# NUMERICAL ANALYSIS I

Lecture Notes

Dr. OMAR AL-TAMEEMI

Department of Mathematics and Computer Applications

omar.ismael@nahrainuniv.edu.iq

# Contents


I Error Analysis

1 Errors Analysis . . . . . 7

1.1 Introduction 7

1.2 Accuracy and Errors 9

1.2.1 Sources of Error . . . . . 9

1.2.2 Measurement of Errors . . . . . 10

1.2.3 Taylor Polynomials and their Error . . . . . 11

1.2.4 Some Causes of Serious Rounding Errors and Possible Remedies . . . . . 15

1.2.5 Conclusions . . . . . 16

1.3 Algorithms and Convergence 17

1.3.1 Algorithms . . . . . 17

1.3.2 Convergence . . . . . 18

II Solutions of Equations in One Variable

2 Solutions of Equations in One Variable . . . . . 21

2.1 Introduction 21

2.2 Bisection Method 22

2.2.1 Bisection method algorithm . . . . . 25

2.2.2 Stopping criteria . . . . . 26

**2.3 Newton's Method** **28**
2.3.1 Newton's Method Algorithm . . . . . 30
2.3.2 Disadvantage of Newton's method . . . . . 30

**III Interpolation and Polynomial Approximation**

**3 Interpolation . . . . . 33**
**3.1 Barycentric Interpolation (Modern Default) 33**
**3.2 Chebyshev Nodes (Avoiding Runge) 33**
**3.3 introduction 34**
**3.4 Lagrange Polynomial Interpolation 36**
3.4.1 Linear Interpolation . . . . . 38
3.4.2 Quadratic Interpolation . . . . . 41
3.4.3 Higher Order Interpolation . . . . . 44
**3.5 Newton's Form of Polynomial Interpolation 46**
3.5.1 Divided Differences Scheme . . . . . 49
3.5.2 Divided-Difference Algorithm . . . . . 56
3.5.3 Error Analysis for Polynomial Interpolation . . . . . 59

**IV Numerical Differentiation and Integration**

**4 Numerical Differentiation and Integration . . . . . 65**
**4.1 Adaptive Simpson (Practical Quadrature Default) 65**
**4.2 NUMERICAL DIFFERENTIATION 65**
4.2.1 Derivation Using Taylor Series Expansion . . . . . 66
4.2.2 Derivation Using Lagrange Polynomial Interpolation . . . . . 68
4.2.3 Round-off Error . . . . . 72
**4.3 Richardson Extrapolation Method 73**
**4.4 Elements of Numerical Integration 76**
4.4.1 Newton-Cotes Formulas . . . . . 83
**4.5 Composite Numerical Integration 86**
**4.6 Romberg Integration 93**

# I Error Analysis

1 Errors Analysis ..... 7

1.1 Introduction

1.2 Accuracy and Errors

1.3 Algorithms and Convergence

# 1. Errors Analysis

## 1.1 Introduction

**Three sources of error:** *modeling error*, *conditioning* (sensitivity of the true problem), and *algorithmic stability*. Keep them conceptually separate.

### Floating-Point Essentials

Let $\mathrm{fl}(x)$ denote rounding to working precision with unit roundoff $u$. For basic operations,

$$\mathrm{fl}(x \circ y) = (x \circ y)(1+\delta), \qquad |\delta| \le u,\ \circ \in \{+,-,\times,/\}.$$

**Catastrophic cancellation:** avoid forms like $\dfrac{1-\cos x}{x^2}$ for small $x$. Use identities, e.g. $\dfrac{1-\cos x}{x^2} = \frac{1}{2}\left(\dfrac{\sin(x/2)}{x/2}\right)^2$.

### Stability (Backward View)

**Definition 1.1.1 — Backward stability.** An algorithm is backward stable if its output $\widehat{x}$ is the exact solution of a nearby problem with relative data perturbation $\lesssim u$.

[Quadratic formula (stable form)] To compute roots of $ax^2+bx+c=0$, use

$$x_1 = \frac{-b-\text{sgn}(b)\sqrt{b^2-4ac}}{2a}, \qquad x_2 = \frac{2c}{-b-\text{sgn}(b)\sqrt{b^2-4ac}},$$

to avoid subtractive cancellation when $|b| \gg |\sqrt{b^2-4ac}|$.

Why do we need numerical methods and numerical analysis? "real life" problems lead to mathematical models which do not have analytic solutions, i.e. a formula for the solution in terms of simple functions, for example:

1. In a calculation concerned with finding the magnetic field near a recording head in a tape deck the following integral arises:

$$\int_0^\infty \frac{k\sin(k)\cos(k)\sin(\alpha k)\cos(\alpha k)\coth(\beta k)}{[k^2-(m\pi)^2]\left[k^2-\left\{\left(n-\frac{1}{2}\right)\pi\right\}^2\right]}dk$$

   $\alpha, \beta$ - constants; m, n - integers.

2. If a loan of £$S$ is to be paid off in $N$ monthly payments of £$p$, the true annual rate of interest, known as the annual percentage rate (APR), $a$, is given by the solution of:

$$p = \frac{S\left((1+a)^{1/12}-1\right)}{1-(1+a)^{-N/12}}$$

3. Calculations involved in determining the strength of dams, the stability of bridges, tall buildings, etc., air flow around aircraft, trains, cars, etc., tomorrow's weather, etc. usually reduce to solving systems of simultaneous linear equations. Tens or even hundreds of thousands of such equations and unknown variables are quite normal.

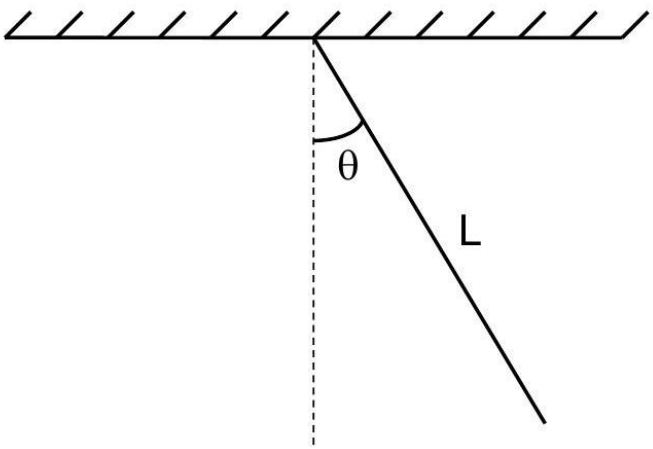
4. Keeping spacecraft on course, predicting tidal motion, the problems mentioned in (3) above, etc., all involve solving complex systems of ordinary and partial differential equations.

Even "simple" problems do not have analytic solutions, as the following examples show.

1. Evaluate (a) $\int_0^t e^{-x^2} dx$ (b) $\int_0^1 x \tan x dx$
2. Find $x$ such that $e^x - \sin x = 0$
3. Solve the pendulum equation,

$$\frac{d^2\theta}{dt^{2^2}} - \frac{g}{L}\sin\theta = 0$$

when $\theta$ is not assumed to be small and so we cannot approximate $\sin\theta$ by $\theta$.




4. How does your calculator or a computer evaluate functions such as $e^x, \sin x, \tan^{-1} x, \sqrt{x}$, etc.?

## 1.2 Accuracy and Errors

### 1.2.1 Sources of Error

1. Human error:
   - programming errors.
   - try to test the procedure using input data with known results.
2. Idealised model:
   - simplifying assumptions used to obtain a mathematical model of a real problem, e.g. ignoring air resistance in a model of a falling stone.

- care needed when comparing computed results with real observations.

3. Measurement error:
   - data used as input to a program may only be measured to a few significant figures.
   - see how sensitive your method/program is to uncertainties in the input data.
4. Truncation error:
   - the result of replacing an infinite process by a finite one,

     e.g.1. computing an approximation to $e^x$ from the first few terms of its Taylor series

     $$\underbrace{e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!}}_{\text{approximation}} \Big| \underbrace{+\frac{x^4}{4!} + \frac{x^5}{5!} + \cdots}_{\text{truncation error}}$$

     e.g.2. solving $f(x) = 0$ using the Newton-Raphson process and stopping when successive estimates are "close".
5. Round-off error:
   - the effect of using finite instead of infinite precision in computers to store real numbers.

     e.g. $1/3$ cannot be stored exactly as a decimal.

### 1.2.2 Measurement of Errors

Let $\hat{x}$ be an approximation to the exact value $x$. We may define the following measures of the error:

- **absolute error**$= |x - \hat{x}|$ which is a measure of the number of correct decimal places.
- **relative error**$= \frac{|x-\hat{x}|}{|x|}$ which is a measure of the number of correct significant figures (if $x \neq 0$)
- **percentage error** $= \frac{|x-\hat{x}|}{|x|} \times 100$ (if $x \neq 0$)

Usually relative error is a more meaningful measure of error. Procedures requiring some sort of stopping criterion, such as the Newton-Raphson procedure, should normally

include relative error tests.

■ **Example 1.1** Evaluate the absolute, relative and percentage errors in each of the following cases:

(i) $x = 1.2, \ \hat{x} = 1.21.$

(ii) $x = 12000, \ \hat{x} = 12100.$

(iii) $x = 0.00120, \ \hat{x} = 0.00121.$

| | Absolute error | Relative error | Percentage error |
|---|---|---|---|
| (i) | | | |
| (ii) | | | |
| (iii) | | | |

■

### 1.2.3 Taylor Polynomials and their Error

In the analysis of numerical methods, Taylor polynomials and their associated error term are frequently used.

For a function f(x) which satisfies certain continuity and differentiability conditions, it is possible to write

$$f(x) = f(0) + xf'(0) + \frac{x^2}{2!}f''(0) + \cdots + \frac{x^n}{n!}f^{(n)}(0) + \cdots$$

Note that this is a Taylor series expansion about $x = 0$, which is also called a Maclaurin expansion. To get a Taylor polynomial of degree $n$, just truncate the infinite series after $(n+1)$ terms.

■ **Example 1.2** Write down the first 5 terms in the Taylor series expansion for $f(x) = e^x$, expanded about x = 0. ■

■ **Example 1.3** For $f(x) = e^x$, find the linear and quadratic Taylor polynomials expanded about $x = 0$ and sketch the function and the polynomials for $-1 \leq x \leq 1$. ■

In order to use a Taylor polynomial in a practical way we need to know its error. Taylor's theorem tells us that, under certain conditions, if we take the polynomial

$$p_n(x) = f(0) + xf'(0) + \frac{x^2}{2!}f''(0) + \cdots + \frac{x^n}{n!}f^{(n)}(0)$$

as an approximation to $f(x)$, the infinite number of error terms

$$\frac{x^{n+1}}{(n+1)!}f^{(n+1)}(0) + \frac{x^{n+2}}{(n+2)!}f^{(n+2)}(0) + \cdots$$

may be written as the single term,

$$\frac{x^{n+1}}{(n+1)!}f^{(n+1)}(c)$$

where c is some (unknown) point between 0 and x.

■ **Example 1.4** If $e^x$ is approximated by its $3^{rd}$ order Taylor polynomial, find an upper bound for the error if $x$ lies in the interval $[-1,1]$. ■

The error term may also be used to decide in advance how many terms in a polynomial approximation are needed to ensure a certain accuracy.

■ **Example 1.5** If $e^x$ is to be approximated by the first $(n+1)$ terms of its Taylor series, what should n be such that the error for $-1 \leq x \leq 1$ is less than $10^{-5}$ ? ■

■ **Example 1.6** Write down the first 4 non-zero terms in the Taylor series expansion for $\sin(x)$, expanded about x $= 0$. ■

Now write down an expression for a general $n^{\text{th}}$ term in the series that you produce and also an expression for the truncation error, if this series is approximated after n terms. Taylor series may be found by expanding about points other than $x = 0$. More generally Taylor's theorem is:

$$f(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{(x - x_0)^2}{2!} f''(x_0) + \cdots$$
$$\cdots + \frac{(x - x_0)^n}{n!} f^{(n)}(x_0) + \frac{(x - x_0)^{n+1}}{(n+1)!} f^{(n+1)}(c)$$

where c is between $x_0$ and x.

Setting $x_0 = 0$ gives the previous result.

A useful form for future work is:

$$y(x+h) = y(x) + hy'(x) + \frac{h^2}{2!}y''(x) + \cdots + \frac{h^n}{n!}y^{(n)}(x) + \frac{h^{n+1}}{(n+1)!}y^{(n+1)}(c)$$

where $c$ is between $x$ and $x+h$ (in the expansion of $f(x)$ above, replace $f(x)$ by $y(x)$, $x_0$ by $x$ and $x$ by $x+h$, then $x-x_0$ becomes $h$ ).

### 1.2.4 Some Causes of Serious Rounding Errors and Possible Remedies

Rounding errors are unavoidable in computer calculations with real numbers. Unfortunately their effects can sometimes be very serious. Here we shall look at two particular situations where rounding errors occur and suggest how their effect can be reduced.

Note that if you express a real number in normalised floating point form $\pm m \times 10^{\mathrm{e}}$ then m is called the mantissa and e is the exponent.

▪ **Example 1.7** Consider a computer which stores the mantissa of a real number using 4 decimal digits. What is the result of the sum $0.1 + 5000$ using this machine? First the numbers are stored (exactly) in normalised floating point form to give

$$0.1000 \times 10^0 + 0.5000 \times 10^4$$

Then, in order to perform the addition, the exponents are made the same to give

$$\begin{array}{r} 0.00001 \times 10^4 \\ +0.50000 \times 10^4 \\ \hline 0.50001 \times 10^4 \end{array}$$

Finally the result is stored using 4 decimal digits as $0.5000 \times 10^4 = 5000$ ▪

In the above example the result should of course have been 5000.1 but adding a large number to a small number has resulted in a loss of significant figures. The number 0.1 has been lost entirely.

■ **Example 1.8** The quadratic formulas for computing the roots of $ax^2+bx+c=0$, when $a \neq 0$, are

$$x_1 = \frac{-b+\sqrt{b^2-4ac}}{2a} \qquad \text{and} \qquad x_2 = \frac{-b-\sqrt{b^2-4ac}}{2a}.$$

Consider the quadratic equation $x^2+62.10x+1=0$ and discuss the numerical results.

■

### 1.2.5 Conclusions

1. Avoid calculations in your programs which may lead to loss of significant figures by reformulating the problem.
2. We can use a higher level of precision (i.e. more significant figures) in all of the calculations, but program running times will then be longer.

## 1.3 Algorithms and Convergence

### 1.3.1 Algorithms

**Definition 1.3.1 — Algorithm.** An algorithm is a procedure that describes a finite sequence of steps to be performed in a specified order.

■ **Example 1.9** Give an algorithm to compute $\sum_{i=1}^{n} x_i$, where $n$ and $x_1, x_2, \ldots, x_n$ are given.

■

**Definition 1.3.2 — Stable.** An algorithm is called stable if small changes in the initial data of the algorithm produce correspondingly small changes in the final results.

**Definition 1.3.3 — Unstable.** An algorithm is unstable if small errors made at one stage of the algorithm are magnified and propagated in subsequent stages and seriously degrade the accuracy of the overall calculation.

■ **Example 1.10** Consider the following recurrence algorithm

$$\begin{cases} x_0 = 1, & x_1 = \frac{1}{3} \\ x_{n+1} = \frac{13}{3}x_n - \frac{4}{3}x_{n-1} \end{cases}$$

for computing the sequence of $\{x_n = (\frac{1}{3})^n\}$. This algorithm is **unstable**. Why? ■

### 1.3.2 Convergence

**Definition 1.3.4** Suppose $\{\beta_n\} \to 0$ and $\{x_n\} \to x^*$. If $\exists\ c > 0$ and an integer $N > 0$ such that

$$|x_n - x^*| \leq c|\beta_n|, \quad \forall\ n \geq N,$$

then we say $\{x_n\}$ converges to $x^*$ with rate of convergence $O(\beta_n)$, and write $x_n = x^* + O(\beta_n)$.

■ **Example 1.11** Compare the convergence behavior of $\{x_n\}$ and $\{y_n\}$, where

$$x_n = \frac{n+1}{n^2}, \quad \text{and} \quad y_n = \frac{n+3}{n^3}.$$

■

# II Solutions of Equations in One Variable


2 Solutions of Equations in One Variable ..... 21

2.1 Introduction

2.2 Bisection Method

2.3 Newton's Method

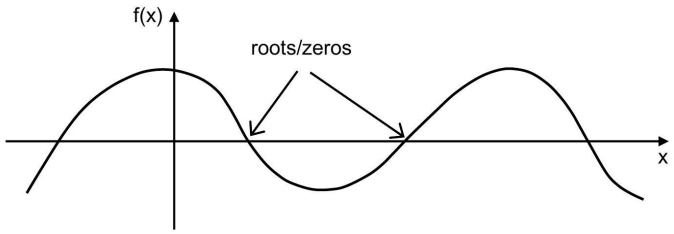
# 2. Solutions of Equations in One Variable

## 2.1 Introduction

**Unified stopping rule (use throughout):** stop when both

$$\frac{|x_k - x_{k-1}|}{\max(1, |x_k|)} \leq \text{tol} \quad \text{and} \quad |f(x_k)| \leq \text{tol}_{\text{res}}.$$

Given a function $f(x)$ we wish to find values of $x$ for which $f(x) = 0$. Such points are known as the zeros of $f(x)$ or the roots of $f(x) = 0$.




