Math 371 Homework 5

1. The polynomial $p(x) = x^4 - x^3 + x^2 - x + 1$ has the values shown below.

x	-2	-1	0	1	2	3
$p(x)$	31	5	1	1	11	61

   Using as little work as possible to find a polynomial q which has values

x	-2	-1	0	1	2	3
$q(x)$	31	5	1	1	11	30

2. Prove that a polynomial interpolantion of degree at most n through the $(n+1)$ points $\{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\}$ must be unique. (Hint: Assume that there are two such polynomials, P and Q, and argue that they must be identical, that is $R(x) = P(x) - Q(x)$ must be zero.)

3. (a) Verify that the polynomials

$$p(x) = 5x^3 - 27x^2 + 45x - 21, \qquad q(x) = x^4 - 5x^3 + 8x^2 - 5x + 3$$

   both interpolate the points in the below table.

x	1	2	3	4
y	2	1	6	47

   You showed that interpolation polynomials are unique. Does this problem contradict that? Why?

   (b) Verify that the polynomials

$$p(x) = 3 + 2(x-1) + 4(x-1)(x+2), \qquad q(x) = 3\frac{(x+2)x}{3} - 3\frac{(x-1)x}{6} - 7\frac{(x-1)(x+2)}{-2}$$

   both interpolate the points in the below table.

x	1	-2	0
y	3	-3	-7

   You showed that interpolation polynomials are unique. Does this problem contradict that? Why?

4. Prove the recursion formula for computing Newton divided differences.

$$f[x_0, x_1, \ldots, x_k] = \frac{f[x_1, x_2, \ldots, x_k] - f[x_0, x_1, \ldots, x_{k-1}]}{x_k - x_0}$$

   To do this, let P be the interpolating polynomial for $\{(x_0, y_0), (x_1, y_1), \ldots, (x_{k-1}, y_{k-1})\}$ and Q the interpolating polynomial for $\{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$ and consider the polynomial

$$R(x) = \frac{x_k - x}{x_k - x_0} P(x) + \frac{x - x_0}{x_k - x_0} Q(x).$$

   (a) Prove R is the unique polynomial of at most degree k which interpolates points $\{(x_0, y_0), (x_1, y_1), \ldots, (x_k, y_k)\}$.

   (b) Determine the coefficient of x^k on each side of the equation.

5. Consider $y = \sin x$ on $[0, 2\pi]$ with five equally spaced interpolation points. Find the interpolation polynomial by hand via the following method
   (a) Vandermonde matrix.
   (b) Newton's form using direct computing of the coefficients.
   (c) Newton's form via Newton divided difference (attach your Newton divided difference matrix.)
   (d) Lagrange interpolation.

6. Write a Matlab function `p = NewtonInt(x,y)` for polynomial interpolation using Newton form and direct computing of the coefficients. The input for this function should be an array x with $(n+1)$ distinct points $x_0, x_1, x_2, \ldots, x_n$ and also an array of function values y, with $(n+1)$ function values $y_0, y_1, y_2, \ldots, y_n$. Your function should return to a vector which contains the coefficients (in descend order) of your interpolation polynomial p such that $p(x_i) = f(x_i)$ for $i = 0, 1, \ldots, n$.

7. Write a Matlab function `p = NewtonDiv(x,y)` for polynomial interpolation using Newton form and Newton divided difference. The input for this function should be an array x with $(n+1)$ distinct points $x_0, x_1, x_2, \ldots, x_n$ and also an array of function values y, with $(n+1)$ function values $y_0, y_1, y_2, \ldots, y_n$. Your function should return to a vector which contains the coefficients (in descend order) of your interpolation polynomial p such that $p(x_i) = f(x_i)$ for $i = 0, 1, \ldots, n$.

