```
% Daoud Burghal Dec 2014 "E{dt/di} Integrals"
% This function calculates the E{dt/di} for given location error (e_0),
% % % INPUT:
% SN : (vector) Segmentation parameters
% GN : (vector) Guarding Parameters
% Gamma : (1x2) propagation exponents before and after the break distance
% dbreak : the break distance , prop. exponent change before and after.
% NB_Side : number of blocks at side of the cell.
% saveMat : save the results on a file.
% HD : (1x2) Orientation of the interfering segment
%             HD(1): horozontal , HD(2) : Diagonal
% e: location error in meter
%
% % % OUTPUT:
% di_dtMat: the average value of E{dt/di}, for horiznotal and diagonal

function di_dtMat = Dt_DiGammaMatHDLE(SN, GN, Gamma, dbreak,
NB_Side,saveMat,HD,e)

% % % Default values
if isempty(SN); SN = 0 : 1:5; end
if isempty(GN); GN = 0:0.5:1; end

if length(Gamma)>1
    gamma1 = Gamma(1);
    gamma2 = Gamma(2);
else
    gamma1 = Gamma(1);
    gamma2 = Gamma(1);
end
Lc = 25;    % raduius of the training
d0 = 0.5;   % Near filed comm. - to avoid singularities in path loss
check = 0;  % plot the areas, for verifying purposes
% % %

di_dtDFar = zeros(length(GN),length(SN),NB_Side-1);
di_dtHFar = zeros(length(GN),length(SN),NB_Side-1);

%disp('START with ------------*--Far_Interf-*--------------------------------
--')
R = find(HD>0);
for r = R
    for n_lp = 0 :NB_Side-2

        for g_lp = 1: length(GN)
            for s_lp =1 : length(SN)

                Ls = 2*Lc/(2*SN(s_lp)+1);
                Lg = GN(g_lp)*Ls;

                Nr = 2000; % number of points in the traing disk
                Nt = min(round(1.1*Ls)^2,Nr); % number of points in the tx
segment
                Ni = min(round(1.1*Ls)^2,Nr); % number of points in the
interf segment
```

```
                % modify the intensity of the points according to the
                % distribution of devices
                tx_loc = DevLoc(Nt,Ls,e,0,0); % for tx segm.
                Int_loc_o = DevLoc(Ni,Ls,e,0,0); % for interf segm.

                % distribution in the rx disk
                ur_o = 2*Lc *(rand(1,Nr)-.5) + 1i *2*Lc *(rand(1,Nr)-.5);
                ur_o = ur_o(abs(ur_o)<Lc);

                % Shifting the center of points distribution
                Nr = length(ur_o);
                Shift = Lc + 2*Lg + Lc+ n_lp*(2*Lg+2*Lc) ;
                if r == 2       % i.e., HD(2) = 1
                    Shift2 = Shift +1i* Shift;
                elseif r == 1   % i.e., HD(1) = 1
                    Shift2 = Shift;
                end
                Int_loc = Int_loc_o + Shift2 ;

                UR_ACT =  tx_loc(:,ones(1,Nr))+ ur_o(ones(Nt,1),:);
                if check > 0
                    plot(real(Int_loc),imag(Int_loc),'+')
                    hold on
                    plot(real(ur_o),imag(ur_o),'g')
                    plot(real(UR_ACT(:)),imag(UR_ACT(:)),'m.')
                    plot(real(tx_loc(:)),imag(tx_loc(:)),'r+')
                    plot(real(ur_o),imag(ur_o),'g')
                end
                Di  = abs(UR_ACT - Int_loc(:,ones(1,Nr)));
                Di = Di(:);

                Dt = abs(ur_o(ones(Nt,1),:));
                Dt = Dt(:) ;
                % replace the distances less that d0 with d0
                Dt(Dt< d0) = d0;
                Di(Di< d0) = d0;

                % Using the appropriate propagation exponent
                Dt_p = (Dt/dbreak).^gamma1 .* (Dt<dbreak) + ...
                    (Dt/dbreak).^gamma2 .* (Dt >= dbreak);
                Di_p = (Di/dbreak).^gamma1 .*(Di<dbreak) + ...
                    (Di/dbreak).^gamma2 .* (Di >= dbreak);
                 if r == 2       % i.e., HD(2) = 1
                    di_dtDFar(g_lp,s_lp,n_lp+1) = sum((Dt_p./Di_p
))/(length(Dt));
                 elseif r == 1       % i.e., HD(1) = 1
                    di_dtHFar(g_lp,s_lp,n_lp+1) = sum((Dt_p./Di_p
))/(length(Dt));
                 end

                clear Di UR_ACT Int_loc Dt

            end
        end
```

```
        di_dtMat = (di_dtDFar + di_dtHFar)/sum(HD);
    end
end
if saveMat
    save di_dtMat
end
```