HW7

Thursday, October 19, 2023 7:42 PM

1. In this problem we find the the polynomial $p(x) = c_n + c_{n-1}x + c_{n-3}x^2 + \cdots + c_1 x^{n-1}$ that interpolates the data $(x_j, y_j) = (x_j, f(x_j)), j = 1, \ldots, n$.

   (a) Assume $(x_j, y_j), j = 1, \ldots, n$ are given. Derive the system $V\mathbf{c} = \mathbf{y}$ that determines the coefficients $\mathbf{c} = [c_1, \ldots, c_n]^T$ (here $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$), that is, find how the matrix V looks like.

   To solve the system of equations you can simply use inversion from Numpy's linear algebra package. You will write your polynomial evaluator.

   (b) Find the polynomial (i.e. the coefficients $\mathbf{c}$) that interpolates

   $$f(x) = \frac{1}{1 + (10x)^2},$$

   in the points $x_i = -1 + (i-1)h, h = \frac{2}{N-1}, i = 1, \ldots, N$. Plot data points as circles (`plot(x,f,'o')`) and, in the same plot, plot the polynomial and $f(x)$ on a finer grid (still on $x \in [-1, 1]$), say with 1001 points. Observe what happens when you increase N. Try $N = 2, 3, 4, \ldots$ and continue until the maximum value of $p(x)$ is about 100 (should be for $N \sim 17 - 20$). As you can see the polynomial behaves badly near the endpoints of the interval due to Runge's phenomena.

a)

$$p(x) = c_n + c_{n-1}x + c_{n-2}x^2 + \cdots + c_1 x^{n-1}$$

$$\{(x_i, f(x_i)\}_{i=1}^{n}$$

$$p(x_1) = f(x_1) = c_n + c_{n-1}x_1 + \cdots + c_1 x_1^{n-1}$$

$$p(x_2) = f(x_2) = c_n + c_{n-1}x_2 + \cdots + c_1 x_2^{n-1}$$

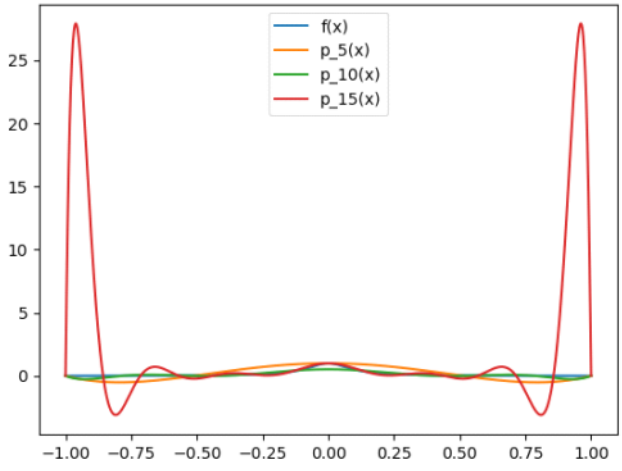
$$\vdots$$

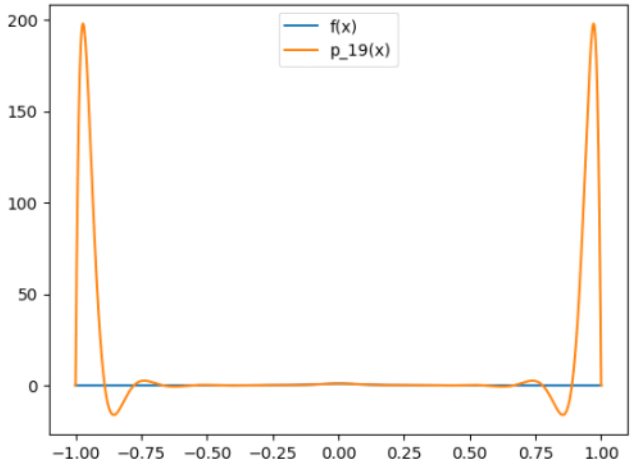
$$p(x_n) = f(x_n) = c_n + c_{n-1}x_n + \cdots + c_1 x_n^{n-1}$$

