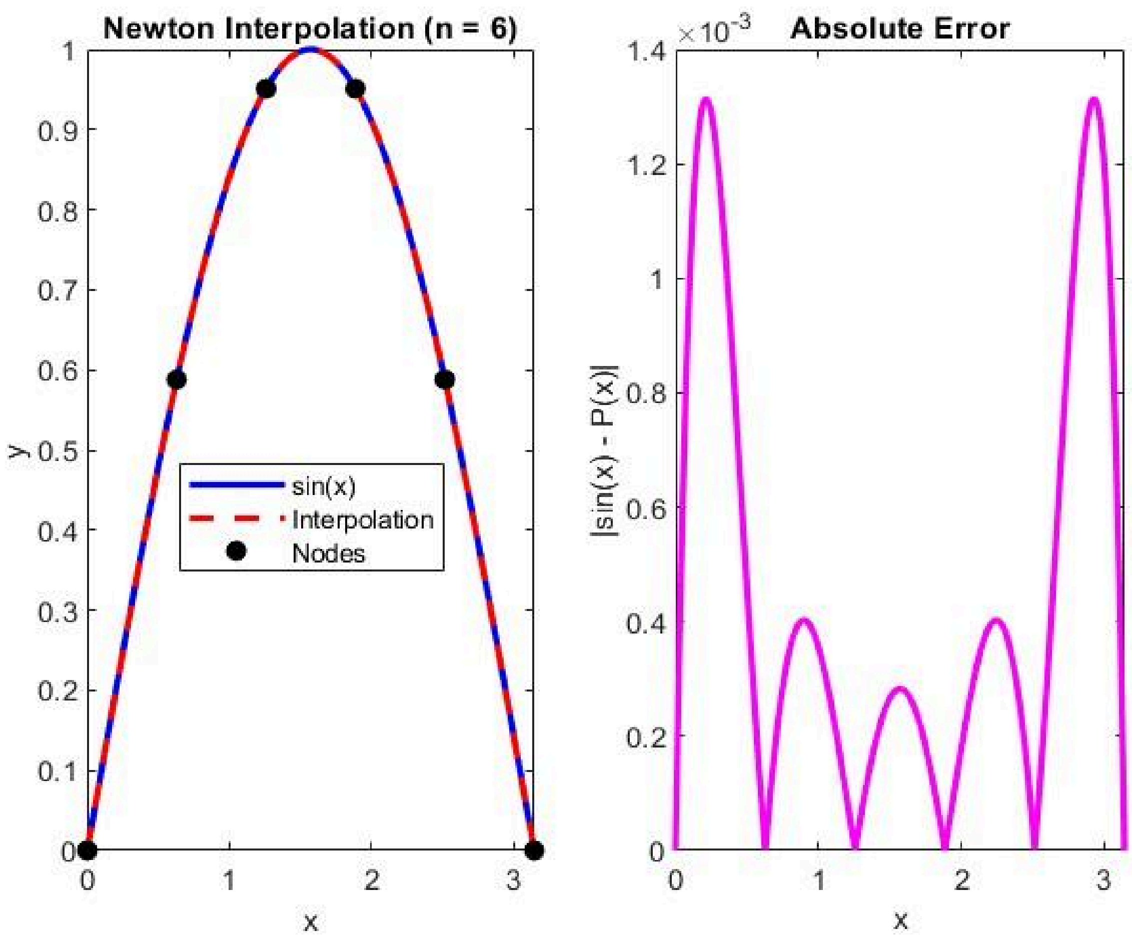
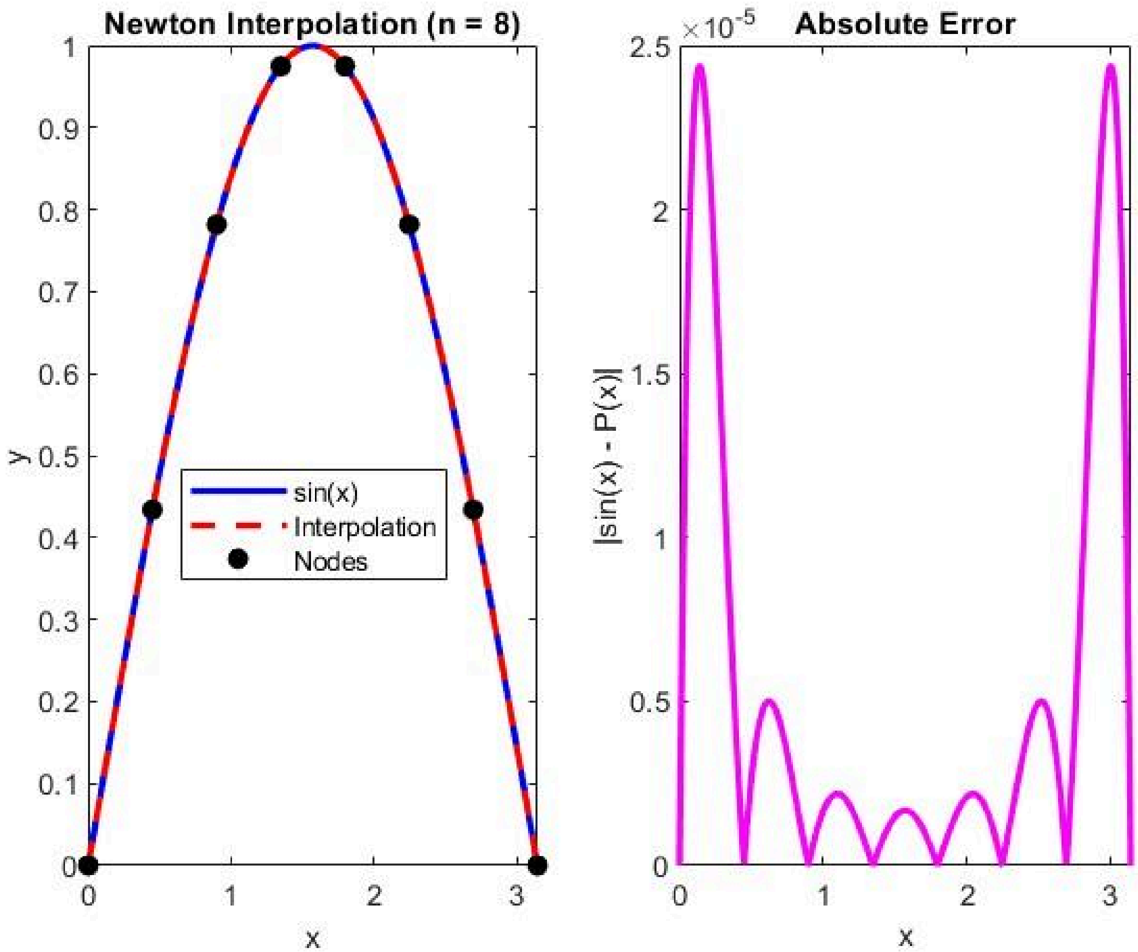
33.

```matlab
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NAME: Josh Derrow
% JMU-EID: derrowjb
% DATE: April 17, 2025
%
% PROGRAM: newtonpoly1.m
% PURPOSE: Approximates the function f(x) = sin(x) via Newton polynomials.
