Overview

• Preliminaries (Ch 1) • Solving nonlinear equation (Ch 3) • Interpolation (Ch 4) • Integration (Ch 5)

• Solving linear system (Ch 2+8) • Initial value problem (Ch 7)

Nonlinear equations

• Root-finding problem: find x s.t. f(x)=0

• Methods to be studied – Bisection method (3.1) – Newton’s method (3.2) – Secant method (3.3)

• Convergence analysis • Comparison and discussions

Interpolation (Phép nội suy)

. A special case: super-resolution

• Polynomial interpolation

• Error analysis

• Data fitting → machine learning

Chart, scatter chart

Description automatically generated

• Riemann sum

• Methods to be studied – Trapezoid method (5.1)

– Simpson’s rules (5.3) – Gaussian Quadrature formula (5.4)

• Error analysis

Solving Ax = b

• Gaussian elimination (2.1) • Pivoting (2.2) • Tridiagonal and banded system (2.3)

• Matrix factorization (8.1) • Singular Value Decomposition (8.2) • Power method (8.3) • Iterative method (8.4)

Initial value problem

• Problem Text, letter

Description automatically generated

Do you have ODE background?

• Methods – Taylor series method (7.1) – Runge-Kutta method (7.2)

Table

Description automatically generated with medium confidence

**Chapter 1.1: MATHEMATICAL PRELIMINARIES**

Significant digits of precision

• Non-zero digits within given measurements are significant.

• **Zeros to the right of the last non-zero digit are significant if within the measurement**.

• Zeros to the left of the first nonzero digit are NOT significant.

• An exact number has an infinite number of significant digits (or figures).



. Data thought to be accurate should be carried with full precision and not be rounded prior to each of the calculation.

Errors

Text

Description automatically generated with medium confidence

• In computing **absolute error**, the roles of these two values are the same, whereas in relative errors it is essential to distinguish the correct one.

• In practice, relative error is more meaningful. Sometimes, **percentage error** is used.

• The **exact value** may be the **true value** or the **best-known value**.

Rounding and chopping

• Rounding reduces the number of significant digits in a number.

• The result of rounding is a number similar in magnitude that is a shorter number having fewer nonzero digits.

• The round-to-even rule tends to reduce the total rounding error with (on average) an equal portion of numbers rounding up and down.

• Compared to chopping, rounding is preferrable.

Nested multiplication

. To evaluate the polynomial



. we group the terms in a nested multiplication:



**The pseudocode** that evaluates p(x) starts with the innermost parentheses and works out-ward. It can be written as

**integer** i, n; real p, x; **real array** (ai)0:n

p ← an

**for** i = n − 1 **to** 0 **do** p ← ai + x\*p

**end for**

Review theoretical tools for numerical analysis:

• Taylor series/Theorem • Ratio test • Alternating

Theorem Computational tools:

• Horner’s algorithm

Taylor series

A picture containing text

Description automatically generated

Use Taylor for computation

• Use the Taylor series for the natural logarithm

A picture containing clock, watch

Description automatically generated

• With 𝑥 = 1

Chart, box and whisker chart

Description automatically generated

• Add the eight terms

Chart, scatter chart

Description automatically generated

Alternative

• Use a different Taylor series

A picture containing text, clock, watch

Description automatically generated

• With 𝑥 = 1/3

A picture containing text, watch, clock, gauge

Description automatically generated

• Add the four terms and multiply by 2

A screenshot of a computer

Description automatically generated with low confidence

Take-home message

Fast convergence of a Taylor series can be expected near the point of expansion.

• Taylor series for 𝑓(𝑥) at a point c

Text

Description automatically generated with medium confidence

• Maclaurin series if 𝑐 = 0.

• How to compute? Horner’s algorithm.

Deflation (Giảm phát)

Given a polynomial



We Have: A picture containing watch

Description automatically generated

Recall that if n <= m, we write A picture containing antenna

Description automatically generated

And Text

Description automatically generated

By convention, whenever m < n, we define A picture containing logo

Description automatically generated

Horner’s algorithm can be used in the deflation of a polynomial. This is the process of removing a linear factor from a polynomial. If r is a root of the polynomial p, then x − r is a factor of p. The remaining roots of p are the n − 1 roots of a polynomial q of degree 1 less than the degree of p such that

**and a number 𝑟 find another polynomial s.t.**



Where: 

• A special case of polynomial long division.

