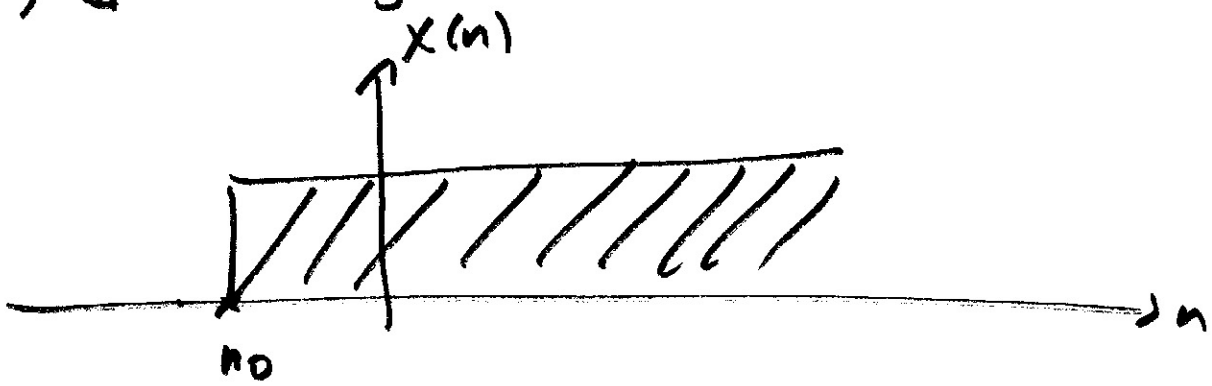
How to choose B.C. for 2D L.C.C.D.E

Feb 10/06

Approach: Generalize the approach in 1.D.

$y(n) \longleftarrow y(n-1) + x(n)$ causal



To obtain LTI system $y(n_0-1) = 0$

---

① Compute Region of support of output given the D.E. $\longrightarrow R_y$

② set B.C. $= 0$ outside of R_y.

Detailed

(1) ROS of input R_x

(2) Given Comp. procedure for ΔE
→ Know impulse response $h(n)$
→ Compute R_h = Region of support for impulse response.

(3) Use R_x and R_h To compute
R_y = Region of support of output

(4) B.C = 0 $\forall n \notin R_y$

Ex $y(n) \leftarrow y(n-1) + x(n)$ causal.

$x(n) = \delta(n)$

R_y = what is ROS for output

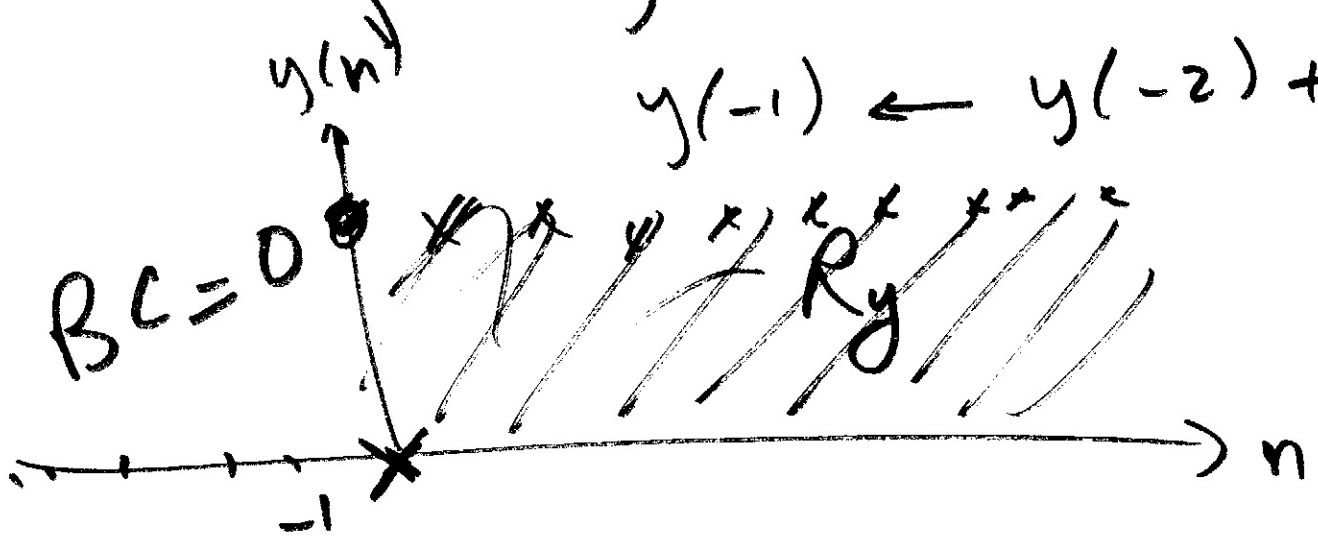
$y(0) \leftarrow y(-1) + \delta(0)$

$y(1) \leftarrow y(0) + 0$

$y(2) \leftarrow y(1)$

$y(3) \leftarrow y(2)$

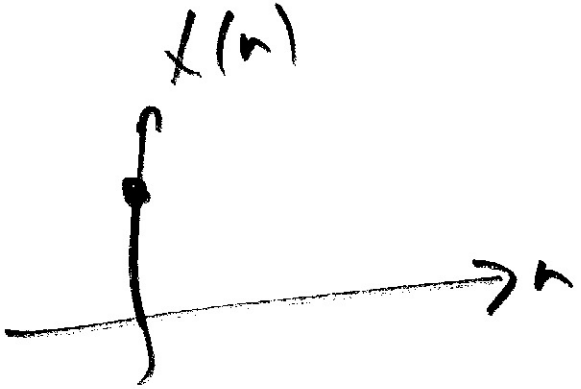
$y(-1) \leftarrow y(-2) + 0$



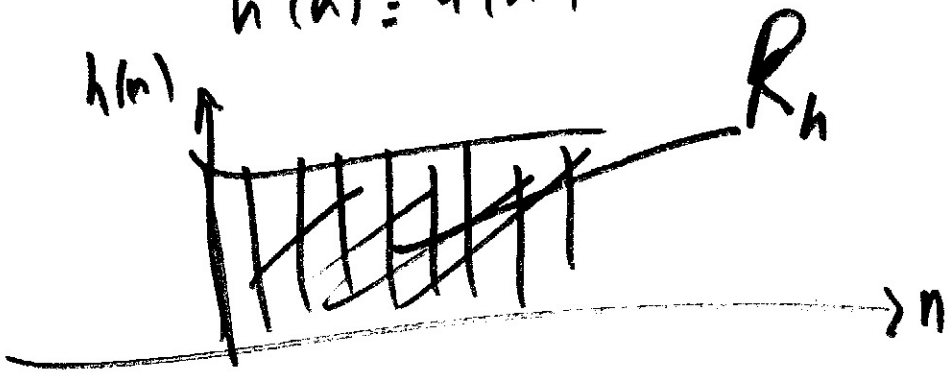
Alitan $h(n)$?

$$Y(z) = z^{-1} Y(z) + X(z)$$

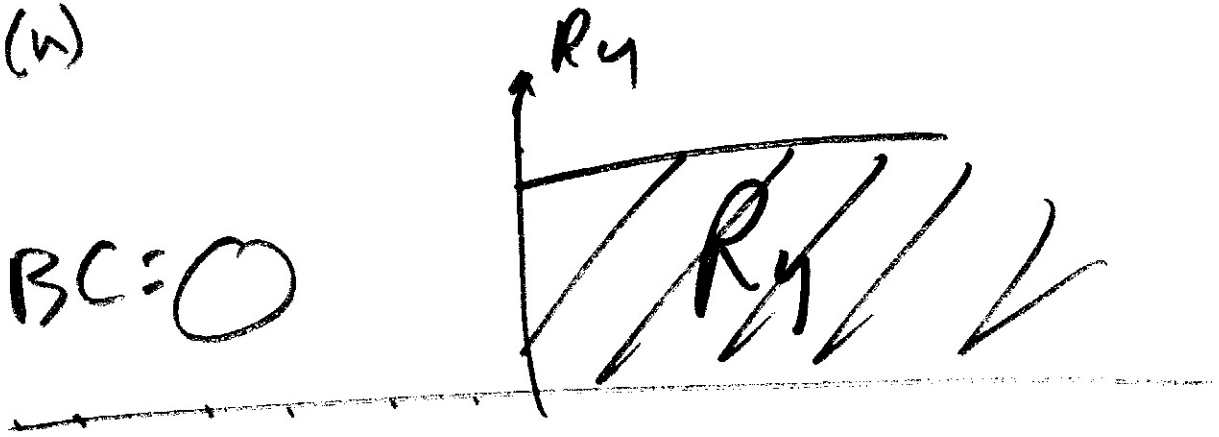
$$H(z) = \frac{Y(z)}{X(z)} = \frac{1}{1 - z^{-1}}$$



$h(n) = u(n)$



$x(n) = \delta(n)$



BC: 0

What happens in ≥ 1?

Exact same steps

(1) Given Comp procedure $\longrightarrow R_h$.

(2) Given input $\longrightarrow R_x$

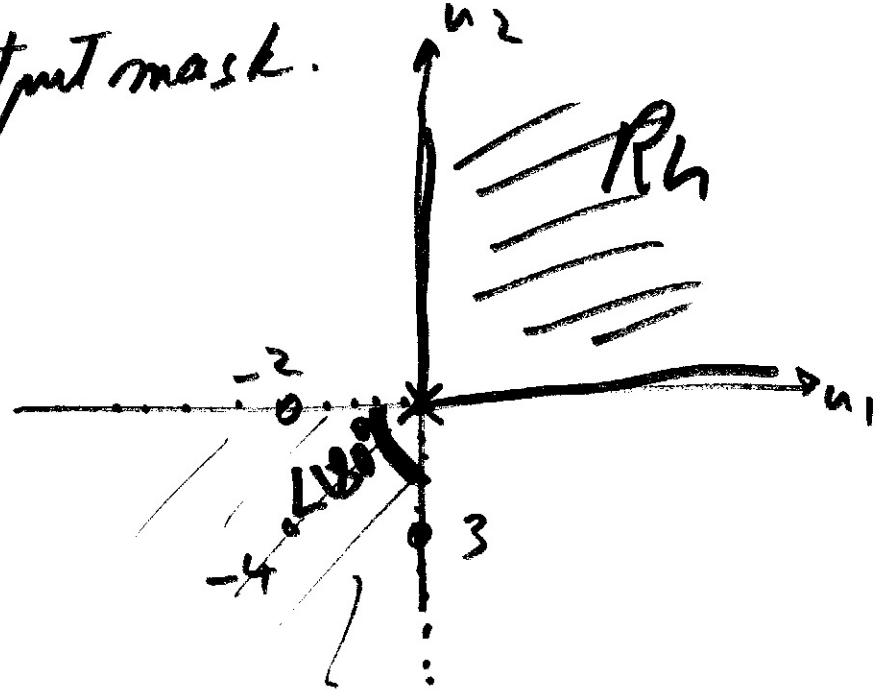
(3) $R_x + R_h \longrightarrow R_y$

(4) $B.C = 0$ $\forall n_1, n_2 \notin R_y$.

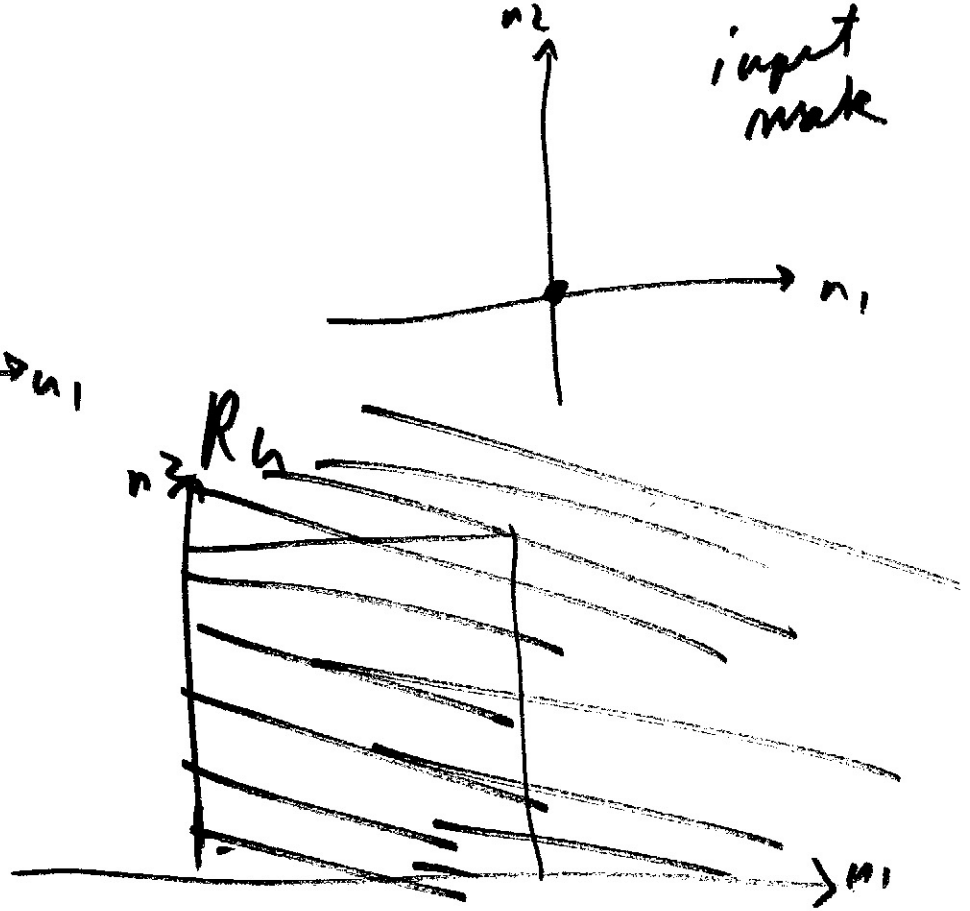
Ex Comp. Proc:

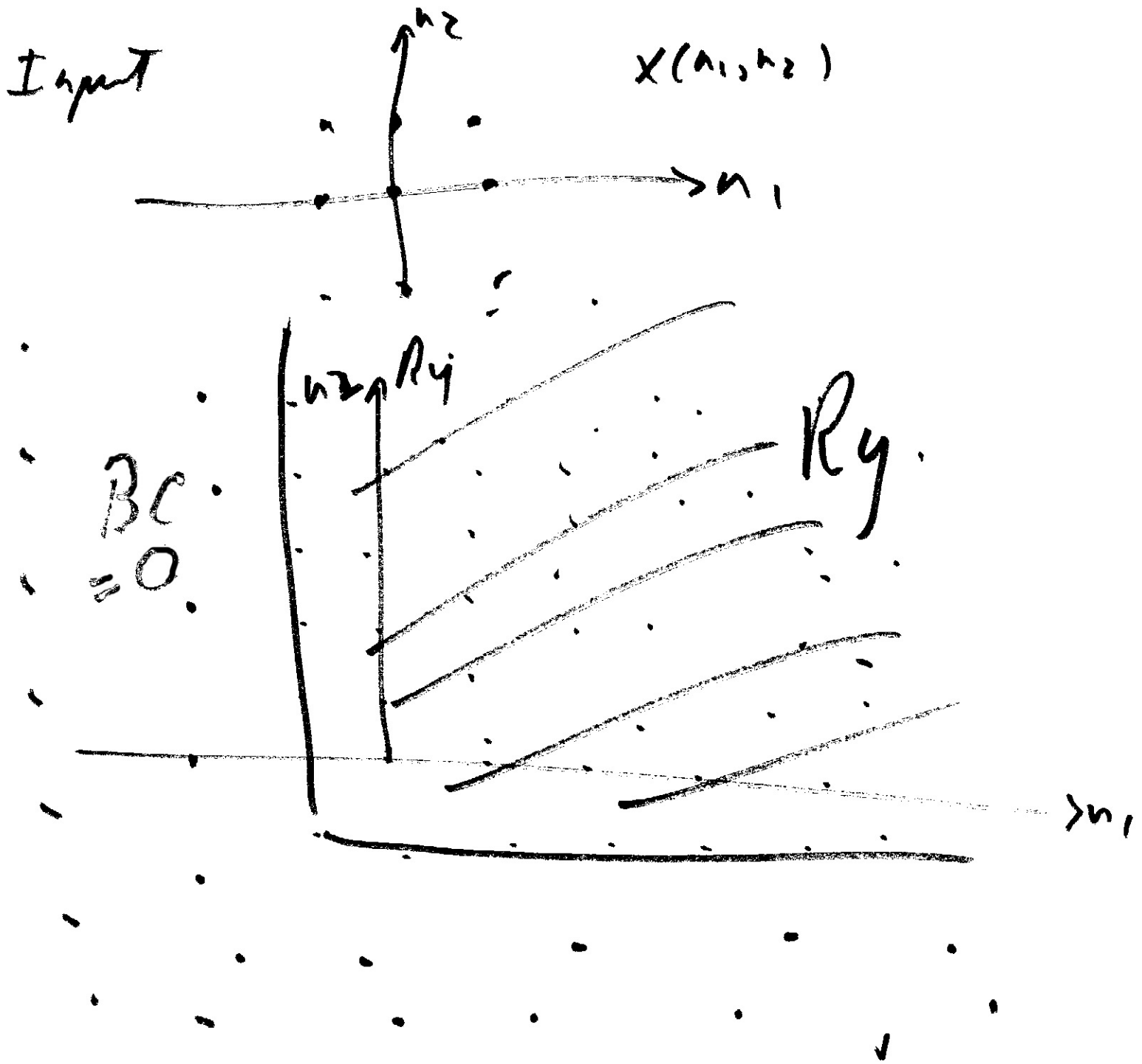
$$y(n_1,n_2) \leftarrow -2\,y(n_1-1,n_2) - 3\,y(n_1,n_2-1) - 4\,y(n_1-1,n_2-1) + x(n_1,n_2)$$

output mask.

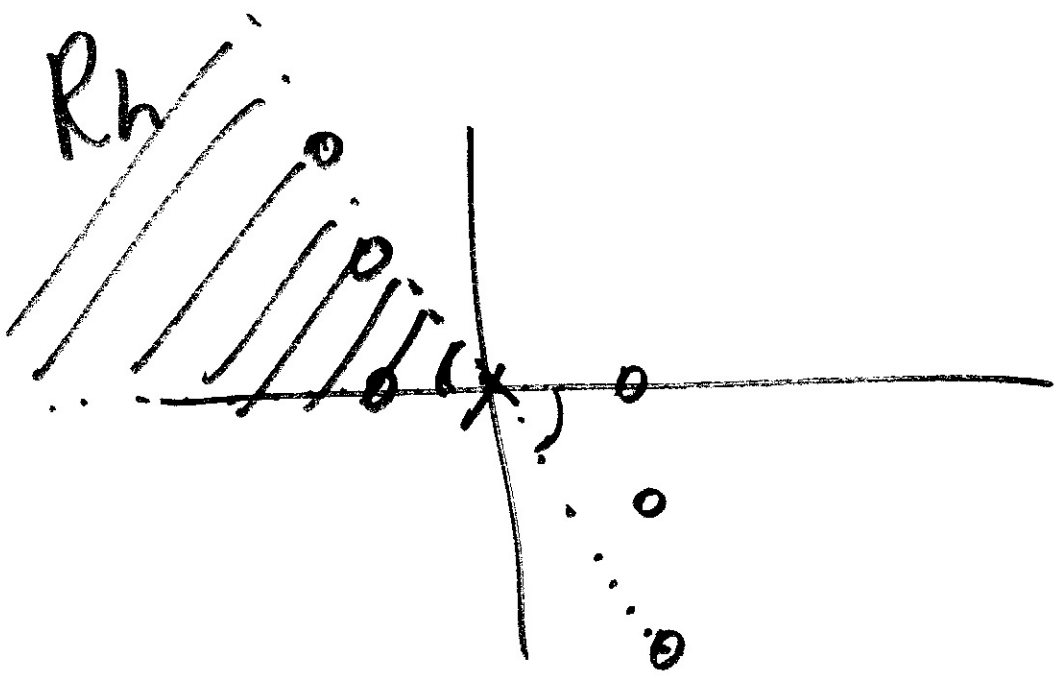


input mask




Input
$x(n_1, n_2)$
n_2
n_1
n_2
R_y
R_y
BC
$= 0$
n_1

p. 98 – 100 J.S. Lin's book



Ex $y(n_1, n_2) = y(n_1 - 1, n_2 - 1) + x(n_1, n_2)$

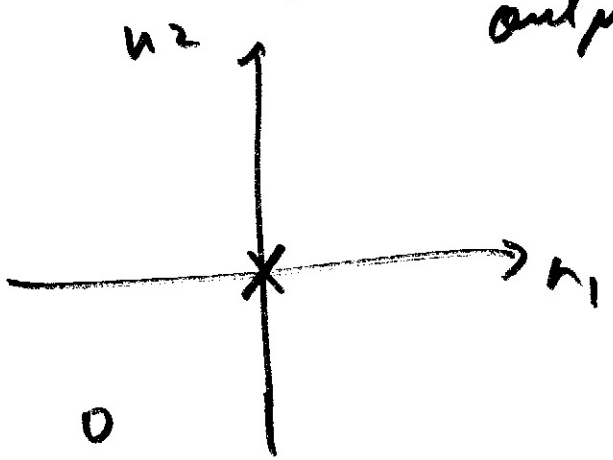
Find Comp. Proc $\longrightarrow$ $h(n_1, n_2)$ is 3rd Quadrant.

---

only 2 comp. proc.

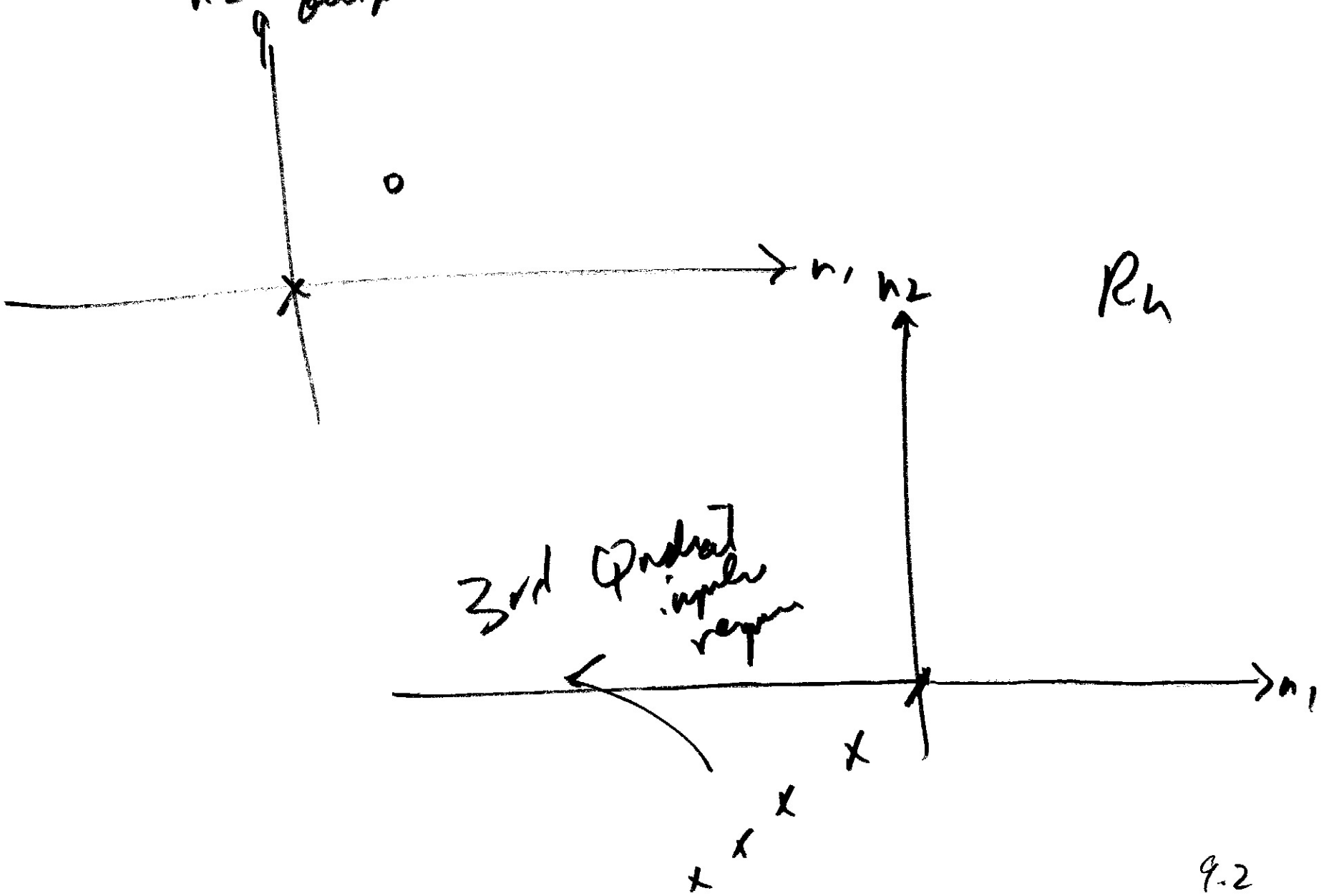
(1) $y(n_1, n_2) \leftarrow y(n_1 - 1, n_2 - 1) + x(n_1, n_2)$

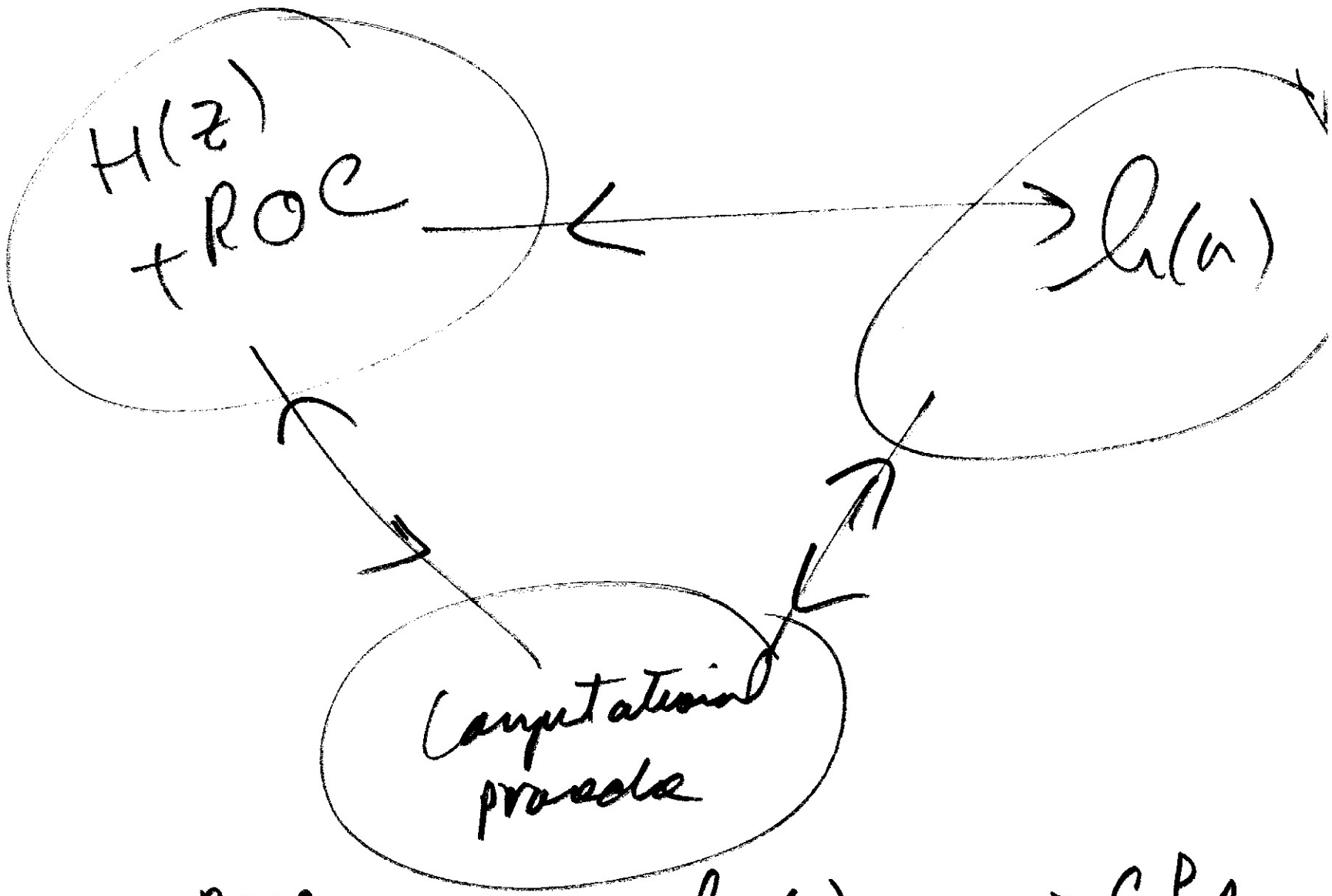
output mark



first Quadrant support.

$y(n_1, n_2) \longleftarrow y(n_1+1, n_2+1) - x(n_1+1, n_2+1)$

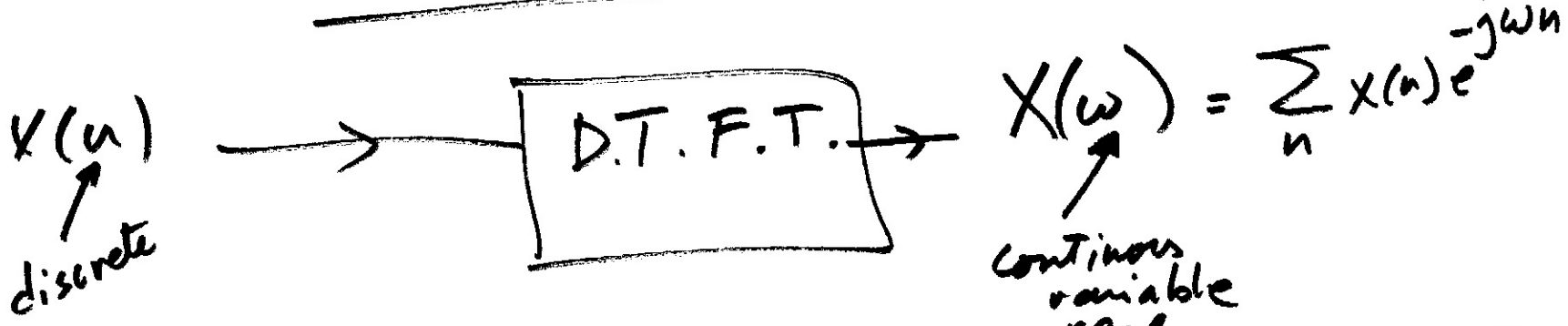


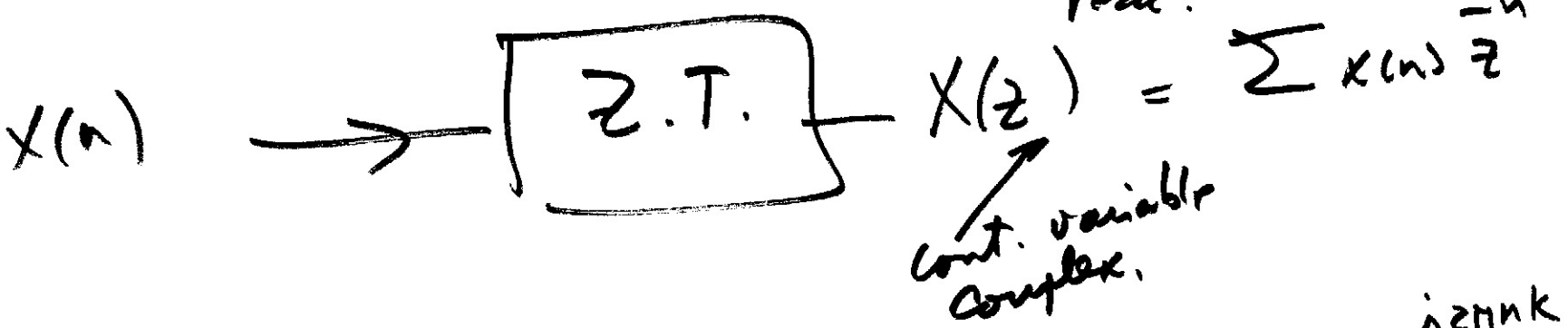


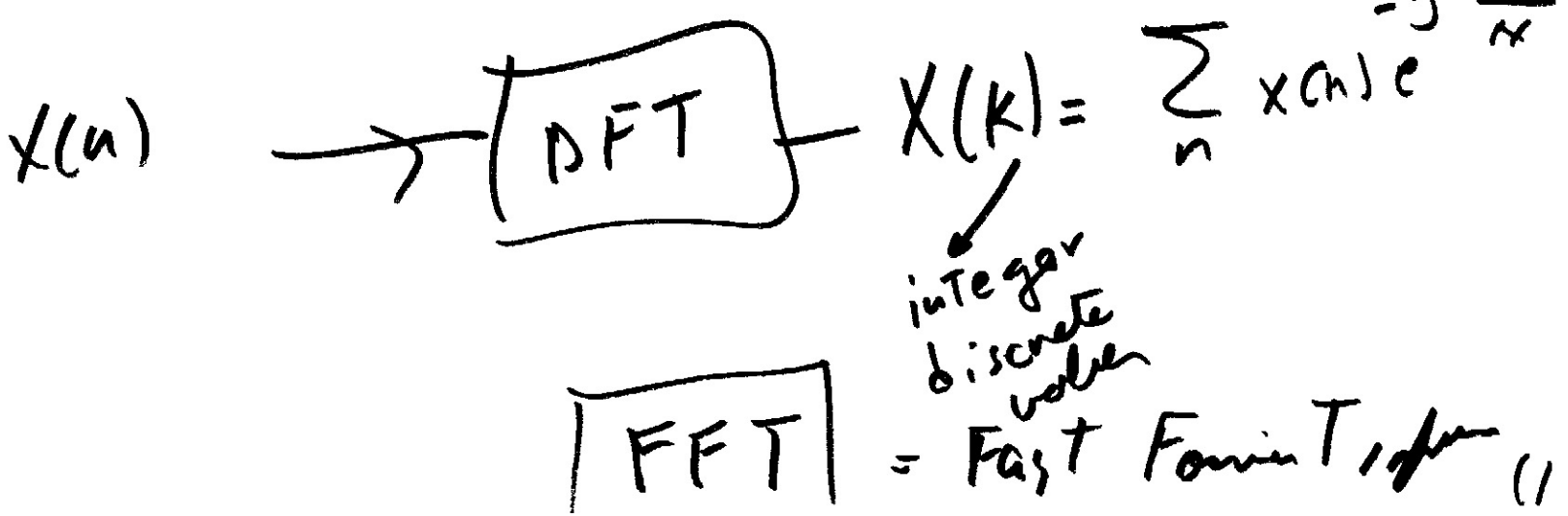
$H(z) + ROC_1 \longrightarrow h_1(n) \longrightarrow C.P.1$

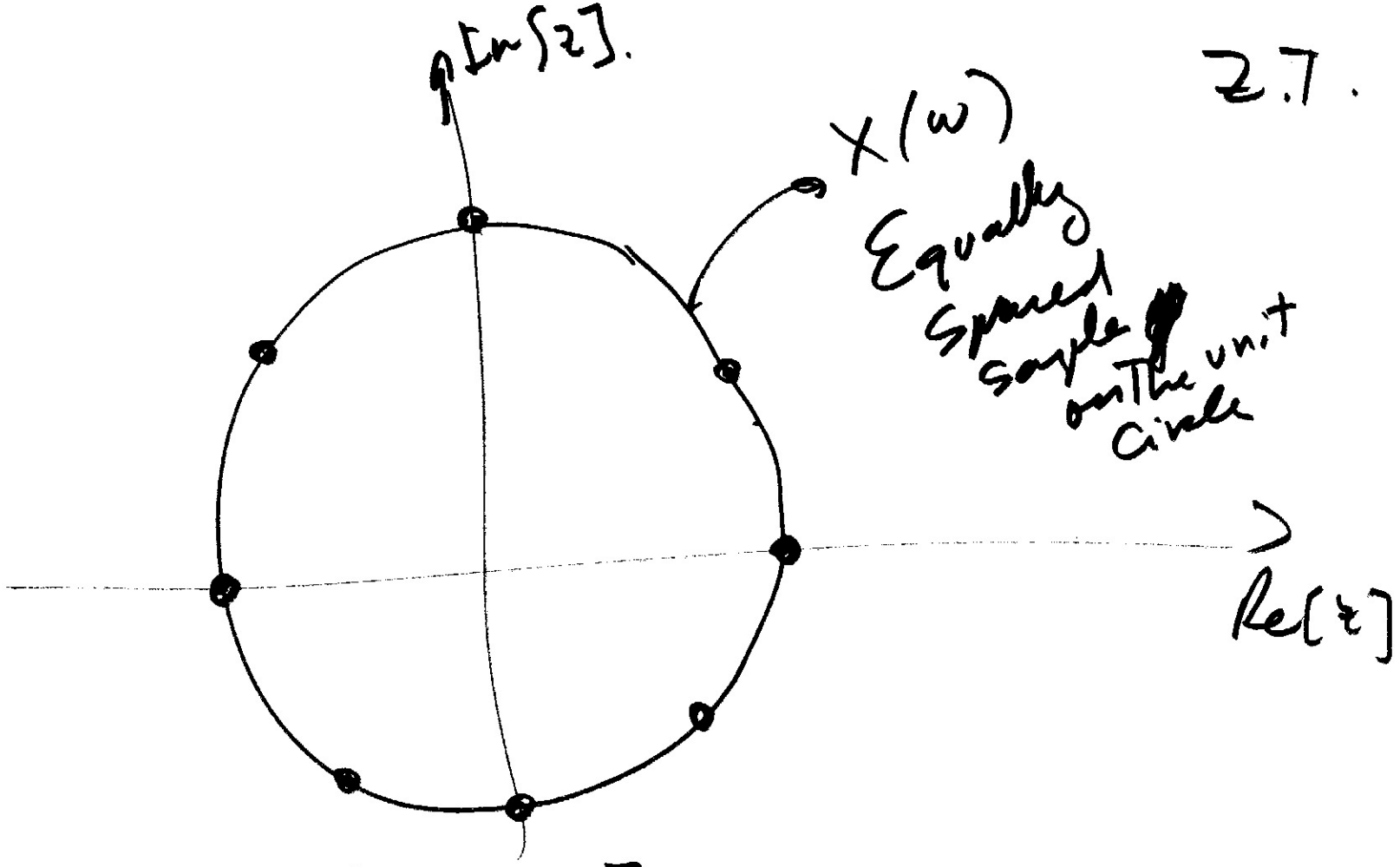
$H(z) + ROC_2 \longrightarrow h_2(n) \longrightarrow CP.2.$

2.D. D.F.T


x(n)
discrete
D.T.F.T.
X(ω) = Σ x(n) e^{-jωn}
continous variable real.


x(n)
Z.T.
X(z) = Σ x(n) z^{-n}
Cont. variable complex.


x(n)
DFT
X(k) = Σ x(n) e^{-j2πnk/N}
integer discrete value
FFT = Fast Fourier T



$$X(k) = \left[X(\omega) \right]_{\omega = \frac{2\pi k}{N}}$$

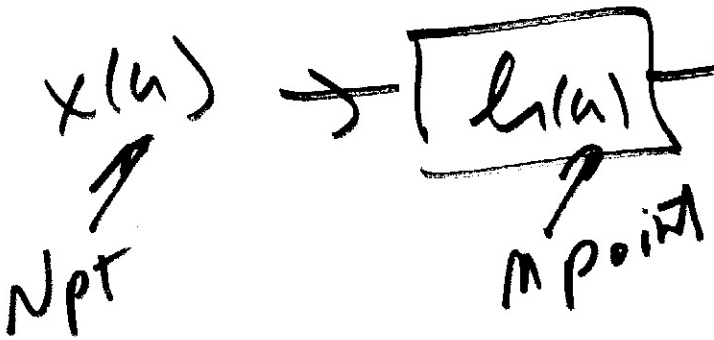
2.D. DFT

$N_1 \times N_2$

$$\mathrm{DFT}\{x(n_1,n_2)\} = X(k_1,k_2) = \begin{cases} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x(n_1,n_2)\, e^{-j2\pi \frac{n_1 k_1}{N_1}} e^{-j2\pi \frac{n_2 k_2}{N_2}} & 0 \le k_1 < N_1 \\ & 0 \le k_2 < N_2 \\ 0 & \text{otherwise} \end{cases}$$

$$x(n_1,n_2) = \mathrm{IDFT}\{X(k_1,k_2)\} = \begin{cases} \frac{1}{N_1 N_2} \sum\sum X(k_1,k_2)\, e^{j2\pi \frac{k_1 n_1}{N_1}} e^{j2\pi \frac{k_2 n_2}{N_2}} & 0 \le n_1 < N_1 \\ & 0 \le n_2 < N_2 \\ 0 & \text{otherwise} \end{cases}$$

Use of DFT in Convolution



Choose $P \geq N+M-1$

(1) Compute P point DFT of input $\longrightarrow X(K)$

(2) " " " " h $\longrightarrow H(K)$

(3) $X(K)\,H(K) = Y(K) = DFT\{y(n)\}$

(4) $IDFT\{Y(K)\} = y(n)$

All of this applies in 2D

How To Compute 2D DFT

3 ways

1. Direct method
2. Row/Col. decomposition
3. Vector-Radix ← True FFT

Factors affect speed

- # of arithmetic ops
- Complexity of # program
  - memory for executible → footprint
  - memory usage at program run.
- language / platform.

Direct Method

$N_1 = N_2 = N$

① For each (k_1, k_2) $\longrightarrow$ N^2 mult, N^2 adds. $\longrightarrow$

② how many (k_1, k_2) ? N^2.

N^4 mult

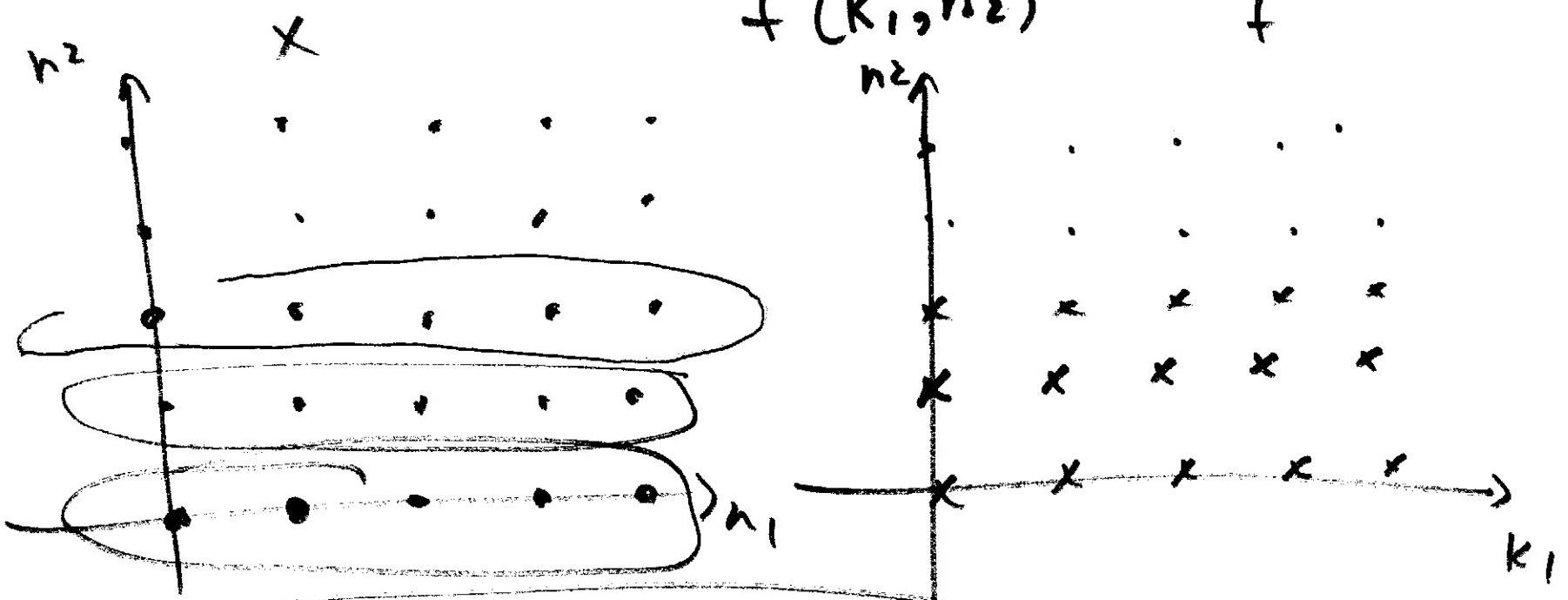
N^4 adds

Row 1.

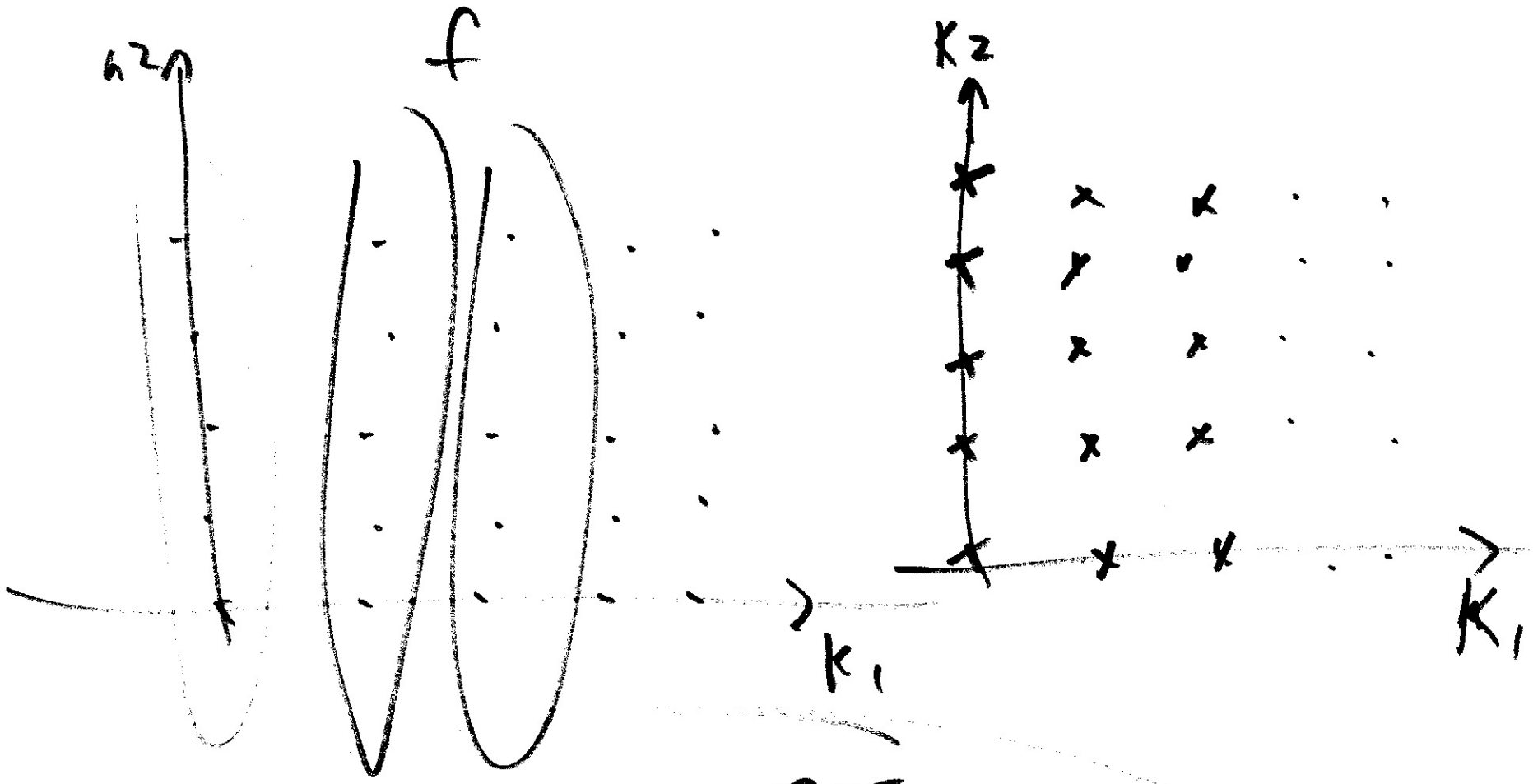
Row Column Decomposition

$$X(k_1,k_2) = \sum_{n_2=0}^{N-1} \left(\sum_{n_1=0}^{N-1} x(n_1,n_2)\, e^{-j\frac{2\pi n_1 k_1}{N}} \right) e^{-j\frac{2\pi n_2 k_2}{N}}$$

$f(k_1,n_2)$ $\quad$ f

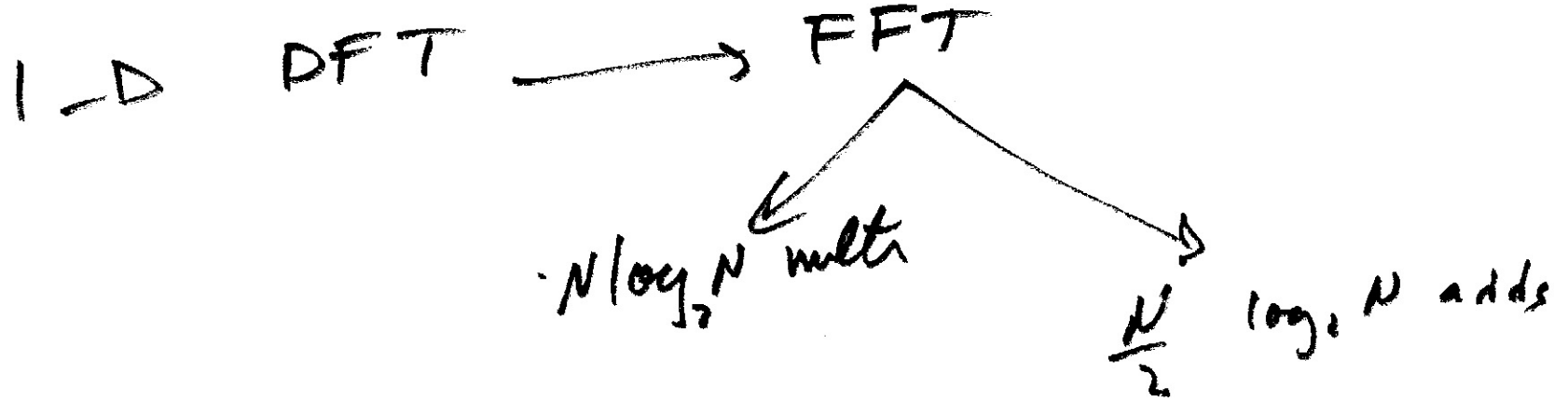


N - 1-D DFT of rows
each 1-D DFT is N point



N — 1D DFT.

each 1D DFT was N pt



R/C decomp $N^2 \log_2 N$ operations

LOT lower direct method