Lecture 1
chapter 1
Introduction

Numerical analysis is concerned with the development and analysis of methods for the numerical solution of practical problems. Traditionally, these methods have been mainly used to solve problems in the physical sciences and engineering. However, they are finding increasing relevance in a much broader range of subjects including economics and business studies.

The first stage in the solution of a particular problem is the formulation of a mathematical model. Mathematical symbols are introduced to represent the variables involved and physical (or economic) principles are applied to derive equations which describe the behavior of these variables. Unfortunately, it is often impossible to find the exact solution of the resulting mathematical problem using standard techniques. In fact, there are very few problems for which an analytical solution can be determined. For example, there are formulas for solving quadratic, cubic and quartic polynomial equations, but no such formula exists for polynomial equations of degree greater than four or even for a simple equation such as

$$x = \cos(x)$$

Similarly, we can certainly evaluate the integral

$$A = \int_a^b e^x dx = e^x|_a^b$$

as $e^b - e^a$, but we cannot find the exact value of

$$A = \int_a^b e^{x^2} dx$$

since no function exists which differentiates to e^{x^2}. Even when an analytical solution can be found it may be of more theoretical than practical use. For example, if the solution of a differential equation

$$y'' = f(x, y, y')$$

is expressed as an infinite sum of Bessel functions, then it is most unsuitable for calculating the numerical value of y corresponding to some numerical value of x.

Errors

Computations generally yield approximations as their output. This output may be an approximation to a true solution of an equation, or an approximation of a true value of some quantity. Errors are commonly measured in one of two ways: absolute error and relative error as the following definition.

Definition 1. *If x_A is an approximation to x_T, the **error** is defined as*

$$err(x_A) = x_T - x_A \tag{1}$$

*The **absolute error** is defined as*

$$Aerr(x_A) = |err(x_A)| = |x_T - x_A| \tag{2}$$

*And the **relative error** is given by*

$$rel(x_A) = \frac{Absolute\ error}{True\ value} = \frac{|x_T - x_A|}{|x_T|}, \qquad x_T \neq 0 \tag{3}$$

Note that if the true value happens to be zero, $x = 0$, the relative error is regarded as undefined. The relative error is generally of more significance than the absolute error.

Let $x_T = \frac{19}{7} \approx 2.714285$ and $x_A = 2.718281$. Then

$$err\,(x_A) = x_T - x_A = \frac{19}{7} - 2.718281 \approx -0.003996$$

$$Aerr\,(x_A) = |err\,(x_A)| \approx 0.003996$$

$$rel\,(x_A) = \frac{Aerr\,(x_A)}{x_T} = \frac{0.003996}{2.7142857} \approx 0.00147$$

Example 0.1. *Consider the following table*

x_T	x_A	*Absolute Error*	*Relative Error*
1	*0.99*	*0.01*	*0.01*
1	*1.1*	*0.01*	*0.01*
–1.5	*–1.2*	*0.3*	*0.2*
100	*99.99*	*0.01*	*0.0001*
100	*99*	*1*	*0.01*

Example 0.2. *Consider two different computations. In the first one, an estimate* $x_A = 0.003$ *is obtained for the true value* $x_T = 0.004$*. In the second one,* $y_A = 1238$ *for* $y_T = 1258$*. Therefore, the absolute errors are*

$$Aerr(x_A) = |x_T - x_A| = 0.001, \qquad Aerr(y_A) = |y_T - y_A| = 20$$

The corresponding relative errors are

$$rel(x_A) = \frac{Aerr(x_A)}{x_T} = \frac{0.001}{0.004} = 0.25,$$

$$rel(y_A) = \frac{Aerr(y_A)}{y_T} = \frac{20}{1258} = 0.0159$$

We notice that the absolute errors of 0.001 *and* 20 *can be rather misleading, judging by their magnitudes. In other words, the fact that* 0.001 *is much smaller than* 20 *does not make the first error a smaller error relative to its corresponding computation.*

In fact, looking at the relative errors, we see that 0.001 *is associated with a* 25% *error, while* 20 *corresponds to* 1.59% *error, much smaller than the first. Because they convey a more specific type of information, relative errors are considered more significant than absolute errors.*

Lecture 2

Computational and Errors

Numerical methods are procedures that allow for efficient solution of a mathematically formulated problem in a finite number of steps to within an arbitrary precision. Computers are needed in most cases. A very important issue here is the errors caused in computations.
A numerical algorithm consists of a sequence of arithmetic and logical operations which produces an approximate solution to within any prescribed accuracy. There are often several different algorithms for the solution of any one problem. The particular algorithm chosen depends on the context from which the problem is taken. In economics, for example, it may be that only the general behavior of a variable is required, in which case a simple, low accuracy method which uses only a few calculations is appropriate. On the other hand, in precision engineering, it may be essential to use a complex, highly accurate method, regardless of the total amount of computational effort involved.
Once a numerical algorithm has been selected, a computer program is usually written for its implementation. The program is run to obtain numerical results, although this may not be the end of the story. The computed solution could indicate that the original mathematical model needs modifying with a corresponding change in both the numerical algorithm and the program.
Although the solution of 'real problems' by numerical techniques involves the use of a digital computer or calculator, Determination of the eigenvalues of large matrices, for ex-

ample, did not become a realistic proposition until computers became available because of the amount of computation involved. Nowadays any numerical technique can at least be demonstrated on a microcomputer, although there are some problems that can only be solved using the speed and storage capacity of much larger machines.

There exist four possible sources of error:

1. **Errors in the formulation of the problem** to be solved.
   (a) Errors in the mathematical model. For example, when simplifying assumptions are made in the derivation of the mathematical model of a physical system. (Simplifications).
   (b) Error in input data. (Measurements).
2. **Approximation errors**
   (a) Discretization error.
   (b) Convergence error in iterative methods.
   (c) Discretization/convergence errors may be estimated by an analysis of the method used.
3. **Roundoff errors**: This error is caused by the computer representation of numbers.
   (a) Roundoff errors arise everywhere in numerical computation because of the finite precision arithmetic.
   (b) Roundoff errors behave quite unorganized.
4. **Truncation error**: Whenever an expression is approximated by some type of a mathematical method. For

example, suppose we use the Maclaurin series representation of the sine function:

$$\sin\alpha = \sum_{n=odd}^{\infty} \frac{(-1)^{\frac{(n-1)}{2}}}{n!}\alpha^n = \alpha - \frac{1}{3!}\alpha^3 + \frac{1}{5!}\alpha^5 - \cdots + \frac{(-1)^{\frac{(m-1)}{2}}}{3!}\alpha^m + E_m$$

where E_m is the tail end of the expansion, neglected in the process, and known as the truncation error.

0.1 ERRORS AND STABILITY

The majority of numerical methods involve a large number of calculations which are best performed on a computer or calculator. Unfortunately, such machines are incapable of working to infinite precision and so small errors occur in nearly every arithmetic operation. Even an apparently simple number such as $2/3$ cannot be represented exactly on a computer. This number has a non-terminating decimal expansion

$$0.66666666666666\cdots$$

and if, for example, the machine uses ten-digit arithmetic, then it is stored as

$$0.666\,666\,666\,7$$

(In fact, computers use binary arithmetic. However, since the substance of the argument is the same in either case, we restrict our attention to decimal arithmetic for simplicity). **The difference between the exact and stored values is called the rounding error** which, for this example, is

$$-0.000\,000\,000\,033\,33...$$

Suppose that for a given real number α the digits after the decimal point are

$$d_1 d_2 \cdots d_n d_{n+1} \cdots$$

To round α to n decimal places (abbreviated to nD) we proceed as follows. If $d_{n+1} < 5$, then α is rounded down; all digits after the nth place are removed. If $d_{n+1} \geq 5$, then α is rounded up; d_n is increased by one and all digits after the nth place are removed. It should be clear that in either case the magnitude of the rounding error does not exceed 0.5×10^{-n}.
In most situations the introduction of rounding errors into the calculations does not significantly affect the final results. However, in certain cases it can lead to a serious loss of accuracy so that computed results are very different from those obtained using exact arithmetic. The term instability is used to describe this phenomenon.
There are two fundamental types of instability in numerical analysis - **inherent** and **induced**. The first of these is a fault of the problem, the second of the method of solution.

Definition 2. *A problem is said to be **inherently unstable** (or **ill - conditioned**) if small changes in the data of the problem cause large changes in its solution.*

This concept is important for two reasons. Firstly, the data may be given as a set of readings from an analogue device such as a thermometer or voltmeter and as such cannot be measured exactly. If the problem is ill-conditioned then any numerical results, irrespective of the method used to obtain them, will be highly inaccurate and may be worthless. The second reason is that even if the data is exact it will not necessarily be stored exactly on a computer. Consequently, the problem which the computer is attempting to solve may differ slightly from the one originally posed. This does not usually matter, but if the problem is ill-conditioned then the computed results may differ wildly from those expected.

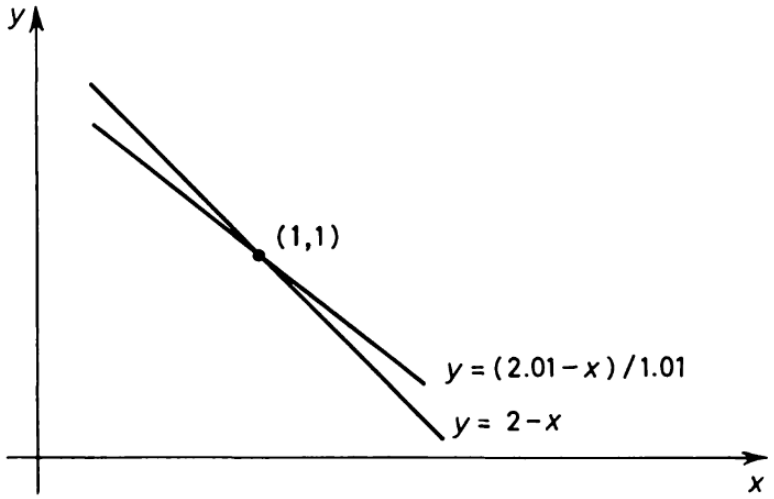

Figure 1: sketche of example 0.3

Example 0.3. *Consider the simultaneous linear equations*

$$x + y = 2$$
$$x + 1.01y = 2.01$$

which have solution $x = y = 1$*. If the number* 2.01 *is changed to* 2.02*, the corresponding solution is* $x = 0$*,* $y = 2$*. We see that a* 0.5% *change in the data produces a* 100% *change in the solution. It is instructive to give a geometrical interpretation of this result. The solution of the system is the point of intersection of the two lines* $y = 2 - x$ *and* $y = (2.01 - x)/1.01$*. These lines are sketched in figure 1. It is clear that the point of intersection is sensitive to small movements in either of these lines since they are nearly parallel. In fact, if the coefficient of* y *in the second equation is* 1.00*, the two lines are exactly parallel and the system has no solution. This is fairly typical of ill-conditioned problems. They are often close to 'critical' problems which either possess infinitely many solutions or no solution whatsoever.*

Example 0.4. *Consider the initial value problem*

$$y'' - 10y' - 11y = 0; \quad y(0) = 1, \quad y'(0) = -1$$

defined on $x \geq 0$*. The corresponding auxiliary equation has roots* -1 *and* 11*, so the general solution of the differential equation is*

$$y = Ae^{-x} + Be^{11x}$$

for arbitrary constants A *and* B*. The particular solution which satisfies the given initial conditions is*

$$y = e^{-x}$$

Now suppose that the initial conditions are replaced by

$$y(0) = 1 + \delta, \qquad y'(0) = -1 + \epsilon$$

for some small numbers δ *and* ϵ*. The particular solution satisfying these conditions is*

$$y = \left(1 + \frac{11\delta}{12} - \frac{\epsilon}{12}\right) e^{-x} + \left(\frac{\delta}{12} + \frac{\epsilon}{12}\right) e^{11x}$$

and the change in the solution is therefore

$$\left(\frac{11\delta}{12} - \frac{\epsilon}{12}\right) e^{-x} + \left(\frac{\delta}{12} + \frac{\epsilon}{12}\right) e^{11x}$$

The term $\dfrac{(\delta + \epsilon)e^{11x}}{12}$ *is large compared with* e^{-x} *for* $x > 0$*, indicating that this problem is ill-conditioned.*
To inherent stability depends on the size of the solution to the original problem as well as on the size of any changes in the data. Under these circumstances, one would say that the problem is ill-conditioned.

We now consider a different type of instability which is a consequence of the method of solution rather than the problem itself.

Lecture 3

Definition 3. *A method is said to suffer from* ***induced instability*** *if small errors present at one stage of the method lead to bad effect in subsequent stages to such final results are totally inaccurate.*

Nearly all numerical methods involve a repetitive sequence of calculations and so it is inevitable that small individual rounding errors accumulate as they proceed. However, the actual growth of these errors can occur in different ways. If, after n steps of the method, the total rounding error is approximately $C\,n\,\epsilon$, where C is a positive constant and ϵ is the size of a typical rounding error, then the growth in rounding errors is usually acceptable. For example, if $C = 1$ and $\epsilon = 10^{-11}$, it takes about 50000 steps before the sixth decimal place is affected. On the other hand, if the total rounding error is approximately $Ca^n\epsilon$ or $Cn!\,\epsilon$, for some number $a > 1$, then the growth in rounding errors is usually unacceptable. For example, in the first case, if $C = 1$, $\epsilon = 10^{-11}$ and $a = 10$, it only takes about five steps before the sixth decimal place is affected. The second case is illustrated by the following example.

Example 0.5. *Many successful algorithms are available for calculating individual real roots of polynomial equations of the form*

$$p_n(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0 = 0$$

Some of these are described later. An attractive idea would be to use these methods to estimate one of the real roots, α say, and then to divide $P_n(x)$ by $x - \alpha$ to produce a polynomial of degree $n - 1$ which contains the remaining roots. This process can then be repeated until all of the roots have been

located. This is usually referred to as the ***method of deflation****. If* α *were an exact root of* $P_n(x) = 0$*, then the remaining* $n-1$ *roots would, of course, be the zeros of the deflated polynomial of degree* $n-1$*. However, in practice* α *might only be an approximate root and in this case the zeros of the deflated polynomial can be very different from those of* $P_n(x)$*. For example, consider the cubic*

$$p_3(x) = x^3 - 13x^2 + 32x - 20 = (x-1)(x-2)(x-10)$$

and suppose that an estimate of its largest zero is taken as 10.1*. If we divide* $p_3(x)$ *by* $x-10.1$*, the quotient is* $x^2-2.9x+2.71$ *which has zeros* $1.45 \pm 0.78i$*. Clearly an error of* 0.1 *in the largest zero of* $p_3(x)$ *has induced a large error into the calculation of the remaining zeros.*
It is interesting to note that if we divide $p_3(x)$ *by* $x - 1.1$*, the corresponding quadratic has zeros* 1.9 *and* 10.0 *which are perfectly acceptable. The deflation process can be applied successfully provided that certain precautions are taken. In particular, the roots should be eliminated in increasing order of magnitude.*

Of the two types of instability discussed, that of inherent instability is the most serious. Induced instability is a fault of the method and can be avoided either by modifying the existing method, as we did for some examples given in this section, or by using a completely different solution procedure. Inherent instability, however, is a fault of the problem so there is relatively little that we can do about it. The extent to which this property is potentially disastrous depends not only on the degree of ill-conditioning involved but also on the context from which the problem is taken.

