# A Pragmatic Approach to Shaped Coded Modulation

Frans M.J. Willems and Jos J. Wuijts

Eindhoven University of Technology, Electrical Engineering Department,
P.O. Box 513, 5600 MB Eindhoven,The Netherlands

## Abstract

We discuss shaping codes in an elementary way, which is from the standpoint of enumerative coding. Then we show how to combine our shaping methods with a error correcting codes. It turns out that we can easily gain a decibel by using shaping techniques, with or without error correction coding. Finally we describe the combination of our shaping method with the 'pragmatic' approach to coded modulation, which can be realized with VLSI circuits that are widely available.

## 1 PAM

Consider uncoded transmission over an additive Gaussian noise channel, i.e.

$$Y_t = X_t + N_t, \text{ for } t = 1, 2, \cdots,$$

where the noise variables $N_t$ are independent and normally distributed, with mean 0 and variance $\sigma^2$. The (independent) input symbols $X_t$ assume values in the alphabet $\{1-M, 3-M, \cdots, M-1\}$, each value has probability of occurrence of $1/M$, and the information rate $R = \log M$ bits per transmission[1]. The squared Euclidean distance $d_E^2$ between any pair of different input sequences is larger than or equal to 4. For the average input power $P^{av}$ we have that $P^{av} = (M^2-1)/3$. For example, if $M = 4$ the rate $R = 2$ bits per transmission, and $P^{av} = 5$. Error probabilities are determined by the variance $\sigma^2$ of the noise.

The signalling method described above is known as *pulse amplitude modulation (PAM).* In what follows we will discuss alternative methods. The *asymptotic gain* of an alternative method with rate $R$ and minimum squared Euclidean distance $d_{E,min}^2$ over the PAM case, is defined as

$$G \triangleq \frac{2^{2R}-1}{3P^{av}} \cdot \frac{d_{E,min}^2}{4}. \tag{1}$$

To understand this definition, note that we can associate the power $(2^{2R}-1)/3$ to a PAM-system with rate $R$.

## 2 Block shaping

We first consider transmission of sequences with block length $T$. The energy of an input sequence $x^T = x_1 x_2 \cdots x_T$ is defined as

$$E(x^T) \triangleq \sum_{t=1,T} x_t^2.$$

In a PAM block system, the signal points form a T-dimensional cube. It is well known (see [3]) that we can save energy, at most $\pi e/6 = 1.53$ dB, if we choose a spherical signal structure. Using spherical signal structures is called *shaping.* It is easy to see that a signal structure achieves the lowest possible average energy iff all (lattice-) points with energy smaller than $E_{max}$ are signal points, where $E_{max}$ is the maximal energy over all signal points. In other words a set of energy-constrained sequences is minimizes the average energy.

We start our investigations with a simple shaping problem. Let $T = 4$ and $E_{max} = 28$. How many sequences $x^T$ are there with components in $\{\cdots, -3, -1, +1, +3, \cdots\}$ and energy $E(x^T) \leq E_{max}$? Are there efficient ways to enumerate these sequences ? The answer to this questions is affirmative and follows from the work of Schalkwijk [6] and Cover [2].

To make things simple we restrict ourselves first to the 'one-sided' alphabet $\{1, 3, \cdots\}$ and consider

[1]The base of the log is 2.

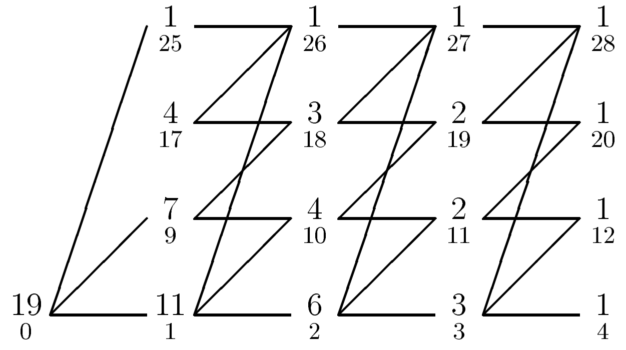

Figure 1: Trellis for $T = 4$ and $E_{max} = 28$.

sequences $u^T$ over this alphabet with energy not larger than 28. Next, we observe that the energy of the first $t$ components in a sequence equals $t$ plus a multiple of 8. Any sequence can be regarded as a path in a trellis, the states corresponding to energies, the branches to a symbol in $\{1, 3, \cdots\}$ (see figure). A path corresponding to one of our energy-constrained sequences starts in the state with energy 0 and ends in one of the final states, having energy 4, 12, 20 or 28. In the figure the energy of a state is denoted by a small number. The largely denoted number in each state is the number of alternatives $A$ that lead from this state to a final state. These numbers can be computed recursively, starting from the final states as follows. For each $t$ consider only those $e$ such that $e - t$ is a multiple of 8. Then

$$\begin{aligned} A(T, e) &:= 1, \\ &\quad \text{for } 0 \leq e \leq E_{max} \text{ and 0 otherwise,} \\ A(t, e) &:= \sum_{u \in \{1,3,\cdots\}} A(t + 1, e + u^2), \\ &\quad \text{for } t = T - 1, T - 2, \cdots, 0. \end{aligned}$$

From the figure we can see that $A(0, 0) = 19$, which means that there are 19 sequences with components in $\{1, 3, \cdots\}$ having energy not exceeding 28. The rate of this one-sided code is $R_u = (\log 19)/4 = 1.062$. We can similarly determine recursively the number of sequences with a given energy. It turns out that there is one sequence with energy 4, four with energy 12, six with energy 20, and eight with energy 28. Therefore the average energy is $(4 + 4 \cdot 12 + 6 \cdot 20 + 8 \cdot 28)/19 = 20.842$, and $P^{av} = 5.211$. There is no one-sided code with 19 codewords having smaller average power.

Now that we have computed the number of sequences of length 4 with components in $\{1, 3, \cdots\}$ with energy not larger than 28, the question arises how we can enumerate these sequences. In other words, does there exist an ordering that gives the sequences an index which can easily be computed from the sequence and vice versa ? It follows from [6] and [2] that, if we assume a lexicographical ordering over the sequences, the index can be determined efficiently from the sequence. It turns out that

$$i(u^T) = \sum_{t=1,T} \sum_{w<u_t \vee w \in \{1,3,\cdots\}} A(t, \sum_{s=1,t-1} u_s^2 + w^2).$$

For example, $i(3131) = A(1, 1^2) + A(3, 3^2 + 1^2 + 1^2) = A(1, 1) + A(3, 11) = 11 + 2 = 13$, which can be found recursively in the array $A(t, e)$, as we can see.

Computing the sequence $u^T$ with a given index $i(u^T)$ is also not very difficult. Suppose $i(u^T) = 8$. Then the first component of $u^T$ can only be 1. If it would be 3 or larger, the index of the sequence would be $A(1, 1) = 11$ or more. If the second component would also be 1, the index would be smaller than $A(2, 2) = 6$. With the second component equal to 3, we would get an index not smaller than $A(2, 2) = 6$ and smaller than $A(2, 2) + A(2, 10) = 6 + 4 = 10$, hence the second component must be 3. If the third component was 1, the index would be smaller than $A(2, 2) + A(3, 11) = 6 + 2 = 8$. A third component equal to 3, gives an index not smaller than $A(2, 2) + A(3, 11) = 6 + 2 = 8$ and smaller than $A(2, 2)+A(3, 11)+A(3, 19) = 6+2+2 = 10$, thus the third component is also 3. The fourth component must be 1. If it would be 3, this would result in an index $A(2, 2) + A(3, 11) + A(4, 20) = 6 + 2 + 1 = 9$. Conclusion is that, by comparing the index with 'index boundaries', computed from the array $A(t, e)$ recursively, we can find the sequence corresponding to a given index.