• If 𝑝 (𝑟) = 0, 𝑟 is a root of the polynomial.

The pseudocode for Horner’s algorithm can be written as follows:

Text, letter

Description automatically generated**integer** i, n; real p,r; **real array** (ai)0:n, (bi)0:n−1

bn−1 ← an

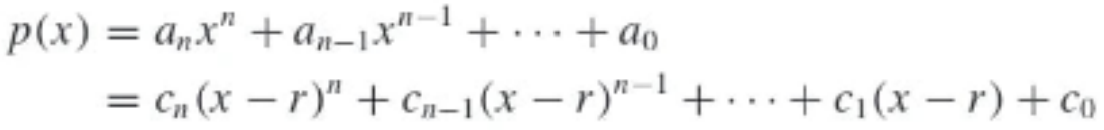
**for** i = n − 1 **to** 0 **do** bi−1 ← ai + rbi

Text

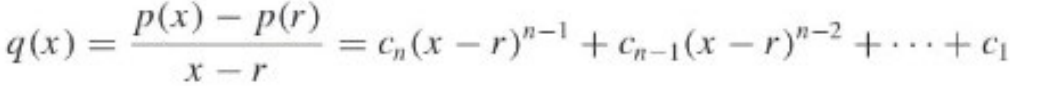
Description automatically generated**end for**

Back to Taylor expansion

• Recall:



• Deflating the polynomial



• Pseudocode

Text, letter

Description automatically generated

Taylor Theorem

Text

Description automatically generated

• In practice, it is necessary to truncate→partial sum.

• E is called the remainder or error term.

• Convergence can be established in some cases.

Chart

Description automatically generatedMean Value Theorem

A special case of Taylor Theorem

Taylor Theorem for 𝑓(𝑥 + ℎ)

A picture containing graphical user interface

Description automatically generated

• Error term converges to zero with the same rate as ℎ𝑛𝑛+1.

• Introduce big O notation, 𝐸𝑛+1 = 𝑂(ℎ𝑛+1) , which means Icon

Description automatically generated

Examples

• It holds for every n

Diagram

Description automatically generated

• Some commonly used ones:

Text, letter

Description automatically generated

Alternating series

Text

Description automatically generated

• It only applies to alternating series.

• It gives an upper bound for the error.

• Back to ln 2 for an example.

**Recap on Taylor**

A picture containing graphical user interface

Description automatically generated

• Error term converges to zero with the same rate as ℎ𝑛𝑛+1.

• Introduce big O notation, 𝐸𝑛+1 = 𝑂(ℎ𝑛+1) , which means Icon

Description automatically generated

Examples

• It holds for every n

Diagram

Description automatically generated

• Some commonly used ones:

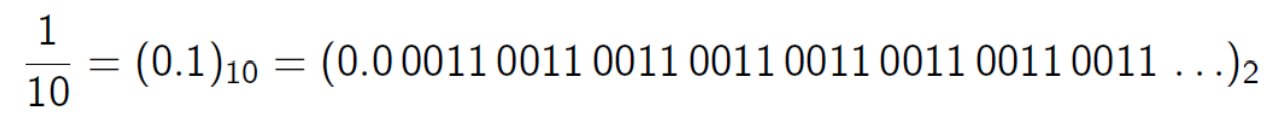
Text, letter

Description automatically generated

Today’s Overview

• Computers usually do not use base-10 numbers.

• Numbers that have a finite expression in one number system may have an infinite one in

another, e.g. 

• We will discuss

– Floating-point number system

– Roundoff error

Floating-point (FP) Repre

For example, 37.2345, 0.0003541, -3093453.32

• **Decimal form** – Integer part – A decimal point – Fractional part

• Normalized scientific notation: leading digit in the fraction is NOT zero.

– e.g., 37.2345 = 0.372345 × 102

• (Standard) scientific notation:

– e.g., 2.99 × 108 m/s

Normalized FP Repre.

