

MATH 446/OR 481

Project 5

October 30, 2022

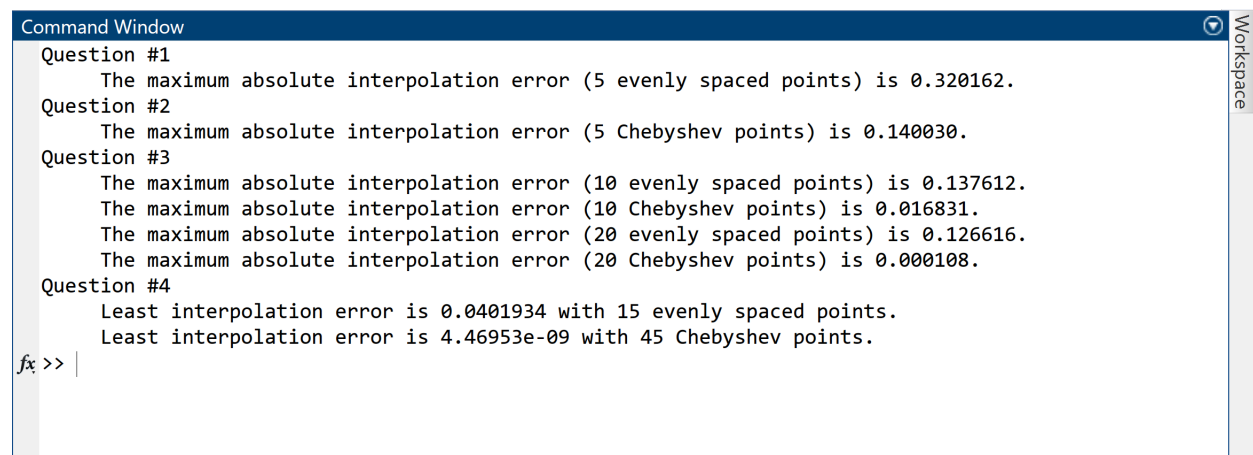
Project Conclusion

A function $f(x) = \cos(\tan(\sin(x)))$ is interpolated on the interval $[0, 4]$ with evenly spaced and Chebyshev points. We see that the errors are decreased when using Chebyshev points compared to an equal number of evenly spaced points.

The least error when using evenly spaced points occurs with 15 points and the error is 0.0401934. The least error when using Chebyshev points occurs with 45 Chebyshev points, and the least error is 4.46953e-09.

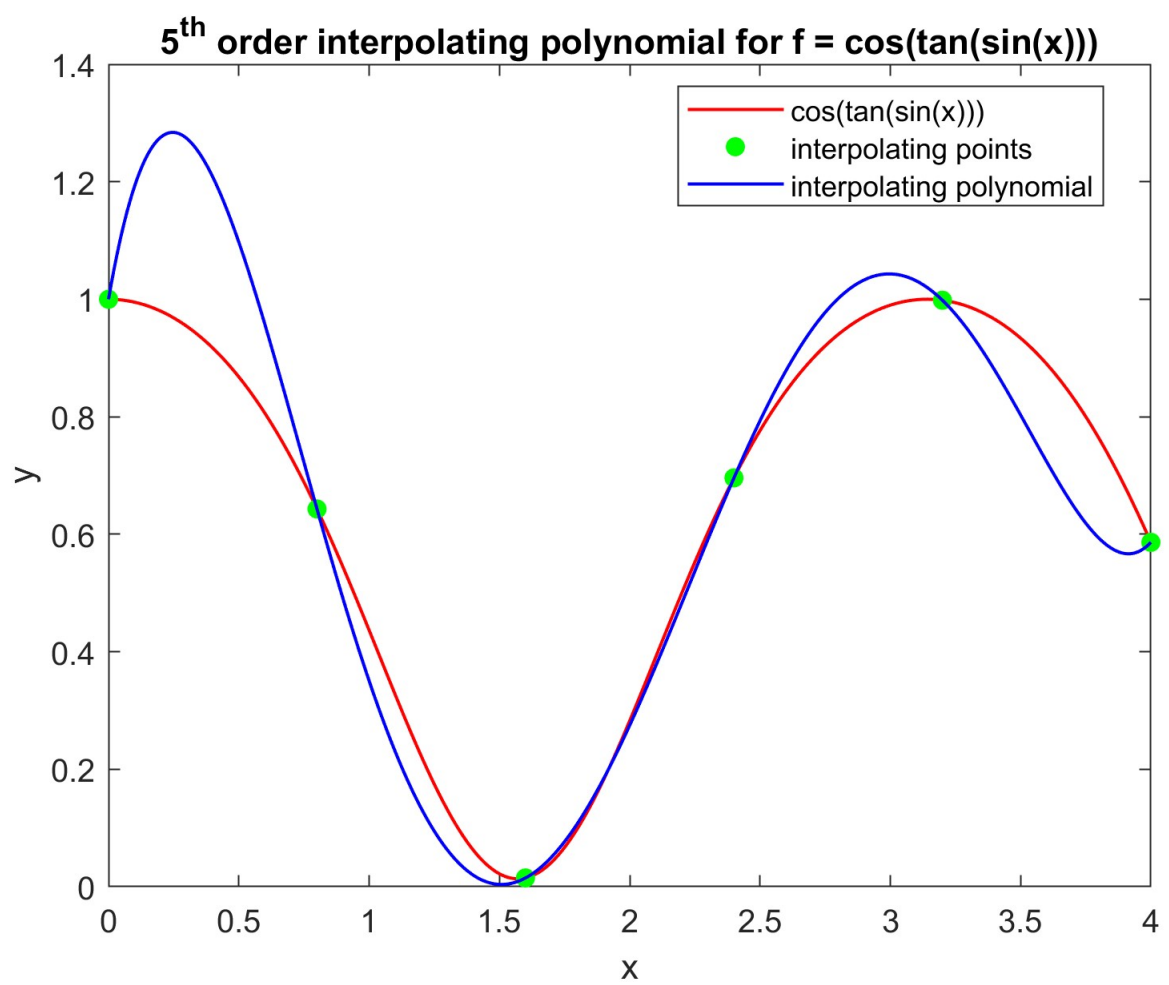
We can see that with enough Chebyshev points, we can achieve near perfect interpolation. However, even with just 10 Chebyshev points the error is 0.016831, which is less than what we get with the best case scenario for evenly spaced points. With 20 Chebyshev points the error is 0.000108, which is very good performance, and hence we need not use 45 points unless really necessary.

