MA22037 Coursework 2024/25

The coursework counts for 25% of the final grade.

You are expected to upload your solutions to the submission point on Moodle as a **single document**, containing both the code and the mathematical writeup. Please submit two versions of your report: one as a .pdf file and one as a .ipynb file.

You are expected to type up your solutions. Handwritten solutions are not accepted.

The online session on Friday 28th March will focus on how to type mathematics using markdown in Python notebooks. In addition, you are welcome to email the lecturer (James Foster, jmf68@bath.ac.uk) or come to his office if you would like further advice on typing mathematics.

Due: Wednesday 23rd April at 5pm

The coursework will be graded as follows:

- [60%] Content.
- [40%] Presentation.

Each question in the assignment is equally weighted (although not all questions are of the same length). There are 50 marks in total.

All questions in the assignment must be fully typed up. Any handwritten submission risks losing presentation marks.

Quadrature in Python

Python gives us two ways of evaluating integrals. Symbolically, Python can sometimes find exact values, such as for $\int_{-1}^{1}\sqrt{1-x^2}\,dx$:

In [1]:
```
from sympy import *
x = Symbol('x')

integrate(sqrt(1-x**2),  (x, -1, 1))
```

Out[1]: $\frac{\pi}{2}$

Sometimes, the result of a symbolic function is returned using special functions, which are difficult to interpret. For example, $\int_0^1 x^{-\frac{1}{2}}\sqrt{1-x^2}\,dx$:

In [2]:
```
integrate(sqrt(1-x**2) / sqrt(x),  (x, 0, 1))
```

Out[2]:

$$\frac{\Gamma\left(\frac{1}{4}\right) {}_{2}F_{1}\left(\left.\begin{matrix} -\frac{1}{2}, \frac{1}{4} \\ \frac{5}{4} \end{matrix}\right| 1\right)}{2\Gamma\left(\frac{5}{4}\right)}$$

Here Γ and ${}_2F_1$ are special functions which can be evaluated numerically using Python.

In [3]:

```
from scipy.special import gamma, hyp2f1

gamma(1/4) * hyp2f1(-1/2, 1/4, 5/4, 1) / (2 * gamma(5/4))
```

Out[3]:

```
1.7480383695280801
```

Sometimes symbolic computation gives no useful information, such as for $\int_{-1}^{1} e^x \sqrt{1-x^2}\, dx$:

In [4]:

```
integrate(exp(x)*sqrt(1-x**2),  (x, -1, 1))
```

Out[4]:

$$\int\limits_{-1}^{1} \sqrt{-\left(x-1\right)\left(x+1\right)} e^{x}\, dx$$

However, Python also allows us to numerically evaluate integrals.

In [5]:

```
from scipy.integrate import quad
import numpy as np

result, error = quad(lambda x : np.sqrt(1-x**2), -1, 1)

print('True integral value:\t', np.pi / 2)
print('Approx integral value:\t', result)
```

```
True integral value:     1.5707963267948966
Approx integral value:   1.5707963267948986
```

In this assignment, we will develop a quadrature method for computing $\int_{-1}^{1} g(x)\sqrt{1-x^2}\, dx$ for a given $g : [-1,1] \to \mathbb{R}$. You can use the `scipy.integrate` to compare your results and computer errors for use in Section B.

Section A: Theory

We are interested in finding quadrature rules

$$\int_{-1}^{1} g(x)\sqrt{1-x^2}\, dx \approx \sum_{i=1}^{n} g(x_i) w_i\,,$$

for integrals with a weight function $\sqrt{1-x^2}$, for a smooth function $g : [-1,1] \to \mathbb{R}$, using weights w_i and nodes $x_i \in [-1,1]$. To do this, let $p_0(x) := 1$ and, for $n = 1, 2, \cdots$,

$$p_n(x) := \frac{1}{\sin\theta}\sin((n+1)\theta), \quad \theta := \cos^{-1}(x), \quad x \in [-1,1].$$

Let $\mathcal{P}_n$ denote the vector space of polynomials up to and including degree n.

A1 [5 Marks]

Show that $p_1(x) = 2x$. By using a trigonometric identity, show that

$$p_n(x) = 2xp_{n-1}(x) - p_{n-2}(x), \quad n = 2, 3, \cdots$$

Hence, show that $p_n(x)$ is a polynomial of degree n and that $p_0, \cdots, p_n$ form a basis for $\mathcal{P}_n$.

A2 [5 Marks]

Let $x_1, \cdots, x_n$ denote the roots of $p_n(x)$. Show that the matrix

$$J_n := \begin{pmatrix} p_0(x_1) & p_0(x_2) & \cdots & p_0(x_n) \\ p_1(x_1) & p_1(x_2) & \cdots & p_1(x_n) \\ p_2(x_1) & p_2(x_2) & \cdots & p_2(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ p_{n-1}(x_1) & p_{n-1}(x_2) & \cdots & p_{n-1}(x_n) \end{pmatrix}$$

is non-singular. Hence, show that there exist $w_1, \cdots, w_n$ such that

$$\sum_{i=1}^{n} p_k(x_i) w_i = \begin{cases} A, & k = 0 \\ 0, & k = 1, \cdots, n-1, \end{cases} \qquad A := \int_{-1}^{1} \sqrt{1-x^2}\, dx.$$

Hint: Suppose that there exists a $\mathbf{c} \in \mathbb{R}^n$ such that $\mathbf{c}^\top J_n = \mathbf{0}$.

