DTS104TC
NUMERICAL METHODS

INTRODUCTION & LECTURE 1

LONG HUANG

Xi'an Jiaotong-Liverpool University
西交利物浦大学

CONTENTS

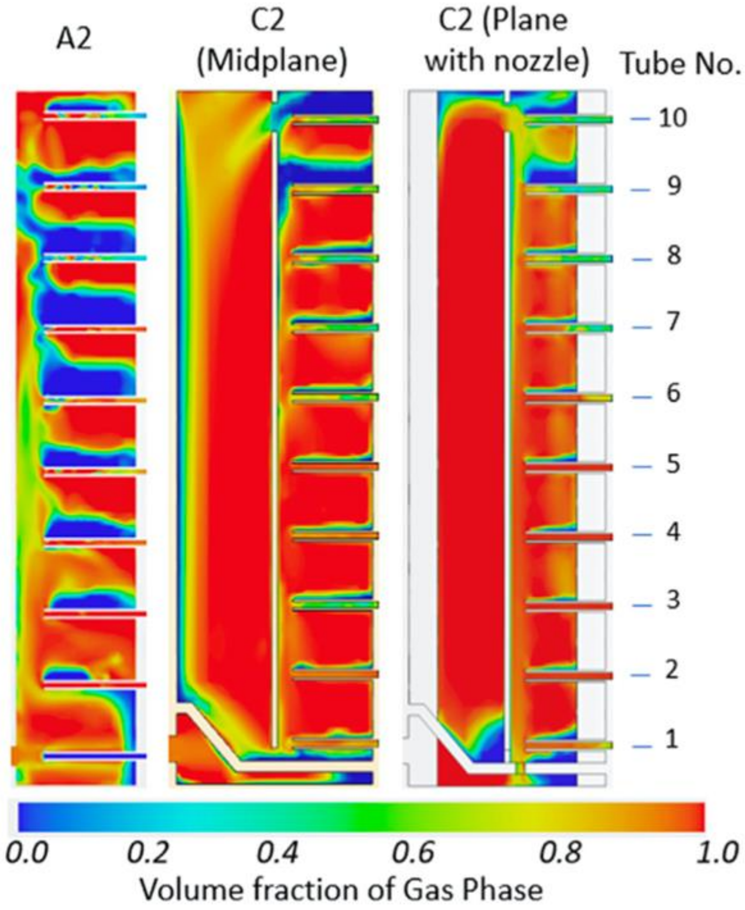
- Course Introduction
- Handbook Walkthrough
- Errors, Accuracy and Precisions
- Non-linear Equation Solving

COURSE INTRODUCTION

Why this is a required module for your degree:

A bridge to connect numerical computation and other subjects. Think how data scientists work with domain experts.

The emphasis is to equip students with the knowledge to obtain numerical solutions to mathematical problems that otherwise may not be solved analytically.



Numerical simulation of two-phase flow separation

Handbook Walkthrough

KEY INFORMATION

Module name: Numerical Methods

Module code: DTS104TC

Credit value: 2.5

Semester in which the module is taught: Second Part of Semester 2, 2020-2021

Pre-requisites needed for the module: NA (Math modules required to opt in big data program)

Programme: BEng Data Science and Big Data Technology with Contemporary Entrepreneurialism

SCHEDULE

Lectures	
Time (Week 8 – Week 12)	Teaching mode: onsite/online
Mon. 4-6 pm	Mon. SC169 / Wed. SB102
Wed. 9-11 am	https://learningmall.xjtlu.edu.cn/course/view.php?id=1389
Tutorial	
Time (Week 8 – Week 13)	Teaching mode: onsite/online
Thu. 11-1 pm for Class Group 1/D1	BS394
Thu. 2-4 pm for Class Group 2/D1	BS394
Fri. 2-4 pm for Class Group 3/D1	BS2114 https://learningmall.xjtlu.edu.cn/course/view.php?id=1389
Seminar	
Time (Week 14)	Teaching mode: online/onsite
Thu. 11-1 pm for Class Group 1/D1	BS492
Thu. 4-6 pm for Class Group 2/D1	PB202
Fri. 9-11 am for Class Group 3/D1	MB119 https://learningmall.xjtlu.edu.cn/course/view.php?id=1389

*In person attendance expected for students on campus

LEARNING OUTCOMES

A. Apply numerical methods in a number of different contexts.

B. Solve systems of linear and nonlinear algebraic equations to specified precision.

C. Compute eigenvalues and eigenvectors by the power method.

D. Solve boundary value and initial problems to finite precision.

E. Develop quadrature methods for numerical integration.

SYLLABUS

(a) Solution of non-linear equations in one variable and multi-variables, Newton-Raphson, Jacobian Matrix and error behaviour and its consequences.
(b) Norms, Spectral Radius, Gauss elimination and LU factorization for solving systems of equations. Partial pivoting.
(c) Iterative methods (Gauss Seidel, Jacobi, SOR), convergence of linear iterative methods.
(d) The eigenvalue problem: the power method and inverse power method
(e) Ordinary and partial differential equations, multi-step methods, finite difference methods.
(f) Lagrange interpolation and quadrature methods: trapezoidal rule, Simpson's rule, Gaussian quadrature; numerical solution of initial value problems and Fredholm integral equations.

*This module focuses on selections and applications of methods
**Students are expected to contribute self-study time if interested in detailed proof of methods

ASSESSMENT

Initial Assessment

Sequence	Method	Assessment Type (EXAM or CW)[2]	Learning outcomes assessed (use codes under Learning Outcomes)	% of Final Mark	Resit(Y/N/S)[3]
001	Assignment	CW	A, B, C, D, E	80	S
002	Assignment	CW	A, B, E	20	S

- Coursework assignment 1 (80%): submission deadline 10th June 2021, 5pm.
- Coursework assignment 2 (20%): submission deadline 2nd June 2021, 5pm.

*subject to coursework moderation at XJTLU, Liverpool and External

Resit Assessment

Sequence	Assessment Type (EXAM or CW)	Learning outcomes assessed (use codes under Learning Outcomes)	% of Final Mark
R001	CW	A, B, C, D, E	100

TEACHING PLAN

Week Number and/or Date	Lecture/Seminar/ Field Trip/Other	Topic/Theme/Title
Week 8 Monday	Lecture 1	Solution of non-linear equations
Week 8 Wednesday	Lecture 2	Solving systems of equations
Week 9 Monday	Lecture 3	Iterative methods
Week 9 Wednesday	Lecture 4	Convergence of linear iterative methods.
Week 10 Monday	Lecture 5	The power method
Week 10 Wednesday	Lecture 6	Inverse power method
Week 11 Monday	Lecture 7	Ordinary differential equations
Week 11 Wednesday	Lecture 8	Partial differential equations
Week 12 Monday	Lecture 9	Lagrange interpolation and quadrature methods
Week 12 Wednesday	Lecture 10	Initial value problems

*Detailed contents may be adjusted based on learning and teaching progress

Week Number and/or Date	Tutorial	Topic/Theme/Title
Week 8	Tutorial 1	Syllabus(a)
Week 9	Tutorial 2	Syllabus(b)
Week 10	Tutorial 3	Syllabus(c)
Week 11	Tutorial 4	Syllabus(d)
Week 12	Tutorial 5	Syllabus(e)
Week 13	Tutorial 6	Syllabus(f)

** Tutorials are for:
-Answering frequently asked questions
-Make-up lectures
-Matlab based practices

**Arrangements
Thursday Morning: Long Huang
Thursday Afternoon & Friday : Yue Wang

Week Number and/or Date	Topic/Theme/Title	Lecturer/Instructor
Week 14 for all groups	Question and Answer Session	Long Huang

REFERENCE BOOKS

Title	Author	ISBN/Publisher
NUMERICAL ANALYSIS., 3RD EDITION	TIMOTHY SAUER	9780134697321 /PEARSON
AN INTRODUCTION TO NUMERICAL ANALYSIS 1ST EDITION	ENDRE SULI	978-052100794 /CAMBRIDGE UNIVERSITY
NUMERICAL METHODS FOR ENGINEERS	STEVEN C. CHAPRA, RAYMOND P. CANALE	9780073397924 /MCGRAW-HILL
NUMERICAL METHODS FOR ENGINEERS AND SCIENTISTS	J.N. SHARMA	9781842653654 /ALPHA SCIENCE
MATLAB: A PRACTICAL INTRODUCTION TO PROGRAMMING AND PROBLEM SOLVING	ATTAWAY, STORMY	9780124058767 /BUTTERWORTH-HEINEMANN
NUMERICAL METHODS USING MATLAB	GEOGRE LINDFIELD and JOHN PENNY	9780123869425

ADDITIONAL INFORMATION

- Students who are able to be on campus are reminded of the Academic Policy requiring no less than 80% attendance at classes. Failure to observe this requirement may lead to failure or exclusion from resit or retake examinations.
- Attendance QR code will be provided when class begins and during break.
- Certain level of proficiency in Matlab is needed to complete the assignments.
- Slides will be provided before lecture starts.
- Feedbacks are always encouraged during the teaching period.
- Questions will be answered during office hour, or in a collective manner during tutorials.
- LMO and emails are preferred means of communication.

MATLAB

- Software which was originally developed as a matrix laboratory. A variety of numerical functions, symbolic computations, and visualization tools have been added to the matrix manipulations.

- Excellent for numerical calculations.

