Introduction to Mathematical Modeling and Numerical Errors

Ashok Basnet

https://github.com/realashok

dev.ashokbasnet@gmail.com

Lecture 1 - Mathematical Modeling and Numerical Errors

Introduction

- Role of numerical methods in engineering

Mathematical Modeling

- What is a mathematical model?
- Steps in developing a model
- Example: The Falling Parachutist

Conservation Laws in Engineering

- Mass, momentum, energy conservation
- Translating physical problems → mathematical form → numerical algorithm.

Transition: From Modeling to Computation

- Why models need to be approximated numerically.
- Limitations of digital computation (finite precision).

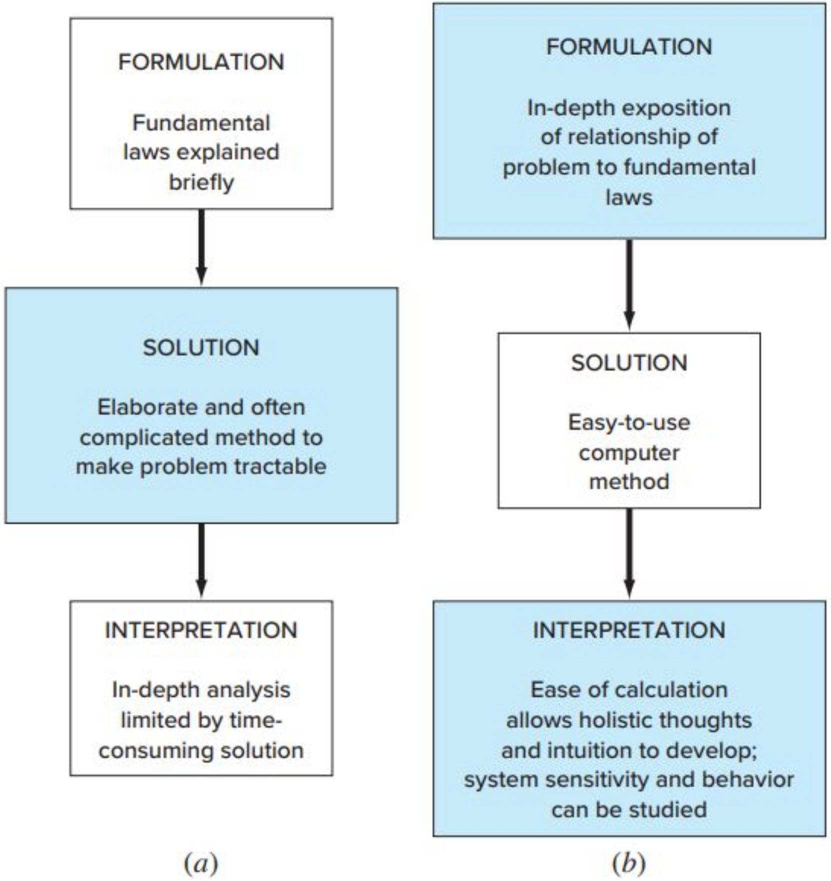
Approximation Concepts

- Significant Figures, Accuracy, and Precision

Learning Objectives

- Explain what a mathematical model is and how it relates to engineering systems.
- Distinguish between analytical and numerical solutions.
- Describe the role of conservation laws in building models.
- Define and compute approximation errors: absolute, relative, and percentage.
- Understand the meaning of accuracy, precision, and significant figures.
- Explain how computers represent numbers and the origin of round-off errors.

The Engineering Problem-Solving Process



FORMULATION

Fundamental laws explained briefly

SOLUTION

Elaborate and often complicated method to make problem tractable

INTERPRETATION

In-depth analysis limited by time-consuming solution

(*a*)

FORMULATION

In-depth exposition of relationship of problem to fundamental laws

SOLUTION

Easy-to-use computer method

INTERPRETATION

Ease of calculation allows holistic thoughts and intuition to develop; system sensitivity and behavior can be studied

(*b*)

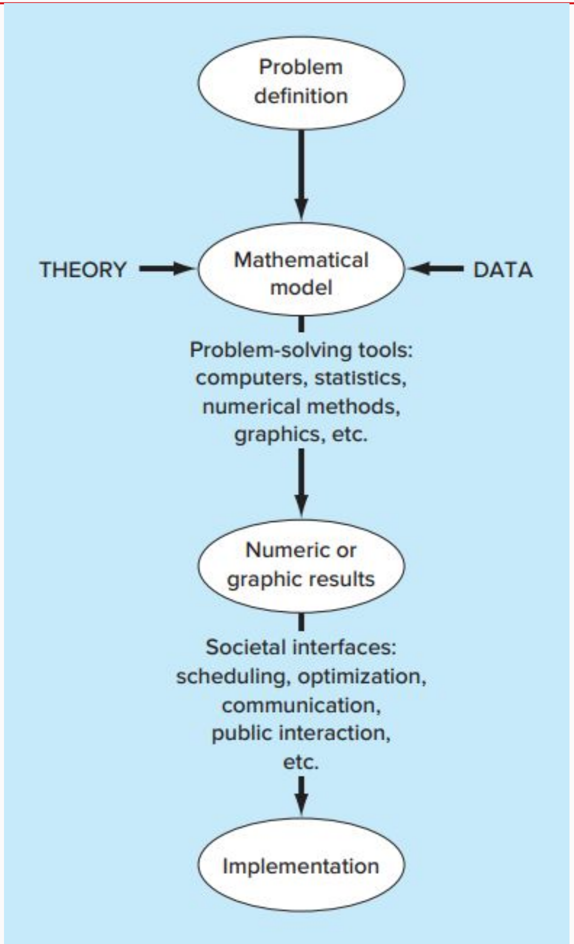
FIGURE PT1.1

The three phases of engineering problem solving in (*a*) the precomputer and (*b*) the computer era. The sizes of the boxes indicate the level of emphasis directed toward each phase. Computers facilitate the implementation of solution techniques and thus allow more emphasis to be placed on the creative aspects of problem formulation and interpretation of results.

The Engineering Problem-Solving Process

- As engineers, we are problem solvers. But many real-world problems are too **complex** for a simple, exact **textbook** solution.
    - *“Can we exactly solve airflow around a drone blade?”*
    - *“Can we derive a closed-form equation for heat loss in a complex building?”*
- We cannot solve these with simple algebra.
- **The Two-Pronged Approach:**
    - **Empiricism (Observation & Experiment):** We collect data from the real world. This is essential but can be expensive, time-consuming, and limited to specific conditions.
    - **Theoretical Analysis (Fundamental Laws):** Over time, we've discovered general laws that govern physical behavior—Newton's Laws, Conservation of Mass/Energy, Kirchhoff's Laws, etc.
- **These two approaches feed each other -** Observations inform theories, and theories guide new experiments.
- **The Mathematical Model -** This is the tool that allows us to take our theoretical understanding and make quantitative predictions.

The Engineering Problem-Solving Process


Problem definition
THEORY
Mathematical model
DATA
Problem-solving tools: computers, statistics, numerical methods, graphics, etc.
Numeric or graphic results
Societal interfaces: scheduling, optimization, communication, public interaction, etc.
Implementation

What is a Mathematical Model?

- A mathematical model is a formulation, an equation, that expresses the essential features of a physical system or process in mathematical terms.

- It is an idealization and simplification of reality. We ignore negligible details to focus on the core behavior. (e.g., when modeling a falling object near the earth, we ignore relativistic effects).
- General Functional Relationship :

Dependent variable = f(independent variables, parameters, forcing functions)

- **Dependent Variable:** A characteristic that reflects the system's behavior or state (e.g., velocity, temperature, stress). This is often what we want to predict.
- **Independent Variables:** Dimensions along which the system's behavior is determined (e.g., time, space).
- **Parameters:** Reflect the system's properties or composition (e.g., mass, drag coefficient, resistance).
- **Forcing Functions:** External influences acting on the system (e.g., gravitational force, applied voltage).

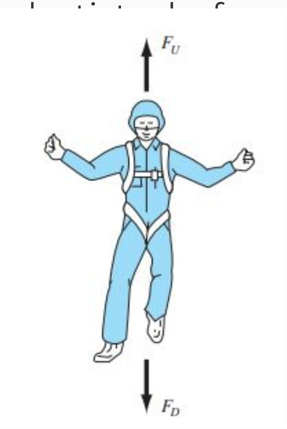
What is a Mathematical Model?

Newton's Second Law

- Physical Law: The rate of change of momentum is equal to the net force.
- Mathematical Model: F = ma
- Let's put it in the form of Eq. :
  - **a = F/m**
- **Dependent Variable:** Acceleration, a (the behavior we're predicting).
- **Forcing Function:** Net Force, F (the external influence).
- **Parameter:** Mass, m (a property of the system).
- **Independent Variable:** In this simple form, there isn't one. We are not predicting how acceleration varies over time or space yet.

Case Study: The Falling Parachutist

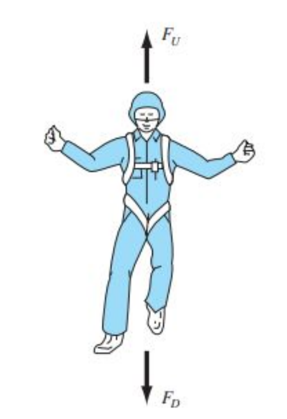
- Let's develop a model to predict the velocity of a free-falling para the chute opens.
- Two primary forces act on them:
  - Downward force (positive direction): Gravity, F_D.
  - Upward force (negative direction): Air Resistance, F_U.



Apply a Fundamental Law

- We apply Newton's Second Law in the vertical direction:
  - Net Force = mass × acceleration.
  - **F = F_D + F_U = m * a** ... but a = dv/dt.
  - So, m * (dv/dt) = F_D + F_U. (Eq. A)

Case Study: The Falling Parachutist



Formulate the Forces (Model the Components)

- Force due to Gravity: **F_D = m * g** where g ≈ 9.81 m/s^2.
- Force due to Air Resistance: This is where modeling comes in. model assumes it's linearly proportional to velocity.
    - **F_U = -c * v**
    - c is the drag coefficient (a parameter that depends on shape, suit, etc.).
    - The negative sign indicates this force acts upward, opposing the motion.

Develop the Complete Mathematical Model

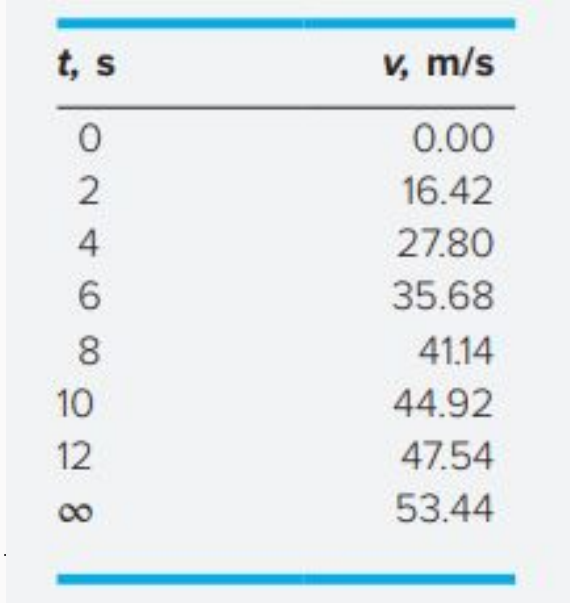
- Substitute the forces back into Eq. (A):
- m * (dv/dt) = mg - cv
- Divide both sides by m:
- **dv/dt = g - (c/m) * v** (The Governing Differential Equation)
- This is our model. It describes how the velocity v changes with time t.

Case Study: The Falling Parachutist

Solve the Model - The Analytical Solution

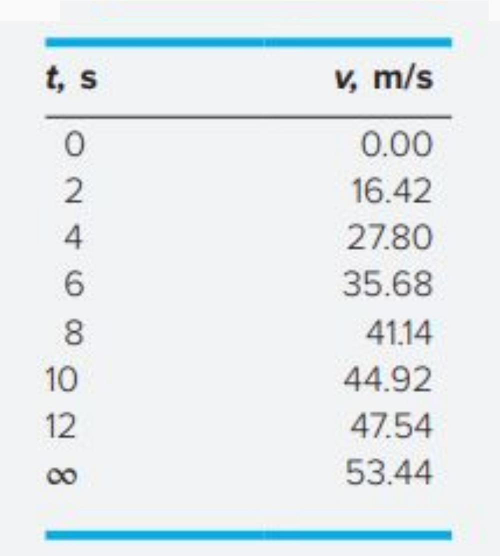
- "For this specific case, we can use calculus to find an exact, or analytical solution."
- Given the initial condition v(0) = 0, the solution is:
  - $\boldsymbol{v(t) = (g * m / c) * (1 - e^{-(c/m)t})}$
- Walk through Example:
  - m = 68.1 kg, c = 12.5 kg/s, g = 9.81 m/s².
  - $v(t) = (9.81*68.1/12.5) * (1 - e^{-(12.5/68.1)t})$

    = 53.44 * (1 - e^(-0.18355t))
- Key Observations from the Solution:
  - Velocity increases over time.
  - It approaches a terminal velocity: v_term = gm/c.
  - This happens when the net dt=0), meaning gravity is balanced by air resistance.

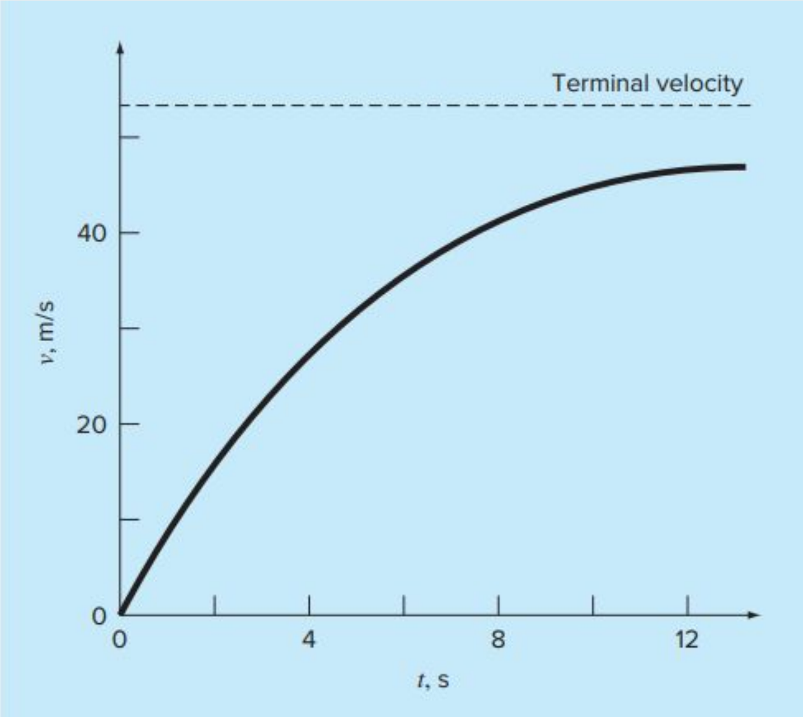


t, s	v, m/s
0	0.00
2	16.42
4	27.80
6	35.68
8	41.14
10	44.92
12	47.54
∞	53.44

The Falling Parachutist: Analytical Solution



t, s	*v*, m/s
0	0.00
2	16.42
4	27.80
6	35.68
8	41.14
10	44.92
12	47.54
∞	53.44



Case Study: The Falling Parachutist

- Equation above is called an analytical, or exact, solution because it exactly satisfies the original differential equation.
- Unfortunately, there are many mathematical models that cannot be solved exactly.
- In many of these cases, the only alternative is to develop a numerical solution that approximates the exact solution.
- As mentioned previously, numerical methods are those in which the mathematical problem is reformulated so that it can be solved by arithmetic operations.
- The Core Idea: We can approximate the continuous derivative dv/dt with a finite difference.
    - dv/dt ≈ Δv/Δt = (v(t_i+1) - v(t_i)) / (t_i+1 - t_i)
- This approximates the slope of the tangent with the slope of a secant line.

Case Study: The Falling Parachutist

Derive the Formula:

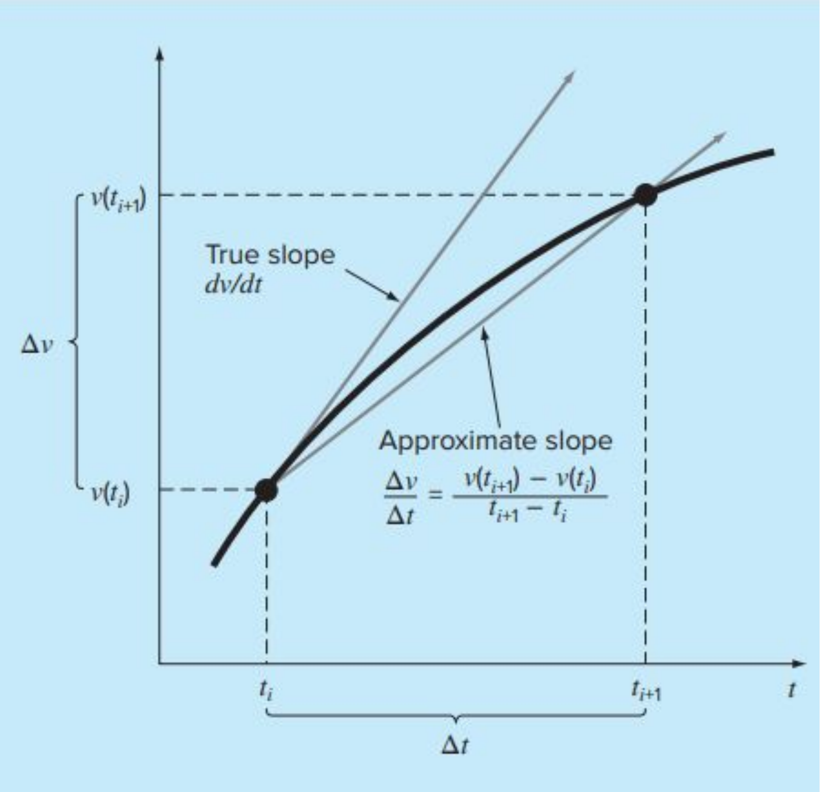
- We start with the model: **dv/dt = g - (c/m)v.**
- Substitute the finite difference approximation:
    - (v(t_i+1) - v(t_i)) / (t_i+1 - t_i) ≈ g - (c/m)v(t_i)
- Solve for the future velocity, v(t_i+1):
- v(t_i+1) ≈ v(t_i) + [g - (c/m)v(t_i)] * (t_i+1 - t_i)
- Let Δt = t_i+1 - t_i. The final formula is:
- v_new = v_old + [g - (c/m) * v_old] * Δt

  (Eq. - **Euler's Method**)

This is incredibly powerful!

