Resource Link:
https://serc.carleton.edu/teaching_computation/workshop_2018/activities/210996.html

EA413 COMPUTER PROGRAMMING FOR ENGINEERS

ASSIGNMENT

INSTRUCTIONS: All students should strictly adhere to ethical writing style. Avoid plagiarism and any form of ethical misconduct. Acknowledge, paraphrase and properly cite the reference materials used.

Note: The semester assignment will provide students with hands-on experience on Matlab programming skills covered in the class.

Assignment Question

You are to assume a constant thrust and constant mass of the SATURN V rocket. NASA fires the motors at negative ***t*** and lifts off at ***t = 0***. In contrast the Russians fire the motor ***at = 0*** and lifts off at positive ***t***. Use different values of ***t*** to launch the rocket. (i.e., ***t = 5, 10, 15, …*** and so on).

1. Using the MATLAB program given in RockLaunch.m file as an example, design, build and test MATLAB codes which simulates a SATURN V rocket launch, using the constant acceleration model. Write a MATLAB script to launch the rocket specifying the initial conditions at the beginning of the program so that they can be changed. Modify the function GetAccelaration to GetConstAcceleration. The new function will be renamed as GetConstAcceleration.m file.
2. Add MatLab codes to plot a graph of the height against time, velocity against time, acceleration against time for the SATURN V rocket launch.
3. Modify the plot function to produce three figures of height, velocity and acceleration with three subplots command, plotting height in blue and velocity in green and acceleration in red colors.
4. Modify the RocketLaunch.m function to RocketLaunchH.m and GetConstAcceleration function into GetHAcceleration.m, and modify them to determine the horizontal displacement at which the rocket exceeds the speed of sound. Assume that, instead of launching vertically, the rocket launched horizontally.

Please keep in mind the following policy:

- ✓ Submit the hard copies of your work to your class representative on the due date. Also note that the file(s) of your work should be turned in by e-mail by the due date to your class representatives.
- ✓ When your work is not turned in on time, a late penalty will be given. Notice that it is 10% a day within a week, and thereafter, your score is 0.
- ✓ You may discuss problems with your friends, but all work must be done individually and

you must be able to prove that you understand everything that you hand in. **Any copied work will be given 0, for both the copied work and the work it was copied from.**

- ✓ **Ethical research is the foundation of qualitative education. The students should strictly adhere to ethical standards, avoid plagiarism and any form of ethical misconduct.**

The following should be turned in, in the form of a technical report. Remember that all the files should be immediately turned in by e-mail to your class representatives.

- ✓ - Complete source code of your program (M-Files).
- ✓ - Compile messages (screen capture)
- ✓ - Input and output data of your running test, including log messages (screen capture)
- ✓ - Documentation, including (i) explanation of your program and (ii) discussion on your validation.

Grading criteria

- ✓ Clarity: Does it clearly state what to develop/solve and what have been done?
- ✓ Originality: Is the proposed idea/solution original?
- ✓ Approach: How much are the development process and experiment environment realistic?
- ✓ Presentation/Demo: Are the presentation and demo persuasive?
- ✓ Report: Are the reports (final report) nicely organized and well presented?

Modeling Rocket Flight Trajectory

Abstract

This problem is a computational problem solving. Computational problem solving often involves applying a consistent and structured approach to solving problems. As essential as problem solving is, several approaches are combined that will lead to the final solution. However, problems must be approached methodically, applying algorithms or step by step procedure by which one arrives at a solution. In this classroom activity, the students will develop a model to determine the time of flight of rocket missile launch from the surface of the earth and distance travelled when the rocket returns to the ground. To explain the problem in a clear way, the student will write MATLAB codes to calculate the range that a rocket missile would travel when it is launched with an initial velocity of 20 m/s at an initial angle of θ. Calculate this range for all angles between 0° and 90° in steps of 1° degree. Determine the angle that will result in the maximum range of the rocket. Plot the trajectory of the rocket missile for angles between 10° and 90° in 1° increments. Plot the maximum-range trajectory in a different color and with a thicker line. Assume that the atmospheric air has no effect on the trajectory of flight. Furthermore, the flight of a rocket launch can be confronted with real world situations. The Saturn V rocket used in the Apollo 11 spacecraft was employed to provide better understanding of a real-life scenario. The methodology makes use of the second order differential equation to model the Saturn V rocket launch. The rocket can be modelled using MATLAB codes including its position, time derivatives, velocity and acceleration.

1.0 Introduction

The computation of rocket trajectories is mainly done using purely numerical methods that will take charge of all the relevant parameters and yield the desired end product results. A demand for analytical methods that can offer more explicit the effect of the various rocket's atmospheric parameters of the trajectory that can be utilized as test cases with infinite accuracy. An analytical method can also be used. The analytical techniques consider the following: (1) propellant consumption resulting from variable rocket mass; (2) nonlinear aerodynamic forces proportional to the square of the velocity; (3) exponential dependence of the mass density with altitude for an isothermal atmospheric layer. A research paper [1] proposes four analytical methods of calculation of rocket trajectories in an isothermal atmospheric layer using new exact solutions of the equations of motion. The proposed methods are developed for the simpler case of a vertical climb and will then be extended to the more practically case scenario of a gravity turn.

UC Rocketry is a rocket research group developed by the University of Canterbury (UC). The rocket has broken the world altitude record for an I-class motor, an achieved impulse of 320-540 Ns. The university has run the rocketry course for the first time in New Zealand. The research paper presented the development and results of the world record rocket namely, Milly. In addition, detail fundamental elements of the rocketry final year engineering course were provided. These include; manufacturing process; wind tunnel testing; avionics and control. Furthermore, a final rocket launch called Smokey was provided. The two rockets Milly and Smokey are used as an example of the design, implementation and testing processes. This development paradigm has immensely contributed to the graduate space program research in New Zealand [2].

