \frac{F}{F'}. \quad (9b)$$

**Programming Task:** modify your program to recalculate the root of equation (4), and the double root of equation (5a), using Newton-Raphson iteration.

**Question 5** For equation (4), experiment with various  $x_0$  until you have demonstrated a case that converges, and also a case that has not converged in 10 iterations. In the unconverted case, show graphically what happened in the first few iterations.

For both equation (4) and equation (5a) do your (converged) results bear out the theoretical orders of convergence? Comment on the effects of rounding error.

*Hint:* you may want to use a smaller value for  $\epsilon$ .

---

<sup>§</sup> The references at the end may prove helpful.

## References

- [1] Epperson, J.F., *An Introduction to Numerical Methods and Analysis*, John Wiley & Sons (2007). ISBN-13: 9780470049631.
- [2] Kharab, A. and Guenther, R.B., *An Introduction to Numerical Methods: A MATLAB Approach*, Second Edition, CRC Press (2005). ISBN-13: 9781584885573
- [3] Press, W.H., Teukolsky, S.A. and Vetterling, W.T. and Flannery, B.P., *Numerical Recipes: The Art of Scientific Computing*, Third Edition, Cambridge University Press (2007). ISBN-10: 0521880688.
- [4] Süli, E. and Mayers, D., *An Introduction to Numerical Analysis*, Cambridge University Press (2003). ISBN-10: 0521810264 (hardback), ISBN-10: 0521007941 (paperback).