A3 [5 Marks]

By making a substitution, show that

$$\int_{-1}^{1} p_n(x) p_m(x) \sqrt{1-x^2}\, dx = 0, \quad n \neq m.$$

A4 [5 Marks]

By using the basis $\{p_0, \cdots, p_n\}$ for $\mathcal{P}_n$ and A3, prove that

$$\int_{-1}^{1} p(x) \sqrt{1-x^2}\, dx = \sum_{i=1}^{n} p(x_i) w_i\,, \quad \forall p \in \mathcal{P}_n\,,$$

for the x_i and w_i defined above.

Now show that the relationship also holds for $p \in \mathcal{P}_{2n-1}$ to show the quadrature rule has degree of precision at least $2n-1$ and hence is a Gaussian quadrature rule.

Hint: you may use any standard result about polynomials covered in MA12001 (Algebra 1A).

A5 [5 Marks]

Let $q_0(x) := 1$ and $q_n(x) = 2^n \det(xI - A_n)$ for $n = 1, 2, \cdots$, where A_n is the $n \times n$ matrix

$$A_n := \begin{pmatrix} 0 & \frac{1}{2} & 0 & \cdots & & \\ \frac{1}{2} & 0 & \frac{1}{2} & \ddots & & \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & & \\ \vdots & \ddots & \frac{1}{2} & \ddots & \ddots & \\ & & & \ddots & 0 & \frac{1}{2} \\ & & & & \frac{1}{2} & 0 \end{pmatrix}$$

(all entries zero, apart from the two main off-diagonals). Show that $q_n = p_n$ for $n = 1, 2, \cdots$

Hence, show that eigenvalues of A_n equal the quadrature nodes x_i .

A6 [5 Marks]

Denote the i-th column of J_n (from A2) by

$$\mathbf{v}^i := \left[p_0(x_i), p_1(x_i), \cdots, p_{n-1}(x_i)\right]^\top, \quad i = 1, 2, \cdots, n.$$

Show that $\mathbf{v}^i$ is an eigenvector of A_n corresponding to the eigenvalue x_i . Hence, using the fact that eigenvectors of a symmetric matrix are orthogonal, show that the quadrature weights satisfy

$$w_i = A\frac{1}{\|\mathbf{v}^i\|^2}(v_1^i)^2, \quad A := \int_{-1}^{1} \sqrt{1-x^2}\, dx = \frac{1}{2}\pi,$$

where v_1^i is the first component of the i-th eigenvector $\mathbf{v}^i$.

Section B: Computing

In Section A, we showed that the nodes x_i are the eigenvalues of A_n (see A5) and the corresponding weights are

$$w_i = \frac{\pi}{2}\frac{1}{\|\mathbf{v}^i\|^2}(v_1^i)^2,$$

where v_1^i is the first component of the eigenvector $\mathbf{v}^i$ corresponding to x_i (see A6). This is an important way of calculating Gaussian quadrature rules and applies to many different types of integral (e.g. with different weighting functions).

The process of finding the relevant eigenvalues and eigenvectors can be done using `np.linalg.eig` .

Given an $n \times n$ matrix A, Python can find the eigenvectors and eigenvalues with the commands:

```
eigenvalues, eigenvectors = np.linalg.eig(A)

print(eigenvalues)          # print the eigenvalues
j = 1
print(eigenvalues[j])       # print the jth eigenvalue
print(eigenvectors[:,j])    # print the jth eigenvector
```

B1 [5 Marks]

Write Python code to compute the quadrature nodes $x_1, \cdots, x_n$ and weights $w_1, \cdots, w_n$ for the quadrature rule developed in Section A. Fill in

```
def getquad(n):
    '''Returns a vector of quadrature nodes x and weights w of dimension n.'''

    return x, w
```

B2 [5 Marks]

Write a routine to evaluate the quadrature for a given function g. Fill in

```
def myquad(g, x, w):
    '''Evaluate sum_{i=1,...,n} w_i g(x_i)'''

    assert len(x) == len(w), "x and w should have the same length"

    return result
```

It should run with the call

```
myquad((lambda x : x), x, w)
```

to evaluate $\int_{-1}^{1} x\sqrt{1-x^2}\, dx$.

Verify the degree of precision is $2n-1$ for $n = 10$ (using Python).

B3 [5 Marks]

Use Python to find reference values for

$$\int_{-1}^{1} g(x)\sqrt{1-x^2}\, dx,$$

in the case $g_1(x) = \exp(x)$, $g_2(x) = x\sin(x)$ and $g_3(x) = \sqrt{1-x^2}$. Using these values, you can now evaluate the error of quadarture approximations.

Thus, using the quadrature method developed in B2, evaluate

$$I_i := \int_{-1}^{1} g_i(x)\sqrt{1-x^2}\, dx, \quad \text{for } i = 1, 2, 3.$$

Using `matplotlib.pyplot.semilogy`, plot each error $|I_i - Q_n|$ against n for $n = 1, 2, \cdots, 20$.

All three of your lines should be labelled and on the **same plot**.

B4 [5 Marks]

In B3, you computed a sequence of quadarture approximations to the integral

$$I_i := \int_{-1}^{1} g_i(x)\sqrt{1-x^2}\, dx.$$

Explain why the quadarture approximation for I_3 is the slowest to converge.

Let $\{Q_n\}_{1\leq n\leq 20}$ denote the sequence for approximating I_3.

Using $\{Q_n\}$ and a known "sequence acceleration" technique, compute a new sequence $\{A_n\}$ which shows faster convergence to I_3.

Demonstrate this acceleration by plotting of the error of $\{Q_n\}$ and $\{A_n\}$ against n. As before, these errors should appear on the **same plot**.

Note:

- "Sequence acceleration" is not discussed in the lectures or lecture notes, so you will have to do your own research into this!
- Your approximations Q_n do not need to be defined for all $n \in \{1, \cdots, 20\}$.

In [ ]: