Answers

Problem 1.

Electric field from an infinitesimally thin spherical shell of a charge with radius R can be investigated from a ring along its central axis and subsequent integration of the rings to form a spherical shell.

Let dA represent the area of a ring formed by a slice of the shell centered on the z axis:

$dA = 2\pi r\sin\theta r d\theta = 2\pi r^2\sin\theta d\theta$

Based on a uniform surface charge density σ, charge on the ring would equal:

$dq = \sigma dA = 2\sigma\pi r^2\sin\theta d\theta$

Differential of the electric field E generated by the ring at point P along the central z axis can be defined as:

$dE = (4\pi\varepsilon_0)^{-1}(2\sigma\pi r^2)(z - r\cos\theta)(r^2+z^2 - 2zr\cos\theta)^{-3/2}\sin\theta d\theta$.

Total field E at P:

$$E = \frac{1}{4\pi\varepsilon_0} 2\pi r^2 \sigma \int_0^\pi \frac{z - r\cos\theta}{(r^2 + z^2 - 2zr\cos\theta)^{3/2}} \sin\theta d\theta$$

Let $\sigma(4\pi\varepsilon_0)^{-1}$ adopt a value equal to 1. Set radius r equal to 1.

$$E = 2\pi \int_0^\pi \frac{z - \cos\theta}{(1^2 + z^2 - 2z\cos\theta)^{3/2}} \sin\theta d\theta$$

An analytical solution was determined to compare the integrated values to the exact solution.

Further transform with $u=\cos\theta$

$$E = 2\pi \int_{-1}^{1} \frac{z - u}{(1^2 + z^2 - 2zu)^{3/2}} du$$

The analytical solution shall equal:

$$E = \frac{4\pi z}{z^2|z|} \quad |z| > r$$

$$E = 0 \; when \; |z| < r$$

The integral constructed with E as a function of u was evaluated via 3-point integration (Figure 1), whose results were compared to scipy.integrate.quad (Figure 2). Tolerance level was set at 10^{-4}. Singularities are shown for z=-r and z=+r. Both the *integrate* routine and *quad* were able to avoid errors in the vicinity of z= $\pm$r.

The difference between the calculated integrated values for the former method and the corresponding analytical solution revealed that the error range was [-10^{-5}, 10^{5}], evidencing good agreement between the numerical and analytical solutions (Figure 3). Performance of *quad* was superior and was consistent throughout all examined z values, whereas *integrate* had fluctuating errors, albeit with continual good agreement with the analytic solution.

