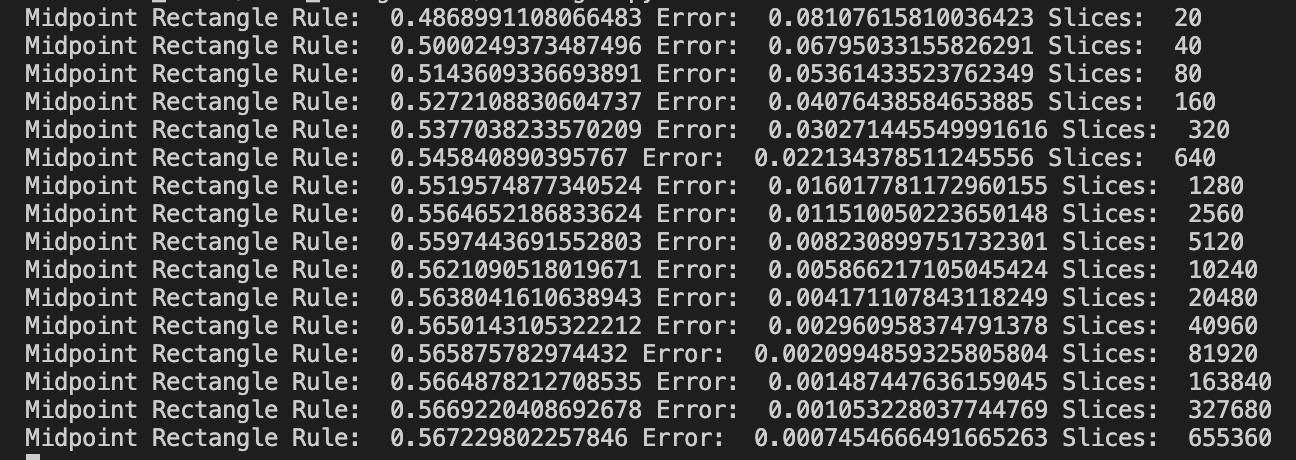
1)

a) Rectangle Rule

For this method, I have initially tried the left rectangle rule to integrate the given function. However, this method produced results which were too inaccurate, and took very long. After trying different methods, the middle point rectangle rule was much more accurate and fast. The results given were calculated using the middle rectangle rules.

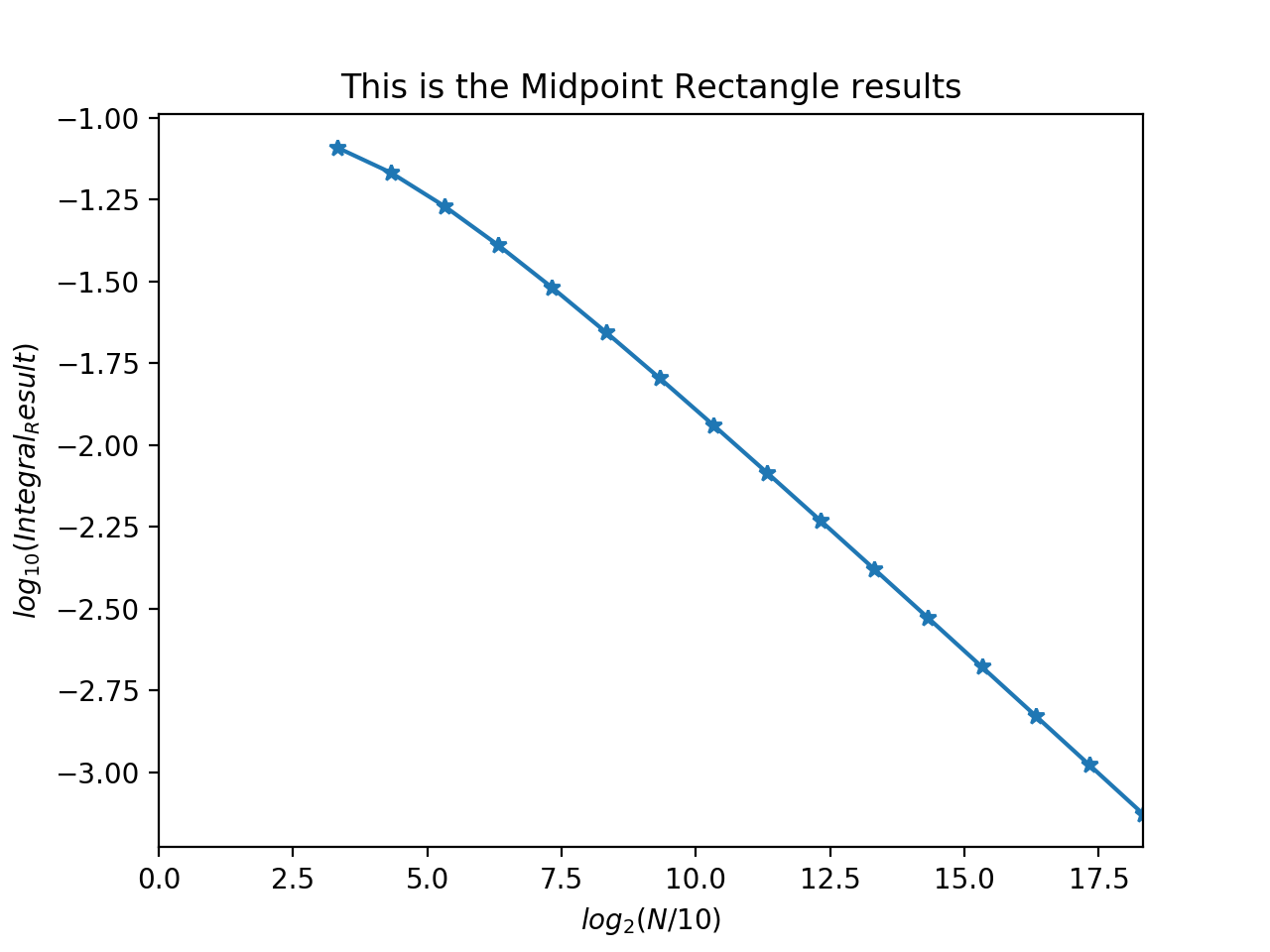


To achieve error of 1e-4, it took as many as slices.

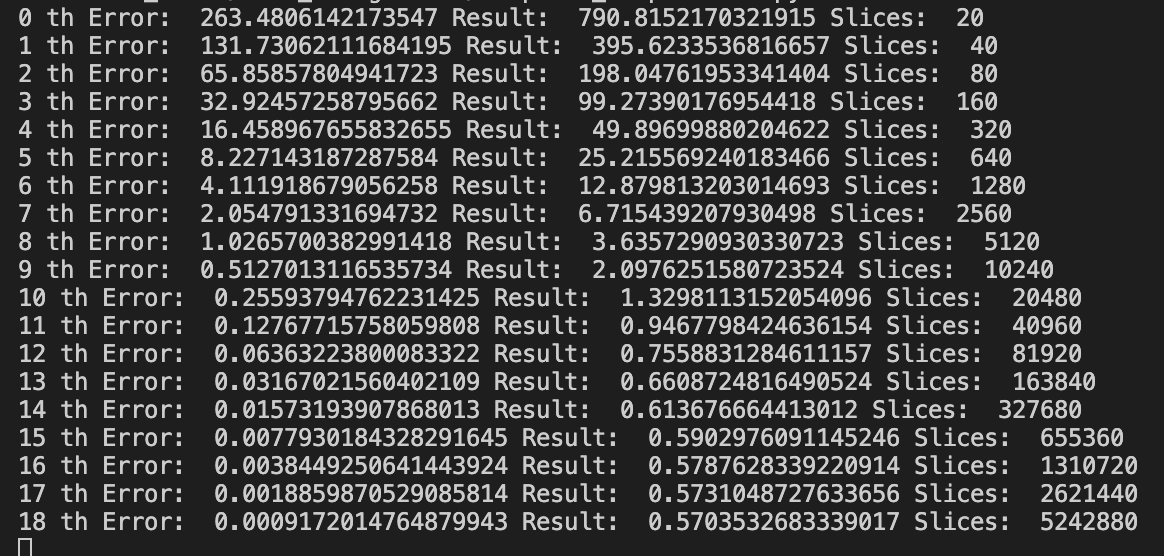
Using time.process\_time():

print("Computation Time: ", time.process\_time())

Computation Time: 17.495598

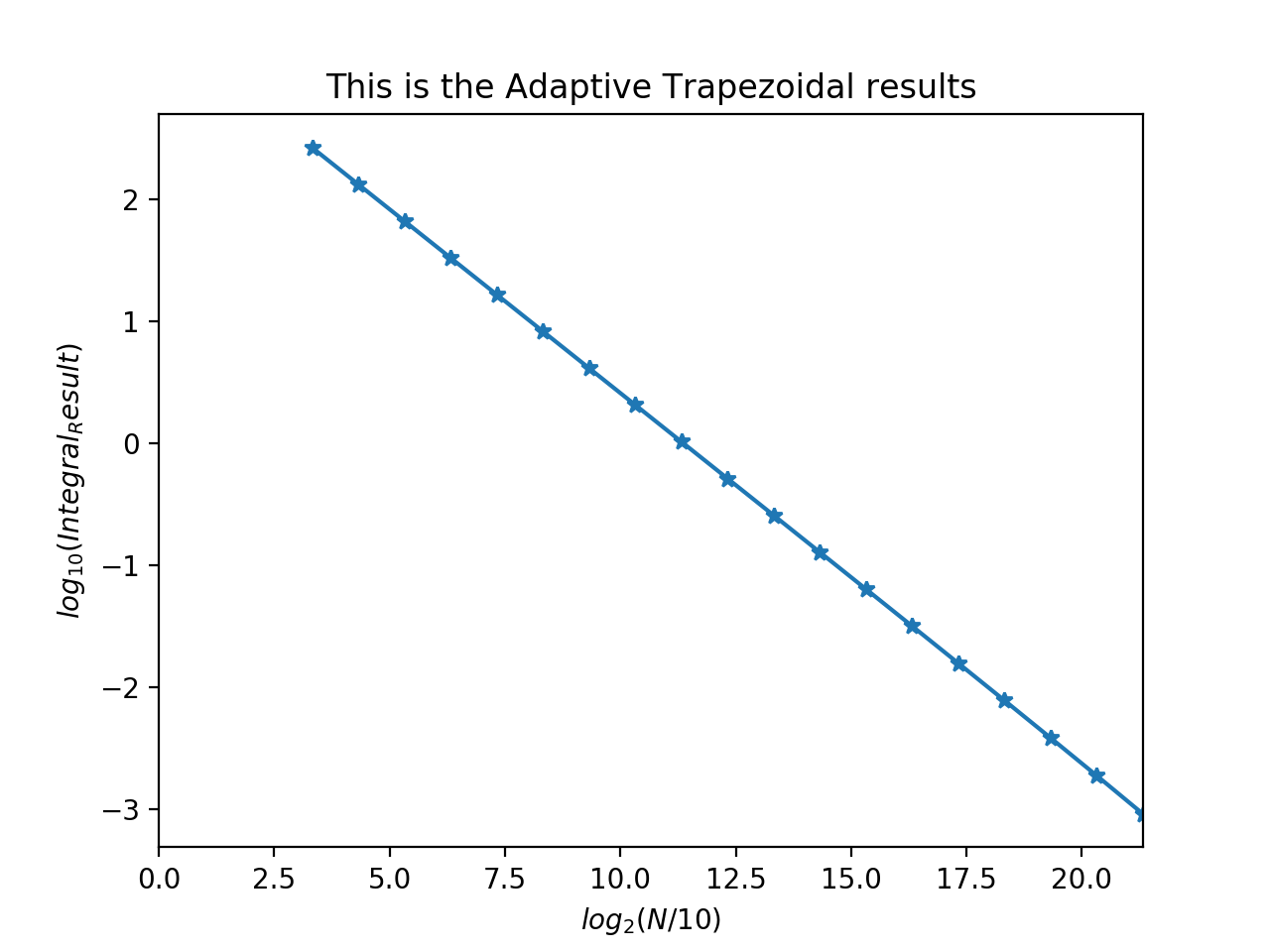


b) Adaptive Trapezoidal



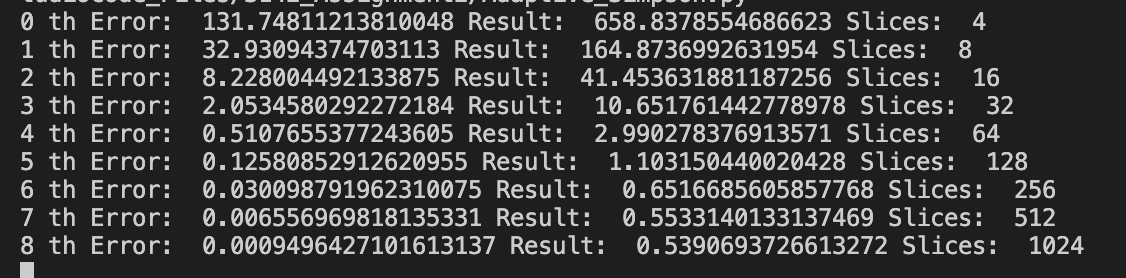
Computation Time: 32.80422

To achieve error of 1e-4, it took as many as slices.

