

Vector Modulus Algorithms for Blind Equalization of Shaped Modulation

Mark A. Haun
Dept. of ECE
University of Illinois
markhaun@uiuc.edu

Vanessa Y. Yang
TRW
One Space Park
Redondo Beach, CA 90278

Douglas L. Jones
Dept. of ECE
University of Illinois
dl-jones@uiuc.edu

16 September 1999

Abstract

The vector constant modulus algorithm (VCMA) was recently introduced as an extension of CMA which can equalize data from shaped sources having nearly Gaussian marginal distributions.

VCMA was developed with the shaping technique of shell mapping in mind, but it also equalizes data from other shaping methods, such as trellis shaping. Some simple changes in the structure of VCMA permit its use in fractionally spaced equalizers, the benefits of which include shorter equalizer lengths, built-in matched filtering, and the possibility of global convergence on FIR channels. The use of a vector modulus in the cost function has general applicability and is used to obtain the vector reduced constellation algorithm (VRCA) and the vector multimodulus algorithm (VMMA), both based on constant modulus criteria and derived from their scalar modulus counterparts, RCA and MMA. Although the global convergence of VCMA has not yet been fully proved, some theoretical results and the close parallels with CMA suggest that it inherits the desirable properties of CMA. Simulations verify the superior performance of $T/2$ -spaced VCMA compared to baud-spaced VCMA on various short FIR channels, the ability of VCMA to equalize trellis-shaped data, and the performance of one of the related vector modulus algorithms.

EDICS: SP 2.5.4 Signal Reconstruction: Deconvolution and System Identification
SP 2.8.1 DSP Applications: Communications

Corresponding author: Mark Haun, Beckman Institute MC-251, 405 N. Matthews, Urbana, IL 61801; voice: (217) 353-7437; fax: (217) 244-1642; email: markhaun@uiuc.edu

1 Introduction

The constant modulus algorithm (CMA), first described in [8] and [16], is a popular algorithm for blind equalization in digital communication systems. CMA uses a cost function which penalizes deviations of the received signal's magnitude from a circle of constant modulus. Interestingly, CMA works well even with source signal constellations like QAM, in which the symbols are not of constant modulus. Its convergence ability does depend, however, on the probability distribution of the transmitted symbols; those symbols must have sub-Gaussian marginal distributions [1].

Unfortunately, this requirement is at odds with source shaping techniques which attempt to increase system performance by using non-equiprobable signaling, as an approximately Gaussian source is precisely the goal of source shaping. Source shaping is gaining in popularity; the most prominent example is the V.34 telephone modem standard which incorporates a shaping technique known as shell mapping [3]. A modest amount of shaping gain is inexpensive to implement when compared to the computationally intensive coding schemes already being used in many applications. Thus, it is likely that source shaping will be included in many more standards.

When shaping is applied to a two-dimensional constellation, the marginal source symbol distributions become closer to Gaussian. Under these conditions, the convergence of CMA slows down and eventually fails altogether [1]. This limitation afflicts not only CMA, but other methods based on higher-order statistics (HOS) as well [14]. Algorithms based on second-order statistics (SOS) are expensive and have not yet been shown to work reliably [2]. With shaped signal constellations becoming more common, an alternative blind equalization algorithm is clearly needed.

The vector constant modulus algorithm (VCMA) [20] was recently introduced as an extension of CMA which can equalize data from shaped sources having nearly Gaussian marginal distributions, but uniform (or at least moderately sub-Gaussian) vector distributions. In [20], Yang and Jones

developed VCMA for baud-spaced equalizers, using the source shaping technique of shell-mapping as a conceptual guide. Shell-mapping and other shaping techniques will be described in Section 2; however, from a simplified viewpoint shell-mapping can be seen as choosing signal points uniformly distributed in a $2N$ -dimensional sphere and transmitting them as a sequence of N complex values. The key insight is that vectors of N successive samples ought to be distributed uniformly in $2N$ -dimensional space, so a version of CMA which operates on vectors of the received samples should be able to equalize the shaped data. In the present paper it is further shown through simulations that this ability of VCMA is not restricted to one particular shaping method.

In Section 3 we review the VCMA cost function and tap update equation, then show how to adapt the algorithm to fractionally spaced equalizers. This important result removes a barrier to the use of VCMA in many applications where fractionally spaced equalizers are standard practice due to their ability to perfectly equalize many FIR channels and perform other receiver functions at the same time [17]. Also in this section, computational short-cuts which reduce the cost of VCMA to less than twice that of CMA are presented in detail. Current knowledge about the convergence properties of VCMA is reviewed in Section 4.

VCMA is not the only algorithm that may be used for shaped data equalization. Other CMA-like algorithms exist whose cost functions may be suitably modified to form new vector modulus algorithms. Two of these are derived in Section 5 and may offer advantages over VCMA for certain applications.

Section 6 is devoted to simulations of the algorithms discussed in the previous sections, and verifies the practicality of basic VCMA and many of its variations.

2 Source Shaping

Source shaping is able to provide gain, independent of coding, by using efficient constellation shapes. Consider a simple example in two dimensions: A circular constellation has an inherent shaping gain of about 0.2 dB over a more traditional square containing the same number of signal points and unchanged spacing between the points, because the average energy of the constellation points is smaller. The symbol error probability will be similar for both constellations due to the common spacing, but the square constellation is wasting some energy on the corner points that would be better spread out evenly around the edges.

By choosing signal points from uniform spherical constellations in higher dimensions, higher gains are possible, approaching 1.53 dB for the shaping gain of an N -dimensional sphere over an N -dimensional cube as $N \rightarrow \infty$ [5]. The points chosen from the hypersphere are simply transmitted as a sequence of successive two-dimensional signal points. As the dimension of the underlying constellation increases beyond two, this has the effect of imposing a nonuniform probability distribution across the two-dimensional constellation. In the limit of large N , it is equivalent to imposing marginal Gaussian distributions on the two-dimensional constellation, which renders the signal unequalizable by the conventional constant modulus algorithm.

In theory, source shaping does not seem difficult to accomplish. One conceptually simple approach is to select points uniformly from a high-dimensional hypersphere and transmit them as a series of symbols from a two-dimensional base constellation. In practice, the challenge lies in finding a simple way to select only the points which are inside the hypersphere, while providing a way to index those points efficiently. A variety of techniques have been used for this task.

2.1 Shell Mapping

Shell mapping is the most common source shaping technique in current use and is the shaping method specified by the V.34 telephone modem standard [3]. In shell mapping, a two-dimensional circular constellation is divided into M rings of increasing energy. Each ring has an equal number of points and is assigned a cost label proportional to the energy of the points (often the labels are simply taken to be the integers counting rings outward). Now consider the $2N$ -dimensional region containing all of the points described by sequences of N symbols from the original two-dimensional constellation. Shell mapping provides an efficient way to select the 2^b lowest-cost signal points from this higher-dimensional set, where the cost is a measure of signal energy determined by adding the N individual ring costs. The subconstellation selected by shell mapping in this way is very close to a sphere in $2N$ dimensions. Thus, b bits can be sent using N of the two-dimensional signal points transmitted in sequence. Inherent in this technique is the ability to send a fractional number of bits per symbol [12].

The applicability of VCMA to shell-mapped data is rather intuitive: By using vectors of the received signal points, the algorithm is able to operate on signal vectors having a uniform distribution and avoid the problem of the almost-Gaussian marginal distributions of signal points in two dimensions [20]. By this reasoning, any shaping technique which achieves non-equiprobable signaling by projecting points from a higher-dimensional uniform constellation should produce data equalizable by VCMA.

2.2 Trellis Shaping

A different shaping approach was described by Forney in [4]. In contrast to shell mapping, which chooses points in a spherical constellation of finite dimension, trellis shaping searches the trellis of a convolutional code to find the lowest-energy sequence from a larger set of possible transmitted

sequences.

A simple form of trellis shaping, which Forney calls “sign-bit shaping,” is accomplished by starting with a square two-dimensional QAM constellation. This constellation is partitioned into its four quadrants, with any given point being addressed by a combination of two sign bits to select the quadrant plus other bits that select the signal point within the quadrant. The original data sequence consisting of points in the square constellation is shaped by adding (modulo-2) its sequence of sign bits to a sequence in the convolutional code, thereby scrambling the quadrants of the original points in a constrained manner. A Viterbi algorithm search through the trellis of the convolutional code is performed using the energies of the resultant modified signal points as the branch metrics in order to select the convolutional codeword that minimizes the transmitted signal energy. It is not possible to describe this operation in a specific, finite-dimensional space, yet simulations (presented later in this paper) have shown that VCMA is able to equalize trellis-shaped data.

3 The Vector Constant Modulus Algorithm

3.1 The Constant Modulus Algorithm (CMA)

Consider the complex baseband model of a digital communication system shown in Figure . The transmitted data sequence $\{a_n\}$ consists of independent, identically distributed symbols chosen from a signal constellation with symmetries satisfying $E[a_n^2] = 0$. The transmitted sequence is filtered by a channel \mathbf{h} and passed through an equalizer \mathbf{c} . The equalizer output is $z = \mathbf{y} * \mathbf{c}$, where \mathbf{y} is a vector of samples of the channel output $y = \mathbf{a} * \mathbf{h}$.

Suppose the constant modulus algorithm (CMA) is used to adapt the equalizer [8], [16]. The cost function for CMA is

$$CF_{\text{CMA}} = E(|z_n|^p - R_p)^2 \tag{1}$$

where

$$R_p = \frac{E|a_n|^{2p}}{E|a_n|^p} \quad (2)$$

is a constant which depends on the signal constellation and acts as a scaling factor for the equalized data. A stochastic gradient update [11] with step size μ is used to update the filter coefficients:

$$\mathbf{c}_{n+1} = \mathbf{c}_n - \mu \left[\frac{\partial CF_{\text{CMA}}}{\partial \mathbf{c}_n^*} \right] \quad (3)$$

where, by convention, the derivative is with respect to \mathbf{c}_n^* . Recalling that $z_n = \mathbf{y}_n^T \mathbf{c}_n$, this is calculated as

$$\begin{aligned} \left[\frac{\partial CF_{\text{CMA}}}{\partial \mathbf{c}_n^*} \right] &= \\ E \left[2(|z_n|^p - R_p) p |z_n|^{p-1} \frac{\partial}{\partial \mathbf{c}_n^*} ((\mathbf{y}_n^T \mathbf{c}_n)^* (\mathbf{y}_n^T \mathbf{c}_n))^{1/2} \right] &= \\ E \left[2(|z_n|^p - R_p) p |z_n|^{p-1} \frac{1}{2} |z_n|^{-1} \mathbf{y}_n^* z_n \right] & \end{aligned} \quad (4)$$

Dropping the expectation and simplifying, we obtain the desired tap update equation

$$\mathbf{c}_{n+1} = \mathbf{c}_n - \lambda \mathbf{y}_n^* z_n |z_n|^{p-2} (|z_n|^p - R_p) \quad (5)$$

where the step size λ is μ multiplied by the constant factors arising in the differentiation. For computational convenience, p is normally set to 2.

3.2 Baud-Spaced VCMA

The vector constant modulus algorithm (VCMA) is an extension of CMA which operates on vectors of the received signal samples. Let $\mathbf{a}_n = [a_n \ a_{n-1} \ \cdots \ a_{n-N+1}]^T$ be a vector of N transmitted complex signal points corresponding to a point in the $2N$ -dimensional uniform constellation (shell-mapping is in mind here). Let $\mathbf{z}_n = [z_n \ z_{n-1} \ \cdots \ z_{n-N+1}]^T$ be a vector of the last N output samples from the equalizer; it is calculated by $\mathbf{z}_n = \mathbf{Y}_n^T \mathbf{c}_n$, where

$$\mathbf{Y}_n = \begin{bmatrix} y_n & y_{n-1} & \cdots & y_{n-N+1} \\ y_{n-1} & y_{n-2} & \cdots & y_{n-N} \\ \vdots & \vdots & & \vdots \\ y_{n-L_c+1} & y_{n-L_c} & \cdots & y_{n-L_c-N+2} \end{bmatrix} \quad (6)$$

and L_c is the number of equalizer taps.

VCMA uses a cost function identical to that of CMA except that a vector modulus is used:

$$CF_{\text{VCMA}} = E(|\mathbf{z}_n|^p - R_p)^2 \quad (7)$$

where

$$R_p = \frac{E|\mathbf{a}_n|^{2p}}{E|\mathbf{a}_n|^p} \quad (8)$$

We again use a stochastic gradient update with step size μ to update the filter coefficients:

$$\mathbf{c}_{n+1} = \mathbf{c}_n - \mu \left[\frac{\partial CF_{\text{VCMA}}}{\partial \mathbf{c}_n^*} \right] \quad (9)$$

Recalling that $\mathbf{z}_n = \mathbf{Y}_n^T \mathbf{c}_n$, the derivative is calculated as

$$\begin{aligned} \left[\frac{\partial CF_{\text{VCMA}}}{\partial \mathbf{c}_n^*} \right] &= \\ E \left[2(|\mathbf{z}_n|^p - R_p)p|\mathbf{z}_n|^{p-1} \frac{\partial}{\partial \mathbf{c}_n^*} ((\mathbf{Y}_n^T \mathbf{c}_n)^H (\mathbf{Y}_n^T \mathbf{c}_n))^{1/2} \right] &= \\ E \left[2(|\mathbf{z}_n|^p - R_p)p|\mathbf{z}_n|^{p-1} \frac{1}{2} |\mathbf{z}_n|^{-1} \mathbf{Y}_n^* \mathbf{z}_n \right] & \quad (10) \end{aligned}$$

Dropping the expectation and simplifying, we obtain the desired tap update equation

$$\mathbf{c}_{n+1} = \mathbf{c}_n - \lambda \mathbf{Y}_n^* \mathbf{z}_n |\mathbf{z}_n|^{p-2} (|\mathbf{z}_n|^p - R_p) \quad (11)$$

where the step size λ is μ multiplied by the constant factors arising in the differentiation. It is also possible to calculate a normalized step size by setting the *a posteriori* error to zero; descriptions of normalized VCMA may be found in [19] and [10]. As with CMA, p is normally set to 2 for computational convenience.

The equalizer taps are updated every symbol period. Under these conditions, the computational cost of VCMA is surprisingly low. If all data values are kept in memory between tap updates, several computational short-cuts are possible. For many of these simplifications it is necessary to assume that the tap weights \mathbf{c}_n change slowly over time so that \mathbf{c}_n is an acceptable substitute for \mathbf{c}_{n-m} , $0 < m < N$.

The suggested steps of computation and their cost in real multiplications and additions are shown in Table 1. It will be necessary to run the recursive update steps through at least N iterations to properly initialize the matrices and vectors before starting filter adaptation. Following are some notes on the calculations:

1. The last column of \mathbf{Y} is saved for later use.
2. The matrix \mathbf{Y} is updated with new data by shifting each row to the right and discarding the last column. The first column is updated by shifting it down m elements for a T/m -spaced equalizer (see the next section), then filling in the top m positions with new data.
3. The new first element of \mathbf{z} is calculated, requiring L_c complex multiplications and $L_c - 1$ complex additions.
4. The vector norm of \mathbf{z} is denoted b , and is recursively updated here by adding the squared magnitude of the new first element and subtracting the squared magnitude of the old last element. The latter quantity was computed N iterations earlier and can be stored in memory so that only one magnitude need be calculated. The cost is two real multiplications and one real addition for the magnitude calculation, plus two real additions to find the new b .
5. The error e is real and requires just one real addition to calculate.
6. The vector \mathbf{x} represents the recursively updated product $\mathbf{Y}^*\mathbf{z}$. Two column/scalar multiplies are used—the first and last columns of \mathbf{Y} with the new first element and the old last element of \mathbf{z} , respectively, requiring $2L_c$ complex multiplications. Then, a column addition and subtraction are performed using $2L_c$ complex additions.
7. In preparation for the next iteration, \mathbf{z} is updated by shifting the elements down, discarding the last one, and placing the previously computed v in the first position.

8. Both λ and e are real, so the filter update equation now requires only one real multiplication to find λe , $2L_c$ real multiplications to multiply the complex column vector \mathbf{x} by this a real number, and L_c complex additions to update the filter taps \mathbf{c} .

The end result is a cost of $14L_c + 3$ real multiplications and $14L_c + 2$ real additions per symbol period, which is only about 75% more expensive than conventional CMA. Considering the transformation of the update equation from a scalar/vector form (CMA) to a vector/matrix form (VCMA), this is a pleasant surprise!

“Decimating the update” by updating the taps less frequently than every symbol period does not deliver the computational savings that might be expected, since some of the savings offered by the recursive updates are forfeited. For example, if the update is computed only every other symbol period, there is an inherent halving of the required computations (or so it would seem at first glance), but this is almost entirely offset because steps 3, 4, and 6 (above) now require twice as many computations.

It is also worth noting that the computational cost of VCMA is virtually independent of the block size N when implemented with the speed-ups discussed here. In Section 6 it will be shown that N may often be set lower than the theoretical requirement while preserving good performance. This does not, however, lead to a significant computational gain.

3.3 Fractionally Spaced VCMA

It is often desirable to use an equalizer with taps spaced at some fraction of the data symbol period T , most commonly $T/2$. A fractionally spaced equalizer (FSE), as this configuration is termed, has the extra degrees of freedom necessary to perform additional receiver operations such as matched filtering and adjustment of sampling phase. This makes for simpler, better receivers [17]. More recently it was shown that while an ordinary baud-spaced equalizer of

arbitrary length cannot perfectly equalize a general FIR channel, a CMA FSE of length equal to or greater than the channel delay spread can, in theory, achieve global convergence given some simple channel restrictions [13] (but see [17] for a discussion of required FSE lengths for real-world channels and hardware). A “good” FSE is therefore often shorter than a “good” baud-spaced equalizer (BSE) for a given channel. For these reasons, fractionally-spaced equalizers are now the norm in most applications.

In a FSE, the channel is sampled at the desired multiple of the symbol rate and the equalizer output is calculated only at T -spaced intervals to obtain the equalized data. Clearly, when an adaptive algorithm like CMA is used with a FSE, the tap update operation should be performed not more frequently than at time intervals of T to avoid over-constraining the filter output, since only the decimated rate $1/T$ equalizer output is of interest. When applied to VCMA, this requires some simple changes in the structure of the computations. Recall that the equalizer output vector is $\mathbf{z}_n = \mathbf{Y}_n^T \mathbf{c}_n$, with the rectangular matrix \mathbf{Y}_n given by (6) for the baud-spaced case. For a $T/2$ -spaced FSE, the samples entering the equalizer need to be $T/2$ -spaced, so each column of \mathbf{Y}_n should be $T/2$ -spaced. The equalizer output vector \mathbf{z}_n , however, needs to be T -spaced, so each row of \mathbf{Y}_n should be T -spaced. Thus, for a $T/2$ -spaced FSE, we have

$$\mathbf{Y}_n = \begin{bmatrix} y_n & y_{n-2} & \cdots & y_{n-2N+2} \\ y_{n-1} & y_{n-3} & \cdots & y_{n-2N+1} \\ \vdots & \vdots & & \vdots \\ y_{n-L_c+1} & y_{n-L_c-1} & \cdots & y_{n-L_c-2N+3} \end{bmatrix} \quad (12)$$

Simulations performed with FS-VCMA and presented in Section 6 have verified its ability to equalize several short FIR channels with a required equalizer length comparable to the channel delay spread.

Analyzing the computational cost of FS-VCMA in like manner to that of the baud-spaced case, one finds that the required steps in calculating the update are the same. The only change in the structure of the calculations is the added space between columns of \mathbf{Y}_n , and this does not affect any of the shortcuts. Therefore, FS-VCMA still requires $14L_c + 3$ real multiplications and

$14L_c + 2$ real additions *per symbol period* T . A real-world analysis would, of course, have to account for the change in required L_c when moving from a BSE to a FSE.

4 Convergence of VCMA

At the present time, relatively little has been proven about the convergence properties of VCMA. The first convergence analysis was by Yang in [19], who followed Foschini [6] in assuming an infinitely long equalizer and examining the solutions to $[\partial\mathcal{D}^{(2)}/\partial\mathbf{s}] = 0$, where $\mathcal{D}^{(2)}$ is the VCMA cost function (with $p = 2$) and \mathbf{s} is the cascaded impulse response of the channel and equalizer. The desired finding would be that the only stable equilibria correspond to global minima of $\mathcal{D}^{(2)}$ and, furthermore, that these global minima correspond to combined channel–equalizer responses \mathbf{s} of the form $[\cdots 0 0 s_k 0 0 \cdots]$, where s_k is some complex number. This type of analysis is complicated greatly by the fact that source shaping introduces correlations between neighboring symbols. For example, in shell-mapped data, individual 2-D symbols within the same length- N block may be correlated, although symbols from different blocks remain uncorrelated. Yang was able to show that ideal channel–equalizer responses of the above form were indeed global minima of $\mathcal{D}^{(2)}$. (The proof is too involved to repeat here, but can be found in [19].) It was not proven whether non-ideal responses (those having more than one nonzero element, up to the block length N) were stable or unstable equilibria.

More recently, Touzni, Tong, Casas, and Johnson presented a another study of VCMA equilibria [15]. For sub-Gaussian sources, their analysis confirms the existence of global minima which correspond to the global minima of the CMA cost function. In addition, they found conditions under which VCMA admits equalizing global minima for Gaussian and super-Gaussian sources. The characterization of other equilibria corresponding to non-zero-forcing solutions remains a challenge, however.

The close parallels between CMA and VCMA suggest that VCMA will inherit the desirable features of CMA, and simulations have supported that assumption so far.

5 Other Vector Modulus Algorithms

The constant modulus algorithm can be viewed as just one member of a larger family of blind equalization algorithms having similar cost functions and typically implemented with a stochastic gradient algorithm. For example, the reduced constellation algorithm (RCA) proposed in [9] uses the cost function

$$CF_{\text{RCA}} = E|z_n - R\text{csgn}(z_n)|^2 \quad (13)$$

where csgn is the complex sign function which returns one of the values in $\{1 + j, 1 - j, -1 + j, -1 - j\}$ according to the quadrant occupied by z_n . This cost function attempts to fit the received constellation points to the corners of a square. Although reportedly more prone to misconvergence than other algorithms in the same family, RCA is simple to implement and has the advantage of correcting for carrier phase, since unlike CMA the cost function is sensitive to constellation rotation [7]. In this section, vector versions of RCA and another algorithm are derived.

5.1 Vector Reduced Constellation Algorithm

A vector RCA (VRCA) is easily derived by using vector quantities in place of the scalar quantities in the cost function:

$$CF_{\text{VRCA}} = E|\mathbf{z}_n - R\text{csgn}(\mathbf{z}_n)|^2 \quad (14)$$

The complex sign function csgn returns a vector the same length as \mathbf{z} , working on an element-by-element basis. Whereas we can think of VCMA attempting to fit vectors of received symbols to a hypersphere in $2N$ dimensions, VRCA attempts to fit these vectors to the corners of

a $2N$ -dimensional hypercube. The update equation for VRCA (after dropping the expectation) is

$$\begin{aligned}
\mathbf{c}_{n+1} &= \mathbf{c}_n - \lambda \left[\frac{\partial CF_{\text{VRCA}}}{\partial \mathbf{c}_n^*} \right] \\
&= \mathbf{c}_n - \lambda \frac{\partial}{\partial \mathbf{c}_n^*} [(\mathbf{c}_n^H \mathbf{Y}_n^* - R \text{csgn}(\mathbf{c}_n^H \mathbf{Y}_n^*))(\mathbf{Y}_n^T \mathbf{c}_n - R \text{csgn}(\mathbf{Y}_n^T \mathbf{c}_n))] \\
&= \mathbf{c}_n - \lambda \mathbf{Y}_n^* (\mathbf{z}_n - R \text{csgn}(\mathbf{z}_n))
\end{aligned} \tag{15}$$

Computationally, VRCA is very similar to VCMA. If all data values are kept in memory between tap updates, and given the same assumption of a slowly-varying \mathbf{c}_n used for the VCMA simplifications (see Section 3), several computational short-cuts are possible. For an equalizer of length L_c , the suggested sequence of operations and the corresponding costs are illustrated in Table 2. The following comments refer to the numbered steps in the figure:

1. The last column of \mathbf{Y} is saved for later use.
2. The matrix \mathbf{Y} is updated with new data by shifting each row to the right and discarding the last column. The first column is updated by shifting it down m elements for a T/m -spaced equalizer, then filling in the top m positions with new data.
3. The new first element of \mathbf{z} is denoted v and is computed with L_c complex multiplications and $L_c - 1$ complex additions.
4. The new first element of the error vector $\mathbf{e} = \mathbf{z} - R \text{csgn}(\mathbf{z})$ is t . Since there are only four possible values of $R \text{csgn}(v)$, it is assumed here that the only calculation required to find t is one complex addition.
5. The vector x represents the recursively updated product $\mathbf{Y}^*(\mathbf{z} - R \text{csgn}(\mathbf{z}))$. Two column/scalar multiplies are used—the first and last columns of \mathbf{Y} with the new first element and the old last element of the error vector $\mathbf{e} = \mathbf{z} - R \text{csgn}(\mathbf{z})$, respectively, requiring $2L_c$ complex multiplications. Then, a column addition and subtraction are performed using $2L_c$ complex additions.

6. In preparation for the next iteration, \mathbf{e} is updated by shifting the elements down, discarding the last one, and placing the previously computed t in the first position.
7. The filter update equation requires $2L_c$ real multiplications to multiply the complex column vector \mathbf{x} by the real scalar λ , then L_c complex additions to update the filter taps \mathbf{c} .

As with VCMA, the rate of computation required for fractionally-spaced VRCA is the same per symbol period as baud-spaced VRCA, and performing the tap update less frequently does not result in significant savings, due to the loss of computational speedups.

Note that there is no particular computational advantage to VRCA over VCMA. A principal advantage of RCA over CMA, its ability to correct a carrier phase offset, may or may not be a factor with VRCA depending on the type and degree of shaping used for the signal constellation. Shaped constellations are often sufficiently circular that VRCA will not be sensitive to constellation rotation. On the other hand, in trials with lightly trellis-shaped data, enough “squareness” remained for VRCA to correct for carrier phase offsets (see the simulations in the next section).

5.2 Vector Multimodulus Algorithm

Similar derivations and experiments were carried out with a vector version of the multimodulus algorithm (MMA) described in [18]. The simplest form of MMA has the cost function

$$CF_{\text{MMA}} = E \left[(\mathcal{R}e^2\{z_n\} - R)^2 + (\mathcal{I}m^2\{z_n\} - R)^2 \right] \quad (16)$$

The update equation for MMA, found by taking the derivative with respect to \mathbf{c}_n^* and dropping the expectation, is

$$\begin{aligned} \mathbf{c}_{n+1} &= \mathbf{c}_n - \mu \left[\frac{\partial CF_{\text{MMA}}}{\partial \mathbf{c}_n^*} \right] \\ &= \mathbf{c}_n - \mu \frac{\partial}{\partial \mathbf{c}_n^*} \left[\left(\frac{1}{2} (\mathbf{y}_n^T \mathbf{c}_n + \mathbf{y}_n^H \mathbf{c}_n^*) \right)^2 - R \right]^2 + \left[\left(\frac{1}{2j} (\mathbf{y}_n^T \mathbf{c}_n - \mathbf{y}_n^H \mathbf{c}_n^*) \right)^2 - R \right]^2 \end{aligned}$$

$$= \mathbf{c}_n - \lambda \left[\mathcal{R}e\{z_n\} \mathbf{y}_n^* (\mathcal{R}e^2\{z_n\} - R) + j \mathcal{I}m\{z_n\} \mathbf{y}_n^* (\mathcal{I}m^2\{z_n\} - R) \right] \quad (17)$$

where $\lambda = 2\mu$.

This is easily generalized to a vector MMA (VMMA) by making the z_n 's vector quantities. The resulting update equation is

$$\mathbf{c}_{n+1} = \mathbf{c}_n - \lambda \mathbf{Y}_n^* \left[\mathcal{R}e\{\mathbf{z}_n\} \left(|\mathcal{R}e\{\mathbf{z}_n\}|^2 - R \right) + j \mathcal{I}m\{\mathbf{z}_n\} \left(|\mathcal{I}m\{\mathbf{z}_n\}|^2 - R \right) \right] \quad (18)$$

Simulations yielded results much like those for VRCA, and similar comments apply.

As is apparent from these examples, the vector modulus concept from VCMA has general applicability and can be used to obtain other blind equalization algorithms by suitable modifications of the original cost function.

6 Simulations

An extensive set of simulations was performed using fractionally-spaced VCMA and VRCA; some of the more interesting results are presented here.

6.1 Data Sets

Two source data sets, each of length 160,000 samples, were duplicated as many times as necessary at the input of the simulation to provide for the required number of iterations. The first data set was generated by the shell mapping algorithm using a 192-point constellation (Figure 2) divided into six rings. Each mapping operation used 36 bits to address one of 2^{36} points in 16-dimensional space, generating an output of eight complex symbols. This data is quite highly shaped with the marginal distributions having a kurtosis κ_s of about 2.64 using the definition

$$\kappa_x = \frac{E|x|^4}{(E|x|^2)^2} \quad (19)$$

where x is a real random variable (other definitions of kurtosis are in common use). Under this definition, a Gaussian distribution has $\kappa = 3$, a uniform distribution has $\kappa = 1.8$, and sub-Gaussian distributions have kurtoses between 1 and 3.

The second data set was generated by a simple trellis-shaping implementation (Forney’s “sign bit shaping” [4]) using the convolutional code $[1 + D^2, 1 + D + D^2]$ and 256-QAM for the base constellation. The Viterbi algorithm memory was allowed to span the entire 160,000 samples. The marginal distributions of this data have a kurtosis of about $\kappa_t = 2.49$ using the definition above. This data is therefore less highly shaped than the shell-mapped data.

6.2 Channels

A variety of $T/2$ -spaced channels were used in the simulations, but the specific cases presented in this chapter used the following two channels:

$$\mathbf{h}_1 = [0.23, -0.51, 0.84, 0.71, 0, -0.35, 0.61, -0.40] \quad (20)$$

$$\mathbf{h}_2 = [0.5, 0.05, 0.1 + j0.4, -0.4, 0, 0.2 - j0.2] \quad (21)$$

Note that the second channel will impose a phase shift (constellation rotation) in addition to intersymbol interference.

6.3 FS-VCMA

Figure 3 is a scatter plot showing 10,000 received symbols of the shell-mapped data after passing through channel \mathbf{h}_1 . $T/2$ -spaced VCMA with a step size of $\lambda = 10^{-6}$ and an equalizer of 8 $T/2$ -spaced taps (the same as the delay spread of the channel) was able to equalize this data. Figure 4 shows some of the equalized symbols after 80,000 iterations. Figure 5 is a plot of the equalizer taps versus time for this simulation.

6.3.1 Improvement of fractionally spaced over baud-spaced VCMA

Figure 6 illustrates the result of applying conventional baud-spaced VCMA to the same shell-mapped data and channel as just described. The data was first passed through the $T/2$ -spaced channel, then decimated to T -spaced samples prior to equalization. Six hundred thousand iterations with a 20-tap equalizer were required to achieve the poor equalization shown here, demonstrating the ability of fractionally spaced VCMA to obtain good equalization with many fewer taps than baud-spaced VCMA, at least over some channels.

6.3.2 Trellis-shaped data

Figure 7 is a plot of equalized data points after 60,000 iterations of $T/2$ -spaced VCMA on the trellis-mapped data through channel \mathbf{h}_1 . The step size was $\lambda = 10^{-6}$, and the block size N was set to 8 (the same as for the shell-mapped data). This clearly shows the ability of VCMA to equalize data from shaped sources that did not use shell mapping as the shaping method; this is in spite of the conceptual link between shell mapping and VCMA discussed in Section 2.

6.3.3 Block size N

When equalizing shell-mapped data, the correct value of the block size N is obvious from the shell-mapping algorithm. When other shaping techniques are employed, the appropriate block size is not as clear. In either case, it is natural to ask whether smaller values for N might work, and if so, how well.

Many simulations were performed using channel \mathbf{h}_1 and $T/2$ -spaced VCMA with both shell-mapped and trellis-shaped data to determine the effect of the block size N on the number of iterations required for acceptable equalization. Acceptable equalization was determined by inspection of a scatter plot of the received points. The results are shown in Table 3, from which it

is clear that N is quite a flexible parameter. Figure 8 is a typical result obtained using VCMA when N is too small.

Note that the trellis-shaped data is sufficiently sub-Gaussian that all of the block sizes tested were acceptable. Comparison tests using $T/2$ -spaced CMA either failed completely or had a tendency to drift in and out of good equalization. It should also be noted that these results are just one example; similar results would not necessarily be obtained if a different channel were used.

6.3.4 Block offset

In many practical situations using shell mapping, the receiver may not know the frame boundaries where one block of N symbols ends and another begins. It would therefore be very desirable if VCMA could operate with arbitrary synchronization to the incoming data.

Simulations were performed over channel \mathbf{h}_2 with the shell-mapped data having a block size of 8 complex symbols and with N set to 8 in the equalizer as well. In every case, $T/2$ -spaced VCMA achieved equalization regardless of the block offset used.

6.4 FS-VRCA

Figure 9 shows the equalized data points resulting from a $T/2$ -spaced vector reduced constellation algorithm (VRCA) trial of 480,000 iterations with step size $\lambda = 10^{-5}$ and block size $N = 8$ using the trellis-shaped data set. Comparing this with Figure 10, an identical simulation using FS-VCMA, we see that the rotation-sensitive nature of the VRCA cost function allowed the algorithm to correct for the constellation rotation which VCMA ignores.

7 Conclusions

The vector constant modulus algorithm, first developed in a baud-spaced equalizer framework with shell-mapped source data, has been shown effective in other scenarios as well. Its ability to work with fractionally spaced equalizers and its modest computational cost over CMA make it viable for use in the increasing number of data communication applications which use source shaping. Indeed, the vector modulus algorithms continue to be the only practical techniques available for the blind equalization of shaped data.

The success of VCMA with both shell-mapped and trellis-shaped data is encouraging. The key concept of using a vector modulus in order to present uniformly-distributed data to the underlying algorithm seems to be valid even when the source data is not generated from a uniform constellation of a particular finite dimension. This suggests that VCMA may remain a viable algorithm for equalizing shaped data regardless of the specific shaping techniques used in the future.

This concept is sufficiently general to be applied to other algorithms based on constant modulus criteria, including the reduced constellation algorithm (RCA) and the multimodulus algorithm (MMA). This yields the vector modulus algorithms VRCA and VMMA. Both algorithms perform about as well as VCMA, but VRCA may be more applicable when the signal constellation has enough asymmetry to permit automatic correction of a carrier phase offset, thus avoiding the added complexity of a de-rotator block after the equalizer.

Future study into the convergence properties of VCMA and related algorithms would still be desirable, since no proof yet exists showing the global convergence of VCMA. Simulations, however, suggest that it shares similar convergence properties with the well-studied CMA. Further simulations would also be useful to study the performance of VCMA when used with multicarrier signals like OFDM and DMT, which, like single-carrier shaped modulation, have

almost Gaussian marginal distributions but decidedly sub-Gaussian vector block distributions.

References

- [1] C.K. Chan and J.J. Shynk. Stationary points of the constant modulus algorithm for real Gaussian signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(12):2176–2181, December 1990.
- [2] T. Endres. *Equalizing with fractionally-spaced constant modulus and second-order-statistics blind receivers*. PhD thesis, Cornell University, Ithaca, New York, May 1997.
- [3] M.V. Eyuboglu, G.D. Forney, Jr., P. Dong, and G. Long. Advanced modulation techniques for V.Fast. *European Transactions on Telecommunications*, 4(3):9–22, May–June 1993.
- [4] G.D. Forney, Jr. Trellis shaping. *IEEE Transactions on Information Theory*, 38(2):281–300, March 1992.
- [5] G.D. Forney, Jr. and L.-F. Wei. Multidimensional constellations—part i: Introduction, figures of merit, and generalized cross constellations. *IEEE Journal on Selected Areas in Communications*, 7(6):877–892, August 1989.
- [6] G.J. Foschini. Equalizing without altering or detecting data. *AT&T Technical Journal*, 64:1885–1911, October 1985.
- [7] L.M. Garth, J. Yang, and J. Werner. Blind start-up for VDSL systems. Unpublished manuscript.
- [8] D.N. Godard. Self-recovering equalization and carrier tracking in two-dimensional data communication systems. *IEEE Transactions on Communications*, 28(11):1867–1875, November 1980.
- [9] D.N. Godard and P.E. Thirion. Method and device for training an adaptive equalizer by means of an unknown data signal in a QAM transmission system. U.S. Pat. 4 227 152, October 1980.
- [10] M.A. Haun. The fractionally spaced vector constant modulus algorithm. Master’s thesis, University of Illinois at Urbana-Champaign, 1999.
- [11] S. Haykin. *Adaptive filter theory*. Prentice Hall, Englewood Cliffs, NJ, 1991.
- [12] R. Laroia, N. Farvardin, and S.A. Tretter. On optimal shaping of multidimensional constellations. *IEEE Transactions on Information Theory*, 40(4):1044–1056, July 1994.
- [13] Y. Li and Z. Ding. Global convergence of fractionally spaced Godard (CMA) adaptive equalizers. *IEEE Transactions on Signal Processing*, 44(4):818–826, April 1996.
- [14] O. Shalvi and E. Weinstein. New criteria for blind deconvolution of nonminimum phase systems (channels). *IEEE Transactions on Information Theory*, 36:312–321, March 1990.
- [15] A. Touzni, L. Tong, R.A. Casas, and C.R. Johnson, Jr. Vector-CM stable equilibrium analysis. To appear in signal processing letters.
- [16] J.R. Treichler and B.G. Agee. A new approach to multipath correction of constant modulus signals. *IEEE Transactions on Communications*, 31(2):459–472, April 1983.

