L4 (Multidimensional Signals and Systems)

2D Discrete Signals

Separable Sequences

$$x[n_1, n_2] = x_1[n_1]x_2[n_2]$$

2D impulse, 2D step, 2D exponential functions are all separable sequences:

$$\delta[n_1, n_2] = \delta[n_1]\delta[n_2]$$

$$\epsilon[n_1, n_2] = \epsilon[n_1]\epsilon[n_2]$$

$$x[n_1, n_2] = a^{n_1}b^{n_2}$$

If a sequence $h[m, n]$ is separable, then 2D convolution can be accomplished by first applying 1D filtering along each row using $h_y(n)$ and then applying 1D filtering to the intermediate result along each column using $h_x(n)$. For example, Sobel filter, we can first apply H_x and then H_y.

Periodic Sequences

$$x[n_1, n_2] = x[n_1 + N, n_2] = x[n_1, n_2 + N_2]$$

Example would be:

$$x[n_1, n_2] = \cos(\pi n_1 + \frac{\pi}{2} n_2)$$

In this course, we focus on **linear shift-invariant(LSI)** systems

Linear Shift-Invariant Systems

LSI have two properties:

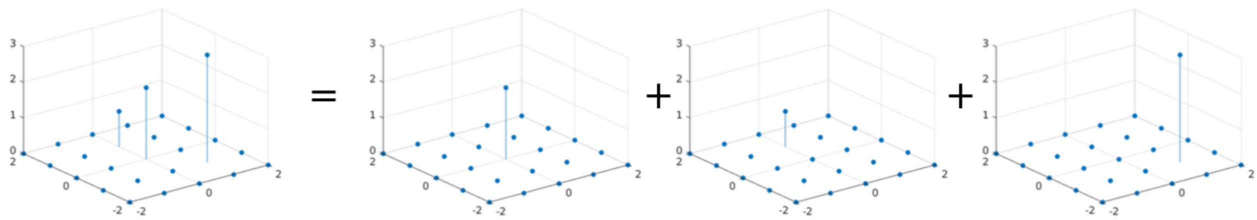
- **linearity** $T(ax_1[n_1, n_2] + bx_2[n_1, n_2]) = aT(x_1[n_1, n_2]) + bT(x_2[n_1, n_2])$
- **shift-invariance**: shift in input implies an equal shift in the output (translational invariance):

$$T(x[n_1 - m_1, n_2 - m_2]) = y[n_1 - m_1, n_2 - m_2]$$

Important Fact

Any 2D sequence can be represented as a weighted combination of shifted 2D unit impulses:

$$x[n_1, n_2] = \sum_{k_1=-\inf}^{+\inf} \sum_{k_2=-\inf}^{+\inf} x[k_1, k_2]\delta[n_1 - k_1, n_2 - k_2]$$



Convolution

For any LSI system, the output $y[n_1, n_2]$ can be computed by convolving the input signal $x[n_1, n_2]$ with the system's impulse response $h[n_1, n_2]$:

$$y[n_1, n_2] = x[n_1, n_2] * h[n_1, n_2] = \sum_{k_1=\inf}^{+\inf} \sum_{k_2=-\inf}^{+\inf} x[k_1, k_2]h[n_1 - k_1, n_2 - k_2]$$

Convolution has 4 main properties. Here, $x[n_1, n_2]$ is the input signal and all others are convolution filters:

- Commutativity: $x[n_1, n_2] * y[n_1, n_2] = y[n_1, n_2] * x[n_1, n_2]$
- Associativity: $(x[n_1, n_2]) * (y[n_1, n_2] + z[n_1, n_2]) = x[n_1, n_2] * y[n_1, n_2] + x[n_1, n_2] * z[n_1, n_2]$
- Distributivity: $x[n_1, n_2] * (y[n_1, n_2] + z[n_1, n_2]) = x[n_1, n_2] * y[n_1, n_2] + x[n_1, n_2] * z[n_1, n_2]$
- Convolution with unit impulse:

  $$x[n_1, n_2] * \delta[n_1 - m_1, n_2 - m_2] = x[n_1 - m_1, n_2 - m_2]$$

  Convolution operation is a cross-correlation where the filter is flipped both horizontally and vertically before being applied to the image.

z-Transform

z-Transform converts signal sequence info frequency representation:

$$X(z_1, z_2) = \sum_{n_1=-\inf}^{+\inf} \sum_{n_2=-\inf}^{+\inf} x[n_1, n_2]z_1^{-n_1}z_2^{-n_2}$$

z here is a complex number: $z_i = A_i e^{j\phi_i} = A_i(\cos\phi_i + j\sin\phi_i)$

