Numerical Method
Purdue University
Spring 2021

Homework 3

Jialei Wang

1 Problem 1

Answers:

$$f(x) = \frac{1}{x} \quad f''(x) = 2x^{-3}$$

$$\begin{aligned} p(f;x) &= \frac{x-x_1}{x_0-x_1}f(x_0) + \frac{x-x_0}{x_1-x_0}f(x_1) \\ &= -(x-2)f(1) + (x-1)f(2) \\ &= -x + 2 + \frac{1}{2}x - \frac{1}{2} \\ &= \frac{3}{2} - \frac{1}{2}x \end{aligned}$$

$$\begin{aligned} f(x) - p_1(f;x) &= (x-x_0)(x-x_1)f''(\zeta(x))/2 \\ 2x^{-3} - (\frac{3}{2} - \frac{1}{2}x) &= (x^2-3x+2)(\zeta(x))^{-3} \\ \frac{1}{2}x^{-1}(2-3x+x^2) &= (x^2-3x+2)(\zeta(x))^{-3} \\ (\zeta(x))^{-3} &= (2x)^{-1} \\ \zeta(x) &= (2x)^{1/3} \end{aligned}$$

The maximum and minimum point of function $\zeta(x)$ with the range of $x \in [1,2]$ is the end point.

$$\max_{1\le x\le 2} \zeta(x) = 1.5874, \quad x = 2$$

$$\min_{1\le x\le 2} \zeta(x) = 1.2599, \quad x = 1$$

2 Problem 2

1. Code for spline interpolation function

```
function spline_interpolation(f, x, xn)
    n = length(xn)
    A = zeros((n, n))
    b = zeros((n,))
    f_inter = zeros((length(x),))
    # Compute values of m
    dx = xn[2:n] .- xn[1:n-1]
    df = (f[2:n] .- f[1:n-1]) ./ dx
    A[1, 1:2] = [2, 1]
    b[1] = 3 * df[1]
    for i in range(2, stop = n - 1)
        A[i, i-1:i+1] = [dx[i], 2 * (dx[i-1] + dx[i]), 2 * dx[i-1]]
        b[i] = 3 * (dx[i] * df[i-1] + dx[i-1] * df[i])
    end
    A[n, [n - 1, n]] = [1, 2]
    b[n] = 3 * df[n-1]
    m = inv(A) * b
    # compute values of c0 to c3
    c0 = f[1:n-1]
    c1 = m[1:n-1]
    c3 = (m[2:n] .+ c1 .- 2 .* df) ./ (dx .^ 2)
    c2 = (df .- m[1:n-1]) ./ dx .- c3 .* dx
    # Calculate x
    j = 1
    for i in range(1, stop = n - 1)
        while (j <= length(x)) && (x[j] <= xn[i+1])
            f_inter[j] =
                c0[i] +
                c1[i] * (x[j] - xn[i]) +
                c2[i] * (x[j] - xn[i]) .^ 2 +
                c3[i] * (x[j] - xn[i]) .^ 3
            j = j + 1
        end
    end
    return f_inter
end
```

2. Test the function on $f(x) = e^{-x}$. The maximum error happens at $x = 0.06006006006006006$ where the error is 0.003139346615462779.

```
println("Spline interpolation of f(x)=e^-x")
f(x) = exp.(-x)
xn = range(0, stop = 1, length = 11)
fn = f(xn)
x = range(0, stop = 1, length = 1000)
fx = f(x)
```

```
f_inter = spline_interpolation(fn, x, xn)
inter_error = abs.(f_inter - fx)
max_error = findmax(inter_error)
println(max_error)
println(x[max_error[2]])
plot(x, fx)
display(plot!(x, f_inter))
```

3. Test the function on $f(x) = x^{5/2}$ using uniformly distributed interpolation points. The maximum error happens at $x = 0.93993993993994$ where the error is 0.013501626367536823.

```
println("Spline interpolation of f(x)=x^(5/2)")
f(x) = x .^ (5 / 2)
xn = range(0, stop = 1, length = 11)
fn = f(xn)
x = range(0, stop = 1, length = 1000)
fx = f(x)
f_inter = spline_interpolation(fn, x, xn)
inter_error = abs.(f_inter - fx)
max_error = findmax(inter_error)
println(max_error)
println(x[max_error[2]])
plot(x, fx)
display(plot!(x, f_inter))
```

4. Test the function on $f(x) = x^{5/2}$ using 11 points $x_i = \left(\frac{i-1}{n-1}\right)^2$. The maximum error happens at $x = 0.8808808808808809$ where the error is 0.020264760345926947.

```
println("Spline interpolation of f(x)=x^(5/2)")
f(x) = x .^ (5 / 2)
xn = range(1, stop = 11)
xn = ((xn .- 1) ./ 10) .^ 2
fn = f(xn)
x = range(0, stop = 1, length = 1000)
fx = f(x)
f_inter = spline_interpolation(fn, x, xn)
inter_error = abs.(f_inter - fx)
max_error = findmax(inter_error)
println(max_error)
println(x[max_error[2]])
plot(x, fx)
display(plot!(x, f_inter))
```

