# Assignment 2

**10 May 2020**

**MAM2046W 2NA**
**KDSMIL001**

## Analytical Problems

1. **Interpolating** $\sin(x)$

   (a) **Lagrange Method** We are trying to approximate the function $f(x) = \sin x$ using a Lagrange interpolating polynomial. We have 4 nodes $0, \pi/6, \pi/3, \pi/2$ and their function values $0, 1/2, \sqrt{3}/2, 1$ which we will use with the form of the Lagrange interpolating polynomial, which is given by

$$P_n(x) = \sum_{i=0}^{n} y_i L_i(x) \tag{1}$$

where

$$L_{i,n} = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \tag{2}$$

From Equation 2 we can find

$$\begin{aligned}
L_0(x) &= -\frac{36x^3}{\pi^3} + \frac{36x^2}{\pi^2} - \frac{11x}{\pi} + 1 \\
L_1(x) &= \frac{108x^3}{\pi^3} - \frac{90x^2}{\pi^2} - \frac{18x}{\pi} \\
L_2(x) &= -\frac{108x^3}{\pi^3} + \frac{72x^2}{\pi^2} - \frac{9x}{\pi} \\
L_3(x) &= \frac{36x^3}{\pi^3} - \frac{18x^2}{\pi^2} + \frac{2x}{\pi}
\end{aligned}$$

Which, when combined with Equation 1 gives us

$$P_3 = x^3 \left( \frac{90 - 54\sqrt{3}}{\pi^3} \right) + x^2 \left( \frac{-63 + 36\sqrt{3}}{\pi^2} \right) + x \left( \frac{22 - 9\sqrt{3}}{2\pi} \right) \tag{3}$$

(b) **Plotting**

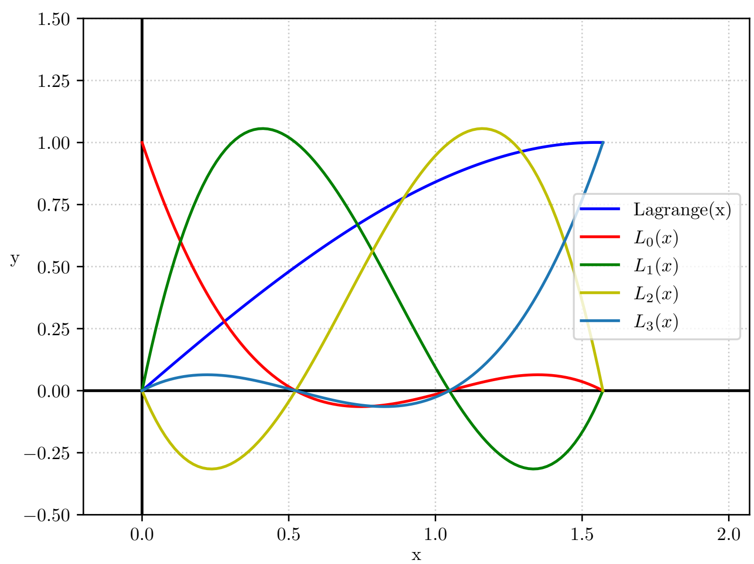


Figure 1: Lagrange Interpolating Polynomial on $[0, \pi/2]$

(c) **Newton Method**

Newton's method of finding the interpolating polynomial is slightly different. The polynomial is of the form

$$P_n(x) = f(x_0) + \sum_{i=1}^{n} f[x_0, \dots, x_i](x - x_0) \dots (x - x_{-1}) \tag{4}$$

where

$$f[x_0, x_1, x_2] = \frac{f[x_0, x_1] - f[x_1, x_2]}{x_0 - x_2} = \frac{\frac{f(x_0) - f(x_1)}{x_0 - x_1} - \frac{f(x_1) - f(x_2)}{x_1 - x_2}}{x_0 - x_2}$$

Finding $P_3(x)$ in this case takes a lot of calculation but is relatively trivial and gives us

$$P_3(x) = \frac{3x}{\pi} \frac{9(\sqrt{3} - 2)}{\pi^2}(x^2 - \frac{x\pi}{6}) + \frac{18(5 - 3\sqrt{3})}{\pi^3}(x^3 - \frac{x^2\pi}{2} + \frac{x\pi^2}{18}) \tag{5}$$