- [17] J.R. Treichler, I. Fijalkow, and C.R. Johnson, Jr. Fractionally spaced equalizers: How long should they really be? *IEEE Signal Processing Magazine*, 13(3):65–81, May 1996.
- [18] J. Yang, J.J. Werner, and G.A. Dumont. The multimodulus blind equalization algorithm. In *Proc. Thirteenth Intl. Conf. on Digital Signal Processing*, pages 127–130, Santorini, Greece, July 1997.
- [19] V.Y. Yang. A vector constant modulus algorithm for shaped constellation equalization. Master's thesis, University of Illinois at Urbana-Champaign, 1997.
- [20] V.Y. Yang and D.L. Jones. A vector constant modulus algorithm for shaped constellation equalization. *IEEE Signal Processing Letters*, 5(4):89–91, April 1998.

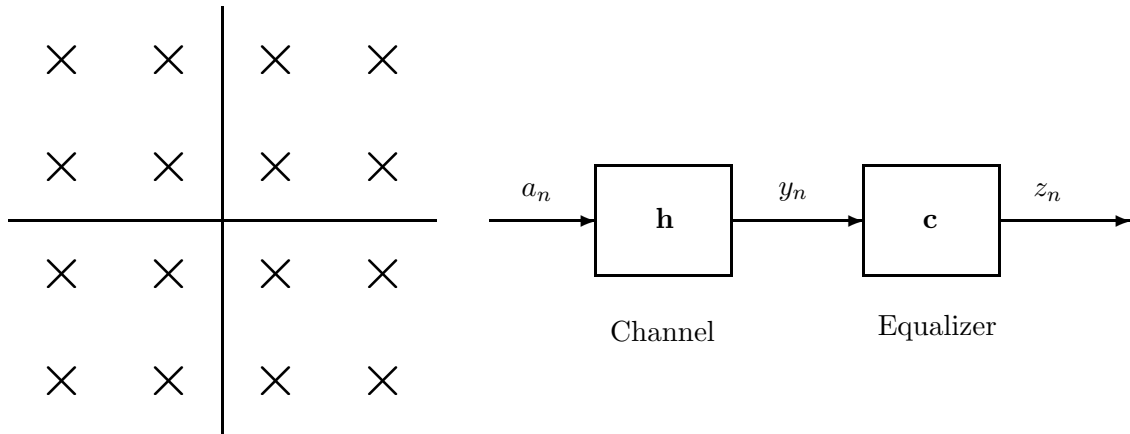


Figure 1: Communications system model. A sequence of symbols $\{a_n\}$ from a set in the complex plane (16-QAM is shown here) are convolved first with the channel impulse response \mathbf{h} and then with the equalizer \mathbf{c} at the receiver.

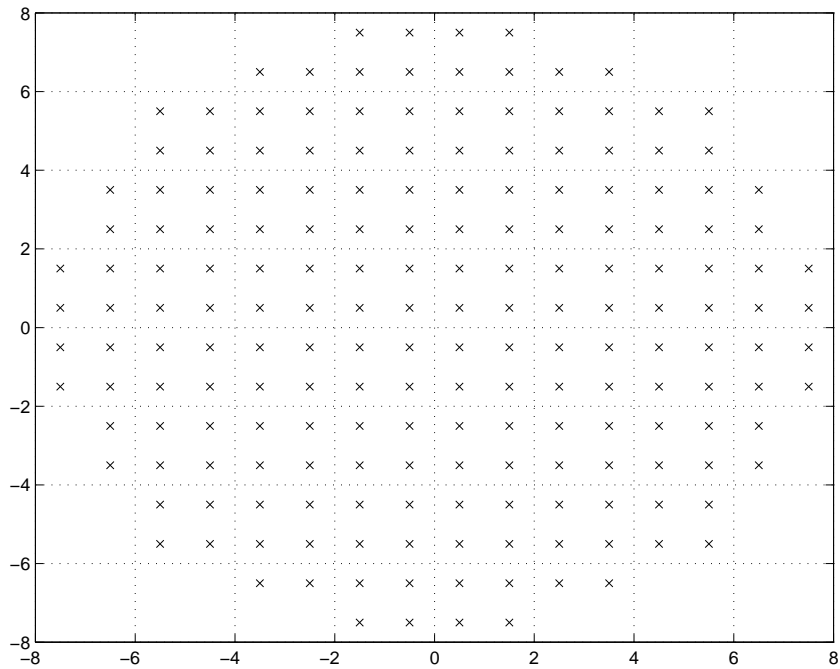


Figure 2: 192-point constellation used for shell mapping.

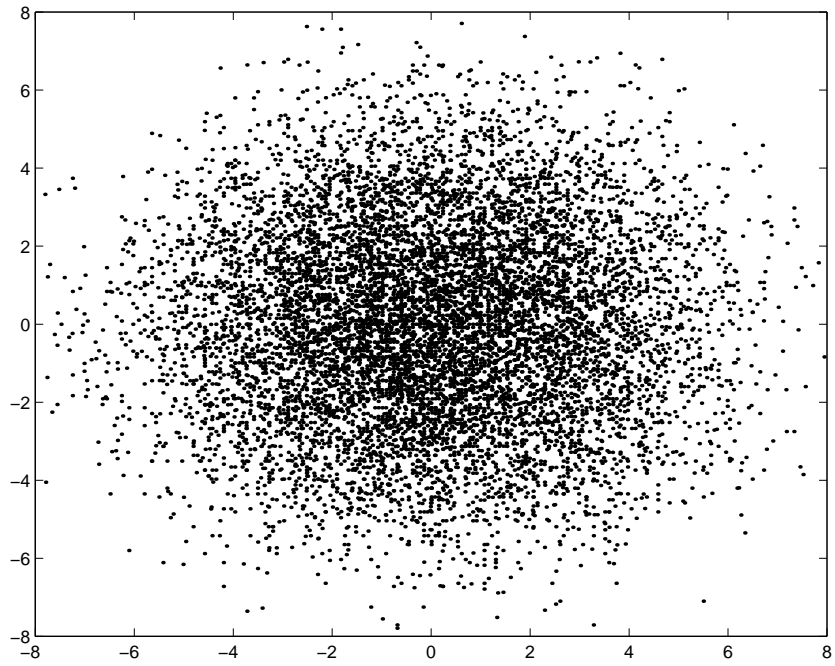


Figure 3: Shell-mapped data after passage through channel \mathbf{h}_1 .

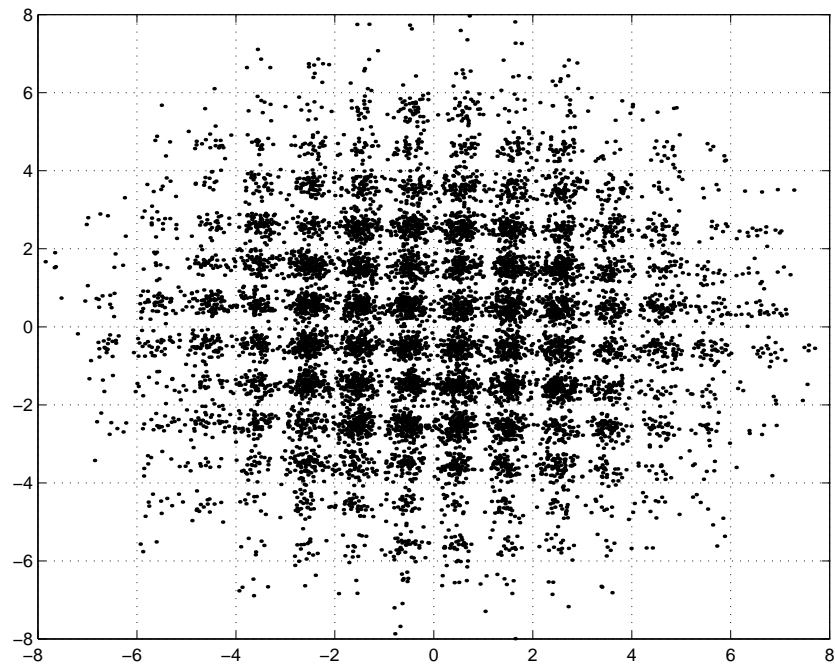


Figure 4: FS-VCMA equalizer output for input data in Figure 3.

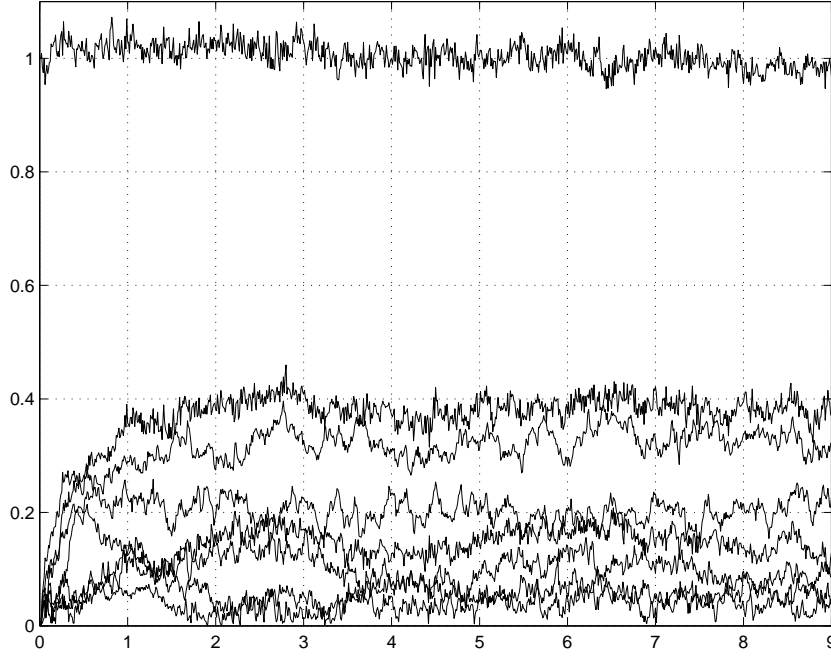


Figure 5: FS-VCMA equalizer tap magnitudes vs. number of iterations ($\times 10^4$).

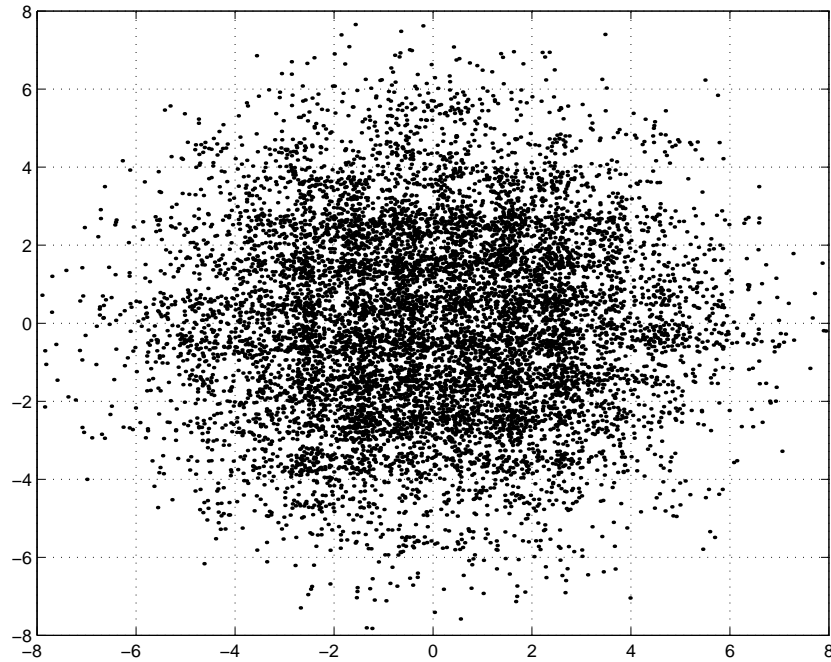


Figure 6: Baud-spaced VCMA equalizer output for input data in Figure 3.

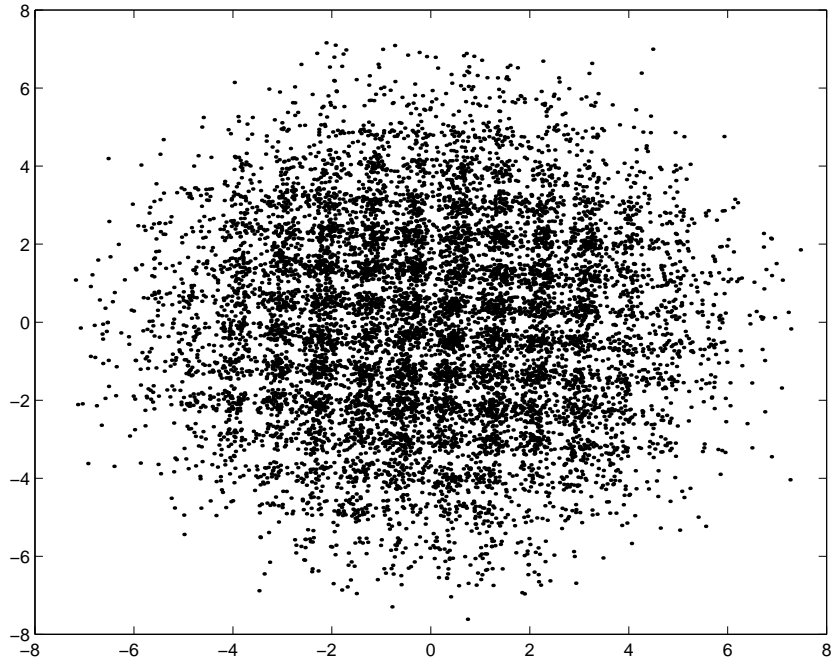


Figure 7: Equalized trellis-shaped data through channel \mathbf{h}_1 using FS-VCMA.

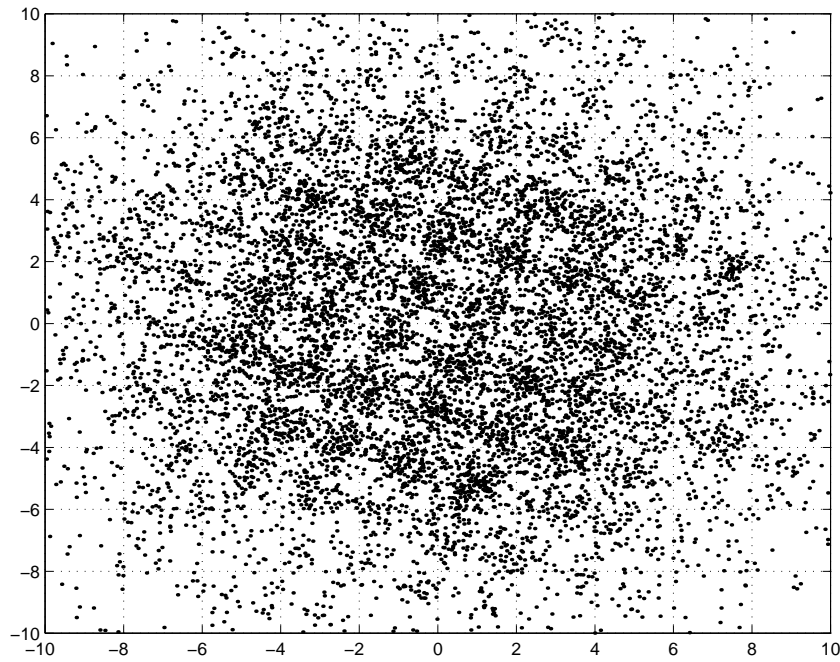


Figure 8: Poor equalization of shell-mapped data with FS-VCMA when $N = 2$.

