PHYS 581: Assignment #1

Scott Salmon UCID: 30093320

January 28th, 2025

Question 1 (10 points)

Solutions for questions 1a) and 1b) are both found in the submitted Mathematica workbook in the top cell. I used Mathematica functions and made comments throughout the workbook to describe every step along the way.

I found the solution using both the recurrence method and the close-form expression, and then verified that they were equivalent to each other and also equivalent to the built in Chebyshev function in Mathematica. An image of the output is shown below:

```
Out[417]= {1, x, -1 + 2 x^2, -3 x + 4 x^3, 1 - 8 x^2 + 8 x^4, 5 x - 20 x^3 + 16 x^5, -1 + 18 x^2 - 48 x^4 + 32 x^6, -7 x + 56 x^3 - 112 x^5 + 64 x^7,
          1 - 32 x^2 + 160 x^4 - 256 x^6 + 128 x^8, 9 x - 120 x^3 + 432 x^5 - 576 x^7 + 256 x^9, -1 + 50 x^2 - 400 x^4 + 1120 x^6 - 1280 x^8 + 512 x^10}

Out[418]= {1, x, Cos[2 ArcCos[x]], Cos[3 ArcCos[x]], Cos[4 ArcCos[x]], Cos[5 ArcCos[x]],
          Cos[6 ArcCos[x]], Cos[7 ArcCos[x]], Cos[8 ArcCos[x]], Cos[9 ArcCos[x]], Cos[10 ArcCos[x]]}

Out[419]= True

Out[421]= True
```

Summary of outputs in image:

- Output 417 is the Chebyshev Polynomials up to n=10 using the recurrence relation.
- Output 418 is Chebyshev Polynomials up to n=10 using the closed form expression.
- Output 419 is the output that the solutions for 1a and 1b are equivalent.
- Output 421 is the output that tells us that the Recurring form (and therefore also the closed form) are equivalent to the built in Chebyshev Polynomial function.

Question 2 (5 points)

The solution for 2 can also be found in the second cell of the submitted Mathematica workbook. I used Mathematica functions and made comments throughout the workbook to describe every step along the way.

The following table is the given output using the recurrence relation:

Out[465]//TableForm=

n	T_n(x)	T_n'(x)	T_n''(x)
0	1	0	0
1	x	1	0
2	$-1+2x^2$	$4x$	4
3	$-3x+4x^3$	$-3+12x^2$	$24x$
4	$1-8x^2+8x^4$	$-16x+32x^3$	$-16+96x^2$
5	$5x-20x^3+16x^5$	$5-60x^2+80x^4$	$-120x+320x^3$
6	$-1+18x^2-48x^4+32x^6$	$36x-192x^3+192x^5$	$36-576x^2+960x^4$
7	$-7x+56x^3-112x^5+64x^7$	$-7+168x^2-560x^4+448x^6$	$336x-2240x^3+2688x^5$
8	$1-32x^2+160x^4-256x^6+128x^8$	$-64x+640x^3-1536x^5+1024x^7$	$-64+1920x^2-7680x^4+7168x^6$
9	$9x-120x^3+432x^5-576x^7+256x^9$	$9-360x^2+2160x^4-4032x^6+2304x^8$	$-720x+8640x^3-24192x^5+18432x^7$
10	$-1+50x^2-400x^4+1120x^6-1280x^8+512x^{10}$	$100x-1600x^3+6720x^5-10240x^7+5120x^9$	$100-4800x^2+33600x^4-71680x^6+46080x^8$

The next table is the equivalent output from the table above, but using the closed form expression instead of the recurrence relation:

Out[12]//TableForm=

n	T_n(x)	T_n'(x)	T_n''(x)
0	1	0	0
1	x	1	0
2	$\text{Cos}[2\,\text{ArcCos}[x]]$	$\frac{2\,\text{Sin}[2\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{4\,\text{Cos}[2\,\text{ArcCos}[x]]}{1-x^2}+\frac{2x\,\text{Sin}[2\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
3	$\text{Cos}[3\,\text{ArcCos}[x]]$	$\frac{3\,\text{Sin}[3\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{9\,\text{Cos}[3\,\text{ArcCos}[x]]}{1-x^2}+\frac{3x\,\text{Sin}[3\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
4	$\text{Cos}[4\,\text{ArcCos}[x]]$	$\frac{4\,\text{Sin}[4\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{16\,\text{Cos}[4\,\text{ArcCos}[x]]}{1-x^2}+\frac{4x\,\text{Sin}[4\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
5	$\text{Cos}[5\,\text{ArcCos}[x]]$	$\frac{5\,\text{Sin}[5\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{25\,\text{Cos}[5\,\text{ArcCos}[x]]}{1-x^2}+\frac{5x\,\text{Sin}[5\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
6	$\text{Cos}[6\,\text{ArcCos}[x]]$	$\frac{6\,\text{Sin}[6\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{36\,\text{Cos}[6\,\text{ArcCos}[x]]}{1-x^2}+\frac{6x\,\text{Sin}[6\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
7	$\text{Cos}[7\,\text{ArcCos}[x]]$	$\frac{7\,\text{Sin}[7\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{49\,\text{Cos}[7\,\text{ArcCos}[x]]}{1-x^2}+\frac{7x\,\text{Sin}[7\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
8	$\text{Cos}[8\,\text{ArcCos}[x]]$	$\frac{8\,\text{Sin}[8\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{64\,\text{Cos}[8\,\text{ArcCos}[x]]}{1-x^2}+\frac{8x\,\text{Sin}[8\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
9	$\text{Cos}[9\,\text{ArcCos}[x]]$	$\frac{9\,\text{Sin}[9\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{81\,\text{Cos}[9\,\text{ArcCos}[x]]}{1-x^2}+\frac{9x\,\text{Sin}[9\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$
10	$\text{Cos}[10\,\text{ArcCos}[x]]$	$\frac{10\,\text{Sin}[10\,\text{ArcCos}[x]]}{\sqrt{1-x^2}}$	$-\frac{100\,\text{Cos}[10\,\text{ArcCos}[x]]}{1-x^2}+\frac{10x\,\text{Sin}[10\,\text{ArcCos}[x]]}{(1-x^2)^{3/2}}$

Question 3 (20 points)

The first thing I did for Question 3 was make a script in Mathematica to give me the expected results from my C code so I could verify if my program was working properly. To do this, I used Mathematica functions and made comments through the workbook to describe every step along the way.

Shown below is a screen shot of what I found the final answers to of what the T_{10}, T'_{10} and T''_{10} values are with 100 points equally spaced from -1 to 1:

Out[21]//TableForm=

x	T10(x)	T10'(x)	T10''(x)
-1.	1.	Undefined	Undefined
-0.98	-0.419189	-45.6236	2187.63
-0.96	-0.954251	-10.6788	1347.92
-0.94	-0.942732	9.77655	730.956
-0.92	-0.632852	19.756	293.683
-0.9	-0.200747	22.4746	-0.801843
-0.88	0.234741	20.4655	-183.882
-0.86	0.599501	15.6846	-282.023
-0.84	0.85354	9.60266	-317.324
-0.82	0.98213	3.28817	-308.026
-0.8	0.988497	-2.52072	-268.981
-0.78	0.887862	-7.3526	-212.082
-0.76	0.702675	-10.9476	-146.655
-0.74	0.458869	-13.2099	-79.8223

Now, in question3.c, I wrote 3 functions that simply returned the expressions determined in Question 2 for the n=10 closed form expression of the Chebyshev polynomials. They all take in a "x" value and evaluate the equation.

