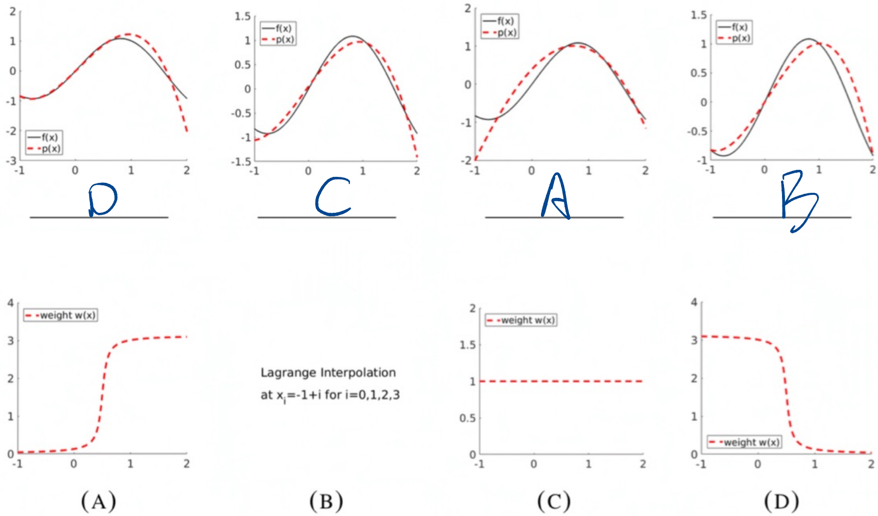
1. **[Best 2-norm approximation, 2+3pt]**

(a) The upper row in the below figure shows a function f together with a polynomial approximation. For three plots, the optimal best 2-norm fit for three different weights $w(x)$ is used, and one is the result of an Lagrange interpolation. Match the approximations in the upper row with the information (weight functions or interpolation points) in the lower row.



(b) Let $\{\varphi_0, \varphi_1, \varphi_2\}$ be a system of orthonormal polynomials on $[-1,1]$ with respect to the weight function $w(x) = \sqrt{1-x^2}$ given by

$$\varphi_0(x) = \sqrt{\frac{2}{\pi}}, \quad \varphi_1(x) = 2x\sqrt{\frac{2}{\pi}}, \quad \varphi_2(x) = (4x^2-1)\sqrt{\frac{2}{\pi}}.$$

Given $f(x) = \frac{2}{\sqrt{1-x^2}}$, find the polynomial best fit of degree 2 in the weighted 2-norm.

It suffices to find f's projection on to each orthonormal polynomial.

$$\langle f, \varphi_0 \rangle = \int_{-1}^{1} \sqrt{1-x^2} \frac{2}{\sqrt{1-x^2}} \cdot \sqrt{\frac{2}{\pi}}\, dx = 4\sqrt{\frac{2}{\pi}}$$

$$\langle f, \varphi_1 \rangle = \int_{-1}^{1} \sqrt{1-x^2} \frac{2}{\sqrt{1-x^2}} 2x\sqrt{\frac{2}{\pi}}\, dx = 0$$

$$\langle f, \varphi_2 \rangle = \int_{-1}^{1} \sqrt{1-x^2} \frac{2}{\sqrt{1-x^2}} (4x^2-1)\sqrt{\frac{2}{\pi}}\, dx = \frac{2}{3} \cdot 2\sqrt{\frac{2}{\pi}} = \frac{4}{3}\sqrt{\frac{2}{\pi}}$$

$$\Rightarrow \text{Best fit: } p(x) = 4\sqrt{\frac{2}{\pi}}\,\varphi_0 + \frac{4}{3}\sqrt{\frac{2}{\pi}}\,\varphi_2 = \frac{8}{\pi} + \frac{8}{3\pi}(4x^2-1) = \frac{32}{3\pi}x^2 + \frac{16}{3\pi}$$

2. **[Interpolation and optimal 2-norm approximation, 2+2+2+2pt]** For an interval (a, b), $n \in \mathbb{N}$ and disjoint points $x_0, \ldots, x_n$ in $[a, b]$, we define[1] for polynomials p, q

$$\langle p, q \rangle := \sum_{i=0}^{n} p(x_i) q(x_i).$$

(a) Show that $\langle \cdot, \cdot \rangle$ is an inner product for each $\mathcal{P}_k$ with $k \leq n$, where $\mathcal{P}_k$ denotes the space of polynomials of degree k or less.

(a) It suffices to verify.

(1) Symmetry: $\langle p, q \rangle := \sum_{i=0}^{n} p(x_i) q(x_i) = \sum_{i=0}^{n} q(x_i) p(x_i) = \langle q, p \rangle$

(2) linearity. $\langle p_1 + p_2, q \rangle = \sum_{i=0}^{n} (p_1(x_i) + p_2(x_i)) q(x_i) = \sum_{i=0}^{n} p_1(x_i) q(x_i) + \sum_{i=0}^{n} p_2(x_i) q(x_i)$
$= \langle p_1, q \rangle + \langle p_2, q \rangle$

$\langle \alpha p, q \rangle = \sum_{i=0}^{n} \alpha p(x_i) q(x_i) = \alpha \sum_{i=0}^{n} p(x_i) q(x_i) = \alpha \langle p, q \rangle$

The linearity in second entry followed by symetry and above reasoning.

(3) positive definite.

$\langle p, p \rangle = \sum_{i=0}^{n} p^2(x_i) \geq 0$

If $\langle p, p \rangle = 0$, $p(x_i) = 0$ $\forall$ $i = 0, \ldots, n$. $(n+1)$ zeros

As $p \in P_n$, it has at most n zeros

It follows that p is constantly 0.

(b) Why is $\langle \cdot , \cdot \rangle$ not an inner product for $k > n$?

Conter example:

Let $p(x) = \prod_{i=0}^{n} (x - x_i) \in P_{n+1}$

$<p, p> = \sum_{i=0}^{n} p^2(x_i) = 0$ but p is not 0

(c) Show that the Lagrange polynomials L_i corresponding to the nodes $x_0, \ldots, x_n$ are orthonormal with respect to the inner product $\langle \cdot , \cdot \rangle$.

(C)

$$<L_i, L_j> = \sum_{k=0}^{n} L_i(x_i) L_j(x_i)$$

Vanishes when $k \neq i$ vanishes when $k \neq j$.

$$<L_i, L_j> := \begin{cases} 0 & i \neq j \\ L_i(x_i) \cdot L_i(x_i) = 1 & i = j \end{cases}$$

It follows $\{L_i\}$ is an othonomal set w.r.t. $<\cdot, \cdot>$

(d) For a continuous function $f : [a, b] \to \mathbb{R}$, compute its optimal approximation in $\mathcal{P}_n$ with respect to the inner product $\langle \cdot\,, \cdot \rangle$ and compare with the interpolation of f.

(d) Since L_i is orthonormal, it suffices to take inner product.

$$< f, L_i(x) > = \sum_{k=0}^{n} f(x_i)\, L_i(x_k) = f(x_i)$$

$$\Rightarrow P = \sum_{i=0}^{n} f(x_i)\, L_i(x)$$ which happens to be the Lagrange polynomials.

3. **[Newton-Cotes vs. Gauss Quadrature, 2+2+2+1pt]** We discussed two methods to integrate functions numerically, namely the Newton-Cotes formulas and Gauss quadrature.

(a) Recall that we calculated the first three orthogonal polynimals with respect to $w \equiv 1$ on $(0,1)$ in class to be $\{\varphi_0, \varphi_1, \varphi_2\} = \{1, x - 1/2, x^2 - x + 1/6\}$. Calculate $\varphi_3(x)$ using the ansatz $\varphi_3(x) = x^3 - a_2\varphi_2(x) - a_1\varphi_1(x) - a_0\varphi_0(x)$, with appropriately computed $a_2, a_1, a_0 \in \mathbb{R}$.

(a) we first normalize $\{\varphi_0, \varphi_1, \varphi_2\}$, transform them to be orthonormal.

$\hat{\varphi}_0 = 1$

$\hat{\varphi}_1 = 2\sqrt{3}\,(x - \frac{1}{2})$

$\hat{\varphi}_2 = 6\sqrt{5}\,(x^2 - x + \frac{1}{6})$

$\Rightarrow \varphi_3 = x^3 - \hat{a}_0\hat{\varphi}_0 - \hat{a}_1\hat{\varphi}_1 - \hat{a}_2\hat{\varphi}_2$

$= x^3 - \frac{3}{2}x^2 + \frac{3}{5}x - \frac{1}{20}$

$\Rightarrow \hat{a}_0 = \langle x^3, \hat{\varphi}_0 \rangle = \frac{1}{4}$

$\hat{a}_1 = \langle x^3, \hat{\varphi}_1 \rangle = -\frac{\sqrt{3}}{10}$

$\hat{a}_2 = \langle x^3, \hat{\varphi}_2 \rangle = \frac{1}{4\sqrt{5}}$

(b) Derive the Gaussian Quadrature formula for $n = 2$, i.e., calculate both the quadrature points x_0, x_1, x_2 (these are the roots of φ_3 and the corresponding weights W_0, W_1, W_2.[2]

(b)

Nodes of Gaussian Quadrature are roots of φ_3:

$$\begin{cases} x_0 = \frac{1}{2} \\ x_1 = \frac{1}{2} + \frac{1}{2}\sqrt{\frac{3}{5}} \\ x_2 = \frac{1}{2} - \frac{1}{2}\sqrt{\frac{3}{5}} \end{cases}$$

$\Rightarrow$

$x_0 = \frac{1}{2}$

$x_1 = \frac{1}{2} + \frac{1}{2}\sqrt{\frac{3}{5}}$

$x_2 = \frac{1}{2} - \frac{1}{2}\sqrt{\frac{3}{5}}$

$$W_0 = \int_0^1 l_0^2(x)\,dx = \int_0^1 \left[\frac{(x - x_1)(x - x_2)}{(\frac{1}{2}\sqrt{\frac{3}{5}})(-\frac{1}{2}\sqrt{\frac{3}{5}})}\right]^2 dx = \frac{4}{9}$$

$$W_1 = \int_0^1 l_1^2(x)\,dx = \frac{5}{18} = W_2$$

$\Rightarrow$

$$\int_0^1 f(x)\,dx \approx \sum_{i=0}^{2} f(x_i) W_i$$

$$= \frac{5}{18} f\left(\frac{1}{2} - \frac{1}{2}\sqrt{\frac{3}{5}}\right) + \frac{4}{9} f\left(\frac{1}{2}\right) + \frac{5}{18} f\left(\frac{1}{2} + \frac{1}{2}\sqrt{\frac{3}{5}}\right)$$

(c) Now we want to compare Gaussian quadrature derived in (b) with the Simpson's Rule. Use both methods to numerically find

$$I_k = \int_0^1 x^k \, dx, \qquad \text{for} \quad k = 0, \ldots, 7.$$

Plot the errors arising in each method as a function of k. Note that to find the error, you will need to calculate the exact values for I_k (by hand).

$$I_k = \int_0^1 x^k \, dx = \left. \frac{x^{k+1}}{k+1} \right|_0^1 = \frac{1}{k+1} \leftarrow \text{Exact Error.}$$

See Code file.

(d) For a continuous function $f : [a, b] \to \mathbb{R}$, compute its optimal approximation in $\mathcal{P}_n$ with respect to the inner product $\langle \cdot , \cdot \rangle$ and compare with the interpolation of f.

See Code file.

4. **[Orthogonal polynomials on $[0, \infty)$, 2+2+2pt extra credit]** We do Gram-Schimdt here.

(a) Find orthogonal polynomials l_0, l_1, l_2, l_3 for the unbounded interval $[0, \infty)$ with the weight function $\omega(x) = \exp(-x)$.[3] Plot these polynomials (they are called *Laguerre polynomials*).

See codefile.

(a)

$l_0 = 1 \rightarrow \langle l_0, x \rangle = \int_0^{+\infty} \exp(-x)\, x = -\exp(-x) \Big|_0^{+\infty} = 0 - (-1) = 1$

$l_1 = -x+1$ ← substract, and normalization.

$l_2 = \frac{1}{2}(x^2 - 4x + 2)$ --- by some tedious calculations.

$l_3 = \frac{1}{6}(-x^3 + 9x^2 - 18x + 6)$ we have the result.

(b) As these are orthogonal polynomials, they correspond to a quadrature rule for weighted integrals on $[0, \infty)$. The resulting quadrature points and weight are given in Table 1. Verify

Table 1: Gauss quadrature points and weights for quadrature on $[0, \infty)$.

n	x_i	W_i
2	0.585786	0.853553
	3.41421	0.146447
3	0.415775	0.711093
	2.29428	0.278518
	6.28995	0.0103893
4	0.322548	0.603154
	1.74576	0.357419
	4.53662	0.0388879
	9.39507	0.000539295

that for $n = 2$, $n = 3$, the quadrature nodes x_i are the roots of the polynomials $l_2(x), l_3(x)$ (up to round-off).

$n=2$ $\quad l_2(x) = \frac{1}{2}(x^2 - 4x + 2) = 0$

$$x_{12} = \frac{4 \pm 2\sqrt{2}}{2} = 2 \pm \sqrt{2}$$

$x_1 \doteq 0.5857$

$x_2 \doteq 3.4142$

$n=3$ $\quad l_3(x) = \frac{1}{6}(-x^3 + 9x^2 - 18x + 6) = 0$

According to the calculator, we have

$x_1 = 0.4157$

$x_2 = 2.2943$

$x_3 = 6.2899$

It shows that the quadrature nodes x_i are roots of the polynomial.

(c) Use the quadrature rules from Table 1 to approximate the integrals

$$\int_0^\infty \exp(-x)\exp(-x)\,dx \quad \text{and} \quad \int_0^\infty \exp(-x^2)\,dx.$$

Note that, to take into account the weight $\omega(x) = \exp(-x)$, for the first integral $f(x) = \exp(-x)$ and for the second $f(x) = \exp(-x^2 + x)$. Report the errors for $n = 2, 3, 4$ using that the exact values for the integrals are $1/2$ and $\sqrt{\pi}/2$.

HW_7_code

May 6, 2022

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

0.1 Q3

[2]:
```
def quadrature(f):
    X = np.array([1/2, 1/2 + (1/2)*np.sqrt(3/5), 1/2 - (1/2)*np.sqrt(3/5)])
    W = np.array([4/9, 5/18, 5/18])
    y = np.array([f(x) for x in X])

    return np.dot(y, W)


def simpson(f, a=0, b=1):
    c = (a + b) / 2
    return ((b-a)/6) * (f(a) + 4*f(c) + f(b))
```

[3]:
```
error_qua_hist = []
error_sim_hist = []
for k in range(0, 8):
    f = lambda x: x**k
    exact = 1/(k+1)
    error_qua = abs(exact - quadrature(f))
    error_sim = abs(exact - simpson(f))
    error_qua_hist.append(error_qua)
    error_sim_hist.append(error_sim)
```

[4]:
```
k_space = list(range(0, 8))
plt.plot(k_space, error_qua_hist, label='Quadrature Rule')
plt.plot(k_space, error_sim_hist, label='Simpson Rule')
plt.title('Comparisons of error')
plt.legend()
plt.show()
```

- As we can see from the plot, the quadrature rule, which comes from hermite interpolation, is exact up to degree $2n + 1 = 5$.
- In contrast, as for simpson's rule, which comes from Lagrange interpolation, is exact up to degree $n = 2$.
- Note that, the plot reveal that it is exact up to degree $n = 3$. It is merely a coincidence. As the lagrange interpolation of x^3 at the three nodes gives us

