Math 449: Numerical Methods
Homework 6
Wenzhen Zhu

Part 1: Theory

Problem 1. Given $x_0, x_1, \dots, x_n \in \mathbb{R}$, define the *Vandermonde matrix*

$$V = \begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix}.$$

Prove that V is invertible if and only if $x_0, \dots, x_n$ are distinct. Hint: Let $p_n(x) = a_0 + a_1 x + \cdots + a_n x^n$, and consider the meaning of the linear system

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}.$$

If $x_0, \dots, x_n$ are distinct, $\vec{a}$ is always nontrivial solution

so $p_n(x)$ always have a list of coefficients where $\vec{a}$ is unique.

so V is invertible.

if $x_0, \dots, x_n$ are not distinct,

$\exists$ some i, j. s.t. $x_i = x_j$,

so V is singular

Problem 2. Recall that the interpolating polynomial, in Lagrange form, is

$$p_n(x) = \sum_{k=0}^{n} L_k(x) y_k, \quad \text{where} \quad L_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{x - x_i}{x_k - x_i}. \tag{1}$$

This formula, however, does not give the most efficient or numerically stable way to evaluate p_n at a point x. In this problem, we will explore an alternate approach, called *barycentric Lagrange interpolation*[1], which is generally preferred for numerical implementation.

a. Explain why evaluating (1), as written, requires $\mathcal{O}(n^2)$ operations.

b. For $k = 0, \ldots, n$, define the *barycentric weights*

$$w_k = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{1}{x_k - x_i}.$$

Show that the interpolating polynomial can be rewritten as

$$p_n(x) = \pi_{n+1}(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k} y_k, \qquad x \neq x_0, \ldots, x_n, \tag{2}$$

where (following Süli–Mayers notation) $\pi_{n+1}(x) = \prod_{i=0}^{n} (x - x_i)$.

(a)

$$L_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{x - x_i}{x_k - x_i}$$

for each $L_k(x)$, we need $(2n)$ subtraction

n division

n multiplication

$$p_n = \sum_{k=0}^{n} L_k(x) \cdot y_k$$

for each $p_n(x)$, it need n addition, $n+1$ multiplication

In total: $(2n + n + n)(n + n+1) = 4n \cdot (2n+1) = 8n^2 + 4n = O(n^2)$

(b)

$$p_n(x) = \sum_{k=0}^{n} \left(\prod_{i=0}^{n} \frac{x - x_i}{x_k - x_i} \right) y_k = \sum_{k=0}^{n} \prod_{i=0}^{n} \frac{(x - x_i)}{(x - x_k)} \prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{y_k}{x_k - x_i}$$

$$= \underbrace{\prod_{i=0}^{n} (x - x_i)}_{\pi_{n+1}(x)} \sum_{k=0}^{n} \Big(\prod_{\substack{i=0 \\ i \neq k}}^{n} \frac{1}{x - x_k} \overbrace{\frac{1}{x_k - x_i}}^{w_k} \cdot y_k \Big) = \pi_{n+1}(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k} y_k$$

c. Show that for every $x \neq x_0, \ldots, x_n$,

$$1 = \pi_{n+1}(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k}$$

(hint: interpolate the constant function 1), and combine this with (2) to deduce the *barycentric interpolation formula*

$$p_n(x) = \frac{\displaystyle\sum_{k=0}^{n} \frac{w_k}{x - x_k} y_k}{\displaystyle\sum_{k=0}^{n} \frac{w_k}{x - x_k}}, \qquad x \neq x_0, \ldots, x_n. \tag{3}$$

d. Show that computing the barycentric weights requires $\mathcal{O}(n^2)$ operations, but that subsequently evaluating (2) or (3) requires only $\mathcal{O}(n)$ operations. (Note that the barycentric weights only need to be computed once, since they are independent of x.)

(c.)

$$P_1(x) = \pi_{n+1}(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k} = 1 \qquad \text{since } x^0 = 1 \ \forall x \ \text{ with } y_k = 1$$

$$\Rightarrow \quad \pi_{n+1}(x) = \frac{1}{\displaystyle\sum_{k=0}^{n} \frac{w_k}{x - x_k}}$$

$$p_n(x) = \pi_{n+1}(x) \sum_{k=0}^{n} \frac{w_k}{x - x_k} y_k = \frac{\displaystyle\sum_{k=0}^{n} \frac{w_k}{x - x_k} y_k}{\displaystyle\sum_{k=0}^{n} \frac{w_k}{x - x_k}}$$

Part 2

Problem 3

In [6]:
```
from __future__ import division
from pylab import *

def weights(xk):
    n = len(xk)
    [x1, x2] = meshgrid(xk, xk)
    z = subtract(x2, x1) + eye(n)
    w = zeros(n)
    for i in range(n):
        w[i] = 1/prod(z[i])
    return w
```

In [7]:
```
weights(linspace(-5,5,11))
```

Out[7]:
```
array([  2.75573192e-07,  -2.75573192e-06,   1.24007937e-05,
        -3.30687831e-05,   5.78703704e-05,  -6.94444444e-05,
         5.78703704e-05,  -3.30687831e-05,   1.24007937e-05,
        -2.75573192e-06,   2.75573192e-07])
```

Problem 4 ¶

In [8]:
```
def f(x):
    return 1/(1+x**2)

def interpolate(x, xk, yk, wk):
    i = find(x == xk)
    if size(i) > 0:
        p = yk[i[0]]
    else:
        p = sum(wk*yk / (x - xk))/sum(wk/(x-xk))
    return p
```

In [9]:
```
xk = linspace(-5,5,11)
yk = f(xk)
wk = weights(xk)
interpolate(4.75, xk,yk,wk)
```

Out[9]:
```
1.9236311497192033
```

In [10]:
```
xk = linspace(-5,5,21)
yk = f(xk)
wk = weights(xk)
interpolate(4.75, xk,yk,wk)
```

Out[10]: -39.952449033039777

In [11]:
```
xk = linspace(-5,5,51)
yk = f(xk)
wk = weights(xk)
interpolate(4.75, xk,yk,wk)
```

Out[11]: -122974.68471651913