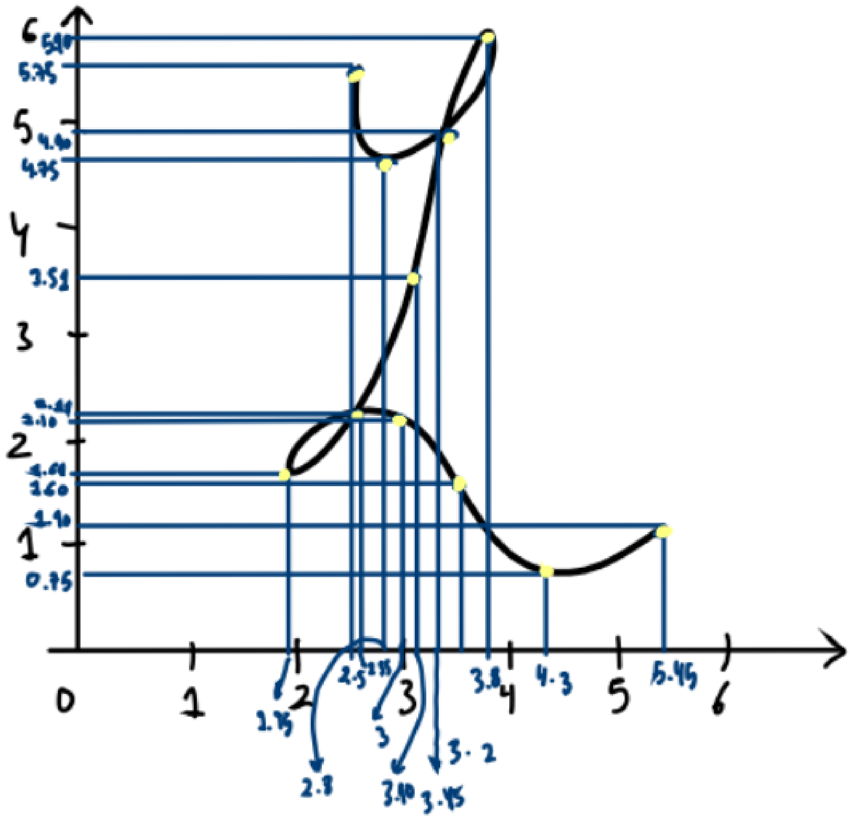
PROJECT 2 -CS514

FALL 2022

ELENA DE TOLEDO HERNANDEZ

FIRST PART: SECTION 6.4, PROBLEM 7

For this problem, I decided to use the letter L. Find below my drawing of said letter and the 11 interpolation knots I extracted from the picture.



After extracting the points, I created to matrices; A and E, both had 11 rows and 2 columns. Because I was asked to compute Sx and Sy, I ordered in A the knots in ascending order in the first row, and it's associated y was placed on the same row on the coordinate y. For matrix E, the first column had the knots in ascending order of y, and in the other column the associated value for x.

I used the same tile to plot the knots, Sx and Sy.

For obtaining both Sx and Sy, I followed the same method of natural cubic splines. Following the instructions of the book, I computed the distances between each of the nodes, and stored them in a vector h. I also computed the vector u using the formula u(i) = 2*(h(i+1)+h(i)), and I computed the matrix 'mmatrix'. Finally, I obtained the vector b, following the formula b(i) = (6/h(i))*(y(i+1)-y(i)) and I used this vector to compute v(i) = b(i)-b(i-1).

When I had both the matrix and the vector, I used the command linsolve of matlab to obtain the solution vector z.

Once I had z, I computed the vectors A,B,C (I used different names because the letters where already in use). The formulas are the following:

A(i) = (1/(6*h(i)))*(z(i+1)-z(i));

B(i) = z(i)/2;

C(i) = (-h(i)/6)*z(i+1)-(h(i)/3)*z(i)+(1/h(i))*(y(i+1)-y(i));

I ended up with the different splines, which were computed as follows:

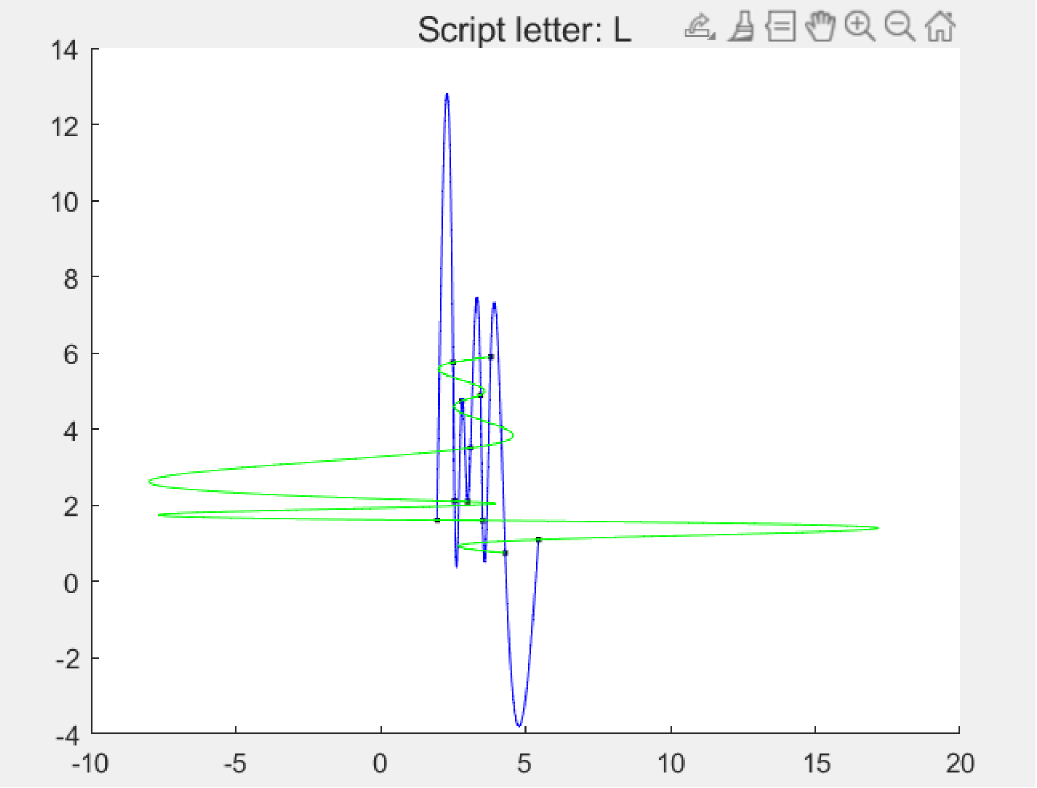
F(x) = y(i)+(x-x(i))*(C(i)+(x-x(i))*(B(i)+(x-x(i))*A(i)));

Note that the notation used in my code is different because I used a matrix to store the points, and because I had to use letter D to denote the vector that's named A in the book because that variable name was already in use.

To plot the splines, I used the function linspace of matlab, which returns a row vector of 100 evenly spaced points between each of the nodes xi, xi+1.

I made the plot of the function at each of these points, and this was the result.

In black, see the points I extracted from the picture. In blue, the representation of Sx, and in green, the representation of Sy.



My conclusion is that the obtained plots are not very accurate because I used a modest number of points from the original drawing to obtain the solution. My guess is that I had I chosen a much bigger number of points, the approximation would be much better.

SECOND PART: SECTION 6.4, PROBLEM 8

PART A

For Lagrange.

Table of results:

f(x)	p(x)	f(x)-p(x)
0.142857	0.142857	-0.000000
0.155547	0.005241	0.150307
0.169884	0.172917	-0.003033
0.186137	0.198621	-0.012484
0.204626	0.203923	0.000703
0.225728	0.223863	0.001865
0.249889	0.250086	-0.000198
0.277622	0.278041	-0.000420
0.309519	0.309448	0.000071
0.346241	0.346110	0.000131
0.388491	0.388524	-0.000033
0.436964	0.437017	-0.000053
0.492240	0.492222	0.000019
0.554602	0.554575	0.000028
0.623748	0.623760	-0.000013
0.698380	0.698397	-0.000018
0.775727	0.775716	0.000010
0.851139	0.851126	0.000013
0.918078	0.918087	-0.000010
0.968876	0.968887	-0.000011
0.996443	0.996433	0.000010
0.996443	0.996433	0.000010
0.968876	0.968887	-0.000011

0.918078	0.918087	-0.000010
0.851139	0.851126	0.000013
0.775727	0.775716	0.000010
0.698380	0.698397	-0.000018
0.623748	0.623760	-0.000013
0.554602	0.554575	0.000028
0.492240	0.492222	0.000019
0.436964	0.437017	-0.000053
0.388491	0.388524	-0.000033
0.346241	0.346110	0.000131
0.309519	0.309448	0.000071
0.277622	0.278041	-0.000420
0.249889	0.250086	-0.000198
0.225728	0.223863	0.001865
0.204626	0.203923	0.000703
0.186137	0.198621	-0.012484
0.169884	0.172917	-0.003033
0.155547	0.005241	0.150307

For this method, I computed the required distance to have 21 equispaced and used it to compute the nodes. I later computed its images (y) and the coefficients of the different li(x). Later, I computed the polynomials l(x) and finally computed the sum of all the yi(x)*li(x). I displayed the results, adding one column for the difference between the function and the approximation.

PART B

f(x)	p(x)	f(x)-p(x)
0.142857	1872.457143	-1872.314286
0.155547	-128785630.108417	128785630.263964
0.169884	2590104.259254	-2590104.089371
0.186137	10677030.103611	-10677029.917475
0.204626	-598724.356058	598724.560684
0.225728	-1611865.130747	1611865.356476
0.249889	168642.046051	-168641.796163
0.277622	375248.346907	-375248.069286

0.309519	-60180.663393	60180.972912
0.346241	-125805.612802	125805.959043
0.388491	26009.991476	-26009.602986
0.436964	58651.029794	-58650.592830
0.492240	-12266.010678	12266.502918
0.554602	-36686.446303	36687.000906
0.623748	4454.996655	-4454.372908
0.698380	28897.272819	-28896.574439
0.775727	2802.305596	-2801.529869
0.851139	-25413.994508	25414.845647
0.918078	-11970.812873	11971.730950
0.968876	20043.719233	-20042.750357
0.996443	21078.866786	-21077.870343
0.996443	-10443.212939	10444.209382
0.968876	-25808.681632	25809.650508
0.918078	972.378150	-971.460072
0.851139	28650.242323	-28649.391184
0.775727	6642.854291	-6642.078564
0.698380	-35703.708701	35704.407080
0.623748	-15322.125645	15322.749393
0.554602	56339.079673	-56338.525071
0.492240	31527.936664	-31527.444424
0.436964	-119895.127968	119895.564932
0.388491	-73865.610327	73865.998818
0.346241	355963.403677	-355963.057436
0.309519	217141.666529	-217141.357010
0.277622	-1522986.046807	1522986.324428
0.249889	-856611.405206	856611.655095
0.225728	10029256.605465	-10029256.379737
0.204626	4874829.082671	-4874828.878046
0.186137	-119518407.282664	119518407.468801
0.169884	-45077171.039972	45077171.209856

0.155547	4624030802.662806	-4624030802.507258

Because the nodes were given, I just had to apply Lagrange interpolation to said nodes. After doing so, I displayed the results.

PART C

f(x)	p(x)	f(x)-p(x)
0.142857	16.320481	-16.177624
0.155547	15.142291	-14.986744
0.169884	14.023603	-13.853719
0.186137	12.962853	-12.776716
0.204626	11.958478	-11.753852
0.225728	11.008914	-10.783185
0.249889	10.112597	-9.862708
0.277622	9.267962	-8.990340
0.309519	8.473447	-8.163927
0.346241	7.727487	-7.381246
0.388491	7.028518	-6.640027
0.436964	6.374977	-5.938013
0.492240	5.765299	-5.273059
0.554602	5.197920	-4.643318
0.623748	4.671278	-4.047530
0.698380	4.183807	-3.485427
0.775727	3.733944	-2.958218
0.851139	3.320126	-2.468986
0.918078	2.940787	-2.022709
0.968876	2.594365	-1.625489
0.996443	2.279295	-1.282851
0.996443	1.994013	-0.997570
0.968876	1.736956	-0.768080
0.918078	1.506560	-0.588482
0.851139	1.301260	-0.450121
0.775727	1.119494	-0.343767

0.698380	0.959696	-0.261316
0.623748	0.820303	-0.196556
0.554602	0.699752	-0.145149
0.492240	0.596478	-0.104238
0.436964	0.508917	-0.071953
0.388491	0.435506	-0.047015
0.346241	0.374680	-0.028439
0.309519	0.324877	-0.015357
0.277622	0.284531	-0.006909
0.249889	0.252079	-0.002190
0.225728	0.225957	-0.000228
0.204626	0.204601	0.000025
0.186137	0.186447	-0.000311
0.169884	0.169932	-0.000048
0.155547	0.153491	0.002056

I used a method quite similar to the one applied to the previous problem in this project. I obtained the results and ordered them in the table above.

CONCLUSIONS

I believe I made have done something wrong with Chebyshev interpolation, as it is giving me a huge error when it should be giving the least error of them all because the nodes have been carefully chosen.

Between Lagrange and the cubic splines, I believe Lagrange has a smaller error, although they seem quite similar. I acknowledge this fact to the greater degree the Lagrange interpolation has.