Only for simple functions $f(x)$ may this be done analytically.

■ **Example 2.1** Find the maximum and minimum values of $y = e^x + \cos x$.

This is apparently an easy problem. We differentiate $y$ and set the derivative equal to zero, $e^x - \sin x = 0$. But how do we solve this equation? ■

■ **Example 2.2 — Population Modelling.** Let N(t) be the size of a population at time t which is growing continuously at a rate which is proportional to the population size at that time. Let $\lambda$ be the constant growth rate.

(a) Ignoring other factors, this may be modelled by the differential equation

$$\frac{\mathrm{dN}}{\mathrm{dt}} = \lambda \mathrm{N}$$

with solution

$$N(t) = N_0 e^{\lambda t}$$

where $N_0$ is the population when $t = 0$.

(b) If the model is to take into account net migration at a constant rate in time of $v$, the appropriate differential equation is now

$$\frac{\mathrm{dN}}{\mathrm{dt}} = \lambda \mathrm{N} + \mathrm{v}$$

with solution

$$N(t) = N_0 e^{\lambda t} + \frac{v}{\lambda}\left(e^{\lambda t} - 1\right)$$

Suppose we know $\mathrm{N}_0$, $v$ and the population N at some time t in the future. How can we determine the birth rate $\lambda$ from this information? We need to solve a non-linear equation for $\lambda$. ■

## 2.2 Bisection Method

The first technique to find the root of a function, based on the Intermediate Value Theorem, is called the Bisection method. First of all, lets recall the **Intermediate Value Theorem**.

**Theorem 2.2.1 — Intermediate Value Theorem.** If $f \in C[a,b]$ and $K$ is in between $f(a)$ and $f(b)$, then $\exists c \in (a,b)$ s.t. $f(c) = K$.

In words, this just says that a continuous function on $[a,b]$ must attain all intermediate values between $f(a)$ and $f(b)$. We are frequently interested in the special case where $K = 0$, which would show that $f$ has a root on the interval $[a,b]$.

Most of us probably discovered the bisection algorithm to calculate square roots in elementary school. Suppose I give you a 4 function calculator and ask you to approximate $\sqrt{19}$ as accurately as possible. Since $4^2 = 16$ and $5^2 = 25$, you know $\sqrt{19} \in [4,5]$. You might consider $4.5^2 = 20.25$ next to deduce $\sqrt{19} \in [4,4.5]$. Then you might further bisect this interval, notice $4.25^2 = 18.0625$, and that $\sqrt{19} \in [4.25,4.5]$.

And then we proceed in this fashion until the interval becomes smaller and smaller after continual halving and we get an approximation to $\sqrt{19}$ to as many digits as we desire. But this is precisely the bisection algorithm for rootfinding applied to $f(x) = x^2 - 19$.

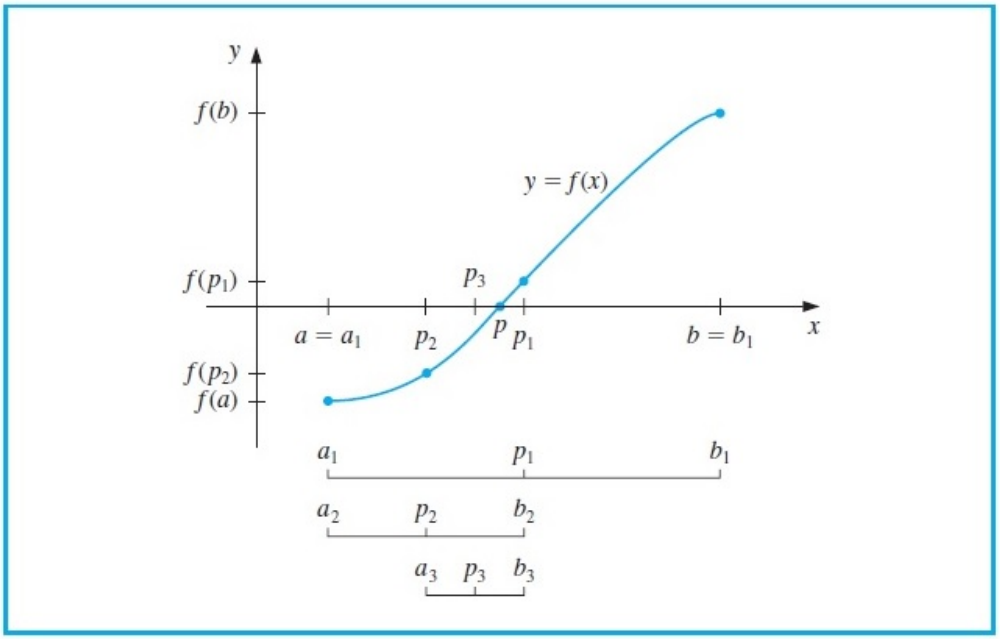



Figure 2.1: Bisection Method

The method proceeds as follows:

1. Choose the midpoint c = (a + b)/2 of [a, b].

2. Evaluate $f(c)$.
3. If $f(a)$ and $f(c)$ have opposite signs, the root lies in [a,c], while if $f(c)$ and $f(b)$ have opposite signs, the root lies in [c,b].
4. In either case the interval containing the root is halved. The process is now repeated on this new interval and continues until the root is found to the required accuracy. For a single root in the interval $[a,b]$ there are two possible situations.

■ **Example 2.3** Consider the problem of finding the root of $f(x) = x - e^{1/x}$ for $x > 0$. Use the Bisection method to find an approximate value of the root. Continuing the iterations in a table, and noting that only the sign of the function needs to be recorded.

| iteration | $a$ | $b$ | $c$ | $f(a)$ | $f(b)$ | $f(c)$ |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

■

### 2.2.1 Bisection method algorithm

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 $a,b$; stop criteria: tolerance $TOL$ and $\varepsilon$; maximum number of iterations $n$.

**OUTPUT** approximate solution $c$ or message of failure.

**Step 1** Set $i = 1$;

$$FA = f(a).$$

**Step 2** While $i \leq n$ do Steps 3-6.

**Step 3** Set $c = a + (b-a)/2$; (Compute $c_i$.)

$$FC = f(c).$$

**Step 4** If $FC \leq \varepsilon$ or $c \leq TOL$ then

**OUTPUT** $(c)$; (Procedure completed successfully.) STOP.

**Step 5** Set $i = i+1$.

**Step 6** If $FA \cdot FP > 0$ then set $a = c$; (Compute $a_i, b_i$.)

$$FA = FC$$

else set $b = c$.

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

### 2.2.2 Stopping criteria

Let $\{c_n\}$ be the sequence of numbers produced. The algorithm should stop if one of the following conditions is satisfied.

1. the iteration number $i > n$,
2. $|c_i - c_{i-1}| \leq TOL$, or
3. $|f(c_i)| < \varepsilon$.

**Theorem 2.2.2** 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:

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

**Proof**

For each $n \geq 1$, we have

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

Since $p_n = \frac{1}{2}(a_n + b_n) \forall n \geq 1$, it follows that:

$|p_n - p| \leq \frac{1}{2}(b_n - a_n) = \frac{b-a}{2^n}$

Since

$|p_n - p| \leq (b-a)\frac{1}{2^n}$ the sequence $\{p_n\}_{n=1}^{\infty}$ converges to $p$ with rate of convergence $O(\frac{1}{2^n})$; that is,

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

It is important to realize that the above Theorem gives only a bound for approximation

error and that this bound may be quite conservative.

■ **Example 2.4** In Example 2.3 we carried out 6 iterations of the Bisection method. Evaluate the maximum error after these iterations. ■

■ **Example 2.5** How many steps would be required in Example 2.3 to find the root with an error of less than $0.5 \times 10^{-4}$. ■

## 2.3 Newton's Method

Newton's (or the Newton–Raphson) method is one of the most powerful and well-known numerical methods for solving a root-finding problem. There are many ways to derive Newton's method. In this lectures We shall derive this method of solving $f(x) = 0$ in two ways:

I. **Graphically**

Taking $p_0$ as an approximation to the root $p$, a better approximation appears to be the point $p_1$, which is the point where the tangent to the curve $f(x)$ at $x = p_0$ cuts the x-axis.

Now,

$$f'(p_0) = \frac{f(p_0)}{p_0 - p_1}$$

giving

$$p_1 = p_0 - \frac{f(p_0)}{f'(p_0)}$$

We use an estimate of the root as a value of $p_0$ and then use this formula to successively generate $p_1, p_2, p_3, \ldots$ until the required accuracy is achieved.

II. **Using Taylor Series**

Expand $f(x)$ about the point $p_0$

$$f(x) = f(p_0) + (x - p_0) f'(p_0) + \dots$$

Assuming $p_0$ is close to $p$ we may omit the other terms in the expansion

$$\left[ \frac{(x - p_0)^2}{2!} f''(p_0) \text{ etc.} \right]$$

If we would like $p$ to be the root satisfying $f(x) = 0$, we take $x = p$ and then $f(x) = f(p) = 0$ and

$$0 \approx f(p_0) + (p - p_0) f'(p_0)$$

or on rearranging

$$p \approx p_0 - \frac{f(p_0)}{f'(p_0)}$$

is a better approximation to the root than $p_0$.

■ **Example 2.6** Find an approximation to the root of $f(x) = x - e^{1/x}$ using the Newton-Raphson method with a starting value $p_0 = 1.5$. ■

■ **Example 2.7** Use the Newton-Raphson method to derive an iterative scheme for computing $\sqrt{A}$ where $A$ is a positive real number. Use this scheme to compute $\sqrt{2}$. ■

### 2.3.1 Newton's Method Algorithm

To find a solution to $f(x) = 0$ given an initial approximation $p_0$ :

**INPUT** initial approximation $p_0$; tolerance $TOL$; maximum number of iterations $N_0$.

**OUTPUT** approximate solution $p$ or message of failure.

**Step 1** Set $i = 1$.

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

**Step 3** Set $p = p_0 - f(p_0)/f'(p_0)$. ( Compute $p_i$.)

**Step 4** If $|p - p_0| < TOL$ then, OUTPUT $(p)$; (The procedure was successful.) STOP.

**Step 5** Set $i = i + 1$.

**Step 6** Set $p_0 = p$. ( Update $p_0$.)

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

The stopping-technique inequalities given with the Bisection method are applicable to Newton's method. That is, select a tolerance $\varepsilon > 0$, and construct $p_1, \ldots p_N$, until

$$|p_N - p_{N-1}| < \varepsilon$$
$$\frac{|p_N - p_{N-1}|}{|p_N|} < \varepsilon, \quad p_N \neq 0$$

or

$$|f(p_N)| < \varepsilon.$$

### 2.3.2 Disadvantage of Newton's method

In many applications, the derivative $f'(x)$ is very expensive to compute, or the function $f(x)$ is not given in an algebraic formula so that $f'(x)$ is not available.

# III Interpolation and Polynomial Approximation


3 Interpolation ..... 33

3.1 Barycentric Interpolation (Modern Default)

3.2 Chebyshev Nodes (Avoiding Runge)

3.3 introduction

3.4 Lagrange Polynomial Interpolation

3.5 Newton's Form of Polynomial Interpolation

# 3. Interpolation

## 3.1 Barycentric Interpolation (Modern Default)

Given nodes $x_0, \dots, x_n$ and values $y_i$, the second barycentric form is

$$P(x) = \frac{\sum_{i=0}^{n} \frac{w_i y_i}{x - x_i}}{\sum_{i=0}^{n} \frac{w_i}{x - x_i}}, \qquad w_i = \frac{1}{\prod_{j \neq i}(x_i - x_j)}.$$

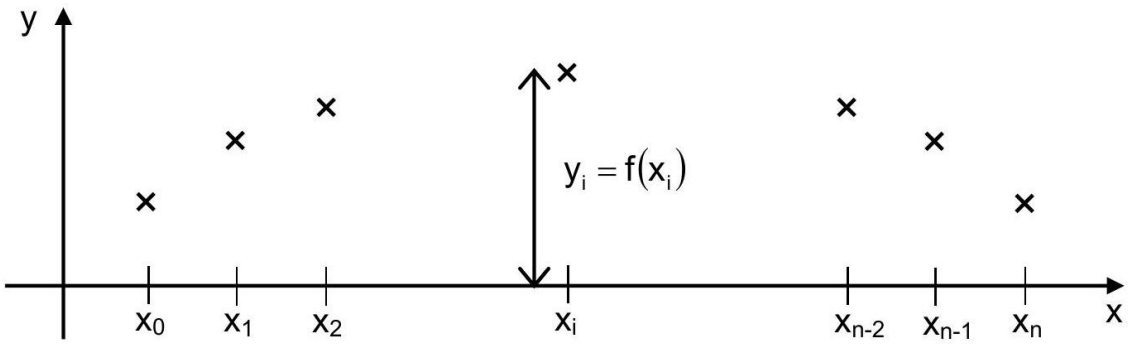
It evaluates $P(x)$ in $\mathcal{O}(n)$ and is numerically stable for well-chosen nodes.

## 3.2 Chebyshev Nodes (Avoiding Runge)

On $[-1,1]$, choose $x_k = \cos\left(\frac{2k+1}{2(n+1)}\pi\right)$, $k = 0, \dots, n$. Map general $[a,b]$ by an affine change of variables. These nodes minimize the interpolation error in the maximum norm up to a constant.

## 3.3 introduction

Interpolation is the problem of finding and evaluating a function which passes through a given set of points.

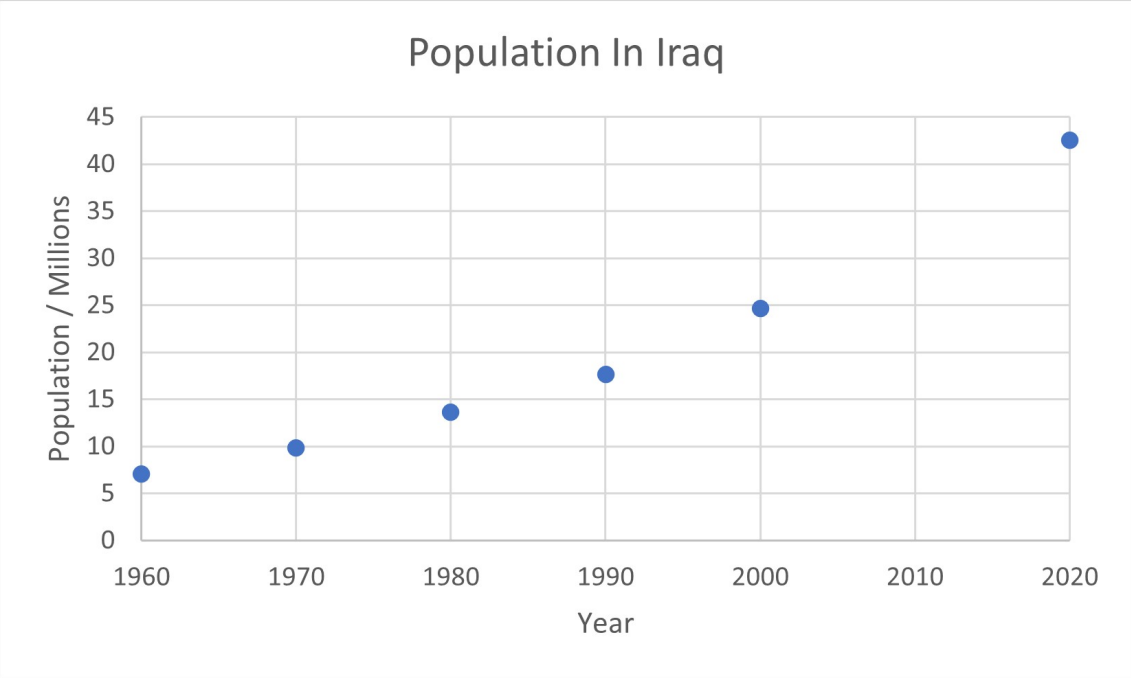



The points may be measurements from a physical problem $[x_i, y_i], i = 0, 1, 2, \ldots, n$, or values of a mathematical function $[x_i, f(x_i)], i = 0, 1, 2, \ldots, n$. The function we fit is usually a polynomial because polynomials are easy to work with.

Why do we need interpolation?

- to estimate the value of the function at a point not in the table of measurements/values.
- to fit a smooth function to the data as a mathematical model.
- to have a representation for functions such as $\sin(x), \ln(x)$, etc. which we may use to evaluate these functions in a computer/calculator
- numerical differentiation or integration of functions or discrete data is often done by differentiating or integrating a good polynomial (or other) representation of the function.

■ **Example 3.1** A census of the population of Iraq is taken every 10 years. The following table lists the population, in thousands of people, from 1960 to 2020. ■




In reviewing these data, we might ask whether they could be used to provide a reasonable estimate of the population, say, in 1975, or even in the year 2030. Predictions of this type can be obtained by using a function that fits the given data. This process is called interpolation.

■ **Example 3.2**

| $x_i$ | 1 | 2 | 3 |
|---|---|---|---|
| $y_i$ | 2 | 0 | 0 |

■

The following four functions all pass through the three given points.

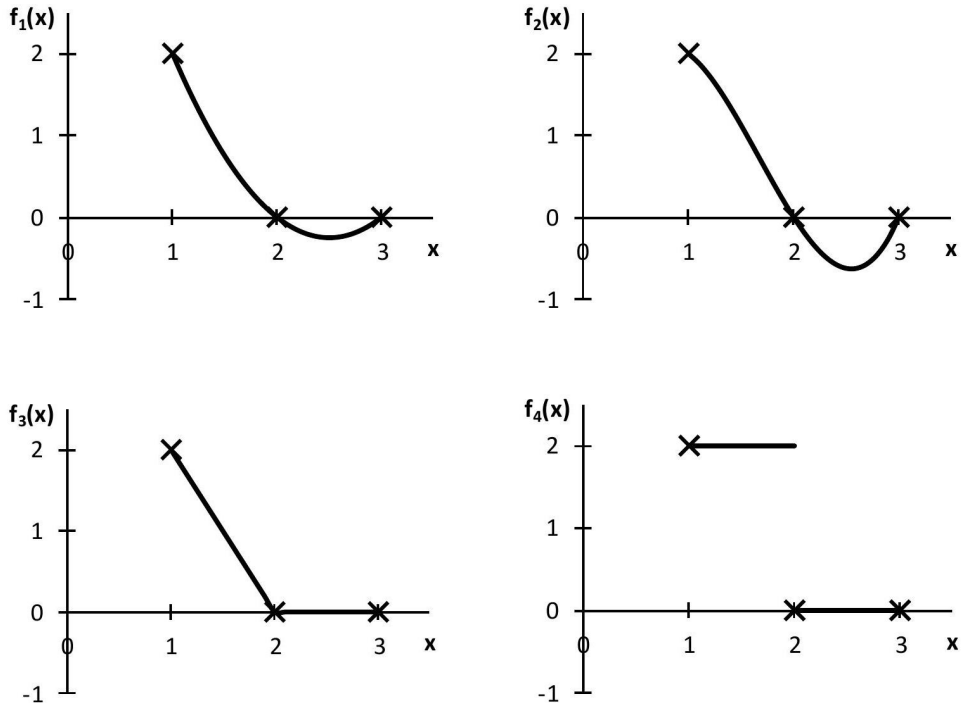
$$f_1(x) = x^2 - 5x + 6$$

$$f_2(x) = x\left(x^2 - 5x + 6\right)$$

$$f_3(x) = \begin{cases} 4 - 2x & 1 \le x \le 2 \\ 0 & 2 < x \le 3 \end{cases}$$

$$f_4(x) = \begin{cases} 2 & 1 \le x < 2 \\ 0 & 2 \le x \le 3 \end{cases}$$

You can probably think of many more!




If these interpolating functions are used to estimate a value at $x = 1.5$ we get four very different answers:

$$f_1(1.5) = 0.75, \quad f_2(1.5) = 1.125, \quad f_3(1.5) = 1, \quad f_4(1.5) = 2$$

Which is correct? Is there a correct answer?!

The point of this example is to indicate some of the problems associated with interpolation.

## 3.4 Lagrange Polynomial Interpolation

The polynomial interpolation problem, also called Lagrange interpolation, can be described as follows: Given $n+1$ data points $(x_i, y_i)$, $i = 0, 1, \ldots, n$, find a polynomial $P$ of lowest possible degree such that

$$y_i = P(x_i), \qquad i = 0, 1, \ldots, n. \tag{3.1}$$

Such a polynomial is said to interpolate the data. Here $y_i$ may be the value of some unknown function $f$ at $x_i$, i.e., $y_i = f(x_i)$.

One reason for considering the class of polynomials in approximation of functions is that they uniformly approximate continuous function.

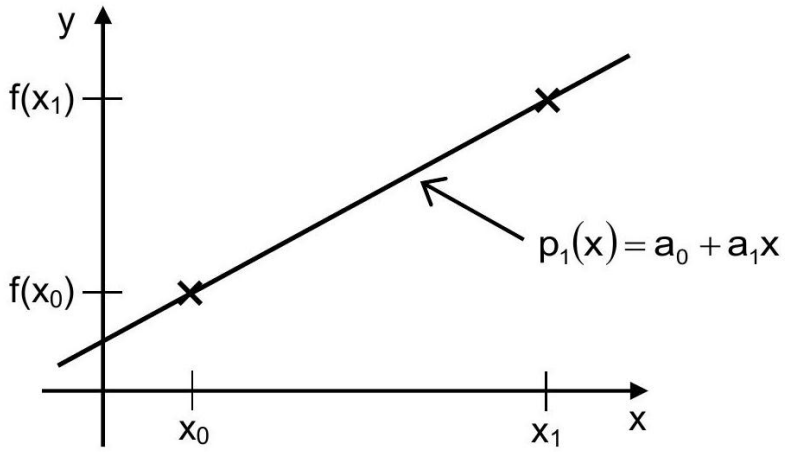
**Theorem 3.4.1 — Weierstrass Approximation Theorem.** Suppose that $f$ is defined and continuous on $[a,b]$. For any $\varepsilon > 0$, there exists a polynomial $P(x)$, defined on $[a,b]$, with the property that

$$|f(x) - P(x)| < \varepsilon, \qquad \text{for all } x \text{ in } [a,b].$$

Another important reason for considering the class of polynomials in approximation of functions is that the derivatives and indefinite integral of a polynomial are easy to determine and are also polynomials. For these reasons, polynomials are often used for approximating continuous functions.

### 3.4.1 Linear Interpolation

Suppose we have two points $[x_0, f(x_0)]$ and $[x_1, f(x_1)]$. We may fit a linear polynomial, $p_1(x) = a_0 + a_1 x$.




To find the coefficients $a_0$ and $a_1$ we may set up the two linear equations:

$$\text{at} \quad x = x_0, \quad a_0 + a_1 x_0 = f(x_0)$$
$$\text{at} \quad x = x_1, \quad a_0 + a_1 x_1 = f(x_1)$$

Solving gives,

$$a_1 = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \quad a_0 = f(x_0) - a_1 x_0$$

and then

$$p_1(x) = f(x_0) + \left[\frac{f(x_1) - f(x_0)}{x_1 - x_0}\right](x - x_0)$$

■ **Example 3.3** Find the linear polynomial through the points $(1,1)$ and $(4,2)$. Assuming that these data points actually came from the function $f(x) = \sqrt{x}$, use the linear polynomial to estimate $\sqrt{2}$. ■



■ **Example 3.4** Determine the linear polynomial through the following two points and then estimate $e^{0.87}$.

| x | 0.8 | 0.9 |
|---|---|---|
| $e^x$ | 2.22554 | 2.45960 |

■

We note the following concerning linear interpolation:

1. Clearly there is only one linear polynomial through two distinct data points.
2. Observe that $p_1(x) = f(x_0) + \left[\frac{f(x_1)-f(x_0)}{x_1-x_0}\right](x-x_0)$ may be written as

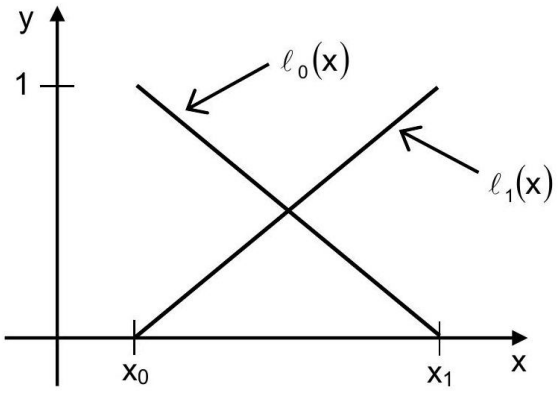
Hence

$$p_1(x) = \frac{(x-x_1)}{(x_0-x_1)}f(x_0) + \frac{(x-x_0)}{(x_1-x_0)}f(x_1)$$
$$= \ell_0(x)f(x_0) + \ell_1(x)f(x_1)$$

where $\ell_0(x) = \frac{x-x_1}{x_0-x_1}$ and $\ell_1(x) = \frac{x-x_0}{x_1-x_0}$

$\ell_0(\mathrm{x})$ and $\ell_1(\mathrm{x})$ are both linear functions with the properties

$$\ell_0(\mathrm{x}_0) = 1, \quad \ell_0(\mathrm{x}_1) = 0$$
$$\ell_1(\mathrm{x}_0) = 0, \quad \ell_1(\mathrm{x}_1) = 1$$



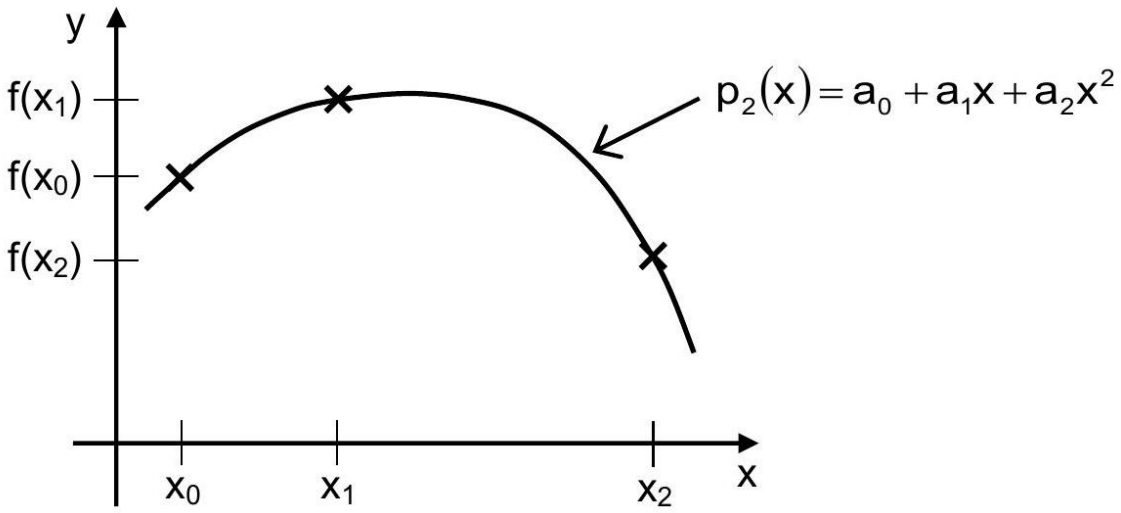

Hence $p_1(x)$ is also linear and,

$$\text{at } x_0 \quad p_1(x_0) = (1)f(x_0) + (0)f(x_1) = f(x_0)$$
$$\text{at } x_1 \quad p_1(x_1) = (0)f(x_0) + (1)f(x_1) = f(x_1)$$

i.e. it goes through the two data points as required.

### 3.4.2 Quadratic Interpolation

Given three points $[x_0, f(x_0)], [x_1, f(x_1)]$ and $[x_2, f(x_2)]$ we may fit a quadratic polynomial $p_2(x) = a_0 + a_1 x + a_2 x^2$.




To find the coefficients $a_0, a_1$ and $a_2$ we could set up and solve the three linear equations,

$$\begin{aligned} \text{at } x = x_0 \quad & a_0 + a_1 x_0 + a_2 x_0^2 = f(x_0) \\ \text{at } x = x_1 \quad & a_0 + a_1 x_1 + a_2 x_1^2 = f(x_1) \\ \text{at } x = x_2 \quad & a_0 + a_1 x_2 + a_2 x_2^2 = f(x_2) \end{aligned}$$

Alternatively consider,

$$\begin{aligned} p_2(x) &= \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} f(x_0) + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} f(x_1) + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} f(x_2) \\ &= \ell_0(x) f(x_0) + \ell_1(x) f(x_1) + \ell_2(x) f(x_2) \end{aligned}$$

where,

$$\begin{aligned} \ell_0(x) &= \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \\ \ell_1(x) &= \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \\ \ell_2(x) &= \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \end{aligned}$$

Now, $\ell_0(x)$ is a quadratic polynomial which is zero when $x = x_1$ or $x = x_2$ and is unity when $x = x_0$. The functions $\ell_1(x)$ and $\ell_2(x)$ have similar properties.

These may be summarized as:

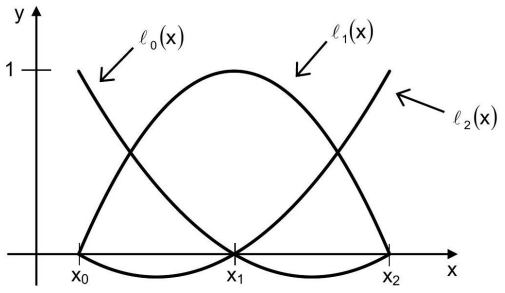
$$\left.\begin{array}{l} \ell_i(x_i) = 1 \\ \ell_i(x_j) = 0 \quad i \neq j \end{array}\right\} \quad \text{for} \quad i, j = 0, 1, 2$$

or,

$$\ell_i(x_j) = \delta_{ij} \quad \text{for } i, j = 0, 1, 2$$

where $\delta_{ij}$ is the Kronecker delta function with the property

$$\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$




Clearly $p_2(x)$ is a quadratic polynomial and it does pass through the three given data points.

i.e. $p_2(x_i) = f(x_i)$ for $i = 0, 1, 2$ since,

$$\begin{aligned} p_2(x) &= \ell_0(x) f(x_0) + \ell_1(x) f(x_1) + \ell_2(x) f(x_2) \\ p_2(x_i) &= \ell_0(x_i) f(x_0) + \ell_1(x_i) f(x_1) + \ell_2(x_i) f(x_2) \\ &= f(x_i) \end{aligned}$$

The three functions $\ell_0(x), \ell_1(x)$ and $\ell_2(x)$ are called the Lagrange interpolation basis functions and the above form of the interpolating polynomial $p_2(x)$ is known as the Lagrange form.

■ **Example 3.5** Determine the quadratic polynomial through the following three points and then estimate $e^{0.87}$.

| x | 0.8 | 0.9 | 1.0 |
|---|---|---|---|
| $e^x$ | 2.22554 | 2.45960 | 2.71828 |

■

### 3.4.3 Higher Order Interpolation

In general through $(n+1)$ points $[x_i, f(x_i)], i = 0, 1, \ldots, n$ where the $x_i$ values do not need to be equally spaced, we may pass a polynomial of degree $n$,

$$\begin{aligned} p_n(x) &= \ell_{n,0}(x) f(x_0) + \ell_{n,1}(x) f(x_1) + \ldots + \ell_{n,n}(x) f(x_n) \\ &= \sum_{k=0}^{n} \ell_{n,k}(x) f(x_i) \end{aligned}$$

where,

$$\ell_{n,k}(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)(x_k-x_1)\cdots(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)}$$

is the Lagrange basis function.

The interpolating polynomial is easily described once the form of $L_{n,k}$ is known. This polynomial, called the nth Lagrange interpolating polynomial, is defined in the following theorem.

**Theorem 3.4.2** If $x_0, x_1, \ldots, x_n$ are $n+1$ distinct numbers and $f$ is a function whose values are given at these numbers, then a unique polynomial of degree at most $n$ exists with

$$f(x_k) = P(x_k), \quad \text{for } k = 0, 1, \ldots, n.$$

This polynomial is given by

$$P(x) = \sum_{k=0}^{n} f(x_k) L_{n,k}(x)$$

which is called the $n$th Lagrange interpolating polynomial.

■ **Example 3.6** Given the following 4 data points,

| $x_i$ | 0 | 1 | 3 | 5 |
|---|---|---|---|---|
| $y_i$ | 1 | 2 | 6 | 7 |

find a polynomial in Lagrange form to interpolate these data. ■

$$
\begin{aligned}
L_0(x) &= \frac{(x-1)(x-3)(x-5)}{(0-1)(0-3)(0-5)} = -\frac{1}{15}(x-1)(x-3)(x-5),\\
L_1(x) &= \frac{(x-0)(x-3)(x-5)}{(1-0)(1-3)(1-5)} = \frac{1}{8}x(x-3)(x-5),\\
L_2(x) &= \frac{(x-0)(x-1)(x-5)}{(3-0)(3-1)(3-5)} = -\frac{1}{12}x(x-1)(x-5),\\
L_3(x) &= \frac{(x-0)(x-1)(x-3)}{(5-0)(5-1)(5-3)} = \frac{1}{40}x(x-1)(x-3).
\end{aligned}
$$

The interpolating polynomial in the Lagrange form is

$$P_3(x) = L_0(x) + 2L_1(x) + 6L_2(x) + 7L_3(x).$$

■

## 3.5 Newton's Form of Polynomial Interpolation

Since there is one and only one polynomial of degree $\leq n$ that takes prescribed values at given $n+1$ distinct points, various algorithms will produce the same polynomial but in different forms. For example, the polynomial can be constructed as a linear combination of the basic polynomials $1, x, x^2, \ldots, x^n$. For numerical work, it is preferable to use the following basis

$$\begin{aligned} q_0(x) &= 1, \\ q_1(x) &= (x - x_0), \\ q_2(x) &= (x - x_0)(x - x_1), \\ &\vdots \\ q_n(x) &= (x - x_0)(x - x_1)\cdots(x - x_{n-1}), \end{aligned}$$

and express $P(x)$ as

$$\begin{aligned} P_n(x) &= c_0 + c_1(x - x_0) + c_2(x - x_0)(x - x_1) + \cdots + c_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}). \\ &= \sum_{k=0}^{n} c_k q_k(x) \end{aligned} \tag{3.2}$$

This polynomial is called the interpolating polynomial in Newton's form. The interpolation conditions

$$P_n(x_i) = \sum_{k=0}^{n} c_k q_k(x_i) = y_i, \qquad i = 0, 1, \ldots, n,$$

give rise to $n+1$ equations with $n+1$ parameters $c_0, c_1, \ldots, c_n$ to be determined

$$\begin{aligned}
c_0 &= y_0, \\
c_0 + c_1(x_1 - x_0) &= y_1, \\
c_0 + c_1(x_2 - x_0) + c_2(x_2 - x_0)(x_2 - x_1) &= y_2, \\
&\vdots \\
c_0 + c_1(x_n - x_0) + c_2(x_n - x_0)(x_n - x_1) + \cdots + c_n(x_n - x_0)\cdots(x_n - x_{n-1}) &= y_n.
\end{aligned}$$

In matrix equation, we have a triangular linear system

$$\begin{bmatrix}
1 & 0 & 0 & \cdots & 0 \\
1 & x_1 - x_0 & 0 & \cdots & 0 \\
1 & x_2 - x_0 & (x_2 - x_0)(x_2 - x_1) & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_n - x_0 & (x_n - x_0)(x_n - x_1) & \cdots & (x_n - x_0)\cdots(x_n - x_{n-1})
\end{bmatrix}
\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}
=
\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}. \tag{3.3}$$

Use forward substitution, we can solve for $c_0, c_1, \ldots, c_n$. Computational work involved in this approach includes forming and solving the system.

■ **Example 3.7** Given the following 4 points ($n = 3$)

| $x_i$ | 0 | 1 | 3 | 5 |
|---|---|---|---|---|
| $y_i$ | 1 | 2 | 6 | 7 |

find a polynomial of degree 3 in Newton's form to interpolate these data. ■

**Solution**

Write

$$\begin{aligned} P(x) &= c_0 + c_1(x-0) + c_2(x-0)(x-1) + c_3(x-0)(x-1)(x-3) \\ &= c_0 + c_1 x + c_2 x(x-1) + c_3 x(x-1)(x-3). \end{aligned}$$

Plug in the interpolation conditions,

$$\begin{aligned} P(0) &= c_0 = 1. \\ P(1) &= c_0 + c_1 = 2. \\ P(3) &= c_0 + 3c_1 + 6c_2 = 6. \\ P(5) &= c_0 + 5c_1 + 20c_2 + 40c_3 = 7, \end{aligned}$$

which lead to

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 3 & 6 & 0 \\ 1 & 5 & 20 & 40 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 6 \\ 7 \end{bmatrix} \quad \Longrightarrow \quad \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \frac{1}{3} \\ -\frac{17}{120} \end{bmatrix}.$$

Thus

$$P(x) = 1 + x + \frac{1}{3}x(x-1) - \frac{17}{120}x(x-1)(x-3).$$

■

### 3.5.1 Divided Differences Scheme

With proper arrangement, the triangular linear system in the previous subsection can be solved with the formulations

$$\begin{aligned} c_0 &= f(x_0) \\ c_1 &= \frac{f(x_1)-f(x_0)}{x_1-x_0} \\ c_2 &= \frac{1}{x_2-x_0}\left[\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}\right] \\ &\vdots \end{aligned}$$

and so on. An efficient procedure, called the **divided-difference method**, can be derived for computing the $n+1$ coefficients $c_0, c_1, \cdots, c_n$ in the Newton's form interpolating polynomial.

First we introduce the divided-difference notation. The zero divided difference of the function $f$ with respect to $x_i$,

$$f[x_i] = f(x_i), \tag{3.4}$$

is simply the value of $f$ at $x_i$. The first divided difference of $f$ with respect to $x_i$ and $x_{i+1}$ is denoted and defined as

$$f[x_i, x_{i+1}] = \frac{f[x_{i+1}] - f[x_i]}{x_{i+1} - x_i}. \tag{3.5}$$

The second divided difference of $f$ is defined as

$$f[x_i, x_{i+1}, x_{i+2}] = \frac{f[x_{i+1}, x_{i+2}] - f[x_i, x_{i+1}]}{x_{i+2} - x_i}. \tag{3.6}$$

In general, the $k$-th divided difference relative to $x_i, x_{i+1}, \ldots, x_{i+k}$ is given by

$$f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}. \tag{3.7}$$

As might be expected from the evaluation of $c_0, c_1, \ldots, c_n$ in the Newton form interpolating polynomial, the required coefficients are given by

$$c_k = f[x_0, x_1, \ldots, x_k], \qquad k = 0, 1, \ldots, n. \tag{3.8}$$

Therefore the interpolating polynomial in Newton's form can be expressed as

$$P_n(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k](x - x_0)(x - x_1)\cdots(x - x_{k-1}). \tag{3.9}$$

This formula is known as the Newton's interpolatory divided-difference formula. The determination of the divided differences is best described by using tabular form as shown in the following table as an example.

| $x_i$ | $k=0$ | $k=1$ | $k=2$ | $k=3$ |
|---|---|---|---|---|
| $x_0$ | $f[x_0]$ | | | |
| | | $> f[x_0, x_1]$ | | |
| $x_1$ | $f[x_1]$ | | $> f[x_0, x_1, x_2]$ | |
| | | $> f[x_1, x_2]$ | | $> f[x_0, x_1, x_2, x_3]$ |
| $x_2$ | $f[x_2]$ | | $> f[x_1, x_2, x_3]$ | |
| | | $> f[x_2, x_3]$ | | |
| $x_3$ | $f[x_3]$ | | | |

| $x_i$ | $f[x_i]$ | $f[x_i,x_{i+1}]$ | $f[x_i,x_{i+1},x_{i+2}]$ | $\cdots$ | $f[x_i,x_{i+1},\ldots,x_n]$ |
|---|---|---|---|---|---|
| $x_0$ | $f[x_0]$ | | | | |
| | | $> f[x_0,x_1]$ | | | |
| $x_1$ | $f[x_1]$ | | $> f[x_0,x_1,x_2]$ | | |
| | | $> f[x_1,x_2]$ | | | |
| $x_2$ | $f[x_2]$ | | $> f[x_1,x_2,x_3]$ | | |
| | | $> f[x_2,x_3]$ | $\vdots$ | $> \cdots$ | |
| $x_3$ | $f[x_3]$ | $\vdots$ | $\vdots$ | | |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | | |
| $\vdots$ | $\vdots$ | $\vdots$ | | | |
| $\vdots$ | $\vdots$ | $> f[x_{n-1},x_n]$ | | | |
| $x_n$ | $f[x_n]$ | | | | |

■ **Example 3.8** Given the following 4 points ($n = 3$)

| $x_i$ | 0 | 1 | 3 | 5 |
|---|---|---|---|---|
| $y_i$ | 1 | 2 | 6 | 7 |

find a polynomial of degree 3 in Newton's form to interpolate these data. ■

**Solution**

| $x_i$ | $k=0$ | $k=1$ | $k=2$ | $k=3$ |
|---|---|---|---|---|
| 0 | 1 | | | |
| | | $> 1$ | | |
| 1 | 2 | | $> \frac{1}{3}$ | |
| | | $> 2$ | | $> -\frac{17}{120}$ |
| 3 | 6 | | $> -\frac{3}{8}$ | |
| | | $> \frac{1}{2}$ | | |
| 5 | 7 | | | |

So,

$$P(x) = 1 + x + \frac{1}{3}x(x-1) - \frac{17}{120}x(x-1)(x-3).$$

Note that $x_i$ can be reordered, but must be distinct. When the order of $x_i$ are changed, one obtains the same polynomial but in different form.

■ **Example 3.9** Given the following 4 points ($n = 3$)

| $x_i$ | 3 | 1 | 5 | 0 |
|---|---|---|---|---|
| $y_i$ | 6 | 2 | 7 | 1 |

find a polynomial of degree 3 in Newton's form to interpolate these data. ■

**Solution**

| $x_i$ | $k=0$ | $k=1$ | $k=2$ | $k=3$ |
|---|---|---|---|---|
| 3 | 6 | | | |
| | | $> 2$ | | |
| 1 | 2 | | $> -\frac{3}{8}$ | |
| | | $> \frac{5}{4}$ | | $> -\frac{17}{120}$ |
| 5 | 7 | | $> \frac{1}{20}$ | |
| | | $> \frac{6}{5}$ | | |
| 0 | 1 | | | |

So,

$$P(x) = 6 + 2(x-3) - \frac{3}{8}(x-3)(x-1) - \frac{17}{120}(x-3)(x-1)(x-5).$$

■

Note that $c_0, c_1, \ldots, c_{n-1}$ are changed, but $c_n$ is unchanged, and the polynomial is expressed in different form.

**Theorem 3.5.1** Suppose $f \in C^n[a,b]$ and $x_0, x_1, \ldots, x_n$ are distinct numbers in $[a,b]$. Then there exists $\xi \in (a,b)$ such that

$$f[x_0, x_1, \ldots, x_n] = \frac{f^{(n)}(\xi)}{n!}.$$

*Proof.* Let

$$P_n(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k](x - x_0)(x - x_1)\cdots(x - x_{k-1})$$

be the interpolating polynomial of $f$ in Newton's form. Define

$$g(x) = f(x) - P_n(x).$$

Since $P_n(x_i) = f(x_i)$ for $i = 0, 1, \ldots, n$, the function $g$ has $n+1$ distinct zeros in $[a,b]$. By the generalized Rolle's Theorem, there exists $\xi \in (a,b)$ such that

$$g^{(n)}(\xi) = f^{(n)}(\xi) - P_n^{(n)}(\xi) = 0.$$

Note that

$$P_n^{(n)}(x) = n! f[x_0, x_1, \ldots, x_n].$$

As a consequence

$$f[x_0, x_1, \ldots, x_n] = \frac{f^{(n)}(\xi)}{n!}.$$

■

An algorithm, to be introduced, for computing the divided-difference table is very efficient and is recommended as the best scheme for producing an interpolating polynomial in Newton's form. First we change the notation to

$$\begin{aligned} a_{i,k} &= f[x_i, x_{i+1}, \ldots, x_{i+k}] \\ &= \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_{i+k} - x_i} = \frac{a_{i+1,k-1} - a_{i,k-1}}{x_{i+k} - x_i} \end{aligned}$$

so that the divided-difference table has the entries as shown in the following.

| $x_i$ | $k=0$ | $k=1$ | $k=2$ | | $k=n$ |
|---|---|---|---|---|---|
| $x_0$ | $a_{00}$ | | | | |
| | | $\rangle$ $a_{01}$ | | | |
| $x_1$ | $a_{10}$ | | $\rangle$ $a_{02}$ | | |
| | | $\rangle$ $a_{12}$ | | $\ddots$ | |
| $x_2$ | $a_{20}$ | | $\rangle$ $a_{12}$ | $\ddots$ | |
| | | $\rangle$ $a_{21}$ | | | $\cdots$ $a_{0,n}$ |
| $x_3$ | $a_{30}$ | | $\vdots$ | $\ddots$ | |
| | | $\vdots$ $\vdots$ | | $\ddots$ | |
| $\vdots$ | $\vdots$ | | $\ddots$ | | |
| | | $\rangle$ $a_{n-1,1}$ | | | |
| $x_n$ | $a_{n,0}$ | | | | |

The interpolating polynomial, of course, is

$$P_n(x) = a_{00} + \sum_{k=1}^{n} a_{0,k} \prod_{j=0}^{k-1} (x - x_j). \tag{3.10}$$

### 3.5.2 Divided-Difference Algorithm

Given $n+1$ distinct data points $(x_i, y_i)$, $i = 0, 1, \ldots, n$, this algorithm computes the coefficients of the interpolating polynomial in Newton's form by constructing the divided-difference table.

This algorithm uses one 2-d array of $(n+1) \times (n+1)$ for the storage of all $a_{ij}$. The inner $i$-loop can be performed in parallel. If only the coefficients in the Newton's interpolating polynomial are required for one set of data, an improved algorithm which uses only one 1-d array of size $n+1$ can be derived. The modified algorithm, however, can not be executed in parallel.

Given $n+1$ distinct points $(x_i, y_i)$, $i = 0, 1, \ldots, n$, this algorithm computes the coefficients of the interpolating polynomial in Newton's form using an 1-d array of size $n+1$..

Newton's interpolatory divided-difference formula can be expressed in a simplified form when $x_0, x_1, \ldots, x_n$ are arranged consecutively with equal spacing. Let

$$h = \frac{x_n - x_0}{n} = x_{i+1} - x_i, \qquad i = 0, 1, \ldots, n-1.$$

Then each $x_i = x_0 + i * h$, $i = 0, 1, \ldots, n$. For any $x \in [a, b]$, write

$$x = x_0 + s * h, \qquad s \in .$$

Hence $x - x_i = (s - i) * h$ and

$$\begin{aligned} P_n(x) &= f[x_0] + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k](x - x_0)(x - x_1) \cdots (x - x_{k-1}) \\ &= f(x_0) + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k](s-0) * h * (s-1) * h * \cdots (s-k+1) * h \\ &= f(x_0) + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k] s(s-1) \cdots (s-k+1) h^k \\ &= f(x_0) + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k] k! \binom{s}{k} h^k, \end{aligned} \tag{3.11}$$

where the binomial formula

$$\binom{s}{k} = \frac{s(s-1)\cdots(s-k+1)}{k!}$$

is used. This formula is called the Newton forward divided-difference formula.

Next we introduce the forward difference notation $\triangle$

$$\triangle f(x_i) = f(x_{i+1}) - f(x_i) \tag{3.12}$$

and

$$\triangle^k f(x_i) = \triangle^{k-1} f(x_{i+1}) - \triangle^{k-1} f(x_i) = \triangle\left(\triangle^{k-1} f(x_i)\right), \tag{3.13}$$

for $i = 0, 1, \ldots, n-1$. With this notation,

$$\begin{aligned} f[x_0, x_1] &= \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{1}{h}\triangle f(x_0) \\ f[x_0, x_1, x_2] &= \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = \frac{\frac{1}{h}\triangle f(x_1) - \frac{1}{h}\triangle f(x_0)}{2h} = \frac{1}{2h^2}\triangle^2 f(x_0), \end{aligned}$$

and, in general

$$f[x_0, x_1, \ldots, x_k] = \frac{1}{k!\,h^k}\triangle^k f(x_0). \tag{3.14}$$

Using the forward difference $\triangle$ notation, the Newton's forward divided-difference formula can be expressed as

$$P_n(x) = f(x_0) + \sum_{k=1}^{n} \binom{s}{k} \triangle^k f(x_0). \tag{3.15}$$

This formula is called the Newton forward-difference formula. If the interpolation nodes are arranged as $x_n, x_{n-1}, \ldots, x_0$, a formula for the interpolating polynomial similar to results

$$P_n(x) = f[x_n] + \sum_{k=1}^{n} f[x_n, x_{n-1}, \ldots, x_{n-k}](x - x_n)(x - x_{n-1}) \cdots (x - x_{n-k+1}). \quad (3.16)$$

If the nodes are equally spaced with

$$h = \frac{x_n - x_0}{n}, \qquad x_i = x_n - (n-i) * h, \qquad x = x_n + s * h,$$

then

$$\begin{aligned} P_n(x) &= f[x_n] + \sum_{k=1}^{n} f[x_n, x_{n-1}, \ldots, x_{n-k}] s * h * (s+1) * h * \cdots * (s+k-1) * h \\ &= f(x_n) + \sum_{k=1}^{n} f[x_n, x_{n-1}, \ldots, x_{n-k}](-1)^k k! \binom{-s}{k} h^k \end{aligned} \quad (3.17)$$

where the binomial formula is extended to include all real value $s$

$$\binom{-s}{k} = \frac{-s(-s-1)\cdots(-s-k+1)}{k!} = (-1)^k \frac{s(s+1)\cdots(s+k-1)}{k!}.$$

This form is called the Newton backward divided-difference formula. Similarly, we can introduce the backward-difference notation

$$\nabla^k f(x_i) = \nabla^{k-1} f(x_i) - \nabla^{k-1} f(x_{i-1}) = \nabla \left( \nabla^{k-1} f(x_i) \right), \quad (3.18)$$

then

$$f[x_n, x_{n-1}] = \frac{1}{h} \nabla f(x_n), \qquad f[x_n, x_{n-1}, x_{n-2}] = \frac{1}{2h^2} \nabla^2 f(x_n),$$

and, in general,

$$f[x_n, x_{n-1}, \ldots, x_{n-k}] = \frac{1}{k!\, h^k} \nabla^k f(x_n). \quad (3.19)$$

Using this backward-difference $\nabla$ notation, the Newton backward divided-difference formula can be written as

$$P_n(x) = f(x_0) + \sum_{k=1}^{n} (-1)^k \binom{-s}{k} \nabla^k f(x_k). \tag{3.20}$$

This is called the Newton backward-difference formula.

### 3.5.3 Error Analysis for Polynomial Interpolation

**Theorem 3.5.2** Suppose $f(x) \in C^{n+1}[a,b]$, and $P(x)$ is a polynomial of degree $\leq n$ that interpolates $f$ at $n+1$ given distinct points $x_0, x_1 \ldots, x_n$ in $[a,b]$. Then for each $x \in [a,b]$, there exists a point $\xi_x \in (a,b)$ such that

$$f(x) - P(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{k=0}^{n} (x - x_k). \tag{3.21}$$

*Proof.* Note first that if $x = x_i$ for some $0 \leq i \leq n$, then $P(x) = f(x)$, the theorem holds for any arbitrary $\xi_x \in (a,b)$. Suppose that $x \in [a,b]$ but $x \neq x_i$ for all $i = 0, 1, \ldots, n$. Define a new function

$$g(t) = f(t) - P(t) - [f(x) - P(x)] \prod_{k=0}^{n} \frac{(t - x_k)}{(x - x_k)}, \qquad t \in [a,b].$$

Since $f \in C^{n+1}[a,b]$, $P \in C^{\infty}[a,b]$, and $x \neq x_i, \ \forall\, i$, it follows that $g(t)$ is well-defined for all $t \in [a,b]$, and $g \in C^{n+1}[a,b]$. Then

$$g(x_i) = f(x_i) - P(x_i) - [f(x) - P(x)] \prod_{k=0}^{n} \frac{(x_i - x_k)}{(x - x_k)} = 0, \qquad i = 0, 1, \ldots, n.$$

Moreover

$$g(x) = f(x) - P(x) - [f(x) - P(x)] \prod_{k=0}^{n} \frac{(x - x_k)}{(x - x_k)} = 0.$$

Thus $g \in C^{n+1}[a,b]$ and $g$ has $n+2$ zeros in $[a,b]$. By the generalized Rolle's theorem,

there exists $\xi_x \in (a,b)$ for which $g^{(n+1)}(\xi_x) = 0$. Hence

$$\begin{aligned} 0 = g^{(n+1)}(\xi_x) &= f^{(n+1)}(\xi_x) - P^{(n+1)}(\xi_x) - [f(x) - P(x)] \left[ \frac{d^{n+1}}{dt^{n+1}} \prod_{k=0}^{n} \frac{(t-x_k)}{(x-x_k)} \right] \Bigg|_{t=\xi_x} \\ &= f^{(n+1)}(\xi_x) - 0 - [f(x) - P(x)] \cdot \frac{(n+1)!}{\prod (x-x_k)}. \end{aligned}$$

Therefore

$$f(x) - P(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{k=0}^{n} (x-x_k).$$

■

▪ **Example 3.10** Suppose $f(x) = \sin x$ is approximated by an interpolating polynomial $p(x)$ of degree 9 in $[0,1]$. Estimate $|f(x) - p(x)|$, for all $x \in [0,1]$. ▪

**Solution**

With $n = 9$, and since $f(x) = \sin x$, $\left| f^{(10)}(\xi) \right| \leq 1$, and $x \in [0,1]$,

$$|x - x_i| \leq 1, \quad \Longrightarrow \quad \left| \prod_{i=0}^{10} (x - x_i) \right| \leq 1,$$

we have

$$|f(x) - p(x)| = \frac{1}{10!} \left| f^{(10)}(\xi) \right| \left| \prod_{i=0}^{10} (x - x_i) \right| \leq \frac{1}{10!}.$$

■

■ **Example 3.11**

1. Suppose that we prepare a table for the function $f(x) = e^x$ for $x \in [0,1]$.
2. $x_{j+1} - x_j = h$ for $j = 0, 1, \ldots, n-1$.
3. What should $h$ be for linear interpolation to give an absolute error of at most $10^{-6}$?

**Solution**

Suppose $x \in [x_j, x_{j+1}]$. Equation (3.21) implies that

$$\begin{aligned}
|f(x) - P(x)| &= \left| \frac{f^{(2)}(\xi)}{2!}(x - x_j)(x - x_{j+1}) \right| \\
&= \frac{|f^{(2)}(\xi)|}{2!}|x - x_j||x - x_{j+1}| \\
&= \frac{e^{\xi}}{2}|x - jh||x - (j+1)h| \\
&\leq \frac{1}{2}\left( \max_{\xi \in [0,1]} e^{\xi} \right) \left( \max_{x_j \leq x \leq x_{j+1}} |x - jh||x - (j+1)h| \right) \\
&= \frac{e}{2}\left( \max_{x_j \leq x \leq x_{j+1}} |x - jh||x - (j+1)h| \right).
\end{aligned}$$

Let

$$g(x) = (x - jh)(x - (j+1)h), \text{ for } jh \leq x \leq (j+1)h.$$

Then

$$\max_{x_j \leq x \leq x_{j+1}} |g(x)| = \left| g\left( \left( j + \frac{1}{2} \right) h \right) \right| = \frac{h^2}{4}.$$

Consequently,

$$|f(x) - P(x)| \leq \frac{eh^2}{8} \leq 10^{-6},$$

which implies that

$$h < 1.72 \times 10^{-3}.$$

Since $n = (1-0)/h$ must be an integer, one logical choice for the step size is $h = 0.001$. ■

■ **Example 3.12** If $f(x) = \sqrt{x}$ on $[0,1]$, then

$$f'(x) = \frac{1}{2}x^{-\frac{1}{2}}, \qquad f''(x) = -\frac{1}{4}x^{-\frac{3}{2}}, \qquad f'''(x) = \frac{3}{8}x^{-\frac{5}{2}}, \qquad \cdots$$

Thus

$$f^{(n+1)}(x) \longrightarrow \infty \quad \text{for } x \in (0,1).$$

■

# IV Numerical Differentiation and Integration


4 Numerical Differentiation and Integration ..... 65

4.1 Adaptive Simpson (Practical Quadrature Default)

4.2 NUMERICAL DIFFERENTIATION

4.3 Richardson Extrapolation Method

4.4 Elements of Numerical Integration

4.5 Composite Numerical Integration

4.6 Romberg Integration

# 4. Numerical Differentiation and Integratio

## 4.1 Adaptive Simpson (Practical Quadrature Default)

Use Simpson on $[a,b]$ and on halves $[a,m],[m,b]$ with $m=\frac{a+b}{2}$. If the difference is below a scaled tolerance, accept; otherwise, recurse on subintervals and sum.

## 4.2 NUMERICAL DIFFERENTIATION

Suppose we want to differentiate (i.e. find the rate of change of) a function which is only tabulated at discrete points, perhaps because it is data from an experiment, or we have a function which is difficult or impossible to differentiate analytically, perhaps because although we may calculate the function at any point we do not have an explicit representation for it.

Consider a function $f(x)$ and the problem of finding $f'(x)$ at some point $x_0$. The derivative of the function $f$ at $x_0$ is:

$$f'(x_0)=\lim_{h\to 0}\frac{f(x_0+h)-f(x_0)}{h}.$$

This formula gives an obvious way to generate an approximation to $f\prime(x)$; simply

compute:

$$\frac{f(x_0+h)-f(x_0)}{h}$$

However, what if you want to get a sense of how accurate this approximation is?

Do you see any relation between the choice of $h$ and the size of the error?

What can we do if we want to get a bound on how large the actual error could get?

### 4.2.1 Derivation Using Taylor Series Expansion

To compute difference formula which approximate derivatives at a point is to use Taylor Approximations. This is useful because again one gets a sense of how good the approximation is as one is making it.

Suppose a given function $f$ has continuous first derivative and $f''$ exists. From Taylor's theorem

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(\xi)h^2,$$

where $\xi$ is between $x$ and $x+h$, one has

$$f'(x) = \frac{f(x+h)-f(x)}{h} - \frac{h}{2}f''(\xi) = \frac{f(x+h)-f(x)}{h} + O(h).$$

Hence it is reasonable to use the approximation

$$f'(x) \approx \frac{f(x+h)-f(x)}{h} \tag{4.1}$$

which is called **forward finite difference**, and the error involved is

$$|e| = \frac{h}{2}|f''(\xi)| \leq \frac{h}{2}\max_{t\in(x,x+h)}|f''(t)|.$$

Similarly one can derive the **backward finite difference** approximation

$$f'(x) \approx \frac{f(x)-f(x-h)}{h} \tag{4.2}$$

which has the same order of truncation error as the forward finite difference scheme.

The forward difference is an $O(h)$ scheme. An $O(h^2)$ scheme can also be derived from the Taylor's theorem

$$\begin{aligned} f(x+h) &= f(x)+f'(x)h+\frac{1}{2}f''(x)h^2+\frac{1}{6}f'''(\xi_1)h^3 \\ f(x-h) &= f(x)-f'(x)h+\frac{1}{2}f''(x)h^2-\frac{1}{6}f'''(\xi_2)h^3, \end{aligned}$$

where $\xi_1$ is between $x$ and $x+h$ and $\xi_2$ is between $x$ and $x-h$. Hence

$$f(x+h)-f(x-h) = 2f'(x)h+\frac{1}{6}[f'''(\xi_1)+f'''(\xi_2)]h^3$$

and

$$f'(x) = \frac{f(x+h)-f(x-h)}{2h}-\frac{1}{12}[f'''(\xi_1)+f'''(\xi_2)]h^2$$

Let

$$M = \max_{z\in[x-h,x+h]} f'''(z) \quad \text{and} \quad m = \min_{z\in[x-h,x+h]} f'''(z).$$

If $f'''$ is continuous on $[x-h,x+h]$, then by the intermediate value theorem, there exists $\xi \in [x-h,x+h]$ such that

$$f'''(\xi) = \frac{1}{2}[f'''(\xi_1)+f'''(\xi_2)].$$

Hence

$$f'(x) = \frac{f(x+h)-f(x-h)}{2h}-\frac{1}{6}f'''(\xi)h^2 = \frac{f(x+h)-f(x-h)}{2h}+O(h^2). \tag{4.3}$$

This is called **center difference approximation** and the truncation error is

$$|e| = \frac{h^2}{6} f'''(\xi)$$

Similarly, we can derive an $O(h^2)$ scheme from Taylor's theorem for $f''(x)$

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - \frac{1}{12} f^{(4)}(\xi) h^2, \tag{4.4}$$

where $\xi$ is between $x-h$ and $x+h$.

### 4.2.2 Derivation Using Lagrange Polynomial Interpolation

Another way to compute difference formula which approximate derivatives at a point is to use our knowledge of Lagrange Interpolation to come up with formulas that allow us to approximate the **derivative** of a function as a point, even though we are only given a number of values of the function.

Suppose that $(x_0, f(x_0))$, $(x_1, f(x_1)) \cdots, (x_n, f(x_n))$ have been given, we apply the Lagrange polynomial interpolation scheme to derive

$$P(x) = \sum_{i=0}^{n} f(x_i) L_i(x), \qquad \text{where} \qquad L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}.$$

Since $f(x)$ can be written as

$$f(x) = \sum_{i=0}^{n} f(x_i) L(x) + \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) w(x), \qquad \text{where} \qquad w(x) = \prod_{j=0}^{n} (x - x_j),$$

we have,

$$\begin{aligned} f'(x) &= \sum_{i=0} n f(x_i) L_i'(x) + \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) w'(x) \\ &+ \frac{1}{(n+1)!} w(x) \frac{d}{dx} f^{(n+1)}(\xi_x). \end{aligned}$$

Note that

$$w'(x) = \sum_{j=0}^{n} \prod_{i=0, i\neq j}^{n} (x - x_i).$$

Hence a reasonable approximation for the first derivative of $f$ is

$$f'(x) \approx \sum_{i=0}^{n} f(x_i) L_i'(x).$$

When $x = x_k$ for some $0 \leq k \leq n$,

$$w(x_k) = 0 \quad \text{and} \quad w'(x_k) = \prod_{i=0, i\neq k}^{n} (x_k - x_i).$$

Hence

$$f'(x_k) = \sum_{i=0}^{n} f(x_i) L_i'(x_k) + \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{i=0, i\neq k}^{n} (x_k - x_i), \tag{4.5}$$

which is called an **(n+1)-point formula to approximate** $f'(x)$.

- **Two Point Formulas**

• **Three Point Formulas** Since

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}$$

we have

$$L_0'(x) = \frac{2x-x_1-x_2}{(x_0-x_1)(x_0-x_2)}.$$

Similarly,

$$L_1'(x) = \frac{2x-x_0-x_2}{(x_1-x_0)(x_1-x_2)} \text{ and } L_2'(x) = \frac{2x-x_0-x_1}{(x_2-x_0)(x_2-x_1)}.$$

Hence

$$\begin{aligned} f'(x_j) &= f(x_0)\left[\frac{2x_j-x_1-x_2}{(x_0-x_1)(x_0-x_2)}\right] + f(x_1)\left[\frac{2x_j-x_0-x_2}{(x_1-x_0)(x_1-x_2)}\right] \\ &+ f(x_2)\left[\frac{2x_j-x_0-x_1}{(x_2-x_0)(x_2-x_1)}\right] + \frac{1}{6}f^{(3)}(\xi_j)\prod_{k=0,k\neq j}^{2}(x_j-x_k), \end{aligned}$$

for each $j = 0,1,2$. Assume that

$$x_1 = x_0 + h \text{ and } x_2 = x_0 + 2h, \text{ for some } h \neq 0.$$

Then

$$\begin{aligned} f'(x_0) &= \frac{1}{h}\left[-\frac{3}{2}f(x_0) + 2f(x_1) - \frac{1}{2}f(x_2)\right] + \frac{h^2}{3}f^{(3)}(\xi_0), \\ f'(x_1) &= \frac{1}{h}\left[-\frac{1}{2}f(x_0) + \frac{1}{2}f(x_2)\right] - \frac{h^2}{6}f^{(3)}(\xi_1), \\ f'(x_2) &= \frac{1}{h}\left[\frac{1}{2}f(x_0) - 2f(x_1) + \frac{3}{2}f(x_2)\right] + \frac{h^2}{3}f^{(3)}(\xi_2). \end{aligned}$$

That is

$$\begin{aligned} f'(x_0) &= \frac{1}{h}\left[-\frac{3}{2}f(x_0)+2f(x_0+h)-\frac{1}{2}f(x_0+2h)\right]+\frac{h^2}{3}f^{(3)}(\xi_0), \\ f'(x_0+h) &= \frac{1}{h}\left[-\frac{1}{2}f(x_0)+\frac{1}{2}f(x_0+2h)\right]-\frac{h^2}{6}f^{(3)}(\xi_1), \qquad (4.6) \\ f'(x_0+2h) &= \frac{1}{h}\left[\frac{1}{2}f(x_0)-2f(x_0+h)+\frac{3}{2}f(x_0+2h)\right]+\frac{h^2}{3}f^{(3)}(\xi_2). \qquad (4.7) \end{aligned}$$

Using the variable substitution $x_0$ for $x_0+h$ and $x_0+2h$ in (4.6) and (4.7), respectively, we have

$$\begin{aligned} f'(x_0) &= \frac{1}{2h}[-3f(x_0)+4f(x_0+h)-f(x_0+2h)]+\frac{h^2}{3}f^{(3)}(\xi_0), \qquad (4.8) \\ f'(x_0) &= \frac{1}{2h}[-f(x_0-h)+f(x_0+h)]-\frac{h^2}{6}f^{(3)}(\xi_1), \\ f'(x_0) &= \frac{1}{2h}[f(x_0-2h)-4f(x_0-h)+3f(x_0)]+\frac{h^2}{3}f^{(3)}(\xi_2). \qquad (4.9) \end{aligned}$$

Note that (4.9) can be obtained from (4.8) by replacing $h$ with $-h$.

