AN INTROUCTION TO 2-DIMENSIONAL DSP

Richard M. Stern

18-792 lecture

December 2, 2024

Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

INTRODUCTION

- **Background:** Many types of analyses make use of 2- dimensional images ....
  – Photographs
  – Satellite images
  – X-rays and other medical images

- **Many concepts from 1-D DSP are directly extensible to two dimensions, but some are not**

INTRODUCTION

- **Goals of this lecture:**
  - To summarize basic 2-D relationships
  - To identify which concepts do or do not extend to 2-D
  - To discuss briefly 2-D filter design approaches
- **For further reading:**
  - *Two-Dimensional Signal and Image Processing* by Jae Lim
  - Chapter *Two-Dimensional Signal Processing* by Lim in the edited book by Lim and Oppenheim on Advanced DSP (pseudo-text for ADSP)
  - Many many other texts and resources

Some examples of original and processed images

- **Peppers …**

Effects of lowpass filtering

- **Original image:**

- **After lowpass filter:**

Effects of highpass filtering

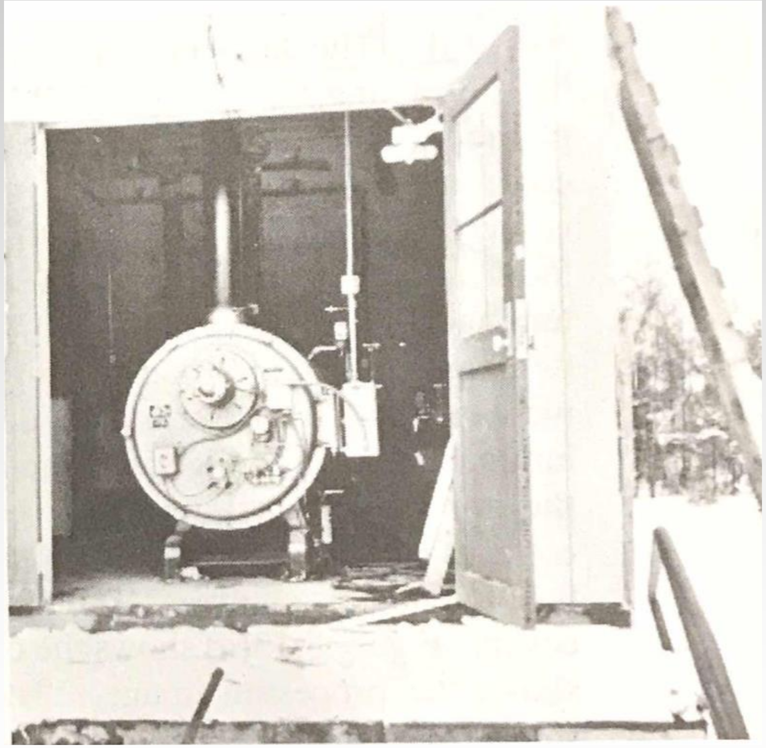
- **Original image:**

- **After highpass filter:**

An example of nonlinear processing

- **Original:**

Enhancement via homomorphic homomorphic filtering:



Some examples of 2-D signals

- **The unit sample function:**

$$\delta[n_1, n_2] = \begin{cases} 1, & n_1 = n_2 = 0 \\ 0, & \text{otherwise} \end{cases}$$

- **The unit step function:**

$$u[n_1, n_2] = \begin{cases} 1, & n_1 \geq 0, n_2 \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

- **The exponential function:**

$$x[n_1, n_2] = \alpha^{n_1} \beta^{n_2}$$

Some examples of 2-D signals

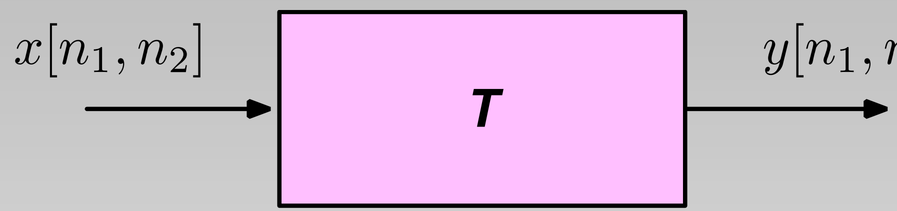
- **Cosine functions:**

$$x[n_1, n_2] = \cos(\omega_1 n_1 + \phi_1)\cos(\omega_2 n_2 + \phi_2)$$

- **Note:** A sequence is **separable** if

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

2-D LSI systems



- **A system is linear if**

$$ax_1[n_1, n_2] + bx_2[n_1.n_2] \Rightarrow ay_1[n_1, n_2] + by_2[n_1.n_2]$$

- **A system is shift invariant if for all k, l**

$$x[n_1 - k, n_2 - l] \Rightarrow y[n_1 - k, n_2 - l]$$

- **If a 2-D system is LSI, then**

$\delta[n_1, n_2] \Rightarrow h[n_1, n_2]$ **(this is called the point-spread function)**

The convolution sum

- As in 1-D, we can represent an input as a linear combination of shifted and scaled delta functions producing …
- 1-D convolution:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

- 2-D convolution:

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

Convolving separable functions

- **If both $x[n_1,n_2]$ and $h[n_1,n_2]$ are separable, then**

$$
\begin{aligned}
y[n_1,n_2] &= \sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty} x[k_1,k_2]h[n_1-k_1,n_2-k_2] \\
&= \sum_{k_1=-\infty}^{\infty}\sum_{k_2=-\infty}^{\infty} x_1[k_1]x_2[k_2]h_1[n_1-k_1]h_2[n_2-k_2]
\end{aligned}
$$

or

$$
y[n_1,n_2] = \sum_{k_1=-\infty}^{\infty} x_1[k_1]h_1[n_1-k_1] \sum_{k_2=-\infty}^{\infty} x_2[k_2]h_2[n_2-k_2]
$$

- In other words, if x and h are separable, the 2-D convolution becomes the product of two 1-D convolutions. For finite sequences of length *N*, this reduces the number of multiplys from N^4 to $2N^2$

Some system properties

- A system is **causal** if

$$h[n_1, n_2], = h[n_1, n_2]u[n_1, n_2]$$

(This is not usually a big deal in 2-D)

- A system is **stable** if

$$\sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} |h[n_1, n_2]| < \infty$$

Difference equations for causal systems

- **In 1 dimension:**

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{l=0}^{M} b_l x[n-l]$$

- **In 2 dimensions:**

$$\sum_{k_1=0}^{N_1} \sum_{k_2=0}^{N_2} a_{k_1,k_2} y[n_1-k_1, n_2-k_2] = \sum_{l_1=0}^{N_2} \sum_{l_2=0}^{N_2} b_{l_1,l_2} x[n_1-l_1, n_2-l_2]$$

The 2-D discrete-time Fourier transform

- **In 1-D we have**

$$e^{j\omega n} \longrightarrow \boxed{T} \longrightarrow H(e^{j\omega})e^{j\omega n}$$

- **In 2-D we have**

$$e^{j\omega_1 n_1} e^{j\omega_2 n_2} \longrightarrow \boxed{T} \longrightarrow H(e^{j\omega_1}, e^{j\omega_2})e^{j\omega_1 n_1} e^{j\omega_2 n_2}$$

The 2-D discrete-time Fourier transform

$$e^{j\omega_1 n_1} e^{j\omega_2 n_2} \longrightarrow \boxed{T} \longrightarrow H(e^{j\omega_1}, e^{j\omega_2}) e^{j\omega_1 n_1} e^{j\omega_2 n_2}$$

- **From the convolution sum definition we can obtain**

$$H\left(e^{j\omega_1}, e^{j\omega_2}\right) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} h[n_1, n_2] e^{-j\omega_1 n_1} e^{-j\omega_2 n_2}$$

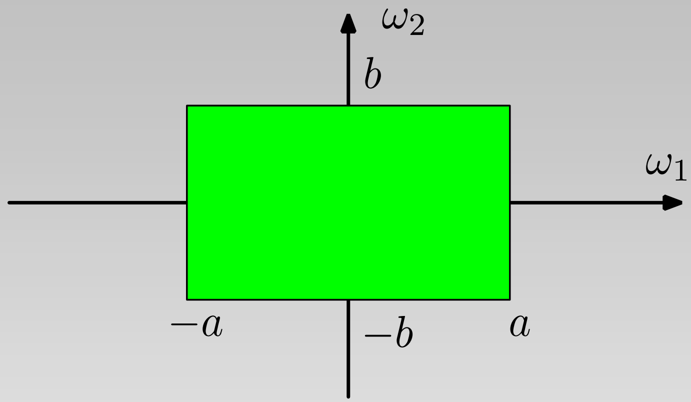
and

$$h[n_1, n_2] = \frac{1}{(2\pi)^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} H(e^{j\omega_1}, e^{j\omega_2}) e^{j\omega_1 n_1} e^{j\omega_2 n_2} d\omega_1 d\omega_2$$

The 2-D discrete-time Fourier transform

- **Comments:**
  - $H(e^{j\omega_1}, e^{j\omega_2})$ is periodic with period 2π in ω_1 and ω_2
  - If $h[n_1, n_2]$ is separable, $H(e^{j\omega_1}, e^{j\omega_2})$ is as well, and computing the 2-D DTFT becomes just a matter of computing the product of two 1-D DTFTs
  - Convolution in time ⇔ multiplication in frequency

An example of frequency response

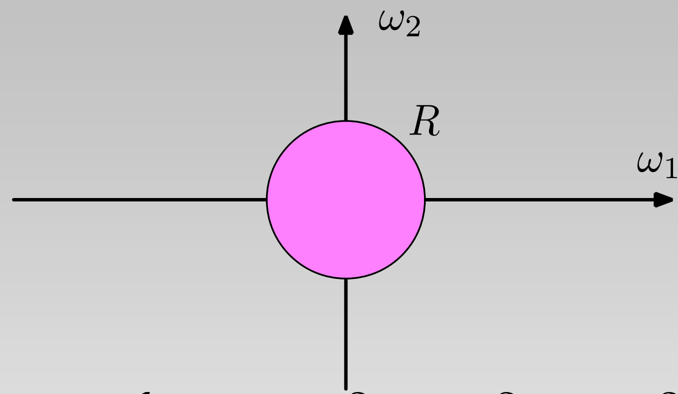


$$H(e^{j\omega_1}, e^{j\omega_2}) = \begin{cases} 1, & |\omega_1| \leq a, |\omega_2| \leq b, \\ 0, & \text{otherwise} \end{cases}$$

- **The DTFT is separable and**

$$h[n_1, n_2] = \frac{\sin(an_1)}{\pi n_1} \frac{\sin(bn_2)}{\pi n_2}$$

A second example of a frequency response



$$H(e^{j\omega_1}, e^{j\omega_2}) = \begin{cases} 1, & \omega_1^2 + \omega_2^2 \leq R^2 \\ 0, & \text{otherwise} \end{cases}$$

