DS284-Numerical Linear Algebra - Assignment 4

LOCHAN SURYA TEJA NEELI(06-18-01-10-22-23-1-23198)

October 24, 2023

Answer 1

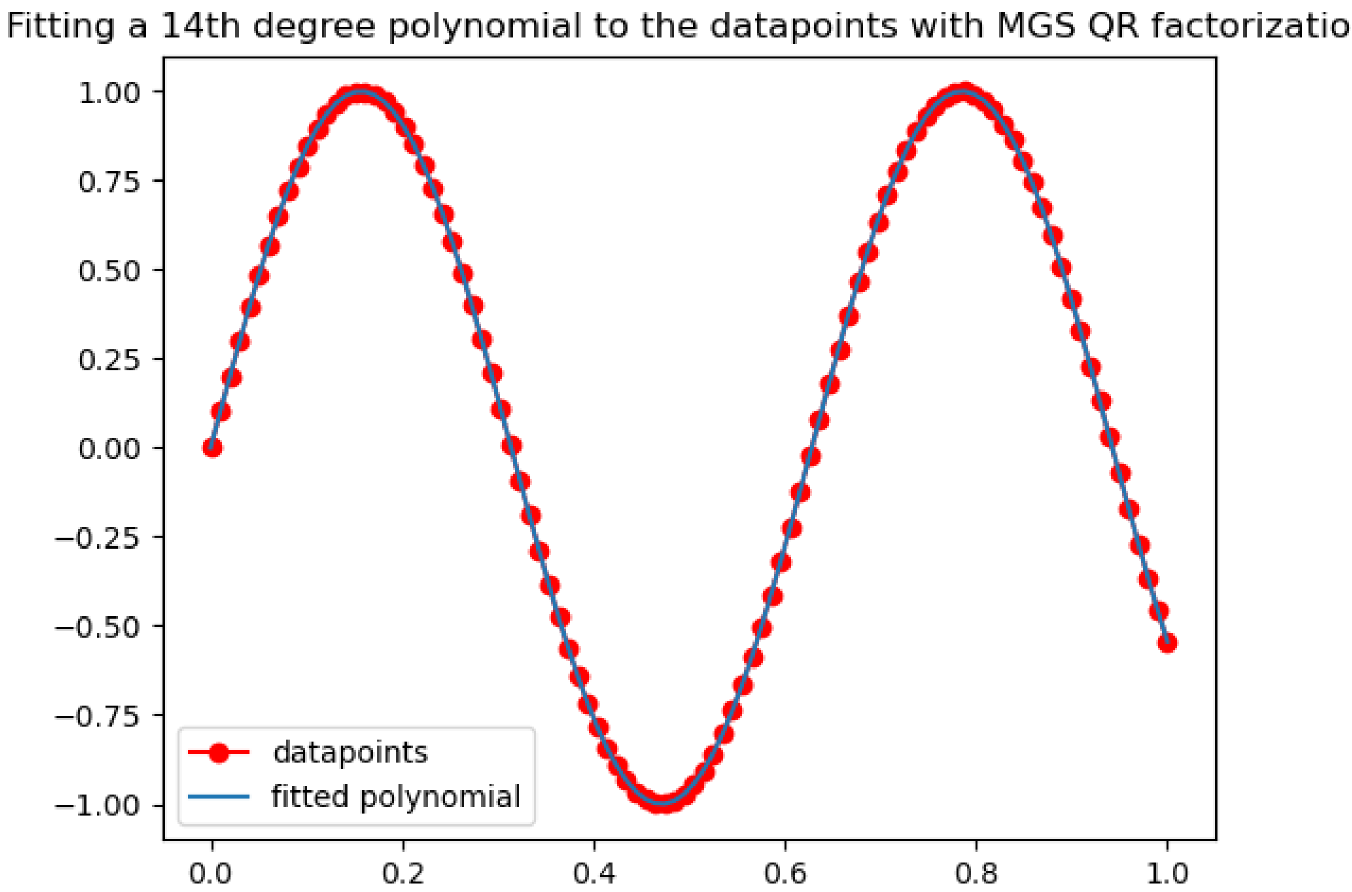