and plotting this gives us

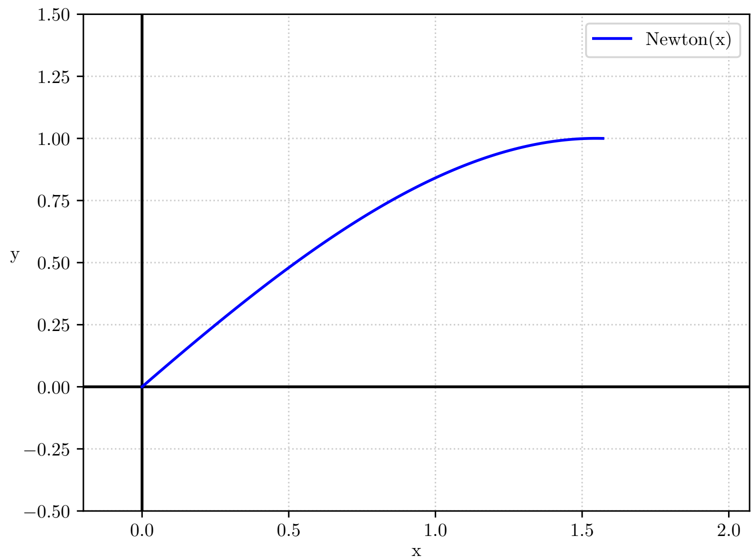


Figure 2: Newton Interpolating Polynomial on $[0, \pi/2]$

which is identical to Figure 1. They also simplify to the same thing if we check using Wolfram Alpha.

2. **Error Analysis**

(a) **Polynomial Approximation** The error bound formula for polynomial approximation $p_n(x)$ to a function $f(x)$ is given by

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)\dots(x - x_n) \tag{6}$$

and we are to show that it satisfies

$$|f(x) - p_3(x)| \le \frac{h^4}{24} \max_{\xi \in [x_0, x_3]} |f^4(\xi)| \tag{7}$$

where $x_0, x_1, x_2, x_3$ are equally spaced nodes with step-size $h$. In terms of interpreting Equation 7, we can see that Equation 6 gives us an upper and lower bound of the error of the polynomial. This means that taking the absolute value of that error bound will give us the maximum absolute value of the error. This simplifies things nicely and means that we can effectively ignore everything in Equation 7 apart from $\frac{h^4}{24}$, leaving us with having to show that

$$\frac{1}{4!}(x-x_0)\dots(x-x_n)=\frac{h^4}{24}$$

and we know that $4! = 24$. Now we can use a substitution to simplify the rest, which we'll call $w(x)$. If we let $x = t + x_1 + h/2$ and substitute in, we get

$$\begin{aligned}
w(t) =&(t+x_1+\frac{h}{2}-x_0)(t+x_1+\frac{h}{2}-x_1)(t+x_1+\frac{h}{2}-x_2)(t+x_1+\frac{h}{2}-x_3)\\
=&(t+\frac{3h}{2})(t+\frac{h}{2})(t-\frac{h}{2})(t-\frac{3h}{2})\\
=&(t^2-\frac{9h^2}{4})(t^2-\frac{h^2}{4})
\end{aligned}$$

We want to find the absolute maximum of this function, so we must find its stationary points and take the absolute values to find the max. Finding these stationary points is relatively trivial and we find them to be at $t_{root} = 0, \pm\sqrt{5/4}h^2$ with the function being at its absolute maximum at $t = 0$ with $w(0) = -h^4 \implies |w(0)| = h^4$, which is what we aimed to show.

(b) **Hermite Approximation**

Now we aim to show that, for a cubic Hermite Interpolating Polynomial, the error bound satisfies

$$|f(x)-H_3(x)| \leq \frac{(b-a)^4}{384}\max_{\xi\in[a,b]}|f^4(\xi)| \tag{8}$$

The error bound formula in question is

$$f(x)-H_{2n+1}(x)=\frac{f^{(2n+2)}(\xi)}{(2n+2)!}(x-x_0)^2\dots(x-x_n)^2 \tag{9}$$

which, similarly to Question 2a above, we can ignore most of. We need only focus on showing that

$$\begin{aligned}
\frac{1}{4!}(x-a)^2(x-b)^2 &= \frac{(b-a)^4}{384}\\
\implies (x-a)^2(x-b)^2 &= \frac{(b-a)^4}{16}
\end{aligned}$$

For simplicity, we'll call the left hand side of this $w(x)$, as before. We can also make the same substitution as before, letting $x = t + b + \frac{b-a}{2}$, which gives us

$$w(t) = (t + b - a + \frac{b-a}{2})^2(t + \frac{b-a}{2})^2$$
$$= (t + \frac{3(b-a)}{2})^2(t + \frac{b-a}{2})$$

Once again we find the absolute maximum by taking the derivative and setting it to 0, which is relatively trivial and gives us stationary points at $t_{root} = \frac{3(a-b)}{2}, \frac{(a-b)}{2}, (a-b)$, with $w(t)$ being at its maximum when $t = (a-b)$, with $w(a-b) = \frac{(a-b)^4}{16} \implies |w(a-b)| = \frac{(b-a)^4}{16}$. This is what we aimed to show.

## Numerical Problems

3. **Expanding the Lagrange Approximation to any** $x \in [0, \infty)$

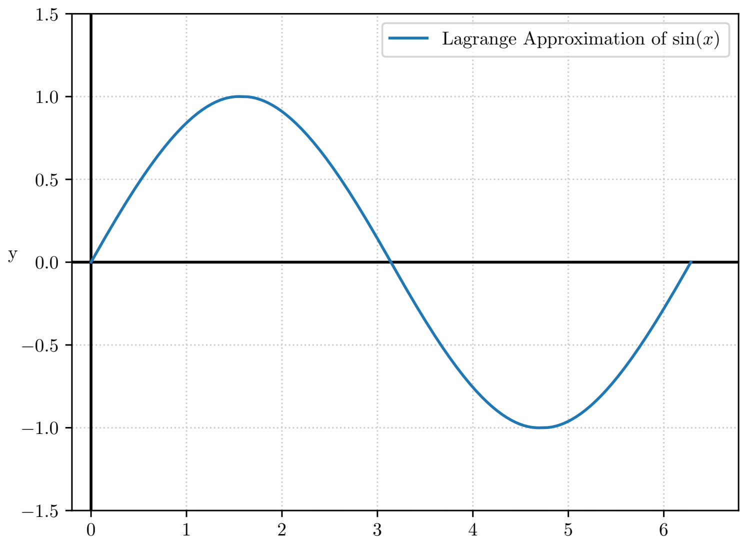


Figure 3: Lagrange Approximation Expanded onto $[0, 2\pi]$

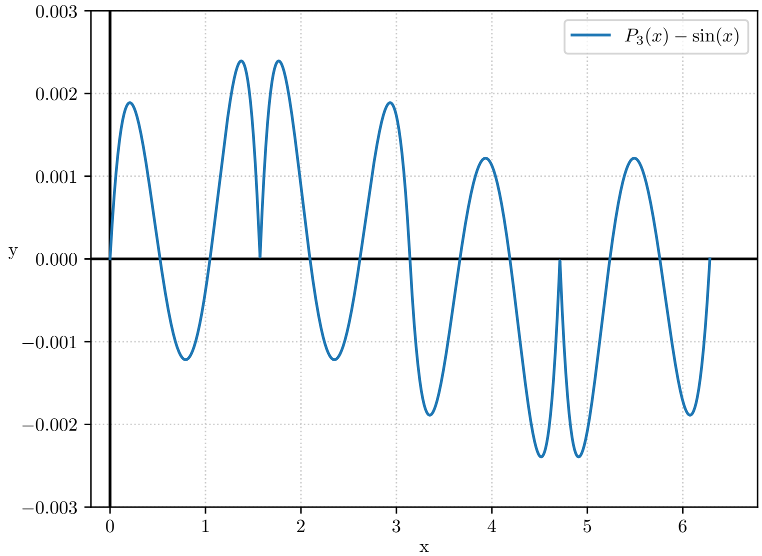


Figure 4: Expanded Lagrange Error

Code 1: Expanding Lagrange Approximation

```
from matplotlib import pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
from numpy import cos, pi, sin, sqrt, exp, random
import matplotlib
matplotlib.use('pgf')
matplotlib.rcParams.update({
    'pgf.texsystem': 'pdflatex',
    'font.family': 'serif',
    'text.usetex': True,
    'pgf.rcfonts': False,
})

def Lagrange(x):
    return ((x**3)*((90-(54*sqrt(3)))/(pi**3)))+((x**2)
        *((-63+(36*sqrt(3)))/(pi**2)))+((x)*((22-(9*sqrt(3)))/(2*
        pi)))

xs = np.linspace(0, 2*pi, 2000)
ys = np.zeros(2000)
zeros = np.zeros(1000)
axis = np.linspace(-10, 10, 1000)

```

```python
for i in range(xs.size):
    mod = xs[i]-np.floor(xs[i]/(2*pi))*(2*pi)
    if 0 < mod <= (pi/2):
        ys[i] = Lagrange(mod)
    elif (pi/2) < mod <= (pi):
        ys[i] = Lagrange(pi-mod)
    elif (pi) < mod <= (3*pi/2):
        ys[i] = Lagrange(mod-pi)*(-1)
    elif (3*pi/2) < mod <= (2*pi):
        ys[i] = Lagrange((2*pi)-mod)*(-1)

plt.plot(axis, zeros, '#000000')
plt.plot(zeros, axis, '#000000')
plt.plot(xs, ys, label='Lagrange Approximation of $\sin(x)$')
plt.grid(color='#CCCCCC', linestyle=':')
plt.xlabel('x')
plt.ylabel('y', rotation=0)
plt.legend()
plt.xlim(-0.2, (2*pi)+0.5)
plt.ylim(-1.5, 1.5)
# plt.show()
plt.savefig('2NA Assignments\Assignment 2\LagrangeExpanded.pgf')
plt.plot(axis, zeros, '#000000')
plt.plot(zeros, axis, '#000000')
plt.plot(xs, ys-sin(xs), label='$P_3(x)-\sin(x)$')
plt.grid(color='#CCCCCC', linestyle=':')
plt.xlabel('x')
plt.ylabel('y', rotation=0)
plt.legend()
plt.xlim(-0.2, (2*pi)+0.5)
plt.ylim(-.003, .003)
# plt.show()
plt.savefig('2NA Assignments\Assignment 2\LagrangeError.pgf')
```