
High Performance Computing
Lecture 17 - Fourier Transforms

Ole Nielsen

Case study: Fourier Transforms

Objectives

- Parallel computation of the Fast Fourier Transform in Python
- Static load balancing with a complex communication pattern
- Performance analysis

Outline

- **Lecture 17:** Discrete Fourier Transform (DFT)
Overview, Motivation
- Lecture 18: Fast Fourier Transform (FFT)
- Lecture 19: Parallel Fast FFT
- Lecture 20: Parallel Fast FFT

The Continuous Fourier Transform

Forward

$$g(\xi) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi\xi x} dx$$

Inverse

$$f(x) = \int_{-\infty}^{\infty} g(\xi) e^{i2\pi\xi x} d\xi$$

Provided

- $\int_{-\infty}^{\infty} |f(x)| dx$ exists.
- f has a finite number of discontinuities.
- f has bounded variation.

The Discrete Fourier Transform (DFT)

$$g_k = \frac{1}{N} \sum_{l=0}^{N-1} f_l e^{-i2\pi kl/N}, \quad k = 0, 1, \dots, N - 1$$

where

$$f_l = f(l/N), \quad l = 0, 1, \dots, N - 1$$

$$g_l = g(k/N), \quad k = 0, 1, \dots, N - 1$$

and f and g periodic functions.

Compact notation for DFT

$$g_k = \frac{1}{N} \sum_{l=0}^{N-1} f_l w^{kl}, \quad k = 0, 1, \dots, N - 1$$

where

$$w = e^{-i2\pi/N}$$

DFT Matrix

DFT

$$g_k = \frac{1}{N} \sum_{l=0}^{N-1} f_l w^{kl}, \quad k = 0, 1, \dots, N - 1$$

Matrix formulation of DFT

$$\mathbf{f} = (f_0, f_1, \dots, f_{N-1})^T$$

$$\mathbf{g} = (g_0, g_1, \dots, g_{N-1})^T$$

$$\mathbf{g} = \mathbf{F} \mathbf{f}, \quad [\mathbf{F}]_{kl} = w^{kl}$$

DFT matrix example ($N = 4$)

DFT

$$g_k = \frac{1}{N} \sum_{l=0}^3 f_l w^{kl}, \quad k = 0, 1, \dots, 3$$

Matrix formulation

$$\begin{pmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \end{pmatrix} = \begin{pmatrix} w^0 & w^0 & w^0 & w^0 \\ w^0 & w^1 & w^2 & w^3 \\ w^0 & w^2 & w^4 & w^6 \\ w^0 & w^3 & w^6 & w^9 \end{pmatrix} \begin{pmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

Properties

- Parsevals identity:

$$\langle \mathbf{F} \mathbf{x}, \mathbf{F} \mathbf{x} \rangle = N \langle \mathbf{x}, \mathbf{x} \rangle$$

- Convolution theorem

$$[\mathbf{f} * \mathbf{g}]_k = \sum_{l=0}^{N-1} f_l g_{k-l}$$

$$[\mathbf{F}(\mathbf{f} * \mathbf{g})]_k = [\mathbf{F} \mathbf{f}]_k [\mathbf{F} \mathbf{g}]_k$$

- Diagonalisation of differential operator
(See next slide)

The d 'th derivative of f

Use DFT to approximate derivatives

$$\mathbf{f}^{(d)} = \mathbf{F}^{-1} \mathbf{D}^{(d)} \mathbf{F} \mathbf{f}$$

where

$$\mathbf{f}^{(d)} = (f^{(d)}(0), f^{(d)}(1/N), \dots, f^{(d)}((N-1)/N))$$

Diagonal Differentiation matrix

$$[\mathbf{D}]_{k,k} = (2\pi k)^d, \quad k = 0, 1, \dots, N-1$$

A nonlinear PDE example

Nonlinear initial value PDE: Find $u(x, t)$ where

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + u^2, \quad t > 0 \quad \text{and} \quad u(x, 0) = f(x)$$

Spatial discretization

$$\mathbf{u}(t) = \left\{ u \left(\frac{k}{N}, t \right) \right\}_{k=0,1,\dots,N-1}$$

$$\frac{d\mathbf{u}}{dt}(t) = (\mathbf{L}^{(2)} + \mathbf{u}^2)\mathbf{u}(t)$$

where $\mathbf{L}^{(2)}$ the usual linear central difference operator and \mathbf{u}^2 taken elementwise.

Fourier split-step method

Exact solution

$$\mathbf{u}(t + \Delta t) = \exp \left(\Delta t \mathbf{L}^{(2)} + \int_t^{t+\Delta t} \mathbf{u}(\tau)^2 d\tau \right) \mathbf{u}(t)$$

Approximation

$$\mathbf{u}(t + \Delta t) = \mathbf{E} \mathbf{G} \mathbf{E} \mathbf{u}(t)$$

$$\mathbf{G} = \exp \left(\frac{\Delta t}{2} [\mathbf{u}_n^2 + \mathbf{u}_{n+1}^2] \right) \text{ diagonal (good)}$$

$$\mathbf{E} = \exp \left(\frac{\Delta t}{2} \mathbf{L}^{(2)} \right) \text{ dense (bad)}$$

Fourier Transform Diagonalises Differential Operator

$$\mathbf{f}^{(2)} = \mathbf{F}^{-1} \mathbf{D}^{(2)} \mathbf{F} \mathbf{f}$$

Approximation then becomes

$$\mathbf{u}(t + \Delta t) = (\mathbf{F}^{-1} \mathbf{E} \mathbf{F}) \mathbf{G} (\mathbf{F}^{-1} \mathbf{E} \mathbf{F}) \mathbf{u}(t)$$

$$\mathbf{G} = \exp\left(\frac{\Delta t}{2} [\mathbf{u}_n^2 + \mathbf{u}_{n+1}^2]\right) \text{ (diagonal)}$$

$$\mathbf{E} = \exp\left(\frac{\Delta t}{2} \mathbf{D}^{(2)}\right) \text{ (diagonal)}$$

Cost \propto the cost of a Fourier transform

Applications of Fourier Transforms

- Differential Equations
- Image processing: deconvolution and compression
- Sound: Synthesizing, analysing, modifying
- Tomography (e.g. Geology, Medical imaging)
- Many many more...

Fourier Transforms need to be fast!

Today's exercise

- Implement discrete Fourier transform
- Test correctness using some mathematical properties
- Test algorithmic complexity with timings