New Value = Old Value + Slope × Step Size.

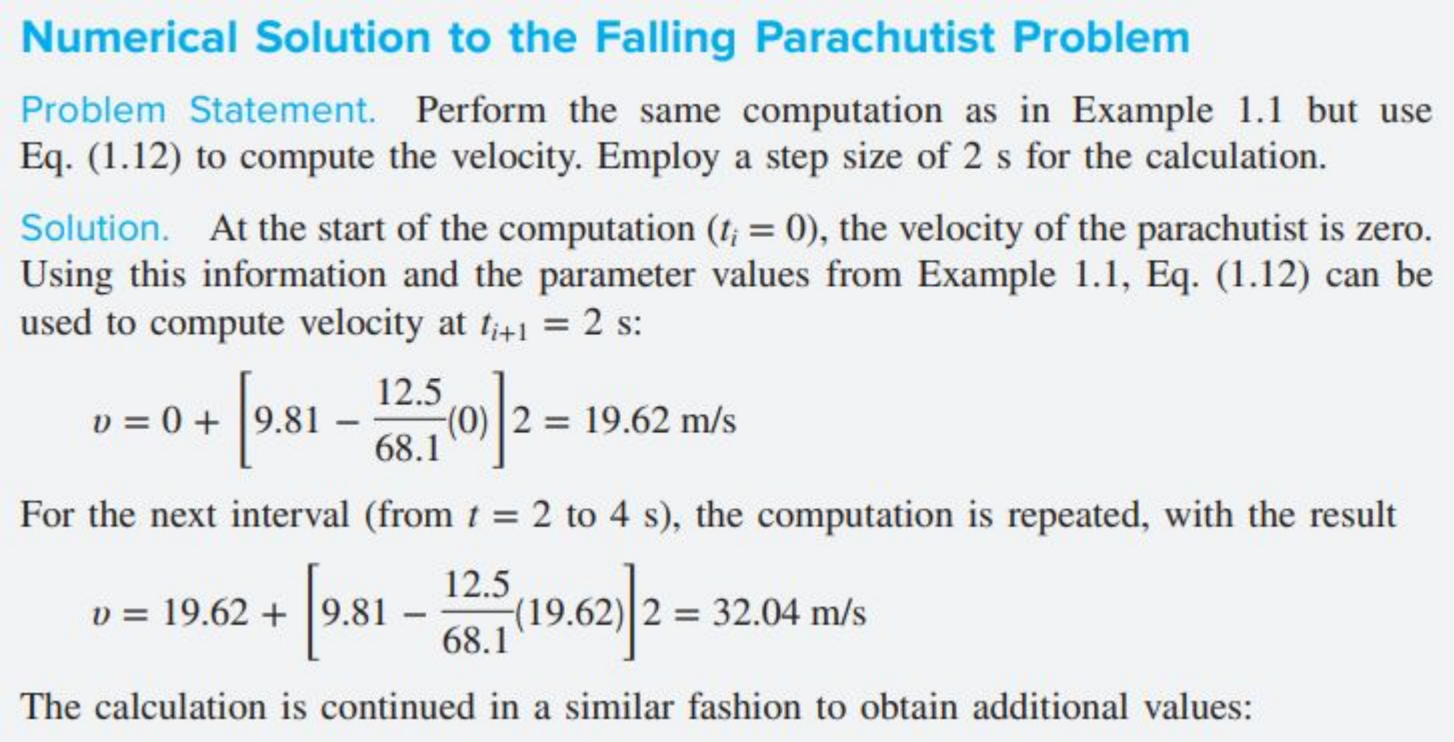
We can start with a known initial velocity and simply march forward in time, step by step."



Case Study: The Falling Parachutist

- Equation above is called an analytical, or exact, solution because it exactly satisfies the original differential equation.
- Unfortunately, there are many mathematical models that cannot be solved exactly.
- In many of these cases, the only alternative is to develop a numerical solution that approximates the exact solution.
- As mentioned previously, numerical methods are those in which the mathematical problem is reformulated so that it can be solved by arithmetic operations.
- The Core Idea: We can approximate the continuous derivative dv/dt with a finite difference.
    - $dv/dt \approx \Delta v/\Delta t = (v(t_{i+1}) - v(t_i)) / (t_{i+1} - t_i)$
- This approximates the slope of the tangent with the slope of a secant line.

Case Study: The Falling Parachutist



Numerical Solution to the Falling Parachutist Problem

Problem Statement. Perform the same computation as in Example 1.1 but use Eq. (1.12) to compute the velocity. Employ a step size of 2 s for the calculation.

Solution. At the start of the computation ($t_i = 0$), the velocity of the parachutist is zero. Using this information and the parameter values from Example 1.1, Eq. (1.12) can be used to compute velocity at $t_{i+1} = 2$ s:

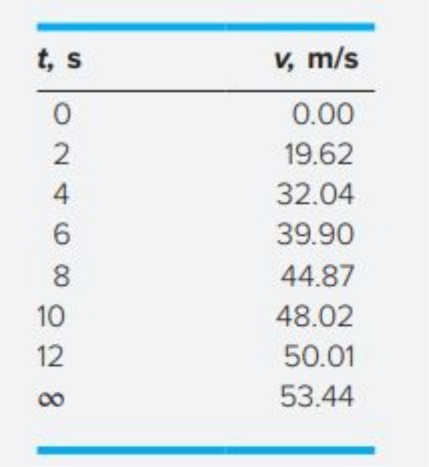
$$\upsilon = 0 + \left[9.81 - \frac{12.5}{68.1}(0)\right]2 = 19.62 \text{ m/s}$$

For the next interval (from $t = 2$ to 4 s), the computation is repeated, with the result

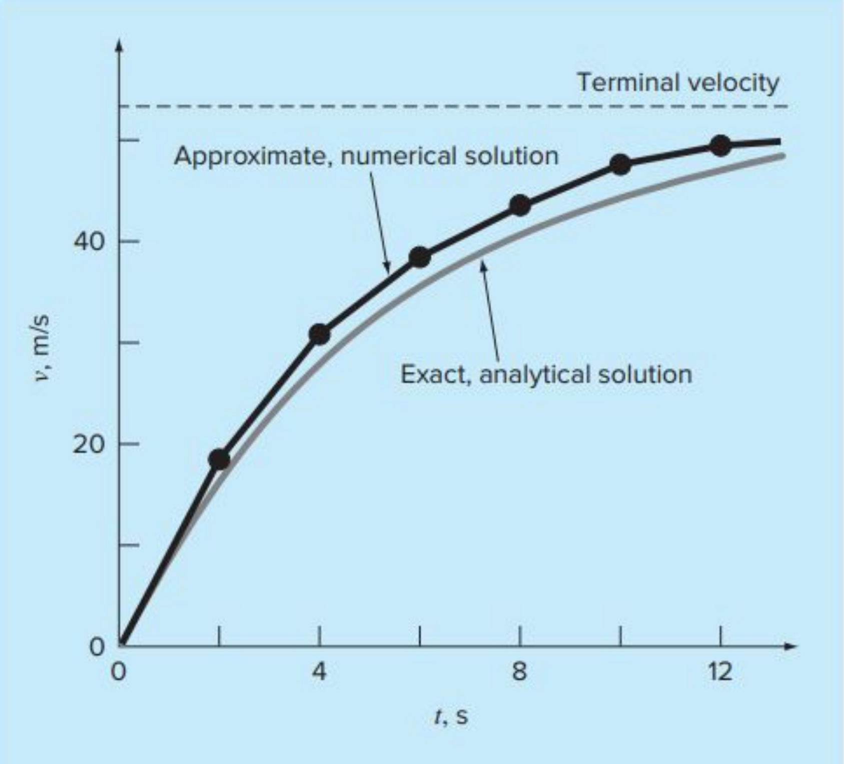
$$\upsilon = 19.62 + \left[9.81 - \frac{12.5}{68.1}(19.62)\right]2 = 32.04 \text{ m/s}$$

The calculation is continued in a similar fashion to obtain additional values:

The Falling Parachutist: Numerical Solution



t, s	*v*, m/s
0	0.00
2	19.62
4	32.04
6	39.90
8	44.87
10	48.02
12	50.01
∞	53.44

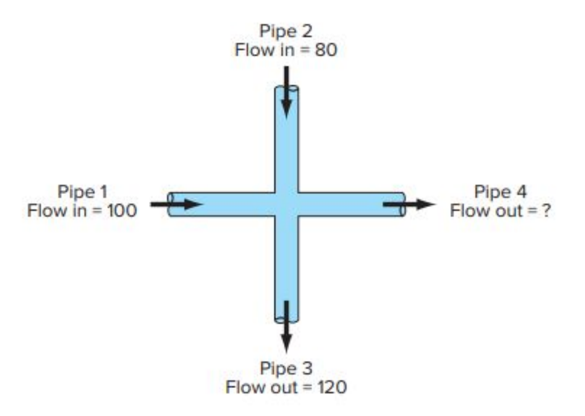


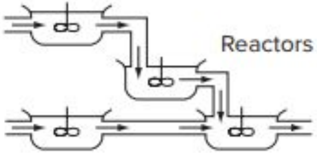
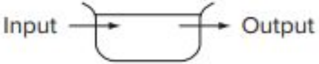
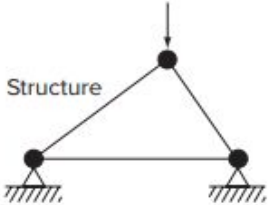
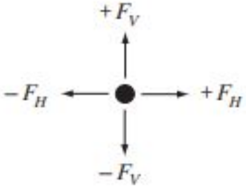
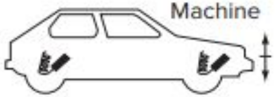
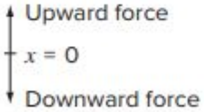
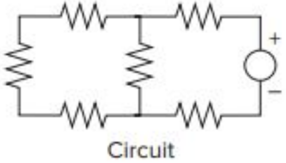
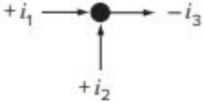
Summary: The Falling Parachutist

- It can be seen that the numerical method captures the essential features of the exact solution.
- However, because we have employed straight-line segments to approximate a continuously curving function, there is some discrepancy between the two results.
- One way to minimize such discrepancies is to use a smaller step size. For example, applying Eq. at l-s intervals results in a smaller error, as the straight-line segments track closer to the true solution.
- Using hand calculations, the effort associated with using smaller and smaller step sizes would make such numerical solutions impractical.
- However, with the aid of the computer, large numbers of calculations can be performed easily.
- Thus, you can accurately model the velocity of the falling parachutist without having to solve the differential equation exactly.
- A computational price must be paid for a more accurate numerical result.
- Thus, we see that there is a trade-off between accuracy and computational effort

Conservation Laws in Engineering

- "Newton's Second Law is one organizing principle. Another powerful set of tools are the Conservation Laws."
- The General Balance: Change = Increases - Decreases
- **Two Fundamental Modes:**
  - **Time-Variable (Transient):** Change ≠ 0. We predict how a system evolves over time (like the parachutist).
  - **Steady-State:** Change = 0, so Increases = Decreases. The system is in equilibrium. (e.g., for the parachutist, this gave us terminal velocity: mg = cv).



Field	Device	Organizing Principle	Mathematical Expression
Chemical engineering		Conservation of mass	Mass balance:  Over a unit of time period Δmass = inputs − outputs
Civil engineering		Conservation of momentum	Force balance:  At each node Σ horizontal forces (F_H) = 0 Σ vertical forces (F_V) = 0
Mechanical engineering		Conservation of momentum	Force balance:  $m\dfrac{d^2x}{dt^2}$ = downward force − upward force
Electrical engineering		Conservation of charge	Current balance:  For each node Σ current (i) = 0

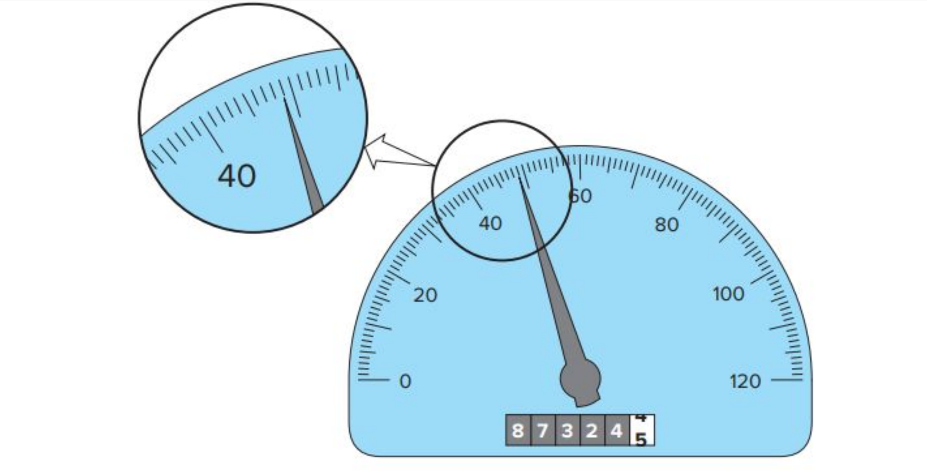
Approximations and Round-Off Errors

This section shifts from modeling to the practical realities of computation.

The goal is to make students aware that :-

> “
>
> computers are not perfect mathematical machines and that understanding their limitations is crucial for an engineer.

Significant Figures



- Visual inspection of the speedometer indicates that the car is traveling between 48 and 49 km/h.
- Because the indicator is higher than the midpoint between the markers on the gauge, we can say with assurance that the car is traveling at approximately 49 km/h.
- However, let us say that we insist that the speed be estimated to one decimal place, one person might say 48.8, whereas another might say 48.9 km/h

Significant Figures

- Therefore, because of the limits of this instrument, only the first two digits can be used with confidence.
- Estimates of the third digit (or higher) must be viewed as approximations.
- It would be ludicrous to claim, on the basis of this speedometer, that the automobile is traveling at 48.8642138 km/h.

In contrast

- The odometer provides up to six certain digits.
- From Fig. we can conclude that the car has traveled slightly less than 87,324.5 km during its lifetime. In this case, the seventh digit (and higher) is uncertain.

The concept of a significant figure, or digit, has been developed to formally designate the reliability of a numerical value.

Significant Figures

The significant digits of a number are those that can be used with confidence. They correspond to the number of certain digits plus one estimated digit.

- For example, the speedometer and the odometer in Fig. yield readings of three and seven significant figures, respectively.
- For the speedometer, the two certain digits are 48.
- It is conventional to set the estimated digit at one-half of the smallest scale division on the measurement device.
- Thus the speedometer reading would consist of the three significant figures: 48.5.
- In a similar fashion, the odometer would yield a seven significant-figure reading of 87,324.45.

Significant Figures

- Although it is usually a straightforward procedure to ascertain the significant figures of a number, some cases can lead to confusion.
- For example, zeros are not always significant figures because they may be necessary just to locate a decimal point.
- The numbers 0.00001845, 0.0001845, and 0.001845 all have four significant figures.
- Similarly, when trailing zeros are used in large numbers, it is not clear how many, if any, of the zeros are significant.
- For example, at face value the number 45,300 may have three, four, or five significant digits, depending on whether the zeros are known with confidence.
- Such uncertainty can be resolved by using scientific notation, where 4.53×10^4, 4.530×10^4, 4.5300×10^4 designate that the number is known to three, four, and five significant figures, respectively.

Summary: Significant Figures

- In science and engineering, many real-world problems cannot be solved exactly.
- Examples:
    - Solving nonlinear equations,
    - Integrating complex functions,
    - Modeling motion (e.g., a falling parachutist with air resistance).
- To handle such problems, we use numerical methods — algorithms that produce approximate (not exact) solutions.
- Since numerical methods give approximations, we must ask:
    - How accurate is my result?
    - Can I trust this number?
- Significant figures provide a way to express that confidence. They tell us how many digits in a computed value are reliably correct.

Summary: Significant Figures

Suppose the true velocity of a parachutist is 49.2568 m/s, but your numerical method gives 49.26 m/s.

This result is correct to four significant figures. The difference (error) is in the fifth digit and beyond.

Hence, we can confidently say the result is accurate to 4 significant figures.

- So, instead of saying:
  - "My result is approximately 49.26 m/s."
- you can say:
  - "My result is accurate to four significant figures."

This gives a quantitative measure of precision — not just a vague sense of "approximate."

Summary: Significant Figures

Exact Constants (π, e, √7) Cannot Be Represented Exactly — Round-Off Error

a. Infinite Non-Repeating Values

Certain numbers like π, e, and irrational roots (e.g., √7) have infinite, non-repeating decimal expansions:

- π = 3.1415926535897932384626433832795... (and so on forever)
- e = 2.718281828459045235360287...
- 7 = 2.64575131106459059...

These numbers have no exact finite decimal form — they go on infinitely.

b. Computer Limitation

Computers have finite memory and can only store a limited number of digits. For instance, a 64-bit floating-point variable in most programming languages stores around 15–17 significant digits.

