MA5233 Computational Mathematics

Lecture 8: Fast Fourier Transform

Simon Etter

National University of Singapore

2019/2020

Fast Fourier Transform

Recap: eigenvalues and -vectors of Laplacian

The eigenvalues and -vectors of

$$\Delta_n^{(1)} := (n+1)^2 \begin{pmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix}$$

are given by

$$\lambda_\ell = (n+1)^2 \left(2\cos\left(\pi \tfrac{\ell}{n+1}\right) - 2\right) \quad \text{and} \quad (u_\ell)_k = \sin\left(\pi \tfrac{\ell k}{n+1}\right)$$

for $\ell \in \{1, \dots n\}$.

Remarks

- More precisely, we showed $\Delta_n^{(1)} u_\ell = \lambda_\ell \, u_\ell$.
- To be certain that u_ℓ with $\ell \in \{1, \dots, n\}$ are all the eigenvectors, we need to show that they are linearly independent.

Fast Fourier Transform

Theorem (orthogonality of sin vectors)

The vectors $(u_\ell)_k = \sin\left(\pi \frac{\ell k}{n+1}\right)$ with $\ell \in \{1, \ldots, n\}$ are orthogonal,

$$u_{\ell_1}^T u_{\ell_2} = \tfrac{n+1}{2}\, \delta_{\ell_1 \ell_2} := \begin{cases} \frac{n+1}{2} & \text{if } \ell_1 = \ell_2, \\ 0 & \text{otherwise.} \end{cases}$$

Proof of this theorem will be based on lemma on the following slide.

Fast Fourier Transform

Fundamental lemma of (discrete) Fourier theory

$$\sum_{k=0}^{n-1} \exp\bigl(2\pi\iota \tfrac{\ell k}{n}\bigr) = \begin{cases} n & \text{if } \ell \in n\mathbb{Z} = \{\ldots, -n, 0, n, \ldots\}, \\ 0 & \text{otherwise.} \end{cases}$$

Proof. Result is obvious for $\ell \in n\mathbb{Z}$. For $\ell \in \mathbb{Z} \setminus \bigl(n\mathbb{Z}\bigr)$, we compute using formula for geometric sums that

$$\sum_{k=0}^{n-1} \exp\bigl(2\pi\iota \tfrac{\ell k}{n}\bigr) = \frac{\exp\bigl(2\pi\iota \frac{\ell n}{n}\bigr) - 1}{\exp\bigl(2\pi\iota \frac{\ell}{n}\bigr) - 1} = 0.$$

Remark

Continuous version of the above lemma:

$$\int_0^1 \exp\bigl(2\pi\iota\ell\, x\bigr)\, dx = \begin{cases} 1 & \text{if } \ell = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Fast Fourier Transform

Proof of orthogonality of sin vectors.

$$
\begin{aligned}
u_{\ell_1}^T u_{\ell_2} &= -\tfrac{1}{4} \sum_{k=1}^{n} \left(\exp\left(\pi\iota \tfrac{\ell_1 k}{n+1}\right) - \exp\left(-\pi\iota \tfrac{\ell_1 k}{n+1}\right) \right) \left(\exp\left(\pi\iota \tfrac{\ell_2 k}{n+1}\right) - \exp\left(-\pi\iota \tfrac{\ell_2 k}{n+1}\right) \right) \\
&= -\tfrac{1}{4} \left(\sum_{k=1}^{n} \exp\left(\pi\iota \tfrac{(\ell_1+\ell_2)k}{n+1}\right) + \sum_{k=1}^{n} \exp\left(-\pi\iota \tfrac{(\ell_1+\ell_2)k}{n+1}\right) \right. \\
&\qquad \left. - \sum_{k=1}^{n} \exp\left(\pi\iota \tfrac{(\ell_1-\ell_2)k}{n+1}\right) - \sum_{k=1}^{n} \exp\left(-\pi\iota \tfrac{(\ell_1-\ell_2)k}{n+1}\right) \right) \\
&= -\tfrac{1}{4} \left(\sum_{k=1}^{n} \exp\left(\pi\iota \tfrac{(\ell_1+\ell_2)k}{n+1}\right) + \sum_{k=1}^{n} \exp\left(\pi\iota \tfrac{(\ell_1+\ell_2)(2n+2-k)}{n+1}\right) \right. \\
&\qquad \left. - \sum_{k=1}^{n} \exp\left(\pi\iota \tfrac{(\ell_1-\ell_2)k}{n+1}\right) - \sum_{k=1}^{n} \exp\left(-\pi\iota \tfrac{(\ell_1-\ell_2)(2n+2-k)}{n+1}\right) \right) \\
&= -\tfrac{1}{4} \left(\sum_{k=0}^{2n+1} \exp\left(2\pi\iota \tfrac{(\ell_1+\ell_2)k}{2(n+1)}\right) - \sum_{k=0}^{2n+1} \exp\left(2\pi\iota \tfrac{(\ell_1-\ell_2)k}{2(n+1)}\right) \right)
\end{aligned}
$$

Fast Fourier Transform

Proof of orthogonality of sin vectors (continued).

Main steps on previous slide:

- Expand $\sin(x) = \frac{1}{2\iota}\big(\exp(x) - \exp(-x)\big)$.
- Rearrange to get terms of the form $\exp\big(\pi\iota\frac{\ell k}{n+1}\big)$.
- Use $\exp(2\pi\iota) = 1$ and cancellation to go from $k \in \{1,\ldots,n\}$ to $k \in \{0,\ldots,2n+1\}$.