%
% VARIABLES:
%    ns = list of interpolation points
%    i = index of current interpolation point (n value)
%    n = current interpolation point (n value)
%    x = vector of equally spaced points where f(x) is sampled
%    y = the function of interest (f(x) = sin(x))
%    coeff = Newton coefficients of the polynomial
%    z = vector of equally spaced points where f(x) is evaluated
%    true_y = vector containing the true values of sin(x)
%    interp_y = vector containing the interpolated values of the Newton
%               polynomial
%    error = vector that contains the absolute differences between the true
%            and interpolated values
%    max_error = maximum difference between the true and interpolated
%                values
%
% JMU PLEDGE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Maximum error for n = 6: 1.31e-03
% Maximum error for n = 8: 2.44e-05
% Maximum error for n = 10: 3.01e-07
function newtonpoly1
   ns = [6, 8, 10];
   for i = 1:length(ns)
       n = ns(i);
       x = linspace(0, pi, n);
       y = sin(x);
       coeff = newtoncoeff(x, y);
       % Evaluation points
       z = linspace(0, pi, 1000);
       true_y = sin(z);
       interp_y = arrayfun(@(zval) newtoneval(coeff, x, zval), z);
       error = abs(true_y - interp_y);
       max_error = max(error);
       % Plot the function and interpolation
       figure;
       subplot(1, 2, 1);
       plot(z, true_y, 'b-', 'LineWidth', 2); hold on;
       plot(z, interp_y, 'r--', 'LineWidth', 2);
       plot(x, y, 'ko', 'MarkerFaceColor', 'k');
```

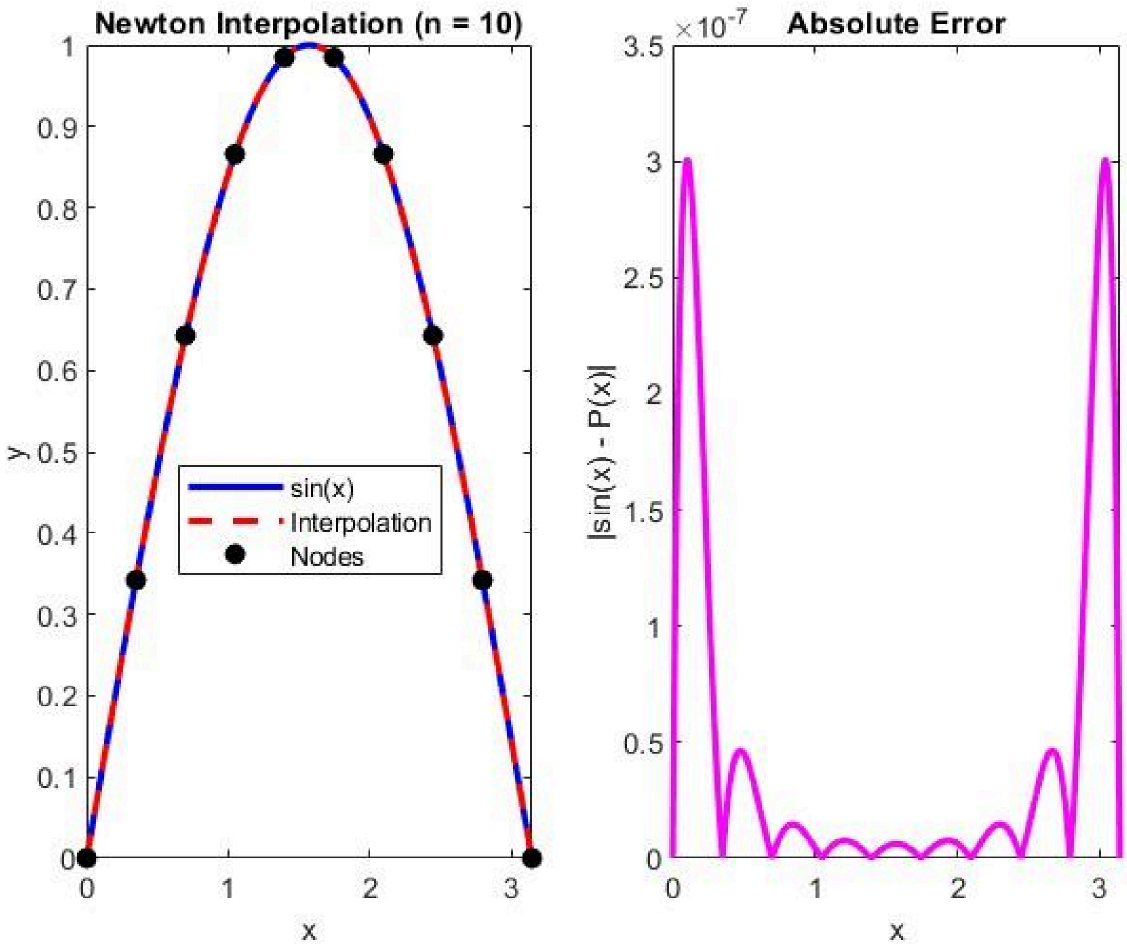
```
        title(['Newton Interpolation (n = ', num2str(n), ')']);
        legend('sin(x)', 'Interpolation', 'Nodes', 'Location', 'best');
        xlabel('x');
        ylabel('y');
        % Plot the absolute error
        subplot(1, 2, 2);
        plot(z, error, 'm', 'LineWidth', 2);
        title('Absolute Error');
        xlabel('x');
        ylabel('|sin(x) - P(x)|');
        % Display maximum error
        fprintf('Maximum error for n = %d: %.2e\n', n, max_error);
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NAME: Josh Derrow
% JMU-EID: derrowjb
% DATE: April 17, 2025
%
% PROGRAM: newtoncoeff.m
% PURPOSE: Helper method to interpolate Newton coefficients.
