Chapter DFT

2D DFT

Contents

(Related to Ch. 3 of Lim.)

Introduction to Discrete Transforms

This continues our "EECS 451 in 2D" coverage. See [1, Ch. 3] and [2].

Overview
- DS orthogonal representation
- DFS, properties, circular convolution
- DFT, properties, circular convolution
- sampling the DSFT, spatial aliasing
- matrix representation
- DCT, properties
- FFT
- two FFT's for the price of one, etc.

The DFT is what we often *compute* because we can do so quickly via an FFT. But often we are really interested in something else, like the FT, or linear convolution, and we must "make do" with the DFT.

2D discrete-space orthogonal representation

For discrete-space signals that are either
- periodic with period (N, M),
- or finite-extent (nonzero only for $n = 0, \dots, N-1$ and $m = 0, \dots, M-1$),

the following **inner product** is appropriate:

$$\langle f,\, g \rangle = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n,m]\, g^*[n,m] \,.$$

There are many **orthogonal bases** with respect to this inner product.

Example. $\phi_{k,l}[n,m] = \delta_2[n-k, m-l]$ for $k = 0, \dots, N-1$ and $l = 0, \dots, M-1$ is a trivial orthonormal basis.

This basis does not provide any new information about the signal. Complex exponential signals are a desirable choice of basis because they are eigenfunctions of LSI systems.

The Fourier basis is as follows. We choose to scale by $1/NM$ for agreement with the DFT convention:

$$\phi_{k,l}[n,m] = \frac{1}{NM} \mathrm{e}^{\imath 2\pi (kn/N + lm/M)} = \frac{1}{NM} \mathrm{e}^{\imath \frac{2\pi}{N} kn} \mathrm{e}^{\imath \frac{2\pi}{M} lm} \,.$$

This is a separable 2D basis. We should verify that the members of this set of signals are orthogonal:

$$\begin{aligned} \langle \phi_{k,l},\, \phi_{k'l'} \rangle &= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \frac{1}{NM} \phi_{k,l}[n,m] \frac{1}{NM} \phi^*_{k'l'}[n,m] \\ &= \frac{1}{(NM)^2} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \mathrm{e}^{\imath \frac{2\pi}{N}(k-k')n} \mathrm{e}^{\imath \frac{2\pi}{M}(l-l')m} = \frac{1}{NM} \delta[[k-k', l-l']]_{(N,M)} \end{aligned} \qquad \text{(DFT-1)}$$

where we have introduced the shorthand

$$\delta[[n,m]]_{(N,M)} \triangleq \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta_2[n - kN, m - lM] \,.$$

The entire set $\{\phi_{k,l}\}$ for $k, l \in \mathbb{Z}$ is *not* an orthogonal basis, but the restricted set $\{\phi_{k,l} \,\colon\, k = 0, \dots, N-1, l = 0, \dots, M-1\}$ is an orthogonal basis, with each $\phi_{k,l}$ having the same energy:

$$E_{k,l} = \|\phi_{k,l}\|^2 = \langle \phi_{k,l},\, \phi_{k,l} \rangle = \frac{1}{NM}.$$

2D discrete Fourier series

Given a periodic signal $\tilde{x}[n,m]$ with period (N, M), we would like to find its 2D **discrete Fourier series (DFS)** representation:

$$\tilde{x}[n,m] = \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{X}[k,l]\, \phi_{k,l}[n,m] = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{X}[k,l]\, \mathrm{e}^{\imath 2\pi (kn/N + lm/M)} \,.$$

That is, we want to represent $\tilde{x}[n,m]$ as a sum of harmonically related complex exponentials having frequencies

$$(0,0),\ (0, \frac{2\pi}{N}),\ \dots,\ \left(\frac{2\pi}{N}(N-1), \frac{2\pi}{M}(M-1) \right). \qquad \text{(DFT-2)}$$

For similarity to the DFT notation, we use $\tilde{X}[k,l]$ to denote the (k,l)th **DFS coefficient**, *i.e.*, the value that multiplies the exponential having frequency $(\frac{2\pi}{N}k, \frac{2\pi}{M}l)$,

From the earlier discussion of general orthogonal representations of signals, the DFS coefficients are given by

$$\tilde{X}[k,l] = \frac{\langle \tilde{x}[\cdot],\, \phi_{k,l} \rangle}{E_{k,l}} = NM \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \tilde{x}[n,m] \frac{1}{NM} \mathrm{e}^{-\imath 2\pi (kn/N + lm/M)} \,.$$

Summarizing then, for a periodic signal $\tilde{x}[n,m]$ with period (N,M), we have the following representation:

$$\boxed{\tilde{X}[k,l] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \tilde{x}[n,m] \, \mathrm{e}^{-\imath 2\pi(kn/N + lm/M)}},$$

$$\boxed{\tilde{x}[n,m] = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{X}[k,l] \, \mathrm{e}^{\imath 2\pi(kn/N + lm/M)}}.$$

These are the **analysis** and **synthesis** formulas, which as mentioned before, represent $\tilde{x}[n,m]$ as a linear combination of NM harmonically related complex exponential signals.

Exact equalities hold in both the analysis and synthesis formulas (as long as all signal values are finite). There are no questions of convergence here.

Due to the periodicity of $\tilde{x}[n,m]$ and the complex exponentials, we could equally well evaluate the analysis formula by summing over *any* $N \times M$ rectangular region. That is for any integers n_o, m_o,

$$\tilde{X}[k,l] = \sum_{n=n_o}^{n_o+N-1} \sum_{m=m_o}^{m_o+M-1} \tilde{x}[n,m] \, \mathrm{e}^{-\imath 2\pi(kn/N + lm/M)}.$$

Although the synthesis formula above represents $\tilde{x}[n,m]$ in terms of coefficients $\tilde{X}[k,l]$, $k = 0, \ldots, N-1$, $l = 0, \ldots, M-1$, and corresponding complex exponentials having frequencies (DFT-2), one could equally well represent $\tilde{x}[n,m]$ in terms of complex coefficients and complex exponentials in any $N \times M$ "box" in frequency space. That is, for any integers k_o, l_o, one could compute $\tilde{x}[n,m]$ by summing the products of the coefficients $\tilde{X}[k,l]$, $k = k_o, \ldots, k_o+N-1$, $l = l_o, \ldots, l_o+M-1$ and their corresponding exponentials. For this reason, we consider the DFS coefficients $\tilde{X}[k,l]$ to be defined for *all* $k,l \in \mathbb{Z}$. Note, however, that they form a periodic image with period (N,M), because $\tilde{X}[k+N,l] = \tilde{X}[k,l]$ and $\tilde{X}[k,l+M] = \tilde{X}[k,l]$. In contrast, the DFT, to be defined later, produces only a finite number of coefficients.

Because equalities hold in both the analysis and synthesis formulas, we can say there is a one-to-one correspondence between periodic signals with period (N,M) and periodic sequences of Fourier coefficients with period (N,M).

Example. Consider the following signal

$$\begin{aligned} \tilde{x}[n,m] &= 4\cos(\pi n/3)\cos(\pi m/4) = \left[\mathrm{e}^{\imath 2\pi n/6} + \mathrm{e}^{-\imath 2\pi n/6}\right]\left[\mathrm{e}^{\imath 2\pi m/8} + \mathrm{e}^{-\imath 2\pi m/8}\right] \\ &= \mathrm{e}^{\imath 2\pi(n/6+m/8)} + \mathrm{e}^{\imath 2\pi(n/6-m/8)} + \mathrm{e}^{\imath 2\pi(-n/6+m/8)} + \mathrm{e}^{\imath 2\pi(-n/6-m/8)}. \end{aligned}$$

What is the period? **??**

In this case, we need not apply the formula for $\tilde{X}[k,l]$ above, since we have directly represented $\tilde{x}[n,m]$ as a sum of complex exponentials. The coefficients in this sum are the $\tilde{X}[k,l]$'s:

$$\begin{aligned} \tilde{X}[k,l] &= \begin{cases} 48, & k \bmod 6 = \pm 1, l \bmod 8 = \pm 1 \\ 0, & \text{otherwise} \end{cases} \\ &= 48\left(\delta_2[[k-1,l-1]]_{(6,8)} + \delta_2[[k+1,l-1]]_{(6,8)} + \delta_2[[k-1,l+1]]_{(6,8)} + \delta_2[[k+1,l+1]]_{(6,8)}\right). \end{aligned}$$

(Because of the one-to-one correspondence mentioned previously, there can be one and only one representation of the signal in terms of complex exponentials signals having the given range of harmonic frequencies. Therefore, the above $\tilde{X}[k,l]$'s are the one and only possible $\tilde{X}[k,l]$'s for this representation.

Example. What signal has constant coefficients? *i.e.*, $\tilde{X}[k,l] = 1$. Presumably some sort of impulse-like signal. But we are working with periodic signals, so it must be a periodic impulse-like signal. In fact it is a 2D "impulse train:"

$$\tilde{x}[n,m] = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \mathrm{e}^{\imath 2\pi(kn/N + lm/M)} = \delta[[n,m]]_{(N,M)} = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \delta_2[n-kN, m-lM].$$

(Picture)

This signal is useful for deriving properties of DFS.

Properties of the DFS

Most properties are analogous to those of the 2D CS FS, except the **scaling property** is absent, since scaling changes the period. (Presumably there is some such property, but it probably differs significantly from the usual scaling properties.)

- **linearity** if $\tilde{x}[n,m]$ and $\tilde{y}[n,m]$ have the same period:

$$\alpha\,\tilde{x}[n,m] + \beta\,\tilde{y}[n,m] \overset{\text{DFS}}{\longleftrightarrow} \alpha\,\tilde{X}[k,l] + \beta\,\tilde{Y}[k,l]$$

- **separability**

$$\tilde{x}[n,m] = \tilde{x}_1[n]\,\tilde{x}_2[m] \overset{\text{DFS}}{\longleftrightarrow} \tilde{X}_1[k]\,\tilde{X}_2[l],$$

  where $\tilde{X}[k]$ denotes the usual 1D DFS.

- **shift**

$$\tilde{x}[n-n_0, m-m_0] \overset{\text{DFS}}{\longleftrightarrow} \mathrm{e}^{-\imath 2\pi(kn_0/N + lm_0/M)}\,\tilde{X}[k,l]$$

- **Average value** (caution!)

$$\frac{1}{NM}\tilde{X}[0,0] = \frac{1}{NM}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\tilde{x}[n,m]$$

- **Parseval's theorem**
  In general, if $f = \sum_{k=-\infty}^{\infty} c_k\phi_k$ and $g = \sum_{k=-\infty}^{\infty} d_k\phi_k$ where the ϕ_k's are orthogonal, then $\langle f,\, g\rangle = \sum_{k=-\infty}^{\infty}\mathcal{E}_k c_k d_k^*$. Thus, for the DFS:

$$\sum_{n=0}^{N-1}\sum_{m=0}^{M-1}\tilde{x}[n,m]\,\tilde{y}^*[n,m] = \frac{1}{NM}\sum_{k=0}^{N-1}\sum_{l=0}^{M-1}\tilde{X}[k,l]\,\tilde{Y}^*[k,l]$$

- **Symmetry properties**

$$\tilde{x}^*[n,m] \overset{\text{DFS}}{\longleftrightarrow} \tilde{X}^*[-k,-l]$$

  If $\tilde{x}[n,m]$ is real, then $\tilde{X}[k,l]$ is Hermitian symmetric. And vice-versa.

- **Duality** ***(Homework)***

$$\text{If } \tilde{x}[n,m] \overset{\text{DFS}}{\longleftrightarrow} \tilde{X}[k,l], \text{ then } \tilde{X}[n,m] \overset{\text{DFS}}{\longleftrightarrow} \boxed{\textbf{??}} \;\&\; \tilde{X}^*[n,m] \overset{\text{DFS}}{\longleftrightarrow} \boxed{\textbf{??}}$$

- **Change of period**
  If a signal $\tilde{x}[n,m]$ is periodic with period (N,M), then it is also periodic with period (n_0N, m_0M) for any positive integers n_0, m_0. Therefore, it can also be represented with a DFS in terms of $n_0N \times m_0M$ complex exponentials with frequencies $(0,0), (0, 2\pi/(n_0N)), \ldots, \left(\frac{2\pi}{n_0N}(n_0N-1), \frac{2\pi}{m_0M}(m_0M-1)\right)$. With $\tilde{X}[k,l]$ representing the usual DFS coefficients, the "new" DFS coefficients are

$$\hat{X}[k,l] = \begin{cases} \tilde{X}[k/n_0, l/m_0], & \text{if } k/n_0 \text{ and } l/m_0 \text{ are integers} \\ 0, & \text{otherwise.} \end{cases}$$

  Thus, the "new" DFS coefficients are just the original ones with zeros in between.

- Pairs of Hermitian DFS terms form sinusoids

$$\tilde{X}[k,l]\,\mathrm{e}^{-2\pi(kn/N+lm/M)} + \tilde{X}[-k,-l]\,\mathrm{e}^{-\imath 2\pi(kn/N+lm/M)} = 2\left|\tilde{X}[k,l]\right|\cos\left(2\pi\left(\frac{kn}{N}+\frac{lm}{M}\right) + \angle\tilde{X}[k,l]\right)$$

"Convolution" property

For the DSFT, we know that $x[n,m] \circledast\circledast h[n,m] \overset{\text{DSFT}}{\longleftrightarrow} X(\omega_{\text{X}},\omega_{\text{Y}})\, H(\omega_{\text{X}},\omega_{\text{Y}})$, which is very useful for filtering without performing convolution. What is the corresponding result for the DFS?

(The corresponding result *cannot* be "$\tilde{x}[n,m] \circledast\circledast \tilde{h}[n,m]$" because linear convolution of two nonzero periodic signals results in infinite or undefined values.)

Suppose we have two periodic signals $\tilde{x}[n,m]$ and $\tilde{h}[n,m]$ with the same period (N,M). If we multiply their DFS coefficients to form $\tilde{Y}[k,l] = \tilde{H}[k,l]\,\tilde{X}[k,l]$ and then compute the inverse DFS to get a signal $\tilde{y}[n,m]$, what is the relationship between $\tilde{y}[n,m]$ and the original $\tilde{x}[n,m]$ and $\tilde{h}[n,m]$? The following derivation shows that the result is periodic convolution, just as in the 1D case:

$$\begin{aligned}
\tilde{y}[n,m] &= \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{Y}[k,l]\, \mathrm{e}^{\imath 2\pi(kn/N + lm/M)} \\
&= \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{H}[k,l]\, \tilde{X}[k,l]\, \mathrm{e}^{\imath 2\pi(kn/N + lm/M)} \\
&= \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \left[\sum_{n'=0}^{N-1} \sum_{m'=0}^{M-1} \tilde{x}[n',m']\, \mathrm{e}^{-\imath 2\pi(kn'/N + lm'/M)} \right] \\
&\qquad \cdot \left[\sum_{n''=0}^{N-1} \sum_{m''=0}^{M-1} \tilde{h}[n'',m'']\, \mathrm{e}^{-\imath 2\pi(kn''/N + lm''/M)} \right] \mathrm{e}^{\imath 2\pi(kn/N + lm/M)} \\
&= \sum_{n'=0}^{N-1} \sum_{m'=0}^{M-1} \sum_{n''=0}^{N-1} \sum_{m''=0}^{M-1} \tilde{x}[n',m']\, \tilde{h}[n'',m'']\, \left[\frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \mathrm{e}^{-\imath 2\pi(k(n'+n''-n)/N + l(m'+m''-m)/M)} \right] \\
&= \sum_{n'=0}^{N-1} \sum_{m'=0}^{M-1} \tilde{x}[n',m'] \left[\sum_{n''=0}^{N-1} \sum_{m''=0}^{M-1} \tilde{h}[n'',m']\, \delta[[n'+n''-n, m'+m''-m]]_{(N,M)} \right] \\
&= \sum_{n'=0}^{N-1} \sum_{m'=0}^{M-1} \tilde{x}[n',m']\, \tilde{h}[n-n', m-m'],
\end{aligned}$$

where the last equality follows from the fact that the sum over n'',m'' has only NM terms, but $\delta[[\cdot]]_{(N,M)}$ is zero for only one of every NM terms, and the periodicity of $\tilde{h}[n,m]$.

Thus we have the **convolution property** of the DFS:

$$\tilde{y}[n,m] = \sum_{n'=0}^{N-1} \sum_{m'=0}^{M-1} \tilde{x}[n',m']\, \tilde{h}[n-n', m-m'] \overset{\text{DFS}}{\longleftrightarrow} \tilde{Y}[k,l] = \tilde{X}[k,l]\, \tilde{H}[k,l]\,.$$

This is called **circular convolution**, since the summations are finite. In ordinary **linear convolution**, the summations are infinite. In these notes, circular convolution is denoted $\tilde{x}[n,m] \bigotimes \tilde{h}[n,m]$.

$$\boxed{\tilde{x}[n,m] \bigotimes \tilde{h}[n,m] \overset{\text{DFS}}{\longleftrightarrow} \tilde{X}[k,l]\, \tilde{H}[k,l]\,.}$$

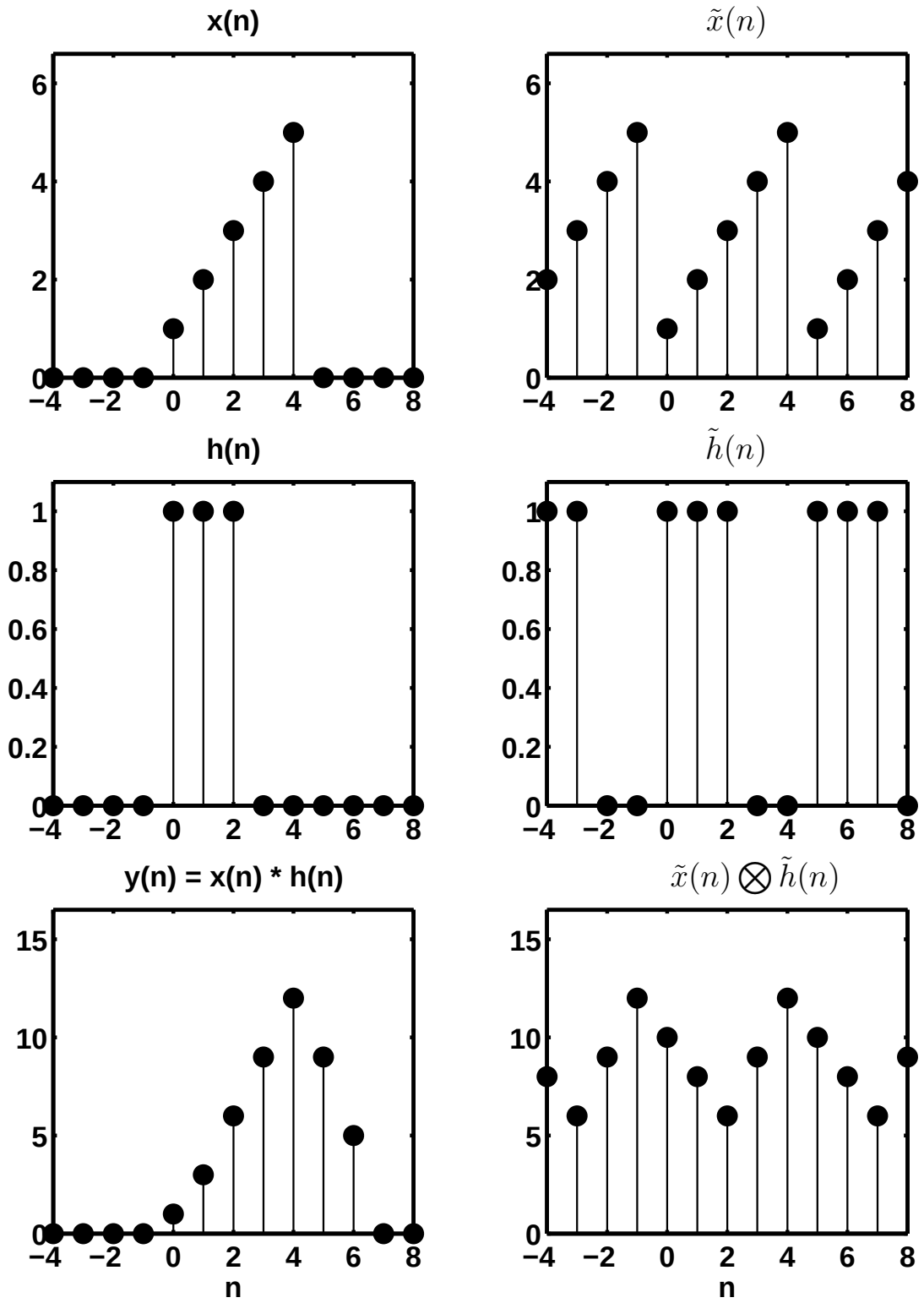
Although it was stated earlier that one could use the DFS for either periodic or finite-support signals, the above property holds only for periodic signals.

Multiplication property

Using duality, one can show

$$\tilde{x}[n,m]\, \tilde{y}[n,m] \overset{\text{DFS}}{\longleftrightarrow} \frac{1}{NM}\, \tilde{X}[k,l] \bigotimes \tilde{Y}[k,l]\,.$$

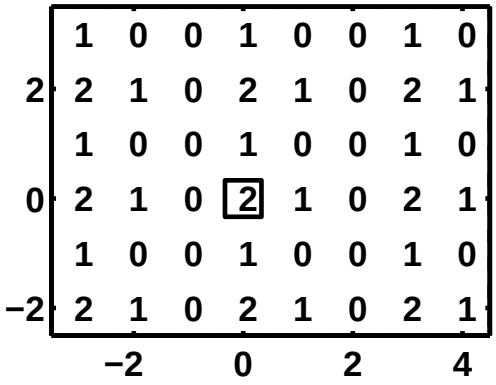
Example. Here is a 1D illustration of linear vs circular convolution.



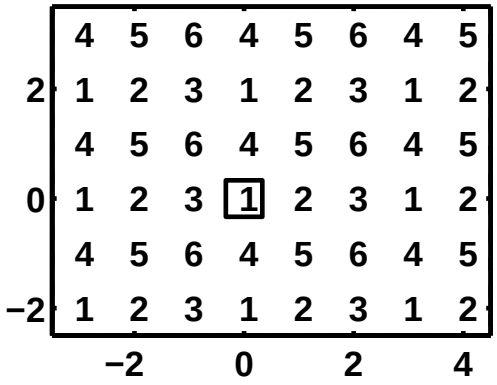
The linear convolution of a 5-point sequence with a 3-point sequence yields a 7-point sequence.

Periodic convolution of two 5-periodic sequences is still 5-periodic.

