Matlab'de Sinyal Tanımlama ve Tersine Fourier Dönüşümü

Belirli bir zaman aralığı için tanımlanan bir sinyali matlab programında bir kod yazıp inceleyelim

Örnek: Açısal frekans ekseninde -10π noktasından başlayarak artan ve sıfır noktasında maksimum değeri olan 1'e ulaşan ardından azalarak ilerleyip +10π noktasında sıfır değerine ulaşan fonksiyonu matlab programı ile çizip inceleyiniz.

Kodumuzu yazmaya açısal frekans ekseni ve kodun hızlı çalışması için boş bir vektör olarak $X_c(\omega)$ fonksiyonunu tanımlayarak başlayalım. Boş vektör tanımını zeros komutunu ve tanımladığımız açısal frekans ekseninin boyunu kullanarak yapacağız.

```
clc;clear all;close all;
w=-10*pi:pi/10:10*pi;
Xcw=zeros(1,length(w));
```

`zeros(1,length(w))` komutu ile satır sayısı 1 sütun sayısı `w` kadar olan, sıfırlardan oluşan bir vektör tanımladık, komutu daha iyi anlamak için komut penceresine `zeros(5)` ve `zeros(1,5)` komutlarını yazıp çalıştırınız. Fonksiyonu tanımlamak için öncelikle açısal frekans eksenini taramamız ve -10π ile 0 aralığını bulmamız gerekiyor bu işlemler için sırası ile for ve if komutlarını kullanacağız. Daha önceden yazmış olduğumuz kodu aşağıdaki gibi güncelleyiniz.

```
clc;clear all;close all;
w=-10*pi:pi/10:10*pi;
Xcw=zeros(1,length(w));


for i=1:length(w)
    if -10*pi<=w(i) && w(i)<0
```

Yukarıdaki kodda `i=1:length(w)komutu` ile bütün açısal frekans eksenini taradık ve `if -10*pi<=w(i) && w(i)<0` komutu ile açısal frekansın -10 π ve 0 aralığında olduğu değerleri bulduk, bu işlemi yaparken `&&` skaler ve komutunu kullandık. Şimdi bu aralık için fonksiyonumuzun birinci kısmını tanımlayacağız bu tanımlamayı yapabilmek için örnekte verilen bilgilerden doğru denklemini yazarak yapacağız. Doğru denklemini yazmak için $y=ax+b$ gösterimini kullanacağız, bu formülde y fonksiyonu a eğimi ve b'de kesişim noktasını göstermektedir. Sinyalin -10 π <=`w` <0 aralığında eğimi 1/10 π ve b değeride 1'e karşılık gelmekte, bu durumda kodumuzu aşağıdaki gibi devam ettirelim.

```
clc;clear all;close all;
w=-10*pi:pi/10:10*pi;
Xcw=zeros(1,length(w));


for i=1:length(w)
    if -10*pi<=w(i) && w(i)<0
        Xcw(i)=w(i)/(10*pi)+1;
```

Kodu yukarıdaki gibi düzenledikten sonra sinyalin ikinci koşulunu yani sıfır noktasında aldığı 1 değerini tanımlayalım. Bu koşulu yazmak için elseif komutunu kullanacağız.

```
clc;clear all;close all;
w=-10*pi:pi/10:10*pi;
Xcw=zeros(1,length(w));


for i=1:length(w)
    if -10*pi<=w(i) && w(i)<0
        Xcw(i)=w(i)/(10*pi)+1;
    elseif w(i)==0
            Xcw(i)=1;
```

Bir sonraki adımda ise sinyalimizin azalan kısmını yazacağız. Azalan bölge 0 ile 10 π aralında bu tanımı elseif komutu ile tanımlayacağız ve komutun altına sinyalin bilgilerini ekleyeceğiz bunun için yine $y=ax+b$ gösterimi ile sinyalimizi ifade edeceğiz, bu kısımda sinyal azalmakta olduğu için eğimimiz negatif çıkacaktır ve kesişim noktamız yine 1 olacaktır. Bu verilere göre kodumuzu aşağıdaki gibi devam ettirelim.

```
clc;clear all;close all;
w=-10*pi:pi/10:10*pi;
Xcw=zeros(1,length(w));


for i=1:length(w)
    if -10*pi<=w(i) && w(i)<0
        Xcw(i)=w(i)/(10*pi)+1;
    elseif w(i)==0
            Xcw(i)=1;
    elseif 0<w(i) && w(i)<=10*pi
                Xcw(i)=-(w(i)/(10*pi))+1;
```

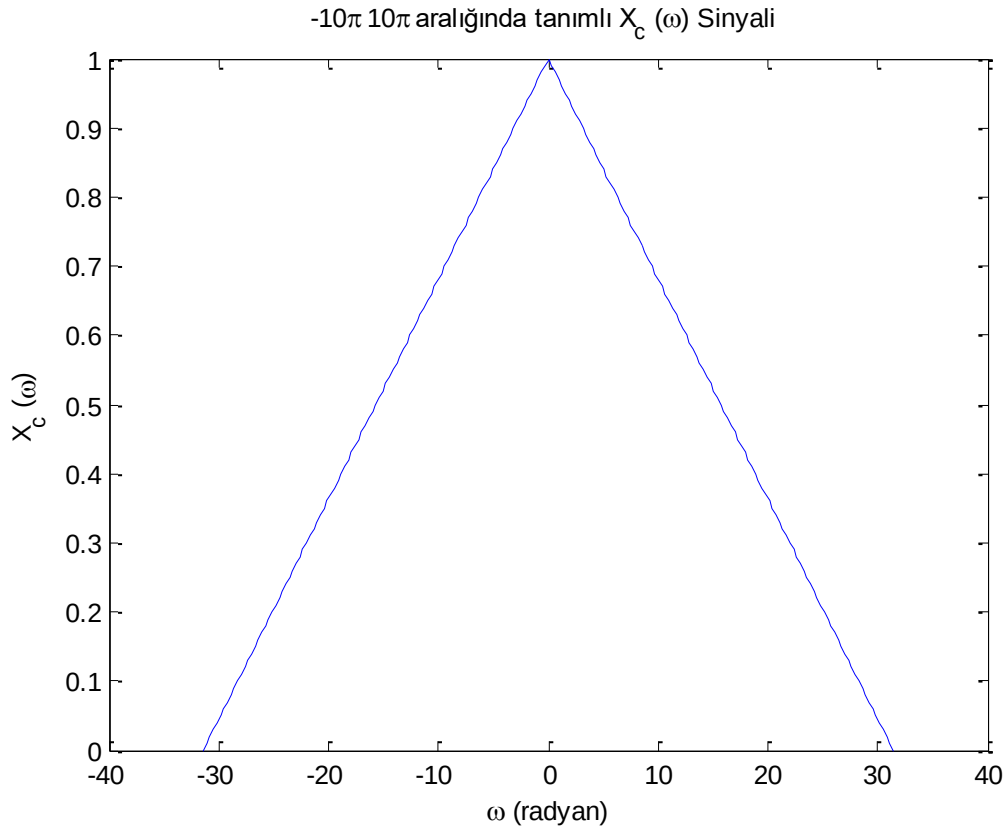
Son olarak sinyalin değer almadığı bölgeleri yani 0 olduğu yerleri tanımlayalım. Bunun için else komutunu kullanıp if döngüsünü end komutu ile bitirmemiz gerekiyor. Ardından for döngüsünde end komutu ile sonlandıra biliriz. Yazdığımız kodu aşağıdaki gibi devam ettirelim

```
for i=1:length(w)
    if -10*pi<=w(i) && w(i)<0
        Xcw(i)=w(i)/(10*pi)+1;
    elseif w(i)==0
            Xcw(i)=1;
    elseif 0<w(i) && w(i)<=10*pi
                Xcw(i)=-(w(i)/(10*pi))+1;
            else
            Xcw(i)=0;
    end
end
```

Şimdi yukarıda tanımladığımız kodu çizdirip inceleyelim. Bunun için plot komutunu kullanacağız.

```
clc;clear all;close all;
w=-10*pi:pi/10:10*pi;
Xcw=zeros(1,length(w));


for i=1:length(w)
    if -10*pi<=w(i) && w(i)<0
        Xcw(i)=w(i)/(10*pi)+1;
    elseif w(i)==0
            Xcw(i)=1;
    elseif 0<w(i) && w(i)<=10*pi
                Xcw(i)=-(w(i)/(10*pi))+1;
            else
            Xcw(i)=0;
    end
end
plot(w,Xcw)
xlabel('\omega (radyan)')
ylabel('X_c (\omega)')
title('-10\pi 10\pi aralığında tanımlı X_c (\omega) Sinyali ')
```

Kodumuzu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



Örnek: Bir önceki kodda yazılan sinyalin tersine Fourier dönüşümünü alıp inceleyiniz.

Bunun için ilk önce yukarıda yazdığımız koda zaman eksenini tanımlayalım. Bu tanımlamayı hali hazırda kullandığımız açısal frekans ekseninin eleman sayısını kullanarak -1s ve 1s aralığına linspace komutu ile yapalım. Sonrasında boş bir $x_c(t)$ vektörünü zeros komutunu kullanarak tanımlayalım. Aşağıdaki satırları yukarıda yazdığımız kodun altına ekleyerek devam edebiliriz.

```
title('-10\pi 10\pi aralığında tanımlı X_c (\omega) Sinyali ')


t=linspace(-1,1,length(w));
xct=zeros(1,length(t));
```

Tersine Fourier dönüşümünün matematiksel formülü aşağıdaki gibidir.

$$x_c(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty} x_c(w)e^{j\omega t}dw$$

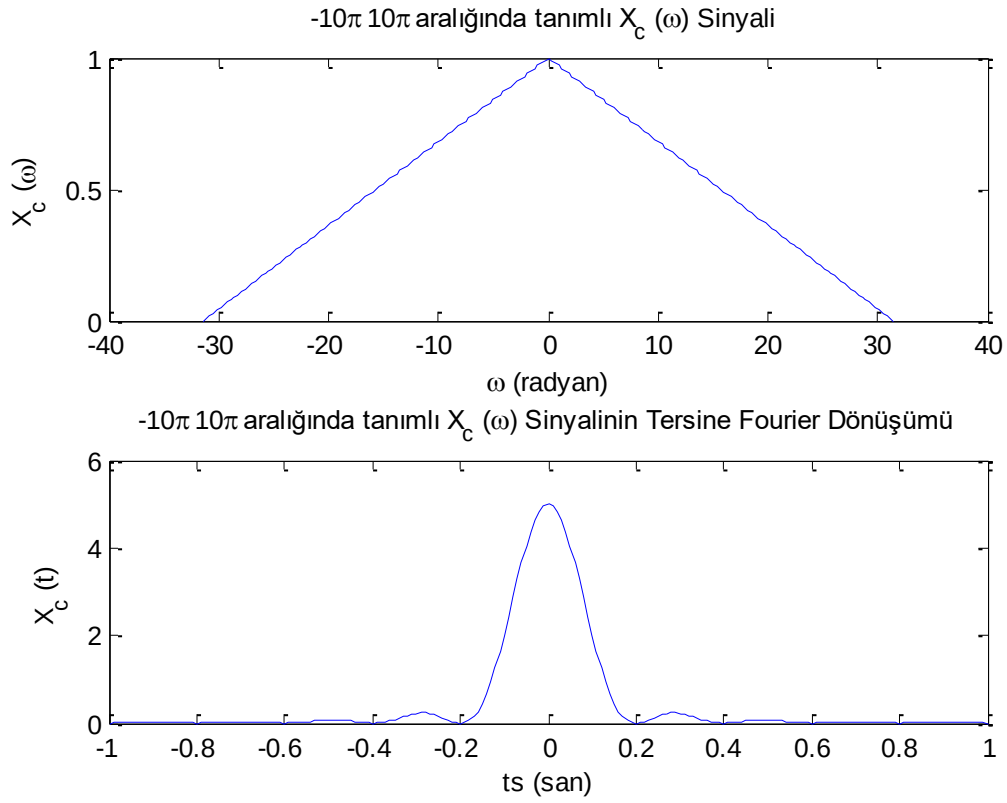
Bu formülü ve trapz komutunu kullanarak tersine Fourier dönüşümünü matlab kodumuza aşağıdaki gibi tanımlayalım.

```
t=linspace(-1,1,length(w));
xct=zeros(1,length(t));


for i=1:length(w)
    xct(i)=1/(2*pi).*trapz(w,(Xcw.*exp(j.*t(i).*w)));
end
```

Kodumuzun çizimlerini yapmak için subplot komutunu kullanıp, aşağıdaki gibi düzenleyelim.

```
subplot(2,1,1)
plot(w,Xcw)
xlabel('\omega (radyan)')
ylabel('X_c (\omega)')
title('-10\pi 10\pi aralığında tanımlı X_c (\omega) Sinyali ')

t=linspace(-1,1,length(w));
xct=zeros(1,length(t));

for i=1:length(w)
    xct(i)=1/(2*pi).*trapz(w,(Xcw.*exp(j.*t(i).*w)));
end

subplot(2,1,2)
plot(t,abs(xct))
xlabel('ts (san)')
ylabel('X_c (t)')
title('-10\pi 10\pi aralığında tanımlı X_c (\omega) Sinyalinin
Tersine Fourier Dönüşümü')
```

Kodumuzu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



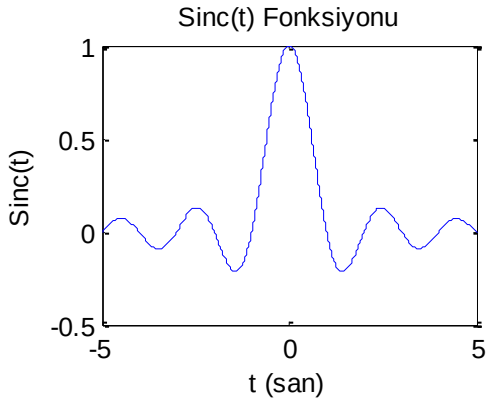
Figürün ilk çiziminde bir üçgen sinyal gözlemlenmekte ve tersine Fourier dönüşümünde ise sinc fonksiyonuna benzer ancak sıfırın altına düşmeyen bir fonksiyon gözlemlenmektedir. Bilindiği üzere üçgen sinyalleri elde etmenin bir yöntemi iki adet kare sinyalin konvolüsyon işlemine tabi tutulmasıdır, diğer bir bildiğimiz konuda kare sinyallerin tersine veya normal Fourier dönüşümü alındığında sinc fonksiyonunu vermesidir. Fourier veya zaman düzlemlerinden birinde yapılan konvolüsyon işlemi aynı sinyallerin diğer düzlemde çarpılmasına karşılık gelmektedir. Diğer bir deyişle zaman ekseninde bulunan iki sinyali konvolüsyon işlemine tabi tutup Fourier dönüşümünü hesaplamamız, bu sinyallerin ayrı ayrı Fourier dönüşümünü alıp bu dönüşümlerin birbiriyle çarpılması ile aynı sonucu vermektedir. Bu örneği şu şekilde düşüne biliriz, Fourier düzleminde bulunan bir kare sinyalin tersine Fourier dönüşümü sinc fonksiyonunu verecektir, bu kare sinyallerin konvolüsyonunu alırsak üçgen sinyal elde ederiz ve bu üçgen sinyalin tersine Fourier dönüşüm işlemi bize iki kare sinyalin ayrı ayrı tersine Fourier dönüşümlerinin hesaplanması ile elde edilen sinc fonksiyonlarının çarpımına eşit olacaktır, yani biz bu örnekte tersine Fourier dönüşüm işlemi ile bir sinc^2 fonksiyonu elde ettik. Sıradaki örnek bu konsepti anlamamıza daha yardımcı olacaktır.

Örnek: Matlab programını kullanarak bir adet *sinc(t)* fonksiyonu tanımlayınız, sonrasında bu fonksiyonun ilk olarak karesini sonra da Fourier dönüşümünü hesaplayınız.

Kodumuzu yazmaya önce zaman aralığını ve *sinc(t)* fonksiyonu ile beraber dörde böldüğümüz subplot komutu ile başlayalım. Aşağıda verilen kodu yazınız.

```
clc;clear all;close all;
t=-5:1/100:5;
xct=sinc(t);
Subplot(2,2,1)
plot(t,xct)
xlabel('t (san)')
ylabel('Sinc(t)')
Title('Sinc Fonksiyonu')
```

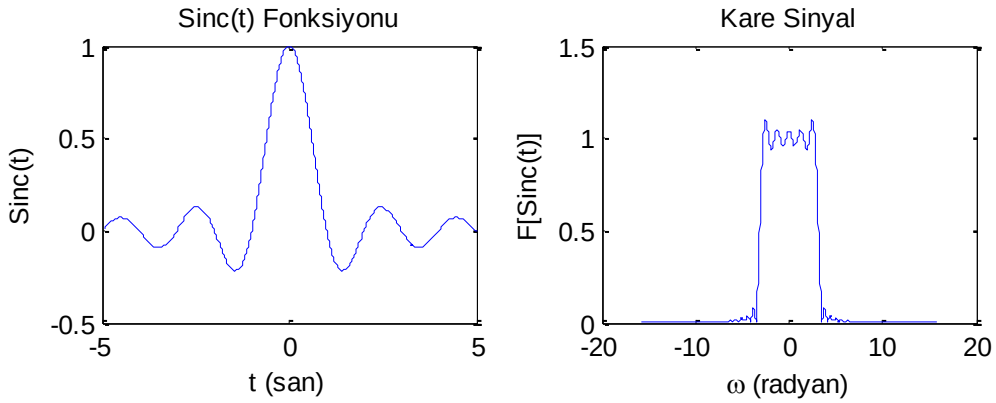
Yukarıda verilen kod çalıştırıldığında aşağıdaki sonucu elde edeceğiz.



Yazdığımız sinc fonksiyonunun Fourier dönüşümünü alarak devam edelim, kodu aşağıdaki gibi devam ettirelim. Bunun için ilk önce açısal frekans eksenini tanımlayıp, trapz komutunu kullanarak Foruier dönüşümünü hesaplayalım.

```
clc;clear all;close all;
t=-5:1/100:5;
xct=sinc(t);
Subplot(2,2,1)
plot(t,xct)
xlabel('t (san)')
ylabel('Sinc(t)')
Title('Sinc Fonksiyonu')
w=linspace(-5*pi,5*pi,length(t));

for i=1:length(w)
    xcw(i)=trapz(t,xct.*exp(-j*w(i).*t));
end
subplot(2,2,2)
plot(w,abs(xcw))
xlabel('\omega (radyan)')
ylabel('F[Sinc(t)]')
Title('Kare Sinyal')
```

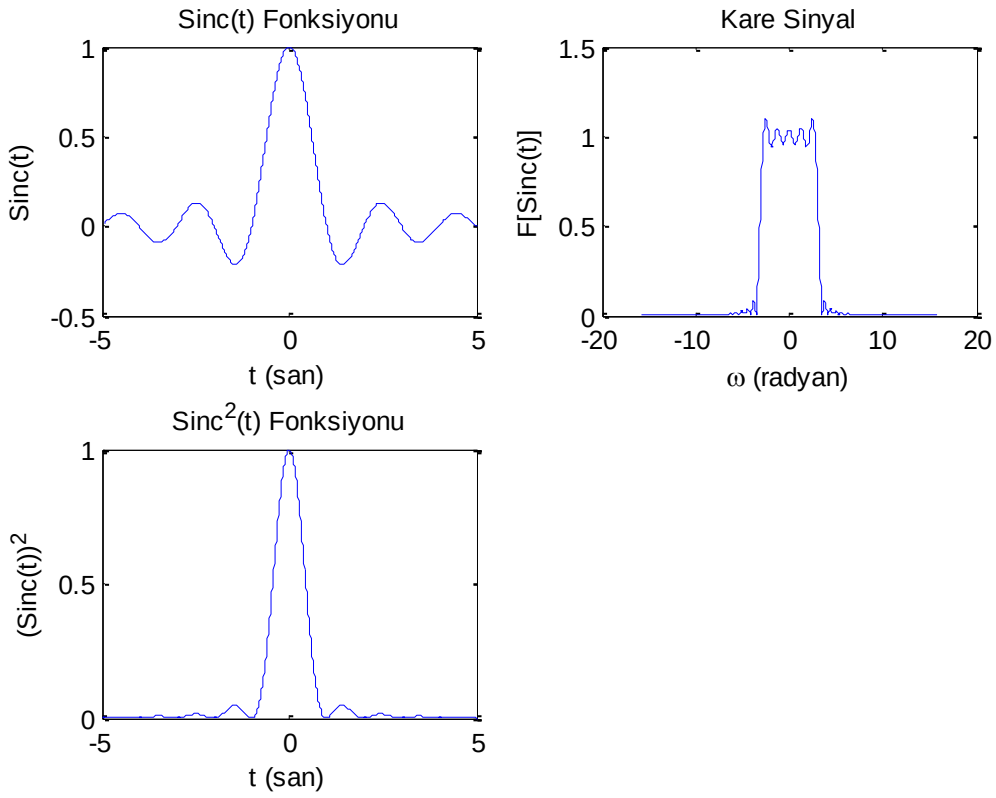
Yukarıdaki kodu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



Şimdi *sinc(t)* fonksiyonunun karesini alıp Fourier dönüşümünü hesaplayalım. Aşağıda verilen kodu daha önceden yazdığımız kodun altına ekleyip çalıştırınız.

```
xct2=xct.^2;
Subplot(2,2,3)
plot(t,xct)
xlabel('t (san)')
ylabel('(Sinc(t))^2')
Title('Sinc^2(t) Fonksiyonu')
```

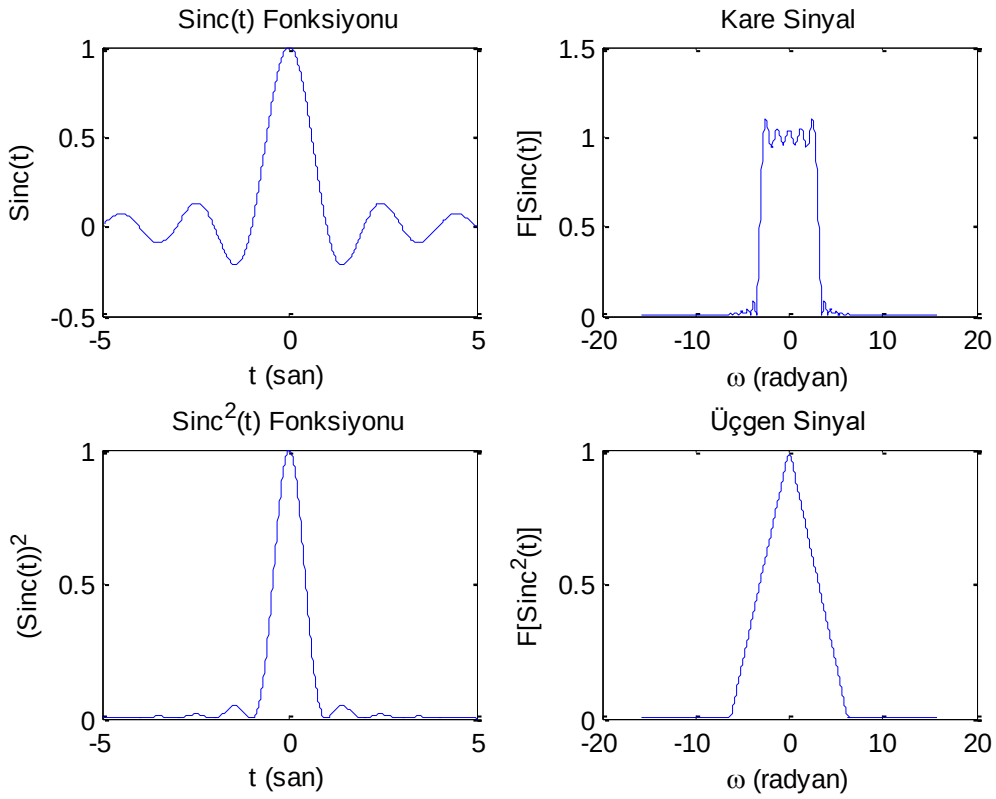
Yukarıdaki kodu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



Son olarak bulduğumuz *sinc²(t)* fonksiyonunun Fourier dönüşümünü hesaplayalım, bunun için yazdığımız koda aşağıdaki gibi devam edelim.

```
for i=1:length(w)
    xcw2(i)=trapz(t,xct2.*exp(-j*w(i).*t));
end
subplot(2,2,4)
plot(w,abs(xcw2))
xlabel('\omega (radyan)')
ylabel('F[Sinc^2(t)]')
Title('Üçgen Sinyal')
```

Yukarudaki kodu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



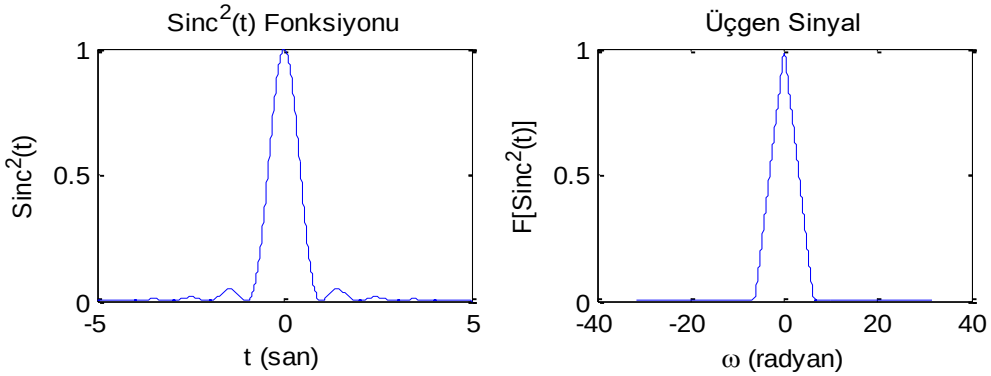
Yukarıdaki figürden de görüldüğü üzere nasılki üçgen sinyal kare sinyallerin konvolüsyonu ile elde edilebiliyor, kare sinyallerin tersine Fourier dönüşümü olan sinc fonksiyonları da bir birleri ile çarpılıp Fourier dönüşümleri alınınca aynı sonucu veriyor.

Örnek: Bir matlab kodu yazarak sürekli ve Ts=1/5 ile örneklenmiş $Sinc^2(t)$ fonksiyonlarının Fourier dönüşümlerini inceleyiniz.

Örneğimizin ilk adımında $Sinc^2(t)$ fonksiyonunu tanımlayıp Fourier dönüşümünü alıp subplot ile çizdirelim. Aşağıdaki kodu yazınız.

```
clc;clear all;close all;
t=-5:1/100:5;
xct=(sinc(t)).^2;
Subplot(2,2,1)
plot(t,xct)
xlabel('t (san)')
ylabel('Sinc^2(t)')
Title('Sinc^2(t) Fonksiyonu')
w=linspace(-10*pi,10*pi,length(t));
for i=1:length(t)
    xcw(i)=trapz(t,xct.*exp(-j*w(i).*t));
end
subplot(2,2,2)
plot(w,abs(xcw))
xlabel('\omega (radyan)')
ylabel('F[Sinc^2(t)]')
Title('Üçgen Sinyal')
```

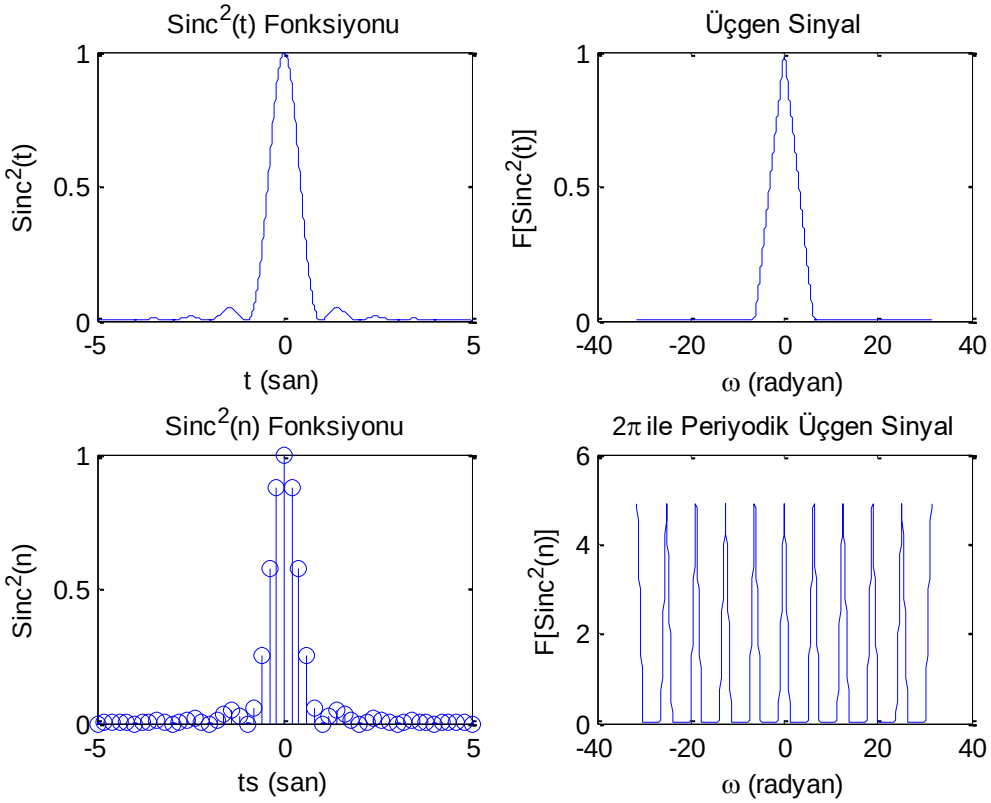
Yukarıda yazdığımız kodu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



İkinci adımda ise fonksiyonumuzu Ts=1/5 ile örnekleyip Fourier dönüşümünü alalım. Yazdığımız kodu aşağıdaki gibi devam ettirin.

```
Ts=1/5;
ts=-5:Ts:5;
n=ts./Ts;
xn=(sinc(ts)).^2;
Subplot(2,2,3)
stem(ts,xn)
xlabel('ts (san)')
ylabel('Sinc^2(n)')
Title('Sinc^2(n) Fonksiyonu')
for i=1:length(w)
    xsw(i)=sum(xn.*exp(-j.*w(i).*n));
end
subplot(2,2,4)
plot(w,abs(xsw))
xlabel('\omega (radyan)')
ylabel('F[Sinc^2(n)]')
Title('2\pi ile Periyodik Üçgen Sinyal')
```

Yukarıdaki kodu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



Yukarıdaki çizimden de anlaşılacağı gibi sürekli bir sinyal örneklenince Fourier dönüşümünde genliği Ts değerine bölünür, yatay ekseni Ts ile çarpılır, ve 2π ile periyodik hale gelir.

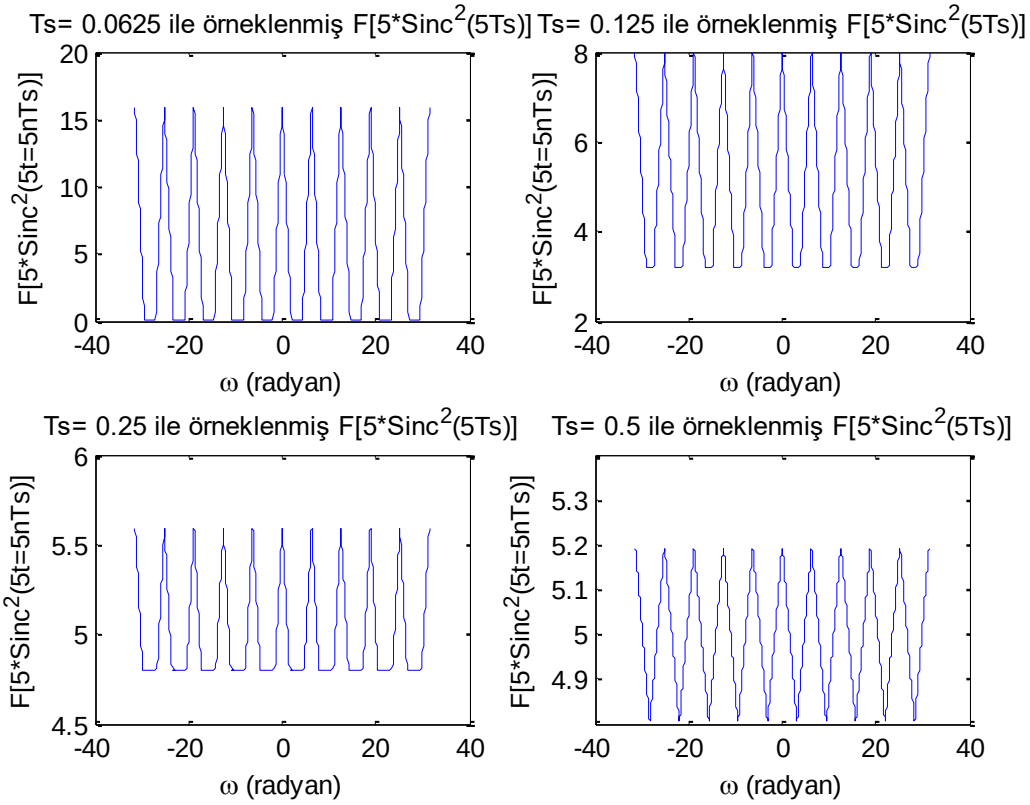
Örnek: *5*Sinc²(5t)* sinyalini Ts=1/16, Ts=1/8, Ts=1/4, Ts=1/2, örnekleme periyodları ile örnekleyip Fourier dönüşümlerini inceleyiniz ve hangi örnekleme periyodlarına spektral örtüşme olduğuna bakınız

Bu örnekte sadece sinyalimizin sadece Fourier dönüşümleri kıyaslayacağız. Kodu aşağıdaki gibi yazıp inceleyin.

```
clc;clear all; close all;

t=-5:1/100:5;

Tsarr=[1/16 1/8 1/4 1/2];

w=linspace(-10*pi,10*pi,length(t));
```

```
for Ts=1:length(Tsarr)
xsw=zeros(1,length(w));
ts=-5:Tsarr(Ts):5;
n=ts./Tsarr(Ts);
xn=zeros(1,length(n));
xn=5*((sinc(5*ts)).^2);
for i=1:length(w)
    xsw(i)=sum(xn.*exp(-j.*w(i).*n));
end
subplot(2,2,Ts)
plot(w,abs(xsw))
xlabel('\omega (radyan)')
ylabel('F[5*Sinc^2(5t=5nTs)]')
Title(['Ts= ' num2str(Tsarr(Ts)) ' ile örneklenmiş
F[5*Sinc^2(5Ts)]'])
end
```

Yukarıdaki kodu çalıştırdığımızda aşağıdaki sonucu elde edeceğiz.



Yukarıdaki çizimden anlaşılacağı gibi örnekleme periyodu 1/8 ve daha büyük periyodlar için dönüşümde spektral örtüşme (Aliasing) durumu ortaya çıkmıştır.

Örnek Azaltma İşlemi

Örneklenmiş sinyallerin örnek sayısını azalatmak için yapılan bir işlemdir. Örneklenmiş sinyalimizi *x[n]* ile ifade ediyoruz *x[n]* sinyalinin elemanlarından her M kat artışında eleman alarak oluşturduğumuz sinyale de *y[n]* diyelim, bu iki sinyal arasındaki ilişkiyi *y[n]=x[Mn]* şeklinde ifade edebiliriz.

Örnek: Örneklenmiş bir sinyal olan x[n] dizisinin elemanları (1,2,a,b,3,4,c,d,5,6,e,f,7,8) gibi olsun x[Mn] dizisini M1=2, ve M2=3 için oluşturup gösteriniz.

x[n] dizisinin ilk elemanı olan '1' değerinin indeksi 1 olsun bu budurumda son eleman olan '8' değerinin indeksi 14 olacaktır. Bu indeks değerleri Matlab programı ile örtüşsün diye seçilmiştir. M1=2 için indeks değerleri 1,3,5,7,9,11,13 olan elemanlar yani y1[n]=(1,a,3,c,5,e,7) dizisinin oluşması lazım, M2=3 için ise indeks değerleri 1,4,7,10,13 olan yani y2=(1,b,c,6,7) dizisinin oluşması gerekmektedir.

Kodu yazmaya a,b,c,d,e,f alfa numerik değerlerini sembol olarak tanıtım x[n] vektörünü tanıtarak başlayalım.

```
clc; clear all;close all;
syms a b c d e f;
xn=[1 2 a b 3 4 c d 5 6 e f 7 8 ];
```

Yukardaki tanımlamalımızı yaptıktan sonra M1 ve M2 değerlerini 2 ve 3 olarak ve yeni oluşturacağımız y1 ve y2 dizilerinin indeksleri için i ve j değerlerini 1 olarak tanımlayalım.

```
clc; clear all;close all;
syms a b c d e f;
x=[1 2 a b 3 4 c d 5 6 e f 7 8 ];
M1=2;
M2=3;
i=1;
j=1;
```

Artık y1 ve y2 dizilerini x dizisinin indeksinin M1 ve M2 katları artışından eleman seçerek oluşturalım, bu işlem için for döngüleri kullancağız. Bu döngülerin sonunda y1 ve y2 değerlerini komut penceresinden gözlemleyelim.

```
clc; clear all;close all;
syms a b c d e f;
x=[1 2 a b 3 4 c d 5 6 e f 7 8 ];
M1=2; M2=3; i=1; j=1;
for indeks=1:M1:length(x)
    y1(i)=x(indeks);
    i=i+1;
end
for indeks=1:M2:length(x)
    y2(j)=x(indeks);
    j=j+1;
end
y1
y2
```