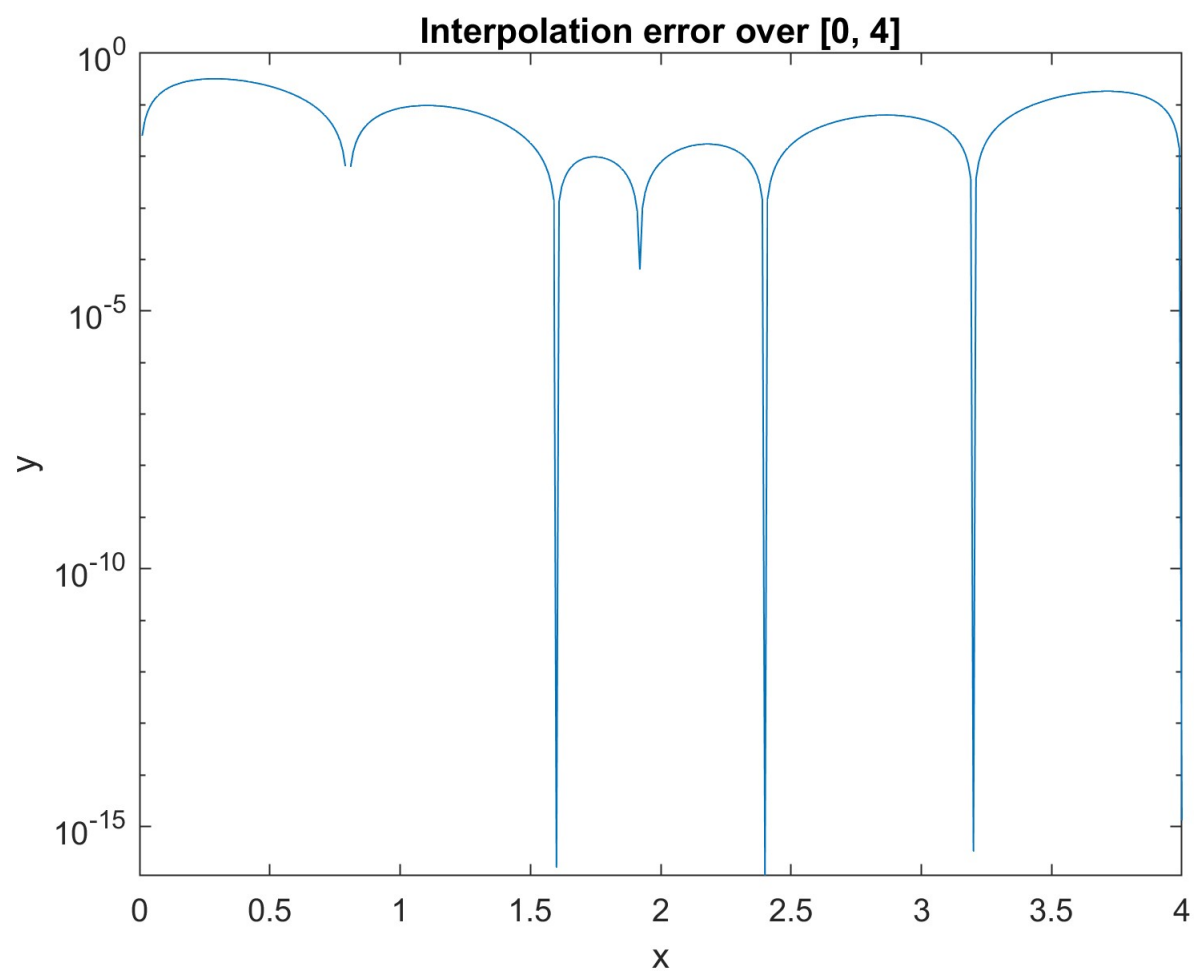
The following MATLAB Command Window Output summarizes the results.

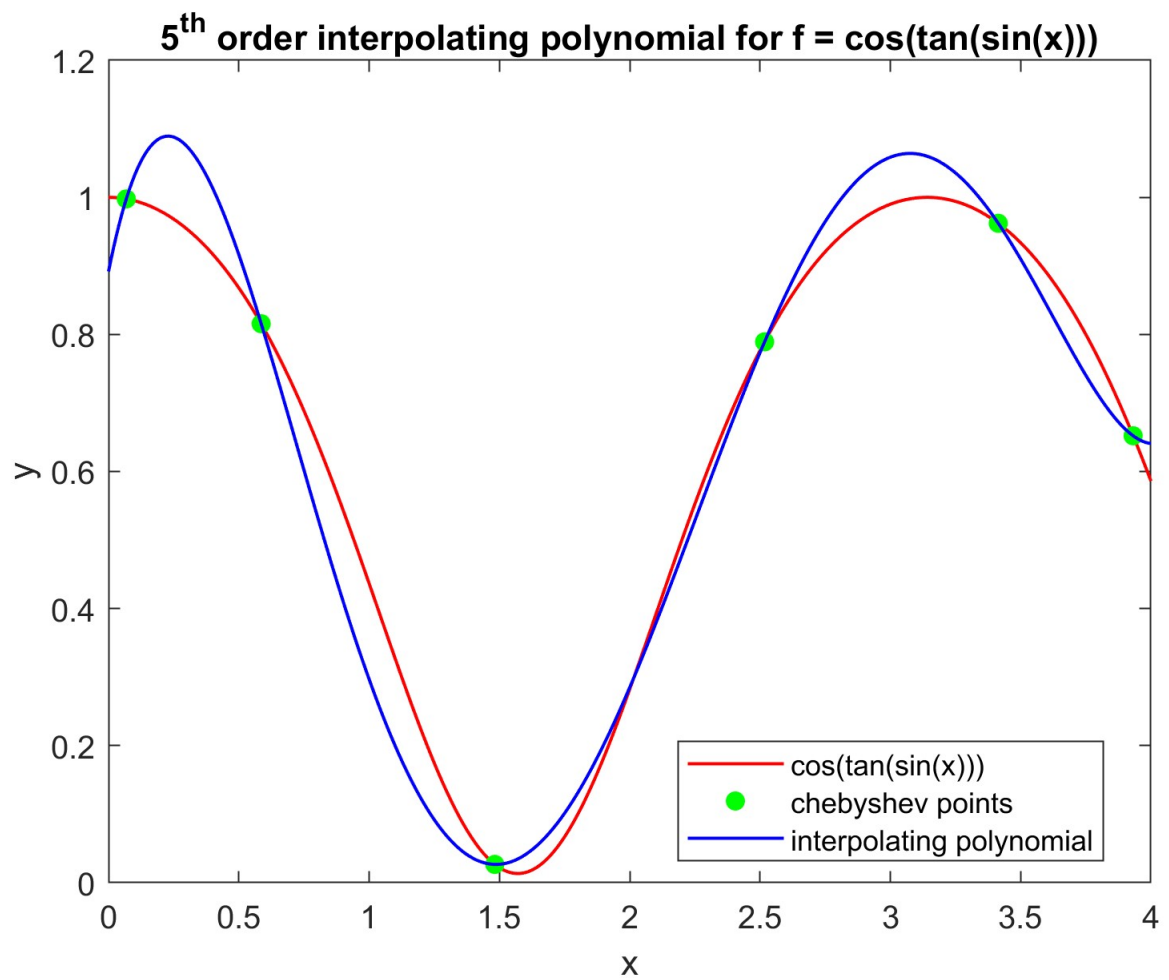
A screenshot of the MATLAB Command Window. The window has a dark blue title bar with the text 'Command Window' and a small icon on the right. The main area is white with black text. The text shows the results of four questions. Question #1: 'The maximum absolute interpolation error (5 evenly spaced points) is 0.320162.' Question #2: 'The maximum absolute interpolation error (5 Chebyshev points) is 0.140030.' Question #3: 'The maximum absolute interpolation error (10 evenly spaced points) is 0.137612. The maximum absolute interpolation error (10 Chebyshev points) is 0.016831. The maximum absolute interpolation error (20 evenly spaced points) is 0.126616. The maximum absolute interpolation error (20 Chebyshev points) is 0.000108.' Question #4: 'Least interpolation error is 0.0401934 with 15 evenly spaced points. Least interpolation error is 4.46953e-09 with 45 Chebyshev points.' At the bottom left, there is a prompt 'fx >> |'. On the right side of the window, there is a vertical grey bar with the word 'Workspace' written vertically.

