

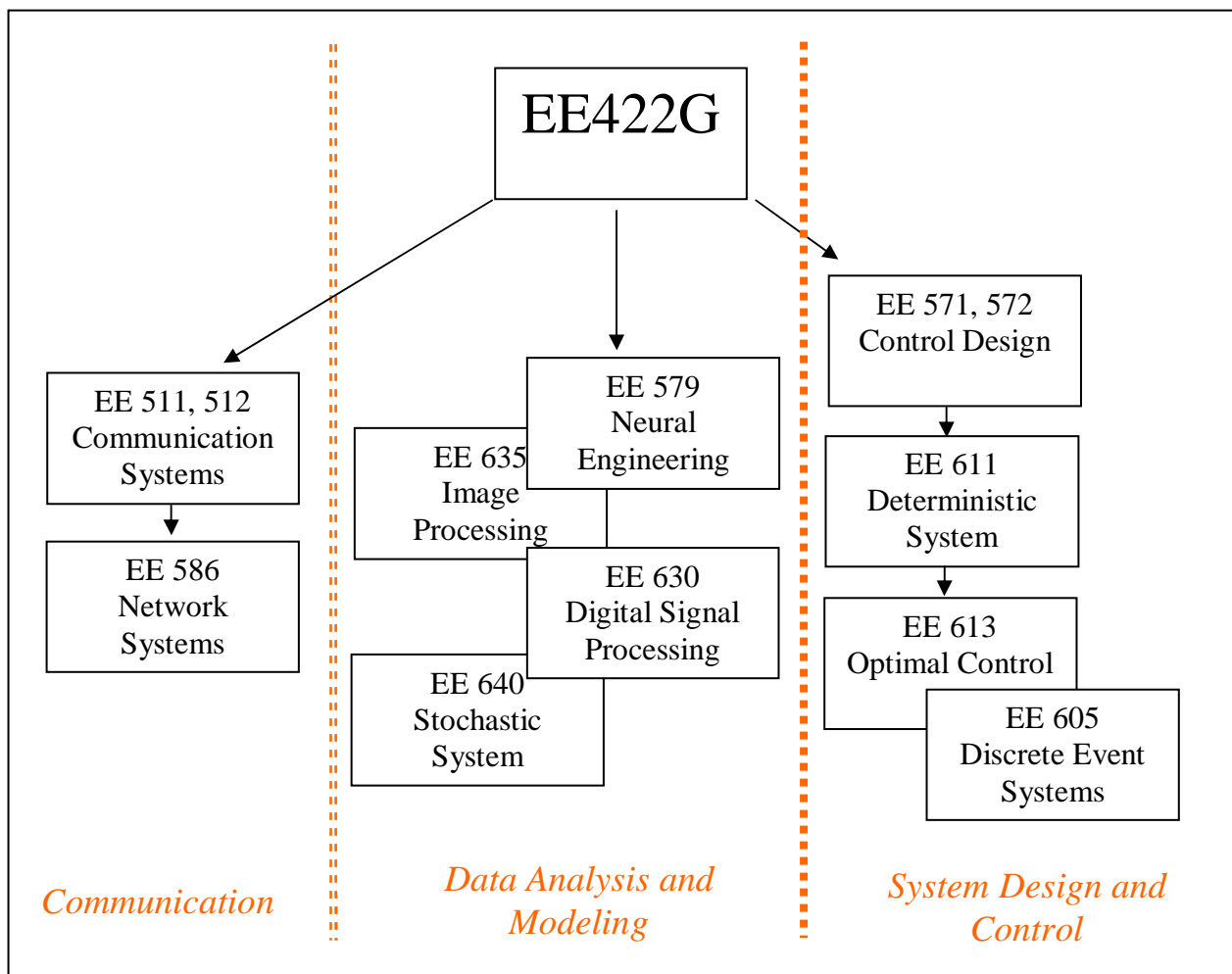
## EE422G: Signal and Systems II

### Overview

#### 1. Objective of the course

Provide you with the necessary mathematical background to design, analyze, and model different kinds of signals and systems used in engineering.