(a) **Pseudocode**	*(b)* **MATLAB**

IF/THEN:

```
IF condition THEN                    if b ~= 0
  True block                           r1 = -c / b;
ENDIF                                end
```

IF/THEN/ELSE:

```
IF condition THEN                    if a < 0
  True block                           b = sqrt(abs(a));
ELSE                                 else
  False block                          b = sqrt(a);
ENDIF                                end
```

IF/THEN/ELSEIF:

```
IF condition1 THEN                   if class == 1
  Block1                               x = x + 8;
ELSEIF condition2                    elseif class < 1
  Block2                               x = x - 8;
ELSEIF condition3                    elseif class < 10
  Block3                               x = x - 32;
ELSE                                 else
  Block4                               x = x - 64;
ENDIF                                end
```

CASE:

```
SELECT CASE Test Expression          switch a + b
  CASE Value1                          case 1
    Block1                               x = -5;
  CASE Value2                          case 2
    Block2                               x = -5 - (a + b) / 10;
  CASE Value3                          case 3
    Block3                               x = (a + b) / 10;
  CASE ELSE                            otherwise
    Block4                               x = 5;
END SELECT                           end
```

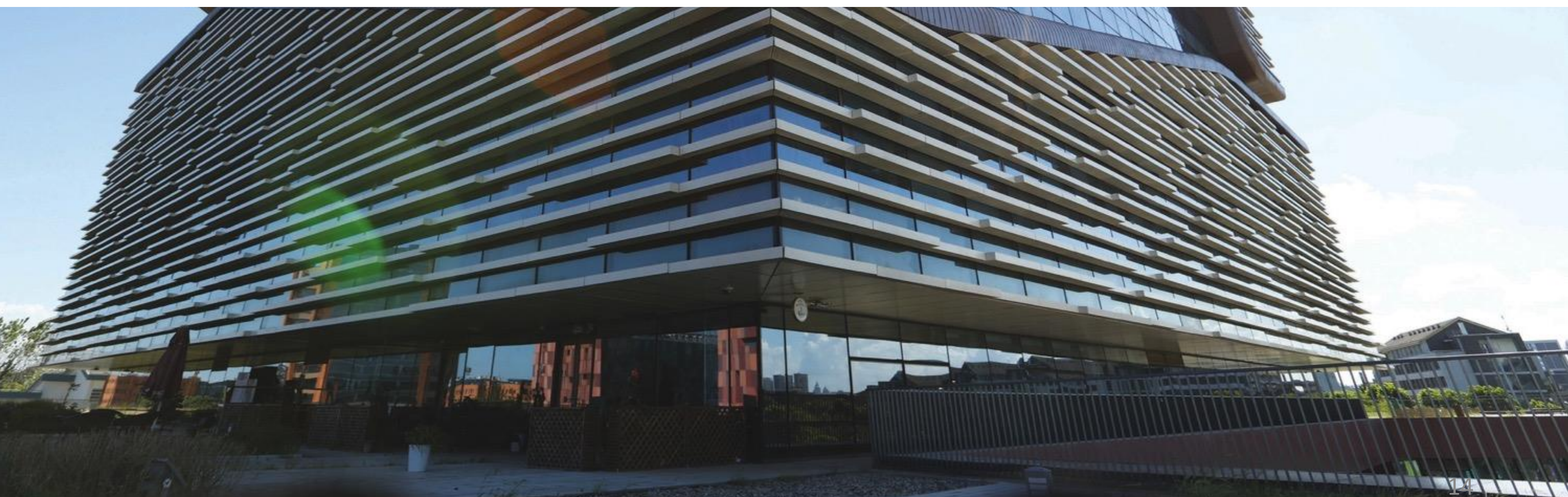
DOEXIT:

```
DO                                   while (1)
  Block1                               i = i + 1;
  IF condition EXIT                    if i >= 10, break, end
  Block2                               j = i*x;
ENDIF                                end
```

COUNT-CONTROLLED LOOP:

```
DOFOR i = start, finish, step        for i = 1:10:2
  Block                                x = x + i;
ENDDO                                end
```

Errors & Accuracy



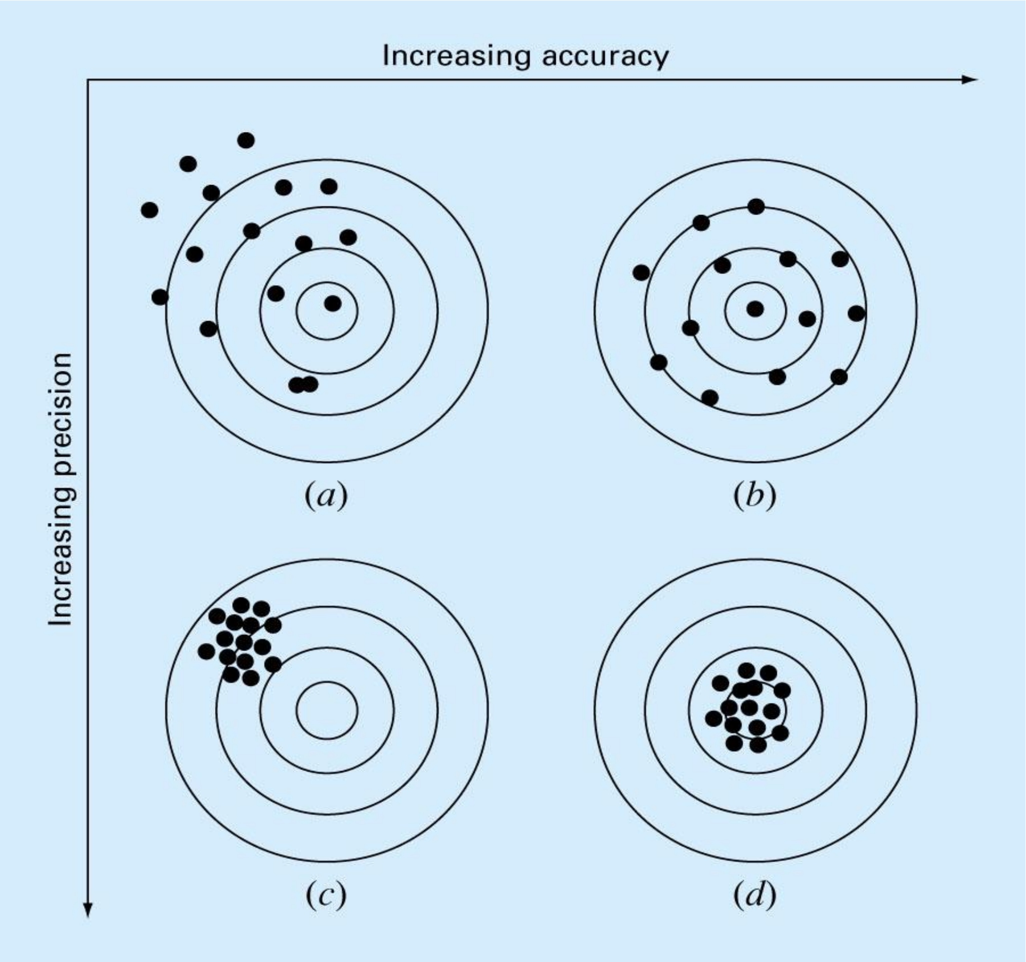
APPROXIMATIONS AND ROUND-OFF ERRORS

- Numerical methods yield approximate results, results that are close to the exact analytical solution. We cannot exactly compute the errors associated with numerical methods.
- Only rarely given data are exact, since they originate from measurements. Therefore there is probably error in the input information.
- Algorithm itself usually introduces errors as well, for example, unavoidable round-offs, etc …
- The output information will then contain error from both of these sources.
- How confident we are in our approximate result?
- The question is “how much error is present in our calculation and is it tolerable?”

ACCURACY & PRECISION

- Accuracy. How close is a computed or measured value to the true value.
- Precision (or reproducibility). How close is a computed or measured value to previously computed or measured values.
- Inaccuracy (or bias). A systematic deviation from the actual value.
- Imprecision (or uncertainty). Magnitude of scatter.

TARGETS ILLUSTRATING ACCURACY & PRECISION



DEFINITION OF ERROR

- True Value = Approximation + Error

$$\text{True fractional relative error} = \frac{\text{true error}}{\text{true value}}$$

$$\text{True percent relative error, } \varepsilon_{\text{t}} = \frac{\text{true error}}{\text{true value}} \times 100\%$$

DEFINITION OF ERROR

- Number of significant figures indicates precision. Significant digits of a number are those that can be used with confidence, for example, the number of certain digits plus one estimated digit.

53,8<u>00</u> How many significant figures?

5.38×10^4	3
5.380×10^4	4
5.3800×10^4	5

- Zeros are sometimes used to locate the decimal point not significant figures.

0.00001753	4
0.0001753	4
0.001753	4

DEFINITION OF ERROR

- For numerical methods, the true value will be known only when we deal with functions that can be solved analytically (simple systems). In real world applications, we usually not know the answer a priori. Then

$$\varepsilon_{\mathrm{a}} = \frac{\text{Approximate error}}{\text{Approximation}} \times 100\%$$

- Iterative approach, example Newton's method.

$$\varepsilon_{\mathrm{a}} = \frac{\text{Current approximation} - \text{Previous approximation}}{\text{Current approximation}} \times 100\%$$

DEFINITION OF ERROR

- Use absolute value.
- Computations are repeated until stopping criterion is satisfied.