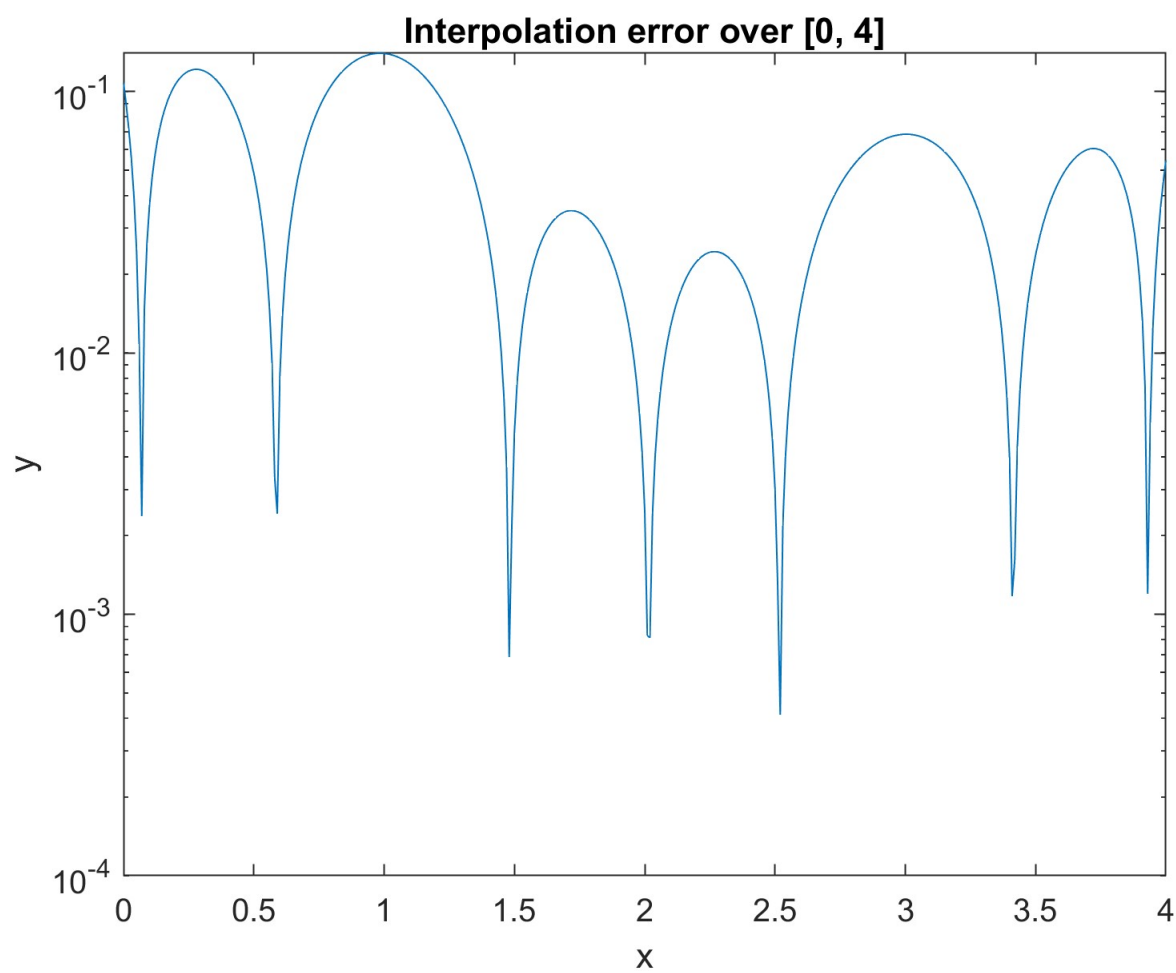
```
Command Window
Question #1
    The maximum absolute interpolation error (5 evenly spaced points) is 0.320162.
Question #2
    The maximum absolute interpolation error (5 Chebyshev points) is 0.140030.
Question #3
    The maximum absolute interpolation error (10 evenly spaced points) is 0.137612.
    The maximum absolute interpolation error (10 Chebyshev points) is 0.016831.
    The maximum absolute interpolation error (20 evenly spaced points) is 0.126616.
    The maximum absolute interpolation error (20 Chebyshev points) is 0.000108.
Question #4
    Least interpolation error is 0.0401934 with 15 evenly spaced points.
    Least interpolation error is 4.46953e-09 with 45 Chebyshev points.
fx >> |
```

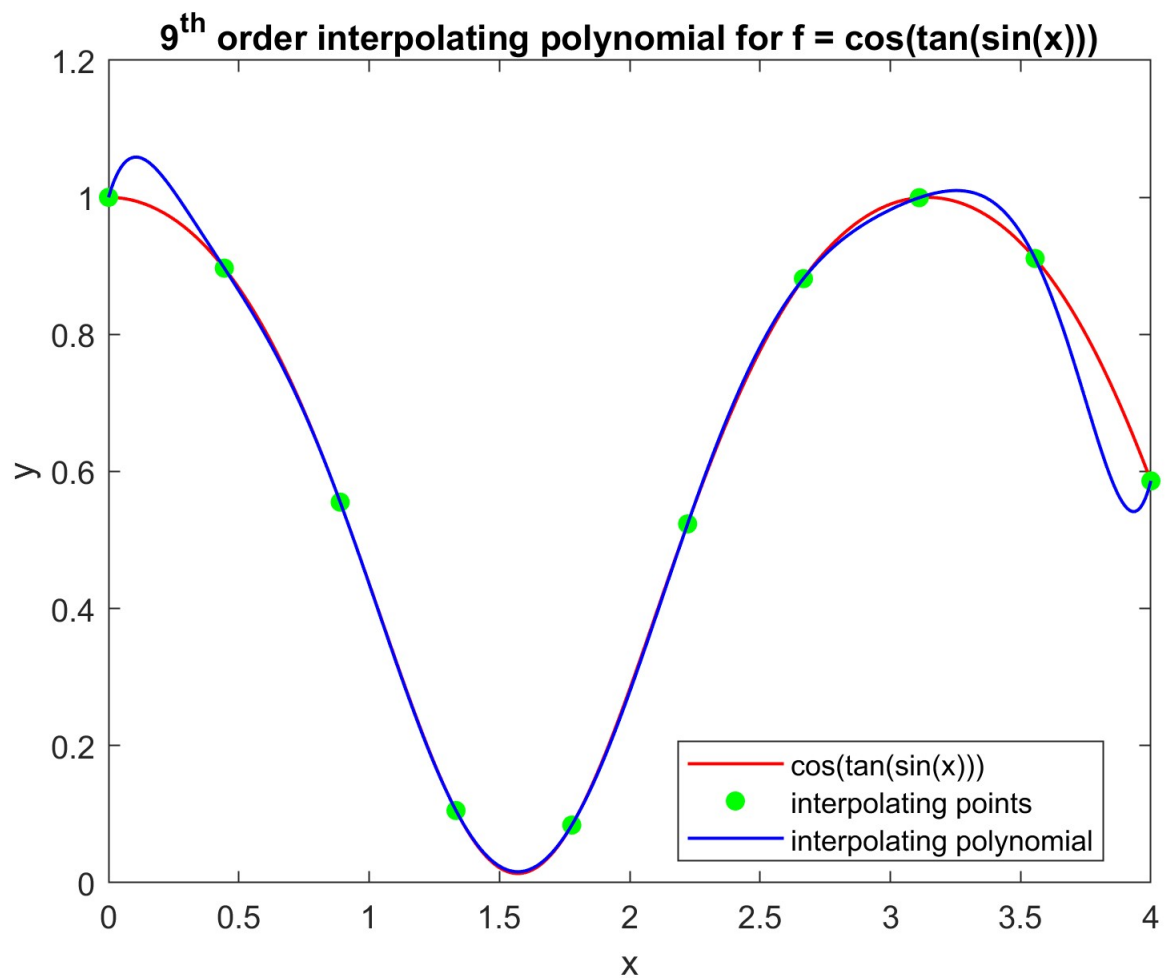
The interpolating polynomial plots and the interpolation errors plots follow.