In another development, a customized vertical wind tunnel was built by the UC Rocketry space program research group. This development has been critical for the success of the UC Rocketry as it lets the optimization of avionics and control system parameters for an earlier flight. The research paper summaries the construction of the wind tunnel and includes an analysis of flow quality including swirl. In addition, a minimal modelling methodology for roll dynamics was developed. This methodology can extrapolate wind tunnel behavior at low wind speeds in relation to much higher velocities encountered during flight. The outcomes demonstrate the models get the roll flight dynamics in two rocket launches with mean roll angle errors fluctuating from 0.26 to 1.5 points across the flight data ranges. A predictable fluctuation on wind tunnel tests and flight, including canard-fin interaction behavior are presented in the model parameters. The presented results prove that the vertical wind tunnel is a significant tool for the modelling and control of sounding rockets [3].

The German Aerospace Center (DLR) launched a support program for students to acquire, construct and set up their rockets in April 2012. The program goes by the acronym STERN (STudentische Experimental-RaketeN) which is also the German word for star. The STERN was funded by the Federal Ministry of Economics and Technology (BMWi) and conducted by the DLR Space Administration of the German Aerospace Center DLR. The STERN program provides aerospace engineering faculties at universities with opportunities to introduce students realistically to subjects linked to space transport. It was envisaged that within the project time frame of three years the soon-to-be engineers will develop their own rockets. Another important issue is that there are no limits regarding peak altitude or the propulsion system used. The designed rockets should have a minimal telemetry system that will transmit vital trajectory and housekeeping data back to earth during flight and provide information to the students as well as the rocket altitude [4].

An approach for a Problem-Based Learning (PBL) experience for undergraduate students of aerospace engineering was provided by the work of the authors in [5]. The experience affords the students the ability to construct a model rocket using locally available materials. They also calculate all the relevant measures to design and characterize the rocket and they examine the hardiness of their intent. Moreover, the students launch the rocket with the matching payload and verify the flight parameters using an on-board altimeter. Lastly, they also compare the flight parameters with the theoretically expected values. Practicing this simple system, the students are later presented in the simulation of complex flows, utilizing standard techniques. The authors recognize that their students get rapidly involved in the project, permitting them to take several practical abilities, besides making an exact knowledge of the physics of rockets and of fluid dynamics.

The goal of the Center for Simulation of Advanced Rockets is a comprehensive simulation of solid propellant rockets. Achieving this goal will require advancements in technical matters linked to several scientific disciplines, in system integration, and in the computational infrastructure for supporting such large-scale simulations. It involves a multidisciplinary team of applied scientists, physical scientists, and computer scientists to cultivate and conduct away the necessary mathematical models, algorithms, and software to create a virtual rocket. The authors gave an outline approach for merging rocket design, simulation and computer science principles [6]. Another approach for teaching model rocketry in science was provided by the authors in [7].

The research work that introduces a model rocket project suitable for sophomore aerospace engineering students has been presented. The project incorporates elements of drag estimation, thrust determination and analysis using digital data acquisition, statistical analysis of data, computer aided drafting, programming, team work and written communication skills. The student-

made rockets are set up in the university baseball field with the objective of taking a specific amount of payload so that the rocket achieves a specific altitude before the parachute is installed. Throughout the course of the project, the students are ushered into real-world engineering practice through written report submission of their design works. For several years, the project has proven to enhance the learning objectives, provides cost effective and has proven good outcome results [8].

The National Aeronautics and Space Administration (NASA), provides a classroom topic that gives awesome experience in rocket design. The educator guide provides the scientific, technical, engineering and mathematical foundations of rocketry. The guide provides exciting classroom opportunities for authentic hands-on, minds-on experimentation to enhance the educational experience. The activities and lesson plans contained in this educator guide emphasize hands-on science, prediction, data collection and interpretation, teamwork, and problem-solving processes. In summation, the guide contains background information about the history of rockets and basic rocket science. The rocket activities in this guide support national curriculum standards for science, mathematics and engineering [9].

A stochastic six-degree of freedom flight simulator for passively controlled high power rockets was proposed by the authors in [10]. The research paper presented a method for simulating the flight of a passively controlled rocket in six degrees of freedom, and the descent under parachute in three degrees of freedom. Also exhibited is a method for modeling the uncertainty in both the rocket dynamics and the atmospheric conditions using stochastic parameters and the Monte Carlo method. Furthermore, the authors introduced a method for measuring the uncertainty in the atmospheric conditions using historical atmospheric data. The center masterpiece of the simulation algorithm is a numerical integration of the rocket's equations of motion using the Runge-Kutta-Fehlberg method. The location of the rocket's center of mass is drawn using three dimensional cartesian coordinates and the rocket's orientation is described using quaternions. The input parameters for the simulation process are made using stochastic process by adding Gaussian noise. For parameters related to the atmosphere at the adjacent altitude, the correlation of the variation of the noise is a function of altitude and noise. The main simulation algorithm with stochastic parameters, was tested within a Monte Carlo wrapper to evaluate the total uncertainty in the rocket's flight path trajectory. The results of the demonstration of the simulator showed a promising correlated value.

2.0 Problem Background

This classroom activity is a computational problem solving. Computational problem solving often requires using a coherent and structured approach to resolving problems. For instance, a rocket missile is launched into flight from the ground at a speed of 20 miles per second at 45^{o} degrees above the horizontal over level ground. Determine the time of flight and the distance travelled when the rocket returns to the ground.

Let's begin with the theoretical background using object motion in physics [11], [12], [13]. Suppose, we assume the negligible effect of the atmospheric air friction and ignore the curvature of the earth, a rocket missile that is launched into the air from the surface of the earth will follow a parabolic flight path as illustrated in Fig. 1 below. The height of the rocket at any time *t* after it is launched is given by Equation (1).

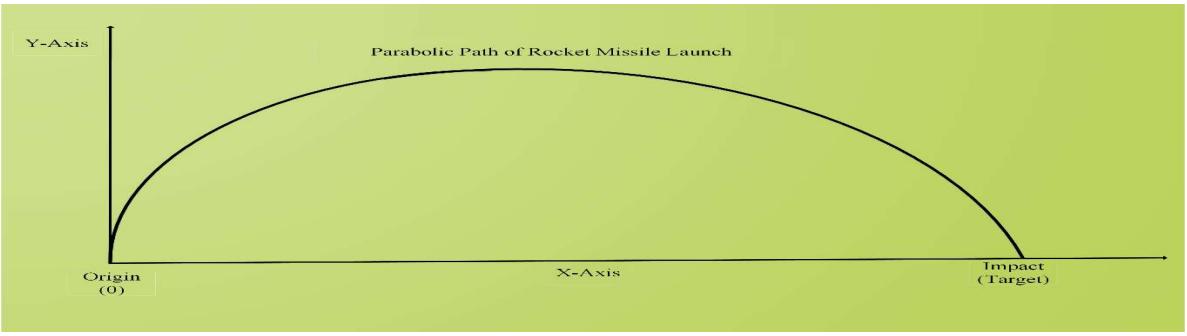


Fig. 1 A parabolic path movement of a rocket missile launch

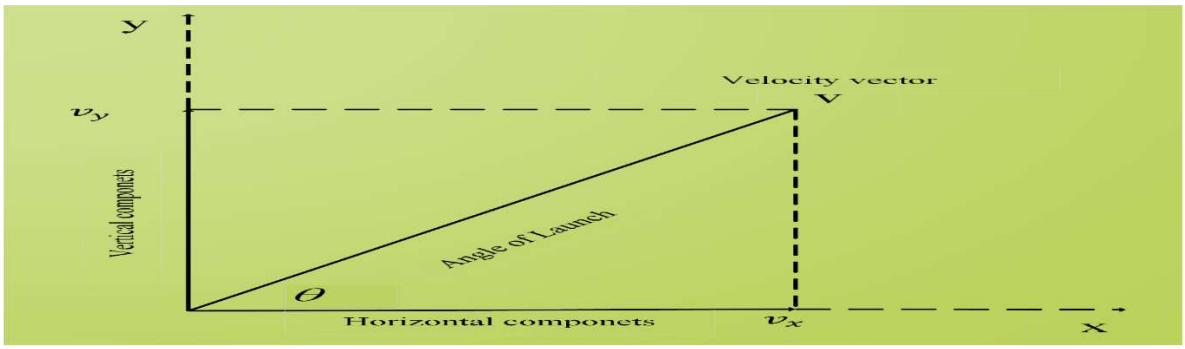


Fig. 2 The horizontal and vertical components of a velocity vector v at an angle θ with respect to the horizontal.

$$y(t) = y_o + v_{yo}t + \frac{1}{2}gt^2 \qquad (1)$$

The parameter y_o is the initial height of the of the rocket above the ground, v_{yo} is the initial vertical velocity of the rocket, and g is the acceleration due to the earth's gravity. The horizontal distance (range) traveled by the rocket as a function of time after it is launched is given by Equation (2).

$$x(t) = x_o + v_{xo}t \qquad (2)$$

The parameter x_o is the initial horizontal position of the rocket on the ground, and v_{xo} is the initial horizontal velocity of the rocket. Assume the rocket is launched with some initial velocity at an angle of degrees with respect to the earth's surface. The rocket will have the initial horizontal and vertical components of velocity as given in Equations (3) and (4).

$$v_{xo} = v_o \cos\theta \qquad (3)$$

$$v_{yo} = v_o \sin\theta \qquad (4)$$

Herein, we assume that the rocket is initially launched from the position $(x_o, y_o) = (0, 0)$with an initial velocity of say 20 meters per second at an initial angle of degrees. Write a MATLAB script (program) that will plot the trajectory of the rocket and also determine the horizontal distance traveled before it touches the ground. The MATLAB codes should plot the trajectories of the rocket for all angles from 5 to 85° in 1° steps, and it should determine the horizontal distance traveled for all angles θ from 0 to 90° in 1° steps. Lastly, it should determine the angle that maximizes the range of the rocket and plot that particular trajectory in a different color with a bolder line.

To propose a solution to this problem, an additional information is required that is, an equation for the time that the rocket returns to the ground. At that time, we can calculate the (*x*, *y*) position of the rocket using Equations (1) to (6). In doing this, for several times between 0 and the time that the rocket returns to the ground, those points can be used to plot the rocket's trajectory [13], [14].

$$y(t) = y_o + v_{yo}t + {}^{1}/_{2}\, gt^2 \qquad (5)$$

$$0 = 0 + v_{yo} + {}^{1}/_{2}\, gt^2$$

$$0 = \left(v_{yo} + {}^{1}/_{2}\, gt\right) t$$

The time that the rocket will remain in the air after it is launched may be calculated from Equation (1). The rocket touched the ground at the time *t, for* which *y* (*t*) = 0. Recalling that the rocket will start from ground level (y (0) = 0), and solving for, t, we can obtain Equations 5 and 6. Hence, the rocket will be at ground level at a time $t_1 = 0$, when launched, and at a time t_2 as given in Equation (6).

$$t_2 = -{}^{2\, v_{yo}}/_{g} \qquad (6)$$

As stated in the problem statement, we know that the initial velocity is 20 meters per second and that the rocket will be launched at all angles from 0° to 90° in 1° steps. Lastly, the acceleration due to the earth's gravity is –9.81 meters per second squared. Let's use the computational problem design technique and MABLAB codes to solve this problem.

2.1 Problem Definition

1. Problem statement
   Precisely, the problem statement can be stated as: Write a MATLAB script to calculate the range that a rocket missile would travel when it is launched with an initial velocity of 20 m/s at an initial angle θ. Calculate this range for all angles between 0° and 90° in 1° steps. Determine the angle that will result in the maximum range of the rocket. Plot the trajectory of the rocket for angles between 5° and 85° in 1° increments. Plot the maximum-range trajectory in a different color and with a bolder line. Assume that the atmospheric air has no effect on the trajectory of flight. To test the accuracy of the MATLAB program, we run the Matlab program and view the numerical and plotted graphs.
2. Define the inputs and outputs.
   The problem is already defined. Therefore, no inputs are required. The values of the parameters v_o *and* θ are known. The outputs from this program will be a table showing the range of the rocket for each angle θ, the angle θ for which the range is maximized, and a plot of the specified trajectory.
3. Devise the algorithm.
   To devise the algorithm the following steps should be followed:
   i) Calculate the range of the rocket for angle θ between 0º and 90° degrees.
   ii) Write a table of ranges.
   iii) Determine the maximum range and write it out.

iv) Plot the trajectories for angle θ between 5 and 85° degrees.

v) Plot the maximum-range trajectory.

As listed above, to calculate the maximum range of the rocket for each angle θ, we will first calculate the initial horizontal and vertical velocity from Equations (3) and (4). Then, we will determine the time when the rocket returns to earth from Equation (6). Finally, we computed the range at that time from Equation (5).

This pseudocode creates and initialize an array to hold ranges.

```
for ii = 1:91
        theta <- ii - 1
        vxo <- vo * cos(theta*conv)
        vyo <- vo * sin(theta*conv)
        max_time <- -2 * vyo / g
        range(ii) <- vxo * max_time
end
```

Next, we must write a table of ranges. The pseudocode for this step is to write heading.

```
for ii = 1:91
        theta <- ii - 1
        print angle theta and range
end
```

The maximum range can be found with the max function. This function returns both the maximum value and its location. The pseudocode for this step is given below:

```
[maxrange index] <- max(range)
Print out maximum range and angle (=index-1)
```

To plot the flight trajectories, a nested for loops are used. To get all of the plots to appear on the screen, we must plot the first trajectory and then set the ***hold on*** command before plotting any other trajectories. After plotting the last trajectory, we must set ***hold off*** command. To perform this calculation, we will divide each trajectory into 21-time steps and find the ***x*** and ***y*** positions of the rocket for each time step. Then, those (x, y) positions are plotted. The pseudocode for this step is:

```
for ii = 5:10:85
        % Get velocities and max time for this angle
        theta <- ii - 1
        vxo <- vo * cos(theta*conv)
        vyo <- vo * sin(theta*conv)
        max_time <- -2 * vyo / g
        Initialize x and y arrays
        for jj = 1:21
```

```
            time <- (jj-1) * max_time/20
            x(time) <- vxo * time
            y(time) <- vyo * time + 0.5 * g * time^2
        end
plot (x, y) with thin green lines
Set "hold on" after first plot
end
Add titles and axis labels
```

Finally, we must plot the maximum range trajectory in a different color and with a thicker line.

```
vxo <- vo * cos(max_angle*conv)
vyo <- vo * sin(max_angle*conv)
max_time <- -2 * vyo / g
Initialize x and y arrays
for jj = 1:21
        time <- (jj-1) * max_time/20
        x(jj) <- vxo * time
        y(jj) <- vyo * time + 0.5 * g * time^2
end
plot (x, y) with a thick red line
hold off
```

4. Implement the algorithm using MATLAB codes.
   At this stage, the algorithm is transformed into MATLAB codes. This is provided in a m-file titled (ROCKET_MISSILE. m).
5. Test the program.
   To test this program, the Matlab program is run and generates the desired output. Please, refer to the program's output file (Rocket_Trajectory1.svg, Rocket_Trajectory1.tif, Numerical_Computation_Results.pdf). Figs. 3 and 4 illustrate the output generated by the MATLAB program codes.

