Chapter 2
Fast Fourier Transform

Songting Luo

Department of Mathematics
Iowa State University

MATH 561 Numerical Analysis

Discrete Least Squares Approximation

Consider the approximation of the data $\{(x_i, y_i)\}$ for $i = 1, \dots, m$ by a polynomial (or trigonometric polynomial)

$$P_n(x) = a_n\phi_n(x) + a_{n-1}\phi_{n-1}(x) + \cdots + a_1\phi_1(x) + a_0\phi_0(x)$$

with $\phi_0, \dots, \phi_n$ a basis by minimizing

$$E \equiv E(a_0, a_1, \dots, a_n) = \sum_{i=1}^{m} (P_n(x_i) - y_i)^2 w(x_i),$$

over all $\{a_0, a_1, \dots, a_n\}$. The coefficients $\mathbf{a} = (a_0, \dots, a_n)$ are given by the solution to the Normal Equations by setting

$$\frac{\partial E}{\partial a_j} = 0$$

for $j = 0, 1, \dots, n$.

Least Squares Approximation of Functions

Consider approximating $f \in C[a,b]$ by a polynomial (or trigonometric polynomial)

$$P_n(x) = a_n\phi_n(x) + a_{n-1}\phi_{n-1}(x) + \cdots + a_1\phi_1(x) + a_0\phi_0(x)$$

with $\phi_0, \ldots, \phi_n$ a basis by minimizing

$$E \equiv E(a_0, a_1, \ldots, a_n) = \int_a^b (f(x) - \sum_{k=0}^{n} a_k\phi_k(x))^2 w(x)dx$$

The coefficients **a** $= (a_0, \ldots, a_n)$ are given by the solution to the Normal Equations by setting

$$\frac{\partial E}{\partial a_j} = 0$$

for $j = 0, 1, \ldots, n$.

Orthogonal Functions

- Legendre polynomial: $[-1,\ 1]$, weight $w(x) \equiv 1$.
  - Other application: Gaussian Quadrature
- Chebyshev polynomial: $[-1,\ 1]$, weight $w(x) = \frac{1}{\sqrt{1-x^2}}$
  - Minimization Property
  - Chebyshev Nodes
- Orthogonal Trigonometric polynomial: $[-\pi,\ \pi]$, weight $w(x) \equiv 1$. For each integer $n > 0$, the set $\{\phi_0, \phi_1, \ldots, \phi_{2n-1}\}$, where

$$\begin{aligned} \phi_0(x) &= 1/2, \\ \phi_k(x) &= \cos kx, \qquad \text{for each } k = 1, 2, \ldots, n \\ \phi_{n+k}(x) &= \sin kx, \qquad \text{for each } k = 1, 2, \ldots, n-1. \end{aligned}$$

  is orthogonal on $[-\pi, \pi]$ with respect to $w(x) = 1$.

Trigonometric Polynomial Approximation; Least Squares

- Trigonometric polynomials of degree $\leqslant n$:
  $\mathbf{T}_n = \text{span}(\{\phi_0, \phi_1, \ldots, \phi_{2n-1}\})$, i.e., linear combinations.
- Least squares approximation: approximating $f \in C[-\pi, \pi]$ by functions $S_n(x) \in \mathbf{T}_n$:

$$S_n(x) = \frac{a_0}{2} + a_n \cos nx + \sum_{k=1}^{n-1} (a_k \cos kx + b_k \sin kx)$$

by minimizing

$$E \equiv E(a_0, a_1, \ldots, a_n, b_1, \ldots, b_{n-1}) = \int_{\pi}^{\pi} (f(x) - S_n(x))^2 dx$$

Trigonometric Polynomial Approximation; Least Squares

Least square approximation of $f \in C[-\pi, \pi]$ by functions in $\mathbf{T}_n$:

$$S_n(x) = \frac{a_0}{2} + a_n \cos nx + \sum_{k=1}^{n-1} (a_k \cos kx + b_k \sin kx)$$

The solution to the Normal equations are:

$$a_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos kx dx, \quad k = 0, 1, \ldots, n$$
$$b_k = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin kx dx, \quad k = 1, \ldots, n-1$$

S_n when $n \to \infty$ is the Fourier series of f.

Discrete Trigonometric Approximation

- Consider the $2m$ points $\{(x_j, y_j)\}_{j=0}^{2m-1}$, with

$$x_j = -\pi + \frac{j}{m}\pi, \ j = 0, \ldots, 2m-1$$

- Find the trigonometric polynomial $S_n \in \mathbf{T}_n$ that minimizes

$$E(S_n) = \sum_{j=0}^{2m-1} [y_j - S_n(x_j)]^2$$

- Simplified by discrete orthogonality:

$$\sum_{j=0}^{2m-1} \phi_k(x_j)\phi_l(x_j) = 0$$

Discrete Trigonometric Approximation

Theorem

The trigonometric polynomial

$$S_n(x) = \left[\frac{a_0}{2} + a_n \cos nx + \sum_{k=1}^{n-1} (a_k \cos kx + b_k \sin kx) \right]$$

that minimizes the least squares sum

$$E(a_0, \ldots, a_n, b_1, \ldots, b_{n-1}) = \sum_{j=0}^{2m-1} (y_j - S_n(x_j))^2$$

are

$$a_k = \frac{1}{m} \sum_{j=0}^{2m-1} y_j \cos kx_j, \;\; k = 0, 1, \ldots, n$$

$$b_k = \frac{1}{m} \sum_{j=0}^{2m-1} y_j \sin kx_j, \;\; k = 1, \ldots, n-1$$

Interpolatory Trigonometric Polynomials

- Given $2m$ points (equally space) $\{(x_j, y_j)\}_{j=0}^{2m-1}$,

$$x_j = -\pi + \frac{j}{m}\pi, \; j = 0, \ldots, 2m-1$$

- Find a trigonometric polynomial $S_m \in \mathbf{T}_m$

$$S_m(x) = \frac{a_0 + a_m \cos mx}{2} + \sum_{k=1}^{m-1} (a_k \cos kx + b_k \sin kx)$$

such that

$$S_m(x_j) = y_j, \quad j = 0, 1, \ldots, 2m-1$$

- Note: NOT Discrete Trigonometric Approximation !!

Interpolatory Trigonometric Polynomials

The interpolatory trigonometric polynomial in $\mathbf{T}_m$ on the $2m$ points $\{(x_j, y_j)\}_{j=0}^{2m-1}$,

$$x_j = -\pi + \frac{j}{m}\pi, \; j = 0, \ldots, 2m-1$$

is

$$S_m(x) = \frac{a_0 + a_m \cos mx}{2} + \sum_{k=1}^{m-1} (a_k \cos kx + b_k \sin kx)$$

where

$$a_k = \frac{1}{m} \sum_{j=0}^{2m-1} y_j \cos kx_j, \; k = 0, 1, \ldots, m$$

$$b_k = \frac{1}{m} \sum_{j=0}^{2m-1} y_j \sin kx_j, \; k = 1, \ldots, m-1$$

Complex Form of the DFT

- DFT = Discrete Fourier Transform.
- Consider the complex coefficients c_k in the Fourier Transform

$$S_m(x) = \frac{1}{m} \sum_{k=0}^{2m-1} c_k e^{ikx}$$

where

$$c_k = \sum_{j=0}^{2m-1} y_j e^{ik\pi j/m}, \text{ for } k = 0, \ldots, 2m-1$$

- a_k and b_k can be recovered from c_k using Euler's formula $e^{iz} = \cos z + i \sin z$ as for $k = 0, 1, \ldots, m$.

$$a_k + ib_k = \frac{(-1)^k}{m} c_k$$

The Fast Fourier Transform (FFT) Algorithm

- Split the DFT into even and odd indices (assume $m = 2^p$): for $k = 0, 1, \ldots, m-1$,

$$\begin{aligned} c_k &= \sum_{j=0}^{2m-1} y_j e^{ik\pi j/m} = \sum_{j=0}^{m-1} y_{2j} e^{ik\pi 2j/m} + \sum_{j=0}^{m-1} y_{2j+1} e^{ik\pi(2j+1)/m} \\ &= \sum_{j=0}^{m-1} y_{2j} e^{ik\pi j/(m/2)} + e^{ik\pi/m} \sum_{j=0}^{m-1} y_{2j+1} e^{ik\pi j/(m/2)} \\ &= E_k + e^{ik\pi/m} O_k \end{aligned}$$

  where E_k is the DFT of the even index inputs x_{2j} and O_k is the DFT of the odd index inputs x_{2j+1}.

- For $k = m, m+1, \ldots, 2m-1$, compute c_k use

$$e^{i(k+m)\pi/m} = e^{i\pi} e^{ik\pi/m} = -e^{ik\pi/m}$$

The Fast Fourier Transform; Reduction of Operatrions

- The FFT algorithm can then be written:

$$c_k = \begin{cases} E_k + e^{ik\pi/m} Q_k & \text{if } k < m, \\ E_{k-m} - e^{ik\pi/m} Q_{k-m} & \text{if } k \geqslant m \end{cases}$$

- Only need to compute $E_k, \ Q_k$, for $k = 0, 1, \ldots, m-1$.
- From $(2m)^2 = 4m^2$ multiplications to $m(m + (m+1)) = m(2m+1) = 2m^2 + m$ multiplications
- $E_k, \ Q_k$ has similar form as c_k, further split into two parts, number of operations is further reduced;
- repeat $r = p + 1$ times, since $m = 2^p$.
- Use recursion, $O(m \log m)$ total work