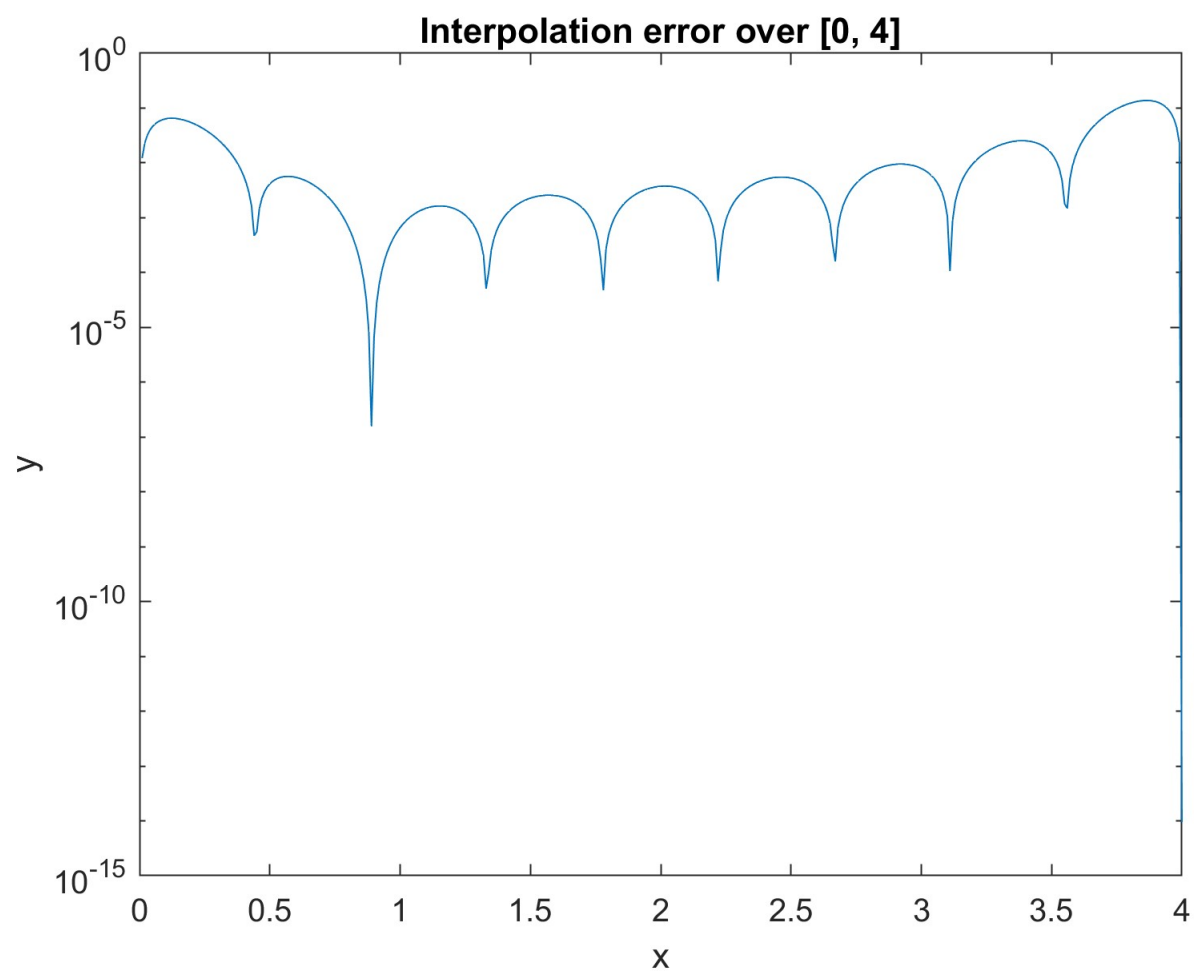


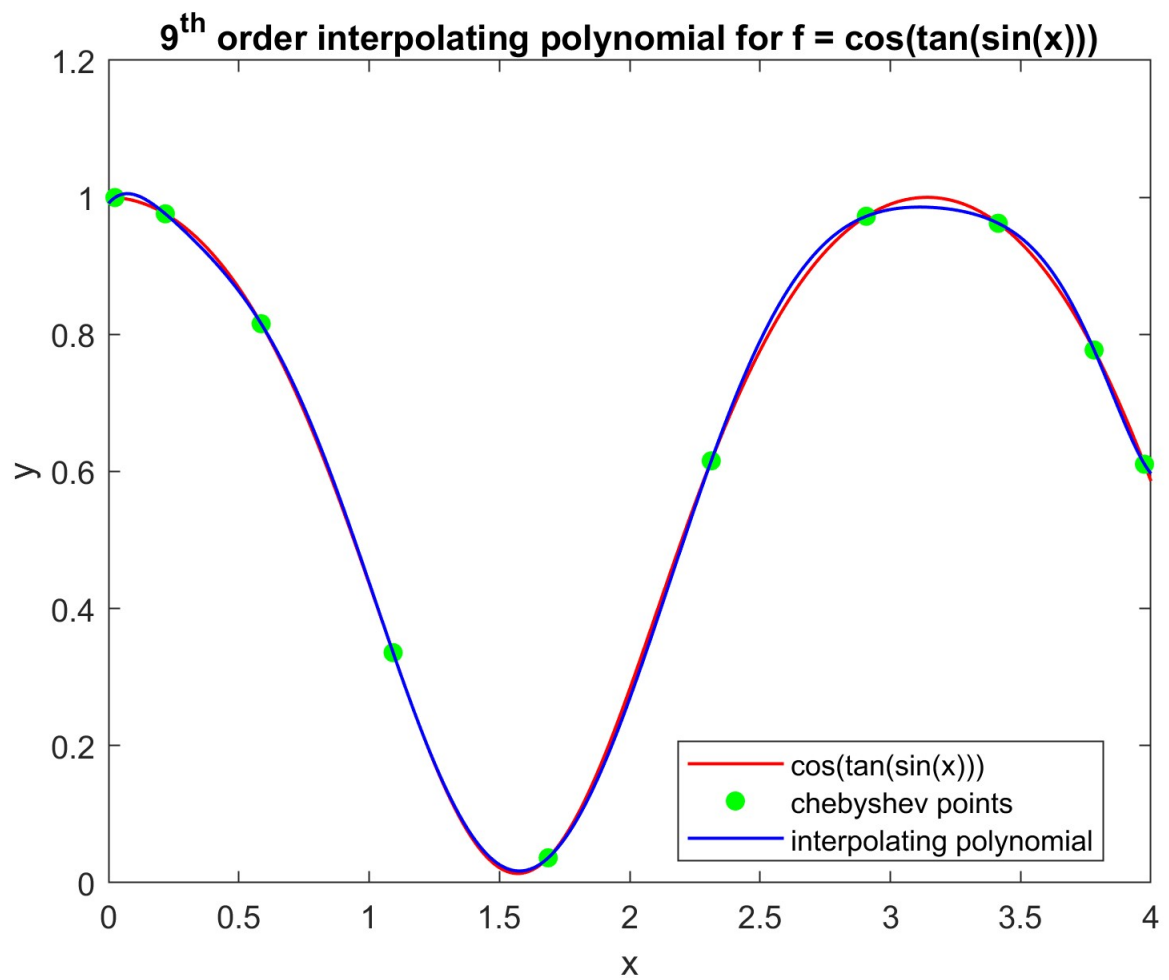


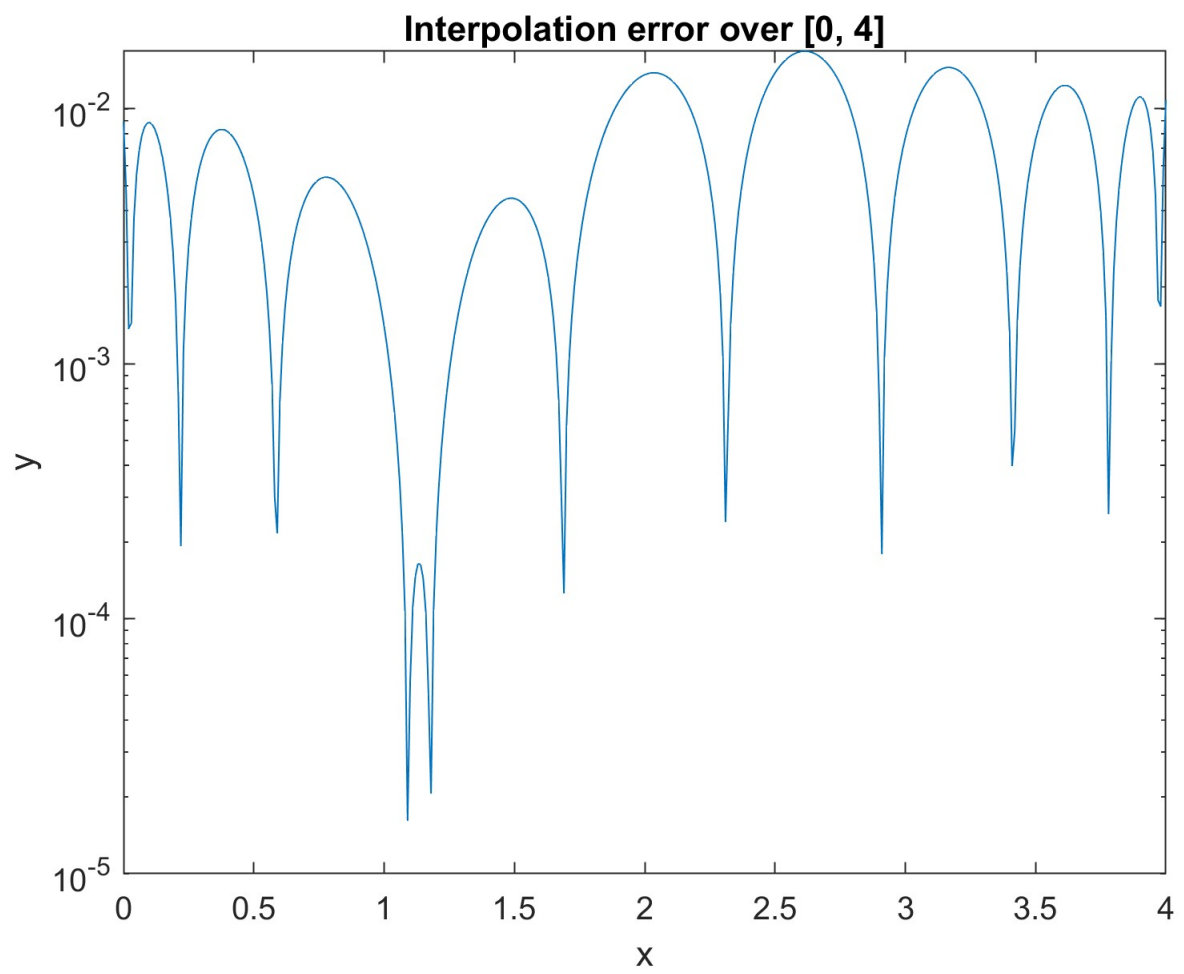


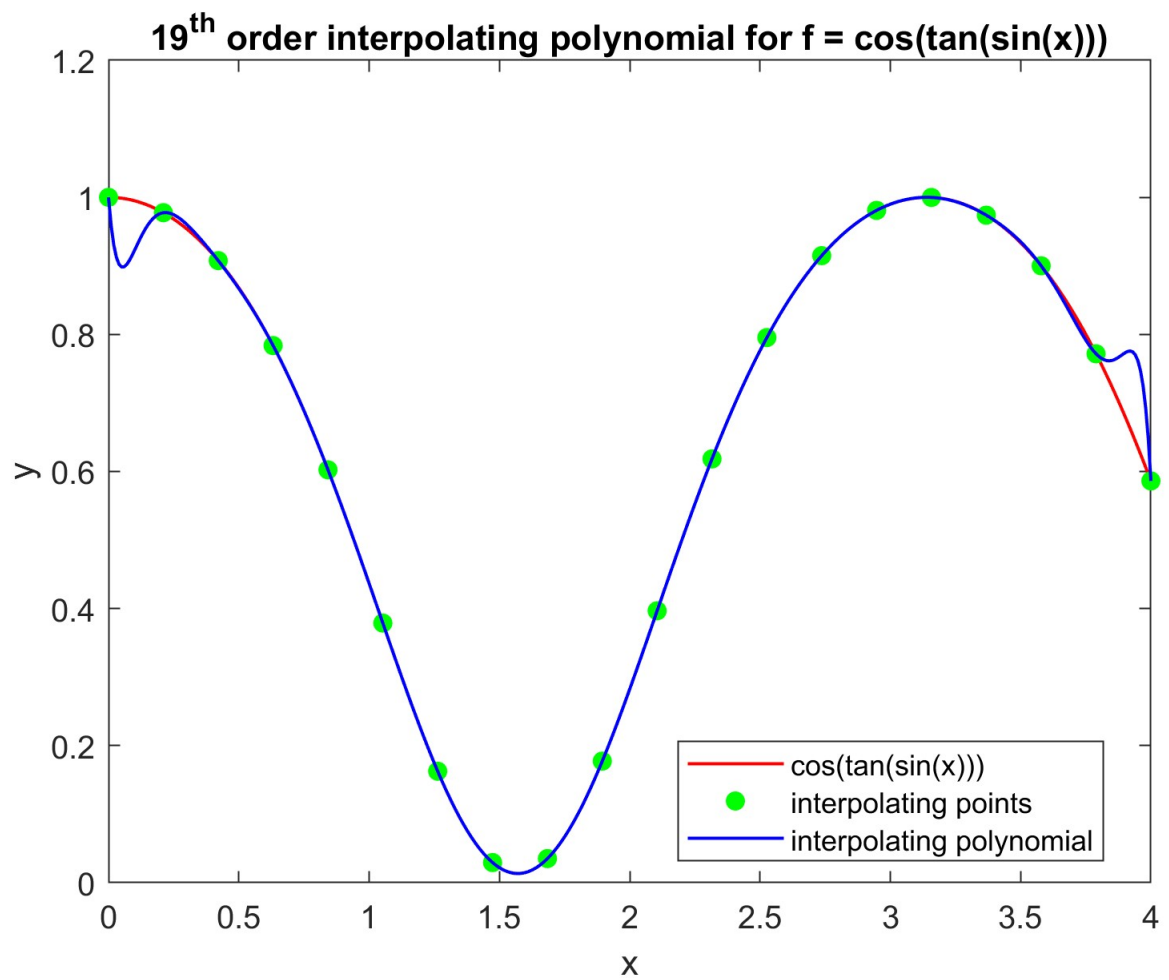


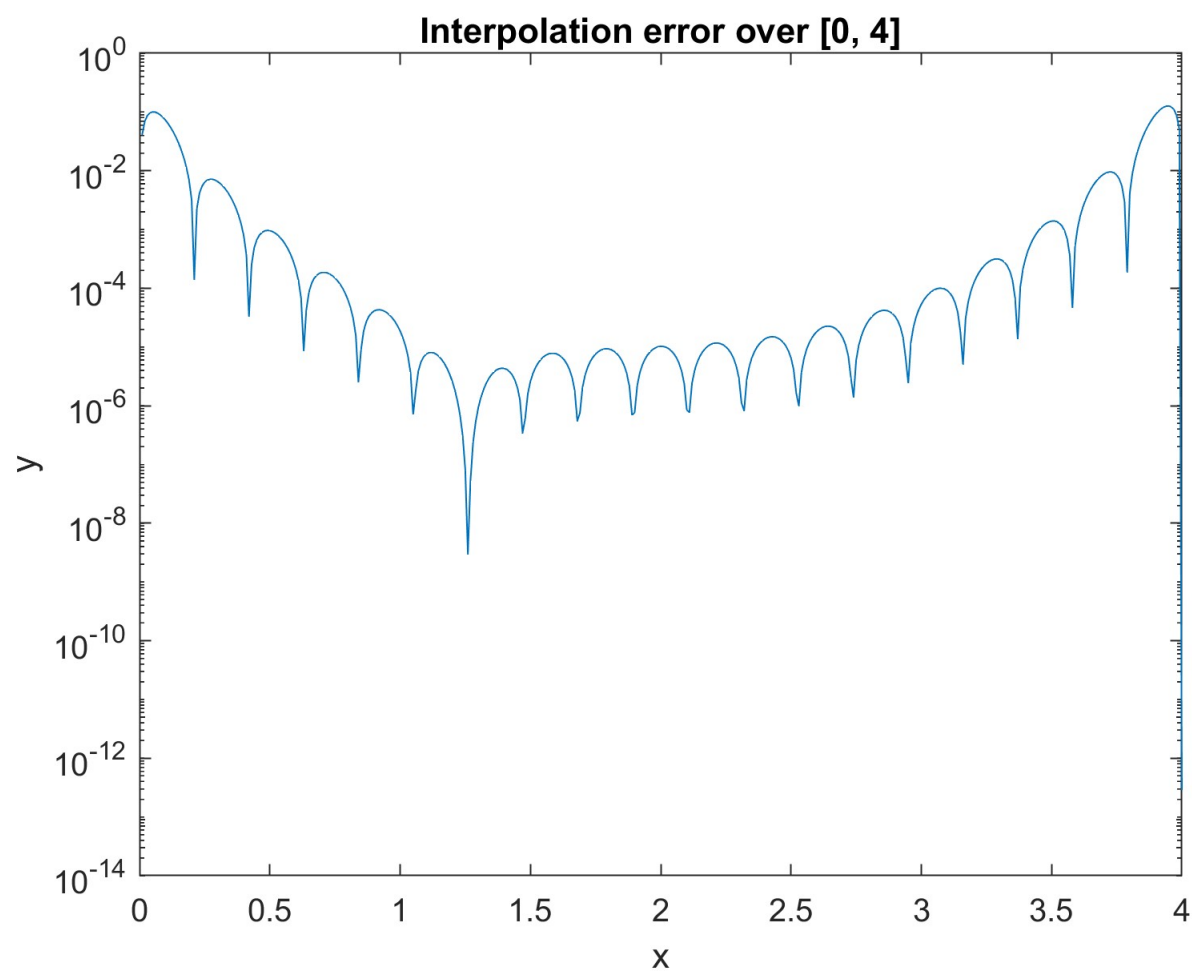


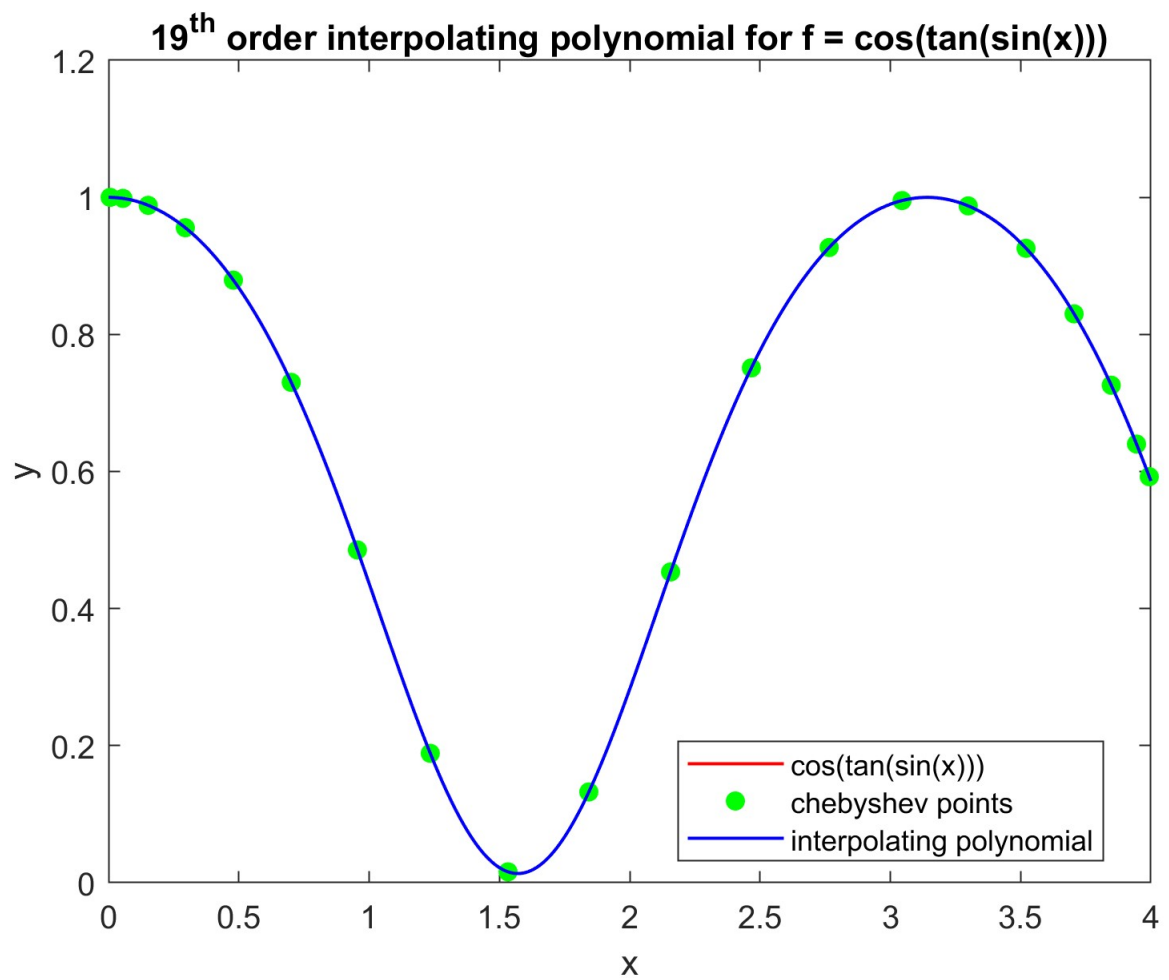


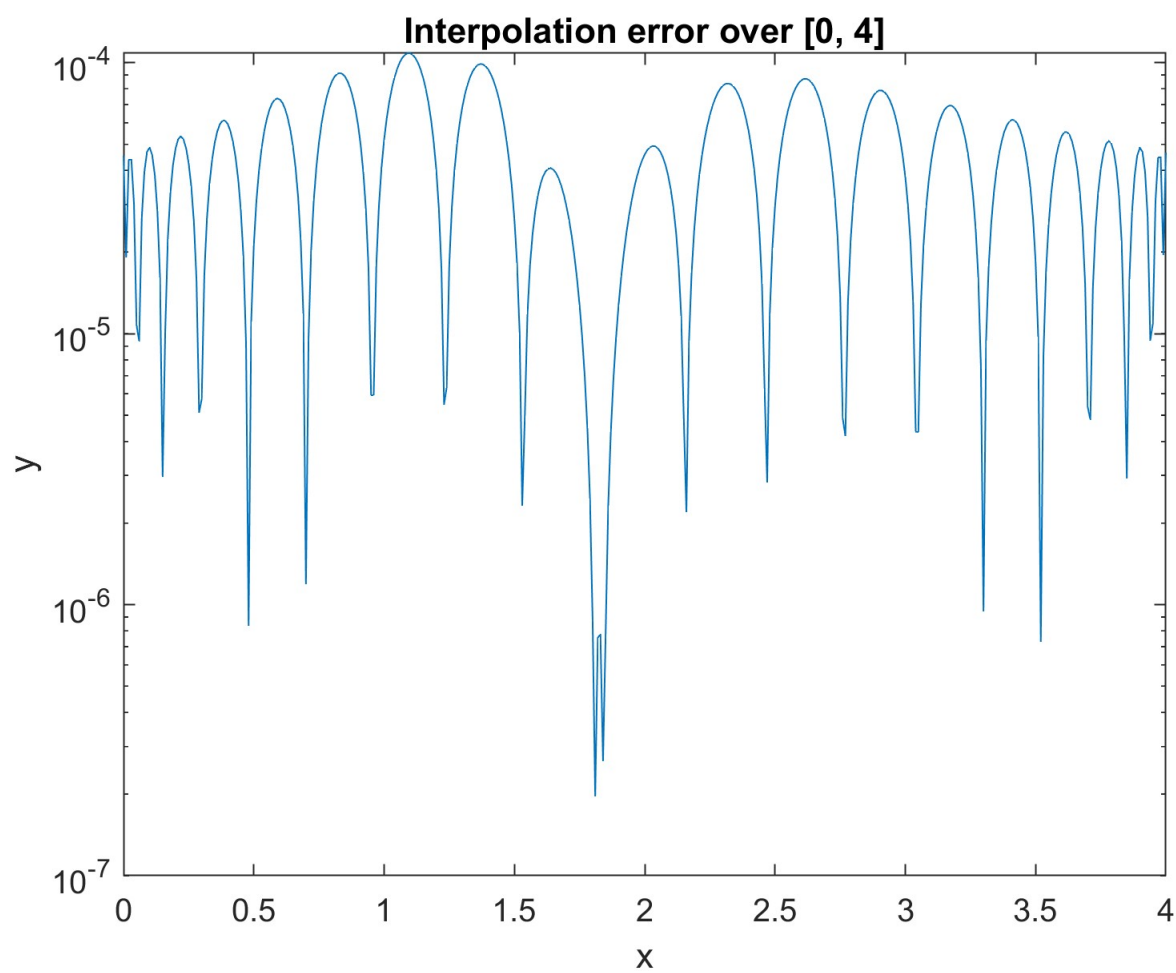












MATLAB Code - nest.m

```
% Program 0.1 Nested multiplication
% Evaluates polynomial from nested form using Horner s Method
% Input: degree d of polynomial,
%        array of d+1 coefficients c (constant term first),
%        x-coordinate x at which to evaluate, and
%        array of d base points b, if needed
% Output: value y of polynomial at x
function y = nest(d,c,x,b)
    if nargin<4, b=zeros(d,1); end
    y=c(d+1);

    for i=d:-1:1
        y = y.*(x-b(i))+c(i);
    end
end
```

MATLAB Code - newtdd.m

```
% Program 3.1 Newton Divided Difference Interpolation Method
% Computes coefficients of interpolating polynomial
% Input: x and y are vectors containing the x and y coordinates
%       of the n data points
% Output: coefficients c of interpolating polynomial in nested form
% Use with nest.m to evaluate interpolating polynomial
function c = newtdd(x,y,n)
    for j=1:n
        v(j,1)=y(j); % Fill in y column of Newton triangle
    end

    for i=2:n % For column i,
        for j=1:n+1-i % fill in column from top to bottom
            v(j,i)=(v(j+1,i-1)-v(j,i-1))/(x(j+i-1)-x(j));
        end
    end

    for i=1:n
        c(i)=v(1,i); % Read along top of triangle
    end % for output coefficients
end
```


