COMS E6998: Algorithms for Massive Data (Fall'23) Oct 11, 2023

Lecture 11: Discrete Fourier Transform

Instructor: *Alex Andoni* Scribes: *Peter Davis*

1 Final Projects

Project proposals will be due in 2 weeks. Proposals should be roughly one page and cover team members, papers to be considered, and general scope. The last two days of class will be presentations of final projects. There are three general types of projects (some projects may be a combination of these):

1. **Reading/survey:** Focus on a particular problem, read a few recent papers on that problem, and summarize in your own words. This should be to a level of detail as if you are preparing a lecture for your classmates.
2. **Implementation:** Code up some implementations of algorithms we have covered, and analyze it's practical performance. Projects of this form should be comparative in nature to other algorithms, and should have some message that can be taken away beyond simply "I implemented this algorithm and it ran in x seconds on this dataset".
3. **Research/open-ended:** Produce some novel theoretical result, such as proving an unproven theorem, or developing a new algorithm and proving it's properties.

Projects are expected to be completed in 2-3 person teams. 3 team members is a hard upper limit. This is intended as a collaborative exercise, so one-person projects are discouraged.

2 Recap on Compressed Sensing

Given a measurement $y \in \mathbb{R}^m$, and a signal $x \in \mathbb{R}^n$, and some sampling operator A such that $y = Ax$, we want to recover an $x' \approx x$, assuming x is approximately k-sparse. We showed that we can do this using $m \approx k \log \frac{n}{k}$.

In this case, we managed to extract $\approx m$ "pieces" of information from x, and showed that that is enough to recover x with good accuracy. If $m << n$, then we can say this is a sublinear compression. But the recovery process itself takes time $\mathcal{O}(poly(n))$ for LP, and $\mathcal{O}(nk)$ for Iterative Hard Thresholding.

Question: Can we get *sublinear* ($o(n)$) recovery time? **Yes**. For example, for IHT, we can design a more structured A matrix s.t. we can recover x' in $\mathcal{O}(k(\log n)^{\mathcal{O}(1)})$ time.

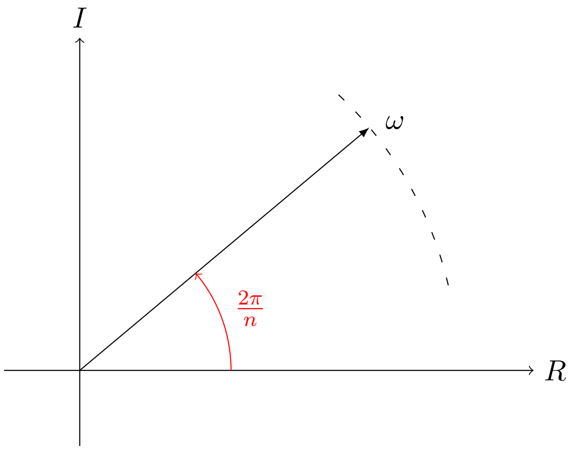

Figure 1: ω projected on the complex plane.

3 Fourier Transform

Definition 1. *Discrete Fourier Transform (DFT)*

Assume $n = 2^\ell$, $a \in \mathbb{R}^n$. Then $\hat{a} \in \mathbb{C}^n$ is the discrete Fourier transform of a, defined element-wise as:[1]

$$\hat{a}_u = \frac{1}{n} \sum_{j=0}^{n-1} a_j e^{-\frac{2\pi i}{n} uj}$$

(In this notation, a_0 and $\hat{a}_0$ are the first elements of a and $\hat{a}$)

Equivalently, we can define $\hat{a} = \mathcal{F}a$, where

$$\mathcal{F}_{uj} = \frac{1}{n} e^{-\frac{2\pi i}{n} uj} = \frac{\omega^{-uj}}{n}$$

where $\omega = e^{\frac{2\pi i}{n}}$. Intuitively, ω represents a $\frac{1}{n}$ rotation around the origin in the complex plane (figure 1), so the exponent of ω can be assumed to be modulo n.

We also have the **inverse** DFT, $a = \mathcal{F}^{-1}\hat{a}$, with $\mathcal{F}^{-1}_{ju} = \omega^{uj}$

Fact 2.

$$\|\mathcal{F}a\|_2^2 = \frac{1}{n}\|a\|_2^2$$

$$\|\mathcal{F}^{-1}\hat{a}\|_2^2 = n\|\hat{a}\|_2^2$$

The DFT has many uses in signal processing, where it is often thought of as shifting from a "time" basis to a "frequency" basis.