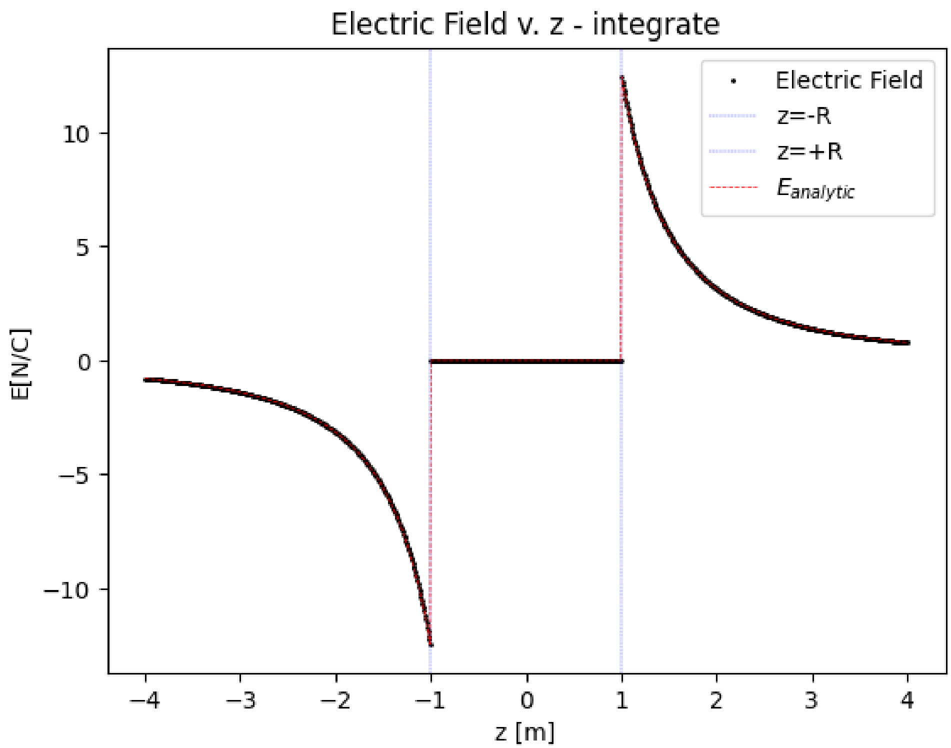

Figure 1. Electric field dependence on z, evaluated via *integrate* routine

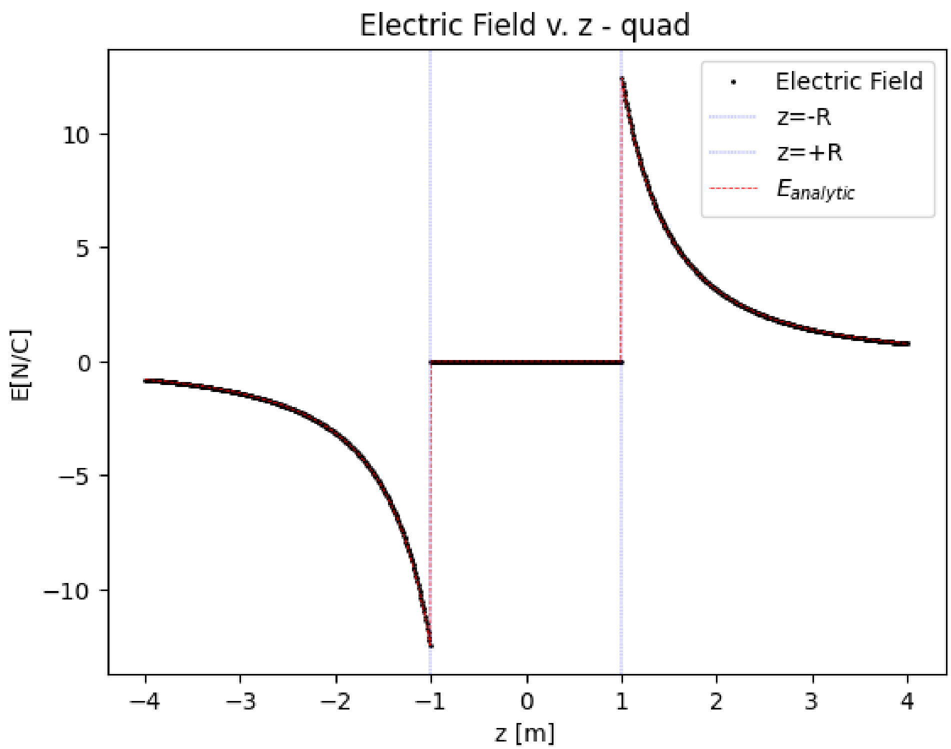

Figure 2. Electric field dependence on z, evaluated via *scipy.integrate.quad* function

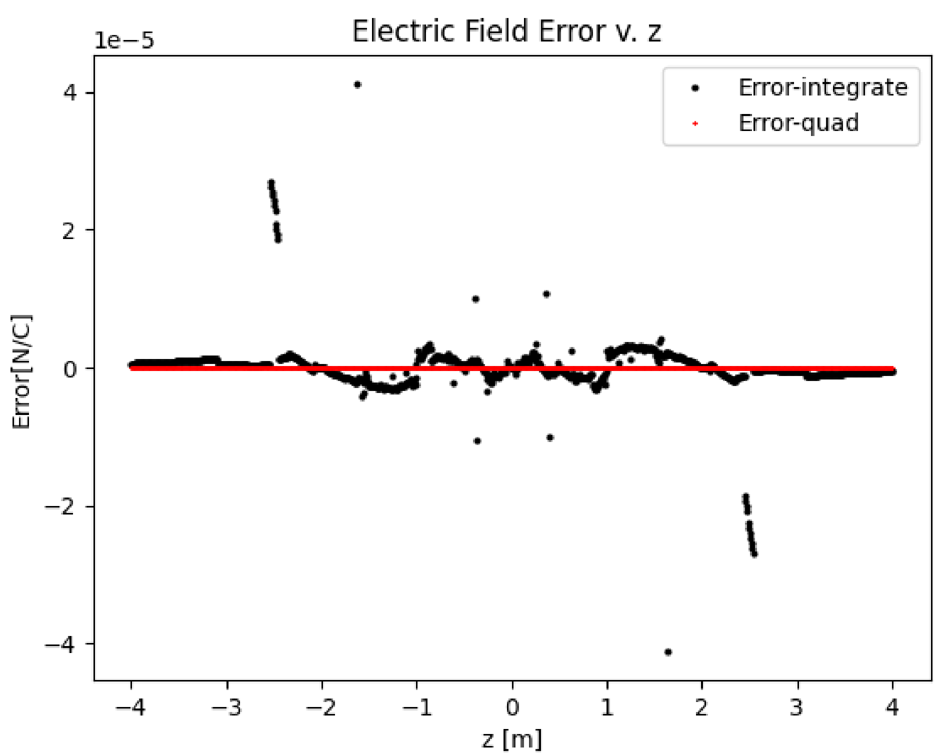

Figure 3. Electric field calculation error dependence on z

Problem 2.

A recursive variable step size integrator like the one discussed during the lectures that does not call f(x) multiple times for the same x was written. The function prototype is:

def integrate_adaptive(fun, a, b, tol, extra=None):

where extra should contain the information that a sub-call would need from preceding calls. On the initial call, extra is set equal to None, so the integrator could be prompted that it is starting off.

A few typical examples with cosh, exp, and Lorentzian were investigated in a set domain of [-5,5] with a tolerance level of 10^{-4}. The number of calls were reduced with each trial (Table 1), supporting the superior performance of the *integrate_adaptive* function.

Table 1. Number of Function Calls with *integrate* and *integrate_adaptive* Methods

Function	No. of calls *integrate*	No. of calls *integrate_adaptive*
cosh(x)	555	225
exp(x)	455	185
$(1+x^2)^{-1}$	275	113

Problem 3.

a)

A function that models the $\log_2 x$ valid from 0.5 to 1 to an accuracy in the region better than 10^{-6} (i.e. tolerance level) was written, utilizing a truncated Chebyshev polynomial fit via *np.polynomial.chebyshev.Chebyshev.fit*. The x-range rescaling was accomplished by implementing the domain parameter such that domain=(0.5,1) parameter. The terms unnecessary for the specified tolerance level were discarded via the *polynomial.chebyshev.Chebyshev.trim*(tol=1e-6) method. According to the method's output, the number of terms needed to achieve the 10^{-6} level was found to be 8. The resultant function and observed error compared to actual $\log_2 x$ is shown in Figures 4 and 5, respectively. The error of the fit was observed to be on the order of 10^{-7}, in agreement with the desired tolerance level.

