```matlab
%---------------Monte Carlo Simulation------------------%
clc
clear

%--------------network setting--------------------------%
gammaE=1;          % Requied SNR for Eav
P=10;              % Transmit Power
Pj=1:1:10;         % Jamming Power
K=3;               % Hops
dje=[4,3,2];       % Distance Between Jammer and Eav
alpha=4;           % Fading coefficient
M=2;               % Number of Eavs
round=10000000;    % Trials
ds=5;              % Distance of a hop

dse(1)=sqrt(7.5^2+10^2);   % Distance between Eav and S1
dse(2)=sqrt(2.5^2+10^2);   % Distance between Eav and S2
dse(3)=dse(2);             % Distance between Eav and S3

%-------------------------Theoretical Results-----------------------%
for l=1:1:length(dje)
    for i=1:1:length(Pj)
        for k=1:1:K
            omega(k)=(gammaE*Pj(i))^(-1)*M*dje(l)^alpha/dse(k)^alpha;
        end
        Pso(l,i)=1-exp(-P*sum(omega));
    end
end

%----------------------Monte Carlo Resutls------------------------%
for l=1:1:length(dje)
    for i=1:1:length(Pj)
        for k=1:1:K
            num(k)=0;
            for r=1:1:round
                hs=exprnd(1);
                for m=1:1:M
                    hj(m)=exprnd(1);
                    SIR(m)=P*hs/dse(k)^alpha/Pj(i)/hj(m)*dje(l)^alpha;
                end
                if(max(SIR)>gammaE)
                    num(k)=num(k)+1;
                end
            end
            Psol(k)=num(k)/round;
        end
        temp=1;
        for k=1:1:K
            temp=temp*(1-Psol(k));
        end
        Pso_Mon(l,i)=1-temp;
    end
```

```
end

figure
hold on
box on
plot(Pj,Pso(1,:),'b-square')
plot(Pj,Pso_Mon(1,:),'bpentagram')
plot(Pj,Pso(2,:),'r-square')
plot(Pj,Pso_Mon(2,:),'rpentagram')
plot(Pj,Pso(3,:),'k-square')
plot(Pj,Pso_Mon(3,:),'kpentagram')
xlabel({'Jamming power, $$P_J$$'},'interpreter','latex')
ylabel({'Secrecy outage probability, $$P_{so}$$'},'interpreter','latex')
legend({'Theoretical (Eq. (1)), $d_{J_i,E_i}=4$','Monte Carlo Simulation, $d_{J_i,E_i}=4$','Theoretical (Eq. (1)), $d_{J_i,E_i}=3$','Monte Carlo Simulation, $d_{J_i,E_i}=3$','Theoretical (Eq. (1)), $d_{J_i,E_i}=2$','Monte Carlo Simulation, $d_{J_i,E_i}=2$'},'interpreter','latex')
```

```
%----------------Jamming Power vs Number of Jammers----------------------%
clc
clear

%-------------------------network setting---------------------------------%
c=1;                        % Unit Power Cost
R=1;                        % Rewards
alpha=4;                    % Fading coefficient
M=[2,3,4,5,6,7,8];          % Number of Eavs
round=1000000;              % Trials
dje1=3;                     % Distance between J1 and E1
dje_mean=[1,2,3];           % Mean Distance between Other Paris of Jammer-Eav

%-------------------------numerical results------------------------------%
for l=1:1:length(dje_mean)
    for i=1:1:length(M)
        for k=1:1:round
            s=1;
            for j=1:1:M(i)-1
                dje(j)=dje_mean(l)+2*rand;
                s=s+dje1^(-alpha)/dje(j)^(-alpha);
            end
            k1=(M(i)-1)*(s-M(i)+1)/c/s^2;
            x(k)=max(R*k1,0);
        end
        PJ1(l,i)=sum(x)/round;
    end
end

figure
hold on
box on
plot(M,PJ1(1,:),'r-square')
plot(M,PJ1(2,:),'b-o')
plot(M,PJ1(3,:),'k-^')
xlabel({'Number of jammers, $M$'},'interpreter','latex')
ylabel({'Optimal jamming power, $P_{J}^{ne}$'},'interpreter','latex')
legend({'$d_{J_i,E_i}\sim \mathcal{U}(1,3)$','$d_{J_i,E_i}\sim \mathcal{U}(2,4)$','$d_{J_i,E_i}\sim
\mathcal{U}(3,5)$'},'interpreter','latex')
```

