Journal of Science-FAS-SEUSL (2025) 06(02) 82-96
ISSN:2738-2184

Modeling Sri lanka's Population Growth using Vital Demographic Indicators: A Comparative Analysis of Lagrange Polynomial and Cubic Spline Interpolation Methods

A.N. Wazeetha Mazari[1*], U.L.M. Althaf[l], M.S.M. Farhan[2] and M.A.A.M. Faham[1]

[1]Department of Mathematical Sciences, Faculty of Applied Sciences, South Eastern University of Sri Lanka 32200

[2]Department of Physical Science, Trincomalee Campus, Eastern University, Sri Lanka 31000

[*]mazari@seu.ac.lk

Abstract

This study models and analyzes Sri Lanka's population growth trends using key Demographic Indicators such as Mid-Year Population, Crude Birth Rate, and Crude Death Rate through Lagrange Polynomial Interpolation and Cubic Spline Interpolation techniques. Demographic data spanning the period from 2015 to 2023 were collected from Official National Demographic Statistics. The years were transformed into numerical time indices by assigning 2015 as Year 0. The year 2019 (Year 4) was selected as the interpolation testing year and the year 2024 (Year 9) was used for extrapolation testing year. Predicted values obtained from both interpolation and extrapolation methods were compared with actual recorded data, and accuracy was evaluated using the Absolute Percentage Error measure. The results indicate that cubic spline interpolation provides slightly more accurate estimates than Lagrange polynomial interpolation, primarily due to its piecewise smooth structure and reduced oscillatory behavior. Although both methods performed satisfactorily within the data range, cubic spline interpolation demonstrated greater stability and reliability for extrapolation, whereas Lagrange interpolation showed significant divergence. This study highlights the usefulness of interpolation based mathematical modeling techniques in demographic trend analysis.

Keywords: Demographic Indicators, Lagrange Polynomial, Cubic Spline Interpolation, Population Modeling, MATLAB, Predictive Modeling.

1. Introduction

Analyzing and understanding population growth patterns is crucial for national planning, policy formulation, and forecasting vital demographic trends. Mathematical modeling provides systematic tools for analyzing historical population behavior and estimate missing or future values. Several studies have employed Machine Learning, Predictive Modeling, and Statistical Techniques to analyze demographic and social datasets [1-5].

Focusing on Sri Lanka's population dynamics from the period 2015 to 2023, this study evaluates three key demographic indicators: Mid-Year Population, Crude Birth Rate, and Crude Death Rate using two classical interpolation techniques:

1. Lagrange Polynomial Interpolation
2. Cubic Spline Interpolation

Corresponding Author: mazari@seu.ac.lk

The primary objective is to compare the performance of these techniques in terms of interpolation accuracy within the data range and extrapolation behavior beyond the available data. Since Demographic indicators typically evolve gradually over time, spline-based interpolation is expected to perform well, while Lagrange interpolation might suffer from oscillation behavior as the degree of the polynomial increases. Predictive accuracy is measured using the Absolute Percentage Error formula.

2. Methodology

Lagrange polynomial interpolation and Cubic Spline interpolation were utilized to model Sri Lanka's population growth using selected vital demographic indicators. Annual demographic data for Sri Lanka over a selected historical period were used to construct interpolation models. An intermediate year within the data range was selected to validate interpolation accuracy. Extrapolation was subsequently performed for future years beyond the available data using the same mathematical models.

MATLAB software was used to implement both interpolation methods and to compute predicted population forecasts based on the selected demographic indicators. Error analysis was carried out by comparing the predicted values with actual recorded data using the Absolute Percentage Error Metric.

2.1 Lagrange's' Interpolation Polynomial Method

The Lagrange interpolation polynomial provides a direct method for constructing a single polynomial that passes through all given data points. The polynomial of degree n is defined as

$$P_n(x) = \sum_{i=0}^{n} f(x_i)\, L_i(x)$$

where the Lagrange basis polynomials $L_i(x)$ are given by

$$L_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right), \quad i = 0, 1, 2, \cdots, n.$$

These basis polynomials satisfy the property

$$L_i(x_k) = \delta_{ik} = \begin{cases} 1 & \text{if} \quad i = k \\ 0 & \text{if} \quad i \neq k \end{cases}$$

where δ_{ik} denotes the Kronecker delta.

Although this method ensures exact interpolation at the given data points, it may suffer from oscillatory behavior, particularly when applied to large datasets or used for extrapolation and is quite laborious.

2.2 Cubic Spline Interpolation

Given a set of $n+1$ data points (x_0, y_0), (x_1, y_1),……., (x_n, y_n), cubic spline interpolation constructs a set of piecewise cubic polynomials $S_i(x)$ defined on each subinterval $|x_i, x_i + 1|$ defined by

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \;\; i = 0,1,2, \ldots . n - 1.$$

Each spline segment satisfies the interpolation conditions

$$S_i(x_i) = y_i, \qquad S_i(x_i + 1) = y_i + 1,$$

Along with continuity of the first and second derivatives at interior points:

$$S_i'(x_i + 1) = S_{i+1}'(x_i + 1), \quad S_i''(x_i + 1) = S_{i+1}''(x_i + 1).$$

For a natural cubic spline, the second derivatives at the endpoints are set to zero:

$$S_0''(x_0) = 0, \quad S_{n-1}''(x_n) = 0.$$

The coefficients a_i, b_i, c_i and d_i can be efficiently determined by solving a tridiagonal matrix derived from the above continuity conditions:

1. Continuity of Values
2. Continuity of First Derivatives
3. Continuity of Second Derivatives

MATLAB's built-in spline functions were used for implementation. This method reduces oscillatory behavior and preserves local smoothness, making it particularly suitable for demographic data.

2.3 Data Collection

Demographic data for Sri Lanka from 2015-2023 were obtained from publicly available official statistics (Source: Department of Census and Statistics/ Registrar General's Department). The three variables considered were:

1. Mid-Year Population (Millions)
2. Crude Birth Rate (Per 1,000 Population)
3. Crude Death Rate (Per 1,000 Population)

3. Results and Discussion

Figure 1 illustrates Cubic Spline appears smoother and more realistic between data points. It avoids large oscillations and better reflects natural population changes. Lagrange Polynomial shows slight irregularities, especially near the boundaries; a common issue due to Runge's phenomenon in high-degree polynomials. Near the end (last few data points), Lagrange interpolation diverges more, while Cubic Spline stays closer to actual trends.

Table1: Interpolation Results (Testing Year: 2019)

Indicator	Actual value	Predicted value (Lagrange)	Predicted value (Cubic Spline)	Error % (Lagrange)	Error % (Cubic Spline)
Population (Millions)	21.803	21.7757	21.7832	0.125	0.091
Crude birth rate	15.11	14.8781	14.8896	1.535	1.459
Crude death rate	6.48	6.3430	6.3882	2.114	1.417

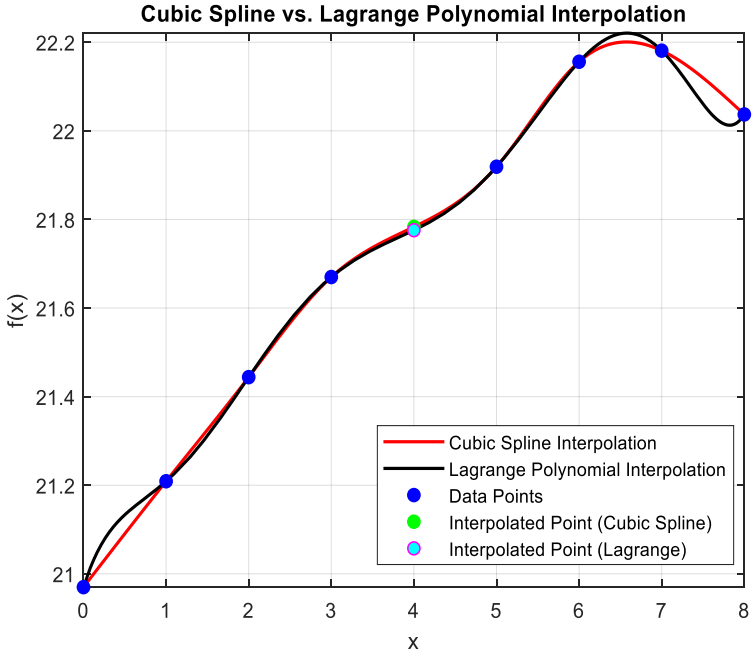

Figure 1: Mid-Year Population

Figure 2 explains how both methods follow a decreasing trend in Birth Rate over time. The Cubic Spline (red) provides a smooth and natural curve with no sharp deviations. The Lagrange Polynomial fits well overall but shows a slight deviation at the start, where it overshoots slightly. The difference between the two interpolations is small here due to the smoother, monotonic nature of the data.

