

# SME206, Signals and Systems

## Extension – 2D FFT

---

黄嘉炜

[huangjw3@mail.sustech.edu.cn](mailto:huangjw3@mail.sustech.edu.cn)

李嘉敏

[lijm3@sustech.edu.cn](mailto:lijm3@sustech.edu.cn)

# Recall: DTFT

analysis  
equation

$$X(e^{jw}) = \sum_{n=-\infty}^{+\infty} x[n]e^{-jwn}$$

synthesis  
equation

$$x[n] = \frac{1}{2\pi} \int_{w_0}^{w_0+2\pi} X(e^{jw})e^{jwn} dw$$

For any DT signals

- When  $0 \leq n < M$ ,  $x[n] \geq 0$
- When  $n < 0$  or  $n \geq M$ ,  $x[n] = 0$

when  $N \geq M$ ,

**X = fftshift(fft(x, N));**  
**w = -pi + [0:(N-1)] \* 2\*pi/N;**

$$\triangleq \begin{bmatrix} e^{-jw_0 0} & \dots & e^{-jw_0(N-1)} \\ \vdots & \ddots & \vdots \\ e^{-jw_{N-1} 0} & \dots & e^{-jw_{N-1}(N-1)} \end{bmatrix} \begin{pmatrix} x[0] \\ x[1] \\ \vdots \\ x[M-1] \\ 0 \\ \vdots \\ 0 \end{pmatrix} = fft(\begin{pmatrix} x[0] \\ x[1] \\ \vdots \\ x[M-1] \\ 0 \\ \vdots \\ 0 \end{pmatrix})$$

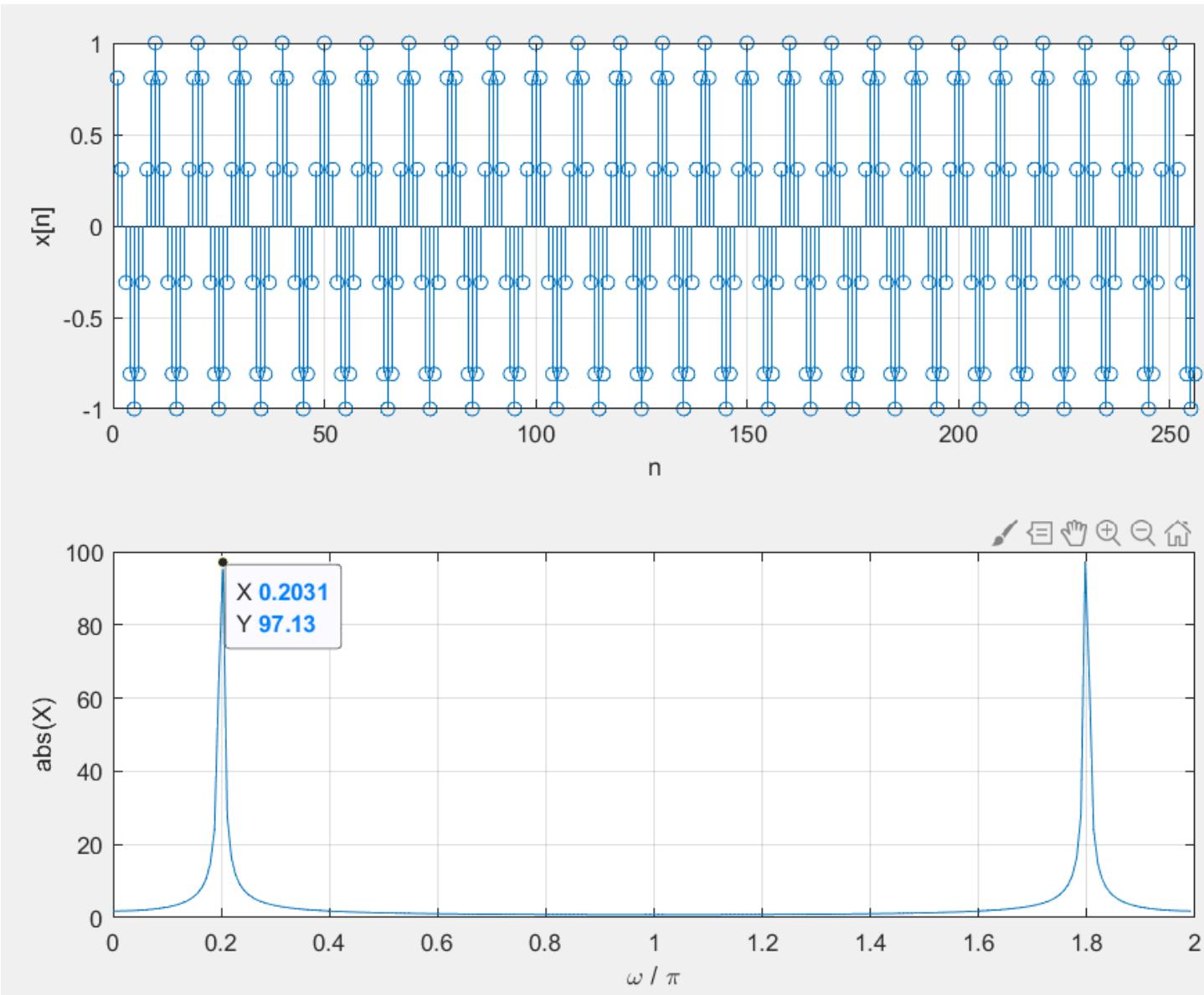
# Example: DTFT

Given a signal  $x[n]$ , as

$$x[n] = \cos(w_h n)$$

```
N = 256;  
wh = 2*pi / 10;
```

```
nx = 1:N;  
x = cos(wh * nx);  
w = [0:N-1] * 2*pi / N;  
X = fft(x);
```



Frequency analysis by 'fft', with sampling window of N

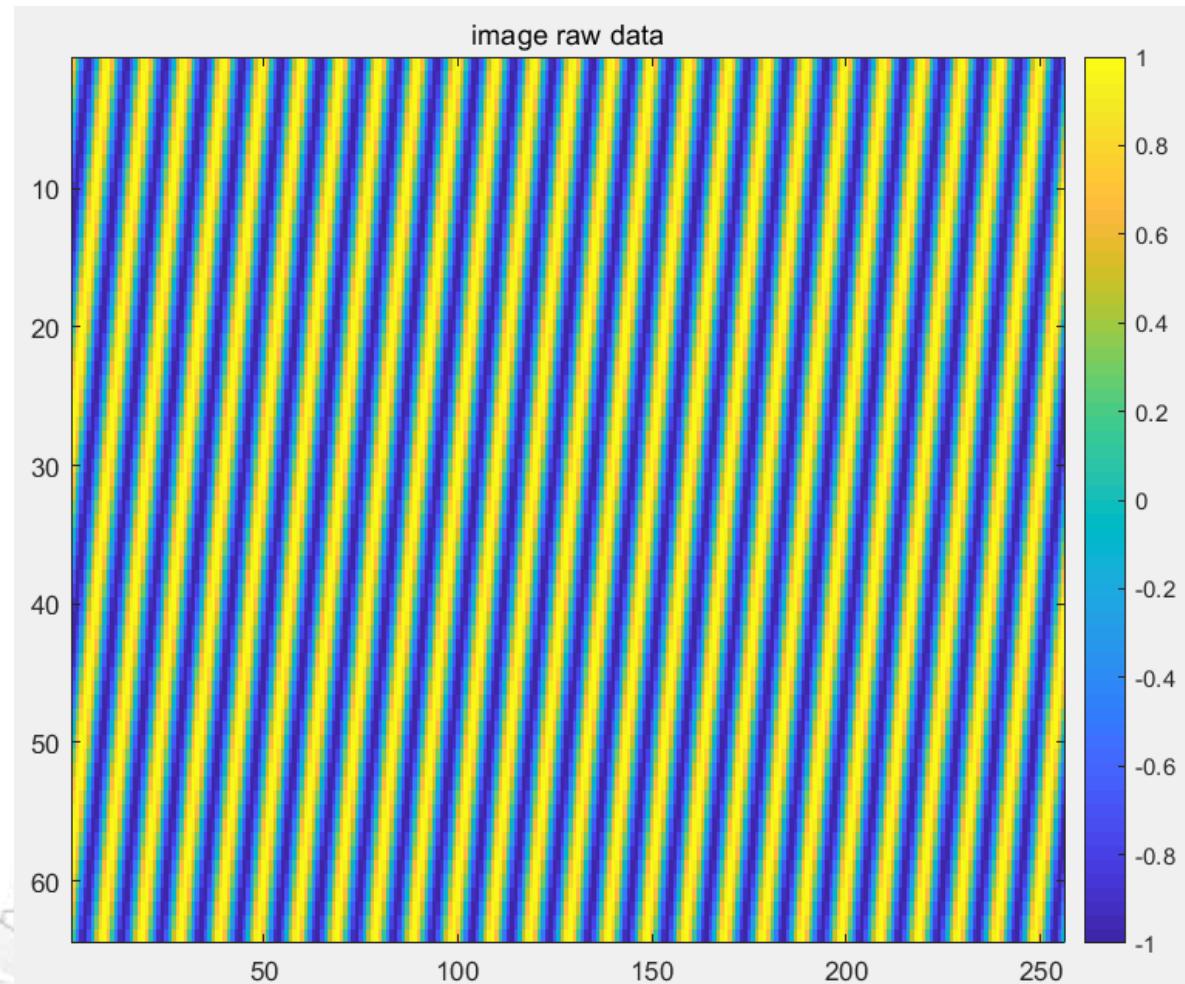
# Extend to 2D signal

For m-th sampling window, given  
a signal  $x[m, n]$ , as

$$x[m, n] = \cos(w_h n + w_v m)$$

```
N = 256;
M = 64;
wh = 2*pi / 10;
wv = 2*pi / 30;

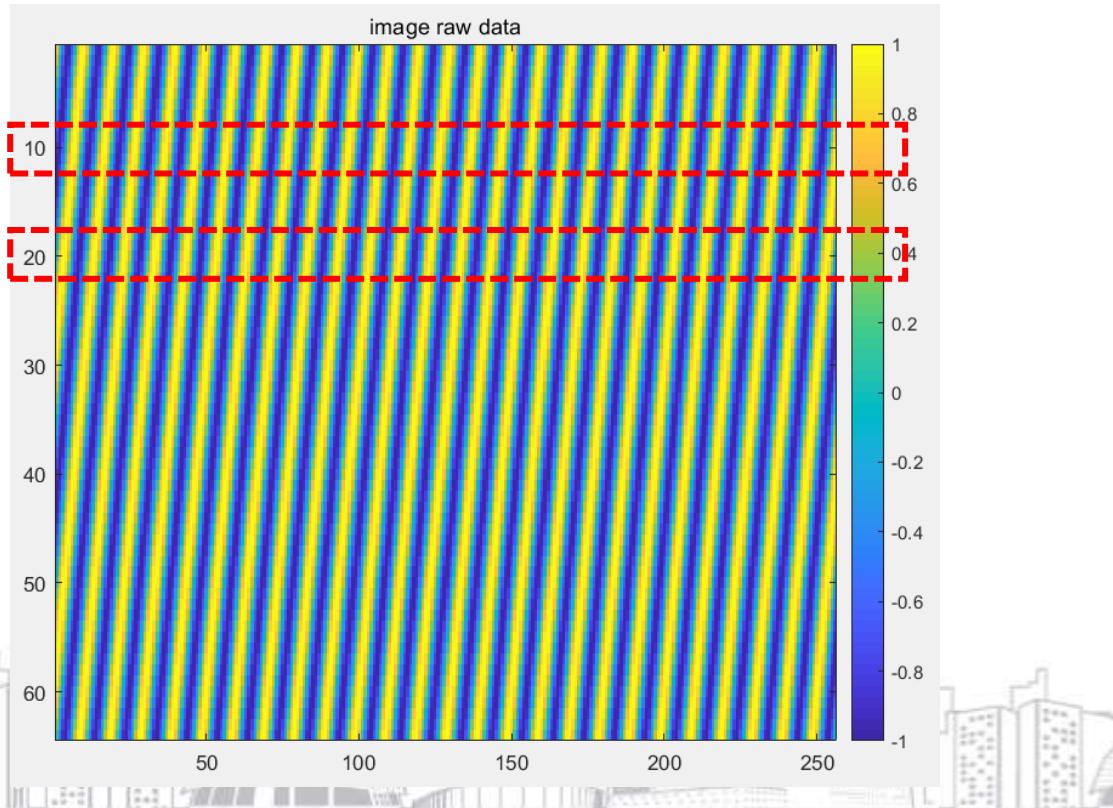
img = zeros(M, N);
for row = 1:M
    row_offset = wv * row;
    ncol = 1:N
    img(row, :) = cos(wh * ncol + row_offset);
end
```



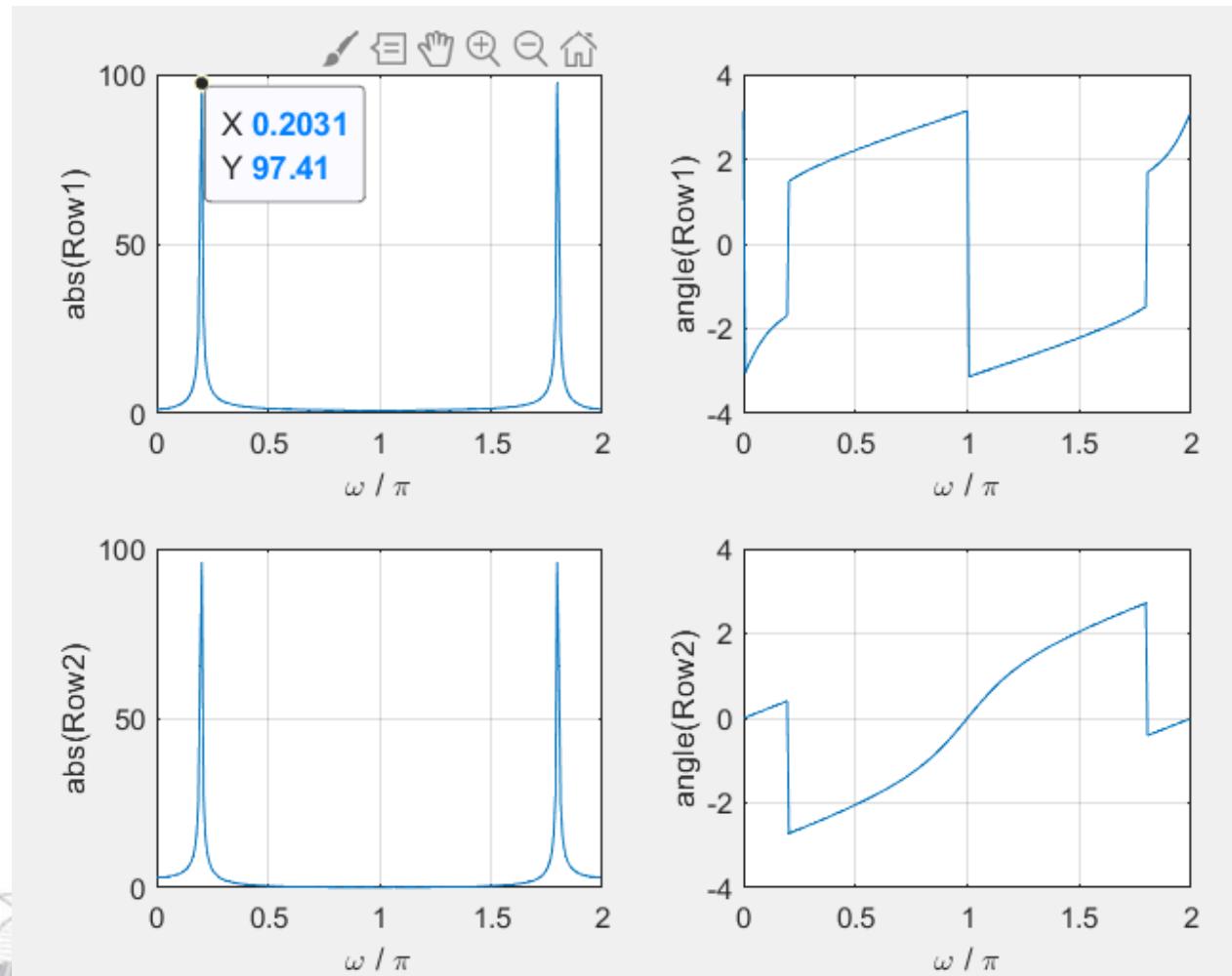
# Extend to 2D signal

For m-th sampling window, given a signal  $x[m, n]$ , as

$$x[m, n] = \cos(w_h n + w_v m)$$



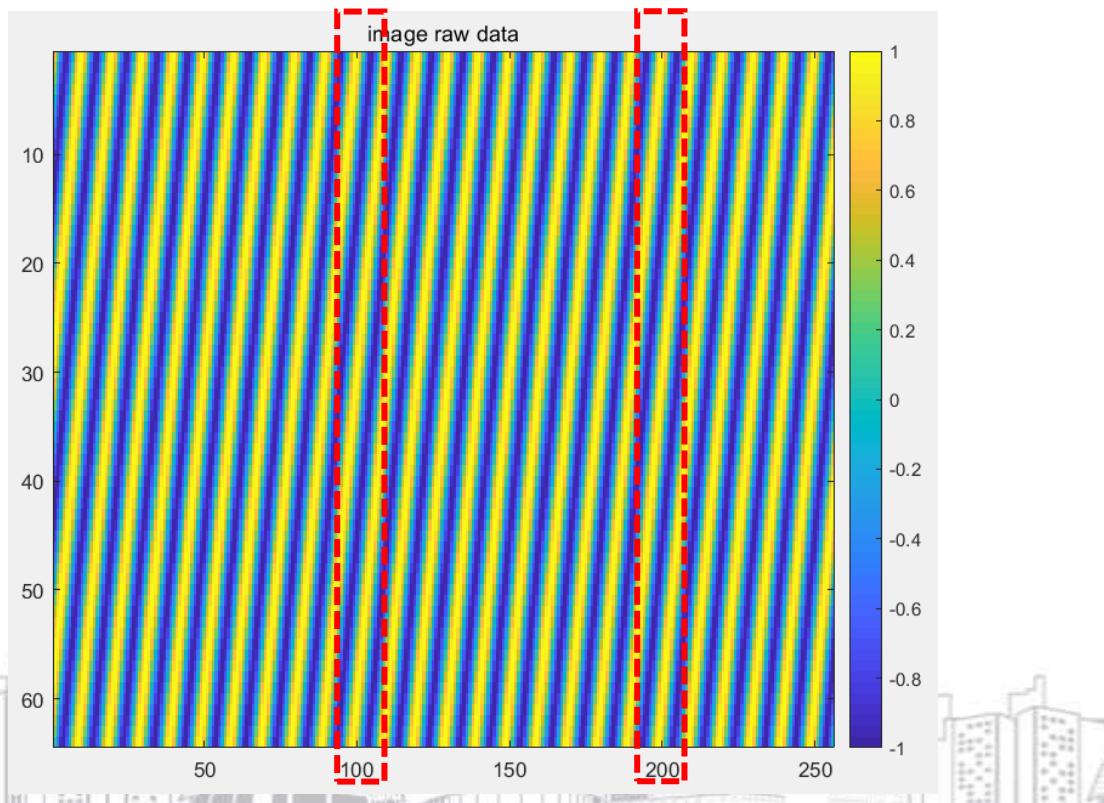
**'fft' analysis: With fixed m, take samples  $x[m, 1:N]$**