- **This DTFT is not separable! It can be shown that**

$$h[n_1, n_2] = \frac{\omega_c}{2\pi\sqrt{n_1^2 + n_2^2}} J_1\left(\omega_c\sqrt{n_1^2 + n_2^2}\right)$$

- **Note: While this DTFT is not separable, it IS rotation invariant in both time and frequency**

2-dimensional *z*-transforms

- In a similar fashion to the 1-D case, we build up z-transforms by modeling functions in space as linear combinations of the function

$$z_1^{n_1} z_2^{n_2} = \left(r_1 e^{j\omega_1}\right)^{n_1} \left(r_2 e^{j\omega_2}\right)^{n_2}$$

- In particular,

$$H(z_1, z_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} h[n_1, n_2] z_1^{-n_1} z_2^{-n_2}$$

and

$$h[n_1, n_2] = \frac{1}{(2\pi j)^2} \int_{C_1} \int_{C_2} H(z_1, z_2) z_1^{n_1-1} z_2^{n_2-1} dz_1 dz_2$$

An example 2-D *z*-transform

- **Consider the simple space function**

$$x[n_1, n_2] = \begin{cases} K^{n_1}, & n_1 = n_2 \text{ and } n_1 \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

- **The corresponding *z*-transform is**

$$X(z_1, z_2) = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} K^{n_1} \delta[n_1 - n_2] z_1^{-n_1} z_2^{-n_2} u[n_1, n_2]$$

$$= \sum_{n_1=0}^{\infty} K^{n_1} (z_1 z_2)^{-n_1} = \frac{1}{1 - K z_1^{-1} z_2^{-1}}$$

which converges for $|K z_1^{-1} z_2^{-1}| < 1$

The fundamental curse of 2D-DSP

$$X(z_1, z_2) = \frac{1}{1 - K z_1^{-1} z_2^{-1}}$$

- **Comments: no poles and zeros (!), so**
  – No easy tests for stability
  – No parallel or cascade implementations
  – No Parks-McClellan algorithm
  – etc. etc.

The 2-dimensional discrete Fourier transform

- **The 2D-DFT is derived in a fashion similar to how it had been in 1-D. Specifically:**

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

and

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

- **Comments:**
  - Multiplying coefficients in frequency corresponds to a 2-D circular ("toroidal") convolution in space
  - Overlap-add, overlap-save algorithms are still valid

Computing the 2-D DFT

- **Again, the 2D-DF is**

$$H[k_1,k_2] = \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} h[n_1,n_2]W_{N_1}^{k_1n_1}W_{N_2}^{k_2n_2}$$

- **This can be rewritten as**

$$H[k_1,k_2] = \sum_{n_1=0}^{N_1-1} W_{N_1}^{k_1n_1} \sum_{n_2=0}^{N_2-1} h[n_1,n_2]W_{N_2}^{k_2n_2}$$

- **Let** $$\sum_{n_2=0}^{N_2-1} h[n_1,n_2]W_{N_2}^{k_2n_2} \equiv g[n_1,k_2]$$

then $$H[k_1,k_2] = \sum_{n_1=0}^{N_1-1} g[n_1,k_2]W_{N_1}^{k_1n_1}$$

Computing the 2-D DFT

- **To compute the 2-D DFT:**
  - Compute the 1-D DFT of each column and replace in the column
  - Compute the row-wise DFTs of the resulting coefficients

- **Comments:**
  - This always works… $x[n_1,n_2]$ need not be separable or anything else
  - Huge computational savings
    - » For example: let $N_1=N_2=1024\approx1000$
    - » Direct computation of 2D-DFT $\approx 10^{12}$ complex mults!
    - » Using the row/column shortcut we have $\approx 2\ 10^9$ complex mults
    - » Using the shortcut and FFT algorithms leaves only 10^7 complex mults

Some summary observations about 2D-DSP

- **Many things are obvious extensions of 1-D DSP**
  - Linearity and shift invariance
  - Convolution sum, difference equations
  - 2-D DTFTs
  - 2-D DFTs

