PRACTICAL COMPUTING USING MATLAB

The Basics

JESSE GABRIEL

Contents

Chapter 1

Introduction

1.1 Overview

Computing is such an interesting field that many of us are undeniably curious and interested in. Computing comes in many forms, types, systems, tools and platforms, but often, we generally describe it as digital technologies. The ones that we are most familiar with and use almost daily include the internet, the apps or applications and software in our phones and computers, the social media applications including Facebook and WhatsApp, even our mobile phones, our desktop or laptop computers and many others. These are all computing devices that are generally grouped as software applications and hardware devices. A computer hardware is any element of a computer that's physical, that we can see or touch. This includes things like monitors, keyboards, and also the insides of devices, like microchips and hard drives, as well as our phones and physical calculators. A software is anything that tells the hardware what to do and how to do it, including computer programs and apps on our phones including Facebook and WhatsApp.

In the context of this book, we are interested in the computer software. Think of computers as a human body. In the context of this book, we are simply interested in the computer's brain and nervious system, where the brain is like the central processing unit (CPU) and the software applications are like the nervious system and together they instruct the different parts of the body (hardware) what to do and how to do things. This brings us to the big question: how do you write such a computer software? The purpose of this book is simply to answer that question: not in words, but in practical software writing. As all things start small, this book provides basic software writing techniques, methods and terminologies especially through solved practice exercises. The MATLAB software is used for the purpose of this.

MATLAB is one of the powerful tools out there to write or create computer software or program, or simply computer code. You can use MATLAB to write different types of code, but it specializes in numerical (numbers) computations mainly applied in engineering, science and economics. We won't go into those big

These settings are general guidelines, and can should refer to the official MathWorks documentation and system requirements for the specific version of MATLAB you intend to install. Additionally, it's advisable to check for any specific recommendations or updates provided by MathWorks at the time of installation.

Community and Support

MATLAB has an active online community of users, and MathWorks provides comprehensive support resources. You can explore forums, documentation, and online tutorials to enhance your MATLAB skills. Additionally, MathWorks offers technical support for licensed users to address any issues or queries.

With MATLAB in your toolkit, you're well-equipped to tackle a wide range of computational tasks. In the upcoming chapters, we'll delve deeper into harnessing the full potential of MATLAB for effective coding and problem-solving.

1.4 Few Prerequsites

It is apparent that this book provides hands-on programming exercises using MATLAB. There are a few loose requirements that you may need to consider for maximum benefit from this document.

- **Basic Computer Literacy**: A fundamental understanding of using a computer; e.g. basic operations such as file management, navigating through directories, and installing software.
- **Basic Math**: A basic understanding of fundamental mathematical concepts like arithmetic operations is crucial.
- **Access to MATLAB Software** Ensure that you have or you can have access to MATLAB through any of the 4 avenues.
- **Eagerness to Learn**: Most importantly, you have to be willing to put in the effort to learn by practice with positive attitude and eagerness to learn. Computing can be both challenging and rewarding, and having a curious mindset will contribute significantly to your success with MATLAB.

1.5 Further Reading

There are other key figures (e.g. Alan Turing, Claude Shannon, Konrad Zuse, etc.) and developments (e.g. transistors, Windows, Linux, world wide web, etc.) in the history of computing that have not been covered. Books [9, 12] provide an excellent and concise coverage of these key developments and contrubutors in computing. Books [8, 3, 4, 14] also provide a comprehensive information on the history of computing. Additionally, there are many online infromation including Wikipedia that provide summarized information on computing. Paper [29] provides a summary of the development of the MATLAB software.