$$\left|\varepsilon_a\right| < \varepsilon_s$$

Pre-specified % tolerance based on the knowledge of your solution

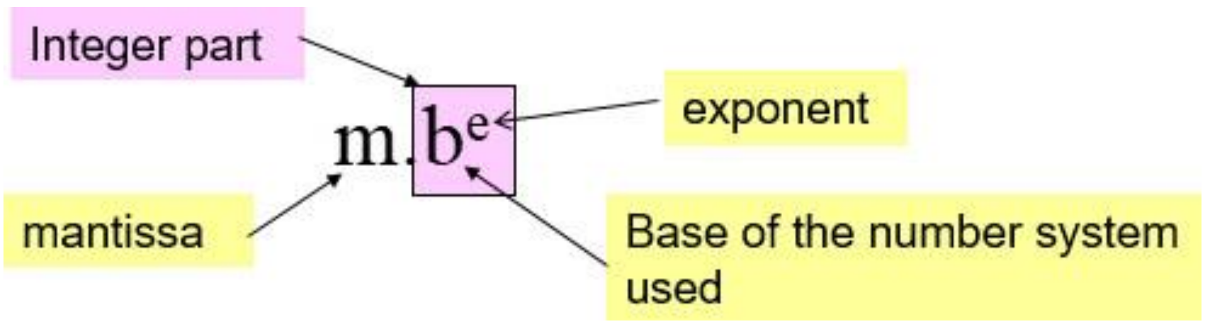
- If the following criterion is met

$$\varepsilon_s = \left(0.5 \times 10^{(2\text{-}n)}\right)\%$$

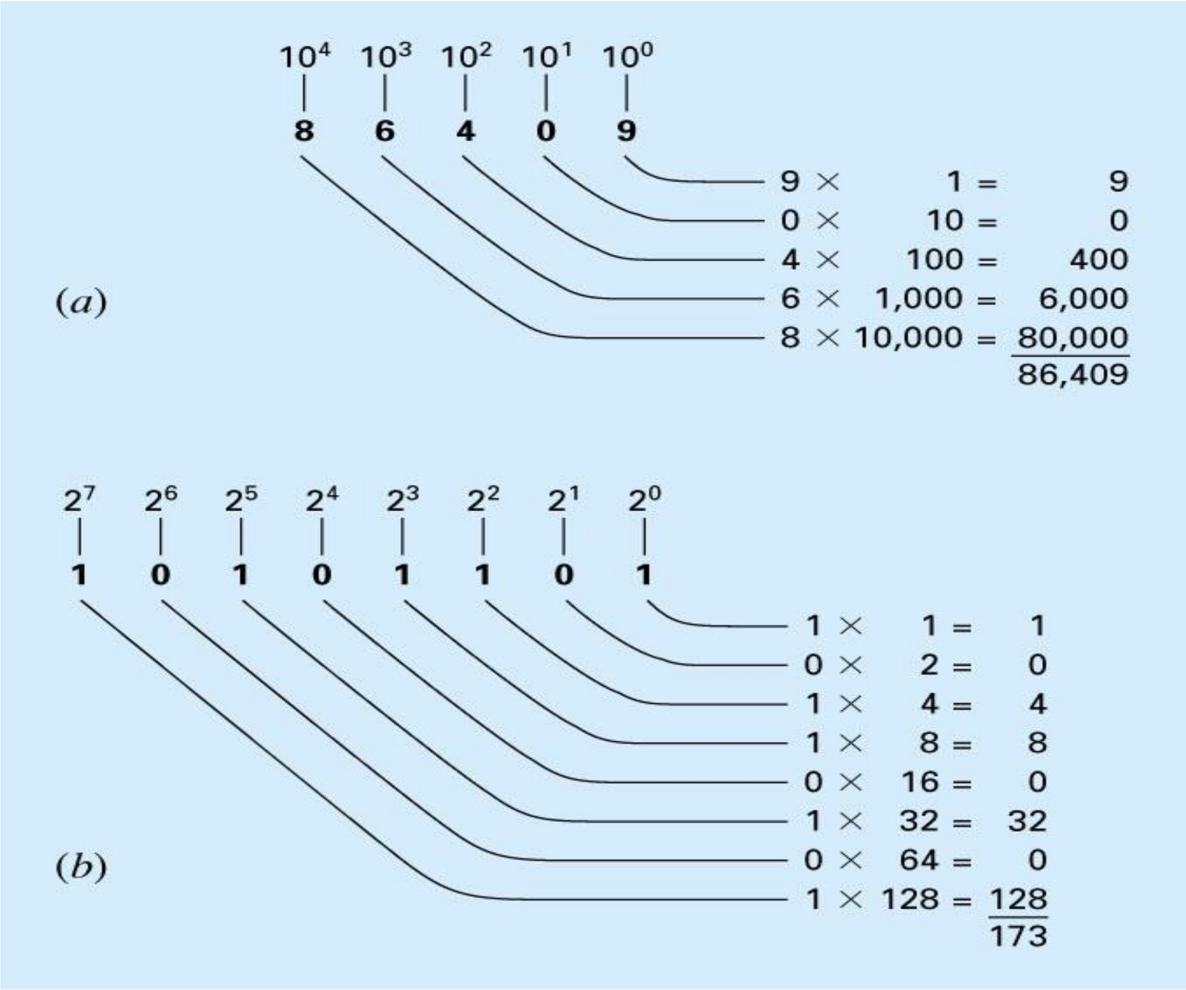
- You can be sure that the result is correct to at least n significant figures.

ROUND-OFF ERRORS

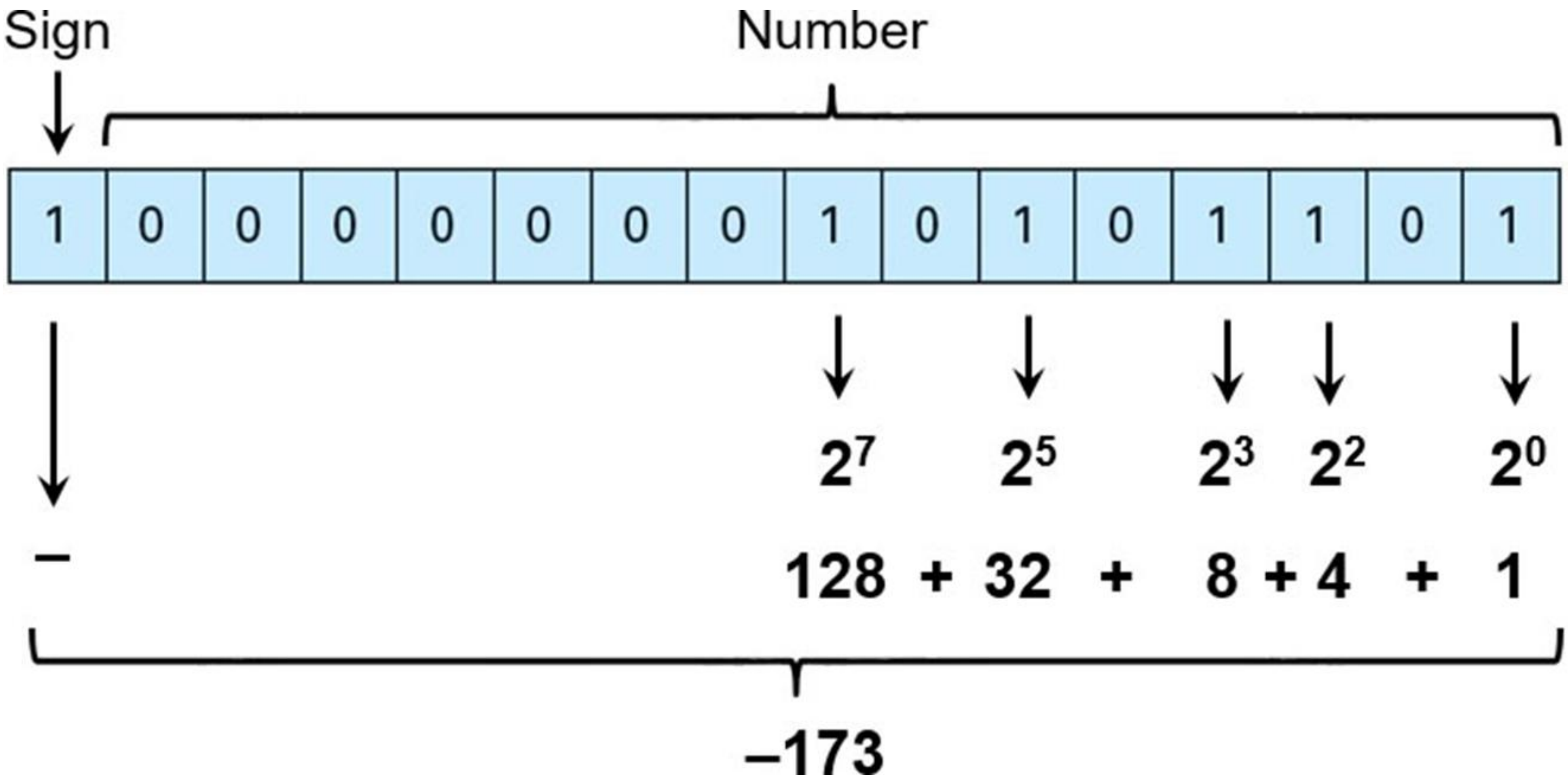
- Numbers such as π, e, or $\sqrt{7}$ cannot be expressed by a fixed number of significant figures.
- Computers use a base-2 representation, they cannot precisely represent certain exact base-10 numbers.
- Fractional quantities are typically represented in computer using “floating point” form, for example,



(A) DECIMAL & (B) BINARY NUMBER SYSTEMS



THE SIGNED MAGNITUDE METHOD FOR INTEGER REPRESENTATION OF –173



HOW A DECIMAL NUMBER WOULD BE STORED IN A HYPOTHETICAL BASE-10 COMPUTER

- Example: $1/34 = 0.029411765$
- Suppose only 4 decimal places can be stored

$$0.029411765 \rightarrow 0.0294 \times 10^{0}$$

- The first zero to right of decimal point is useless. So, normalize by multiplying the mantissa by 10 and lowering the exponent by 1

$$0.0294 \times 10^{0} \rightarrow 0.294\underline{1} \times 10^{-1}$$

Additional significant figure is retained

NORMALIZATION MEANS LIMITED RANGE OF MANTISSA

$$\frac{1}{b} \le |m| < 1$$

- for a base-10 system $0.1 \le m < 1$
- for a base-2 system $0.5 \le m < 1$
- Floating point representation allows both fractions and very large numbers to be expressed on the computer. However,
  - Floating point numbers take up more room.
  - Take longer to process than integer numbers.
  - Round-off errors are introduced because mantissa holds only a finite number of significant figures.

CHOPPING & ROUNDING

- Example:
- $\pi = 3.14159265358\ldots$ to be stored on a base-10 system carrying 7 significant digits.

$$\pi = 3.141592 \qquad \text{chopping error} \;\rightarrow\; \varepsilon_t = 0.00000065$$

- If rounded

$$\pi = 3.141593 \qquad \text{rounding error} \;\rightarrow\; \varepsilon_t = 0.00000035$$

- Some machines use chopping, because rounding adds to the computational overhead. Since number of significant figures is large enough, resulting chopping error is “usually” negligible.

