# Summary of Lecture 4

- We learnt how to generate random images.
- We learnt about histogram matching which enabled us to "match" the histogram of a given image to another image's histogram.
- We learnt how to calculate the "inverses" of discrete functions.
- We learnt about quantization, simple uniform quantization and companding.
  - Calculating errors.
  - MSQE.
  - Choosing thresholds to minimize MSQE.

# Quantization

- $t_n \in \{0, 1, \ldots, 255\}$ a sequence of thresholds $(n = 0, \ldots, P-1)$.
- $P$ "half-open, discrete intervals" $R_n = [t_n, t_{n+1})$ $(t_0 = 0,\ t_P = 256)$.
- $r_n \in R_n$ the reproduction level of the interval $R_n$.
- Quantizer:

$$Q(l) = \{r_k | l \in R_k,\ k = 0, \ldots, P-1\} \tag{1}$$

i.e., $l \in R_k \Leftrightarrow Q(l) = r_k$.

# Designing Good Quantizers

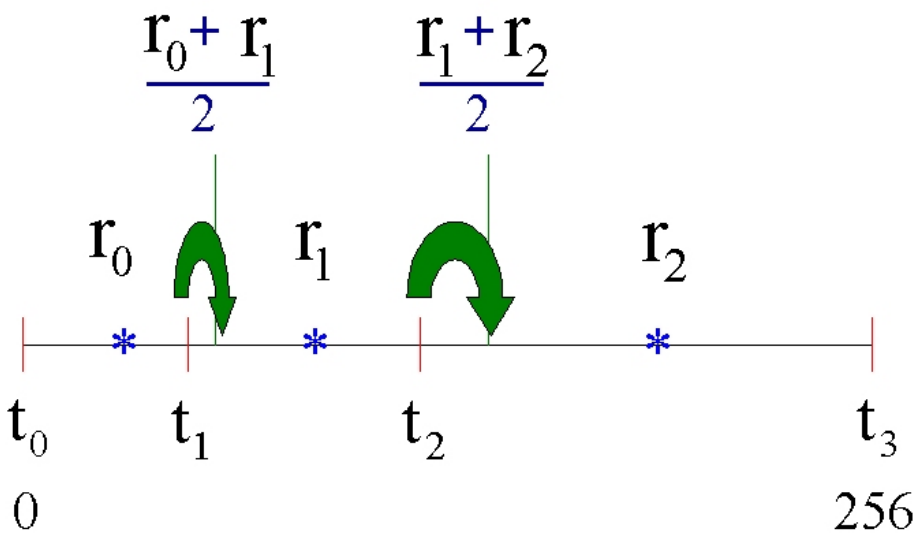
$$Q(l) = \{r_k | l \in R_k, \ k = 0, \ldots, P-1\} \quad (2)$$

$$\text{MSQE} = \sum_{l=0}^{255} (l - Q(l))^2 p_A(l) \quad (3)$$

Assuming $P$ is fixed:

- Around ranges of $l$ where $p_A(l)$ is large, a good quantizer should have many small $R_n$, i.e., since we can have at most $P$ discrete intervals, most of these intervals should be around ranges of $l$ where $p_A(l)$ is large.
- Equivalently, a good quantizer should not "waste" many reproduction levels around ranges of $l$ where $p_A(l)$ is *small*.

# Designing the Thresholds for Given Reproduction Levels




- Assume $P$ and $r_n$ are given.
- For MSQE optimality

$$t_n = \text{round}\left(\frac{r_{n-1} + r_n}{2}\right) \tag{4}$$

# Designing the Reproduction Levels for Given Thresholds

- Assume $P$ and $t_n$ are given.
- Consider $R_n = [t_n, t_{n+1})$.
- How can we choose $r_n \in R_n$ to minimize MSQE?
- Equation 3 can be written as:

$$\begin{aligned}
\text{MSQE} &= \sum_{l=0}^{255} (l - Q(l))^2 p_A(l) \\
&= \sum_{m=0}^{P-1} \sum_{l=t_m}^{t_{m+1}-1} (l - Q(l))^2 p_A(l) \\
&= \sum_{m=0}^{P-1} \sum_{l=t_m}^{t_{m+1}-1} (l - r_m)^2 p_A(l) \\
&= \sum_{m=0, m\neq n}^{P-1} \sum_{l=t_m}^{t_{m+1}-1} (l - r_m)^2 p_A(l) + \sum_{l=t_n}^{t_{n+1}-1} (l - r_n)^2 p_A(l)
\end{aligned}$$

