Lecture 12: Fourier transforms
(Numerical Recipes, Chapter 12)

• Fourier transforms are important in a huge variety of physics applications: optics, quantum mechanics, classical mechanics …

• **Definitions:**
  Let \( h(t) \) be a (possibly complex) function of time defined for \(-\infty < t < \infty\)

  The Fourier transform is \( H(f) = \int_{-\infty}^{\infty} h(t) e^{i2\pi ft} \, dt \)

  and

  the inverse transform is \( h(t) = \int_{-\infty}^{\infty} H(f) e^{-i2\pi ft} \, df \)
Spectral power density

The *power* associated with the function $h(t)$ is defined by

$$P_{TOT} = \int_{-\infty}^{\infty} |h(t)|^2 \, dt = \int_{-\infty}^{\infty} dt \, h(t) \, h^*(t)$$

and is equal to

$$\int_{-\infty}^{\infty} dt \int_{-\infty}^{\infty} df \, H(f) \, e^{-i2\pi ft} \int_{-\infty}^{\infty} df' \, H^*(f') \, e^{i2\pi f't}$$

$$= \int_{-\infty}^{\infty} df \int_{-\infty}^{\infty} df' \, H(f) \, H^*(f') \int_{-\infty}^{\infty} dt \, e^{-i2\pi(f-f')t}$$

$$= \int_{-\infty}^{\infty} |H(f)|^2 \, df \quad \delta(f - f')$$
Spectral power density

The relation $P_{TOT} = \int_{-\infty}^{\infty} |H(f)|^2 df$ allows us define the spectral power density by

$$P(f) = |H(f)|^2 + |H(-f)|^2 \text{ for } f \geq 0$$

such that $P_{TOT} = \int_{0}^{\infty} P(f) df$
Spectral power density

Physical example: \( h(t) = \) electric field in an EM wave

Total flux = \((c/4\pi) |h(t)|^2\)

Monochromatic flux at frequency \( f \)

\[ = P(f) = (c/2\pi) |H(f)|^2 \]

Note: Whenever \( h(t) \) is real, \( H(f) = H^*(-f) \) and \( P(f) = 2 |H(f)|^2 \)
Measurements of $h(t)$

In real measurements of $h(t)$, we never measure $h(t)$ over an infinite time period. Nor do we measure with an infinite density of points. We tend to SAMPLE $h(t)$ on a discrete set of times

i.e. we obtain $h_k = h(k\Delta t)$

where $k = 0, 1, 2, 3, \ldots$
and $\Delta t$ is the “sampling interval”
Nyquist critical frequency

There is a special frequency associated with the sampling frequency called the Nyquist critical frequency:

\[ f_c = \frac{1}{2} \Delta t^{-1} \]

This is the maximum frequency that we can study from our measurements.
Sampling theorem

If $H(f) = 0$ for $|f| > f_c = \frac{1}{2} \Delta t^{-1}$

then $h(t)$ is completely determined by the discrete samples:

$$h(t) = \Delta t \sum_{i=1}^{N} h_k \sin \left[2\pi f_c(t - k\Delta t)\right] / \left[\pi(t - k\Delta t)\right]$$

Need at least 2 samples per cycle of a sine wave at the Nyquist frequency to determine its amplitude
Sampling theorem

However, if $H(f) \neq 0$ for $|f| > f_c$

then we cannot compute a correct $H(f)$ or $P(f)$ even for $f < f_c$

Higher frequencies will “feed down” into the measured range: this is called ALIASING
Aliasing

Example: it is impossible to distinguish a signal at $f = f_c$ from one at $f = 3f_c$. 

\[ \Delta t \]

\[ t \]
Aliasing

- Consider the superposition of two waves at frequencies $f_1$ and $f_2 = f_1 + m/\Delta t$, with $m$ any integer.

$$h(t) = a_1 \exp (i2\pi f_1 t) + a_2 \exp (i2\pi f_2 t)$$

$$h_k = h(k \Delta t) = a_1 \exp (i2\pi f_1 k \Delta t) + a_2 \exp (i2\pi [f_1 k \Delta t + km])$$

$$= [a_1 + a_2 \exp (2\pi imk)] \exp (2\pi i f_1 k \Delta t)$$

$$= [a_1 + a_2] \exp (2\pi i f_1 k \Delta t)$$
The discrete Fourier transform

• Real measurements always sample a function at a finite number of points N over a finite interval T

\[ h_k = h(k\Delta t), \text{ with } k = 0, 1, 2, 3, \ldots, N-1 \]
(We’ll assume N to be even)

