## Computing and Numerical Methods Part II Report:

#### Using C++ to code a simulation, and provide a convergence study of a Vessel System’s Dynamics

**Q6(a)**

Table 1: Case Parameters

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Case** | **System Parameters** | **Vessel Parameters** | | | | | |
|  |  |  |  |  |  |
| Single Vessel, no drag |  | 1.0 | 0.1 | 0.0 | -0.1 | 0.0 | 0.0 |
| Single Vessel, low drag |  | 5.0 | 0.5 | 0.0 | 0.1 | -0.5 | 0.0 |
| Sinking Vessel |  | 100.0 | 0.1 | 1000.0 | 0.0 | 0.0 | 0.0 |
| Three Vessel |  | 1  10.0  50.0 | 0.1  0.5  10.0 | 20  10.0  1.0 | -0.1  -0.1  -0.1 | 0.0  0.0  0.0 | 0.0  0.0  0.0 |

1. Identify and state the maximum time step for a converged solution for FE and RK4.

Running test cases for the first test case (Single Vessel, no drag).

The baseline solution is defined as the solution given by the numerical method using a sensible base time step, as stated below.

The maximum time step is defined as the largest time step falling within the 10% deviation of the baseline solution.

Table 2: Simulation Results and Convergence Tests

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Case** | **Method** | **Vessel** | **Max Step Size**  **(m)** | **Final Displacement**  **(m)** | **Final Velocity**  **(m/s)** | **Final Water Levels**  **(m)** |
| Single Vessel, no drag | RK4 | 1 |  | -0.099986 | -0.317409 | 0.000000 |
| FE | 1 |  | -0.099552 | -0.316048 | 0.000000 |
| Single Vessel, low drag | RK4 | 1 |  | 0.006299 | -0.236070 | 0.000000 |
| FE | 1 |  | 0.006300 | -0.235292 | 0.000000 |
| Sinking Vessel | RK4 | 1 |  | 15.933335 | 15.738578 | 14.439672 |
| FE | 1 |  | 15.933921 | 15.738459 | 14.440190 |
| Three Vessel | RK4 | 1 |  | 0.144690 | -0.033585 | 0.063547 |
| 2 |  | 0.059553 | -0.179188 | 0.004433 |
| 3 |  | 0.010222 | 0.038018 | 0.000005 |
| FE | 1 |  | 0.144497 | -0.033789 | 0.063540 |
| 2 |  | 0.059154 | -0.177866 | 0.004433 |
| 3 |  | 0.010012 | 0.035148 | 0.000005 |

|  |  |
| --- | --- |
| Figure 1: Case 1, RK4 | Figure 2: Case 1, FE |
| Figure 3: Case 2, RK4 | Figure 4: Case 2, FE |
| Figure 5: Case 3, RK4 | Figure 6: Case 3, FE |
| Figure 7: Case 4, RK4 | Figure 8: Case 4, FE |

**Q6(b)**

Table 3: Baseline Parameters for Parameter testing

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Base Case** | **Method** | **System**  **Parameters** | **Vessel Parameters** | | | | | |
|  |  |  |  |  |  |
| Single Vessel, no drag | RK4 |  | 1.0 | 0.1 | 0.0 | -0.1 | 0.0 | 0.0 |

|  |  |
| --- | --- |
| Figure 9: Study of Varying Mass | Figure 10: Study of Varying Area |
| Figure 11: Study of Varying K | Figure 12: Study of Varying Initial Height |
| Figure 13: Study of Varying Initial Displacement | Figure 14: Study of Varying Initial Velocity |

From the graphs above it is clear that the Initial velocity and the initial displacement do not contribute to the frequency response of the system as shown in Figure 13 and Figure 14. In Figure 13 the valley when initial displacement is zero is because if there is no initial perturbation the vessel will not undergo motion, so it is a discontinuity in the graph.

It is however clear that the initial height, K, Area, and Mass affect the dynamic response of the system.

**Q6(c)**

C++ is a low-level language so there is a large benefit to its utility in solving computationally heavy numerical problems.

One advantage is that it is a compiled language so it is directly converted to machine code before executing and so it is faster to execute, which has its benefits for implementing solvers such as those found within FEA and CFD software, which are computationally expensive numerical problems.

There is a large assortment of libraries available compared to other low-level languages such as TensorFlow which allows C++ access to the most advanced machine learning libraries which it can utilize to provide the backbone for Machine learning software, especially applying it to new technologies.

**Q6(d)**

I used the STL library in the form of Vectors. Vectors are a useful data type as they act as dynamically sized arrays and can be used to easily stop data when passing into functions, one use in my program was defining force vectors to output into my Main RK4 code where I could cleanly and efficiently pass outputs from my functions.

I used OOP paradigms to create classes that were used alongside inheritance to define a set of public parameters that could easily be accessed by the derived classes. This can be seen in my code through my simulation class which has 2 derived classes for each numerical solver used RK4 and FE these derived classes were able to utilize and access functions from the public scope of the simulation class as opposed to having to pass the parameters into a function each iteration which saves computational power and reduces the number of definitions of the same parameter within the code, contributing to a more efficient and readable code structure.

**Appendix**

%%

clc;

clear;

close all;

format long

set(groot,'defaultLineLineWidth',2) %sets graph line width as 2

set(groot,'defaultAxesFontSize',24) %sets graph axes font size as 18

set(groot,'defaulttextfontsize',24) %sets graph text font size as 18

set(groot,'defaultLineMarkerSize',8) %sets line marker size as 8

set(groot,'defaultAxesXGrid','on') %sets X axis grid on

set(groot,'defaultAxesYGrid','on') %sets Y axis grid on

set(groot,'DefaultAxesBox', 'on') %sets Axes boxes on

picturewidth = 20; % set this parameter and keep it forever

hw\_ratio = 0.65; % feel free to play with this ratio

%%

data = load("E:\Y2\computing CW\Final CPP CW\Output\_4.txt");

%%

% Extract time and data for vessels

% Assuming each line has: t y1 v1 h1 y2 v2 h2 ...

time = data(:, 1); % Time column

num\_vessels = (size(data, 2) - 1) / 3; % Determine the number of vessels

% Initialize storage for vessel data

positions = zeros(length(time), num\_vessels);

velocities = zeros(length(time), num\_vessels);

water\_levels = zeros(length(time), num\_vessels);

% Extract positions, velocities, and water levels for each vessel

for i = 1:num\_vessels

positions(:, i) = data(:, 3 \* i - 1);

velocities(:, i) = data(:, 3 \* i);

water\_levels(:, i) = data(:, 3 \* i + 1);

end

% Part (a): Plot displacement of vessel base (y) over time for each vessel

% plot Analytical solution

Fig\_name\_1 = "Displacement\_4\_FE";

Line\_Style = [":","-.","-"];

Fig\_name = figure;

%set(Analytical\_solution,"WindowState","maximized");

set(findall(Fig\_name,'-property','FontSize'),'FontSize',24);

set(findall(Fig\_name,'-property','Interpreter'),'Interpreter','latex')

set(findall(Fig\_name,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(Fig\_name,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(Fig\_name,'Position');

set(Fig\_name,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

for i = 1:num\_vessels

plot(time, positions(:, i),'DisplayName', ['Vessel ' num2str(i)],LineStyle = Line\_Style(i));

end

xlabel('Time (s)');

ylabel('Displacement (m)');

% title('Displacement of Vessel Base Over Time');

grid on;

legend(Location= "southeast");

hold off;

saveas(Fig\_name,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_a'+Fig\_name\_1+'.svg');

%%

% %% Plot Velocities

%

% figure;

% hold on;

% for i = 1:num\_vessels

% plot(time, velocities(:, i), 'DisplayName', ['Vessel ' num2str(i)]);

% end

% xlabel('Time (s)');

% ylabel('Velocity (m/s)');

