MATH 315, Fall 2020
Homework 7

James Nagy

Homework Team *(6)*: *(Diego Gonzalez, Yuting Hou, Julie Levine, Yolanda Zheng)*

NOTE: We are returning to team submissions for this assignment. Submit only one set of solutions for all team members.

This assignment is associated with Chapter 5, Differentiation and Integration. It illustrates numerical methods for differentiating a function at a point and across an interval, and also some of the resulting errors including those caused by implementing these methods on a computer.

1. Use Taylor series to show that the one-sided difference approximation

$$f'(x) \approx \frac{-3f(x) + 4f(x+h) - f(x+2h)}{2h}$$

is second order in h.

Answer:

$f(x+h) = f(x) + f'(x)h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \frac{f^{iv}(x)}{4!}h^4 + \cdots\cdots$

$4f(x+h) = 4f(x) + 4f'(x)h + \frac{4f''(x)}{2!}h^2 + \frac{4f'''(x)}{3!}h^3 + \frac{4f^{iv}(x)}{4!}h^4 + \cdots\cdots$

$f(x+2h) = f(x) + f'(x)2h + \frac{f''(x)}{2!}(2h)^2 + \frac{f'''(x)}{3!}(2h)^3 + \frac{f^{iv}(x)}{4!}(2h)^4 + \cdots\cdots$

$4f(x+h) - f(x+2h) = 3f(x) + 2f'(x)h + \frac{(-4)f'''(x)}{3!}h^3 + \frac{(-12)f^{iv}(x)}{4!}h^4 + \cdots\cdots$

$-3f(x) + 4f(x+h) - f(x+2h) = 2f'(x)h + \frac{(-4)f'''(x)}{3!}h^3 + \frac{(-12)f^{iv}(x)}{4!}h^4 + \cdots\cdots$

$\frac{-3f(x) + 4f(x+h) - f(x+2h)}{2h} = f'(x) - \frac{2f'''(x)}{3!}h^2 - \frac{6f^{iv}(x)}{4!}h^3 + \cdots\cdots$

truncate when h is small

Hence, $f'(x) \approx \frac{-3f(x)+4f(x+h)-f(x+2h)}{2h}$

$\therefore$ The order of the one-sided difference approximation is $O(h^2)$.

2. (a) Write a short MATLAB script to "reproduce" the data in Table 5.1 of your text for the derivative of $f(x) = \frac{1}{1+25x^2}$ at $x = 1$. Create a similar table that includes a third column that implements the second order one-sided difference approximation given in Problem 1. Compute an additional table showing the absolute value of the error. Discuss your results. When does round off error begin to dominate for each approximation?

Answer:

Table 1: Approximations

NUMERICAL DERIVATIVES:

n	First Order	Second Order	2nd. Order 1 Sided
0	-0.02856054836253	-0.49504950495050	-0.04010271687482
1	-0.06461538461538	-0.07529411764706	-0.07205821205821
2	-0.07292490548787	-0.07397762848085	-0.07393912262670
3	-0.07385937101934	-0.07396462833950	-0.07396423535097
4	-0.07395397263736	-0.07396449835440	-0.07396449441636
5	-0.07396344448365	-0.07396449705453	-0.07396449701706
6	-0.07396439178492	-0.07396449704100	-0.07396449704100
7	-0.07396448661878	-0.07396449706182	-0.07396449726998
8	-0.07396449563934	-0.07396449668018	-0.07396449598629
9	-0.07396450812935	-0.07396449772101	-0.07396451853769
10	-0.07396451506825	-0.07396451506825	-0.07396451506825
11	-0.07396513956870	-0.07396479262400	-0.07396583345809
12	-0.07397554790955	-0.07396860901565	-0.07398595625041
13	-0.07389922007661	-0.07393391454613	-0.07379513666805
14	-0.07424616477181	-0.07389922007661	-0.07459310946700
15	-0.08326672684689	-0.07632783294298	-0.09020562075079
16	0.00000000000000	-0.03469446951954	0.10408340855861

Table 2: Absolute Value of the Error

ABSOLUTE VALUE OF ERRORS:

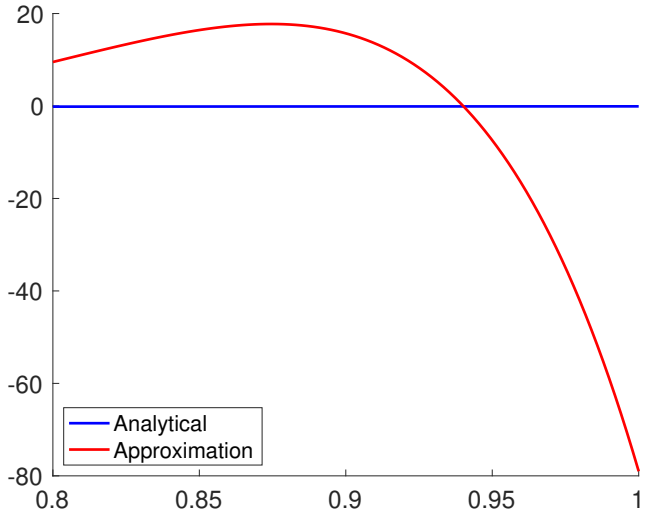
n	First Order	Second Order	2nd. Order 1 Sided
0	0.04540394867889	0.42108500790907	0.03386178016660
1	0.00934911242604	0.00132962060564	0.00190628498321
2	0.00103959155355	0.00001313143943	0.00002537441472
3	0.00010512602208	0.00000013129808	0.00000026169045
4	0.00001052440406	0.00000000131298	0.00000000262506
5	0.00000105255777	0.00000000001311	0.00000000002436
6	0.00000010525650	0.00000000000042	0.00000000000042
7	0.00000001042264	0.00000000002040	0.00000000022856
8	0.00000000140208	0.00000000036124	0.00000000105513
9	0.00000001108793	0.00000000067959	0.00000002149627
10	0.00000001802683	0.00000001802683	0.00000001802683
11	0.00000064252728	0.00000029558258	0.00000133641667
12	0.00001105086813	0.00000411197423	0.00002145920899
13	0.00006527696481	0.00003058249529	0.00016936037337
14	0.00028166773039	0.00006527696481	0.00062861242558
15	0.00930222980547	0.00236333590156	0.01624112370937
16	0.07396449704142	0.03927002752188	0.17804790560003

The results indicate that as n increases the error decreases until the round off error begins to dominate again. For the first order approximation, we can see that the error decreases and is at its lowest value when n = 8, then it starts to increase again as n gets larger. For the second order approximation, we can see that the error decreases up until n = 6 before and then starts to increase as n increases. For the second order one sided difference approximation, the error also decreases until n = 6 then starts to increase as n increases. By comparing the smallest error we see in each approximation, we see the error is the smallest for the

second order and one sided difference approximations. This makes sense since the order of the second order approximation and the second order one-sided difference approximation is $O(h^2)$ while the order of the first order approximation is $O(h)$. So when h is really small, h^2 is much smaller than h, so the first term that we ignore from the Taylor series is smaller for the second order approximation and thus, we'd expect the absolute error to be smaller. The round off error begins to dominate in these functions right after we see the error hit its lowest value and before the error hit its lowest value when n is small. For first order we see the round off error begin to dominate at n = 0 and get smaller as n increases and at n = 9 again after reaching its minimum at n=8, while for the second order and one sided difference we see the round off begin to dominate at n = 0 and get smaller as n get smaller and at n = 7 again after reaching its minimum at n=6.

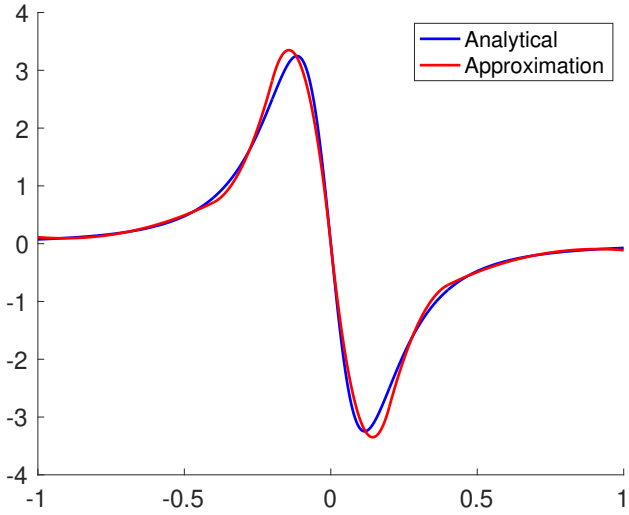
(b) Use `polyfit` to determine the unique 10th order polynomial that interpolates $\frac{1}{1+25x^2}$ at equally spaced points on the interval $[-1, 1]$. Differentiate and evaluate this polynomial using `syms`, `diff`, and `subs` to approximate the derivative at $x = 1$. Create a plot that shows the analytical derivative of $\frac{1}{1+25x^2}$ on the interval $[0.8, 1]$ and the result obtained using this approximation (i.e. the derivative of the function on the interval $[0.8, 1]$). *Hint: you can use* `double(subs(f,x))` *to evaluate the symbolic function for the vector of values,* `x`.