Figure 1: **(a)**

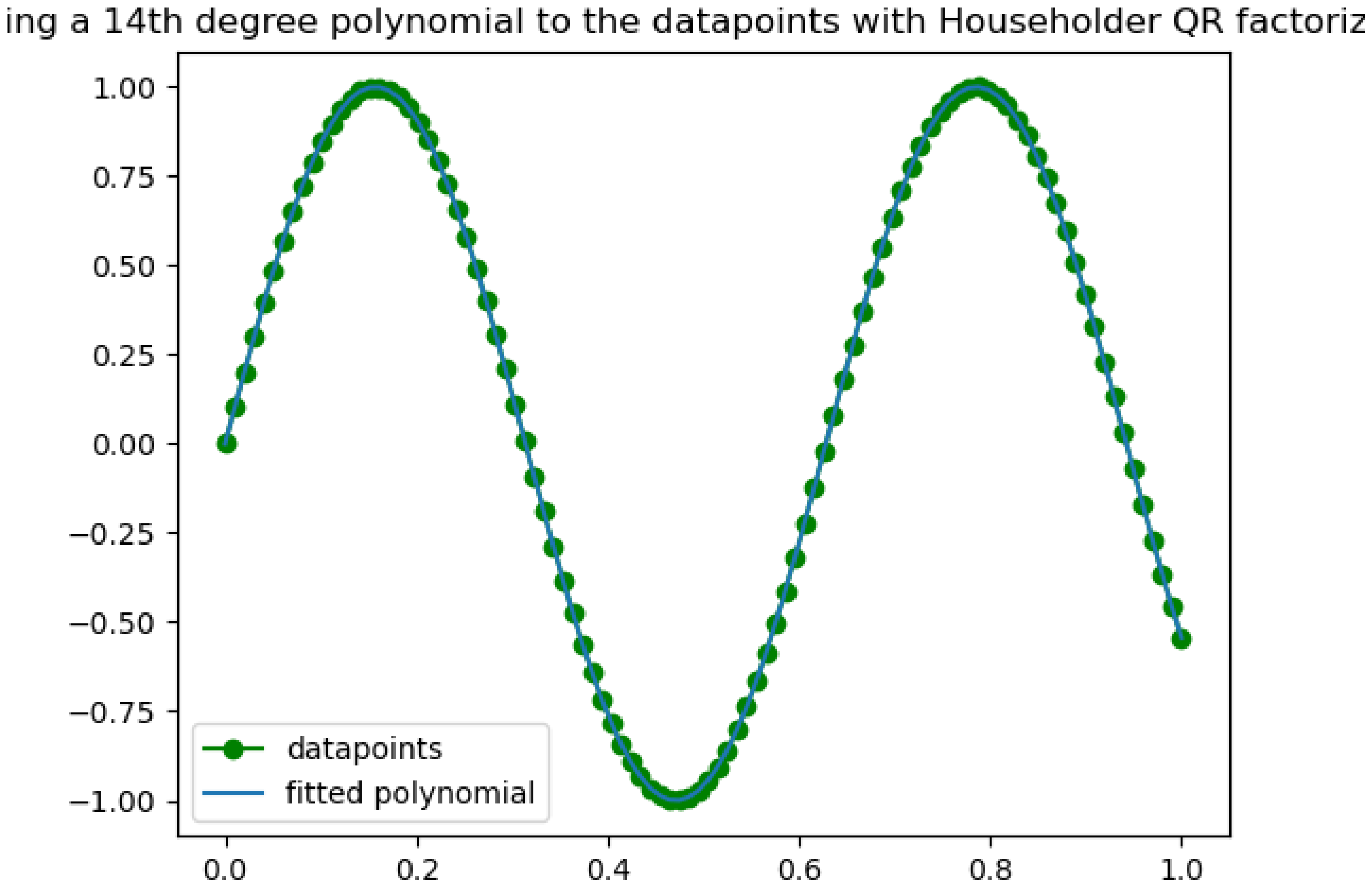
