Postdoc Discovers One Weird Fourier Trick for Combinatorial Data

Graph theorists hate him!

tom denton

York University and the Fields Institute
Toronto, Canada

A Friday Morning in Montreal

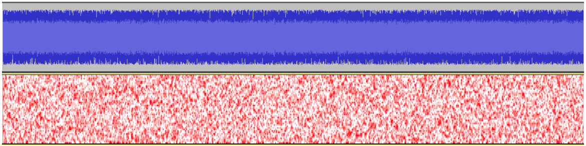
Usual Fourier Transform

- Time Domain: Function $f : S^1 \to \mathbb{C}$.
- Frequency Domain: Fourier transform $\hat{f} = (\ldots, \hat{f}_{-1}, \hat{f}_0, \hat{f}_1, \hat{f}_2, \ldots)$.
- Extremely useful: Frequency domain often has simpler structure, and some operations become very easy. (Convolution, etc.)

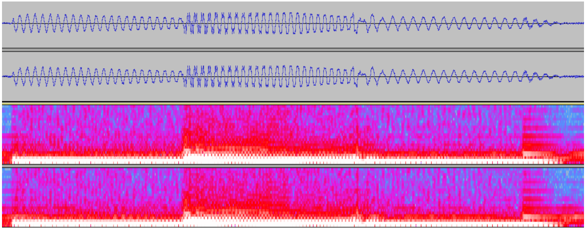
- We call f **band-limited** if all but a few of the coefficients $\hat{f}_i$ are zero.

Band Limited Functions

- Noise:



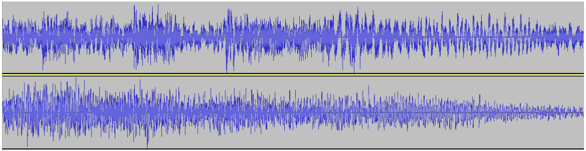
- Music:



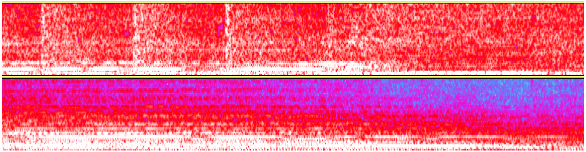
Usage in Machine Learning

It is often easier to find structural differences (ie, learn) in frequency space.

- Metallica vs Mozart: Time Domain



- Metallica vs Mozart: Frequency Domain



Fourier Transforms over S_n

Take $f : S_n \to \mathbb{C}$.
Choose a representation ρ of S_n; in particular, $\rho(\sigma)$ is a matrix for any $\sigma \in S_n$.

Definition

The Fourier transform of f at ρ is the matrix:

$$\hat{f}_\rho := \sum_{\sigma \in S_n} f(\sigma)\rho(sigma).$$

For the FT at an irreducible representation ρ_λ, we write $\hat{f}_\lambda$.

Properties of S_n Fourier Transform

- *Fourier Inversion Theorem:* For a collection $\{\rho_\lambda\}$ of irreducible representations of S_n, the collection of $\hat{f}_\lambda$ give a complete description of f.
- *Mean value* is given by trivial representation. Constant functions have $\hat{c}_\lambda = 0$ in all but trivial component.
- *Plancharel Formula* exists.
- *Convolution* is easy in 'frequency space:'

$$\widehat{f * g} = \hat{f}\hat{g}.$$

- *Translation*: Set $f^\pi(\sigma) := f(\pi^{-1}\sigma)$. Then:

$$\widehat{f^\pi} = \rho(\pi)\hat{f}.$$

Noise

Fourier transform of a randomly generated function on S_4.

'Time'

```
sage: f
[0.434977836537475*[1, 2, 3, 4],
 0.392063492238919*[1, 2, 4, 3],
 0.521516704615953*[1, 3, 2, 4],
 0.579389780448483*[1, 3, 4, 2],
 0.523673497481391*[1, 4, 2, 3],
 0.437116226899831*[1, 4, 3, 2],
 0.940504932065627*[2, 1, 3, 4],
 0.112353444528742*[2, 1, 4, 3],
 0.323783686524932*[2, 3, 1, 4],
 0.587326505347836*[2, 3, 4, 1],
 0.650670380618791*[2, 4, 1, 3],
 0.955910936983525*[2, 4, 3, 1],
 0.601537158504575*[3, 1, 2, 4],
 0.794394823511594*[3, 1, 4, 2],
 0.679884674700317*[3, 2, 1, 4],
 0.592560066726996*[3, 2, 4, 1],
 0.446587863295809*[3, 4, 1, 2],
 0.663383056294419*[3, 4, 2, 1],
 0.973295975527333*[4, 1, 2, 3],
 0.417614417812606*[4, 1, 3, 2],
 0.193123698781817*[4, 2, 1, 3],
 0.0491074958654644*[4, 2, 3, 1],
 0.701972353428726*[4, 3, 1, 2],
 0.182927737939954*[4, 3, 2, 1]]
```

'Frequency'

```
sage: for lam in irr.keys(): print lam, '\n', matrix(irr[lam]), '\n'
[1, 1, 1, 1]
[-2.25844652968]

[4]
[9.38378872464]

[3, 1]
[-0.635195612906  0.594839310665 -0.372979286523]
[ 0.487581803321 -0.392255077964   0.36462945528]
[-0.695137662289  -1.29950187318  -1.16837072065]

[2, 2]
[ -0.902808786609  -0.191227218508]
[  0.446283890917 0.00128951491858]

[2, 1, 1]
[   1.0138981529  0.523357218317  0.505581181152]
[-0.744064342263 -0.516668991817  0.265599994452]
[ 0.949840006734  -2.21664192545    1.1634017955]
```

Table 1: Fourier transform of a noisy function.

