```
%--------------------Monte Carlo Simulation for SOP----------------------%

clc
clear

%-------------Network Settings-----------------%
K=3;                    % Hops
ds=5;                   % Distance of One Hop
dse(1)=sqrt(5^2+10^2);  % Distance between Eav and S1 (S)
dse(2)=10;              % Distance between Eav and S2
dse(3)=dse(1);          % Distance between Eav and S3
M=2;                    % Number of Eavs (Jammers)
PJ_TH=1:.1:10;          % Jamming Power for Theoretical Formulation (PJ1=PJ2)
PJ_MC=1:1:10;           % Jamming Power for Monte Carlo Simulation (PJ1=PJ2)
alpha=4;                % Path Loss Exponent
gammaE=1;               % SNR Threshold for Eav Wiretapping Successfully

dje=2.5;                  % Distance between an Eav and Its Corresponding Jammer
dJE=[2.5,22.5;22.5,2.5];  % Distance between Ei and Jj
PS=10;                     % Transmitteing Power (PS1=PS2=PS3)

round=10000000;            % Monte Carlo Trials

%---------------------------Theoretical Formulation---------------------%
for l=1:1:length(PJ_TH)
    sum=0;
    for k=1:1:K
        sum=sum+dse(k)^(-alpha);
    end
    Pso_TH(l)=1-exp(-PS*dje^alpha*M/gammaE/PJ_TH(l)*sum);  % Formula (2)
end

%---------------------------Monte Carlo Simulation----------------------%
for l=1:1:length(PJ_MC)
    num=0;
    for r=1:1:round
        for k=1:1:K
            hS(k)=exprnd(1);                % Fading Coefficient of Link lk

            for i=1:1:M                   % Fading Coefficient, h_{J_j,E_i}
                for j=1:1:M
                    hJ(i,j)=exprnd(1);
                end
            end

            for m=1:1:M                                       % SNR at Em
                sum=0;
                for j=1:1:M
                    sum=sum+PJ_MC(l)*hJ(m,j)/dJE(m,j)^alpha;
                end
                SNR(m)=PS*hS(k)/dse(k)^alpha/sum;
            end
```

```
            if(max(SNR)>gammaE)
                num=num+1;
                break;
            end
        end
    end
    Pso_MC(1)=num/round;
end

plot(PJ_TH,Pso_TH,'k-')
hold on
box on
plot(PJ_MC,Pso_MC,'kv')
xlabel({'Jamming power, $$P_{J_i}$$'},'interpreter','latex')
ylabel({'Secrecy outage probability, $$P_{so}$$'},'interpreter','latex')
legend({'$P_{S_k}=10$, $d_{J_i,E_i}=2.5$, Th','$P_{S_k}=10$, $d_{J_i,E_i}=2.5$, MC',
},'interpreter','latex')
```

```
%-------------------Monte Carlo Simulation for COP------------------------%
clc
clear

%-------------------------Network Settings-------------------------------%
K=3;                              % Hops
ds=5;                             % Distance of One Hop
M=2;                              % Number of Jammers
alpha=4;                          % Path Loss Exponent
gammaC=1;                         % SNR Threshold for Receiver Decoding Successfully
PJ_TH=1:.1:10;                    % Jamming Power for Theoretical Formulation (PJ1=PJ2)
PJ_MC=1:1:10;                     % Jamming Power for Monte Carlo Simulation (PJ1=PJ2)

dje=2.5;                          % Distance between an Eav and Its Corresponding Jammer
djd(1)=10+dje;                    % Distance between Jammer and D1
djd(2)=sqrt(ds^2+(10+dje)^2);     % Distance between Jammer and D2
djd(3)=sqrt((2*ds)^2+(10+dje)^2); % Distance between Jammer and D3 (D)

PS=10;                            % Transmitteing Power (PS1=PS2=PS3)



round=10000000;                      % Monte Carlo Trials


%-------------------------Theoretical Formulation------------------------%
for l=1:1:length(PJ_TH)
    temp=0;
    for k=1:1:K
        temp=temp+djd(k)^(-alpha);
    end
    Pco_TH(l)=1-exp(-gammaC*ds^alpha*M*PJ_TH(l)/PS*temp);
end

%----------------------Monte Carlo Simulation---------------------------%