• **Five-point Formulas**

$$\begin{aligned} f'(x_0) &= \frac{1}{12h}[f(x_0-2h)-8f(x_0-h)+8f(x_0+h)-f(x_0+2h)] \\ &+ \frac{h^4}{30}f^{(5)}(\xi), \end{aligned}$$

where $\xi \in (x_0-2h, x_0+2h)$ and

$$\begin{aligned} f'(x_0) &= \frac{1}{12h}[-25f(x_0)+48f(x_0+h)-36f(x_0+2h) \\ &+ 16f(x_0+3h)-3f(x_0+4h)]+\frac{h^4}{5}f^{(5)}(\xi), \end{aligned}$$

where $\xi \in (x_0, x_0+4h)$.

### 4.2.3 Round-off Error

Consider

$$f'(x_0) = \frac{1}{2h}[-f(x_0-h)+f(x_0+h)] - \frac{h^2}{6}f^{(3)}(\xi_1),$$

where $\frac{h^2}{6}f^{(3)}(\xi_1)$ is called truncation error. Let $\tilde{f}(x_0+h)$ and $\tilde{f}(x_0-h)$ be the computed values of $f(x_0+h)$ and $f(x_0-h)$, respectively. Then

$$f(x_0+h) = \tilde{f}(x_0+h)+e(x_0+h)$$

and

$$f(x_0-h) = \tilde{f}(x_0-h)+e(x_0-h).$$

Therefore, the total error in the approximation

$$f'(x_0) - \frac{\tilde{f}(x_0+h)-\tilde{f}(x_0-h)}{2h} = \frac{e(x_0+h)-e(x_0-h)}{2h} - \frac{h^2}{6}f^{(3)}(\xi_1)$$

is due in part to round-off error and in the part to truncation error.

Assume that

$$|e(x_0 \pm h)| \leq \varepsilon \text{ and } |f^{(3)}(\xi_1)| \leq M.$$

Then

$$\left| f'(x_0) - \frac{\tilde{f}(x_0+h)-\tilde{f}(x_0-h)}{2h} \right| \leq \frac{\varepsilon}{2h} + \frac{h^2}{6}M \equiv e(h).$$

Note that $e(h)$ attains its minimum at $h = \sqrt[3]{3\varepsilon/M}$.

## 4.3 Richardson Extrapolation Method

Suppose $\forall h \neq 0$ we have a formula $N_1(h)$ that approximates an unknown value $M$

$$M - N_1(h) = K_1 h + K_2 h^2 + K_3 h^3 + \cdots, \tag{4.10}$$

for some unknown constants $K_1, K_2, K_3, \ldots$. If $K_1 \neq 0$, then the truncation error is $O(h)$. For example,

$$f'(x) - \frac{f(x+h) - f(x)}{h} = -\frac{f''(x)}{2!}h - \frac{f'''(x)}{3!}h^2 - \frac{f^{(4)}(x)}{4!}h^3 - \cdots.$$

Find an easy way to produce formulas with a higher-order truncation error.

Replacing $h$ in (4.10) by $h/2$, we have

$$M = N_1\left(\frac{h}{2}\right) + K_1\frac{h}{2} + K_2\frac{h^2}{4} + K_3\frac{h^3}{8} + \cdots. \tag{4.11}$$

Subtracting (4.10) with twice (4.11), we get

$$M = N_2(h) - \frac{K_2}{2}h^2 - \frac{3K_3}{4}h^3 - \cdots, \tag{4.12}$$

where

$$N_2(h) = 2N_1\left(\frac{h}{2}\right) - N_1(h) = N_1\left(\frac{h}{2}\right) + \left[N_1\left(\frac{h}{2}\right) - N_1(h)\right],$$

which is an $O(h^2)$ approximation formula.

Replacing $h$ in (4.12) by $h/2$, we get

$$M = N_2\left(\frac{h}{2}\right) - \frac{K_2}{8}h^2 - \frac{3K_3}{32}h^3 - \cdots. \tag{4.13}$$

Subtracting (4.12) from 4 times (4.13) gives

$$3M = 4N_2\left(\frac{h}{2}\right) - N_2(h) + \frac{3K_3}{8}h^3 + \cdots,$$

which implies that

$$M = \left[N_2\left(\frac{h}{2}\right) + \frac{N_2(h/2) - N_2(h)}{3}\right] + \frac{K_3}{8}h^3 + \cdots \equiv N_3(h) + \frac{K_3}{8}h^3 + \cdots .$$

Using induction, $M$ can be approximated by

$$M = N_m(h) + O(h^m),$$

where

$$N_m(h) = N_{m-1}\left(\frac{h}{2}\right) + \frac{N_{m-1}(h/2) - N_{m-1}(h)}{2^{m-1} - 1}.$$

**Centered difference formula**. From the Taylor's theorem

$$\begin{aligned} f(x+h) &= f(x) + hf'(x) + \frac{h^2}{2!}f''(x) + \frac{h^3}{3!}f'''(x) + \frac{h^4}{4!}f^{(4)}(x) + \frac{h^5}{5!}f^{(5)}(x) + \frac{h^6}{6!}f^{(6)}(x) + \cdots \\ f(x-h) &= f(x) - hf'(x) + \frac{h^2}{2!}f''(x) - \frac{h^3}{3!}f'''(x) + \frac{h^4}{4!}f^{(4)}(x) - \frac{h^5}{5!}f^{(5)}(x) + \frac{h^6}{6!}f^{(6)}(x) + \cdots \end{aligned}$$

we have

$$f(x+h) - f(x-h) = 2hf'(x) + \frac{2h^3}{3!}f'''(x) + \frac{2h^5}{5!}f^{(5)}(x) + \cdots ,$$

and, consequently,

$$\begin{aligned} f'(x_0) &= \frac{f(x_0+h) - f(x_0-h)}{2h} - \left[\frac{h^2}{3!}f'''(x_0) + \frac{h^4}{5!}f^{(5)}(x_0) + \cdots\right], \\ &\equiv N_1(h) - \left[\frac{h^2}{3!}f'''(x_0) + \frac{h^4}{5!}f^{(5)}(x_0) + \cdots\right]. \end{aligned} \tag{4.14}$$

Replacing $h$ in (4.14) by $h/2$ gives

$$f'(x_0) = N_1\left(\frac{h}{2}\right) - \frac{h^2}{24}f'''(x_0) - \frac{h^4}{1920}f^{(5)}(x_0) - \cdots . \tag{4.15}$$

Subtracting (4.14) from 4 times (4.15) gives

$$f'(x_0) = N_2(h) + \frac{h^4}{480} f^{(5)}(x_0) + \cdots,$$

where

$$N_2(h) = \frac{1}{3}\left[4N_1\left(\frac{h}{2}\right) - N_1(h)\right] = N_1\left(\frac{h}{2}\right) + \frac{N_1(h/2) - N_1(h)}{3}.$$

In general,

$$f'(x_0) = N_j(h) + O(h^{2j})$$

with

$$N_j(h) = N_{j-1}\left(\frac{h}{2}\right) + \frac{N_{j-1}(h/2) - N_{j-1}(h)}{4^{j-1} - 1}.$$

| $O(h)$ | $O(h^2)$ | $O(h^3)$ | $O(h^4)$ |
|---|---|---|---|
| 1: $N_1(h) = N(h)$ | | | |
| 2: $N_1(h/2) = N(h/2)$ | 3: $N_2(h)$ | | |
| 4: $N_1(h/4) = N(h/4)$ | 5: $N_2(h/2)$ | 6: $N_3(h)$ | |
| 7: $N_1(h/8) = N(h/8)$ | 8: $N_2(h/4)$ | 9: $N_3(h/2)$ | 10: $N_4(h)$ |

■ **Example 4.1** Suppose that $x_0 = 2.0$, $h = 0.2$ and $f(x) = xe^x$. Compute an approximated value of $f'(2.0) = 22.16716829679195$ to six decimal places. ■

**Solution**. By centered difference formula, we have

$$\begin{aligned} N_1(0.2) &= \frac{f(2.0+0.2) - f(2.0-0.2)}{2h} = 22.414160, \\ N_1(0.1) &= \frac{f(2.0+0.1) - f(2.0-0.1)}{h} = 22.228786. \end{aligned}$$

It implies that

$$N_2(0.2) = N_1(0.1) + \frac{N_1(0.1) - N_1(0.2)}{3} = 22.166995$$

which does not have six decimal digits. Adding $N_1(0.05) = 22.182564$, we get

$$N_2(0.1) = N_1(0.05) + \frac{N_1(0.05) - N_1(0.1)}{3} = 22.167157$$

and

$$N_3(0.2) = N_2(0.1) + \frac{N_2(0.1) - N_2(0.2)}{15} = 22.167168$$

which contains six decimal digits.

## 4.4 Elements of Numerical Integration

The basic method involved in approximating the integration

$$\int_a^b f(x)\,dx, \tag{4.16}$$

is called numerical quadrature and uses a sum of the type

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} c_i f(x_i). \tag{4.17}$$

The method of quadrature in this section is based on the polynomial interpolation. We first select a set of distinct nodes $\{x_0, x_1, \ldots, x_n\}$ from the interval $[a,b]$. Then the Lagrange polynomial

$$P_n(x) = \sum_{i=0}^{n} f(x_i)L_i(x) = \sum_{i=0}^{n} f(x_i) \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j}$$

is used to approximate $f(x)$. With the error term we have

$$f(x) = P_n(x) + E_n(x) = \sum_{i=0}^{n} f(x_i)L_i(x) + \frac{f^{(n+1)}(\zeta_x)}{(n+1)!}\prod_{i=0}^{n}(x-x_i),$$

where $\zeta_x \in [a,b]$ and depends on $x$, and

$$\begin{aligned}\int_a^b f(x)\,dx &= \int_a^b P_n(x)\,dx + \int_a^b E_n(x)\,dx \\ &= \sum_{i=0}^{n} f(x_i)\int_a^b L_i(x)\,dx + \frac{1}{(n+1)!}\int_a^b f^{(n+1)}(\zeta_x)\prod_{i=0}^{n}(x-x_i)\,dx. \quad (4.18)\end{aligned}$$

The quadrature formula is, therefore,

$$\int_a^b f(x)\,dx \approx \int_a^b P_n(x)\,dx = \sum_{i=0}^{n} f(x_i)\int_a^b L_i(x)\,dx \equiv \sum_{i=0}^{n} c_i f(x_i), \quad (4.19)$$

where

$$c_i = \int_a^b L_i(x)\,dx = \int_a^b \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x-x_j}{x_i-x_j}\,dx. \quad (4.20)$$

Moreover, the error in the quadrature formula is given by

$$E = \frac{1}{(n+1)!}\int_a^b f^{(n+1)}(\zeta_x)\prod_{i=0}^{n}(x-x_i)\,dx, \quad (4.21)$$

for some $\zeta_x \in [a,b]$.

Let us consider formulas produced by using first and second Lagrange polynomials with equally spaced nodes. This gives the Trapezoidal rule and Simpson's rule, respectively.

**Trapezoidal rule**: Let $x_0 = a, x_1 = b, h = b-a$ and use the linear Lagrange polynomial:

$$P_1(x) = \frac{(x-x_1)}{(x_0-x_1)}f(x_0) + \frac{(x-x_0)}{(x_1-x_0)}f(x_1).$$

Then

$$\begin{aligned}\int_a^b f(x)dx &= \int_{x_0}^{x_1}\left[\frac{(x-x_1)}{(x_0-x_1)}f(x_0)+\frac{(x-x_0)}{(x_1-x_0)}f(x_1)\right]dx \\ &\quad +\frac{1}{2}\int_{x_0}^{x_1} f''(\zeta(x))(x-x_0)(x-x_1)dx. \end{aligned} \tag{4.22}$$

**Theorem 4.4.1 — Weighted Mean Value Theorem for Integrals.** Suppose $f \in C[a,b]$, the Riemann integral of $g(x)$

$$\int_a^b g(x)dx = \lim_{\max \triangle x_i \to 0}\sum_{i=1}^{n} g(x_i)\triangle x_i,$$

exists and $g(x)$ does not change sign on $[a,b]$. Then $\exists\, c \in (a,b)$ with

$$\int_a^b f(x)g(x)dx = f(c)\int_a^b g(x)dx.$$

Since $(x-x_0)(x-x_1)$ does not change sign on $[x_0,x_1]$, by the Weighted Mean Value Theorem, $\exists\, \zeta \in (x_0,x_1)$ such that

$$\begin{aligned}&\int_{x_0}^{x_1} f''(\zeta(x))(x-x_0)(x-x_1)dx = f''(\zeta)\int_{x_0}^{x_1}(x-x_0)(x-x_1)dx \\ &= f''(\zeta)\left[\frac{x^3}{3}-\frac{x_1+x_0}{2}x^2+x_0x_1x\right]_{x_0}^{x_1} = -\frac{h^3}{6}f''(\zeta).\end{aligned}$$

Consequently, Eq. (4.22) implies that

$$\begin{aligned}\int_a^b f(x)dx &= \left[\frac{(x-x_1)^2}{2(x_0-x_1)}f(x_0)+\frac{(x-x_0)^2}{2(x_1-x_0)}f(x_1)\right]_{x_0}^{x_1} - \frac{h^3}{12}f''(\zeta) \\ &= \frac{x_1-x_0}{2}\left[f(x_0)+f(x_1)\right]-\frac{h^3}{12}f''(\zeta) \\ &= \frac{h}{2}\left[f(x_0)+f(x_1)\right]-\frac{h^3}{12}f''(\zeta),\end{aligned}$$

which is called the Trapezoidal rule.

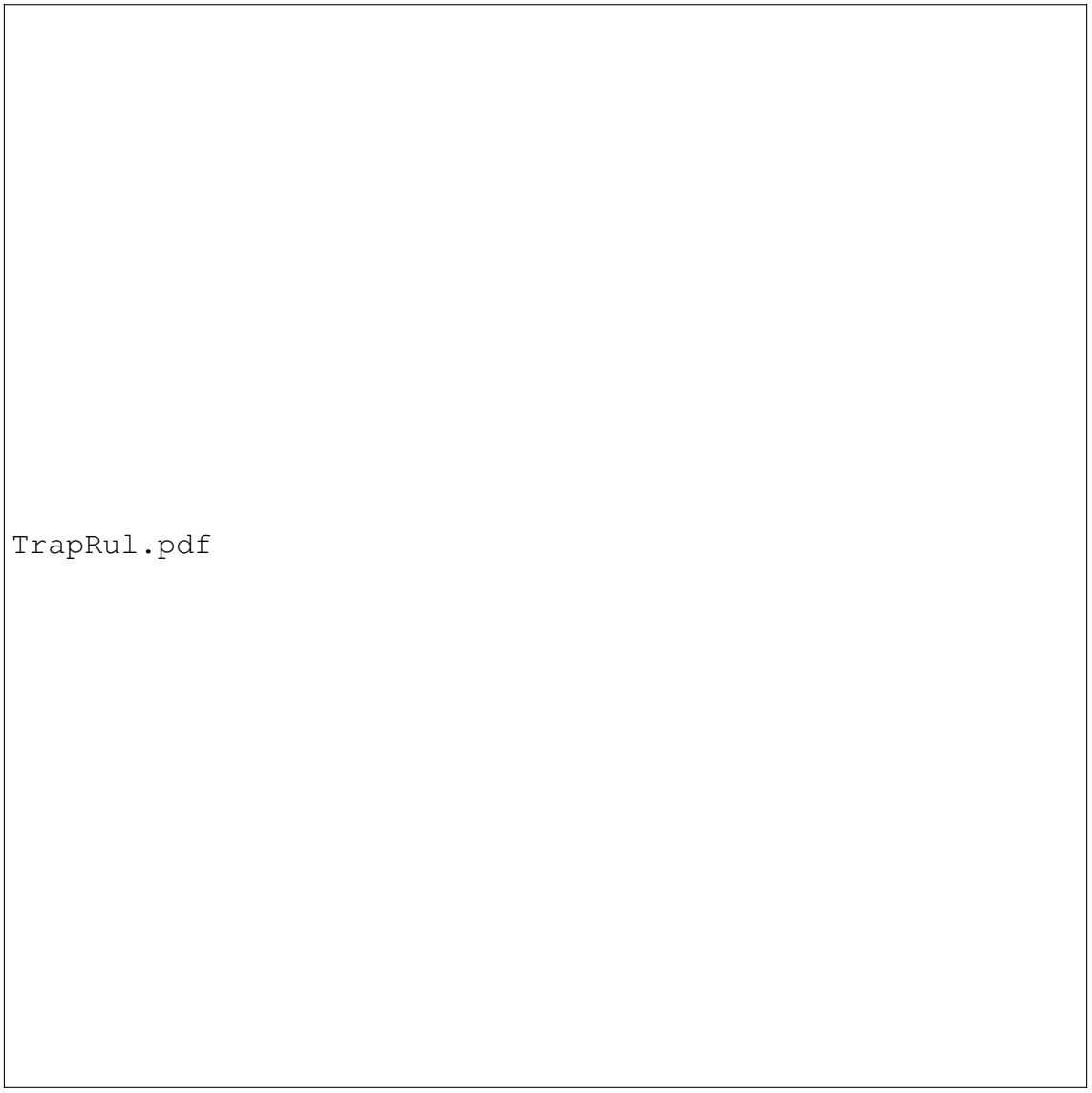



Figure 4.1: Existence and uniqueness of fixed point

If we choose $x_0 = a$, $x_1 = \frac{1}{2}(a+b)$, $x_2 = b$, $h = (b-a)/2$, and the second order Lagrange polynomial

$$\begin{aligned} P_2(x) &= f(x_0)\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + f(x_1)\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} \\ &\quad + f(x_2)\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} \end{aligned}$$

to interpolate $f(x)$, then

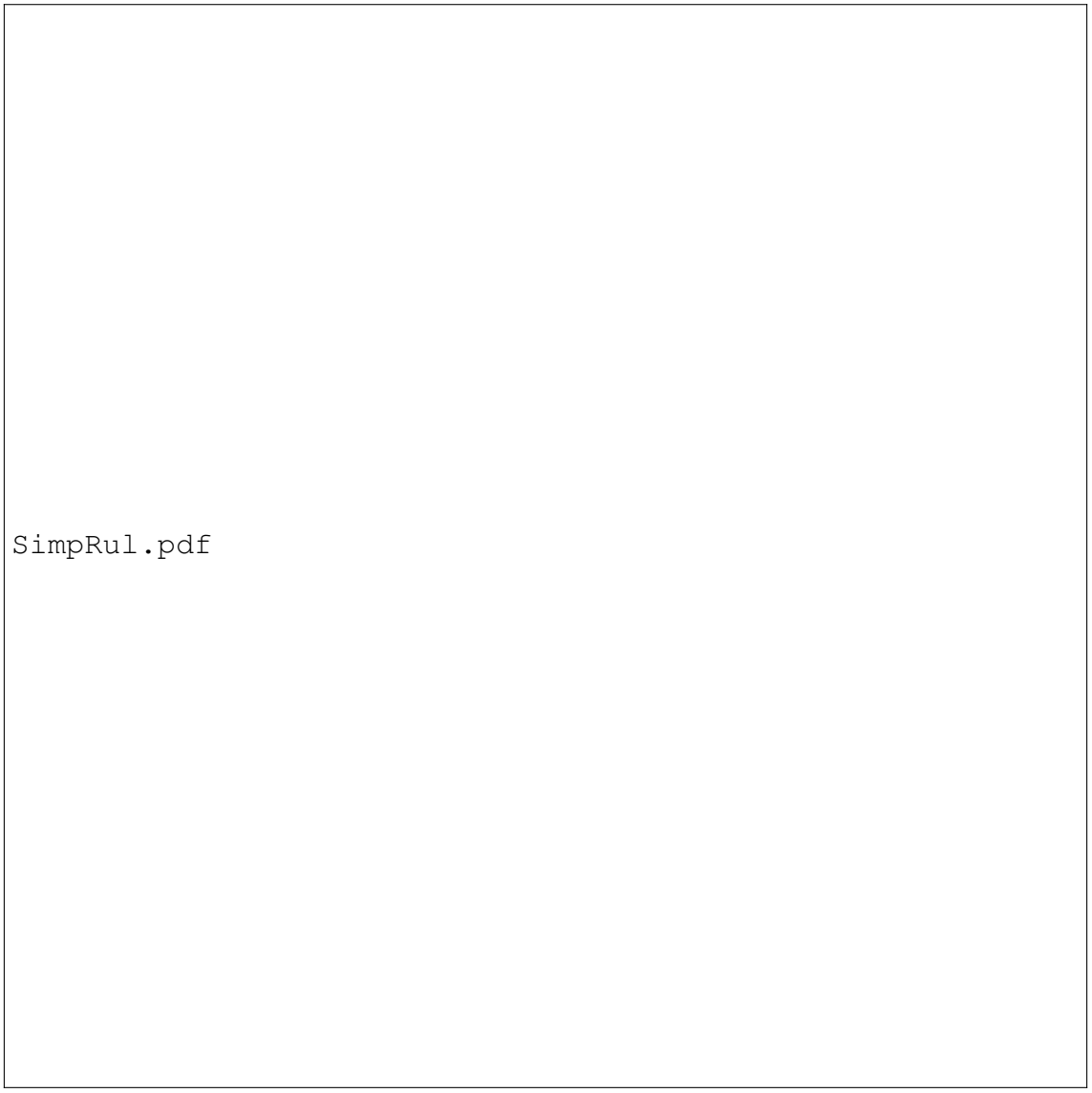



Figure 4.2: Existence and uniqueness of fixed point

$$\begin{aligned}\int_a^b f(x)dx &= \int_{x_0}^{x_2}\left[\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}f(x_0)+\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}f(x_1)\right.\\ &\quad\left.+\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}f(x_2)\right]dx\\ &\quad+\int_{x_0}^{x_2}\frac{(x-x_0)(x-x_1)(x-x_2)}{6}f^{(3)}(\zeta(x))dx.\end{aligned}$$

Since, letting $x = x_0 + th$,

$$\begin{aligned}
\int_{x_0}^{x_2} \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}dx &= h\int_0^2 \frac{t-1}{0-1}\cdot\frac{t-2}{0-2}\,dt \\
&= \frac{h}{2}\int_0^2 (t^2-3t+2)\,dt = \frac{h}{3}, \\
\int_{x_0}^{x_2} \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}dx &= h\int_0^2 \frac{t-0}{1-0}\cdot\frac{t-2}{1-2}\,dt \\
&= -h\int_0^2 (t^2-2t)\,dt = \frac{4h}{3},
\end{aligned}$$

$$\int_{x_0}^{x_2} \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}dx = h\int_0^2 \frac{t-0}{2-0}\cdot\frac{t-1}{2-1}\,dt\frac{h}{2}\int_0^2 (t^2-t)\,dt = \frac{h}{3},$$

it implies that

$$\begin{aligned}
\int_a^b f(x)dx &= h\left[\frac{1}{3}f(x_0)+\frac{4}{3}f(x_1)+\frac{1}{3}f(x_2)\right] \\
&\quad + \int_{x_0}^{x_2} \frac{(x-x_0)(x-x_1)(x-x_2)}{6} f^{(3)}(\zeta(x))dx,
\end{aligned}$$

which is called the Simpson's rule and provides only an $O(h^4)$ error term involving $f^{(3)}$. A higher order error analysis can be derived by expanding $f$ in the third Taylor's formula about $x_1$. $\forall\, x \in [a,b]$, $\exists\, \zeta_x \in (a,b)$ such that

$$\begin{aligned}
f(x) &= f(x_1)+f'(x_1)(x-x_1)+\frac{f''(x_1)}{2}(x-x_1)^2 \\
&\quad + \frac{f'''(x_1)}{6}(x-x_1)^3 + \frac{f^{(4)}(\zeta_x)}{24}(x-x_1)^4.
\end{aligned}$$

Then

$$\begin{aligned}
\int_a^b f(x)\,dx &= \left[f(x_1)(x-x_1)+\frac{f'(x_1)}{2}(x-x_1)^2+\frac{f''(x_1)}{6}(x-x_1)^3\right. \\
&\quad \left. + \frac{f'''(x_1)}{24}(x-x_1)^4\right]\Bigg|_a^b + \frac{1}{24}\int_a^b f^{(4)}(\zeta_x)(x-x_1)^4\,dx.
\end{aligned}$$

Note that $(b-x_1) = h$, $(a-x_1) = -h$, and since $(x-x_1)^4$ does not change sign in $[a,b]$,

by the Weighted Mean-Value Theorem for Integral, there exists $\xi_1 \in (a,b)$ such that

$$\int_a^b f^{(4)}(\zeta_x)(x-x_1)^4\,dx = f^{(4)}(\xi_1)\int_a^b (x-x_1)^4\,dx = \frac{2f^{(4)}(\xi_1)}{5}h^5.$$

Consequently,

$$\int_a^b f(x)\,dx = 2f(x_1)h + \frac{f''(x_1)}{3}h^3 + \frac{f^{(4)}(\xi_1)}{60}h^5.$$

Finally we replace $f''(x_1)$ by the central finite difference formulation $f''(x_1) = \frac{f(x_0)-2f(x_1)+f(x_2)}{h^2} - \frac{f^{(4)}(\xi_2)}{12}h^2$,

for some $\xi_2 \in (a,b)$, to obtain

$$\begin{aligned}\int_a^b f(x)\,dx &= 2hf(x_1)+\frac{h}{3}\left(f(x_0)-2f(x_1)+f(x_2)\right)\\ &\quad -\frac{f^{(4)}(\xi_2)}{36}h^5+\frac{f^{(4)}(\xi_1)}{60}h^5\\ &= h\left[\frac{1}{3}f(x_0)+\frac{4}{3}f(x_1)+\frac{1}{3}f(x_2)\right]\\ &\quad +\frac{1}{90}\left[\frac{3}{2}f^{(4)}(\xi_1)-\frac{5}{2}f^{(4)}(\xi_2)\right]h^5.\end{aligned}$$

It can show that there exists $\xi \in (a,b)$ such that

$$\int_a^b f(x)\,dx = \frac{h}{3}\left[f(x_0)+4f(x_1)+f(x_2)\right]-\frac{f^{(4)}(\xi)}{90}h^5.$$

This gives the Simpson's rule formulation.

**Definition 4.4.1** The degree of accuracy, or precision, of a quadrature formula is the largest positive integer $n$ such that the formula is exact for $x^k$, when $k = 0,1,\ldots,n$.

- The Trapezoidal and Simpson's rules have degrees of precision one and three, respectively.
- The degree of accuracy of a quadrature formula is $n$ if and only if the error $E = 0$ for all polynomials $P(x)$ of degree less than or equal to $n$, but $E \neq 0$ for some polynomials of degree greater than $n$.

### 4.4.1 Newton-Cotes Formulas

**Definition 4.4.2 — Newton-Cotes formula.** A quadrature formula of the form

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} c_i f(x_i)$$

is called a Newton-Cotes formula if the nodes $\{x_0,x_1,\ldots,x_n\}$ are equally spaced.

Consider a uniform partition of the closed interval $[a,b]$ by

$$x_i = a+ih, \quad i = 0,1,\ldots,n, \quad h = \frac{b-a}{n},$$

where $n$ is a positive integer and $h$ is called the step length.

By introduction a new variable $t$ such that $x = a + ht$, the fundamental Lagrange polynomial becomes

$$L_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j} = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{a + ht - a - jh}{a + ih - a - jh} = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{t - j}{i - j} \equiv \varphi_i(t).$$

Therefore, the integration (4.20) gives

$$c_i = \int_a^b L_i(x)\,dx = \int_0^n \varphi_i(t) h\,dt = h \int_0^n \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{t - j}{i - j}\,dt, \tag{4.23}$$

and the general Newton-Cotes formula has the form

$$\int_a^b f(x)\,dx = h \sum_{i=0}^{n} f(x_i) \int_0^n \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{t - j}{i - j}\,dt + \frac{1}{(n+1)!} \int_a^b f^{(n+1)}(\zeta_x) \prod_{i=0}^{n} (x - x_i)\,dx. \tag{4.24}$$

**Theorem 4.4.2 — Closed Newton-Cotes Formulas.** Suppose that $\sum_{i=0}^{n} a_i f(x_i)$ denotes the $(n+1)$-point closed Newton-Cotes formula with $x_0 = a, x_n = b$ and $h = (b-a)/n$. If $n$ is even and $f \in C^{n+2}[a,b]$, then

$$\int_a^b f(x)\,dx = h \sum_{i=0}^{n} \alpha_i f(x_i) + \frac{h^{n+3} f^{(n+2)}(\xi)}{(n+2)!} \int_0^n t^2 (t-1) \cdots (t-n)\,dt, \tag{4.25}$$

and if $n$ is odd and $f \in C^{n+1}[a,b]$, then

$$\int_a^b f(x)\,dx = h \sum_{i=0}^{n} \alpha_i f(x_i) + \frac{h^{n+2} f^{(n+1)}(\xi)}{(n+1)!} \int_0^n t (t-1) \cdots (t-n)\,dt, \tag{4.26}$$

where $\xi \in (a,b)$ and $\alpha_i = \int_0^n \prod_{j=0,\ j\neq i}^{n} \frac{t-j}{i-j}\,dt$ for $i = 0, 1, \ldots, n$. Consequently, the degree of accuracy is $n+1$ when $n$ is an even integer, and $n$ when $n$ is an odd integer.

- $n = 1$: Trapezoidal rule

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\,[f(a)+f(b)] - \frac{h^3}{12}f''(\xi),\ a < \xi < b.$$

- $n = 2$: Simpson's rule

$$\int_a^b f(x)\,dx = h\left[\frac{1}{3}f(x_0) + \frac{4}{3}f(x_1) + \frac{1}{3}f(x_2)\right] - \frac{f^{(4)}(\xi)}{90}h^5,\ a < \xi < b.$$

- The error term of the Trapezoidal rule is $O(h^3)$.
- Since the rule involves $f''$, it gives the exact result when applied to any function whose second derivative is identically zero, e.g., any polynomial of degree 1 or less.
- The degree of accuracy of Trapezoidal rule is one.
- The Simpson's rule is an $O(h^5)$ scheme and the degree of accuracy is three.

Another class of Newton-Cotes formulas is the open Newton-Cotes formulas in which the nodes

$$x_i = x_0 + ih, \quad i = 0, 1, \ldots, n,$$

where

$$x_0 = a + h \quad \text{and} \quad h = \frac{b-a}{n+2},$$

are used. This implies that $x_n = b - h$, and the endpoints, $a$ and $b$, are not used. Hence we label $a = x_{-1}$ and $b = x_{n+1}$. The formulas become

$$\int_a^b f(x)dx = \int_{x_{-1}}^{x_{n+1}} f(x)dx \approx \sum_{i=0}^{n} a_i f(x_i),$$

where

$$a_i = \int_a^b L_i(x)dx.$$

The following theorem summarizes the open Newton-Cotes formulas.

**Theorem 4.4.3 — Open Newton-Cotes Formulas.** Suppose that $\sum_{i=0}^{n} a_i f(x_i)$ denotes the $(n+1)$-point open Newton-Cotes formula with $x_{-1} = a$, $x_{n+1} = b$ and $h = (b-a)/(n+2)$. If $n$ is even and $f \in C^{n+2}[a,b]$, then

$$\int_a^b f(x)\,dx = h\sum_{i=0}^{n} \alpha_i f(x_i) + \frac{h^{n+3} f^{(n+2)}(\xi)}{(n+2)!}\int_{-1}^{n+1} t^2(t-1)\cdots(t-n)\,dt, \tag{4.27}$$

and if $n$ is odd and $f \in C^{n+1}[a,b]$, then

$$\int_a^b f(x)\,dx = h\sum_{i=0}^{n} \alpha_i f(x_i) + \frac{h^{n+2} f^{(n+1)}(\xi)}{(n+1)!}\int_{-1}^{n+1} t(t-1)\cdots(t-n)\,dt, \tag{4.28}$$

where $\xi \in (a,b)$ and $\alpha_i = \int_{-1}^{n+1} \prod_{j=0, j\neq i}^{n} \frac{t-j}{i-j}\,dt$ for $i = 0,1,\ldots,n$. Consequently, the degree of accuracy is $n+1$ when $n$ is an even integer, and $n$ when $n$ is an odd integer.

The simplest open Newton-Cotes formula is choosing $n = 0$ and only using the midpoint $x_0 = \frac{a+b}{2}$. Then the coefficient and the error term can be computed easily as

$$\alpha_0 = \int_1^{-1} dt = 2, \quad \text{and} \quad \frac{h^3 f''(\xi)}{2!}\int_{-1}^{1} t^2\,dt = \frac{1}{3}f''(\xi)h^3.$$

These gives the so-called Midpoint rule or Rectangular rule.