Lecture 4

chapter 2
Solutions of Equations in One Variable

One of the fundamental problems of mathematics is that of solving equations of the form

$$f(x) = 0 \tag{4}$$

where f is a real valued function of a real variable x. Any number α satisfying equation (4) is called a **root** of the equation or a zero of f.
Most equations arising in practice are non-linear and are rarely of a form which allows the roots to be determined exactly. Consequently, numerical techniques must be used to find them.
Graphically, a solution, or a root, of Equation (4) refers to the point of intersection of $f(x)$ and the $x-$axis. Therefore, depending on the nature of the curve of $f(x)$ in relation to the $x-$axis, Equation (4) may have a unique solution, multiple solutions, or no solution. A root of an equation can sometimes be determined analytically resulting in an exact solution. For instance, the equation $e^{2x} - 3 = 0$ can be solved analytically to obtain a unique solution $x = \frac{1}{2}\ln 3$. In most situations, however, this is not possible and the root(s) must be found using a numerical procedure.

Bisection Technique

This technique based on the Intermediate Value Theorem. Suppose f is a continuous function defined on the interval $[a, b]$, with $f(a)$ and $f(b)$ of opposite sign. The Intermediate Value Theorem implies that a number p exists in (a, b) with

$f(p) = 0$. The method calls for a repeated halving of subintervals of $[a, b]$ and, at each step, locating the half containing p. To begin, set $a_1 = a$ and $b_1 = b$, and let p_1 be the midpoint of $[a, b]$; that is,

$$p_1 = a_1 + \frac{b_1 - a_1}{2} = \frac{a_1 + b_1}{2}$$

1. If $f(p_1) = 0$, then $p = p_1$, and we are done.
2. If $f(p_1) \neq 0$, then $f(p_1)$ has the same sign as either $f(a_1)$ or $f(b_1)$.
   - If $f(p_1)$ and $f(a_1)$ have the same sign, $p \in (p_1, b_1)$. Set $a_2 = p_1$ and $b_2 = b_1$.
   - If $f(p_1)$ and $f(a_1)$ have opposite signs, $p \in (a_1, p_1)$. Set $a_2 = a_1$ and $b_2 = p_1$.

Then reapply the process to the interval $[a_2, b_2]$. See Figure 2.

We can select a tolerance $\epsilon > 0$ and generate $p_1, p_2, \cdots, p_N$ until one of the following conditions is met:

- $|p_N - p_{N-1}| < \epsilon$,
- $\frac{|p_N - p_{N-1}|}{|p_N|} < \epsilon, \quad p_N \neq 0, \quad$ or
- $f(p_N) < \epsilon$,

When using a computer to generate approximations, it is good practice to set an upper bound on the number of iterations. This eliminates the possibility of entering an infinite loop, a situation that can arise when the sequence diverges (and also when the program is incorrectly coded).

Example 0.6. *The function* $f(x) = x^3 + 4x^2 - 10$ *has a root in* $[1, 2]$, *because* $f(1) = -5$ *and and* $f(2) = 14$ *the Intermediate*

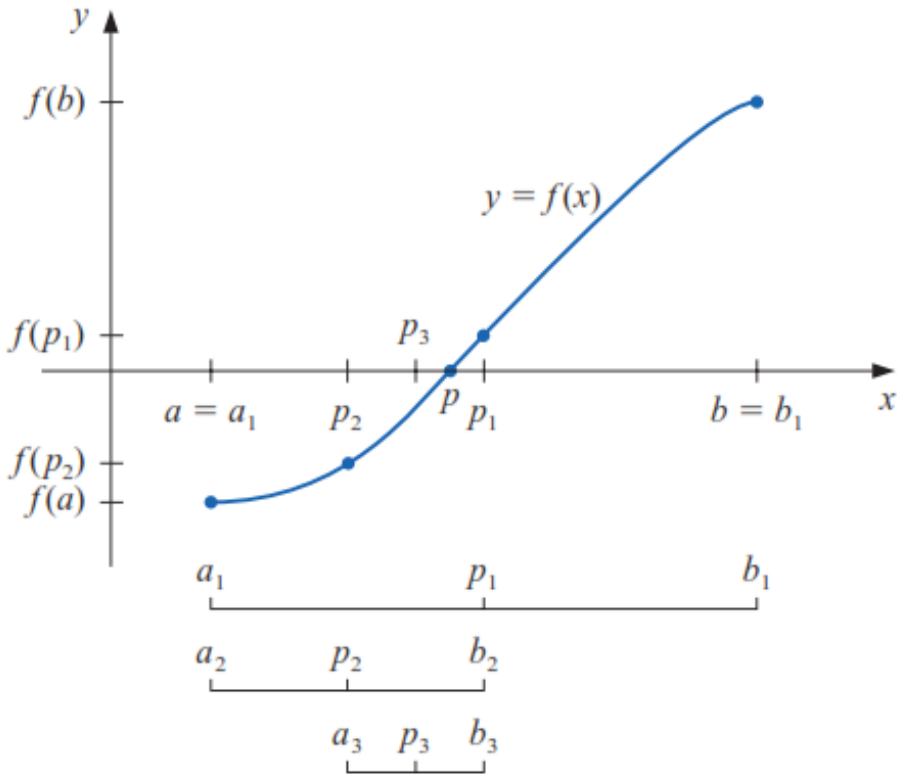

Figure 2: Produces of Bisection Technique

Value Theorem ensures that this continuous function has a root in $[1, 2]$.
Using Bisection method with the Matlab code to determine an approximation to the root.

Example 0.7. *The function* $f(x) = (x+1)^2 e^{(x^2-2)} - 1$ *has a root in* $[0, 1]$ *because* $f(0) < 0$ *and* $f(1) > 0$. *Use Bisection method to find the approximate root with* $\epsilon = 0.00001$.

MaTlab built-In Function fzero

The fzero function in MATLAB finds the roots of $f(x) = 0$ for a real function $f(x)$. FZERO Scalar nonlinear zero finding.

$X = FZERO(FUN, X_0)$ tries to find a zero of the function FUN near X_0, if X_0 is a scalar.
For example 0.6 use the following Matlab code:

```
clc
clear
fun = @(x) x.^3+4*x.^2-10; % function
x0 = 1; % initial point
x = fzero(fun,x0)
```

the resulte is:
$x = 1.365230013414097$

Theorem 0.8. *Suppose that $f \in C[a,b]$ and $f(a)f(b) < 0$. The Bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approximating a zero p of f with*

$$|p_n - p| < \frac{b-a}{2^n}, \qquad n \geq 1$$

Proof. For each $n \geq 1$, we have

$$b_1 - a_1 = \frac{1}{2}(b-a), \quad and \quad p_1 \in (a_1, b_1)$$

$$b_2 - a_2 = \frac{1}{2}\left[\frac{1}{2}(b-a)\right] = \frac{1}{2^2}(b-a), \quad and \quad p_2 \in (a_2, b_2)$$

$$b_3 - a_3 = \frac{1}{2}(b_2 - a_2) = \frac{1}{2^3}(b-a), \quad and \quad p_3 \in (a_3, b_3)$$

and so for the n step we can get

$$b_n - a_n = \frac{1}{2^n}(b-a), \quad and \quad p_n \in (a_n, b_n)$$

Since $p_n \in (a_n, b_n)$ and $|(a_n, b_n)| = b_n - a_n$ for all $n \geq 1$, it follows that

$$|p_n - p| < b_n - a_n = \frac{b-a}{2^n}$$

the sequence $\{p_n\}_{n=1}^{\infty}$ converges to p with rate of convergence of order $\frac{1}{2^n}$; that is

$$p_n = p + O\left(\frac{1}{2^n}\right)$$

□

It is important to realize that Theorem 0.8 gives only a bound for approximation error and that this bound might be quite conservative. For example, this bound applied to the problem in Example 0.6 ensures only that

$$|p - p_9| < \frac{2-1}{2^9} = 0.001953125 \approx 2 \times 10^{-3}$$

but the actual error is much smaller:

$$\begin{aligned} |p - p_9| &\leq |1.365230013414097 - 1.365234375| \\ &\approx -0.000004361585903 \\ &\approx 4.4 \times 10^{-6} \end{aligned}$$

Example 0.9. *Determine the number of iterations necessary to solve* $f(x) = x^3 + 4x^2 - 10 = 0$ *with accuracy* 10^{-3} *using* $a_1 = 1$ *and* $b_1 = 2$*.*

Solution: *We we will use logarithms to find an integer* N *that satisfies*

$$\begin{aligned} |p - p_n| &< 2^{-N}(b_1 - a_1) \\ &= 2^{-N}(2-1) \\ &= 2^{-N} < 10^{-3} \end{aligned}$$

One can use logarithms to any base, but we will use base-10 *logarithms because the tolerance is given as a power of* 10*. Since* $2^{-N} < 10^{-3}$ *implies that* $\log_{10} 2^{-N} < \log_{10} 10^{-3} = -3$*, we have*

$$-N \log_{10} 2 < -3 \qquad \textit{and} \qquad N > \frac{3}{\log_{10} 2} \approx 9.96$$

Hence, 10 iterations will ensure an approximation accurate to within 10^{-3}.

Lecture 5

Fixed-Point Iteration

A fixed point for a function is a number at which the value of the function does not change when the function is applied.