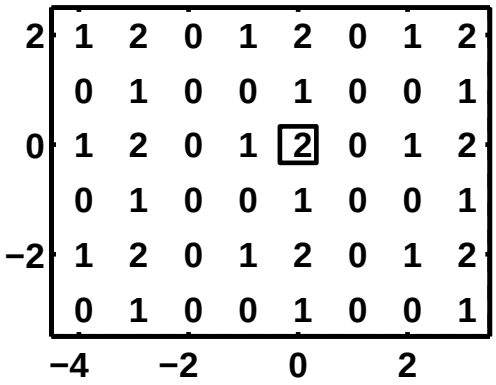
$\tilde{h}[n,m]$



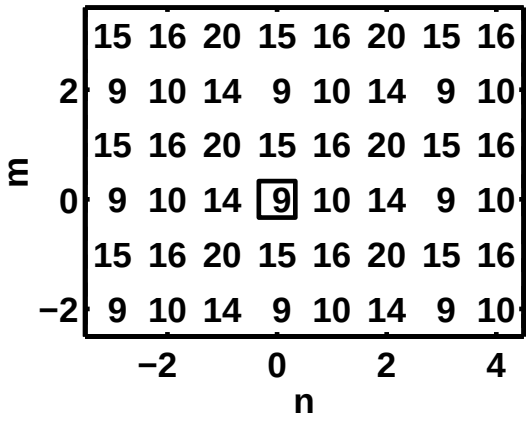
$\tilde{x}[n,m]$



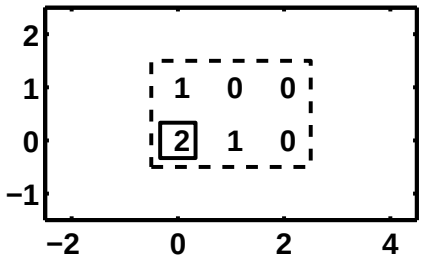
$\tilde{h}[n,m]$ after `rot90`



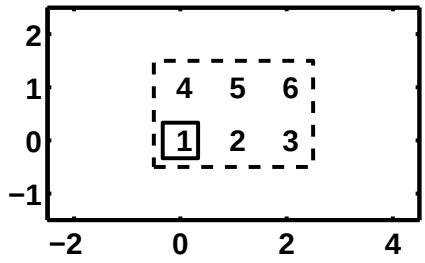
$\tilde{h}[n,m] \circledast \tilde{x}[n,m]$



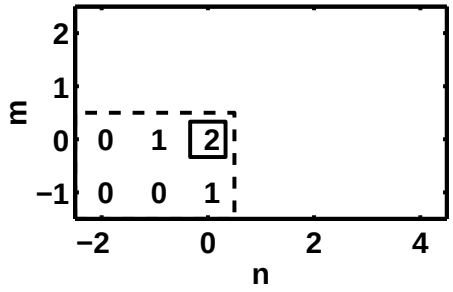
$h[n,m]$



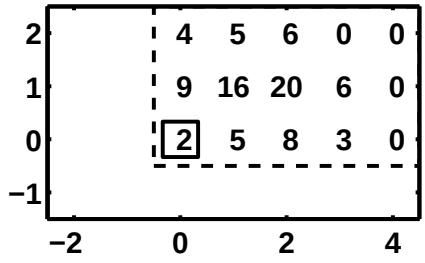
$x[n,m]$



$h[n,m]$ after `rot90`



$h[n,m] \ast\ast x[n,m]$



Discrete Fourier transform (DFT)

Now we switch from the "extreme" of periodic sequences to the other extreme of **finite-support** (also called finite-extent) signals. The connection between the two is that one can form a periodic signal from a finite-support signal by replication, and one can "extract" a finite-support signal from a periodic signal by a rectangular window.

Given a 2D DS signal $x[n,m]$, we can form periodic signals from $x[n,m]$ in two distinct ways.

- One way is called the (N,M)**-point circular extension** of $x[n,m]$:

$$\tilde{x}[n,m] \triangleq x[n \bmod N,\, m \bmod M]\,. \tag{DFT-3}$$

  Notice that this **circular extension** depends only on the values of $x[n,m]$ for $n = 0,\ldots,N-1$ and $m = 0,\ldots,M-1$, regardless of what the support of $x[n,m]$ is.

- Another way is called the (N,M)**-point periodic superposition** of $x[n,m]$:

$$x_{\mathrm{ps}}[n,m] = x[[n,m]]_{(N,M)} = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x[n-kN, m-lM]\,. \tag{DFT-4}$$

  This periodic signal depends on *all* of the values of the original signal $x[n,m]$.

Many books do not distinguish between the above two ways of creating a periodic signal from $x[n,m]$, and refer to one or the other (or both) as simply the "**periodic extension**" of $x[n,m]$. In general the two methods are distinct. However, there is a very important "special case" where the two methods yield identical signals. If $x[n,m]$ is a $N \times M$ **finite-support** signal, meaning (unless otherwise specified) $x[n,m]$ nonzero only for $n = 0,\ldots,N-1$ and $m = 0,\ldots,M-1$, then $\tilde{x}[n,m]$ and $x_{\mathrm{ps}}[n,m]$ are identical. So in this case the generic term "periodic extension" is unambiguous.

Furthermore, in this finite-support case, we can "extract" the original finite-support signal $x[n,m]$ from the periodic signal $\tilde{x}[n,m]$ (or from $x_{\mathrm{ps}}[n,m]$) by applying a rectangular window in the space domain:

$$x[n,m] = \tilde{x}[n,m]\, R_{NM}[n,m] \text{ where } R_{NM}[n,m] \triangleq \begin{cases} 1, & n = 0,\ldots,N-1, m = 0,\ldots,M-1 \\ 0, & \text{otherwise.} \end{cases} \tag{DFT-5}$$

Focusing for now on the finite-support case, what is the DFS of the periodic extension signal $\tilde{x}[n,m]$? Applying the analysis formula:

$$\tilde{X}[k,l] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \tilde{x}[n,m]\, \mathrm{e}^{-\imath 2\pi(kn/N + lm/M)} = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m]\, \mathrm{e}^{-\imath 2\pi(kn/N + lm/M)}\,.$$

(The equality follows from the limits on the sum.) The DFS coefficients are *defined and generally nonzero* for all $k,l \in \mathbb{Z}$.

We can truncate this periodic set of coefficients to form a finite-extent set, called the **discrete Fourier transform** (**DFT**) of $x[n,m]$:

$$\begin{aligned} X[k,l] &\triangleq \tilde{X}[k,l]\, R_{NM}[k,l] \\ &= \begin{cases} \tilde{X}[k,l], & k = 0,\ldots,N-1,\ l = 0,\ldots,M-1 \\ 0, & \text{otherwise.} \end{cases} \\ &= \begin{cases} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m]\, \mathrm{e}^{-\imath 2\pi(kn/N + lm/M)}\,, & k = 0,\ldots,N-1, l = 0,\ldots,M-1 \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

- For our purposes, it will be safest to consider $X[k,l]$ to be defined only for $k = 0,\ldots,N-1$ and $l = 0,\ldots,M-1$.
- Alternatively, one can consider $X[k,l]$ to be *defined* for all $k,l \in \mathbb{Z}$, but to be *nonzero* only for $k = 0,\ldots,N-1$ and $l = 0,\ldots,M-1$.
- Alternatively, some texts choose to define $X[k,l]$ to be a periodic function of k and l, just like $\tilde{X}[k,l]$ is.
  The disadvantage of these latter two conventions is that tools for computing DFTs only work with finite arrays. Trying to evaluate $X[-k,l]$ will cause a program fault usually. To avoid such problems, it is safest to have the theory match practice as closely as possible, so we usually choose $X[k,l]$ to be *undefined* except when $k = 0,\ldots,N-1$ and $l = 0,\ldots,M-1$.

Note the similarity in notation between $X[k,l]$ and $X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}})$. Which transform is meant should always be obvious from context.

Of course we can always "recover" the entire periodic sequence $\tilde{X}[k,l]$ from the DFT by applying a (N,M)-point circular extension as follows:

$$\tilde{X}[k,l] = X[k \bmod N, l \bmod M] \,.$$

Since the inverse DFS formula depends on $\tilde{X}[k,l]$ only for $n = 0, \ldots, N-1$ and $m = 0, \ldots, M-1$, clearly we can recover $x[n,m]$ from $X[k,l]$:

$$\begin{aligned} x[n,m] &= \tilde{x}[n,m]\, R_{NM}[n,m] \\ &= \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{X}[k,l]\, \mathrm{e}^{\imath 2\pi (kn/N + lm/M)}\, R_{NM}[n,m] \\ &= \begin{cases} \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X[k,l]\, \mathrm{e}^{\imath 2\pi (kn/N + lm/M)}, & n = 0, \ldots, N-1, m = 0, \ldots, M-1 \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

This is the **inverse DFT (iDFT)** formula in 2D.

In summary then, the DFT/iDFT pair are given as follows.

$$X[k,l] = \begin{cases} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m]\, \mathrm{e}^{-\imath 2\pi (kn/N + lm/M)}, & k = 0, \ldots, N-1, l = 0, \ldots, M-1 \\ ?, & \text{otherwise,} \end{cases} \quad \text{(DFT-6)}$$

$$x[n,m] = \begin{cases} \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X[k,l]\, \mathrm{e}^{\imath 2\pi (kn/N + lm/M)}, & n = 0, \ldots, N-1, m = 0, \ldots, M-1 \\ ?, & \text{otherwise,} \end{cases} \,. \quad \text{(DFT-7)}$$

where the "?" depends on interpretation. (It may be undefined, or zero, or the circular extension.)

Relationship between DFT and DSFT

For a finite-support signal $x[n,m]$, we can relate the DFT to the DSFT as follows:

$$\begin{aligned} X[k,l] &= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m]\, \mathrm{e}^{-\imath 2\pi (kn/N + lm/M)} \\ &= \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} x[n,m]\, \mathrm{e}^{-\imath 2\pi (kn/N + lm/M)} \\ &= X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}})\big|_{\omega_{\mathrm{X}} = 2\pi k/N,\, \omega_{\mathrm{Y}} = 2\pi l/M} = X\left(\frac{2\pi}{N} k,\, \frac{2\pi}{M} l\right), \qquad k = 0, \ldots, N-1, l = 0, \ldots, M-1. \end{aligned}$$

Provided $x[n,m]$ has finite support, we can also write:

$$\tilde{X}[k,l] = X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \Big|_{\omega_{\mathrm{X}} = 2\pi k/N,\, \omega_{\mathrm{Y}} = 2\pi l/M} = X\left(\frac{2\pi}{N} k,\, \frac{2\pi}{M} l\right). \quad \text{(DFT-8)}$$

Caution: recall that if $g[n,m] = g_a(n\Delta_{\mathrm{X}}, m\Delta_{\mathrm{Y}})$, then

$$G(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) = \frac{1}{\Delta_{\mathrm{X}} \Delta_{\mathrm{Y}}} \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} G_a\left(\frac{\omega_{\mathrm{X}}/2\pi - k}{\Delta_{\mathrm{X}}} \frac{\omega_{\mathrm{Y}}/2\pi - l}{\Delta_{\mathrm{Y}}}\right).$$

This holds for any sampling rate regardless of whether $g_a(x,y)$ is band-limited. But if $g_a(x,y)$ were band-limited, then the central replicate of $G(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}})$ would tell us everything about the original analog signal spectrum. And if (DFT-8) also held, we could relate the analog spectrum to the DFT coefficients. But (DFT-8) requires a finite-support signal, which is (in general) incompatible with the analog signal being band-limited.

Find a band-limited analog signal that, when critically sampled, is a finite-support signal. **??**

What happens though if you shift that analog signal slightly? **??**

Properties of the DFT

The following properties are all written so that only the values of $x[n,m]$ for $n = 0, \ldots, N-1$ and $m = 0, \ldots, M-1$ are involved, as well as only the values of $X[k,l]$ for $k = 0, \ldots, N-1$ and $l = 0, \ldots, M-1$.

Missing properties (relative to CS FS): scaling, since scaling changes the extent. (Presumably there is such a property, but it probably differs significantly from the usual scaling properties.)

Since the DFT is defined in terms of the DFS, it inherits many of its properties. However, there are some differences too, due to the finite-extent nature of both $x[n,m]$ and $X[k,l]$.

Properties that are identical to those of the DFS

- **linearity** if $x[n,m]$ and $y[n,m]$ have the same period:

$$\alpha\, x[n,m] + \beta\, y[n,m] \overset{\text{DFT}}{\longleftrightarrow} \alpha\, X[k,l] + \beta\, Y[k,l]$$

- **separability**

$$x[n,m] = x_1[n]\, x_2[m] \overset{\text{DFT}}{\longleftrightarrow} X_1[k]\, X_2[l],$$

  where $X[k]$ denotes the usual 1D DFT.

- **average value** (caution!)

$$\frac{1}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m] = \frac{1}{NM}\, X[0,0]$$

- **Parseval's theorem**

$$\sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m]\, y^*[n,m] = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X[k,l]\, Y^*[k,l]$$

Properties that differ somewhat from those of the DFS

Many of these properties depend on $x[n \bmod N,\ m \bmod M]$, the (N,M)-point circular extension of a signal, as defined in (DFT-3), or the (N,M)-point circular extension of the DFT coefficients $X[k,l]$.

- **complex conjugate**

$$x^*[n,m] \overset{\text{DFT}}{\longleftrightarrow} X^*[-k \bmod N,\ -l \bmod M]$$

- **duality**

$$x[n,m] \overset{\text{DFT}}{\longleftrightarrow} X[k,l] \implies \begin{array}{ll} X[n,m] & \overset{\text{DFT}}{\longleftrightarrow} \quad NM\, x[-k \bmod N,\ -l \bmod M] \\ X^*[n,m] & \overset{\text{DFT}}{\longleftrightarrow} \quad NM\, x^*[k,l] \end{array}$$

- **circular shift / complex modulation**

  A circular shift of a 1D signal by n_0, transforms $\{x[0], \ldots, x[N-1]\}$ into $\{x[n_0], \ldots, x[N-1], x[0], \ldots, x[n_0-1]\}$, which can be written $x[(n-n_0) \bmod N]$.

  For a 2D signal, the circular shift by n_0, m_0 and its (N,M)-point DFT are

$$x[(n-n_0) \bmod N,\ (m-m_0) \bmod M] \overset{\text{DFT}}{\longleftrightarrow} \mathrm{e}^{-\imath 2\pi(kn_0/N + lm_0/M)}\, X[k,l]\,.$$

- **circular frequency shift**

$$\mathrm{e}^{\imath 2\pi(k_0 n/N + l_0 m/M)}\, x[n,m] \overset{\text{DFT}}{\longleftrightarrow} X[(k-k_0) \bmod N,\ (l-l_0) \bmod M]$$

- **Circular space reversal**

$$x[-n \bmod N,\ -m \bmod M] \overset{\text{DFT}}{\longleftrightarrow} X[-k \bmod N,\ -l \bmod M]$$

For the DSFT, we know that if $x[n,m] \overset{\text{DSFT}}{\longleftrightarrow} X(\omega_{\text{X}}, \omega_{\text{Y}})$, then $x[-n,-m] \overset{\text{DSFT}}{\longleftrightarrow} X(-\omega_{\text{X}}, -\omega_{\text{Y}})$.
What is the corresponding formula for the DFT?
The **circular space reversal** of a 2D signal $x[n,m]$ is given by $x[-n \bmod N,\ -m \bmod M]$.

Example. If $x[n,m] = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ \underline{9} & 10 & 11 & 12 \end{bmatrix}$, then $x[-n \bmod 4, -m \bmod 3] = \begin{bmatrix} 5 & 8 & 7 & 6 \\ 1 & 4 & 3 & 2 \\ \underline{9} & 12 & 11 & 10 \end{bmatrix}$.

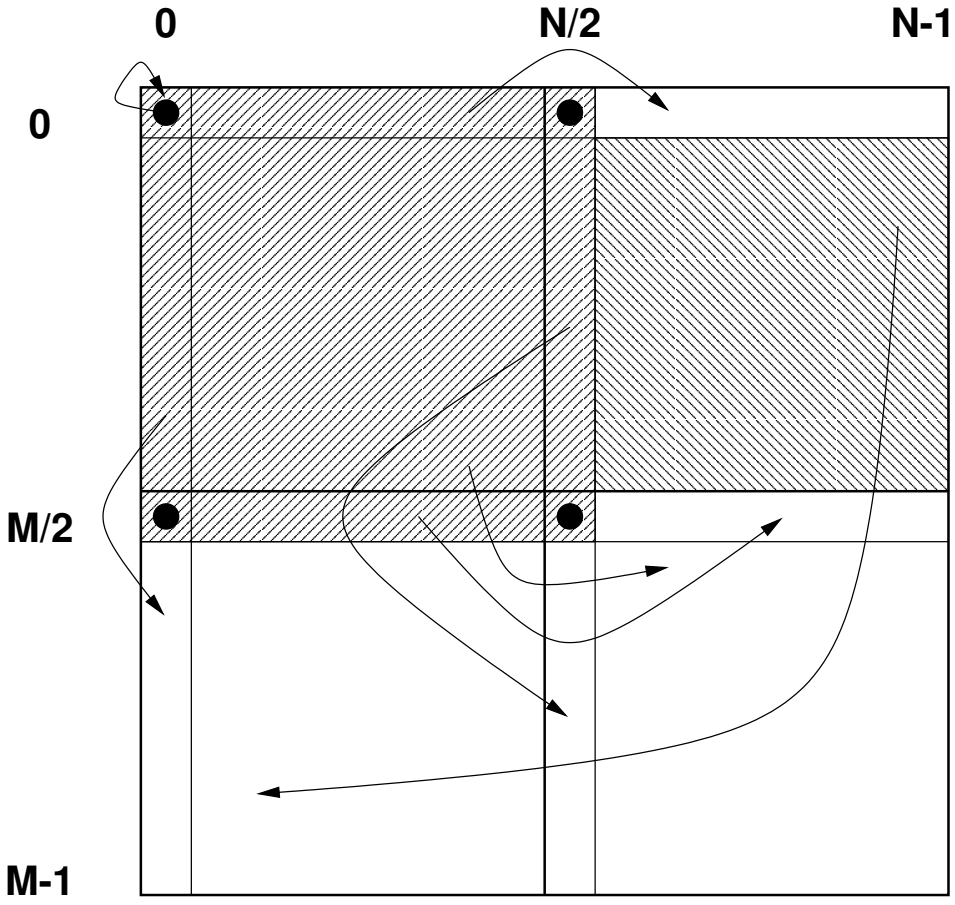
- **Symmetry properties**

$$x^*[n,m] \overset{\text{DFT}}{\longleftrightarrow} X^*[-k \bmod N,\ -l \bmod M]$$

If $x[n,m]$ is **circularly even**, *i.e.,* if $x[-n \bmod N,\ -m \bmod M] = x[n,m]$, then $X[k,l]$ is also **circularly even**.
If $x[n,m]$ is real, then $X[k,l]$ is Hermitian **circularly symmetric**, *i.e.,* $X[-k \bmod N,\ -l \bmod M] = X^*[k,l]$.
And vice-versa.

If $x[n,m]$ is real, the formula $X[-k \bmod N,\ -l \bmod M] = X^*[k,l]$ may not convey visually the symmetry properties of the DFT coefficients. If we arrange the DFT coefficients as a $N \times M$ array, the following diagram illustrates which coefficients are complex conjugates of which other coefficients. The shaded area represents a set of "sufficient" coefficients; when storage is at a premium, for real signals it suffices to store the values in the shaded region.

The points indicated by the dots are values that are their own complex conjugates, and hence must be real.



Proof of complex conjugate property

Here is a proof of the complex conjugate property for 1D signals, to provide an example of how one performs such proofs.

By the synthesis property:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] \, \mathrm{e}^{\imath 2\pi nk/N} \,,$$

so taking the complex conjugate of both sides yields

$$\begin{aligned}
x^*[n] &= \frac{1}{N} \sum_{k=0}^{N-1} X^*[k] \, \mathrm{e}^{-\imath 2\pi nk/N} \\
&= \frac{1}{N} X[0] + \frac{1}{N} \sum_{k=1}^{N-1} X^*[k] \, \mathrm{e}^{-\imath 2\pi nk/N} \\
&\quad \text{let } l = N - k\text{:} \\
&= \frac{1}{N} X[0] + \frac{1}{N} \sum_{l=1}^{N-1} X^*[N-l] \, \mathrm{e}^{-\imath 2\pi n(N-l)/N} \\
&= \frac{1}{N} X[0 \bmod N] + \frac{1}{N} \sum_{l=1}^{N-1} X^*[N-l] \, \mathrm{e}^{\imath 2\pi nl/N} \\
&= \frac{1}{N} X[0 \bmod N] + \frac{1}{N} \sum_{l=1}^{N-1} X^*[-l \bmod N] \, \mathrm{e}^{\imath 2\pi nl/N} \\
&= \frac{1}{N} \sum_{l=0}^{N-1} X^*[-l \bmod N] \, \mathrm{e}^{\imath 2\pi nl/N} \,.
\end{aligned}$$

Since this is the DFT synthesis expression for $x^*[n]$, we have shown the following property:

$$x^*[n] \overset{\mathrm{DFT}}{\longleftrightarrow} X^*[-k \bmod n] \,.$$

Convolution property

One of the main uses of the DFT is to implement fast convolution via the FFT. So the convolution property is particularly important. In fact, this property is probably the primary reason why the DFT is used so much more frequently than the many alternative orthogonal transforms.

What happens if we take two sets of (N, M)-point DFT coefficients, multiply them, and take the inverse DFT? Clearly the answer cannot in general be the linear convolution, since linear convolution of two signals results in a signal with larger support.

Derivation:

Suppose $h[n,m]$ and $x[n,m]$ are both finite-support signals, with corresponding DFT coefficients $H[k,l]$ and $X[k,l]$.

Define $Y[k,l] = H[k,l]\,X[k,l]$, and let $y[n,m]$ be the inverse (N,M)-point DFT of $Y[k,l]$. Then by the above construction,

$$y[n,m] = \tilde{y}[n,m]\, R_{NM}[n,m],$$

where $R_{NM}[n,m]$ was defined in (DFT-5), and where

$$\tilde{y}[n,m] \overset{\text{DFS}}{\longleftrightarrow} \tilde{Y}[k,l] = \widetilde{H[k,l]\,X}[k,l] = \tilde{H}[k,l]\,\tilde{X}[k,l],$$

where $\tilde{H}[k,l]$ and $\tilde{X}[k,l]$ are the periodic extensions of $H[k,l]$ and $X[k,l]$ respectively.

It follows that $\tilde{y}[n,m] = \tilde{h}[n,m] \bigotimes \tilde{x}[n,m]$, and so

$$\begin{aligned} y[n,m] &= (\tilde{h}[n,m] \bigotimes \tilde{x}[n,m])\, R_{NM}[n,m] \\ &= \begin{cases} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \tilde{h}[k,l]\,\tilde{x}[n-k,m-l], & n=0,\ldots,N-1, m=0,\ldots,M-1 \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

This is a correct but incomplete answer since we want a direct relationship between $y[n,m]$ and the original signals $h[n,m]$ and $x[n,m]$. Considering that the summation limits extend only from 0 to $N-1$ and 0 to $M-1$, we can write

$$\begin{aligned} y[n,m] &= \begin{cases} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} h[k,l]\, x[(n-k) \bmod N, (m-l) \bmod M], & n=0,\ldots,N-1, m=0,\ldots,M-1 \\ 0, & \text{otherwise} \end{cases} \\ &\triangleq x[n,m] \bigotimes h[n,m]\,. \end{aligned}$$

Note that for finite-support signals we have a *slightly* different definition of circular convolution than for periodic signals. In words, if we start with two finite-support signals, multiplying their DFT coefficients is equivalent to: first forming their periodic extension, then performing circular convolution, then truncating the result.