• With N data points, we can only determine \( H(f) \) at N frequencies in the interval \(-f_c < f < f_c\)

\[ \rightarrow \text{ suggests we should consider only the finite set of frequencies, } f_n = n / N \Delta t \]
with \( n = -\frac{1}{2}N, -\frac{1}{2}N+1, \ldots, -1, 0, 1, \ldots, +\frac{1}{2}N \)
The discrete Fourier transform

The Fourier transform at these frequencies is

\[ H(f_n) = \int_{-\infty}^{\infty} h(t) \exp(i2\pi f_n t) \, dt \]

\[ \sim \quad \sum_{k=0}^{N-1} h_k \exp(i2\pi f_n k\Delta t) \Delta t \]

\[ = \quad \sum_{k=0}^{N-1} h_k \exp(i2\pi nk/N) \Delta t \equiv H_n \Delta t \]
The discrete Fourier transform

\[ H_n = \sum_{k=0}^{N-1} h_k \exp\left(i2\pi nk/N\right) \]

**Notes:**

- \( H_n \) is periodic in \( n \) with a period \( N \)
  - (since \( \exp i2\pi k = 1 \) for any integer \( k \))
  - \( H_n = H_{N+n} \) or \( H_{-n} = H_{N-n} \)

This allows us to renumber with \( n \) from 0 to \( N-1 \) instead of from \( -N/2 \) to \( +N/2 \)
The discrete Fourier transform

The inverse transform is

\[ h_k = \frac{1}{N} \sum_{n=0}^{N-1} H_n \exp(-i2\pi nk/N) \]

where we have made use of the fact that

\[ \sum_{n=0}^{N-1} \exp(-i2\pi n([k - k']/N)) = N \delta_{k,k'} \]
The discrete Fourier transform

The total power is defined as

$$P_{TOT} = \sum_{k=0}^{N-1} |h_k|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |H_n|^2$$
The discrete Fourier transform

- The calculation of a discrete FT is essentially the multiplication of a vector by a matrix

\[ H_n = \sum_{\kappa=0}^{N-1} h_\kappa \exp(i2\pi nk/N) = \sum_{\kappa=0}^{N-1} W^{nk} h_\kappa \]

where \( W = e^{2\pi i/N} \)

\( \Rightarrow \) requires \( \sim N^2 \) operations
The Fast Fourier Transform

• The direct calculation costing $\sim N^2$ operations is probably the easiest approach for $N < 1000$
• It is not the fastest way, however.
• It was realized in the 1960’s that there are much more efficient ways of doing the calculation: we describe next a one such method
Danielson-Lanczos Lemma

Write \( H_n = \sum_{k=0}^{N-1} h_k \exp(i2\pi nk/N) \)

\[
\begin{align*}
&= \sum_{k=0}^{[N/2]-1} h_{2k} \exp[i2\pi(2k)n/N] + \sum_{k=0}^{[N/2]-1} h_{2k+1} \exp[i2\pi(2k+1)n/N] \\
&= \sum_{k=0}^{[N/2]-1} h_{2k} \exp[i2\pi kn/(N/2)] + W^n \sum_{k=0}^{[N/2]-1} h_{2k+1} \exp[i2\pi kn/(N/2)] \\
&= H_n^e + W^n H_n^o
\end{align*}
\]

\( W = \exp(2\pi i/N) \) as before
The Fast Fourier Transform

• Key point: $H_n^0$ and $H_n^e$ are sampled half as densely as $H_n$
  – There are only $N/2$ sample points and the Nyquist frequency is only one-half that for $H_n$
  – $H_n^0$ and $H_n^e$ therefore repeat after the first $N/2$ n-values $\Rightarrow$ only need to calculate them for the first $N/2 – 1$ terms
  – The we gain a factor of 2 in computation speed