Thus, when you write in code:

- pi = 3.141592653589793

the computer **truncates or rounds the infinite true value.**

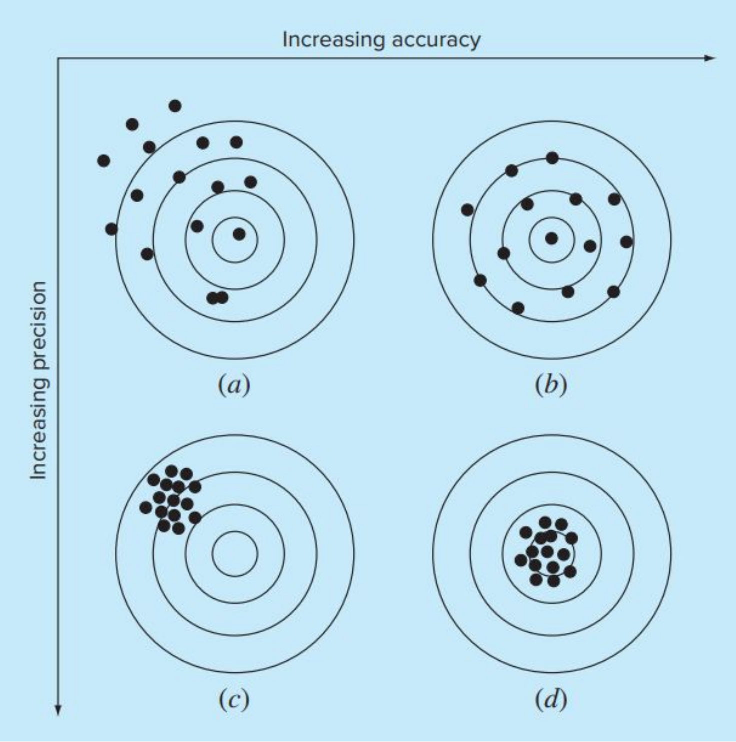
Accuracy and Precision

- Accuracy and Precision are not interchangeable in engineering and science.

- The errors associated with both calculations and measurements can be characterized with regard to their **accuracy and precision.**

- **Accuracy** refers to how closely a computed or measured value agrees with the true value.

- **Precision** refers to how closely individual computed or measured values agree with each other.



Error Definitions: Quantifying the Doubt

To manage error, we must first be able to measure it. We use several standard formulas.

- Numerical errors arise from the use of approximations to represent exact mathematical operations and quantities.
- These include **truncation errors**, which result when approximations are used to represent exact mathematical procedures, and **round-off errors**, which result when numbers having limited significant figures are used to represent exact numbers.
- For both types, the relationship between the exact, or true, result and the approximation can be formulated as:
  - **True value = approximation + error**
- By rearranging Eq. , we find that the numerical error is equal to the discrepancy between the true value and the approximation, as in
  - **E_t = true value − approximation**

where E_t is used to designate the exact value of the error. The subscript t is included to designate that this is the “true” error.

Error Definitions: Quantifying the Doubt

- A shortcoming of this definition is that it takes no account of the order of magnitude of the value under examination.
- For example, an error of a centimeter is much more significant if we are measuring a rivet rather than a bridge.
- One way to account for the magnitudes of the quantities being evaluated is to normalize the error to the true value, as in

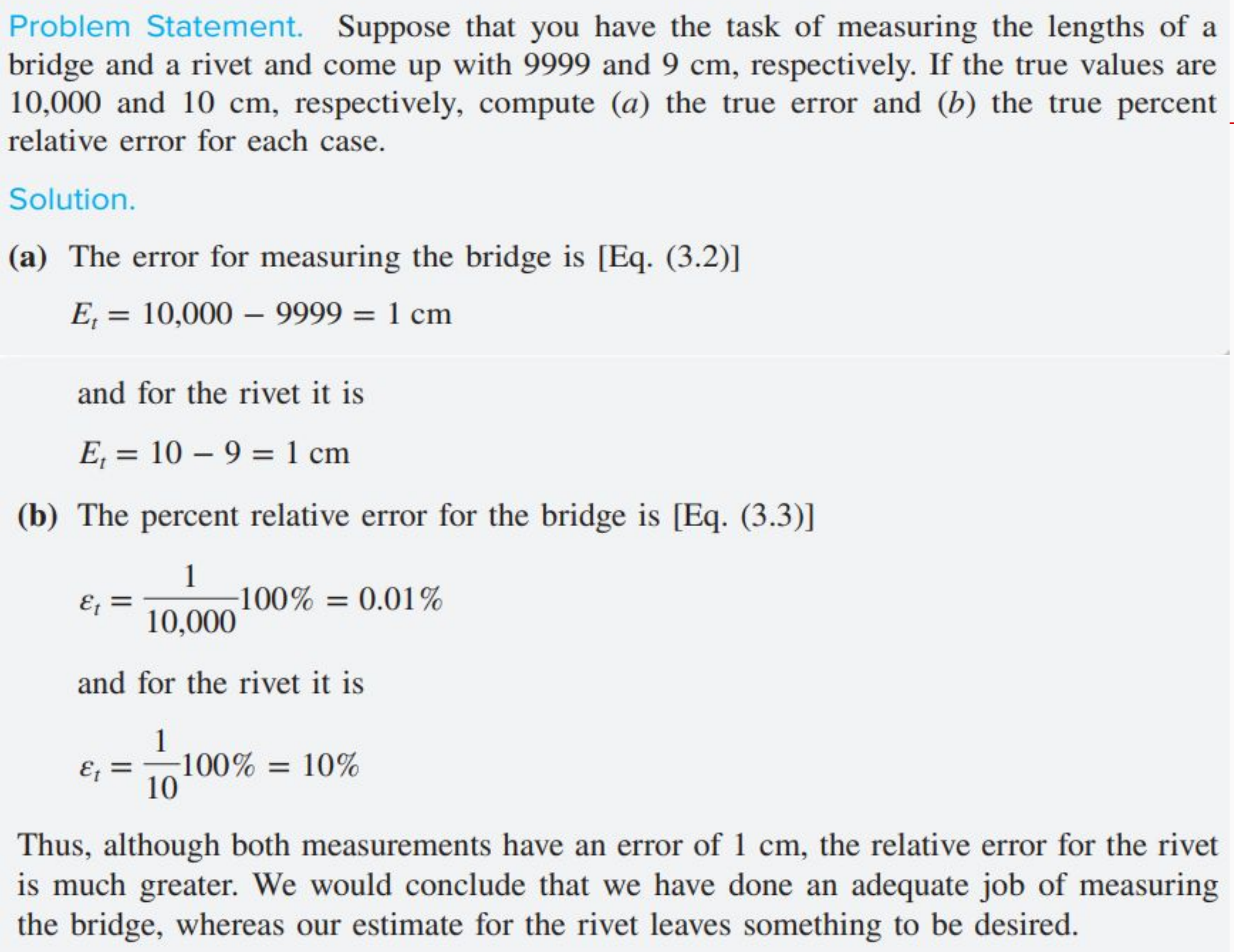
True fractional relative error = true error / true value

where, as specified by Eq. , error = true value − approximation.

The relative error can also be multiplied by 100 percent to express it as:

$$\varepsilon_t = (\text{true error} / \text{true value}) * 100\%$$

where ε_t designates the true percent relative error.



Problem Statement. Suppose that you have the task of measuring the lengths of a bridge and a rivet and come up with 9999 and 9 cm, respectively. If the true values are 10,000 and 10 cm, respectively, compute (*a*) the true error and (*b*) the true percent relative error for each case.

Solution.

(a) The error for measuring the bridge is [Eq. (3.2)]

$$E_t = 10{,}000 - 9999 = 1 \text{ cm}$$

and for the rivet it is

$$E_t = 10 - 9 = 1 \text{ cm}$$

(b) The percent relative error for the bridge is [Eq. (3.3)]

$$\varepsilon_t = \frac{1}{10{,}000} 100\% = 0.01\%$$

and for the rivet it is

$$\varepsilon_t = \frac{1}{10} 100\% = 10\%$$

Thus, although both measurements have an error of 1 cm, the relative error for the rivet is much greater. We would conclude that we have done an adequate job of measuring the bridge, whereas our estimate for the rivet leaves something to be desired.

Error Definitions: Quantifying the Doubt

- E and ε are subscripted with a t to signify that the error is normalized to the true value.
- However, in actual situations such information is rarely available.
- For numerical methods, the true value will be known only when we deal with functions that can be solved analytically.
- Such will typically be the case when we investigate the theoretical behavior of a particular technique for simple systems.
- However, in real-world applications, we will obviously not know the true answer a priori.
- For these situations, an alternative is to normalize the error using the best available estimate of the true value, that is, to the approximation itself, as in
    - **ε_a = (approximate error / approximation) * 100%**

  where the subscript a signifies that the error is normalized to an approximate value.

Error Definitions: Quantifying the Doubt

One of the challenges of numerical methods is to determine error estimates in the absence of knowledge regarding the true value.

- For example, certain numerical methods use an iterative approach to compute answers.
- In such an approach, a present approximation is made on the basis of a previous approximation.
- This process is performed repeatedly, or iteratively, to successively compute (we hope) better and better approximations.
- For such cases, the error is often estimated as the difference between previous and current approximations.
- Thus, percent relative error is determined according to
  - **ε_a = ((current approximation – previous approximation) / current approximation) * 100%**

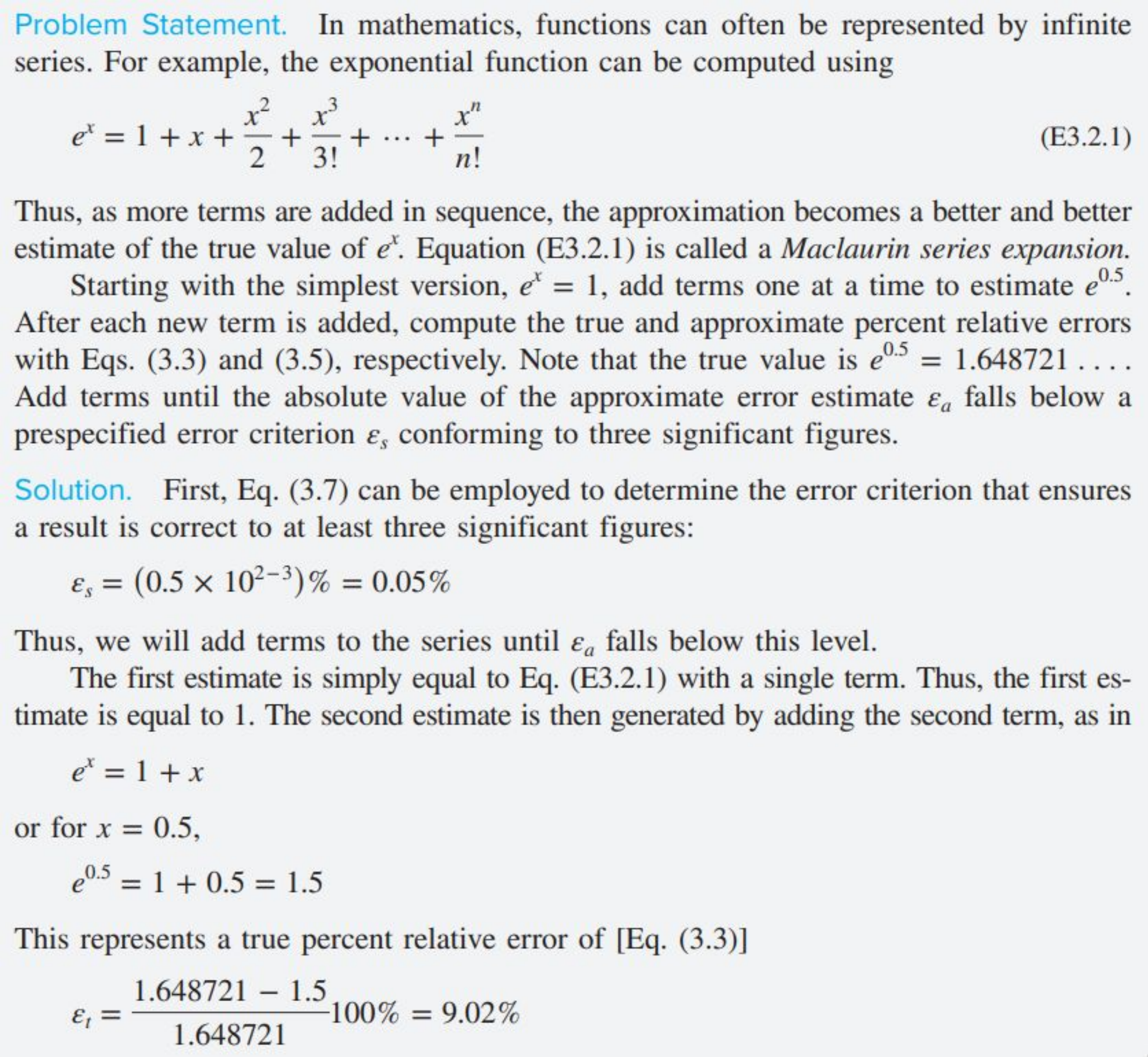
Often, when performing computations, we may not be concerned with the sign of the error, but we are interested in whether the percent absolute value is lower than a prespecified percent tolerance **ε_s.**

Error Definitions: Quantifying the Doubt

The computation is repeated until

$$|\varepsilon_a| < \varepsilon_s$$

- If this relationship holds, our result is assumed to be within the prespecified acceptable level ε_s.
- Note that for the remainder of this text, we will almost exclusively employ absolute values when we use relative errors.
- It is also convenient to relate these errors to the number of significant figures in the approximation.
- It can be shown (Scarborough 1966) that if the following criterion is met, we can be assured that the result is correct to at least n significant figures.
  - $\varepsilon_s = (0.5 \times 10^{2-n})\ \%$



Problem Statement. In mathematics, functions can often be represented by infinite series. For example, the exponential function can be computed using

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} \tag{E3.2.1}$$

Thus, as more terms are added in sequence, the approximation becomes a better and better estimate of the true value of e^x. Equation (E3.2.1) is called a *Maclaurin series expansion.*

Starting with the simplest version, $e^x = 1$, add terms one at a time to estimate $e^{0.5}$. After each new term is added, compute the true and approximate percent relative errors with Eqs. (3.3) and (3.5), respectively. Note that the true value is $e^{0.5} = 1.648721 \ldots$. Add terms until the absolute value of the approximate error estimate ε_a falls below a prespecified error criterion ε_s conforming to three significant figures.

Solution. First, Eq. (3.7) can be employed to determine the error criterion that ensures a result is correct to at least three significant figures:

$$\varepsilon_s = (0.5 \times 10^{2-3})\% = 0.05\%$$

Thus, we will add terms to the series until ε_a falls below this level.

The first estimate is simply equal to Eq. (E3.2.1) with a single term. Thus, the first estimate is equal to 1. The second estimate is then generated by adding the second term, as in

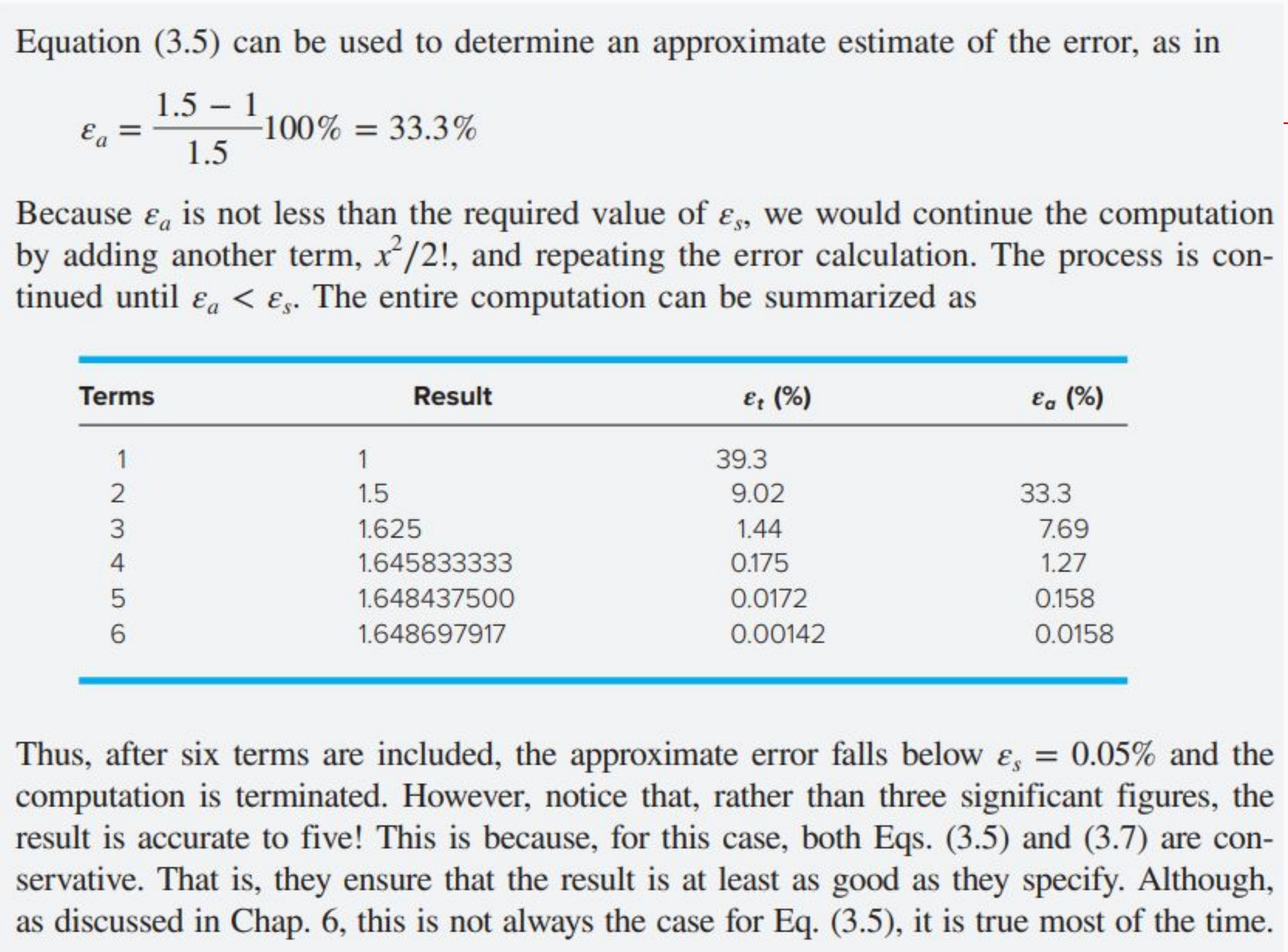
$$e^x = 1 + x$$

or for $x = 0.5$,

$$e^{0.5} = 1 + 0.5 = 1.5$$

This represents a true percent relative error of [Eq. (3.3)]

$$\varepsilon_t = \frac{1.648721 - 1.5}{1.648721} 100\% = 9.02\%$$



Equation (3.5) can be used to determine an approximate estimate of the error, as in

$$\varepsilon_a = \frac{1.5 - 1}{1.5}100\% = 33.3\%$$

Because ε_a is not less than the required value of ε_s, we would continue the computation by adding another term, $x^2/2!$, and repeating the error calculation. The process is continued until $\varepsilon_a < \varepsilon_s$. The entire computation can be summarized as

Terms	Result	ε_t (%)	ε_a (%)
1	1	39.3	
2	1.5	9.02	33.3
3	1.625	1.44	7.69
4	1.645833333	0.175	1.27
5	1.648437500	0.0172	0.158
6	1.648697917	0.00142	0.0158

Thus, after six terms are included, the approximate error falls below $\varepsilon_s = 0.05\%$ and the computation is terminated. However, notice that, rather than three significant figures, the result is accurate to five! This is because, for this case, both Eqs. (3.5) and (3.7) are conservative. That is, they ensure that the result is at least as good as they specify. Although, as discussed in Chap. 6, this is not always the case for Eq. (3.5), it is true most of the time.