2D Discrete Space Fourier Transform (DSFT)

Any discrete signal can be represented as a weighted sum of Fourier basis functions (complex-valued functions of frequency)
It is a particular case of z-transform where $z_1 = e^{j\Omega_1}$, $z_2 = e^{j\Omega_2}$ with continuous spatial frequencies $\Omega_{1,2} = [0, \ldots, 2\pi]$.
Direct transform (natural $->$ frequency):

$$X(e^{j\Omega_1}, e^{j\Omega_2}) = \sum_{n_1=-\inf}^{+\inf} \sum_{n_2=-\inf}^{+\inf} x[n_1, n_2] e_1^{-j\Omega_1 n_1} e_2^{-j\Omega_2 n_2}$$

Inverse Transform (frequency $->$ to natural):

$$x[n_1, n_2] = \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} X(e^{j\Omega_1}, e^{j\Omega_2}) e^{-jn_1\Omega_1} e^{-jn_2\Omega_2} d\Omega_1 d\Omega_2$$

Discrete Fourier Transform (DFT)

DFT

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi kn}{N}}$$

Inverse DFT

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{\frac{j2\pi kn}{N}}$$

Important
Fourier transforms(both DFT and inverse DFT) are separable.

2D DFT Properties

Extending DFT from 1D to 2D, we get the following:

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

Inverse 2D DFT is similar, just change the sign and order

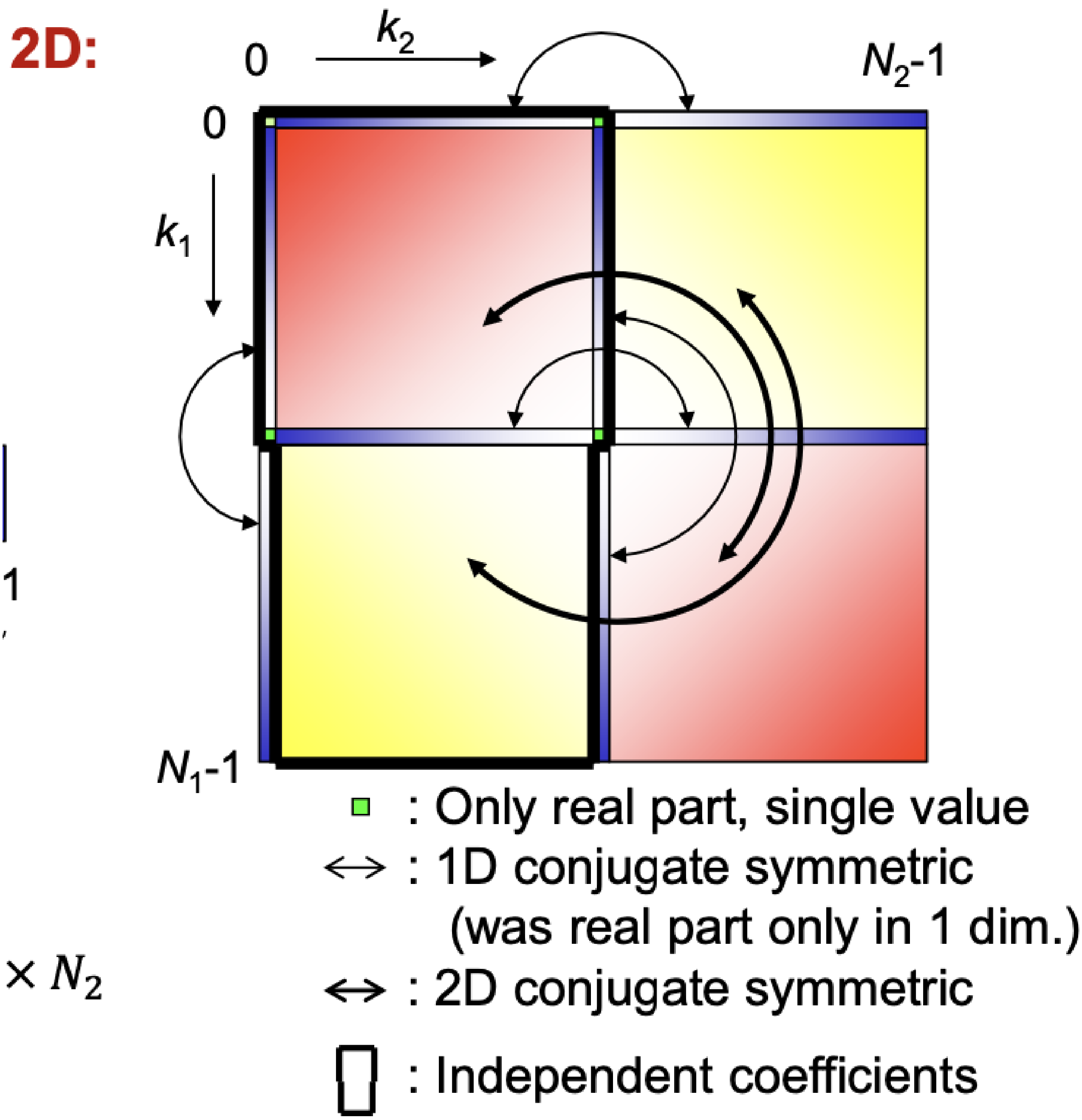
Linearity:

$$ax[n_1, n_2] + by[n_1, n_2] \rightarrow aX[k_1, k_2] + bY[k_1, k_2]$$

Circular Convolution

$$x[n_1, n_2] \circledast y[n_1, n_2] \rightarrow X[k_1, k_2]Y[k_1, k_2]$$

Conjugate Symmetry Property



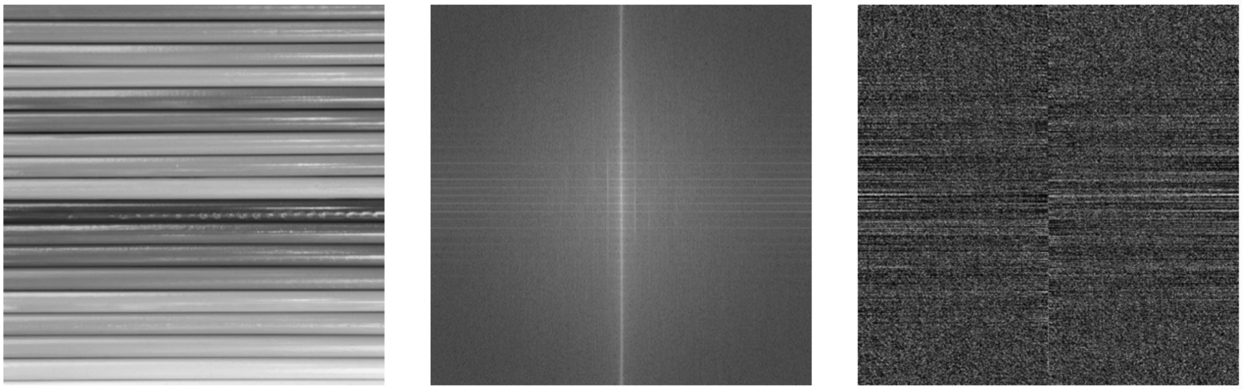
For 2D DFT of real-valued signals $x[n_1, n_2]$, the conjugate symmetry is :

$$X[k_1, k_2] = X^*[N_1 - k_1, N_2 - k_2]$$

where X^* denotes complex conjugate.

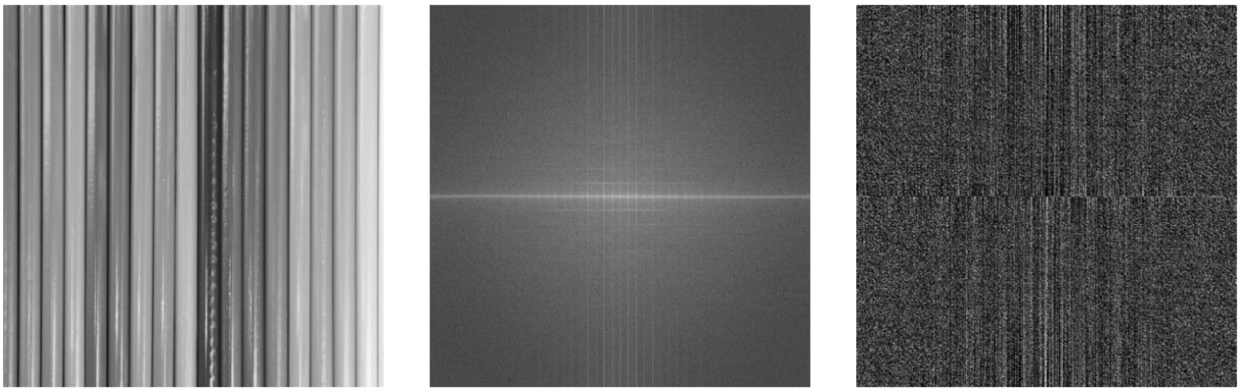
Special cases: $X[0, 0]$, $X[N_1/2, N_2/2]$ are real-valued. $X[k_1, 0]$ and $X[k_1, N_2/2]$ follow 1D conjugate symmetry in k_1 and $X[0, k_2]$ and $X[N_1/2, k_2]$ follow 1D conjugate symmetry in k_2.

2D DFT examples



Original image　log-magnitude　Phase

Horizontal edges are due to **vertical frequencies** (vertically varying Fourier basis functions).



Original image　log-magnitude　Phase

Vertical edges are due to **horizontal frequencies** (horizontally varying Fourier basis functions).

Cross-correlation

1D:

$$\phi_{yx}[k] = \phi_{xx}[k] * h[k]$$

2D:

$$\phi_{yx}[k_1, k_2] = \phi_{xx}[k_1, k_2] * h[k_1, k_2]$$

Linear Image Filtering

Each pixel is replaced by a weighted sum of its neighbors (convolution):
Low pass filter

$$h = \frac{1}{6}\begin{pmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

This convolution filter acts as a low-pass filter, blurring the image.
High pass filter(sharpening):

$$h = \frac{1}{6}\begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

Gaussian Filter

Acts as a linear low-pass filter (Gaussian blur/smoothing):

$$h_G[n_1, n_2] = \frac{1}{2\pi\sigma^2}\exp(-\frac{n_1^2 + n_2^2}{2\sigma^2})$$

n_1 and n_2 are the size of the filter $(3, 3)$. Bigger σ makes the Gaussian squeeze more.

Normalized Gaussian Filter

We apply convolution with the Gaussian filter. Remember the output of convolution operation is given by:

$$y[n_1, n_2] = x[n_1, n_2] * h[n_1, n_2] = \sum_{k_1=\inf}^{+\inf} \sum_{k_2=-\inf}^{+\inf} x[k_1, k_2]h[n_1 - k_1, n_2 - k_2]$$

Now, let's replace h with our Gaussian above:

$$y[n_1, n_2] = x[n_1, n_2] * h[n_1, n_2] = \frac{1}{2\pi\sigma^2} \sum_{k_1=\inf}^{+\inf} \sum_{k_2=-\inf}^{+\inf} x[k_1, k_2]\exp(-\frac{(n_1 - k_1)^2 + (n_2 - k_2)^2}{2\sigma^2})$$

Non-Linear Filtering

The common filter is median filter. It is calculated over square window of size $3x3$ or $5x5$. Goal is to:

- remove single outliers
- remove salt&pepper noise. Median filter is very effective here.
- non-linear prediction

Bilateral Filter

Problem with low-pass filters is that they smooth **all** high frequency content(including edges)
Bilateral filter instead reduces noise but still preserves the edges. The idea is to apply Gaussian blur to similar pixels only and it is non-linear.
It has two kernels: spatial kernel and range kernel. Spatial kernel smoothes differences in coordinates while range kernel smoothes differences in intensities:

$$y[n] = \frac{1}{C} \sum_{n_i \in A} x[n_i] K_{\sigma_s}(||n - n_i||^2) K_{\sigma_r}(|x[n] - x[n_i]|^2)$$

with normalization factor:

$$C = \sum_{n_i \in A} K_{\sigma_s}(||n - n_i||^2) K_{\sigma_r}(|x(n) - x(n_i)|^2)$$

Strong bilateral filters create cartoon-like appearance (in combination with color quantization)

Wiener Filter

Wiener filter is the MSE-optimal stationary linear filter for images degraded by additive noise and blurring. It is applied in the frequency domain. Given a degraded image $x[n, m]$, one takes the DFT to obtain $X(u, v)$. The original image spectrum is estimated by taking the product of $X(u, v)$ with the Wiener filter $H(u, v)$:

$$S(u, v) = H(u, v) X(u, v)$$

where $H(u, v)$ is given by:

$$\boxed{H(\mathrm{e}^{\mathrm{j}\Omega}) = \frac{\Phi_{xx}(\mathrm{e}^{\mathrm{j}\Omega}) G^*(\mathrm{e}^{\mathrm{j}\Omega})}{\Phi_{xx}(\mathrm{e}^{\mathrm{j}\Omega}) |G(\mathrm{e}^{\mathrm{j}\Omega})|^2 + \Phi_{vv}(\mathrm{e}^{\mathrm{j}\Omega})} = \frac{1}{G(\mathrm{e}^{\mathrm{j}\Omega})} \frac{|G(\mathrm{e}^{\mathrm{j}\Omega})|^2}{|G(\mathrm{e}^{\mathrm{j}\Omega})|^2 + \Phi_{vv}(\mathrm{e}^{\mathrm{j}\Omega}) / \Phi_{xx}(\mathrm{e}^{\mathrm{j}\Omega})}}$$