MATLAB Code - project_5.m

```
clc; clear; close all

%% Question 1 - interpolating with evenly spaced points
n = 6;
x0 = 4 * (0 : (n-1))/(n-1);

y0 = cos(tan(sin(x0)));

% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% plot function vs interpolant
figure()
plot(x, f, 'r-', 'LineWidth', 1)
hold on
scatter(x0, y0, 'filled', 'green')
plot(x, p, 'b-', 'LineWidth', 1)
hold off
xlabel('x')
ylabel('y')
legend('cos(tan(sin(x)))', 'interpolating points', ...
'interpolating polynomial', 'location', 'best')
title('5th order interpolating polynomial for f = cos(tan(sin(x)))')

% plot interpolation error
interperror = abs(f - p);
figure()
semilogy(x, interperror)
xlabel('x')
ylabel('y')
title('Interpolation error over [0, 4]')

% compute maxerror
maxerror = max(interperror);
```

```

fprintf(['Question #1 \n\t The maximum absolute interpolation error (5 evenly spa
' points) is %f.\n'], maxerror)

%% Question 2 - interpolating with Chebyshev points
x0=2+2*cos((1:2:2*n-1)*pi/(2*n));

y0 = cos(tan(sin(x0)));

% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% plot function vs interpolant
figure()
plot(x, f, 'r-', 'LineWidth', 1)
hold on
scatter(x0, y0, 'filled', 'green')
plot(x, p, 'b-', 'LineWidth', 1)
hold off
xlabel('x')
ylabel('y')
legend('cos(tan(sin(x)))', 'chebyshev points', ...
'interpolating polynomial', 'location', 'best')
title('5th order interpolating polynomial for f = cos(tan(sin(x)))')

% plot interpolation error
interperror = abs(f - p);
figure()
semilogy(x, interperror)
xlabel('x')
ylabel('y')
title('Interpolation error over [0, 4]')

% compute maxerror
maxerror = max(interperror);
fprintf(['Question #2 \n\t The maximum absolute interpolation error (5 Chebyshev
' is %f.\n'], maxerror)

```

```

%% Question 3 - interpolating with n = 10 points

% evenly spaced points
n = 10;
x0 = 4 * (0 : (n-1))/(n-1);

y0 = cos(tan(sin(x0)));

% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% plot function vs interpolant
figure()
plot(x, f, 'r-', 'LineWidth', 1)
hold on
scatter(x0, y0, 'filled', 'green')
plot(x, p, 'b-', 'LineWidth', 1)
hold off
xlabel('x')
ylabel('y')
legend('cos(tan(sin(x)))', 'interpolating points', ...
'interpolating polynomial', 'location', 'best')
title('9^{th} order interpolating polynomial for f = cos(tan(sin(x)))')

% plot interpolation error
interperror = abs(f - p);
figure()
semilogy(x, interperror)
xlabel('x')
ylabel('y')
title('Interpolation error over [0, 4]')

% compute maxerror
maxerror = max(interperror);
fprintf(['Question #3 \n\t The maximum absolute interpolation error (10 evenly spaced points) is: %f\n', maxerror])

```

```

' points) is %f.\n'], maxerror)

% interpolating with Chebyshev points
x0=2+2*cos((1:2:2*n-1)*pi/(2*n));

y0 = cos(tan(sin(x0)));

% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% plot function vs interpolant
figure()
plot(x, f, 'r-', 'LineWidth', 1)
hold on
scatter(x0, y0, 'filled', 'green')
plot(x, p, 'b-', 'LineWidth', 1)
hold off
xlabel('x')
ylabel('y')
legend('cos(tan(sin(x)))', 'chebyshev points', ...
'interpolating polynomial', 'location', 'best')
title('9th order interpolating polynomial for f = cos(tan(sin(x)))')

% plot interpolation error
interperror = abs(f - p);
figure()
semilogy(x, interperror)
xlabel('x')
ylabel('y')
title('Interpolation error over [0, 4]')

% compute maxerror
maxerror = max(interperror);
fprintf(['\t The maximum absolute interpolation error (10 Chebyshev points)' ...
' is %f.\n'], maxerror)

```

```

%% Question 3 - interpolating with n = 20 points

% evenly spaced points
n = 20;
x0 = 4 * (0 : (n-1))/(n-1);

y0 = cos(tan(sin(x0)));

% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% plot function vs interpolant
figure()
plot(x, f, 'r-', 'LineWidth', 1)
hold on
scatter(x0, y0, 'filled', 'green')
plot(x, p, 'b-', 'LineWidth', 1)
hold off
xlabel('x')
ylabel('y')
legend('cos(tan(sin(x)))', 'interpolating points', ...
'interpolating polynomial', 'location', 'best')
title('19^{th} order interpolating polynomial for f = cos(tan(sin(x)))')

% plot interpolation error
interperror = abs(f - p);
figure()
semilogy(x, interperror)
xlabel('x')
ylabel('y')
title('Interpolation error over [0, 4]')

% compute maxerror
maxerror = max(interperror);
fprintf(['\t The maximum absolute interpolation error (20 evenly spaced' ...
' points) is %f.\n'], maxerror)

```

```

% interpolating with Chebyshev points
x0=2+2*cos((1:2:2*n-1)*pi/(2*n));

y0 = cos(tan(sin(x0)));

% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% plot function vs interpolant
figure()
plot(x, f, 'r-', 'LineWidth', 1)
hold on
scatter(x0, y0, 'filled', 'green')
plot(x, p, 'b-', 'LineWidth', 1)
hold off
xlabel('x')
ylabel('y')
legend('cos(tan(sin(x)))', 'chebyshev points', ...
'interpolating polynomial', 'location', 'best')
title('19th order interpolating polynomial for f = cos(tan(sin(x)))')

% plot interpolation error
interperror = abs(f - p);
figure()
semilogy(x, interperror)
xlabel('x')
ylabel('y')
title('Interpolation error over [0, 4]')

% compute maxerror
maxerror = max(interperror);
fprintf(['\t The maximum absolute interpolation error (20 Chebyshev points)' ...
' is %f.\n'], maxerror)

%% Question 4 - find 'n' that results in smallest possible interpolation error

```

```
% evenly spaced points
% error for n = 30 > error for n = 20
% take n = 30 as upper limit

maxerror = 1E6;
nbest = 30;
for n = 30:-1:5
    x0 = 4 * (0 : (n-1))/(n-1);

    y0 = cos(tan(sin(x0)));

    % compute interpolating polynomial
    c = newtdd(x0, y0, n);

    % compute polynomial over a grid
    x = 0:0.01:4;
    p = nest(n-1, c, x, x0);
    f = cos(tan(sin(x)));

    % max interpolation error
    interperror = abs(f - p);
    if max(interperror) < maxerror
        maxerror = max(interperror);
        nbest = n;
    end
end

fprintf(['Question #4 \n\t Least interpolation error is %g with %d evenly' ...
' spaced points.\n'], maxerror, nbest)

% chebyshev points

maxerror = 1E6;
nbest = 5;

for n = 5:1:60
    x0 = 2+2*cos((1:2:2*n-1)*pi/(2*n));

    y0 = cos(tan(sin(x0)));
```

```
% compute interpolating polynomial
c = newtdd(x0, y0, n);

% compute polynomial over a grid
x = 0:0.01:4;
p = nest(n-1, c, x, x0);
f = cos(tan(sin(x)));

% max interpolation error
interperror = abs(f - p);
if max(interperror) < maxerror
    maxerror = max(interperror);
    nbest = n;
end
end

fprintf(['\t Least interpolation error is %g with %d' ...
' Chebyshev points.\n'], maxerror, nbest)
```