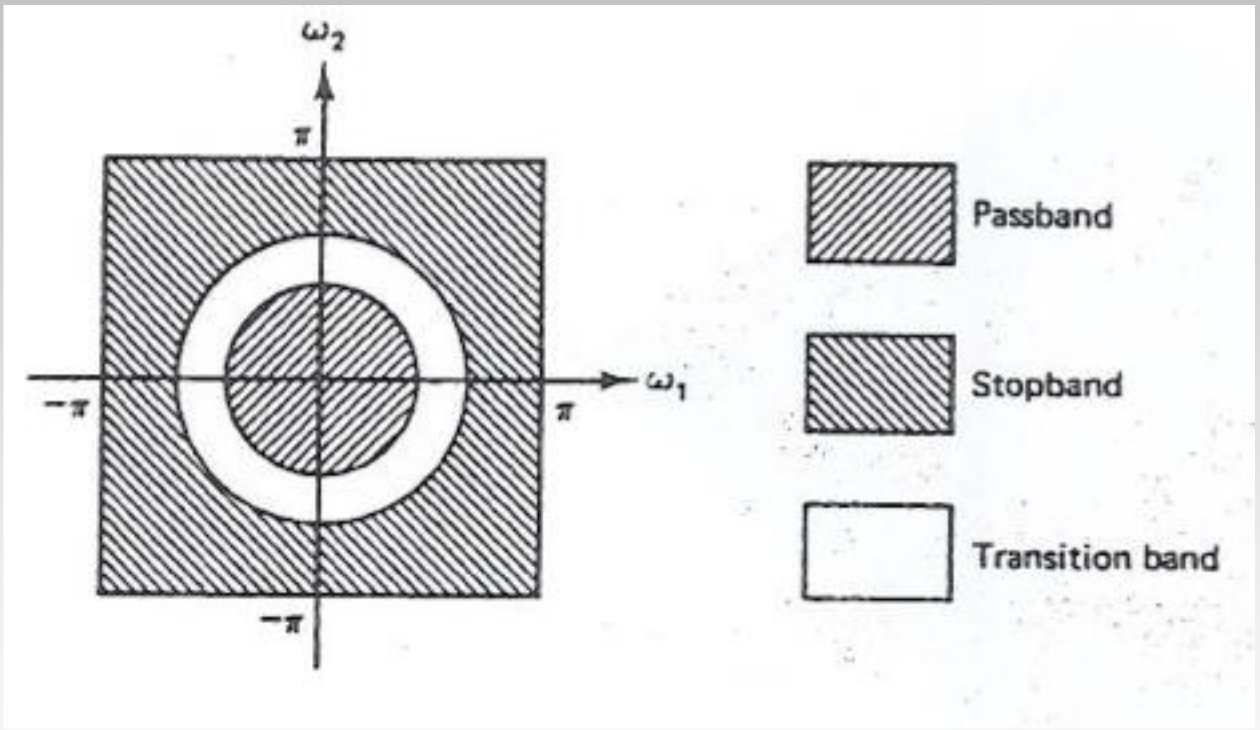
- **Some things are fundamentally different:**
  - 2-D *z*-transforms
    - » No poles and zeros as we know them

Some summary observations about 2D-DSP

- **Some other things to keep in mind:**
  - Tradeoff between separability and rotation invariance
  - Physical significance of 2-D complex exponentials
  - Efficiencies provided by separability
  - Efficient computation of the 2-D DFT

- **Next topic of discussion:**
  - 2-D discrete-space filter design

The general 2-D filter design problem



Designing digital filters in two dimensions

- **We will focus on FIR designs for now because of stability issues with IIR filters (despite computational efficiencies)**

- **Major FIR techniques:**
  – Window designs
  – Frequency-sampled design
  – Parks-McClellan algorithm

- **For the most part, 2-D FIR filters are designed by using successful 1-D techniques and extending to 2-D**
- **Zero-phase filtering is much more important in 2D than 1D**

2-dimensional FIR design using windows

- **Let** $h[n_1,n_2] = h_d[n_1,n_2]w[n_1,n_2]$

- **Separable window approach:**

$$w[n_1,n_2] = w[n_1]w[n_2]$$

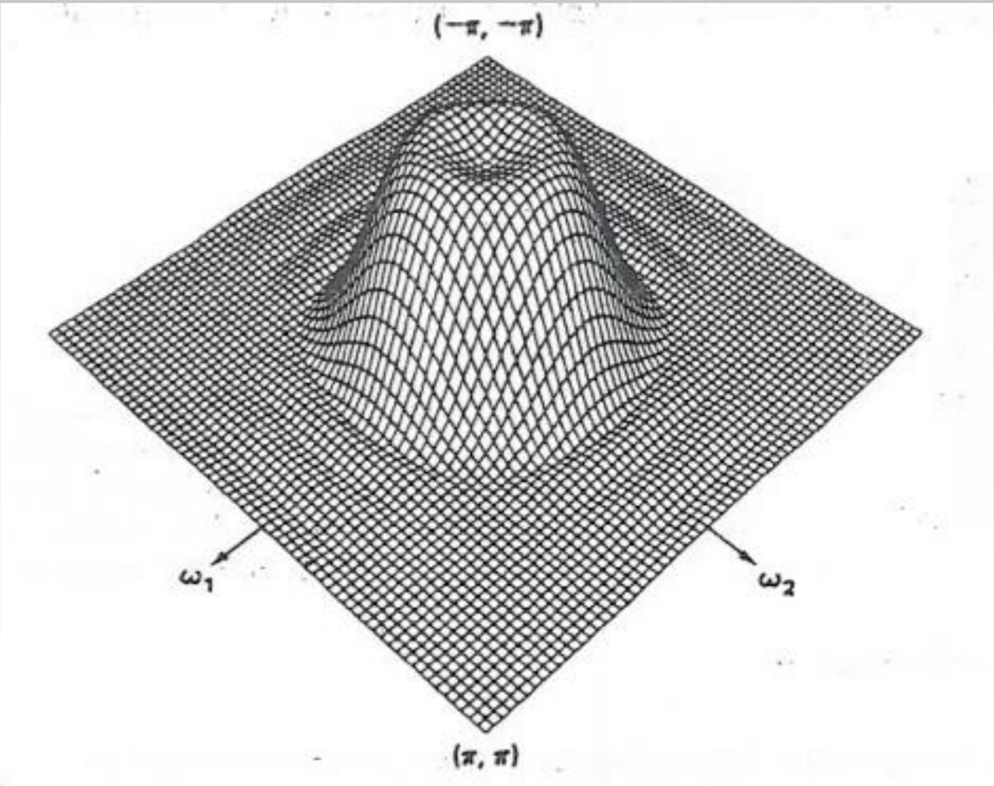
- **Rotation-invariant window approach:**

$$w[n_1,n_2] = w\left[\sqrt{n_1^2+n_2^2}\right]$$

where *w[n]* is a successful 1-D window, usually a Kaiser window

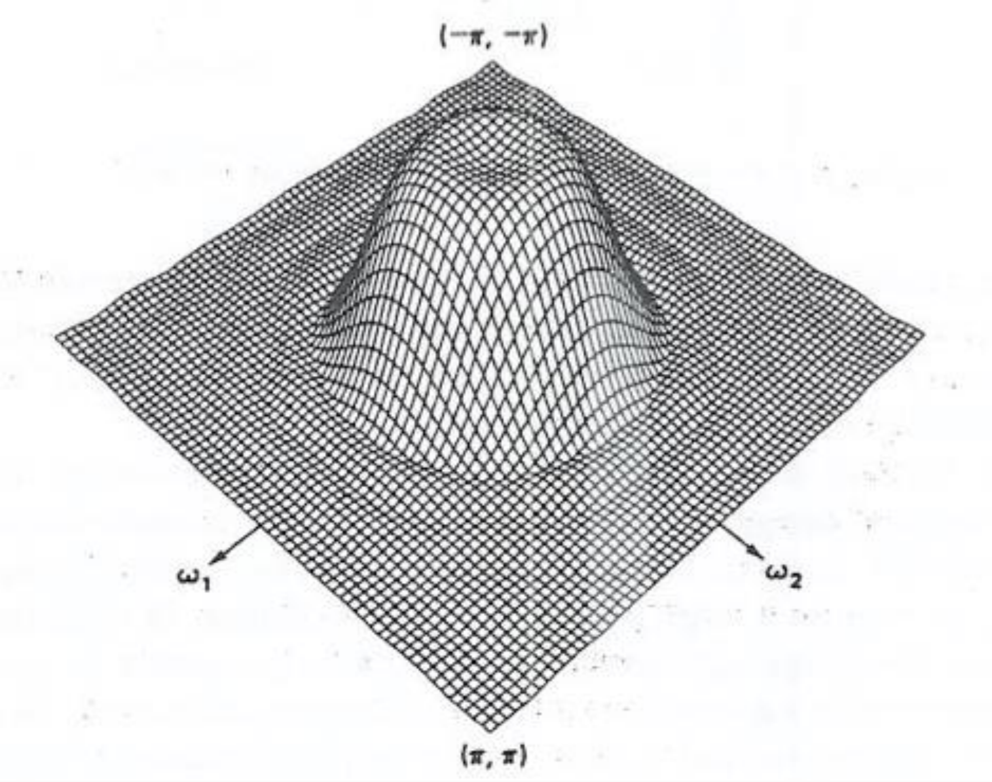
A lowpass example using a separable window

- **Separable 9x9 Kaiser window, $\omega_c = 0.4\pi$**



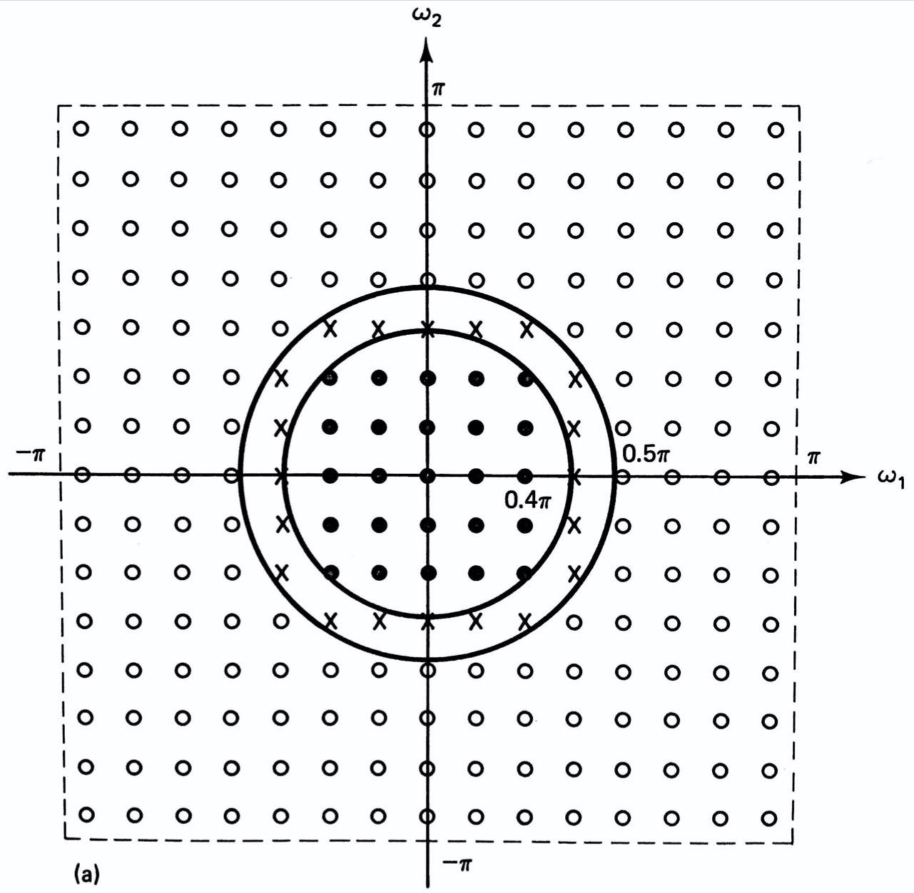
LPF example with a rotation-invariant window

- **Rotation-invariant 9x9 Kaiser window, $\omega_c = 0.4\pi$**



The frequency-sampling approach

- 15 x 15-point design using frequency sampling

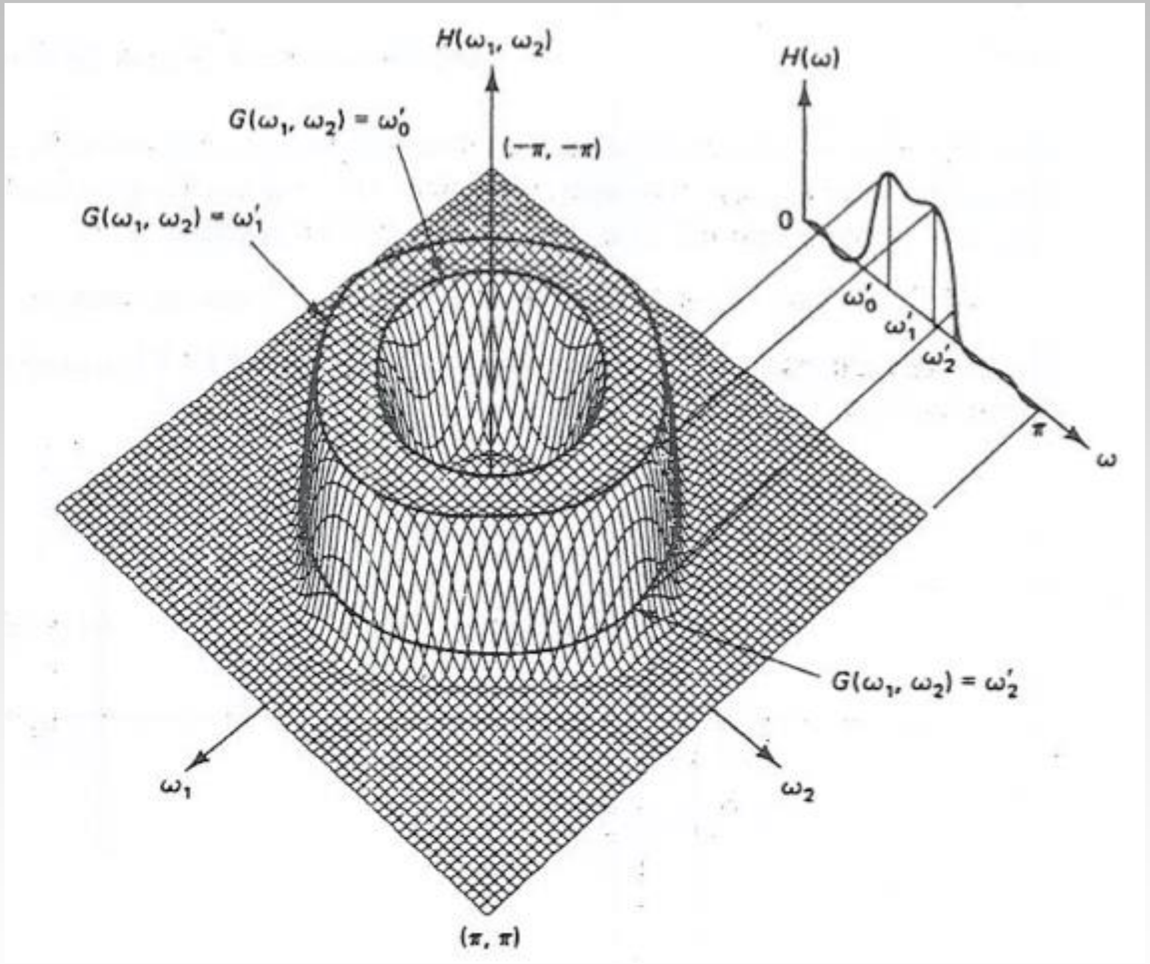


Optimum 2-D FIR filters

- **General approach:**
  - Design optimal 1-D filter using Parks-McClellan algorithm
  - Transform from 1-D to 2-D using method also developed in McClellan thesis (!)

$$H(\omega_1,\omega_2) = H(\omega)\,\big|_{\omega=G(\omega_1,\omega_2)}$$

The general approach



The McClellan transformation

- **As you will recall, for a zero-phase Type I FIR filter we have**

$$H(\omega) = \sum_{n=-N}^{N} h[n]e^{-j\omega n} = h[0] + \sum_{n=1}^{N} 2h[n]cos(\omega n)$$

$$= \sum_{n=0}^{N} a[n]\cos(\omega n) = \sum_{n=0}^{N} b[n](\cos(\omega))^n$$

- **The 2-D response is obtained by**

$$H(\omega_1, \omega_2) = H(\omega)|_{\cos(\omega)=T(\omega_1,\omega_2)} = \sum_{n=0}^{N} b[n]T[\omega_1, \omega_2)]^n$$

The 2-D to 2-D transform

- **From before,**

$$H(\omega_1, \omega_2) = H(\omega)|_{\cos(\omega)=T(\omega_1,\omega_2)} = \sum_{n=0}^{N} b[n]T[\omega_1, \omega_2)]^n$$