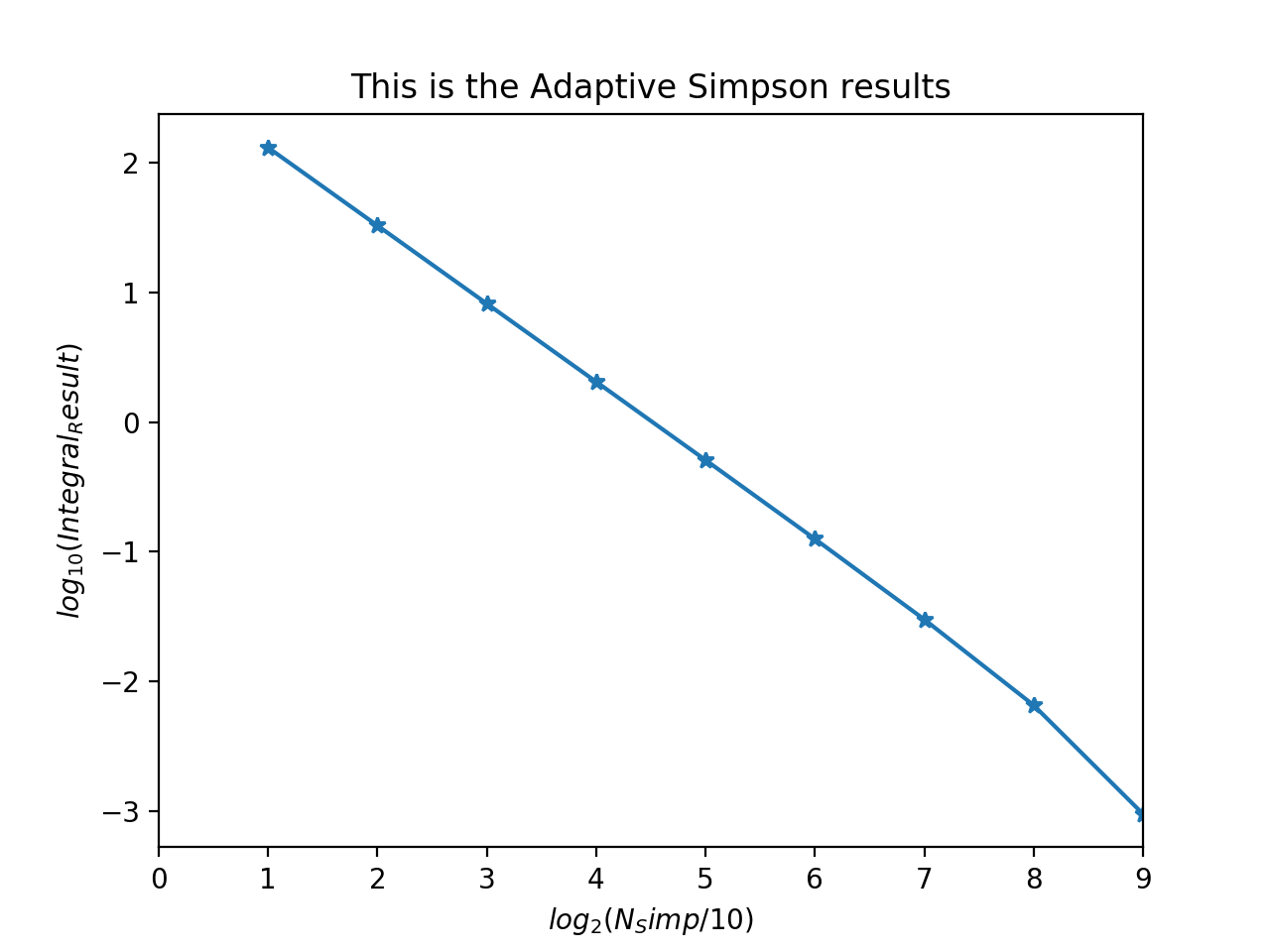


c) Adaptive Simpson

Adaptive Simpson’s method was the fastest until now with only 1024 slices, which is really small and fast compared to the two methods above.

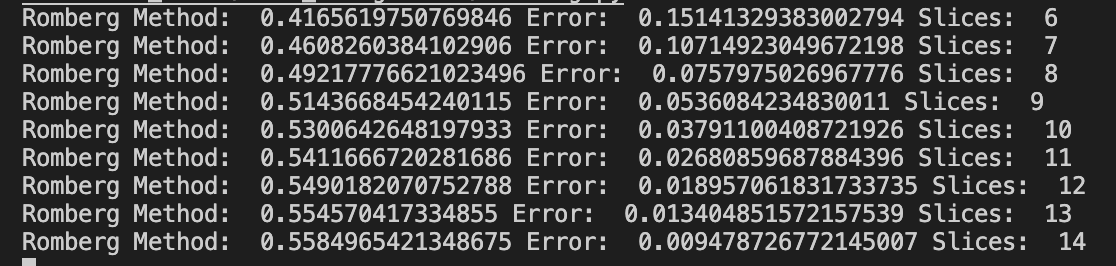


Computation Time: 0.76363

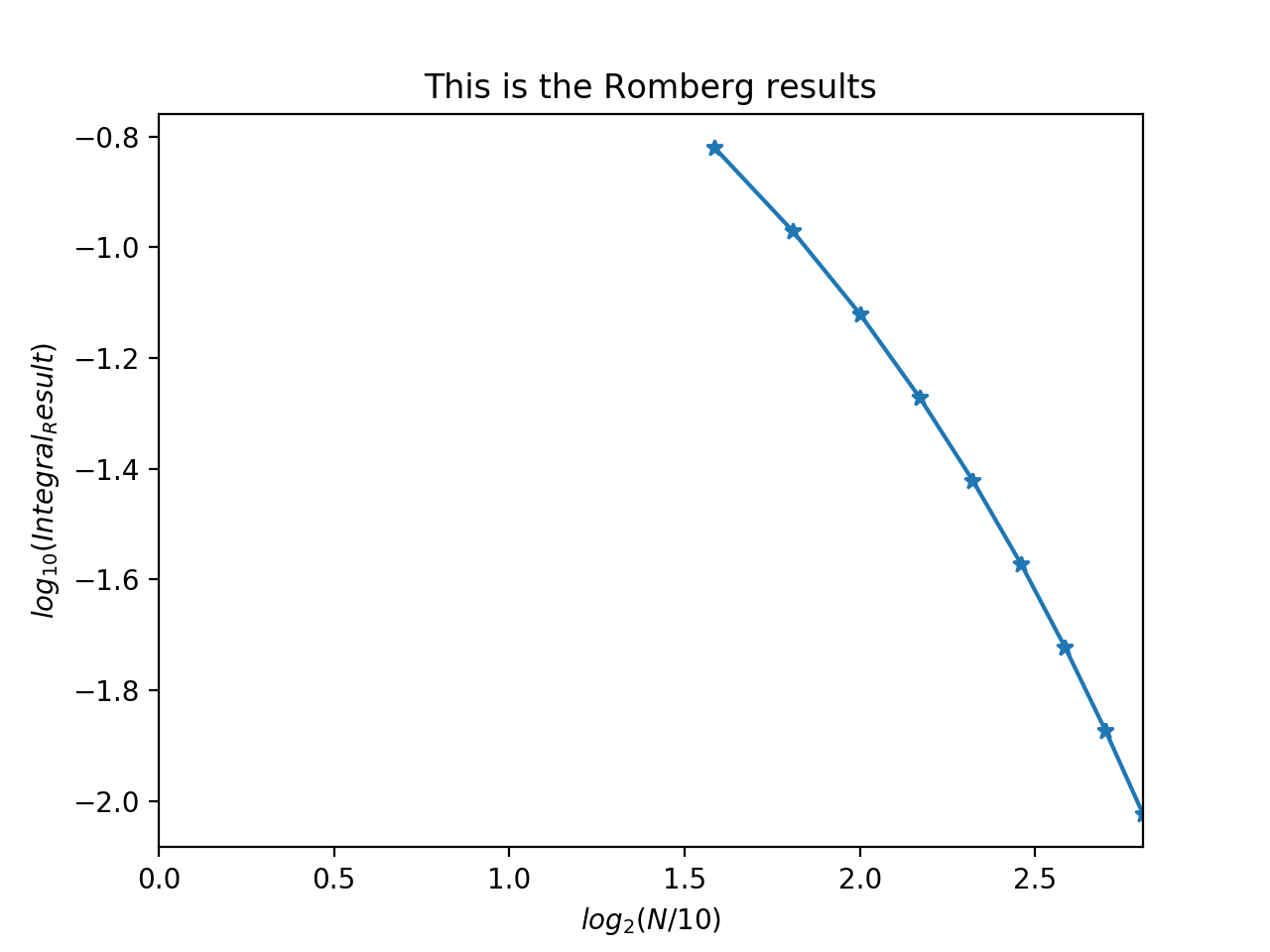


d) Romberg

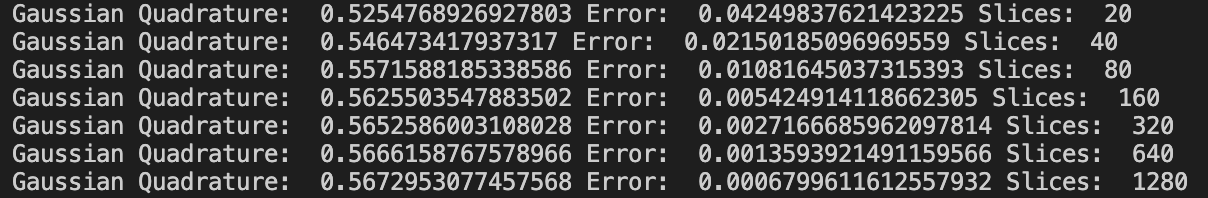
To increase the slices by a factor of 2 was too expensive in terms of computation time. Therefore, to at least allow the display of results, the slice, which already had a smaller start, was increased in increments of 1. Increasing the slices by factor of 2 could not be computed when the slices were at 40. It makes sense, because the computer would be computing a huge matrix. But it was relatively quick to compute for small slices.



Computation Time: 1.821842

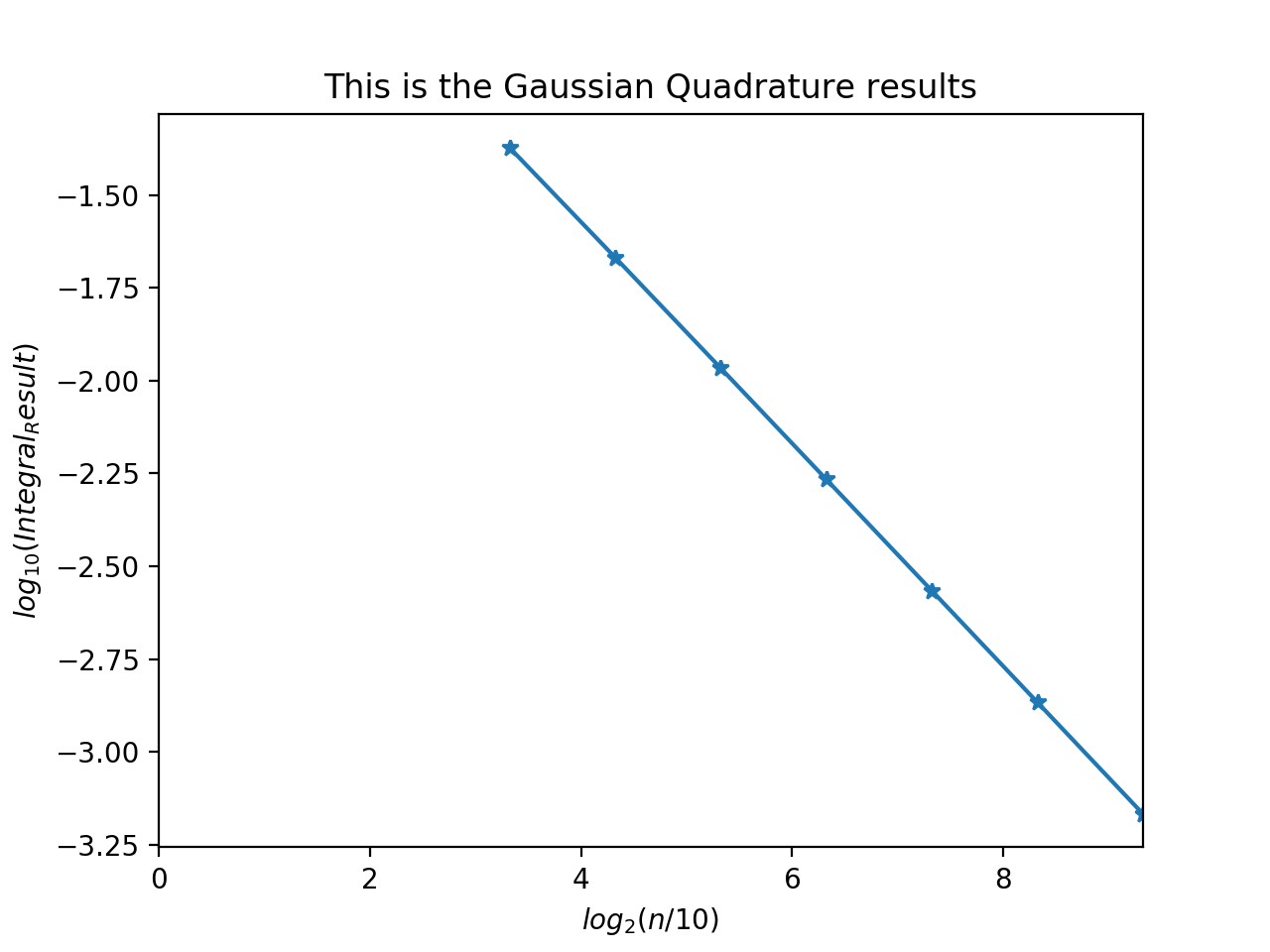


e) Gaussian Quadrature



Computation Time: 1.553066

As expected from the king of integral methods, it was very quick to compute and took relatively small amount of slices at 1280 to achieve error of 1e-4. However, it is similar to Adaptive Simpson’s method in part (c), but in fact Adaptive Simpson’s method requires less slices! However, it is acceptable if one views the error- Gaussian quadrature has ~200 more slices, but the error is also that much smaller when compared to the Adaptive Simpson’s method.

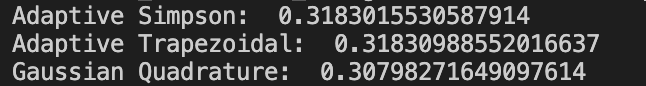


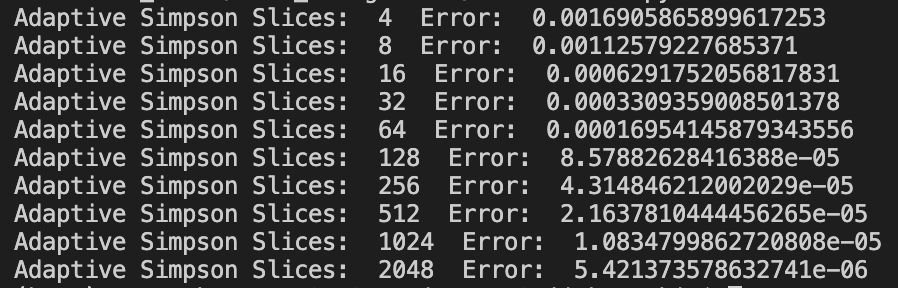
Comments on question 1:

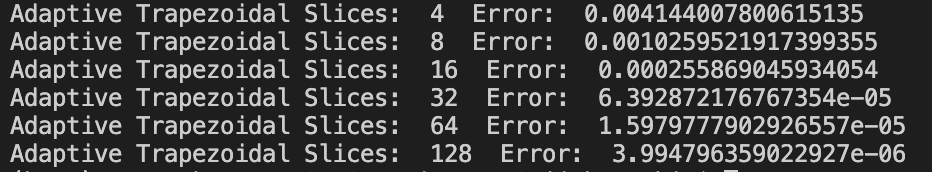
I expected Gaussian Quadrature to be the best method before trying it out, but after multiple trials, it seems like Adaptive Simpson is the quickest, and takes fewer slices compared to other methods.

2) Weierstrass Function

Comparing the three methods at N=10000. (Reason explained below)



The Adaptive Simpson and Adaptive Trapezoidal had been calculated starting at N = 2, and increasing N by a factor of 2 until the error was 1e-6.



The Adaptive Trapezoidal took much less slices to compute, while still giving smaller error.

For the Gaussian Quadrature, it was impossible to get an accurate result, because the nature of the Weierstrass function was extremely poorly behaved- with a lot of rapid changes in the graph, almost like a fuzzy signal. Therefore, Gaussian Quadrature cannot provide accurate results, and small error, because the assumption of the function requires a well behaved function. Therefore, instead, the three methods had been compared at N=10000.

At N=10000, it is assumed that the Adaptive Simpson and Adaptive Trapezoidal methods would give very accurate results. Therefore, based on this assumption, the result of the Gaussian Quadrature can be compared to the previous two methods to give an estimate of its error. The trapezoidal should give the most accurate result- therefore comparing the trapezoidal result and gaussian quadrature result at N=10000, we can estimate the error of the gaussian quadrature to be 0.01032716902919023. Still very large compared to the other two methods.