- Choose $r_n$ to minimize:

$$\sum_{l=t_n}^{t_{n+1}-1} (l - r_n)^2 p_A(l) \tag{5}$$

# Designing the Reproduction Levels contd.

- Choose $r_n$ to minimize:

$$\sum_{l=t_n}^{t_{n+1}-1} (l - r_n)^2 p_A(l)$$

- Taking the derivative with respect to $r_n$ and equating the result to $0$ yields:

$$-\sum_{l=t_n}^{t_{n+1}-1} 2(l - r_n) p_A(l) = 0$$

$$r_n = \text{round}\left(\frac{\sum_{l=t_n}^{t_{n+1}-1} l p_A(l)}{\sum_{k=t_n}^{t_{n+1}-1} p_A(k)}\right) \tag{6}$$

# MSQE Optimal Lloyd-Max Quantizer

Algorithm: Find the MSQE optimal quantizer in *iterations* $v = 0, 1, \ldots$. Start with $v = 0$ and a given set of reproduction levels $r_n^0$. Let $\epsilon > 0$ be a *small* number.

1. $v \rightarrow v + 1$.
2. Calculate MSQE optimal $t_n^v$ via:

$$t_n^v = \text{round}\left(\frac{r_{n-1}^{v-1} + r_n^{v-1}}{2}\right) \tag{7}$$

3. Calculate MSQE optimal $r_n^v$ via:

$$r_n^v = \text{round}\left(\frac{\sum_{l=t_n^v}^{t_{n+1}^v - 1} l p_A(l)}{\sum_{k=t_n^v}^{t_{n+1}^v - 1} p_A(k)}\right) \tag{8}$$

4. Calculate MSQE via:

$$e^v = \sum_{m=0}^{P-1} \sum_{l=t_m^v}^{t_{m+1}^v - 1} (l - r_m^v)^2 p_A(l) \tag{9}$$

5. If $e^{v-1} - e^v < \epsilon$ terminate (quantizer designed) else goto 1.

# Properties

- $e^{v-1} - e^v \geq 0$, the MSQE *is non-increasing with* $v$. The algorithm is *guaranteed* to converge.
- The MSQE optimal final quantizer satisfies the optimality conditions for the thresholds and reproduction levels.
- The MSQE optimal final quantizer is *locally optimal*, i.e., the final quantizer (and hence $e^v$) depends on the initial $r_n^0$ that we started with.
- The final quantizer must be implemented in the *general way* as described in Lecture 4.
- A good way to choose the $r_n^0$ is by making them the reproduction levels of a companding quantizer.

# Example

Companded, P=64, MSQE=1.14

Lloyd-Max, P=64, MSQE=1.03

Companded, P=32, MSQE=3.80

Lloyd-Max, P=32, MSQE=3.65

# Example - contd.

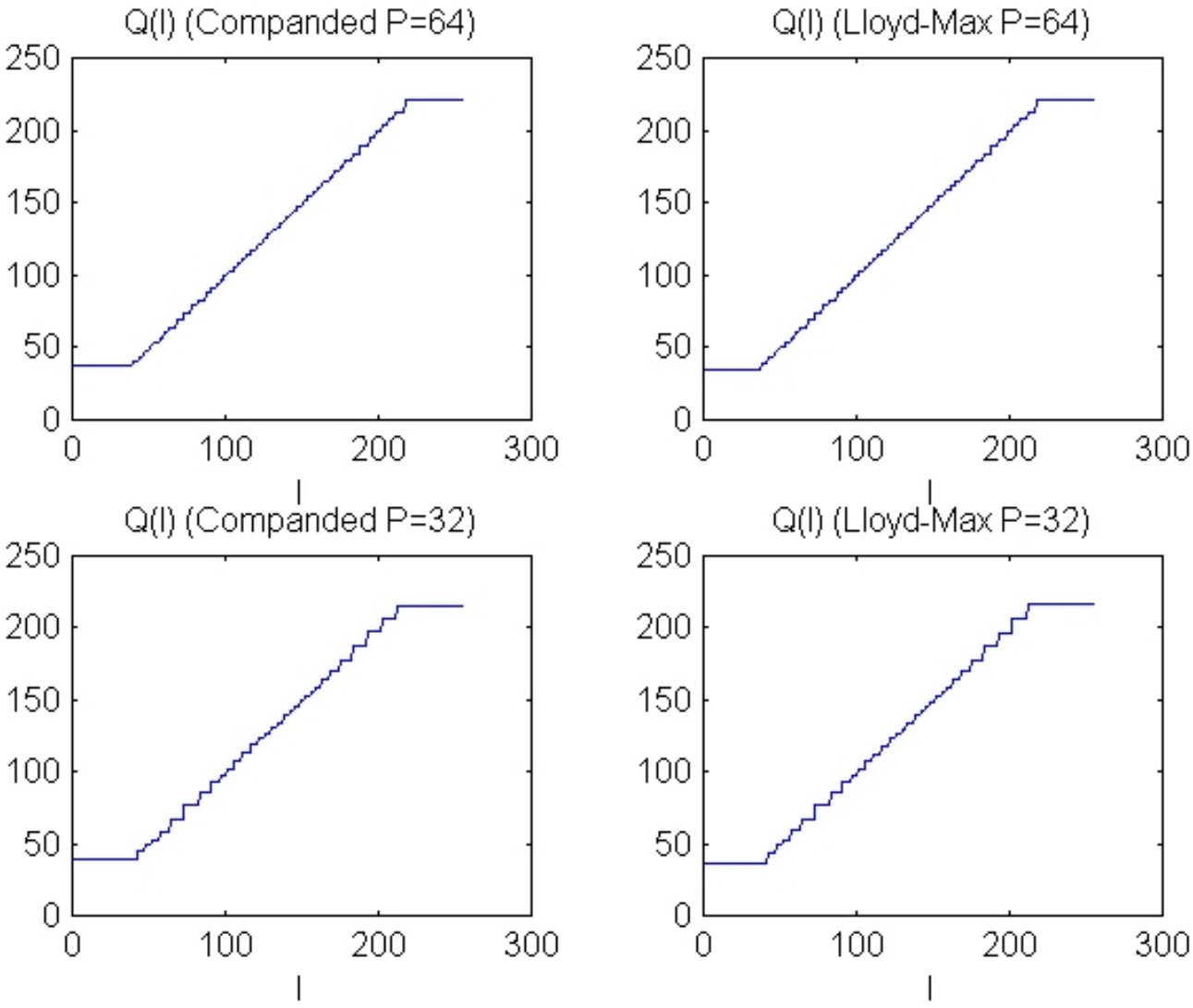
Companded, P=16, MSQE=13.15

Lloyd-Max, P=16, MSQE=12.36

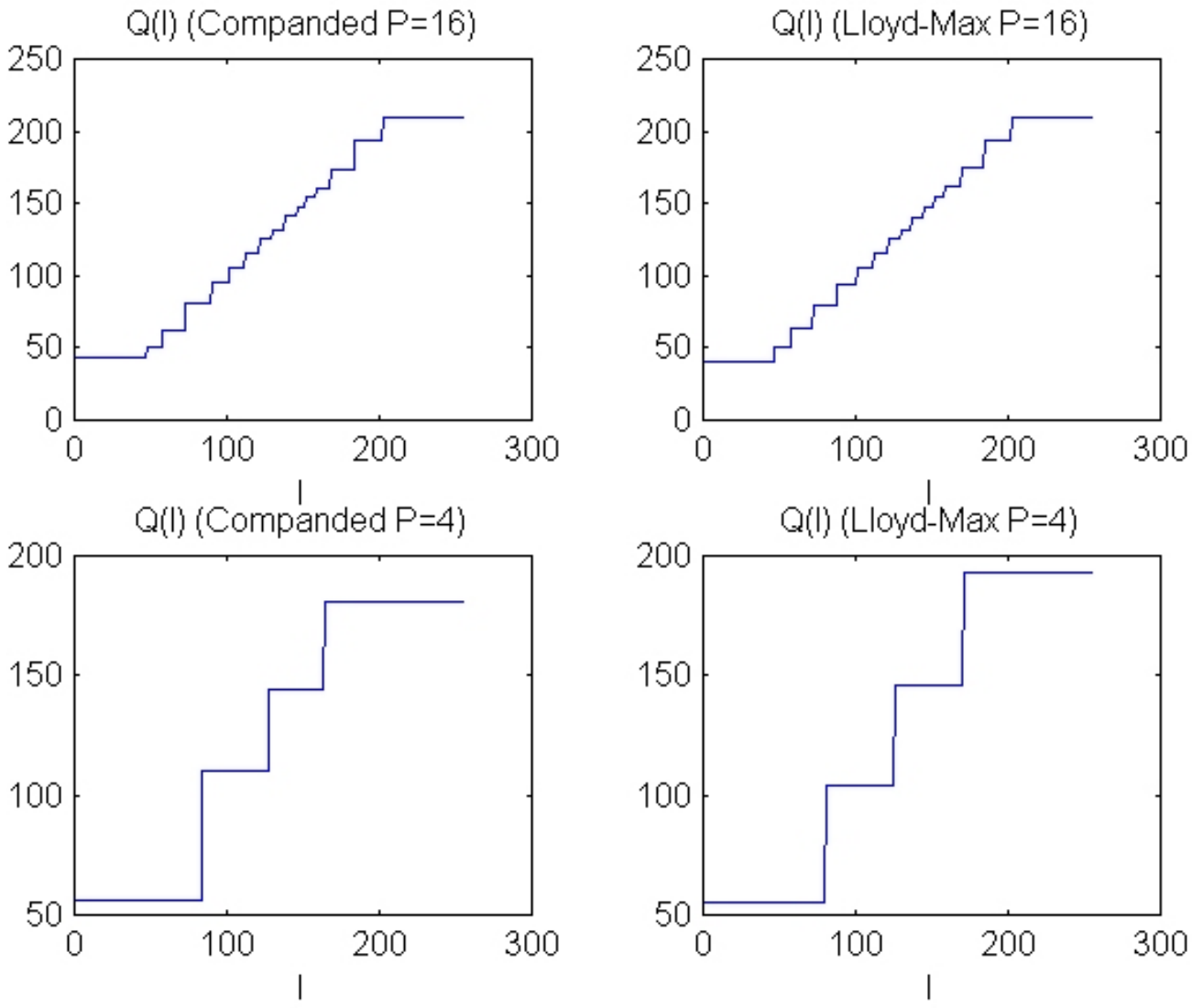
Companded, P=4, MSQE=186.04

Lloyd-Max, P=4, MSQE=161.91

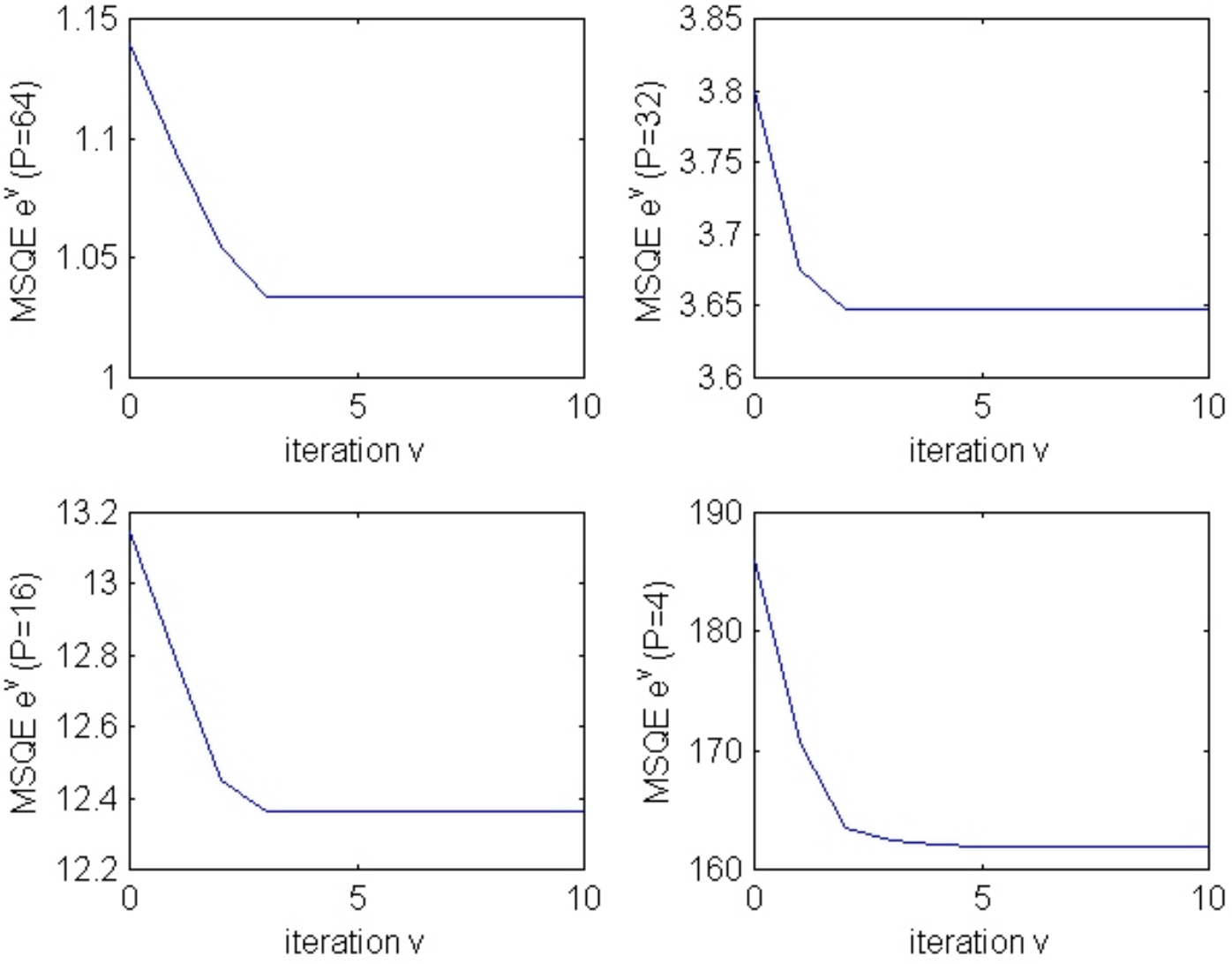
# Example - contd.



# Example - contd.



# Example - contd.



# Systems

- Image processing operations can be modeled by utilizing system theory.
- For now assume $\mathbf{A}$ and $\mathbf{B}$ are general two-dimensional sequences.
- Consider $\mathbf{A} \overset{\mathcal{S}}{\Rightarrow} \mathbf{B}$.
  - $\mathcal{S}$ is a system which "converts" the input $\mathbf{A}$ into the output $\mathbf{B}$.
  - The output $\mathbf{B}$ *depends* on the input $\mathbf{A}$, i.e., in general different inputs *may* give rise to different outputs.
  - The system $\mathcal{S}$ is characterized by its input-output relationship $(\mathcal{H})$:

$$\mathbf{B} = \mathcal{H}(\mathbf{A}) \tag{10}$$

We will also write:

$$B(m,n) = \mathcal{H}(\mathbf{A}(i,j)), \qquad m,n,i,j \in \{-\infty,\ldots,-1,0,1,\ldots,+\infty\}$$

# Linear Systems

- Let $a_1$ and $a_2$ be constants.
- A linear system $\mathcal{S}$ is a system such that

$$\mathcal{H}(a_1\mathbf{A_1} + a_2\mathbf{A_2}) \;=\; a_1\mathcal{H}(\mathbf{A_1}) + a_2\mathcal{H}(\mathbf{A_2}) \tag{11}$$

  for all $a_1,\ a_2,\ \mathbf{A_1},\ \mathbf{A_2}$.

- A large number of image processing/formation operations can be *modeled* by linear systems.

# Linear Shift Invariant (LSI) Systems

- A shift invariant system $\mathcal{S}$ is a system such that if

$$B(m,n) = \mathcal{H}(A(i,j))$$

then

$$B(m-i_0, n-j_0) = \mathcal{H}(A(i-i_0, j-j_0)) \tag{12}$$

for all $\mathbf{A}$ and $i_0, j_0 \in \{-\infty, \ldots, -1, 0, 1, \ldots, +\infty\}$.

- A linear shift invariant (LSI) system $\mathcal{S}$ is a linear system that is *also* shift invariant.
- Many image processing/formation operations can be *modeled* by linear shift invariant systems.

# Impulse Representation of 2D Sequences/Images

$$A(i,j) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k, j-l) \tag{13}$$

where

$$\delta(i,j) = \begin{cases} 1, & i = j = 0 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

is the Kronecker delta function or discrete-time impulse function,
$\delta(i,j) = \delta(i)\delta(j)$.

(This is not to be confused with the Dirac delta function or continuous-time impulse function $\delta(x,y)$).

# Impulse Response for LSI Systems and Convolution

Assume $\mathcal{S}$ is a linear shift invariant system with input-output relationship $\mathcal{H}$. Using Equation 13:

$$\begin{aligned}\mathcal{H}(A(i,j)) &= \mathcal{H}(\sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k,j-l)) \\ &= \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)\mathcal{H}(\delta(i-k,j-l)) \qquad (15)\end{aligned}$$

Let $h(i,j) = \mathcal{H}(\delta(i,j))$ denote the impulse response of the system $\mathcal{S}$. Then:

$$\mathcal{H}(\delta(i-k,j-l)) = h(i-k,j-l)$$

and we have:

$$\mathcal{H}(A(i,j)) = \sum_{k=-\infty}^{+\infty}\sum_{l=-\infty}^{+\infty} A(k,l)h(i-k,j-l) \qquad (16)$$

which is called the convolution sum.

- This holds for all $\mathbf{A}$, i.e., everything about the LSI system $\mathcal{S}$ is "in" $h(i,j)$.

# Convolution

$$B(i,j) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)h(i-k,j-l)$$

We will also write $\mathbf{B} = \mathbf{A} \otimes \mathbf{h}$.

Let $k' = i-k,\ \ l' = j-l$.

$$\begin{aligned} \mathbf{A} \otimes \mathbf{C} &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)C(i-k,j-l) \\ &= \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} A(i-k',j-l')C(k',l') \\ &= \mathbf{C} \otimes \mathbf{A} \end{aligned}$$

In particular, $\delta \otimes \mathbf{h} = \mathbf{h} \otimes \delta = \mathbf{h}$ and for all $\mathbf{A}$:

$$\begin{aligned} \mathbf{A} \otimes \delta &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l)\delta(i-k,j-l) \\ &= \mathbf{A} \end{aligned} \tag{17}$$

# Convolution - contd.

- In practice we will be interested in the convolution of two dimensional sequences of finite extent, for e.g.,

$$A(i,j) \quad \begin{cases} \neq 0, & 0 \le i \le N_1 - 1,\ 0 \le j \le M_1 - 1 \\ = 0 & \text{otherwise} \end{cases}$$

$$B(i,j) \quad \begin{cases} \neq 0, & 0 \le i \le N_2 - 1,\ 0 \le j \le M_2 - 1 \\ = 0 & \text{otherwise} \end{cases}$$

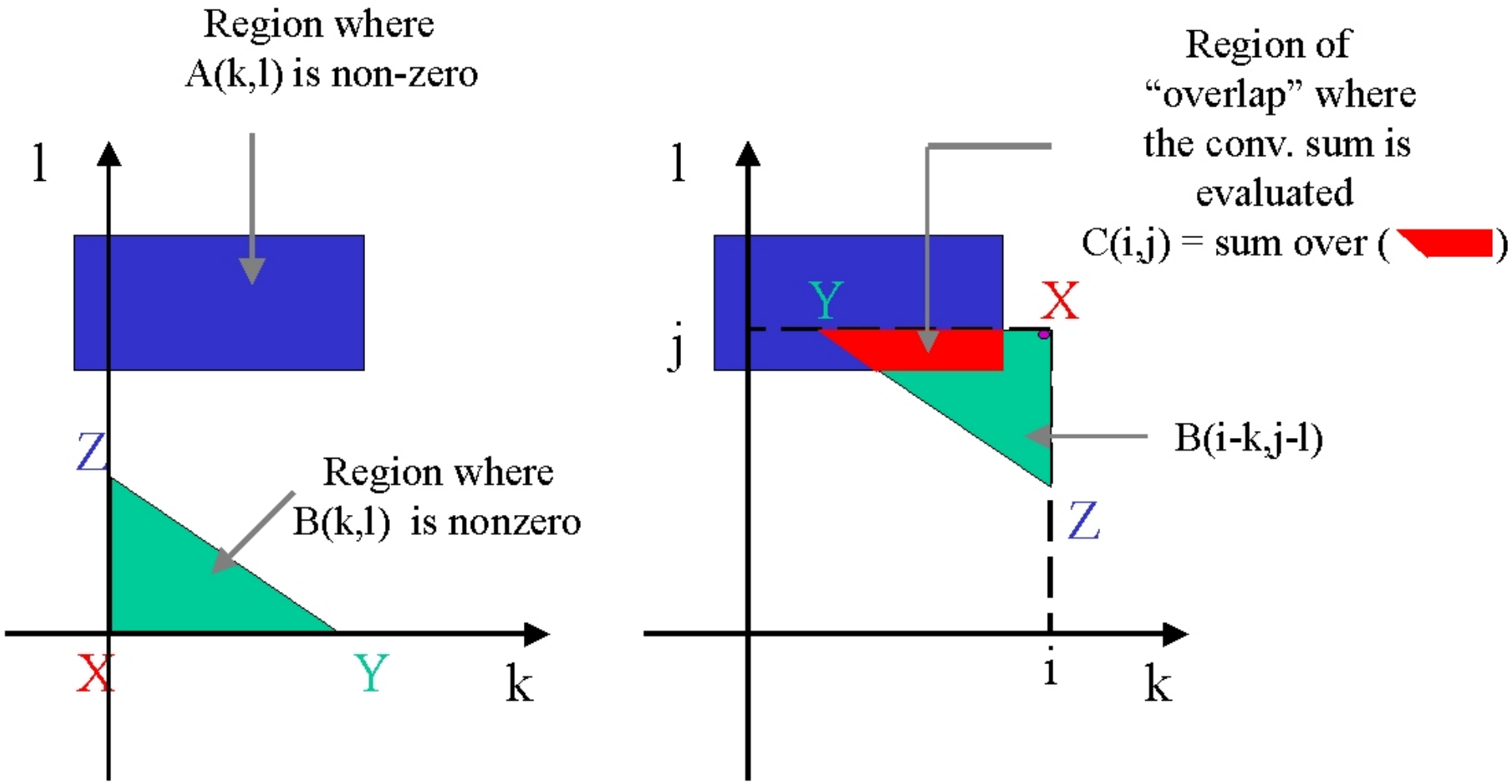
- Let $\mathbf{A}$ and $\mathbf{B}$ be as above.

$$\begin{aligned} \mathbf{A} \otimes \mathbf{B} &= \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} A(k,l) B(i-k, j-l) \\ &= \sum_{k=0}^{N_1-1} \sum_{l=0}^{M_1-1} A(k,l) B(i-k, j-l) \end{aligned} \tag{18}$$

  Note that the above convolution sum includes the term $A(N_1 - 1, M_1 - 1)B(i - (N_1 - 1), j - (M_1 - 1))$.

- Thus if $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$, $\mathbf{A}$ is $N_1 \times M_1$ and $\mathbf{B}$ is $N_2 \times M_2$, then $\mathbf{C}$ is in general $(N_1 + N_2 - 1) \times (M_1 + M_2 - 1)$.

# Example I


Region where
A(k,l) is non-zero
l
Z
Region where
B(k,l) is nonzero
X
Y
k
Region of
"overlap" where
the conv. sum is
evaluated
C(i,j) = sum over (
)
l
Y
X
j
B(i-k,j-l)
Z
i
k

# Convolution - contd.

Equation 18 can be further simplified in various ways so that sums of $0$ are avoided for implementation purposes.

- $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$. $\mathbf{A}$ $(N_1 \times M_1)$ and $\mathbf{B}$ $(N_2 \times M_2)$.
  Suppose $N_2 << N_1,\ M_2 << N_2$ and let

$$x_i = \begin{cases} i - (N_2 - 1), & i > N_2 - 1 \\ 0, & \text{otherwise} \end{cases}$$

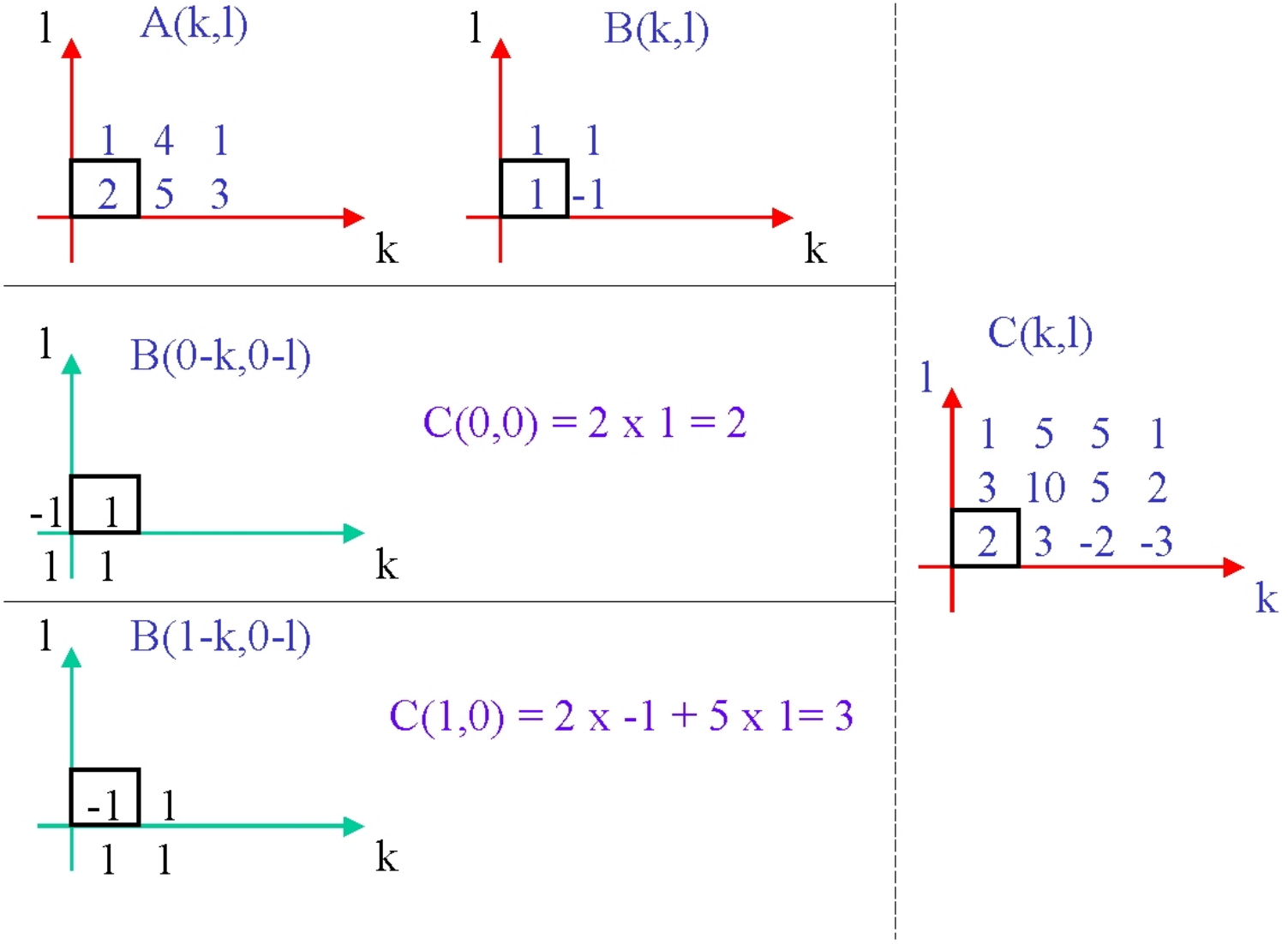
$$y_i = \begin{cases} N_1 - 1, & i > N_1 - 1 \\ i, & \text{otherwise} \end{cases}$$

  (Replace $i$ with $j$, $N$ with $M$ to get $x_j,\ y_j$).

$$C(i,j) = \sum_{k=x_i}^{y_i} \sum_{l=x_j}^{y_j} A(k,l)B(i-k, j-l) \tag{19}$$

- Note that $i = 0, \ldots, N_1 + N_2 - 1 - 1$ and $j = 0, \ldots, M_1 + M_2 - 1 - 1$ for this *special case* of finite extent sequences.

# Example II


A(k,l)
1 4 1
2 5 3
B(k,l)
1 1
1 -1
B(0-k,0-l)
C(0,0) = 2 x 1 = 2
B(1-k,0-l)
C(1,0) = 2 x -1 + 5 x 1= 3
C(k,l)
1 5 5 1
3 10 5 2
2 3 -2 -3

# Summary

- In this lecture we learnt how to pick the reproduction levels for the given thresholds.
- We learnt how to design MSQE optimal quantizers.
- We reviewed linear systems, linear shift invariant systems and the convolution sum.
- Please read pages 11-19 from the textbook.

# Homework V

1. Implement the Lloyd-Max quantizer for your image. Do everything I did between pages 9-13. Show all results as well as your original image and its sample probability mass function.

2. Convolve the 2-d sequences with nonzero portions shown, i.e., obtain $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$. Show your computations graphically for at least 5 different values of $\mathbf{C}$ similar to the earlier example.

$A(i,j)=$

| | $j=0$ | 1 | 2 |
|---|---|---|---|
| $i=0$ | 0 | 4 | 4 |
| 1 | $-2$ | 1 | 3 |
| 2 | 5 | 1 | 1 |

, $B(i,j)=$

| | $j=0$ | 1 | 2 |
|---|---|---|---|
| $i=0$ | $-1$ | 3 | 4 |
| 1 | 1 | 2 | 2 |
| 2 | $-3$ | 2 | 6 |

3. Implement a convolution script in matlab taking note of the computational simplifications as discussed. Convolve your image with *itself*. Normalize the result and show it as an image. Comment on the result as well as the execution time. What is the dimension of the resulting image? Now do the same using the *conv*2 command in matlab. Make sure the results of your script and *conv*2 are the same. (You can do so by calculating an error image and summing its absolute value. The result should be 0. Experiment with small images till you get it right.)

4. Let $A(i,j) =$

| | $j=0$ | 1 | 2 |
|---|---|---|---|
| $i=0$ | $-1$ | 0 | 1 |
| 1 | $-2$ | 0 | 2 |
| 2 | $-1$ | 0 | 1 |

Convolve your image with **A** (your script). Take the absolute value of the result, normalize and show. Comment on the result and the execution time. What is the size of the resulting image? Now do the same with *conv*2 as above.

## References

[1] A. K. Jain, *Fundamentals of Digital Image Processing.* Englewood Cliffs, NJ: Prentice Hall, 1989.