

Prestazioni Modulazioni

Francesco Musto 0612707371

Andrea Savastano 0612707904

Mario Zito 0612708073

November 18, 2024

Contents

1	Costellazioni Modulazioni	2
1.1	Costellazioni con segnali ricevuti	2
2	Confronto PAM - PPM	3
2.1	2-PAM con 2-PPM	3
2.2	16-PAM con 16-PPM	3
3	Prestazioni 16-QAM	4
3.1	Senza Fading	4
3.2	Con Fading	4
3.3	Con Tecnica di Diversità in tempo/frequenza	4
4	Prestazioni 16-PSK al variare di L	5
4.1	L=2	5
4.2	L=5	5
5	Prestazioni 8-PPM al variare di MC	6
5.1	MC=10 ⁴	6
5.2	MC=10 ⁶	6
6	Resoconto finale	7
	Appendice: Codice Matlab	7
A	Main	7
B	Generatore costellazioni	8
B.1	Generatore PAM	8
B.2	Generatore QAM	8
B.3	Generatore PSK	9
B.4	Generatore PPM	9
C	Stima $P_s(e)$	10
C.1	$P_s(e)$ senza Fading	10
C.2	$P_s(e)$ con Fading	11
C.3	$P_s(e)$ con Fading e tecnica diversità	12
C.3.1	myexprnd ausiliaria	13

1 Costellazioni Modulazioni

Delle 4 modulazioni prese in esame nelle simulazioni (PSK, PAM, QAM, PPM) ne vengono stampate le costellazioni, ove possibile, nel caso di $M=16$ ed $E_{av}=1$.

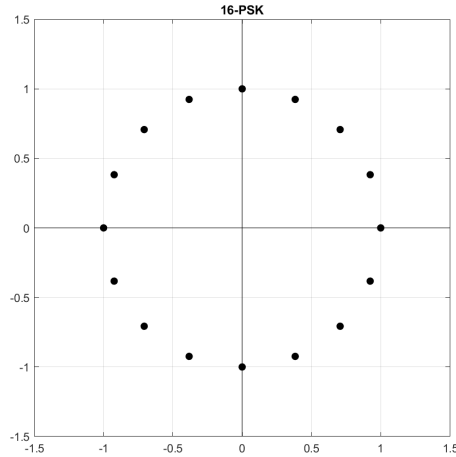


Figure 1: 16-PSK

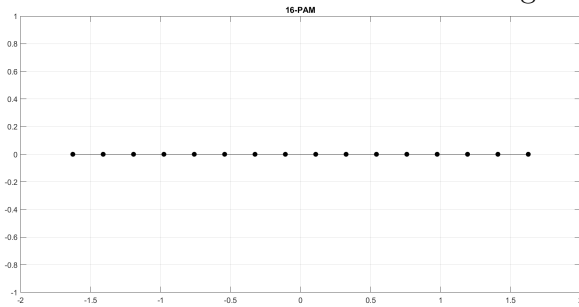


Figure 2: 16-PAM

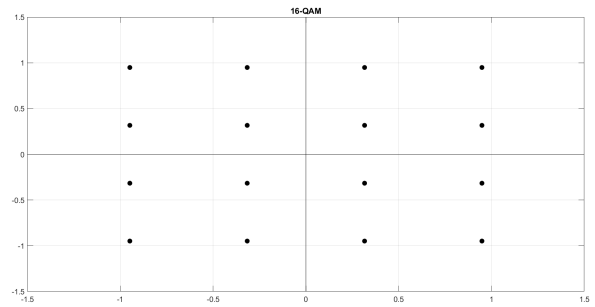
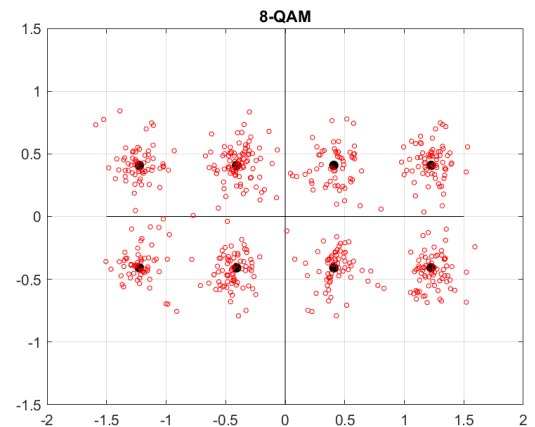
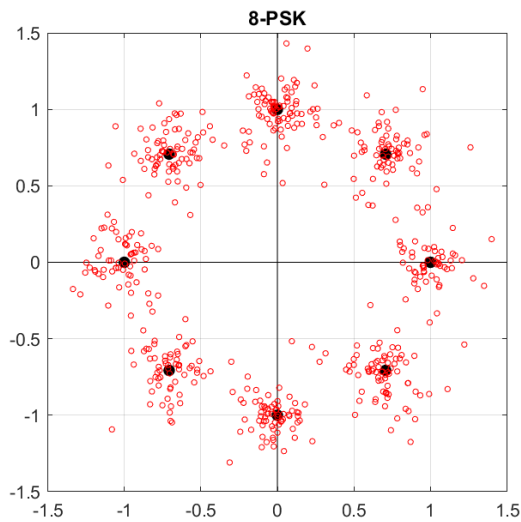


Figure 3: 16-QAM

1.1 Costellazioni con segnali ricevuti

Prendendo in considerazione le modulazioni PSK e QAM con $M=8$, $E_{av}=1$, $SNR_{dB}=[10:20]$ e 50 prove MonteCarlo, ne vengono stampate le costellazioni con i segnali ricevuti in rosso.

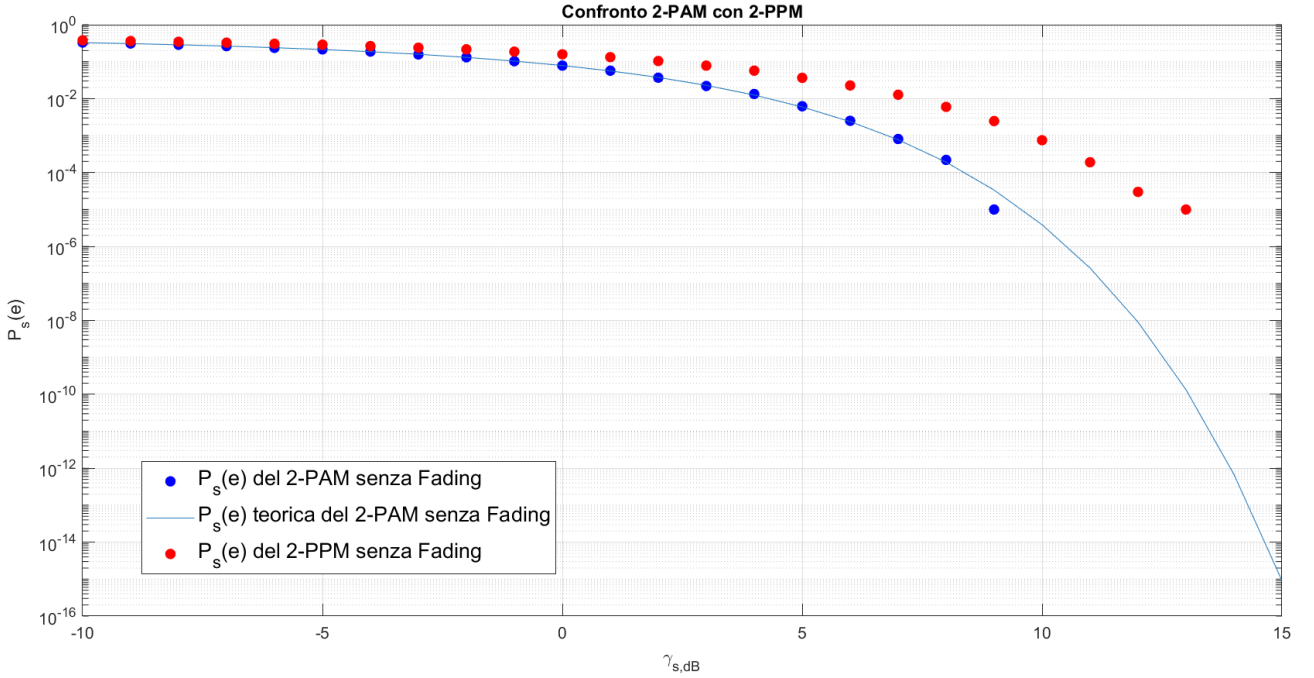


Si noti che i segnali ricevuti si trovano vicino a quelli teorici, grazie all' SNR_{dB} alto.

2 Confronto PAM - PPM

2.1 2-PAM con 2-PPM

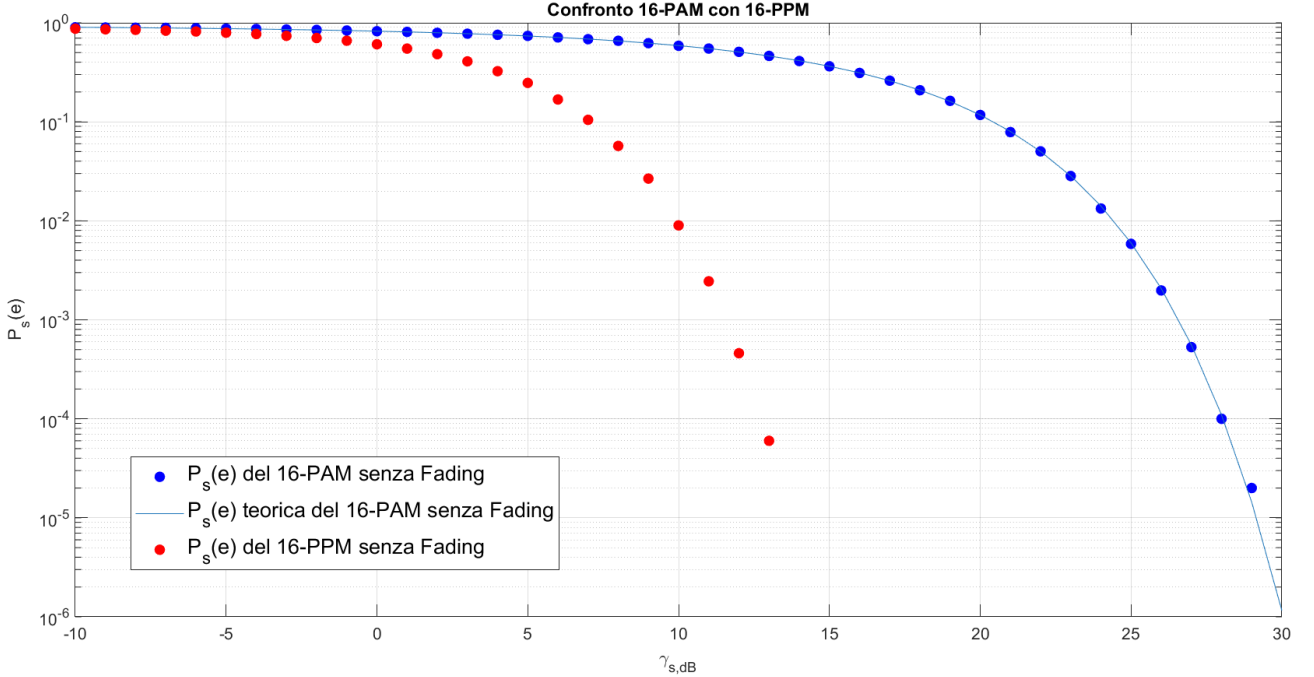
Simulazione delle modulazioni 2-PAM e 2-PPM con $SNR_{dB}=[-10:15]$, 10^5 prove MonteCarlo.



In questa prima simulazione la curva blu, relativa alle prestazioni del 2-PAM, decresce più velocemente rispetto alla curva rossa del 2-PPM, al crescere dell' SNR_{dB} . Questo dimostra che per il numero di bit scelto ($k=2$), il PAM risulta più prestante del PPM.

2.2 16-PAM con 16-PPM

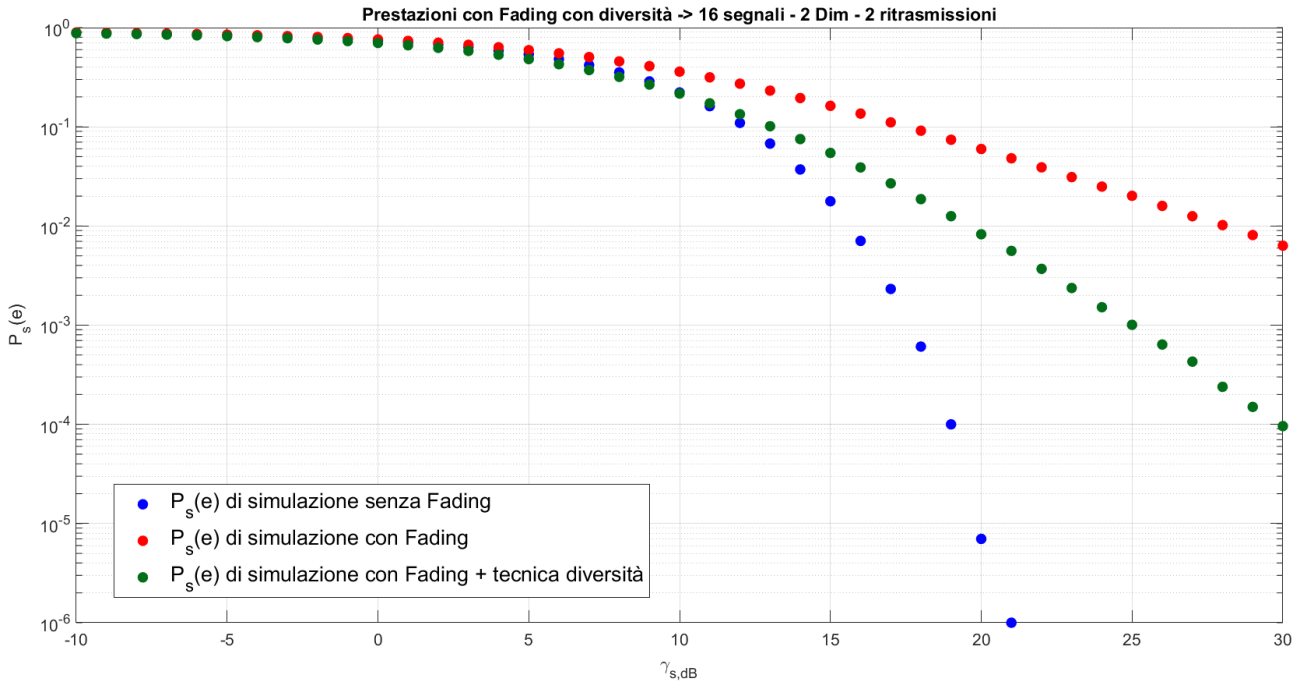
Simulazione delle modulazioni 16-PAM e 16-PPM con $SNR_{dB}=[-10:30]$, 10^5 prove MonteCarlo.



Rispetto alla simulazione precedente, il PPM risulta molto più prestante del PAM per il numero di bit scelto ($k=4$). Se si continuasse ad aumentare il numero k di bit in ulteriori simulazioni, si noterebbe che le prestazioni del PAM continuerebbero a peggiorare, mentre quelle del PPM a migliorare.

3 Prestazioni 16-QAM

Simulazione di una modulazione 16-QAM con $SNR_{dB}=[-10:30]$, 10^6 prove MonteCarlo e $L=2$ ritrasmissioni



3.1 Senza Fading

La curva azzurra mostra come la $P_s(e)$ decresce esponenzialmente all'aumentare dell'SNR considerato.

3.2 Con Fading

La curva rossa mostra come la $P_s(e)$ decresce iperbolicamente (più lentamente) all'aumentare dell'SNR.

Si noti come le prestazioni risultano pessime rispetto al caso in cui il Fading è assente.

3.3 Con Tecnica di Diversità in tempo/frequenza

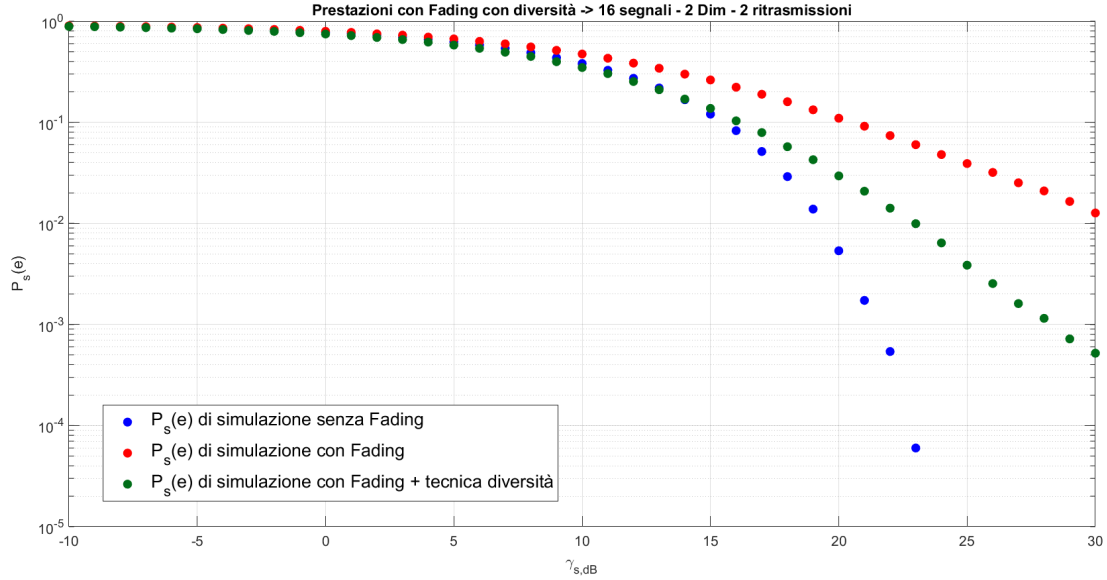
La curva verde mostra come la $P_s(e)$ decresce in maniera più veloce rispetto a quella in presenza di Fading.

Utilizzando $L=2$ ritrasmissioni si riesce a fronteggiare il Fading, ottenendo delle prestazioni tollerabili.

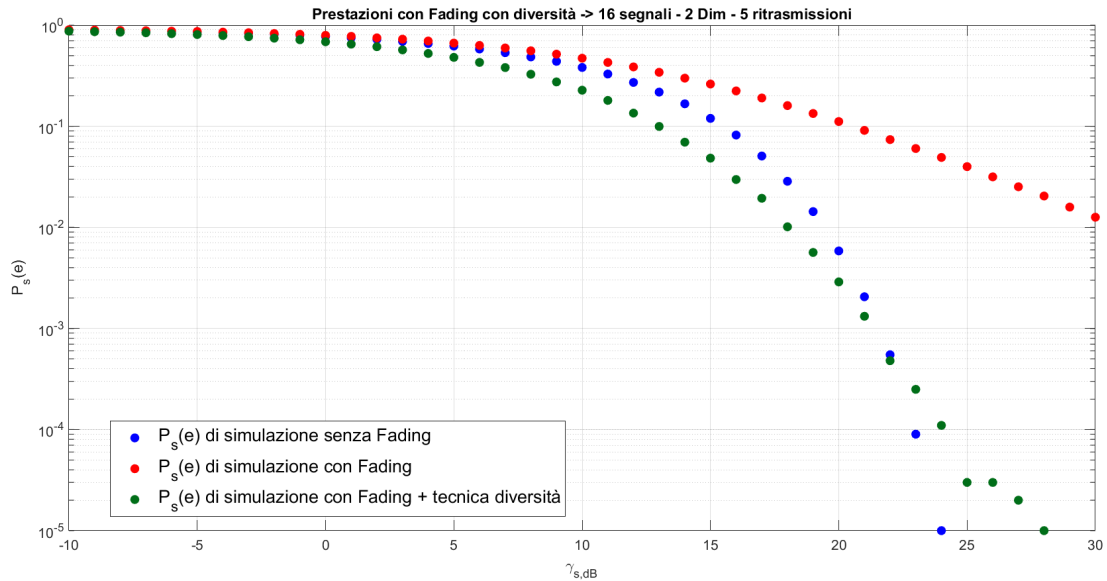
4 Prestazioni 16-PSK al variare di L

Simulazione di una modulazione 16-PSK con $SNR_{dB}=[-10:30]$, 10^5 prove MonteCarlo e ripetuta due volte: la prima con un numero L di ritrasmissioni pari a 2 e la seconda con L=5.

4.1 L=2



4.2 L=5



Confrontando le curve verdi e blu dei 2 grafici è possibile notare che all'aumentare delle ritrasmissioni (da L=2 a L=5) le prestazioni migliorano in maniera significativa.

Entrando nel dettaglio, all'aumentare dell'SNR, la $P_s(e)$ nel primo grafico (L=2) segue inizialmente quella del caso senza Fading per poi allontanarsi da quest'ultima, rimanendo comunque al di sotto di quella con il Fading.

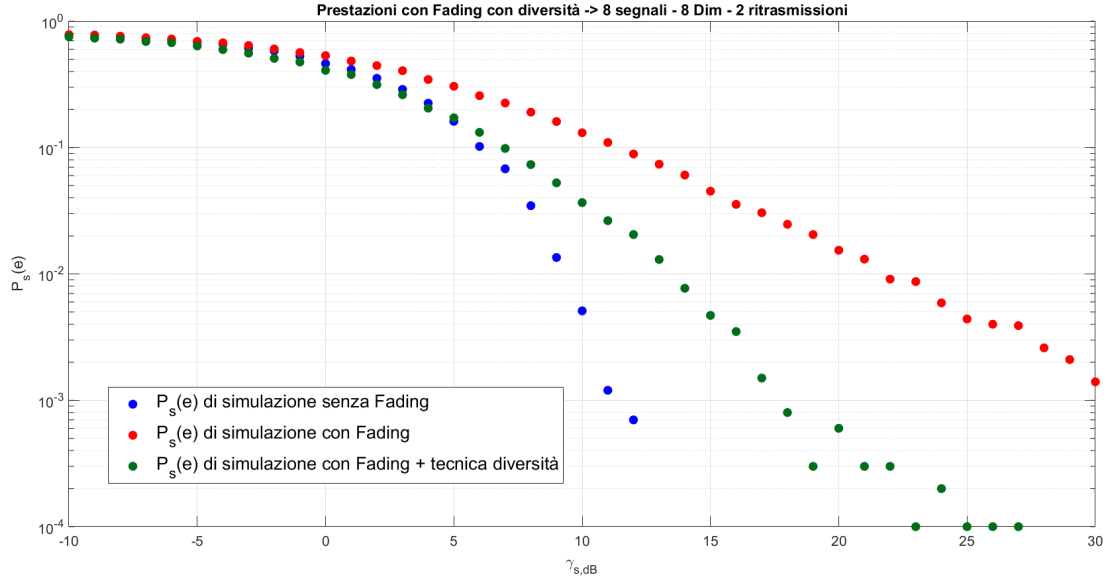
Questo è dovuto al fatto che la tecnica di diversità scelta faccia selezionare tra le L=2 ritrasmissioni quella con maggiore SNR.

Nel secondo grafico (L=5) si può notare che la $P_s(e)$ si trova addirittura al di sotto di quella del caso in cui il fenomeno del Fading non è considerato; sinonimo del fatto che le prestazioni sono migliorate ancor di più.

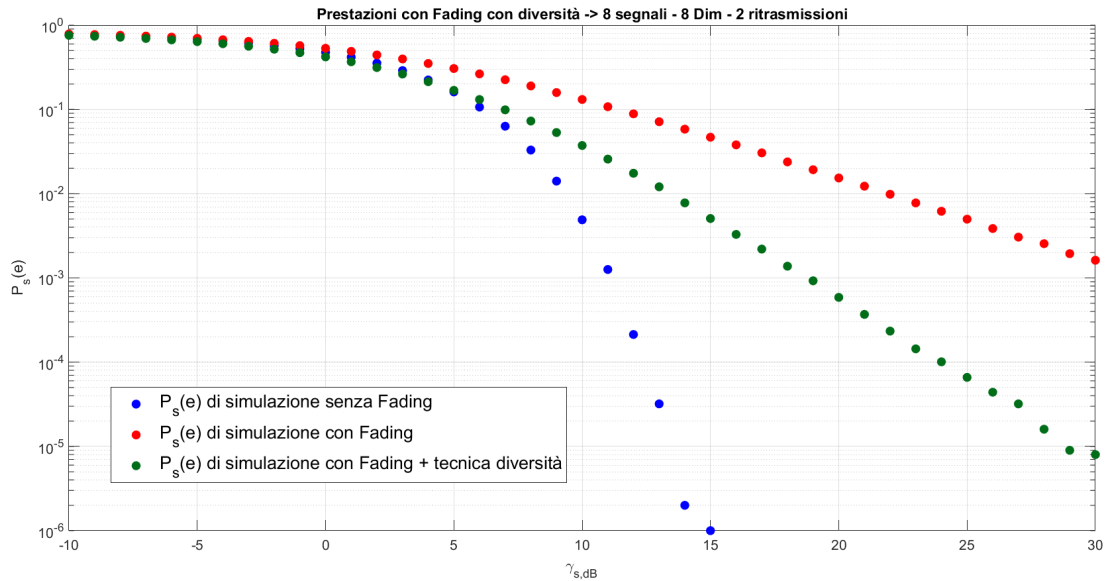
5 Prestazioni 8-PPM al variare di MC

Simulazione di una modulazione 8-PPM con $SNR_{dB}=[-10:30]$, $L=2$ e ripetuta due volte: la prima con un numero MonteCarlo di prove pari a 10^4 e la seconda con $MC=10^6$.

5.1 $MC=10^4$



5.2 $MC=10^6$



Osservando la prima figura ($MC=10^4$), la curva azzurra non presenta stime precise della $P_s(e)$ per valori di SNR_{dB} più alti, in particolare da $SNR_{dB}=10$ fino a 30.

Questo è dovuto al numero insufficiente di prove MC, sulle quali è stata stimata una $P_s(e)=0$ per gli SNR_{dB} alti. In questo caso la $P_s(e)$ non si può mostrare perchè è in scala logaritmica.

Invece per la curva rossa vengono mostrate le stime della $P_s(e)$ per ogni SNR_{dB} perchè, in presenza del Fading (SNR come variabile aleatoria esponenziale), l'SNR effettivo è minore e questo facilita la stima della $P_s(e)$, perchè vi sono più errori.

Osservando la seconda figura ($MC=10^6$), la curva azzurra presenta stime più precise della $P_s(e)$, fornendone una fino a $SNR_{dB}=15$. Dopo questo valore, anche in questo caso, non è possibile ottenere stime della $P_s(e)$ perchè il numero MC di prove risulta ancora insufficiente.

6 Resoconto finale

Per effettuare un'analisi complessiva delle simulazioni, si evince che un aumento del rapporto segnale-rumore (SNR) nelle trasmissioni conduce a un miglioramento delle prestazioni.

Tuttavia, tale incremento risulta insufficiente per ottenere prestazioni accettabili in presenza di Fading, rendendo pertanto necessaria l'introduzione di almeno una tecnica di diversità. Quella implementata nelle simulazioni prevede la ritrasmissione dello stesso segnale, consentendo di migliorare le prestazioni anche in condizioni di Fading.

Appendice: Codice Matlab

A Main

```
1      function proj_main(Str, k, SNRdB, MC, flag, L)
2      %% parametri In
3      % --INPUT--
4      % Str:   e' la modulazione da utilizzare
5      % k:     numero di bit da trasmettere
6      % SNRdB: Rapporto segnale rumore per decibel per simbolo
7      % MC:    numero MonteCarlo di trasmissioni per ogni SNR
8      % flag:  a 1 per simulare il Fading
9      % L:     numero di ritrasmissioni per la tecnica di diversita' ...
10             (flag=1)
11
12      %% Controllo
13      if k==0
14          fprintf('k deve essere maggiore di 0\n')
15          return;
16      end
17
18      %% Scelta modulazione
19      if strcmpi(Str, 'PAM')
20          Cost = proj_PAM_generator(k);
21      elseif strcmpi(Str, 'PPM')
22          Cost = proj_PPM_generator(k);
23      elseif strcmpi(Str, 'PSK')
24          Cost = proj_PSK_generator(k);
25      elseif strcmpi(Str, 'QAM')
26          Cost = proj_QAM_generator(k);
27      else
28          fprintf('Str deve essere PAM / PPM / PSK / QAM \n')
29          return;
30      end
31
32      %% Simulazione con o senza Fading
33      proj_estimate.Pe(SNRdB, Cost, MC);
34      if flag
35          proj_estimate.Pe_Fading(SNRdB, Cost, MC);
36      if L>1
37          proj_estimate.Pe_diversity(SNRdB, Cost, MC, L);
38      end
39      end
```

B Generatore costellazioni

B.1 Generatore PAM

```
1      function Cost = proj_PAM_generator(k)
2      %% parametri In-Out
3      % --INPUT--
4      % k:      numero di bit da trasmettere
5
6      % --OUTPUT--
7      % Cost:   Costellazione dei segnali in forma matriciale -> Mx1 (PAM)
8
9      %% calcolo matrice Cost
10     M = 2^k;
11     Am = linspace(-(M-1), M-1, M);
12     Eg = 3/(M^2-1);
13     Cost = (Am*sqrt(Eg))';
14
15     %% stampa costellazione PAM
16     plot(Cost,zeros(1,M),'ko-','MarkerSize',6,'MarkerFaceColor','k')
17     hold on
18     title(M+'-PAM')
19     grid on
```

B.2 Generatore QAM

```
1      function Cost = proj_QAM_generator(k)
2      %% parametri In-Out
3      % --INPUT--
4      % k:      numero di bit da trasmettere
5
6      % --OUTPUT--
7      % Cost:   Costellazione dei segnali in forma matriciale -> Mx2 (QAM)
8
9      %% calcolo matrice Cost
10     M = 2^k;
11     qam_symbols = qammod(0:M-1, M);
12     Cost(:,1) = real(qam_symbols);
13     Cost(:,2) = imag(qam_symbols);
14
15     Eav = sum(Cost(:,1).^2+Cost(:,2).^2)/M;
16     Cost = Cost./sqrt(Eav);
17
18     %% stampa costellazione PSK
19     plot(Cost(:,1),Cost(:,2),'ko','MarkerSize',6,'MarkerFaceColor','k')
20     hold on
21     plot([-1.5 1.5],[0 0],'k-','MarkerSize',6,'MarkerFaceColor','k')
22     plot([0 0],[-1.5 1.5],'k-','MarkerSize',6,'MarkerFaceColor','k')
23     title(M+'-QAM')
24     grid on
```


B.3 Generatore PSK

```
1      function Cost = proj_PSK_generator(k)
2      %% parametri In-Out
3      % --INPUT--
4      % k:      numero di bit da trasmettere
5
6      % --OUTPUT--
7      % Cost:   Costellazione dei segnali in forma matriciale -> Mx2 (PSK)
8
9      %% calcolo matrice Cost
10     M = 2^k;
11     D = M/2;
12     Cost = zeros(M,2);
13
14     % Avendo posto Eav=1 i segnali sono equienergetici: Es=1 per ogni ...
15     % m --> r=1
16     % quindi gli M segnali sono equidistanti dal centro della ...
17     % circonferenza (r=1)
18     % sx ed sy sono le coordinate dei singoli segnali su x e su y e ...
19     % quindi
20     % gli elementi di ogni riga della costellazione Cost.
21     for ii=1:M
22         sx = cos(ii*pi/D);
23         sy = sin(ii*pi/D);
24         Cost(ii,1) = sx;
25         Cost(ii,2) = sy;
26     end
27
28     %% stampa costellazione PSK
29     plot(Cost(:,1),Cost(:,2),'ko','MarkerSize',6,'MarkerFaceColor','k')
30     hold on
31     plot([-1.5 1.5],[0 0],'k-','MarkerSize',6,'MarkerFaceColor','k')
32     plot([0 0],[-1.5 1.5],'k-','MarkerSize',6,'MarkerFaceColor','k')
33     title(M+'-PSK')
34     grid on
35     axis('square')
```

B.4 Generatore PPM

```
1      function Cost = proj_PPM_generator(k)
2      %% parametri In-Out
3      % --INPUT--
4      % k:      numero di bit da trasmettere
5
6      % --OUTPUT--
7      % Cost:   Costellazione dei segnali in forma matriciale -> MxM (PPM)
8
9      %% calcolo matrice Cost
10     %tutti gli M segnali sono equienergetici e Es = 1 per ogni m
11     %quindi Eav = = Esm = 1
12     M = 2^k;
13     Cost = eye(M);
```

C Stima $P_s(e)$

C.1 $P_s(e)$ senza Fading

```
1      function Pe_s = proj_estimate_Pe(SNRdB, Cost, MC)
2      %% parametri In-Out
3      % --INPUT--
4      % SNRdB: Rapporto segnale rumore per decibel per simbolo
5      % Cost:  M righe = M regnali, N colonne = dimensionalita'
6      % MC:    numero MonteCarlo di trasmissioni per ogni SNR
7
8      % --OUTPUT--
9      % Pe_s:  Probabilita' di errore per simbolo su MC prove e per ...
10             diversi SNR
11
12      %% parametri utili
13      SNRnf = 10.^(SNRdB/10); % SNR no Fading per SIMBOLO
14      Eav = 1;
15      N0 = Eav./SNRnf; % varia N0
16      M = length(Cost(:,1));
17      N = length(Cost(1,:));
18      Pe_s = zeros(1,length(SNRnf));
19      %% calcolo Ps(e)
20      for ii=1:length(SNRnf)
21          N0_now = N0(ii);
22          errori = zeros(1,MC);
23          for jj=1:MC
24              indexTx = randi(M);
25              % scegliamo una delle M righe = una degli M segnali
26              s = Cost(indexTx,:);
27              r = s + randn(1,N)*sqrt(N0_now/2);
28
29              % plot del segnale ricevuto
30              % if N==1
31              %     plot(r,0,'ro','MarkerSize',3);
32              % elseif N==2
33              %     plot(r(1),r(2),'ro','MarkerSize',3);
34              % end
35
36              % calcolo della minima distanza
37              dmin = norm(r - Cost(1,:));
38              indexRx = 1;
39              for zz=2:M
40                  dtmp = norm(r - Cost(zz,:));
41                  if(dtmp < dmin)
42                      dmin = dtmp;
43                      indexRx = zz;
44                  end
45              end
46
47              errori(jj) = indexTx~=indexRx;
48          end
49          Pe_s(ii) = mean(errori);
50      end
51
52      %% Stampa
53      figure
54      semilogy(SNRdB, Pe_s, 'bo', 'MarkerSize', 6, 'MarkerFaceColor', 'b')
55
56      hold on
```

```

56     title('Prestazioni Modulazione - '+M+' segnali - '+N+' Dim')
57     xlabel('\gamma_{s,dB}')
58     ylabel('P_s(e)')
59     lgd = legend('P_s(e) di simulazione senza Fading');
60
61     if N==1 % la modulazione e' il PAM
62     % non mettiamo log2(M) perche' SNR e' gia' per simbolo
63     Pe_s_th = 2*(M-1)/M * qfunc(sqrt(6/(M^2-1)*SNRnf));
64     semilogy(SNRdB, Pe_s_th, 'Color','#1f77ba');
65     lgd = legend('P_s(e) di simulazione senza Fading', ...
66     'P_s(e) teorica senza Fading');
67     end
68     lgd.FontSize = 13;
69     grid on

```

C.2 $P_s(e)$ con Fading

```

1     function Pe_s = proj_estimate_Pe_Fading(SNRdB, Cost, MC)
2     %% parametri In-Out
3     % --INPUT--
4     % SNRdB: Rapporto segnale rumore per decibel per simbolo
5     % Cost:  M righe = M regnali, N colonne = dimensionalita'
6     % MC:    numero MonteCarlo di trasmissioni per ogni SNR
7
8     % --OUTPUT--
9     % Pe_s:  Probabilita' di errore per simbolo su MC prove e per ...
10            diversi SNR
11
12     %% parametri utili
13     SNRnom = 10.^(SNRdB/10);
14     Eav = 1;
15     M = length(Cost(:,1));
16     N = length(Cost(1,:));
17     Pe_s = zeros(1,length(SNRnom));
18     %% calcolo Ps(e)
19     for ii=1:length(SNRnom)
20         errori = zeros(1,MC);
21         for jj=1:MC
22             SNRrv = myexprnd(SNRnom(ii),1,1);
23             NO_now = Eav/SNRrv;
24
25             indexTx = randi(M);
26             s = Cost(indexTx,:);
27             r = s + randn(1,N)*sqrt(NO_now/2);
28
29             d_min = norm(r - Cost(1,:));
30             indexRx = 1;
31             for zz=2:M
32                 d_tmp = norm(r - Cost(zz,:));
33                 if(d_tmp < d_min)
34                     d_min = d_tmp;
35                     indexRx = zz;
36                 end
37             end
38             errori(jj) = indexTx~=indexRx;
39         end
40         Pe_s(ii) = mean(errori);
41     end

```

```

42
43 %% Stampa
44 semilogy(SNRdB, Pe_s, 'ro', 'MarkerSize', 6, 'MarkerFaceColor', 'r')
45
46 title('Prestazioni con Fading '+M+' segnali - '+N+' Dim')
47 lgd = legend('P_s(e) di simulazione senza Fading', ...
48 'P_s(e) di simulazione con Fading');
49
50 if N==1
51 % Calcolo Pe teorica con Fading per il PAM
52 Pe_s_th_F = zeros(1,length(SNRnom));
53 for ii=1:length(SNRnom)
54 Pe_s_th_F(ii) = (M-1)/M * (1-sqrt(1/(1+(M^2-1)/(3*SNRnom(ii)))) );
55 end
56 semilogy(SNRdB, Pe_s_th_F, 'Color', '#FF6506')
57 lgd = legend('P_s(e) di simulazione senza Fading', ...
58 'P_s(e) teorica senza Fading', ...
59 'P_s(e) di simulazione con Fading', ...
60 'P_s(e) teorica con Fading');
61 end
62 lgd.FontSize = 13;

```

C.3 $P_s(e)$ con Fading e tecnica diversità

```

1 function Pe_s = proj_estimate_Pe_diversity(SNRdB, Cost, MC, L)
2 %% parametri In-Out
3 % --INPUT--
4 % SNRdB: Rapporto segnale rumore per decibel per simbolo
5 % Cost: M righe = M regnali, N colonne = dimensionalita'
6 % MC: numero MonteCarlo di trasmissioni per ogni SNR
7 % L: numero di ritrasmissioni per la tecnica di diversita'
8
9 % --OUTPUT--
10 % Pe_s: Probabilita' di errore per simbolo su MC prove e per ...
11 % diversi SNR
12 %% parametri utili
13 SNRnom = 10.^(SNRdB/10);
14 Eav = 1;
15 M = length(Cost(:,1));
16 N = length(Cost(1,:));
17 Pe_s = zeros(1,length(SNRnom));
18 %% calcolo Ps(e)
19 for ii=1:length(SNRnom)
20 errori = zeros(1,MC);
21 for jj=1:MC
22 indexTx = randi(M);
23 s = Cost(indexTx,:);
24
25 SNRrv = myexprnd(SNRnom(ii),1,L);
26 NO_now = Eav/max(SNRrv);
27 r = s + randn(1,N)*sqrt(NO_now/2);
28
29 d_min = norm(r - Cost(1,:));
30 indexRx = 1;
31 for zz=2:M
32 d_tmp = norm(r - Cost(zz,:));
33 if(d_tmp < d_min)
34 d_min = d_tmp;

```

```

35     indexRx = zz;
36     end
37     end
38
39     errori(jj) = indexTx≠indexRx;
40     end
41     Pe_s(ii) = mean(errori);
42     end
43
44     %% Stampa
45     semilogy(SNRdB, Pe_s, 'o','Color', '#006e18','MarkerSize', 6, ...
46             'MarkerFaceColor', '#006e18')
47
48     title('Prestazioni con Fading con diversita' -> '+M+' segnali - ...
49           '+N+' Dim - '+L+' ritrasmissioni')
50     lgd = legend('P_s(e) di simulazione senza Fading', ...
51                 'P_s(e) di simulazione con Fading', ...
52                 'P_s(e) di simulazione con Fading + tecnica diversita');
53
54     if N==1
55         lgd = legend('P_s(e) di simulazione senza Fading', ...
56                     'P_s(e) teorica senza Fading', ...
57                     'P_s(e) di simulazione con Fading', ...
58                     'P_s(e) teorica con Fading', ...
59                     'P_s(e) di simulazione con Fading + tecnica diversita');
60     end
61     lgd.FontSize = 13;

```

C.3.1 myexprnd ausiliaria

```

1     function y=myexprnd(mu,r,c)
2     y=-mu*log(rand(r,c));

```