for l=1:1:length(PJ_MC)
    num=0;
    for r=1:1:round
        for k=1:1:K
            hs(k)=exprnd(1);                      % Fading Coefficient of Link lk

            for m=1:1:M
                hj(m)=exprnd(1);                  % Fading Coefficient, h_{J_m,D_k}
                in(m)=PJ_MC(l)*hj(m)/djd(k)^alpha;  % Artificial Noise of J_m at D_k
            end

            SNR=PS*hs(k)/ds^alpha/sum(in);       % SNR at D_k

            if(SNR<gammaC)
                num=num+1;
                break;
            end
```

```
        end
    end
    Pco_MC(l)=num/round;
end

plot(PJ_TH,Pco_TH,'r--')
hold on
plot(PJ_MC,Pco_MC,'ro')
xlabel({'Jamming power, $$P_{J_i}$$'},'interpreter','latex')
ylabel({'Connection outage probability, $$P_{co}$$'},'interpreter','latex')
legend({'$P_{S_k}=10$, $d_{J_i,E_i}=2.5$, Th','$P_{S_k}=10$, $d_{J_i,E_i}=2.5$,
MC'},'interpreter','latex')
```

```
%--------Nash Equilibrium vs Numer of Jammers ---------------------------%

clc
clear

%------------------------Network Settings--------------------------------%
c=1;                                              % Unit Cost
R=10;                                             % Rewards
alpha=4;                                          % Path Loss Exponent
M=[2,3,4,5,6];                                    % Number of Jammers
dje(1)=2;                                         % Distance between J1 and E1
dje_mean=[1.5,2,2.5];                             % Mean Distance between Ohter J-E Pair

round=10000000;                                    % Trials

for z=1:1:length(dje_mean)
    for i=1:1:length(M)
        for k=1:1:round
            s=1;
            for j=2:1:M(i)
                dje(j)=dje_mean(z)+rand-0.5;
                s=s+dje(1)^(-alpha)/dje(j)^(-alpha);
            end
            k1=(s-M(i)+1)/s^2;                            % \kappa_1
            x(k)=max((M(i)-1)*R/c*k1,0);                  % Nash Equilibrium
        end
        PJ1(z,i)=sum(x)/round;                            % Averaged
    end
end

figure
hold on
box on
plot(M,PJ1(1,:),'k-v')
plot(M,PJ1(2,:),'b-o')
plot(M,PJ1(3,:),'r-s')
xlabel({'Number of jammers, $M$'},'interpreter','latex')
ylabel({'Optimal jamming power, $P_{J_1}^{ne}$'},'interpreter','latex')
legend({'$d_{J_i,E_i}\sim \mathcal{U}(1,2)$','$d_{J_i,E_i}\sim \mathcal{U}(1.5,2.5)$','$d_{J_i,E_i}↙
\sim \mathcal{U}(2,3)$'},'interpreter','latex')
```

```matlab
%-----------------Jammer Utility vs Number of Jammers--------------------%

clc
clear

%------------------------Network Settings-------------------------------%
c=1;                                              % Unit Cost
R=10;                                             % Rewards
alpha=4;                                          % Path Loss Exponent
M=[2,3,4,5,6];                                    % Number of Jammers
dje(1)=2;                                         % Distance between J1 and E1
dje_mean=[1.5,2,2.5];                             % Mean Distance between Ohter J-E Pair

round=100000;                                    % Trials

for z=1:1:length(dje_mean)
    for i=1:1:length(M)
        for k=1:1:round
            for j=2:1:M(i)
                dje(j)=dje_mean(z)+rand-0.5;
            end

            for j=1:1:M(i)
                s(j)=0;
                for l=1:1:M(i)
                    s(j)=s(j)+dje(j)^(-alpha)/dje(l)^(-alpha);
                end
                ka(j)=(s(j)-M(i)+1)/s(j)^2;               % \kappa_1
                P(j)=max((M(i)-1)*R/c*ka(j),0);           % Nash Equilibrium
            end

            temp=0;
            for j=1:1:M(i)
                temp=temp+P(j)*dje(j)^(-alpha);
            end

            Uj1(k)=max(P(1)*dje(1)^(-alpha)/temp*R-c*P(1),0.0001);  %U_{J_1}
        end

        UJ1(z,i)=sum(Uj1)/round;                                      % Averaged
    end
end

figure
hold on
box on
plot(M,UJ1(1,:),'k-v')
plot(M,UJ1(2,:),'b-o')
plot(M,UJ1(3,:),'r-s')
xlabel({'Number of jammers, $M$'},'interpreter','latex')
ylabel({'Optimal jammer utility, $U_{J_1}$'},'interpreter','latex')
legend({'$d_{J_i,E_i}\sim \mathcal{U}(1,2)$','$d_{J_i,E_i}\sim \mathcal{U}(1.5,2.5)$','$d_{J_i,E_i}↙
```

```
\sim \mathcal{U}(2,3)$'},'interpreter','latex')
```

```
%----------------------Security-QoS Tradeoff-----------------------------%

clc
clear

%-------------------------Network Settings-------------------------------%
K=3;                    % Hops
ds=5;                   % Distance of One Hop
dse(1)=sqrt(5^2+10^2);  % Distance between Eav and S1 (S)
dse(2)=10;              % Distance between Eav and S2
dse(3)=dse(1);          % Distance between Eav and S3
M=2;                    % Number of Eavs (Jammers)
alpha=4;                % Path Loss Exponent
gammaC=1;               % SNR Threshold for Receiver Decoding Successfully
gammaE=1;               % SNR Threshold for Eav Wiretapping Successfully
R=10;                   % Rewards

beta=0.01:0.01:0.2;                     % COP Constraint
dje=[1.5,2,2.5];                        % Distance between an Eav and Its Corresponding Jammer

for z=1:1:length(dje)
    djd(1)=10+dje(z);                   % Distance between Jammer and D1
    djd(2)=sqrt(ds^2+(10+dje(z))^2);    % Distance between Jammer and D2
    djd(3)=sqrt((2*ds)^2+(10+dje(z))^2);  % Distance between Jammer and D3 (D)

%------------------Determine PJ----------------------%
    kappa=1/M^2;
    PJ=R*kappa;

%----------------Compute SOP and PS*----------------------%
    weight=0;
    for k=1:1:K
        Xi(k)=gammaC*ds^alpha*M*PJ/djd(k)^alpha;
        Psi(k)=1/gammaE*M*dje(z)^alpha/PJ/dse(k)^alpha;
        weight=weight+sqrt(Xi(k)*Psi(k));
    end

    for i=1:1:length(beta)
        Upsilon=log(1/(1-beta(i)));

        Pso(z,i)=1-exp(-1/Upsilon*weight^2);

        PS1(z,i)=1/Upsilon*sqrt(Xi(1)/Psi(1))*weight;
        PS2(z,i)=1/Upsilon*sqrt(Xi(2)/Psi(2))*weight;
        PS3(z,i)=1/Upsilon*sqrt(Xi(3)/Psi(3))*weight;

%         for k=1:1:K
%             Ps(i,k)=1/Upsilon*sqrt(Xi(k)/Psi(k))*weight;
%         end
    end
end
```

```
figure
xlim([0.01 0.2])
xticks([0.01 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2])
xticklabels({'1','2','4','6','8','10','12','14','16','18','20'})
yticks([0 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18])
yticklabels({'0','2','4','6','8','10','12','14','16','18'})
hold on
box on
plot(beta,Pso(1,:),'k-')
plot(beta,Pso(2,:),'b--')
plot(beta,Pso(3,:),'r-.')
xlabel({'Constraint on COP, $\beta_{co} (\%)$'},'interpreter','latex')
ylabel({'Minimal achievable SOP, $P_{so}^* (\%)$'},'interpreter','latex')
legend({'$d_{J_i,E_i}=1.5$','$d_{J_i,E_i}=2$','$d_{J_i,E_i}=2.5$'},'interpreter','latex')

figure
xlim([0.01 0.2])
xticks([0.01 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2])
xticklabels({'1','2','4','6','8','10','12','14','16','18','20'})
hold on
box on
plot(beta,PS1(1,:),'k-')
plot(beta,PS1(2,:),'b--')
plot(beta,PS1(3,:),'r-.')
xlabel({'Constraint on COP, $\beta_{co} (\%)$'},'interpreter','latex')
ylabel({'Optimal transmission power, $P_{S_1}^*$'},'interpreter','latex')
legend({'$d_{J_i,E_i}=1.5$','$d_{J_i,E_i}=2$','$d_{J_i,E_i}=2.5$'},'interpreter','latex')

figure
xlim([0.01 0.2])
xticks([0.01 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2])
xticklabels({'1','2','4','6','8','10','12','14','16','18','20'})
hold on
box on
plot(beta,PS2(1,:),'k-')
plot(beta,PS2(2,:),'b--')
plot(beta,PS2(3,:),'r-.')
xlabel({'Constraint on COP, $\beta_{co} (\%)$'},'interpreter','latex')
ylabel({'Optimal transmission power, $P_{S_2}^*$'},'interpreter','latex')
legend({'$d_{J_i,E_i}=1.5$','$d_{J_i,E_i}=2$','$d_{J_i,E_i}=2.5$'},'interpreter','latex')

figure
xlim([0.01 0.2])
xticks([0.01 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2])
xticklabels({'1','2','4','6','8','10','12','14','16','18','20'})
hold on
box on
plot(beta,PS3(1,:),'k-')
plot(beta,PS3(2,:),'b--')
plot(beta,PS3(3,:),'r-.')
xlabel({'Constraint on COP, $\beta_{co} (\%)$'},'interpreter','latex')
ylabel({'Optimal transmission power, $P_{S_3}^*$'},'interpreter','latex')
```

```
legend({'$d_{J_i,E_i}=1.5$','$d_{J_i,E_i}=2$','$d_{J_i,E_i}=2.5$'},'interpreter','latex')
```

```matlab
%------------------------Snapshot Generation-----------------------------%
clc
clear

L=20;             % Square Length
N=30;             % Number of Legitimate Devices
M=3;              % Number of Eavs

%--------------------Position of Legitimate Devices---------------------%
Xl=L*rand(1,N);
Yl=L*rand(1,N);

%--------------------Position of Eavesdroppers---------------------------%
Xe=L*rand(1,M);
Ye=L*rand(1,M);

%--------------------Position of Jammers---------------------------%
for m=1:1:M
    Xj(m)=max(0,min(20,Xe(m)+4*rand-2));
    Yj(m)=max(0,min(20,Ye(m)+4*rand-2));
end

figure
box on
hold on

h=plot(Xl,Yl,'o','LineWidth',1,'Markersize',6,'Color',[0.50,0.50,0.50]);
set(h,'MarkerFaceColor',get(h,'color'));

plot(Xe,Ye,'kx','LineWidth',1,'Markersize',8)

h=plot(Xj,Yj,'g^','LineWidth',1,'Markersize',6,'Color',[0.93,0.69,0.13]);
set(h,'MarkerFaceColor',get(h,'color'));

xlabel('X','interpreter','latex')
ylabel('Y','interpreter','latex')
legend({'Network Nodes','Eavedroppers','Jammers'},'interpreter','latex')
```

```
%---------QIS Route Selection Algorithm (Based on Dijkstra)--------------%

function[OR,K,weight,w,dje,kappa,a]=QIS_algorithm(Lx,Ly,Ex,Ey,Jx,Jy,alpha,c,max_hopdis,dest_id)

N=length(Lx);              % Number of Nodes
M=length(Ex);              % Number of Eavs (Jammers)

a=zeros(N);                % Initialize the distance matrix
for i=1:1:N
    a(i,i)=inf;
    for j=(i+1):1:N
        a(i,j)=sqrt((Lx(i)-Lx(j))^2+(Ly(i)-Ly(j))^2);
        if(a(i,j)>max_hopdis)
            a(i,j)=inf;
        end
    end
end
a=a+a';               % Constitute the distance matrix

%-----------------------Compute Link Weight------------------------------%
for m=1:1:M
    dje(m)=sqrt((Jx(m)-Ex(m))^2+(Jy(m)-Ey(m))^2);  % Distance between Jammer and Eav
end

for i=1:1:M
    s(i)=0;
    for j=1:1:M
        s(i)=s(i)+dje(i)^(-alpha)/dje(j)^(-alpha);
    end
    kappa(i)=(s(i)-M+1)/s(i)^2;     %\kappa
end

for i=1:1:N
    for j=1:1:N
        if a(i,j)==inf
            w(i,j)=inf;
        else
            temp1=0;
