

## Project #14

### Properties of Continuous Time Fourier Series, Multiplication

In this project we will demonstrate the validity of “multiplication property” of Fourier series (FS). Let two signals  $x(t)$  and  $y(t)$  be both periodic with period  $T$ , then the product signal  $x(t)y(t)$  will also be periodic with the same period. If FS representations for the two periodic signal are known, then FS representation for the product signal can be obtained by using this property which we can express as follows.

$$x(t) \xrightarrow{FS} a_k, y(t) \xrightarrow{FS} b_k \Rightarrow x(t)y(t) \xrightarrow{FS} c_k = \sum_{l=-\infty}^{\infty} a_l b_{k-l} \quad (14.1)$$

As we see, the sequence of FS coefficients of the product signal  $x(t)y(t)$ ,  $c_k$ 's, is the discrete time convolution of two sequences  $a_k$ 's and  $b_k$ 's, FS coefficients of  $x(t)$  and  $y(t)$  respectively.

This time, we will work with periodic square wave and cosine signals shown in Figure 14.1 (upper and middle panels), to construct our example. For the square wave signal with  $T=1$  and  $T_1=0.25$ , we already know FS coefficients  $a_k$ 's, expressed as a discrete time sequence with index  $k$ , are

$$a[0] = \frac{2T_1}{T}, \quad a[k] = \frac{\sin(k\pi/2)}{k\pi} \text{ for } k \neq 0. \quad (14.2)$$

For the 1Hz cosine signal,  $\cos(2\pi t)$ , FS coefficients  $b_k$ 's are

$$b_k = \begin{cases} 0.5, & k = \pm 1 \\ 0, & \text{otherwise} \end{cases}. \quad (14.3)$$

We can also express  $b_k$ 's as a discrete time sequence with index  $k$  as

$$b[k] = 0.5\delta[k+1] + 0.5\delta[k-1]. \quad (14.4)$$

To carry out the convolution of  $a_k$  and  $b_k$  sequences, we have used three different approaches.

In the first one, we have directly implemented the sum formula for  $c_k$  given in statement (14.1), using a “for loop”.

In the second technique, we used the following property of impulse function

$$x[k] * \delta[k \pm k_0] = x[k \pm k_0], \quad (14.5)$$

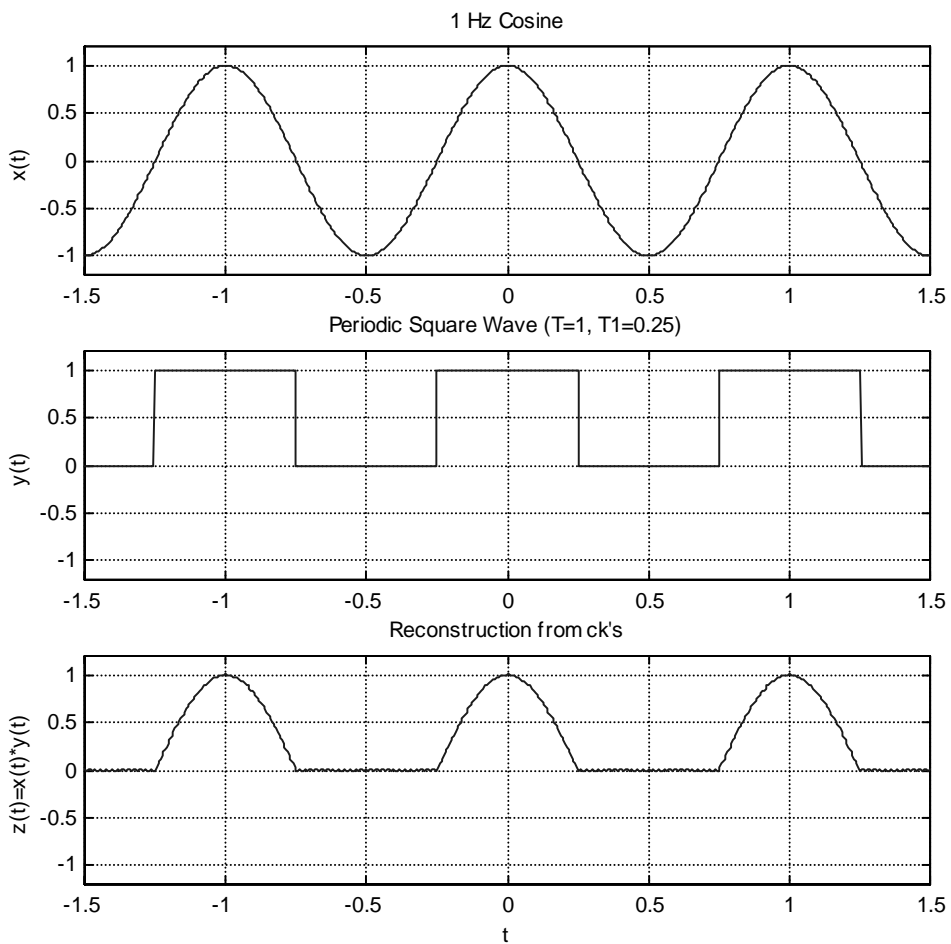
where “\*” denotes the convolution operation. Using equations (14.4) and (14.5), we expressed the convolution of the sequences  $a[k]$  and  $b[k]$  as follows.

$$\begin{aligned} a[k] * b[k] &= a[k] * (0.5\delta[k+1] + 0.5\delta[k-1]) \\ &= 0.5a[k] * \delta[k+1] + 0.5a[k] * \delta[k-1] \\ &= 0.5a[k+1] + 0.5a[k-1] \end{aligned} \quad (14.6)$$

This last formula indicates that the convolution  $a[k]*b[k]$  can be obtained by adding two shifted and scaled versions of  $a[k]$ . We basically implemented this idea to carry out the convolution.

In the third technique, we simply used Matlab’s *conv* function to convolve  $a[k]$  and  $b[k]$  sequences. For more information about this function, type “help conv” at the Matlab prompt.

You can find Matlab code of this project below, in mfile “project14.m”.



**Figure 14.1** Square wave signal with  $T = 1$ ,  $T_1 = 0.25$  (upper panel), 1Hz cosine (middle panel), and reconstructed signal from convolution of FS coefficients of these two sequences (lower panel).

```
% Project #14
% Title: Properties of Continuous Time Fourier Series
%       Multiplication

% Generation of 1Hz cosine
t=-1.5:0.005:1.5;
xt=cos(2*pi*t);
```

```
% Generation of periodic square wave
yt=xt>0;

% FS coefficients of periodic square wave
k=-50:50;
T=1;T1=0.25;
ak=sin(k*2*pi*(T1/T))./(k*pi);
% Manual correction for a0 -> ak(51)
ak(51)=2*T1/T;

% 1st way to create ck's (For loop implementation
% of convolution)
% FS coefficients of 1Hz cosine (over k=-100..100)
bk=zeros(1,201);
bk(100)=0.5;bk(102)=0.5;
ck1=zeros(1,101);
for k=-50:50
    u=0;
    for L=-50:50
        u=u + ak(L+51)*bk(k-L+101);
    end
    ck1(k+51)=u;
end

% 2nd way to create ck's (A simpler way of implementing
% convolution using properties of impulse)
% FS coefficients of 1Hz cosine (over k=-50..50)
bk=zeros(1,101);
bk(50)=0.5;bk(52)=0.5;
ck2=0.5*[ak(2:101) 0]+0.5*[0 ak(1:100)];

% 3rd way to create ck's (Implementation of convolution via
% Matlab's "conv" function that does convolution)
```

```

% FS coefficients of 1Hz cosine (over k=-1..1)
bk=zeros(1,3);
bk(1)=0.5;bk(3)=0.5;
ck3=conv(ak,bk);
ck3(103)=[];ck3(1)=[];

% Check the equality of ck1, ck2, and ck3
ck1-ck2
ck1-ck3
% Have seen that both differences produced vectors
% filled with zeros?

% Reconstruction from ck1's with 101 terms (M=50)
w0=2*pi/T;zt=zeros(1,length(t));
for k=-50:50
    zt=zt + ck1(k+51)*exp(j*k*w0*t);
end

figure(1);set(gcf,'defaultaxesfontsize',8)
subplot(3,1,1);plot(t,xt);ylabel('x(t)')
title('1 Hz Cosine')
set(gca,'ylim',[-1.2 1.2]);grid

subplot(3,1,2);plot(t,yt);ylabel('y(t)')
title('Periodic Square Wave (T=1, Tl=0.25)')
set(gca,'ylim',[-1.2 1.2]);grid

subplot(3,1,3);plot(t,real(zt))
xlabel('t');ylabel('z(t)=x(t)*y(t)')
title('Reconstruction from ck''s')
set(gca,'ylim',[-1.2 1.2]);grid

```