Questoion 3 2025

Q1. Find the value of √7 using Newton-Raphson method correct up to 4 decimal places.  
  
Code:

1. clc;

2. clear;

3.

4. % Define the function and its derivative

5. f = @(x) x.^2 - 7;

6. df = @(x) 2\*x;

7.

8. % Initial guess

9. x0 = 2.5;

10.

11. % Tolerance and max iterations

12. tol = 1e-4;

13. max\_iter = 100;

14.

15. % Store iteration values for plotting

16. x\_vals = x0;

17.

18. % Newton-Raphson Iteration

19. for i = 1:max\_iter

20. x1 = x0 - f(x0)/df(x0);

21. x\_vals(end+1) = x1; % Store new approximation

22. if abs(x1 - x0) < tol

23. break;

24. end

25. x0 = x1;

26. end

27.

28. % Plot convergence

29. figure;

30. plot(1:length(x\_vals), x\_vals, '-o', 'LineWidth', 2, 'MarkerSize', 6);

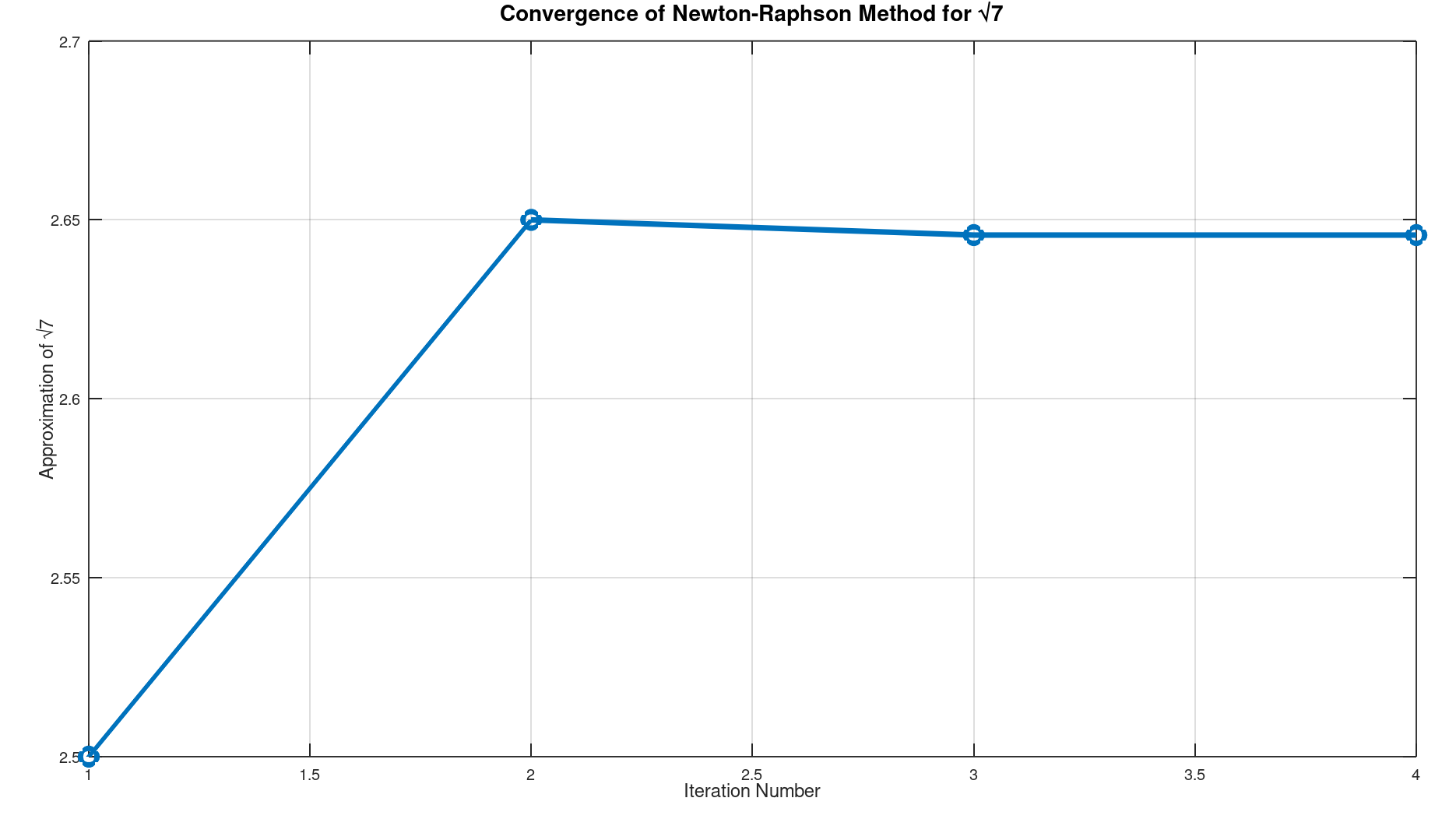
31. xlabel('Iteration Number', 'FontSize', 12);

32. ylabel('Approximation of \surd7', 'FontSize', 12);

33. title('Convergence of Newton-Raphson Method for \surd7', 'FontSize', 14);

34. grid on;

Graph:



Q2. Using Newton’s Interpolation, calculate P(2.3) from the given table  
Code:

1. clc;

2. clear;

3.

4. % Given data

5. x = [0 2 4 6];

6. y = [2.2 4.2 5.2 8.6];

7.

8. % Number of data points

9. n = length(x);

10.

11. % Construct forward difference table

12. diff\_table = zeros(n, n);

13. diff\_table(:,1) = y';

14.

15. for j = 2:n

16. for i = 1:n-j+1

17. diff\_table(i,j) = diff\_table(i+1,j-1) - diff\_table(i,j-1);

18. end

19. end

20.

21. % Compute u and interpolate at x = 2.3

22. x\_interp = 2.3;

23. h = x(2) - x(1);

24. u = (x\_interp - x(1)) / h;

25.

26. % Newton Forward Formula

27. P = diff\_table(1,1);

28. u\_term = 1;

29.

30. for i = 1:n-1

31. u\_term = u\_term \* (u - i + 1) / i;

32. P = P + u\_term \* diff\_table(1,i+1);

33. end

34.

35. % Display result

36. fprintf('Interpolated value P(2.3) = %.4f\n', P);

37.

38. % Plot

39. xx = linspace(min(x), max(x), 100);

40. yy = interp1(x, y, xx, 'spline'); % Smooth curve for visualization

41.

42. figure;

43. plot(x, y, 'ro', 'MarkerSize', 8, 'DisplayName', 'Data Points'); hold on;

44. plot(xx, yy, 'b-', 'LineWidth', 1.5, 'DisplayName', 'Spline Fit');

45. plot(x\_interp, P, 'ks', 'MarkerFaceColor','g', 'DisplayName', 'P(2.3)');

46. legend show;

47. xlabel('x');

