CMSC878R/AMSC698R **Homework 1**

This is similar to the simple example given in class. The goal is to develop fast algorithms for the following matrix-vector product

$$\mathbf{v} = \mathbf{A}\mathbf{u}, \tag{1}$$

where

$$v_i = \sum_{j=1}^{N} u_j \left(x_i - x_j\right)^n, \quad j = 1, ..., N, \tag{2}$$

and

$$\mathbf{A} = \begin{pmatrix} 0 & (x_1 - x_2)^n & (x_1 - x_3)^n & ... & (x_1 - x_N)^n \\ (x_2 - x_1)^n & 0 & (x_2 - x_3)^n & ... & (x_2 - x_N)^n \\ (x_3 - x_1)^n & (x_3 - x_2)^n & 0 & ... & (x_3 - x_N)^n \\ ... & ... & ... & ... & ... \\ (x_N - x_1)^n & (x_N - x_2)^n & (x_N - x_3)^n & ... & 0 \end{pmatrix}, \quad \mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ ... \\ u_N \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ ... \\ v_N \end{pmatrix}. \tag{3}$$

Here $x_1, ..., x_N$, $u_1, ..., u_N$, are given. The matrix dimension $N > 0$, and the power, $n > 0$, are given (fixed) positive integers.

1. Derive a formula that can be used for developing a "fast" $(O(N))$ method for arbitrary n.
2. Write a program that implements both straightforward computation based on Eq. (2) and the "Fast" method. Use Matlab. Be sure that you can vary $x_1, ..., x_N$, and $u_1, ..., u_N$.
3. Check that both methods produce the same results (within machine precision). Plot the absolute maximum error between the straightforward and the "Fast" method for $n = 5$ and N varying between 10^2 and 10^3.
4. Plot a comparison of the CPU time required by the straightforward and the "Fast" methods for $n = 5$ and N varying between 10^2 and 10^3 for the straightforward method, and with N varying between 10^2 and 10^4 for the "Fast" method. Make the plots log-log. Also plot lines corresponding to linear and quadratic dependences of the CPU time on N and compare with your computational results.
5. Make a conclusion about the asymptotic complexity of each method.

Hints

1. Use the Newton binomial theorem

$$(a + b)^n = a^n + \begin{pmatrix} n \\ 1 \end{pmatrix} a^{n-1}b + \begin{pmatrix} n \\ 2 \end{pmatrix} a^{n-2}b^2 + ... + b^n = \sum_{k=0}^{n} \begin{pmatrix} n \\ k \end{pmatrix} a^{n-k}b^k, \tag{4}$$

   where the binomial coefficients are

$$\begin{pmatrix} n \\ k \end{pmatrix} = \frac{n(n-1)...(n-k+1)}{1 \cdot 2 \cdot ... \cdot k} = \frac{n!}{(n-k)!k!}. \tag{5}$$

2. For CPU time measurement use the Matlab function `cputime` (see Matlab help).