Engineering Computation

A Tutorial for Beginners in Computation for Applied Mechanics

Serhat Beyenir

14 August 2025

Contents

List of Figures

List of Tables

Preface

This book presents a collection of lecture notes on engineering computation, designed to provide learners with succinct yet essential insights into key topics covered in class. Each chapter is accompanied by a problem set to facilitate comprehension and reinforce understanding.

Chapter 2: The International System of Units (SI) is the globally accepted standard for measurement. Established to provide a consistent framework for scientific and technical measurements, SI units facilitate clear communication and data comparison across various fields and countries. The system is based on seven fundamental units: the meter for length, the kilogram for mass, the second for time, the ampere for electric current, the kelvin for temperature, the mole for substance, and the candela for luminous intensity.

Part I

Beginning

Chapter 1

Python Tutorial

This tutorial will help you get started with Python, covering basic concepts and providing examples to illustrate key points. We will use Python as a glorified calculator first, then move on to variables, functions, and control flow.

1.1 Requirements

To follow along with this tutorial, you will need to have Python installed on your computer. Python is available for Windows, macOS, and Linux. You can use Python 3.13.6, which is the version used for this tutorial. Additionally, ensure you have a text editor or an IDE (Integrated Development Environment) to write your Python code. I recommend using Positron for this purpose, as it provides a user-friendly interface and a built-in terminal for running Python scripts.

1.2 Basic Syntax

Python uses indentation to define blocks of code. Make sure to use consistent indentation (usually 4 spaces) throughout your code.

Here are some basic syntax rules:

- Comments start with `#` and extend to the end of the line.
- Strings can be enclosed in single quotes (`'`), double quotes (`"`), or triple quotes (`'''` or `"""`) for multi-line strings.
- Statements end with a newline character. You can use a backslash (`\`) to continue a statement on the next line.
- Python is case-sensitive, so `Variable` and `variable` are considered different identifiers.

1.3 The `print()` Function

In Python, `print()` is the built-in function used to display output.

```
name = "Rudolf Diesel"
year = 1858
print(f"{name} was born in {year}.")
```

```
# Output: Rudolf Diesel was born in 1858.
```

1.4 Formatting in `print()`

Format	Code	Example	Output
Round to 2 decimals	`f"{x:.2f}"`	`print(f"{3.14159:.2f}")`	`3.14`
Round to whole number	`f"{x:.0f}"`	`print(f"{3.9:.0f}")`	`4`
Thousands separator	`f"{x:,.2f}"`	`print(f"{1234567.89:,.2f}")`	`1,234,567.89`
Percentage	`f"{x:.1%}"`	`print(f"{0.756:.1%}")`	`75.6%`
Currency style	`f"${x:,.2f}"`	`print(f"${1234.5:,.2f}")`	`$1,234.50`

1.5 Variables and Data Types

Variables are used to store data. You can assign a value to a variable using the `=` operator.

```
x = 10
y = 3.14
name = "Rudolph"
```

Python has several built-in data types, including:

- Integers (`int`): Whole numbers, e.g., `10`, `-5`
- Floating-point numbers (`float`): Decimal numbers, e.g., `3.14`, `-0.001`
- Strings (`str`): Text, e.g., `"Hello"`, `'World'`
- Booleans (`bool`): `True` or `False`

You can check the type of a variable using the `type()` function.

```
print(type(x))  # Output: <class 'int'>
print(type(name))  # Output: <class 'str'>
```

1.5.1 Arithmetic Operations

```
a = 10
b = 3
print(a + b)  # Addition: 13
print(a - b)  # Subtraction: 7
print(a * b)  # Multiplication: 30
print(a / b)  # Division: 3.3333...
print(a // b) # Integer Division: 3
print(a ** b) # Exponentiation: 1000
```

1.5.2 String Operations

```
first_name = "Rudolph"
last_name = "Diesel"
full_name = first_name + " " + last_name  # Concatenation
print(full_name)  # Output: Rudolph Diesel
print(full_name * 2)  # Repetition: Rudolph DieselRudolph Diesel
print(full_name.upper())  # Uppercase: RUDOLPH DIESEL
```

1.6 Python as a Calculator in Interactive Mode

You can use Python as a calculator in interactive mode, which allows you to enter Python commands directly and see the results immediately. This is useful for quick calculations and testing small snippets of code. To start Python in interactive mode, open macOS terminal and type:

```
    python3
```

(or `python` on Windows)

You should see:

```
    >>>
```

Now you can type math expressions and press **Enter**.

1.6.1 Basic Arithmetic

```
    2 + 3     # addition → 5
    7 - 4     # subtraction → 3
    6 * 9     # multiplication → 54
    8 / 2     # division → 4.0 (float result, decimal)
    8 // 2    # integer division → 4
    2 ** 3    # exponent → 8
```

1.6.2 Parentheses for Grouping

```
    (2 + 3) * 4    # → 20
    2 + (3 * 4)    # → 14
```

1.6.3 Variables

```
    x = 10
    y = 3
    x / y        # → 3.3333333333333335
```

1.6.4 Exiting Python Interactive Mode

To exit the interactive mode, you can type:

```
    exit()  # Type this to exit the interactive mode
```

Or:

- **Ctrl+D** (macOS/Linux)
- **Ctrl+Z** then Enter (Windows)

1.7 Control Flow

Control flow statements allow you to control the execution of your code based on certain conditions.

1.7.1 Conditional Statements

Conditional statements let you execute different blocks of code based on certain conditions. The most common conditional statements are `if`, `elif`, and `else`.

`if` statements check a condition and execute a block of code if the condition is true. `elif` (short for “else if”) allows you to check multiple conditions, and `else` provides a fallback option if none of the previous conditions are true.

```
# Conditional Statements: This script checks
# the age of a person and categorizes them as
# Minor, Adult, or Senior.
age = 19
if age < 18:
    print("Minor")
elif age >= 18 and age < 65:
    print("Adult")
else:
    print("Senior")
```

```
# Output: Adult
```

1.7.2 For Loop

A `for` loop iterates over a sequence (like a list or string) and executes a block of code for each item in the sequence.

```
components = ["piston", "liner", "connecting rod"]
for component in components:
    print(component)
```

```
# Output:
# piston
# liner
# connecting rod
```

1.7.3 While Loop

A `while` loop continues to execute a block of code as long as a specified condition is true.

```
count = 0
while count <= 5:
    print(count)
    count += 1
```

```
# Output:
# 0
# 1
# 2
# 3
# 4
# 5
```

1.8 Functions

Functions are reusable blocks of code that perform a specific task. You can define a function using the `def` keyword.

```
def greet(name):
    return "Hello, " + name + "!"
print(greet("Alice"))  # Output: Hello, Alice!
```

You can also define functions with default parameters and final results in decimal points.

```
def add(a, b=0):
    return a + b
print(add(5))      # Output: 5
print(add(5, 3))   # Output: 8
def multiply(*args):
    result = 1
    for num in args:
        result *= num
    return result
print(multiply(2, 3, 4))  # Output: 24
```

`lambda` a special syntax in Python for creating a function object without using the `def` keyword. Two ways to create the same function:

```
celsius_to_fahrenheit = lambda c: (c * 9 / 5) + 32
```

```
def celsius_to_fahrenheit(c):
    return (c * 9 / 5) + 32
```

Both behave identically:

```
celsius_to_fahrenheit(25)  # 77.0
```

1.9 The `math` Module

Python has a built-in `math` module that provides various mathematical functions and constants. You can import the `math` module using `import math`, and then use its functions like `math.sqrt()`, `math.sin()`, and `math.cos()`.

```
import math
print(math.sqrt(16))         # Output: 4.0
print(math.pi)               # Output: 3.141592653589793
```

```
import math
angle = math.pi / 4  # 45 degrees in radians
print(math.sin(angle))  # Output: 0.7071067811865475
print(math.cos(angle))  # Output: 0.7071067811865476
print(math.tan(angle))  # Output: 1.0
```

The `math.log()` function can be used to calculate the natural logarithm, and you can specify a base for logarithms as well.

```
import math
print(math.log(10))  # Natural logarithm of 10
print(math.log(100, 10))  # Logarithm of 100 with base 10
```

1.10 Writing Python Scripts

You can also write Python code in a text file and run it as a script. Create a file named `script.py` and add the following code:

```
# script.py
import math
print("Square root of 16 is:", math.sqrt(16))
print("Value of pi is:", math.pi)
print("Sine of 90 degrees is:", math.sin(math.pi / 2))
print("Natural logarithm of 10 is:", math.log(10))
print("Logarithm of 100 with base 10 is:", math.log(100, 10))
```