If we assume constant noise to signal ration K:

$$K = \frac{\Phi_{vv}(e^{j\Omega})}{\Phi_{xx}(e^{j\Omega})}$$

We get simplified Wiener filter for image restoration:

$$H(e^{j\Omega}) = \frac{1}{G(e^{j\Omega})} \frac{|G(e^{j\Omega)}|^2}{|G(e^{j\Omega}|^2 + K}$$

And if there is no additive noise, which means noise to signal ratio K is 0, then we have a more simpler term:

$$H(e^{j\Omega}) = \frac{1}{G(e^{j\Omega})}$$

If we falsely set $K = 0$ in Wiener filter when there is actually additive noise, then we get a really bad result.
Goal: estimate a clean image signal x from the observed signal y. It frames this problem as minimization of MSE:

$$\epsilon = E\{|e[n]|^2\} = E\{|x[n] - \hat{x}[n]|^2\}$$

where $\hat{x}[n] = \sum_{l=-\inf}^{\inf} h[l]y[n-l]$. So substituting for $\hat{x}[n]$, we get the following:

$$\epsilon = E\{|e[n]|^2\} = E\{|x[n] - \hat{x}[n]|^2\} = x[n] - \sum_{l=-\inf}^{\inf} h[l]y[n-l]$$

Orthogonality principle

$$E\{e[n]y[n-k]\} = 0$$

for $-\inf < k < \inf$
We get **Wiener-Hopf Equations** to determine $h[n]$:

$$\phi_{xy}[k] = h[k] * \phi_{yy}[k]$$

In frequency domain, this gives for 2D image signals

$$\Phi_{xy}(e^{j\Omega_1}, e^{j\Omega_2}) = H(e^{j\Omega_1}, e^{j\Omega_2})\Phi_{yy}(e^{j\Omega_1}, e^{j\Omega_2}$$

and

$$H_{xy}(e^{j\Omega_1}, e^{j\Omega_2}) = \frac{\Phi_{xy}(e^{j\Omega_1}, e^{j\Omega_2})}{\Phi_{yy}(e^{j\Omega_1}, e^{j\Omega_2})}$$

L4(exercise)

2D DFT properties

Conjugate Symmetry

The conjugate symmetry property says that for 2D DFT of real-valued signals $x[n_1, n_2]$, we have this property:

$$X[k_1, k_2] = X^*[N_1 - k_1, N_2 - k_2]$$

Periodicity

$$X[k_1, k_2] = X[k_1 + c_1N_1, k_2 + c_2N_2]$$

For example, if we want to calculate $X[4, 4]$ this is equal to $X[0, 0]$. Here
$k_1, k_2 \in \{0, 1, \ldots, N-1\}$
These values does not contain complex parts: $X[0, 0]$, $X[0, 2]$, $X[2, 0]$, $X[2, 2]$.

Every other value is a conjugate pair according to conjugate symmetry property above.
Problem solving
If you are asked to find errors in a given 2D DFT matrix, just check the boundary values that are real, and check if other values have complex conjugates by visual inspection.

Separability of 2D DFT

If we have row transformation $X_R[n_1, k_2]$, and transformation matrix T_{DFT}, we can calculate the final 2D DFT my matrix multiplication:

$$X[k_1, k_2] = T_{DFT} X_R[n_1, k_2]$$

2D DFT interpretation

If a column is filled with zeros, we can subsample the 2D DFT without losing information. DC part which corresponds to $X[0, 0]$ (because there are no frequency component, just write down the formula for proof) indicates the energy of the image. DC value is the average intensity of the entire image.

DFST vs DFT

DFST is in continuous frequency domain $(w_1, w_2) \in [-\pi, \pi]$ and the result is continuous and periodic in frequency.
DFT has both spatial and frequency domains in discrete and contains finite summation ($N_1 x N_2$ product)

Linear Filtering (Convolution)

Shift: The filter is shifted since the indices run from -1 to 1.
Flip: it should always be flipped. If not, the operation would be correlation rather than convolution. If the filter is symmetric $[1, 2, 1]$, it doesn't matter if we flip it or not