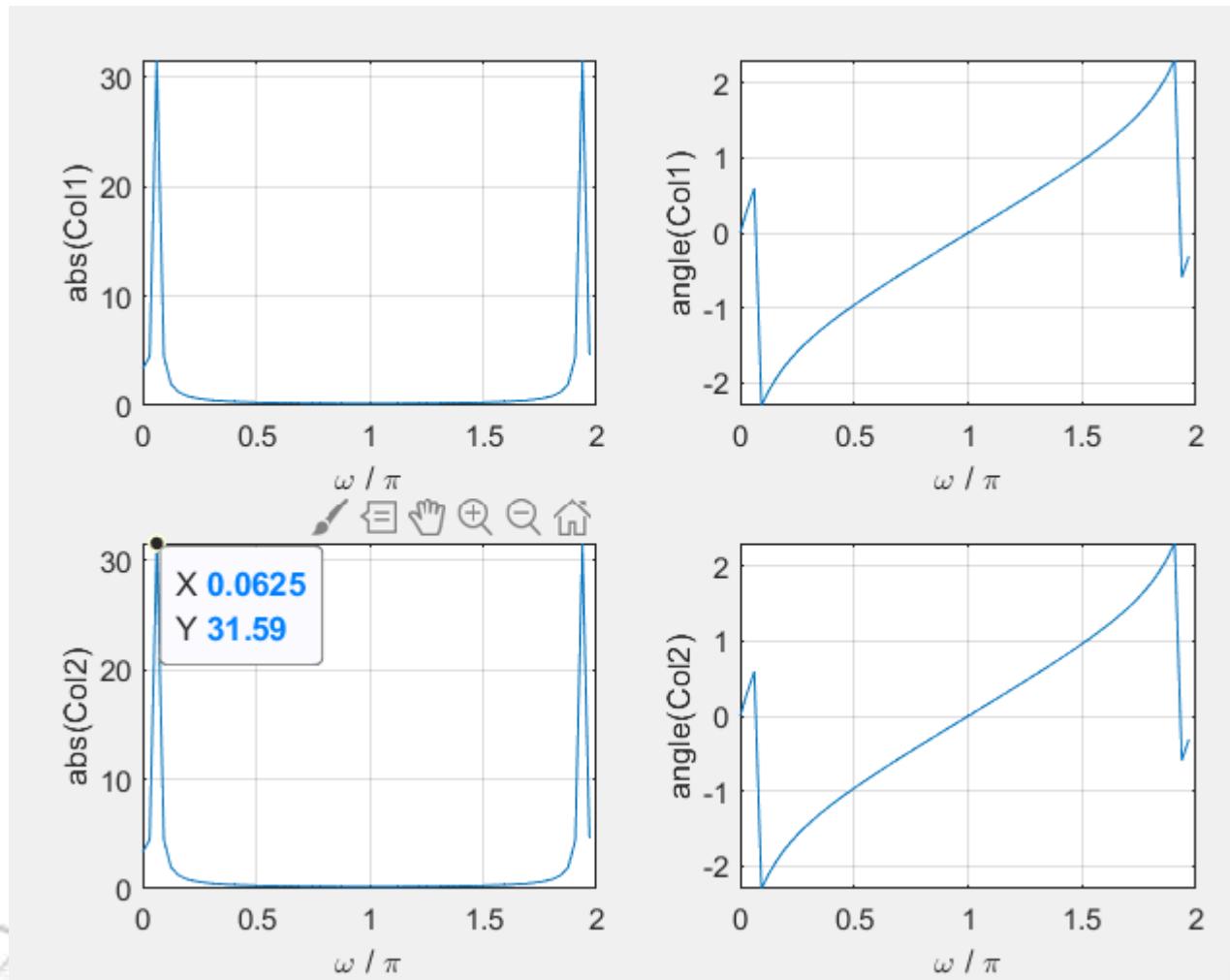
# Extend to 2D signal

For m-th sampling window, given a signal  $x[m, n]$ , as

$$x[m, n] = \cos(w_h n + w_v m)$$



**'fft' analysis: With fixed n, take samples  $x[1:M, n]$**



# 2D DTFT / Background

$$x[m, n] = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} X[k, l] e^{j(w_l n + w_k m)}$$

synthesis  
equation

$$X[k, l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m, n] e^{-j(w_l n + w_k m)}$$

analysis  
equation

$$w_l = \frac{2\pi}{N} l, \text{ where } l = 0, 1, \dots, N - 1$$

$$w_k = \frac{2\pi}{M} k, \text{ where } k = 0, 1, \dots, M - 1$$

# 2D DTFT / Matlab fft2

$$X[k, l] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x[m, n] e^{-j(w_l n + w_k m)}$$
$$= \left( \sum_{m=0}^{M-1} e^{-jw_k m} \right) \left( \sum_{n=0}^{N-1} x[m, n] e^{-jw_l n} \right)$$