To run the script, open your terminal and navigate to the directory where `script.py` is located, then type:

```
python3 script.py  # or python script.py on Windows
```

You should see the output of the script in your terminal. This is a great way to write and save your Python code for later use.

```
# Output:
Square root of 16 is: 4.0
Value of pi is: 3.141592653589793
Sine of 90 degrees is: 1.0
Natural logarithm of 10 is: 2.302585092994046
Logarithm of 100 with base 10 is: 2.0
```

1.11 Summary

This tutorial covered the basics of Python programming, including syntax, variables, data types, operations, control flow, and functions. Python is a powerful and versatile language, and has a rich ecosystem of libraries and modules that extend its functionality. Some popular libraries include:

- **NumPy**: For numerical computations and array manipulations.
- **Matplotlib**: For data visualization and plotting.
- **Pint**: For handling units and performing unit conversions automatically.

You can explore these libraries to enhance your Python programming skills further. For example installing them can be done using `pip`:

```
pip install numpy matplotlib pint
```

`pip` is the package manager for Python, and it allows you to install and manage additional libraries that are not included in the standard Python distribution.

Chapter 2

International System of Units

2.1 SI Units

The International System of Units (SI) is the globally accepted standard for measurement. Established to provide a consistent framework for scientific and technical measurements, SI units facilitate clear communication and data comparison across various fields and countries. The system is based on seven fundamental units: the meter for length, the kilogram for mass, the second for time, the ampere for electric current, the kelvin for temperature, the mole for substance, and the candela for luminous intensity.

Table 2.1: Base SI units.

Physical Quantity	SI Base Unit	Symbol
Length	Meter	m
Mass	Kilogram	kg
Time	Second	s
Electric Current	Ampere	A
Temperature	Kelvin	K
Amount of Substance	Mole	mol
Luminous Intensity	Candela	cd

Table 2.2: Derived SI units.

Physical Quantity	Derived SI Unit	Symbol
Area	Square meter	m^2
Volume	Cubic meter	m^3
Speed	Meter per second	m/s
Acceleration	Meter per second squared	m/s^2
Force	Newton	N
Pressure	Pascal	Pa
Energy	Joule	J
Power	Watt	W
Electric Charge	Coulomb	C
Electric Potential	Volt	V
Resistance	Ohm	Ω
Capacitance	Farad	F
Frequency	Hertz	Hz
Luminous Flux	Lumen	lm
Illuminance	Lux	lx
Specific Energy	Joule per kilogram	J/kg
Specific Heat Capacity	Joule per kilogram Kelvin	$J/(kg \cdot K)$

Table 2.3: Common multiples and submultiples for SI units.

Factor	Prefix	Symbol
10^9	giga	G
10^6	mega	M
10^3	kilo	k
10^2	hecto	h
10^1	deca	da
10^{-1}	deci	d
10^{-2}	centi	c
10^{-3}	milli	m
10^{-6}	micro	µ

2.2 SI System Rules and Common Mistakes

Using the SI system correctly is crucial for clear communication in science and engineering. Below are common mistakes in using the SI system, examples of incorrect usage, and how to correct them.

Table 2.4: SI system rules and common mistakes

Concept	**Mistake**	**Correct Usage**	**Notes**
Use of SI Unit Symbols	`m./s`	`m/s`	Use the correct format without additional punctuation.
Spacing Between Value & Unit	`10kg`	`10 kg`	Always leave a space between the number and the unit symbol.
Incorrect Unit Symbols	`sec`, `hrs`, `°K`	`s`, `h`, `K`	Use the proper SI symbols; symbols are case-sensitive.
Abbreviations for Units	`5 kilograms (kgs)`	`5 kilograms (kg)`	Avoid informal abbreviations like “kgs”; adhere to standard symbols.
Multiple Units in Expressions	`5 m/s/s`, `5 kg/meter`^{2}	`5 m/s`^{2}, `5 kg/m`^{2}	Use compact, standardized formats for derived units.

