
```
% CONTINUOUS TIME FOURIER TRANSFORM OF ONES-SIDED EXPONENTIAL SIGNAL

%1)
help fourier;
help ifourier;

%2)
syms t;
a=1;
f=exp(-a*t)*heaviside(t);
T=-5:0.01:5;
y1=subs(f,T);
subplot(121);
plot(T,y1);

F=fourier(f);
y2=subs(F,T);
subplot(122);
plot(T,y2);

%3)
disp(F);
% w is the default variable in which is return by the function "fourier"

%4)
W=-10:10;
subplot(121);
y3=abs(subs(F,W));
plot(W,y3,'linewidth',2);
subplot(122);
y4=angle(subs(F,W));
plot(W,y4,'linewidth',2);

%5)
x1=exp(-1*t)*heaviside(t);
x2=exp(-4*t)*heaviside(t);
T=-5:0.01:5;
subplot(321);
y5=subs(x1,T);
plot(T,y5,'linewidth',2,'color','b');
subplot(322);
y6=subs(x2,T);
plot(T,y6,'linewidth',2,'color','r');

F1=fourier(x1);
F2=fourier(x2);

subplot(323);
plot(T,subs(F1,T),'linewidth',2,'color','b');
subplot(324);
plot(T,subs(F2,T),'linewidth',2,'color','r');
```

```

subplot(325);
y7=abs(subs(F1,T));
y8=abs(subs(F2,T));
plot(T,y7,'linewidth',2,'color','b'); hold ON;
plot(T,y8,'linewidth',2,'color','r'); hold OFF;

subplot(326);
y9=angle(subs(F1,T));
y10=angle(subs(F2,T));
plot(T,y9,'linewidth',2,'color','b'); hold ON;
plot(T,y10,'LineWidth',2,'color','r'); hold OFF;

% UNDERSTANDING AN IDEAL LOW PASS FILTER

%1)
wc=20;
A=2;
syms w;
LPF=A*(heaviside(w+wc)-heaviside(w-wc));
lpf=ifourier(LPF);

T=-10:0.01:10;
T(1001)=0.01; %This is done in order to resolve the zero division error.
subplot(211);
plot(T,subs(LPF,T),'linewidth',2);
subplot(212);
plot(T,subs(lpf,T),'linewidth',2);

%2)
lim=limit(lpf,0);
disp(lim);

%3
%{
lpf(t) is non-causal because for a causal signal at t<0 it should have
value=0. But as we can see lpf(t) has non-zero value at t<0.
%}

% AMPLITUDE MODULATION

%1)
syms t;
m=exp(-0.5*abs(t-1))+exp(-0.5*abs(t+1));
T=-5:0.01:5;
F=fourier(m);
subplot(121);
plot(T,subs(m,T),'linewidth',2);
subplot(122);
plot(T,subs(F,T),'linewidth',2);

%2)
w0=20;
s=m*cos(w0*t);
W=-40:0.1:40;

```

```

T=-10:0.1:10;
Fs=fourier(s);
subplot(121);
plot(T,subs(s,T), 'linewidth', 2);
subplot(122);
plot(W,subs(Fs,W), 'linewidth', 2);

%3)
%{
From the property of DTFT we can say that if  $y[n]=\text{convolution}(x[n],h[n])$ 
Their DTFT follow
 $Y(W)=X(W)*H(W)$ 
This is the multiplicative proeprty and can be observed in the plot.
%}

%4)
d=s*cos(w0*t);
Dw=fourier(d);
W=-60:0.1:60;
subplot(121);
plot(T,subs(d,T), 'linewidth', 2);
subplot(122);
plot(W,subs(Dw,W), 'linewidth', 2);

%5)
W=-40:0.1:40;
R=Dw*LPF;
subplot(111);
plot(W,subs(R,W), 'linewidth', 2);

%6)
T=-5:0.01:5;
r=ifourier(R);
plot(T,subs(r,T), 'linewidth', 2);

%7)
subplot(121);
plot(T,subs(m,T), 'linewidth', 2);
subplot(122);
plot(T,subs(r,T), 'linewidth', 2);

%8)
%{
We can observe that  $r(t)$  has breaks at its peak. Also, it is smoother than
 $m(t)$ . Both  $r(t)$  and  $m(t)$  have the same amplitude as expected.
%}

% THE DISCRETE-TIME FOURIER TRANSFORM

%1)
% Function named DTFT_Analysis is made below.

% DTFT OF A RECTANGULAR PULSE