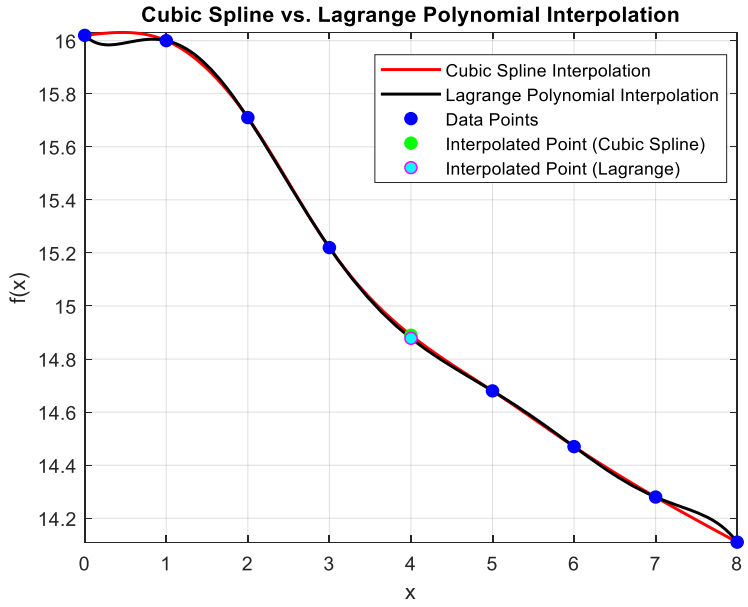

Figure 2: Crude Birth Rate

Figure 3 interprets that Cubic Spline Interpolation is clearly superior for modeling Crude Death Rate, as it avoids the large oscillations seen in the Lagrange method. The Lagrange polynomial is unsuitable here due to its instability over wider intervals with more data points.

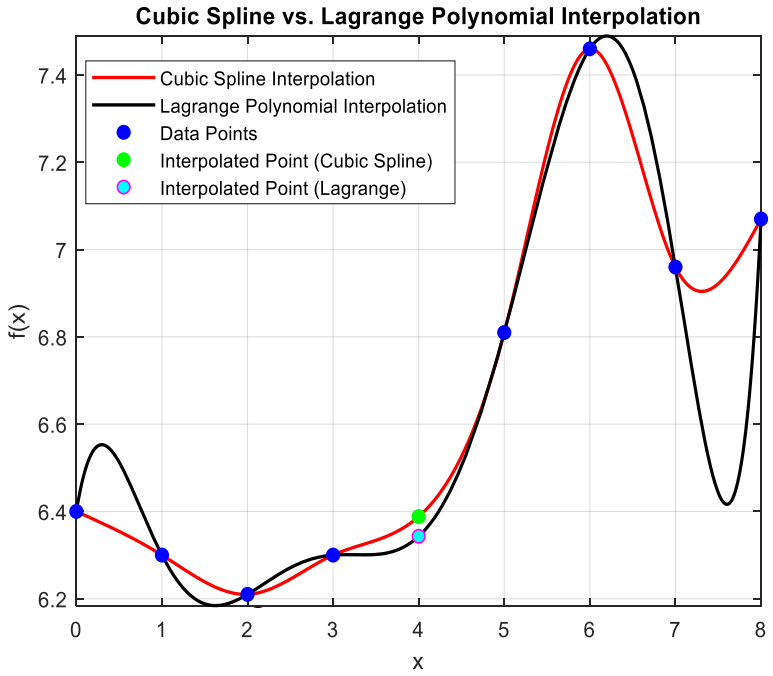

Figure 3: Crude Death Rate

Table 2: Extrapolation Results (Testing Year: 2024)

Indicator	Actual value	Predicted value (Lagrange)	Predicted value (Cubic Spline)	Error % (Lagrange)	Error % (Cubic Spline)
Population (Millions)	21.916	28.098	21.9018	28.208	0.065
Crude Birth Rate	10.1	41.67	14.0978	312.574	39.582
Crude Death Rate	7.8	42.19	10.5342	440.897	35.054

Figure 4-6 demonstrates that Cubic spline interpolation significantly outperforms Lagrange interpolation in extrapolation accuracy. Lagrange interpolation produces extreme overestimations due to polynomial oscillations, rendering it unsuitable for long-term extrapolation. While Population size follows a relatively smooth trend and is well captured by cubic splines rate-based indicators such as Birth and Death Rates exhibit higher variability resulting in larger extrapolation errors for both methods.

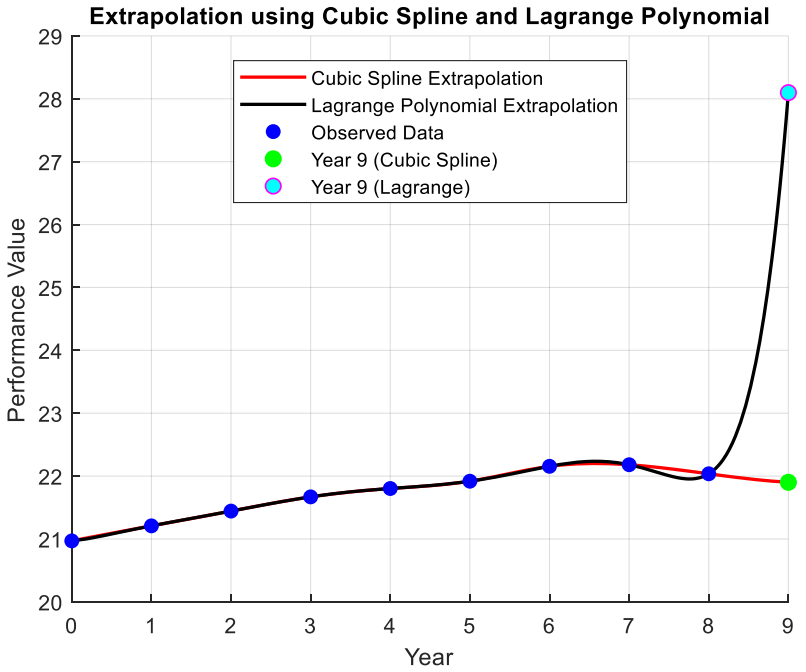

Figure 4: Mid-Year Population

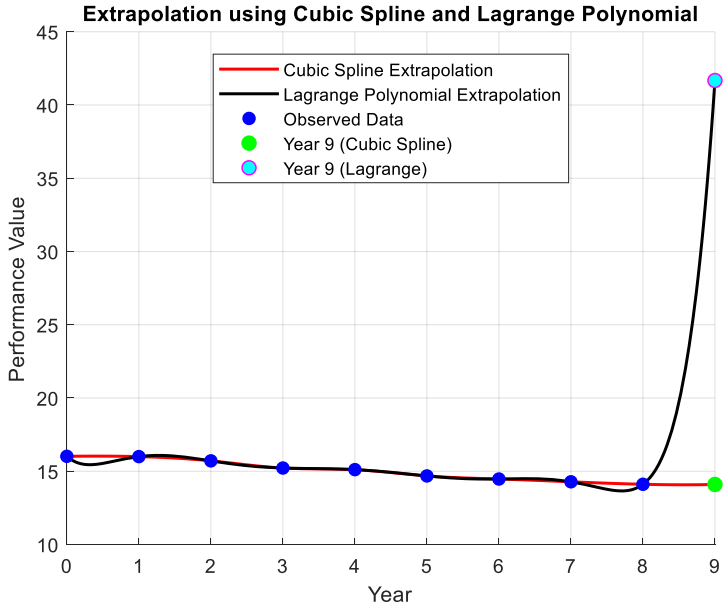

Figure 5: Crude Birth Rate

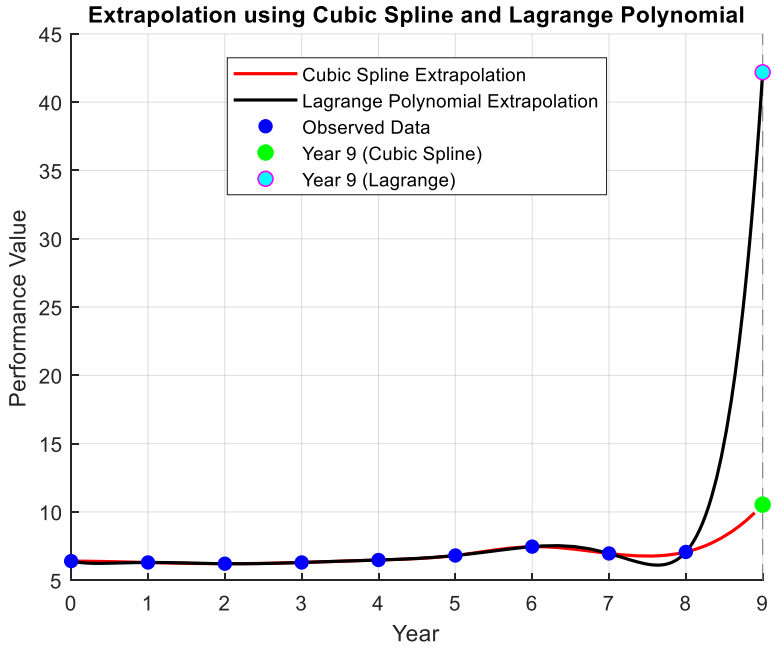

Figure 6: Crude Death Rate

4. Conclusion

This study analyzed and examined Sri Lanka's population growth using Lagrange polynomial and cubic spline interpolation methods applied to demographic data from 2015 to 2023. The results for the testing year 2019 indicate that cubic spline interpolation provides more accurate predictions with lower error percentages compared to the Lagrange method. Although Lagrange interpolation performs reasonably well within the data range, it suffers from inherent limitations due to oscillations issues associated with high-degree polynomials. Both techniques remain stable within the selected interval; however, extrapolation to the year 2024 indicates that cubic spline interpolation is more robust and reliable for demographic forecasting.

Nevertheless, caution is required when extrapolating rate-based indicators, as their natural variability can diminish prediction accuracy. Overall, the findings suggest that cubic spline interpolation is better suited for demographic modeling where population changes occur gradually over time. Future research may enhance predictive accuracy and robustness by incorporating smoothing splines, regression based models, machine learning techniques, and hybrid interpolation forecasting approaches.

5. References

1. de Boor, C (1978). *A practical guide to splines*. Applied Mathematical Sciences, 27, Springer, New York.
2. Preston S.H., Heuveline, P & Guillot, m (2001). *Demography: Measuring and modeling population processes*. Population and Development Review, Blackwell Publishers, Oxford 27(2), 365-367.
3. Logan, J. R., Xu, Z. & Stults, B. (2014). Interpolating population census data. *The Professional Geographer,* 66(3), 412-422.
4. Ibrahim, A. & Adeyemi, O. (2013). Comparative study of polynomial and spline interpolation methods. *International Journal of Mathematical Analysis,* 7(27), 1329–1338.
5. Lee, R. & Carter, L. (1992). Modeling and forecasting population growth. *Journal of the American Statistical Association,* 87(419), 659–671.

6. Appendices

Appendix A: MATLAB Program for Lagrange's Interpolation Method

```
syms x
P = 0;
u=x_nodes;
v=y_nodes;
n = length(u);

if length(x_nodes) ~= length(y_nodes)
   error('ERROR: Length of input vectors u and v must be equal.');
end

% Construct Lagrange polynomial
for i = 1:n
   N = 1; D = 1;
   for j = 1:n
      if j ~= i
```

```
            N = N*(x - x_nodes(j));
            D = D*(x_nodes(i) - x_nodes(j));
        end
    end
    P = P + (N/D)*y_nodes(i);
end

f(x) = simplify(P);

% Display polynomial
disp('Symbolic Lagrange Polynomial f(x):');
pretty(f(x))

% Evaluate at x_eval (if provided) and mark on the plot

 y_eval_lag = double(subs(f, x, x_interp));

 fprintf('\nInterpolated value at x = %.4f uisng Lagrange method is S(x) = %.4f\n', x_interp,
y_eval_lag);
```

