1. Prove the Equioscillation theorem.

   **Equioscillation theorem:** Let $f \in C[-1,1]$ and $p(x)$ be a polynomial whose degree doesn't exceed $n$. $p$ minimizes $\|f-p\|_\infty$ iff $f-p$ equioscillates at $n+2$ points.

   **Proof:**

   Let us define the supremum norm of $f$ to be:

   $$\|f\| = sup\{f(x) : x \in [a,b]\}$$

   and the best minimax error of degree $n$ by $d_n = \inf\{\|f-p\|$: $p$ is a polynomial on $[a,b]$ of degree $\leq n\}$

   The theorem is trivially true if $f$ is itself a polynomial of degree $<= n$. We assume not, and so $d_n > 0$.

   Since $f$ be continuous on $[-1,1]$, and let $p$ be a polynomial of degree $<= n$. If $f,\ p$ has a non-uniform alternating set $X = (x_0, ..., x_{n+1})$ of length $n+2$, then $d_n \geq min\{|e_i| : i = 0, 1, ..., n+1\}$.

   Suppose that $f,\ p_n$ has an alternating set of length $n+2$. We have $\|f-p_n\| \leq d_n$. As $d_n \leq \|f-p_n\|$ by the definition of $d_n$, it follows that $p_n$ is a polynomial of best approximation to $f$.

   Now suppose that $p_n$ is a polynomial of best approximation to $f$ and $f \neq p$. Then $f,\ p_n$ has an alternating set of length 2m and it can be extended into a sectioned alternating set of length $m$. We must have $m \geq n+2$, for if $m \leq n+1$ then we could add a polynomial $q$ of degree $\leq n$ to $p_n$ and get a better approximation than $p_n$, which is impossible. Thus every polynomial of best approximation has an alternating set of length at least $n+2$.

   To show uniqueness, suppose that $p_n$ and $q_n$ are both polynomials of best approximation, and we will show that they are equal.

   Note that $(p_n+q_n)/2$ is a polynomial of best approximation, as:

   $$\left\|f - \frac{p_n+q_n}{2}\right\| = \left\|\frac{f-p_n}{2} + \frac{f-q_n}{2}\right\| \leq \frac{1}{2}\|f-p_n\| + \frac{1}{2}\|f-q_n\| = d_n$$

   Therefore, there are $n+2$ alternating points at which $(f-p_n)/2 + (f-q_n)/2 = \pm d_n$.

   At each of these alternating points, $f-p_n$ and $f-q_n$ are both $d_n$ or both $-d_n$. So $f-p_n$ and $f-q_n$ agree on $n+2$ points, and so $(f-p_n)-(f-q_n) = q_n - p_n = 0$ at these $n+2$ points. Since $q_n - p_n$ is a polynomial of degree $\leq n$, $q_n$ and $p_n$ must be identical. Therefore the polynomial $p_n$ of best approximation is unique.

2. Consider the function $f(x) = |x|$ on the interval $[-1,1]$.

   - Prove that of all polynomials whose degree doesn't exceed 3, $p(x) = x^2 + \dfrac{1}{8}$ is the best approximation in the $\|\cdot\|_\infty$ norm.
   - Interpolate the function using 4 Legendre nodes and Chebyshev nodes. Call the polynomials obtained as $p_L(x)$ and $p_C(x)$.
   - Fill in the table below. You should be able to complete the table by hand.

| Approximation | $\|\cdot\|_2$ | $\|\cdot\|_\infty$ |
|---|---|---|
| $f(x)-p(x)$ | | |
| $f(x)-p_L(x)$ | | |
| $f(x)-p_C(x)$ | | |

   Comment on the errors you obtain using the different norm. Which one is optimal under the $\|\cdot\|_2$ and $\|\cdot\|_\infty$? For each norm, order the different approximations in increasing order of accuracy.

3. **Error estimate of Gaussian quadrature:** Prove that

$$\int_a^b w(x)f(x)dx - \sum_{i=1}^{n} w_i f(x_i) = \frac{f^{(2n)}(\xi)}{(2n)!}\|p_n(x)\|_w^2$$

for some $\xi \in (a,b)$ and $p_n(x)$ is the monic orthogonal polynomial corresponding to the weight function $w(x)$.

4. Let $f(x)$ be periodic function on $[0,1]$ of the form

$$f(x) = a_0 + \sum_{k=1}^{n}(a_k\ cos(2k\pi x) + b_k\ sin(2k\pi x))$$

Take the case of $n = 20$ and $a_k, bk$ are uniformly distributed on $[-1,1]$. Approximate $\int_{-1}^{1} f(x)dx$ using trapezoidal rule with $k$ points where $k \in \{1,2,...,80\}$. Compare the error with the exact integral and comment on the result you obtain. Prove that the trapezoidal rule give you the exact integral for $k > n$.