In the main function, I divided the distance from x=-1 to x=1 into "m" number of steps, and then I ran a loop that evaluated T_{10}, T'_{10} and T''_{10} at each of these x-values. A screenshot of the final results are shown in the screenshot below:

```
x              T10(x)          T10'(x)         T10''(x)
-----------------------------------------------------------------
-1.000000      1.000000        -inf    nan
-0.980000      -0.419189       -45.623636      2187.628396
-0.960000      -0.954251       -10.678811      1347.917850
-0.940000      -0.942732       9.776547        730.955838
-0.920000      -0.632852       19.755981       293.682634
-0.900000      -0.200747       22.474552       -0.801843
-0.880000      0.234741        20.465511       -183.881942
-0.860000      0.599501        15.684565       -282.023157
-0.840000      0.853540        9.602663        -317.324148
-0.820000      0.982130        3.288174        -308.025995
-0.800000      0.988497        -2.520719       -268.980787
-0.780000      0.887862        -7.352600       -212.081542
...
```

As we can see, the C code aligns with the results determined in Mathematica, which means the algorithm in question3.c is working as expected.

Question 4 (20 points)

For question4.c, instead of having functions that gave an exact analytical solution, I instead had to use finite difference methods to solve T'_{10} and T''_{10}. To do this, for the majority of points I used the central finite difference equation:

$$f(x) = cos(10cos^{-1}(x)) = T_{10}(x)$$

$$f'(x) = \frac{T_{10}(x+a) - T_{10}(x-a)}{2a}$$

$$f''(x) = \frac{T_{10}(x+a) - 2 \cdot T_{10}(x) + T_{10}(x-a)}{a^2}$$

Unfortunately, at the boundaries we cannot use the CFD method, as there is no "$(x+a)$" or "$(x-a)$". At these points, we need to use either the forwards finite difference or backwards finite difference methods:

$$f'_{forwards}(x) = \frac{T_{10}(x+a) - T_{10}(x)}{a}$$

$$f'_{backwards}(x) = \frac{T_{10}(x) - T_{10}(x-a)}{a}$$

$$f''_{forwards}(x) = \frac{T_{10}(x+2a) - 2 \cdot T_{10}(x+a) + T_{10}(x)}{a^2}$$

$$f''_{backward}(x) = \frac{T_{10}(x) - 2 \cdot T_{10}(x-a) + T_{10}(x-2a)}{a^2}$$

Other than changing the initial functions to apply this differentiation method (instead of simply returning the analytical result), the algorithm in question4.c is essentially the same as in question3.c.

Question 5 (20 points)

For this question, I basically copy pasted the functions from question3.c and question4.c because we need to use both the analytical and finite difference methods.

In my main function, I initialize a list of a various number of points to compare the two methods with different spacing, "a" values. I then essentially ran the same algorithm I created for question3.c and question4.c many times. Each time the loop ran, it would find the difference between the analytical and numerical solutions, and then take the absolute value of this difference and store it in variables denoted as "averageprime" and "averagedoubleprime".

These variables then were printed at the end of every loop. I did this so I could find the amount of error present every time the loop ran itself with a different spacing constant. I ignored the points at $i = 0$ and $i = m - 1$ in the comparison process, as those are the boundary points and the analytic functions return a

"nan" (as when $x = \pm 1$, there is a division by zero).

The image shown below shows the final results given by question5.c:

# of Points	Point Spacing (a)	Average Error T10'(x)	Average Error T10''(x)
10	0.222222	3.068475	57.367208
15	0.142857	4.843842	50.194157
20	0.105263	3.863943	35.174128
25	0.083333	2.684250	25.156882
30	0.068966	2.162683	19.409288
35	0.058824	1.679828	14.979475
40	0.051282	1.360720	11.967991
45	0.045455	1.119144	9.785257
50	0.040816	0.928909	8.086310
75	0.027027	0.453909	3.871511
100	0.020202	0.267358	2.258322
125	0.016129	0.175757	1.475986
...			
350	0.005731	0.023983	0.198512
500	0.004008	0.011883	0.098114
999	0.002004	0.003015	0.024826

This aligns with our expected results that as the point spacing constant, "a", becomes smaller, the average error in both the $T'_{10}(x)$ and $T''_{10}(x)$ calculations also become significantly smaller.

Question 6 (15 points)

Back in Mathematica, I harvested the data from question5.c as shown in the table. I then plotted the data and used the "LinearModelFit" function to do a quadratic fit on the data.

The reason I did this was because due to the way that central finite difference is defined, the truncation error should be approximately proportional to a^2. Here's why:

1. The finite difference method uses Taylor Series expansion to approximate derivatives. The following are the defined Taylor Series expansions for $f(x+a)$ and $f(x-a)$ respectively:

$$f(x+a) = f(x) + af'(x) + \frac{a^2}{2}f''(x) + \frac{a^3}{6}f'''(x) + ...$$
$$f(x-a) = f(x) - af'(x) + \frac{a^2}{2}f''(x) - \frac{a^3}{6}f'''(x) + ...$$

2. Now, if we subtract $f(x-a)$ from $f(x+a)$, all of the even numbered terms will disappear and odd numbered terms will double:

$$f(x+a) - f(x-a) = 2af'(x) + \frac{2a^3}{6}f'''(x) + ...$$

3. Divide both sides by $2a$:

$$\frac{f(x+a)-f(x-a)}{2a} = f'(x) + \frac{a^2}{6}f'''(x) + ...$$

(notice that the LHS is the CFD formula now, and the RHS has the first derivative by itself, which is exactly what we want)

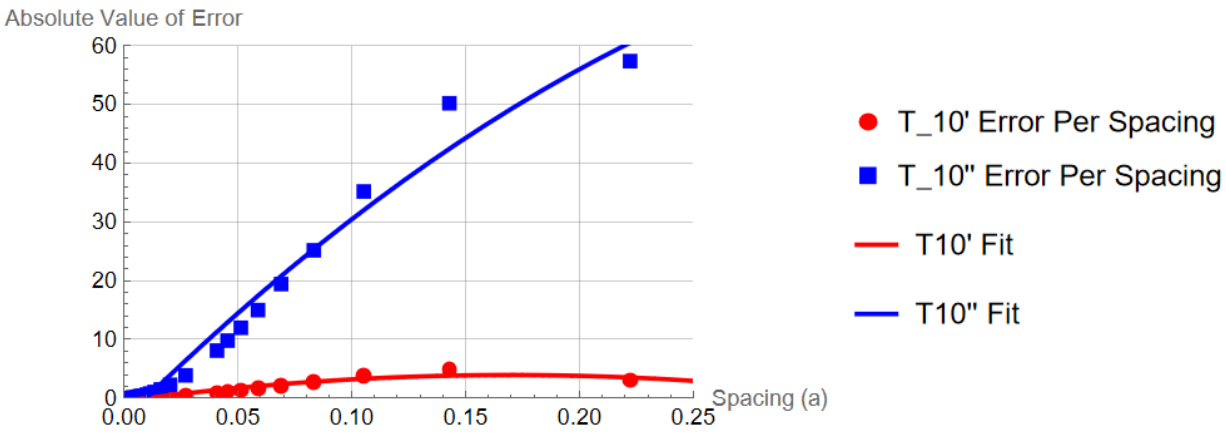
4. From this, we can see that every term past $f'(x)$ contributes to the error of the CFD. The leading term of this error is $\frac{a^2}{6}f'''(x)$, which scales as a^2 changes. Higher order terms (i.e. a^4, a^6, etc. will be much smaller than a^2 for small "a" values.

5. Therefore, we know that the error present in our central difference model is going to be approximately proportional to a^2. This is what is meant when it's said that CFD is considered a second order method.

6. Using this exact same train of logic, we can also determine that the leading term of forwards/backwards finite difference is only "a", as the "odd" terms never get canceled out in Step #2. Therefore, for forwards/backwards difference, the expected truncation error will be proportional to "a" instead of "a^2". This makes sense as forwards/backwards difference is only considered as a "first" order method.