Music!

Consider the length function on S_n. Set:

$$f := \sum l(\sigma)\sigma$$

'Time' (n=4)

```
sage: f
[0,
 [1, 2, 4, 3],
 [1, 3, 2, 4],
 2*[1, 3, 4, 2],
 2*[1, 4, 2, 3],
 3*[1, 4, 3, 2],
 [2, 1, 3, 4],
 2*[2, 1, 4, 3],
 2*[2, 3, 1, 4],
 3*[2, 3, 4, 1],
 3*[2, 4, 1, 3],
 4*[2, 4, 3, 1],
 2*[3, 1, 2, 4],
 3*[3, 1, 4, 2],
 3*[3, 2, 1, 4],
 4*[3, 2, 4, 1],
 4*[3, 4, 1, 2],
 5*[3, 4, 2, 1],
 3*[4, 1, 2, 3],
 4*[4, 1, 3, 2],
 4*[4, 2, 1, 3],
 5*[4, 2, 3, 1],
 5*[4, 3, 1, 2],
 6*[4, 3, 2, 1]]
```

'Frequency' (n=6)

```
[5, 1]
[  -24.0  -41.57  -58.79  -75.89  -92.95]
[ -41.57   -72.0 -101.82 -131.45  -161.0]
[ -58.79 -101.82  -144.0  -185.9 -227.68]
[ -75.89 -131.45  -185.9  -240.0 -293.94]
[ -92.95  -161.0 -227.68 -293.94  -360.0]

[6]
[5400.0]

[4, 1, 1]
[ -12.0  -8.49  -14.7  -6.57 -11.38  -16.1  -5.37   -9.3 -13.15 -16.97]
[ -8.49   -6.0 -10.39  -4.65  -8.05 -11.38  -3.79  -6.57   -9.3  -12.0]
[ -14.7 -10.39  -18.0  -8.05 -13.94 -19.72  -6.57 -11.38  -16.1 -20.78]
[ -6.57  -4.65  -8.05   -3.6  -6.24  -8.82  -2.94  -5.09   -7.2   -9.3]
[-11.38  -8.05 -13.94  -6.24  -10.8 -15.27  -5.09  -8.82 -12.47  -16.1]
[ -16.1 -11.38 -19.72  -8.82 -15.27  -21.6   -7.2 -12.47 -17.64 -22.77]
[ -5.37  -3.79  -6.57  -2.94  -5.09   -7.2   -2.4  -4.16  -5.88  -7.59]
[  -9.3  -6.57 -11.38  -5.09  -8.82 -12.47  -4.16   -7.2 -10.18 -13.15]
[-13.15   -9.3  -16.1   -7.2 -12.47 -17.64  -5.88 -10.18  -14.4 -18.59]
[-16.97  -12.0 -20.78   -9.3  -16.1 -22.77  -7.59 -13.15 -18.59  -24.0]
```

Table 2: Fourier transform of a musical function.

All other $\hat{f}_\lambda = 0$!

Fast Fourier Transform on S_n

- Early 1990's: Clausen develops FFT over S_n.
- Uses embedding of S_{n-1} in S_n,
- Young's Seminormal/Orthogonal representation, and
- Branching of partitions in Young's lattice.

$$\begin{aligned}
\hat{f}_\lambda &= \sum_{\sigma \in S_n} f(\sigma)\rho_\lambda(\sigma) \\
&= \sum_{k=1}^{n} \sum_{\tau \in S_{n-1}} f(g_{k,n}\tau)\rho_\lambda(g_{k,n}\tau) \\
&= \sum_{k=1}^{n} \rho_\lambda(g_{k,n}) \sum_{\tau \in S_{n-1}} f(g_{k,n}\tau)\rho_\lambda(\tau)
\end{aligned}$$

Fast Fourier Transform on S_n II

- Then FFT over S_n can be written as a sum of FFT's over S_{n-1}.
- Furthermore, restriction to S_{n-1} is a 'twisted' block diagonal matrix, with blocks given by down-set of λ:

$$\rho_\lambda(g_{k,n}) \sum_{\tau \in S_{n-1}} f(g_{k,n}\tau)\rho_\lambda(\tau) = \rho_\lambda(g_{k,n}) \sum_{\tau \in S_{n-1}} \bigoplus_{\mu} f(g_{k,n}\tau)\rho_\mu(\tau)$$

- Efficient Algorithm: Roughly speaking, simultaneously sort all permutations, progressively building matrices for $\hat{f}$ at level S_{k+1} from matrices at level S_k.
- Uses $O(n!)$ memory, and approximately $O(n!n^3)$ time. (Clausen '93)

Band Restriction and the FFT

- Recall that constant functions are zero in all but trivial component.
- Then if a function h is constant on (say) S_{n-2}, its Fourier transform on restriction to S_{n-2} is zero away from the trivial component.
- Start from this trivial component, induct up twice to get the full set of non-zero components of $\hat{h}$.
- Set $h_{i,j}(\sigma) = 1$ if σ has an inversion at (i, j), and $h_{i,j}(\sigma) = 0$ otherwise.
- Then $h_{n-1,n}$ is constant on S_{n-2}, so band restricted.
- But all of the $h_{i,j}$ are *translations* of $h_{n-1,n}$. Then:

$$\widehat{h_{i,j}} = \rho(\pi)\widehat{h_{n-1,n}}.$$

Thus, $h_{i,j}$ is band-restricted as well.

Theorem for Length Function

Theorem (D?)

The length function is band-restricted. In particular, the only non-zero components are those associated to the partitions (n), $(n-1,1)$, and $(n-2,1,1)$.
The length function's Fourier transform can be computed in $O(n^5)$ time, and stored with $O(n^2)$ memory.

Note: Conceivably, could have non-zero coefficients in $\lambda = (n-2,2)$, but this ends up also being zero.

Similar theorems are easily written for a wide variety of interesting combinatorial statistics, including maj, number of peaks, number of descents, noninv(k) for fixed k, and more.

Encoding Graphs in $\mathbb{Z}S_n$

Cosider a labeled graph G with adjacency matrix A. Set :

$$f_G(\sigma) = A_{\sigma(n-1),\sigma(n)}.$$

- Then f_G completely encodes the adjacency matrix of G, and is constant on S_{n-2}; thus, severely band-restricted.
- Can compute the Fourier transform of f_G in $O(n^3)$ time (Kondar, 2008).
- Relabeling G with π induces a translation of f_G by π. Orthogonal representation gives *power invariants*:

  $$\hat{f}^t \hat{f}.$$

  These are graph invariants, due to the translation property of the FT.
- Similar games yield many other invariants.

What are these invariants?

For graphs on n vertices, consider the variables $X = \{x_{i,j}\}$ with $1 \leq i \neq j \leq n$ with action of the symmetric group:

$$\sigma \cdot x_{i,j} = x_{\sigma(i),\sigma(j)}.$$

- Form polynomials $p(X)\ldots$
- Then we can evaluate at a graph $p(G)$ by plugging in entries from the adjacency matrix.
- The symmetric group action is graph relabeling.
- Symmetrize a polynomial by Reynold's operator

$$R(p) = \sum_{\sigma} \sigma \cdot p(X).$$

Algebra of graph invariants

- The symmetric functions in the variables $X = \{x_{i,j}\}$ calculate invariants of unlabeled graphs.
- Introduce extra relation $x_{i,j}^2 = x_{i,j}$ to get a finite dimensional algebra B_n^*.
- Dimension of B_n^* is number of unlabeled graphs on n vertices, graded by number of edges.
- Can encode basically every hard problem in graph theory as polynomials in B_n^*.
- Example: Set $m_G = R(\prod x_{i,j})$ for $(i,j) \in E(G)$.
  Then $m_G(H)$ counts embeddings of G into H.
  Take $G = C_n$ to count Hamiltonian cycles. NP-complete!

Fourier transform?

- Fourier transform gives a way to efficiently *evaluate* certain collections of invariants.
- For two functions $f, g \in \mathbb{C}[S_n]$ (not symmetric), take **invariant product**:

$$f \odot g(\sigma) = \sum_{\tau \in S_n} f(\tau\sigma) g(\tau).$$

- Then $\widehat{f \odot g} = \hat{f}^t \hat{g}$, translation invariants.
- Can form f, g from any matrix associated to graph G: Powers of adjacency matrix, all-pairs-shortest-lengths, etc.
- These $\hat{f}^t \hat{g}$ are evaluations of symmetric polynomials in B_n^*.

Upshot

- The Fourier transform allows us to mix and match cheap invariants using simple matrix operations.
- The invariant product is algebraically different from addition, multiplication in B_n^*: Get interesting (or at least non-trivial) invariants.
- (Kondar, Borgwaldt) This is actually useful for machine learning problems involving weighted graphs.
  'Skew spectrum' is a collection of 49 invariants derived in this way; outperforms established feature sets for chemical data in three out of four tests.

Questions

- Numerous obvious directions for generalization.
- Symmetric functions on species?
- Integer eigenvalue mysteries. . .
- The algebras B_n^* and relatives are grossly understudied.
  I want:
  - Free generators for B_∞ indexed by connected graphs on k edges,
  - Hall inner product,
  - Polynomial-time evaluation of algebraic generators for B_n^*,
  - A pony.

Questions