

Intro to Digital Signal Processing

Intro to Digital Signal Processing

Collection edited by: Robert Nowak

Content authors: Robert Nowak, Don Johnson, Michael Haag, Behnaam Aazhang, Nick Kingsbury, Benjamin Fite, Melissa Selik, Richard Baraniuk, Dan Calderon, Ivan Selesnick, Bill Wilson, Hyeokho Choi, Douglas Jones, Swaroop Appadwedula, Matthew Berry, Mark Haun, Dima Moussa, Daniel Sachs, Roy Ha, Justin Romberg, and Phil Schniter

Online: <<http://cnx.org/content/col10203/1.4>>

This selection and arrangement of content as a collection is copyrighted by Robert Nowak.

It is licensed under the Creative Commons Attribution License: <http://creativecommons.org/licenses/by/1.0>

Collection structure revised: 2004/01/22

For copyright and attribution information for the modules contained in this collection, see the "[Attributions](#)" section at the end of the collection.

Intro to Digital Signal Processing

Table of Contents

- [Chapter 1. DSP Systems I](#)
 - [1.1. Image Restoration Basics](#)
 - [Image Restoration](#)
 - [Frequency Domain Analysis](#)
 - [1.2. Digital Image Processing Basics](#)
 - [Digital Image Processing](#)
 - [1.3. 2D DFT](#)
 - [2D DFT](#)
 - [Inverse 2D DFT](#)
 - [2D DFT and Convolution](#)
 - [Circular Convolution](#)
 - [Zero Padding](#)
 - [Computing the 2D DFT](#)
 - [1.4. Images: 2D signals](#)
 - [Image Processing](#)
 - [Linear Shift Invariant Systems](#)
 - [2D Fourier Analysis](#)
 - [2D Sampling Theory](#)
 - [Nyquist Theorem](#)
 - [1.5. DFT as a Matrix Operation](#)
 - [Matrix Review](#)
 - [Representing DFT as Matrix Operation](#)
 - [1.6. Fast Convolution Using the FFT](#)
 - [Important Application of the FFT](#)
 - [Summary of DFT](#)
 - [What is the DFT](#)
 - [Inverse DFT \(IDFT\)](#)
 - [Circular Convolution](#)
 - [Regular Convolution from Circular Convolution](#)
 - [The Fast Fourier Transform \(FFT\)](#)
 - [Fast Convolution](#)
 - [1.7. The FFT Algorithm](#)
 - [The Fast Fourier Transform FFT](#)
 - [How does the FFT work?](#)
 - [Decimation in Time FFT](#)
 - [Derivation](#)
 - [1.8. The DFT: Frequency Domain with a Computer Analysis](#)

- Introduction
 - Sampling DTFT
 - Choosing M
 - Case 1
 - Case 2
- Discrete Fourier Transform (DFT)
 - Interpretation
 - Remark 1
 - Remark 2
- Periodicity of the DFT
- A Sampling Perspective
 - Inverse DTFT of $S(\omega)$
- Connections
- 1.9. Discrete-Time Processing of CT Signals
 - DT Processing of CT Signals
 - Analysis
 - Summary
 - Note
 - Application: 60Hz Noise Removal
 - DSP Solution
 - Sampling Period/Rate
 - Digital Filter
- 1.10. Sampling CT Signals: A Frequency Domain Perspective
 - Understanding Sampling in the Frequency Domain
 - Sampling
 - Relating $x[n]$ to sampled $x(t)$
- 1.11. Filtering with the DFT
 - Introduction
 - Compute IDFT of $Y[k]$
 - DFT Pair
 - Regular Convolution from Periodic Convolution
 - DSP System
- 1.12. Ideal Reconstruction of Sampled Signals
 - Reconstruction of Sampled Signals
 - Step 1
 - Step 2
 - Ideal Reconstruction System
- 1.13. Amplitude Quantization
- 1.14. Classic Fourier Series
- Chapter 2. Random Signals
 - 2.1. Introduction to Random Signals and Processes
 - Signals: Deterministic vs. Stochastic
 - Deterministic Signals

- [Stochastic Signals](#)
 - [Random Process](#)
 - [2.2. Introduction to Stochastic Processes](#)
 - [Definitions, distributions, and stationarity](#)
 - [2.3. Random Signals](#)
 - [Example - Detection of a binary signal in noise](#)
 - [2.4. Stationary and Nonstationary Random Processes](#)
 - [Introduction](#)
 - [Distribution and Density Functions](#)
 - [Stationarity](#)
 - [First-Order Stationary Process](#)
 - [Second-Order and Strict-Sense Stationary Process](#)
 - [Wide-Sense Stationary Process](#)
 - [2.5. Random Processes: Mean and Variance](#)
 - [Mean Value](#)
 - [Properties of the Mean](#)
 - [Mean-Square Value](#)
 - [Variance](#)
 - [Standard Deviation](#)
 - [Properties of Variance](#)
 - [Time Averages](#)
 - [Estimating the Mean](#)
 - [Estimating the Variance](#)
 - [Example](#)
 - [2.6. Correlation and Covariance of a Random Signal](#)
 - [Covariance](#)
 - [Useful Properties](#)
 - [Correlation](#)
 - [Correlation Coefficient](#)
 - [Example](#)
 - [Matlab Code for Example](#)
 - [2.7. Autocorrelation of Random Processes](#)
 - [Autocorrelation Function](#)
 - [Properties of Autocorrelation](#)
 - [Estimating the Autocorrelation with Time-Averaging](#)
 - [Examples](#)
 - [2.8. Crosscorrelation of Random Processes](#)
 - [Crosscorrelation Function](#)
 - [Properties of Crosscorrelation](#)
 - [Examples](#)
- [Chapter 3. Filter Design I \(Z-Transform\)](#)
 - [3.1. Difference Equation](#)
 - [Introduction](#)

- General Formulas for the Difference Equation
 - Difference Equation
 - Conversion to Z-Transform
 - Conversion to Frequency Response
- Example
- Solving a LCCDE
 - Direct Method
 - Homogeneous Solution
 - Particular Solution
 - Indirect Method
- 3.2. The Z Transform: Definition
 - Basic Definition of the Z-Transform
 - The Complex Plane
 - Region of Convergence
- 3.3. Table of Common z-Transforms
- 3.4. Poles and Zeros
 - Introduction
 - Pole/Zero Plots
 - Repeated Poles and Zeros
 - Pole-Zero Cancellation
- 3.5. Rational Functions and the Z-Transform
 - Introduction
 - Properties of Rational Functions
 - Roots
 - Discontinuities
 - Domain
 - Intercepts
 - Rational Functions and the Z-Transform
 - Conclusion
- 3.6. The Complex Plane
 - Complex Plane
 - Rectangular Coordinates
 - Polar Form
- 3.7. Region of Convergence for the Z-transform
 - Introduction
 - The Region of Convergence
 - Properties of the Region of Convergence
 - Examples
 - Graphical Understanding of ROC
 - Conclusion
- 3.8. Understanding Pole/Zero Plots on the Z-Plane
 - Introduction to Poles and Zeros of the Z-Transform
 - The Z-Plane

- [Examples of Pole/Zero Plots](#)
 - [Interactive Demonstration of Poles and Zeros](#)
 - [Applications for pole-zero plots](#)
 - [Stability and Control theory](#)
 - [Pole/Zero Plots and the Region of Convergence](#)
 - [Frequency Response and Pole/Zero Plots](#)
- [3.9. Zero Locations of Linear-Phase FIR Filters](#)
 - [Zero Locations of Linear-Phase Filters](#)
 - [ZEROS LOCATIONS](#)
 - [ZERO LOCATIONS: AUTOMATIC ZEROS](#)
 - [ZERO LOCATIONS: EXAMPLES](#)
- [3.10. Discrete Time Filter Design](#)
 - [Estimating Frequency Response from Z-Plane](#)
 - [Drawing Frequency Response from Pole/Zero Plot](#)
 - [Interactive Filter Design Illustration](#)
 - [Conclusion](#)
- [Chapter 4. Filter Design II](#)
 - [4.1. Bilinear Transform](#)
- [Chapter 5. Filter Design III](#)
 - [5.1. Linear-Phase FIR Filters](#)
 - [THE AMPLITUDE RESPONSE](#)
 - [WHY LINEAR-PHASE?](#)
 - [WHY LINEAR-PHASE: EXAMPLE](#)
 - [WHY LINEAR-PHASE: EXAMPLE \(2\)](#)
 - [WHY LINEAR-PHASE: MORE](#)
 - [5.2. Four Types of Linear-Phase FIR Filters](#)
 - [FOUR TYPES OF LINEAR-PHASE FIR FILTERS](#)
 - [TYPE I: ODD-LENGTH SYMMETRIC](#)
 - [TYPE II: EVEN-LENGTH SYMMETRIC](#)
 - [TYPE III: ODD-LENGTH ANTI-SYMMETRIC](#)
 - [TYPE IV: EVEN-LENGTH ANTI-SYMMETRIC](#)
 - [5.3. Design of Linear-Phase FIR Filters by DFT-Based Interpolation](#)
 - [DESIGN OF FIR FILTERS BY DFT-BASED INTERPOLATION](#)
 - [Types I and II](#)
 - [Types III and IV](#)
 - [EXAMPLE: DFT-INTERPOLATION \(TYPE I\)](#)
 - [SUMMARY: IMPULSE AND AMP RESPONSE](#)
 - [5.4. Design of Linear-Phase FIR Filters by General Interpolation](#)
 - [DESIGN OF FIR FILTERS BY GENERAL INTERPOLATION](#)
 - [EXAMPLE](#)
 - [LINEAR-PHASE FIR FILTERS: PROS AND CONS](#)
 - [5.5. Linear-Phase FIR Filters: Amplitude Formulas](#)
 - [SUMMARY: AMPLITUDE FORMULAS](#)

- [AMPLITUDE RESPONSE CHARACTERISTICS](#)
- [EVALUATING THE AMPLITUDE RESPONSE](#)
 - [Types I and II](#)
 - [Types III and IV](#)
 - [Modules for Further Study](#)
- [5.6. FIR Filter Design using MATLAB](#)
 - [FIR Filter Design Using MATLAB](#)
 - [Design by windowing](#)
 - [Parks-McClellan FIR filter design](#)
- [5.7. MATLAB FIR Filter Design Exercise](#)
 - [FIR Filter Design MATLAB Exercise](#)
 - [Design by windowing](#)
 - [Parks-McClellan Optimal Design](#)
- [5.8. Parks-McClellan Optimal FIR Filter Design](#)
- [Chapter 6. Wiener Filter Design](#)
- [Chapter 7. Adaptive Filtering](#)
 - [7.1. Adaptive Filtering: LMS Algorithm](#)
 - [Introduction](#)
 - [Gradient-descent adaptation](#)
 - [MATLAB Simulation](#)
 - [Processor Implementation](#)
 - [Extensions](#)
 - [References](#)
- [Chapter 8. Wavelets and Filterbanks](#)
 - [8.1. Haar Wavelet Basis](#)
 - [Introduction](#)
 - [Basis Comparisons](#)
 - [Haar Wavelet Transform](#)
 - [Haar Wavelet Demonstration](#)
 - [8.2. Orthonormal Wavelet Basis](#)
 - [8.3. Continuous Wavelet Transform](#)
 - [8.4. Discrete Wavelet Transform: Main Concepts](#)
 - [Main Concepts](#)
 - [8.5. The Haar System as an Example of DWT](#)
- [Index](#)

Chapter 1. DSP Systems I

1.1. Image Restoration Basics*

Image Restoration

In many applications (*e.g.*, satellite imaging, medical imaging, astronomical imaging, poor-quality family portraits) the imaging system introduces a slight distortion. Often images are slightly blurred and image restoration aims at **deblurring** the image.

The blurring can usually be modeled as an LSI system with a given PSF $h[m, n]$.

$$h[m,n] \xleftrightarrow{\text{FT}} H(u,v)$$

Figure 1.1.

Fourier Transform (FT) relationship between the two functions.

The observed image is

$$g[m, n] = h[m, n] * f[m, n] \quad 0$$

$$G(u, v) = H(u, v)F(u, v) \quad 0$$

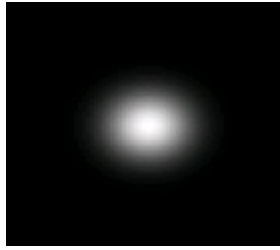
$$F(u, v) = \frac{G(u, v)}{H(u, v)} \quad 0$$

Example 1.1. Image Blurring

Above we showed the equations for representing the common model for blurring an image. In [Figure 1.2](#) we have an original image and a PSF function that we wish to apply to the image in order to model a basic blurred image.



(a)



(b)

Figure 1.2.

Once we apply the PSF to the original image, we receive our blurred image that is shown in [Figure 1.3](#):



Figure 1.3.

Frequency Domain Analysis

[Example 1.1](#) looks at the original images in its typical form; however, it is often useful to look at our images and PSF in the frequency domain. In [Figure 1.4](#), we take another look at the image blurring example above and look at how the images and results would appear in the frequency domain if we applied the fourier transforms.

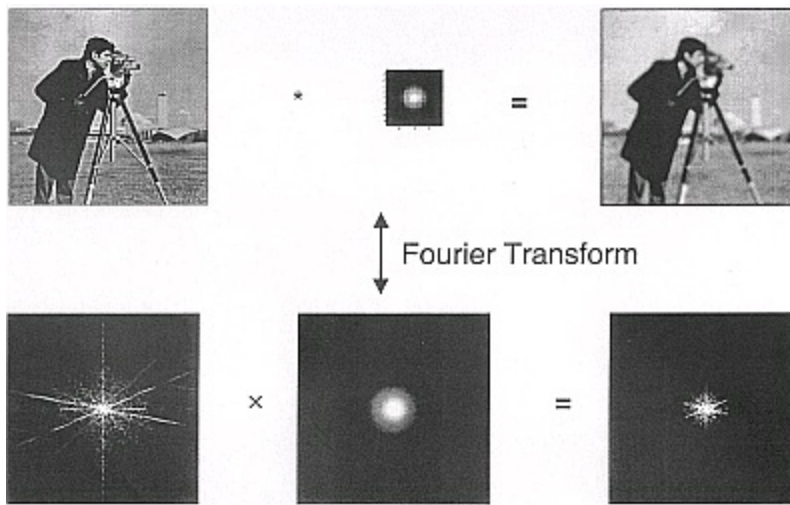


Figure 1.4.

1.2. Digital Image Processing Basics*

Digital Image Processing

A sampled image gives us our usual 2D array of pixels $f[m, n]$ (Figure 1.5):

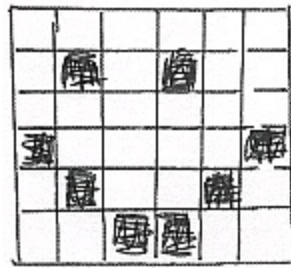


Figure 1.5.

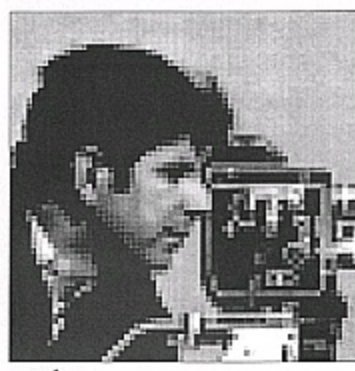
We illustrate a "pixelized" smiley face.

We can filter $f[m, n]$ by applying a 2D discrete-space **convolution** as shown below (where $h[m, n]$ is our PSF):

$$\begin{aligned}
 g[m, n] &= h[m, n] * f[m, n] \\
 &= \sum_{k=-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} (h[m-k, n-l]f[k, l]) \right)
 \end{aligned}$$

0

Example 1.2. Sampled Image



close-up shows "pixelized" (sampled) nature of the image

Figure 1.6.

Illustrate the "pixelized" nature of all digital images.

We also have discrete-space FTS:

$$F[u, v] = \sum_{m=-\infty}^{\infty} \left(\sum_{n=-\infty}^{\infty} (f[m, n] e^{-i\hat{u}m} e^{-i\hat{v}n}) \right) \quad 0$$

where $F[u, v]$ is analogous to **DTFT** in 1D.

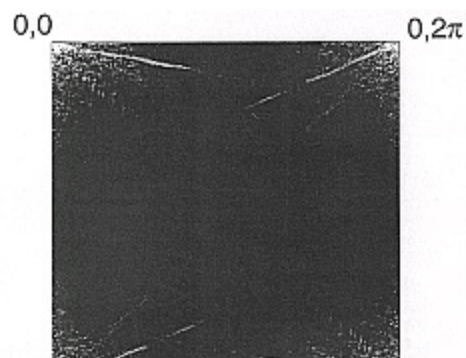
"Convolution in Time" is the **same** as "Multiplication in Frequency"

$$g[m, n] = h[m, n] * f[m, n] \quad 0$$

which, as stated above, is the same as:

$$G[u, v] = H[u, v] F[u, v] \quad 0$$

Example 1.3. Magnitude of FT of Cameraman Image



2π,0 2π,2π

Figure 1.7.

To get a better image, we can use the `fftshift` command in Matlab to center the Fourier Transform. The resulting image is shown in [Figure 1.8](#):

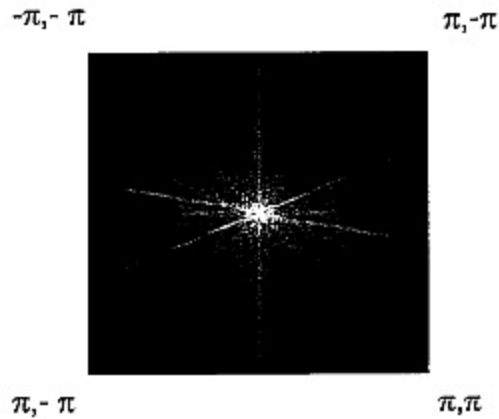


Figure 1.8.

1.3. 2D DFT*

2D DFT

To perform image restoration (and many other useful image processing algorithms) in a computer, we need a Fourier Transform (FT) that is discrete and two-dimensional.

$$F[k, l] = F(u, v) \Big|_{u = \frac{2\pi k}{N}, v = \frac{2\pi l}{N}} \quad 0$$

for $k = \{0, \dots, N-1\}$ and $l = \{0, \dots, N-1\}$.

$$F(u, v) = \sum_m \left(\sum_n \left(f[m, n] e^{-i\hat{u}m} e^{-i\hat{v}n} \right) \right) \quad 0$$

$$F[k, l] = \sum_{m=0}^{N-1} \left(\sum_{n=0}^{N-1} \left(f[m, n] e^{(-i)\frac{2\pi km}{N}} e^{(-i)\frac{2\pi ln}{N}} \right) \right) \quad 0$$

where the above equation ([Equation](#)) has finite support for an $N \times N$ image.

Inverse 2D DFT

As with our regular fourier transforms, the 2D DFT also has an inverse transform that allows us to reconstruct an image as a weighted combination of complex sinusoidal basis functions.

$$f[m, n] = \frac{1}{N^2} \sum_{k=0}^{N-1} \left(\sum_{l=0}^{N-1} \left(F[k, l] e^{\frac{i2\pi km}{N}} e^{\frac{i2\pi ln}{N}} \right) \right)$$

0

Example 1.4. Periodic Extensions

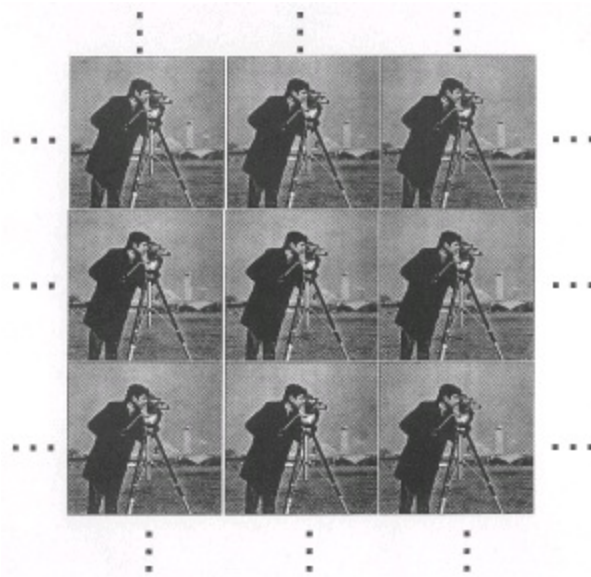


Figure 1.9.

Illustrate the periodic extension of images.

2D DFT and Convolution

The regular 2D convolution equation is

$$g[m, n] = \sum_{k=0}^{N-1} \left(\sum_{l=0}^{N-1} (f[k, l] h[m-k, n-l]) \right)$$

0

Example 1.5.

Below we go through the steps of convolving two two-dimensional arrays. You can think of f as representing an image and h represents a PSF, where $h[m, n]=0$ for m and $n > 1$ and

m and $n < 0$. $h = \begin{pmatrix} h[0, 0] & h[0, 1] \\ h[1, 0] & h[1, 1] \end{pmatrix}$ $f = \begin{pmatrix} f[0, 0] & \dots & f[0, N-1] \\ \vdots & \ddots & \vdots \\ f[N-1, 0] & \dots & f[N-1, N-1] \end{pmatrix}$ Step 1 (Flip h):

$$h[-m, -n] = \begin{pmatrix} h[1, 1] & h[1, 0] & 0 \\ h[0, 1] & h[0, 0] & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Step 2 (Convolve):

$$g[0, 0] = h[0, 0]f[0, 0]$$

We use the standard 2D convolution equation ([Equation](#)) to find the first element of our convolved image. In order to better understand what is happening, we can think of this visually. The basic idea is to take $h[-m, -n]$ and place it "on top" of $f[k, l]$, so that just the bottom-right element, $h[0, 0]$ of $h[-m, -n]$ overlaps with the top-left element, $f[0, 0]$, of $f[k, l]$. Then, to get the next element of our convolved image, we slide the flipped matrix, $h[-m, -n]$, over one element to the right and get the following result:

$g[0, 1] = h[0, 0]f[0, 1] + h[0, 1]f[0, 0]$ We continue in this fashion to find all of the elements of our convolved image, $g[m, n]$. Using the above method we define the general formula to find a particular element of $g[m, n]$ as:

$$g[m, n] = h[0, 0]f[m, n] + h[0, 1]f[m, n-1] + h[1, 0]f[m-1, n] + h[1, 1]f[m-1, n-1]$$

Circular Convolution

Exercise 1.

What does $H[k, l]F[k, l]$ produce?

2D Circular Convolution

$$\begin{aligned} \tilde{g}[m, n] &= \text{IDFT}(H[k, l]F[k, l]) \\ &= \text{circular convolution in 2D} \end{aligned}$$

Due to periodic extension by DFT ([Figure 1.10](#)):

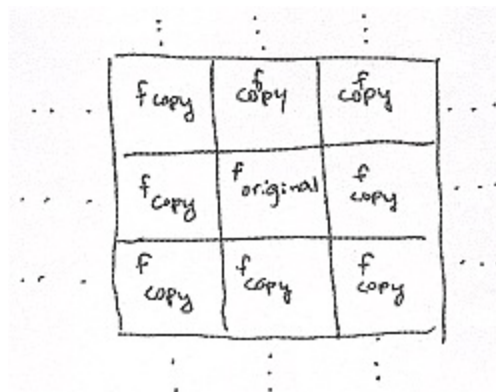


Figure 1.10.

Based on the above solution, we will let

$$\tilde{g}[m, n] = \text{IDFT}\{H[k, l]F[k, l]\} \quad 0$$

Using this equation, we can calculate the value for each position on our final image, $\tilde{g}[m, n]$. For example, due to the periodic extension of the images, when circular convolution is applied we will observe a **wrap-around** effect.

$$\tilde{g}[0, 0] = h[0, 0]f[0, 0] + h[1, 0]f[N-1, 0] + h[0, 1]f[0, N-1] + h[1, 1]f[N-1, N-1] \quad 0$$

Where the last three terms in [Equation](#) are a result of the wrap-around effect caused by the presence of the images copies located all around it.

Zero Padding

If the support of h is $M \times M$ and f is $N \times N$, then we zero pad f and h to $M+N-1 \times M+N-1$ (see [Figure 1.11](#)).

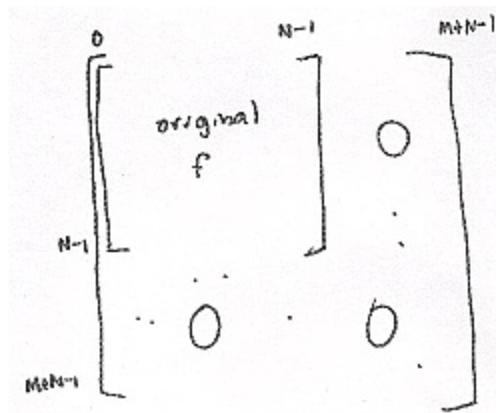


Figure 1.11.

Circular Convolution = Regular Convolution

Computing the 2D DFT

$$F[k, l] = \sum_{m=0}^{N-1} \left(\sum_{n=0}^{N-1} \left(f[m, n] e^{(-j)\frac{2\pi kn}{N}} e^{(-j)\frac{2\pi ln}{N}} \right) \right) \quad 0$$

where in the above equation, $\sum_{n=0}^{N-1} \left(f[m, n] e^{(-j)\frac{2\pi ln}{N}} \right)$ is simply a 1D DFT over n . This means that we will take the 1D FFT of each row; if we have N rows, then it will require $N \log N$ operations per row. We can rewrite this as

$$F[k, l] = \sum_{m=0}^{N-1} \left(f[m, l] e^{(-j) \frac{2\pi k m}{N}} \right)$$

where now we take the 1D FFT of each column, which means that if we have N columns, then it requires $N \log N$ operations per column.

Therefore the overall complexity of a 2D FFT is $O(N^2 \log N)$ where N^2 equals the number of pixels in the image.

1.4. Images: 2D signals*

Image Processing

Image not finished

Figure 1.12.

Images are 2D functions $f(x, y)$

Linear Shift Invariant Systems

Image not finished

Figure 1.13.

H is LSI if:

1. $H(\alpha f_1(x, y) + \alpha_2 f_2(x, y)) = H(f_1(x, y)) + H(f_2(x, y))$ for all images f_1, f_2 and scalar.
2. $H(f(x - x_0, y - y_0)) = g(x - x_0, y - y_0)$

LSI systems are expressed mathematically as 2D convolutions: $g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x - \alpha, y - \beta) f(\alpha, \beta) d\alpha d\beta$

where $h(x, y)$ is the 2D impulse response (also called the **point spread function**).

2D Fourier Analysis

Thank You for previewing this eBook

You can read the full version of this eBook in different formats:

- HTML (Free /Available to everyone)
- PDF / TXT (Available to V.I.P. members. Free Standard members can access up to 5 PDF/TXT eBooks per month each month)
- Epub & Mobipocket (Exclusive to V.I.P. members)

To download this full book, simply select the format you desire below