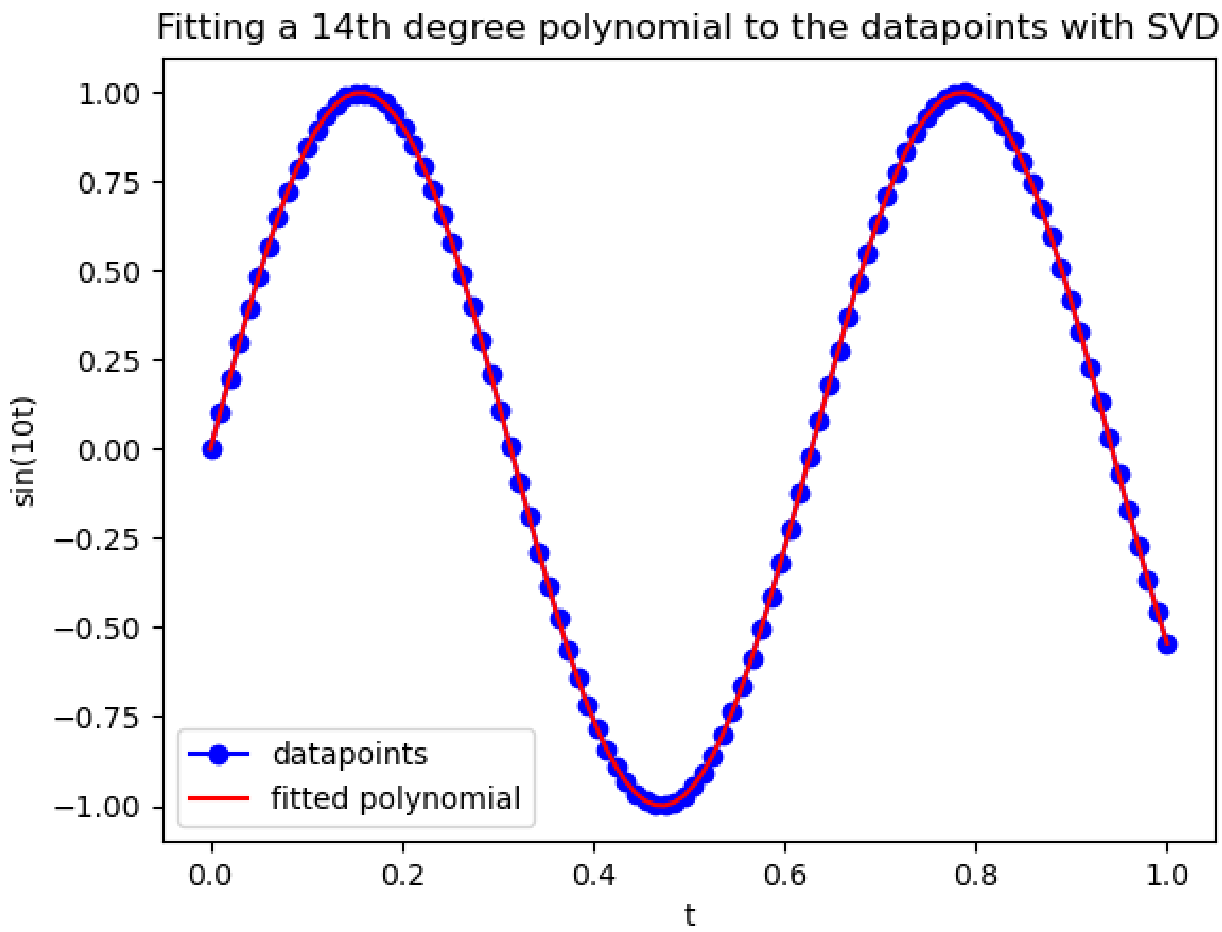
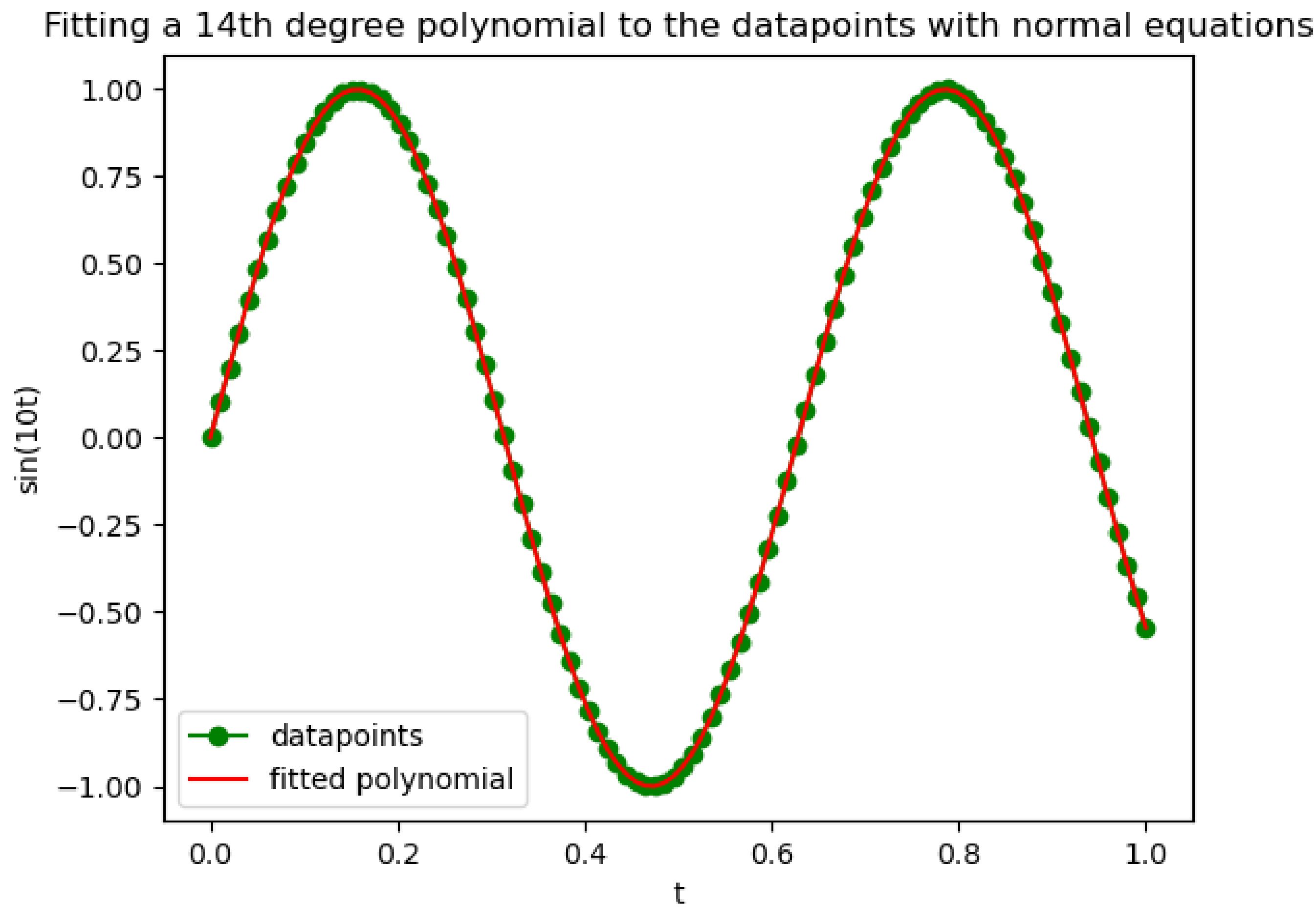

ing a 14th degree polynomial to the datapoints with Householder QR factoriz
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.0
0.2
0.4
0.6
0.8
1.0
datapoints
fitted polynomial

Figure 2: **(b)**



Coefficients of the 14 degree polynomial for the function $f(t) = sin(10t)$ approximation:

$p_{14}(x) = a_0+a_1x+a_2x^2+a_3x^3+a_4x^4+a_5x^5+a_6x^6+a_7x^7+a_8x^8+a_9x^9+a_{10}x^{10}+a_{11}x^{11}+a_{12}x^{12}+a_{13}x^{13}+a_{14}x^{14}$



Fitting a 14th degree polynomial to the datapoints with normal equations
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
sin(10t)
0.0
0.2
0.4
0.6
0.8
1.0
t
datapoints
fitted polynomial

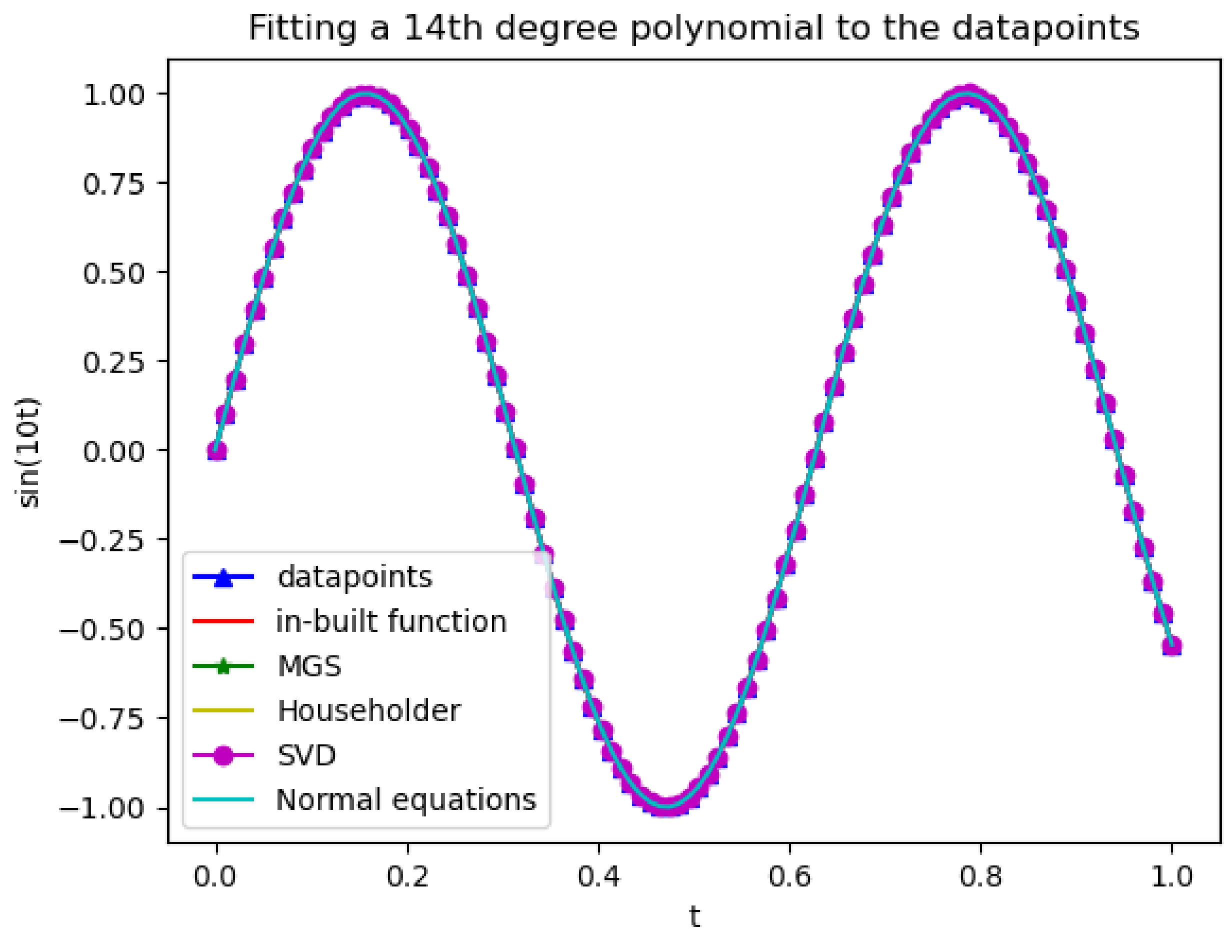
Figure 4: **(d)**

Coefficients of the 14 degree polynomial for the function $f(t) = sin(10t)$ approximation:

$p_{14}(x) = a_0+a_1x+a_2x^2+a_3x^3+a_4x^4+a_5x^5+a_6x^6+a_7x^7+a_8x^8+a_9x^9+a_{10}x^{10}+a_{11}x^{11}+a_{12}x^{12}+a_{13}x^{13}+a_{14}x^{14}$

	MGS	**Householder**	**normal equations**	**SVD**