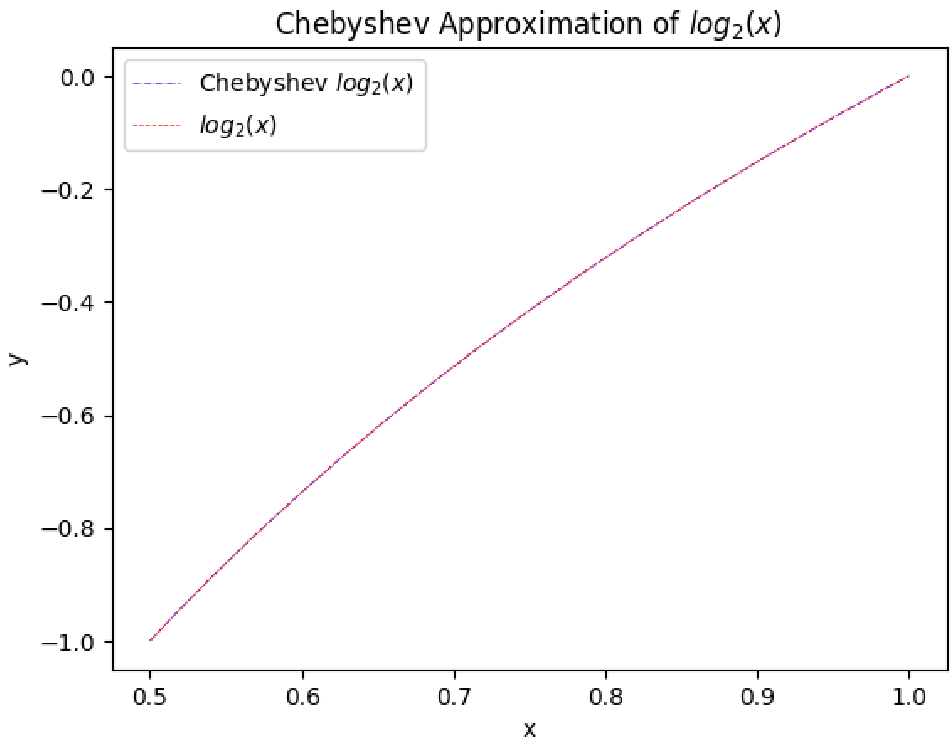

Figure 4. The Chebyshev Approximation of $\log_2(x)$

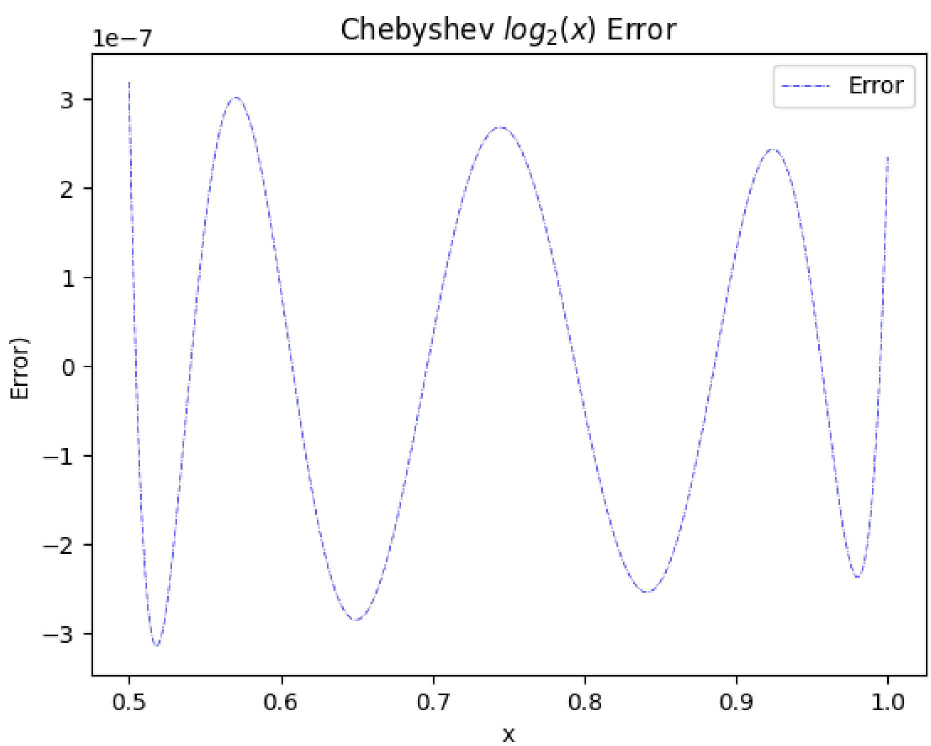

Figure 5. Error profile of the Chebyshev Approximation of $\log_2(x)$

b)

Once the Chebyshev expansion for 0.5 to 1 was accomplished, a routine called *mylog2* was constructed to take the natural log of any positive number x, ln(x) of base e, resorting to the routine *np.frexp,* breaking up a floating point number into its mantissa and exponent. Natural log may be expressed on the base of 2 as follows:

$\ln(x)=\log_2 x[\log_2 e]^{-1}$

Decomposed x should equal:

$x=m2^p$ where m and p are the mantissa and the exponent, respectively.