The result was

$$u_{\ell_1}^T u_{\ell_2} = -\frac{1}{4}\left(\sum_{k=0}^{2n+1}\exp\Big(2\pi\iota\frac{(\ell_1+\ell_2)k}{2\,(n+1)}\Big) - \sum_{k=0}^{2n+1}\exp\Big(2\pi\iota\frac{(\ell_1-\ell_2)k}{2\,(n+1)}\Big)\right).$$

Recall $\ell_1, \ell_2 \in \{1,\ldots,n\}$.

- First term is zero since $\ell_1 + \ell_2 \notin 2\,(n+1)\,\mathbb{Z}$.
- Second term is zero except if $\ell_1 - \ell_2 = 0$, in which case we obtain

$$u_\ell^T u_\ell = \frac{2n+2}{4} = \frac{n+1}{2}.$$

Fast Fourier Transform

Sine matrix $(S_n)_{k\ell} := \sin\left(\pi \frac{\ell k}{n+1}\right)$

Corollaries

- Orthogonality of sin vectors may be written as $S_n^T S_n = \frac{n+1}{2} I$.
- Eigenvalue equation for 1d Laplacian $\Delta_n^{(1)}$ may be written as

$$\Delta_n^{(1)} = \frac{2}{n+1} S_n \Lambda_n S_n$$

where Λ_n is diagonal matrix given by

$$(\Lambda_n)_{\ell\ell} := (n+1)^2 \left(2\cos\left(\pi \frac{\ell}{n+1}\right) - 2\right).$$

- Eigenvalue equation for 2d Laplacian $\Delta_n^{(2)}$ may be written as

$$\Delta_n^{(2)} = \frac{4}{(n+1)^2} \left(S_n \otimes S_n\right) \left(\Lambda_n \otimes I + I \otimes \Lambda_n\right) \left(S_n \otimes S_n\right).$$

These matrices are orthogonal / diagonal, so inversion is easy!

Fast Fourier Transform

Vectorisation of matrices

Let $A \in \mathbb{K}^{n \times n}$ be a matrix. Its *vectorisation* $\mathrm{vec}(A) \in \mathbb{K}^{n^2}$ is obtained by stacking the columns of A into a long vector,

$$\mathrm{vec}(A)_{i+n\,(j-1)} := A_{ij}.$$

Theorem

Let $A, B, C \in \mathbb{K}^{n \times n}$ be matrices. Then,

$$\big(A \otimes B\big)\,\mathrm{vec}(C) = \mathrm{vec}\big(ACB^T\big).$$

Proof. Straightforward but tedious computations.

Fast Fourier Transform

Solving Poisson equation via sine transform, algorithm

Denote by $f_{k_1 k_2}$ the matrix (!) of point-values of $f(x, y)$ on the equispaced $n \times n$ mesh on $[0, 1]^2$.

The corresponding matrix of point-values $u_{k_1 k_2}$ of the solution to $-\Delta u = f$ can be computed as follows.

1. $\hat{f} := S_n \, f \, S_n$
2. $\hat{u}_{\ell_1 \ell_2} = \frac{\hat{f}_{\ell_1 \ell_2}}{-\lambda_{\ell_1} - \lambda_{\ell_2}}$
3. $u = \frac{4}{(n+1)^2} \, S_n \, \hat{u} \, S_n$

Rule of thumb for prefactor:

- Every factor of S_n requires a factor $\sqrt{\frac{2}{n+1}}$.
- Algorithm above uses S_n four times; hence factor is $\frac{4}{(n+1)^2}$.

Fast Fourier Transform

Solving Poisson equation via sine transform

Good: only matrix products required.

Bad: $S_n\, f\, S_n$ seems to require $\mathcal{O}(n^3) = \mathcal{O}(N^{3/2})$ FLOP.

No speedup compared to LU factorisation.

Fast Fourier Transform

Matrix-vector product $S_n\, v$ with $v \in \mathbb{R}^n$ can be evaluated using only $\mathcal{O}\big(n \log(n)\big)$ instead of $\mathcal{O}(n^2)$ FLOP!

Similar statements hold for multiplication with

- Fourier matrix: $F_{k\ell} := \exp\big(2\pi\iota \frac{k\ell}{n}\big)$ for $k, \ell \in \{0, \ldots, n-1\}$,
- cosine matrix: (similar to S_n, but more complicated boundary conditions).

Corollary

- $S_n\, f\, S_n$ can be computed in $\mathcal{O}\big(n^2 \log(n)\big) = \mathcal{O}\big(N \log(N)\big)$ FLOP.
- Even 2d (and 3d) Poisson equation is easy to solve. . .
- . . . iff $\Omega = [a, b]^d$ and $D = \text{const}$!

Fast Fourier Transform

The FFTW package

- FFTW: Fastest Fourier Transform in the West.
- Fastest publicly available code for Fourier and related transforms.
- Available in Julia as a `FFTW` package.
- See `8_fast_fourier_transform.jl` on how to use it, and

  `http://www.fftw.org/fftw3_doc/1d-Real_002dodd-DFTs-_0028DSTs_0029.html`

  for documentation regarding sine transform.