a_0	4.13e-7	-1.604e-7	0.0000022188	-1.604e-7
a_1	9.9998424247	10.0000600749	10.0017959484	10.0000600747
a_2	0.0095821262	-0.0031492289	-0.1558430004	-0.0031492228
a_3	-166.9113404969	-166.6106191969	-162.496532603	-166.6106192929
a_4	3.4840726795	-0.2959493046	-53.2919958934	-0.2959482308
a_5	801.5976640372	830.4565467514	1219.9893792245	830.4565385493
a_6	199.3841776048	55.2238419012	-1734.0963064463	55.2238843827
a_7	-2884.4179382225	-2392.0438688348	3003.4347635999	-2392.0440199592
a_8	2979.4731872136	1801.1100507212	-9120.6421024678	1801.1104252418
a_9	-4503.7856651089	-2508.830344659	12323.3515274771	-2508.8309956998
a_{10}	12907.8718186067	10530.107804164	-2596.3576586784	10530.1085943036
a_{11}	-18825.4162729622	-16874.0560276336	-9907.4873359055	-16874.0566831362
a_{12}	13836.7980869608	12787.2865803221	-9907.4873359055	12787.2869345805
a_{13}	-5133.4271555276	-4800.4499588965	-4908.1733685353	-4800.4500712968
a_{14}	774.7959199589	727.5610133192	832.8919068456	727.5610292058