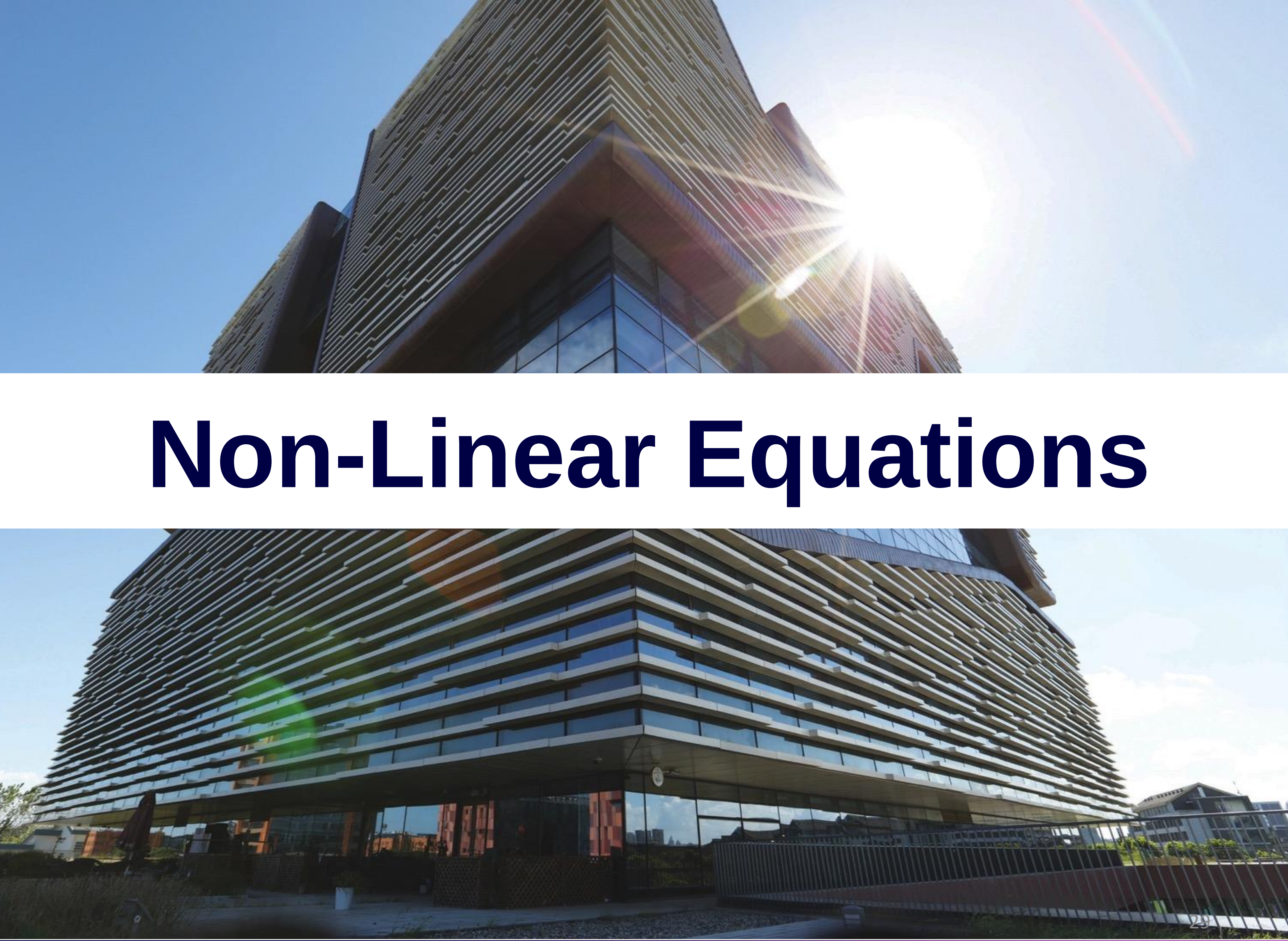
SOURCES OF ROUNDING

- Large interdependent computations (billions of operations); tiny errors can compound (add up).
- Adding a large & small number.
- Smearing: Occurs whenever the individual terms in a summation are larger than the summation itself.
- Subtractive cancellation: round-off induced when subtracting two nearly equal floating-point numbers.

Suggested reading of weaknesses with floating-point numbers:

https://ece.uwaterloo.ca/~dwharder/NumericalAnalysis/02Numerics/Weaknesses/

Non-Linear Equations



ROOTS OF EQUATIONS

- Why?

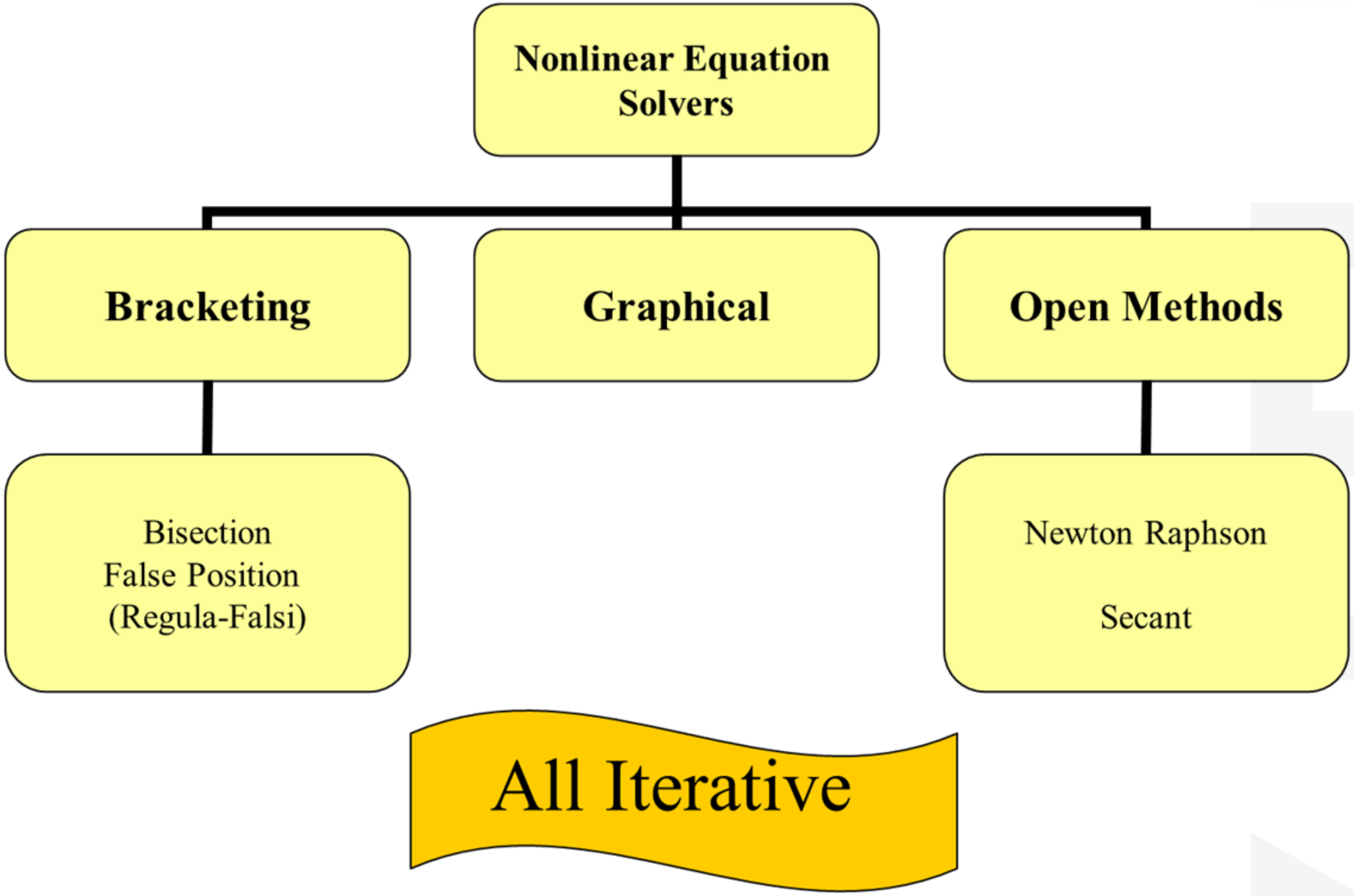
$$ax^2 + bx + c = 0 \quad \Rightarrow \quad x = \frac{-b \mp \sqrt{b^2 - 4ac}}{2a}$$

- But,

$$ax^5 + bx^4 + cx^3 + dx^2 + ex + f = 0 \quad \Rightarrow x = ?$$

$$\sin x + x = 0 \quad \Rightarrow x = ?$$

OUTLINE OF APPROACHES



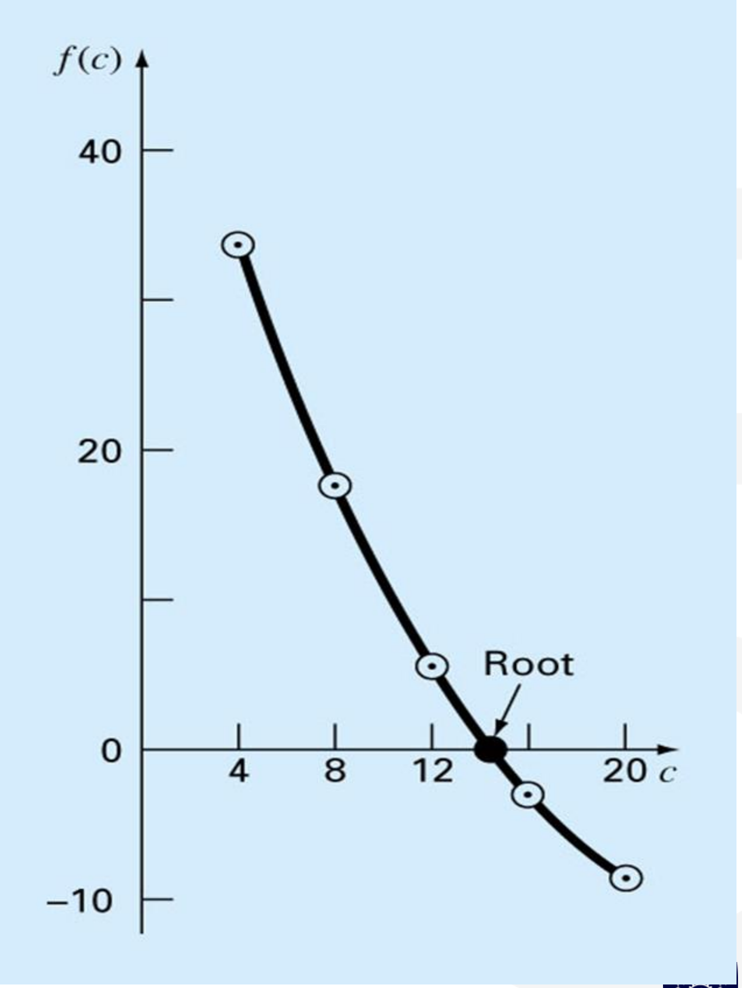
BRACKETING METHODS

- Two initial guesses for the root are required. These guesses must "bracket" or be on either side of the root.