With that explanation out of the way, here is the generated plots of error as a function of the spacing constant, "a", along with the lines of best fit:

Out[379]=



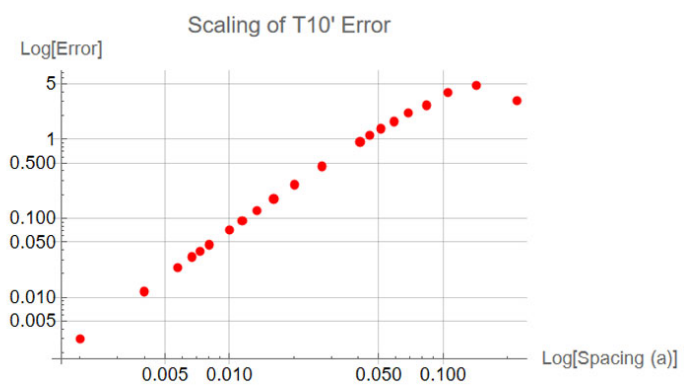
I then calculated the R^2 values of these lines of best fit using Mathematica again. The closer these values are to 1.00, the better the fit is to the data. A screenshot of the resulting output is shown below:

```
Equation for T10' error fit:FittedModel[ -0.503 + 52.2 x - 154. x^2 ]

R^2 value for T10' error fit: 0.930433

Equation for T10'' error fit: FittedModel[ -3.67 + 384. x - 429. x^2 ]

R^2 value for T10'' error fit: 0.974511
```

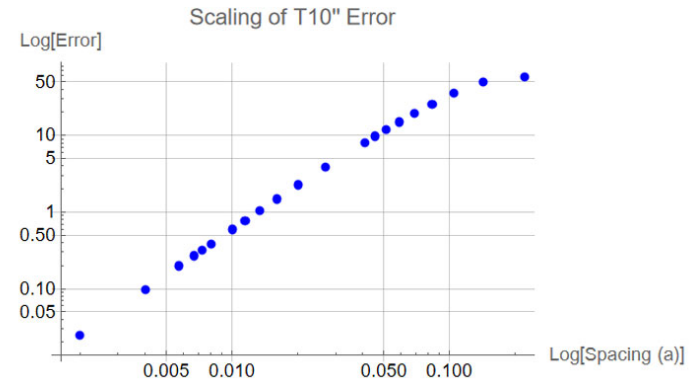
As these numbers are both above 0.90, I was fairly satisifed with these lines of best fit. However, I wanted to go even further to ensure that my model is correct. As discussed above, if the truncation error is approximately proportional to a^2, then if we were to take the logarithm of the given plot, then we should see a linear plot with a slope of approximately 2:

$$\text{Error} = a^2$$
$$log(\text{Error}) = log(a^2)$$
$$log(\text{Error}) = 2log(a)$$

So I took the logarithm of the Error and spacing constants and then found the resulting slopes, as shown in the image below:



Slope for T10' errors: 1.65495



Slope for T10'' errors: 1.74296

Fortunately, from our data shown in the log figures, we can see that the change in log(error) is mostly linear and its slope is somewhat close to 2, meaning that our change in error does line up with the expected truncation error results from the central finite difference model.

A few reasons why it's not exactly 2 are likely due to a combination of using forward/backwards difference at boundaries (expected truncation error of forward/backwards difference is "a", meaning the expected slope of the logarithm is only 1), and also some numerical noise that becomes more and more present whenever a approaches 0. We also see in both data sets that the larger "a" values are the biggest outliers, and if they were to be "removed", the line would likely be even closer to 2.

So, in combination with the R^2 values for our line of best fit being over 0.90, the slope of the logarithm of the error being approximately equal to 2, and then the obvious result of error decreasing significantly as "a" decreases, we can safely say that the dependence of error on "a" does indeed follow expectations.