Algorithm: Iterative Computation of e^x

check the repo for code

Given:

x (value for which e^x is to be computed),

e_s (stopping error criterion, e.g., 0.0001%),

maxit (maximum number of iterations).

Steps:

```
Initialize variables:
    iter = 0
    sol = 1.0 (first term of the series)
    term = 1.0
    ea = 100.0 (set high to start loop)
Loop while $e_a > e_s$ and iter < maxit:
    Increment iteration count: iter = iter + 1
    Compute next term:
        term = term * x / iter
    Save old solution: sol_old = sol
    Update solution: sol = sol + term
    Compute approximate error:
        $e_a$ = abs((sol - sol_old) / sol) * 100
End loop when stopping criterion met.
Output:
    Final value of sol
    Approximate error $e_a$
    Iteration count iter
```

Round-Off Errors

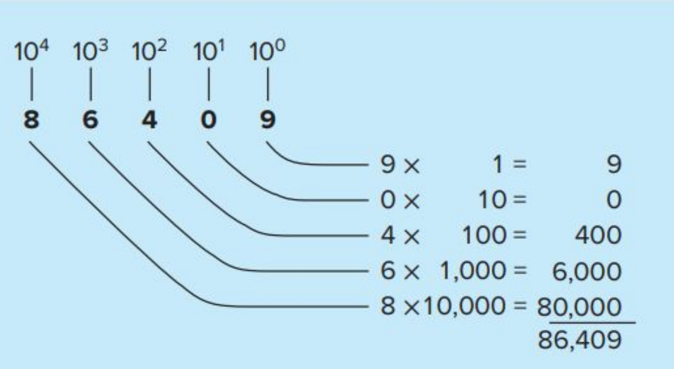
- Round-off errors originate from the fact that computers retain only a fixed number of significant figures during a calculation.
- Numbers such as π, e, or √7 cannot be expressed by a fixed number of significant figures.
- Therefore, they cannot be represented exactly by the computer.
- In addition, because computers use a base-2 representation, they cannot precisely represent certain exact base-10 numbers.
- The discrepancy introduced by this omission of significant figures is called **round-off error.**

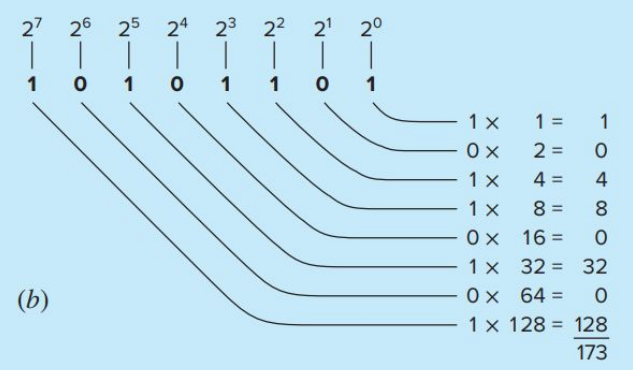
Computer Representation of Numbers

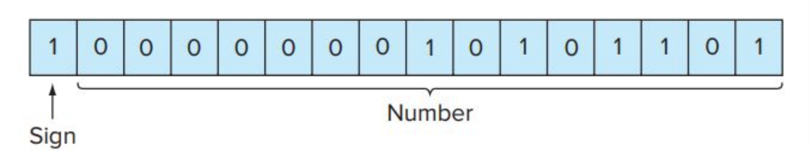
Numerical round-off errors are directly related to the manner in which numbers are stored in a Computer.

- The fundamental unit whereby information is represented is called a word.
- This is an entity that consists of a string of binary digits, or bits.
- Numbers are typically stored in one or more words.

To understand how this is accomplished, we must first review some material related to number systems.



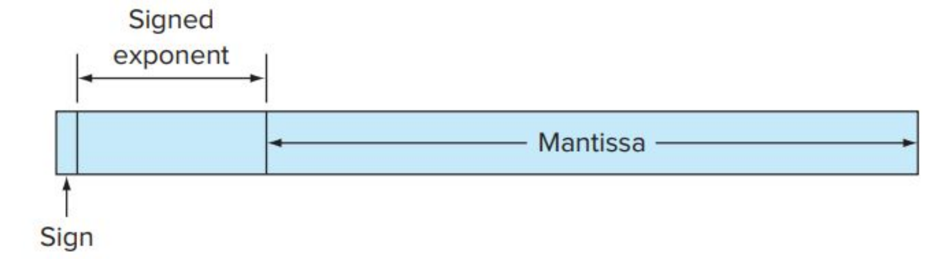




Floating-Point Representation

- Fractional quantities are typically represented in computers using floating-point form. In this approach, the number is expressed as a fractional part, called a mantissa or significand, and an integer part, called an exponent or characteristic, as in
    - $m \cdot b^e$

  where m = the mantissa, b = the base of the number system being used, and e = the exponent. For instance, the number 156.78 could be represented as 0.15678 × 10^3 in a floating-point base-10 system.
- Figure shows one way that a floating-point number could be stored in a word. The first bit is reserved for the sign, the next series of bits for the signed exponent, and the last bits for the mantissa.



Floating-Point Representation

Note that the mantissa is usually normalized if it has leading zero digits.

- For example, suppose the quantity 1/34 = 0.029411765 . . . was stored in a floating-point base-10 system that allowed only four decimal places to be stored.
- Thus, 1/34 would be stored as 0.0294×10^{0}
- However, in the process of doing this, the inclusion of the useless zero to the right of the decimal forces us to drop the digit 1 in the fifth decimal place.
- The number can be normalized to remove the leading zero by multiplying the mantissa by 10 and lowering the exponent by 1 to give 0.2941×10^{-1}

Thus, we retain an additional significant figure when the number is stored.

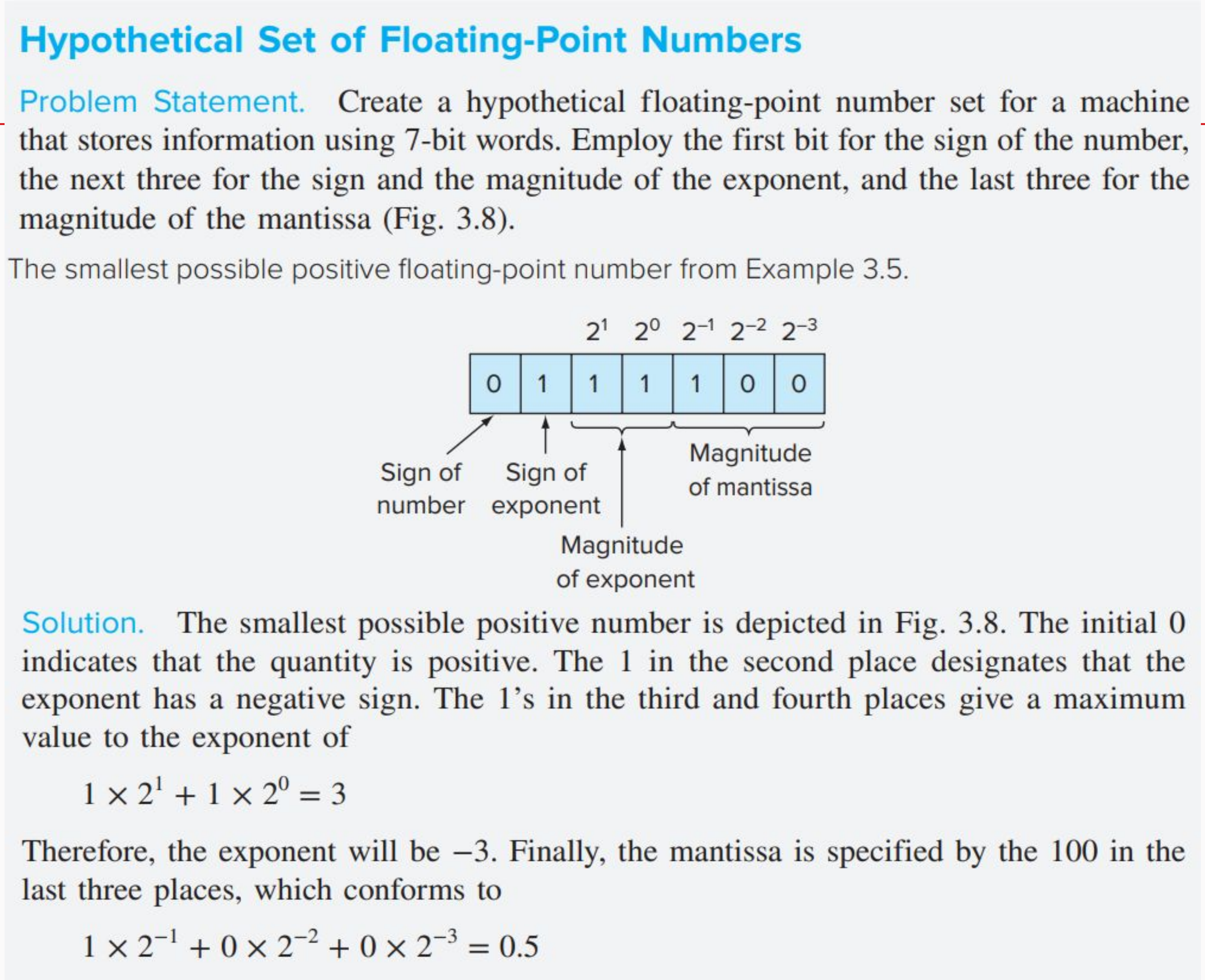
Floating-Point Representation

The consequence of normalization is that the absolute value of m is limited. That is,

$$1 / b \leq m < 1 \qquad \text{where } b = \text{the base.}$$

- For example, for a base-10 system, m would range between 0.1 and 1, and for a base-2 system, between 0.5 and 1.
- Floating-point representation allows both fractions and very large numbers to be expressed on the computer.
- However, it has some disadvantages. For example, floating-point numbers take up more room and take longer to process than integer numbers.
- More significantly, however, their use introduces a source of error because the mantissa holds only a finite number of significant figures.

Thus, a **round-off** error is introduced.



Hypothetical Set of Floating-Point Numbers

Problem Statement. Create a hypothetical floating-point number set for a machine that stores information using 7-bit words. Employ the first bit for the sign of the number, the next three for the sign and the magnitude of the exponent, and the last three for the magnitude of the mantissa (Fig. 3.8).

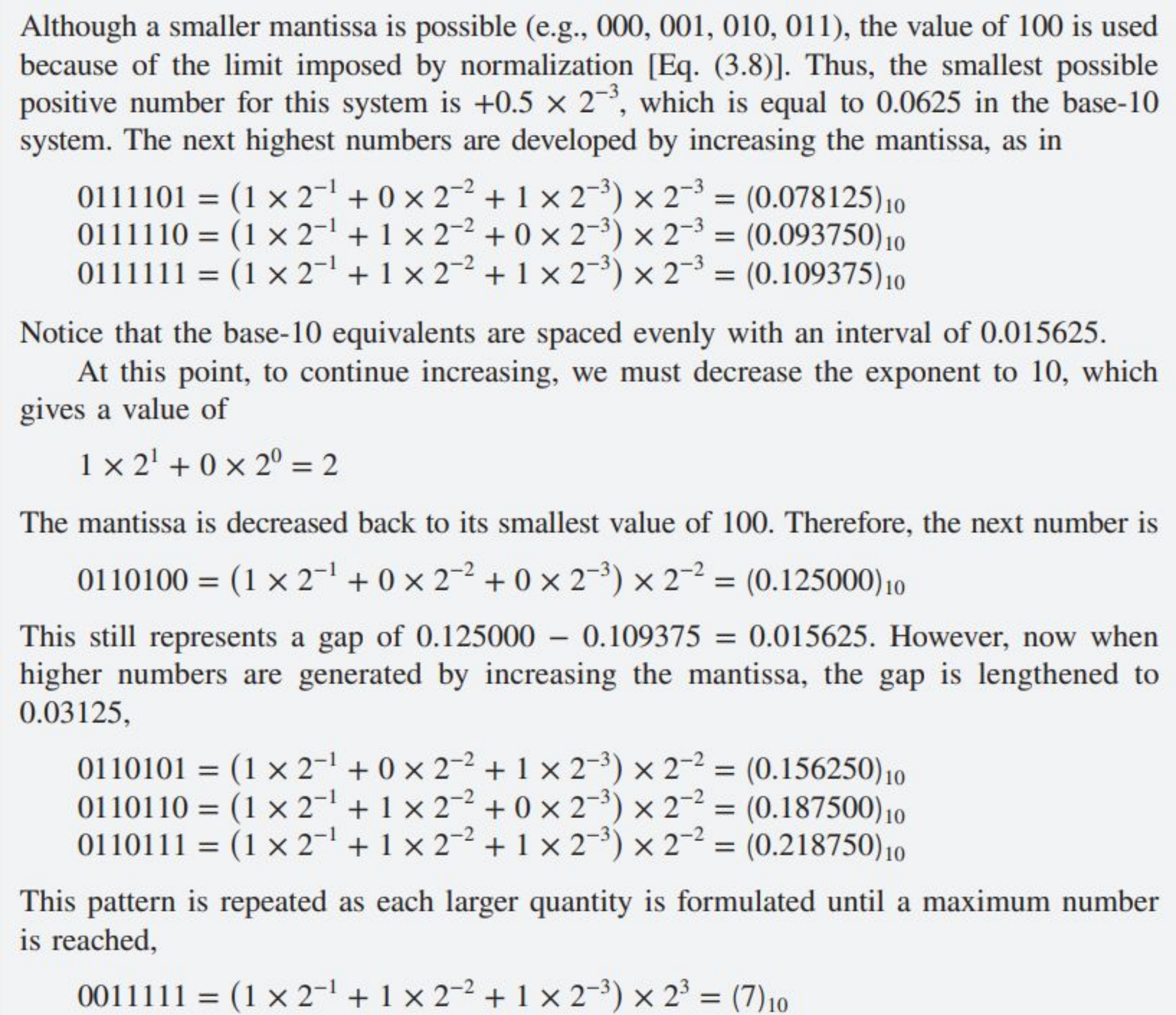
The smallest possible positive floating-point number from Example 3.5.

Solution. The smallest possible positive number is depicted in Fig. 3.8. The initial 0 indicates that the quantity is positive. The 1 in the second place designates that the exponent has a negative sign. The 1's in the third and fourth places give a maximum value to the exponent of

$$1 \times 2^1 + 1 \times 2^0 = 3$$

Therefore, the exponent will be −3. Finally, the mantissa is specified by the 100 in the last three places, which conforms to

$$1 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3} = 0.5$$



Although a smaller mantissa is possible (e.g., 000, 001, 010, 011), the value of 100 is used because of the limit imposed by normalization [Eq. (3.8)]. Thus, the smallest possible positive number for this system is $+0.5 \times 2^{-3}$, which is equal to 0.0625 in the base-10 system. The next highest numbers are developed by increasing the mantissa, as in

$$
\begin{aligned}
0111101 &= (1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}) \times 2^{-3} = (0.078125)_{10} \\
0111110 &= (1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3}) \times 2^{-3} = (0.093750)_{10} \\
0111111 &= (1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}) \times 2^{-3} = (0.109375)_{10}
\end{aligned}
$$

Notice that the base-10 equivalents are spaced evenly with an interval of 0.015625.

