$\boxed{7.30}$ Write a function called *u_synbank* that synthesizes a composite signal $x(i)$ from N low-bandwidth subsignals $x_i(k)$ using a uniform DFT synthesis filter bank. The calling sequence for *u_synbank* is as follows.

```
x = u_synbank (X,m,alpha,win,fs);
```

On entry to *u_synbank*, X is a p by N matrix with the ith subsignal in column i, m is the anti-imaging filter order, *alpha* is a factor between 0 and 1 which controls the filter cutoff frequency, *win* is the window type as in Table 6.2.2, and fs is the sampling frequency. Use a lowpass cutoff frequency of $F_0 = alpha(fs)/(2N)$. On exit from *u_synbank*, x is the complex composite signal of bandwidth $Nf_s/2$ containing the frequency translated subsignals in its subbands.

Test function *u_synbank* by writing a script that uses the FDSP toolbox function *f_subsignals* to construct a 32 by 4 matrix X with the samples of the kth subsignal in column k. The function *f_subsignals* produces signals whose spectra are given in (7.5.6). Use $alpha = 0.5$, $fs = 200$ Hz, and a windowed filter of order $m = 90$ with a Hamming window. Save x and fs in a MAT-file named *prob7_30* and plot the following

(a) The real and imagninary parts of the complex composite signal $x(i)$. Use *subplot* to construct a 2×1 array of plots on one screen.

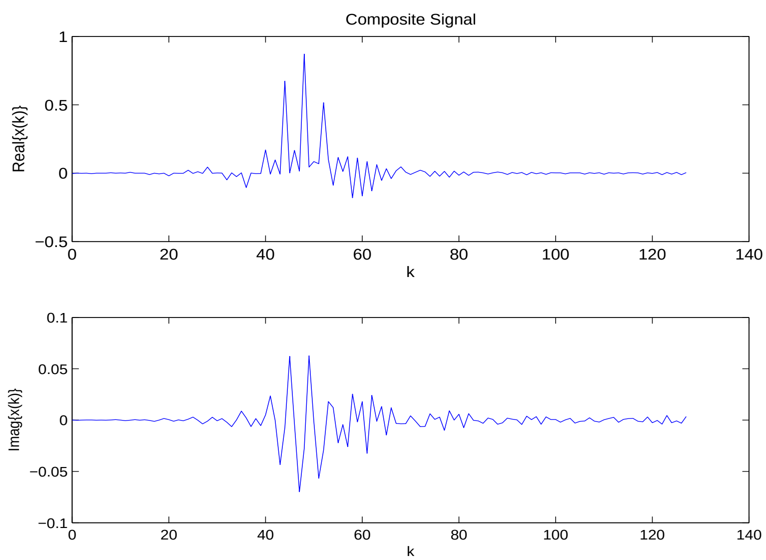
(b) The magnitude spectrum $A(f) = |X(f)|$ for $0 \le f \le fs$.

Solution

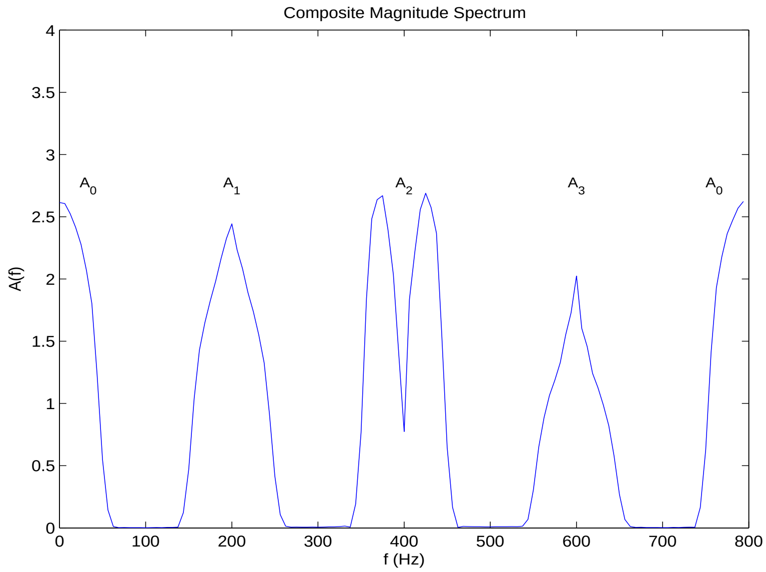
```
% Problem 7.30

% Initialize

function p730
clear
clc
fs = 200;
N = 4;
p = 32;
m = 90;
win = 2;
alpha = 0.5;

% Compute and plot complex composite signal

X = f_subsignals (p);
x = u_synbank (X,m,alpha,win,fs);
q = length(x);
k = 0 : q-1;
figure
subplot(2,1,1)
plot(k,real(x))
```

```
set (gca,'Fontsize',11);
f_labels('Composite Signal','k','Real\{x(k)\}')
subplot(2,1,2)
plot (k,imag(x))
f_labels('','k','Imag\{x(k)\}')
f_wait
save prob7_30 x fs

% Compute and plot composite magnitude spectrum

H_x = fft(x,q);
A_x = abs(H_x);
f_x = linspace(0,(q-1)*N*fs/q,q);
figure
plot(f_x,A_x)
set (gca,'Fontsize',11);
f_labels ('Composite Magnitude Spectrum','f (Hz)','A(f)')
axis([0 N*fs 0 4])
a = 2.75;
text (fs/6,a,'A_0','HorizontalAlignment','Center')
text (fs,a,'A_1','HorizontalAlignment','Center')
text (2*fs,a,'A_2','HorizontalAlignment','Center')
text (3*fs,a,'A_3','HorizontalAlignment','Center')
text (N*fs-fs/5,a,'A_0','HorizontalAlignment','Center')
f_wait

function x = u_synbank (X,m,alpha,win,fs)
% U_SYNBANK: Synthesize a complex composite signal from subsignals using a DFT filter bank
%
% Inputs: X     = p by N matrix containing subsignal i in column i
%         m     = order of anti-imaging filter
%         alpha = relative cutoff frequency: F_0 = alpha*fs/(2N)
%         win   = an integer specifying the desired window type
%
%                 0 = rectangular
%                 1 = Hanning
%                 2 = Hamming
%                 3 = Blackman
%
%         fs    = sampling frequency
%
% Outputs: x = complex vector of length q = Np containing samples of composite
%              signal. x contains N frequency-multiplexed subsignals.  The
%              bandwidth of x is N*fs/2 and the ith subsignal is in band i

% Change sampling rate

[p,N] = size(X);
L = N;
```

```
q = L*p;
y = zeros(q,N);
type = 2;
for i = 1 : N
    Y(:,i) = f_tocol(f_interpol(X(:,i),fs,L,m,type,alpha));
end

% Construct composite signal

k = [0 : q-1]';
W_N = exp(-j*2*pi/N);
x = zeros(q,1);
for i = 1 : N
   x = x + W_N.^(-(i-1)*k) .* Y(:,i);
end
```



(a) Components of the Complex Composite Signal



(b) Magnitude Spectrum of the Complex Composite Signal