Observation 3. *Many signals can be well-approximated by a sparse DFT, i.e. $\hat{a}$ is approximately k-sparse in many applications.*

[1]The leading $\frac{1}{n}$ is a normalization factor that has some flexibility. The invariant is that the normalization of $\mathcal{F}$ and $\mathcal{F}^{-1}$ multiply to $\frac{1}{n}$. Different literature may use 1 for $\mathcal{F}$ and $\frac{1}{n}$ for $\mathcal{F}^{-1}$, or $\frac{1}{\sqrt{n}}$ for both. Different normalizations will change Fact 2 accordingly.

$$\begin{array}{ccccc} a & & \mathcal{F}^{-1} & & \hat{a} \\ \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ \vdots \\ a_n \end{bmatrix} & = & \begin{bmatrix} \mathcal{F}^{-1}{}_{11} & \mathcal{F}^{-1}{}_{12} & \cdots & \mathcal{F}^{-1}{}_{1n} \\ \mathcal{F}^{-1}{}_{21} & \mathcal{F}^{-1}{}_{22} & \cdots & \mathcal{F}^{-1}{}_{2n} \\ \mathcal{F}^{-1}{}_{31} & \mathcal{F}^{-1}{}_{32} & \cdots & \mathcal{F}^{-1}{}_{3n} \\ \mathcal{F}^{-1}{}_{41} & \mathcal{F}^{-1}{}_{42} & \cdots & \mathcal{F}^{-1}{}_{4n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{F}^{-1}{}_{n1} & \mathcal{F}^{-1}{}_{n2} & \cdots & \mathcal{F}^{-1}{}_{nn} \end{bmatrix} & \cdot & \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{a}_3 \\ \hat{a}_4 \\ \vdots \\ \hat{a}_n \end{bmatrix} \end{array}$$

Figure 2: Random selection of rows of the inverse Fourier transform $\mathcal{F}^{-1}$

Goal: compute a' or $\hat{a}'$ such that:

$$\|\hat{a} - \hat{a}'\|_2 \le c \cdot \min_{\hat{a}^k \in \text{k-sp.}} \|\hat{a} - \hat{a}^k\|_2$$

Intuitively, we are trying to find an $\hat{a}'$ with error that is similar to the error from a k-sparse approximation of $\hat{a}$, to a constant factor.

Theorem 4 (Candes-Tao '06)**.** *Let a compressed sensing matrix A be composed of m random rows from $\mathcal{F}^{-1}$, then for $m = \mathcal{O}\left(\frac{k}{\varepsilon^2}(\log n)^4\right)$, A is $(\mathcal{O}(k), \varepsilon)$-RIP with probability ≥ 0.9.*

In other words, we can get an approximation of $\hat{a}$ which is roughly as good as the best k-sparse approximation of $\hat{a}$ by looking at just a few rows of $\mathcal{F}^{-1}$ (figure 2) and their corresponding entries in a. Importantly, this result **does not extend** to other changes of basis; it is unique to the Fourier transform.

4 Sparse Fourier Transform (SFT)

Theorem 5. *For some $c = \mathcal{O}(1)$, it is possible to recover $\hat{a}'$ in time $\mathcal{O}\left(k(\log n)^{\mathcal{O}(1)}\right)$.*

This informally implies that an algorithm to recover $\hat{a}'$ uses at-most $\mathcal{O}\left(k(\log n)^{\mathcal{O}(1)}\right)$ positions in A, making it similar to a compressed sensing algorithm. We will not prove this theorem to full generality, but will prove some cases.

Case 1: $k = 1$ *and there is no error.*

This means the Fourier transform is approximated perfectly by a 1-sparse vector, e.g.:

$$\hat{a} = (0, 0, \ldots \hat{a}_u, \ldots 0)$$

Observation 6.

$$a_j = \sum_{v=0}^{n-1} \hat{a}_j \cdot \omega^{vj} = \hat{a}_u \cdot \omega^{uj}$$

This comes directly from applying $\mathcal{F}^{-1}$ to $\hat{a}$. Now if we look at the ratios between adjacent entries:

$$\frac{a_{j+1}}{a_j} = \frac{\hat{a}_u \cdot \omega^{u(j+1)}}{\hat{a}_u \cdot \omega^{uj}} = \omega^u$$

Using the geometric interpretation, we can extract u as

$$u = \frac{n}{2\pi} \angle(a_{j+1}, a_j)$$

And finally:

$$\hat{a}_u = \frac{a_j}{\omega^{uj}} = a_j \cdot \omega^{-uj}, \forall j$$

Looking ahead: when transition from the time domain to the frequency domain, it is possible that the noise will compact into specific elements, so in the noisy case we will pick j at random.