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ 1 & x_3 & x_3^2 & \cdots & x_3^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} c_n \\ c_{n-1} \\ c_{n-2} \\ \vdots \\ c_1 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} \quad \text{where } y_i = f(x_i)$$

$$V \qquad \vec{c} = \vec{y}$$

b) See hwT.py





2. Solving the interpolation problem using the monomial basis (as above) is notoriously ill-conditioned, in fact it is possible to show $\texttt{cond(V)} \sim \pi^{-1} e^{\pi/4} (3.1)^n$. A better way of interpolating is to use either of the barycentric Lagrange interpolation formulas:

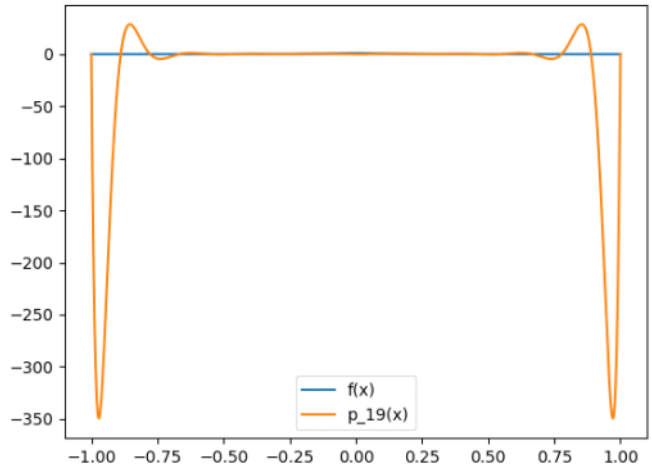
$$p(x) = \Phi_n(x) \sum_{j=0}^{n} \frac{w_j}{x - x_j} f(x_j).$$

$$p(x) = \frac{\sum_{j=0}^{n} \frac{w_j}{x-x_j} f(x_j)}{\sum_{j=0}^{n} \frac{w_j}{x-x_j}}, \quad x \neq x_j.$$

Where

$$\Phi_n(x) = \prod_{i=0}^{n} (x - x_i), \quad w_j = \frac{1}{\prod_{i=0, i\neq j}^{n} (x_j - x_i)}.$$

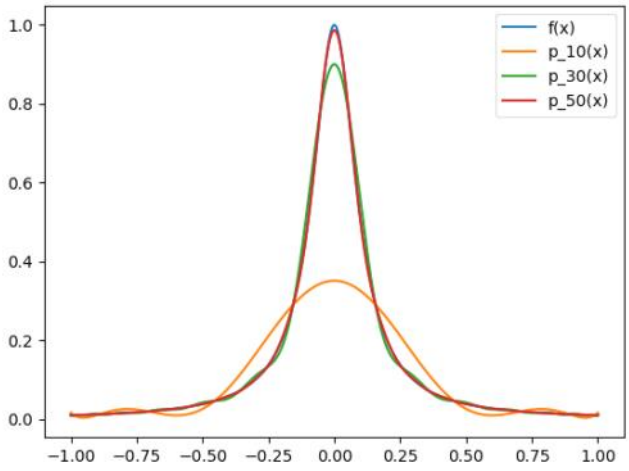
Using either of the above formulas try again to interpolate $f(x)$. Show with some pictures that you still get the same bad behavior close to the endpoints (this is a property of the function $f(x)$ and the distribution of the grid points not of the form of interpolation) but that the approximation is well behaved for small x for very large n.



3. It is much better to interpolate on a grid made up of points that are clustered towards the endpoints. Try to interpolate $f(x)$ in the Chebyshev points

$$x_j = \cos \frac{(2j-1)\pi}{2N}, \quad i = 1, \ldots, N,$$

using either of the methods above. Can you get the interpolation to fail now?


f(x)
p_10(x)
p_30(x)
p_50(x)
1.0
0.8
0.6
0.4
0.2
0.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00