$\ln(x) = \ln(m2^p) = \log_2(m2^p)[\log_2 e]^{-1} = (p+\log_2 m)[\log_2 e]^{-1}$

Both $\log_2 m$ and $\log_2 e$ could subsequently be evaluated via the *mylog2* function for Chebyshev fits. Evaluated over the domain [0.5,1], resultant ln(x) and its corresponding error profile are shown in Figures 6 and 7, in turn. The error of the fit was observed to be on the order of 10^{-7}, reflecting the desired tolerance level.

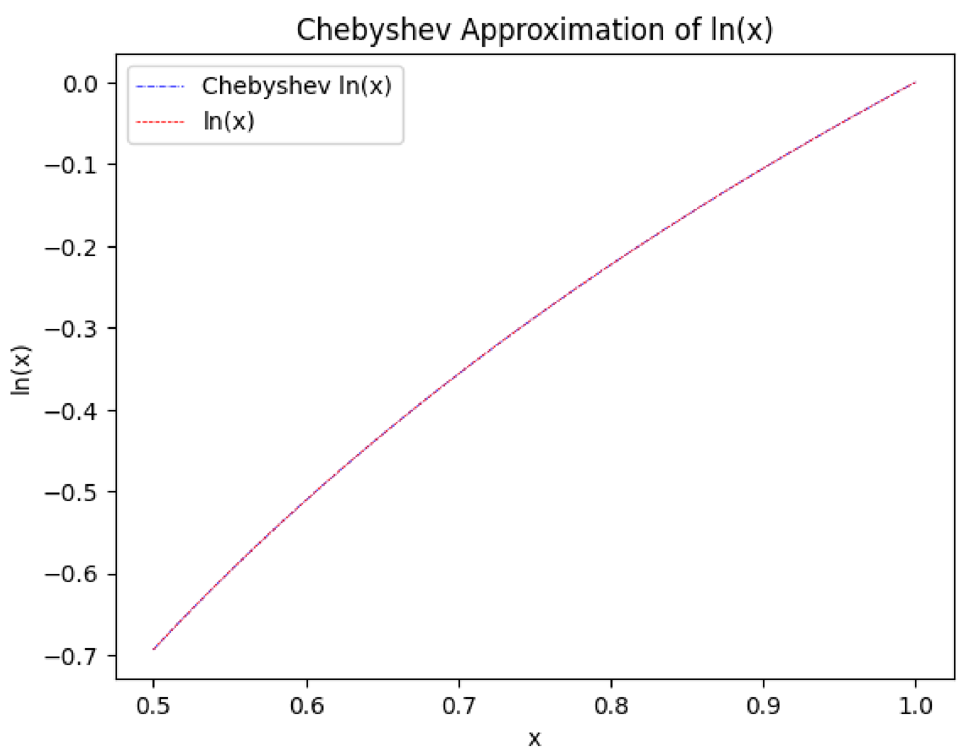

Figure 6. The Chebyshev Approximation of ln(x)

