Homework No.3, 553.481/681, Due March 8, 2024.

Problem 1. Define the *Vandermonde matrix* by its elements:

$$V_{ij}[x_0, ..., x_n] = x_i^j, \quad 0 \le i, j \le n,$$

for any set of $(n+1)$ real numbers $x_0, ..., x_n$.

(a) Show that

$$\det \mathbf{V}[x_0, ..., x_n] = \prod_{i=0}^{n-1} (x_n - x_i) \cdot \det \mathbf{V}[x_0, ..., x_{n-1}].$$

Hint: Show that the determinant on the left is a polynomial of degree n in x_n and find its roots and the coefficient of its highest-order term.

(b) Use part (a) and induction to show that

$$\det \mathbf{V}[x_0, ..., x_n] = \prod_{0 \le i < j \le n} (x_j - x_i)$$

Problem 2. Consider the following seven points:

$$(x_1, y_1) = (1, -17),\ (x_2, y_2) = (2, -29),\ (x_3, y_3) = (3, -55),\ (x_4, y_4) = (4, -155),$$

$$(x_5, y_5) = (5, -581),\ (x_6, y_6) = (6, -1537),\ (x_7, y_7) = (7, 661)$$

For this data, find the 6th-degree interpolating polynomial (a) in the monomial basis, (b) in the barycentric Lagrange form, and (c) in the Newtonian form with divided-differences. Compare the wall clock times to compute the coefficients in the monomial basis, the barycentric weights, and the divided-differences. Which form of the interpolating polynomial is computed the fastest in this example? Plot the seven points along with the interpolating polynomial over the interval $[0, 8]$.

Problem 3. Use the expression

$$f[x_0, ..., x_n] = \sum_{i=0}^{n} w_i f(x_i), \quad w_i = 1/\Psi_n'(x_i)$$

for the divided-difference, with $\Psi_n(x) = \prod_{j=0}^{n}(x - x_j)$, to verify that

$$f[x_0, ..., x_n] = \frac{f[x_1, ..., x_n] - f[x_0, ..., x_{n-1}]}{x_n - x_0}.$$

(b) Show that the polynomial $p_n(x)$ interpolating $f(x)$ can be written as

$$p_n(x) = \frac{\sum_{j=0}^{n} \frac{w_j f(x_j)}{x-x_j}}{\sum_{j=0}^{n} \frac{w_j}{x-x_j}}$$

provided x is not a node point.

Problem 4. This problem explores the utility of *inverse polynomial interpolation* in which data (x_i, y_i), $i = 1, , , , n$ are interpolated by a polynomial $x = Q(y)$ rather than by a polynomial $y = P(x)$. Both interpolation schemes can be applied to the same data if all x_i-values are distinct for $i = 1, ..., n$ and if also all y_i-values are distinct for $i = 1, ..., n$. If both of these conditions are satisfied, then the same algorithms can be applied to evaluate both polynomial interpolants.

(a) Here we consider the following seven points:

$$(x_1, y_1) = (1, 17),\ (x_2, y_2) = (1.2, 17.8929291),\ (x_3, y_3) = (1.4, 19.2467442),$$
$$(x_4, y_4) = (1.6, 21.1810760),\ (x_5, y_5) = (1.8, 23.8244495),$$
$$(x_6, y_6) = (2, 27.3137093),\ (x_7, y_7) = (3, 29.0457592)$$

Could this data have been consistently generated by $(x_i, f(x_i))$ and also by $(g(y_i), y_i)$ for a function f and its inverse $g = f^{-1}$?

(b) Give both 6th-degree polynomials for the data in (a), the direct interpolant $P_6(x)$ and the inverse interpolant $Q_6(y)$, both in the Newton form.

(c) Plot the data and both polynomials over the range $0 < x < 4$ and $10 < y < 60$. Is it true that $Q_6(y) = P_6^{-1}(y)$?

Problem 5. Another example of the Runge phenomenon is provided by the function

$$f(x) = \frac{1}{(1 + 8000|x|^5)^{1/5}}.$$

(a) Define the n-point Chebyshev grid in the interval $[-1, 1]$ by

$$x_k = \cos\left(\frac{(2k-1)\pi}{2n}\right), \quad k = 1, ..., n.$$

Interpolate the above function with a polynomial on the Chebyshev grid for $n = 30, 40, 50, 60, 70, 80$ and plot the results. What would you conjecture about the limit $n \to \infty$ of the interpolating polynomial on the basis of these plots?

(b) Compare the results in (a) with those obtained by polynomial interpolation on the uniform grid

$$x_k = \frac{2k - (n+1)}{2(n+1)}, \quad k = 1, ..., n$$

for $n = 30, 40, 50, 60, 70, 80$.

Problem 6*. In the root-finding method of inverse quadratic interpolation a quadratic polynomial $x = p_2(y)$ is fit to the three points (f_a, a), (f_b, b), (f_c, c).

(a) Use the Lagrange form of the interpolating quadratic $p_2(y)$ to show that it crosses the x-axis at the unique point

$$x = a\frac{f_b f_c}{(f_a - f_b)(f_a - f_c)} + b\frac{f_a f_c}{(f_b - f_a)(f_b - f_c)} + c\frac{f_a f_b}{(f_c - f_a)(f_c - f_b)}.$$

(b) Use the result in (a) to derive the expression used in `fzerotx.m`, i.e. show that if

$$r = f_b/f_a, \quad s = f_b/f_c, \quad t = f_c/f_a$$

$$p = s[(a-b)t(t-r) - (b-c)(r-1)], \quad q = (r-1)(s-1)(t-1),$$

then $\hat{x} = b - p/q$. [*Hint:* In the latter expression for $\hat{x}$ gather together all of the terms proportional to a, b, and c.]

(c) Generalize the formula in (a) to give an approximate root x by inverse cubic interpolation, using a cubic polynomial $x = p_3(y)$ to fit four points (f_a, a), (f_b, b), (f_c, c), (f_d, d).

Problem 7*. (a) By modifying the code `newtondif.m`, write a script `hermitedif.m` to find the coefficients of the Hermite interpolating polynomial for general data $\mathbf{x} = (x_1, ..., x_n)$, $\mathbf{y} = (y_1, ..., y_n)$ and $\mathbf{y}' = (y'_1, ..., y'_n)$. *Hint*: Essentially only the first loop over $i = 1$ needs to be modified, if the vector is doubled to $\mathbf{X} = (x_1, x_1, x_2, x_2, ..., x_n, x_n)$.

(b) Similarly, write a script `hermiteint.m` to evaluate the Hermite interpolating polynomial, by modifying `newtonint.m`.

(c) Apply the code in (b) to Hermite interpolate the function $f(x) = e^{-x}/\sqrt{x}$ on the points $0.5, 1.0, 1.5, 2.0, 2.5$, and 3.0. Plot the interpolating polynomial on the interval $[0, 4]$ along with the original function $f(x)$. Comment on the relative accuracy of extrapolation versus interpolation.

(d) Compare the results in (c) with the Lagrange interpolating polynomial for the same points and function. Which is more accurate and over what ranges of x?