Blake Van Dyken

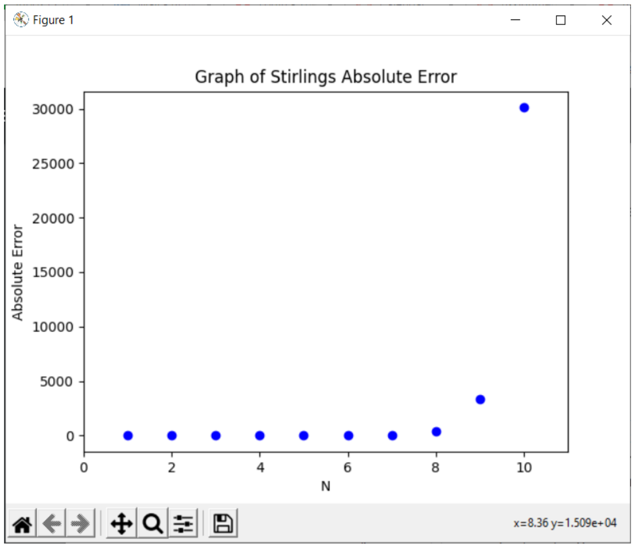
Assignment 4

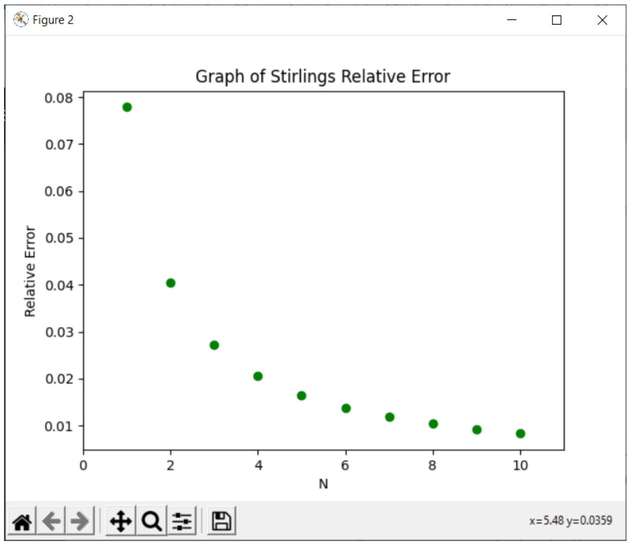
Methods Used:

- **Stirlings Approximation for factorial function**
  - $$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$
  - Used this for n = 1,2,3,...10
- **Approximated pi/4**
  - $$\frac{\pi}{4} = \tan^{-1}(1) = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$$
  - I did this for up to 10,000 terms
- **Chudnovsky Approx.**
  - $$x = \sqrt{2}$$ $$\pi_{approx} = 2 + \sqrt{2}$$ $$y = 2^{1/4}$$
  - $$x = \frac{1}{2}\left(\sqrt{x} + \frac{1}{\sqrt{x}}\right)$$ $$\pi_{approx} = \pi_{approx}\left(\frac{x+1}{y+1}\right)$$ $$y = \frac{y\sqrt{x} + \frac{1}{\sqrt{x}}}{y+1}$$
  - I did this for pi_10th terms because python could not support any more decimal places and thus the result was always the same after around 10 iterations
- I used the following formula below to calculate my errors

- **ABS Error** = True value - computed value
- **Relative Error** = True value - computed value / True value

Results and Figures:





Looking at the first graph, it shows us that our absolute error grows as n increases because we have to approximate larger and larger factorial values, although our error is small compared to the larger values because the actual value is larger compared to other values of computed values, we get an increasing graph. The second graph is of the relative error of Stirling's factorial approximation. This graph decreases because the larger values we approximate the smaller the error becomes. The error computed is small compared to the actual value of n! for larger values, but for the smaller factorials our relative error is greater; thus, we get a shrinking graph as n increases.

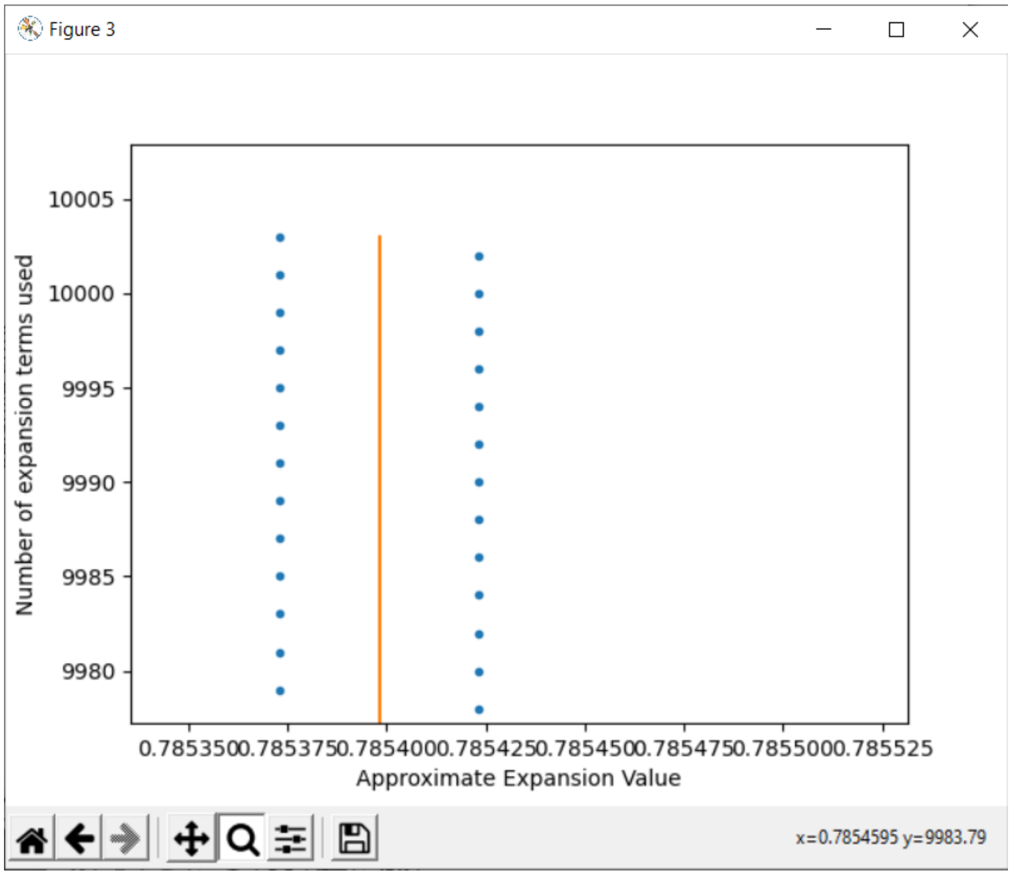


Figure 3
10005
10000
9995
9990
9985
9980
Number of expansion terms used
0.785350
0.785375
0.785400
0.785425
0.785450
0.785475
0.785500
0.785525
Approximate Expansion Value
x=0.7854595 y=9983.79

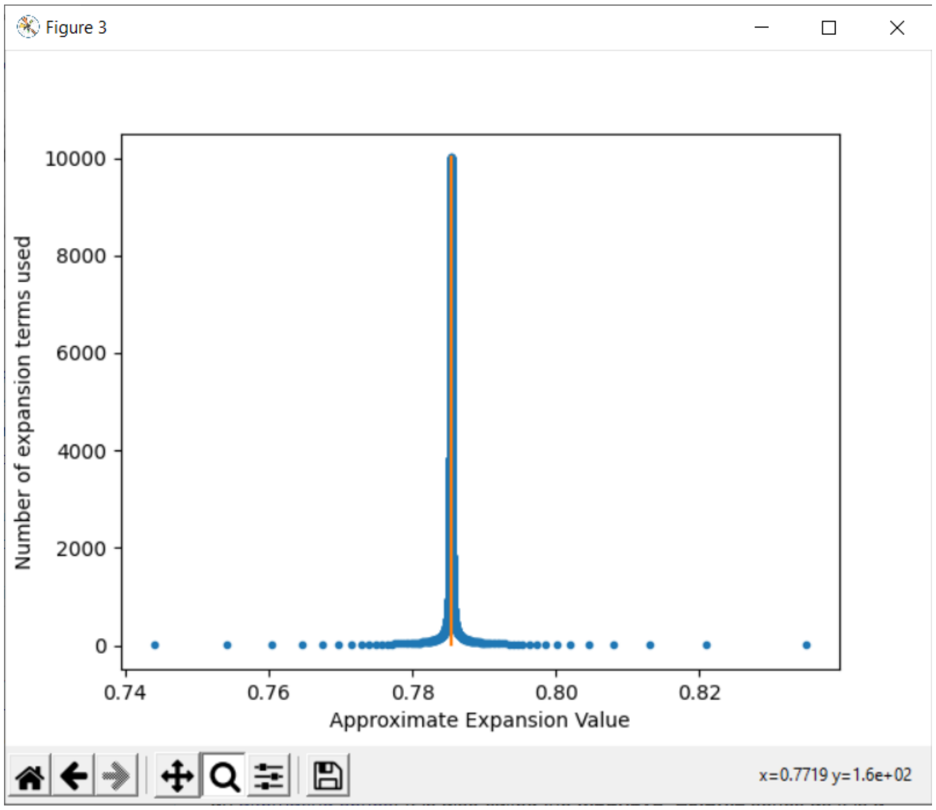
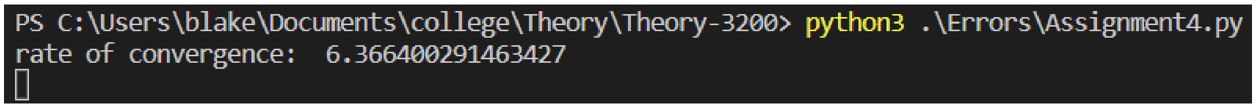
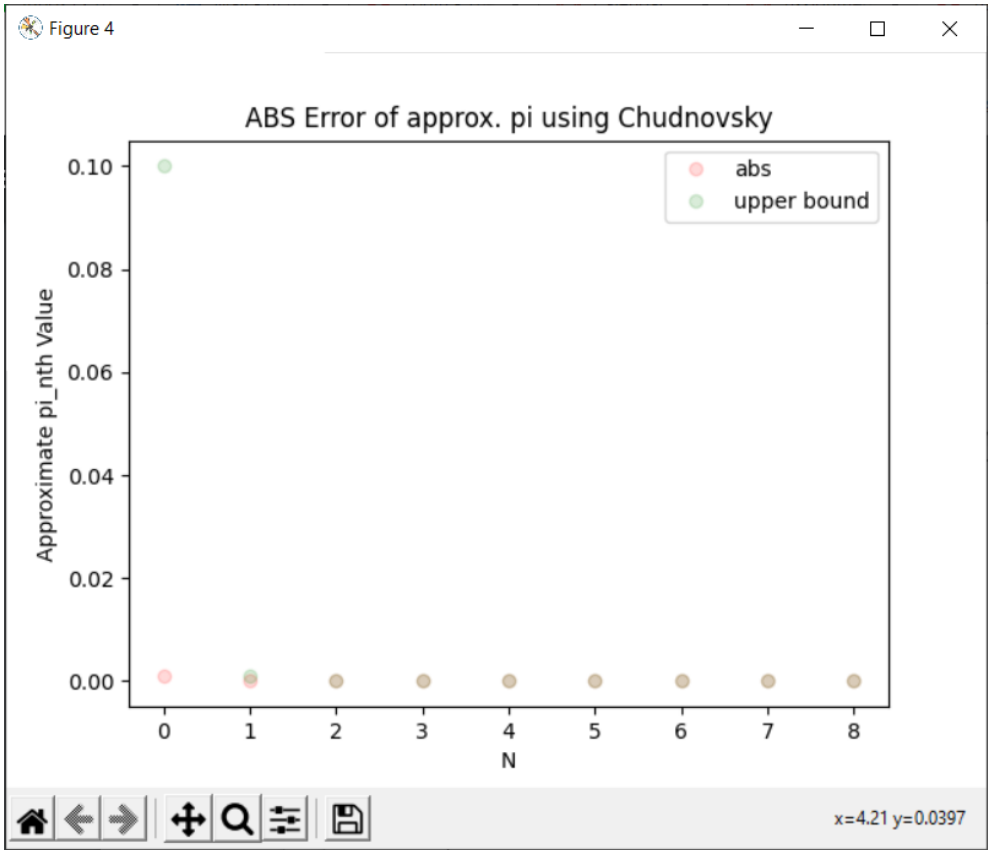


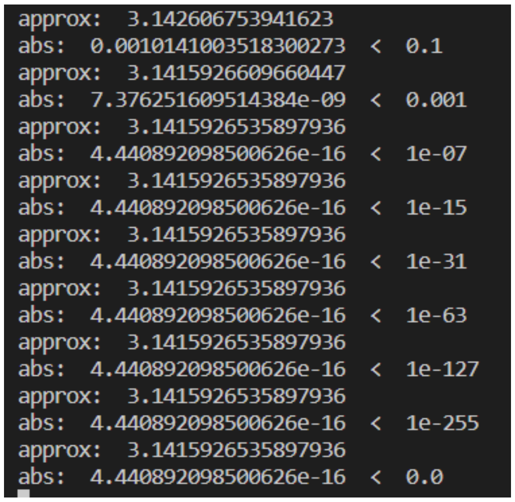
Figure 3
10000
8000
6000
4000
2000
0
Number of expansion terms used
0.74
0.76
0.78
0.80
0.82
Approximate Expansion Value
x=0.7719 y=1.6e+02



```
PS C:\Users\blake\Documents\college\Theory\Theory-3200> python3 .\Errors\Assignment4.py
rate of convergence:  6.366400291463427
```

The graph above (third graph) shows us how slowly the Leibniz formula converges to pi/4. The orange line is the actual value of pi/4 and each blue dot is the approximate value with a certain number of expansion terms(the y-axis). I calculated it up to 100000 terms, but the graph only shows up to 10000, but both show us that it converges very very slowly as seen by its rate which is about 6.366. However, this convergence rate is probably inaccurate because python's float does not support more than 15 to 17 decimal places of precision. Looking at the second image, it is zoomed in at the tip of the graph just to visually show how it is still fairly accurate but it still converges very slowly.





```
approx:  3.142606753941623
abs:  0.0010141003518300273  <  0.1
approx:  3.1415926609660447
abs:  7.376251609514384e-09  <  0.001
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  1e-07
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  1e-15
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  1e-31
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  1e-63
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  1e-127
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  1e-255
approx:  3.1415926535897936
abs:  4.440892098500626e-16  <  0.0
```

Looking at the console above, we can see that the first few Chudnovsky approximations are valid and hold true that pi_approx converges to pi, however, python does not support enough decimal places for it to continue any further but we can see that

$|\pi - \pi_n| < 10^{-(2^{n+1})}$ in the graph and in the console. This method figures out pi very fast and gives accurate results. Thus this method converges the fastest out of all the previous ones done in this assignment.

Sources:

All my code is submitted with the project and on GitHub here:

https://github.com/iPupkin/Theory-3200