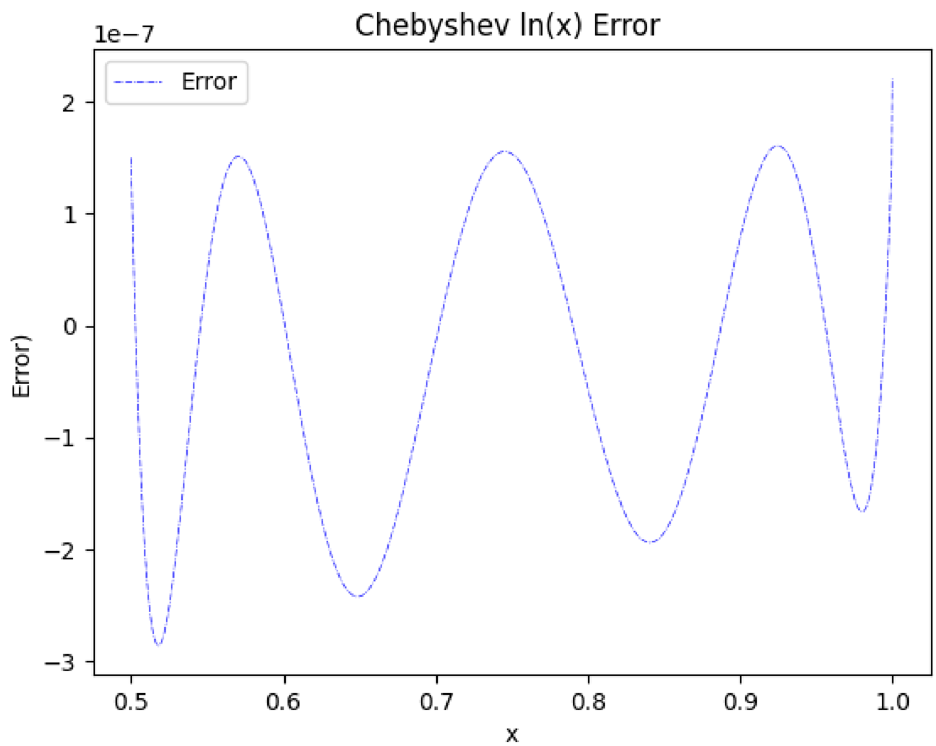

Figure 7. Error profile of the Chebyshev Approximation of ln(x)

A least squares fit to data via Legendre series was explored valid from 0.5 to 1 to an accuracy in the region better than 10^{-6}. A function that models the $\log_2 x$ valid from 0.5 to 1 to an accuracy in the region better than 10^{-6} (i.e. tolerance level) was written, utilizing a Legendre fit via *np.polynomial.legendre.legfit*. A new routine *mylog2leg* was constructed, with appropriate modifications of the previous function *mylog2*. The resultant function and observed error compared to actual $\log_2 x$ is shown in Figures 8 and 9, respectively. The error of the fit was observed to be approximately on the order of 10^{-13}, in agreement with the specified tolerance level.

Similarly, both $\log_2 m$ and $\log_2 e$ could subsequently be evaluated via the *mylog2leg* function for Legendre fits. Evaluated over the same domain, resultant ln(x) and its corresponding error profile are shown in Figures 10 and 11, in turn. The error of the fit was observed to be on the order of 10^{13}, reflecting the desired tolerance level.

