```
%bfdrv2d.m : Driver for nonperiodic/periodic sampling
% on the group G = Z_L X Z_L = {0, ..., L-1} X {0, ..., L-1}
% with addition modulo L.
% Explanation of variables:
% L = number elements in Z_L
% The subgroups are generated by the column vector
% [h1(k):h2(k)] with h1 and h2 as described below
% h1: vector where h1(k) is the divisor of L which generates
% the points in the horizontal direction with the form <h1(k)>, i.e.,
% <h1(k)> = {0,h1(k),2h1(k),...,L-h1(k)}
% h2: vector where h2(k) is the divisor of L which generates the
% points in the vertical direction with the form <h2(k)>, i.e.,
% <h2(k)> = {0,h2(k),2h2(k),...,L-h2(k)}
% x1: vector with shifts in the horizontal direction. M_k = x1(k) + <h1(k)>
% x2: vector with shifts in the vertical direction. M_k = x2(k) + <h2(k)>
% eta1: eta1(k) corresponds to eta_{k+1} in the theorem in the
% horizontal direction.
% eta2: eta2(k) corresponds to eta_{k+1} in the theorem in the
% vertical direction.
% filt: characteristic function of spectrum

%Input parameters:

L=512;                    % Length of Z_L
h1=[8 4 4];         % Specify horizontal direction of the subgroups
h2=[8 8 4];             % Specify vertical direction of the subgroups
x1=[1 1 0];             % Specify horizontal shifts; need x1 in {0,...,h1-1}
x2=[1 0 1];             % Specify vertical shifts; need x2 in {0,...,h2-1}
eta1=[0 0 256]/L;      % Specify eta_{k+1} in the horizontal direction
eta2=[0 64 128]/L;      % Specify eta_{k+1} in the vertical direction
% End of input section

% Randomly generate signal to be sampled and reconstructed
fhat=complex(rand(L,L),rand(L,L));   % Generate random
filt = spectrum(L,h1,h2,N,eta1,eta2);% Spectrum
fhat=fhat.*filt;                     % Filtered signal, i.e., set frequencies
                                     % outside of spectrum to zero
fexact=ifft2(fhat);
fexact=fexact/norm(fexact);          % Normalize signal
N = max(size(h1));                   % Number of subgroups
% Compute sampled values
f = zeros(L,L);
for k=1:N
 Hx1=x1(k)+[0:h1(k):L-h1(k)]; % Coset M1_k = x1(k) + <h1(k)> in the horizontal direction
 Hx2=x2(k)+[0:h2(k):L-h2(k)]; % Coset M2_k = x2(k) + <h2(k)> in the vertical direction
```

```
 f(1+Hx1,1+Hx2) = fexact(1+Hx1,1+Hx2);% Sampled values on [M1_k;M2_k]
end
% Reconstruct signal
F = bfmethod(f,L,h1,h2,eta1,eta2,x1,x2);

%Compute the l2 relative reconstruction error
relerr = norm(fexact - F)
%-------------------------------------------------------------------------
function filt = spect(L,h1,h2,N,eta1,eta2)
% Computes the spectrum according to definition 2 of the paper
% Eta needs to be chosen appropriately to satisfy the definition
M = max(size(eta1));
ETA1=0;
ETA2=0;
filt=zeros(L,L);
for m=N:-1:1
    v=(L*ETA1)+(1:L/h1(m));
    w=(L*ETA2)+(1:L/h2(m));
   filt(v,w)=1;
    for k = (L*ETA2)+(1:L/h2(m))    % Create the plot of the spectrum
        hold on
        grid on
        subplot(1,2,2)
        plot(v,k,'*')
    end
    ETA1 = ETA1 + eta1(m);
    ETA2 = ETA2 + eta2(m);
end
%-----------------------------------------------------------------
function F=bfmethod(f,L,h1,h2,eta1,eta2,x1,x2)

N = max(size(h1));
Hx1 = x1(N)+[0:h1(N):L-h1(N)];              % Coset M1_N = x1(N) + <h1(N)>
Hx2 = x2(N)+[0:h2(N):L-h2(N)];              % Coset M2_N = x2(N) + <h2(N)>
fH = f(1+Hx1,1+Hx2);                        % Sampled values on [M1_N;M2_N]
SMf = SM(fH,L,h1(N),h2(N),x1(N),x2(N));
V=([0:L-1]-x1(N));
W=([0:L-1]-x2(N));
TMP1 = zeros(L,L);
TMP = zeros(L,L);
if N==1
  F = SMf;
else
    for k = 1:L
        for m = 1:L
```

```
            tmp = 1-exp(2*pi*i*[V(k)*eta1(N)+W(m)*eta2(N)]);
            tmp1 = tmp;
            tmp1(find(abs(tmp1 < 1.e-14)))=1;   % Avoid zero divisions
            TMP(k,m) = tmp;
            TMP1(k,m) = tmp1;
        end
    end
f1 = (f - SMf)./TMP1;
fN1 = bfmethod(f1,L,h1(1:N-1),h2(1:N-1),eta1(1:N-1),eta2(1:N-1),x1(1:N-1),x2(1:N-1));
F = fN1.*TMP + SMf;
end
%--------------------------------------------------------------------
function S = SM(f,L,h1,h2,x1,x2)
%Computes S_Mf(z) for z in G
%G = {0,1,...,L-1} X {0,1,...,L-1} with addition mod L
%H = {0,h1,2h1,...,L-h1} X {0,h2,2h2,...,L-h2}
%f = row vector of length L/h1 X L/h2, with sampled
% values on x+H where x=(x1,x2).
% x = shift. Need x1 in {0,...,h1-1} and x2 in {0,...,h2-1}

chi = zeros(L,L);
chi(1:L/h1,1:L/h2) = fft2(f);
S = h1*h2*ifft2(chi);
if x1 > 0
  tmp = S(L-x1+1:L,1:L);
  S(x1+1:L,1:L)=S(1:L-x1,1:L);
  S(1:x1,1:L) = tmp;
end
if x2 > 0
    tmp = S(:,L-x2+1:L);
    S(:,x2+1:L) = S(:,1:L-x2);
    S(:,1:x2) = tmp;
end
%---------------------------------------
```