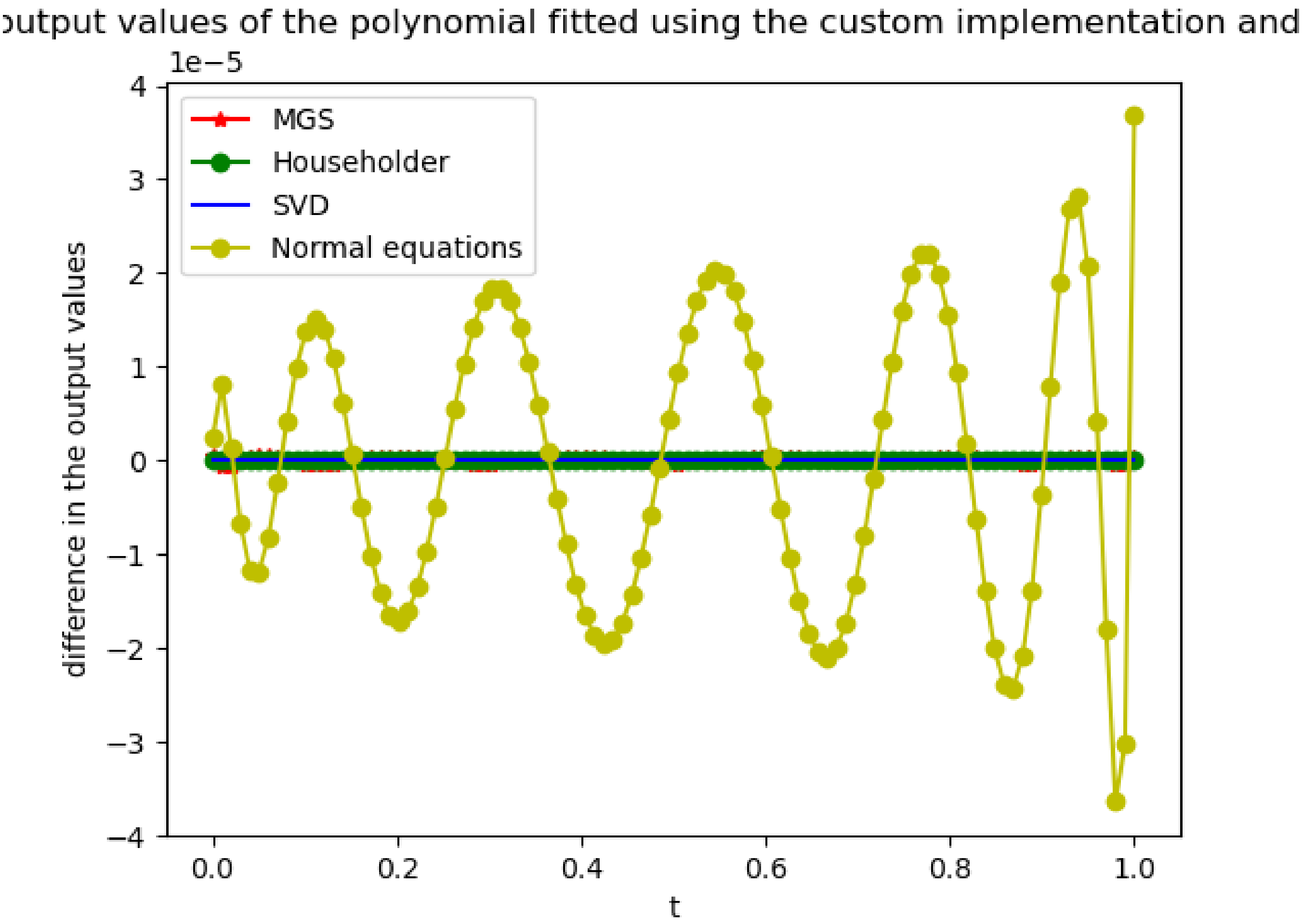
Table 1: Coefficients of the fitted polynomial.



As you can see from the above graph, almost all of the methods approximate the original function accurately upto 4 decimal places after the decimal. So, you cannot say which one is more accurate on a graphical level.

Comparing the models based on the error:

The difference is calculated by taking the difference between the output values obtained from the custom implemented algorithm and the output values obtained from the in-built function.



You can see from the graph that Normal Equations performed the worst among the implemented algorithms.
On comparing the obtained solutions of the different models with solution obtained using the in-built function, **"np.linalg.lstsq()"**, I got the maximum difference, calculated in terms of 2-norm, in the case of normal equations:
error in MGS is 2.4394222148706133e-06
error in Householder is 2.6622356547742706e-11
error in SVD is 6.563324947367568e-12
error in Normal equations is 0.0001522964946523137
Condition Number of the matrix A.T @ A =1.0489572501244311e+18
This is ill-conditioned matrix and thus, the problem of solving least squares through Normal Equations is unstable. This is the reason that the error is the highest in the normal equations case.
So, **SVD** > **Householder** > **MGS** > **Normal Equations** ('>' means 'better than.')