A picture containing text, watch, gauge

Description automatically generatedNormalized floating-point representation: 𝑥 = ±𝑟 × 10𝑛

• A sign that is either + or –

• A number 𝑟 ∈ [ 1 /10 , 1)

– called **normalized mantissa**

• An **integer power** of 10

– 𝑛 is called exponent.

Binary system

• If 𝑥 ≠ 0, it can be written as 𝑥 = ±𝑞 × 2𝑚 ( 1/ 2 ≤ 𝑞 < 1)

• The mantissa would be expressed a sequence of binary values (0 or 1) 𝑞 = (0. 𝑏1𝑏2𝑏3 ⋯)2

• 𝑏1 ≠ 0 → 𝑏1 = 1 → 𝑞 ≥ 1/ 2 .

• Next example: list all the numbers can be expressed as and 𝑘 = 0 or 1.

Example 1

Table

Description automatically generated

Computer number system

• Every computer can only represent a **finite** number of digits.

• The real numbers that are representable in a computer are called its **machine number**.

• **Overflow/underflow** describe something is too big/small.

• An **overflow** often results in a fatal error.

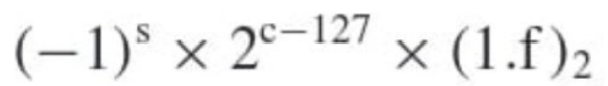
• An **underflow** is usually treated automatically by setting to zero with a warning message.

Common levels of precision

Table

Description automatically generated

Single precision

• Recall 

• 0 < 𝑐 < ( 11 111 111) 2 = 255 ⇒ −127 < 𝑐 − 127 < 128

• 1 ≤ (1. 𝑓)2 = 2 − 2−23

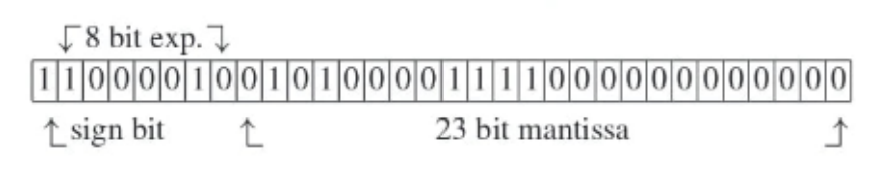
• Largest machine number: 3.4 × 1038

• Smallest machine number: 1.2 × 10−38

• **Machine epsilon**: smallest number 1 + 𝜖 ≠ 1

* 𝜖 = 2−24 ≈ 6 × 10−8 ⇒ **7 significant decimal digits**

Example 2 Determine -52.234375 in single precision.



Double precision

• 11 bits for exponent and 52 for mantissa

• Largest machine number: 1.8 × 10308

• Smallest machine number: 2.2 × 10−308

• Machine epsilon: 2−53 ≈ 1.11 × 10−16

– **15 significant decimal digits**

Computer errors

• The process of replacing a number by its nearest machine number is called **correct rounding**; the error involved is called **roundoff error**.

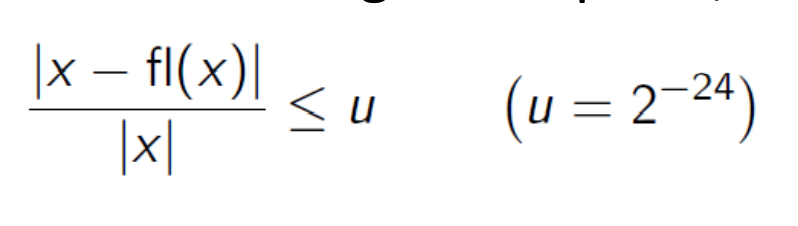
• If a number is **overflow or underflow**, **roundoff** error could be huge.

FP machine number

• Define fl(x) be the FL machine number that corresponds to x

• The function fl depend on the computer.