19) **AI (Artificial Intelligence)**: The development of computer systems that can perform tasks that typically require human intelligence, such as speech recognition and decision-making.
20) **Machine Learning**: A subset of AI involving development of algorithms that enable computers to learn patterns and make predictions from data.
21) **Classical Computing**: Traditional computing using bits, based on classical physics and logic.
22) **Quantum Computing**: Under research and development; utilizes the principles of quantum mechanics to perform computations, potentially solving certain problems exponentially faster than classical computers.
23) **MATLAB**: A proprietary multi-paradigm programming language and numeric computing environment developed by MathWorks.
24) **MATLAB Features**: Include matrix manipulation, plotting functions, and a large set of built-in mathematical functions, making it suitable for scientific and engineering applications.
25) **MATLAB Applications**: Used in various fields, including engineering, finance, biology, and signal processing, for tasks like data analysis, modeling, and simulation.
26) **Matrices**: Matrices provide a powerful way to represent and manipulate data, making them fundamental to many areas of science, economics and engineering.
27) **MATLAB & Matrices**:MATLAB's ability to perform matrix operations with ease and efficiency sets it apart from other programming languages.
28) **Getting Access to MATLAB**: Typically involves (1) purchasing license, (2) through institutions, (3) online free use, (4) install for 30-day free trial.

Review Questions

You may need to use the information provided in this chapter and/or use other resources from the recommended references or internet to answer the following questions.

1) In one or two sentence(s), explain how computing became relevant and started.
2) Create a small timeline outlining the key developments in computing from the Abacus up to the present Quantum Computing research. You can write the years and beside each year, provide a short desctription of the developed technology. Mention names of people where possible. Include at least 10 items in the timeline.
3) What boosted the development of computing technologies around the mid-20th century?
4) Why is Ada Lovelace referred to as the first computer programmer?
5) Can you try do a brief research and categorize the different programming languages into generations? Start from machine language up to the laatest generation languages.

2.31. Transposing a column vector

Let's implement the above example in MATLAB.

```
>> R=R=[1; 2; 3; 4]
   R =
      1
      2
      3
      4
>> R_transpose = R'
   R_transpose =
                1 2 3 4
```

Let's extend this to a 2D matrix in the following:

$$A = \begin{Bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \\ 17 & 18 & 19 & 20 \end{Bmatrix}$$

To find the transpose A^T, we switch the rows and columns:

$$A^T = \begin{Bmatrix} 1 & 5 & 9 & 13 & 17 \\ 2 & 6 & 10 & 14 & 18 \\ 3 & 7 & 11 & 15 & 19 \\ 4 & 8 & 12 & 16 & 20 \end{Bmatrix}$$

2.32. Transposing 2D array

Create a matrix A and find its transpose A^T in MATLAB. We use the colon (:) operator covered previously to generate our matrix.

```
>> A = [1:4; 5:8; 9:12; 13:16; 17:20]
   A =
      1  2  3  4
      5  6  7  8
      9  10 11 12
      13 14 15 16
      17 18 19 20
>> A_transpose = A';
```

If you check the value of `A_transpose`, it will be the transpose of matrix A.

```
>> A_transpose
   ans =
      1  5  9  13  17
      2  6  10 14  18
      3  7  11 15  19
      4  8  12 16  20
```

Understanding the transpose allows you to manipulate and analyze data more effectively by rearranging its structure. It is essential in various mathematical and engineering applications.

You can also use the `hold on` function in MATLAB to create many plots on the same figure. Try the following code in the editor.

```
x = linspace(0, 2*pi, 100);
y_sin = sin(x); y_cos = cos(x); y_2sin = 0.5*sin(2*x);
plot(x, y_sin, 'b', 'LineWidth', 1.2); hold on;
plot(x, y_cos, 'r', 'LineWidth', 1.2);
plot(x, y_2sin, 'g', 'LineWidth', 1.2); hold off;
xlabel('x'); ylabel('f(x)'); title('Trigonometric Functions');
xticks([0,pi/2,pi,3*pi/2,2*pi]);
xticklabels({'0','\pi/2','\pi','3\pi/2','2\pi'}); yticks
    ([-1,-0.5,0,0.5,1]);
legend('\sin(x)', '\cos(x)', '\frac{1}{2}\sin(2x)');
```

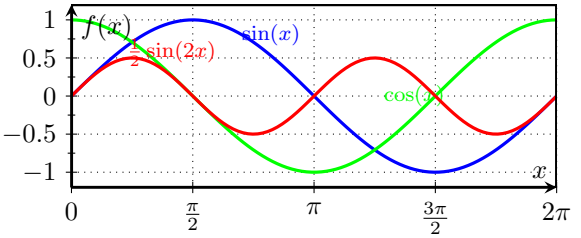


Figure 2.9: Trigonometric functions.

Scatter Plots

Scatter plots are used to visualize the relationship between variables by plotting individual data points on a plane. Each data point is represented by a marker at its corresponding $x - y$ coordinates. To create a 2D scatter plot, you need two vectors of equal length: one for the x-values and one for the corresponding y-values. Here's an example:

```
>> x = [1 2 3 4 5];
>> y = [5 2 7 4 8];
>> scatter(x, y);
>> xlabel('x'); ylabel('y');
>> title('Scatter Plot');
```

In this example, we have two vectors x and y representing the x and y coordinates of the data points. The scatter function is used to generate the scatter plot by plotting the data points on the graph. This generates a plot similar to Figure 2.10: Scatter plots are particularly useful for visualizing patterns, trends, or relationships between variables. They can also be customized by changing the marker shape, size, color, and adding legends to distinguish different groups of data points.

Bar Plots

Bar plots are commonly used to represent categorical data or to compare different quantities across categories. Each category is represented by a bar whose height

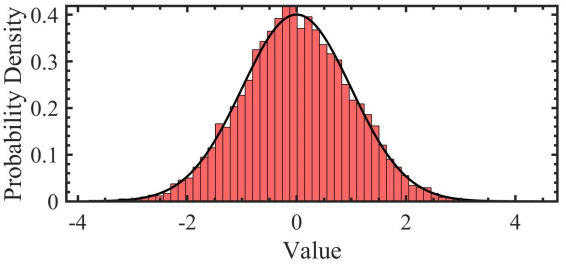


Figure 2.16: Histogram plot of Gaussian (normal) data distribution.

2.8 Further Reading

There are many books and online resources which aim at teaching MATLAB, including the official documentation of MATLAB from MathWorks. The topics covered here are basics and the keen reader is encouraged use these as pointers to explore more detailed online resources. References [20, 21] consist of a complete introduction to programming in MATLAB. This list is not restricted to those references though and among the online books there is a huge variety of books written for teaching MATLAB. Books [22, 24] also provide concrete introduction to MATLAB with specialization in numerical computing in science and engineering. The book [23] teaches data science in MATLAB. In terms of arrays including matrices that we briefly covered, a complete reference on matrices and linear algebra can be found from introductory books to linear algebra written by Gilbert Strang [27, 28].

The MATLAB software package also offers comprehensive help and documentation resources to assist you in learning and using the software effectively. The Help Browser provides access to the MATLAB documentation, which includes detailed explanations, examples, and references for each function and toolbox. You can search for specific topics or browse through the available documentation that can be accessed as shown in Figure 2.17.

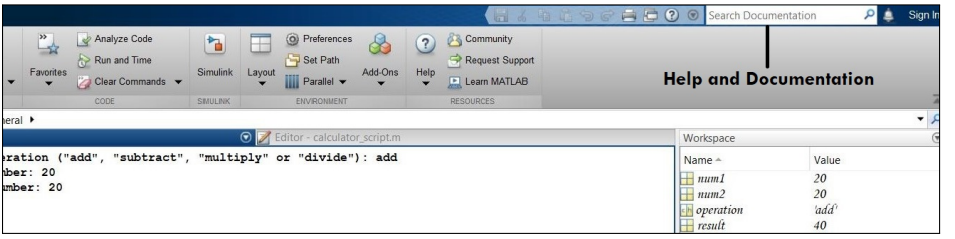


Figure 2.17: Help can be searched in the indicated space. This brings out comprehensive documentation of the search results.

Note that in the above code, we have used the **structure** data type that we briefly described in Table 2.3, to store the details of the excnange rates. The structure data type in MATLAB allows grouping different data elements together under a single variable. In the above code, the structure named `exchangeRates` is created to store the exchange rates for various currencies. Each currency is associated with a specific exchange rate.

Structure data type

A structure data type in MATLAB is a like a flexible container that allows the grouping of related data using named fields, providing a way to organize and access different types of information within a single variable.

Let's use the Currency Converter with some examples, then see how data is stored in the structure data type. Run the script and provide inputs in the command window as follows.

```
Enter the amount: 100
Enter source currency code (e.g., USD): USD
Enter target currency code (e.g., AUD): AUD
Converted Amount: 136 AUD
```

```
Enter the amount: 1000
Enter source currency code (e.g., USD): PGK
Enter target currency code (e.g., AUD): USD
Converted Amount: 281.6901 USD
```

Our currency converter is working perfectly! Now let's see how the data elements are stored in the structure data type`exchangeRates`. Type `exchangeRates` in the command window and press "ENTER".

```
>> exchangeRates
      exchangeRates =
        struct with fields:
        PGK: 3.5500
        AUD: 1.3600
        USD: 1
        NZD: 1.4800
        CNY: 6.4400
```

`exchangeRates` has five fields with data corresponding to the currency conversion factors. We can access each conversion factor as follows:

```
>> exchangeRates.PGK
        ans =
                3.5500
```

```
>> exchangeRates.CNY
        ans =
                6.4400
```

You can also create structure data type in `exchangeRates` is as follows:

4.6 Temperature Converter

Temperature is crucial in various aspects of everyday life, and its accurate measurement and conversion are essential for activities such as cooking, weather forecasting, scientific research, medical applications, and industrial processes. The ability to convert temperatures between different scales can ensure compatibility and practical applicatoin in these areas. In this mini-project, we'll develop a simple Temperature Converter to convert temperatures between Celsius (°C), Fahrenheit (°F), and Kelvin (K). Below are the conversion formulas.

i) **Celsius to Fahrenheit:**

$$F = \frac{9}{5}C + 32 \tag{4.4}$$

ii) **Celsius to Kelvin:**

$$K = C + 273.15 \tag{4.5}$$

iii) **Fahrenheit to Celsius:**

$$C = \frac{5}{9}(F - 32) \tag{4.6}$$

iv) **Fahrenheit to Kelvin:**

$$K = \frac{5}{9}(F - 32) + 273.15 \tag{4.7}$$

v) **Kelvin to Celsius:**

$$C = K - 273.15 \tag{4.8}$$

vi) **Kelvin to Fahrenheit:**

$$F = \frac{9}{5}(K - 273.15) + 32 \tag{4.9}$$

We can write and call functions in our main script.

```
% Main Temperature Converter Program
temperature = input('Enter temperature: ');
unit = lower(input('Enter unit (C, F, or K): ', 's'));
% Perform conversion based on the specified unit
switch unit
    case 'c'
        disp(['Converted Temperatures: Fahrenheit = ', num2str(
            celsiusToFahrenheit(temperature)), 'degrees F,
            Kelvin = ', num2str(celsiusToKelvin(temperature)), '
             K']);
    case 'f'
        disp(['Converted Temperatures: Celsius = ', num2str(
            fahrenheitToCelsius(temperature)), 'degrees C,
            Kelvin = ', num2str(fahrenheitToKelvin(temperature))
            , ' K']);
    case 'k'
        disp(['Converted Temperatures: Celsius = ', num2str(
            kelvinToCelsius(temperature)), 'degrees C,
            Fahrenheit = ', num2str(kelvinToFahrenheit(
            temperature)), ' degrees  F']);
    otherwise
        disp('Invalid unit. Please enter C, F, or K.');
end
%.....continued to next page.....
```

Index of Examples

PRACTICAL COMPUTING USING MATLAB

Solved Primary and Secondary School Exercises

JESSE GABRIEL

Title: PRACTICAL COMPUTING USING MATLAB: Solved Primary and Secondary School Exercises

Contents

Chapter 2

Math Exercises

2.1 Introduction

We begin the practical applications of using MATLAB, starting with solving exercises in the primary and secondary school mathematics. The exercises have been adopted from the subjects offered by Flexible Open and Distance Education (FODE) and the electronic copies of their lessons are freely available in PDF files and can be viewed or downloaded from their website URL link: `https://fode.education.gov.pg/courses.html`.

Once again, the main focus of this text is on writing MATLAB code; only relevant information like equations or formulas related to the questions are covered. It is assumed that the reader will go through the concepts in the lesson documents and other sources and understand them before implementing the solutions in MATLAB as demonstrated in this text. In addition, the approach used in solving the questions may not strictly be the same as they have been presented in other documents. The questions were selected based primarily on their suitability to be implemented in MATLAB. Most of the exercises are implemented in the MATLAB Editor as*script* files along with a few *function* files.

2.2 Grade Eight Mathematics

The Grade 8 Mathematics course at FODE has 6 strands: *1: Numbers and Applications*, *2: Space and Shapes*, *3: Measurements (1)*, *4: Measurements (2)*, *5: Data and Change*, and *6: Patterns and Algebra.*. The course is based on the National Department of Education (NDOE) upper Primary Mathematics Syllabus and Curriculum Framework for Grade 8. It forms part of the continuum of Mathematics learning from Grade 6 to 8, providing a foundation for the grades 9 and 10 mathematics courses.

Gr11 Gen.Math, U2, Act 11.2.3.4, Q3

The original price of a car is K40 000. If it depreciates at the rate of 2.5 per year, what is its value after 5 years?

Solution

Eq 2.7 will be used here.

```
P = 40000; R = 2.5; n = 5;
A = P*(1-R/100)^n; %next year's value
```

Gr11 Gen.Math, U2, Act 11.2.4.4, Q2

Jennifer wants to buy a brand-new car. She has really looked into models and prices and has decided to buy a hybrid. The vehicle costs K46,000 at a 3% interest rate over 6 years. Monthly payments are K699 per month.

a) How much will Jennifer pay overall for her hybrid?

b) How much will she pay in interest?

Solution

```
% (a) Overall payment
P=46000; R=3; mthlypay=699; mthsperyr=12; yrs=6;
totalpay = mthlypay*mthsperyr*yrs; %total payment
% (b) Interest
interest=totalpay-P; %interest
```

2.4.2 Statistics (Unit 3)

Gr11 Gen.Math, U3, Act 11.3.3.5, Q1

Compute the variance and standard deviation for the ungrouped data.

Table 2.6: Ungrouped data.

Data	**Frequency**	**Frequency x data**	**Deviation**	**Squared deviation**	**Frequency x squared deviation**
x	f	fx	$x-\bar{x}$	$(x-\bar{x})^2$	$f(x-\bar{x})^2$
11	2	–	–	–	–
12	3	–	–	–	–
13	5	–	–	–	–
14	6	–	–	–	–
15	9	–	–	–	–
16	7	–	–	–	–
17	3	–	–	–	–
18	4	–	–	–	–
19	1	–	–	–	–
Total	–	$\sum fx =$	–	–	$\sum f(x-\bar{x})^2 =$

2.7.2 Trigonometry and Vectors (Unit 2)

Gr12 Adv.Math, U2, Act 12.2.1.4.3, QA

Sketch the graph of each function (1) $f(x) = 2 + cos(x)$ *and (2)* $g(x) = -cos(x)$*:*

Solution

```
x=-1.5*pi:0.1:3.2*pi; % choose domain -1.5pi to 3.2pi
fx=2*cos(x); gx=-cos(x);
plot(x,fx,'b-',x,gx,'r-')
legend('f(x)=2+cos(x)','g(x)=-cos(x)')
```

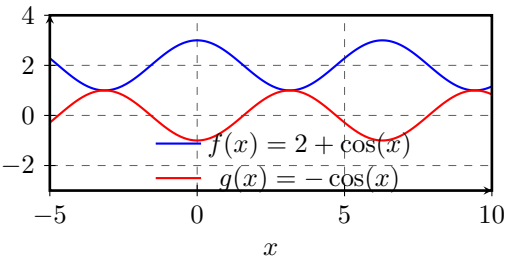


Figure 2.14: Plot of $f(x) = 2 + \cos(x)$, $g(x) = -\cos(x)$

Gr12 Adv.Math, U2, Act 12.2.1.5, Q3

Sketch the graph of $y = 2\sec(x)$ *in the domain* $0 \leq x \leq 2\pi$

Solution

To plot the function $y = 2\sec(x)$ in the domain $0 \leq x \leq 2\pi$ (radians) in MATLAB script, taking into consideration the asymptotes, we can write our script as follows. The asymptotes occur at $x = \frac{\pi}{2}$, $x = \frac{3\pi}{2}$, etc.

```
x = linspace(0, 2*pi, 1000); % Define the domain
% Calculate y values, avoiding asymptotes
y = zeros(size(x));
for i = 1:length(x)
    if abs(mod(x(i), pi) - pi/2) < 1e-4
        y(i) = NaN;  % Avoid division by zero at asymptotes
    else
        y(i) = 2 / cos(x(i));
    end
end
plot(x,y, 'b', 'LineWidth',2); xlabel('x'); ylabel('y');
title('Plot of y = 2sec(x)'); hold 'on';
asymptotes = pi/2:pi:3*pi/2;  % Locations of asymptotes
for i = 1:length(asymptotes)
    xline(asymptotes(i), '--r'); % Add vertical asymptote lines
    %xline(asymptotes(i), '--r', 'Asymptote'); %labeling
end
axis([0, 2*pi, -10, 10]); legend('y = 2sec(x)'); hold 'off';
```

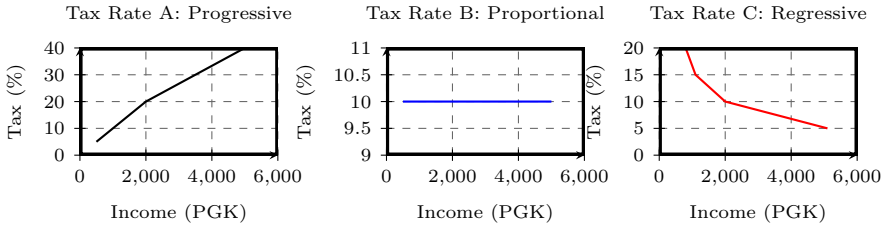
```
>> results
results =
         500          25           5          50          10         100          20
        1000         100          10         100          10         150          15
        2000         400          20         200          10         200          10
        5000        2000          40         500          10         200           4
```

Gr12 Econ, M2, Act 12.2.3, Q9

Sketch a graph illustrating each method of tax rate. Make sure to label all the axis and give a title to each graph.

Solution

```
subplot(1,3,1);plot(AnnualIncome,TaxARate,'k','LineWidth',1.5);
title('Tax Rate A: Progressive')
xlabel('Income (PGK)'); ylabel('Tax (%)')
subplot(1,3,2);plot(AnnualIncome,TaxBRate,'b','LineWidth',1.5);
title('Tax Rate B: Proportional')
xlabel('Income (PGK)'); ylabel('Tax (%)')
subplot(1,3,3);plot(AnnualIncome,TaxCRate,'r','LineWidth',1.5);
title('Tax Rate C: Regressive')
xlabel('Income (PGK)'); ylabel('Tax (%)')
```



3.3.2 The Global Economy (Module 3)

Gr12 Econ, M3, Act 12.3.2, Q9

Given the following information in the table, calculate the answers in the fourth (4^{th}) in the table.

a) Compute the terms of trade for year 1 and 2.

b) What is the 'terms of trade' for year 2 and 3?

Year	Export Price Index	Import Price Index	Terms of Trade Index
1	100	100	100
2	120	125	-
3	130	120	-

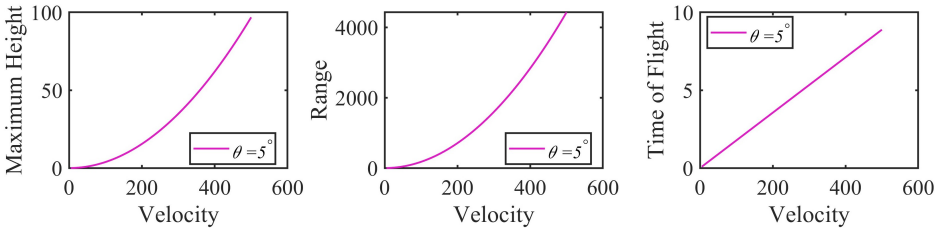


Figure 5.9: Maximum height, range, and time of flight for a projectile as a function of velocity (0-500 m/s) and angle of launch at 5 degrees.