$$p(x) = \frac{3}{2}x^2 - \frac{1}{2}x$$

and

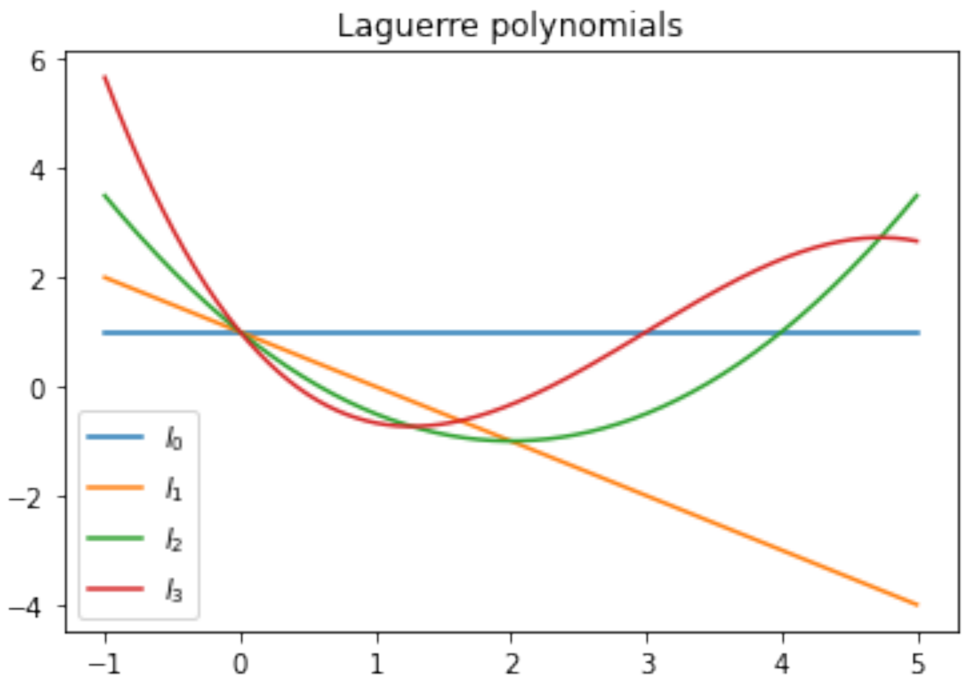
$$\int_0^1 x^3 dx = \int_0^1 p(x) dx = \frac{1}{4}$$

0.2 Q4

[5]:
```
l_0 = lambda x: 1
l_1 = lambda x: -x + 1
l_2 = lambda x: (1/2) * (x**2 - 4*x + 2)
l_3 = lambda x: (1/6) * (-x**3 + 9 * x**2 - 18*x + 6)
```

[6]:
```
x_space = np.linspace(-1, 5, 1001)
plt.plot(x_space, [l_0(x) for x in x_space], label=r'$l_0$')
plt.plot(x_space, [l_1(x) for x in x_space], label=r'$l_1$')
plt.plot(x_space, [l_2(x) for x in x_space], label=r'$l_2$')
plt.plot(x_space, [l_3(x) for x in x_space], label=r'$l_3$')
plt.legend()
```

```python
plt.title('Laguerre polynomials')
plt.show()
```



[7]:
```python
def quadrature(n, f):
    if n == 2:
        X = np.array([0.58576, 3.41421])
        W = np.array([0.853553, 0.146447])
    if n == 3:
        X = np.array([0.4157745, 2.29428, 6.28995])
        W = np.array([0.711093, 0.278518, 0.0103893])
    if n == 4:
        X = np.array([0.322548, 1.74576, 4.53662, 9.39507])
        W = np.array([0.603154, 0.357419, 0.0388879, 0.000539295])

    y = np.array([f(x) for x in X])

    return np.dot(y, W)
```

[8]:
```python
exact = 1/2
f = lambda x: np.exp(-x)
for n in range(2, 5):
    print(f'n = {n}, error = {abs(quadrature(n, f) - exact)}')
```

```
n = 2, error = 0.020023421556548896
n = 3, error = 0.0026968391830128335
```

```
n = 4, error = 0.00034432387742505677
```

[9]:
```
exact = np.pi ** .5 /2
f = lambda x: np.exp(-x**2 + x)
for n in range(2, 5):
    print(f'n = {n}, error = {abs(quadrature(n, f) - exact)}')
```

```
n = 2, error = 0.20176421812245082
n = 3, error = 0.0346771708037783
n = 4, error = 0.0385477868412718
```