#### 2. How does EE422G fit into your curriculum and professional career?



Some examples of jobs:

**Position** Digital Signal Processing Engineer

Position will serve as a member of a wireless technology team. Teams focus is low powered, low cost, embedded software radios. General responsibilities will take place in the following areas: Wireless Signal Processing, System Engineering, Waveform Development, General Signal Processing. This position involves the design and implementation of wireless modems. This includes the design, documentation, simulation, analysis and implementation of algorithms, communications systems and protocols.

**Position** Image Processing Engineer

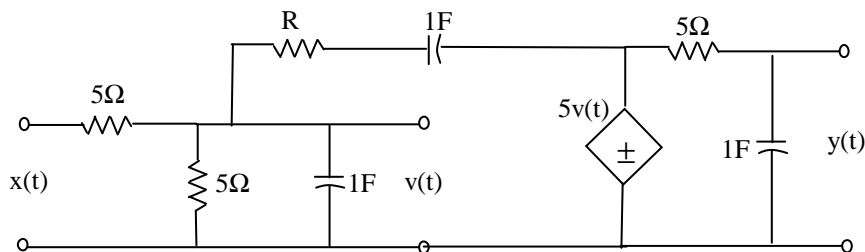
Develop, implement, test, debug, and document automated machine vision inspection algorithms in C/C++. Integrate motion control, frame grabbers, light control, and camera control for specific applications. Generate specifications for image acquisition systems. Experience with C/C++, computer vision, image & signal processing, digital signal processing, and system integration

**Position** Video Processing Engineer

Apple Computer is looking for a senior engineer / architect to focus on the next generation video architectures with emphasis on digital video camera signal processing algorithms and software development. He/she will be responsible for design of most efficient software architecture for video processing, will also ensure optimal and consistently improving video quality through the entire procession pipeline, and help track and help select generations of video camera. This individual should also have demonstrated expertise in the following areas of video and image processing: fix pattern and temporal noise reduction, bad pixel correction, color correction, Bayer demosaicing, auto-white balancing, edge enhancement, auto exposure, auto-focus.

2. What will we cover in this course? Four main topics.

Analysis of Continuous-Time System using Laplace Transform and State-Variable Techniques



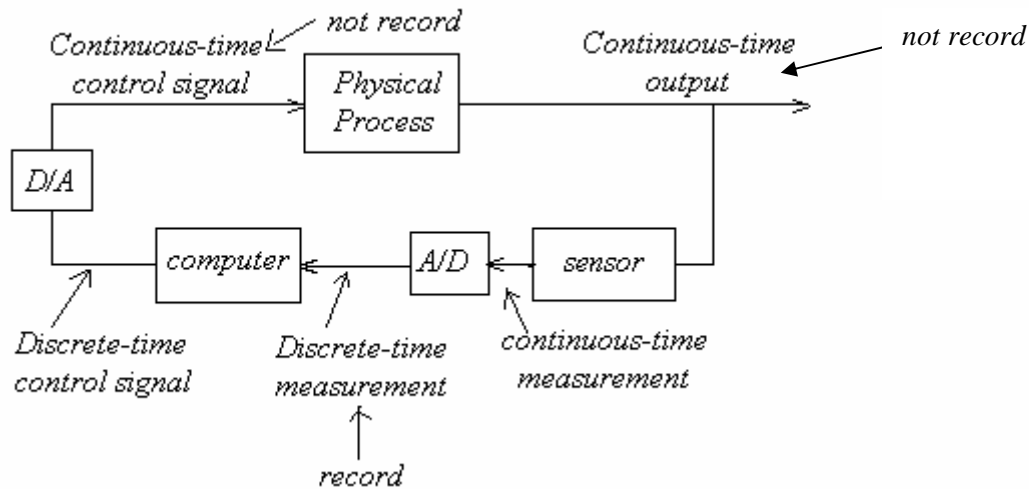
Based on the techniques that you learnt from EE421G, you should be able to compute the steady output of  $y(t)$  when  $x(t)$  is a sinusoid. But can you answer the following questions?

- What is the output  $y(t)$  if  $x(t) = 10te^{0.5t}\cos(100\pi t)u(t)$  (*Not sinusoidal input*) and  $v(t) = 5$  for  $t=0$  (*Initial Condition*)?
- What are the possible values of  $R$  such that the system is stable, i.e. a bounded input signal will deliver a bounded output signal?
- Can you write a program to simulate this circuit for any input?

The first two questions can be easily addressed by a new transform method called the Laplace Transform, which is the first topic of this course.

The last question is tougher to answer. Towards the end of this course, we will introduce an advanced technique called State Variable Technique that is instrumental for simulating the time-domain behavior of dynamic systems.

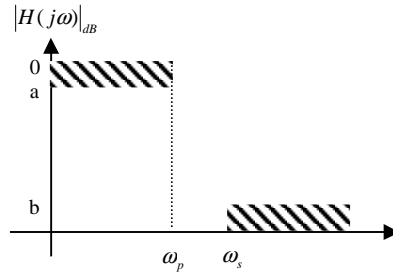
### Discrete-Time Signals and Systems



Computer offers great flexibility in design and analysis. Many of the modern design tools are developed for discrete-time system. The major design tool introduced is the Z-transform (analogous to the Laplace transform in continuous-time). We will also discuss different issues such as sampling rate, quantization, interpolation for interfacing discrete and continuous-time systems.

Filter Design

Low Pass Specification:  
 Pass Band:  $0 \leq \omega \leq \omega_p$   
 $a \leq |H(j\omega)|_{dB} \leq 0$   
 Stop Band:  $\omega_s \leq \omega$   
 $|H(j\omega)|_{dB} \leq b$



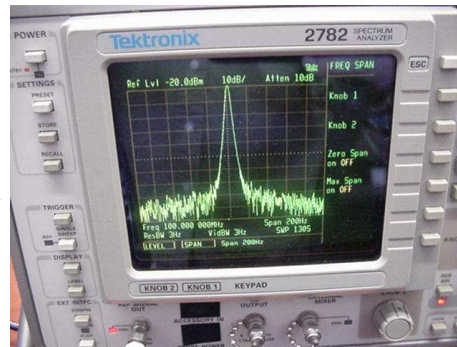
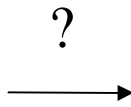
The main design component of this course will focus on various approaches for designing analog (continuous-time) and discrete (discrete-time) filters that satisfy a particular design specification.

Computational Signal Processing

Analysis :  $X(j\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt$

Synthesis :  $x(t) = \int_{-\infty}^{\infty} X(j\omega)e^{j\omega t} d\omega$

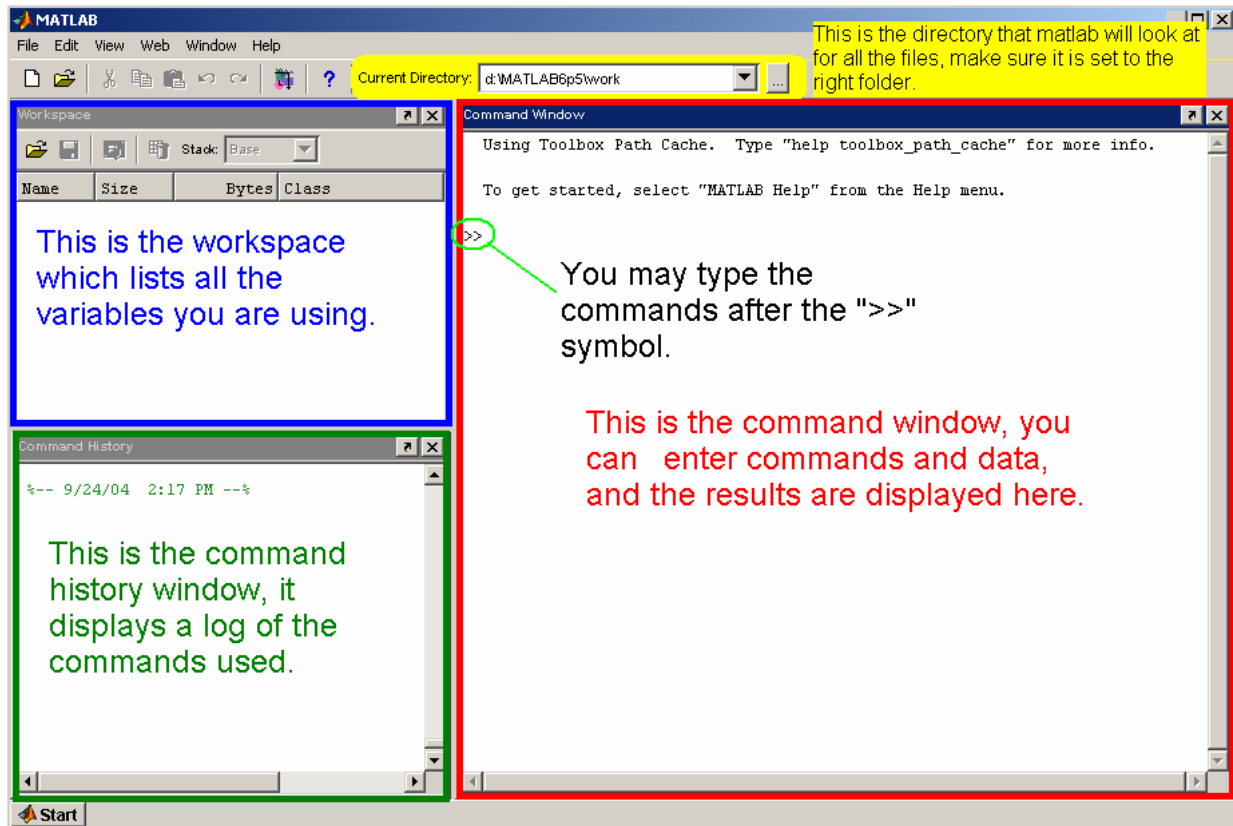
Fourier Transform



How do you build a digital spectrum analyzer? We will learn how to sample Fourier Transform and how to compute it efficiently. The fundamental concept behind the Fast Fourier Transform is the cornerstone of many modern algorithms.

## Introduction to Matlab

### 1. Starting Matlab



- It might be different if you use an older version of Matlab (mine is 7.0.1).
- Make sure you have Signal Processing Toolbox, Control System Toolbox, and Symbolic Math Toolbox. To check, type
 

```
>> help
```

 and you should see these toolboxes listed among all the help pages:
 

signal\signal	-	Signal Processing Toolbox
control\control	-	Control System Toolbox
toolbox\symbolic	-	Symbolic Math Toolbox

### 2. Numbers, constants, variables and data types

- Numbers
 

```
>> 4
      ans =
          4
```
- Constants
 

```
>> pi
```

```
ans =
    3.1416
>> 3+4i
ans =
    3 + 4.0000i
```

- **MATLAB** variable names must begin with a letter, which may be followed by any combination of letters, digits, and underscores. **MATLAB** distinguishes between uppercase and lowercase characters, so **A** and **a** are not the same variable.

- **Vectors**

```
>> v = [2 3 4 5 6 7]
v =
     2     3     4     5     6     7
>> v(4)
ans =
     5
>> w = 2:0.5:7
w =
Columns 1 through 6
    2.0000    2.5000    3.0000    3.5000    4.0000    4.5000
Columns 7 through 11
    5.0000    5.5000    6.0000    6.5000    7.0000
```

- **Matrices** – the formal name of Matlab is **MATrix LABoratory**, so Matlab is very efficient in handling matrices

```
>> A = [1 2 3; 4 5 6; 7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
>> A(1,2)
ans =
     2
```

- **Strings**

```
>> w = 'matlab is simple'
w =
matlab is simple
```

- **Symbolic variables**

```
>> syms x y z
>> x
x =
x
```

### 3. Operations and functions

- Simple operations all work as they should be. Type “help ops” for a list of operators and “help elfun” for a list of elementary mathematical functions. Almost all of them can be applied to scalar, vectors, and matrices.

```
>> ((3+4*5)/(4+5i))^0.44
ans =
    1.6206 - 0.6743i
>> sin([1 2 3 4])
ans =
    0.8415    0.9093    0.1411   -0.7568
>> exp([1 2 3; 4 5 6; 7 8 9])
ans =
    1.0e+003 *
    0.0027    0.0074    0.0201
    0.0546    0.1484    0.4034
    1.0966    2.9810    8.1031
```

- Entry-wise operations are preceded with a “.”

```
>> a = [1 2 3; 4 5 6; 7 8 9]; % <-comment sign
>> b = [10 11 12; 13 14 15; 16 17 18]; % ;<- no echo
>> a*b
ans =
    84    90    96
    201   216   231
    318   342   366
>> a.*b
ans =
    10    22    36
    52    70    90
    112   136   162
```