Answer: The evaluated derivative at x=1 is -79.0158.

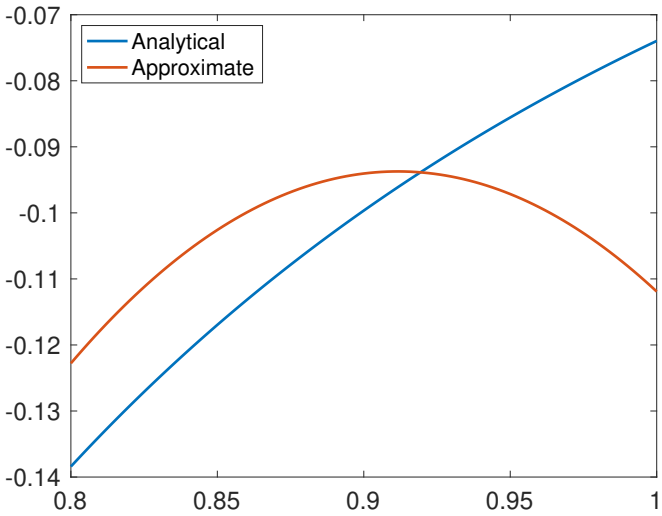


(c) Use `spline` and `unmkpp` to determine the spline that interpolates $\frac{1}{1+25x^2}$ on the interval $[-1, 1]$ (use the same interpolation points you used in Problem 2b). Differentiate and evaluate this appropriate piecewise polynomial using `syms`, `diff`, and `subs` to approximate the derivative at $x = 1$. Create a plot that shows the analytical derivative and the result obtained using this approximation.

Answer: The evaluated derivative at x=1 is -0.1119



If we take a closer look at $x \in [0.8, 1]$, we get:



(d) Compare your results for the approximation of $f'(1)$ from Problems 2b and 2c to that obtained using `gradient`. (For the spacing, use the interpolation points you used in Problem 2b and 2c)

Answer:

```
COMPARISON BETWEEN NUMERICAL DERIVATIVES:============================================================
  The derivative of the interpolating polynomial (using polyfit), f'(x=1) =   -79.01583710409002
  The derivative of the interpolating polynomial (using splines), f'(x=1) =    -0.11190931566917
  The derivative of the function using gradient                  f'(x=1) =    -0.10180995475113
```

For comparison, the value of $f'(x = 1) = -0.073964497$. The results that we get from these numerical derivatives make sense. The derivative of the interpolating polynomial using `polyfit` is the worst because, as we've seen in the past week, interpolating polynomials that use equally spaced points suffer from oscillations which increase the absolute error. The result that we get using `spline` is certainly better, which also makes sense. As we've seen, splines are better for controlling the oscillations when interpolating polynomials. However, we still get an incorrect result, as the interpolation using splines is still not exact. Lastly, the

result that we get using `gradient` is the best (but only slightly better than using `spline`). That is because the algorithm that MATLAB implements in this function is a better numerical approximation. As the documentation for this function states, `gradient` calculates the central difference for interior data points, and calculates values along the edges of the matrix with single-sided differences. This means that `gradient` obtains the numerical derivative dealing directly with the data that we provide it, which is better than first obtaining an approximating polynomial and then taking the derivative of such function. Of course, the result that we get isn't correct, which is caused by the relatively large separation between the eleven points (a separation $h = 0.2$). Therefore, we would expect to get a better result from `gradient` if we had more than just the 11 equally spaced points.