Outline

• Uniform and Nonuniform Discrete Fourier Transforms
  – DFT, FFT, NUFFT, IDFT, IFFT, INUFFT
  – Statement of the problems

• INUFFT
  – Data Structure
  – Kernel Decomposition
  – “Middleman” part of the algorithm
  – FMM part of the algorithm
  – Error bounds
  – Complexity and Optimization

• Some Numerical Results
  – Error Analysis
  – Performance
Band-limited functions

\[ f(x) = \sum_{n=0}^{N-1} c_n e^{i nx} \]

- 2π-periodic function (complex)
- Fourier coefficients (complex)
- real
- bandwidth
Inverse Discrete Fourier Transform

IDFT: \( \{c_n\} \rightarrow \{f_k\} \)

\[
f_k = \sum_{n=0}^{N-1} c_n e^{i\pi n k / N}
\]

Straightforward: \( O(N^2) \)
IFFT: \( O(N \log N) \)
Inverse Nonuniform Discrete Fourier Transform

**INUDFT:** \( \{c_n\} \rightarrow \{g_j\} \)

\[ g_j = \sum_{n=0}^{N-1} c_n e^{i\pi y_j} \]

Straightforward: \( O(N^2) \)
Looking for: \( O(N\log N) \) (INUFFT)

\[ g_j = f(y_j), \quad j = 0, \ldots, N-1 \]
Forward Discrete Fourier Transform

\[ c_n = \frac{1}{N} \sum_{k=0}^{N-1} f_k e^{-inx_k} \]

Straightforward: \( O(N^2) \)
FFT: \( O(N \log N) \)
Forward Nonuniform Discrete Fourier Transform

$$\text{NUDFT: } \{g_j\} \rightarrow \{c_n\}$$

Straightforward ($O(N^3)$)

Looking for: $O(N\log N)$

(NUFFT)

$$g_j = f(y_j), \quad j = 0, \ldots, N-1$$
Inverse Nonuniform Fast Fourier Transform

INUFFT: $\{c_n\} \rightarrow \{g_j\}$

FFT

$\{f_k\}$

FFIA = “Fast Fourier Interpolation Algorithm”
\[ g_j = f(y_j) = \sum_{n=0}^{N-1} c_n e^{i\nu_j} = \sum_{k=0}^{N-1} \left[ \frac{1}{N} \sum_{n=0}^{N-1} e^{-i\nu_k} e^{i\nu_j} \right] f_k, \quad j = 0, \ldots, N - 1, \]

\[ \{g_j\} = \{K_{jk}\}\{f_k\}, \quad K_{jk} = \frac{1}{N} \sum_{n=0}^{N-1} e^{-i\nu_k} e^{i\nu_j}, \quad j, k = 0, \ldots, N - 1. \]

\[ K_{jk} = \frac{1}{N} \sum_{n=0}^{N-1} e^{in(y_j-x_k)} = \frac{1}{N} \frac{e^{iN\nu_j} - 1}{e^{i(y_j-x_k)} - 1} = F_j G(y_j - x_k), \]

where

\[ F_j = \frac{e^{iN\nu_j} - 1}{N}, \quad G(t) = \frac{1}{e^{it} - 1} = -\frac{1}{2} - i\frac{1}{2}H(t), \quad H(t) = \cot \frac{t}{2}. \]
The kernel $H(y_j - x_k)$ is a $2\pi$-periodic function. Due to this, we can make transforms of the form $\bar{x}_k = x_k + 2\pi n$, $n = 0, \pm 1, \ldots$, which do not change the function, and keep $y_j - \bar{x}_k$ in $-\pi \leq y_j - \bar{x}_k \leq \pi$, i.e., $y_j$ and $x_k$ can be considered as points on a unit circle, and then $x_k$ and $\bar{x}_k$ can be considered as identical.
Kernel Decomposition

\[ h(y_j) = \sum_{x_k \in \Omega_1(P_n)} f_k \cot \frac{y_j - x_k}{2} + \sum_{x_k \in \Omega_2(P_n)} f_k \cot \frac{y_j - x_k}{2}, \]

First sum:

\[ \cot \frac{t}{2} = \frac{2}{t} - 2 \sum_{m=1}^{\infty} \frac{|B_{2m}|}{(2m)!} t^{2m-1}, \quad |t| < 2\pi, \]

where \( B_m \) are the Bernoulli numbers. When applied to the computation of the sum, we note that for \( x_k \in \Omega_1(P_n) \) we have \(-\pi \leq y_j - \hat{x}_k \leq \pi\), so for \( t = y_j - \hat{x}_k \) we have \( |t| \leq \pi \). This provides fast convergence.