Concept	**Mistake**	**Correct Usage**	**Notes**
Incorrect Use of Prefixes	`0.0001 km`	`100 mm`	Choose prefixes to keep numbers in the range (0.1 x < 1000).
Misplaced Unit Symbols	`5/s`, `kg10`	`5 s` ^{1}, `10 kg`	Symbols must follow numerical values, not precede them.
Degrees Celsius vs. Kelvin	`300°K`	`300 K`	Kelvin is written without "degree"
Singular vs. Plural Units	`5 kgs`, `1 meters`	`5 kg`, `1 meter`	Symbols do not pluralize; full unit names follow grammar rules.
Capitalization of Symbols	`Kg`, `S`, `Km`, `MA`	`kg`, `s`, `km`, `mA`	Symbols are case-sensitive; use uppercase only where specified (e.g., `N`, `Pa`).
Capitalization of Unit Names	`Newton`, `Pascal`, `Watt`	`newton`, `pascal`, `watt`	Unit names are lowercase, even if derived from a person's name, unless starting a sentence.
Prefix Capitalization	`MilliMeter`, `MegaWatt`	`millimeter`, `megawatt`	Prefixes are lowercase for (10^{-1}) to (10^{-9}), uppercase for (10^6) and larger (except `k` for kilo).
Formatting in Reports	`5`, `Temperature: 300`	`5 kg`, `Temperature: 300 K`	Always specify units explicitly.

2.3 Unity Fraction

The **unity fraction** method, or **unit conversion using unity fractions**, is a systematic way to convert one unit of measurement into another. This method relies on multiplying by fractions that are equal to one, where the numerator and the denominator represent the same quantity in different units. Since any number multiplied by one remains the same, unity fractions allow for seamless conversion without changing the value.

The principle of unity fractions is based on:

1. **Setting up equal values**: Write a fraction where the numerator and denominator are equivalent values in different units, so the fraction equals one. For example, $\frac{1km}{1000m}$ is a unity fraction because 1 km equals 1000 m.
2. **Multiplying by unity fractions**: Multiply the initial quantity by the unity fraction(s) so that the undesired units cancel out, leaving only the desired units.

2.4 Classwork

Example 2.1. Suppose we want to convert 5 kilometers to meters.

1. Start with 5 kilometers:
$$5\,\mathrm{km}$$
2. Multiply by a unity fraction that cancels kilometers and introduces meters. We use $(\frac{1000\,\mathrm{m}}{1\,\mathrm{km}}), since\ 1\,\mathrm{km} = 1000\,\mathrm{m}$:
$$5\,\mathrm{km} \times \frac{1000\,\mathrm{m}}{1\,\mathrm{km}} = 5000\,\mathrm{m}$$
3. The kilometers km cancel out, leaving us with meters m:
$$5\,\mathrm{km} = 5000\,\mathrm{m}$$

This step-by-step approach illustrates how the unity fraction cancels the undesired units and achieves the correct result in meters.

Unity fractions can be extended by using multiple conversion steps. For example, converting hours to seconds would require two unity fractions: one to convert hours to minutes and another to convert minutes to seconds. This approach ensures accuracy and is widely used in science, engineering, and other fields that require precise unit conversions.

Example 2.2. Convert $15\,\mathrm{m/s}$ to km/h.

1. Start with $15\,\mathrm{m/s}$.
2. To convert meters to kilometers, multiply by $\frac{1\,\mathrm{km}}{1000\,\mathrm{m}}$.
3. To convert seconds to hours, multiply by $\frac{3600\,\mathrm{s}}{1\,\mathrm{h}}$.

$$15\,\mathrm{m/s} \times \frac{1\,\mathrm{km}}{1000\,\mathrm{m}} \times \frac{3600\,\mathrm{s}}{1\,\mathrm{h}} = 54\,\mathrm{km/h}$$

The meters and seconds cancel out, leaving kilometers per hour: 54 km/h.

2.5 Problem Set

Instructions:

1. Use unity fraction to convert between derived SI units.
2. Show each step of your work to ensure accuracy.
3. Simplify your answers and include correct units.

---

1. **Speed**
   Convert 72 km/h to m/s.
2. **Force**
   Convert 980 N (newtons) to $\text{kg} \cdot \text{m/s}^2$.
3. **Energy**
   Convert 2500 J (joules) to kJ.
4. **Power**
   Convert 1500 W (watts) to kW.
5. **Pressure**
   Convert 101325 Pa (pascals) to kPa.
6. **Volume Flow Rate**
   Convert $3\,\text{m}^3/\text{min}$ to L/s.
7. **Density**
   Convert $1000\,\text{kg/m}^3$ to g/cm^3.
8. **Acceleration**
   Convert $9.8\,\text{m/s}^2$ to cm/s^2.
9. **Torque**
   Convert $50\,\text{N} \cdot \text{m}$ to $\text{kN} \cdot \text{cm}$.
10. **Frequency**
    Convert 500 Hz (hertz) to kHz.
11. **Work to Energy Conversion**
    A force of 20 N moves an object 500 cm. Convert the work done to joules.
12. **Kinetic Energy Conversion**
    Calculate the kinetic energy in kilojoules of a 1500 kg car moving at 72 km/h.

13. **Power to Energy Conversion**
A machine operates at $2\,\text{kW}$ for 3 hours. Convert the energy used to megajoules.
14. **Pressure to Force Conversion**
Convert a pressure of $200\,\text{kPa}$ applied to an area of $0.5\,\text{m}^2$ to force in newtons.
15. **Density to Mass Conversion**
Convert $0.8\,\text{g/cm}^3$ for an object with a volume of $250\,\text{cm}^3$ to mass in grams.

2.5.1 Answer Key

1. $72\,\text{km/h} = 20\,\text{m/s}$
2. $980\,\text{N} = 980\,\text{kg}\cdot\text{m/s}^2$
3. $2500\,\text{J} = 2.5\,\text{kJ}$
4. $1500\,\text{W} = 1.5\,\text{kW}$
5. $101325\,\text{Pa} = 101.325\,\text{kPa}$
6. $3\,\text{m}^3/\text{min} = 50\,\text{L/s}$
7. $1000\,\text{kg/m}^3 = 1\,\text{g/cm}^3$
8. $9.8\,\text{m/s}^2 = 980\,\text{cm/s}^2$
9. $50\,\text{N}\cdot\text{m} = 5\,\text{kN}\cdot\text{cm}$
10. $500\,\text{Hz} = 0.5\,\text{kHz}$
11. $20\,\text{N} \times 5\,\text{m} = 100\,\text{J}$
12. Kinetic energy $= 1500\,\text{kg} \times (20\,\text{m/s})^2/2 = 300\,\text{kJ}$
13. $2\,\text{kW} \times 3\,\text{hours} = 21.6\,\text{MJ}$
14. $200\,\text{kPa} \times 0.5\,\text{m}^2 = 100{,}000\,\text{N}$
15. $0.8\,\text{g/cm}^3 \times 250\,\text{cm}^3 = 200\,\text{g}$

2.6 Further Reading

Introduction in Russell et al. (2021) and SI units in Bolton (2021) for additional information.

Part II

Making progress

Part III

Going forward

Chapter 3

Review

We have used several books by Ahrens (2022), Russell et al. (2021), Bolton (2021), Polya & Conway (2014), Bird & Ross (2020) and Bird (2021). These sources have helped you understand complex concepts.

Chapter 2:

- Purpose of SI Units: Provide a consistent framework for scientific and technical measurements.
- Advantages of SI Units: Facilitate clear communication and data comparison across various fields and countries.
- Fundamental Units of SI: Meter, kilogram, second, ampere, kelvin, mole, and candela.
- Method Name: Unity fraction method.
- Purpose: Converting one unit of measurement into another.
- Methodology: Multiplying by fractions equal to one, where the numerator and denominator represent the same quantity in different units.

References

Ahrens, S. (2022). *How to take smart notes: One simple technique to boost writing, learning and thinking* (2nd ed. edition). Sönke Ahrens.

Bird, J. O. (2021). *Bird's engineering mathematics* (Ninth edition). Routledge.

Bird, J. O., & Ross, C. T. F. (2020). *Mechanical engineering principles* (Fourth edition). Routledge.

Bolton, W. (2021). *Engineering science* (Seventh edition). Routledge.

Polya, G., & Conway, J. H. (2014). *How to solve it: A new aspect of mathematical method* (With a Foreword by John H. Con ed. edition). Princeton University Press.

Russell, P. A., Jackson, L., & Embleton, W. (2021). *Applied mechanics for marine engineers* (7th edition). Reeds.

Index