At this point, to continue increasing, we must decrease the exponent to 10, which gives a value of

$$1 \times 2^{1} + 0 \times 2^{0} = 2$$

The mantissa is decreased back to its smallest value of 100. Therefore, the next number is

$$0110100 = (1 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3}) \times 2^{-2} = (0.125000)_{10}$$

This still represents a gap of $0.125000 - 0.109375 = 0.015625$. However, now when higher numbers are generated by increasing the mantissa, the gap is lengthened to 0.03125,

$$
\begin{aligned}
0110101 &= (1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}) \times 2^{-2} = (0.156250)_{10} \\
0110110 &= (1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3}) \times 2^{-2} = (0.187500)_{10} \\
0110111 &= (1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}) \times 2^{-2} = (0.218750)_{10}
\end{aligned}
$$

This pattern is repeated as each larger quantity is formulated until a maximum number is reached,

$$0011111 = (1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}) \times 2^{3} = (7)_{10}$$

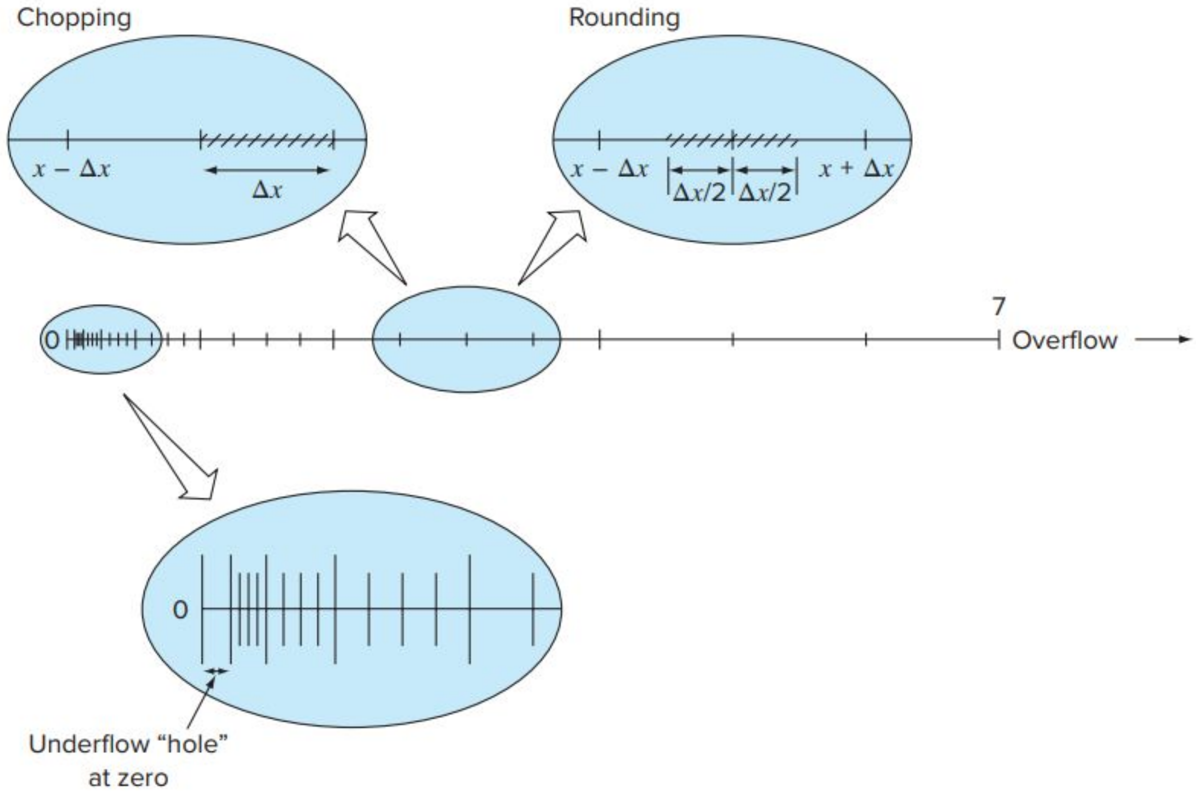


FIGURE 3.9
The hypothetical number system developed in Example 3.5. Each value is indicated by a tick mark. Only the positive numbers are shown. An identical set would also extend in the negative direction.

Limited Range — Overflow and Underflow

A floating-point system can represent only numbers within a finite range.

- **Overflow:** When a number is too large to be represented (e.g., 10^{400} in a system that can only go up to 10^{308}), a runtime error occurs.
- Underflow: When a number is too small (close to zero) to be represented, it becomes zero.
- The range between 0 and the smallest representable positive number is called the underflow hole.
- It exists because of normalization — floating-point numbers are stored so that their mantissa has a specific form (like 1.mmm... in binary).

Takeaway:

- Computers can’t handle infinitely large or infinitely small numbers — they “overflow” or “underflow.”

Finite Precision — Quantizing, Chopping, and Rounding

Because floating-point systems use a finite number of bits to store numbers, only a finite number of distinct values can exist between the smallest and largest representable values.

- Quantizing Error: Any real number must be approximated to the nearest representable value.
- Two methods for this approximation:
- a. Chopping : Simply discard extra digits beyond the available precision. Example:
  - π=3.14159265358
  - → Chopped to 7 significant figures: 3.141592   **Error E_t = 0.00000065**

Bias: All errors are positive (values always rounded down).

- b. Rounding : Adjust the last retained digit based on the next discarded one.
  - 3.14159265358 → 3.141593   **Error E_t = −0.00000035**
- No bias: Some errors go up, some down, More accurate (error ≤ half of chopping's).
- However, rounding adds a bit of computational cost, so some systems historically used chopping.

Takeaway: Rounding minimizes bias and produces smaller average error than chopping.

Variable Spacing — Interval Between Representable Numbers (Δx)

In floating-point systems, the spacing between consecutive representable numbers (Δx) increases as numbers get larger.

That’s because floating-point format keeps the same number of significant digits but shifts the decimal point (the “floating point”) to represent larger magnitudes.

Hence:

- Small numbers → closely spaced
- Large numbers → widely spaced
- Relative error remains roughly constant.

Takeaway

Machine epsilon defines how precisely a computer can represent real numbers — smaller ε means higher precision.

Thank you

https://github.com/realashok

dev.ashokbasnet@gmail.com

Lecture 2 - Round-Off, Truncation, and Taylor Series

Round-Off Errors

- Origin: Finite representation of numbers (floating-point systems).

Truncation Errors and the Taylor Series Introduction

- Difference between a mathematical operation and its approximation.
- Revisit Example 1.1 (Falling Parachutist) to show truncation in derivative approximation.
- Taylor's Theorem

Application of the Taylor Series

- Derivation of finite difference formulas using Taylor expansion.

Error Propagation and Total Numerical Error

- How round-off and truncation combine.
- Finding the optimal step size (trade-off between two errors).

Blunders and Data Uncertainty

- Distinction between: Blunders (human errors), Modeling/formulation errors, Data uncertainty.
- Engineering decision: how much error is acceptable?

The Taylor Series

- Truncation errors are those that result from using an approximation in place of an exact mathematical procedure.
- For example, in previous class we approximated the derivative of velocity of a falling parachutist by a finite-divided-difference equation of the form:

$$d\upsilon / dt \cong \Delta\upsilon / \Delta t = (\upsilon(t_{i+1}) - \upsilon(t_i)) / (t_{i+1} - t_i)$$

- A truncation error was introduced into the numerical solution because the difference equation only approximates the true value of the derivative.
- In order to gain insight into the properties of such errors, we now turn to a mathematical formulation that is used widely in numerical methods to express functions in an approximate fashion— the **Taylor series.**

The Taylor Series

Taylor's theorem and its associated formula, the Taylor series, are of great value in the study of numerical methods.

- In essence, the Taylor series provides a means to predict a function value at one point in terms of the function value and its derivatives at another point.
- In particular, the theorem states that any smooth function can be approximated as a polynomial. A useful way to gain insight into the Taylor series is to build it term by term.

For example, the first term in the series is

$$f(x_{i+1}) \cong f(x_i)$$

- This relationship, called the **zero-order approximation**, indicates that the value of f at the new point is the same as its value at the old point.
- This result makes intuitive sense because if x_i and x_{i+1} are close to each other, it is likely that the new value is probably similar to the old value.
- Eqn above provides a perfect estimate if the function being approximated is, in fact, a constant.

The Taylor Series

However, if the function changes at all over the interval, additional terms of the Taylor series are required to provide a better estimate.

- For example, the first-order approximation is developed by adding another term to yield

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

The additional first-order term consists of a slope ***f'(x$_i$)*** multiplied by the difference between x_{i+1} and x_i

- . Thus, the expression is now in the form of a straight line and is capable of predicting an increase or decrease of the function between xi and xi+1.
- Although Eq. above can predict a change, it is exact only for a straight-line, or linear, trend.
- Therefore, a second-order term is added to the series to capture some of the curvature that the function might exhibit:

$$f(x_{i+1}) \cong f(x_i) + f'(x_i)(x_{i+1} - x_i) + f''(x_i) / 2! \, (x_{i+1} - x_i)$$

The Taylor Series

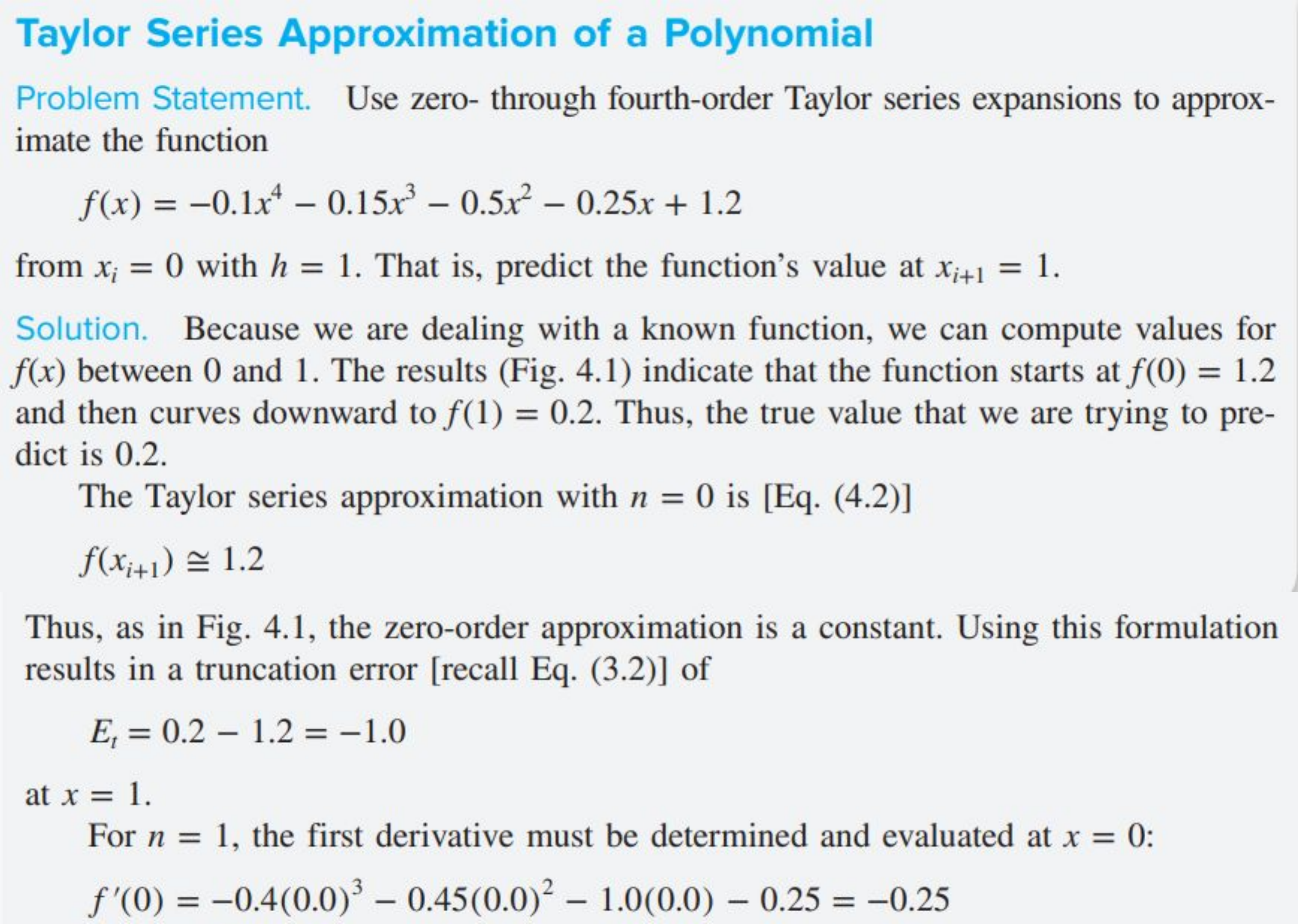
In a similar manner, additional terms can be included to develop the complete Taylor series expansion:

$$f(x_{i+1}) = f(x_i) + f'(x_i)h + \frac{f''(x_i)}{2!}h^2 + \frac{f^{(3)}(x_i)}{3!}h^3 + \cdots + \frac{f^{n}(x_i)}{n!}h^n + R_n$$

- A remainder term is included to account for all terms from n + 1 to infinity:

$$R_n = \frac{f^{(n+1)}(\xi)}{(n+1)!}h^{n+1}$$

where the subscript n connotes that this is the remainder for the nth-order approximation and **ξ** is a value of x that lies somewhere between x_i and x_{i+1}.



Taylor Series Approximation of a Polynomial

Problem Statement. Use zero- through fourth-order Taylor series expansions to approximate the function

$$f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$$

from $x_i = 0$ with $h = 1$. That is, predict the function's value at $x_{i+1} = 1$.

Solution. Because we are dealing with a known function, we can compute values for $f(x)$ between 0 and 1. The results (Fig. 4.1) indicate that the function starts at $f(0) = 1.2$ and then curves downward to $f(1) = 0.2$. Thus, the true value that we are trying to predict is 0.2.

The Taylor series approximation with $n = 0$ is [Eq. (4.2)]

$$f(x_{i+1}) \cong 1.2$$

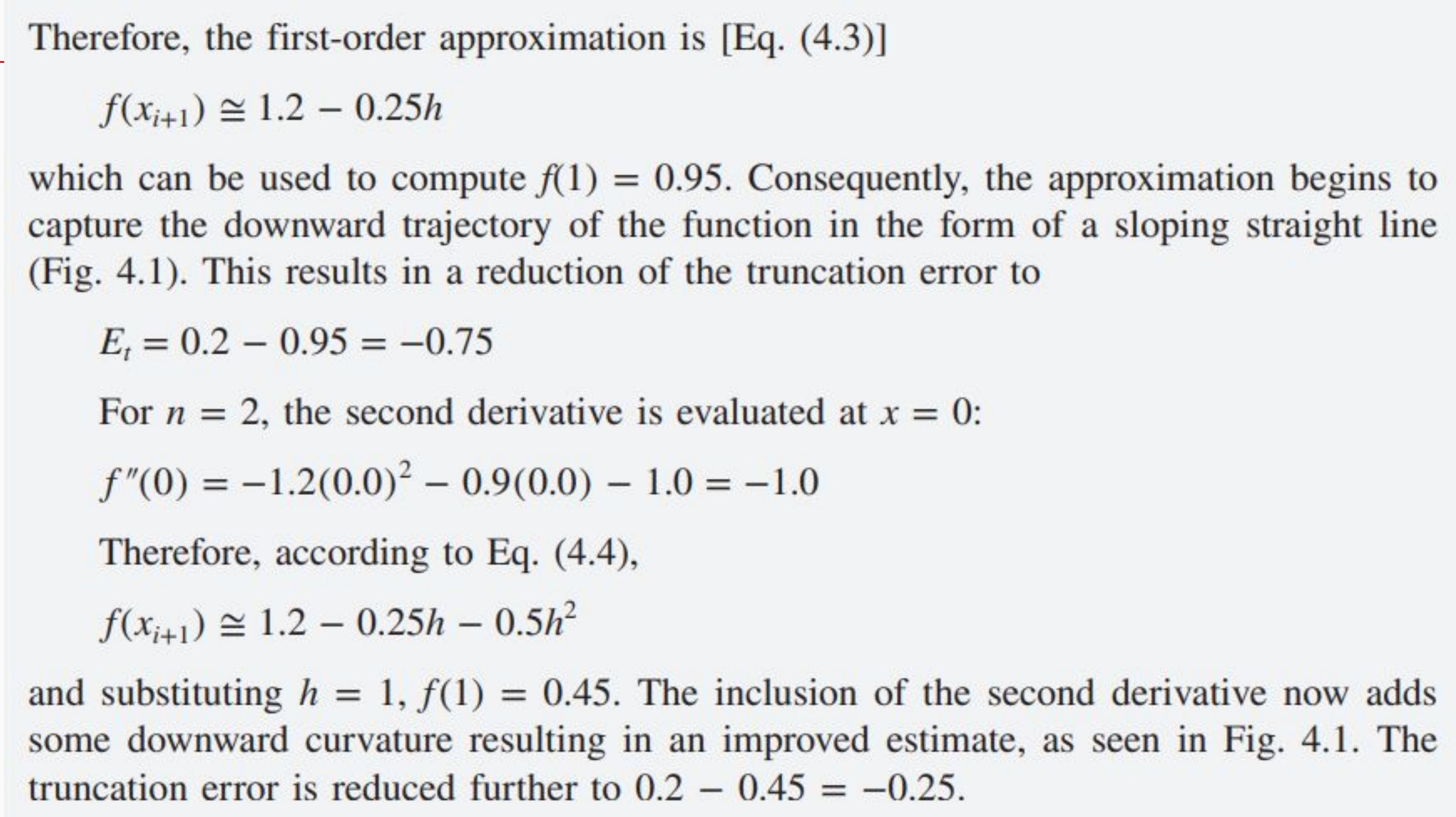
Thus, as in Fig. 4.1, the zero-order approximation is a constant. Using this formulation results in a truncation error [recall Eq. (3.2)] of

$$E_t = 0.2 - 1.2 = -1.0$$

at $x = 1$.

For $n = 1$, the first derivative must be determined and evaluated at $x = 0$:

$$f'(0) = -0.4(0.0)^3 - 0.45(0.0)^2 - 1.0(0.0) - 0.25 = -0.25$$



Therefore, the first-order approximation is [Eq. (4.3)]

$$f(x_{i+1}) \cong 1.2 - 0.25h$$

which can be used to compute $f(1) = 0.95$. Consequently, the approximation begins to capture the downward trajectory of the function in the form of a sloping straight line (Fig. 4.1). This results in a reduction of the truncation error to

$$E_t = 0.2 - 0.95 = -0.75$$

For $n = 2$, the second derivative is evaluated at $x = 0$:

$$f''(0) = -1.2(0.0)^2 - 0.9(0.0) - 1.0 = -1.0$$

Therefore, according to Eq. (4.4),

$$f(x_{i+1}) \cong 1.2 - 0.25h - 0.5h^2$$

and substituting $h = 1$, $f(1) = 0.45$. The inclusion of the second derivative now adds some downward curvature resulting in an improved estimate, as seen in Fig. 4.1. The truncation error is reduced further to $0.2 - 0.45 = -0.25$.