Second sum:
We introduce \( t = y_j - \hat{x}_k \), where \( \hat{x}_k = x_k \pm \pi \) is a point opposite to \( x_k \) with respect to the circle center. We have \(-\pi/2 \leq y_j - \hat{x}_k \leq \pi/2\), or \( |t| \leq \pi/2 \), while

\[ \cot \frac{y_j - x_k}{2} = -\tan \frac{t}{2} = -2 \sum_{m=1}^{\infty} \frac{(2^m - 1)|B_{2m}|}{(2m)!} t^{2m-1}, \quad |t| < \pi. \]
Errors of Truncation

\[ |B_{2m}| < \frac{2(2m)!}{(2\pi)^{2m}} \frac{1}{1 - 2^{1-2m}}, \quad m = 1, 2, ... \]  \hspace{1cm} (9)

This shows that the truncated part of the series can be majorated by the geometric progression. Using \(|t| \leq \pi\), the error bound is

\[ |\epsilon_q^{(1)}| = \left| \sum_{m=q+1}^{\infty} \frac{|B_{2m}| \cdot t^{2m-1}}{(2m)!} \right| < \frac{2^{1-2q}}{3\pi (1 - 2^{-2q-1})} = \epsilon_q. \]  \hspace{1cm} (10)

The truncation error, \(\epsilon_q^{(2)}\), in (8) can be bounded similarly, using \(|t| \leq \pi/2\):

\[ |\epsilon_q^{(2)}| < 2\epsilon_q. \]
Error Bound for the Regular Part Computation ("Middleman Error")

\[
\left| \varepsilon_{q}^{reg} \right| = \left| \sum_{k=0}^{N-1} \left( K_{jk} - K_{jk}^{(q)} \right) f_k \right| < \frac{2}{N} \left( \frac{3N\varepsilon_q}{4} + \frac{2N\varepsilon_q}{4} \right) = \frac{5\varepsilon_q}{2}.
\]
FMM and Total Error

\[ \left| \epsilon_p^{\\text{FMM-P}} \right| < \frac{5}{\pi} \cdot 2^{l_{\max}} 3^{-p}, \quad (11) \]

where \( l_{\max} \) is the maximum level of space subdivision and \( p \) is the truncation number used in the MLFMM. Therefore the total truncation error for the present method can be estimated as

\[ \epsilon \lesssim \frac{5}{3\pi} \left( 4^{-q} + 2^{l_{\max}} 3^{1-p} \right). \quad (12) \]

By requiring that the error in the singular and regular parts be the same in (12) we relate \( p \) and \( q \), and obtain a combined bound as

\[ q \gtrsim \frac{1}{2} \log_2 \frac{3\pi}{10\epsilon}, \quad p \gtrsim \log_3 \frac{3\pi}{10\epsilon} + \frac{l_{\max}}{\log_2 3} + 1. \quad (13) \]
Complexity

\[ h(y_j) = 2 \sum_{x_k \in \Omega_1(P_n)} \frac{f_k}{y_j - \tilde{x}_k} - \sum_{l=0}^{2q-1} \frac{d_l}{l!} \left( y_j - x_c^{(n)} \right)^l, \quad (14) \]

\[ d_l = \sum_{m=[l/2]+1}^q \frac{|B_{2m}|}{m} \left[ \alpha_{2m-l-1}^{(1)} + \left( 2^{2m} - 1 \right) \alpha_{2m-l-1}^{(2)} \right], \]

\[ \alpha_{l}^{(s)} = \frac{1}{l!} \sum_{x_k \in \Omega_s(P_n)} f_k \left( x_c^{(n)} - \tilde{x}_k \right)^l, \quad s = 1, 2. \]

For \( 2q \ll \min(N, M) \) the second sum in Eq. (14) is \( O\left( 2q(N + M) \right) \). For the complexity of the MLFMM in the present case we have [4]

\[ C_{FM} = O \left( p(N + M) + \frac{3NM}{2^{l_{\max}}} + \frac{6P}{2^{-l_{\max}}} \right), \quad (15) \]
Optimization

The total cost of the fast Fourier interpolation algorithm (FFIA) can be estimated, and minimized by selection of $l_{\text{max}}$ for given error (12). A simplified estimate assuming that $p$ and $q$ change slower than $2^{l_{\text{max}}}$, yields

$$l_{\text{max}}^{(opt)} \sim \frac{1}{2} \log_2 \frac{NM}{2p},$$

and the total complexity of the optimized algorithm will be

$$C_{FFIA}^{(opt)} = O \left( (N + M)(p + 2q) + 6[2NMP(p)]^{1/2} \right). \quad (17)$$

For $N \sim M$, this yields $C_{FFIA}^{(opt)} = O \left( N (\log N + \log \epsilon^{-1}) \right)$. 
Numerical Results: Error

\[ M = N, \quad q = q(\varepsilon), \quad p = p(\varepsilon, l_{\text{max}}) \]
Numerical Results: Optimization

\[ M = N, \, q = q(\varepsilon), \, p = p(\varepsilon, I_{\text{max}}) \]

Graph showing the relationship between CPU time and the maximum level for different values of \( N \). The graph includes markers for different values of \( N \) and lines indicating optimal \( I_{\text{max}} \) for various error tolerances.
Numerical Results: Performance

![Graph showing performance comparison between Straightforward and FFIA methods.]
Numerical Results: Truncation Numbers

$M = N, q = q(\varepsilon), p = p(\varepsilon, l_{\text{max}})$.