- Function calls work pretty much the same way. The most powerful aspect of Matlab is a large collection of functions commonly used in many engineering and scientific communities. Here are some of the signal processing commands we are going to use in this course. We will discuss their usages later.

residue	Partial-fraction expansion
fourier, ifourier	Symbolic forward and inverse Fourier Transform
laplace, ilaplace	Symbolic forward and inverse Laplace Transform
tf	Create transfer function for a system
bode	Magnitude and Phase response of a system
pzmap	Pole zero plot of a system
ss	Create state-space model for a system

ss2tf, tf2ss	Conversion between transfer function and state-space representation
lsim	Simulate time response of LTI models
eig	Eigenvalues of matrix
expm	Matrix exponential
c2d	Continuous to discrete time
ztrans, iztrans	Symbolic forward and Inverse Z-transform
residuez	Partial-fraction expansion in z-domain
filter	Discrete-time filtering
conv	Discrete-time convolution
butter	Butterworth filter design
cheby1	Chebyshev 1 filter design
impinvar	Impulse invariance method for analog-to-digital filter conversion
bilinear	Bilinear transformation method for analog-to-digital filter conversion
lp2bp, lp2bs, lp2hp, lp2lp	Frequency transformation
fir1, fir2	FIR filter design
freqs, freqs	Numerical Fourier transform
fft	Fast Fourier Transform

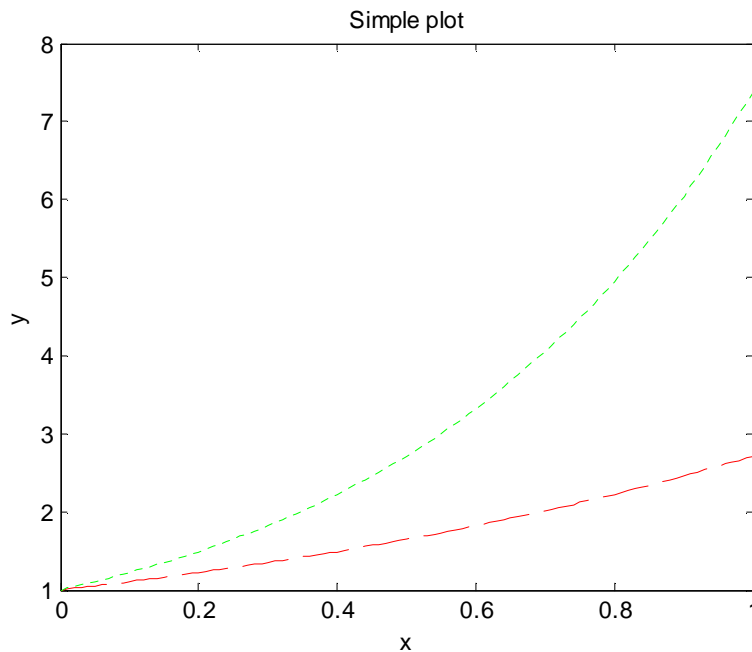
- Symbolic operations – most of the basic operations can be applied directly to symbolic variables. To actual evaluate an expression of symbolic variables, use the “subs” command.

```
>> syms a b c
>> s = sin(a)*b*exp(c)
s =
sin(a)*b*exp(c)
>> a=0.5;b=9;c=1;
>> subs(s)
ans =
    11.7289
>> a=0;b=0;c=0;
>> subs(s)
ans =
    0
```

#### 4. Miscellaneous

- Plotting

```
>> x=0:0.01:1; y=exp(x);  
>> x=0:0.01:1;  
>> plot(x,exp(x),'r--',x,exp(2*x),'g:');  
>> xlabel('x');  
>> ylabel('y');  
>> title('Simple plot');
```



- Saving the workspace

```
>> save everything.mat % save workspace to "everything.mat"  
>> load everything.mat % load "everything.mat"
```

- Saving what you type and the matlab output to a text file

```
>> diary foo.txt % output & response goto "foo.txt"  
>> ...  
>> diary off % stop logging
```

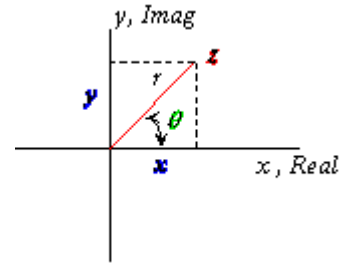
- You can also save all your commands in a file and execute it later on. The file should have a ".m" extension