            for m=1:1:M       % Note!!! m: Jammer Index; j: Rec Index
                djd(m,j)=sqrt((Jx(m)-Lx(j))^2+(Jy(m)-Ly(j))^2);
                temp1=temp1+kappa(m)/djd(m,j)^alpha;
            end
            temp2=0;
            for m=1:1:M       % Note!!! m: Eav Index; i: Tran Index
                dse(i,m)=sqrt((Lx(i)-Ex(m))^2+(Ly(i)-Ey(m))^2);
                temp2=temp2+dje(m)^alpha/kappa(m)/dse(i,m)^alpha;
            end
            w(i,j)=sqrt(a(i,j)^alpha*temp1*temp2);     % Link Weight Matrix
        end
    end
end
```

```
%----------Select the Minimum Sum Link Weight Route (Dijkstra)-----------%
ind(1:length(w))=0;    % Indicator Vector: indicate a node is whether (1) or not (0) added into the found set
ind(1)=1;

join=1;                % Found Set

pre_hop=ones(1,length(w));         % Vector for storing the previous hop on the optimal route

weight(1:length(w))=inf;           % Initialize the route weight
weight(1)=0;

new_join=1;                        % Index of the node that is newly added into the found set

while sum(ind)<length(w)                                            % Check whether all the nodes have been added into the found set
    njoin=find(ind==0);                                             % Update the unfounded set of nodes
    weight(njoin)=min(weight(njoin),weight(new_join)+w(new_join,njoin));   % Update the minimum route weight from the source to the unfounded nodes
    index=find(weight(njoin)==min(weight(njoin)));                  % Find the unfound node that has the minimum route weight
    new_join=njoin(index(1));                                       % This node is the new node that will be added into the found set
    ind(new_join)=1;                                                % Update the indicator of the new-join node
    join=[join,new_join];                                           % Add the new-join node into the found set
    temp=find( roundn(weight(join),-4)==roundn(weight(new_join)-w(join,new_join),-4)' );
    pre_hop(new_join)=join(temp(1));                                % Determine the previous hop device of the new-join node
end

weight;    % Minimum Route Weight from the Source to Any Other Node
join;      % We can check the order that nodes are added into the found set
pre_hop;   % Previous hop of a node on the optimal route from the source to this node

phop=dest_id;
K=0;              % Hops of the selected route
while phop~=1     % Source node in this snapshot is node 1
    phop=pre_hop(phop);
    K=K+1;
    S(K)=phop;    % Record the node of each hop in a backward manner
end

OR=[fliplr(S),dest_id];  % Route selected by QIS
```

```matlab
%----------QIS Routing Scheme & Performance Comparison-------------------%

clc
clear

%------------------------System Parameters------------------------------%
gammaE=1;                 % SNR Threshold for Eav Wiretapping Successfully
gammaC=1;                 % SNR Threshold for Receiver Decoding Successfully
R=10;                     % Rewards
alpha=4;                  % Path Loss Exponent
c=1;                      % Unit Power Cost
beta=0.05;                % Constraint on COP
Upsilon=-log(1-beta);

%----------------------Network Topology--------------------------------%
load snapshot_case1.mat;     % Load a Snapshot: Nodes (Lx,Ly); Eav (Ex,Ey); Jammer (Jx,Jy);
% load snapshot_case2.mat;
max_hopdis=7;                % Maximum Distance of One Hop
N=length(Lx);                 % Number of Nodes
M=length(Ex);                % Number of Eavs (Jammers)
dest_id=6;                   % Destination ID

[OR,K,weight,w,dje,kappa,a]=QIS_algorithm(Lx,Ly,Ex,Ey,Jx,Jy,alpha,c,max_hopdis,dest_id);
% OR: Optimal Route->Node ID Vector
% K: Hops of Optimal Route
% weight: Optimal Route Weight Vectors,e.g., weight(6) represents the optimal route weight between node 1 and node 6;
% w: Link Weight Matrix, e.g., w(i,j) is the weight of link_i,j, if link_i_j does not exist, w(i,j)=inf
% dje: Vector of Distance between Eav and Jammer
% a: Distance Matrix, e.g., a(i,j) is the distance between node i and node j, if a(i,j)>max_hopdis, a(i,j)=inf, means link_i_j does not exist

OR  %Optimal Route->Node ID Vector

for k=1:1:K
    tran_id=OR(k);
    rec_id=OR(k+1);
    ds(k)=a(tran_id,rec_id);   % Distance between S_k and D_k
    for i=1:1:M
        dse(k,i)=sqrt((Lx(tran_id)-Ex(i))^2+(Ly(tran_id)-Ey(i))^2);  % Distance between S_k and E_i
        djd(k,i)=sqrt((Jx(i)-Lx(rec_id))^2+(Jy(i)-Ly(rec_id))^2);    % Distance between J_i and D_k
    end
end

for i=1:1:M
    PJopt(i)=R*(M-1)/c*kappa(i);
end
PJopt     % Jamming Power, Nash Equilibrium

weight_opt=weight(dest_id)   % Weight of Optimal Route
```

```
Pso_opt=1-exp(-1/Upsilon*gammaC/gammaE*weight_opt^2)  % SOP of Optimal Route

for k=1:1:K
    Xi(k)=0;
    Psi(k)=0;
    for i=1:1:M
        Xi(k)=Xi(k)+gammaC*ds(k)^alpha*PJopt(i)/djd(k,i)^alpha;
        Psi(k)=Psi(k)+1/gammaE*dje(i)^alpha/PJopt(i)/dse(k,i)^alpha;
    end
    temp3(k)=sqrt(Xi(k)*Psi(k));
end

for k=1:1:K
    Ps_opt(k)=1/Upsilon*sqrt(Xi(k)/Psi(k))*sum(temp3);
end
Ps_opt   % Transmitting Power on Optimal Route

figure
hold on
box on

h=plot(Lx(1:6),Ly(1:6),'-p','Color',[0.00,1.00,0.00],'LineWidth',1,'Markersize',6);
set(h,'MarkerFaceColor',get(h,'color'));

%----Random Choose Route 1, 1->23->7->22->26->14->6-----------------------%
h=plot(Lx([1,23,7,22,26,14,6]),Ly([1,23,7,22,26,14,6]),'--square','LineWidth',1,'Markersize',
6,'Color',[1.00,0.00,0.00]);
set(h,'MarkerFaceColor',get(h,'color'));

%----Random Choose Route 2, 1->16->28->29->30->6--------------------------%
h=plot(Lx([1,16,28,29,30,6]),Ly([1,16,28,29,30,6]),'--diamond','LineWidth',1,'Markersize',6,'Color',