In summary, one usually expresses the **convolution property** of the DFT as follows:

$$\boxed{h[n,m] \bigotimes x[n,m] \overset{\text{DFT}}{\longleftrightarrow} H[k,l]\,X[k,l],}$$

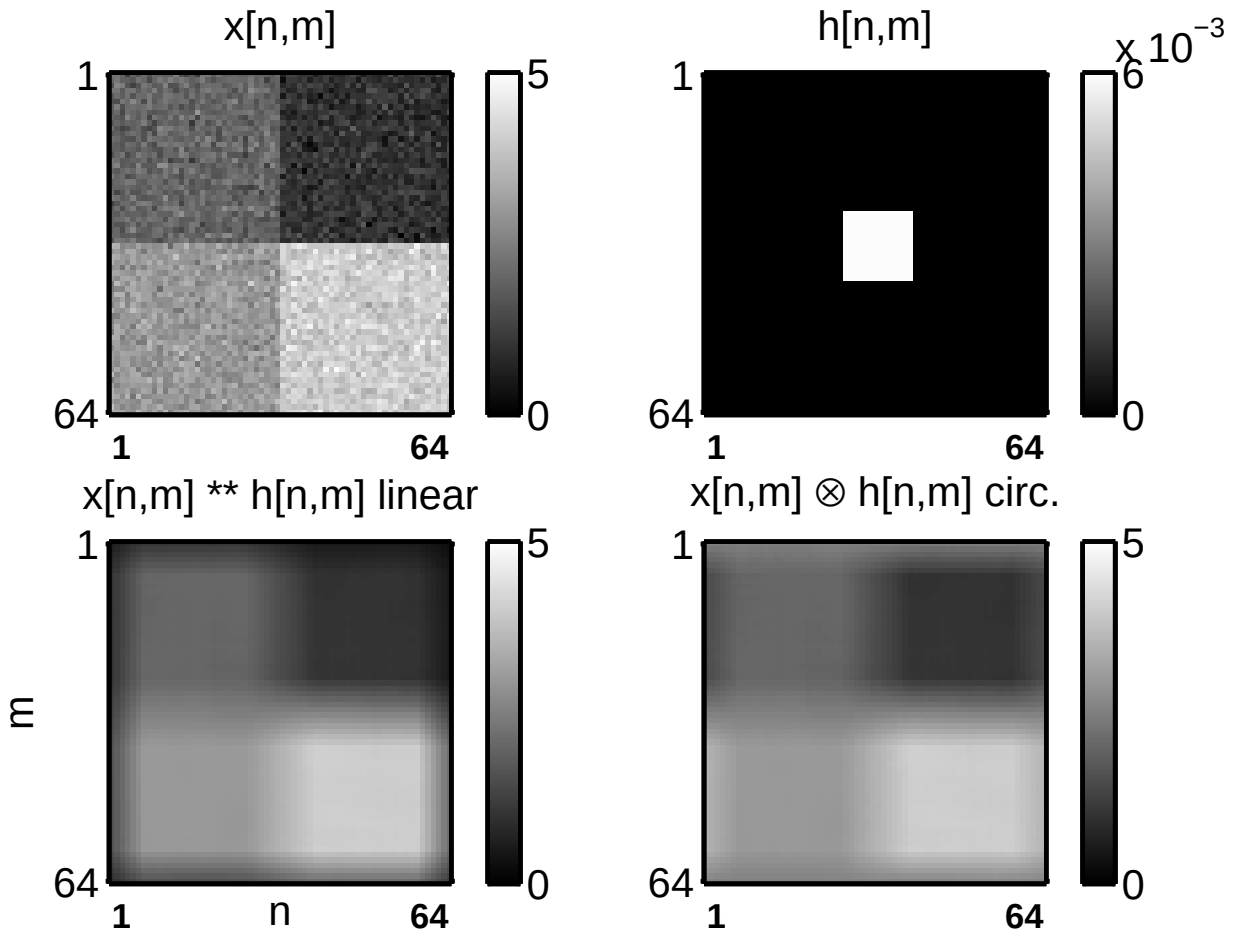
where one must remember the truncation and the slightly modified meaning of **circular convolution**.

Multiplication property

Using duality, one can show

$$x[n,m]\, y[n,m] \overset{\text{DFS}}{\longleftrightarrow} NM\, X[k,l] \bigotimes Y[k,l]\,.$$

Illustration of wrap around effect



Emulating linear convolution

The effect of circular convolution is a **wrap around effect**, where signal values from one border of an image can "wrap around" to affect values on the other side after filtering via a DFT. This effect is generally undesirable.

How can we eliminate this wrap around effect of the DFT? Using **zero padding**.

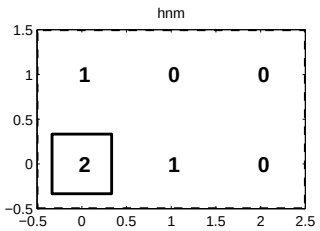
Convolving an $N_1 \times N_2$ image $x[n,m]$ with a $M_1 \times M_2$ filter $h[n,m]$ yields a $L_1 \times L_2$ result $y[n,m]$, where $L_k = N_k + M_k - 1$. Thus, if we **zero pad** both $x[n,m]$ and $h[n,m]$ to a size $L_1 \times L_2$ before taking the DFT, then the final result of `iDFT(DFT(x) .* DFT(h))` will be exactly equivalent to the result of **linear convolution** of $x[n,m]$ with $h[n,m]$. (Of course this only works for finite-support images, but that is always adequate in practice for finite-support filters!)

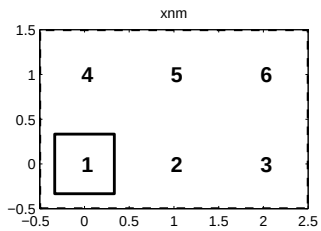
There is a practical inconvenience with zero padding however. Suppose we wish to convolve a 256×256 image with a 17×17 filter. The result will be 272×272. The prime factors of 272 are 2 and 17. The FFT works fastest for small prime factors (especially 2), so a prime factor of 17 is computationally undesirable. One could pad to a 512×512 image, but then only 28% of the final image would be the part we care about (the rest would be zero in exact arithmetic). There are a few choices.

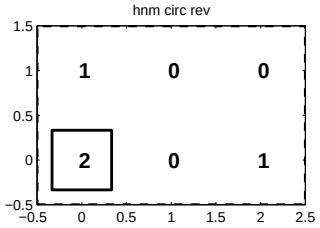
- Abandon the zero padding, accept the wrap around, and ignore the edges of the image.
- Zero pad to a value smaller than 512; *e.g.*, 288 has factors 2 and 3 and is only slightly bigger than 272. MATLAB's `fft2` routine is clever enough to exploit this. Consider the following results of `tic, fft2(rand(N)); toc` on a Pentium II (200MHz).

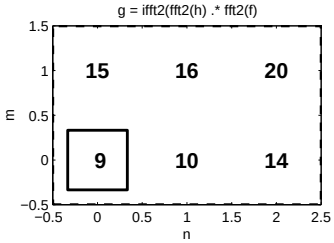
N	time
256	0.32
272	0.45
288	0.37
512	1.25

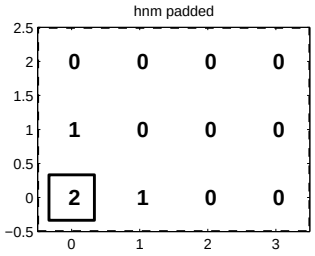
  This approach *still uses circular convolution*, but the zero padding has the effect of yielding the same *results* as linear convolution, *for finite-support signals*.
  Clearly intelligent zero padding can reduce computation considerably over the popular "next higher power of 2" approach.
- Use the **overlap-add method** or **overlap-save method** described below.

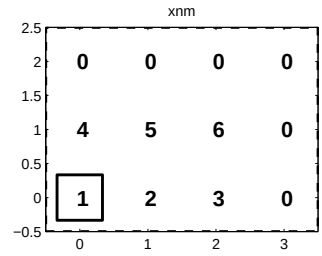

hnm
1 0 0
2 1 0

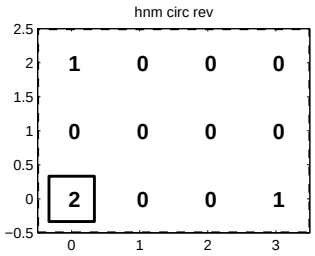

xnm
4 5 6
1 2 3

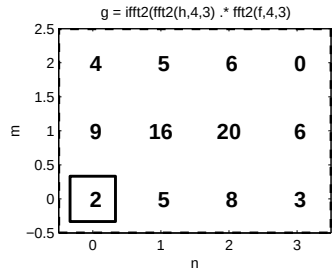

hnm circ rev
1 0 0
2 0 1

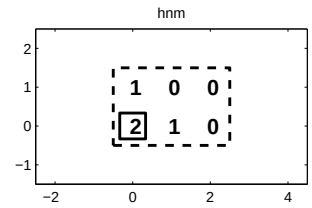

g = ifft2(fft2(h) .* fft2(f)
15 16 20
9 10 14
m
n

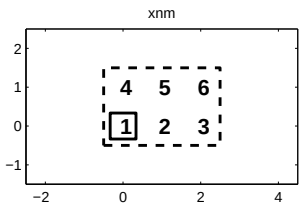

hnm padded
0 0 0 0
1 0 0 0
2 1 0 0

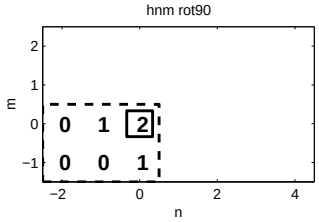

xnm
0 0 0 0
4 5 6 0
1 2 3 0

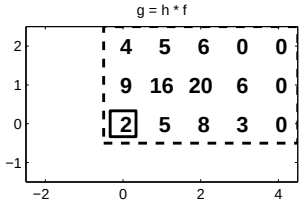

hnm circ rev
1 0 0 0
0 0 0 0
2 0 0 1


g = ifft2(fft2(h,4,3) .* fft2(f,4,3)
4 5 6 0
9 16 20 6
2 5 8 3
m
n


hnm
1 0 0
2 1 0


xnm
4 5 6
1 2 3


hnm rot90
0 1 2
0 0 1
m
n


g = h * f
4 5 6 0 0
9 16 20 6 0
2 5 8 3 0

The overlap-add method

Since convolution is linear, we can decompose convolution of a large image with a small filter into the sum of the convolution of the filter with each of several modest-sized blocks of the image.

Let

$$x[n,m] = \sum_k x_k[n,m] \quad \textbf{\textit{(Picture)}} \,,$$

then

$$\begin{aligned} y[n,m] &= h[n,m] \ast\ast x[n,m] = h[n,m] \ast\ast \sum_k x_k[n,m] \\ &= \sum_k h[n,m] \ast\ast x_k[n,m] \triangleq \sum_k y_k[n,m], \text{ where } y_k[n,m] \triangleq h[n,m] \ast\ast x_k[n,m] \,. \end{aligned}$$

If block is $M_1 \times M_2$ and the filter is $L_1 \times L_2$, then the result of each block convolution will be $(M_1 + L_1 - 1) \times (M_2 + L_2 - 1)$. So the output blocks $y_k[n,m]$ will overlap. Nevertheless, superposition still applies, so we will get the correct final result provided we

- implement linear convolution for each block,
- and properly *add* the resulting overlapping blocks.

The MATLAB command `fftfilt` does this in 1D.
Is there a 2D overlap-add method in* MATLAB*? (The `conv2` routine is a MEX file.)

The **overlap-save method** is fairly similar. See text.

To implement linear convolution for each block, we can either use space-domain convolution (which would be reasonable for small filters, especially if separable), or use zero-padded convolution via the FFT. One should choose the block size such that the zero-padded FFT size factors into small primes.

Sampling the DSFT

We have seen that if $x[n,m]$ is a finite-support signal, then its DFT consists of samples of its DSFT:

$$X[k,l] = X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \Big|_{\omega_{\mathrm{X}}=2\pi k/N,\ \omega_{\mathrm{Y}}=2\pi l/M}. \tag{DFT-9}$$

Often, however, we "take samples" of an analytical expression for the DSFT of some unknown signal, and then compute the iDFT to examine the signal. What happens if the underlying signal does not have finite extent?

In other words, suppose $x[n,m] \overset{\mathrm{DSFT}}{\longleftrightarrow} X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}})$ for *any* DS signal $x[n,m]$, and define

$$Y[k,l] = X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \Big|_{\omega_{\mathrm{X}}=2\pi k/N,\ \omega_{\mathrm{Y}}=2\pi l/M},\ k = 0, \ldots, N-1,\ l = 0, \ldots, M-1$$

for some user-selected values (N, M).
If we then compute the iDFT of $Y[k,l]$ to get some signal $y[n,m]$, how does that $y[n,m]$ relate to the original $x[n,m]$?

$$x[n,m] \to \boxed{\text{DSFT}} \to X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \to \boxed{\text{sample/truncate}} \to Y[k,l] \to \boxed{\text{iDFT}} \to y[n,m]$$

To aid in the analysis, define

$$\begin{aligned}
Y(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) &= \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} Y[k,l]\, \delta_2((\omega_{\mathrm{X}} - 2\pi k/N, \omega_{\mathrm{Y}} - 2\pi l/M))_{(2\pi,2\pi)} \\
&= \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \Big|_{\omega_{\mathrm{X}}=2\pi k/N,\ \omega_{\mathrm{Y}}=2\pi l/M} \delta_2((\omega_{\mathrm{X}} - 2\pi k/N, \omega_{\mathrm{Y}} - 2\pi l/M))_{(2\pi,2\pi)} \\
&= X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \delta_2((\omega_{\mathrm{X}} - 2\pi k/N, \omega_{\mathrm{Y}} - 2\pi l/M))_{(2\pi,2\pi)}.
\end{aligned}$$

Then the output signal is as follows:

$$\begin{aligned}
y[n,m] &= \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} Y[k,l]\, \mathrm{e}^{\imath 2\pi(nk/N + ml/M)} \\
&= \frac{1}{NM} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} Y(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}})\, \mathrm{e}^{\imath(\omega_{\mathrm{X}} n + \omega_{\mathrm{Y}} m)}\, \mathrm{d}\omega_{\mathrm{X}}\, \mathrm{d}\omega_{\mathrm{Y}} \qquad \text{(substitute } Y(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \text{ to verify)} \\
&= \frac{1}{NM} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} \left[X(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \delta_2((\omega_{\mathrm{X}} - 2\pi k/N, \omega_{\mathrm{Y}} - 2\pi l/M))_{(2\pi,2\pi)} \right] \mathrm{e}^{\imath(\omega_{\mathrm{X}} n + \omega_{\mathrm{Y}} m)}\, \mathrm{d}\omega_{\mathrm{X}}\, \mathrm{d}\omega_{\mathrm{Y}} \\
&= x[n,m] \circledast\circledast h[n,m],
\end{aligned}$$

where we see

$$h[n,m] \overset{\mathcal{F}_2}{\longleftrightarrow} \frac{(2\pi)^2}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \delta_2((\omega_{\mathrm{X}} - 2\pi k/N, \omega_{\mathrm{Y}} - 2\pi l/M))_{(2\pi,2\pi)},$$

so by recalling the DSFT of complex exponential signals:

$$h[n,m] = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} \mathrm{e}^{-\imath 2\pi(kn/N + lm/M)} = \delta[[n,m]]_{(N,M)}.$$

So

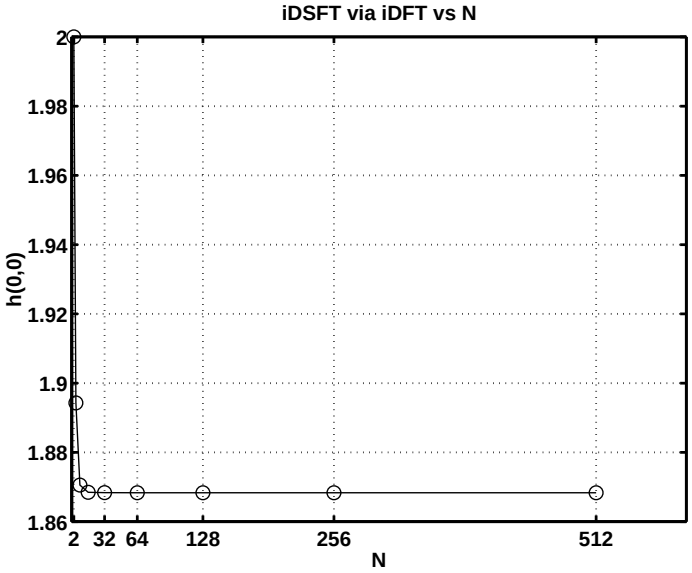
$$y[n,m] = x[n,m] \circledast\circledast h[n,m] = x[n,m] \circledast\circledast \delta[[n,m]]_{(N,M)} = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x[n - kN, m - lM] = x_{\mathrm{ps}}[n,m],$$

i.e., sampling in the Fourier domain causes superimposed replicas in the space domain, which means spatial **aliasing** if $x[n,m]$ is not a finite-extent signal.

Example. Earlier, in the discussion of filter design 2 at the end of Chapter DS, we wanted to find the impulse response of the filter having spectrum

$$H(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) = 4(1 + \cos\omega) \operatorname{rect}\Big(\frac{\omega}{2\pi}\Big), \omega = \sqrt{\omega_{\mathrm{X}}^2 + \omega_{\mathrm{Y}}^2}.$$

I did this numerically by sampling the DSFT, and then computing the iDFT. Since the $h[n, m]$ in this example has infinite extent, this sampling results in a spatially aliased reconstruction of $h[n, m]$. In the example I used $N = M = 128$. Let us see if that was a reasonable value by computing iDFT for various values of N and seeing how $h[0, 0]$ varies with N.

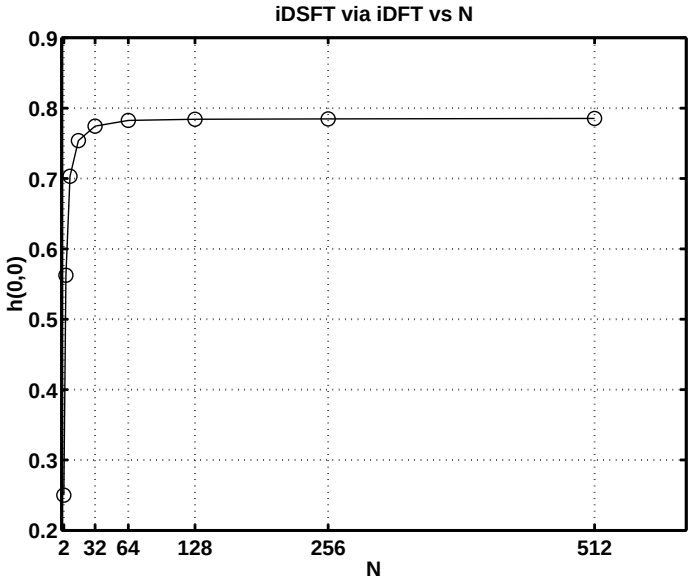


In this case, $h[n, m]$ decays rapidly, so even $N = 32$ appears to suffice.

As a more challenging example, consider the ideal lowpass filter spectrum:

$$H(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) = \operatorname{rect}\Big(\frac{\omega}{2\pi}\Big).$$

Here is $h[0, 0]$ versus N. Ideally $h[n, m] = \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} H(\omega_{\mathrm{X}}, \omega_{\mathrm{Y}}) \, \mathrm{d}\omega_{\mathrm{X}} \, \mathrm{d}\omega_{\mathrm{Y}} = \frac{1}{(2\pi)^2} \pi\pi^2 = \pi/4 \approx 0.785$



In this case $h[n, m]$ is a jinc function, which decays more slowly, so larger N is needed to achieve negligible effects of the spatial aliasing.

Matrix representation of DFT

1D DFT using matrices

The DFT is a linear transformation of the sequence $x[0], \ldots, x[N-1]$ to the coefficients $X[0], \ldots, X[N-1]$. We can always represent any such linear transformation in matrix-vector form.

Define

$$\boldsymbol{x} = \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix} \text{ and } \boldsymbol{X} = \begin{bmatrix} X[0] \\ X[1] \\ \vdots \\ X[N-1] \end{bmatrix}.$$

Then we can write the 1D DFT expression (*cf.* (DFT-6)):

$$X[k] = \sum_{n=0}^{N-1} x[n] \, \mathrm{e}^{-\imath 2\pi nk/N}$$

in matrix-vector form as follows [3, Section 5.1.3]:

$$\boldsymbol{X} = \boldsymbol{W}\boldsymbol{x}, \tag{DFT-10}$$

where $\boldsymbol{W}$ is a $N \times N$ matrix with elements

$$\boldsymbol{W}_{kn} = W_N^{kn} \text{ where } W_N \triangleq \mathrm{e}^{-\imath 2\pi/N}.$$

Since harmonic complex exponential signals are orthogonal (see (DFT-1)), the columns (and rows) of the matrix $\boldsymbol{W}$ are orthogonal. Hence the matrix $\boldsymbol{W}$ is an orthogonal matrix, meaning $\boldsymbol{W}'\boldsymbol{W}$ is a diagonal matrix. Specifically,

$$\boldsymbol{W}'\boldsymbol{W} = N\boldsymbol{I} = \boldsymbol{W}\boldsymbol{W}'$$

where $\boldsymbol{I}$ is the $N \times N$ identity matrix.

Thus, by definition of matrix inverse:

$$\boldsymbol{W}^{-1} = \frac{1}{N}\boldsymbol{W}'.$$

Thus, from (DFT-10),

$$\boldsymbol{x} = \boldsymbol{W}^{-1}\boldsymbol{X} = \frac{1}{N}\boldsymbol{W}'\boldsymbol{X},$$

which is the matrix-vector form of (DFT-7).

We can also view **Parseval's relation** for the DFT in matrix vector form:

$$\boldsymbol{x}'\boldsymbol{y} = \left(\frac{1}{N}\boldsymbol{W}'\boldsymbol{X}\right)' \left(\frac{1}{N}\boldsymbol{W}'\boldsymbol{Y}\right) = \frac{1}{N^2}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{W}'\boldsymbol{Y} = \frac{1}{N^2}\boldsymbol{X}'(N\boldsymbol{I})\boldsymbol{Y} = \frac{1}{N}\boldsymbol{X}'\boldsymbol{Y},$$

and as a special case:

$$\|\boldsymbol{x}\|^2 = \boldsymbol{x}'\boldsymbol{x} = \frac{1}{N}\boldsymbol{X}'\boldsymbol{X} = \frac{1}{N}\|\boldsymbol{X}\|^2.$$

The MATLAB command `dftmtx` constructs the matrix $\boldsymbol{W}$ above.

2D DFT using matrices (See [2, Section 5.2].)

What about the 2D DFT? The 2D DFT is also a linear transformation of the NM values

$$x[n,m],\ n=0,\ldots,N-1,\ m=0,\ldots,M-1$$

to the NM coefficients

$$X[k,l],\ k=0,\ldots,N-1,\ l=0,\ldots,M-1.$$

We can also represent this linear transformation by a $NM \times NM$ matrix. Yet apparently MATLAB does not have a single built-in command for constructing this matrix. How can we construct it?

The first thing we must do is arrange the 2D set of image values into a 1D vector. The standard method for this is called **lexicographic ordering**, defined as follows:

$$\boldsymbol{x} = [x[0,0]\ x[1,0]\ \ldots\ x[N-1,0]\ x[0,1]\ \ldots\ x[N-1,1]\ \ldots\ x[0,M-1]\ \ldots\ x[N-1,M-1]]^T$$

In MATLAB, if the image array is represented by

$$\texttt{xarray} = \begin{bmatrix} x[0,0] & \ldots & x[0,M-1] \\ \vdots & \ddots & \vdots \\ x[N-1,0] & \ldots & x[N-1,M-1] \end{bmatrix},$$

then forming $\boldsymbol{x}$ from `xarray` is as simple as typing:

```
x = xarray(:),
```

which is sometimes written in papers as

$$x = \mathrm{vec}(\mathrm{xarray}).$$

We can similarly arrange the DFT coefficients $X[k,l]$ in lexicographic ordering as a vector $\boldsymbol{X}$. The relationship between $\boldsymbol{X}$ and $\boldsymbol{x}$ is given as follows:

$$\boldsymbol{X} = (\boldsymbol{W}_M \otimes \boldsymbol{W}_N)\boldsymbol{x}, \tag{DFT-11}$$

where $\boldsymbol{A} \otimes \boldsymbol{B}$ denotes the **Kronecker product** of two matrices, defined by

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \ldots & a_{1n}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & \ldots & a_{mn}\boldsymbol{B} \end{bmatrix}.$$

MATLAB's `kron` command computes the Kronecker product. So `kron(dftmtx(M),dftmtx(N))` constructs the 2D DFT matrix. For a 128×128 image, this matrix would be $128^2 \times 128^2$, which would require 4Gbyte to store in the usual double precision format. And even if we could store it, the expression (DFT-11) is very inefficient computationally compared to the 2D FFT. However, for analysis purposes, the representation (DFT-11) can be quite useful.

The above formulation is useful for certain types of analyses.

Example. If $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X}$, where $\boldsymbol{X}$ is a random vector with covariance matrix $\mathsf{Cov}\{\boldsymbol{X}\}$, then the covariance matrix of $\boldsymbol{Y}$ is given by

$$\mathsf{Cov}\{\boldsymbol{Y}\} = \mathsf{Cov}\{\boldsymbol{A}\boldsymbol{X}\} = \boldsymbol{A}\,\mathsf{Cov}\{\boldsymbol{X}\}\,\boldsymbol{A}'.$$

Suppose $x[n]$ includes both a deterministic signal component and an additive random noise component, *i.e.*,

$$x[n] = \mu[n] + \varepsilon[n],$$

where $\mu[n]$ is deterministic and $\varepsilon[n]$ is an uncorrelated sequence of random variables all having the same variance σ^2. Then of course $\mathsf{Cov}\{\boldsymbol{x}\} = \sigma^2\boldsymbol{I}$. Suppose we take the N-point DFT of $x[n]$; what is the covariance of the DFT coefficients?

$$\mathsf{Cov}\{\boldsymbol{X}\} = \mathsf{Cov}\{\boldsymbol{W}\boldsymbol{x}\} = \boldsymbol{W}\,\mathsf{Cov}\{\boldsymbol{x}\}\,\boldsymbol{W}' = \boldsymbol{W}\sigma^2\boldsymbol{I}\boldsymbol{W}' = \sigma^2\boldsymbol{W}\boldsymbol{W}' = \sigma^2 N\boldsymbol{I}.$$

So if the signal values are uncorrelated (and have the same variance) then the DFT coefficients are also uncorrelated (and have the same variance).

This general conclusion applies to any orthogonal transformation, and is the foundation for modern image processing methods like "denoising using wavelets" [4].

Discrete cosine transforms (DCT)

The DFT/FFT are excellent for convolution, and useful for frequency-domain analysis of sampled analog signals. So why did someone invent a new transform, the DCT?

For good image compression, we would like **energy compaction**; a good transform will result in "Fourier coefficients" that are mostly near zero; we can discard or coarsely quantize the small coefficients, and use most of the bits to represent the larger coefficients. Note that for any orthogonal transform $\{\phi_{k,l}\}$ total energy is always preserved by Parseval's theorem:

$$\sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} |x[n,m]|^2 = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} \|\phi_{k,l}\|^2 |c_{k,l}|^2 .$$

For image compression, what matters is how the energy is distributed among the various components.

Image compression is often done in blocks. Suppose we select a small block from some natural image. The DFT of the block gives us the values of the discrete Fourier series of the periodic extension of that signal. Suppose the periodic extension has a discontinuity at the block boundaries. Then the DFT coefficients will decay slowly, just like the FT of a square wave (discontinuous) decay as $1/k$, whereas those of a triangle wave decay as $1/k^2$. So *any* discontinuities in an image, including at the boundary of a block, lead to poor energy compaction of the DFT coefficients.

As an additional drawback of the DFT, if the image is real, then its coefficients are complex. All other things being equal, when developing image compression methods one would usually prefer real valued quantities over complex values if the original image is real.

To overcome these drawbacks of the DFT, **discrete cosine transform** (**DCT**) uses the trick of taking the image (block) and forming a symmetrized version of it before computing a DFT. This symmetrization has the effect of
- eliminating discontinuities at the block edges, and
- yielding real coefficients.

Interestingly though, the DCT is derived via the DFT. So Fourier analysis is still the fundamental tool, even for this new transform.

1D DCT

Consider the signal of extent $N = 4$: $x[n] = \{\underline{2}, 4, 6, 8\}$. Its 4-point circular extension is $\tilde{x}[n] = \{\ldots, 2, 4, 6, 8, \underline{2}, 4, 6, 8, 2, \ldots\}$. Note the discontinuity.

Now consider the new signal of length $2N$:

$$y[n] = \begin{cases} x[n], & 0 \le n \le N-1 \\ x[2N-1-n], & N \le n \le 2N-1 \\ 0, & \text{otherwise}, \end{cases}$$

which is $y[n] = \{\underline{2}, 4, 6, 8, 8, 6, 4, 2\}$ in the example.
This signal has no jumps at the boundaries. We now derive the DCT via the DFT. For $0 \le k \le 2N-1$:

$$\begin{aligned} Y[k] &= \sum_{n=0}^{2N-1} y[n] \, \mathrm{e}^{-\imath \frac{2\pi}{2N} kn} = \sum_{n=0}^{2N-1} y[n] \, W_{2N}^{kn} \\ &= \sum_{n=0}^{N-1} x[n] \, W_{2N}^{kn} + \sum_{n=N}^{2N-1} x[2N-1-n] \, W_{2N}^{kn} \\ &= \sum_{n=0}^{N-1} x[n] \, W_{2N}^{kn} + \sum_{n=0}^{N-1} x[n] \, W_{2N}^{k(2N-1-n)} = \sum_{n=0}^{N-1} x[n] \, W_{2N}^{kn} [1 + W_{2N}^{-k}] \\ &= W_{2N}^{-k/2} \sum_{n=0}^{N-1} x[n] \left(W_{2N}^{k(n+1/2)} + W_{2N}^{-k(n+1/2)} \right) = W_{2N}^{-k/2} \sum_{n=0}^{N-1} x[n] \, 2 \cos\left(\frac{2\pi k (2n+1)}{4N} \right), \end{aligned}$$

where $W_N \triangleq \mathrm{e}^{-\imath \frac{2\pi}{N}}$.

Suppose we started with a length N signal $x[n]$. Now we have $2N$ *complex* DFT coefficients $Y[0], \ldots, Y[2N-1]$. This hardly seems like progress towards compaction! But obviously there must be some redundancies in the $Y[k]$'s.

- Note that $Y[N] = 0$, which is a consequence of the symmetry in the construction of $y[n]$.
- $W_{2N}^{k/2} Y[k]$ is *real* if $x[n]$ is real, since it is a sum of $x[n]$'s times a cosine.
- One can verify that we really need only save *half* of the $Y[k]$'s due to the following form of odd symmetry:

$$-W_{2N}^{(2N-k)/2} Y[2N-k] = W_{2N}^{k/2} Y[k], \quad k = 1, \ldots, 2N-1.$$

In light of these properties, the 1D DCT is defined as follows:

$$C_x[k] \triangleq \begin{cases} W_{2N}^{k/2} Y[k], & 0 \le k \le N-1 \\ 0, & \text{otherwise.} \end{cases}$$

$$\boxed{C_x[k] = \sum_{n=0}^{N-1} 2\, x[n] \cos\left(\frac{\pi k (2n+1)}{2N} \right), \quad k = 0, \ldots, N-1.}$$

A few properties of the DCT:

- Maps an N-point sequence to another N-point sequence.
- If $x[n]$ is real, then so is its DCT.
- $C_x[0] = 2\, Y[0]$, so the 0th component of the DCT is twice that of the DFT.
- We can express it using basis functions: $C_x[k] = \langle x, \phi_k \rangle$, where $\phi_k[n] \triangleq 2 \cos\left(\frac{\pi k (2n+1)}{2N} \right)$.
- Is the set of signals $\{\phi_k\}$ an orthogonal set over $n = 0, \ldots, N-1$? **??**
  Why does it matter? Simplicity in reconstruction: $x[n] = \sum_{k=0}^{N-1} \frac{\langle x, \phi_k \rangle}{\|\phi_k\|^2} \phi_k[n]$.

For the inverse DCT, one can recover the $2N$-point DFT coefficients $Y[k]$ from the DCT coefficients $C_x[\cdot]$ as follows:

$$Y[k] = \begin{cases} W_{2N}^{-k/2} C_x[k], & k = 0, \ldots, N-1 \\ 0, & k = N \\ -W_{2N}^{-k/2} C_x[2N-k], & k = N+1, \ldots, 2N-1. \end{cases}$$

Substituting into the iDFT formula and simplifying (or applying the orthogonality of the DCT basis) yields:

$$\boxed{x[n] = y[n] = \frac{1}{N} \sum_{k=0}^{N-1} w[k]\, C_x[k] \cos\left(\frac{\pi k (2n+1)}{2N} \right), \quad n = 0, \ldots, N-1,}$$

where, because $Y[0] = 2 \sum_{n=0}^{N-1} x[n]$, we have

$$w[k] \triangleq \begin{cases} 1/2, & k = 0 \\ 1, & \text{otherwise.} \end{cases} \tag{DFT-12}$$

Rarely does one *implement* the DCT using the two boxed formulae above, since that would require $O(N^2)$ operations. Instead one uses an FFT-based algorithm.

Basic algorithm for 1D DCT

- extend $x[n]$ to form $y[n]$. (Use MATLAB's `fliplr` or `flipud` command.)
- compute $2N$-point DFT $Y[k]$ from $y[n]$. (Use MATLAB's `fft` command.)
- $C_x[k] = W_{2N}^{k/2} Y[k], \; k = 0, \ldots, N-1$ (Use MATLAB's `real` command after scaling by the $W_{2N}^{k/2}$'s since there will be some residual complex part due to finite precision.)

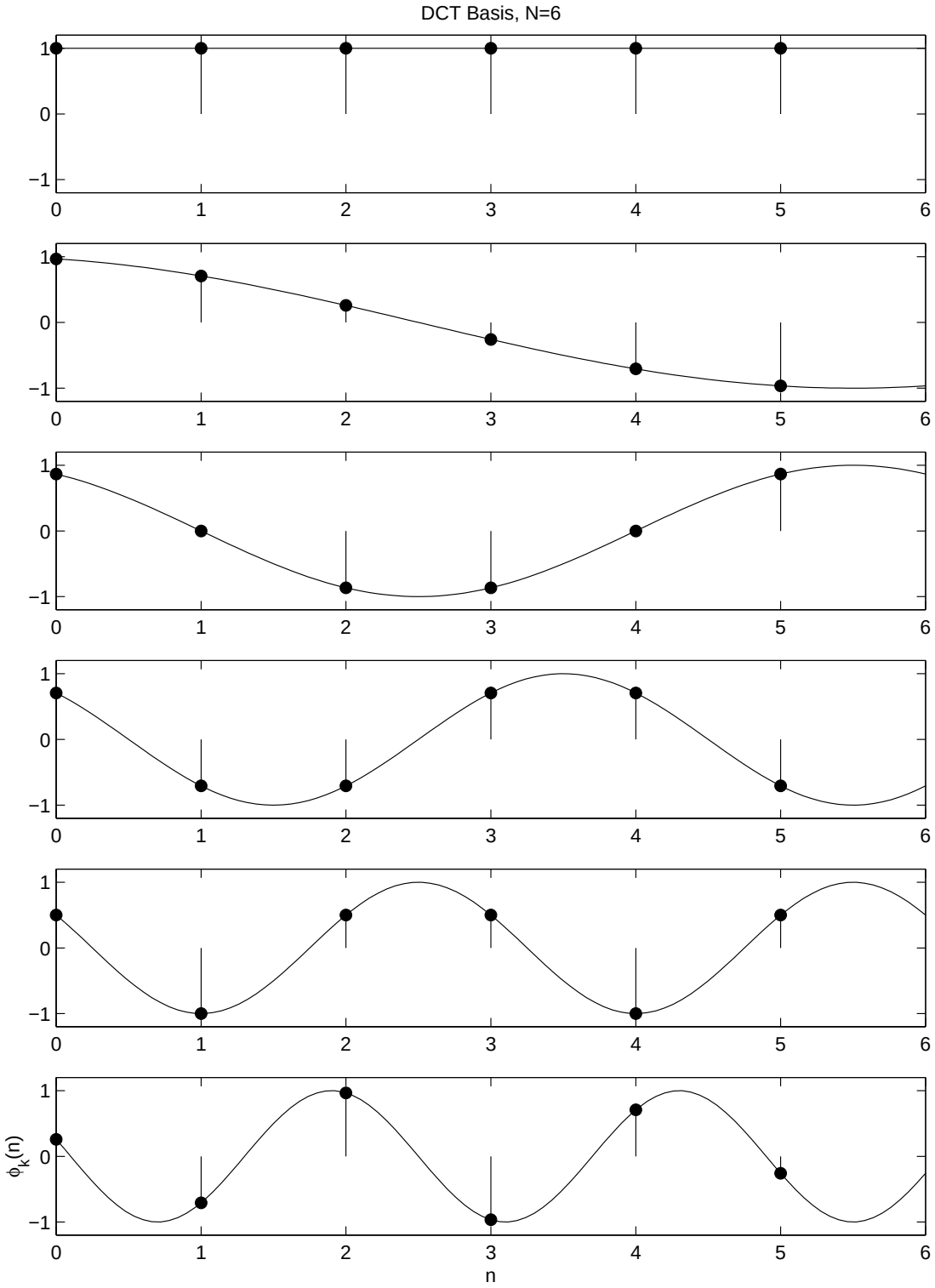
Similar for inverse DCT. In fact it can be done using N-point DFTs too. (A problem in Lim.)

Caution! MATLAB's `dct` command uses a slightly different definition of the DCT that is normalized so that it is an **orthonormal** transformation, following [2, p. 150-3].

Example: $x[n] = \{\underline{2}, 4, 6, 8\}$ has DFT $\{20, -4 + \imath 4, -4, -4 - \imath 4\}$. The DCT is $\{40, -12.6, 0, -0.897\}$, which has nominally better compaction since one of the entries is zero.

Since the DCT input sequence $y[n]$ has no extraneous sharp discontinuities, it will lead to better energy compaction in the frequency domain than the DFT input sequence $x[n]$, *i.e.*, more energy is concentrated in low frequency components.

What is the catch? What signals are better compacted by the DFT? **??**



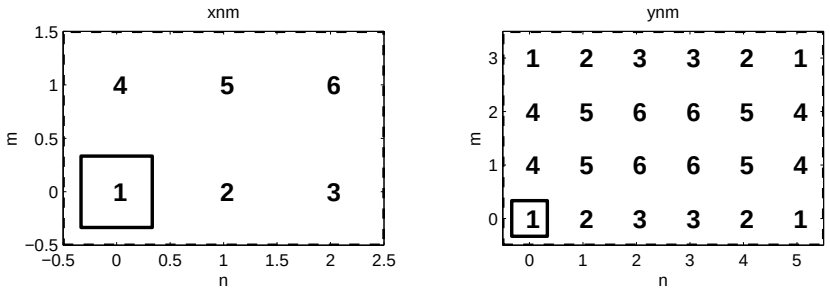
2D DCT

The 2D DCT is defined similarly to the 1D DCT, except that the symmetrizing extension is a 2D operation:

$$y[n,m] = \begin{cases} x[n,m] + x[2N-1-n,m] \\ \quad + \; x[n,2M-1-m] + x[2N-1-n,2M-1-m], & n = 0,\ldots,N-1, m = 0,\ldots,M-1 \\ 0, & \text{otherwise,} \end{cases}$$

assuming $x[n,m]$ is a finite-extent signal that is nonzero only over $n = 0,\ldots,N-1$, $m = 0,\ldots,M-1$.

Example.



By a similar derivation as the 1D case (see text):

$$\boxed{C_x[k,l] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} 4\, x[n,m] \cos\left(\frac{\pi k(2n+1)}{2N}\right) \cos\left(\frac{\pi l(2m+1)}{2M}\right),}$$

and

$$\boxed{x[n,m] = \frac{1}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} w[k]\, w[l]\, C_x[k,l] \cos\left(\frac{\pi k(2n+1)}{2N}\right) \cos\left(\frac{\pi l(2m+1)}{2M}\right),}$$

where $w[k]$ was defined in (DFT-12).

Is the 2D DCT based on a **separable** basis? **??**

Again, rather than using the boxed equations above, one typically uses an FFT-based algorithm.

Basic algorithm for 2D DCT

- Extend $x[n,m]$ to form $y[n,m]$.
  Use MATLAB's `fliplr`, `flipud`, and `rot90` routines.
- compute $2N, 2M$-point DFT $Y[k,l]$ from $y[n,m]$. Use MATLAB's `fft2` routine.
- $C_x[k,l] = W_{2N}^{k/2} W_{2M}^{l/2}\, Y[k,l]$, $k = 0,\ldots,N-1$, $l = 0,\ldots,M-1$.
  Use MATLAB's `ndgrid` or `meshgrid` routine to create the weights, and then take the `real` part to eliminate residual complex component caused by finite numerical precision.

Similar algorithm flow for inverse DCT.

There is ongoing research on even faster algorithms for the 2D DCT, *e.g.*, [5].

Properties of the 2D DCT

linearity, **separability**

symmetry: $x^*[n,m] \overset{\text{DCT}}{\longleftrightarrow} C_x^*[k,l]$

If $x[n,m]$ is real, then $C_x(k,l)$ is real.

Parseval: $\sum_{n=0}^{N-1} \sum_{m=0}^{M-1} |x[n,m]|^2 = \frac{1}{4NM} \sum_{k=0}^{N-1} \sum_{l=0}^{M-1} w[k]\, w[l]\, |C_x[k,l]|^2$

3.3.4 **The discrete-space cosine transform** (for causal sequences $x[n]$.) ***skip***

3.4

Fast Fourier transforms (FFT)

Brute-force evaluation of the 2D DFT would require $O((NM)^2)$ flops. By recognizing that the DFT is a **separable** operation, we can reduce greatly the computation.

Ignoring the braces, we can rewrite the DFT expression as follows:

$$\begin{aligned} X[k,l] &= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} x[n,m] \, \mathrm{e}^{-\imath 2\pi (kn/N + lm/M)} \\ &= \sum_{n=0}^{N-1} \mathrm{e}^{-\imath 2\pi kn/N} \left[\sum_{m=0}^{M-1} x[n,m] \, \mathrm{e}^{-\imath 2\pi lm/M} \right]. \end{aligned}$$

This is called the **row-column decomposition**. Apply the 1D DFT to each column of the image, and then apply the 1D DFT to each row of the result. Naturally we will want to use the **fast Fourier transform** (**FFT**) for these 1D DFTs, which thus reduces the computation to $NO(M \log M)$ for the inner set of 1D FFTs, and then $MO(N \log N)$ for the outer set of 1D FFTs, for a total of $O(MN \log MN)$ flops. For a 512^2 image, the savings in using the row-column with 1D FFTs is about a factor of 15000 relative to the brute-force 2D DFT!

In MATLAB, the `fft` routine applies the 1D FFT to each column of the supplied matrix. So the most basic version of the `fft2` routine could be written in one line as follows:

```
fft(fft(x).').'
```

Why the `.'` in this? **??**

There are also "vector FFT" approaches that can improve over this row-column approach [6].

3.4.2 **Minicomputer implementation**

skip (The book was written in the late 80's...)

3.4.3 **Vector radix FFT**

skip " ... do not offer any significant advantages..."

3.4.4 **Fast algorithms for DFT**

skip (Hard to program. See EECS 658 if interested...)

Nonuniform FFTs

When would one not use the FFT? In some applications, such as MRI and certain versions of tomography, one needs frequency samples that are nonuniformly spaced. Recently several papers have addressed fast algorithms for this problem beginning with [7] and including [8–15]. Such methods are often called the nonuniform FFT, or NUFFT. Many of these algorithms have been presented only for the case of 1D signals. We have recently addressed the multidimensional case [16].

Tricks

There are many useful FFT tricks to further accelerate DFT calculations.

Example. One can compute the DFT of a $2N$-point *real* signal by just one N-point FFT call [3, p. 476].

Example. One can compute the DFT of two *real* signals by just one FFT call [3, p. 475].

This trick, when extended to 2D, is quite useful for convolving two real images.

Summary

Major topics presented.

- DS orthogonal representation
- DFS, properties, circular convolution
- DFT, properties, circular convolution
- sampling the DSFT, spatial aliasing
- matrix representation
- DCT, properties
- FFT

Bibliography

[1] J. S. Lim. *Two-dimensional signal and image processing*. Prentice-Hall, New York, 1990.

[2] A. K. Jain. *Fundamentals of digital image processing*. Prentice-Hall, New Jersey, 1989.

[3] J. Proakis and D. Manolakis. *Digital signal processing: Principles, algorithms and applications*. Prentice-Hall, New York, 1996.

[4] D. L. Donoho. De-noising by soft-thresholding. *IEEE Tr. Info. Theory*, 41(3):613–27, May 1995.

[5] G. Bi, G. Li, K. K. Ma, and T. C. Tan. On the computation of the two-dimensional DCT. *IEEE Tr. Sig. Proc.*, 48(4):1171, April 2000.

[6] H. R. Wu and J. Paoloni. The structure of vector radix fast Fourier transforms. *IEEE Tr. Acoust. Sp. Sig. Proc.*, 37(9):1415–24, September 1989.

[7] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data. *SIAM J. Sci. Comp.*, 14(6):1368–93, November 1993.

[8] C. Anderson and M. D. Dahleh. Rapid computation of the discrete Fourier transform. *SIAM J. Sci. Comp.*, 17(4):913–9, July 1996.

[9] G. Beylkin. On the fast Fourier transform of functions with singularities. *Applied and Computational Harmonic Analysis*, 2(4):363–81, October 1995.

[10] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data, II. *Applied and Computational Harmonic Analysis*, 2(1):85–100, January 1995.

[11] Q. H. Liu and N. Nguyen. An accurate algorithm for nonuniform fast Fourier transforms (NUFFT's). *IEEE Microwave and Guided Wave Letters*, 8(1):18–20, January 1998.

[12] Q. H. Liu and X. Y. Tang. Iterative algorithm for nonuniform inverse fast Fourier transform. *Electronics Letters*, 34(20):1913–4, October 1998.

[13] N. Nguyen and Q. H. Liu. The regular Fourier matrices and nonuniform fast Fourier transforms. *SIAM J. Sci. Comp.*, 21(1):283–93, 1999.

[14] G. Steidl. A note on the fast Fourier transforms for nonequispaced grids. *Advances in computational mathematics*, 9(3):337–52, 1998.

[15] A. F. Ware. Fast approximate Fourier transforms for irregularly spaced data. *SIAM Review*, 40(4):838–56, December 1998.

[16] J. A. Fessler and B. P. Sutton. Nonuniform fast Fourier transforms using min-max interpolation. *IEEE Tr. Sig. Proc.*, 51(2):560–74, February 2003.