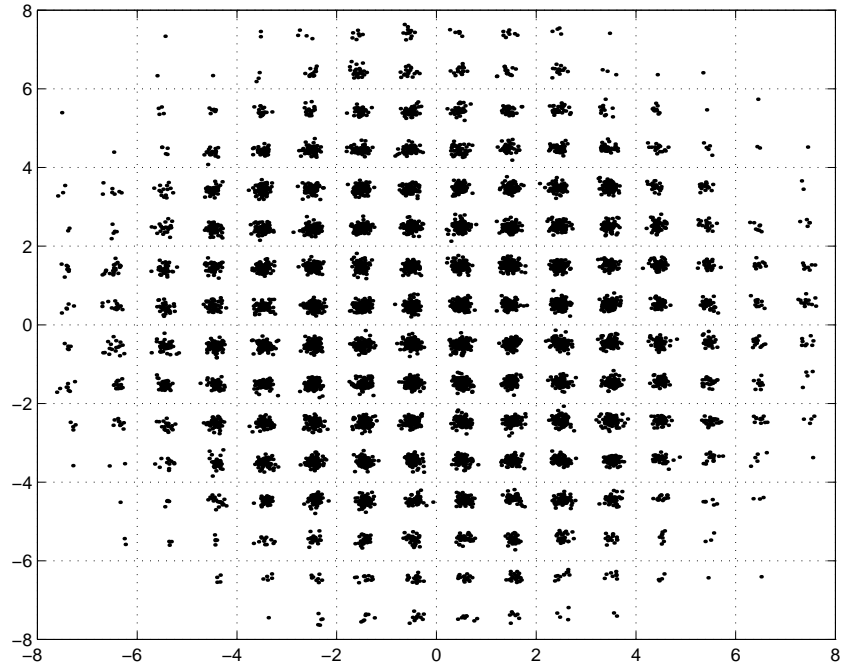


Figure 9: Equalized trellis-shaped data through channel h_2 using FS-VRCA.

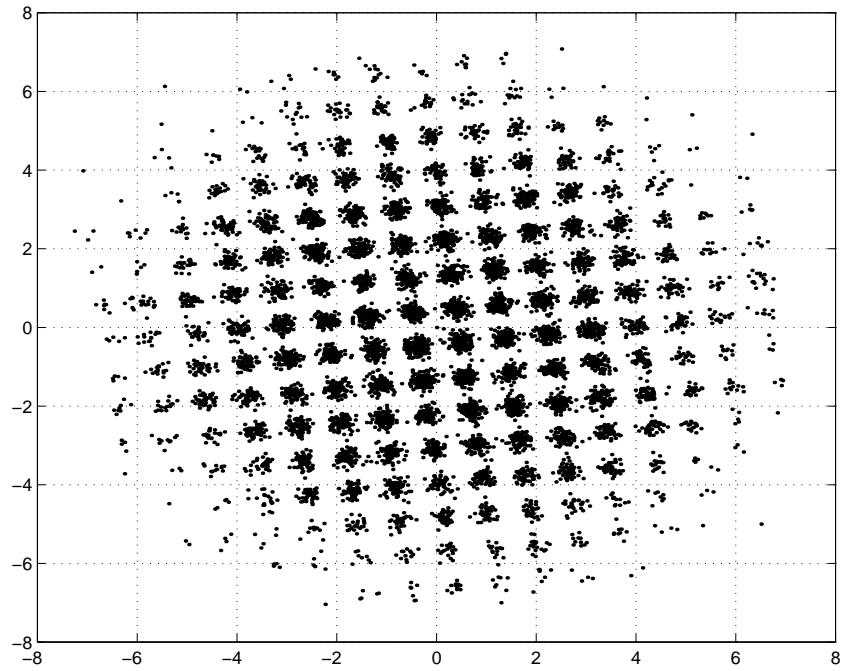


Figure 10: Equalized trellis-shaped data through channel h_2 using FS-VCMA.

Table 1: Computational steps and their associated costs for one iteration of VCMA.

Operation	Real multiplications	Real additions
1. $\mathbf{u} = \mathbf{y}_N$		
2. ⟨Update \mathbf{Y} — see text⟩		
3. $v = \mathbf{y}_1^T \mathbf{c}$	$4L_c$	$4L_c - 2$
4. $b = b + v ^2 - z_N ^2$	2	3
5. $e = b - R_2$		1
6. $\mathbf{x} = \mathbf{x} + \mathbf{y}_1^* v - \mathbf{u}^* z_N$	$8L_c$	$8L_c$
7. $\mathbf{z} = [v \ z_1 \ z_2 \ \cdots \ z_{N-1}]^T$		
8. $\mathbf{c} = \mathbf{c} - \lambda \mathbf{x} e$	$2L_c + 1$	$2L_c$
Total:	$14L_c + 3$	$14L_c + 2$

$$\left(\mathbf{Y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_N \end{bmatrix} ; \mathbf{z} = [z_1 \ z_2 \ \cdots \ z_N]^T \right)$$

Table 2: Computational steps and their associated costs for one iteration of VRCA.

Operation	Real multiplications	Real additions
1. $\mathbf{u} = \mathbf{y}_N$		
2. ⟨Update \mathbf{Y} ⟩		
3. $v = \mathbf{y}_1^T \mathbf{c}$	$4L_c$	$4L_c - 2$
4. $t = v - R \text{csgn}(v)$		2
5. $\mathbf{x} = \mathbf{x} + \mathbf{y}_1^* t - \mathbf{u}^* e_N$	$8L_c$	$8L_c$
6. $\mathbf{e} = [t \ e_1 \ e_2 \ \cdots \ e_{N-1}]^T$		
7. $\mathbf{c} = \mathbf{c} - \lambda \mathbf{x}$	$2L_c$	$2L_c$
Total:	$14L_c$	$14L_c$

$$\left(\mathbf{Y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_N \end{bmatrix} ; \mathbf{e} = [e_1 \ e_2 \ \cdots \ e_N]^T \right)$$

Table 3: Required number of iterations of FS-VCMA (in thousands) vs. block size N .

Block length $N =$	8	7	6	5	4	3	2
Shell mapped data	20	30	30	40	40	50	≈ 200 (poor convergence)
Trellis shaped data	25	30	30	30	40	40	40

List of Figures

1	Communications system model. A sequence of symbols $\{a_n\}$ from a set in the complex plane (16-QAM is shown here) are convolved first with the channel impulse response \mathbf{h} and then with the equalizer \mathbf{c} at the receiver.	23
2	192-point constellation used for shell mapping.	23
3	Shell-mapped data after passage through channel \mathbf{h}_1	24
4	FS-VCMA equalizer output for input data in Figure 3.	24
5	FS-VCMA equalizer tap magnitudes vs. number of iterations ($\times 10^4$).	25
6	Baud-spaced VCMA equalizer output for input data in Figure 3.	25
7	Equalized trellis-shaped data through channel \mathbf{h}_1 using FS-VCMA.	26
8	Poor equalization of shell-mapped data with FS-VCMA when $N = 2$	26
9	Equalized trellis-shaped data through channel \mathbf{h}_2 using FS-VRCA.	27
10	Equalized trellis-shaped data through channel \mathbf{h}_2 using FS-VCMA.	27

List of Tables

1	Computational steps and their associated costs for one iteration of VCMA.	28
2	Computational steps and their associated costs for one iteration of VRCA.	28
3	Required number of iterations of FS-VCMA (in thousands) vs. block size N	28