```

```

%1)
N=3;
n=-2*N:2*N;
x=(abs(n)<=N);

%2)
subplot(111);
stem(n,x,'linewidth',2);

%3)
syms w;
X=DTFT_Analysis(x,n,w);
disp(X);

%4)
W=-2*pi:0.01:2*pi;
subX=subs(X,W);
flipX=subs(X,-W);
if(flipX==subX)
    disp("X(e^jw) is even over the range");
else
    disp("No, X(e^jw) is not even over the range");
end

if(subX==real(subX))
    disp("X is real.");
else
    disp("X is not real.");
end

%{
From looking at x[n] we can see that it is symmetric about the y-axis and
hence X is even.
Also, X is real, as displayed.
%}

%5)
plot(W,subX);
xticks(-2*pi:pi:2*pi);

%6)
W1=-4*pi:0.01:4*pi;
plot(W1,subs(X,W1));
xticks(-4*pi:pi:4*pi);

%{
We can observe from the graph that X is periodic at a period of 6.28 i.e.
2*pi.
%}

%7)
syms w;
N1=6;
n1=-2*N1:2*N1;

```

```

x1=(abs(n1)<=N1);
X1=DTFT_Analysis(x1,n1,w);
disp(X);
disp(X1);
W1=-2*pi:0.01:2*pi;
subX1=subs(X1,W1);
subplot(121);
plot(W,subX);
xticks(-2*pi:pi:2*pi);
subplot(122);
plot(W1,subX1);
xticks(-2*pi:pi:2*pi);
%{
From the plotted graph and the values displayed we can see that apart the
extra terms in X1. X1(N=6) has a more wiggly nature, more oscillating and a
higher amplitude.
%}

%8)
N=6;
n=-2*N:2*N;
y=(abs(n)<=N).* (2*(mod(n,2)==0)-1);

%9)
subplot(111);
stem(n,y,'linewidth',2);
xticks(-2*N:N/2:2*N);

%10)
syms w;
Y1=DTFT_Analysis(y,n,w);

%11)
subY1=subs(Y1,W1);
subplot(211);
plot(W1,subX1);
xticks(-2*pi:pi/2:2*pi);
subplot(212);
plot(W1,subY1);
xticks(-2*pi:pi/2:2*pi);
%{
From this we observe that the peaks of the two DTFT occur at alternating
locations.
max of one is the min of other and vice-versa.
%}

% TIME EXPANSION AND INVERSE DTFT

%1)
% Nothing to display or explain.

%2)
k=3;
N=3;

```

```

xk=x;
n=-2*N:2*N;
for i=1:length(x)
    if(rem(n(i),k)==0)
        xk(i)=xf(n(i)/k,N);
    else
        xk(i)=0;
    end
end

%3)
x3k=xk;
subplot(111);
stem(n,x3k,'linewidth',2);

%4)
syms w;
X3=DTFT_Analysis(x3k,n,w);
W=-2*pi:0.01:2*pi;
subX3=subs(X3,W);
plot(W,subX3);
xticks(-2*pi:pi:2*pi);
%{
We can see that the frequency is 3 times of that of x.
hence it is indeed. X(e^3jw)
%}

%5)
% Function Inverse_DTFT is made below

%6)
N=3;
n=-2*N:2*N;
syms w;
invX=Inverse_DTFT(X,n,w);
stem(n,invX,'linewidth',2);
%{
From the plot(stem) we can see that the inverse of X is the same as x
defined initially.
%}

% DISCRETE FOURIER TRANSFORM (DFT)

%1)
% N_DFT function is made below

%2)
N=8;
n=0:20;
x=(n>=0).*(n<=3)+(n>=4).*(n<=7).*(-n);
subplot(131);
stem(n,x,'linewidth',2);
Xcap=N_DFT(x,N);
subplot(132);

```

```

stem(0:(N-1),Xcap,'linewidth',2);
subplot(133);
stem(0:(2*N-1),N_DFT(x,N*2),'linewidth',2);

%3)
Xfft=fft(x,(N*2));
subplot(111);
stem(Xfft,'linewidth',2);

% RELATIONSHIP BETWEEN DFT DTFT

%1)
syms w;
n=0:(N-1);
X=DTFT_Analysis(x,n,w);
W=n*2*pi/N;
Y=subs(X,W);

%2)
% Plotting Real Parts
subplot(121);
stem(real(Xcap),'linewidth',2,'color','b'); hold ON;
stem(real(Y),'linewidth',1,'color','r'); hold OFF;
% Plotting Imaginary Parts
subplot(122);
stem(n,imag(Xcap),'linewidth',2,'color','b'); hold ON;
stem(n,imag(Y),'linewidth',1,'color','r'); hold OFF;
% From the graph we see that both the imaginary and real parts are
% overlapping.

% INVERSE DFT

%1)
xinv=IDFT(Xcap,N);

%2)
subplot(121);
stem(n,xinv,'LineWidth',2);
subplot(122);
stem(n,ifft(Xcap,N),'linewidth',2);

% FUNCTIONS

function X=DTFT_Analysis(x,t,w)
INF=10;
len=length(t);
X=0;
for i=1:len
    X=X+x(i)*exp(-(1i)*w*t(i));
end
end

function x=xf(n,N)
x=abs(n)<=N;

```

```

end

function x=Inverse_DTFT(X,n,w)
X1=X*exp((1i)*w*n);
x=int(X1,0,2*pi)/(2*pi);
end

function Xcap=N_DFT(x,N)
y=zeros(1,N);
for i=1:min(N,length(x))
    y(i)=x(i);
end
Xcap=zeros(1,N);
for k=0:N-1
    for n=0:N-1
        Xcap(k+1)=Xcap(k+1)+y(n+1)*exp(-1*(1i)*k*(2*pi/N)*n);
    end
end
end

function x=IDFT(Xcap,N)
x=zeros(N);
for n=0:N-1
    for k=0:N-1
        x(n+1)=x(n+1)+Xcap(k+1)*exp((1i)*k*n*(2*pi)/N)/N;
    end
end
end

```

--- help for sym/fourier ---

FOURIER Fourier integral transform.

F = FOURIER(f) is the Fourier transform of the symbolic expression or function f with default independent variable x. If f does not contain x, then the default variable is determined by SYMVAR. By default, the result F is a function of w. If f = f(w), then F is returned as a function of the variable v, F = F(v).

*By definition, F(w) = c*int(f(x)*exp(s*i*w*x),x,-inf,inf).*

You can set the parameters c,s to any numeric or symbolic values by setting the preference SYMPREF('FourierParameters',[c,s]). By default, the values are c = 1 and s = -1.

F = FOURIER(f,v) returns F as a function of the variable v instead of the default variable w:

*F(v) = c*int(f(x)*exp(s*i*v*x),x,-inf,inf).*

F = FOURIER(f,u,v) treats f as a function of the variable u instead of the default variable x:

*F(v) = c*int(f(u)*exp(s*i*v*u),u,-inf,inf).*

Examples:

syms t v w x f(x)

```

fourier(1/t)      returns   -pi*sign(w)*li
fourier(exp(-x^2),x,t)  returns   pi^(1/2)*exp(-t^2/4)
fourier(exp(-t)*heaviside(t),v)  returns   1/(1+v*li)
fourier(diff(f(x)),x,w)  returns   w*fourier(f(x),x,w)*li

```

See also SYM/IFOURIER, SYM/LAPLACE, SYM/HTRANS, SYM/ZTRANS, SUBS, SYMPREF.

Documentation for sym/fourier
doc sym/fourier

--- help for sym/ifourier ---

IFOURIER Inverse Fourier integral transform.

f = IFOURIER(F) is the inverse Fourier transform of the symbolic expression or function F with default independent variable w. If F does not contain w, then the default variable is determined by SYMVAR. By default, the result f is a function of x. If F = F(x), then f is returned as a function of the variable t, f = f(t).

By definition,

*f(x) = abs(s)/(2*pi*c) * int(F(w)*exp(-s*i*w*x),w,-inf,inf).*

You can set the parameters c,s to any numeric or symbolic values by setting the preference SYMPREF('FourierParameters',[c,s]).

By default, the values are c = 1 and s = -1.

f = IFOURIER(F,u) returns f as a function of the variable u instead of the default variable x:

*f(u) = abs(s)/(2*pi*c) * int(F(w)*exp(-s*i*w*u),w,-inf,inf).*

f = IFOURIER(F,v,u) treats F as a function of the variable v instead of the default variable w:

*f(u) = abs(s)/(2*pi*c) * int(F(v)*exp(-s*i*v*u),v,-inf,inf).*

Examples:

```

syms t u v w f(x)
ifourier(w*exp(-3*w)*heaviside(w))  returns  1/(2*pi*(-3+x*li)^2)
ifourier(1/(1 + w^2),u)    returns  exp(-abs(u))/2
ifourier(v/(1 + w^2),v,u)  returns  -(dirac(1,u)*li)/(w^2+1)
ifourier(fourier(f(x),x,w),w,x)  returns  f(x)

```

See also SYM/FOURIER, SYM/LAPLACE, SYM/HTRANS, SYM/ZTRANS, SUBS, SYMPREF.

Documentation for sym/ifourier
doc sym/ifourier

*1/(1 + w*li)*

40/pi

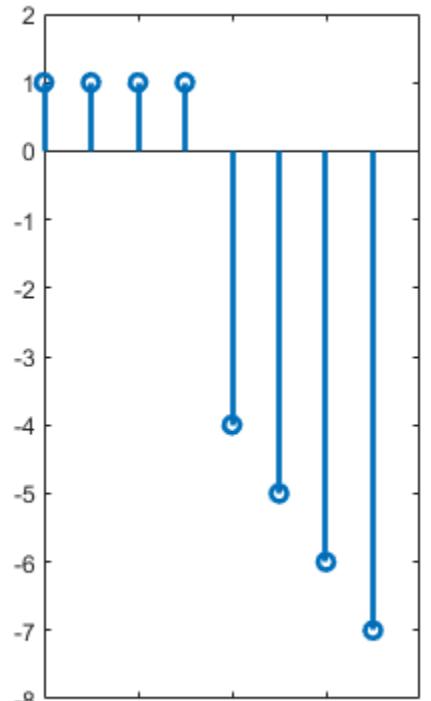
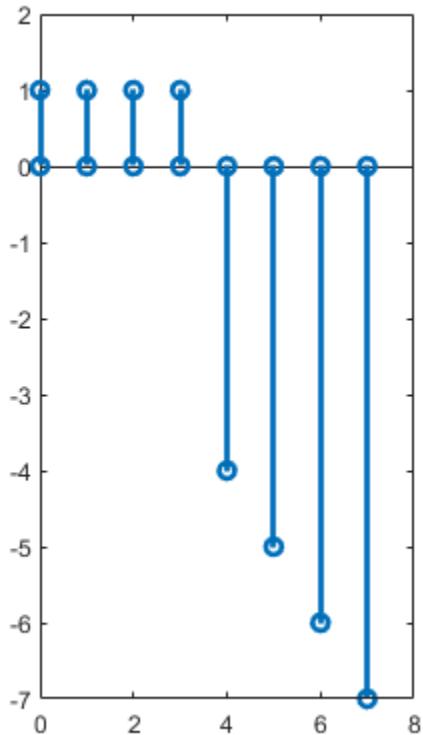
*exp(-w*li) + exp(w*li) + exp(-w*2i) + exp(w*2i) + exp(-w*3i) + exp(w*3i) + 1*

$X(e^{jw})$ is even over the range

X is real.

$$\exp(-w \cdot 1i) + \exp(w \cdot 1i) + \exp(-w \cdot 2i) + \exp(w \cdot 2i) + \exp(-w \cdot 3i) + \exp(w \cdot 3i) + 1$$

$$\exp(-w \cdot 1i) + \exp(w \cdot 1i) + \exp(-w \cdot 2i) + \exp(w \cdot 2i) + \exp(-w \cdot 3i) + \exp(w \cdot 3i) + \\ \exp(-w \cdot 4i) + \exp(w \cdot 4i) + \exp(-w \cdot 5i) + \exp(w \cdot 5i) + \exp(-w \cdot 6i) + \exp(w \cdot 6i) + 1$$



Published with MATLAB® R2023a