Signals and Systems

Lab Assignment I

Mohammed Abed (21903608)

Part 1:

- **a)** For a=[3.2 34/7 -6 24], the output is a row, and for a=[3.2; 34/7; -6; 24], the output is a row.
- **b)** Using the command window: inputs ending with a semicolon are saved to the variable and not displayed, however when the semicolon is not used, the output is displayed. Using the script, if the input is not terminated with (;), it gives an error.
- **c)** Using the 'tic' and 'toc':
    - Elapsed time when not using a semicolon is 0.000757 seconds.
    - When using it for one of the vectors is 0.000315 seconds.
    - When using it for both vectors is 0.000066 seconds.

  Using (;) is useful to suppress the command window output. That is why it is the fastest to use (;) when the run speed matters.

- **d)** Message received is: "Error using * Incorrect dimensions for matrix multiplication", and the reason is mismatch between the matrices' dimensions. Matlab treats the vectors as 1x0 matrices, and since the number of the columns of the first matrix is NOT equal to the number of rows of the second, it gives an error.
- **e)** When the dot (.) is added, Matlab computes the element-wise multiplication, and the result does not change when reversing the variable since the operation is commutative.
- **f)** c = -2436.1. Matlab computes the matrix multiplication of a row and a column. Which is a single number (a 1x1 matrix).

- **g)** Result is a 4x4 matrix:

$$\begin{bmatrix} 0.0186 & 0.0154 & 0.0160 & -0.3264 \\ 0.0282 & 0.0233 & 0.0243 & -0.4954 \\ -0.0348 & -0.0288 & -0.0300 & 0.6120 \\ 0.1392 & 0.1152 & 0.1200 & -2.4480 \end{bmatrix}$$

- **h)** It creates a 1x101 matrix with inputs that are evenly spaced between 1 and 2. And the delta between every two consecutive elements is 0.01
- **i)** Elapsed time is 0.000706 seconds.
- **j)** Elapsed time is 0.001695 seconds.
- **k)** Elapsed time is 0.002101 seconds. Method 1 is the most efficient.
- **l)** Matlab treats the 1x17 array (a=[0:pi/8:2pi]) as the domain of the sin function. Accordingly it iterates through the elements of the domain and outputs the range.
- **m)** The commands **plot(x)** and **plot(t,x)** outputs the cosine function. However, **plot(x,t)** outputs the arccosine (inverse of cosine), because the arguments are reversed.
- **n)** The '+-' and '+' are used to show the points on the graph.
- **o)** 26 time points.
- **p)** t = linspace(0,1,26)
- **q,r,s,t,u,v)**

The output (for illustration) for the first 't' array is:

$[\ 0.8660\ \ 0.9632\ \ 0.9998\ \ 0.9736\ \ 0.8862\ \ 0.7431\ \ 0.5534\ \ 0.3289\ \ 0.0837\ \ -0.1668\ \ -0.4067\ \ -0.6211\ \ -0.7965\ \ -0.9219\ \ -0.9893\ \ -0.9945\ \ -0.9373\ \ -0.8211\ \ -0.6534\ \ -0.4446\ \ -0.2079\ \ 0.0419\ \ 0.2890\ \ 0.5180\ \ 0.7145\ \ 0.8660\ ]_{1x26}$

The best plot for continuous x(t) is $x_2(t=t_2)$ in which t_2 = [0.01,0.02,0.03,....,1] because it has the highest number of points, so the output is more accurate as Matlab computes the $x(t_i)$ for more t_i values.

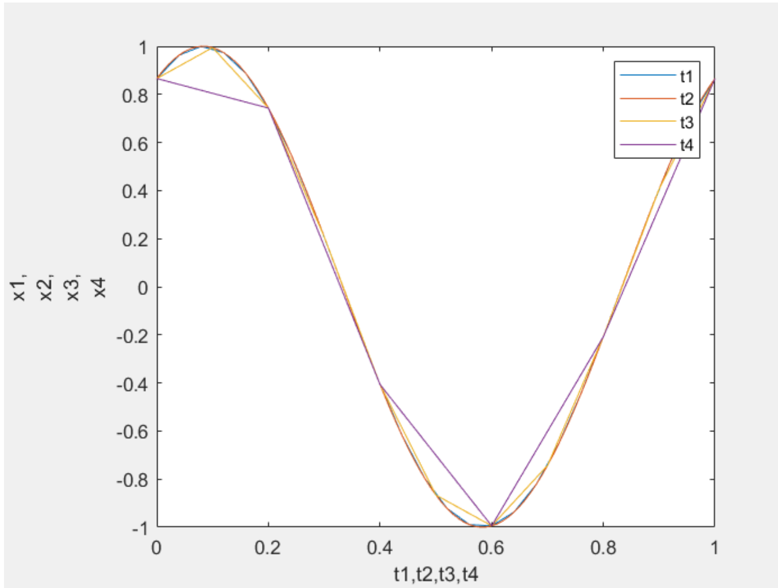


Figure 1: A Sinusoid of Different 't' Domains

- **w)** Matlab computes the values of the function at all t's given in the array, and finds the line of the best fit connecting all the points.
- **x) Plot** command is used to represent continuous functions, and **stem** is used to output discrete functions. In other words, **stem** does not graphically fill the values between points specified in the domain.

Part 2:

- a) **Soundsc** is not appropriate to listen to a discrete signal, however when **sound** was used on the
- b,c,d) $\cos(2\pi f_0 t)$

  As frequency increases, the pitch increases, and the sound becomes sharper (The sinusoid gains more energy).

code:

```
t= [0:1/8192:1];

f_01 = 440; %hz

f_02 = 687;

f_03 = 883;

x01= cos((2*f_01*pi).*t);

x02= cos((2*f_02*pi).*t);

x03 = cos((2*f_03*pi).*t);

figure()

plot(t,x1);

%sound(x01)

%sound(x02)

sound(x03)
```

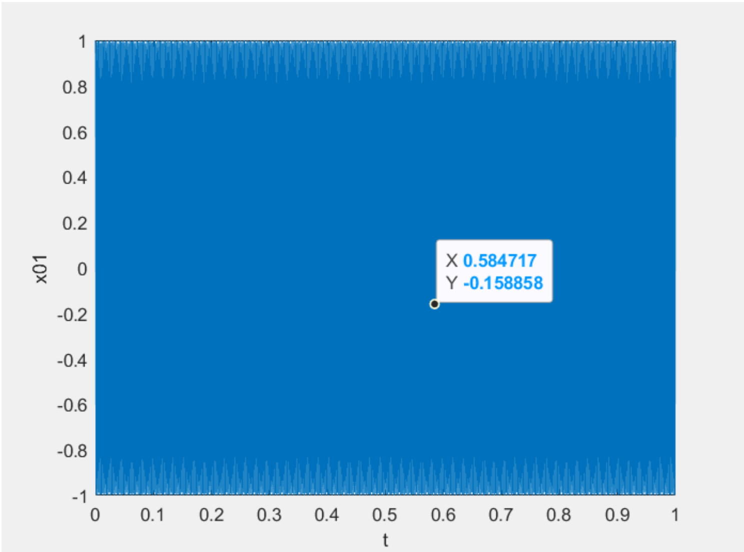


Figure 2: The Sinusoid $\cos(2\pi f_0 t)$

- $x_2(t) = \exp(-at).\cos(2\pi f_0 t)$

The plot is damped by the sinusoid, which coincides with the sound, because the sound fades aways smoothly.

$X_1(t)$ resembles a flute sound, and $x_2(t)$ resembles a piano sound.

As '**a**' increases, the duration is less (the sound wave gets damped faster).