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

%------------------------network setting---------------------------------%
c=1;                         % Unit Power Cost
R=1;                         % Rewards
alpha=4;                     % Fading coefficient
M=[2,3,4,5,6,7,8];           % Number of Eavs
round=1000000;               % Trials
dje(1)=3;                    % Distance between J1 and E1
dje_mean=[1,2,3];            % Mean Distance between Other Paris of Jammer-Eav

%------------------------numerical results-------------------------------%
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)+2*rand;
            end

            for j=1:1:M(i)  % compute s(j)
                s(j)=0;
                for l=1:1:M(i)
                    s(j)=s(j)+dje(l)^alpha/dje(j)^alpha;
                end
            end

            for j=1:1:M(i)
                PJ(j)=max(R/c*(M(i)-1)*(s(j)-M(i)+1)/s(j)^2,0);
            end

            sigma=0;
            for j=1:1:M(i)
                sigma=sigma+PJ(j)/dje(j)^alpha;
            end

            x(k)=PJ(1);
            y(k)=max(PJ(1)/dje(1)^alpha/sigma*R-c*PJ(1),0);
        end

        PJ1(z,i)=sum(x)/round;
        UJ1(z,i)=sum(y)/round;
    end
end

figure
hold on
box on
plot(M,UJ1(1,:),'r-square')
plot(M,UJ1(2,:),'b-o')
plot(M,UJ1(3,:),'k-^')
```

```
xlabel({'Number of jammers, $M$'},'interpreter','latex')
ylabel({'Optimal jammer utility, $U_{J_1}^{ne}$'},'interpreter','latex')
legend({'$d_{J_i,E_i}\sim \mathcal{U}(1,3)$','$d_{J_i,E_i}\sim \mathcal{U}(2,4)$','$d_{J_i,E_i}\sim \mathcal{U}(3,5)$'},'interpreter','latex')
```

```
%----------------------Source Utility vs Rewards-------------------------%
clear
clc

%------------------------network setting--------------------------------%
gammaE=1;               % Requied SNR for Eav
P=10;                   % Transmit Power
lambda=20;              % System Parameter
M=[2,3];                % Number of Eavs
alpha=4;                % Fading cCefficient
c=1;                    % Unit Power Cost
K=3;                    % Hops
dse=10;                 % Distance between Transmitter and Eav
dje=[2,3];              % Distance between Jammer and Eav
R=0:.1:15;              % Rewards

%-----------------------numerical results--------------------------------%
figure
hold on
box on

for X=1:1:length(M)

    for Y=1:1:length(dje)

        for i=1:1:M(X)
            s=0;
            for j=1:1:M(X)
                s=s+dje(Y)^(-alpha)/dje(Y)^(-alpha);
            end
            kappa(i)=(M(X)-1)*(s-M(X)+1)/c/s^2;
        end

        s=0;
        for i=1:1:M(X)
            s=s+dje(Y)^alpha/dse^alpha/kappa(i);
        end
        delta=P*K/gammaE*s;

        for i=1:1:length(R)
            Us(i)=lambda/(1+delta/R(i))-R(i);
        end

        plot(R,Us)

    end

end

xlabel({'Source reward, $R$'},'interpreter','latex')
ylabel({'Source utility, $U_S$'},'interpreter','latex')
legend({'$d_{J_i,E_i}=2, M=2$','$d_{J_i,E_i}=3, M=2$','$d_{J_i,E_i}=2, M=3$','$d_{J_i,E_i}=3,
```

```
M=3$'},'interpreter','latex')
```

```matlab
%------------Optimal Source Rewards vs Number of Eavesdroppers-----------%
clear
clc

%---------------------------network setting--------------------------------%
gammaE=1;              % Requied SNR for Eav
P=10;                  % Transmit Power
lambda=20;             % System Parameter
M=2:1:10;              % Number of Eavs
alpha=4;               % Fading coefficient
c=1;                   % Unit Power Cost
K=[3,4];               % Hops
dse=[10,15];           % Distance Between Transmitter and Eav

%-------------------------numerical results--------------------------------%
for X=1:1:length(K)

    for Y=1:1:length(dse)

        for k=1:1:length(M)
            dje=2*ones(1,M(k));
            for i=1:1:M(k)
                s=0;
                for j=1:1:M(k)
                    s=s+dje(i)^(-alpha)/dje(j)^(-alpha);
                end
                kappa(i)=(M(k)-1)*(s-M(k)+1)/c/s^2;
            end

            s=0;
            for i=1:1:M(k)
                s=s+dje(i)^alpha/dse(Y)^alpha/kappa(i);
            end
            delta=P*K(X)/gammaE*s;
            R_opt(X,Y,k)=sqrt(lambda*delta)-delta;

        end

    end

end

figure
hold on
box on
plot(M,squeeze(R_opt(1,1,:)),'b-o')
plot(M,squeeze(R_opt(1,2,:)),'r-square')
plot(M,squeeze(R_opt(2,1,:)),'-^','Color',[0.23,0.44,0.34])
plot(M,squeeze(R_opt(2,2,:)),'k-v')
xlabel({'Number of eavesdroppers, $M$'},'interpreter','latex')
ylabel({'Optimal source reward, $R^*$'},'interpreter','latex')
legend({'$d_{S_k,E_i}=10, K=3$','$d_{S_k,E_i}=15, K=3$','$d_{S_k,E_i}=10, K=4$','$d_{S_k,E_i}=15, ↙
```

```
K=4$'},'interpreter','latex')
```

```
%------------Optimal Source Utility vs Number of Eavesdroppers-----------%
clear
clc

%--------------------------network setting-------------------------------%
gammaE=1;               % Requied SNR for Eav
P=10;                   % Transmit Power
lambda=20;              % System Parameter
M=2:1:10;               % Number of Eavs
alpha=4;                % Fading coefficient
c=1;                    % Unit Power Cost
K=[3,4];                % Hops
dse=[10,15];            % Distance Between Transmitter and Eav

%------------------------numerical results-------------------------------%
for X=1:1:length(K)

    for Y=1:1:length(dse)

        for k=1:1:length(M)
            dje=2*ones(1,M(k));
            for i=1:1:M(k)
                s=0;
                for j=1:1:M(k)
                    s=s+dje(i)^(-alpha)/dje(j)^(-alpha);
                end
                kappa(i)=(M(k)-1)*(s-M(k)+1)/c/s^2;
            end

            s=0;
            for i=1:1:M(k)
                s=s+dje(i)^alpha/dse(Y)^alpha/kappa(i);
            end
            delta=P*K(X)/gammaE*s;
            U_opt(X,Y,k)=sqrt(lambda)-sqrt(delta);

        end

    end

end

figure
hold on
box on
plot(M,squeeze(U_opt(1,1,:)),'b-o')
plot(M,squeeze(U_opt(1,2,:)),'r-square')
plot(M,squeeze(U_opt(2,1,:)),'-^','Color',[0.23,0.44,0.34])
plot(M,squeeze(U_opt(2,2,:)),'k-v')
xlabel({'Number of eavesdroppers, $M$'},'interpreter','latex')
ylabel({'Optimal source utility, $U_S^*$'},'interpreter','latex')
legend({'$d_{S_k,E_i}=10, K=3$','$d_{S_k,E_i}=15, K=3$','$d_{S_k,E_i}=10, K=4$','$d_{S_k,E_i}=15,
```

```
K=4$'},'interpreter','latex')
```

```
%------------------------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
plot(Xl,Yl,'ro')
plot(Xe,Ye,'kx')
plot(Xj,Yj,'b^')
```

```matlab
%------------IJS Route Selection Algorithm (Based on Dijkstra)-----------%
clc
clear all

%-----------------compute network topology--------------------------------%
load snapshot.mat;   % load a snapshot
N=30;                % Number of Legitimate Devices
M=3;                 % Number of Eavs (Jammers)
alpha=4;             % Fading Coefficient
c=1;                 % Unit Power Costt

a=zeros(30);         % 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)>7)
            a(i,j)=inf;
        end
    end
end
a=a+a'               % constitute the distance matrix

%-----------------------compute link weight-------------------------------%
for i=1:1:M
    dje(i)=sqrt((Jx(i)-Ex(i))^2+(Jy(i)-Ey(i))^2);
end
dje   % Distance between Jammer and Eav

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)=(M-1)*(s(i)-M+1)/c/s(i)^2;
end
kappa

for i=1:1:N
    for j=1:1:N
        if a(i,j)==inf
            w(i,j)=inf;
        else
            w(i,j)=0;
            for k=1:1:M
                dse(i,k)=sqrt((Lx(i)-Ex(k))^2+(Ly(i)-Ey(k))^2);
                w(i,j)=w(i,j)+dje(k)^alpha/kappa(k)/dse(i,k)^alpha;
            end
        end
    end
end
w          % compute the link weight matrix
```

```
%----------select the minimum sum link weight route (Dijkstra)-----------%
ind(1:length(w))=0;    % Indicator Vector: indicate a device 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 device that is newly added into the found set

while sum(ind)<length(w)                                              % check whether all the
devices have been added into the found set
    njoin=find(ind==0);                                               % update the unfounded
set of devices
    weight(njoin)=min(weight(njoin),weight(new_join)+w(new_join,njoin));   % update the minimum
route weight from the source to the unfounded devices
    index=find(weight(njoin)==min(weight(njoin)));                    % find the unfound device
that has the minimum route weight
    new_join=njoin(index(1));                                         % this device is the new
device that will be added into the found set
    ind(new_join)=1;                                                  % update the indicator of
the new-join device
    join=[join,new_join];                                             % add the new-join device
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 devie
end

weight     % minimum route weight from the source to any other devices
join       % we can check the order that devices are added into the found set
pre_hop    % previous hop of a device on the optimal route from the source to this device

dest_id=6;         % Destination Device in this snapshot is node 6
phop=dest_id;
K=0;               % Hops of the selected route
while phop~=1      % Source Device in this snapshot is node 1
    phop=pre_hop(phop);
    K=K+1;
    S(K)=phop;     % record the device of each hop in a backward manner
end

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

```
%--------------Route Performance Comparison of a Snapshot----------------%
clc
clear

%-----------------------------network setting-----------------------------%
gammaE=1;         % Requied SNR for Eav
P=10;             % Transmit Power
alpha=4;          % Fading coefficient
M=3;              % Number of Eavs
c=1;              % Unit Power Costt
lambda=20;        % System Parameter

%---------------------network topology of a snapshot----------------------%
load snapshot.mat

Lx;  % X Coordinate of Legitimate Devices
Ly;  % Y Coordinate of Legitimate Devices

Ex;  % X Coordinate of Eavs
Ey;  % X Coordinate of Eavs

Jx;  % X Coordinate of Jammers
Jy;  % X Coordinate of Jammers

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

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)=(M-1)*(s(i)-M+1)/c/s(i)^2;
end
kappa

%-------------------optimal path selected by IJS------------------------%
%---------------------<S1, S2, S3, S4, S5, S6>--------------------------%
K=5;            % Hops
for k=1:1:K
    ds(k)=sqrt((Lx(k+1)-Lx(k))^2+(Ly(k+1)-Ly(k))^2);
end
ds    % Hop Distance on the Optimal Path

%---calculate the link weight on the optimal path---%
for k=1:1:K
    Xiopt(k)=0;
    for i=1:1:M
        dse(k,i)=sqrt((Lx(k)-Ex(i))^2+(Ly(k)-Ey(i))^2);
        Xiopt(k)=Xiopt(k)+dje(i)^alpha/kappa(i)/dse(k,i)^alpha;
```

```
    end
end

Xiopt       % Link Weight on the Optimal Path
sum(Xiopt)  % Sum Link Weight of the Optimal Path

%---calculate the metrics on the optimal path---%
Delta=P/gammaE*sum(Xiopt);
Ropt=sqrt(lambda*Delta)-Delta        % Rewards
USopt=(sqrt(lambda)-sqrt(Delta))^2   % Source Utility
PJopt(1)=Ropt*kappa(1);
PJopt(2)=Ropt*kappa(2);
PJopt(3)=Ropt*kappa(3);
PJopt                                % Jamming Power

for k=1:1:K
    omega(k)=0;
    for i=1:1:M
        omega(k)=omega(k)+dje(i)^alpha/PJopt(i)/dse(k,i)^alpha/gammaE;
    end
end

PSOopt=1-exp(-P*sum(omega))          % Secrecy Outage Probability

%-------------------------Path 1------------------------------------------%
%---------------------<S1, S23, S24, S25, S26, S6>------------------------%
ds1(1)=sqrt((Lx(23)-Lx(1))^2+(Ly(23)-Ly(1))^2);
ds1(2)=sqrt((Lx(24)-Lx(23))^2+(Ly(24)-Ly(23))^2);
ds1(3)=sqrt((Lx(25)-Lx(24))^2+(Ly(25)-Ly(24))^2);
ds1(4)=sqrt((Lx(26)-Lx(25))^2+(Ly(26)-Ly(25))^2);
ds1(5)=sqrt((Lx(6)-Lx(26))^2+(Ly(6)-Ly(26))^2);
ds1   % Hop Distance on Path 1

%---calculate the link weight on Path 1---%
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);
Xi1(1)=dje(1)^alpha/kappa(1)/dse(1,1)^alpha+dje(2)^alpha/kappa(2)/dse(1,2)^alpha+dje(3)^alpha/kappa↙
(3)/dse(1,3)^alpha;

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);
Xi1(2)=dje(1)^alpha/kappa(1)/dse(2,1)^alpha+dje(2)^alpha/kappa(2)/dse(2,2)^alpha+dje(3)^alpha/kappa↙
(3)/dse(2,3)^alpha;

dse(3,1)=sqrt((Lx(24)-Ex(1))^2+(Ly(24)-Ey(1))^2);
dse(3,2)=sqrt((Lx(24)-Ex(2))^2+(Ly(24)-Ey(2))^2);
dse(3,3)=sqrt((Lx(24)-Ex(3))^2+(Ly(24)-Ey(3))^2);
Xi1(3)=dje(1)^alpha/kappa(1)/dse(3,1)^alpha+dje(2)^alpha/kappa(2)/dse(3,2)^alpha+dje(3)^alpha/kappa↙
(3)/dse(3,3)^alpha;
```

```
dse(4,1)=sqrt((Lx(25)-Ex(1))^2+(Ly(25)-Ey(1))^2);
dse(4,2)=sqrt((Lx(25)-Ex(2))^2+(Ly(25)-Ey(2))^2);
dse(4,3)=sqrt((Lx(25)-Ex(3))^2+(Ly(25)-Ey(3))^2);
Xi1(4)=dje(1)^alpha/kappa(1)/dse(4,1)^alpha+dje(2)^alpha/kappa(2)/dse(4,2)^alpha+dje(3)^alpha/kappa(3)/dse(4,3)^alpha;

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);
Xi1(5)=dje(1)^alpha/kappa(1)/dse(5,1)^alpha+dje(2)^alpha/kappa(2)/dse(5,2)^alpha+dje(3)^alpha/kappa(3)/dse(5,3)^alpha;

Xi1       % Link Weight on the Optimal Path 1
sum(Xi1)  % Sum Link Weight of Path 1

%---calculate the metrics on Path 1---%
Delta=P/gammaE*sum(Xi1);
R1opt=sqrt(lambda*Delta)-Delta          % Rewards
US1opt=(sqrt(lambda)-sqrt(Delta))^2     % Source Utility
PJ1opt(1)=R1opt*kappa(1);
PJ1opt(2)=R1opt*kappa(2);
PJ1opt(3)=R1opt*kappa(3);
PJ1opt                                  % Jamming Power

for k=1:1:K
    omega(k)=0;
    for i=1:1:M
        omega(k)=omega(k)+dje(i)^alpha/PJ1opt(i)/dse(k,i)^alpha/gammaE;
    end
end

PSO1opt=1-exp(-P*sum(omega))            % Secrecy Outage Probability

%--------------------------Path 2-----------------------------------------%
%----------------------<S1, S27, S28, S29, S30, S6>-----------------------%
ds2(1)=sqrt((Lx(27)-Lx(1))^2+(Ly(27)-Ly(1))^2);
ds2(2)=sqrt((Lx(28)-Lx(27))^2+(Ly(28)-Ly(27))^2);
ds2(3)=sqrt((Lx(29)-Lx(28))^2+(Ly(29)-Ly(28))^2);
ds2(4)=sqrt((Lx(30)-Lx(29))^2+(Ly(30)-Ly(29))^2);
ds2(5)=sqrt((Lx(6)-Lx(30))^2+(Ly(6)-Ly(30))^2);

ds2   % Hop Distance on the Path 2

%---calculate the link weight on Path 2---%
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);
Xi2(1)=dje(1)^alpha/kappa(1)/dse(1,1)^alpha+dje(2)^alpha/kappa(2)/dse(1,2)^alpha+dje(3)^alpha/kappa(3)/dse(1,3)^alpha;

dse(2,1)=sqrt((Lx(27)-Ex(1))^2+(Ly(27)-Ey(1))^2);
dse(2,2)=sqrt((Lx(27)-Ex(2))^2+(Ly(27)-Ey(2))^2);
```

```
dse(2,3)=sqrt((Lx(27)-Ex(3))^2+(Ly(27)-Ey(3))^2);
Xi2(2)=dje(1)^alpha/kappa(1)/dse(2,1)^alpha+dje(2)^alpha/kappa(2)/dse(2,2)^alpha+dje(3)^alpha/kappa
(3)/dse(2,3)^alpha;

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);
Xi2(3)=dje(1)^alpha/kappa(1)/dse(3,1)^alpha+dje(2)^alpha/kappa(2)/dse(3,2)^alpha+dje(3)^alpha/kappa
(3)/dse(3,3)^alpha;

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);
Xi2(4)=dje(1)^alpha/kappa(1)/dse(4,1)^alpha+dje(2)^alpha/kappa(2)/dse(4,2)^alpha+dje(3)^alpha/kappa
(3)/dse(4,3)^alpha;

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);
Xi2(5)=dje(1)^alpha/kappa(1)/dse(5,1)^alpha+dje(2)^alpha/kappa(2)/dse(5,2)^alpha+dje(3)^alpha/kappa
(3)/dse(5,3)^alpha;

Xi2       % Link Weight on the Optimal Path 2
sum(Xi2)  % Sum Link Weight of Path 2

%---calculate the metrics on Path 2---%
Delta=P/gammaE*sum(Xi2);
R2opt=sqrt(lambda*Delta)-Delta          % Rewards
US2opt=(sqrt(lambda)-sqrt(Delta))^2     % Source Utility
PJ2opt(1)=R2opt*kappa(1);
PJ2opt(2)=R2opt*kappa(2);
PJ2opt(3)=R2opt*kappa(3);
PJ2opt                                  % Jamming Power

for k=1:1:K
    omega(k)=0;
    for i=1:1:M
        omega(k)=omega(k)+dje(i)^alpha/PJ2opt(i)/dse(k,i)^alpha/gammaE;
    end
end
PSO2opt=1-exp(-P*sum(omega))            % Secrecy Outage Probability

figure
hold on
box on

h=plot(Lx(1:6),Ly(1:6),'-p','Color',[0.00,0.45,0.74],'LineWidth',1,'Markersize',8);
set(h,'MarkerFaceColor',get(h,'color'));

h=plot(Lx([1,23:26,6]),Ly([1,23:26,6]),'--diamond','LineWidth',1,'Markersize',8,'Color',
[0.47,0.67,0.19]);
set(h,'MarkerFaceColor',get(h,'color'));
```

```
h=plot(Lx([1,27:30,6]),Ly([1,27:30,6]),'--square','LineWidth',1,'Markersize',8,'Color',[0.93,0.69,0.13]);
set(h,'MarkerFaceColor',get(h,'color'));

plot(Lx(7:22),Ly(7:22),'ro','LineWidth',1,'Markersize',8)

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

h=plot(Jx,Jy,'g^','LineWidth',1,'Markersize',8,'Color',[0.3,0.75,0.93]);
set(h,'MarkerFaceColor',get(h,'color'));

xlabel('X','interpreter','latex')
ylabel('Y','interpreter','latex')
legend({'optimal path','path 1','path 2','network nodes','eavedroppers','jammers'},'interpreter','latex')
```