5. Test the function on $f(x) = \frac{1}{1+x^2}$ using 25 uniformly distributed interpolation points between -5 and 5. The maximum error happens at $x = 0.5368536853685368$ where the error is 0.005894125285141327.

```
println("Spline interpolation of f(x)=1/(1+x^2)")
f(x) = 1 ./ (1 .+ x .^ 2)
xn = range(-5, stop = 5, length = 25)
```

```
fn = f(xn)
x = range(0, stop = 1, length = 10000)
fx = f(x)
f_inter = spline_interpolation(fn, x, xn)
inter_error = abs.(f_inter - fx)
max_error = findmax(inter_error)
println(max_error)
println(x[max_error[2]])
plot(x, fx)
display(plot!(x, f_inter))
```

3 Problem 3

1. The derivative of the three functions.

   For all the parts of the problem, the function will be numbered as:

$$f_1(x) = e^x$$

$$f_2(x) = \frac{1}{1+x^2}$$

$$f_3(x) = \frac{e^{3x}\sin(200x^2)}{1+20x^2}$$

   The derivative of the threes functions are:

$$f_1'(x) = e^x$$

$$f_2'(x) = -\frac{2x}{(1+x^2)^2}$$

$$f_3'(x) = \frac{3e^{3x}\sin(200x^2)}{1+20x^2} - \frac{40e^{3x}x\sin(200x^2)}{(1+20x^2)^2} + \frac{400e^{3x}\cos(200x^2)}{1+20x^2}$$

   The plot of three derivatives is shown below:

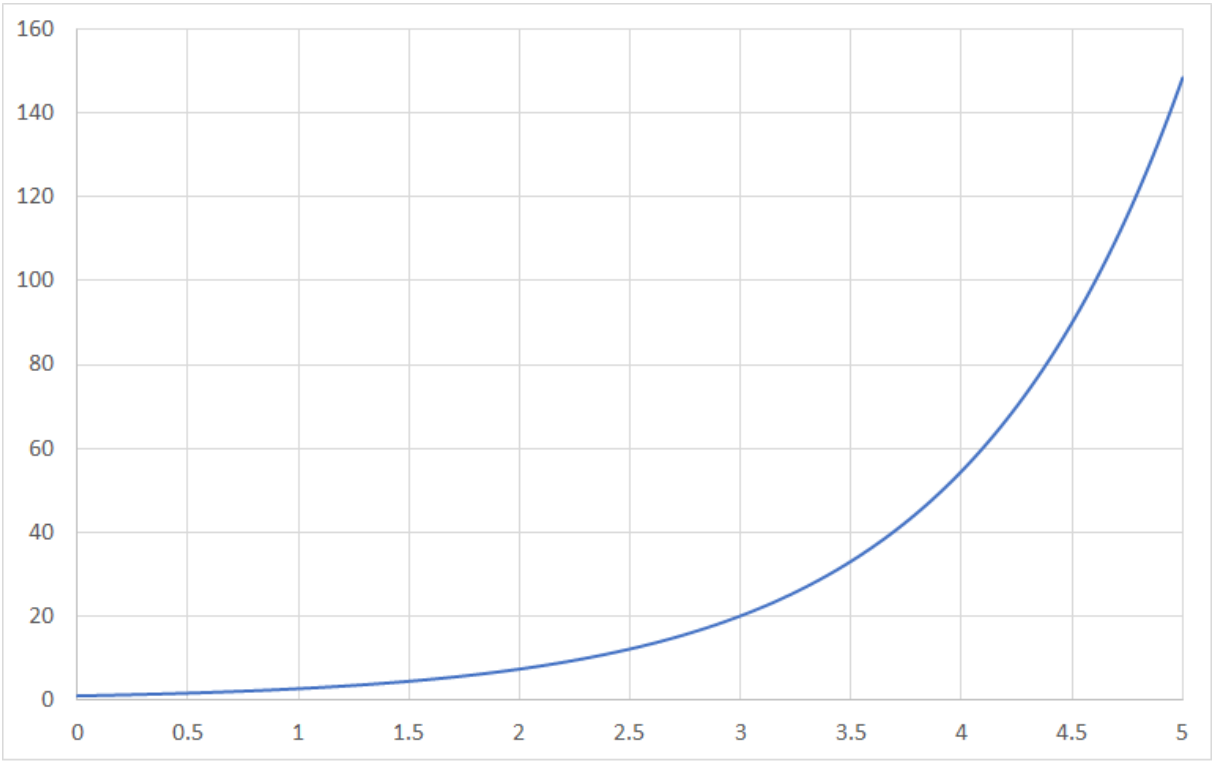


Figure 1: Plot of the expected derivative of $f_1(x) = e^x$

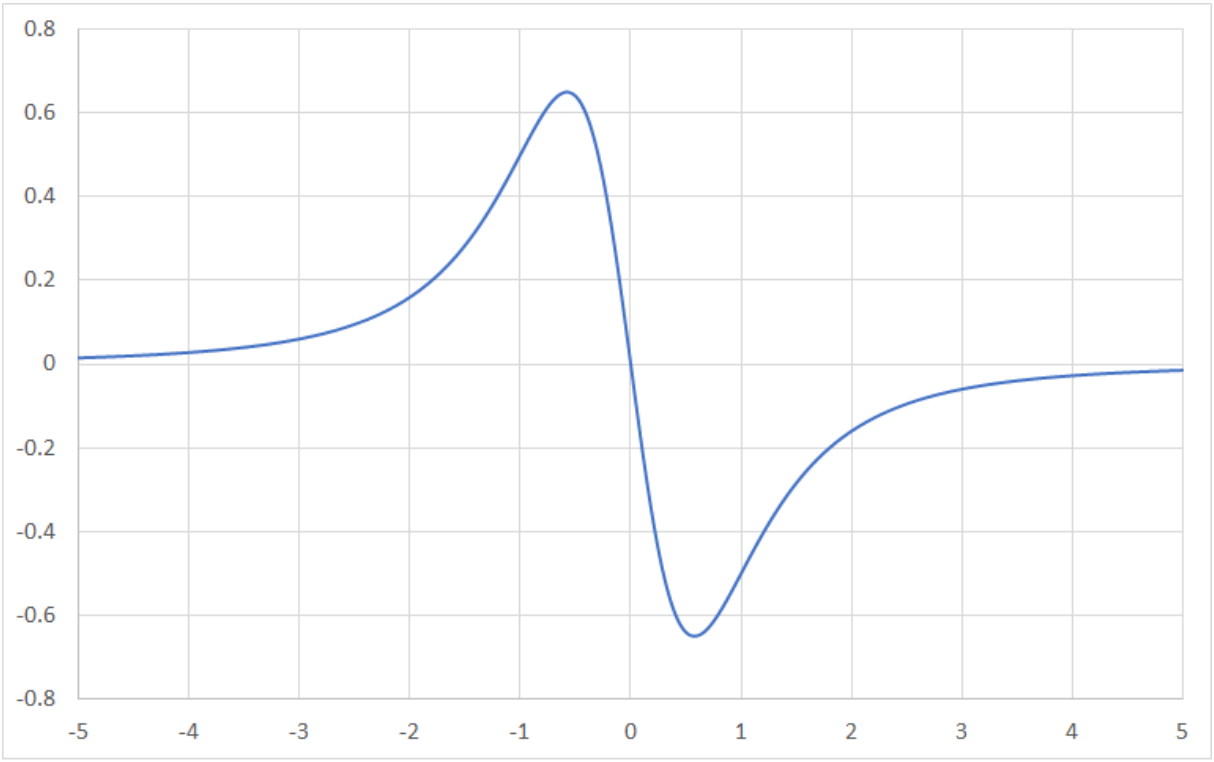


Figure 2: Plot of the expected derivative of $f_2(x) = \frac{1}{1+x^2}$

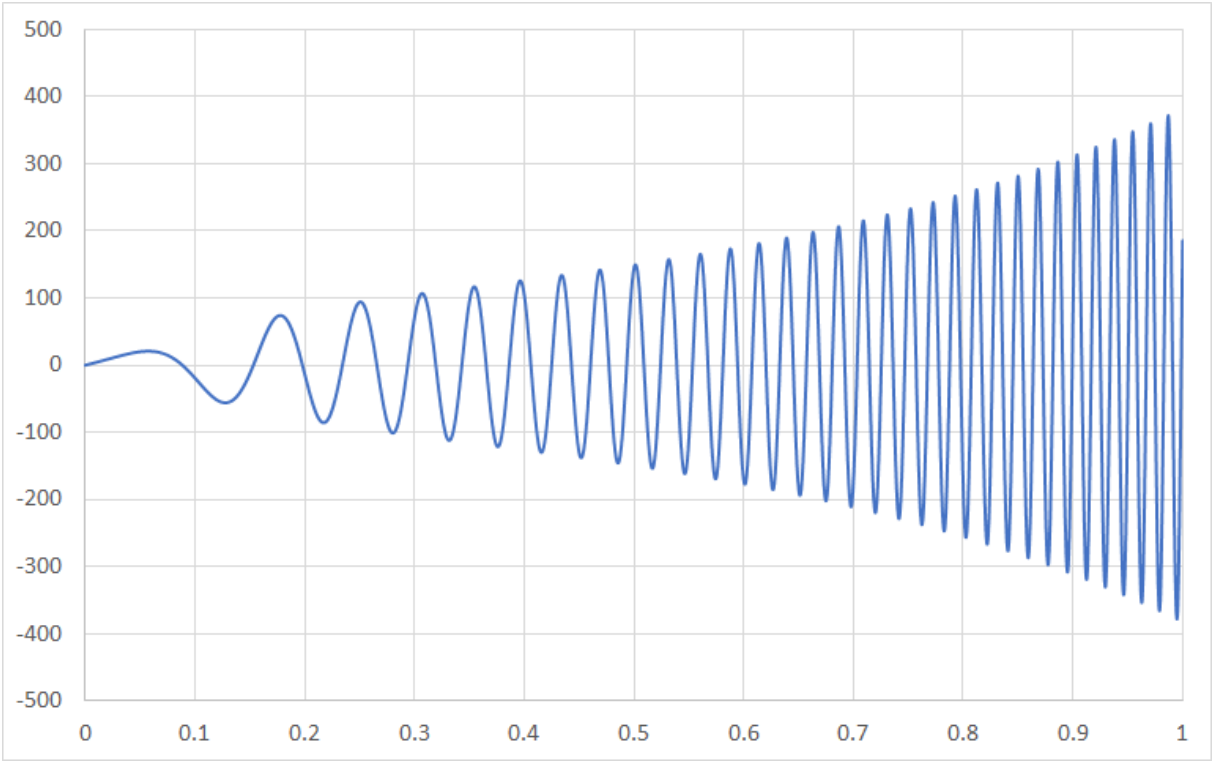
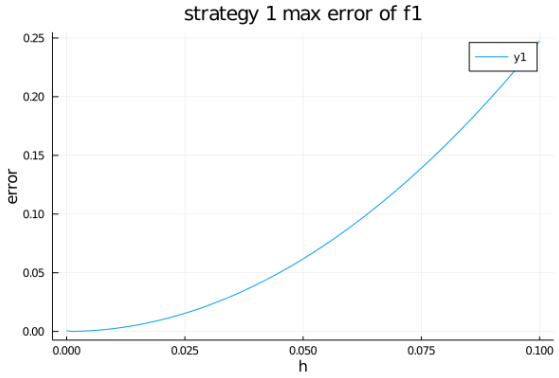


Figure 3: Plot of the expected derivative of $f_3(x) = \frac{e^{3x}\sin(200x^2)}{1+20x^2}$