[0.00,0.00,1.00]);
set(h,'MarkerFaceColor',get(h,'color'));

%------------Other Nodes--------------------%
h=plot(Lx([8:13,15,17:21,24,25,27]),Ly([8:13,15,17:21,24,25,27]),'o','LineWidth',1,'Markersize',
6,'Color',[0.50,0.50,0.50]);
set(h,'MarkerFaceColor',get(h,'color'));

plot(Ex,Ey,'kx','LineWidth',1,'Markersize',8)

h=plot(Jx,Jy,'g^','LineWidth',1,'Markersize',6,'Color',[0.93,0.69,0.13]);
set(h,'MarkerFaceColor',get(h,'color'));

xlabel('X','interpreter','latex')
ylabel('Y','interpreter','latex')
legend({'Optimal Route','Route 1','Route 2','Network
Nodes','Eavedroppers','Jammers'},'interpreter','latex')

%--------------------Performance Comparision-----------------------------%

%-------Route 1: 1->23->7->22->26->14->6------------------------%
```

```
K1=6;    % Hops of Route 1
ds1(1)=a(1,23);
ds1(2)=a(23,7);
ds1(3)=a(7,22);
ds1(4)=a(22,26);
ds1(5)=a(26,14);
ds1(6)=a(14,6);
ds1     % Distance between S_k and D_k

%-------Distance between S_k and E_i--------------%
dse(1,1)=sqrt((Lx(1)-Ex(1))^2+(Ly(1)-Ey(1))^2);
dse(1,2)=sqrt((Lx(1)-Ex(2))^2+(Ly(1)-Ey(2))^2);
dse(1,3)=sqrt((Lx(1)-Ex(3))^2+(Ly(1)-Ey(3))^2);
dse(2,1)=sqrt((Lx(23)-Ex(1))^2+(Ly(23)-Ey(1))^2);
dse(2,2)=sqrt((Lx(23)-Ex(2))^2+(Ly(23)-Ey(2))^2);
dse(2,3)=sqrt((Lx(23)-Ex(3))^2+(Ly(23)-Ey(3))^2);
dse(3,1)=sqrt((Lx(7)-Ex(1))^2+(Ly(7)-Ey(1))^2);
dse(3,2)=sqrt((Lx(7)-Ex(2))^2+(Ly(7)-Ey(2))^2);
dse(3,3)=sqrt((Lx(7)-Ex(3))^2+(Ly(7)-Ey(3))^2);
dse(4,1)=sqrt((Lx(22)-Ex(1))^2+(Ly(22)-Ey(1))^2);
dse(4,2)=sqrt((Lx(22)-Ex(2))^2+(Ly(22)-Ey(2))^2);
dse(4,3)=sqrt((Lx(22)-Ex(3))^2+(Ly(22)-Ey(3))^2);
dse(5,1)=sqrt((Lx(26)-Ex(1))^2+(Ly(26)-Ey(1))^2);
dse(5,2)=sqrt((Lx(26)-Ex(2))^2+(Ly(26)-Ey(2))^2);
dse(5,3)=sqrt((Lx(26)-Ex(3))^2+(Ly(26)-Ey(3))^2);
dse(6,1)=sqrt((Lx(14)-Ex(1))^2+(Ly(14)-Ey(1))^2);
dse(6,2)=sqrt((Lx(14)-Ex(2))^2+(Ly(14)-Ey(2))^2);
dse(6,3)=sqrt((Lx(14)-Ex(3))^2+(Ly(14)-Ey(3))^2);

%-------Distance between J_i and D_k--------------%
djd(1,1)=sqrt((Jx(1)-Lx(23))^2+(Jy(1)-Ly(23))^2);
djd(1,2)=sqrt((Jx(2)-Lx(23))^2+(Jy(2)-Ly(23))^2);
djd(1,3)=sqrt((Jx(3)-Lx(23))^2+(Jy(3)-Ly(23))^2);
djd(2,1)=sqrt((Jx(1)-Lx(7))^2+(Jy(1)-Ly(7))^2);
djd(2,2)=sqrt((Jx(2)-Lx(7))^2+(Jy(2)-Ly(7))^2);
djd(2,3)=sqrt((Jx(3)-Lx(7))^2+(Jy(3)-Ly(7))^2);
djd(3,1)=sqrt((Jx(1)-Lx(22))^2+(Jy(1)-Ly(22))^2);
djd(3,2)=sqrt((Jx(2)-Lx(22))^2+(Jy(2)-Ly(22))^2);
djd(3,3)=sqrt((Jx(3)-Lx(22))^2+(Jy(3)-Ly(22))^2);
djd(4,1)=sqrt((Jx(1)-Lx(26))^2+(Jy(1)-Ly(26))^2);
djd(4,2)=sqrt((Jx(2)-Lx(26))^2+(Jy(2)-Ly(26))^2);
djd(4,3)=sqrt((Jx(3)-Lx(26))^2+(Jy(3)-Ly(26))^2);
djd(5,1)=sqrt((Jx(1)-Lx(14))^2+(Jy(1)-Ly(14))^2);
djd(5,2)=sqrt((Jx(2)-Lx(14))^2+(Jy(2)-Ly(14))^2);
djd(5,3)=sqrt((Jx(3)-Lx(14))^2+(Jy(3)-Ly(14))^2);
djd(6,1)=sqrt((Jx(1)-Lx(6))^2+(Jy(1)-Ly(6))^2);
djd(6,2)=sqrt((Jx(2)-Lx(6))^2+(Jy(2)-Ly(6))^2);
djd(6,3)=sqrt((Jx(3)-Lx(6))^2+(Jy(3)-Ly(6))^2);

weight1=w(1,23)+w(23,7)+w(7,22)+w(22,26)+w(26,14)+w(14,6)   % Weight of Route 1

Pso1=1-exp(-1/Upsilon*gammaC/gammaE*weight1^2)  % SOP of Route 1
```

```
for k=1:1:K1
    Xi(k)=0;
    Psi(k)=0;
    for i=1:1:M
        Xi(k)=Xi(k)+gammaC*ds1(k)^alpha*PJopt(i)/djd(k,i)^alpha;
        Psi(k)=Psi(k)+1/gammaE*dje(i)^alpha/PJopt(i)/dse(k,i)^alpha;
    end
    temp3(k)=sqrt(Xi(k)*Psi(k));
end

for k=1:1:K1
    Ps1(k)=1/Upsilon*sqrt(Xi(k)/Psi(k))*sum(temp3);
end
Ps1   % Transmitting Power on Route 1



%-------Route 2: 1->16->28->29->30->6------------------------%
K2=5;    % Hops of Route 2
ds2(1)=a(1,16);
ds2(2)=a(16,28);
ds2(3)=a(28,29);
ds2(4)=a(29,30);
ds2(5)=a(30,6);
ds2    % Distance between S_k and D_k

%-------Distance between S_k and E_i--------------%
dse(1,1)=sqrt((Lx(1)-Ex(1))^2+(Ly(1)-Ey(1))^2);
dse(1,2)=sqrt((Lx(1)-Ex(2))^2+(Ly(1)-Ey(2))^2);
dse(1,3)=sqrt((Lx(1)-Ex(3))^2+(Ly(1)-Ey(3))^2);
dse(2,1)=sqrt((Lx(16)-Ex(1))^2+(Ly(16)-Ey(1))^2);
dse(2,2)=sqrt((Lx(16)-Ex(2))^2+(Ly(16)-Ey(2))^2);
dse(2,3)=sqrt((Lx(16)-Ex(3))^2+(Ly(16)-Ey(3))^2);
dse(3,1)=sqrt((Lx(28)-Ex(1))^2+(Ly(28)-Ey(1))^2);
dse(3,2)=sqrt((Lx(28)-Ex(2))^2+(Ly(28)-Ey(2))^2);
dse(3,3)=sqrt((Lx(28)-Ex(3))^2+(Ly(28)-Ey(3))^2);
dse(4,1)=sqrt((Lx(29)-Ex(1))^2+(Ly(29)-Ey(1))^2);
dse(4,2)=sqrt((Lx(29)-Ex(2))^2+(Ly(29)-Ey(2))^2);
dse(4,3)=sqrt((Lx(29)-Ex(3))^2+(Ly(29)-Ey(3))^2);
dse(5,1)=sqrt((Lx(30)-Ex(1))^2+(Ly(30)-Ey(1))^2);
dse(5,2)=sqrt((Lx(30)-Ex(2))^2+(Ly(30)-Ey(2))^2);
dse(5,3)=sqrt((Lx(30)-Ex(3))^2+(Ly(30)-Ey(3))^2);

%-------Distance between J_i and D_k--------------%
djd(1,1)=sqrt((Jx(1)-Lx(16))^2+(Jy(1)-Ly(16))^2);
djd(1,2)=sqrt((Jx(2)-Lx(16))^2+(Jy(2)-Ly(16))^2);
djd(1,3)=sqrt((Jx(3)-Lx(16))^2+(Jy(3)-Ly(16))^2);
djd(2,1)=sqrt((Jx(1)-Lx(28))^2+(Jy(1)-Ly(28))^2);
djd(2,2)=sqrt((Jx(2)-Lx(28))^2+(Jy(2)-Ly(28))^2);
djd(2,3)=sqrt((Jx(3)-Lx(28))^2+(Jy(3)-Ly(28))^2);
djd(3,1)=sqrt((Jx(1)-Lx(29))^2+(Jy(1)-Ly(29))^2);
djd(3,2)=sqrt((Jx(2)-Lx(29))^2+(Jy(2)-Ly(29))^2);
```

```
djd(3,3)=sqrt((Jx(3)-Lx(29))^2+(Jy(3)-Ly(29))^2);
djd(4,1)=sqrt((Jx(1)-Lx(30))^2+(Jy(1)-Ly(30))^2);
djd(4,2)=sqrt((Jx(2)-Lx(30))^2+(Jy(2)-Ly(30))^2);
djd(4,3)=sqrt((Jx(3)-Lx(30))^2+(Jy(3)-Ly(30))^2);
djd(5,1)=sqrt((Jx(1)-Lx(6))^2+(Jy(1)-Ly(6))^2);
djd(5,2)=sqrt((Jx(2)-Lx(6))^2+(Jy(2)-Ly(6))^2);
djd(5,3)=sqrt((Jx(3)-Lx(6))^2+(Jy(3)-Ly(6))^2);

weight2=w(1,16)+w(16,28)+w(28,29)+w(29,30)+w(30,6)   % Weight of Route 2

Pso2=1-exp(-1/Upsilon*gammaC/gammaE*weight2^2)    % SOP of Route 2

for k=1:1:K2
    Xi(k)=0;
    Psi(k)=0;
    for i=1:1:M
        Xi(k)=Xi(k)+gammaC*ds2(k)^alpha*PJopt(i)/djd(k,i)^alpha;
        Psi(k)=Psi(k)+1/gammaE*dje(i)^alpha/PJopt(i)/dse(k,i)^alpha;
    end
    temp3(k)=sqrt(Xi(k)*Psi(k));
end

for k=1:1:K2
    Ps2(k)=1/Upsilon*sqrt(Xi(k)/Psi(k))*sum(temp3);
end
Ps2
```