**Program:**

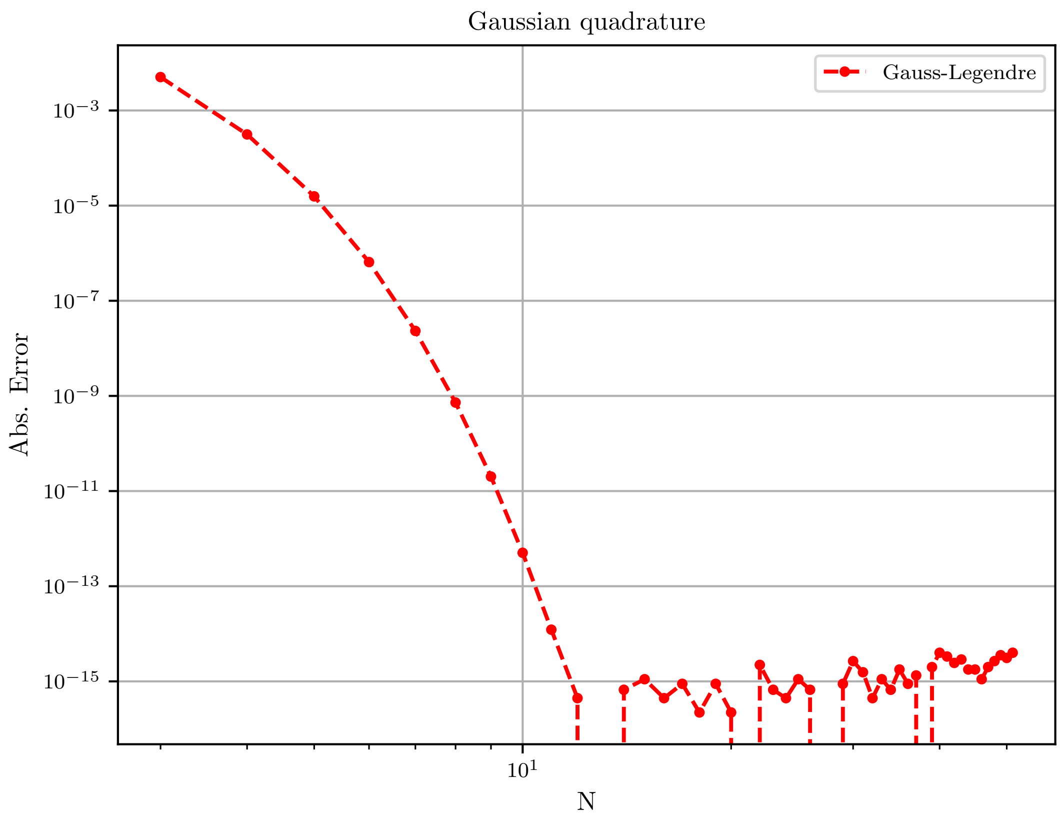
```
#!/usr/bin/env python
# File: trapezoidal.py

""" Script to implement the different quadrature and see how the error behaves """

import numpy as np
from scipy.integrate import quad
import matplotlib
matplotlib.rcParams['pgf.texsystem'] = 'pdflatex'
matplotlib.rcParams.update({'font.family': 'serif', 'font.size': 8,
    'axes.labelsize': 10, 'axes.titlesize': 10, 'figure.titlesize': 10})
matplotlib.rcParams['text.usetex'] = True
import matplotlib.pyplot as plt
matplotlib.use('Agg')

# end points of interval
a, b = -1, 1

# function to be integrated
f = lambda x, k: np.cos(2*k*np.pi*x) + np.sin(2*k*np.pi*x)

# number of different set of grids
Ngrids = 80

# exact value of the integral
exact = np.zeros(Ngrids)

h = np.zeros(Ngrids)        # different grid spacings

trap = np.zeros(Ngrids)     # trapezoidal rule

for k in range(1, Ngrids+1):
    N = 10*2**(k+2)              # number of grid points
    h[k] = (b - a)/(N - 1)       # grid spacing
    x = np.linspace(a,b,N)       # grid points

    trap[k] = h[k]*(np.sum(f(x,k)) - (f(a,k) + f(b,k))/2) # trapezoidal rule
    g = lambda x: np.cos(2*k*np.pi*x) + np.sin(2*k*np.pi*x)
    exact[k] = quad(g, -1, 1)[0]
```

```
# error calculations
trap_err = abs(np.double(trap - exact))      # error in trapezoidal rule

fig, ax = plt.subplots()
ax.loglog(h, trap_err, 'b.--', label=r'trapezoidal')
ax.set(xlabel=r'grid size', ylabel=r'error in quadrature')
ax.set_title(r'Quadrature convergence')
ax.grid(True); ax.legend()
plt.savefig('trapezoidal.pdf')
```

5. Evaluate $\int_{-1}^{1} e^{-x^2} dx$ using Gaussian quadrature with $n$ nodes, where $n \in \{3, 4, 5, ..., 51\}$. Plot the absolute error as a function of $N$ on a log-log plot.

**Program:**

```
#!/usr/bin/env python
# File: gaussleg.py

""" Script to evaluate integral exp(-x**2) from -1 to 1 """

import matplotlib.pyplot as plt
import numpy as np
from scipy.integrate import quad
import matplotlib
matplotlib.rcParams['pgf.texsystem'] = 'pdflatex'
matplotlib.rcParams.update({'font.family': 'serif', 'font.size': 8,
    'axes.labelsize': 10, 'axes.titlesize': 10, 'figure.titlesize': 10})
matplotlib.rcParams['text.usetex'] = True
matplotlib.use('Agg')

# exact value of the integral
f = lambda x: np.exp(-x**2)
exact = quad(f, -1, 1)[0]

n, error = [], []

for k in range(3, 52):

    n.append(k)

    # nodes and weights calculations
    x, w = np.polynomial.legendre.leggauss(k)

    # integration
    integral = np.inner(w, np.exp(-x**2))

    # error calculation
    error.append(abs(np.double(integral - exact)))

fig, ax = plt.subplots()
ax.loglog(n, error, 'r.--', label=r'Gauss-Legendre')
ax.set(xlabel=r'N', ylabel=r'Abs. Error')
ax.set_title(r'Gaussian quadrature')
ax.grid(True); ax.legend()
plt.savefig('gaussleg.pdf')
```


Gaussian quadrature
Gauss-Legendre
Abs. Error
N
$10^{-3}$
$10^{-5}$
$10^{-7}$
$10^{-9}$
$10^{-11}$
$10^{-13}$
$10^{-15}$
$10^{1}$