%
% VARIABLES:
%    coeff = return value representing Newton coefficients
%    x = vector of equally spaced points where f(x) is sampled
%    y = the function of interest: f(x)
%    n = number interpolation points
%    j = index of the current coefficient evaluation
%
% JMU PLEDGE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function coeff = newtoncoeff(x, y)
    n = length(x);
    coeff = y;
    for j = 2:n
        coeff(j:n) = (coeff(j:n) - coeff(j - 1:n - 1)) ./ (x(j:n) - x(1:n - j +
1));
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NAME: Josh Derrow
% JMU-EID: derrowjb
% DATE: April 17, 2025
%
% PROGRAM: newtoneval.m
% PURPOSE: Helper method to evaluate the Newton polynomial at the point z.
%
```

```matlab
% VARIABLES:
%    val = return value representing the Newton polynomial at the point z
%    coeff = return value representing Newton coefficients
%    x = vector of equally spaced points where f(x) is sampled
%    z = point at which the Newton polynomial is evaluated
%    n = number of Newton coefficients
%    k = index of the current Newton interpolation value
%
% JMU PLEDGE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function val = newtoneval(coeff, x, z)
   n = length(coeff);
   val = coeff(n);
   for k = n - 1:-1:1
       val = val * (z - x(k)) + coeff(k);
   end
end
```





Newton Interpolation (n = 8)
Absolute Error
sin(x)
Interpolation
Nodes
y
x
|sin(x) - P(x)|
×10^-5


Newton Interpolation (n = 10)
y
x
sin(x)
Interpolation
Nodes
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
1
2
3
Absolute Error
×10^-7
|sin(x) - P(x)|
0
0.5
1
1.5
2
2.5
3
3.5
0
1
2
3
x

34.
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NAME: Josh Derrow
% JMU-EID: derrowjb
% DATE: April 17, 2025
%
% PROGRAM: newtonpoly2.m
% PURPOSE: Approximates the function f(x) = 1 / (1 + 25x^2) via Newton
%          polynomials.
%
% VARIABLES:
%    ns = list of interpolation points
%    f = the function of interest (f(x) = 1 / (1 + 25x^2))
%    z = interval of evaluation
%    true_y = vector containing the true values of 1 / (1 + 25x^2)
%    i = index of current interpolation point (n value)
%    n = current interpolation point (n value)
%    x = vector of equally spaced points where f(x) is sampled
%    y = function values at the x points
%    coeff = Newton coefficients of the polynomial
%    interp_y = vector containing the interpolated values of the Newton
%               polynomial
%
% JMU PLEDGE
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Using more points is both good and bad in this scenario. In the middle of
% the plot, n = 11 was the most accurate, however, towards the endpoints of
% the interval it was the least accurate. The plots of n = 5 and n = 7
% follow a similar trend but to a smaller degree.
function newtonpoly2
   ns = [5, 7, 11];
   f = @(x) 1 ./ (1 + 25 * x.^2);
   z = linspace(-1, 1, 1000);
   true_y = f(z);
   % Create the figure for plotting and plot the true function
   figure;
   hold on;
   plot(z, true_y, 'b-', 'LineWidth', 2);
   for i = 1:length(ns)
       n = ns(i);
       x = linspace(-1, 1, n);         % Equally spaced nodes
       y = f(x);                       % Function values at the nodes
       % Compute the Newton coefficients and polynomial
       coeff = newtoncoeff(x, y);
       % Interpolate and plot the Newton polynomial
       interp_y = arrayfun(@(zval) newtoneval(coeff, x, zval), z);
       plot(z, interp_y, 'LineWidth', 2);
   end
   % Configure the plot
```

```
    title('Newton Interpolation for f(x) = 1 / (1 + 25x^2)');
    legend('True function', 'n = 5', 'n = 7', 'n = 11', 'Location', 'best');
    xlabel('x');
    ylabel('f(x)');
    grid on;
    hold off;
end
```