Midpoint Rule:

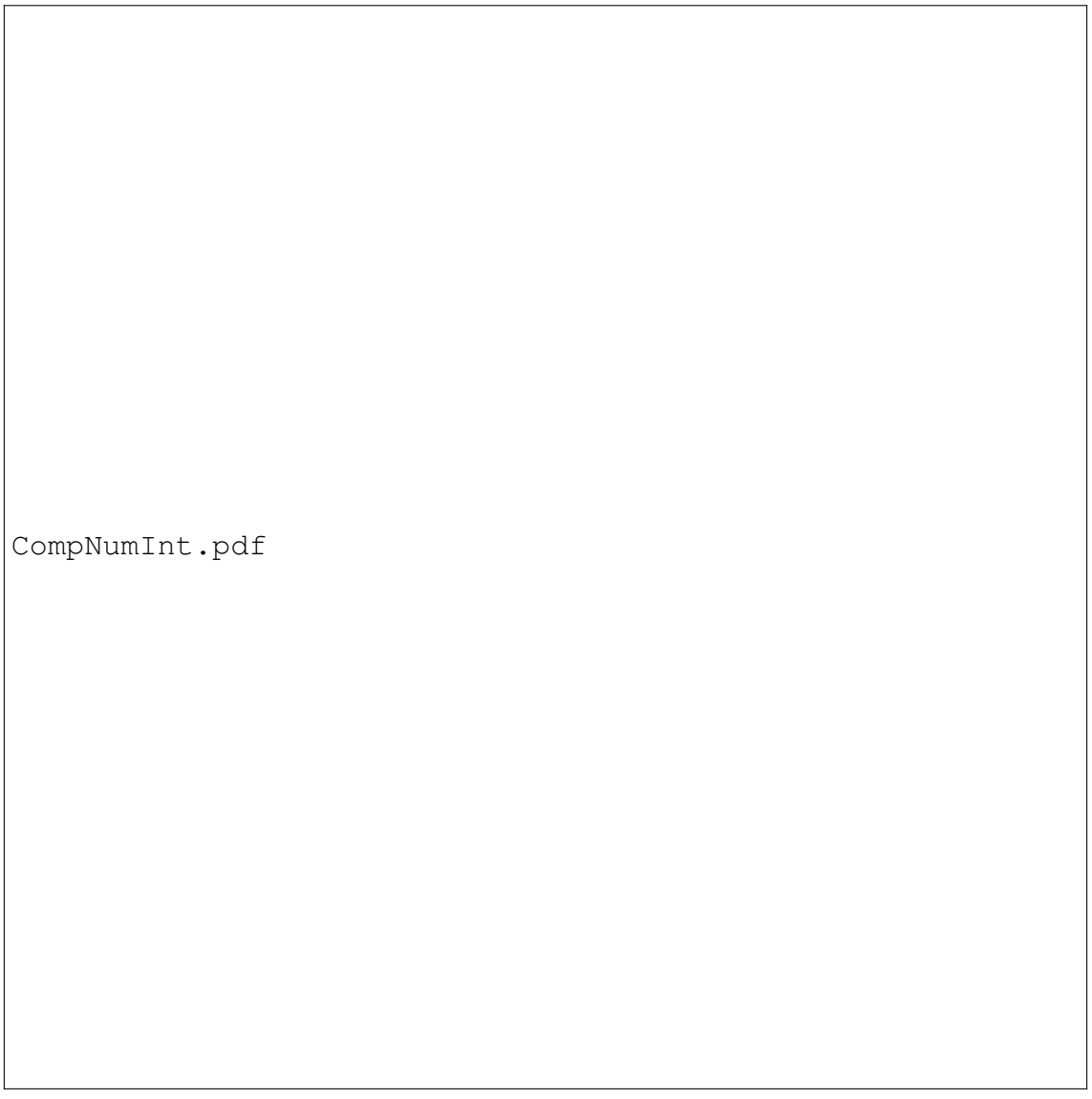
$$\int_a^b f(x)\,dx = 2hf(x_0) + \frac{1}{3}f''(\xi)h^3 = (b-a)f(\frac{a+b}{2}) + \frac{1}{3}f''(\xi)h^3, \tag{4.29}$$

for some $\xi \in (a,b)$.

## 4.5 Composite Numerical Integration

- The Newton-Cotes formulas are generally not suitable for numerical integration over large interval. Higher degree formulas would be required, and the coefficients in these formulas are difficult to obtain.

- Also the Newton-Cotes formulas which are based on polynomial interpolation would be inaccurate over a large interval because of the oscillatory nature of high-degree polynomials.
- Now we discuss a piecewise approach, called composite rule, to numerical integration over large interval that uses the low-order Newton-Cotes formulas.
  - A composite rule is one obtained by applying an integration formula for a single interval to each subinterval of a partitioned interval.




To illustrate the procedure, we choose an even integer $n$ and partition the interval $[a,b]$ into $n$ subintervals by nodes $a = x_0 < x_1 < \cdots < x_n = b$, and apply Simpson's rule on each

consecutive pair of subintervals. With

$$h = \frac{b-a}{n} \quad \text{and} \quad x_j = a + jh, \quad j = 0, 1, \ldots, n,$$

we have on each interval $[x_{2j-2}, x_{2j}]$,

$$\int_{x_{2j-2}}^{x_{2j}} f(x)\,dx = \frac{h}{3}\left[f(x_{2j-2}) + 4f(x_{2j-1}) + f(x_{2j})\right] - \frac{h^5}{90} f^{(4)}(\xi_j),$$

for some $\xi_j \in (x_{2j-2}, x_{2j})$, provided that $f \in C^4[a,b]$.

The composite rule is obtained by summing up over the entire interval, that is,

$$\begin{aligned}
\int_a^b f(x)\,dx &= \sum_{j=1}^{n/2} \int_{x_{2j-2}}^{x_{2j}} f(x)\,dx \\
&= \sum_{j=1}^{n/2} \left[\frac{h}{3}\left(f(x_{2j-2}) + 4f(x_{2j-1}) + f(x_{2j})\right) - \frac{h^5}{90} f^{(4)}(\xi_j)\right] \\
&= \frac{h}{3}[f(x_0) + 4f(x_1) + f(x_2) \\
&\quad + f(x_2) + 4f(x_3) + f(x_4) \\
&\quad + f(x_4) + 4f(x_5) + f(x_6) \\
&\quad \vdots \\
&\quad + f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)] - \frac{h^5}{90} \sum_{j=1}^{n/2} f^{(4)}(\xi_j)
\end{aligned}$$

Hence

$$\begin{aligned}
\int_a^b f(x)\,dx &= \frac{h}{3}[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + 4f(x_5) \\
&\quad + \cdots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)] - \frac{h^5}{90} \sum_{j=1}^{n/2} f^{(4)}(\xi_j) \\
&= \frac{h}{3}\left[f(x_0) + 4\sum_{j=1}^{n/2} f(x_{2j-1}) + 2\sum_{j=1}^{(n/2)-1} f(x_{2j}) + f(x_n)\right] \\
&\quad - \frac{h^5}{90} \sum_{j=1}^{n/2} f^{(4)}(\xi_j).
\end{aligned}$$

To estimate the error associated with approximation, since $f \in C^4[a,b]$, we have, by the Extreme Value Theorem,

$$\min_{x\in[a,b]} f^{(4)}(x) \le f^{(4)}(\xi_j) \le \max_{x\in[a,b]} f^{(4)}(x),$$

for each $\xi_j \in (x_{2j-2}, x_{2j})$.

Hence

$$\frac{n}{2}\min_{x\in[a,b]} f^{(4)}(x) \le \sum_{j=1}^{n/2} f^{(4)}(\xi_j) \le \frac{n}{2}\max_{x\in[a,b]} f^{(4)}(x),$$

and

$$\min_{x\in[a,b]} f^{(4)}(x) \le \frac{2}{n}\sum_{j=1}^{n/2} f^{(4)}(\xi_j) \le \max_{x\in[a,b]} f^{(4)}(x).$$

By the Intermediate Value Theorem, there exists $\mu \in (a,b)$ such that

$$f^{(4)}(\mu) = \frac{2}{n}\sum_{j=1}^{n/2} f^{(4)}(\xi_j).$$

Thus, by replacing $n = (b-a)/h$,

$$\sum_{j=1}^{n/2} f^{(4)}(\xi_j) = \frac{n}{2} f^{(4)}(\mu) = \frac{b-a}{2h} f^{(4)}(\mu).$$

Consequently, the composite Simpson's rule is derived.

**Composite Simpson's Rule**

$$\begin{aligned}\int_a^b f(x)\,dx &= \frac{h}{3}\left[f(a) + 4\sum_{j=1}^{n/2} f(x_{2j-1}) + 2\sum_{j=1}^{(n/2)-1} f(x_{2j}) + f(b)\right] \\ &\quad -\frac{b-a}{180} f^{(4)}(\mu) h^4,\end{aligned}$$

where $n$ is an even integer, $h = (b-a)/n$, $x_j = a + jh$, for $j = 0, 1, \ldots, n$, and some $\mu \in (a,b)$.

The composite Midpoint rule can be derived in a similar way, except the midpoint rule is applied on each subinterval $[x_{2j-1}, x_{2j+1}]$ instead. That is,

$$\int_{x_{2j-1}}^{x_{2j+1}} f(x)\,dx = 2hf(x_{2j}) + \frac{h^3}{3} f''(\xi_j), \qquad j = 1, 2, \ldots, \frac{n}{2}.$$

Note that $n$ must again be even. Consequently,

$$\int_a^b f(x)\,dx = 2h \sum_{j=1}^{n/2} f(x_{2j}) + \frac{h^3}{3} \sum_{j=1}^{n/2} f''(\xi_j).$$

CompTrapRule.pdf

The error term can be written as

$$\sum_{j=1}^{n/2} f''(\xi_j) = \frac{n}{2} f''(\mu) = \frac{b-a}{2h} f''(\mu),$$

for some $\mu \in (a,b)$.

**Composite Midpoint Rule**

$$\int_a^b f(x)\,dx = 2h \sum_{j=1}^{n/2} f(x_{2j}) + \frac{b-a}{6} f''(\mu) h^2, \tag{4.30}$$

where $n$ is an even integer, $h = (b-a)/n$, $x_j = a + jh$, for $j = 0, 1, \ldots, n$, and some $\mu \in (a,b)$.

To derive the composite Trapezoidal rule, we partition $[a,b]$ by $n$ equally spaced nodes $a = x_0 < x_1 < \cdots < x_n = b$, where $n$ can be either odd or even. Apply the trapezoidal rule on $[x_{j-1}, x_j]$ and sum them up to obtain

$$\begin{aligned}
\int_a^b f(x)\,dx &= \sum_{j=1}^{n} \int_{x_{j-1}}^{x_j} f(x)\,dx \\
&= \sum_{j=1}^{n} \left\{ \frac{h}{2} \left[ f(x_{j-1}) + f(x_j) \right] - \frac{h^3}{12} f''(\xi_j) \right\} \\
&= \frac{h}{2} \{ [f(x_0) + f(x_1)] + [f(x_1) + f(x_2)] + \cdots \\
&\quad + [f(x_{n-1}) + f(x_n)] \} - \frac{h^3}{12} \sum_{j=1}^{n} f''(\xi_j)
\end{aligned}$$

Hence,

$$\begin{aligned}
\int_a^b f(x)\,dx &= \frac{h}{2} [f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(x_n)] \\
&\quad - \frac{h^3}{12} \sum_{j=1}^{n} f''(\xi_j) \\
&= \frac{h}{2} \left[ f(a) + 2 \sum_{j=1}^{n-1} f(x_j) + f(b) \right] - \frac{h^3}{12} \sum_{j=1}^{n} f''(\xi_j) \\
&= \frac{h}{2} \left[ f(a) + 2 \sum_{j=1}^{n-1} f(x_j) + f(b) \right] - \frac{b-a}{12} f''(\mu) h^2,
\end{aligned}$$

CompSimpRule.pdf

where each $\xi_j \in (x_{j-1}, x_j)$ and $\mu \in (a,b)$.

**Composite Trapezoidal Rule**

$$\int_a^b f(x)\,dx = \frac{h}{2}\left[f(a) + 2\sum_{j=1}^{n-1} f(x_j) + f(b)\right] - \frac{b-a}{12} f''(\mu)h^2, \tag{4.31}$$

where $n$ is an integer, $h = (b-a)/n$, $x_j = a + jh$, for $j = 0, 1, \ldots, n$, and some $\mu \in (a,b)$.

## 4.6 Romberg Integration

Recall the trapezoidal rule integral formulation

$$\begin{aligned}\int_a^b f(x)\,dx &\approx T(n)\\ &= \frac{h}{2}[f(a)+2f(a+h)+2f(a+2h)+\cdots\\ &\quad +2f(a+(n-1)h)+f(a+nh)],\end{aligned}$$

where $h=\frac{b-a}{n}$. Let $n=2$ and $h=\frac{b-a}{2n}$.

• If we only consider the partitions by $x_0, x_2$, and $x_4$, and apply the Trapezoidal rule, we have the approximation

$$T(n)=\frac{2h}{2}[f(a)+2f(a+2h)+f(a+4h)]=h[f(a)+2f(a+2h)+f(a+4h)].$$

• If we apply Trapezoidal rule by $x_0, x_1, x_2, x_3$, and $x_4$, then we have

$$\begin{aligned}T(2n) &= \frac{h}{2}[f(a)+2f(a+h)+2f(a+2h)+2f(a+3h)+f(a+4h)]\\ &= \frac{1}{2}T(n)+h[f(a+h)+f(a+3h)].\end{aligned}$$

This observation shows that

- if we have computed $T(n)$ and the step size is half, we don't have to compute all the function values all over again, but just at those newly added points.
- In general, suppose $T(n)$ has been computed and the step size becomes $h=\frac{b-a}{2n}$, then

$$T(2n)=\frac{1}{2}T(n)+h\sum_{i=1}^{n} f(a+(2i-1)h). \tag{4.32}$$

With this idea in mind, we can apply the trapezoidal rule recursively, i.e., we partition the interval $[a,b]$ into $2^n$ subintervals with $n=1,2,3,\ldots,$

romberg.pdf

and the integral formulation becomes

$$T(2^n) = \frac{1}{2}T(2^{n-1}) + \frac{b-a}{2^n}\sum_{i=1}^{2^{n-1}} f(a+(2i-1)\frac{b-a}{2^n}). \tag{4.33}$$

Let

$$R_{k,1} = T(2^k).$$

If $f \in C^{\infty}[a,b]$, then

$$\int_a^b f(x)dx - R_{k,1} = K_1 h_k^2 + \sum_{i=2}^{\infty} K_i h_k^{2^i}, \tag{4.34}$$

where each $K_i$ is independent of $h_k$ and depends only on $f^{(2i-1)}(a)$ and $f^{(2i-1)}(b)$. Replacing $h_k$ with $h_{k+1} = h_k/2$, we get

$$\int_a^b f(x)dx - R_{k+1,1} = \frac{K_1 h_k^2}{4} + \sum_{i=2}^{\infty} \frac{K_i h_k^{2^i}}{4^i}. \tag{4.35}$$

Subtracting Eq. (4.34) from 4 times Eq. (4.35), we have

$$\int_a^b f(x)dx - \left[R_{k+1,1} + \frac{R_{k+1,1} - R_{k,1}}{3}\right] = -\frac{K_2}{4} h_k^4 + \sum_{i=3}^{\infty} \frac{K_i}{3} \left(\frac{1-4^{i-1}}{4^{i-1}}\right) h_k^{2i}.$$

Define

$$R_{k,2} = R_{k,1} + \frac{R_{k,1} - R_{k-1,1}}{3}, \quad \text{for } k = 2,3,\ldots,n.$$

Apply the extrapolation procedure to those values. Continuing this notation, we have, for each $k = 2,3,4,\ldots,n$ and $j = 2,\ldots,k$, an $O(h_k^{2j})$ approximation formula defined by

$$R_{k,j} = R_{k,j-1} + \frac{R_{k,j-1} - R_{k-1,j-1}}{4^{j-1} - 1}.$$

This is called the Romberg algorithm.

**Theorem 4.6.1** If $f \in C[a,b]$, then for each column $k$,

$$\lim_{n\to\infty} R(n,k) = \int_a^b f(x)\,dx. \tag{4.36}$$

Moreover, if $f \in C^{2m}[a,b]$, then $R(n,m)$ converges to $\int_a^b f(x)\,dx$ with a rate of $O(h^{2m})$, where $h = \frac{b-a}{2^n}$.