• Of course, we can do this again, and further subdivide $H_n^0$ and $H_n^e$
The Fast Fourier Transform

Write $H_n^e = \sum_{\kappa=0}^{[N/2]-1} h_{2\kappa} \exp[i2\pi kn/(N/2)]$

$= \sum_{\kappa=0}^{[N/4]-1} h_{4\kappa} \exp[i2\pi(2\kappa)n/(N/2)] + \sum_{\kappa=0}^{[N/4]-1} h_{4\kappa+1} \exp[i2\pi(2\kappa+1)n/(N/2)]$

$= \sum_{\kappa=0}^{[N/4]-1} h_{4\kappa} \exp[i2\pi kn/(N/4)] + W^{2n} \sum_{\kappa=0}^{[N/4]-1} h_{4\kappa+1} \exp[i2\pi kn/(N/4)]$

Transform sampled at $N/4$

$= H_n^e + W^{2n} H_n^{eo}$

$W = \exp(2\pi i/N)$ as before
The Fast Fourier Transform

• We have now obtained a factor of \( \sim \) four improvement in speed

• \( H_n = H_{n\text{ee}} + W^{2n}H_{n\text{eo}} + W^n H_{n\text{oe}} + W^{3n} H_{n\text{oo}} \)

• Four items, each with \( N/4 \) samples,
  \( \Rightarrow \) Computational cost \( \sim 4 \times (N/4)^2 \sim N^2/4 \)
The Fast Fourier Transform

\[ H_n = H_{n}^{ee} + W^{2n}H_{n}^{eo} + W^{n}H_{n}^{oe} + W^{3n}H_{n}^{oo} \]

Involves \( h_0, h_4, h_8 \ldots \)

Involves \( h_2, h_6, h_{10} \ldots \)

Involves \( h_1, h_5, h_9 \ldots \)

Involves \( h_3, h_7, h_{11} \ldots \)
## The Fast Fourier Transform

<table>
<thead>
<tr>
<th>Term</th>
<th>Rewrite as</th>
<th>k-values</th>
<th>in binary</th>
<th>Bit-reversed</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_{n \text{ ee}}$</td>
<td>$H_n^{00}$</td>
<td>0, 4, 8, …</td>
<td>0000, 0100, 1000, 1100 …</td>
<td>0000, 0010, 0001, 0011</td>
</tr>
<tr>
<td>$H_{n \text{ eo}}$</td>
<td>$H_n^{01}$</td>
<td>2, 6, 10, …</td>
<td>0010, 0110, 1010, 1110 …</td>
<td>0100, 0110, 0101, 0111</td>
</tr>
<tr>
<td>$H_{n \text{ oe}}$</td>
<td>$H_n^{10}$</td>
<td>1, 5, 9, …</td>
<td>0001, 0101, 1001, 1101 …</td>
<td>1000, 1010, 1001, 1011</td>
</tr>
<tr>
<td>$H_{n \text{ oo}}$</td>
<td>$H_n^{11}$</td>
<td>3, 7, 11, …</td>
<td>0011, 0111, 1011, 1111 …</td>
<td>1100, 1110, 1101, 1111 …</td>
</tr>
</tbody>
</table>

1 for odd, 0 for even
The Fast Fourier Transform

- This suggests a very clever scheme
  - Start with $N = 2^m$
  - Do the subdivision $m$ times, until each term contains a single sample

  e.g. $H_n^{eeoeoeoeoe} = H_n^{001010010} = h_k$

  with $k = 010010100_2 = 148$ in this example
The Fast Fourier Transform

- Resort terms by bit-reversed k values
- Combine adjacent pairs into two-point functions
- Combine adjacent pairs of pairs into 4 point functions
- Combine 4-point functions into 8-point functions

- Each step involves \( \sim N \) operations, and there are \( \sim \log_2 N \) steps
- Total cost \( \sim N \log_2 N \)
The Fast Fourier Transform

• This can be a huge improvement over an $O(N^2)$ algorithm
  – For $N = 10^4 \Rightarrow 750$ times faster
  – For $N = 10^6 \Rightarrow 50,000$ times faster

• For $h(t)$ real, a further factor $\sim 2$ improvement results because $H^*(f) = H(-f)$