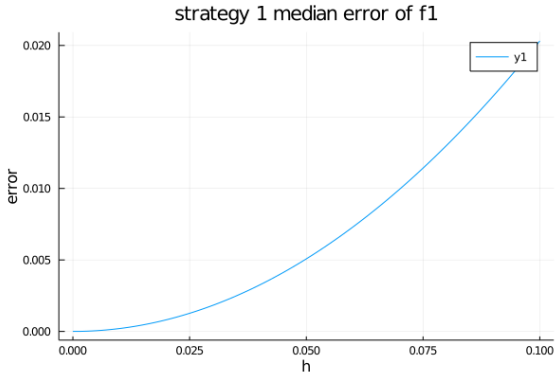
2. Code for implement strategy 1

```
function derivative_1(f, x, h)
    n = length(x)
    df = zeros((n, ))
    df = (f(x .+ h) .- f(x .- h)) ./ (2 .* h)
    return df
end
```

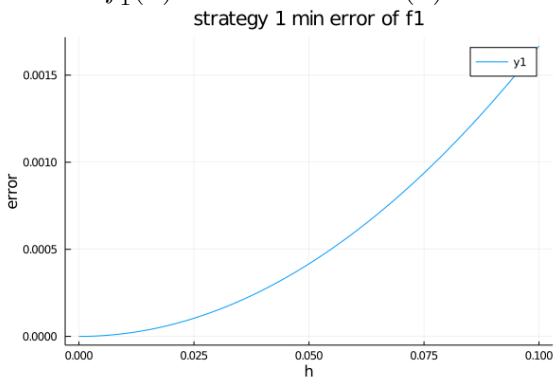
The largest, median, and smallest error in the interpolation with respect the values of h is shown in the picture below:



(a) Maximum Error of $f_1'(x)$

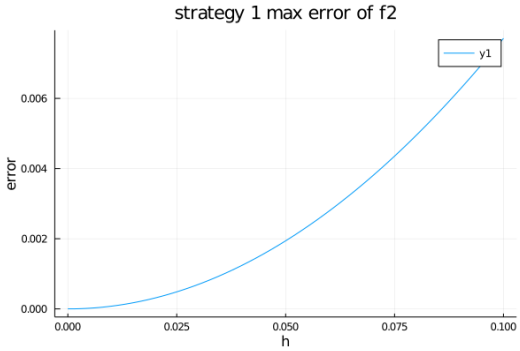


(b) Median Error of $f_1'(x)$



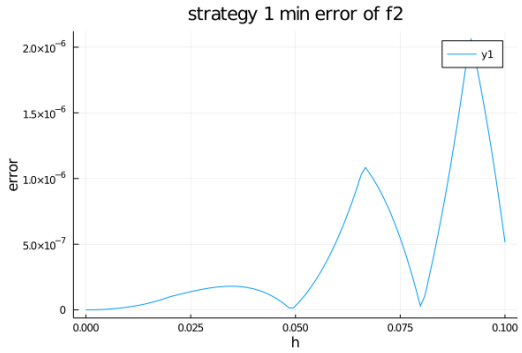
(c) Minimum Error of $f_1'(x)$

Figure 4: Error plot for derivative of $f_1(x)$ using strategy 1



(a) Maximum Error of $f_2'(x)$

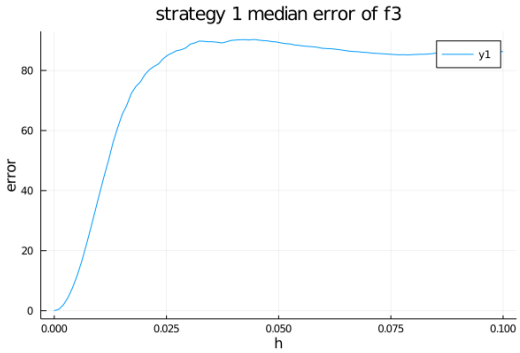
(b) Median Error of $f_2'(x)$



(c) Minimum Error of $f_2'(x)$

Figure 5: Error plot for derivative of $f_2(x)$ using strategy 1

(a) Maximum Error of $f_3'(x)$



(b) Median Error of $f_3'(x)$

(c) Minimum Error of $f_3'(x)$

Figure 6: Error plot for derivative of $f_3(x)$ using strategy 1

3. The expression for the derivative of the Lagrange polynomial is:

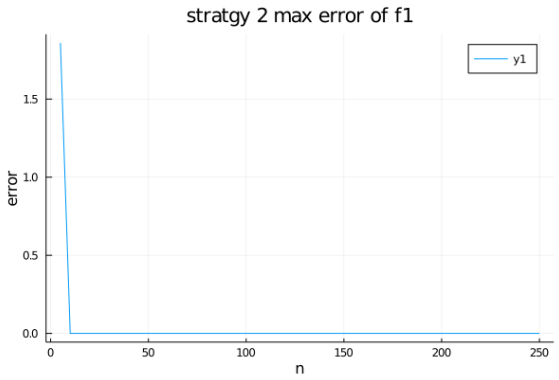
$$l_i(x) = \prod_{\substack{j=0 \\ i \neq j}}^{k} \frac{x - x_j}{x_i - x_j}$$

$$l_i'(x) = \sum_{\substack{j=0 \\ i \neq j}}^{k} \left[\frac{1}{x_i - x_j} \prod_{\substack{m=0 \\ m \neq i \\ m \neq j}}^{k} \frac{x - x_m}{x_i - x_m} \right]$$

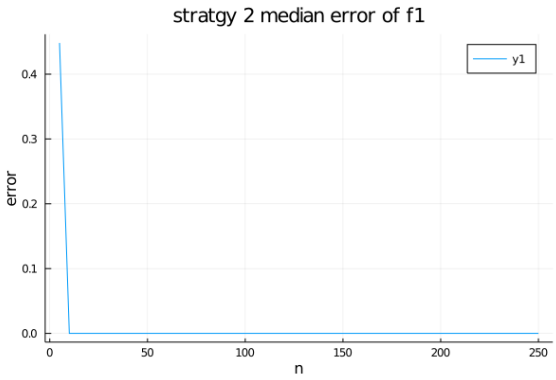
4. Code for implement strategy 2

```
function derivative_2(f, r, n, pts)
    k = 0:n
    x_k = cos.(k .* (pi / n))
    x_k = (r[2] + r[1]) / 2 .+ (r[2] - r[1]) / 2 .* x_k
    y = zeros((pts,))
    x = range(r[1], stop=r[2], length=pts)
    w = ones(Float64, (n + 1, ))
    f_k = f(x_k)
    for ind_x in 1:pts
        l_j = zeros((length(f_k),))
        for j in 1:length(k)
            for i in 1:length(k)
                product = 1
                for m in 1:length(k)
                    if (m != j && m != i)
                        product *= (x[ind_x] - x_k[m])   /
                                (x_k[j] - x_k[m])
                    end
                end
                if (i != j)
                    l_j[j] += product / (x_k[j] - x_k[i])
                end
            end
        end
        y[ind_x] = sum(f_k .* l_j)
    end
    return y
```

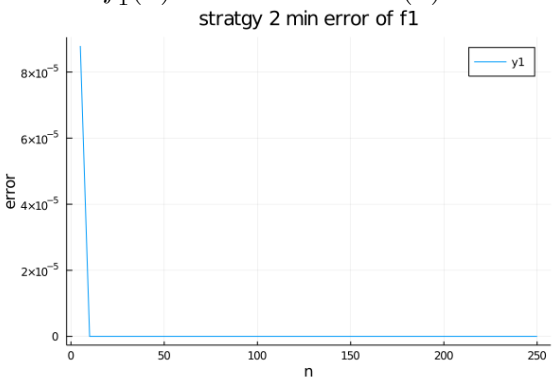
The largest, median, and smallest error in the interpolation with respect the values of n is shown in the picture below:



(a) Maximum Error of $f_1'(x)$

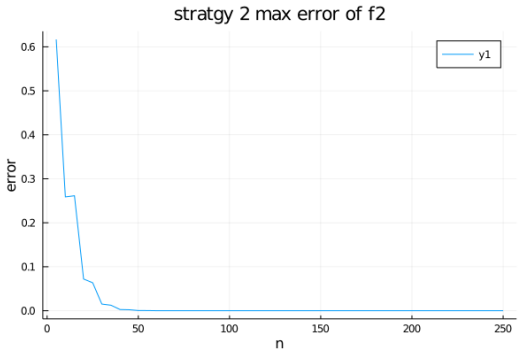


(b) Median Error of $f_1'(x)$

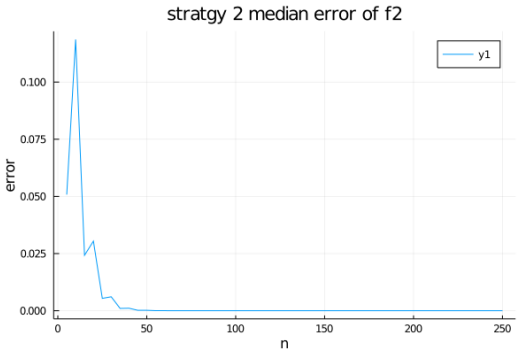


(c) Minimum Error of $f_1'(x)$

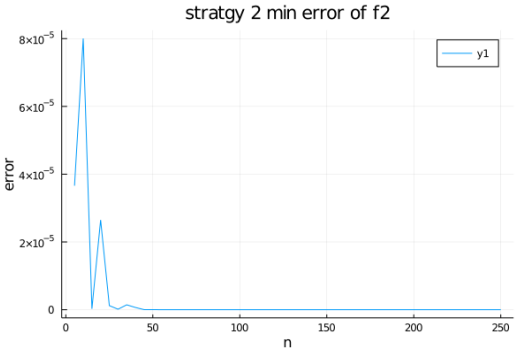
Figure 7: Error plot for derivative of $f_1(x)$ using strategy 2



(a) Maximum Error of $f_2'(x)$



(b) Median Error of $f_2'(x)$

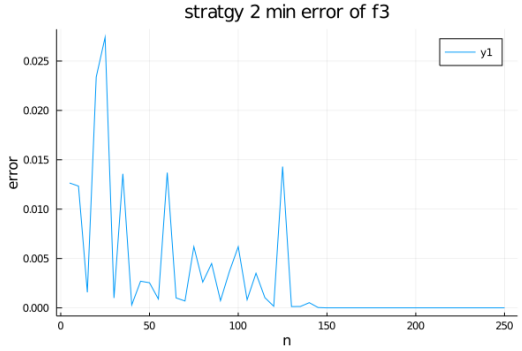


(c) Minimum Error of $f_2'(x)$

Figure 8: Error plot for derivative of $f_2(x)$ using strategy 2

(a) Maximum Error of $f_3'(x)$

(b) Median Error of $f_3'(x)$



(c) Minimum Error of $f_3'(x)$

Figure 9: Error plot for derivative of $f_3(x)$ using strategy 2

5. Discussion of the advantage and disadvantage of strategy 1 and 2

   Strategy 1:

   Advantage: Computationally efficient.

   Disadvantage: The error can be high at locations where is there is shape concave and convex shape. The reason for the high error is because the function between the points samples does not approximate well with linear interpolation. The most notable example is the approximation of $f_3'(x)$, the maximum error is steadily at about 370.

   Strategy 2:

   Advantage: When number of points used for interpolation is high enough, there is barely any interpolation error. For all three functions, the error drops significantly as the number of interpolation points increases.

   Disadvantage: It is requires a lot of computation power. It takes about 30 minutes to approximate 10000 points of derivative when using 250 interpolation points. In comparison, the time it take to approximate 10000 derivative points is near instant.

4 Problem 4

1. The formula to approximate the first derivative is shown as:

$$\begin{aligned} x_{-1} &= x_0 - 2h \\ x_0 &= x_0 \\ x_1 &= x_0 + h \\ x_2 &= x_0 + \frac{3}{2}h \end{aligned}$$

x_{-1}	f_{-1}			
x_0	f_0	$\frac{f_0-f_{-1}}{2h}$		
x_1	f_1	$\frac{f_1-f_0}{2h}$	$\frac{2f_1-3f_0+f_{-1}}{6h^2}$	
x_2	f_2	$\frac{2f_2-2f_1}{2h}$	$\frac{4f_2-6f_0+2f_0}{3h^2}$	$\frac{2(8f_2-14f_1+7f_0-f_{-1}}{42h^3}$