Appendix B: MATLAB Program for Cubic Spline Interpolation Method

```
clc; close all;
syms x

% Step sizes
h = diff(x_nodes);

% --- Construct coefficient matrix for second derivatives M ---
A = sym(zeros(n));
rhs = sym(zeros(n,1));

% Natural boundary conditions: M1 = Mn = 0
A(1,1) = 1;
```

```
A(n,n) = 1;
rhs(1) = 0;
rhs(n) = 0;

% Internal equations
for i = 2:n-1
   A(i,i-1) = h(i-1);
   A(i,i)   = 2*(h(i-1) + h(i));
   A(i,i+1) = h(i);
   rhs(i)   = 6*((y_nodes(i+1) - y_nodes(i))/h(i) - (y_nodes(i) - y_nodes(i-1))/h(i-1));
end

% Solve for M (second derivatives)
M = A\rhs;

% --- Construct spline polynomials symbolically ---
S = sym(zeros(1, n-1)); % store spline for each interval

for i = 1:n-1
   xi  = x_nodes(i);
   xi1 = x_nodes(i+1);
   hi  = h(i);

   % Cubic spline formula in [xi, xi+1]
   S(i) = simplify( M(i)*(xi1 - x)^3/(6*hi) + M(i+1)*(x - xi)^3/(6*hi) + ...
                (y_nodes(i) - M(i)*hi^2/6)*(xi1 - x)/hi + ...
                (y_nodes(i+1) - M(i+1)*hi^2/6)*(x - xi)/hi );
end

% --- Display symbolic spline pieces ---
disp('Symbolic Cubic Spline Functions:')
for i = 1:n-1
   fprintf('\nInterval [%g, %g]:\n', x_nodes(i), x_nodes(i+1));
   pretty(S(i));
```

```
end

% Find which interval x_interp belongs to
interval = find(x_interp >= x_nodes(1:end-1) & x_interp <= x_nodes(2:end), 1);

if isempty(interval)
   error('The given x-value %.4f is outside the interpolation range [%g, %g].', ...
         x_interp, x_nodes(1), x_nodes(end));
end

% Evaluate the spline value at x_interp
y_interp_CS = double(subs(S(interval), x, x_interp));
fprintf('\nInterpolated value at x = %.4f uisng cubic spline method is S(x) = %.4f\n', x_interp,
y_interp_CS);
```

Appendix C: MATLAB Program for Comparison of Lagrange's Interpolation and Cubic Spline Interpolation Method

```
x_nodes = [0 1 2 3 4 5 6 7];     % Example x-values (change as needed)
y_nodes = [0 1 4 5 7 8 10 15];     % Example y-values (change as needed)
n = length(x_nodes);
% --- Input interpolation value ---
x_interp = input('\nEnter the x-value to interpolate: ');

%running cubic spline interpolation
cubic_spline

%running lagrange interpolation
lagrange

% Plot piece wise cubic splines
for i = 1:n-1
   h1 = fplot(S(i), [x_nodes(i), x_nodes(i+1)], 'r', 'LineWidth', 1.5);
   hold on
```

```
end

% --- Plot Lagrange polynomial ---
h2 = fplot(f, [min(u), max(u)], 'k', 'LineWidth', 1.5);
hold on

% --- Plot given data points ---
h3 = plot(x_nodes, y_nodes, 'bo', 'MarkerFaceColor', 'b');

% --- Mark interpolated points ---
h4 = plot(x_interp, y_interp_CS, 'go', 'MarkerSize', 6, 'MarkerFaceColor', 'g');
h5 = plot(x_interp, y_eval_lag, 'mo', 'MarkerSize', 6, 'MarkerFaceColor', 'c');

% --- Add one dummy (invisible) plot to represent all spline segments ---
hSpline = plot(NaN, NaN, 'r', 'LineWidth', 1.5);
hLag    = plot(NaN, NaN, 'k', 'LineWidth', 1.5);

% --- Set labels, title, and grid ---
xlabel('x');
ylabel('f(x)');
title('Cubic Spline vs. Lagrange Polynomial Interpolation');
grid on;

% --- Final clean legend (no repetition) ---
legend([hSpline, hLag, h3, h4, h5], ...
       {'Cubic Spline Interpolation', ...
        'Lagrange Polynomial Interpolation', ...
        'Data Points', ...
        'Interpolated Point (Cubic Spline)', ...
        'Interpolated Point (Lagrange)'}, ...
        'Location', 'best');
```