- **This can be expressed as**

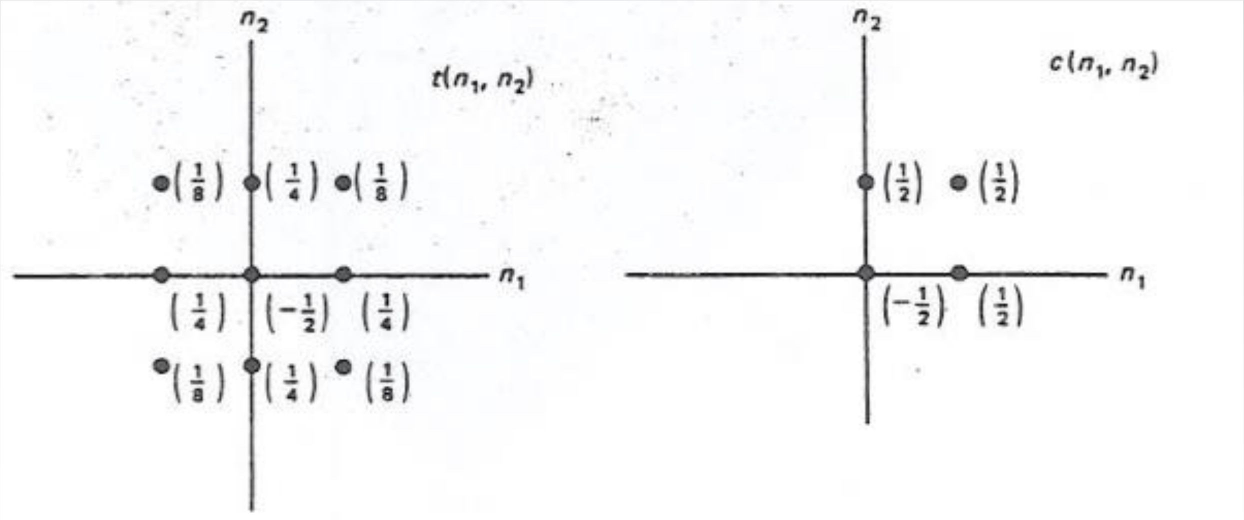
$$T(\omega_1, \omega_2) = \sum_{n_1}\sum_{n_2} t[n_1, n_2]e^{-j\omega_1 n_1}e^{-k\omega_2 n2}$$
$$= \sum_{n_1}\sum_{n_2} c[n_1, n_2]\cos(\omega_1 n_1)\cos(\omega_2 n_2)$$

A particularly common special case

- **An example often used in practice:**

$$T(\omega_1, \omega_2) = \frac{1}{2}\cos(\omega_1) + \frac{1}{2}\cos(\omega_2) + \frac{1}{2}\cos(\omega_1\omega_2) - \frac{1}{2}$$

- **The corresponding sequences *t[n₁,n₂]* and *c[n₁,n₂]*:**



Equivalent contours in 1-D and 2-D

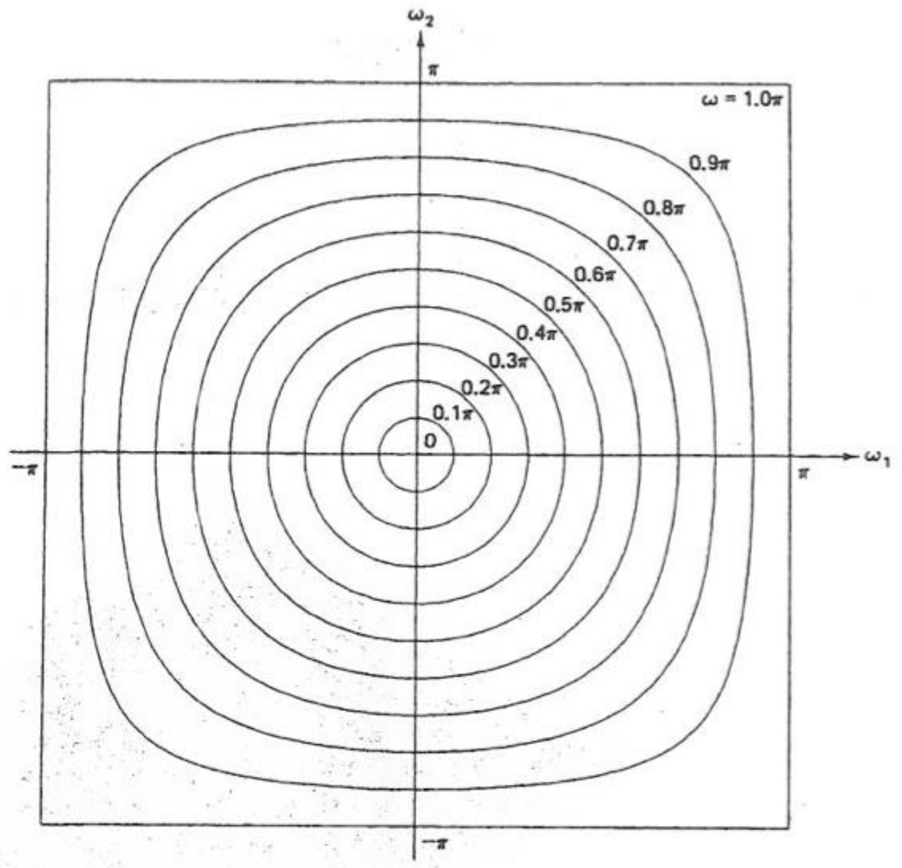
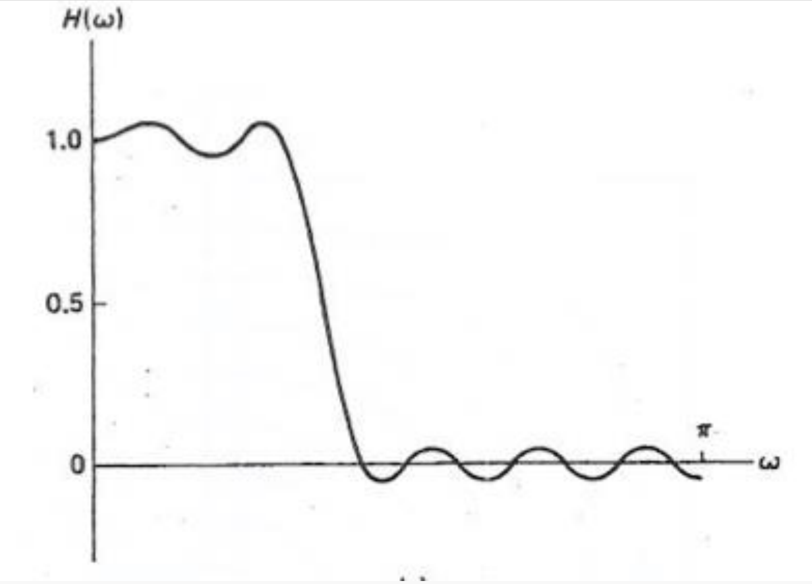


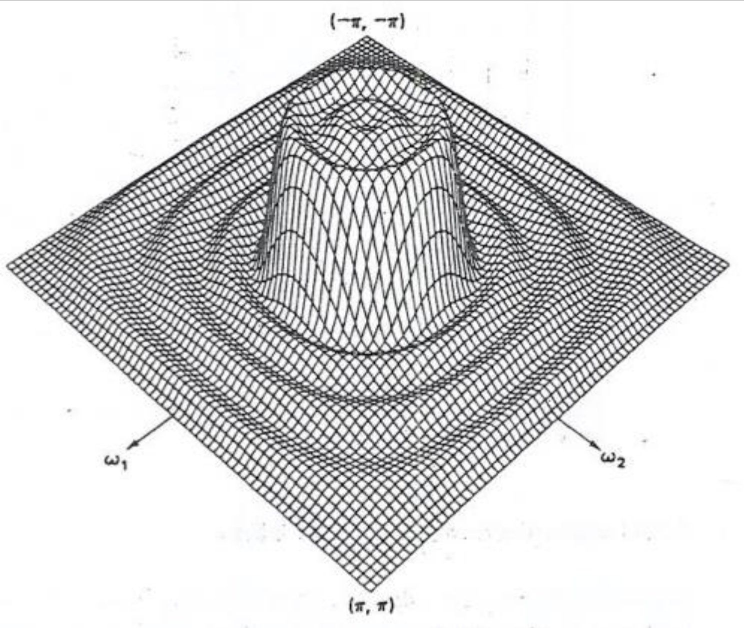
Figure 7.44 The contours obtained by $\cos\,\omega = T(\omega_1, \omega_2)$ for $\omega = 0, \pi/10, \ldots, \pi$ for $T(\omega_1, \omega_2)$ given by Eq. (7.84).

An example frequency response

- 1-D filter:



- 2-D filter

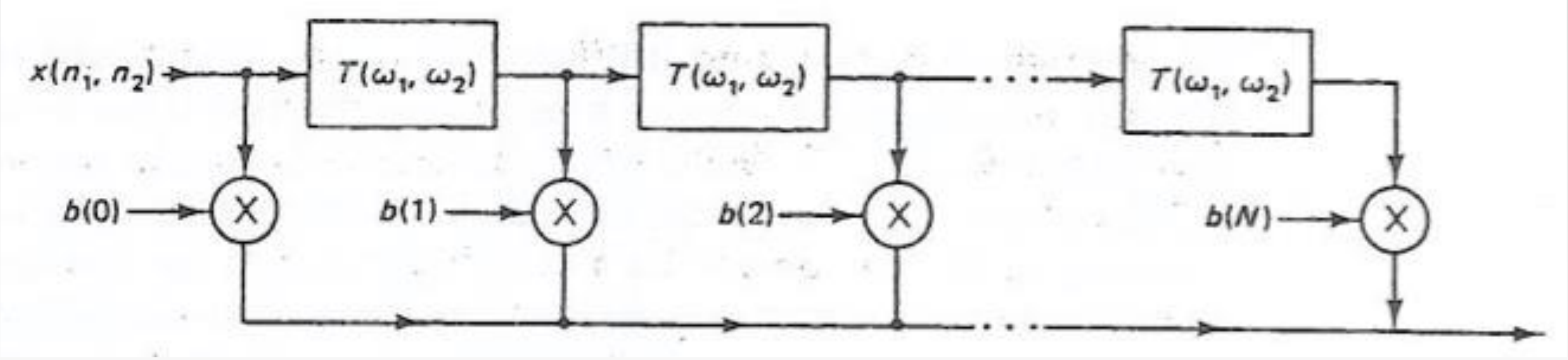


Implementation based on 1-D to 2-D transforms

- **The transfer function:**

$$H(\omega_1, \omega_2) = \sum_{n=0}^{N} b[n] \left[T(\omega_1, \omega_2)\right]^n$$

- **Efficient implementation:**



- **Comment: overlap-add, FFTs, etc. can be used here as well**

Summary: filter design

- **We reviewed FIR filter design using the three methods discussed in 18-491:**
  - Window design with separable or rotation-invariant Kaiser windows
  - Frequency-sampling design, varying symmetric points together
  - Parks-McClellan design, using the McClellan 1-D to 2-D transformation
  - The McClellan transformation implies its own efficient implementation