Definition 4. *The number p is a fixed point for a given function g if $g(p) = p$.*

Suppose that the equation $f(x) = 0$ can be rearranged as

$$x = g(x) \tag{5}$$

Any solution of this equation is called a fixed point of g. An obvious iteration to try for the calculation of fixed points is

$$x_{n+1} = g(x_n) \qquad n = 0, 1, 2, \cdots \tag{6}$$

The value of x_0 is chosen arbitrarily and the hope is that the sequence $x_0, x_1, x_2, \cdots$ converges to a number α which will automatically satisfy equation (5).
Moreover, since equation (5) is a rearrangement of (4), α is guaranteed to be a zero of f.
In general, there are many different ways of rearranging $f(x) = 0$ in the form (5). However, only some of these are likely to give rise to successful iterations, as the following example demonstrates.

Example 0.10. *Consider the quadratic equation*

$$x^2 - 2x - 8 = 0$$

with roots -2 and 4. Three possible rearrangements of this equation are

(a) $x_{n+1} = \sqrt{2x_n + 8}$

(b) $x_{n+1} = \frac{2x_n+8}{x}$

(c) $x_{n+1} = \frac{x_n^2-8}{2}$

Numerical results for the corresponding iterations, starting with $x_0 = 5$*, are given in Matlab code 0.16 with the Table.*

Solution:

```
 k        Xa           Xb          Xc
---  ----------   ----------   --------
 1   4.24264069   3.60000000   8.50000000
 2   4.06020706   4.22222222   32.12500000
 3   4.01502355   3.89473684   512.0078125
 4   4.00375413   4.05405405   131072.0000
 5   4.00093842   3.97333333   8589934592.0
 6   4.00023460   4.01342282   3.6893e+19
```

Consider that the sequence converges for (a) and (b), but diverges for (c).
This example highlights the need for a mathematical analysis of the method. Sufficient conditions for the convergence of the fixed point iteration are given in the following (without proof) theorem.

Theorem 0.11. *If* g' *exists on an interval* $I = [\alpha - A, \alpha + A]$ *containing the starting value* x_0 *and fixed point* α*, then* x_n *converges to* α *provided*

$$|g'(x)| < 1 \quad on \quad I$$

We can now explain the results of Example 0.10

(a) If $g(x) = (2x + 8)^{\frac{1}{2}}$ then $g'(x) = (2x + 8)^{-1/2}$ Theorem 0.11 guarantees convergence to the positive root $\alpha = 4$, because $|g'(x)| < 1$ on the interval $I = [3, 5] = [\alpha - 1, \alpha + 1]$ containing the starting value $x_0 = 5$. which is in agreement with the results of column Xa in the Table.

(b) If $g(x) = \frac{(2x+8)}{x}$ then $g'(x) = \frac{-8}{x^2}$ Theorem 0.11 guarantees convergence to the positive root $\alpha = 4$, because $|g'(x)| < 1$ as (a), which is in agreement with the results of column Xb in the Table.

(c) If $g(x) = \frac{(x^2-8)}{2}$ then $g'(x) = x$ Theorem 0.11 cannot be used to guarantee convergence, which is in agreement with the results of column Xc in the Table.

Example 0.12. *Find the approximate solution for the equation*

$$f(x) = x^4 - x - 10 = 0$$

by fixed point iteration method starting with $x_0 = 1.5$ *with* $|x_n - x_{n-1}| < 0.009$

Solution

The function $f(x)$ has a root in the interval $(1, 2)$, **Why ?**, rearrange the equation as

$$x_{n+1} = g(x_n) = \sqrt{x_n + 10}$$

then

$$g'(x) = \frac{(x+10)^{\frac{-3}{4}}}{4}$$

Achieving the condition

$$|g'(x)| \leq 0.04139 \qquad on \quad (1, 2)$$

then we get the solution sequence $\{1.5, 1.8415, 1.85503, 1.8556, \cdots\}$. consider that $|1.85503 - 1.8556| = 0.00057 < 0.009$.

Lecture 6

Newton-Raphson method

Newton-Raphson method is one of the most popular techniques for finding roots of non-linear equations.

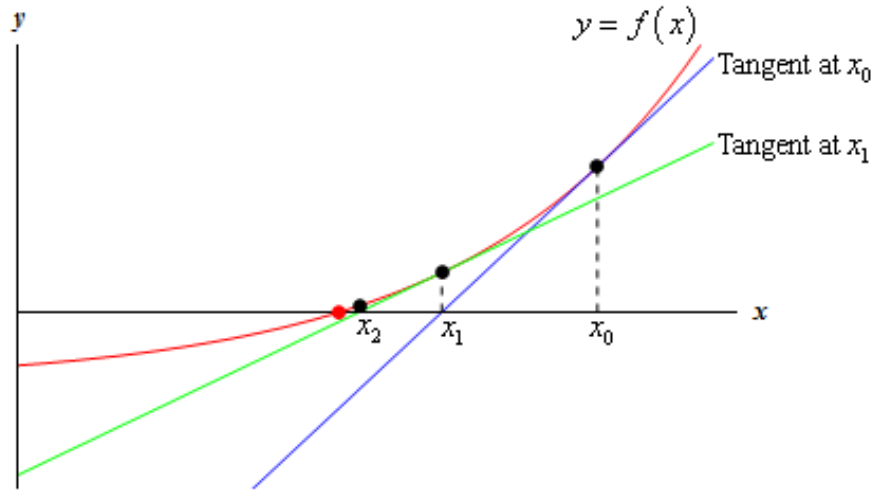

Figure 3: sketch of the Newton Raphson method

Newton-Raphson Formula:

Now Suppose that x_0 is a known approximation to a root of the function $y = f(x)$, as shown in Fig. 3.
The next approximation, x_2 is taken to be the point where tangent graph of $y = f(x)$ at $x = x_0$ intersects the $x-$axis.
From Taylor series we have

$$f(x_1) = f(x_0) + f'(x_0)(x_1 - x_0) + f''(x_0)\frac{(x_1 - x_0)^2}{2!} + f'''(x_0)\frac{(x_1 - x_0)^3}{3!} + \cdots + f^{(n)}(a)\frac{(x_1 - x_0)^n}{n!} + \cdots$$

consider x_1 as a root and take only the first two terms as an approximation:

$$\begin{aligned} 0 &= f(x_0) + f'(x_0)(x_1 - x_0) \\ (x_1 - x_0) &= -\frac{f(x_0)}{f'(x_0)} \\ x_1 &= x_0 - \frac{f(x_0)}{f'(x_0)} \end{aligned}$$

So, we can find the new approximation x_1. Now we can repeat the whole process to find an even better approximation.

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

we will arrive at the following formula.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad n = 0, 1, 2, \cdots \tag{7}$$

Note that when $f'(x_n) = 0$ the calculation of x_{n+1} fails. This is because the tangent at x_n is horizontal.

Example 0.13. *Newton's method for calculating the zeros of*

$$f(x) = e^x - x - 2$$

is given by

$$\begin{aligned} x_{n+1} &= x_n - \frac{e^{x_n} - x_n - 2}{e^{x_n} - 1} \\ &= \frac{e^{x_n}(x_n - 1) + 2}{e^{x_n} - 1} \end{aligned}$$

The graph of f, sketched in Fig. 4, shows that it has two zeros. It is clear from this graph that x_n converges to the negative root if $x_0 < 0$ and to the positive root if $x_0 > 0$, and that it breaks down if $x_0 = 0$. The results obtained with $x_0 = -10$ and $x_0 = 10$ are listed in next table.

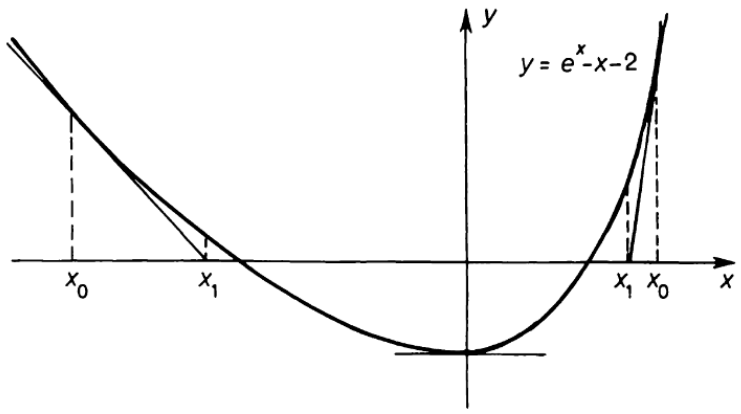

Figure 4: sketch of the Newton Raphson method for example 0.13

Sufficient conditions for the convergence of Newton's method are given in the following theorem.

Theorem 0.14. *If f'' is continuous on an interval $[\alpha - A, \alpha + A]$, then x_n converges to α provided $f'(\alpha) \neq 0$ and x_0 is sufficiently close to α.*

Proof. Comparison of equation

$$x_{n+1} = g(x_n) \qquad n = 0, 1, 2, \cdots$$

and the equation

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

shows that Newton's method is a fixed point iteration with

$$g(x) = x - \frac{f(x)}{f'(x)}$$

By the quotient rule,

$$g'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{(f'(x))^2} = \frac{f(x)f''(x)}{(f'(x))^2}$$

let $x = \alpha$ then

$$g'(\alpha) = \frac{f(\alpha)f''(\alpha)}{(f'(\alpha))^2}$$

This implies that $g'(\alpha) = 0$, because $f(\alpha) = 0$ and $f'(\alpha) \neq 0$. Hence by the continuity of f'', there exists an interval $I = [\alpha-\delta, \alpha+\delta]$, for some $\delta > 0$, on which $|g'(x)| < 1$. Theorem 0.11 then guarantees convergence provided $x_0 \in I$, i.e. provided x_0 is sufficiently close to α. □

0.2 EXERCISE

1. Use the Bisection method and fixed point method to find p_3 for $f(x) = \sqrt{x} - \cos x$ on $[0, 1]$.

2. Let $f(x) = 3(x+1)(x-\frac{1}{2})(x-1)$ Use the Bisection method and fixed point method on the intervals $[-2, 1.5]$ and $[-1.25, 2.5]$ to find p_3.

3. Use the Bisection method on the solutions accurate to within 10^{-2} for $f(x) = x^3 - 7x^2 + 14x - 6 = 0$ on each intervals: $[0, 1]$, $[1, 3.2]$ and $[3.2, 4]$.

4. Find an approximation to $\sqrt{3}$ correct to within 10^{-4} using the Bisection Algorithm. Hint: Consider $f(x) = x^2 - 3$.

5. Use an appropriate fixed point iteration to find the root of

   (a) $x - \cos x = 0$

   (b) $x^2 + \ln x = 0$

   starting in each case with $x_0 = 1$. Stop when $|x_{n+1} - x_n| < 0.5 \times 10^{-2}$.

6. Find the first nine terms of the sequence generated by $x_{n+1} = e^{-x_n}$ starting with $x_0 = 1$.

7. Use Newton's method to find the roots of

(a) $x - \cos x = 0$

(b) $x^2 + \ln x = 0$

starting in each case with $x_0 = 1$. Stop when $|x_{n+1} - x_n| < 10^{-6}$.

8. Find the roots of $x^2 - 3x - 7$ using Newton's method with $\epsilon = 10^{-4}$ or maximum 20 iterations.

Matlab Code 0.15. *Bisection method*

```
% ******************************************
% *************  bisection method  ************
% ****  to find a root of the function f(x) ***
% ******************************************
clc
clear
close all
f=@(x) x.^3+4*x.^2-10 ;
% f=@(x) (x+1)^2*exp(x^2-2)-1;
a=1;
b=2;
c=(a+b)/2;
e=0.00001;
k=1;
fprintf('     k       a           b           f(c)    \n');
fprintf('    ---  ---------   ---------   -------- \n');

while abs(f(c)) > e
c=(a+b)/2;
if f(c)*f(a)<0
     b=c;
   else
     a=c;
 end
fprintf('%6.f  %10.8f  %10.8f  %10.8f \n', k,a,b,f(c));
k=k+1;
end
fprintf('  The approximated root is c= %10.10f  \n', c);
```

The result as the following table:

```
     k        a           b           f(c)
    ---  ---------   ---------   --------
     1  1.00000000  1.50000000  2.37500000
     2  1.25000000  1.50000000  -1.79687500
     3  1.25000000  1.37500000  0.16210938
     4  1.31250000  1.37500000  -0.84838867
    ------------------------------------
    18  1.36522675  1.36523056  0.00000903
  The approximated root is c= 1.3652305603
```

Matlab Code 0.16. *Fixed Point Iteration*

```
1
2 clc
3 clear
4 close all
5
6 xa =5; % Initial value of root
7 xb =5;
8 xc =5;
9 fprintf('     k       Xa          Xb          Xc    \n');
10 fprintf('    ---  ---------   ---------   -------- \n');
11
12 for k=1:1:6
13  xa=sqrt(2*xa+8);
14  xb =(2*xb +8)/xb;
15  xc =(xc^2-8)/2;
16 fprintf('%6.f  %10.8f  %10.8f  %10.8f \n', k, xa , xb , xc);
17 end
```

The result as the following table:

```
     k        Xa          Xb          Xc
    ---  ---------   ---------   --------
     1  4.24264069  3.60000000  8.50000000
     2  4.06020706  4.22222222  32.12500000
     3  4.01502355  3.89473684  512.0078125
     4  4.00375413  4.05405405  131072.0000
     5  4.00093842  3.97333333  8589934592.0
     6  4.00023460  4.01342282  3.6893e+19
```

Matlab Code 0.17. *Newton Raphson method*

```
% ********  Newton Raphson method  ************
% ****  to find a root of the function f(x) ***
clc
clear
close all
f=@(x) exp(x)-x-2 ; % the function f(x)
fp=@(x) exp(x)-1 ; % the derivative f'(x) of f(x)
xa=-10; % Initial value of first root
xb=10;  % Initial value of second root
r = 'failure';
fprintf('     k          Xa             Xb  \n');
fprintf('    ---    ---------     ---------  \n');
fprintf('%6.f    %10.8f    %10.8f \n', 0, xa , xb );
for k=1:1:14
    if fp(xa)==0; r
    return
    elseif fp(xb)==0; r
    return
    end
 xa=xa-f(xa)/fp(xa);
 xb=xb-f(xb)/fp(xb);
 fprintf('%6.f    %10.8f    %10.8f \n', k, xa , xb );
end
```

The result as the following table:

```
     k          Xa             Xb
    ---    ---------     ---------
     1    -1.99959138    9.00049942
     2    -1.84347236    8.00173312
     3    -1.84140606    7.00474864
     --    ----------    ----------
    13    -1.84140566    1.14619325
    14    -1.84140566    1.14619322
>>
```