% title('Velocity of Vessel Base Over Time');

% grid on;

% legend;

% hold off;

%

% %% Plot Water Level

%

% figure;

%

%

% hold on;

% for i = 1:num\_vessels

% plot(time, water\_levels(:, i), 'DisplayName', ['Vessel ' num2str(i)]);

% end

% xlabel('Time (s)');

% ylabel('Water Level (m)');

% title('Water Level of Vessel Base Over Time');

% grid on;

% legend;

% % hold off;

%% Part (b): Maximum timestep for convergence

% For simplicity, this part would involve analyzing output for different dt

% and determining convergence. Here, it is assumed convergence has been

% ensured via proper selection of dt in C++.

% Table for final values

final\_displacements = positions(end, :);

final\_velocities = velocities(end, :);

final\_water\_levels = water\_levels(end, :);

%Display the results in MATLAB's command window

disp('Final values for each vessel:');

disp('Vessel | Final Displacement (m) | Final Velocity (m/s) | Final Water Level (m)');

for i = 1:num\_vessels

fprintf('%6d | %22.6f | %20.6f | %20.6f\n', i, final\_displacements(i), final\_velocities(i), final\_water\_levels(i));

end

%% Part B (ii)

%mass graph

m1 = [0.0005, 0.001, 0.002, 0.003, 0.004, 0.005, 0.05:0.05:1, 1.25, 1.50, 1.75 ,2.00 ];

data\_m = load("E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\m\m1.txt");

m1\_freq = calculateAverageFrequency(data\_m);

mvsfreq = figure;

set(findall(mvsfreq,'-property','FontSize'),'FontSize',24);

set(findall(mvsfreq,'-property','Interpreter'),'Interpreter','latex')

set(findall(mvsfreq,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(mvsfreq,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(mvsfreq,'Position');

set(mvsfreq,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

plot(m1,m1\_freq,"k-x")

xlabel('Mass (kg)');

ylabel('Frequency (Hz)');

grid on;

% legend('Location','best');

hold off;

saveas(mvsfreq,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\m\mvsfreq.svg');

%% A vs

A1 = [0.1, 0.5, 1:1:10, 15, 20:10:70];

data\_A = load("E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\A\A1.txt");

A1\_freq = calculateAverageFrequency(data\_A);

Avsfreq = figure;

set(findall(Avsfreq,'-property','FontSize'),'FontSize',24);

set(findall(Avsfreq,'-property','Interpreter'),'Interpreter','latex')

set(findall(Avsfreq,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(Avsfreq,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(Avsfreq,'Position');

set(Avsfreq,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

plot(A1,A1\_freq,"r-x")

xlabel('Area (m^2)');

ylabel('Frequency (Hz)');

grid on;

% legend('Location','best');

hold off;

saveas(Avsfreq,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\A\Avsfreq.svg');

%% K vs

K1 = [0:5:35,40:20:400];

data\_K = load("E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\K\K1.txt");

K1\_freq = calculateAverageFrequency(data\_K);

Kvsfreq = figure;

set(findall(Kvsfreq,'-property','FontSize'),'FontSize',24);

set(findall(Kvsfreq,'-property','Interpreter'),'Interpreter','latex')

set(findall(Kvsfreq,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(Kvsfreq,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(Kvsfreq,'Position');

set(Kvsfreq,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

plot(K1,K1\_freq,"b-x")

xlabel('K');

ylabel('Frequency (Hz)');

grid on;

% legend('Location','best');

hold off;

saveas(Kvsfreq,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\K\Kvsfreq.svg');

%% H\_0 vs

h1 = [0:0.01:0.04, 0.05:0.02:0.21, 0.22:0.01:0.3 ];

data\_h = load("E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\h\_0\h\_0.txt");

h1\_freq = calculateAverageFrequency(data\_h);

hvsfreq = figure;

set(findall(hvsfreq,'-property','FontSize'),'FontSize',24);

set(findall(hvsfreq,'-property','Interpreter'),'Interpreter','latex')

set(findall(hvsfreq,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(hvsfreq,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(hvsfreq,'Position');

set(hvsfreq,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

plot(h1,h1\_freq,'g-x')

xlabel('h\_0 (m)');

ylabel('Frequency (Hz)');

grid on;

% legend('Location','best');

hold off;

saveas(hvsfreq,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\h\_0\h\_0vsfreq.svg');

%% y\_0 vs

y\_01 = -1:0.2:1;

data\_y\_0 = load("E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\y\_0\y\_0.txt");

y\_01\_freq = calculateAverageFrequency(data\_y\_0);

y\_0vsfreq = figure;

set(findall(y\_0vsfreq,'-property','FontSize'),'FontSize',24);

set(findall(y\_0vsfreq,'-property','Interpreter'),'Interpreter','latex')

set(findall(y\_0vsfreq,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(y\_0vsfreq,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(y\_0vsfreq,'Position');

set(y\_0vsfreq,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

plot(y\_01,y\_01\_freq,Marker= "x", Color="#D95319")

xlabel('y\_0 (m)');

ylabel('Frequency (Hz)');

grid on;

% legend('Location','best');

hold off;

saveas(y\_0vsfreq,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\y\_0\y\_0vsfreq.svg');

%% y\_dot\_0 vs

y\_dot\_01 = -1:0.2:1;

data\_y\_dot\_0 = load("E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\y\_dot\_0\y\_dot\_0.txt");

y\_01\_freq = calculateAverageFrequency(data\_y\_dot\_0);

y\_dot\_0vsfreq = figure;

set(findall(y\_dot\_0vsfreq,'-property','FontSize'),'FontSize',24);

set(findall(y\_dot\_0vsfreq,'-property','Interpreter'),'Interpreter','latex')

set(findall(y\_dot\_0vsfreq,'-property','TickLabelInterpreter'),'TickLabelInterpreter','latex')

set(y\_dot\_0vsfreq,'Units','centimeters','Position',[3 3 picturewidth hw\_ratio\*picturewidth])

pos = get(y\_dot\_0vsfreq,'Position');

set(y\_dot\_0vsfreq,'PaperPositionMode','Auto','PaperUnits','centimeters','PaperSize',[pos(3), pos(4)])

hold on;

plot(y\_dot\_01,y\_01\_freq,marker = "x", Color= "#EDB120")

xlabel('Initial Velocity, v\_0 (m)');

ylabel('Frequency (Hz)');

grid on;

% legend('Location','best');

hold off;

saveas(y\_dot\_0vsfreq,'E:\Y2\computing CW\Final CPP CW\Figures\_Part\_b\y\_dot\_0\y\_dot\_0vsfreq.svg');

%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [dominant\_frequency] = calculateAverageFrequency(data)

% Load the data from the file

% Extract the time column (first column)

time = data(:, 1);

% Determine the number of vessels from the data

num\_vessels = (size(data, 2) - 1) / 3; % Assuming y, v, h per vessel

% Initialize a variable to store the sum of dominant frequencies

dominant\_frequency = zeros(num\_vessels,1);

% Loop through each vessel to calculate its dominant frequency

for i = 1:num\_vessels

% Extract displacement data for the vessel

displacement = data(:, 3 \* i - 1);

% Perform Fourier Transform on the displacement data

Y = fft(displacement);

L = length(time);

% Compute the two-sided spectrum and then the single-sided spectrum

P2 = abs(Y / L);

P1 = P2(1:L/2+1);

P1(2:end-1) = 2 \* P1(2:end-1);

% Define the frequency domain

f = (0:(L/2)) / max(time);

% Find the frequency corresponding to the maximum amplitude

[~, max\_idx] = max(P1);

dominant\_frequency(i) = f(max\_idx);

% fprintf('The frequency of vessel %.1f is %.4f Hz\n',i ,dominant\_frequency);

end

% Calculate the average frequency

% Display the result

end