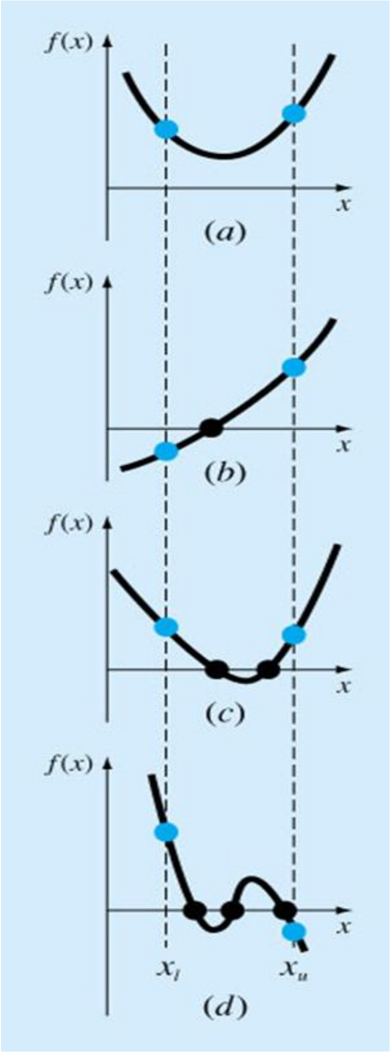
- If one root of a real and continuous function, f(x)=0, is bounded by values x = xl and x = xu then if,

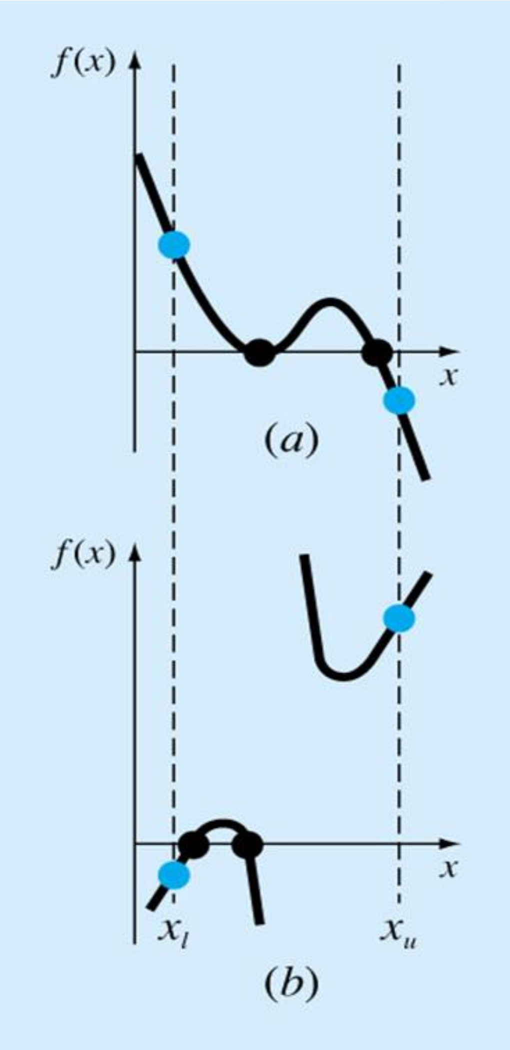
$$f(x_l) \times f(x_u) < 0$$

- The function changes sign on opposite sides of the root).

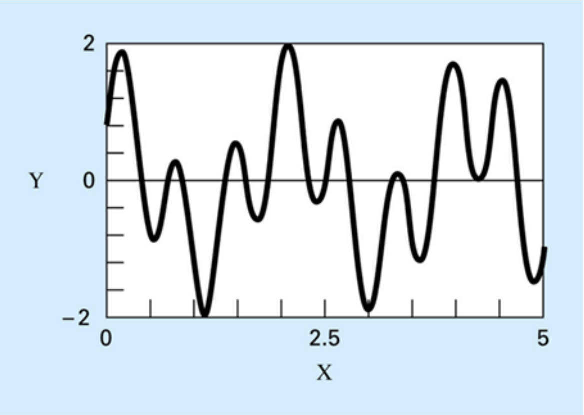


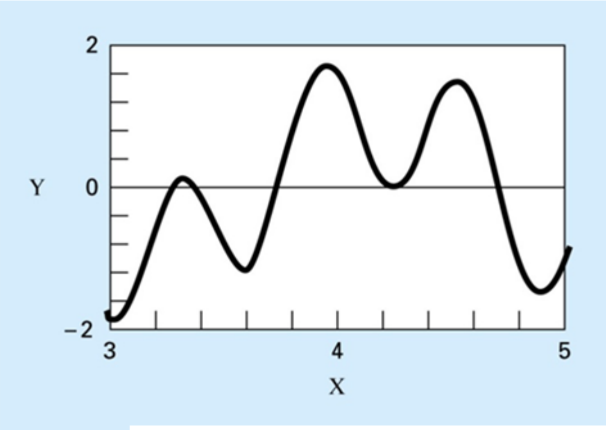
IMPORTANCE OF INITIAL GUESSES

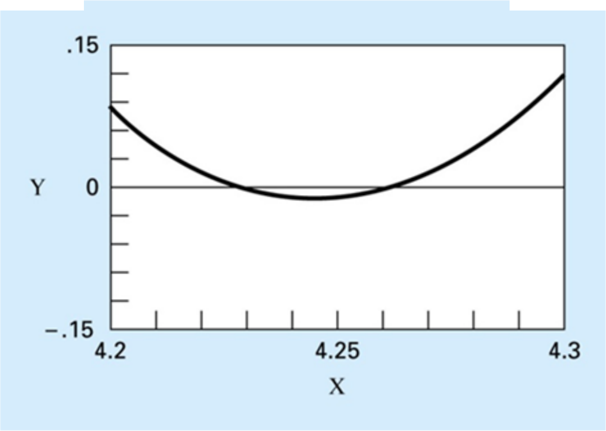




GRAPHICAL APPROACHES







THE BISECTION METHOD

For the arbitrary equation of one variable, $f(x) = 0$.

1. Pick x_l and x_u such that they bound the root of interest check if $f(x_l) \times f(x_u) < 0$.

2. Estimate the root by evaluating $f\left[\left(x_l + x_u\right)/2\right]$.

3. Find the pair.

   If $f\left(x_l\right) \times f\left[\left(x_l + x_u\right)/2\right] < 0,$ root lies in the lower interval, then

   $x_u = \left(x_l + x_u\right)/2$ and go to step 2.

   If $f\left(x_l\right) \times f\left[\left(x_l + x_u\right)/2\right] > 0,$ root lies in the lower interval, then

   $x_l = \left[\left(x_l + x_u\right)/2\right],$ and go to step 2.

THE BISECTION METHOD

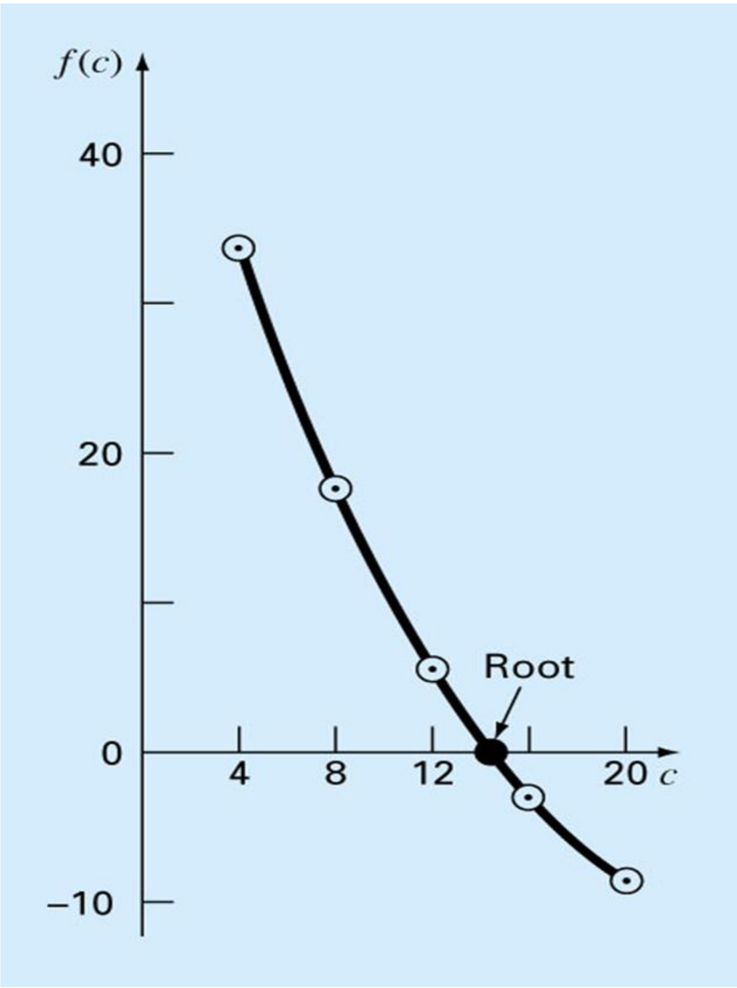
If $f(x_l) \times f\left[(x_l + x_u)/2\right] = 0,$ then root is $(x_l + x_u)/2$, terminate.

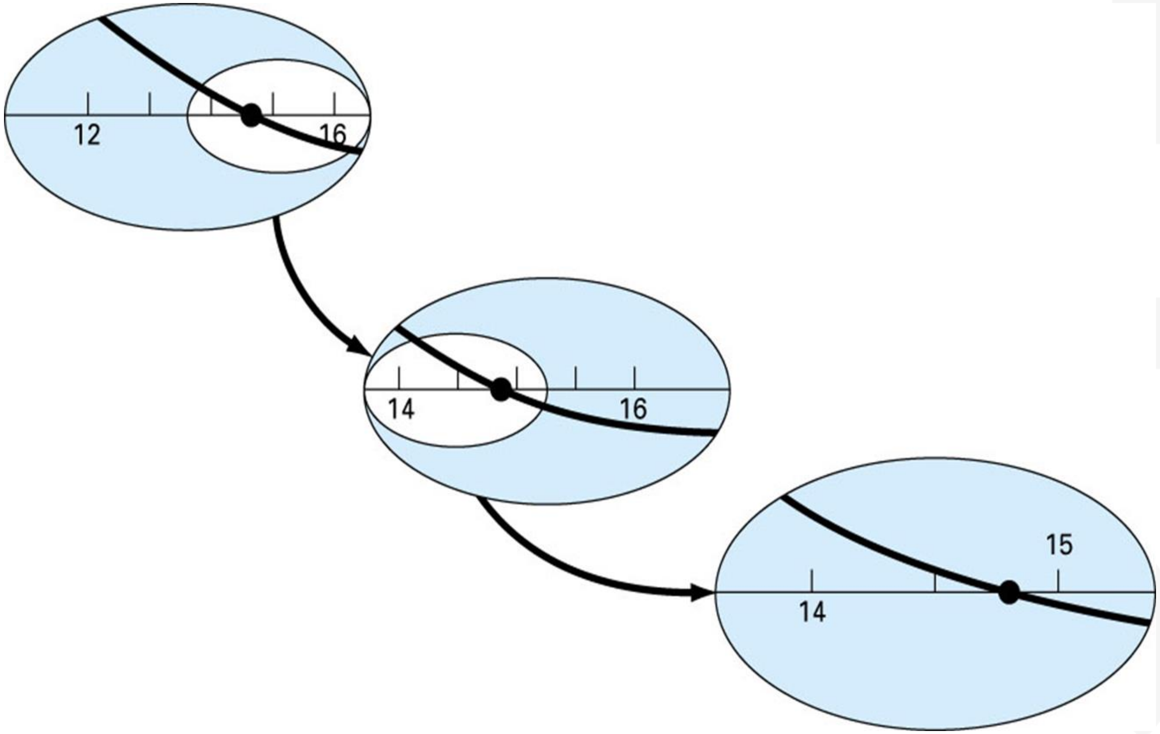
4. Compare stopping criterion with approximate percent relative error ε_s with ε_a.