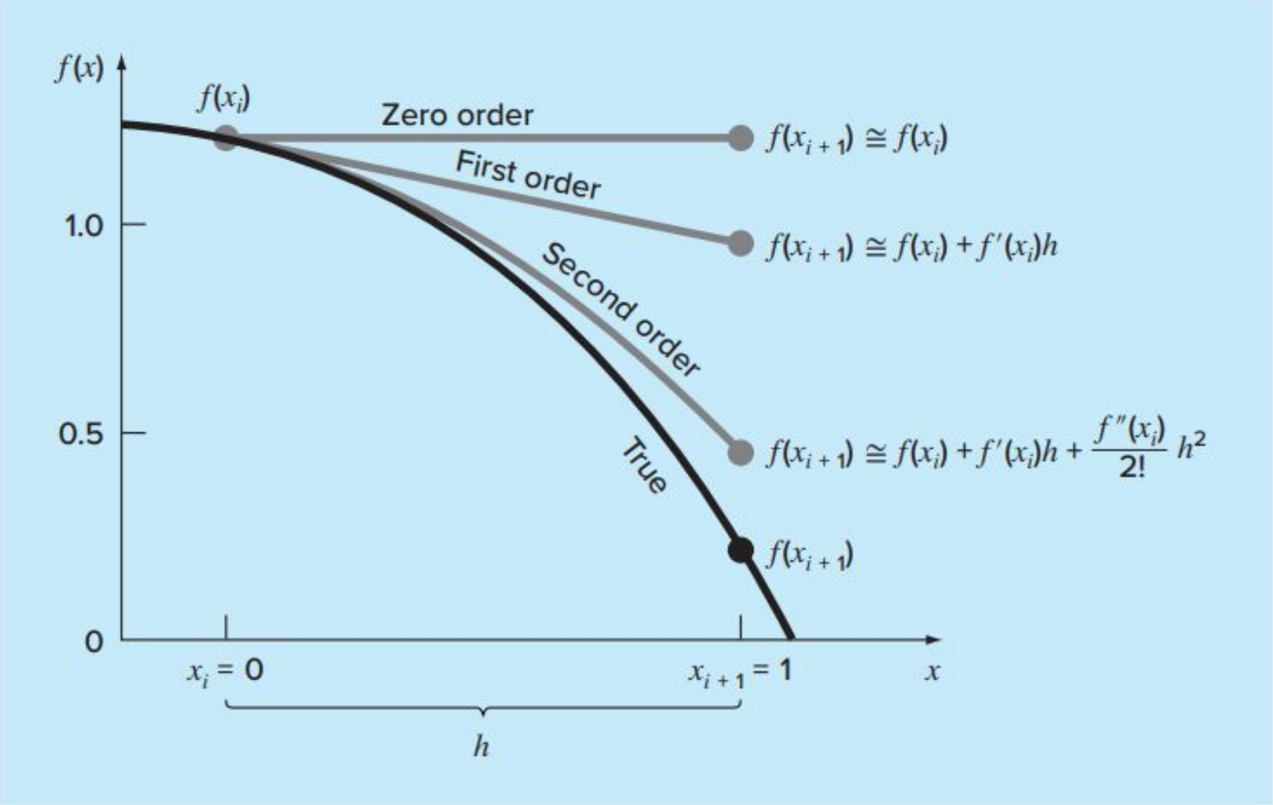
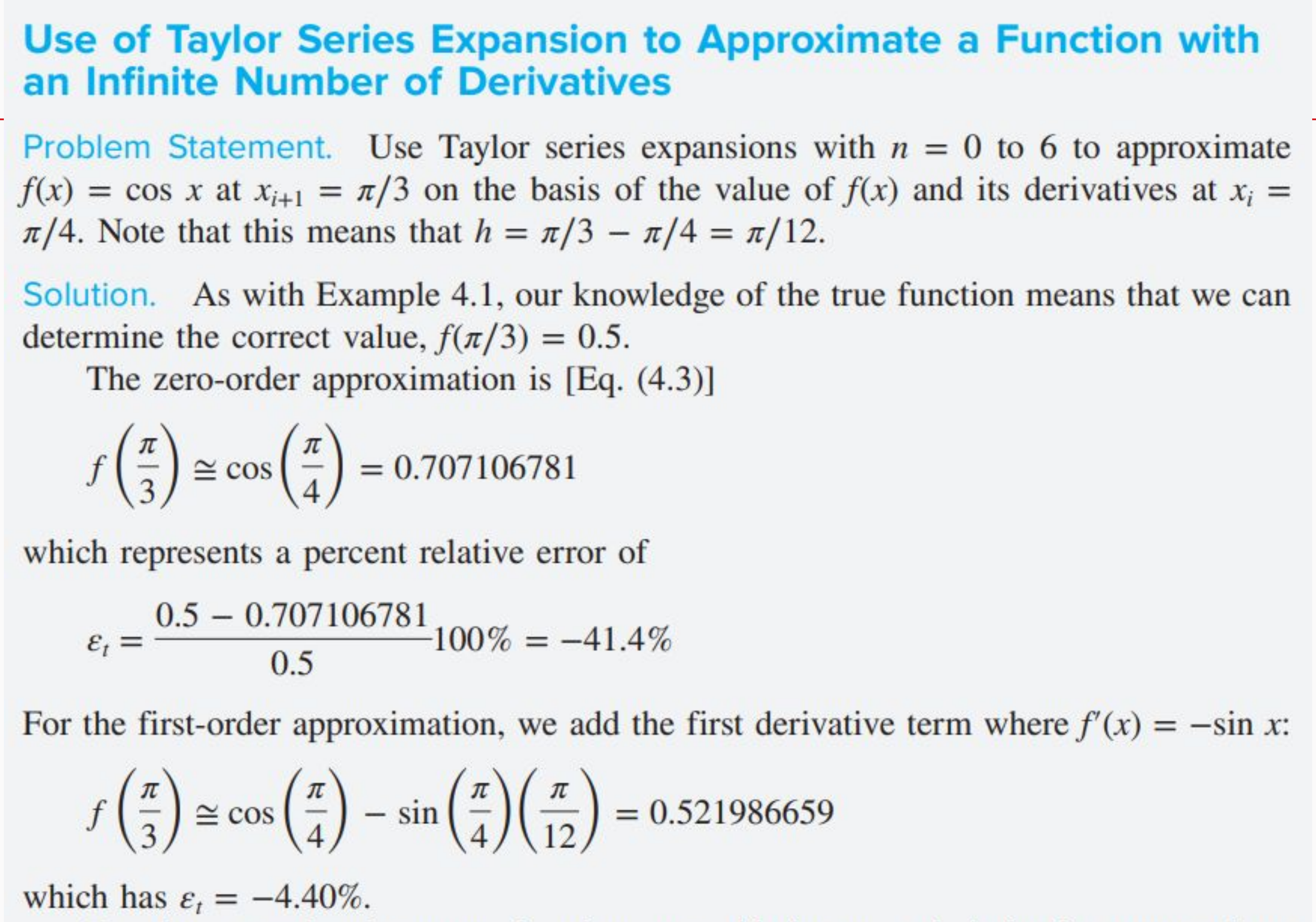


FIGURE 4.1

The approximation of $f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$ at $x = 1$ by zero-order, first-order, and second-order Taylor series expansions.



Use of Taylor Series Expansion to Approximate a Function with an Infinite Number of Derivatives

Problem Statement. Use Taylor series expansions with $n = 0$ to 6 to approximate $f(x) = \cos x$ at $x_{i+1} = \pi/3$ on the basis of the value of $f(x)$ and its derivatives at $x_i = \pi/4$. Note that this means that $h = \pi/3 - \pi/4 = \pi/12$.

Solution. As with Example 4.1, our knowledge of the true function means that we can determine the correct value, $f(\pi/3) = 0.5$.

The zero-order approximation is [Eq. (4.3)]

$$f\left(\frac{\pi}{3}\right) \cong \cos\left(\frac{\pi}{4}\right) = 0.707106781$$

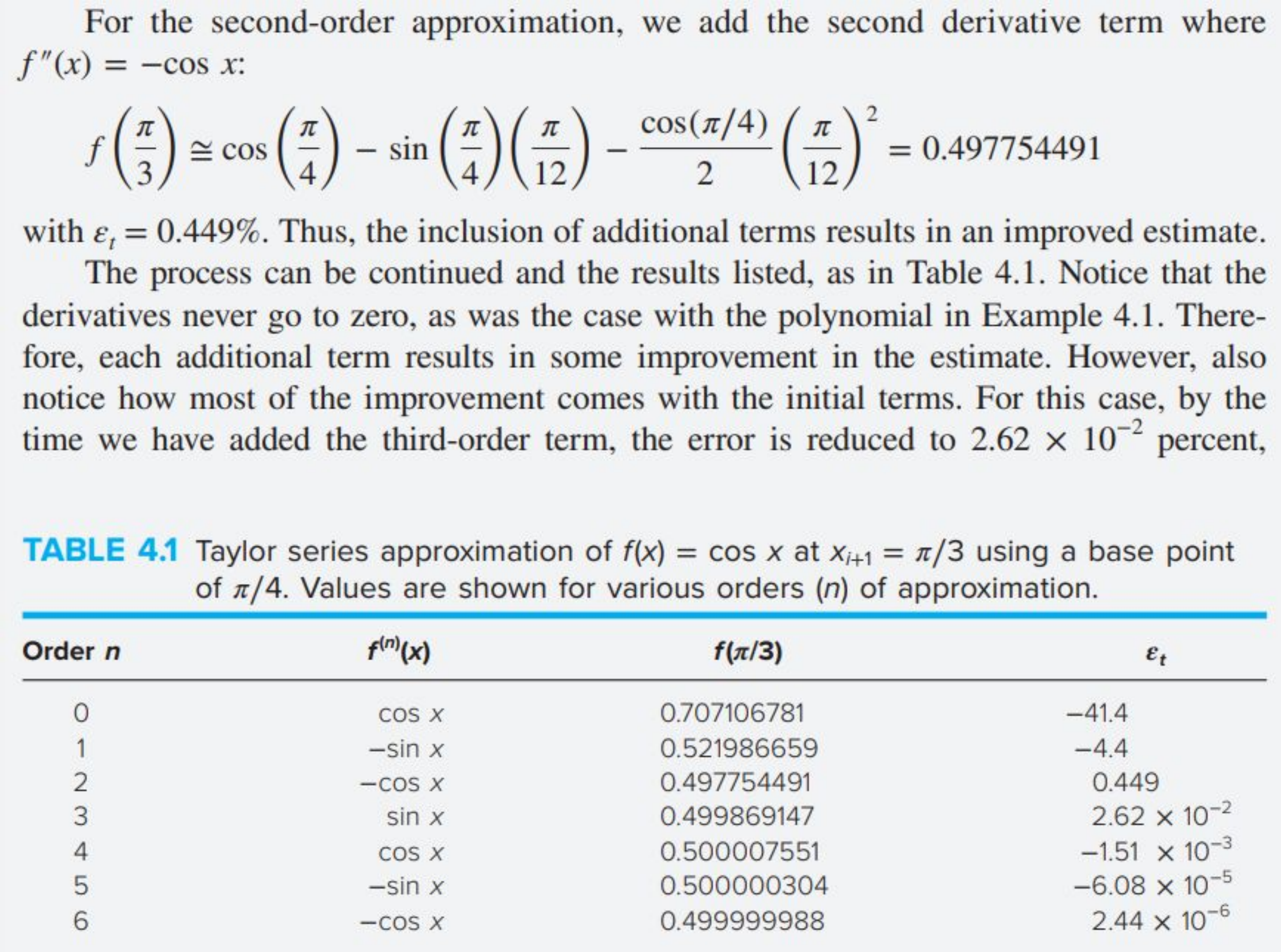
which represents a percent relative error of

$$\varepsilon_t = \frac{0.5 - 0.707106781}{0.5} 100\% = -41.4\%$$

For the first-order approximation, we add the first derivative term where $f'(x) = -\sin x$:

$$f\left(\frac{\pi}{3}\right) \cong \cos\left(\frac{\pi}{4}\right) - \sin\left(\frac{\pi}{4}\right)\left(\frac{\pi}{12}\right) = 0.521986659$$

which has $\varepsilon_t = -4.40\%$.



For the second-order approximation, we add the second derivative term where $f''(x) = -\cos x$:

$$f\left(\frac{\pi}{3}\right) \cong \cos\left(\frac{\pi}{4}\right) - \sin\left(\frac{\pi}{4}\right)\left(\frac{\pi}{12}\right) - \frac{\cos(\pi/4)}{2}\left(\frac{\pi}{12}\right)^2 = 0.497754491$$

with $\varepsilon_t = 0.449\%$. Thus, the inclusion of additional terms results in an improved estimate.

The process can be continued and the results listed, as in Table 4.1. Notice that the derivatives never go to zero, as was the case with the polynomial in Example 4.1. Therefore, each additional term results in some improvement in the estimate. However, also notice how most of the improvement comes with the initial terms. For this case, by the time we have added the third-order term, the error is reduced to 2.62×10^{-2} percent,

TABLE 4.1 Taylor series approximation of $f(x) = \cos x$ at $x_{i+1} = \pi/3$ using a base point of $\pi/4$. Values are shown for various orders (n) of approximation.

Order n	$f^{(n)}(x)$	$f(\pi/3)$	ε_t
0	$\cos x$	0.707106781	−41.4
1	$-\sin x$	0.521986659	−4.4
2	$-\cos x$	0.497754491	0.449
3	$\sin x$	0.499869147	2.62×10^{-2}
4	$\cos x$	0.500007551	-1.51×10^{-3}
5	$-\sin x$	0.500000304	-6.08×10^{-5}
6	$-\cos x$	0.499999988	2.44×10^{-6}

The Remainder for the Taylor Series Expansion

Suppose that we truncated the Taylor series expansion after the zero order term to yield

$$f(x_{i+1}) \cong f(x_i)$$

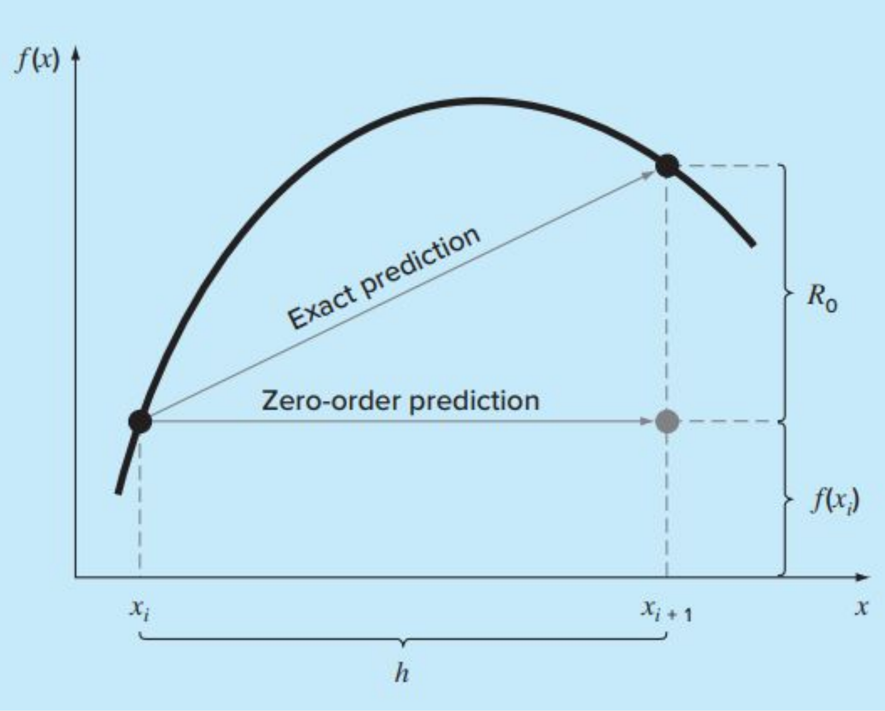
A visual depiction of this zero-order prediction is shown in Fig. . The remainder, or error, of this prediction, which is also shown in the illustration, consists of the infinite series of terms that were truncated:

$$R_0 = f'(x_i) * h + f''(x_i) / 2! * h^2 + f^{(3)} (x_i) / 3! \; * h^3 + . . .$$

It is obviously inconvenient to deal with the remainder in this infinite series format.

One simplification might be to truncate the remainder itself, as in

$$R_0 = f'(x_i) * h$$



The Remainder for the Taylor Series Expansion

- Although, as stated in the previous section, lower-order derivatives usually account for a greater share of the remainder than the higher-order terms, this result is still inexact because of the neglected second- and higher-order terms.
- This “inexactness” is implied by the approximate equality symbol (≅) employed in Eq.
- An alternative simplification that transforms the approximation into an equivalence is based on a graphical insight.
- As in Fig. below, the derivative mean-value theorem states that if a function f(x) and its first derivative are continuous over an interval from x_i to x_{i+1}, then there exists at least one point on the function that has a slope, designated by f'(ξ), that is parallel to the line joining $f(x_i)$ and $f(x_{i+1})$.
- The parameter ξ marks the x value where this slope occurs (Fig.).

A physical illustration of this theorem is that, if you travel between two points with an average velocity, there will be at least one moment during the course of the trip when you will be moving at that average velocity

The Remainder for the Taylor Series Expansion

By invoking this theorem, it is simple to realize that, as illustrated in Fig., the slope f'(ξ) is equal to the rise R_0 divided by the run h, or

$$f'(\xi) = R_0 / h$$

which can be rearranged to give

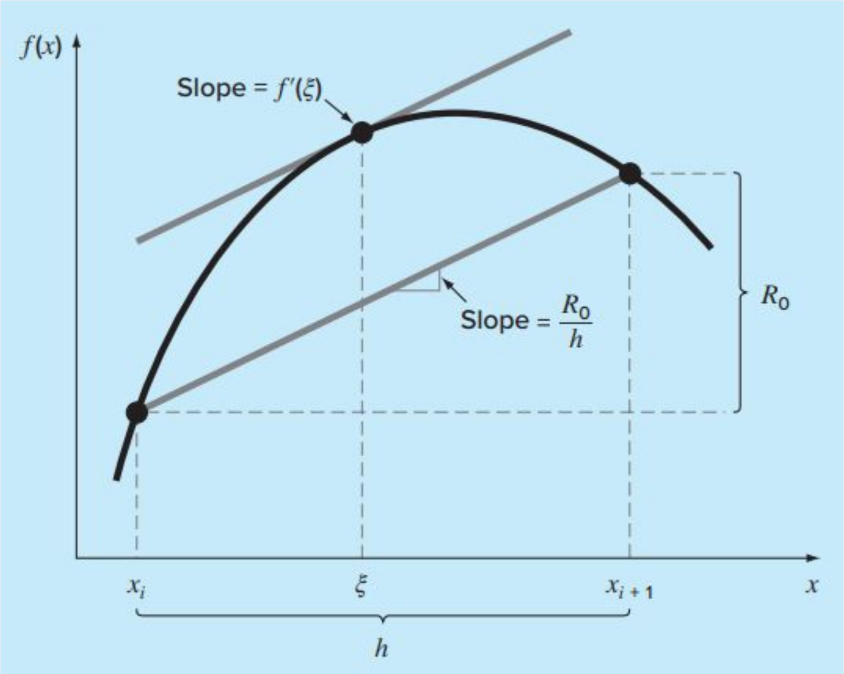
$$R_0 = f'(\xi) * h$$

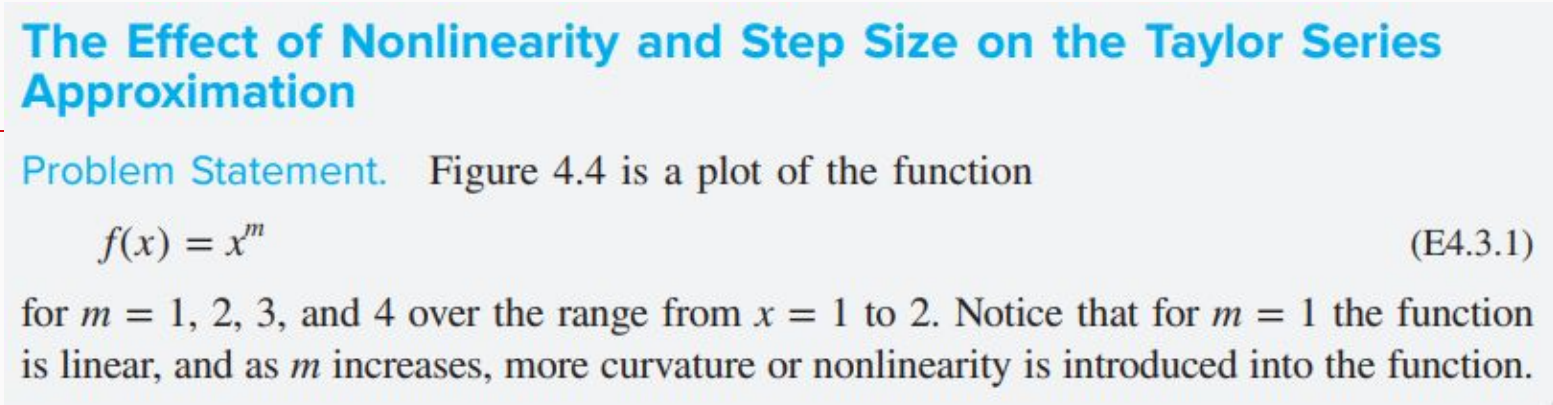
Thus, we have derived the zero-order version of Eq.

The higher-order versions are merely a logical extension of the reasoning used to derive Eq. (4.10). The first-order version is

$$R_1 = f''(\xi) / 2! * h^2$$

For this case, the value of ξ conforms to the x value corresponding to the second derivative that makes Eq.



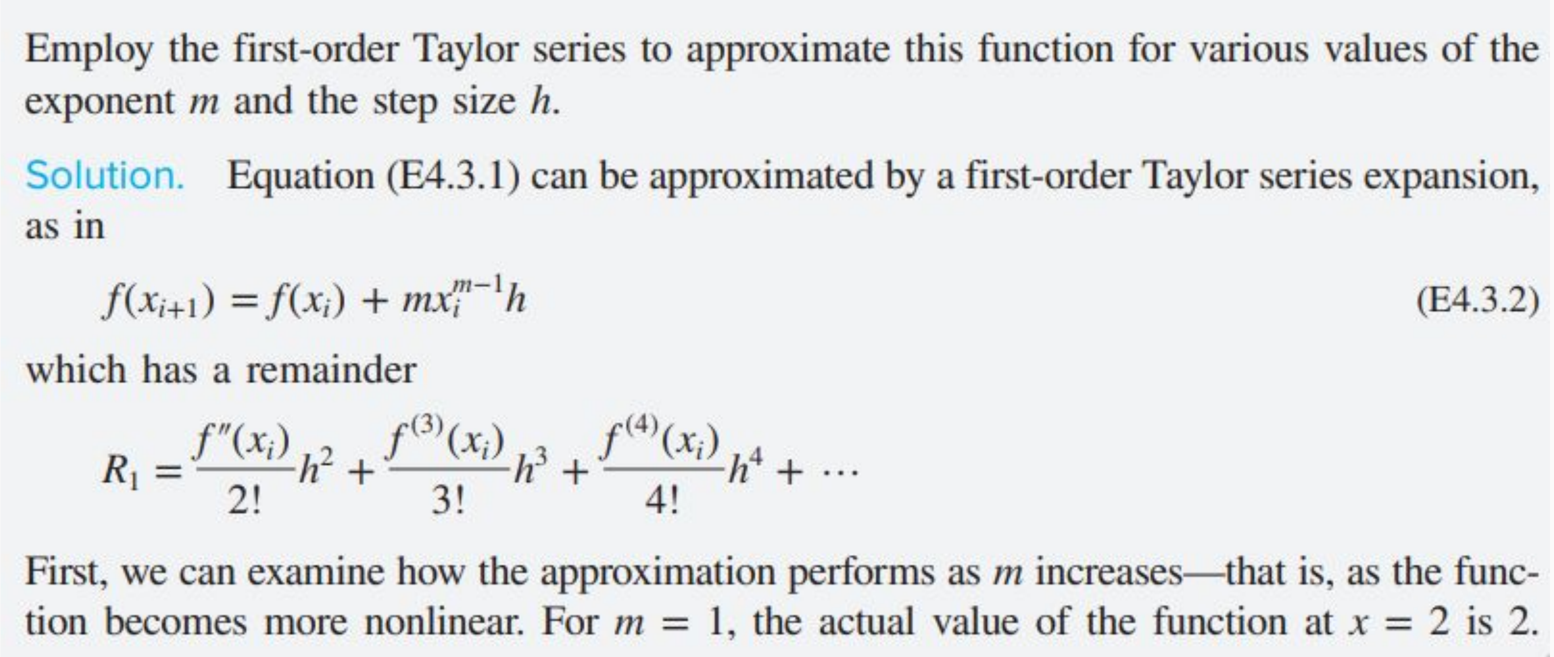


The Effect of Nonlinearity and Step Size on the Taylor Series Approximation

Problem Statement. Figure 4.4 is a plot of the function

$$f(x) = x^m \tag{E4.3.1}$$

for $m = 1, 2, 3$, and 4 over the range from $x = 1$ to 2. Notice that for $m = 1$ the function is linear, and as m increases, more curvature or nonlinearity is introduced into the function.



Employ the first-order Taylor series to approximate this function for various values of the exponent m and the step size h.

Solution. Equation (E4.3.1) can be approximated by a first-order Taylor series expansion, as in

$$f(x_{i+1}) = f(x_i) + mx_i^{m-1}h \tag{E4.3.2}$$

which has a remainder

$$R_1 = \frac{f''(x_i)}{2!}h^2 + \frac{f^{(3)}(x_i)}{3!}h^3 + \frac{f^{(4)}(x_i)}{4!}h^4 + \cdots$$

First, we can examine how the approximation performs as m increases—that is, as the function becomes more nonlinear. For $m = 1$, the actual value of the function at $x = 2$ is 2.

The Taylor series yields

$$f(2) = 1 + 1(1) = 2$$

and

$$R_1 = 0$$

The remainder is zero because the second and higher derivatives of a linear function are zero. Thus, as expected, the first-order Taylor series expansion is perfect when the underlying function is linear.

For $m = 2$, the actual value is $f(2) = 2^2 = 4$. The first-order Taylor series approximation is

$$f(2) = 1 + 2(1) = 3$$

and

$$R_1 = \tfrac{2}{2}(1)^2 + 0 + 0 + \cdots = 1$$

Thus, because the function is a parabola, the straight-line approximation results in a discrepancy. Note that the remainder is determined exactly.

For $m = 3$, the actual value is $f(2) = 2^3 = 8$. The Taylor series approximation is

$$f(2) = 1 + 3(1)^2(1) = 4$$

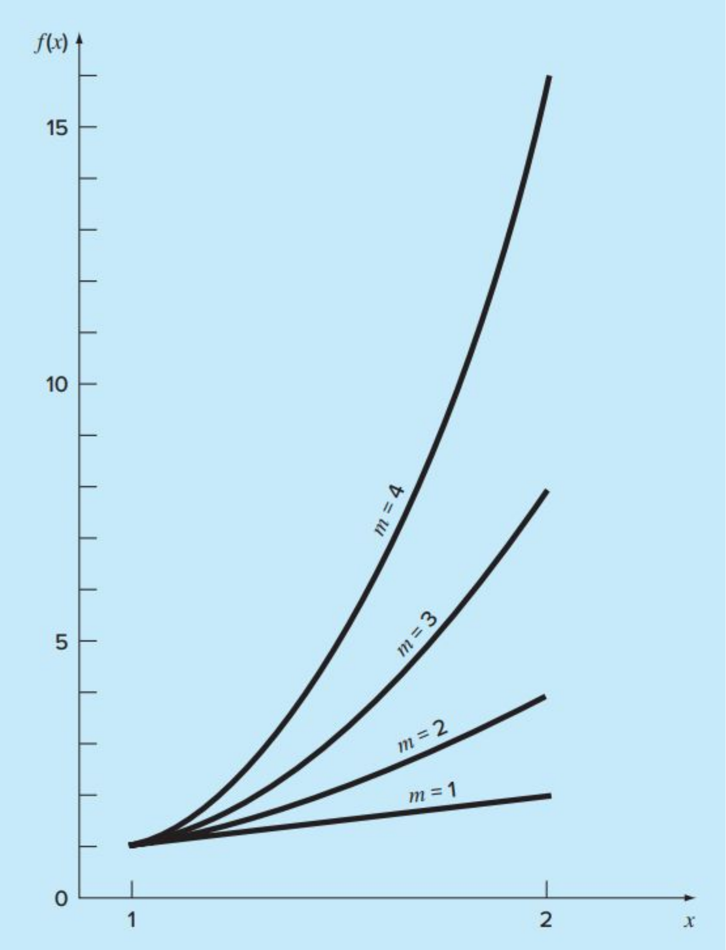
and

$$R_1 = \tfrac{6}{2}(1)^2 + \tfrac{6}{6}(1)^3 + 0 + 0 + \cdots = 4$$

Again, there is a discrepancy that can be determined exactly from the Taylor series.

For $m = 4$, the actual value is $f(2) = 2^4 = 16$. The Taylor series approximation is

$$f(2) = 1 + 4(1)^3(1) = 5$$



and

$$R_1 = \tfrac{12}{2}(1)^2 + \tfrac{24}{6}(1)^3 + \tfrac{24}{24}(1)^4 + 0 + 0 + \cdots = 11$$

On the basis of these four cases, we observe that R_1 increases as the function becomes more nonlinear. Furthermore, R_1 accounts exactly for the discrepancy. This is because Eq. (E4.3.1) is a simple monomial with a finite number of derivatives. This permits a complete determination of the Taylor series remainder.

Next, we will examine Eq. (E4.3.2) for the case $m = 4$ and observe how R_1 changes as the step size h is varied. For $m = 4$, Eq. (E4.3.2) is

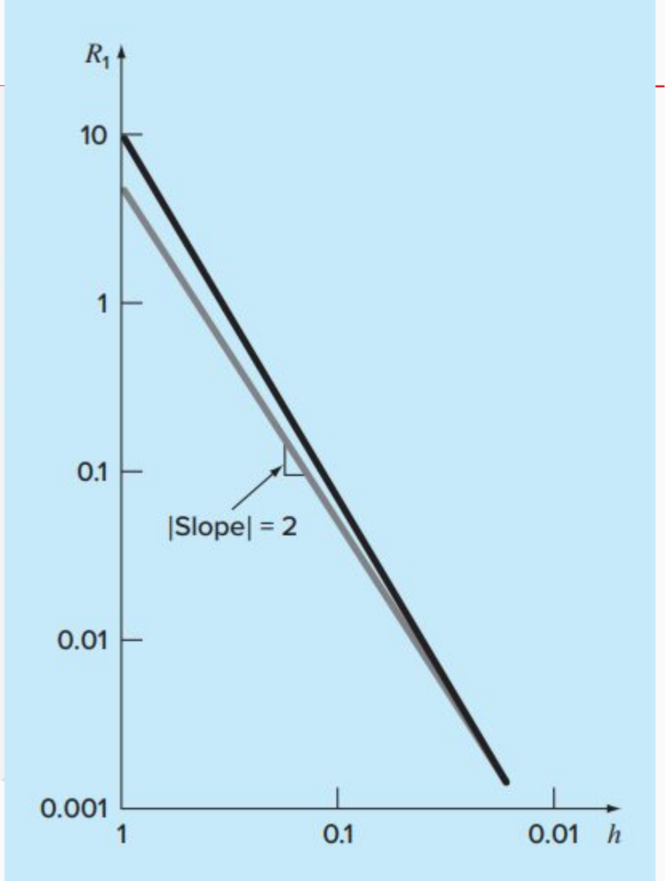
$$f(x + h) = f(x) + 4x_i^3 h$$

If $x = 1$, $f(1) = 1$ and this equation can be expressed as

$$f(1 + h) = 1 + 4h$$

with a remainder of

$$R_1 = 6h^2 + 4h^3 + h^4$$



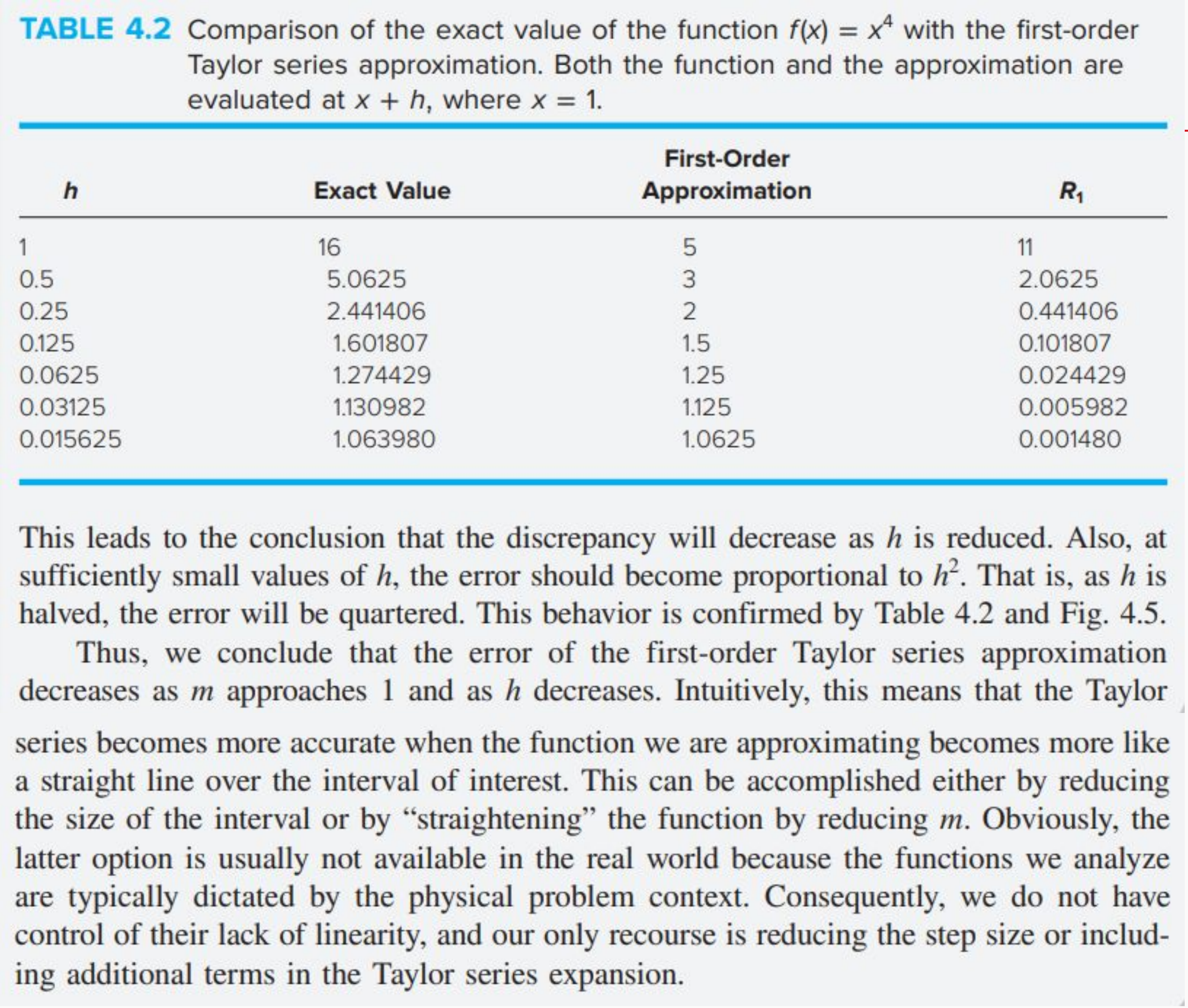
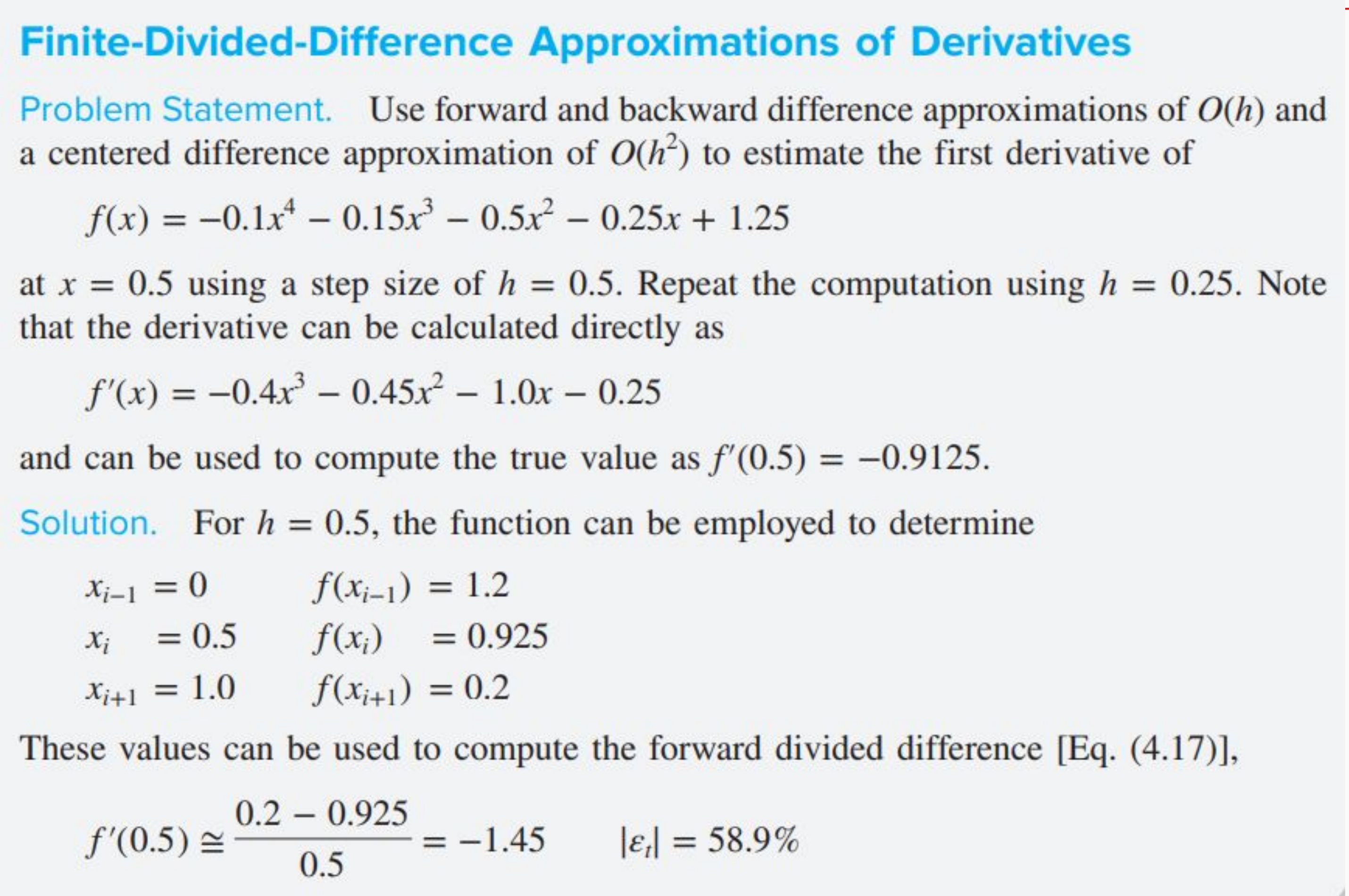


TABLE 4.2 Comparison of the exact value of the function $f(x) = x^4$ with the first-order Taylor series approximation. Both the function and the approximation are evaluated at $x + h$, where $x = 1$.

		First-Order	
h	**Exact Value**	**Approximation**	R_1
1	16	5	11
0.5	5.0625	3	2.0625
0.25	2.441406	2	0.441406
0.125	1.601807	1.5	0.101807
0.0625	1.274429	1.25	0.024429
0.03125	1.130982	1.125	0.005982
0.015625	1.063980	1.0625	0.001480

This leads to the conclusion that the discrepancy will decrease as h is reduced. Also, at sufficiently small values of h, the error should become proportional to h^2. That is, as h is halved, the error will be quartered. This behavior is confirmed by Table 4.2 and Fig. 4.5.

Thus, we conclude that the error of the first-order Taylor series approximation decreases as m approaches 1 and as h decreases. Intuitively, this means that the Taylor series becomes more accurate when the function we are approximating becomes more like a straight line over the interval of interest. This can be accomplished either by reducing the size of the interval or by "straightening" the function by reducing m. Obviously, the latter option is usually not available in the real world because the functions we analyze are typically dictated by the physical problem context. Consequently, we do not have control of their lack of linearity, and our only recourse is reducing the step size or including additional terms in the Taylor series expansion.



Finite-Divided-Difference Approximations of Derivatives

Problem Statement. Use forward and backward difference approximations of $O(h)$ and a centered difference approximation of $O(h^2)$ to estimate the first derivative of

$$f(x) = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.25$$

at $x = 0.5$ using a step size of $h = 0.5$. Repeat the computation using $h = 0.25$. Note that the derivative can be calculated directly as

$$f'(x) = -0.4x^3 - 0.45x^2 - 1.0x - 0.25$$

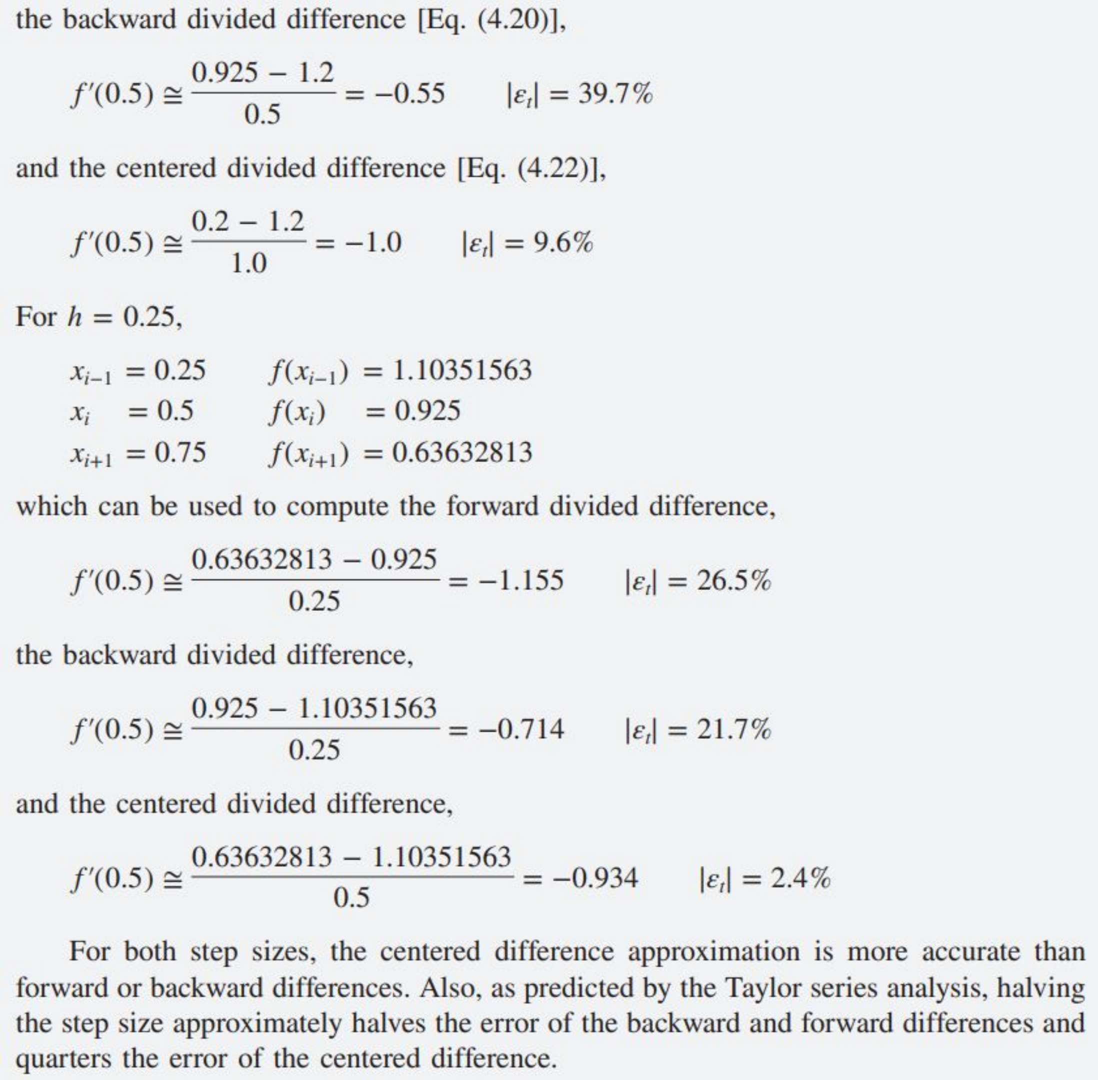
and can be used to compute the true value as $f'(0.5) = -0.9125$.

Solution. For $h = 0.5$, the function can be employed to determine

$$\begin{aligned} x_{i-1} &= 0 & f(x_{i-1}) &= 1.2 \\ x_i &= 0.5 & f(x_i) &= 0.925 \\ x_{i+1} &= 1.0 & f(x_{i+1}) &= 0.2 \end{aligned}$$

These values can be used to compute the forward divided difference [Eq. (4.17)],

$$f'(0.5) \cong \frac{0.2 - 0.925}{0.5} = -1.45 \qquad |\varepsilon_t| = 58.9\%$$



the backward divided difference [Eq. (4.20)],

$$f'(0.5) \cong \frac{0.925 - 1.2}{0.5} = -0.55 \qquad |\varepsilon_t| = 39.7\%$$

and the centered divided difference [Eq. (4.22)],

$$f'(0.5) \cong \frac{0.2 - 1.2}{1.0} = -1.0 \qquad |\varepsilon_t| = 9.6\%$$

For $h = 0.25$,

$$\begin{aligned} x_{i-1} &= 0.25 & f(x_{i-1}) &= 1.10351563 \\ x_i &= 0.5 & f(x_i) &= 0.925 \\ x_{i+1} &= 0.75 & f(x_{i+1}) &= 0.63632813 \end{aligned}$$

which can be used to compute the forward divided difference,

$$f'(0.5) \cong \frac{0.63632813 - 0.925}{0.25} = -1.155 \qquad |\varepsilon_t| = 26.5\%$$

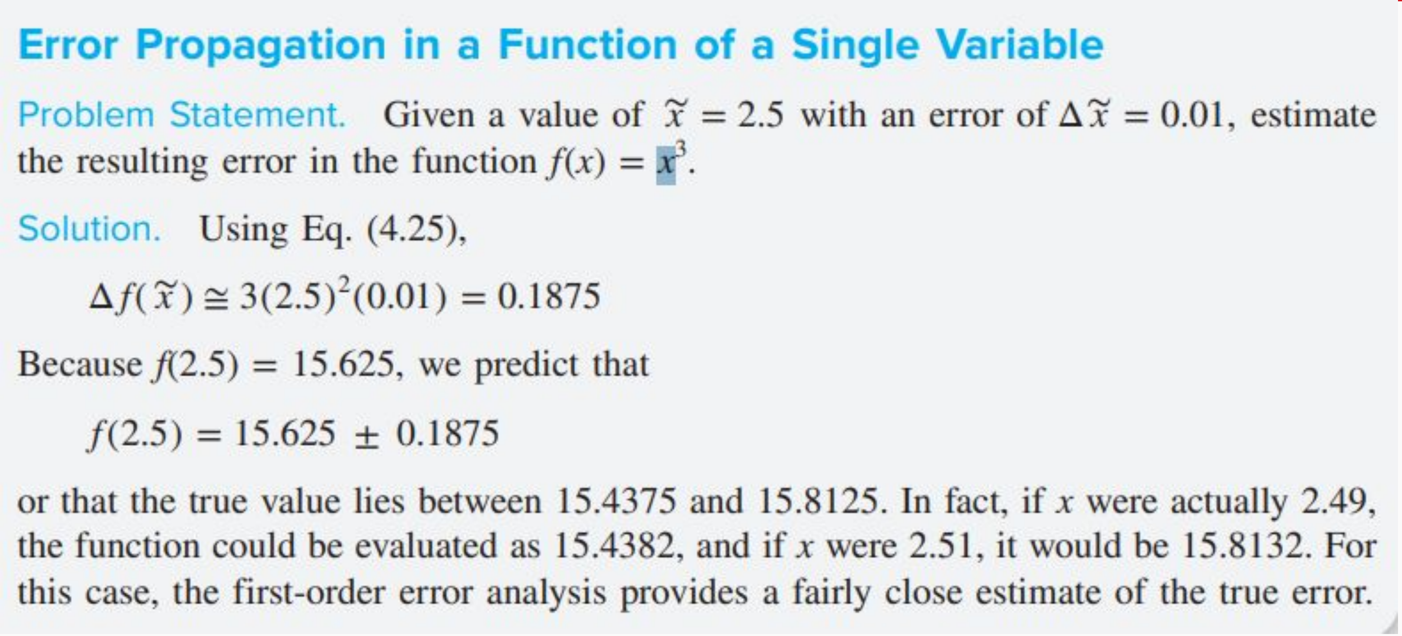
the backward divided difference,

$$f'(0.5) \cong \frac{0.925 - 1.10351563}{0.25} = -0.714 \qquad |\varepsilon_t| = 21.7\%$$

and the centered divided difference,

$$f'(0.5) \cong \frac{0.63632813 - 1.10351563}{0.5} = -0.934 \qquad |\varepsilon_t| = 2.4\%$$

For both step sizes, the centered difference approximation is more accurate than forward or backward differences. Also, as predicted by the Taylor series analysis, halving the step size approximately halves the error of the backward and forward differences and quarters the error of the centered difference.



Error Propagation in a Function of a Single Variable

Problem Statement. Given a value of $\tilde{x} = 2.5$ with an error of $\Delta\tilde{x} = 0.01$, estimate the resulting error in the function $f(x) = x^3$.

Solution. Using Eq. (4.25),

$$\Delta f(\tilde{x}) \cong 3(2.5)^2(0.01) = 0.1875$$

Because $f(2.5) = 15.625$, we predict that

$$f(2.5) = 15.625 \pm 0.1875$$

or that the true value lies between 15.4375 and 15.8125. In fact, if x were actually 2.49, the function could be evaluated as 15.4382, and if x were 2.51, it would be 15.8132. For this case, the first-order error analysis provides a fairly close estimate of the true error.

Blunders (Gross Errors)

Definition: Blunders are mistakes caused by human error or carelessness — not by the numerical method or the model itself.

Examples:

- Typing the wrong number into a program (e.g., entering 9.81 instead of 98.1).
- Misplacing a decimal point.
- Using the wrong units (e.g., cm instead of m).
- Coding mistakes or logic errors.

Impact: Blunders can make results completely wrong — they can dominate all other sources of error.

Prevention:

- Careful problem setup and verification.
- Double-checking input data.
- Following good programming practices (clear structure, testing, comments, validation).
- Comparing results to known benchmarks or analytical checks.

Formulation (Model) Errors

Definition: Formulation errors occur when the mathematical model itself does not accurately represent the real-world system. That is your equations or assumptions are wrong or incomplete.

Example:

- Modeling air resistance as proportional to velocity ($F = -cv$) when in reality it's proportional to v^2.
- Neglecting friction, heat transfer, or relativistic effects when they matter.

Even if your numerical solution is perfect, if your model is wrong — the results are still meaningless.

Key point: "If you are working with a poorly conceived model, no numerical method will provide adequate results."

Minimizing formulation error:

- Use physical insight and experimental validation.
- Compare with known results or simplified test cases.
- Refine the model step-by-step, checking sensitivity to assumptions.

Data Uncertainty

Definition: Data uncertainty arises because measurements of real-world quantities (like temperature, pressure, or velocity) are never exact. Even with a perfect model, uncertain input data causes uncertain results.

Two types of measurement error:

Type	Description	Example
Inaccuracy (Bias)	Systematic error — all measurements are shifted in one direction.	A scale that always reads 0.5 kg too high.
Imprecision (Random Err.)	Random scatter — measurements vary up and down unpredictably.	Repeated velocity readings Fluctuate due to turbulence.

Quantifying uncertainty - We use statistics to describe data uncertainty:

- Mean (average): measures central tendency (bias).
- Standard deviation / variance: measures spread (precision).

These describe how reliable your measured data are.

Why These Three Matter

Even though most numerical methods courses focus on round-off and truncation errors (i.e., the errors due to computation and approximation), in real engineering or scientific work:

Blunders, formulation errors, and data uncertainty often dominate the total error budget.

Thus, before blaming your numerical method, always ask:

- Did I enter the right data? (blunders)
- Is my model physically correct? (formulation error)
- Are my measurements reliable? (data uncertainty)

Thank you

https://github.com/realashok

dev.ashokbasnet@gmail.com