```
    % Quadratic Interpolation (only if idx+2 exists)
    if idx + 2 <= length(t_coarse)
        S_interp_quad(k) = f_quadratic(S_coarse(idx), S_coarse(idx+1), S_coarse(idx+2), t_coarse(idx), t_coarse(idx+1), t_coarse(idx+2), odd_days(k));
        I_interp_quad(k) = f_quadratic(I_coarse(idx), I_coarse(idx+1), I_coarse(idx+2), t_coarse(idx), t_coarse(idx+1), t_coarse(idx+2), odd_days(k));
        R_interp_quad(k) = f_quadratic(R_coarse(idx), R_coarse(idx+1), R_coarse(idx+2), t_coarse(idx), t_coarse(idx+1), t_coarse(idx+2), odd_days(k));
    else
        % Use linear interpolation as a fallback
        S_interp_quad(k) = S_interp_lin(k);
        I_interp_quad(k) = I_interp_lin(k);
        R_interp_quad(k) = R_interp_lin(k);
    end
end

% Compute L2 Errors
L2_error = @(V_interp, V_model) sqrt(sum((V_interp - V_model).^2) / length(V_interp));
S_error_lin = L2_error(S_interp_lin, S_fine(odd_days));
S_error_quad = L2_error(S_interp_quad, S_fine(odd_days));
I_error_lin = L2_error(I_interp_lin, I_fine(odd_days));
I_error_quad = L2_error(I_interp_quad, I_fine(odd_days));
R_error_lin = L2_error(R_interp_lin, R_fine(odd_days));
R_error_quad = L2_error(R_interp_quad, R_fine(odd_days));

error_table = table(["S_Error"; "I_Error"; "R_Error"], [S_error_lin; I_error_lin; R_error_lin], [S_error_quad; I_error_quad; R_error_quad], 'VariableNam
disp(error_table);
```

Command Window

Population	Linear	Quadratic
"S_Error"	15.949	16.016
"I_Error"	8.2041	8.2811
"R_Error"	13.68	13.711

Conclusion: Ultimately, both linear and quadratic interpolation yielded almost the same error. Typically, you see quadratic interpolation yield more accurate results due to its ability to capture the curvature of a function, whereas linear interpolation is simply connecting dots with lines.