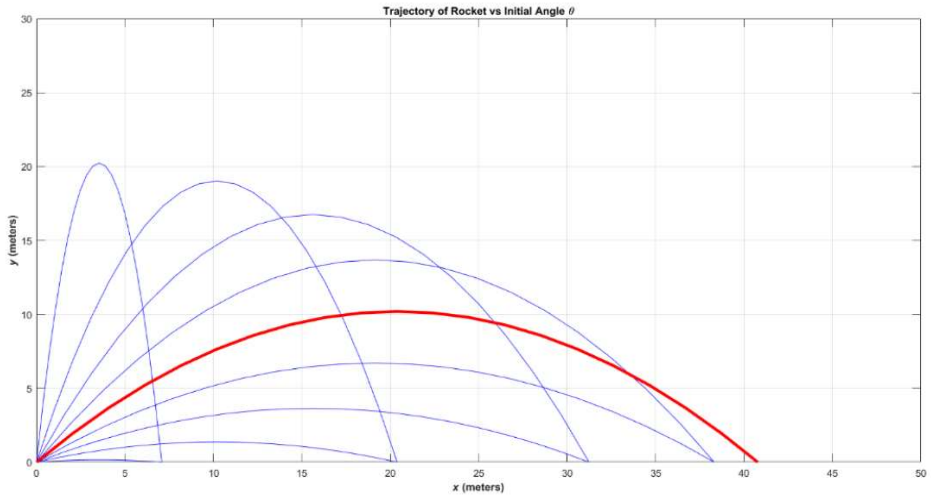


Fig. 3 MATLAB Plot of the Rocket Trajectories

```
>> ROCKET_MISSILE
Range versus angle theta:
  0   0.0000
  1   1.4230
  2   2.8443
  3   4.2621
  4   5.6747
  5   7.0805
  6   8.4775
  7   9.8643
  8  11.2390
  9  12.6001
 10  13.9458
 11  15.2745
 12  16.5846
 13  17.8745
 14  19.1426
 15  20.3874
 16  21.6073
 17  22.8009
 18  23.9668
 19  25.1034
 20  26.2095
 21  27.2836
 22  28.3245
 23  29.3309
 24  30.3015
 25  31.2352
 26  32.1309
 27  32.9874
 28  33.8038
 29  34.5789
 30  35.3119
 31  36.0019
 32  36.6481
 33  37.2496
 34  37.8057
 35  38.3157
 36  38.7791
 37  39.1952
 38  39.5635
 39  39.8837
 40  40.1553
 41  40.3779
 42  40.5514
 43  40.6754
 44  40.7499
 45  40.7747
 46  40.7499
```

```
 47  40.6754
 48  40.5514
 49  40.3779
 50  40.1553
 51  39.8837
 52  39.5635
 53  39.1952
 54  38.7791
 55  38.3157
 56  37.8057
 57  37.2496
 58  36.6481
 59  36.0019
 60  35.3119
 61  34.5789
 62  33.8038
 63  32.9874
 64  32.1309
 65  31.2352
 66  30.3015
 67  29.3309
 68  28.3245
 69  27.2836
 70  26.2095
 71  25.1034
 72  23.9668
 73  22.8009
 74  21.6073
 75  20.3874
 76  19.1426
 77  17.8745
 78  16.5846
 79  15.2745
 80  13.9458
 81  12.6001
 82  11.2390
 83   9.8643
 84   8.4775
 85   7.0805
 86   5.6747
 87   4.2621
 88   2.8443
 89   1.4230
 90   0.0000

Max range is  40.7747 at 45 degrees.
>> ROCKET_MISSILE
Range versus angle theta:
  0   0.0000
```

Fig. 4 MATLAB Numerical Computation of the Rocket Range vs Angle Theta

3.0 Real-life Application

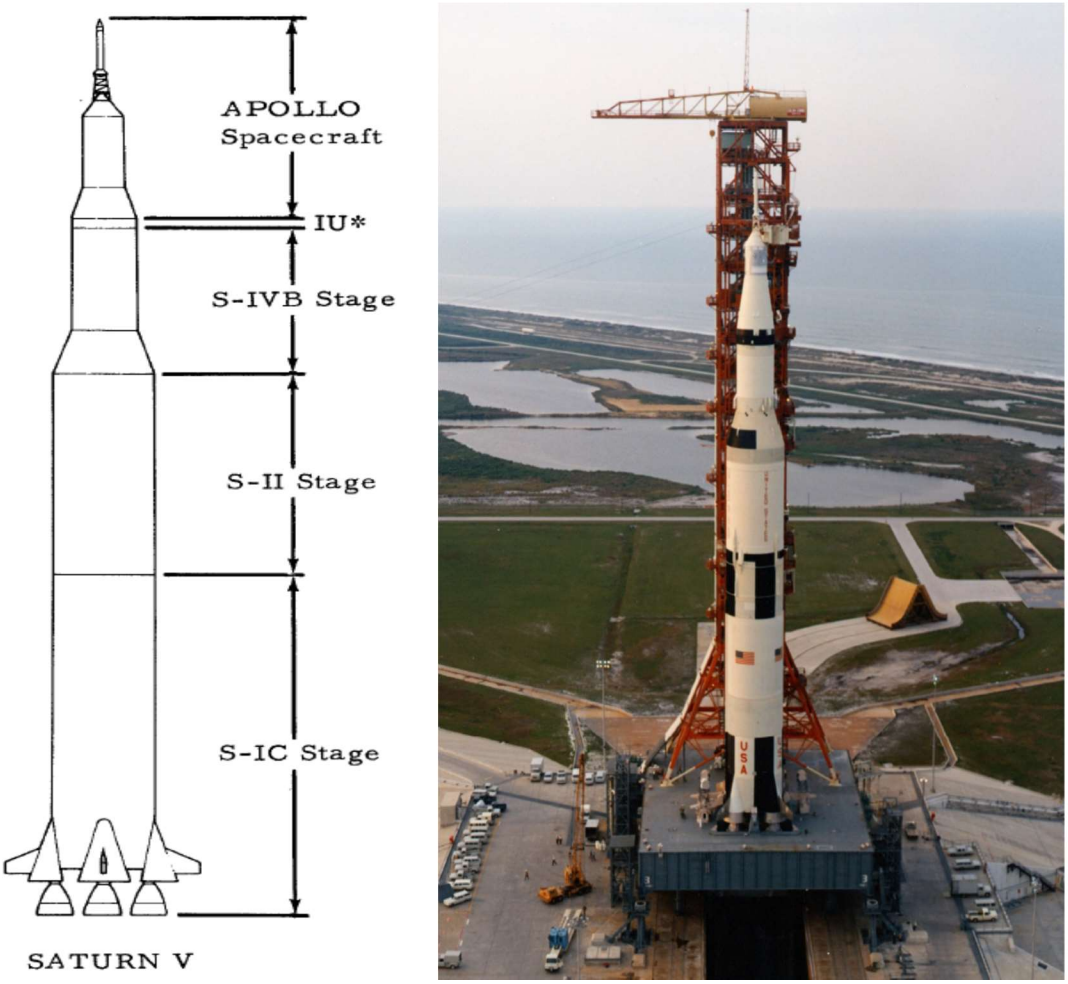


Fig. 3 The Saturn V Rocket (Used in Apollo 11 spacecraft) [9]

The model presented in the previous section is based on physics principles [12], [13]. However, the flight of a rocket launch can be presented in real world situations. Fig. 3 illustrates the Saturn V rocket used in the Apollo 11 spacecraft [9]. This section, employ the use of the second order differential equation to model the Saturn V rocket launch. The rocket can be modelled mathematically, including its position, time derivatives, velocity and acceleration. The second order differential equation can be broken into a number of first order differential equations using the technique called approximating gradients [14]. Let's consider a one-dimensional space, assume that at any given time $\boldsymbol{k}$ the values of the instantaneous acceleration $\boldsymbol{a_k}$, velocity $\boldsymbol{v_k}$, and position $\boldsymbol{x_k}$ of the rocket. Applying the forward differencing method involving the velocity $\boldsymbol{v_{k+1}}$ of the rocket at timestep $\boldsymbol{k+1}$ and a small-time interval change of $\boldsymbol{\Delta t}$, the acceleration of the rocket is given in equation 6 as:

$$a_k = \frac{dv_k}{d_t} \approx \frac{v_{k+1} - v_k}{\Delta t} \tag{7}$$

This implies that,

$$v_{k+1} = v_k + \Delta t\, a_k \tag{8}$$

In the same way, using the position and acceleration parameters we obtain:

$$x_{k+1} = x_k + \Delta t\, v_k \tag{9}$$

The above equations provided the means to compute the acceleration of the rocket at any given time step ***k*** with its position and velocity parameters. In addition, these equations will further provide a means to compute position and velocity at the next timestep ***k+1, k+2, k+3, …*** over all time intervals. The numerical solution to the differential equation is a MATLAB vector of ***x*** values corresponding to vector of time values. For instance, at time ***t*** = 0.1, ***x*** is initially -2 for the SATURN V rocket. Table 1 gives the numerical values of ***k***, ***t*** and ***x*** parameters [14], [15].

Table 1: A Numerical Solution of the Three Parameters ***k***, ***t*** and ***x***

Parameter	Numerical Values								
k	1	2	3	4	5	6	7	8	…
t	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	…
x	-2	1.8	2.3	2.1	4.7	6.0	6.1	7.4	…

4.0 Saturn V Rocket Launch Simulation with Matlab

Table 1 provides an information regarding the Saturn V rocket. Below are MATLAB codes that used the ***for loop*** at each time step, calculates an acceleration follow by the velocity, then, follow by a new position of the rocket [15], [16].

```
v = 0;                  % initial velocity
h = 0;                  % initial height
tstart = 0;             % start time
dt = 0.1;               % time step
tstop = 300;            % how long to simulate for
for t = tstart :dt: tstop
a = GetAcceleration ();     % get current acceleration
% find the new values
h = h + dt * v;             % update height
v = v + dt * a;             % update velocity
end ;
```

In order to plot these parameters, it is important to capture the parameter values into vectors. Implying that the values of ***a, v, h***, and ***t*** are stored in vectors. For instance, the first statement becomes ***v (1) = 0***; and the updates become ***v (k+1) = v (k) +…*** and so on. Nevertheless, in real world rocket launching, the rocket flight telemetry parameters are not stored in the program. The final sets of the updated velocity and height parameters are not stored. However, these parameters

belong to the time (t = tstop + dt) and the height is first updated before the velocity. The updated codes are illustrated below:

```
v = 0;                          % initial velocity
h = 0;                          % initial height
tstart = 0;                     % start time
dt = 0.1;                       % time step
tstop = 300;                    % how long to simulate
k = 0;
for t = tstart :dt: tstop       % vector of times
a = GetAcceleration ();         % get current acceleration
% capture telemetry for plotting
k=k+1; A(k)=a; V(k)=v; H(k)=h; T(k)=t;
% find the new values
h = h + dt * v;                 % update height
v = v + dt * a;                 % update velocity
end
```

The Newton's laws of motion can be employed in order to determine the Trust and Mass of the rocket. We can assume that the thrust and mass of the rocket are constants. Suppose the rocket produces a trust ***F*** and mass ***M***, then, we can have the following equations:

$$Ma = F - Mg \tag{10}$$

Where the parameter g is the acceleration due to gravity at the earth's surface. Therefore, the new acceleration will be as given in equation 10.

$$a = \left(\frac{F}{M} - g\right) \tag{11}$$

In order to formulate the model of the Saturn V rocket, the following assumptions have to be made:

- ✓ A rocket's mass does not remain constant.
- ✓ After a time, it runs out of fuel.
- ✓ The faster a rocket move, the greater the air resistance.
- ✓ The density of the air changes with height.

Suppose the rocket is flying vertically, and ignoring the change in acceleration due to gravity, we can write the force balance on the rocket as:

$$M(t)\, a(t) \;=\; F(t) - D(t, v, h) - M(t)g \tag{11}$$

The parameters F(t) is the thrust at time t, M(t) is the mass, and D(t, v, h) is the drag on the rocket at time ***t*** when travelling at velocity ***v*** at height ***h***. Therefore, the instantaneous acceleration is given as:

$$a(t) = \frac{F(t)-D(t,v,h)-M(t)g}{M(t)} \quad (12)$$

To compute the acceleration at time step k, we simply evaluate the numerical value of each of the terms on the right-hand side of equation 12 at each step of the iteration of the loop. In the previous code's segments, we need a more precise GetAcceleration() function that will use three parameters namely, time, height and velocity. The plotting of the trajectory will be made possible using the plot function in Matlab. The modified code segment is illustrated below:

```
% Initial conditions
v = 0;
h = 0;
% Times and seasons
tstart = 0;
dt = 0.1;
tstop = 300;
k = 0;                                  % Clock tick
% loop through all times
for t = tstart :dt: tstop
a = GetAcceleration (t, h, v);          % get acceleration
% capture telemetry for plotting
k=k+1; A(k)=a; V(k)=v; H(k)=h; T(k)=t;
% update velocity and height
h = h + dt * v;
v = v + dt * a;
end

% graphics
% height Graph
subplot(2,2,1) %Top Graph
plot(T,H,'b')
title ('Height')
xlabel('t')
ylabel('h')
grid

% Velocity Graph
subplot(2,2,2) % Bottom Graph
plot(T,V,'r')
title ('Velocity')
xlabel('t')
ylabel('v')
grid

% Acceleration Graph
subplot(2,2,3) % Bottom Graph
plot(T,A,'r')
```

```
title ('Acceleration')
xlabel('t')
ylabel('a')
grid
```

5.0 The acceleration function

The GetAcceleration function is a global function, it has the capability of calling other local functions. The function can be used to compute the instantaneous Mass, Thrust and Drag. In addition, the function returns the instantaneous acceleration [14], [15], [16]. The code segment below shows the description of the GetAcceleration function.

```
function a = GetAcceleration (t, h, v);
g = GetGravity (h);
f = GetThrust (t);
m = GetMass (t);
d = GetDrag (t,h,v);
a = (f - m*g -d)/m;                    % Newton Second Law

function thrust = GetThrust (t)
burnTime = 150;
if(t >0 & t< burnTime )
thrust = 34e6;
else
thrust = 0;
end

function mass = GetMass (t)
initialMass = 2.9e6; % kg
initialFuelMass = 2.15e6; % kg
burnTime = 150;  % seconds
burnRate =  0.02; % kg/s

if ( t <=0 )
mass = 2.9e6;
elseif (t >0 & t< burnTime)
mass = 2.9e6 - 2.15e6;
else
mass = 0;

function dragcoeff = GetDragCoeff (t,h)
% Assume a constant drag coefficient
% (but we could make it a function of height etc)
dragcoeff = 0.6;
```

```
function drag = GetDrag (t, h, v)
% calculate a quadratic drag
% NB! When v is +ve (up) drag should be +ve (down)
k = GetDragCoeff (t, h);
drag = k * 0.6;

function g = GetGravity ( height )
% ignore variation with height !
g = 9.81;
```

An important function in the above code segment is the GetDrag() function. This function implements a very simple drag model. Herein, the resistance is proportional to the square of the velocity. The constant of proportionality is called a drag coefficient and this is set to a value of 0.6. The constant of proportionality can be customized as a function of height, this means that as the air thins the resistance drops.

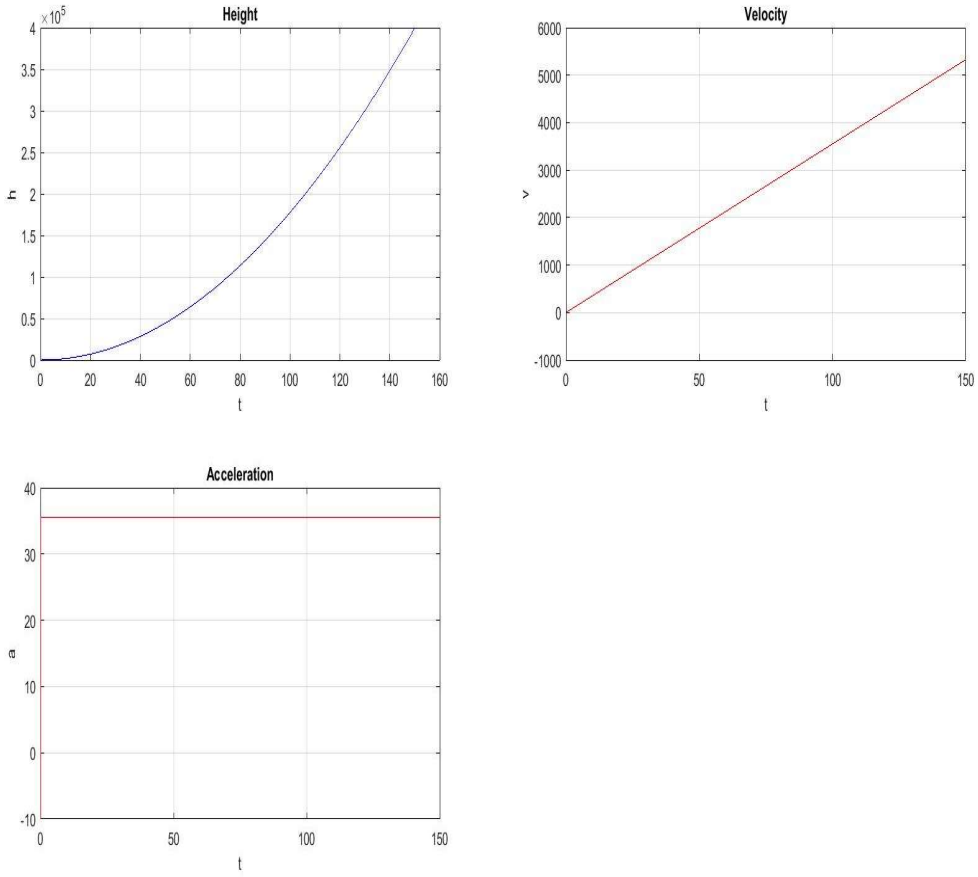
6.0 Program Testing

This section presents the MATLAB program testing. The codes should be tested to ensure that it performs as expected. The students can modify the codes to change the functionality of the Matlab codes as you run on. When you have composed your own modify function in Matlab, check that it executes as required. When all your code appears to run as expected, re-test it by changing input parameters, finding out that the output changes as required. You can try out with the codes by changing different parameters and getting different results. The files coming with the SATURN V Rocket launch are listed beneath:

1. RocketLaunch.m, RocketLaunch.eps, RocketLaunch.asv
2. GetAcceleration.m
3. GetDram.m, GetDram.asv
4. GetDragCoeff.m
5. GetGravity.m
6. GetMass.m, GetMass.asv
7. GetThrust.m

6.1 Program Output

The outputs of the program are the rocket height, velocity and acceleration. This is provided in the program's output files, namely, Output_Program.tif, Output_Program.jpg. Fig. 5 shows the generated MATLAB program's output.