$$\begin{aligned} p_n(f;x) =& f_{-1} + (x-x_0)[x_{-1},x_0]f \\ &+ (x-x_0)(x-x_1)[x_{-1},x_0,x_1]f \\ &+ (x-x_0)(x-x_1)(x-x_2)[x_{-1},x_0,x_1,x_2]f \end{aligned}$$

$$\begin{aligned} p_n'(f;x) &= [x_{-1},x_0]f + (x-x_1)[x_{-1},x_0,x_1]f + (x-x_1)(x-x_2)[x_{-1},x_0,x_1,x_2]f \\ &= \frac{f_0-f_{-1}}{2h} + (-h)\frac{2f_1-3f_0+f_{-1}}{6h^2} + (\frac{3}{2}h^2)\frac{2(8f_2-14f_1+7f_0-f_{-1})}{42h^3} \\ &= \frac{24f_2-56f_1+63f_0-31f_{-1}}{42h} \end{aligned}$$

2. The formula to approximate the second derivative is shown as:

$$\begin{aligned} p''(f;x_0) &= 2[x_{-1},x_0,x_1]f + (2x_0-x_1-x_2)[x_{-1},x_0,x_1,x_2]f \\ &= 2\frac{2f_1-3f_0+f_{-1}}{6h^2} + (-5h)\frac{2(8f_2-14f_1+7f_0-f_{-1})}{42h^3} \\ &= \frac{28f_1-42f_0+14f_{-1}}{42h^2} + \frac{80f_2-140f_1+70f_0-10f_{-1}}{42h^2} \\ &= \frac{-80f_2+168f_1-112f_0+24f_{-1}}{42h^2} \end{aligned}$$

3. Since the 1st and 2nd derivative of e^x is the same, the result of the above formula should be the same. The code that is used to generate the plot is shown below:

```
using Plots

function derivative_1st(f, x_vec, h)
    n = length(x)
    df = zeros((n, ))
    coeff = [-31, 63, -56, 24]
    for i in 1:n
```

```
        x = x_vec[i]
        xn = [x-2*h, x, x+h, x+1.5*h]
        df[i] = sum(f(xn) .* coeff) / (42 * h)
    end
    return df
end

function derivative_2nd(f, x_vec, h)
    n = length(x)
    df = zeros((n, ))
    # coeff = [17, -91, 154, -80]
    coeff = [24, -112, 168, -80]
    for i in 1:n
        x = x_vec[i]
        xn = [x-2*h, x, x+h, x+1.5*h]
        df[i] = sum(f(xn) .* coeff) / (42 * h^2)
    end
    return df
end

h = 1e-5
f(x) = exp.(x)
fp(x) = exp.(x)
x = range(0, stop=5, length=1000)
df = derivative_1st(f, x, h)
df2 = derivative_2nd(f, x, h)
plot(x, df, label="1st derivative approx")
plot!(x, df2, label="2nd derivative approx")
plot!(x, fp(x), label="analytical")
png("unequal_der.png")
```

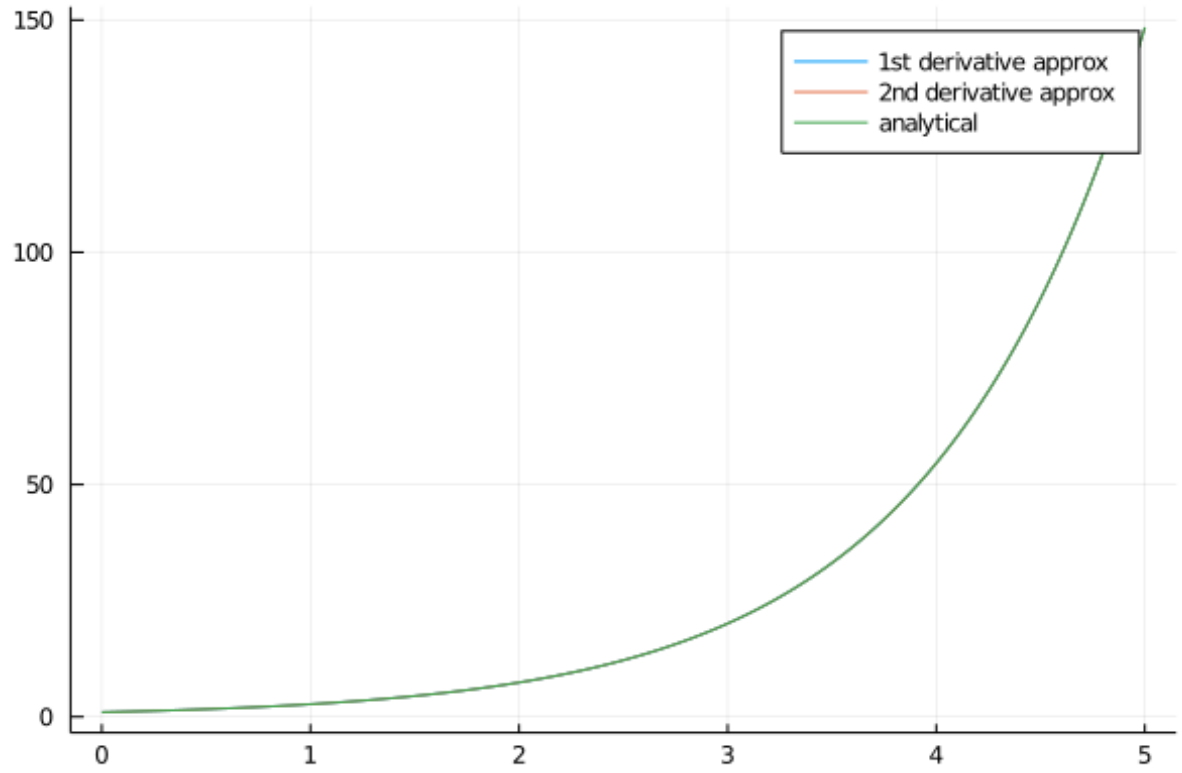


Figure 10: Plot of the 1st and 2nd derivative approximation and the analytical solution

From the plot, both the 1st and 2nd derivative approximation shows good agreement with the analytical solution for $0 < x < 5$.

5 Problem 5

1. The value of a and b can be computed substitute $f(x) = \cos(x)$ and $f(x) = \sin(x)$ which will give two linear equation and two unknown.

$$\begin{aligned} \int_0^h \cos(x) &= a + b\cos(h) \\ \sin(h) &= a + b\cos(h) \end{aligned} \tag{1}$$

$$\begin{aligned} \int_0^h \sin(x) &= b\sin(h) \\ 1 - \cos(h) &= b\sin(h) \end{aligned} \tag{2}$$

From equation 2, the value of b can be calculated.

$$b = \frac{1-\cos(h)}{\sin(h)} = \tan\left(\frac{h}{2}\right)$$

The value of a can then be calculated.

$$\begin{aligned} \sin(h) &= a + \tan\left(\frac{h}{2}\right)\cos(h) \\ a &= \sin(h) - \tan\left(\frac{h}{2}\right)\cos(h) \\ &= 2\sin(h/2)\cos(h/2) - \tan(h/2)[\cos^2(h/2) + \sin^2(h/2)] \\ &= \sin(h/2)\cos(h/2) + \tan(h/2)\sin^2(h/2) \\ &= \sin(h/2)\left[\cos(h/2) + \frac{\sin^2(h/2)}{\cos(h/2)}\right] \\ &= \sin(h/2)\frac{\cos^2(h/2) + \sin^2(h/2)}{\cos(h/2)} \\ &= \frac{\sin(h/2)}{\cos(h/2)} \\ &= \tan(h/2) \end{aligned}$$

The formula becomes:

$$\int_0^h f(x) = \tan(h/2)(f(0) + f(h))$$

As $h \to 0$, $\tan(h/2) \to h/2$, the formula will then be in a trapezoidal-like formula.

2. The error of the approximation when $f(x)$ is a constant is:

$$E(1) = h - 2\tan(h/2)$$

By using the Taylor expansion of $\tan(x)$,

$$E(1) = h - (\frac{1}{2}h + \frac{1}{24}h^3 + \dots) \neq 0$$

This shows the formula from part 1 cannot approximate a constant function exactly.

6 Problem 6

The formula for the area of a unit disk is:

$$\begin{aligned} A &= 2\int_{-1}^{1}(1-t^2)^{1/2}dt \\ &= 2\int_{-1}^{1}(1-t^2)(1-t^2)^{-1/2}dt \end{aligned}$$

The equation matches with $\int_{-1}^{1} f(t)(1-t^2)^{-1/2}dt$. In the case of this problem, $f(t) = 1-t^2$. The area of the unit disk can calculated as:

$$\begin{aligned} A &= 2\left[\frac{\pi}{n}\sum_{k=1}^{n} f(t_k^C)\right] \\ &= 2\left[\frac{\pi}{n}\sum_{k=1}^{n} 1-\cos^2\left(\frac{2k-1}{2n}\pi\right)\right] \\ &= 2\left[\frac{\pi}{n}\left(n-\sum_{k=1}^{n}\frac{1}{2}\left(1+\cos\left(\frac{2k-1}{n}\pi\right)\right)\right)\right] \\ &= 2\left[\frac{\pi}{n}\left(\frac{n}{2}-\sum_{k=1}^{n}\frac{1}{2}\cos\left(\frac{2k-1}{n}\pi\right)\right)\right] \end{aligned}$$

Since $\sum_{k=1}^{n}\cos\left(\frac{2k-1}{n}\pi\right) = 0$,

$$A = 2\cdot\frac{\pi}{n}\cdot\frac{n}{2} = \pi$$

Problem 0

I have worked on this homework on my own.