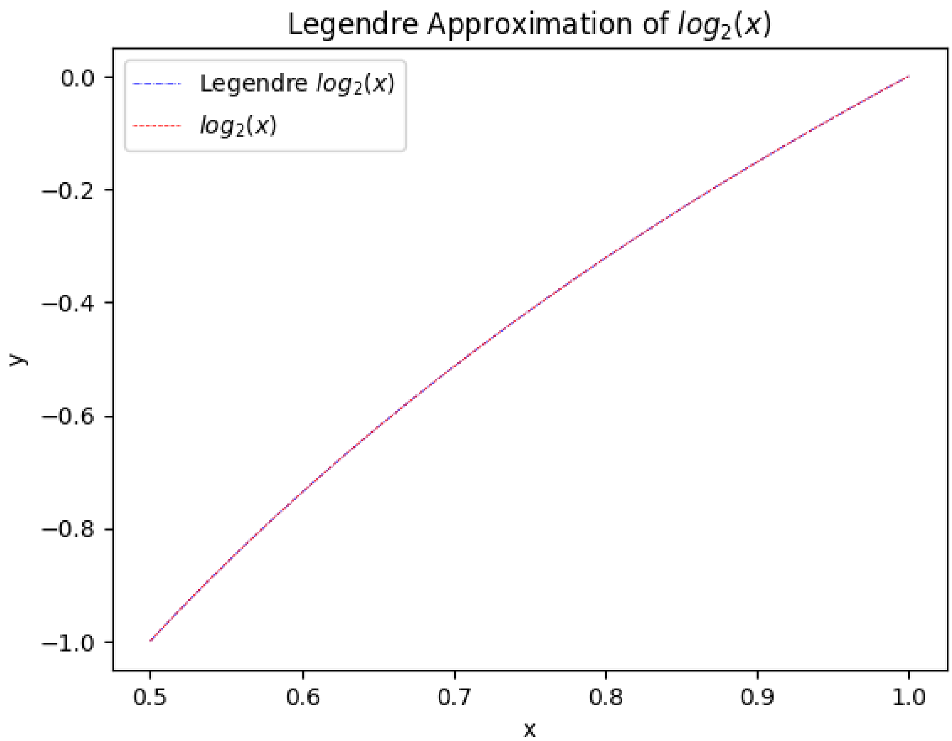

Figure 8. The Legendre Approximation of $\log_2(x)$

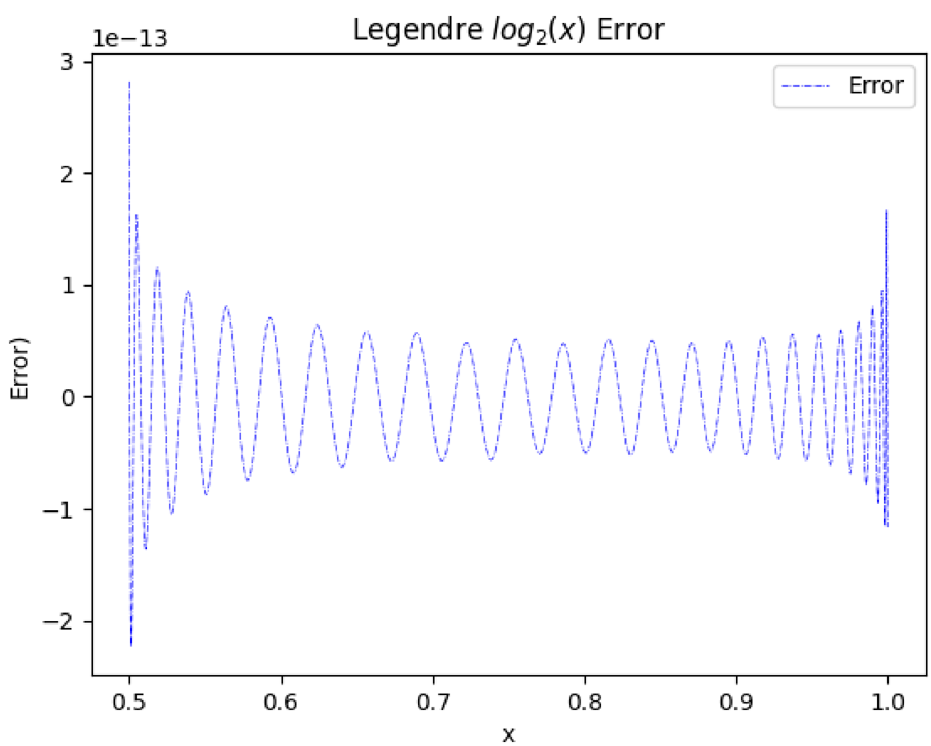

Figure 9. Error profile of the Legendre Approximation of $\log_2(x)$

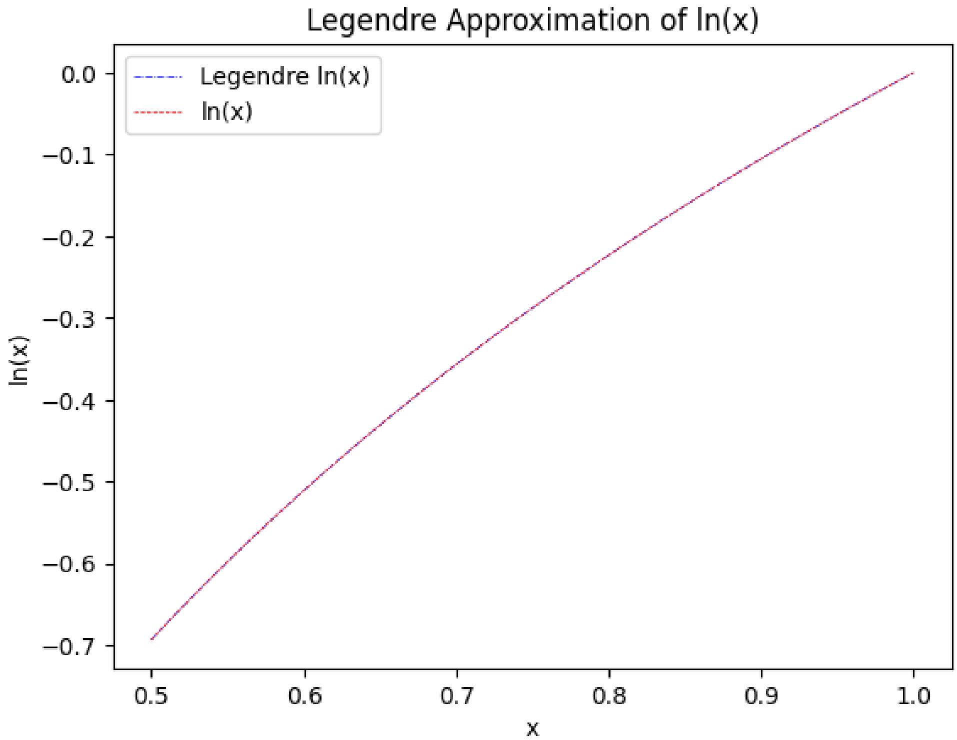

Figure 10. The Legendre Approximation of ln(x)