5. If $\varepsilon_a < \varepsilon_s,$ stop. Otherwise repeat the process.

$$\left\{ \begin{array}{l} \dfrac{\left| x_l - \dfrac{x_l + x_u}{2} \right|}{\left| \dfrac{x_l + x_u}{2} \right|} \prec 100\% \\ \\ \text{or} \\ \dfrac{\left| x_u - \dfrac{x_l + x_u}{2} \right|}{\left| \dfrac{x_l + x_u}{2} \right|} \prec 100\% \end{array} \right\}$$

GRAPHICAL DEPICTION OF THE BISECTION METHOD





EVALUATION OF METHOD

Pros

- Easy.
- Always find root.
- Number of iterations required to attain an absolute error can be computed *a priori*.

Cons

- Slow.
- Know *a* and *b* that bound root.
- Multiple roots.
- No account is taken of $f(x_l)$ and $f(x_u)$, if $f(x_l)$ is closer to zero, it is likely that root is closer to x_l.

HOW MANY ITERATIONS WILL IT TAKE?

- Length of the first Interval. $L_0 = b - a$
- After 1 iteration. $L_1 = L_0/2$
- After 2 iterations. $L_2 = L_0/4$
- After *k* iterations. $L_k = L_0/2^k$

$$\varepsilon_a \leq \frac{L_k}{n} \times 100\% \qquad \varepsilon_a \leq \varepsilon_s$$

HOW MANY ITERATIONS WILL IT TAKE?

- If the absolute magnitude of the error is,

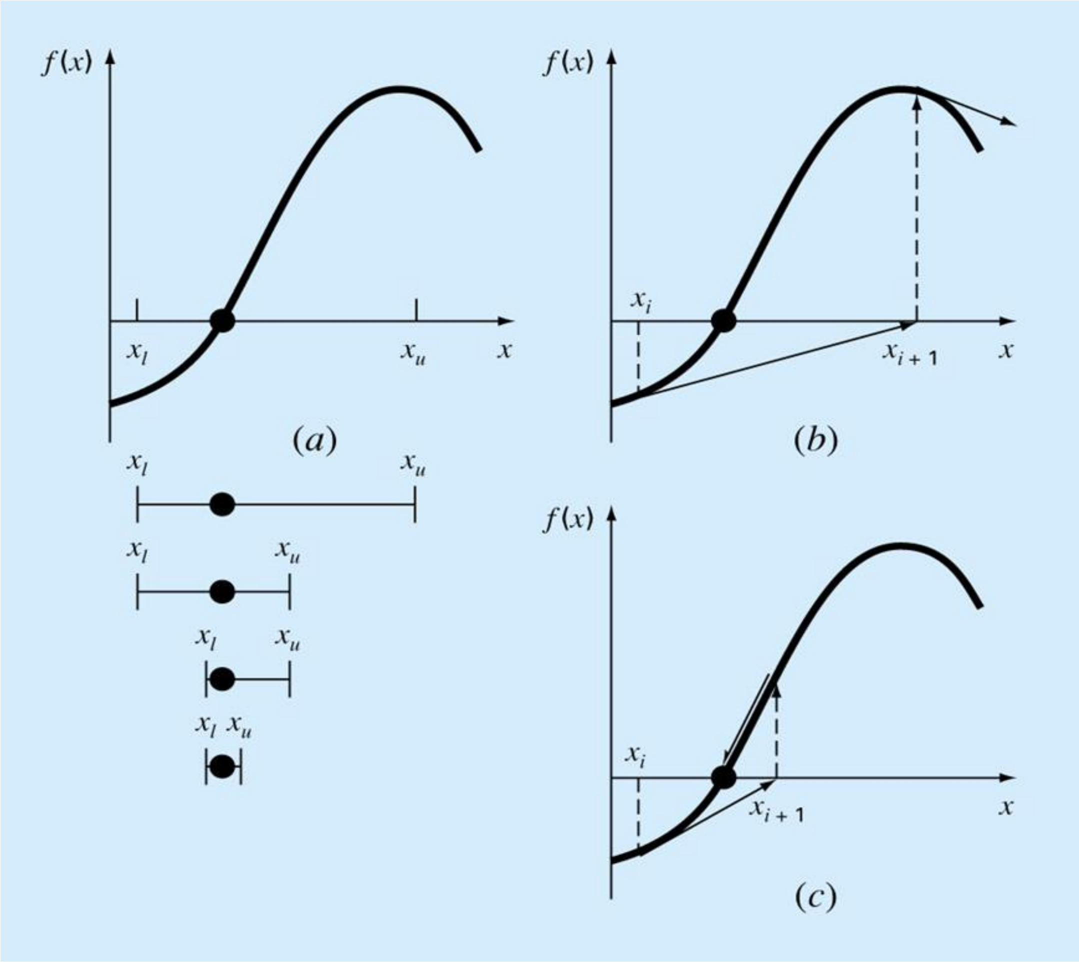
$$\frac{\varepsilon_s \cdot x}{100\%} = 10^{-4}$$

- and L_0 = 2, how many iterations will you have to do to get the required accuracy in the solution?

$$10^{-4} = \frac{2}{2^k} \Rightarrow 2^k = 2 \times 10^4 \Rightarrow k \cong 14.3 = 15$$

OPEN METHODS

- Open methods are based on formulas that require only a single starting value of x or two starting values that do not necessarily bracket the root.



SIMPLE FIXED-POINT ITERATION

- Rearrange the function so that x is on the left side of the equation:

$$f(x) = 0 \quad \Rightarrow \quad g(x) = x$$

$$x_k = g(x_{k-1}) \qquad x_0 \text{ given, } k = 1, 2, \ldots$$

- Bracketing methods are “convergent”.

- Fixed-point methods may sometime “diverge”, depending on the stating point (initial guess) and how the function behaves.

SIMPLE FIXED-POINT ITERATION

$$f(x) = x^2 - x - 2 \qquad x > 0$$

Here are 3 options:

$$g(x) = x^2 - 2$$

- or

$$g(x) = \sqrt{x+2}$$

- or

$$g(x) = 1 + \frac{2}{x}$$

$$\vdots$$

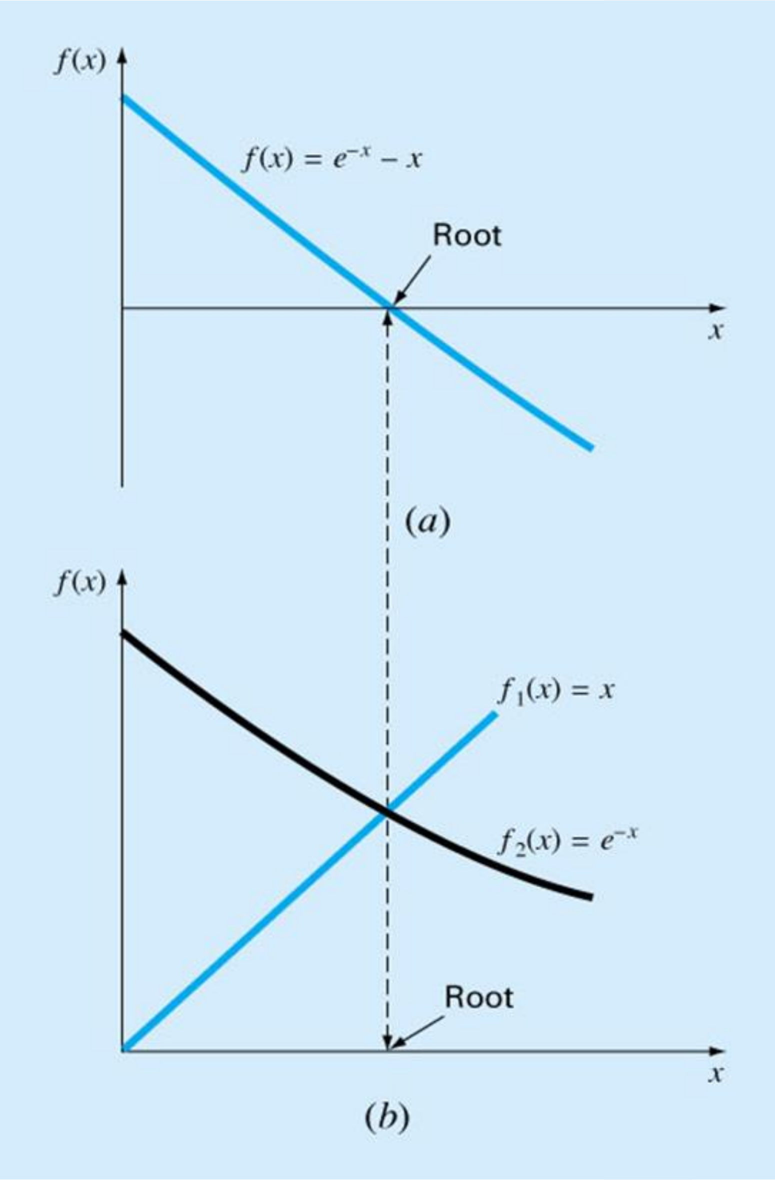
CONVERGENCE

- $x = g(x)$ can be expressed as a pair of equations:

$$y_1 = x$$
$$y_2 = g(x)$$

(component equations)

- Plot them separately.



COBWEB PLOTS

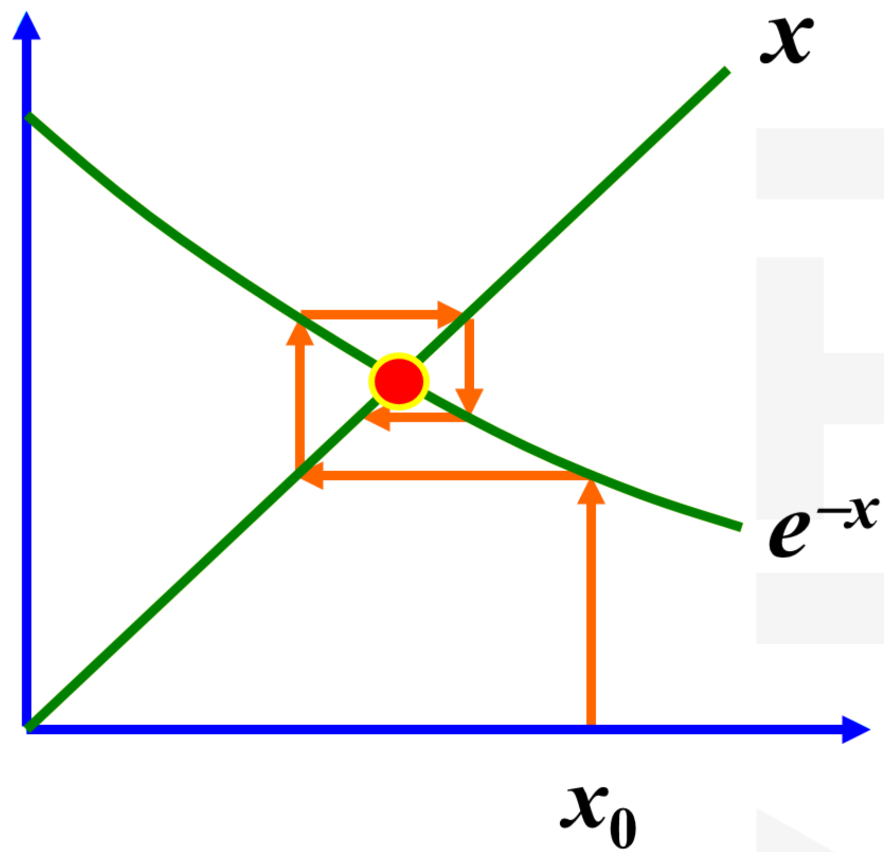
$$x_0$$

$$\downarrow$$

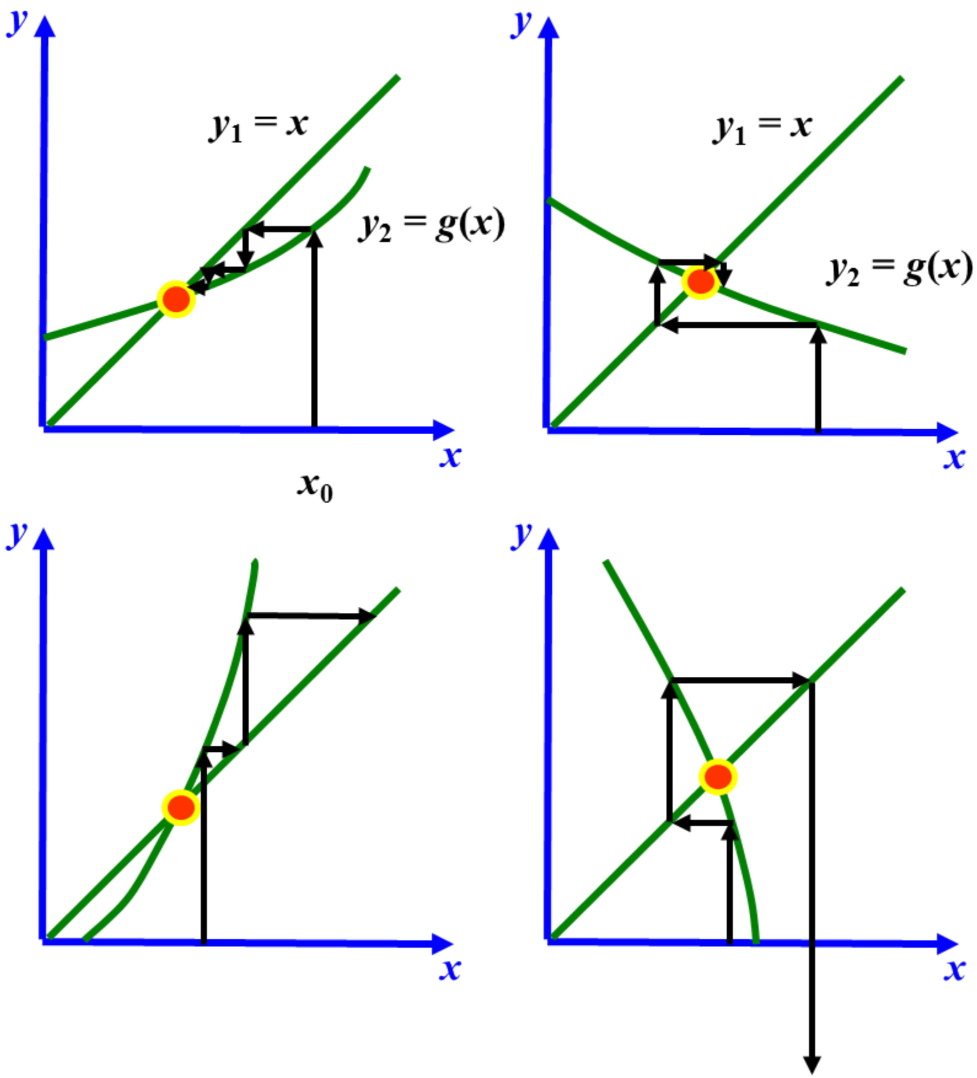
$$x_1 \leftarrow e^{-x_0}$$

$$\searrow$$

$$x_2 \leftarrow e^{-x_1}$$



HOW FIXED-POINT ITERATION CAN WORK OR CAN FAIL



$$E_{i+1} = g'(\xi)\, E_i$$

If $|g'(\xi)| < 1$

converges

If $|g'(\xi)| > 1$

- diverges

If $|g'(\xi)| = 1$

- infinite

CONCLUSION OF FIXED-POINT ITERATION

- Fixed-point iteration converges if,

$$\left|g'(x)\right| < 1 \quad \left(\text{slope of the line } f(x) = x\right)$$

- When the method converges, the error is roughly proportional to or less than the error of the previous step, therefore it is called "linearly convergent."

NEWTON-RAPHSON METHOD

- Most widely used method.
- Based on Taylor series expansion:

$$f(x_{i+1}) = f(x_i) + f'(x_i)\Delta x + f''(x_i)\frac{\Delta x^2}{2!} + O\Delta x^3$$