8. Write a Matlab function `p = LagrangeInt(x,y)` for polynomial interpolation using Lagrange form. The input for this function should be an array x with $(n+1)$ distinct points $x_0, x_1, x_2, \ldots, x_n$ and also an array of function values y, with $(n+1)$ function values $y_0, y_1, y_2, \ldots, y_n$. Your function should return to a vector which contains the coefficients (in descend order) of your interpolation polynomial p such that $p(x_i) = f(x_i)$ for $i = 0, 1, \ldots, n$.

9. Compare the efficiency of `NewtonInt(x,y)`, `NewtonDiv(x,y)` and `LagrangeInt(x,y)`. Try with large amount of sample points until you see a significant difference.

10. To see the accuracy of an interpolation polynomial $p(x)$, we evaluate the interpolation polynomial on evaluation points. To avoid machine error, instead of direct computing, we apply the Horner's method. Check

    `https://en.wikipedia.org/wiki/Horner%27s_method`

    Let (x_i, y_i), $q \leq i \leq m$ be the points to be evaluated, Define $\vec{x} := (x_1, x_2, ..., x_m)^T$ and $\vec{y} := (y_1, y_2, ..., y_m)^T$. The residual are then defined by the average error

    $$\text{residual} = \frac{\|p(\vec{x}) - \vec{y}\|}{m}, \quad \text{where } m \text{ is the number of evaluation points.}$$

    In real life application, you may use different vector norms depending on the problem. Three popular vector norms are

    - L_1 norm: $\|\vec{x}\|_1 := \sum_{i=1}^{n} |x_i|$
    - L_2 norm: $\|\vec{x}\|_2 := \sum_{i=1}^{n} |x_i|^2$
    - L_∞ norm: $\|\vec{x}\|_\infty := \max_{1 \leq i \leq n+1} |x_i|$

    (a) Read and understand the Horner's method

(b) Write a Matlab script `residual = eval(p,x,y,k)` which evaluates the polynomial p at sample points x using Horner's method with one of the norm above. Here p is a vector containing all the coefficients of the interpolation polynomial, x is a vector containing the evaluation points, y is the true value at sample points and $k = 0, 1, 2$ indicates which norm your are going to use (0 for L_∞, 1 for L_1 and 2 for L^2).

11. For each function below, do

(a) $f(x) = \sin(x),\ 0 \le x \le 2\pi$

(b) $g(x) = \cos(x),\ 0 \le x \le 2\pi$

(c) $h(x) = \ln(x),\ 0.5 \le x \le 2$

(d) $i(x) = e^x,\ 0 \le x \le 2$

(a) Apply your code to find the coefficients of Newton's interpolation polynomial and Lagrange's interpolation polynomial. Use 201 equally spaced sample points of the give interval.

(b) Matlab has a built in polynomial fitting function "polyfit". Use this command to fit your sample points using an appropriate degree n.

(c) Evaluate your result in (a) and (b) by computing the evaluation error `residual = eval(p,x,y,k)` with 1001 equally spaced evaluation points of the given interval. Use the three different norms and list your result for each method.

(d) Based on what you see in (c), discuss which vector norm is better.

(e) Read the help file of "polyfit", discuss when to use "polyfit" and when to use interpolation polynomial. Think about real life applications.

12. Consider the Runge function $f(x) = \dfrac{1}{1+25x^2}$ on $[-1, 1]$. Graph the following all on the same plot. Label your graph and include a legend.

(a) Graph $y = f(x)$ on $[-1, 1]$ using 100 uniformly distributed data points.

(b) Find the degree 10 polynomial interpolant of function f through 11 equally spaced nodes in $[-1, 1]$. Graph it using the same data points as in (a).

(c) Repeat (b) with 11 non-equally spaced Chebyshev points $x_j = \cos\left(\dfrac{\pi j}{10}\right)$, $j = 0, 1, \ldots, 10$. Graph your polynomial using the same data points as in (a).

(d) (Challenge) Research on Chebyshev points and explain why it will solve the Runge phenomenon.