•For a 32-bit word length computer, we have



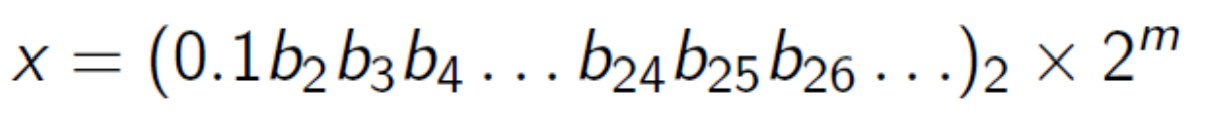
• The inequality can be expressed by

A picture containing text

Description automatically generated

Rounding

• Suppose

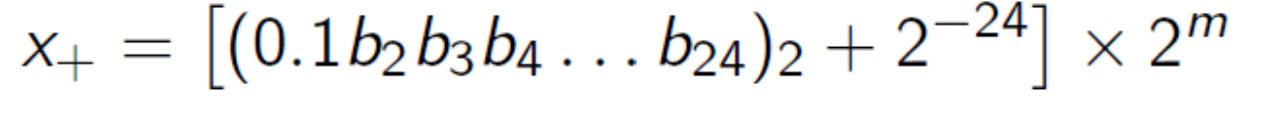


• Round down

A picture containing text

Description automatically generated

• Round up



Unit roundoff error

Diagram, text

Description automatically generated

The **unit roundoff error** for a 32 bit binary computer is 𝑢 = 2−24 , which is equivalent to **machine epsilon**.

Errors in arithmetic operations

Example

• Suppose we have a five-place decimal machine and have two numbers to add



• Perform operations in double-length

Text

Description automatically generated

• Nearest machine number Logo

Description automatically generated with medium confidence

• Error involved

Text

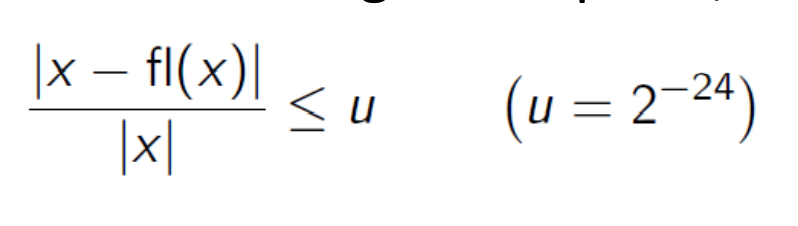
Description automatically generated with medium confidence

FP machine number

• Define fl(x) be the FL machine number that corresponds to x

• The function fl depend on the computer.

•For a 32-bit word length computer, we have

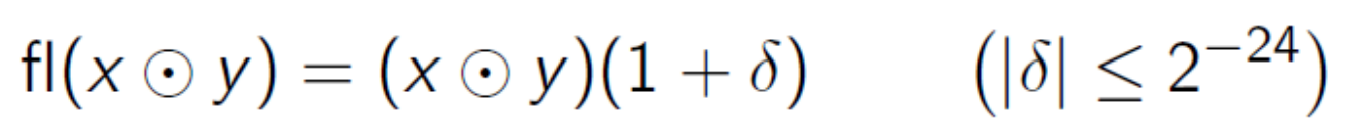


• The inequality can be expressed by

A picture containing text

Description automatically generated

Arithmetic operations



Example: If x, y are real numbers in a 32-bit computer, estimate the relative roundoff error in computing (x+y).

Loss of Significance

• Significance of the digits diminishes from left to right.

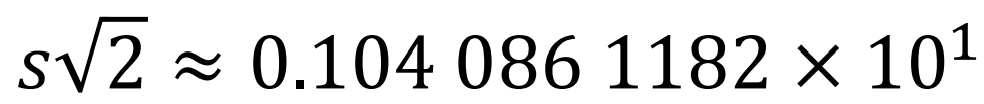
• Every measured quantity involves an error whose magnitude depends on the nature of the measuring device.

• If a meter stick is used, it is not reasonable to get precision better than 1 millimeter, e.g., 2.3453 meters.

• The least significant digit should be in error by at most 5 units, i.e., measured result is **rounded correctly**.

Infinite precision

• If the side of a square is reported to be s = 0.736 meter, then error does not exceed 5 units in the third decimal place.

• The diagonal of the square 

should be reported as 0.1041 × 101.

• The infinite precision in Icon

Description automatically generateddoes not convey any more precision to A picture containing text, clipart

Description automatically generated than was already present in 𝑠.

Loss of significance

Loss of significance

• Consider to execute the statement at x=1/15

A picture containing text

Description automatically generated

• Then

Text

Description automatically generated with medium confidence

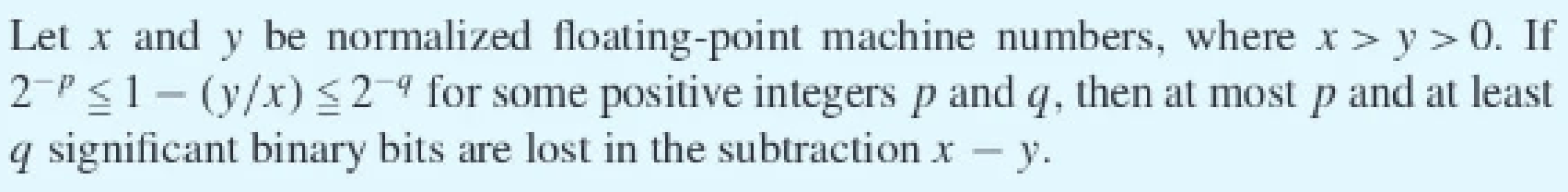
• Correct value

Text

Description automatically generated with medium confidence

Theorem on loss of precision

**Exact how much significant binary digits are lost in subtraction x-y when x is close to y**?



• The closeness of x and y is measured by |1 – 𝑦/ 𝑥 |.

• Double precision may help.

• Taylor series may help

A picture containing text, clock

Description automatically generated

Avoid loss of significance

• Double precision • Taylor series • Rationalization • Trigonometric identities • Logarithmic properties • Range reduction

Root finding

. A number 𝑟 (real or complex), for which 𝑓(𝑟)=0 is called a **root** or a **zero** of 𝑓.

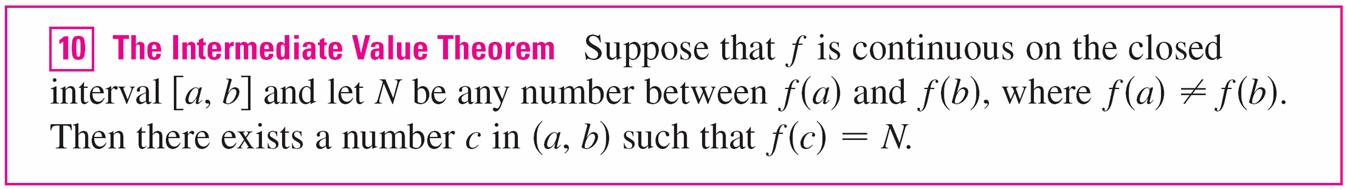
. Why is locating roots important?

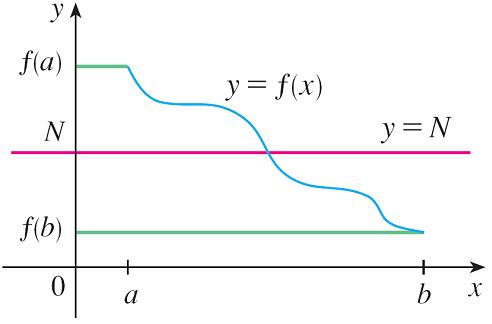
. How to find a root?

* Closed-form solution
* Bisection method

Intermediate Value Theorem (IVT)

Text

Description automatically generated



Bisection method

* Find a root of **a continuous** function .
* At each step, we have an interval with .
* Midpoint and
* If **great**!
* Else either or **why**?
* If then a root exists in we store the value of  **in**  and  **in** .
* If then a root exists in we store the value of **in**  and  **in** .
* In either case, we get to the beginning except that the interval is half as large as the initial one.
* When to stop: interval is sufficiently small, e.g.,
* What’s the final output:

Pseudocode

* Input: (initial interval)
* Output: satisfies

**Remarks**

* Any function value that may be needed later should be stored rather than recomputed.
* Always have stopping conditions in place to avoid **endless** loops.
* Avoid using **== and ~=**
* Underflow/overflow may arise.
* Trace the steps in routine to see it does what is claimed.

Numerical examples

Chart

Description automatically generated with medium confidence

Table

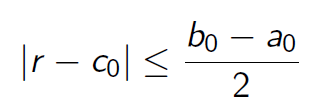
Description automatically generated

A picture containing text, receipt, screenshot

Description automatically generated

**Convergence analysis**

**Analysis**

* Suppose that is a continuous function that takes values of opposite sign at
* Then there exists a root by IVT.
* If we use the midpoint as an estimate of , we have 

Box and whisker chart

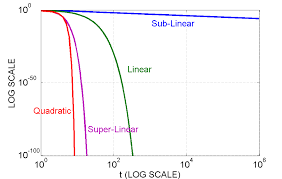
Description automatically generated with low confidence

Convergence Theorem

Graphical user interface, text, application

Description automatically generated

* If error tolerance is prescribed, the number of steps can be calculated.
* The root sought by bisection depends on initial interval and the solution may not be unique.
* Often the bisection method is used to get close to the root before switching to a faster one.

Convergence rate

* Linear convergence:
* Super linear convergence:
* Quadratic convergence

Convergence rate (cont’d)

* A consequence of linear convergence
* Bisection does NOT converge linearly.
* Quadratic convergence doubles the significant digits.

**Bisection variants**

False position method

Rather than selecting the midpoint, this method uses the point where the secant lines intersect the x-axis.

Text, letter

Description automatically generatedChart, line chart

Description automatically generated

Remarks

* False position method uses the values of which is more adaptive to a particular function.
* It may repeatedly select the same endpoint.
* Modified false position method changes the slope of the straight line to get closer to the root. In some cases, super linear convergence rate can be obtained.

**Recap on Bisection**

Bisection method for root finding

• “**Almost**” linear convergence:

Diagram, text

Description automatically generated

Box and whisker chart

Description automatically generated with low confidence• Today’s agenda

– Bisection variants

– Matlab tutorial

– Newton’s method

**Bisection variants**

False position method

. Rather than selecting the midpoint, this method uses the point where the secant lines intersect the x-axis.

Chart, line chart

Description automatically generated

Remarks

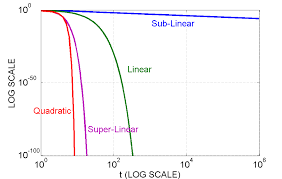
• False position method uses the values of 𝑓(𝑎) , 𝑓(𝑏) , which is more adaptive to a particular function.

• It may repeatedly select the same endpoint.

Chart, line chart

Description automatically generated• Modified false position method changes the slope of the straight line to get closer to the root.

MFP method

Convergence rate

* Linear convergence:
* Superlinear convergence:
* Quadratic convergence

**Matlab tutorial**

Some facts

• Everything in Matlab is a matrix or tensor!

• Matlab does not need any variable declarations, no dimension statements, no storage allocation, no pointers; but better to aware of these for efficiency.

• Programs can be run step by step, with full access to all variables, functions, etc.

Getting help

• To get help on a function type “help function\_name”, e.g., “help plot”.

• To find a topic, type “lookfor topic”, e.g., “lookfor matrix”

Text

Description automatically generated

**Matlab’s Workspace**

• **who, whos** – current workspace vars.

• **save** – save workspace vars to \*.mat file.

• **load** – load variables from \*.mat file.

• **clear all** – clear workspace vars.

• **close all** – close all figures

• **clc** – clear screen

• **clf** – clear figure

Variables

• Variable names:

– Must start with a letter

– May contain only letters, digits, and ‘\_’

– Case sensitive, one & ONE are different

– Built-in variables/functions are all lower-case.

• Assignment:

– Variable = number

– Variable = expression

Basic Commands

• **%** used to denote a comment

• **;** suppresses display of value (when placed at end of a statement)

• **...** continues the statement on next line

• **eps** machine epsilon

• **inf** infinity

• **NaN** not-a number, e.g., 0/0.

**Numbers**

• To change format of numbers: format long, format short, etc. See “**help format**”.

• Mathematical functions: **sqrt(x), exp(x), cos(x), sin(x), sum(x),** etc.

• Operations: **+, -, \* , /,^**

• Constants: **pi, exp(1),** etc.

• Elementwise operator for vectors, .

Arrays and Matrices

• **v = [-2 3 0 4.5 -1.5];** % length 5 row vector.

**• v = v’;** % transposes v.

• **v(1);** % first element of v.

**• v(2:4);** % entries 2-4 of v.

**• v([3,5]);** % returns entries 3 & 5.

• **v=[4:-1:2];** % same as v=[4 3 2];

• **a=1:3; b=2:3; c=[a b];** -> c = [1 2 3 2 3];

. Tic

. Toc

. function handle

**Newton’s method**

Overview

• **Newton’**s method is also called **Newton-Raphson** iteration.

• It has a wider spectrum of applications.

• It requires the function to be differentiable.

Chart, line chart

Description automatically generated• Its basic idea is that the graph at a certain point can be well approximated by its tangent.

Interpretation

• Starting from a point (𝑥0, 𝑓(𝑥0 ))

• Compute the tangent line

• Advance to the next point

Another interpretation

• What correction ℎ should be added to 𝑥𝑥0 to obtain the root precisely?

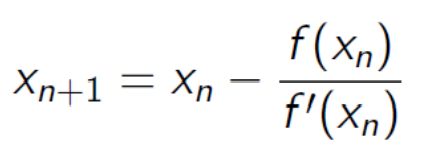
• Taylor series

O = f(x0) + f’(x0)(x1-x0)

• Find an approximated value for ℎ

Summary of Newton

• Newton’s method returns a sequence of points: 𝑥0, 𝑥1, ⋯

• Recursive or inductive definition: 

• Convergence: Text

Description automatically generated with low confidence

Recap on Newton’s method

• Newton’s recursive definition



• Today’s agenda – Pseudocode of Newton – Quadratic Convergence – Secant method.

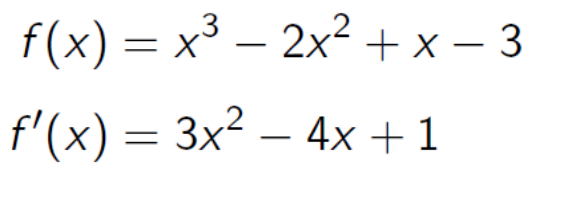
Pseudocode

• Input: f(x), df(x), x0 (initial)

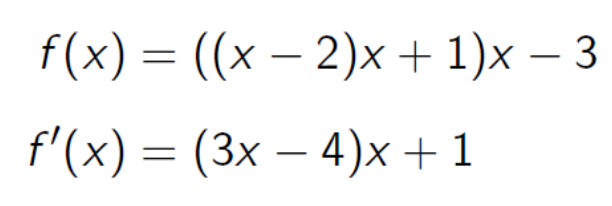
• Output: root of f(x)

Example

• Given



• For efficiency, nested multiplication



Table

Description automatically generatedExample (cont’d)

Chart, line chart

Description automatically generated• Doubling of the accuracy in f(x)

• Rapid convergence!

• 5-10 iterations are generally sufficient.

Convergence analysis

If 𝑓, 𝑓′ , 𝑓′′ are continuous in a neighborhood of a root 𝑟 of 𝑓 and if 𝑓′ (𝑟) ≠ 0 (**simple zero**) , then Newton’s method **converges quadratically**! Recall Taylor Theorem

Text, letter

Description automatically generated

Define

Text

Description automatically generated

Remarks

• Newton’s method relies on a starting point.

• Bisection (initial) + Newton (improve accuracy)

• Convergence depends upon hypotheses that are difficult to verify a priori.

Failure of Newton’s

Chart, line chart

Description automatically generated

Multiplicity

• Quadratic convergence holds only for simple zero, i.e., 𝑓′ (𝑟) ≠ 0

• The multiplicity of the zero is the least 𝑚𝑚 s.t. 𝑓(𝑘)( 𝑟) = 0, ∀𝑘 < 𝑚𝑚

Text, letter

Description automatically generated• Newton’s method converges linearly for a multiple zero.

• Modified Newton’s method with multiplicity

Nonlinear equations

Chart, box and whisker chart

Description automatically generated

Newton’s Method:

Text

Description automatically generated

An example

Text

Description automatically generated with low confidence

Chart, line chart

Description automatically generated**Secant method**

Overview

• **Bisection method** requires two points with opposite signs.

• **Newton method’s** drawback lies in the derivative calculation.

• **Secant method** approximates the function derivative by secant line.

**Remarks**

• Newton’s method could be called tangent method.

• Secant method requires two initial points, but no need to have opposite signs as bisection.

• Overflow may occur if demon is closer to 0.

• Super linear convergence!