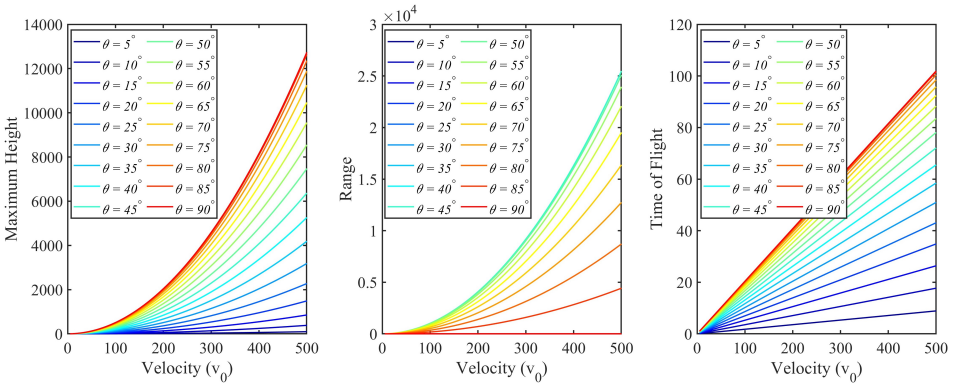


Figure 5.10: Maximum height, range, and time of flight for a projectile as a function of initial velocity and angle of launch.

3. Simple Harmonic Motion (SHM)

A mass-spring system undergoes simple harmonic motion. Its displacement, velocity, and acceleration of the mass as functions of time are given by:

$$\left.\begin{aligned} x(t) &= A\cos(\omega t + \phi) \\ v(t) &= -A\omega\sin(\omega t + \phi) \\ a(t) &= -A\omega^2\cos(\omega t + \phi) \end{aligned}\right\} \tag{5.51}$$

(where A is the amplitude, ω is the angular frequency, and ϕ is the phase constant).

a) Write a MATLAB program (function) to calculate $x(t)$, $v(t)$, and $a(t)$.
b) Test your function in (a) using $t = 5$, $A = 1$, $\omega = 10$, and $\phi = \pi$.
c) Create a new script file. In it, create a vector of t from 0 to 3 at an increment (step size) of 0.1. Call your function in (b) and provide all the inputs for calculation of the values of x, v, and a for the different time points. You will need to write a loop for this. Plot the results on one graph against t.
d) Reduce the step size in (c) to 0.01 and repeat the calculation of x, v, and a. Plot the results again and describe/reason any difference. You will have a plot that looks like Figure 5.11.

Index of Examples

References

[1] Barker, P. (2007). Java Methods for Financial Engineering: Applications in Finance and Investment. Springer Science + Business Media.. https://doi.org/10.1007/978-1-84628-741-1.

[2] Basilio, C. (2016). Grade 10 Mathematics - Unit 5: Trigonometry. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr10/maths.html.

[3] Basilio, C. (2016). Grade 10 Mathematics - Unit 6: Measurement. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr10/maths.html.

[4] Fernandez, L.B. (2016). Grade 8 Mathematics - Strand 1: Numberes and Applications. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr8/maths.html.

[5] Fernandez, L.B. (2016). Grade 10 Mathematics - Unit 4: Functions and Graphs. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr10/maths.html.

[6] Fernandez, L.B. (2014). Grade 12 Mathematics A - Unit Module 1: Patterns and Algebra. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr12/mathsa.html.

[7] Fernandez, L.B. (2016). Grade 12 Mathematics A - Unit Module 2: Trigonometry and Vectors. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr12/mathsa.html.

[8] Fernandez, L.B. (2016). Grade 12 Mathematics A - Unit Module 3: Calculus. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr12/mathsa.html.

[9] Gabriel, J. (2024). Practical Computing using MATLAB: The Basics.

[10] Joseph, M., and Masule, J. (2017). Grade 10 Mathematics - Unit 2: Managing Your Money. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr10/maths.html.

[11] Waken, G. (2017). Grade 12 Physics - Module 2: Temberature and Heat. Flexible Open and Distance Education. Retreived from https://fode.education.gov.pg/Gr12/phy.html.