

# CS 450: Numerical Analysis<sup>1</sup>

## Fast Fourier Transform

University of Illinois at Urbana-Champaign

---

<sup>1</sup>*These 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](#)).*





## Multigrid

- ▶ 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  $t_{n+1} = t_n = b$ .

## Coarse Grid Matrix

- ▶ Multigrid restricts the residual equation on the fine grid  $\mathbf{A}^{(h)}\mathbf{x} = \mathbf{r}^{(h)}$  to the coarse grid:

## Restricting the Residual Equation

- ▶ Given the fine-grid residual  $\mathbf{r}^{(h)}$ , we seek to use the coarse grid to approximate  $\mathbf{x}^{(h)}$  so that  $\mathbf{A}\mathbf{x}^{(h)} \approx \mathbf{r}^{(h)}$

## Discrete Fourier Transform

- ▶ 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

$$\omega_{(n)} = \cos(2\pi/n) - i \sin(2\pi/n) = e^{-2\pi i/n}.$$

## Fast Fourier Transform (FFT)

- ▶ Consider  $\mathbf{b} = \mathbf{F}\mathbf{a}$ , 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

$$\omega_{(n/2)} = \omega_{(n)}^2,$$



## Fast Fourier Transform Derivation

- ▶ The FFT leverages similarity between the first and second half of the output,

$$b_j = \underbrace{\sum_{k=0}^{n/2-1} \omega_{(n/2)}^{jk} a_{2k}}_{u_j} + \omega_{(n)}^j \underbrace{\sum_{k=0}^{n/2-1} \omega_{(n/2)}^{jk} a_{2k+1}}_{v_j}$$

corresponds closely to the entry shifted by  $n/2$ ,

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

## FFT Algorithm Summary

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

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

- ▶ The FFT has  $O(n \log n)$  cost complexity:



## Convolution 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  $T$  corresponds to the imaginary part of a minor of a  $2(n + 1)$ -dimensional DFT matrix:
- ▶ Multidimensional Poisson can be handled with multidimensional FFT: