Discrete Fourier Transform

Nuno Vasconcelos
UCSD
The Discrete-Space Fourier Transform

• as in 1D, an important concept in linear system analysis is that of the Fourier transform
• the Discrete-Space Fourier Transform is the 2D extension of the Discrete-Time Fourier Transform

\[ X(\omega_1, \omega_2) = \sum_{n_1} \sum_{n_2} x[n_1, n_2] e^{-j\omega_1 n_1} e^{-j\omega_2 n_2} \]

\[ x[n_1, n_2] = \frac{1}{(2\pi)^2} \int \int X(\omega_1, \omega_2) e^{j\omega_1 n_1} e^{j\omega_2 n_2} d\omega_1 d\omega_2 \]

• note that this is a continuous function of frequency
  – inconvenient to evaluate numerically in DSP hardware
  – we need a discrete version
  – this is the 2D Discrete Fourier Transform (2D-DFT)

• before that we consider the sampling problem
Sampling in 2D

• consider an analog signal $x_c(t_1, t_2)$ and let its analog Fourier transform be $X_c(\Omega_1, \Omega_2)$
  – we use capital $\Omega$ to emphasize that this is analog frequency
• sample with period $(T_1, T_2)$ to obtain a discrete-space signal

$$x[n_1, n_2] = x_c(t_1, t_2) \bigg|_{t_1=n_1 T_1; t_2=n_2 T_2}$$
Sampling in 2D

- relationship between the Discrete-Space FT of $x[n_1,n_2]$ and the FT of $x_c(t_1,t_2)$ is simple extension of 1D result

\[
X(\omega_1,\omega_2) = \frac{1}{T_1 T_2} \sum_{r_1=-\infty}^{\infty} \sum_{r_2=-\infty}^{\infty} X_c\left(\frac{\omega_1 - 2\pi r_1}{T_1}, \frac{\omega_2 - 2\pi r_2}{T_2}\right)
\]

- DSFT of $x[n_1,n_2]$ “discrete spectrum”
- FT of $x_c(\omega_1,\omega_2)$ “analog spectrum”

- Discrete Space spectrum is sum of replicas of analog spectrum
  - in the “base replica” the analog frequency $\Omega_1 (\Omega_2)$ is mapped into the digital frequency $\Omega_1 T_1 (\Omega_2 T_2)$
  - discrete spectrum has periodicity $(2\pi, 2\pi)$
For example

\[ \Omega' \rightarrow \alpha = \Omega' T_1 \]
\[ \Omega'' \rightarrow \beta = \Omega'' T_2 \]

- no aliasing if
\[
\begin{align*}
\Omega' T_1 &\leq 2\pi - \Omega' T_1 \\
\Omega'' T_2 &\leq 2\pi - \Omega' T_2
\end{align*}
\]
\[\Leftrightarrow\]
\[
\begin{align*}
T_1 &\leq \pi / \Omega' \\
T_2 &\leq \pi / \Omega''
\end{align*}
\]
Aliasing

- the frequency \((\Omega'/\pi, \Omega''/\pi)\) is the critical sampling frequency
- below it we have aliasing
- this is just like the 1D case, but now there are more possibilities for overlap
Reconstruction

• if there is no aliasing we can recover the signal in a way similar to the 1D case

\[
y_c(t_1, t_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} x[n_1, n_2] \frac{\sin \frac{\pi}{T_1} (t_1 - n_1 T_1)}{\frac{\pi}{T_1} (t_1 - n_1 T_1)} \frac{\sin \frac{\pi}{T_2} (t_2 - n_2 T_2)}{\frac{\pi}{T_2} (t_2 - n_2 T_2)}
\]

• note: in 2D there are many more possibilities than in 1D
  – e.g. the sampling grid does not have to be rectangular, e.g. hexagonal sampling when \( T_2 = T_1/sqrt(3) \) and

\[
x[n_1, n_2] = \begin{cases} 
  x_c(t_1, t_2) & |t_1 = n_1 T_1; t_2 = n_2 T_2 \quad n_1, n_2 \text{ both even or odd} \\
  0 & \text{otherwise}
\end{cases}
\]
  – in practice, however, one usually adopts the rectangular grid
• a sequence of images obtained by down-sampling **without any filtering**

• aliasing: the **low-frequency parts are replicated throughout the low-res image**
The role of smoothing

- too little leads to aliasing
- too much leads to loss of information
Aliasing in video

- video frames are the result of temporal sampling
  - fast moving objects are above the critical frequency
  - above a certain speed they are aliased and appear to move backwards
  - this was common in old western movies and become known as the “wagon wheel” effect
  - here is an example: super-resolution increases the frame rate and eliminates aliasing

from “Space-Time Resolution in Video” by E. Shechtman, Y. Caspi and M. Irani
(PAMI 2005).
2D-DFT

- the 2D-DFT is obtained by sampling the DSFT at regular frequency intervals

\[ X[k_1, k_2] = X(\omega_1, \omega_2) \bigg|_{\omega_1 = \frac{2\pi}{N_1} k_1, \omega_2 = \frac{2\pi}{N_2} k_2} \]

- this turns out to make the 2D-DFT somewhat harder to work with than the DSFT
  - it is the same as in 1D
  - you might remember that the inverse transform of the product of two DFTs is not the convolution of the associated signals
  - but, instead, the “circular convolution”
  - where does this come from?

- it is better understood by first considering the 2D Discrete Fourier Series (2D-DFS)
2D-DFS

- it is the natural representation for a periodic sequence
- a sequence $x[n_1, n_2]$ is periodic of period $N_1 \times N_2$ if

$$x[n_1, n_2] = x[n_1 + N_1, n_2] = x[n_1, n_2 + N_2], \quad \forall n_1, n_2$$

- note that

$$X(r_1, r_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} x[n_1, n_2] r_1^{-jn_1} r_2^{-jn_2}$$

- makes no sense for a periodic signal
  - the sum will be infinite for any pair $r_1, r_2$
  - neither the 2D DSFT or the Z-transform will work here
2D-DFS

- the 2D-DFS solves this problem
- it is based on the observation that
  - any periodic sequence can be represented as a weighted sum of complex exponentials of the form
    \[
    X(k_1, k_2) \times e^{j \frac{2\pi}{N_1} k_1 n_1} \times e^{j \frac{2\pi}{N_2} k_2 n_2}, \quad 0 \leq k_1 \leq N_1 - 1, \\
    0 \leq k_2 \leq N_2 - 1
    \]
  - this is a simple consequence of the fact that
    \[
    e^{j \frac{2\pi}{N_1} k_1 n_1} \times e^{j \frac{2\pi}{N_2} k_2 n_2}, \quad 0 \leq k_1 \leq N_1 - 1, 0 \leq k_2 \leq N_2 - 1
    \]
  - is an orthonormal basis of the space of periodic sequences
2D-DFS

- The 2D-DFS relates \( x[n_1, n_2] \) and \( X[k_1, k_2] \)

\[
X(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x[n_1, n_2] e^{-j\frac{2\pi}{N_1} k_1 n_1} e^{-j\frac{2\pi}{N_2} k_2 n_2}
\]

\[
x[n_1, n_2] = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} X[k_1, k_2] e^{j\frac{2\pi}{N_1} k_1 n_1} e^{j\frac{2\pi}{N_2} k_2 n_2}
\]

- Note that \( X[k_1, k_2] \) is also periodic outside

\[
0 \leq k_1 \leq N_1 - 1, \ 0 \leq k_2 \leq N_2 - 1
\]

- Like the DSFT,
  - Properties of the 2D-DFS are identical to those of the 1D-DFS
  - With the straightforward extension of separability
Periodic convolution

• like the Fourier transform,
  – the inverse transform of multiplication is convolution

\[
x[n_1, n_2] \ast x[n_1, n_2] \iff X(k_1, k_2) \times Y(k_1, k_2)
\]

– however, we have to be careful about how we define convolution
– since the sequences have no end, the standard definition

\[
y[n_1, n_2] = \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2]
\]

makes no sense
– e.g. if x and h are both positive sequences, this will always be infinite
Periodic convolution

• to deal with this, we introduce the idea of periodic convolution

• instead of the regular definition

\[ x \ast y = \sum_{k_1 = -\infty}^{\infty} \sum_{k_2 = -\infty}^{\infty} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2] \]

• which, from now on, we refer to as linear convolution

• periodic convolution only considers one period of our sequences

\[ x \circ h = \sum_{k_1 = 0}^{N_1 - 1} \sum_{k_2 = 0}^{N_2 - 1} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2] \]

• the only difference is in the summation limits
Periodic convolution

• this is simple, but produces a convolution which is substantially different
• let’s go back to our example, now assuming that the sequences have period \( (N_1=3, N_2=2) \)

\[
\begin{align*}
\mathbf{x}[n_1, n_2] & \quad \star \quad \mathbf{h}[n_1, n_2]
\end{align*}
\]

• as before, we need **four steps**
Periodic convolution

• step 1): express sequences in terms of \((k_1, k_2)\), and consider one period only

\[
x \circ h = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2]
\]

we next proceed exactly as before
Periodic convolution

• step 2): invert $h(k_1, k_2)$

$$x \circ h = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2]$$

\[ h[k_1, k_2] \quad h[k_1, -k_2] \quad g[k_1, k_2] = h[-k_1, -k_2] \]
Periodic convolution

- **step 3):** shift $g(k_1, k_2)$ by $(n_1, n_2)$

\[
x \circ h = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2]
\]

This sends whatever is at $(0,0)$ to $(n_1,n_2)$

\[
g[k_1, k_2] = h[-k_1,-k_2]
\]

\[
g[k_1 - n_1, k_2 - n_2] = h[n_1 - k_1, n_2 - k_2]
\]
Periodic convolution

- e.g. for \((n_1, n_2) = (1, 0)\)

\[
x[k_1, k_2] 
\]

\[
g[k_1 - n_1, k_2 - n_2] = h[n_1 - k_1, n_2 - k_2] 
\]

the next step would be to point-wise multiply the two signals and sum

\[
x \odot h = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2] 
\]
Periodic convolution

- this is where we depart from linear convolution
- remember that the sequences are periodic, and we really only care about what happens in the fundamental period
- we use the periodicity to fill the values missing in the flipped sequence

\[
\begin{align*}
\quad & h[n_1 - k_1, n_2 - k_2 ] \\
= & h[n_1 - k_1, n_2 - k_2 ]
\end{align*}
\]
Periodic convolution

- **step 4):** we can finally **point-wise multiply** the two signals and sum

\[ x \circ h = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} x[k_1, k_2] h[n_1 - k_1, n_2 - k_2] \]

- e.g. for \((n_1,n_2) = (1,0)\)
Periodic convolution

- note: the sequence that results from the convolution is also periodic
- it is important to keep in mind what we have done
  - we work with a single period (the fundamental period) to make things manageable
  - but remember that we have periodic sequences
  - it is like if we were peeking through a window
  - if we shift, or flip the sequence we need to remember that
  - the sequence does not simply move out of the window, but the next period walks in!!!
  - note, that this can make the fundamental period change considerably
Discrete Fourier Transform

• all of this is interesting,
  – but why do I care about periodic sequences?
  – all images are finite, I could never have such a sequence

• while this is true
  – the DFS is the easiest route to learn about the discrete Fourier transform (DFT)

• recall that the DFT is obtained by sampling the DSFT

\[ X[k_1, k_2] = X(\omega_1, \omega_2) \bigg|_{\omega_1 = \frac{2\pi k_1}{N_1}, \omega_2 = \frac{2\pi k_2}{N_2}} \]

– we know that when we sample in space we have aliasing in frequency

– well, the same happens when we sample in frequency: we get aliasing in time
Discrete Fourier Transform

- this means that
  - even if we have a finite sequence
  - when we compute the DFT we are effectively working with a periodic sequence

- you may recall from 1D signal processing that
  - when you multiply two DFTs, you do not get convolution in time
  - but instead something called circular convolution
  - this is strange: when I flip a signal (e.g. convolution) it wraps around the sequence borders
  - well, in 2D it gets much stranger

- the only way I know how to understand this is to
  - think about the underlying periodic sequence
  - establish a connection between the DFT and the DFS of that sequence
  - use what we have seen for the DFS to help me out with the DFT
Discrete Fourier Transform

• let’s start by the relation between DFT and DFS
• the DFT is defined as

$$X[k_1, k_2] = X(\omega_1, \omega_2)\bigg|_{\omega_1 = \frac{2\pi}{N_1} k_1, \omega_2 = \frac{2\pi}{N_2} k_2}$$

(here $X(\omega_1, \omega_2)$ is the DSFT) which can be written as

$$X[k_1, k_2] = \begin{cases} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x[n_1, n_2] e^{-j\frac{2\pi}{N_1} k_1 n_1} e^{-j\frac{2\pi}{N_2} k_2 n_2}, & 0 \leq k_1 < N_1, 0 \leq k_2 < N_2 \\ 0, & \text{otherwise} \end{cases}$$

$$x[n_1, n_2] = \begin{cases} \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} X[k_1, k_2] e^{j\frac{2\pi}{N_1} k_1 n_1} e^{j\frac{2\pi}{N_2} k_2 n_2}, & 0 \leq n_1 < N_1, 0 \leq n_2 < N_2 \\ 0, & \text{otherwise} \end{cases}$$
Discrete Fourier Transform

• comparing this

\[
X[k_1, k_2] = \begin{cases} 
\sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x[n_1, n_2] e^{-j\frac{2\pi}{N_1}k_1n_1} e^{-j\frac{2\pi}{N_2}k_2n_2}, & 0 \leq k_1 < N_1, \ 0 \leq k_2 < N_2 \\
0, & \text{otherwise}
\end{cases}
\]

\[
x[n_1, n_2] = \begin{cases} 
\frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} X[k_1, k_2] e^{j\frac{2\pi}{N_1}k_1n_1} e^{j\frac{2\pi}{N_2}k_2n_2}, & 0 \leq n_1 < N_1, \ 0 \leq n_2 < N_2 \\
0, & \text{otherwise}
\end{cases}
\]

with the DFS:

\[
X[k_1, k_2] = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x[n_1, n_2] e^{-j\frac{2\pi}{N_1}k_1n_1} e^{-j\frac{2\pi}{N_2}k_2n_2}
\]

\[
x[n_1, n_2] = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} X[k_1, k_2] e^{j\frac{2\pi}{N_1}k_1n_1} e^{j\frac{2\pi}{N_2}k_2n_2}
\]
Discrete Fourier Transform

• we see that inside the boxes

\[
0 \leq k_1 < N_1 \\
0 \leq k_2 < N_2
\]

the two transforms are exactly the same

• if we define the indicator function of the box

\[
R_{N_1 \times N_2}[n_1, n_2] = \begin{cases} 
1, & 0 \leq n_1 < N_1 \\
1, & 0 \leq n_2 < N_2 \\
0, & \text{otherwise}
\end{cases}
\]

• we can write

\[
x[n_1, n_2] = x[n_1, n_2] R_{N_1 \times N_2}[n_1, n_2] \\
X[k_1, k_2] = X[k_1, k_2] R_{N_1 \times N_2}[k_1, k_2]
\]
Discrete Fourier Transform

- note from

\[ x[n_1, n_2] = x[n_1, n_2] \mathcal{R}_{N_1 \times N_2} [n_1, n_2] \quad X[k_1, k_2] = X[k_1, k_2] \mathcal{R}_{N_1 \times N_2} [k_1, k_2] \]

that working in the DFT domain is equivalent to
- working in the DFS domain
- extracting the fundamental period at the end

- we can summarize this as

\[ x[n_1, n_2] \ \rightarrow \ \mathcal{D} \ \leftrightarrow \ \mathcal{F} \ \rightarrow \ \mathcal{D} \ \rightarrow \ X[k_1, k_2] \]

- in this way, I can work with the DFT without having to worry about aliasing
Discrete Fourier Transform

\[
x[n_1, n_2] \rightarrow x[n_1, n_2] \leftrightarrow X[k_1, k_2] \rightarrow X[k_1, k_2]
\]

- this trick can be used to derive all the DFT properties
- e.g. what is the inverse transform of a phase shift?
  - let’s follow the steps

\[
Y[k_1, k_2] = X[k_1, k_2] e^{-j\frac{2\pi}{N_1}k_1m_1} e^{-j\frac{2\pi}{N_2}k_2m_2}
\]

- 1) periodicize: this causes the same phase shift in the DFS

\[
Y[k_1, k_2] = X[k_1, k_2] e^{-j\frac{2\pi}{N_1}k_1m_1} e^{-j\frac{2\pi}{N_2}k_2m_2}
\]
Discrete Fourier Transform

\[ x[n_1, n_2] \xrightarrow{\text{periodicize}} x[n_1, n_2] \xrightarrow{\text{DFS}} X[k_1, k_2] \xrightarrow{\text{truncate}} X[k_1, k_2] \xrightarrow{\text{periodicize}} \]

- 2) compute the inverse DFS: it follows from the properties of the DFS (page 142 on Lim) that we get a shift in space

\[ y[n_1, n_2] = x[n_1 - m_1, n_2 - m_2] \]

- 3) truncate: the inverse DFT is equal to one period of the shifted periodic extension of the sequence

\[ y[n_1, n_2] = x[n_1 - m_1, n_2 - m_2] R_{N_1 \times N_2} [n_1, n_2] \]

- in summary, the new sequence is obtained by making the original periodic, shifting, and taking the fundamental period
Example

- note that *what leaves on one end, enters on the other*
Example

• for this reason it is called a circular shift

• note that this is way more complicated than in 1D
• to get it right we really have to think in terms of the periodic extension of the sequence
• we will see that this shows up in the properties of the DFT,
• namely that convolution becomes circular convolution
Any questions?