## Quick Review of Continuous-time Signal and System

### 1. Complex Numbers

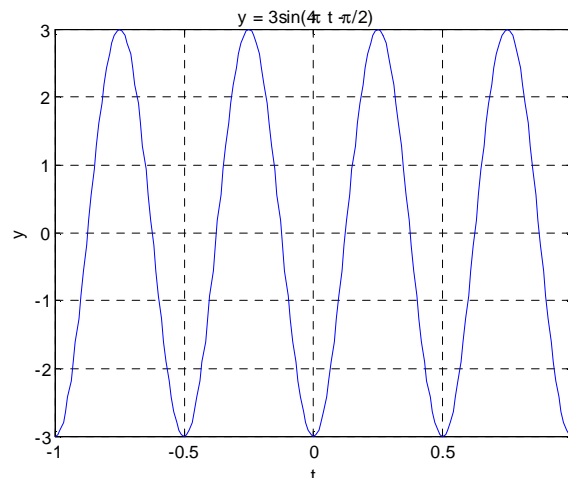
- $z = x + iy$  where  $i = \sqrt{-1}$
- Basic operations
  - Addition/Subtraction:  
 $(x + iy) + (a + ib) = (x + a) + i(y + b)$
  - Multiplication:  
 $(x + iy) \times (a + ib) = (xa - yb) + i(xb + ya)$
  - Division:  

$$\frac{x + iy}{a + ib} = \frac{(xa + yb) + i(ya - xb)}{a^2 + b^2}$$
  - Conjugation :  $\bar{z} = x - iy$
  - Magnitude (modulus) :  $|z| = \sqrt{z\bar{z}} = \sqrt{\text{Im}(z)^2 + \text{Re}(z)^2}$
  - Angle (argument):  $\angle z = \tan^{-1}\left(\frac{\text{Im}(z)}{\text{Re}(z)}\right)$
  - Euler Relation :  $z = |z|(\cos(\angle z) + i \sin(\angle z)) = |z|e^{i\angle z}$
  - De Moivre's Formula:  $z^n = |z|^n e^{in\angle z} = |z|^n (\cos(n\angle z) + i \sin(n\angle z))$



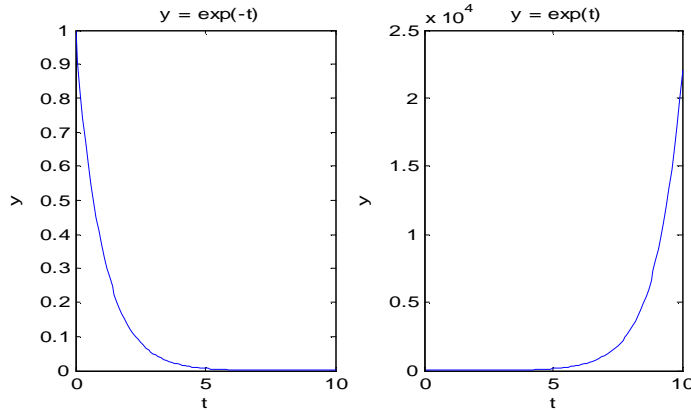
### 2. Common Signal Representation

- Sinusoid :  $y(t) = A \sin(2\pi f t + \theta) = A \sin(\omega t + \theta)$  where  $f$  (in Hz) or  $\omega$  (in rad/s) is the frequency,  $A$ 's the amplitude, and  $\theta$ 's the phase. eg  $y(t) = 3 \sin(4\pi t + \pi/2)$



- Complex sinusoid :  $y(t) = A \exp(i(\omega t + \theta)) = A \cos(\omega t + \theta) + iA \sin(\omega t + \theta)$

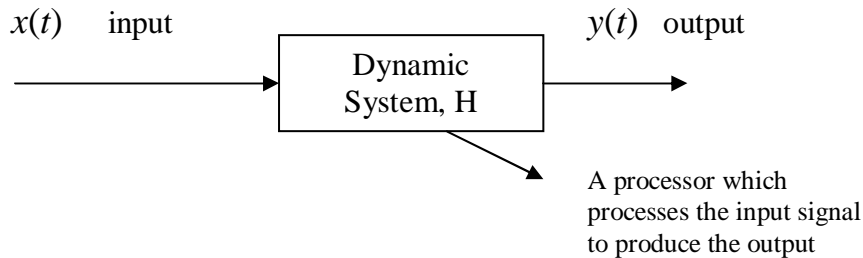
• Real Exponential



• Singular functions

- Brick wall function :  $\Pi(t) = \begin{cases} 1 & |t| \leq 0.5 \\ 0 & \text{otherwise} \end{cases}$
- Impulse function :  $\delta(t) = \lim_{\epsilon \rightarrow 0} \left( \frac{1}{2\epsilon} \Pi\left(\frac{t}{2\epsilon}\right) \right)$ 
  - $\delta(t)$  is zero everywhere, except at  $t=0$ , which is infinity.
  - $\int_{-\infty}^T \delta(t) dt = 1$  for any positive T.
  - $\int_{-\infty}^{\infty} \delta(t)x(t) dt = x(0)$
- Step function :  $u(t) = \int_{-\infty}^t \delta(\lambda) d\lambda$

3. Systems



• Properties

- Linear :  $H(ax_1(t) + bx_2(t)) = aH(x_1(t)) + bH(x_2(t))$
- Time-Invariant : if  $y(t) = H(x(t))$ , then  $y(t - \tau) = H(x(t - \tau))$
- Causal :  $H(x(t_o))$  depends only on  $x(t), t \leq t_o$

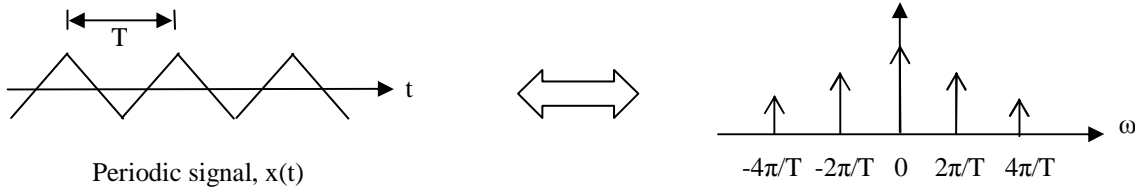
- The output of any Linear Time-Invariant System can be computed by convolving  $x(t)$  with a function  $h(t)$ , called the impulse response which is the output of the system when using the impulse function as input.

$$y(t) = x(t) * h(t) = \int_{-\infty}^{\infty} h(\tau)x(t - \tau)d\tau$$

- Examples of LTI systems include any system described by ordinary (constant coefficient) differential equations – that includes all of the “lumped” electric circuits with resistors, capacitors, and inductors.

4. Transform Methods – methods that transform a system from one domain (time-domain) to another domain (frequency-domain) in such a way that the analysis becomes easier. In EE 421, you have learnt two transform domain methods:

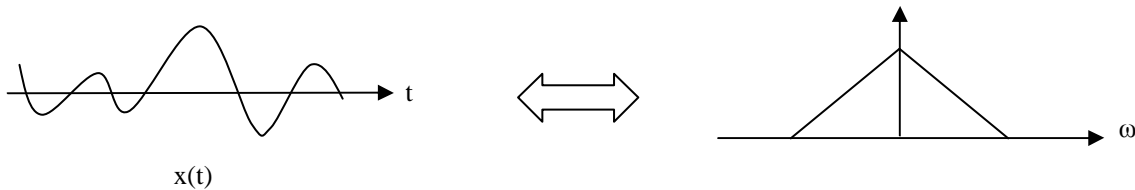
- Fourier Series – continuous-time periodic signals



$$\text{Analysis : } X_k = \frac{1}{T} \int_{-T/2}^{T/2} x(t)e^{-jk\frac{2\pi}{T}t} dt$$

$$\text{Synthesis : } x(t) = \sum_{k=-\infty}^{\infty} X_k e^{jk\frac{2\pi}{T}t}$$

- Fourier Transform – general continuous-time signals



$$\text{Analysis : } X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t} dt$$

$$\text{Synthesis : } x(t) = \int_{-\infty}^{\infty} X(j\omega)e^{j\omega t} d\omega$$

-- derived from Fourier series by letting  $T$  goes to infinity and replacing the summation by integration.