CMSC 666
Spring 2007

Homework I

Roman Stanchak

Problem 1: Prove the Vandermonde matrix is nonsingular by showing that its determininant is nonzero.

Solution 1: In order to prove the Vandermonde matrix is nonzero, it will be shown that the determinant takes a particular form. If A_n is the Vandermonde matrix for $n+1$ interpolation points,

$$|A_n| = (-1)^n \prod_{i=0}^{n} \prod_{j=i+1}^{n} (x_i - x_j)$$

By this proposition, $|A_n| = 0$ if and only if $x_i = x_j$ for some i, j. However, the Vandermonde matrix by definiton is contructed of unique x_i's, so this can not be the case, and therefore $|A_n| \neq 0$.

Proof of the proposition by induction on the number of interpolation nodes.

Base Case. Prove for $n = 1$, x_0, x_1.

$$\begin{aligned} |A_1| &= \begin{vmatrix} 1 & x_0 \\ 1 & x_1 \end{vmatrix} \\ &= (x_1 - x_0) \end{aligned}$$

□

Inductive Case. Assume for $n-1$ (i.e. n interpolation nodes), prove for n (i.e. $n+1$ interpolation nodes).

$$\begin{aligned} |A_n| &= \begin{vmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ \ldots & & & & \\ \ldots & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{vmatrix} \\ &= x_0^n |A_{n/0}| - x_1^n |A_{n/1}| + \cdots + x_j^n |A_n n/j| + \cdots + x_n^n |A_{n-1}| \end{aligned}$$

Where $A_{n/j}$ is the determinant of the cofactor matrix excluding the jth row and nth column. Since the order of the points $x_0 \ldots x_n$ is arbitrary, and each of the $A_{n/j}$ cofactor matrices is size $n \times n$, the IH holds:

$$|A_n| = \sum_{i=0}^{n} x_i^n (-1)^n \prod_{j=0, j\neq i}^{n} \prod_{k=j+1}^{n} (x_j - x_k)$$

Don't know how to get from A to B

$$|A_n| = (-1)^n \prod_{i=0}^{n} \prod_{j=i+1}^{n} (x_i - x_j)$$

□

Problem 2.: Propose a method for finding the root t^* of a function $g(t)$, where it is known that in the neighborhood of $t*$, g has an inverse that is a degree n polynomial, that is:

$$p_n(g(t)) = t$$

for all t near t^*.

Solution 2.: The method should go something as follows:
Let $[a, b]$ be the interval consisting of the 'neighborhood' of t^*. Choose $n+1$ distinct points $t_0, t_1, \ldots, t_n$ from the interval and compute $g(t_j)$ for each. Let $y_j = g(t_j)$. Since $g(t)$ has an inverse in the interval $[a, b]$, then the y_j's are distinct. $p_n(y)$ is then the interpolating polynomial for the interpolation nodes $y_0, y_1, \ldots, y_n$ and interpolation data $t_0, t_1, \ldots, t_n$. The root t^* is found simply by evaluating $p_n(0)$.

Problem 3.: Prove the formula:

$$f[x_0, x_1, \cdots, x_n] = \sum_{i=0}^{n} f(x_i) \prod_{j=0, j\neq i}^{n} (x_i - x_j)^{-1}$$

Solution 3.:

Proof. Proof by induction on number of interpolation points.

Base Case. Prove for $n = 1$, x_0, x_1.
Using the divided difference formula:

$$\begin{aligned} f[x_0, x_1] &= \frac{f[x_1] - f[x_0]}{(x_1 - x_0)} \\ &= \frac{f(x_1) - f(x_0)}{(x_1 - x_0)} \\ &= \frac{f(x_0)}{x_0 - x_1} + \frac{f(x_1)}{(x_1 - x_0)} \\ &= \sum_{i=0}^{1} f(x_i) \prod_{j=0, j\neq i}^{1} (x_i - x_j)^{-1} \end{aligned}$$

□

Inductive Case. Assume that for $\forall k < n$ points, the formula holds. Prove for n points.
Again using the divided difference formula:

$$f[x_0, x_1, \cdots, x_n] = \frac{f[x_1, x_2, \cdots, x_n] - f[x_0, x_1, \cdots, x_{n-1}]}{(x_n - x_0)}$$

Since the ordering of the points and summation indices is arbitrary, and the terms $f[x_1 \cdots x_n]$ and

$f[x_0 \cdots x_{n-1}]$ have fewer than n points, the IH applies, and the following is true:

$$
\begin{aligned}
&= \frac{\sum_{i=1}^{n} f(x_i) \prod_{j=1,j\neq i}^{n} (x_i - x_j)^{-1} - \sum_{i=0}^{n-1} f(x_i) \prod_{j=0,j\neq i}^{n-1} (x_i - x_j)^{-1}}{(x_n - x_0)} \\
&= \sum_{i=1}^{n} \frac{f(x_i)}{(x_n - x_0) \prod_{j=1,j\neq i}^{n} (x_i - x_j)} - \sum_{i=0}^{n-1} \frac{f(x_i)}{(x_n - x_0) \prod_{j=0,j\neq i}^{n-1} (x_i - x_j)} \\
&= \frac{f(x_0)}{(x_n - x_0) \prod_{j=1,j\neq n}^{n} (x_n - x_j)} - \frac{f(x_n)}{(x_n - x_0) \prod_{j=0,j\neq 0}^{n-1} (x_0 - x_j)} + \Phi \\
&= \frac{f(x_0)}{\prod_{j=0,j\neq n}^{n} (x_n - x_j)} + \frac{f(x_n)}{\prod_{j=0,j\neq 0}^{n} (x_0 - x_j)} + \Phi
\end{aligned}
$$

Where:

$$
\begin{aligned}
\Phi &= \sum_{i=1}^{n-1} \frac{f(x_i)}{(x_n - x_0) \prod_{j=1,j\neq i}^{n} (x_i - x_j)} - \sum_{i=1}^{n-1} \frac{f(x_i)}{(x_n - x_0) \prod_{j=0,j\neq i}^{n-1} (x_i - x_j)} \\
&= \sum_{i=1}^{n-1} \frac{f(x_i)}{(x_n - x_0) \prod_{j=1,j\neq i}^{n-1} (x_i - x_j)} \left(\frac{1}{(x_i - x_n)} - \frac{1}{(x_i - x_0)} \right) \\
&= \sum_{i=1}^{n-1} \frac{f(x_i)}{(x_n - x_0) \prod_{j=1,j\neq i}^{n-1} (x_i - x_j)} \cdot \frac{(x_i - x_0) - (x_i - x_n)}{(x_i - x_n)(x_i - x_0)} \\
&= \sum_{i=1}^{n-1} \frac{f(x_i)}{(x_n - x_0) \prod_{j=1,j\neq i}^{n-1} (x_i - x_j)} \cdot \frac{(x_n - x_0)}{(x_i - x_n)(x_i - x_0)} \\
&= \sum_{i=1}^{n-1} \frac{f(x_i)}{\prod_{j=0,j\neq i}^{n} (x_i - x_j)}
\end{aligned}
$$

Substituting Φ back in above:

$$
\begin{aligned}
&= \frac{f(x_0)}{\prod_{j=0,j\neq n}^{n} (x_n - x_j)} + \frac{f(x_n)}{\prod_{j=0,j\neq 0}^{n} (x_0 - x_j)} + \sum_{i=1}^{n-1} \frac{f(x_i)}{\prod_{j=0,j\neq i}^{n} (x_i - x_j)} \\
f[x_0, x_1, \cdots, x_n] &= \sum_{i=0}^{n} f(x_i) \prod_{j=0,j\neq i}^{n} (x_i - x_j)^{-1}
\end{aligned}
$$

□

Thus, by the theory of induction, the proposition holds for $\forall n$. □

Problem 4: Given distinct points $x_0 \ldots x_n$, the value of polynomial $p(x)$ at each, $f_j = p(x_j)$, and the value of the deriviative at each, $f'_j = p'(x_j)$, represent the interpolating polynomial as:

$$p(x) = \sum_{j=0}^{n} f_j \alpha_j(x) + \sum_{j=0}^{n} f'_j \beta_j(x)$$

Solution a: The requirements on α and β are as follows:

$$\begin{aligned}
\alpha_j(x) &= \begin{cases} 1 & x = x_j \\ 0 & \text{otherwise} \end{cases} \\
\alpha'_j(x) &= 0, \forall x_j \\
\beta_j(x) &= 0, \forall x_j \\
\beta'_j(x) &= \begin{cases} 1 & x = x_j \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

Solution b: Given the $n+1$ points and the values of $p(x)$ and $p'(x)$ for each, amounts to $2(n+1)$ constraints on the coefficients of the polynomial. This corresponds to a $2n+1$ degree polynomial. Writing out the constraints for each point in matrix form:

$$\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 0 & 1 & 2 \cdot x_0 & \dots & n \cdot x_0^{n-1} \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ 0 & 1 & 2 \cdot x_1 & \dots & n \cdot x_1^{n-1} \\ \dots \\ 1 & x_n & x_n^2 & \dots & x_n^n \\ 0 & 1 & 2 \cdot x_n & \dots & n \cdot x_n^{n-1} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \dots \\ \alpha_{2n+2} \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f'(x_0) \\ f(x_1) \\ f'(x_1) \\ \dots \\ f(x_n) \\ f'(x_n) \end{bmatrix}$$

Based on my experiments with a CAS with n=1, it is possible that...

$$\begin{aligned}
\alpha_i(x) &= l_i^2(x) l'_i(x) \\
\beta_i(x) &= ?
\end{aligned}$$

Problem 5: Given distinct $x_0, x_1, \dots, x_n$, and $y_0, y_1, \dots, y_n$, we seek the function:

$$p_n(x) = \sum_{j=0}^{n} c_j e^{jx}$$

such that $p_n(x_j) = y_j$. Show there is a unique choice of c_j for which this is true.

Solution 5:

Proof. Each x_j, y_j provides a constraint on the linear system of equations consisting of the c_j's:

$$y_j = \sum_{i=0}^{n} c_i e^{ix_j}$$

Let $z_j = e^{x_j}$. As a result,

$$y_j = \sum_{i=0}^{n} c_i z_j^i$$

This is just the standard polynomial interpolation problem for which there exists a unique set of c_j's if z_j's are unique. x_j's as given are unique. Thus, $z_j = e_j^x$ is also unique because e^x is a strictly increasing function. As a result, there exists a unique set of c_j's as proposed. □

Problem 6a.:

Solution 6a.: See Figure 1 for source code.

Solution 6b.: See Figure 2 for source code and Figures 3 and 4 for function graphs.

```
function A = newton_poly(X,Y)
% A = NEWTON_POLY(X,Y)
%      Compute coefficients of the interpolating polynomial in Newton Form.
% X    set of input values in interval [a,b]
% Y    set of function values f(x_0, x_1, ... x_n)
% A    Calculated Newton polynomial coefficients

% Newton coefficients a_i are calculated recursively as follows:
% a_0 = f[x_i] = y_i
% a_i = f[x0 ... x_i]
% f[x0 ... x_i ] = ( f[x_1 ... x_i] - f[x_0 .. x_i-1] ) / (x_i - x_0)
N = length(X);
A = zeros(N,N);
A(:,1) = Y';
for i=2:N,
        dA = ( A(2:N-i+2,i-1) - A(1:N-i+1,i-1) );
        dx = ( X(i:N) - X(1:N-i+1) )';
        A(1:N-i+1,i) = dA ./ dx;
end%for
A = A(1,:);
```

Figure 1: Listing from source file `newtonpoly.m`

```
function Q = eval_newton(X,A,z)
% input X    set of interpolation nodes
% input A    Newton polynomial coefficients
% input z    point to evaluate polonomial at

% Q_n(x) === f(x)      Interpolation polynomial of degree n
% Q_n(x) = Q_{n-1}(x) + q_k(x)
% Q_0(x) === f(x_0)
% q_k(x) = a_k prod_j=[0,k-1] (x - x_j)
% a_0 = f[x_i]
% a_i = f[x0 ... x_i]
% f[x0 ... x_i ] = ( f[x_1 ... x_i] - f[x_0 .. x_i-1] ) / (x_i - x_0)
N = length(X)-1;
Q = A(1);
x=1
for i=1:N,
        x=x*(z-X(i))
        Q = Q + A(i+1)*x;
end%for
```

Figure 2: Listing from source file `newtoneval.m`

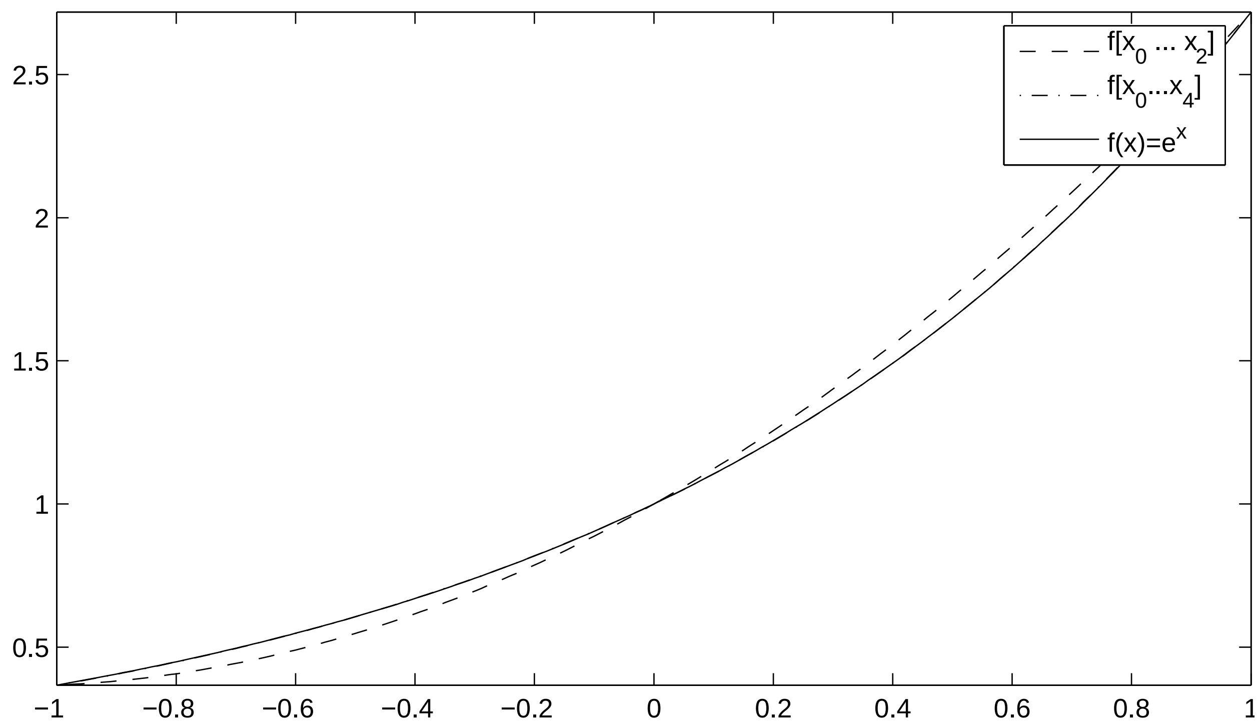

Figure 3: Plot of e^x and $p(x)$ interpolated at 3 and 5 points.

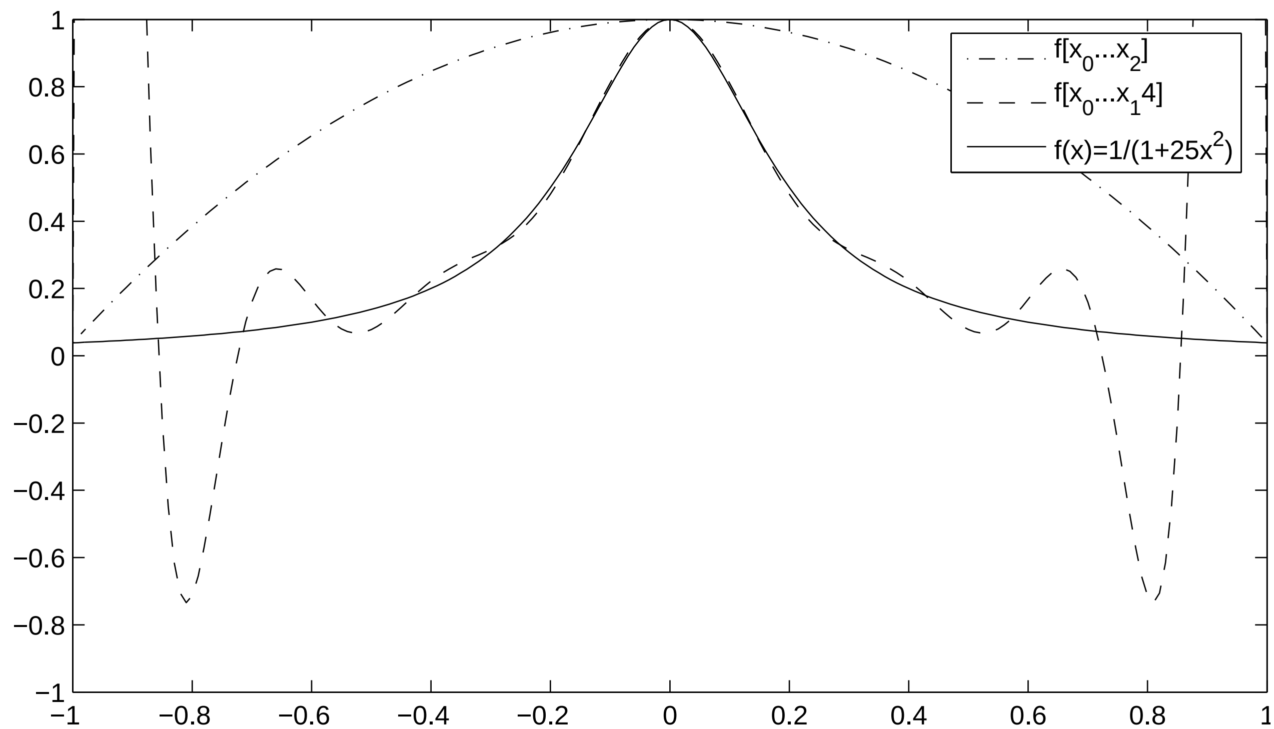

Figure 4: Plot of $\frac{1}{1+25x^2}$ and $p(x)$ interpolated at 3 and 15 points.