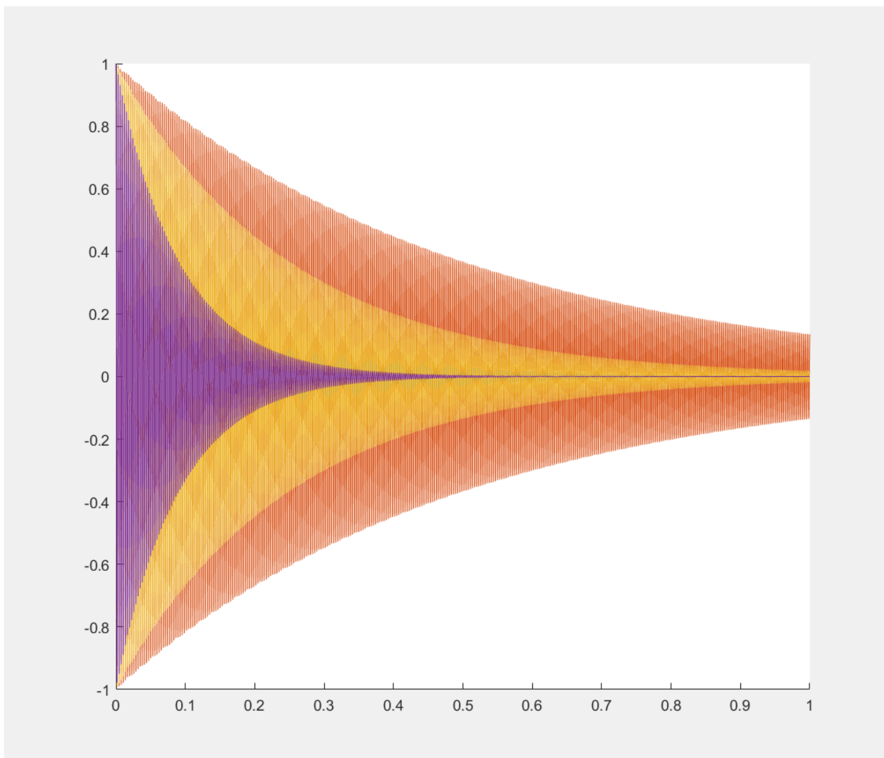


Figure 3: The Sinusoid $\exp(-at).\cos(2\pi f_0 t)$ with Different Damping Coefficients

Code:

f_1=330; %hz

a_1= 7;

```
a_2 = 2;

a_3 = 4;

a_4 = 11;

x20 = (exp(-a_1.*t)).*(cos((2*f_1*pi).*t));

figure()

hold on

plot(t,x20);

sound(x20)

%adding e^-(ax) damps the sinusoid

x_21= (exp(-a_2.*t)).*(cos((2*f_1*pi).*t));

plot(t,x_21);

x_22=(exp(-a_3.*t)).*(cos((2*f_1*pi).*t));

plot(t,x_22);

x_23 = (exp(-a_4.*t)).*(cos((2*f_1*pi).*t));

plot(t,x_23);

%sound(x_21);

%sound(x_22)

sound(x_23)
```

- $\cos(2\pi f_1 t)\cos(2\pi f_0 t)$

  Multiplying with a cosine (of low frequency) makes the sound oscillate (the sound's intensity is changing). In other words, due to the low frequency cosine, the frequency of $x_3(t)$ is varying.

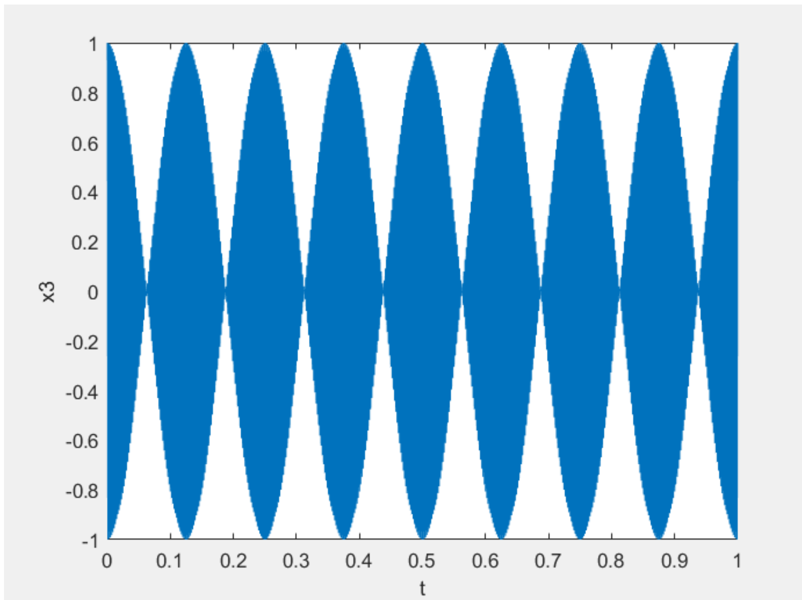
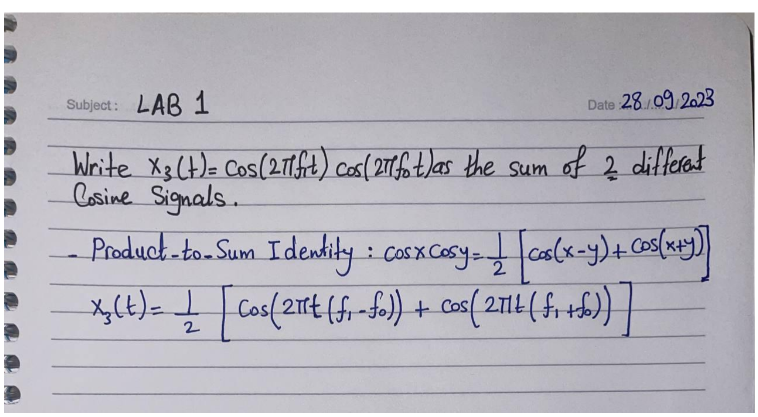


Figure 4: The Sinusoid $\cos(2\pi f_1 t)\cos(2\pi f_0 t)$



Subject: LAB 1 Date 28/09/2023

Write $x_3(t) = \cos(2\pi f_1 t)\cos(2\pi f_0 t)$ as the sum of 2 different Cosine Signals.

- Product-to-Sum Identity : $\cos x \cos y = \frac{1}{2}\left[\cos(x-y) + \cos(x+y)\right]$

$x_3(t) = \frac{1}{2}\left[\cos(2\pi t(f_1 - f_0)) + \cos(2\pi t(f_1 + f_0))\right]$

Figure 5: Sum of Two different Cosine Signals

Code:

```
f0=510;

f1=4;

x_3 = cos((2*pi*f0).*t).*cos((2*pi*f1).*t);

sound(x_3)

figure()

plot(t,x_3)
```

Part 3:

- When ‘a’ is halved the sound lasts longer and it gets deeper. However, when doubling ‘a’ the sound is played faster and it gets squickier.

  Code:

```
x_4 = cos((pi*rand_a.*(t.*t)));

%sound(x_4)

x_41 = cos((pi*(rand_a/2).*(t.*t)));

%sound(x_41)          %when a is halved the sound lasts longer and it gets

%deeper

x_42= cos((pi*(2*rand_a).*(t.*t)));

%sound(x_42)

%when doubling rand_a the sound is played faster and it gets squickier
```

```
x5 = cos((2*pi).*((-500.*t.*t)+(1600.*t)));
```

Part 4:

- As Phase increases the sound wave travels faster which makes the sound sped up.

Code:

```
phi1 = 0;

phi2 = (pi/4);

phi3 = (pi/2);

phi4 = (0.75*pi);

phi5 = pi;

x61 = cos((2*pi*1665).*(t) +phi1);

x62 = cos((2*pi*1665).*(t)) +phi2);

x63 = cos((2*pi*1665).*(t) +phi3);

x64 = cos((2*pi*1665).*(t) +phi4);

x65 = cos((2*pi*1665).*(t) +phi5);

sound(x61)

sound(x62)

sound(x63)

sound(x64)

sound(x65)
```

Part 5:

Part 5 Solution:

Using the identity $\cos(x+y) = \cos(x)\cos(y) - \sin(x)\sin(y)$

$$x_3(t) = A_3\cos(2\pi f_0 t)\cos(\phi_3) - \sin(2\pi f_0 t)\sin(\phi_3)$$

$$= \left[A_1\cos(2\pi f_0 t)\cos(\phi_1) - A_1\sin(2\pi f_0 t)\sin(\phi_1)\right] + \left[A_2\cos(2\pi f_0 t)\cos(\phi_2) - A_2\sin(2\pi f_0 t)\sin(\phi_2)\right]$$

$\rightarrow$ Taking $\cos(2\pi f_0 t)$, $\sin(2\pi f_0 t)$ as Factors

$$= \cos(2\pi f_0 t)\left[A_1\cos(\phi_1) + A_2\cos(\phi_2)\right] - \sin(2\pi f_0 t)\left[A_1\sin(\phi_1) + A_2\sin(\phi_2)\right]$$

Matching terms: $A_3\cos(\phi_3) = A_1\cos(\phi_1) + A_2\cos(\phi_2)$

$A_3\sin(\phi_3) = A_1\sin(\phi_1) + A_2\sin(\phi_2)$

$$A_3^2 = (A_3)^2\cos^2(\phi_3) + (A)^2\sin^2(\phi_3) = A^2\left[\cos^2(\phi_3) + \sin^2\phi_3\right] = A_3^2 \cdot 1$$

$$\sqrt{(A_3)^2} = \boxed{\sqrt{\left[A_1\cos(\phi_1) + A_2\cos(\phi_2)\right]^2 + \left[(A_1\sin(\phi_1) + A_2\sin(\phi_2)\right]^2}} \quad \# A$$

To find ϕ_3:

$$\frac{\cancel{A}\sin(\phi_3)}{\cancel{A}\cos(\phi_3)} = \tan(\phi_3) = \frac{A_1\sin(\phi_1) + A_2\sin(\phi_2)}{A_1\cos(\phi_1) + A_2\cos(\phi_2)}$$

$$\phi_3 = \arctan\left[\frac{A_1\sin(\phi_1) + A_2\sin(\phi_2)}{A_1\cos(\phi_1) + A_2\cos(\phi_2)}\right] \quad \# \phi$$

For $\phi_1 = \phi_2$, A_3 is maximized

If we plug $\phi_1 = \phi_2 \Longrightarrow \cos(\phi_1)\cos(\phi_2) = \cos^2(\phi_2)$

$\Longrightarrow \sin(\phi_1)\sin(\phi_2) = \sin^2(\phi_2)$

$$A_3 = \sqrt{\left[(A_1 + A_2)\cos^2(\phi)\right]^2 + \left[(A_1 + A_2)\sin^2(\phi)\right]^2}$$

$$A_3 = \sqrt{(A_1 + A_2)^2\left[\cos^2(\phi) + \sin^2(\phi)\right]^2} = \boxed{A_1 + A_2} \quad \#$$

Maxima

Subject : Date : ....../....../......

To find a Minima, Let's assume $A_3 = A_1 - A_2$ is a minima

$A_3 > 0$ and Let $\Delta\phi = \pi = \phi_2 - \phi_1 = \phi_1 - \phi_2$

(Difference in phase is π)

then we have: $\cos(\phi_1) = -\cos(\phi_2)$

$\sin(\phi_1) = -\sin(\phi_2)$

$$A_3 = \sqrt{[A_1\cos(\phi_1) + A_2\cos(\phi^2)]^2 + [A_1\sin(\phi_1) + A_2\sin(\phi^2)]^2}$$

$$= \sqrt{[-A_1\cos(\phi_1) + A_2\cos(\phi^2)]^2 + [-A_1\sin(\phi_1) + A_2\sin(\phi^2)]^2}$$

$$= \sqrt{A_1^2\cos^2(\phi_1) - 2A_1A_2\cos(\phi_1)\cos(\phi_2) + A_2^2\cos^2(\phi_2) + A_1^2\sin^2(\phi_1) + -2A_1A_2\sin(\phi_1)\sin(\phi_2) + \sin^2(\phi_1)}$$

$$= \sqrt{A_1^2\underbrace{[\cos^2(\phi_1) + \sin^2(\phi_1)]}_{1} + A_2^2\underbrace{[\cos^2(\phi_1) + \sin^2(\phi_1)]}_{1} - 2AB\underbrace{[\cos(\phi_1)\cos(\phi_2) - \sin(\phi_1)\sin(\phi_1)]}_{1}}$$

$$= \sqrt{A_1^2 + A_2^2 - 2AB} = \sqrt{(A_1 - A_2)^2}$$

$$= \boxed{|\underbrace{A_2 - A_1}_{\text{Max Negative number}}|}$$

Minima