CS 450: Numerical Analysis

Fast Fourier Transform

University of Illinois at Urbana-Champaign

¹These slides have been drafted by Edgar Solomonik as lecture templates and supplementary material for the book “Scientific Computing: An Introductory Survey” by Michael T. Heath (slides).
Sparse Linear Systems and Time-independent PDEs

- The Poisson equation serves as a model problem for numerical methods:

- Dense, sparse direct, iterative, FFT, and Multigrid methods provide increasingly good complexity for the problem:
Multigrid

- Multigrid employs a hierarchy of grids to accelerate iterative methods:

- The multigrid method works by resolving high-frequency error components on finer-grids and low-frequency error components on coarser grids:
Consider the Galerkin approximation with linear finite elements to the Poisson equation $u'' = f(t)$ with boundary conditions $u(a) = u(b) = 0$:

$$
\phi_i^{(h)}(t) = \begin{cases} 
(t - t_{i-1})/h & : t \in [t_{i-1}, t_i] \\
(t_{i+1} - t)/h & : t \in [t_i, t_{i+1}] \\
0 & : \text{otherwise}
\end{cases}
$$

where $t_0 = t_1 = a$ and $t_{n+1} = t_n = b$. 

Multigrid
Coarse Grid Matrix

- Multigrid restricts the residual equation on the fine grid $A^{(h)} x = r^{(h)}$ to the coarse grid:
Restricting the Residual Equation

- Given the fine-grid residual $r^{(h)}$, we seek to use the coarse grid to approximate $x^{(h)}$ so that $Ax^{(h)} \approx r^{(h)}$
The solutions to hyperbolic PDEs like Poisson are wave-like and take on simple representations in the frequency basis, both for continuous and discretized equations. We define the discrete Fourier transform using

\[ \omega(n) = \cos\left(\frac{2\pi}{n}\right) - i \sin\left(\frac{2\pi}{n}\right) = e^{-2\pi i/n}. \]
Fast Fourier Transform (FFT)

Consider \( b = Fa \), we have

\[
\forall j \in [0, n - 1] \quad b_j = \sum_{k=0}^{n-1} \omega_{(n)}^{jk} a_k,
\]

the FFT computes this recursively via 2 FFTs of dimension \( n/2 \), using \( \omega_{(n/2)} = \omega_{(n)}^2 \).
The FFT leverages similarity between the first and second half of the output,

\[
b_j = \sum_{k=0}^{n/2-1} \omega_{(n/2)}^{jk} a_{2k} + \omega_{(n)}^{j} \sum_{k=0}^{n/2-1} \omega_{(n/2)}^{jk} a_{2k+1}
\]

\[
\begin{align*}
\text{corresponds closely to the entry shifted by } n/2, \\
 b_{j+n/2} &= \sum_{k=0}^{n/2-1} \omega_{(n/2)}^{(j+n/2)k} a_{2k} + \omega_{(n)}^{j+n/2} \sum_{k=0}^{n/2-1} \omega_{(n/2)}^{(j+n/2)k} a_{2k+1}
\end{align*}
\]
FFT Algorithm Summary

- Let vectors $u$ and $v$ be two recursive FFTs, $\forall j \in [0, n/2 - 1]$

  $$u_j = \sum_{k=0}^{n/2-1} \omega_{(n/2)}^j a_{2k}, \quad v_j = \sum_{k=0}^{n/2-1} \omega_{(n/2)}^j a_{2k+1}$$

- The FFT has $O(n \log n)$ cost complexity:
Applications of the FFT

▶ We can rapidly multiply degree $n$ polynomials by considering their values $\omega_i^{i} \mod (2n-1)$ for $i \in \{0, \ldots, 2n - 1\}$

▶ More generally the DFT can be used to solve any Toeplitz linear system (convolution):
Convolutions via DFT

The Fourier transform method for computing a convolution is given by

\[ c_k = \frac{1}{n} \sum_s \omega_{(n)}^{-ks} \left( \sum_j \omega_{(n)}^{sj} a_j \right) \left( \sum_t \omega_{(n)}^{st} b_t \right) \]
Solving Numerical PDEs with the FFT

- 1D finite-difference schemes on a regular grid correspond to convolutions:

- For the 1D Poisson model problem, the eigenvectors of $T$ correspond to the imaginary part of a minor of a $2(n + 1)$-dimensional DFT matrix:

- Multidimensional Poisson can be handled with multidimensional FFT: