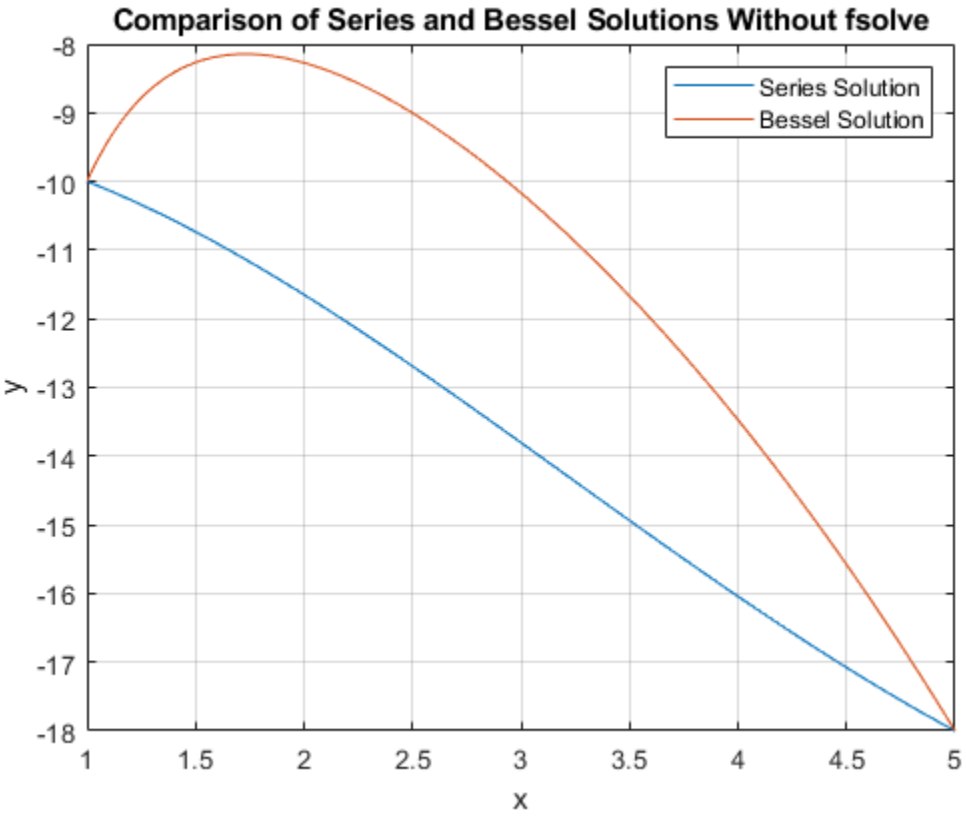
```
function compare_solutions_without_fsolve()
    % Define the u1 and u2 using Bessel functions
    u1 = @(x) 1/sqrt(x) .* besseli(0, 2 * sqrt(x));
    u2 = @(x) 1/sqrt(x) .* besselk(0, 2 * sqrt(x));

    % Compute u1 and u2 at given boundary points (1 and 5)
    u1_values = [u1(1), u1(5)];
    u2_values = [u2(1), u2(5)];

    % Solve the system of equations to determine coefficients using boundary
 conditions
    CC = linsolve([u1_values; u2_values]', [-10; -18]);

    % Define the solution using Bessel
    bessel_trick = @(x) CC(1) * u1(x) + CC(2) * u2(x);

    % Define the series solution to the differential equation using the
 Frobenius method
    function y_val = series(x_val, N)
        if nargin < 2
            N = 50;
        end

        lambda_1 = 0;
        lambda_2 = -1;

        C1 = zeros(1, N);
        C2 = zeros(1, N);

        C1(1) = 1;
        C2(2) = 1;

        for k = 3:N
            C1(k) = -C1(k-1) / (k*(k-1)*(2*k));
            C2(k) = -C2(k-1) / ((k-1)*(k-2)*(2*k-2));
        end

        y = @(x, A, B) sum(A .* C1 .* (x.^(0:N-1)) + B .* C2 .* (x.^(1:N)));

        coef_matrix = [y(1, 1, 0), y(1, 0, 1); y(5, 1, 0), y(5, 0, 1)];
        rhs = [-10; -18];
        AB = coef_matrix \ rhs;

        y_val = y(x_val, AB(1), AB(2));
    end

    x = linspace(1, 5, 400);
    y_vals_series = arrayfun(@(val) series(val), x);
    y_vals_bessel = arrayfun(bessel_trick, x);

    % Plotting the results
    figure;
```

```
    plot(x, y_vals_series, 'DisplayName', 'Series Solution');
    hold on;
    plot(x, y_vals_bessel, 'DisplayName', 'Bessel Solution');
    xlabel('x');
    ylabel('y');
    legend;
    title('Comparison of Series and Bessel Solutions Without fsolve');
    grid on;
end
```



Published with MATLAB® R2023a