Read pp. 14-17!

CM PERFORMANCE WITH RS CODES

I. Introduction

Scenario:

- AWGN
- RS codes
- Multistage decoding

- Hard decision[1] bounded-distance[2] decoding of each RS code independently
- Optimize SNR ($\gamma = E_s/N_0$) for a given total rate R and block error rate[3] (BLER = P_E)
- Analysis allowing real-valued code rates[4]
- MLCM[5]
- Equiprobable, independent input bits
- Ungerboeck set partitioning

Options:

1. BLER $P_E = 10^{-5}$, 10^{-10}, or asymptotic $P_E \to 0$
2. Total rate $R = 0.9\ell$, 0.7ℓ, or 0.5ℓ
3. Constellations 4-PAM, 16-QAM, 8-PSK, 8-DPSK, or 256-QAM[2]
4. Subrates $R_1, \dots, R_L$ optimized for actual P_E, mismatched P_E, or balanced distance rule (BDR)
5. Calculate channel bit error probability p exactly[6], by Q-function union bound (UB), or by better UB[7]
6. Calculate P_E for RS codes exactly[8] or by binary upper bound[9]

Notation:

q = number of bits per RS symbol

(n, k_i, δ_i) = q-ary code parameters for codes $i = 0, \dots, \ell-1$ (same $n = 2^q - 1$)

d_i = Euclidean minimum distance between two subsets at level $i = 0, \dots, \ell-1$

$R_i = \frac{k_i}{n}$ = rate of code i

$M = 2^\ell$ = constellation size

$t_i = \left\lfloor \frac{\delta_i - 1}{2} \right\rfloor$ = error-correcting capability

p_i = channel bit error probability (before decoding) on level $i = 0, \dots, \ell-1$

$R = \sum_{i=0}^{\ell-1} R_i$ = total rate

P_{E_i} = BLER in level $i = 0, \dots, \ell-1$

$\gamma = E_s/N_0$

E_s = average symbol energy of the <u>whole</u> constellation (never subsets)

$N_0/2$ = double-sided noise PSD

$P_E = 1 - \prod_{i=0}^{\ell-1} (1 - P_{E_i})$ = total BLER

Notes to page 1:

1 An extension to soft decision is possible, Difficulties: How do soft decision algorithms for RS work? Do they operate on bits (how?) or on q-ary symbols (what is a proper q-ary metric?)? Do results depend on bit-to-symbol mapping (labeling)?

2 All codewords with $\leq t_i$ symbol errors are corrected and no codewords with $> t_i$ errors.

3 An extension to BER is possible. Difficulties: Dependence on labeling. Error propagation in multistage decoder.

4 This simplifies optimization but is not realistic. Round-off effects must be considered when comparing analysis and simulations.

5 Future work: add an interleaver and obtain similar results for BICM, still without iterative decoding

6 Available for PAM, QAM, PSK [Proakis], DPSK [Pawula] but not odd levels of QAM^2.

7 [Hughes, TIT 1991]

8 [Ebel, TCOM 1995]

9 $$P_{Ei} \leq \sum_{j=t_i+1}^{qn} \binom{qn}{j} p_i^j (1-p_i)^{qn-j} \quad (1)$$

II. Asymptotic coding gain

Options:

1. $P_E \to 0$
2. Any rate
3. Constellations 4-PAM, 16-QAM, 8-PSK, 8-DPSK, 64-PSK², and 256-QAM²
4. Optimized for P_E ($\Rightarrow$ BDR)
5. Any p calculation
6. Any P_E calculation

$P_E \to 0$ means $\gamma \to \infty$ or, equivalently, $N_0 \to 0$. For any constellation, there exist constants A_1 and A_2 such that for sufficiently small N_0,

$$A_1 Q\left(\sqrt{\frac{d_i^2}{2N_0}}\right) \leq p_i \leq A_2 Q\left(\sqrt{\frac{d_i^2}{2N_0}}\right) \qquad (2)$$

where the upper bound is the Q-function UB.

Similarly, both exact[8] or approximate[9] P_E calculation yield for sufficiently small p_i:

$$B_1 p_i^{t_i+1} \leq P_{Ei} \leq B_2 p_i^{t_i+1} \qquad (3)$$

for some constants B_1 and B_2.

Defining ε_{1i} implicitly by

$$P_{E_i} = Q\left(\sqrt{\frac{d_i^2}{2N_0} + \varepsilon_{1i}}\right)^{t_i+1} \qquad (4)$$

and observing that $\frac{d}{dx}\left[\log Q(\sqrt{x})\right] \to -\infty$ as $x \to \infty$, (2) − (3) show that $\varepsilon_{1i} \to 0$ as $N_0 \to 0$, regardless of A_1, A_2, B_1, B_2.

$\boxed{\text{LEMMA 1}}$ For any $t > 0$,

$$\lim_{x\to\infty} \frac{\left[x\sqrt{2\pi}\, Q(x)\right]^t}{x\sqrt{2\pi t}\, Q(x\sqrt{t})} = 1$$

PROOF OUTLINE: Follows from

$$\frac{x}{1+x^2}\frac{1}{\sqrt{2\pi}} e^{-x^2/2} < Q(x) < \frac{1}{x}\frac{1}{\sqrt{2\pi}} e^{-x^2/2} \qquad (6)$$

for all $x > 0$. [Wikipedia, 2009] $\square$

The lemma allows us to rewrite (4) as

$$P_{E_i} = Q\left(\sqrt{\frac{(t_i+1)\, d_i^2}{2N_0} + \varepsilon_{2i}}\right)$$

where again $\varepsilon_{2i} \to 0$ as $N_0 \to 0$.

We wish to minimize

$$P_E = 1 - \prod_i (1 - P_{Ei})$$

$$= \sum_{i=0}^{l-1} P_{Ei} - \sum_{i=0}^{l-1} \sum_{\substack{j=0 \\ j \neq i}}^{l-1} P_{Ei} P_{Ej} + \ldots$$

which asymptotically, when $N_0 \to 0$, approaches

$$\overline{P_E} = \sum_{i=0}^{l-1} Q\left(\sqrt{\frac{(t_i+1)d_i^2}{2N_0}}\right) \qquad (9)$$

For RS codes,

$$\delta_i = n - k_i + 1$$

$$\Rightarrow t_i \doteq \frac{n-k_i}{2} = \frac{n}{2}(1-R_i),$$

ignoring the floor function $\lfloor \cdot \rfloor$ because of the 6th bullet on p.1.

The overall constraint is a constant

$$R = \sum_{i=0}^{l-1} R_i$$

$$\Rightarrow \sum_{i=0}^{l-1} t_i = \frac{nl}{2} - \frac{n}{2} \sum R_i$$

$$= \frac{n(l-R)}{2} \qquad (13)$$

$\boxed{\text{LEMMA 2}}$ For any $a_1>0$, $a_2>0$, and $\gamma>0$, define $\hat{x}_1$ and $\hat{x}_2$ as the values of x_1 and x_2 that minimize

$$f(x_1,x_2) = Q(\sqrt{a_1x_1 z}) + Q(\sqrt{a_2x_2 z})$$

subject to $x_1+x_2=1$. Then

$$\lim_{z\to\infty} \hat{x}_1 = \frac{a_2}{a_1+a_2}$$

$$\lim_{z\to\infty} \hat{x}_2 = \frac{a_1}{a_1+a_2}$$

PROOF: Let $g(x)=f(x,1-x)$. The minimum $g(x)$ occurs where

$$0 = g'(x)$$

$$= Q'(\sqrt{a_1 x z})\cdot\frac{a_1 z}{2\sqrt{a_1 x z}} - Q'(\sqrt{a_2(1-x)z})\frac{a_2 z}{2\sqrt{a_2(1-x)z}}$$

$$= -\frac{1}{\sqrt{2\pi}}e^{-a_1xz/2}\frac{a_1 z}{2\sqrt{a_1xz}} + \frac{1}{\sqrt{2\pi}}e^{-a_2(1-x)z/2}\frac{a_2 z}{2\sqrt{a_2(1-x)z}}$$

$$\Leftrightarrow e^{-a_1xz/2}\sqrt{\frac{a_1}{x}} = e^{-a_2(1-x)z/2}\sqrt{\frac{a_2}{1-x}}$$

$$\Leftrightarrow \sqrt{\frac{a_1}{a_2}\left(\frac{1-x}{x}\right)} = e^{[(a_1+a_2)x-a_2]z/2}$$

$$\Leftrightarrow x = \frac{1}{a_1+a_2}\left(a_2+\frac{1}{z}\log\frac{a_1}{a_2}\left(\frac{1}{x}-1\right)\right) \tag{20}$$

Assume without loss of generality that $a_1 \geq a_2$. Then for any γ,

$$\frac{a_2}{a_1+a_2} \leq x \leq \frac{a_1}{a_1+a_2} \tag{21}$$

because if $x > \frac{a_1}{a_1+a_2}$, then

$$\frac{a_1}{a_2}\left(\frac{1}{x}-1\right) < \frac{a_1}{a_2}\left(\frac{a_1+a_2}{a_1}-1\right)$$
$$= 1$$

and the r.h.s. of (20) is $< \frac{a_2}{a_1+a_2}$,
which is a contradiction. Similarly
if $x < \frac{a_2}{a_1+a_2}$, then

$$\frac{a_1}{a_2}\left(\frac{1}{x}-1\right) > \frac{a_1}{a_2}\left(\frac{a_1+a_2}{a_2}-1\right)$$
$$= \frac{a_1^2}{a_2^2}$$
$$\geqslant 1$$

and the r.h.s of (20) is $> \frac{a_2}{a_1+a_2}$,
which is another contradiction. Hence
x is bounded as (20) for any $z > 0$.

Now let $z \to \infty$. Since x is bounded,
the r.h.s. of (20) converges to $\frac{a_2}{a_1+a_2}$
and the lemma follows by $\hat{x}_1 = x$ and
$\hat{x}_2 = 1-x$.

□

$\boxed{\text{COROLLARY 3}}$ Applying LEMMA 2 recursively
proves that the minimum of

$$\sum_{i=0}^{\ell-1} Q\left(\sqrt{a_i x_i z}\right)$$

subject to $\sum_i x_i = 1$ satisfies

$$a_0 x_0 = a_1 x_1 = \ldots = a_{\ell-1} x_{\ell-1} \qquad (25)$$

asymptotically as $z \to \infty$.

COROLLARY 4 Subsituting $x_i = x_i'/A \ \forall i$ and $z = Az'$ demonstrates that for any constant $A > 0$, the minimum of

$$\sum_{i=0}^{l-1} Q(\sqrt{a_i x_i' z'})$$

subject to $\sum_i x_i' = A$ satisfies

$$a_0 x_0' = a_1 x_1' = \dots = a_{l-1} x_{l-1}' \qquad (27)$$

asymptotically as $z' \to \infty$.

The solution of (27) and $\sum_i x_i' = A$ is $x_i' = \frac{c}{a_i}, \ \forall i,$ where $c = A / \sum_i \frac{1}{a_i}$.

We apply Corollary 4 to (9) and (13), where

$$x_i' = t_i + 1$$
$$a_i = d_i^2$$
$$z' = \frac{1}{2N_0}$$
$$A = \frac{n(l-R)}{2} + l$$

Thus the asymptotic optimum occurs for

$$t_i + 1 = \frac{c}{d_i^2} \qquad i = 0, \dots, l-1 \qquad (32)$$

where

$$c = \frac{n(l-R) + 2l}{2 \sum_i d_i^{-2}}. \qquad (33)$$

As expected, (32) is the BDR.

Combining (32) and (9) yields

$$\overline{P}_E = \sum_{i=0}^{\ell-1} Q\left(\sqrt{\frac{c}{2N_0}}\right)$$

$$= \ell Q\left(\sqrt{\frac{c}{2N_0}}\right) = \ell\, Q\left(\sqrt{\frac{n\ell - nR + 2\ell}{4N_0 \sum_i d_i^{-2}}}\right) \qquad (34)$$

for MLCM at low N_0.

For comparison, we study a system with plain FEC, no MLCM. Then t_i is the same for all i. Still, (13) is satisfied:

$$t_1 = t_2 = \dots = t_\ell = \frac{n}{2}\left(1 - \frac{R}{\ell}\right)$$

Let $\check{d} = \min_i d_i$. Then (9) is, for the FEC case, dominated by the term

$$\overline{P}_{E,FEC} \approx Q\left(\sqrt{\frac{(t_1+1)\check{d}^2}{2N_0}}\right)$$

$$= Q\left(\sqrt{\frac{\left(\frac{n}{2} - \frac{nR}{2\ell} + 1\right)\check{d}^2}{2N_0}}\right)$$

$$= Q\left(\sqrt{\frac{(n\ell - nR + 2\ell)\check{d}^2}{4\ell N_0}}\right)$$

Comparing with (34) shows that the gain of MLCM over FEC in this scenario is

$$ACG = \frac{\ell \check{d}^{-2}}{\sum_i d_i^{-2}} = \frac{\bar{d}^2}{\check{d}^2}$$

where

$$\bar{d} \triangleq \left(\frac{1}{\ell}\sum_i d_i^{-2}\right)^{-1/2}$$

<u>Remark</u> An easier way to prove the same result is by rewriting (6) as

$$Q(x) = e^{-x^2/2 + O(\log x)} \tag{39}$$

Now (2)-(3) implies

$$p_i = e^{-\frac{d_i^2}{4N_0} + O(\log \frac{1}{N_0})}$$

$$P_{E_i} = e^{-\frac{(t_i+1)d_i^2}{4N_0} + O(\log \frac{1}{N_0})}$$

We need no Lemma 1. Lemma 2 has an easier proof using (39):

PROOF OF LEMMA 2:

$$g(x) = e^{-\frac{a_1 x z}{2} + O(\log z)} + e^{-\frac{a_2(1-x)z}{2} + O(\log z)}$$

$$0 = g'(x)$$

$$= -\frac{a_1 z}{2} e^{-\frac{a_1 x z}{2} + O(\log z)} + \frac{a_2 z}{2} e^{-\frac{a_2(1-x)z}{2} + O(\log z)}$$

$$= -e^{-\frac{a_1 x z}{2} + O(\log z)} + e^{-\frac{a_2(1-x)z}{2} + O(\log z)}$$

$$\Rightarrow e^{-\frac{a_1 x z}{2} + O(\log z)} = e^{-\frac{a_2(1-x)z}{2} + O(\log z)}$$

$$\Rightarrow \frac{a_1 x z}{2} + O(\log z) = \frac{a_2(1-x)z}{2} + O(\log z)$$

$$\Rightarrow \frac{a_1 x - a_2(1-x)}{2} z + O(\log z) = 0$$

$$\Rightarrow a_1 x - a_2(1-x) = 0$$

$$\Rightarrow a_1 \hat{x}_1 = a_2 \hat{x}_2$$

and the Lemma follows via $\hat{x}_1 + \hat{x}_2 = 1$. ∎

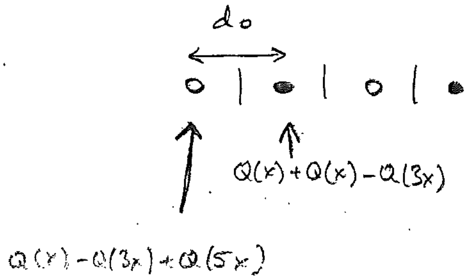
The analysis on this page replaces pp. 4-9.
Continue from p. 10.

III 4-PAM

Options:

1. Varying P_E
2. Varying R
3. 4-PAM
4. Subrates optimized for actual P_E
5. Exact p
6. Exact P_E

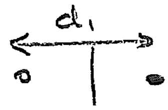
For 4-PAM, there are two levels. Level 0 looks as follows.



where $x=\sqrt{\frac{d_0^2}{2N_0}}$

Thus $p_0 = \frac{1}{2}\left(Q(x)-Q(3x)+Q(5x)\right)+\frac{1}{2}\left(2Q(x)-Q(3x)\right)$
$= \frac{3}{2}Q(x)-Q(3x)+\frac{1}{2}Q(5x)$

Level 1 is simply a binary decision:



Thus $p_1 = Q\left(\sqrt{\frac{d_1^2}{2N_0}}\right)$.

The average symbol energy is

$$E_S = \frac{1}{4}\left(2\left(\frac{d_0}{2}\right)^2+2\left(\frac{3d_0^2}{2}\right)\right)$$

$$=\frac{1}{4}\left(\frac{2d_0^2+18d_0^2}{4}\right)$$

$$=\frac{5}{4}d_0^2 \qquad (49)$$

or $d_0^2 = \frac{4}{5} E_s$

and consequently $d_1^2 = 4 d_0^2 = \frac{16}{5} E_s$.

Thus

$$p_0 = \frac{3}{2} Q\left(\sqrt{\frac{2}{5}\gamma}\right) - Q\left(3\sqrt{\frac{2}{5}\gamma}\right) + \frac{1}{2} Q\left(5\sqrt{\frac{2}{5}\gamma}\right) \qquad (50)$$

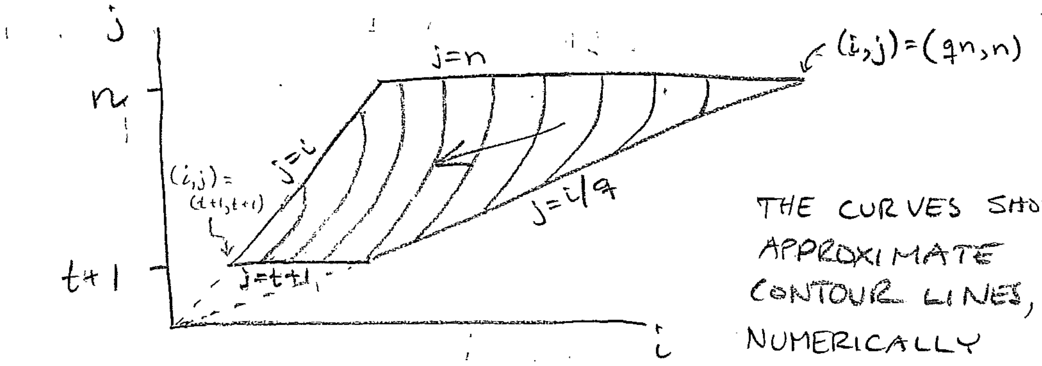
$$p_1 = Q\left(\sqrt{\frac{8}{5}\gamma}\right) \qquad (51)$$

The BLER of an RS code is exactly

$$P_E = \sum_{j=t+1}^{n} \sum_{i=j}^{qj} P(w_j | w_i) P(w_i) \qquad (52)$$

where $P(W_j | w_i)$ and $P(w_i)$ are given in [Ebel 95, (18)-(20)]:

$$P_E = \sum_{j=t+1}^{n} \binom{n}{j} \sum_{i=j}^{qj} p^i (1-p)^{qn-i} \sum_{m=0}^{\lfloor j - i/q \rfloor} (-1)^m \binom{j}{m} \binom{q(j-m)}{i} \qquad (53)$$



THE CURVES SHOW APPROXIMATE CONTOUR LINES, NUMERICALLY OBTAINED, WITH HIGH VALUES TO THE BOTTOM LEFT.

We rewrite (53) as

$$P_E = \sum_{j=t+1}^{n} \sum_{i=j}^{qj} C_{i,j}\, p^i (1-p)^{qn-i} \tag{54}$$

where

$$C_{ij} = \binom{n}{j} \sum_{m=0}^{j-\lceil i/q \rceil} (-1)^m \binom{j}{m} \binom{q(j-m)}{i} \tag{55}$$

These coefficients are integers and do not depend on p or t. They can thus be tabulated offline for given parameters n and q. The calculations can be further simplified using the relation

$$\sum_{j=\lceil \frac{i}{q} \rceil}^{\min(n,i)} C_{i,j} = \binom{qn}{i} \tag{56}$$

for $i = 1, \cdots, qn$

because, with Ebel's definition of $P(W_j|w_i)$,

$$\sum_{j=\lceil \frac{i}{q} \rceil}^{\min(n,i)} P(W_j|w_i) = 1$$

We interchange the order of summation in (54).

$$t+1 \le j \le n \ \wedge \ j \le i \le qj$$

$$\Leftrightarrow \ j_0 \le j \le j_1$$

where

$$\begin{cases} j_0 \triangleq \max\{t+1, \lceil \frac{i}{q} \rceil\} \\ j_1 \triangleq \min\{n, i\} \end{cases}$$

From $j_0 \le j_1$ follows

$$t+1 \le i \le qn \tag{60}$$

Thus (54) becomes

$$P_E = \sum_{i=t+1}^{qn} E_t(i)\, p^i (1-p)^{qn-i} \tag{61}$$

where

$$E_t(i) \triangleq \sum_{j=j_0}^{j_1} c_{ij} \tag{62}$$

depends on t but not p.
From (56) follows

$$\begin{cases} E_t(i) = \binom{qn}{i} & \text{if } i > qt \\ E_t(i) = \binom{qn}{i} - \sum_{j=\lceil \frac{i}{q} \rceil}^{t} c_{ij} & \text{if } i \le qt \end{cases} \tag{63}$$

because $t+1 \le \lceil i/q \rceil \Leftrightarrow i \ge qt+1$.
The final strategy to evaluate P_E is thus:

1. For given n and q, tabulate c_{ij} from (55) offline with high accuracy, for $j = 1, \dots, t_{max}$ and $i = j, \dots, qj$.
2. For a given t, tabulate $E_t(i)$ from (63), for $i = t+1, \dots, qn$.
3. For a given constellation, determine the relation between p and γ as on pp. 13-14.
4. Plot P_E as a function of γ by (61).

In point 1, it is assumed to be known an upper bound on the t's that will ever be considered in item 2. If no such bound is known, use $t_{max} = \lfloor n/2 \rfloor$.

With the definition (62), $E_t(i)$ has the following properties:

$$E_t(i) = 0 \qquad \text{if } 1 \le i \le t$$

$$E_t(i) = \binom{qn}{i} \qquad \text{if } qt < i \le qn$$

Thus it suffices to tabulate $E_t(i)$ for $i = t+1, \dots, qt$.

Returning to the BLER of MLCM systems, the overall BLER of 4-PAM is

$$P_E = 1-(1-P_{E0})(1-P_{E1})$$
$$= P_{E0} + P_{E1} - P_{E0}P_{E1}$$

where P_{E0} and P_{E1} are given by (61) and p_0 and p_1 by (50)-(51).

As a special case, I evaluate P_{EO} for $t=0$ numerically. The result is exactly (within working precision) equal to

$$1-(1-p_{PAM}(\gamma))^{qn} \tag{67}$$

where

$$p_{PAM}(\gamma)=\frac{3}{2}Q\left(\sqrt{\frac{2}{5}\gamma}\right)$$

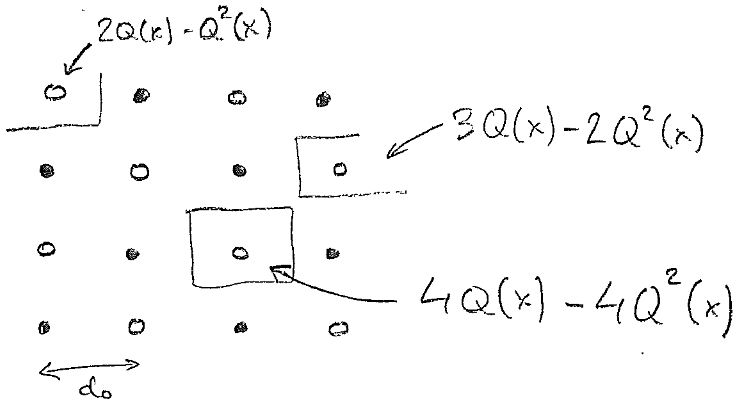
which is reasonable.

IV 16-QAM

1. Varying P_E
2. Varying R
3. 16-QAM
4. Subrates optimized for actual P_E
5. p by Q function UB
6. Exact P_E

For 16-QAM, there are four levels.

<u>Level 0</u> :



On average:

$$p_0 \approx \frac{1}{4}\left(2Q - Q^2\right) + \frac{2}{4}\left(3Q - 2Q^2\right) + \frac{1}{4}\left(4Q - 4Q^2\right)$$

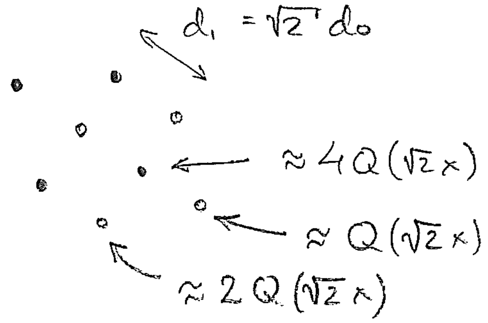
$$= 3Q(x) - \frac{9}{4}Q^2(x) \qquad (69)$$

This is still an <u>approximation</u>, because the possibility of jumping from ∘ to another ∘, which would yield a correct bit decision, has been ignored. This error would partially cancel the $Q^2(x)$ term in (69). It is possible to calculate p_0, p_1, p_2, p_3 exactly, but not now... Thus

$$p_0 \approx 3Q(x)$$

where $x = \sqrt{\frac{d_0^2}{2N_0}}$.

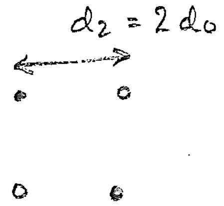
Level 1



$$P_1 \approx \frac{1}{4} Q(\sqrt{2}x) + \frac{2}{4}\cdot 2Q(\sqrt{2}x) + \frac{1}{4}\cdot 4Q(\sqrt{2}x)$$

$$= \frac{9}{4} Q(\sqrt{2}x)$$

Level 2



$$P_2 \approx 2Q(2x)$$

Level 3

$$P_3 \approx Q(2\sqrt{2}x)$$

The average symbol energy is twice that of 4-PAM, see (49):

$$E_s = \frac{5}{2} d_0^2$$

$$\Rightarrow d_0^2 = \frac{2}{5} E_s$$

$$\Rightarrow x = \sqrt{\frac{1}{5}\gamma}$$

The overall BLER of 16-QAM MLCM is now

$$P_E(\gamma) = 1 - \prod_{i=0}^{3} (1 - P_{Ei})$$

$$= \sum_i P_{Ei} - \sum_{\substack{i,j \\ i\neq j}} P_{Ei}P_{Ej} + \sum_{\substack{i,j,k \\ i\neq j\neq k\neq i}} P_{Ei}P_{Ej}P_{Ek} - P_{E0}P_{E1}P_{E2}P_{E3} \qquad (77)$$

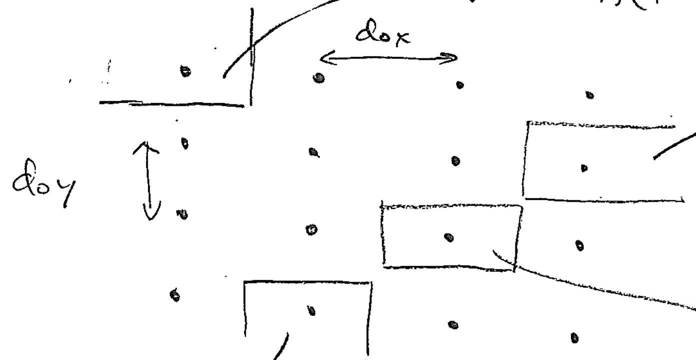
with P_{Ei} given by (61).

V Rectangular 16-QAM

Same scenario as previous, but with an extra parameter α to regulare the relation between the scaling of the two constituent 4-PAM constellations.

Level 0:

$$1-(1-Q(x))(1-Q(y)) = Q(x)+Q(y)-Q(x)Q(y)$$



$$1-(1-2Q(y))(1-Q(x)) = Q(x)+2Q(y)-2Q(x)Q(y)$$

$$1-(1-2Q(x))(1-2Q(y))$$

$$1-(1-2Q(x))(1-Q(y)) = 2Q(x)+Q(y)-2Q(x)Q(y)$$

$$= 2Q(x)+2Q(y)-4Q(x)Q(y)$$

Disregarding higher-order terms $(Q(x)Q(y))$,

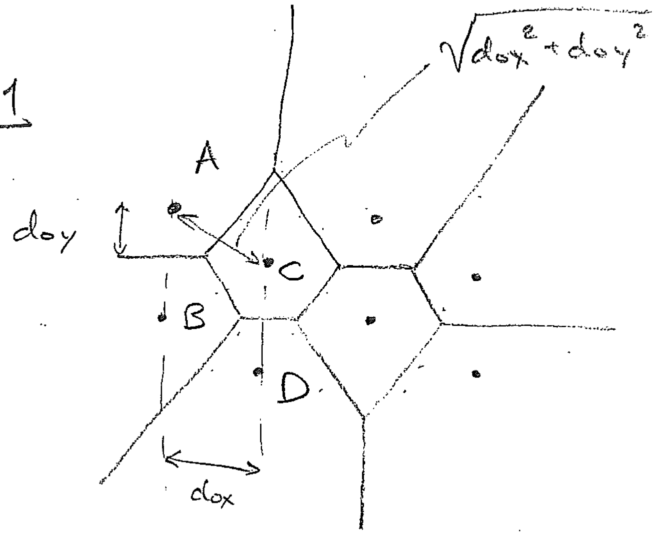
$$p_0 \approx \frac{1}{4}\left(Q(x)+Q(y)+Q(x)+2Q(y)+2Q(x)+2Q(y)+2Q(x)+Q(y)\right)$$

$$= \frac{3}{2}Q(x)+\frac{3}{2}Q(y) \qquad (78)$$

where $x=\sqrt{\frac{d_{0x}^2}{2N_0}}$, $y=\sqrt{\frac{d_{0y}^2}{2N_0}}$

The approximation is simply a UB.

Level 1



Union bound, assuming $d_{oy} < d_{ox}$:

$$p_{1A} \approx Q(2x) + Q(2y) + Q(z)$$

where $z = \sqrt{x^2 + y^2}$

$$p_{1B} \approx Q(2y) + 2Q(z)$$

$$p_{1C} \approx Q(2y) + 4Q(z)$$

$$p_{1D} \approx Q(2x) + Q(2y) + 2Q(z)$$

$$\Rightarrow p_1 \approx \frac{1}{2}Q(2x) + Q(2y) + \frac{9}{4}Q(z) \qquad (84)$$

Level 2: $p_2 \approx Q(2x) + Q(2y)$ (85)

Level 3: $p_3 \approx Q(2z)$ (86)

The average symbol energy is

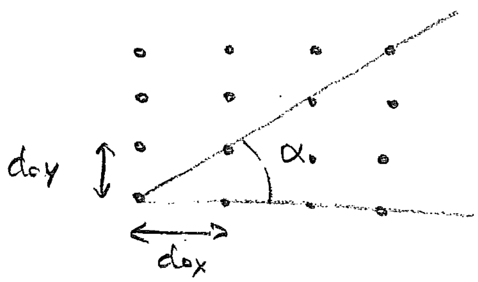
$$E_s = \frac{5}{4}d_{ox}^2 + \frac{5}{4}d_{oy}^2$$

$$= \frac{5}{4} \cdot 2N_0 \cdot (x^2 + y^2)$$

$$= \frac{5N_0}{2} z^2$$

$$\Rightarrow z = \sqrt{\frac{2}{5}\gamma} \qquad (88)$$

Define α as the angle of the diagonal in the rectangle:



Then

$$\begin{cases} x = \sqrt{\frac{2}{5}\gamma}\, \cos\alpha \\ y = \sqrt{\frac{2}{5}\gamma}\, \sin\alpha \end{cases} \qquad (89)$$

The overall BLER is given by (77), with P_{Ei} given by (61), $p_0, \ldots, p_1$ by (78), (84)-(86), and x, y, and z given by (88)-(89). The expressions hold for $0 < \alpha \leq \frac{\pi}{4}$. (For $\frac{\pi}{4} \leq \alpha < \pi$, swap x and y in (84).)

EA Apr. 2007

A 2^m-PSK constellation is used with block-coded modulation (BCM). The m codes are RS codes with length ns symbols (same length for all codes) and dimension ks[[1]],...,ks[[m]] symbols. Coherent detection.

▪ Preliminaries

```
$DefaultFont = {"Times-Roman", 10};
```

```
Off[General::spell];
```

The symbol error probability of coherent M-PSK is proportional to Q(Sqrt[2Es/N0]Sin[Pi/M]), see, e.g., Proakis p. 270. This can be written as Q(del*Sqrt[Es/(2N0)]), where del = 2Sin[Pi/M] is the minimum Euclidean distance in the set, normalized by Es. In a set partitioning of a 2^m-PSK constellation, subset i, for i=1,...,m, corresponds to 2^(m+1-i)-PSK.

```
delPSK[m_, i_] := 2 Sin[Pi / 2^(m - i + 1)];
```

The symbol error probability of M-DPSK is, at high Es/N0, proportional to Q(2Sqrt[Es/N0]Sin[Pi/(2M)]) [Pawula et al., 1982, p. 1834]. This can be written as Q(del*Sqrt[Es/(2N0)]), where del = 2Sqrt[2]Sin[Pi/(2M)] can be considered a virtual Euclidean distance. (The true Euclidean distance is irrelevant, since the received noise is not additive.)

```
delDPSK[m_, i_] := 2 Sqrt[2] Sin[Pi / 2^(m - i + 2)];
```

The asymptotical coding gain over uncoded BPSK with the same energy per information bit is derived in my notes dated 061005. Assumptions:
- AWGN channel
- RS coding
- hard decoding
- asymptotically high SNR

```
acg[ks_List, ns_, del_] := With[{m = Length[ks]}, 10 Log[10, 1 / 8 * (Plus @@ ks) / ns *
      Min[Table[If[ks[[i]] > 0, del[m, i]^2 (ns + 2 - ks[[i]]), Infinity], {i, 1, m}]]]];
```

```
delPSK[1, 1]
```

```
2
```

Examples:

```
acg[{221, 245, 251}, 255, delPSK] // N
acg[{239, 239, 239}, 255, delPSK] // N
```

```
8.69931
```

```
5.68901
```

Define the performance of plain FEC-coded M-PSK (no BCM). Hence, each bit stream has the same amount of redundancy. Still RS coding.

```
acgFEC[kstot_, ns_, m_, del_] := acg[Table[kstot / m, {m}], ns, del];
```

The ACG of uncoded BPSK, QPSK, and 8-PSK agree with my earlier calculations (see notes):

```
acgFEC[255, 255, 1, delPSK]
acgFEC[2 * 255, 255, 2, delPSK]
acgFEC[3 * 255, 255, 3, delPSK] // N
```

```
0
```

```
0
```

```
-3.57199
```

The ACG of some other coded systems that I have analyzed previously with *soft* decoding are all 3 dB worse in this implementations because of the hard decoding:

```
acg[{170, 170, 170}, 255, delPSK] // N
acg[{77, 203, 230}, 255, delPSK] // N
```

```
11.052
```

```
14.2095
```

```
acg[{239, 239, 239}, 255, delPSK] // N
acg[{221, 245, 251}, 255, delPSK] // N
```

```
5.68901
```

```
8.69931
```

A plotting utility:

```
accum[tab_] := Table[Sum[tab[[i]], {i, 1, j}], {j, 0, Length[tab]}];
```

■ Optimization method

Given a fixed total information rate of kstot/ns bits/channel use, the best choice of ks[[1]],...,ks[[m]] is when all m arguments to the Min function above are equal, if the obvious constraints on ks[[ ]] are relaxed (integers 0<=ks<=ns).

```
optimalks[kstot_, ns_, m_, del_] :=
  With[{c = (m (ns + 2) - kstot) / Sum[del[m, i] ^ -2, {i, 1, m}]},
   Table[ns + 2 - c / del[m, i] ^ 2, {i, 1, m}]];
```

Now optimize ks[[j]] under the constraints that 0<=ks[[j]]<=ns. The function returns either the value ks[[j]] for a specific j or, if the argument j is omitted, the list of ks[[1]],...,ks[[m]].

```
optimalks4[kstot_, ns_, m_, del_, j_] :=
  With[{c = (m (ns + 2) - kstot) / Sum[del[m, i] ^ -2, {i, 1, m}]},
   Which[
    ns + 2 - c / del[m, 1] ^ 2 ≥ 0 && ns + 2 - c / del[m, m] ^ 2 ≤ ns, ns + 2 - c / del[m, j] ^ 2,
    ns + 2 - c / del[m, 1] ^ 2 < 0 && j == 1, 0,
    ns + 2 - c / del[m, 1] ^ 2 < 0, optimalks4[kstot, ns, m - 1, del, j - 1],
    ns + 2 - c / del[m, m] ^ 2 > ns && j == m, ns,
    ns + 2 - c / del[m, m] ^ 2 > ns, optimalks4[kstot - ns, ns, m - 1, del[#1 + 1, #2] &, j]
   ]];
optimalks4[kstot_, ns_, m_, del_] := Table[optimalks4[kstot, ns, m, del, j], {j, 1, m}];
```

```
acgdiff[rate_, ns_, m_, del_] := With[{oks = optimalks4[rate * ns, ns, m, del]},
    acg[oks, ns, del]] - Max@Table[
     acgFEC[rate * ns, ns, m1, del], {m1, Ceiling[rate], m}];
```

■ Optimize ks for 8-PSK

```
{ns, m} = {255, 3};
```

Examples:

```
oks1a = optimalks[m*239, ns, m, delPSK] // N
Table[optimalks4[m*239, ns, m, delPSK, j], {j, 1, m}] // N
optimalks4[m*239, ns, m, delPSK] // N
```

```
{219.483, 246.011, 251.506}

{219.483, 246.011, 251.506}

{219.483, 246.011, 251.506}
```

```
oks2a = optimalks[m*100, ns, m, delPSK] // N
oks2b = optimalks4[m*100, ns, m, delPSK] // N
```

```
{-70.2334, 161.156, 209.078}

{0., 114.333, 185.667}
```

```
oks3a = optimalks[m*253, ns, m, delPSK] // N
oks3b = optimalks4[m*253, ns, m, delPSK] // N
```

```
{248.663, 254.558, 255.779}

{249.265, 254.735, 255.}
```

Verify optimality:

```
Plus @@ # / Length@# & /@ {oks2a, oks2b}
Table[delPSK[m, i] ^2 (ns + 2 - #[[i]]), {i, 1, m}] & /@ {oks2a, oks2b} // TableForm
```

```
{100., 100.}
```

191.689	191.689	191.689
150.547	285.333	285.333

```
Plus @@ # / Length@# & /@ {oks3a, oks3b}
Table[delPSK[m, i] ^2 (ns + 2 - #[[i]]), {i, 1, m}] & /@ {oks3a, oks3b} // TableForm
```

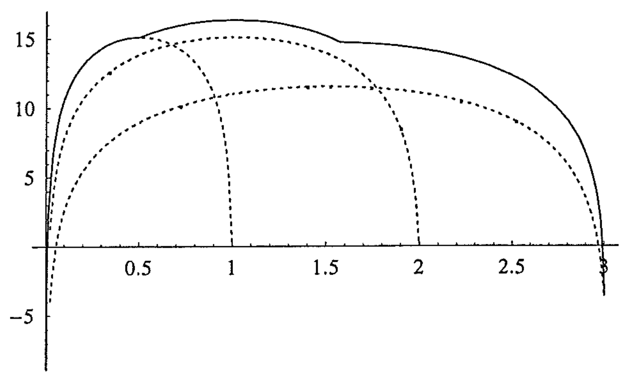
```
{253., 253.}
```

4.88379	4.88379	4.88379
4.53082	4.53082	8.

The following plot gives the ACG of BCM-coded 8-PSK with optimal bit allocation. For comparison, the ACG of uniformly FEC-coded BPSK, QPSK, and 8-PSK are shown.

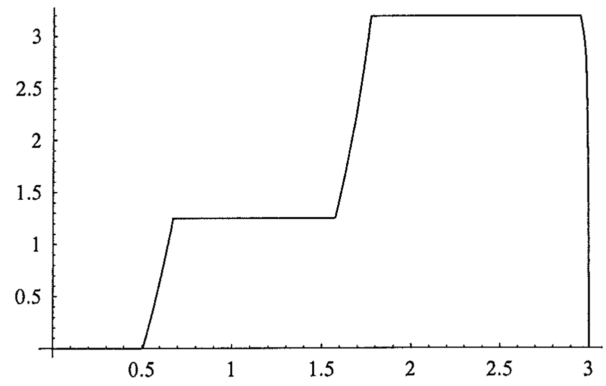
```
fec = {Dashing[{.004, .01}],
   Table[Line@Table[{rate, acgFEC[rate*ns, ns, m1, delPSK]}, {rate, .02, m1, .02}],
    {m1, 1, m}], Dashing[{}]};
```

```
Plot[With[{oks = optimalks4[rate*ns, ns, m, delPSK]},
   acg[oks, ns, delPSK]], {rate, 0, m}, Prolog → fec];
```



Investigate the difference between the black curve above (BCM) and the best of the dotted codes (FEC):

```
Plot[acgdiff[rate, ns, m, delPSK], {rate, 0, 3}];
```



Compare the plateau values with my analysis (see notes date Oct. 2006):

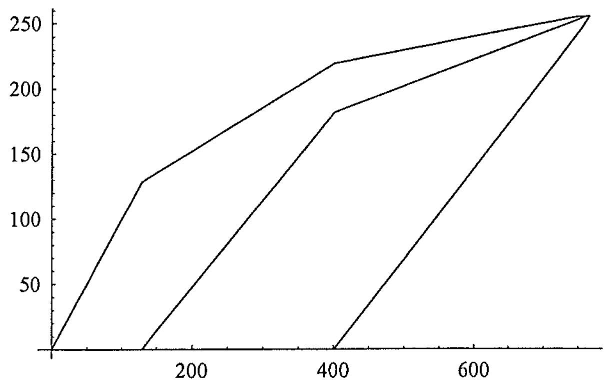
```
acgdiff[#, ns, m, delPSK] & /@ {.4, 1.1, 2.1}
Table[10. Log[10, delPSK[m1, 1] ^-2 / (Sum[delPSK[m1, i] ^-2, {i, 1, m1}] / m1)], {m1, 1, m}]
```

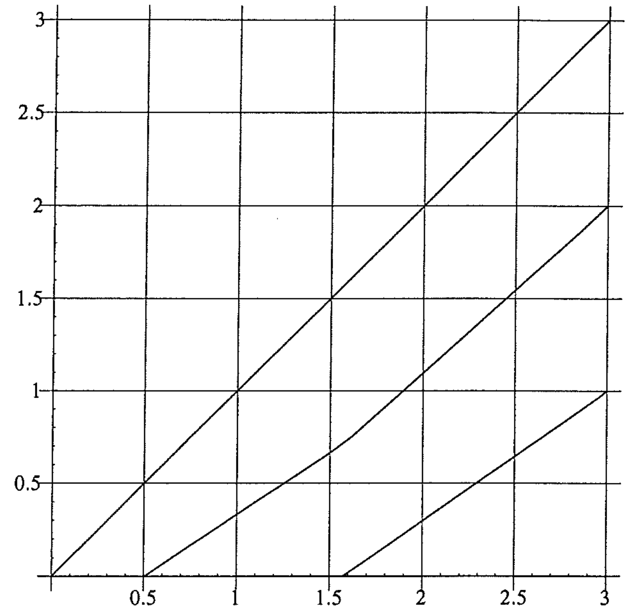
```
{0., 1.24939, 3.18958}
```

```
{0, 1.24939, 3.18958}
```

Illustrate the optimal bit allocation {ks[[1]],ks[[2]],ks[[3]]} for BCM-coded 8-PSK:

```
Plot[Evaluate@optimalks4[kstot, ns, m, delPSK], {kstot, 0, m*ns}];
Plot[Evaluate[accum@optimalks4[rate*ns, ns, m, delPSK] / ns],
  {rate, 0, m}, AspectRatio → Automatic, GridLines → Automatic];
```



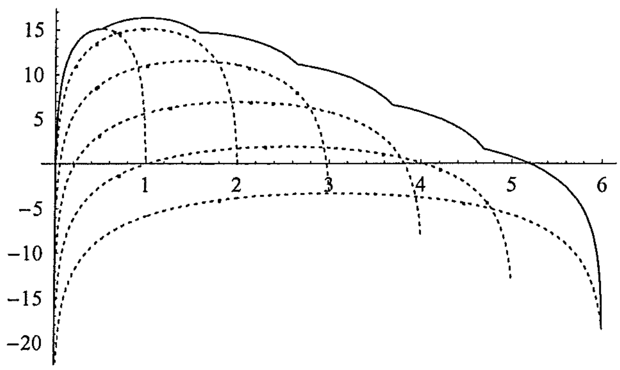


Optimize ks for 64-PSK

```
{ns, m} = {255, 6};
```

```
fec = {Dashing[{.004, .01}],
   Table[Line@Table[{rate, acgFEC[rate * ns, ns, m1, delPSK]}, {rate, .02, m1, .02}],
    {m1, 1, m}], Dashing[{}]};
```

```
Plot[With[{oks = optimalks4[rate * ns, ns, m, delPSK]},
   acg[oks, ns, delPSK]], {rate, 0, m}, Prolog -> fec];
```




```
Table[10. Log[10, delPSK[m1, 1] ^ -2 / (Sum[delPSK[m1, i] ^ -2, {i, 1, m1}] / m1)], {m1, 1, m}]
```

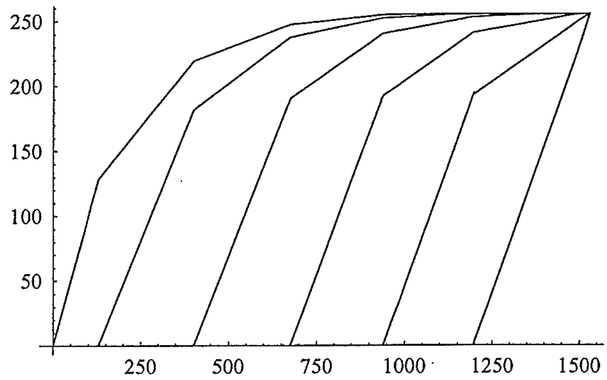
```
{0, 1.24939, 3.18958, 4.6405, 5.69651, 6.51849}
```

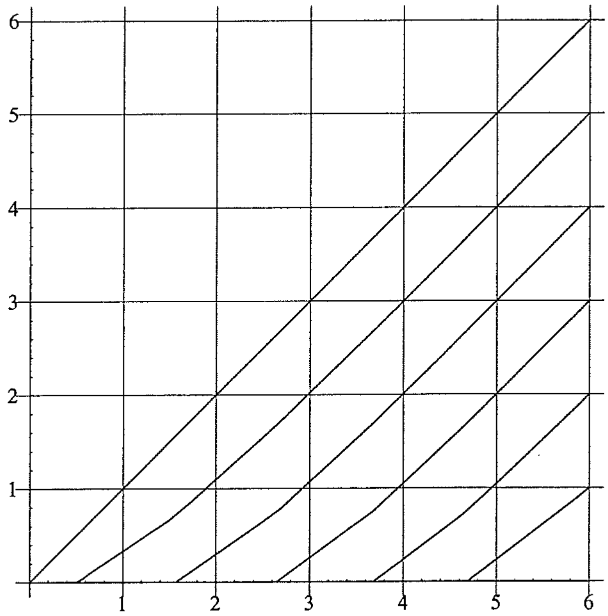
An approximation, which is asymptotically correct for high m (again, see notes):

```
Table[10. Log[10, 3 m1 / 4], {m1, 1, m}]
```

```
{-1.24939, 1.76091, 3.52183, 4.77121, 5.74031, 6.53213}
```

```
Plot[Evaluate@optimalks4[kstot, ns, m, delPSK], {kstot, 0, m * ns}];
Plot[Evaluate[accum@optimalks4[rate * ns, ns, m, delPSK] / ns],
  {rate, 0, m}, AspectRatio -> Automatic, GridLines -> Automatic];
```



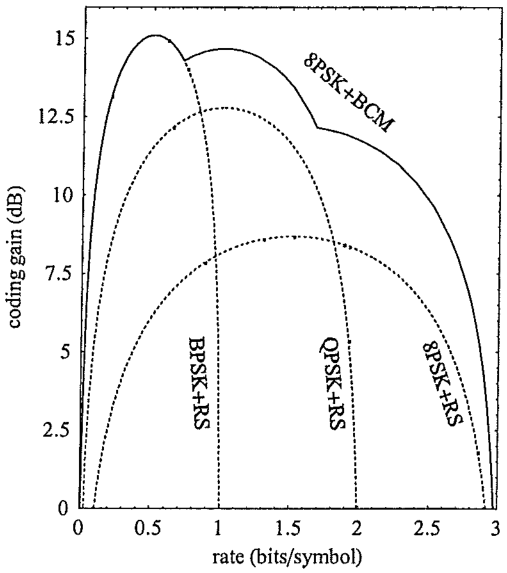


■ Optimize ks for 8-DPSK and 64-DPSK

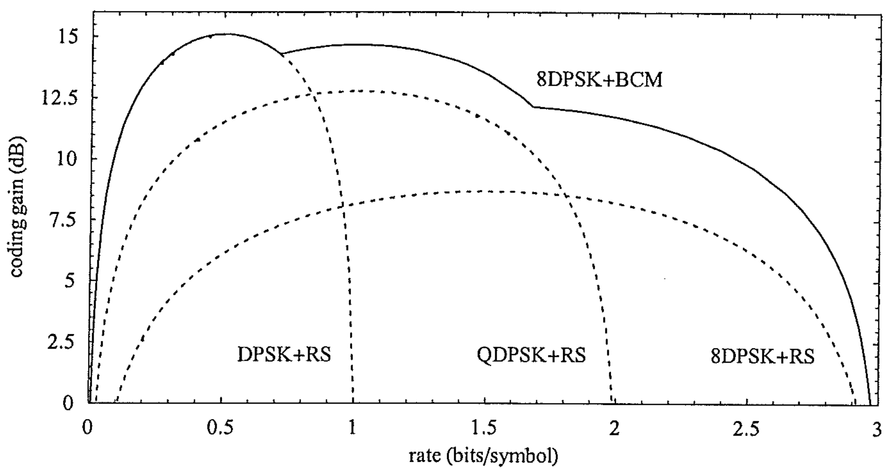
```
{ns, m} = {255, 3};

fec = {Dashing[{.004, .01}],
   Table[Line@Table[{rate, acgFEC[rate * ns, ns, m1, delDPSK]}, {rate, .02, m1, .02}],
    {m1, 1, m}], Dashing[{}]};
```

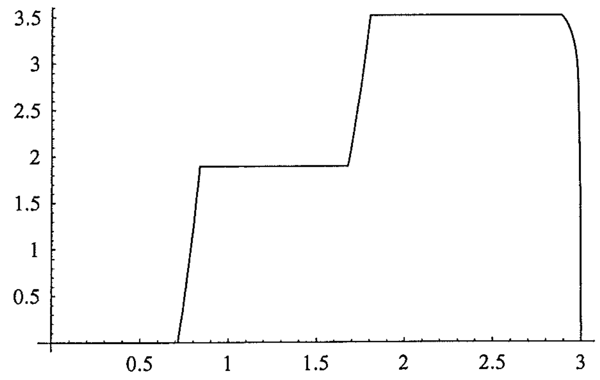
```
Plot[With[{oks = optimalks4[rate * ns, ns, m, delDPSK]},
   acg[oks, ns, delDPSK]], {rate, 0, m}, Prolog → fec, Axes → False,
  Frame → True, PlotRange → {{-10^-6, 3}, {-10^-6, 16}}, AspectRatio → 1.2,
  FrameLabel → {"rate (bits/symbol)", "coding gain (dB)"}, Epilog → {
    Text["BPSK+RS", {.85, 4}, {0, 0}, {.08, -1}],
    Text["QPSK+RS", {1.82, 4}, {0, 0}, {.1, -1}],
    Text["8PSK+RS", {2.6, 4}, {0, 0}, {.35, -1}],
    Text["8PSK+BCM", {1.9, 13.3}, {0, 0}, {1.2, -1}]
  }];
```




```
Plot[With[{oks = optimalks4[rate * ns, ns, m, delDPSK]},
   acg[oks, ns, delDPSK]], {rate, 0, m}, Prolog → fec, Axes → False,
  Frame → True, PlotRange → {{-10^-6, 3}, {-10^-6, 16}}, AspectRatio → .5,
  FrameLabel → {"rate (bits/symbol)", "coding gain (dB)"}, Epilog → {
    Text["DPSK+RS", {.74, 2}, {0, 0}],
    Text["QDPSK+RS", {1.68, 2}, {0, 0}],
    Text["8DPSK+RS", {2.56, 2}, {0, 0}],
    Text["8DPSK+BCM", {1.93, 13.3}, {0, 0}]
  }];
```



```
Plot[acgdiff[rate, ns, m, delDPSK], {rate, 0, 3}];
```

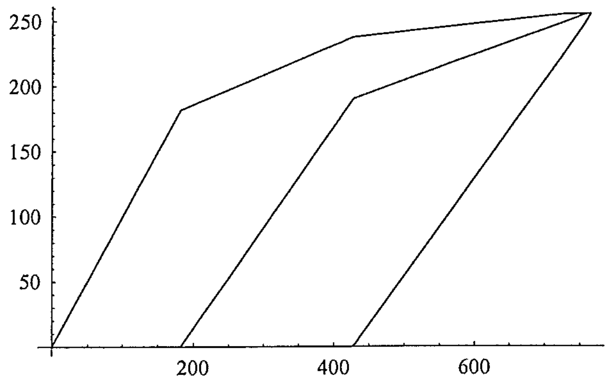


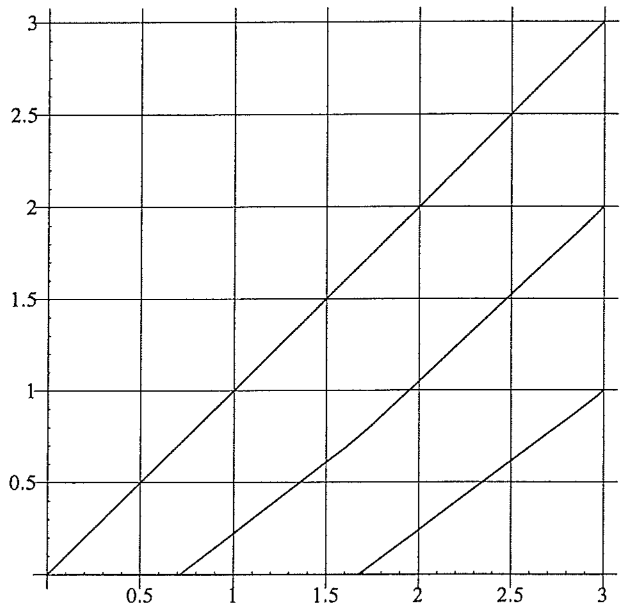

```
acgdiff[#, ns, m, delDPSK] & /@ {.4, 1.1, 2.1}
Table[10. Log[10, delDPSK[m1, 1] ^-2 / (Sum[delDPSK[m1, i] ^-2, {i, 1, m1}] / m1)], {m1, 1, m}]
```

```
{0., 1.89467, 3.51311}
```

```
{0, 1.89467, 3.51311}
```

```
Plot[Evaluate@optimalks4[kstot, ns, m, delDPSK], {kstot, 0, m*ns}];
Plot[Evaluate[accum@optimalks4[rate*ns, ns, m, delDPSK] / ns],
  {rate, 0, m}, AspectRatio → Automatic, GridLines → Automatic];
```

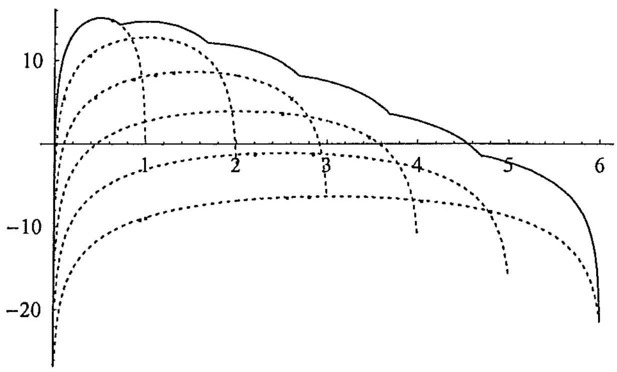





```
{ns, m} = {255, 6};
```

```
fec = {Dashing[{.004, .01}],
   Table[Line@Table[{rate, acgFEC[rate * ns, ns, m1, delDPSK]}, {rate, .02, m1, .02}],
    {m1, 1, m}], Dashing[{}]};

Plot[With[{oks = optimalks4[rate * ns, ns, m, delDPSK]},
   acg[oks, ns, delDPSK]], {rate, 0, m}, Prolog → fec];
```




```
Table[10. Log[10, delDPSK[m1, 1]^-2/(Sum[delDPSK[m1, i]^-2, {i, 1, m1}]/m1)], {m1, 1, m}]

{0, 1.89467, 3.51311, 4.7585, 5.7345, 6.53002}
```