7. Conclusion

This classroom activity demonstrates a computational problem solving. The approach presented in the activity to guide the undergraduate engineering students to understand problem solving techniques and computational problem-solving using MATLAB. This includes using a coherent and structured approach to resolving problems. In computational problem solving, several approaches are combined that will lead to the final solution. Nevertheless, problems must be approached methodically, applying algorithms or step by step procedure by which one arrives at a result. In this classroom activity, the procedure on how to develop a model to determine the time of flight of rocket missile launch from the surface of the earth and distance travelled when the rocket returns to the ground was demonstrated. To explain the problem in a well-defined and precise way, the student should modify and write MATLAB codes to compute the range that a rocket missile would travel when it is launched with an initial velocity of 20 m/s at an initial angle of θ. Compute this range for all angles between 0° and 90° in steps of 1° degree. Determine the angle that will result in the maximum range of the rocket. Plot the trajectory of the rocket missile for angles between 10° and 90° in 1° increments. Plot the maximum-range trajectory in a different color and with a thicker line. Assume that the atmospheric air has no effect on the trajectory of flight. Furthermore, the flight of a rocket launch can be confronted with real world situations. The Saturn V rocket used in the Apollo 11 spacecraft was employed to provide better understanding of a real-life scenario. The methodology makes use of the second order differential equation to model

the Saturn V rocket launch. The rocket can be modelled using MATLAB codes including its position, time derivatives, velocity and acceleration. The classroom activity will assist the undergraduate engineering students to better understand the computational problem-solving using MATLAB.

References:

[1] Luís M. B. C. Campos and Paulo J. S. Gil, On Four New Methods of Analytical Calculation of Rocket Trajectories, Aerospace J. **2018**, 5, 88, pp. 1-32.

[2] George Buchanan et al., The Development of Rocketry Capability in New Zealand—World Record Rocket and First of Its Kind Rocketry Course, *Aerospace J.* **2015**, *2, pp. 92-117.*

[3] Hoani Bryson et al., Vertical Wind Tunnel for Prediction of Rocket Flight Dynamics, Aerospace J. **2016**, 3, 10, pp. 1-27.

[4] Stamminger, A. STERN—A rocket programme for German students. In Proceedings of the 21st ESA Symposium on European Rocket and Balloon Programmes and Related Research, Thun, Switzerland, 9–13 June 2013.

[5] Rojas, J.I.; Prats, X.; Montiaur, A.; Garcia-Berro, E. Model rocket workshop: A problem-based learning experience for engineering students. *Int. J. Emerg. Technol. Learn.* **2008**, *3*, 70–77.

[6] Heath, M.T.; Dick, W.A. Virtual rocketry: Rocket science meets computer science. *Computat. Sci. Eng. IEEE* **1998**, *5*, 16–26.

[7] Cannon, R.L. Model rocketry as a teaching aid in science. *Sch. Sci. Math* **2010**, *74*, 471–475.

[8] Jayaram, S.; Boyer, L.; George, J.; Ravindra, K.; Mitchell, K. Project-based introduction to aerospace engineering course: A model rocket. *Acta Astronaut.* **2010**, *66*, 1525–1533.

[9] NASA. *Rockets Educator Guide*; *EG-2011–11–223-KSC*; NASA: Washington, DC, USA, 2011. Available at: https://www.nasa.gov/audience/foreducators/topnav/materials/listbytype/Rockets.html.

[10] Box, S.; Bishop, C.M.; Hunt, H. Stochastic six-degree-of-freedom flight simulator for passively controlled high-power rockets. *J. Aerosp. Eng.* **2011**, *24*, 31–45.

[11] Open source model rocket simulator. Available online: http://openrocket.sourceforge.net/ download.html (accessed on 21 February 2015).

[12] Stephen J. Chapman, MATLAB® Programming for Engineers, Fourth Edition, Thomson Learning, Thomson Corporation, Canada, 2008, pp. 206-216.

[13] Stormy Attaway, *MATLAB®: A practical Introduction to Programming and Problem Solving*, College of Engineering, Boston University, MA, Butterworth-Heinemann, Elsevier Inc., Third Edition, **2013**, pp. 14-70.

[14] Eric Peasley, 3 DBT Part A: Simulating a Rocket Launch, pp. 37-48, available: https://www.robots.ox.ac.uk/~dwm/Courses/1P5_2011/1P5notes3.pdf.

[15] Eric Peasley, Introduction to the P5 Computing Laboratory, Department of Engineering Science, Oxford University, Britain, pp. 1-67, available at: http://www2.eng.ox.ac.uk/~labejp/Courses/1P5/Lecture1Slides.pdf

[16] Eric Peasley et al. P5 Computing Laboratory and DBT 5 Sessions in Michaelmas, Hilary, and Trinity, 2018-2019, pp. 107-127, available at: www.eng.ox.ac.uk/~labejp/Courses/1P5.