Lecture D. Homework 4

Covered Contents: Newton's Method, Lagrangian Polynomials, (Lec 8 - 10)
Deadline: 10/27/2021, 23:59 PST
Total points: Pen-and-Paper ($10 + 20 + 20 + 20 = 70$) + Coding (30) = 100.
Submit "hw4.zip"

Pen and Paper

D.1. Given $n+1$ data points $\{x_i, f(x_i)\}_{i=0}^n$, where each x_i is assumed to be distinct, recall that the fundamental polynomial interpolation problem is to: find a polynomial $P(x)$ of degree n that satisfies

$$f(x_i) = P(x_i) \qquad (*)$$

for $0 \leq i \leq n$.

We already proved the **existance** of such a polynomial by defining the Lagrange polynomial:

$$P(x) = \sum_{i=0}^{n} f(x_i) L_i(x)$$

where

$$L_i(x) = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}.$$

Now we will show it is **unique** via a proof by contradition.

Assume there are two polynomials $Q(x)$ and $P(x)$ of degree n for which $(*)$ is true. Define a new function, $\phi(x)$ to be the difference between $Q(x)$ and $P(x)$. Necessarily, what degree polynomial is $\phi(x)$? How many roots does $\phi(x)$ have? Given the answer to these two questions, what can we conclude about $P(x)$ and $Q(x)$?

D.2. Let $f(x) = \ln x$.
(a) Construct the Lagrangian polynomial for f passing through the points $(1, \ln 1), (2, \ln 2)$, and $(3, \ln 3)$.
(b) Using the answer in part (a), approximate the values $\ln(1.5)$ and $\ln(2.4)$ and report the error.

D.3. Let $f(x) = \sin(\pi x)$ and $x_0 = 1, x_1 = 1.25, x_2 = 1.6$.
(a) Construct polynomial interpolants of degree at most one and at most two to approximate $f(1.4)$ and find the absolute error. (Hint: pick proper points for the first case.)
(b) Use the theorem expressing the error in Lagrange polynomial interpolation to find an error bound for the approximations.

D.4. This table shows the population in the United States, in thousands, for 1960-2010.

year	1960	1970	1980	1990	2000	2010
population (in thousands)	179,323	203,302	226,542	249,633	281,422	308,746

(a) Construct the interpolating polynomial $P(x)$ for this data (it should be a degree 5 polynomial). It is convenient let x equal the number of years since 1960, so that 1960 corresponds to $x = 0$, 1970 corresponds to $x = 10$ and so on. You may want to use MATLAB to compute this polynomial.
(b) Use $P(x)$ to predict the population of the United States in 2020 (in thousands). How well does it match the data from https://en.wikipedia.org/wiki/2020_United_States_census?

Coding

What to submit: a report, and your code.

Modify the version of Newton's method you used in HW3 to use a stopping condition that insures that the estimated error bound for the root approximation is less than a specified tolerance. Use this modified implementation of Newton's method to find the roots of the following equations with an error bound tolerance of $1 \cdot 10^{-8}$

- (a) $\sin(x/2) - 1 = 0$
- (b) $e^x - \tan(x) = 0$
- (c) $x^3 - 12x^2 + 3x + 1 = 0$

For equations (a) and (b), just find the first root that is greater than zero. For each case, record the value of the starting iterate, the error bound when the iteration terminates,the size of the residual at the final approximation, and the number of iterations that are required to achieve an error bound that is less than the specified tolerance. Output the size of error bound and size of the residuals in scientific notation.

Notes and tips:

- Here you are asked to "use a stopping condition that ensures the estimated error bound for the root approximation is less than some specified tolerance". This means we want

$$|p_n - p| < \tau.$$

So how do we ensure this if we don't already know the true root p? First, recall Taylor's theorem for $n = 1$: $\exists \xi(x)$ s.t.

$$f(x) = f(x_0) + f'(\xi(x))(x - x_0)$$

and $\xi \in (x, x_0)$ or $\xi \in (x_0, x)$. Rearranging it we get

$$\frac{f(x) - f(x_0)}{f'(\xi(x))} = x - x_0.$$

If we let $x = p_n$ and $x_0 = p$, this becomes

$$\frac{f(p_n) - f(p)}{f'(\xi(p_n))} = p_n - p,$$

where $f(p) = 0$ because p is the root. Taking the absolute value of both sides, we get

$$|p_n - p| = |\frac{f(p_n)}{f'(\xi(p_n))}|,$$

with $\xi \in (p_n, p)$ or $\xi \in (p, p_n)$. So, if p_n and p are close, then ξ and p_n are also close. If so, then $f'(\xi) \approx f'(p_n)$. Therefore we get

$$|p_n - p| \approx |\frac{f(p_n)}{f'(p_n)}|.$$

Finally, the definition of Newton's method $p_{n+1} - p_n = -\frac{f(p_n)}{f'(p_n)}$ implies that if $|p_{n+1} - p_n| < \tau$ for some tolerance τ, then $|\frac{f(p_n)}{f'(p_n)}| < \tau$. Therefore approximately $|p_n - p| < \tau$.
Therefore, in practice, we can use

$$|p_{n+1} - p_n| < \tau$$

as the stopping condition. It is a quite good approximation already, in practice.

- When modifying the "newtonRoot.m" program, it is suggested that you first implement the error bound stopping condition on a test problem for which you already know the answer, such as $x^2 - 2 = 0$. After you have verified that your implementation works on a test problem, then use it to find the roots of equations (a)-(c).

- To obtain starting iterates, it is useful to view a plot of the function.

- In this problem youre asked to output the error bounds and residual values in scientific notation. If you are using 'sprintf' then this requires using the 'e' format. For example, here's a snippet of code that can be used to print out the final results:

```
%
% Print out final results
%
disp(sprintf(['Approximate root of %s : %-15.10f '],fstring,xn))
disp(sprintf(['Initial iterate :%-15.10f '],x0))
disp(sprintf(['Residual :%-15.10e '],fn))
disp(sprintf(['Approximation error bound :%-15.10e'],errEst))
disp(sprintf(['Iterations required :%-5d'],iter))
```

and, for the test case of $x^2 - 2 = 0$, this results in the output:

```
Approximate root of x^2 - 2.0 : 1.4142135624
Initial iterate: 1.0000000000
Residual: 4.4408920985e-16
Approximation error bound: 1.5947429103e-12
Iterations required: 5
```