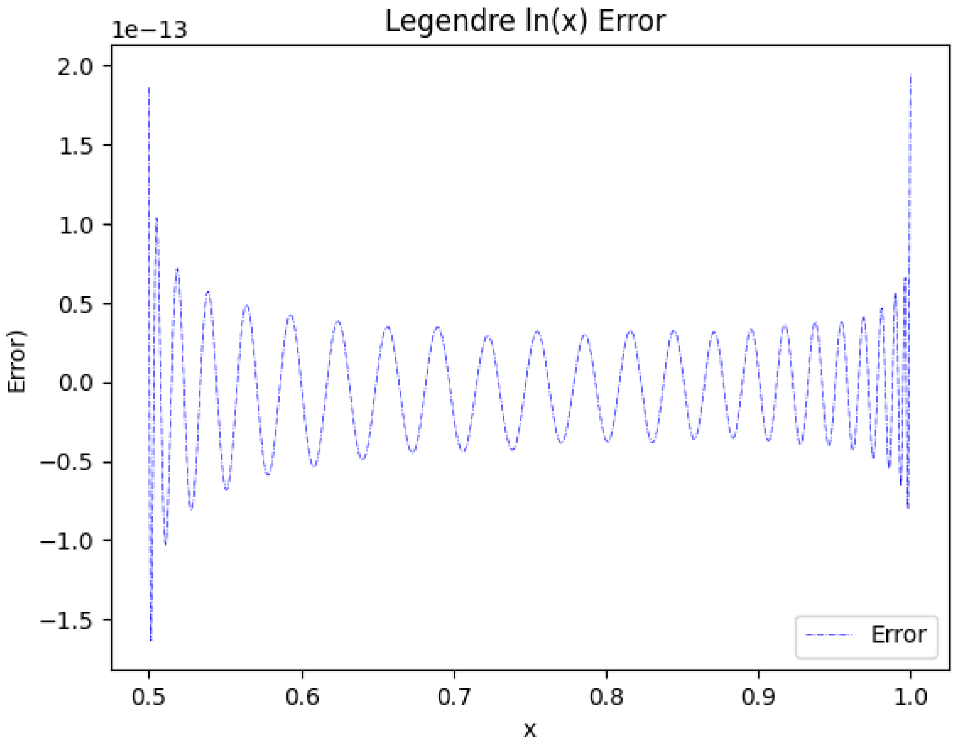

Figure 11. Error profile of the Legendre Approximation of ln(x)

Appendix A: Python Code

Jupyter notebook with relevant Python code and outputs is available at:

https://github.com/ck22512/comp_phys/tree/main/Assignment2