- The root is the value of x_{i+1} when $f(x_{i+1}) = 0$
- Rearranging,

Solve for

$$0 = f(x_i) + f'(x_i)(x_{i+1} - x_i)$$

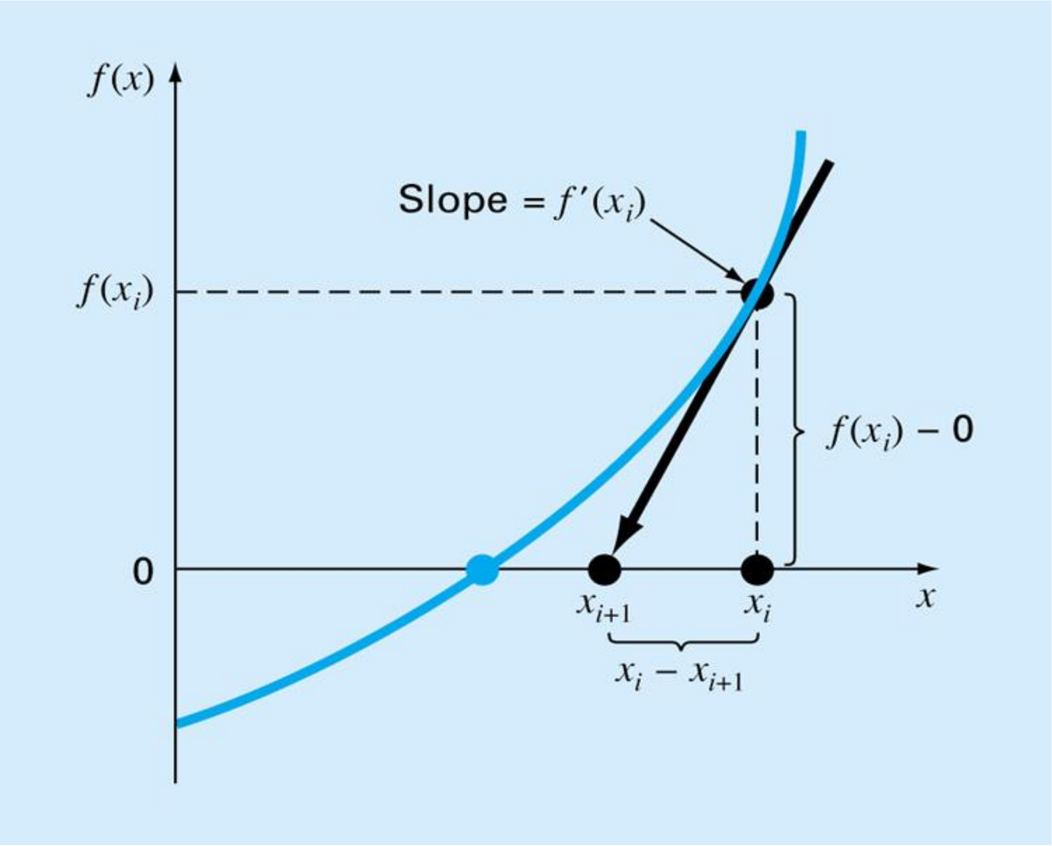
- Solve for,

$$\boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}}$$

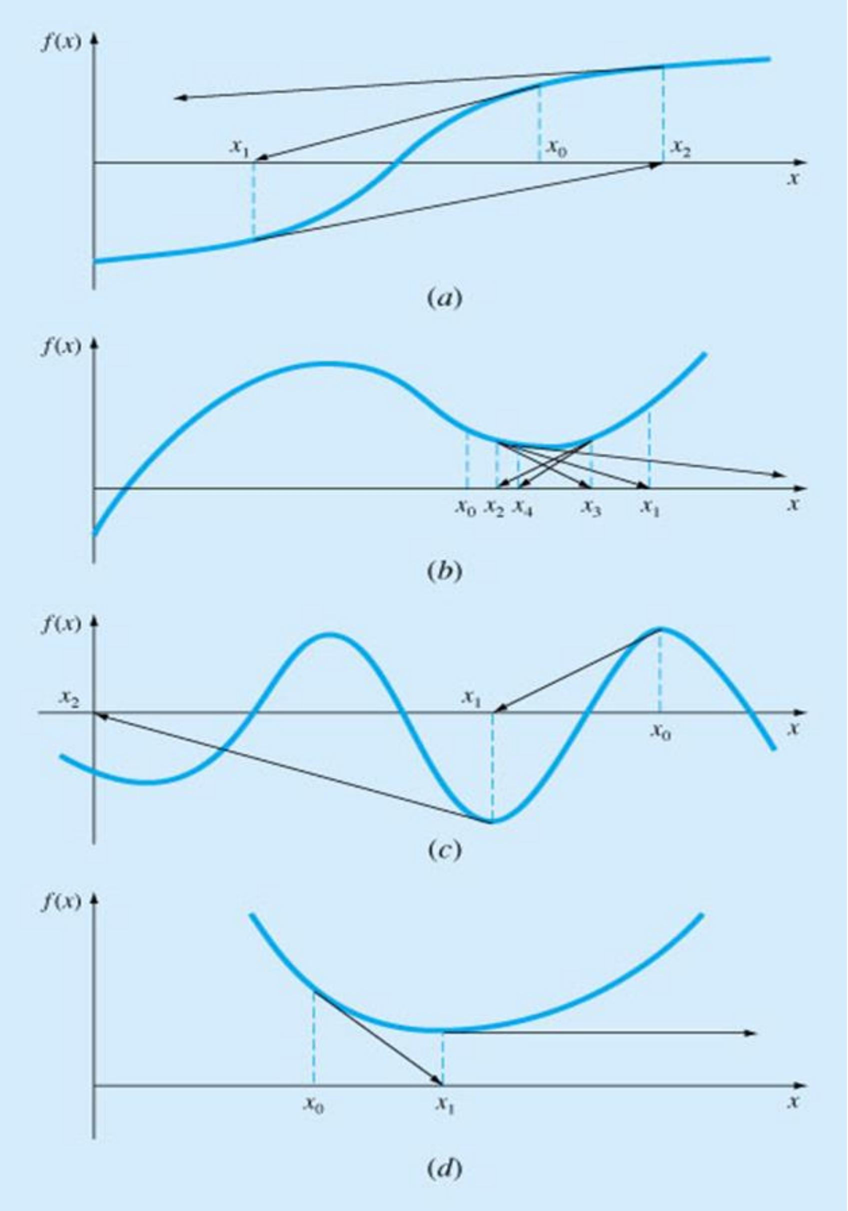
- Newton-Raphson formula

NEWTON-RAPHSON METHOD

- A convenient method for functions whose derivatives **can be evaluated analytically.** It may not be convenient for functions whose derivatives cannot be evaluated analytically.



WHERE NEWTON-RAPHSON EXHIBITS POOR CONVERGENCE



THE SECANT METHOD

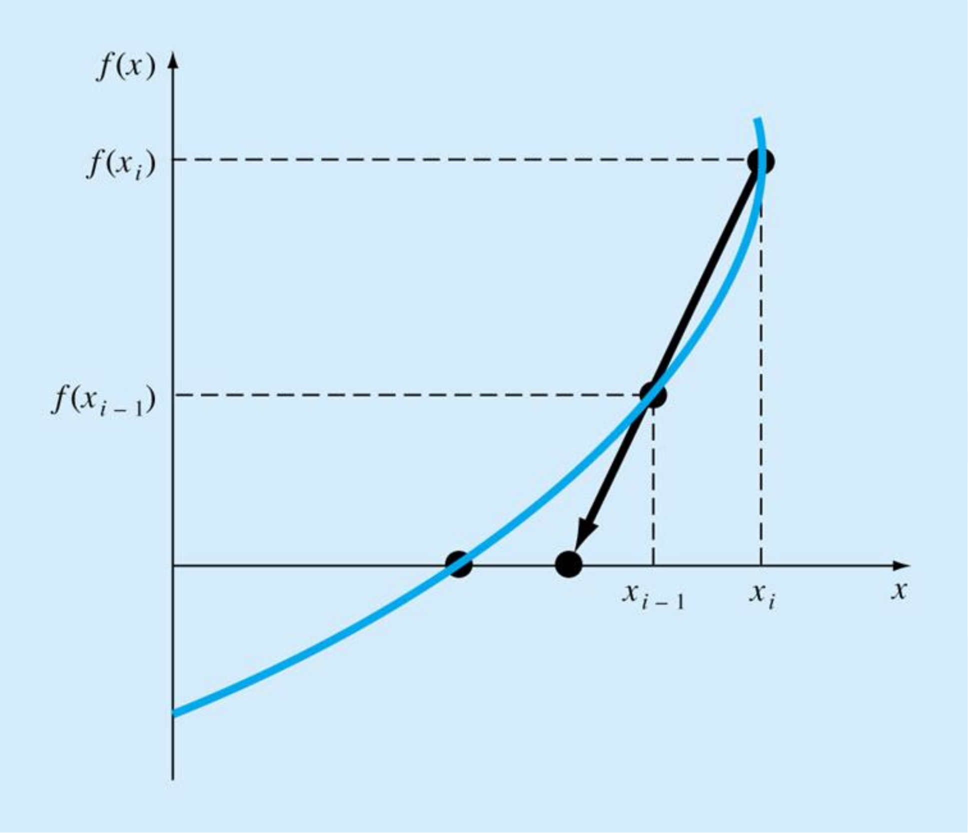
- A slight variation of Newton's method for functions whose derivatives are difficult to evaluate. For these cases the derivative can be approximated by a backward finite divided difference.

$$f'(x_i) \cong \frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})}$$

$$x_{i+1} = x_i - f(x_i)\frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})} \qquad i = 1, 2, 3, \ldots$$

THE SECANT METHOD

- Requires two initial estimates of x , for example, x_0, x_1. However, because $f(x)$ is not required to change signs between estimates, it is not classified as a “bracketing” method.
- The secant method has the same properties as Newton’s method. Convergence is not guaranteed for all initial guesses and functions, $f(x)$.



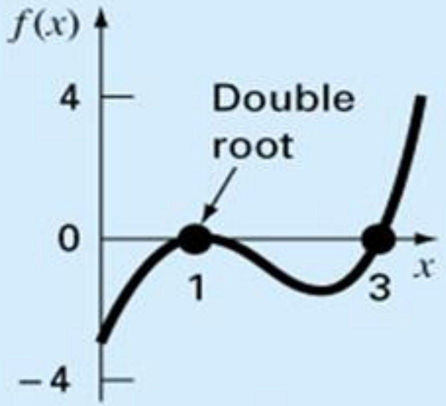
MULTIPLE ROOTS

"Multiple root" corresponds to a point where a function is tangent to the *x* axis.

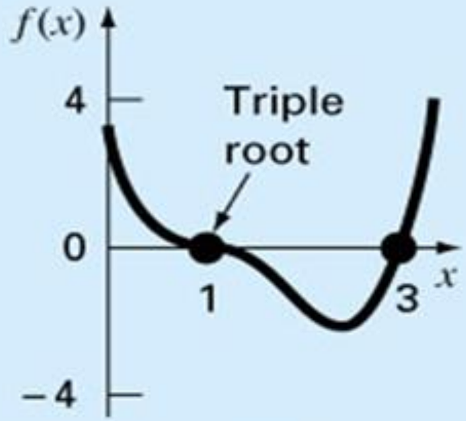
Example $$f(x) = (x-3)(x-1)(x-1)$$

- Function does not change sign at the multiple root, therefore, cannot use bracketing methods.
- Both $f(x)$ and $f'(x) = 0$, division by zero with Newton's and Secant methods.

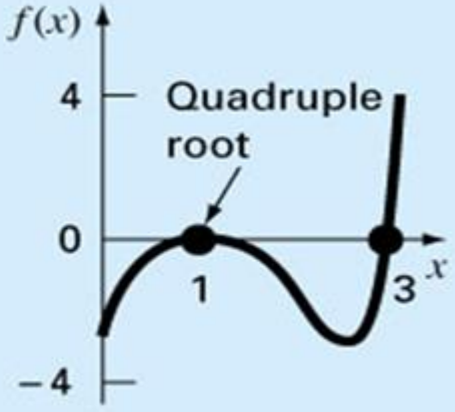
EXAMPLES OF MULTIPLE ROOTS



(a)



(b)



(c)

DETERMINING MULTIPLE ROOTS

- None of the methods deal with multiple roots efficiently, however, one way to deal with problems is as follows:

Set $u(x_i) = \dfrac{f(x_i)}{f'(x_i)}$

This function has roots at all the same locations as the original function.

- Then find $x_i + 1 - \dfrac{u(x_i)}{u'(x_i)}$

SYSTEMS OF NONLINEAR EQUATIONS

$$x^2 + xy = 10$$
$$y + 3xy^2 = 57$$

$$u(x, y) = x^2 + xy - 10 = 0$$
$$v(x, y) = y + 3xy^2 - 57 = 0$$

- Determine the value of (x, y) that make:

$$u(x, y) = 0$$
$$v(x, y) = 0$$

MULTI-DIMENSIONAL NEWTON-RAPHSON

- Taylor series expansion of a function of more than one variable.

$$u_{i+1} = u_i + \frac{\partial u_i}{\partial x}(x_{1i+1} - x_{1i}) + \frac{\partial u_i}{\partial y}(y_{i+1} - y_i)$$

$$v_{i+1} = v_i + \frac{\partial v_i}{\partial x}(x_{1i+1} - x_{1i}) + \frac{\partial v_i}{\partial y}(y_{i+1} - y_i)$$

- The root of the equation occurs at the value of x and y where u_{i+1} and v_{i+1} equal to zero.

MULTI-DIMENSIONAL NEWTON-RAPHSON

$$\frac{\partial u_i}{\partial x} x_{i+1} + \frac{\partial u_i}{\partial y} y_{i+1} = -u_i + x_i \frac{\partial u_i}{\partial x} + y_i \frac{\partial u_i}{\partial y}$$

$$\frac{\partial v_i}{\partial x} x_{i+1} + \frac{\partial v_i}{\partial y} y_{i+1} = -v_i + x_i \frac{\partial v_i}{\partial x} + y_i \frac{\partial v_i}{\partial y}$$

- A set of two linear equations with two unknowns that can be solved for.

MULTI-DIMENSIONAL NEWTON-RAPHSON

$$x_{i+1} = x_i - \frac{u_i \dfrac{\partial v_i}{\partial y} - v_i \dfrac{\partial u_i}{\partial y}}{\dfrac{\partial u_i}{\partial x}\dfrac{\partial v_i}{\partial y} - \dfrac{\partial u_i}{\partial y}\dfrac{\partial v_i}{\partial x}}$$

$$y_{i+1} = y_i - \frac{u_i \dfrac{\partial v_i}{\partial x} - v_i \dfrac{\partial u_i}{\partial x}}{\dfrac{\partial u_i}{\partial x}\dfrac{\partial v_i}{\partial y} - \dfrac{\partial u_i}{\partial y}\dfrac{\partial v_i}{\partial x}}$$

Determinant of the *Jacobian* of the system.