Before we consider the gain of this code, we observe that each of the 19 sequences in this code with components in $\{1, 3, \cdots\}$ having energy not larger than 28, can easily be transformed into 16 sequences with components in $\{\cdots, -3, -1, +1, +3, \cdots\}$ and having the same energy, by placing a sign for each of the four components. By doing so we obtain a code with $16 \cdot 19 = 304$ sequences, which is optimal, i.e. it minimizes $P^{av}$. The rate of this double-sided code is $R = (\log 304)/4 = 2.062$. The minimum squared Euclidean distance is still 4, hence by (1) the gain of this code is $G = (2^{2\cdot 2.062}-1)/(3\cdot 5.211) = 1.051$ or 0.218 dB, so we have gained a little bit. It should be noted that shaping requires constellation expansion. In our example we see that signal components can be $-5$ or $+5$, while the rate is roughly 2. In principle there is no restriction on these values. In practice, restrictions on the signal amplitudes may be necessary. Such restrictions normally do not have a big effect on the gain.

Larger gains can be obtained by increasing the block length $T$. In the table below we have listed for several block lengths, the maximum energy $E_{max}$ that gives a rate $R$ and a gain $G$.

| $T$ | $E_{max}$ | $P^{av}$ | $R$ | $G$ |
|---|---|---|---|---|
| 4 | 28 | 5.211 | 2.062 | 0.218dB |
| 8 | 48 | 5.169 | 2.102 | 0.509dB |
| 16 | 80 | 4.638 | 2.064 | 0.734dB |
| 32 | 136 | 4.100 | 2.006 | 0.901dB |
| 64 | 264 | 4.051 | 2.019 | 1.039dB |

The storage complexity (array size) of this code is quadratical in $T$, the computational complexity is linear. Optimal decoding (minimizing the block-error probability) of these codes requires Viterbi detection on the one-sided trellis. This trellis is needed anyhow for indexing as we saw. Threshold detection results in a small decrease in performance.

## 3 Block Coding

While shaping is concerned with the shape of the signal structure, coding has the intention to enlarge the distances between the signal points. This is accomplished by using Ungerboeck's idea of *set partitioning* [7]. The channel input symbol set $\mathcal{A} = \{\cdots,-3,-1,+1,+3,\cdots\}$ is partitioned into two sets, $\mathcal{A}_0 \triangleq \{\cdots,-3,+1,+5,\cdots\}$ and $\mathcal{A}_1 \triangleq \{\cdots,-5,-1,+3,\cdots\}$. This splitting allows combination of uncoded (but shaped as we shall see later) data with a binary error correcting code. The idea is as follows. Take a binary error correcting code with minimum Hamming distance $d_{H,min}$, block length $T$ and rate $R_b$, and assume that we have uncoded one-sided sequences $u^T$ with the same block length, rate $R_u$ and power $P_u^{av}$. We transform these uncoded one-sided sequences into coded double-sided signals $x^T$ componentwise. The symbol $u_t$, where $t = 1, T$ from the uncoded sequence together with the binary symbol $b_t$ of the codeword $b^T$ determines the double-sided symbol $x_t$ according to the table below.

| $b_t$ \| $u_t$ | 1 | 3 | 5 | 7 | 9 | 11 | $\cdots$ |
|---|---|---|---|---|---|---|---|
| 0 | $+1$ | $-3$ | $+5$ | $-7$ | $+9$ | $-11$ | $\cdots$ |
| 1 | $-1$ | $+3$ | $-5$ | $+7$ | $-9$ | $+11$ | $\cdots$ |

Observe that the table is constructed in such a way that a binary symbol $b_t = 0$ produces symbols in $\mathcal{A}_0$ and $b_t = 1$ leads to a symbol in $\mathcal{A}_1$. The absolute value of the double sided symbol $x_t$ is equal to the one-sided symbol $u_t$.

What about the rate of this code ? Obviously the rate $R = R_u + R_b$. The average power of the code is equal to the power of the one-sided signal, i.e. $P^{av} = P_u^{av}$.

To compute the minimum squared Euclidean distance we consider two situations. First we assume that the error correcting codeword $b^T$ is fixed. In that case the components $x_t$ of all codewords $x^T$ take values in the same set $\mathcal{A}_{b_t}$, where $t = 1, T$. All different codewords differ in at least one component, and two different components in a set $\mathcal{A}_b$ for a $b \in \{0,1\}$ differ at least 4. Therefore, codewords $x^T$ and $\tilde{x}^T$ that result from the same $b^T$ have a squared Euclidean distance of 16 or more.

Secondly we have to investigate the case where we have two codewords $x^T$ and $\tilde{x}^T$ that resulted from different binary codewords $b^T$ and $\tilde{b}^T$. Consider component $t$ and assume that $b_t \neq \tilde{b}_t$. Now $x_t \in \mathcal{A}_{b_t}$ and $\tilde{x}_t, \in \mathcal{A}_{\tilde{b}_t}$. The difference between a symbol from $\mathcal{A}_0$ and a symbol from $\mathcal{A}_1$ is 2 or more. Therefore each differing component in the binary error correcting code contributes at least 4 to the squared Euclidean distance. Since the minimum distance of the binary error correcting code is $d_{H,min}$ we may conclude that for code words $x^T$ and $\tilde{x}^T$ that result from different binary codewords the squared Euclidean distance is $4d_{H,min}$ or more. This leads to

$$d^2_{E,min} = \min(16, 4d_{H,min}),$$

which implies that it is not useful to apply binary codes with $d_{H,min} > 4$. Observe that, although this is not of our concern, we could use this method to combine also unshaped one-sided sequences with a binary error correcting code.

Now that we have expressions for the rate, average power, and minimum squared Euclidean distance, we can compute the gains of this method using (1). We consider two cases. First we assume that the error correcting code is a single parity check code. This code has rate $R_b = (T-1)/T$ and $d_{H,min} = 2$, which leads to $d^2_{E,min} = 8$. For each block length $T$ we have chosen a maximum energy $E_{max}$ such that the total rate $R = R_u + R_b$ is roughly 2 bit per transmission. We have listed this $E_{max}$ and the resulting $P^{av}$, the rate $R$, and gain $G$ in the table below, for several values of $T$.

| $T$ | $E_{max}$ | $P^{av}$ | $R$ | $G$ |
|---|---|---|---|---|
| 4 | 36 | 6.750 | 2.000 | 1.707dB |
| 8 | 56 | 5.961 | 2.079 | 2.755dB |
| 16 | 80 | 4.638 | 2.001 | 3.345dB |
| 32 | 144 | 4.336 | 2.015 | 3.726dB |
| 64 | 264 | 4.051 | 2.004 | 3.949dB |

Secondly we consider the case where an extended Hamming code is used as binary error correcting code. With parameter $m$, the number of parity check equations of the underlying Hamming code, this gives block length $T = 2^m$ and rate $R_b = (T-m-1)/T$. The distance of this code is $d_{H,min} = 4$, therefore $d^2_{E,min} = 16$. Again for each $T$, we have chosen a maximum energy $E_{max}$ such that the total rate $R$ is roughly 2 bit per transmission. Just like before we have listed this $E_{max}$ and the resulting power $P^{av}$, the rate $R$, and gain $G$ in the table below, for several values of $T$.

| $T$ | $E_{max}$ | $P^{av}$ | $R$ | $G$ |
|---|---|---|---|---|
| 4 | 88 | 14.789 | 2.062 | 1.708dB |
| 8 | 88 | 9.169 | 2.013 | 3.471dB |
| 16 | 12 | 6.870 | 2.033 | 4.854dB |
| 32 | 176 | 5.280 | 2.000 | 5.787dB |
| 64 | 296 | 4.537 | 2.007 | 6.489dB |

What follows from these tables is that we can easily gain roughly a decibel by applying shaping techniques.

Optimal decoding for these codes is of course guaranteed when we use the Viterbi algorithm on the product of the shaping trellis and the coding trellis. As in Calderbank, Lee and Mazo [1] however, we can use a suboptimal decoder which operates on the coding trellis only, without loosing too much performance.

## 4 A Pragmatic Approach

Viterbi, Wolf, Zehavi and Padovani described in [8] an elegant method for trellis coded (amplitude) modulation based on a convolutional code for which there exists, by now standard, VLSI decoding hardware. Their technique however does not contain shaping. We show here how to combine their methods with our shaping techniques.

Just like Ungerboeck, Viterbi et al. use set partitioning of the channel input symbol set $\mathcal{A} = \{\cdots, -3, -1, +1, +3, \cdots\}$. They split this set into four sets, $\mathcal{A}_{00} \triangleq \{\cdots, -7, +1, +9, \cdots\}$, $\mathcal{A}_{01} \triangleq \{\cdots, -5, +3, +11, \cdots\}$, $\mathcal{A}_{11} \triangleq \{\cdots, -11, -3, +5, \cdots\}$ and $\mathcal{A}_{10} \triangleq \{\cdots, -9, -1, +7, \cdots\}$, i.e. a Gray code splitting.

Instead of a binary error correcting block code we have a convolutional code here. It produces a pair of binary code symbols $b^1_t b^2_t$ for each input bit $v_t$ that enters the convolutional encoder, for $t = 1, 2, \cdots$.

Again we assume that we have an uncoded (but shaped) one-sided signal $u_1 u_2 \cdots$ with rate $R_u$ and average power $P^{av}_u$. We could try to transform this uncoded one-sided signal into a coded double-sided signal componentwise, just as we did before in section 3. Therefore we construct the table below.

| $b^1_t b^2_t$ | $u_t$ | 1 | 3 | 5 | 7 | 9 | 11 | $\cdots$ |
|---|---|---|---|---|---|---|---|---|
| 00 | | $+1$ | $*$ | $*$ | $-7$ | $+9$ | $*$ | $\cdots$ |
| 01 | | $*$ | $+3$ | $-5$ | $*$ | $*$ | $+11$ | $\cdots$ |
| 11 | | $*$ | $-3$ | $+5$ | $*$ | $*$ | $-11$ | $\cdots$ |
| 10 | | $-1$ | $*$ | $*$ | $+7$ | $-9$ | $*$ | $\cdots$ |

Suppose that the uncoded symbol $u_t = 5$. Then we see from this table that there are only two possible pairs $b^1_t b^2_t$ that lead to an $x_t$ with absolute value 5, i.e. 01 and 11. This implies that we are in trouble if the output of the convolutional encoder would be either 00 or 10. However, since the applied convolutional code has the property that inverting its input symbol $v_t$ results in an inversion of the output pair $b^1_t b^2_t$, there is always an input $v_t$ that yields either 01 or 11. The idea is now, to apply this input to the convolutional encoder. The channel input $x_t$ can now be determined using the table. For example suppose that the output of the convolutional code would be 00 with input 0, and consequently 11 for input 1. Then, with $u_t = 5$, we must set the input $v_t$ to 1. The output pair $b^1_t b^2_t = 11$ now determines, together with $u_t = 5$, the channel symbol $x_t = +5$, as in the table.

It is straightforward to see that the code sequences that are generated by our method can also be generated by the pragmatic encoder of

Viterbi et al. (ignoring for a moment that shaping requires constellation expansion). The consequence of this is that the distance structure remains the same, which leads to a more or less identical error behavior. Moreover it implies that the original pragmatic decoder can be used.

To be able to compare the rate of our method with that of the original pragmatic technique, we first note that in the original pragmatic encoder for each time unit, one information bit enters the convolutional encoder, while the remaining $R-1$ bits address a channel input symbol from one of the four sets $\mathcal{A}_{ij}$, for some $ij \in \{0,1\}^2$ which is specified by the the output of the convolutional encoder. In this case $\mathcal{A} = \{1-2^{R+1}, 3-2^{R+1}, \cdots, 2^{R+1}-1\}$ and $P^{av} = (2^{2R+2}-1)/3$. We can also achieve rate $R$ by using our method, using all one-sided sequences which are composed out of symbols from $\{1, 3, \cdots, 2^{R+1}-1\}$ with equal probability, which means that they are unshaped. The rate of this one-sided code is $R_u = R$ while that of the convolutional code $R_b = 0$ since its inputs are restricted. The average power of the method is equal to the average power of the unshaped one-sided code $P_u^{av}$ which is also $(2^{2R+2}-1)/3$. So the original pragmatic method corresponds to our method with an unshaped one-sided code. Now taking instead of the unshaped code a code which is shaped we can easily gain a decibel as we have seen in section 2.

## 5 Conclusions and Remarks

We have discussed shaping methods for transmission in blocks. These shaping techniques are based on enumerative source coding ideas (see [6] and [2]). We also investigated the combination of shaping methods with error correcting codes. It turns out that a gain of one decibel can easily be achieved, with and without coding. Our shaping techniques differ from that of Laroia et al. [4] in that we use a different ordering over the energy-constrained sequences. Laroia et al. first order the sequences according to their energy level, and then, within the same energy level sequences are ordered lexicographically. We apply a lexicographical ordering over all energy-constrained sequences. This reduces the storage complexity.

The well-known 'pragmatic' approach to coded modulation, applies a convolutional code instead of a block code [8]. VLSI decoding hardware exists for this code. We have demonstrated how to combine our shaping methods with this convolutional coding technique, so that the decoding hardware still can be used. This pragmatic approach to shaping can easily result in an additional gain of a decibel.

What we did not do is the following. To keep things simple we have only discussed one-dimensional signal structures. Also we have left out of consideration properties as 'shaping constellation expansion ratios (CER)' and 'peak-to-average power ratios (PAR)'. It is however very easy to generalize our shaping methods to higher dimensions in such a way that desirable CER's and PAR's can be achieved. Another property that we did not bother about, is the fact that the rates that we have found are not integer. Pruning of the trellis however can be used to overcome this difficulty. Finally we mention that we did only discuss block shaping here. In combination with convolutional code however *stream shaping* seems more appropriate. Therefore it needs to be investigated whether there exist energy constraints from which we can design as in [5] finite-state modulation codes that allow state-independent decoding to limit error propagation, and still achieve a reasonable shaping gain. The constraint we think of, is a sliding constraint on the energy in a window of a given number of symbols.

If we compare our results with that of Laroia et al. we come to the conclusion that there is a considerable overlap. We must remark however that we started our investigations in 1991. This resulted in two reports ([9] and [10]), both in Dutch.

## References

[1] A.R. Calderbank, T.A. Lee, and J.E. Mazo, "Baseband Trellis Codes with a Spectral Null at Zero," *IEEE Trans. Inform. Theory,* vol. IT-34, pp. 425-434, May 1988.

[2] T.M. Cover, "Enumerative Source Encoding," *IEEE Trans. Inform. Theory,* vol. IT-19, pp. 73-77, Jan. 1973.

[3] G.D. Forney, Jr., R.G. Gallager, G.R. Lang, F.M. Longstaff, and S.U. Qureshi, "Efficient Modulation for Band-Limited Chan-

nels," *IEEE Journ. Select. Areas Comm.*, vol. SAC-2, pp. 632-647, Sept. 1984.

[4] R. Laroia, N. Farvardin, S. Tretter, "On Optimal Shaping of Multidimensional Constellations", submitted to *IEEE Trans. on Inform. Theory,* 1992.

[5] B.H. Marcus, P.H. Siegel, and J.K. Wolf, "Finite-State Modulation Codes for Data Storage," IEEE Journ. Select. Areas Comm., vol. SAC-10, pp. 5-37, Jan. 1992.

[6] J.P.M. Schalkwijk, "An Algorithm for Source Coding," *IEEE Trans. Inform. Theory,* vol. IT-18, pp. 395-399, May 1972.

[7] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. Inform. Theory,* vol. IT-28, pp. 55-67, Jan. 1982.

[8] A.J. Viterbi, J.K. Wolf, E. Zehavi, and R. Padovani, "A Pragmatic Approach to Trellis Coded Modulation," *IEEE Comm. Mag.,* p. 11-19, July 1989.

[9] J.J. Wuijts, "De Constructie van een Energiebegrensde Kanaalcode met Behulp van Enumerative Technieken," stageverslag, Eindhoven University of Technology (in Dutch), Oct. 1991.

[10] J.J. Wuijts, "Shaping en Coding bij n-Dimensionale Puls Amplitude Modulatie," afstudeerverslag, Eindhoven University of Technology (in Dutch), Feb. 1993.