Step2, fft on each column

Step1, fft on each row

## fft2

2-D fast Fourier transform

### Syntax

```
Y = fft2(X)  
Y = fft2(X,m,n)
```

### Description

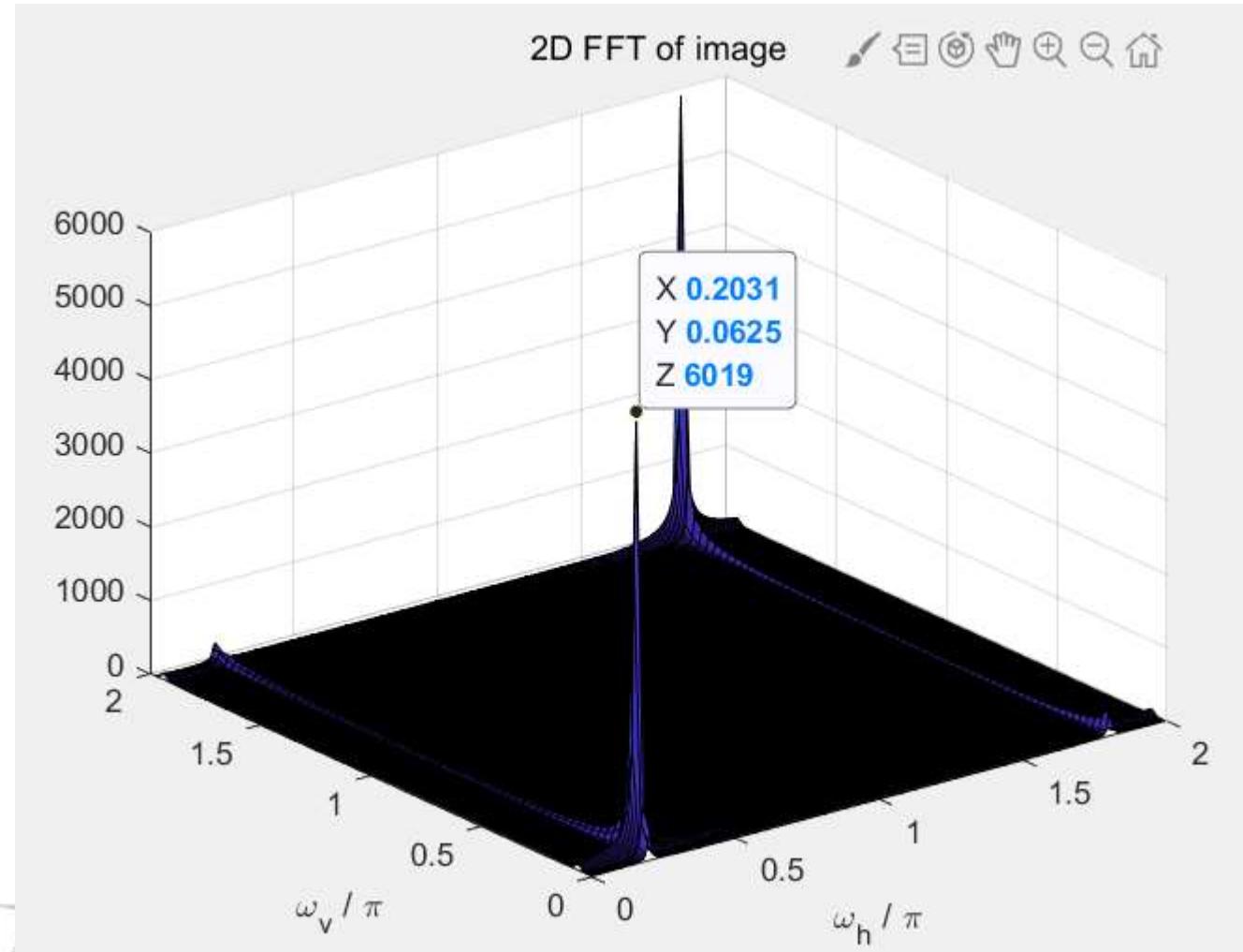
`Y = fft2(X)` returns the two-dimensional Fourier transform of a matrix `X` using a fast Fourier transform algorithm, which is equivalent to computing `fft(fft(X).').'`.

When `X` is a multidimensional array, `fft2` computes the 2-D Fourier transform on the first two dimensions of each subarray of `X` that can be treated as a 2-D matrix for dimensions higher than 2. For example, if `X` is an  $m$ -by- $n$ -by-1-by-2 array, then  $Y(:,:,1,1) = \text{fft2}(X(:,:,1,1))$  and  $Y(:,:,1,2) = \text{fft2}(X(:,:,1,2))$ . The output `Y` is the same size as `X`.

`Y = fft2(X,m,n)` truncates `X` or pads `X` with trailing zeros to form an  $m$ -by- $n$  matrix before computing the transform. If `X` is a matrix, then `Y` is an  $m$ -by- $n$  matrix. If `X` is a multidimensional array, then `fft2` shapes the first two dimensions of `X` according to `m` and `n`.

# 2D DTFT / Matlab fft2

```
Z = fft2(img);
x_axis = [0:N-1] * 2*pi / N;
y_axis = [0:M-1] * 2*pi / M;
surf(x_axis / pi, y_axis / pi, abs(Z));
title("2D FFT of image");
xlabel("\omega_h / \pi");
ylabel("\omega_v / \pi");
```

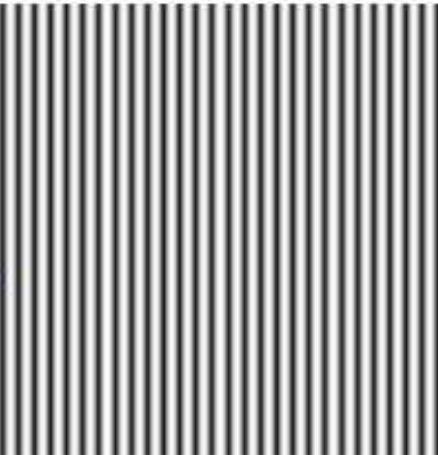


# 2D DTFT / Extend to Image ?

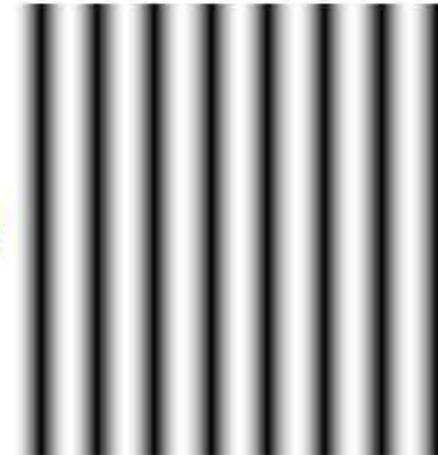
$f(x, y)$



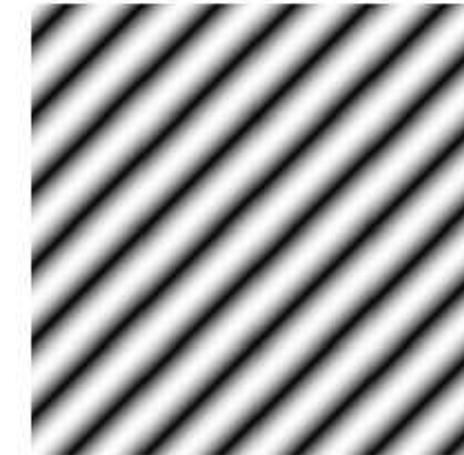
$= \alpha$



$+ \beta$



$+ \gamma$



$+ \dots$

$$x[m, n] = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} X[k, l] e^{j(w_l n + w_k m)}$$

synthesis  
equation

# THANK YOU

---