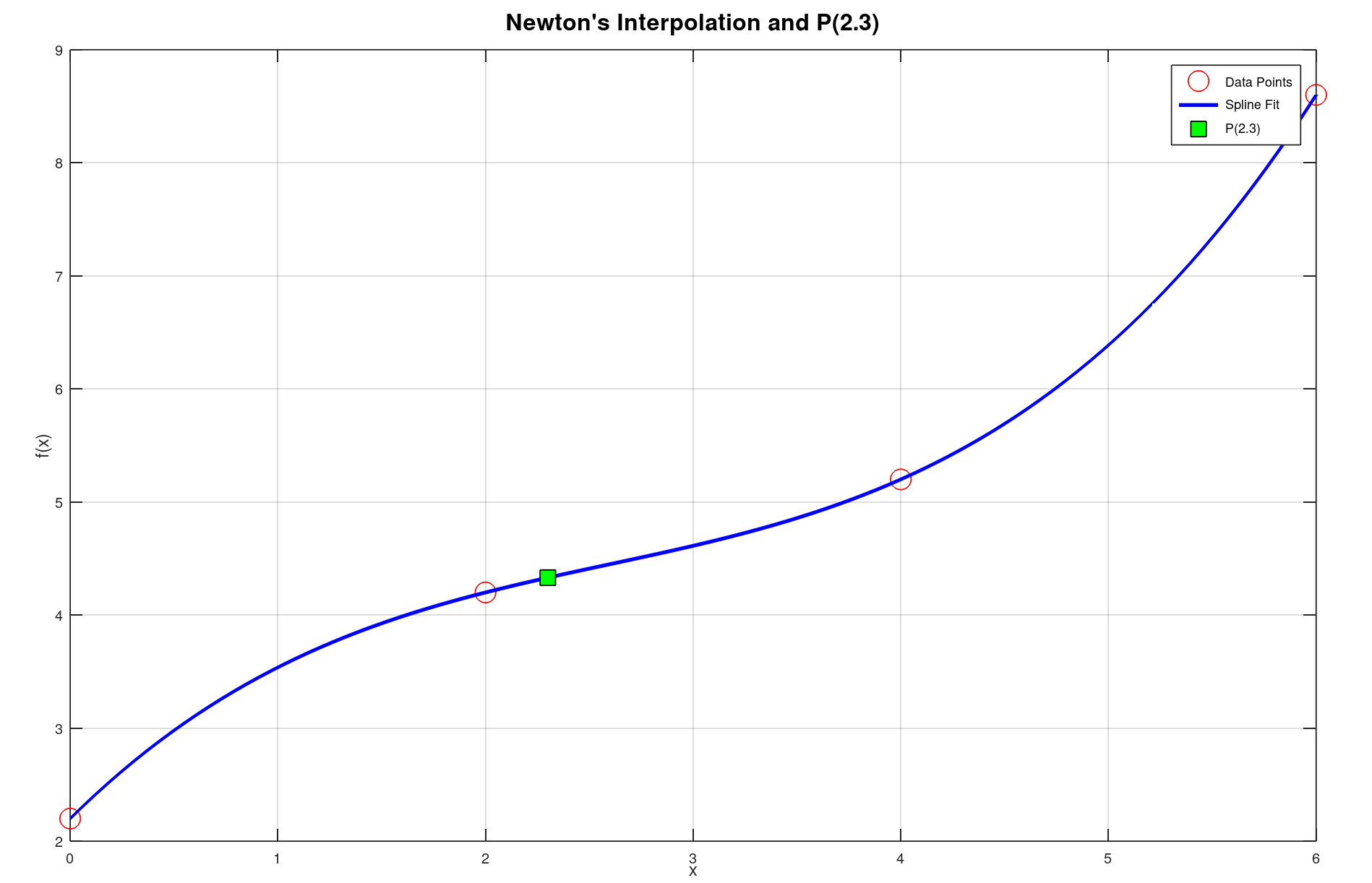
48. ylabel('f(x)');

49. title('Newton''s Interpolation and P(2.3)');

50. grid on;

51.

Graph:



Q3. Using Simpson’s 1/3rd rule, integrate  
  
Code:

1. clc;

2. clear;

3.

4. % Limits of integration

5. a = 0;

6. b = 6;

7.

8. % Number of intervals (must be even)

9. n = 6;

10. h = (b - a)/n;

11.

12. % x values

13. x = a:h:b;

14.

15. % Function values

16. f = @(x) 1./(1 + x.^2);

17. y = f(x);

18.

19. % Apply Simpson's 1/3rd rule

20. I = y(1) + y(end); % f(x0) + f(xn)

21.

22. for i = 2:n

23. if mod(i,2) == 0 % even index (odd x\_i)

24. I = I + 4\*y(i);

25. else % odd index (even x\_i)

26. I = I + 2\*y(i);

27. end

28. end

29.

30. I = (h/3) \* I;

31.

32. % Display result

33. fprintf('Approximate value of the integral = %.6f\n', I);

34.

35. % Optional: Plot the function

36. xx = linspace(a, b, 1000);

37. yy = f(xx);

38. figure;

39. plot(xx, yy, 'b', 'LineWidth', 2);

40. xlabel('x', 'FontSize', 14); ylabel('f(x)', 'FontSize', 14);

41. title('f(x) = 1 / (1 + x^2)', 'FontSize', 16);

42. grid on;

43.

Graph:

