MEMS 1060/2060 Homework 1

Shane Riley
Undergraduate Researcher, University of Pittsburgh
Swanson School of Engineering
3700 O'Hara Street, Benedum Hall of Engineering
Pittsburgh, PA 15261
Email: shane.riley@pitt.edu

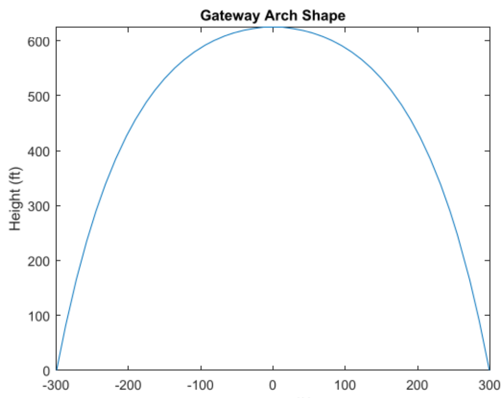

Fig. 1. Gateway Arch.

1 Problem 1

The given equation shown below describes the curve of the Gateway Arch in St. Louis. Plot the curve for $-299.25 < x < 299.25[ft]$*:*

$$y = 693.8 - 68.8cosh(x/99.7)[ft] \tag{1}$$

To create the plot using MATLAB, an anonymous function and a 2 by 1 vector containing the domain bounds are created. Passing both into the **fplot** function creates an autofit plot of the function over the domain. Labels are then added to the graph. See Figure 1.

2 Problem 2

Create a 3by3 determinant function to calculate the determinants of two matrices. The function must use a 2by2 determinant helper function.

The implementation first checks the matrix for proper sizing before evaluation. If the matrix is indeed 3by3, each of the three terms are evaluated separately using a det2by2 helper function. Using the functions defined in the Appendix, the correct determinants are found:

1. $\begin{vmatrix} 1 & 3 & 2 \\ 6 & 5 & 4 \\ 7 & 8 & 9 \end{vmatrix} = -39$
2. $\begin{vmatrix} -2 & 1 & 3 \\ 5 & 2 & -1 \\ 6 & -3 & 4 \end{vmatrix} = -117$

3 Problem 3

Considering the function f(x) and $x = 0.005$*, find the value of the function using a calculator with five significant figures and then with MATLAB format long. Considering the MATLAB value as the true value, calculate the true relative error due to rounding.*

$$f(x) = (e^x - 1)/x \tag{2}$$

To get a calculated value of $f(x)_{calc} = 1.0025$, a TI-84 was used with a function and a table. The MATLAB anonymous function, however, produced a true value of $f_(x)_{true} = 1.002504171880192$, with a computed true relative error of about $4.1615 * 10^{-6}$. The formula for relative error is described, as well as defined in a supporting function:

$$Error = \frac{|f(x)_{calc} - f(x)_{true}|}{|f(x)_{true}|} \tag{3}$$

4 Problem 4

Given the Taylor series for $f(x) = sin(x)$ *centered about* $x = 0$*, calculate* $sin(\pi/4)$ *using two, four, and six terms. Use six digits after the decimal for each term, and compare the results to the true value found using the stock function for sine.*

To calculate the Taylor series in the MATLAB code, an anonymous function is used to generate the value of each term using the input value and the term number. The terms up

Table 1. Taylor series for sine (roundoff at 6 places.)

Num. Terms	$sin(\pi/4)$	Rel. Error
True	0.707106781186547	-
2	0.704652000000000	0.0034716
4	0.707105000000000	2.519e-06
6	0.707105000000000	2.519e-06

to a provided limit are evaluated and rounded independently, and the summed together to yield the estimate for sin. The implementation can be found within the **sinTaylor** function.

The error that comes from taking too few terms of the Taylor series approximation is the truncation error. This type of error decreases as more terms are taken, thus decreasing the relative error. It is worth noting though that when we round off our taylor terms to 6 places after the decimal, the difference between $n = 4$ and $n = 6$ becomes negligible. For this reason, those two values/relative errors are the same.

Appendix: MATLAB Source code

```
%% MEMS 1060/2060 Homework 1
% Author: Shane Riley
% Date: 2/4/2021
format long
%% Problem 1
% Plot the given function for the given span of x

x_min = -299.25;
x_max = 299.25;
y = @(x) 693.8 - 68.8 * cosh(x/99.7);
xspan = [x_min, x_max];

disp(" ");
disp("Problem 1")
fplot(y, xspan);
title("Gateway Arch Shape");
xlabel("Distance (ft)");
ylabel("Height (ft)");


%% Problem 2
% Write a 3x3 determinant function that employs a 2x2 determinant
% subfunction

test_1 = [...
    1 3 2
    6 5 4
    7 8 9];
test_2 = [...
    -2 1 3
    5 2 -1
    6 -3 4];

disp(" ");
disp("Problem 2");
disp("Problem 2 matrix (a): " + det3by3(test_1));
disp("Problem 2 matrix (b): " + det3by3(test_2));

%% Problem 3
% For the given function and x value, use calculator vs format long MATLAB
% to determine relative round-off error

x_3 = 0.005;
f_3 = @(x) (exp(x) - 1)/x;

% Calculator value evaluated using TI-84 using function and table
y_3_calculator = 1.0025;
y_3_matlab = f_3(x_3);

disp(" ");
disp("Problem 3");
disp("True value of function: " + sprintf('%1.15f', y_3_matlab));
disp("Calculator value of function: " + sprintf('%1.15f', y_3_calculator));
disp("Relative error: " + relError(y_3_matlab, y_3_calculator));



%% Problem 4
% Estimate sin(pi/4) using taylor series rounded to 6 decimal places each
% term. Compare to true value (format long evaluation)

x_4 = pi/4;
true_value = sin(x_4);
tries = [2, 4, 6];
roundoff = 6;

disp(" ");
disp("Problem 4");
disp("Taylor series centered around 0; rounded at 6 decimal places for each term");
disp("True value: " + sprintf('%1.15f', true_value))
for n = tries
    disp("First " + n + " terms:");
    value = sinTaylor(x_4, n, roundoff);
    err = relError(true_value, value);
    disp("Value: " + sprintf('%1.15f', value))
```

```
    disp("Relative error vs true: " + err)
end


%% Supporting functions

function d3 = det3by3(A)
% DET3BY3 Local function to evalute 3x3 matrix determinants

    function d2 = det2by2(B)
    % DET2BY2 subfunction
        d2 = (B(1,1)*B(2,2)) - (B(1,2)*B(2,1));
    end

    % Check size
    [x, y] = size(A);
    if (x ~= 3)
    or (y ~= 3)
        throw MException("Not 3x3 matrix");
    end

    % Else
    comp1 = A(1,1) * det2by2(A(2:3,2:3));
    comp2 = A(1,2) * det2by2(A(2:3,1:2:3));
    comp3 = A(1,3) * det2by2(A(2:3,1:2));

    d3 = comp1 - comp2 + comp3;
end

function relative_error = relError(true_value, calc_value)
% RELERROR Local function to evaluate relative error
    absolute_error = abs(calc_value - true_value);
    relative_error = absolute_error / abs(true_value);
end

function value_out = sinTaylor(value_in, n, roundoff)
% SINTAYLOR evaluate n terms of sin taylor series centered at 0
    value_out = 0;
    taylor_term = @(x, term) round((-1)^(term-1) * x^(2*term - 1) / factorial(2*term - 1), roundoff);
    for i = 1:n
        value_out = value_out + taylor_term(value_in, i);
    end

end
```