A Unified Approach to Selecting Optimal Step Lengths for Adaptive Vector Quantizers

Lachlan L. H. Andrew and Marimuthu Palaniswami, *Senior Member, IEEE*

***Abstract*—This paper presents expressions for the optimal step length to use when training a vector quantizer by stochastic approximation. By treating each update as an estimation problem, it provides a unified framework covering both batch and incremental training, which were previously treated separately, and extends existing results to the semibatch case. In addition, the new results presented here provide a measurable improvement over results which were previously thought to be optimal.**

I. Introduction

VECTOR quantization is an important technique in data compression, in which groups of continuous variables are discretized as a single vector rather than individually. An N-level k-dimensional vector quantizer (VQ) is a mapping $V : \Re^k \mapsto C \subset \Re^k$, where the number of members of C is N. Elements $\hat{\mathbf{c}} \in C$ are called reconstruction vectors or levels, and C is called the codebook. The mapping is usually defined to minimize some expected distortion measure, $E[d(\mathbf{x}, V(\mathbf{x}))]$, typically the mean square error (MSE) given by $d(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T(\mathbf{x}-\mathbf{y})$. With nearest neighbor mapping and a tie-break rule, C uniquely determines V.

Selection of C is, effectively, a search in k-dimensional space for a locally optimal codebook. As with most search algorithms, VQ design usually proceeds by finding the direction in which to move and then moves a certain distance in that direction. The direction is usually toward the centroid of one or more vectors which are represented by the particular level in question. This paper is concerned with finding the appropriate distance to move, i.e., the step length.

Earlier work on finding optimal updates includes the seminal papers by Lloyd [1] and Linde, Buzo and Gray [2], which deal exclusively with the case of batch training, and the work of Wu and Fallside [3] which extends the results to the incremental case. In the batch case [1], [2] all of the training data [or the exact analytic probability density function (pdf)] is assumed to be known for each update. Thus, the optimal update is to replace the old reconstruction level with the centroid of those vectors which map to it. Incremental training updates after each single vector is presented [3], which means that the "centroid" (the vector itself) is a *noisy* estimate of the ideal reconstruction level. Some form of averaging is required, which usually takes the form of an autoregressive update which replaces the reconstruction level by the weighted sum of the old level and the training vector. In [3], the weights were chosen such that the reconstruction level was the mean of all of the training vectors which have ever been represented by that level, but it will be shown in Section III-B that this is not generally optimal. This paper proposes a method for determining optimal weights such that the new level is the minimum variance unbiased estimator (MVUE) of the centroid of the previous partition. The approach leads to a general expression which is valid for both batch and incremental training.

The rest of this paper is organized as follows. Section II describes a general update procedure for VQ design. Section III derives expressions for the optimal step length applicable to batch, semibatch, and incremental updates. Section IV presents experimental results justifying the approximations made in some special cases. Section V contains concluding remarks.

II. General Training Framework

Many different schemes have been proposed for VQ codebook design, but below is a general description of a training scheme which includes the LBG algorithm [2] and incremental training as special cases. This description also serves to introduce the notation to be used. It is assumed that training data is drawn from some stationary probability distribution.

Training proceeds in batches, not necessarily consisting of the same data nor necessarily of the same size. Training starts with an initial codebook, consisting of N vectors $\hat{\mathbf{c}}_i(0)$, $i = 1, \cdots, N$. In the tth batch, the current codebook $\{\hat{\mathbf{c}}_i(t)\}$ defines a partition $\{P_i(t)\}$, where $P_i(t)$ is the set of all vectors closer to $\hat{\mathbf{c}}_i(t)$ than to any $\hat{\mathbf{c}}_j(t)$, with $j \neq i$ according to the chosen distortion function. Note that $P_i(t)$ changes each batch as the $\hat{\mathbf{c}}_i(t)$ changes; one of the major advantages of the proposed approach over previous work [3] is that this change is taken into account. For each region $P_i(t)$, there is a vector $\mathbf{c}_i^+(t+1)$, called the true centroid of $P_i(t)$, such that approximating each vector in $P_i(t)$ by $\mathbf{c}_i^+(t+1)$ produces the least distortion (see Fig. 1). This is generally not known and must be approximated from the vectors presented in a training step. In the case of the MSE, it is the mean of all of the vectors in $P_i(t)$ weighted according to their probabilities. (Lloyd's necessary condition for an optimal quantizer [2] can be expressed as $\forall i, \hat{\mathbf{c}}_i(t) = \mathbf{c}_i^+(t)$.) In the tth training step, a batch of training vectors is presented to the suboptimal quantizer given by the $\hat{\mathbf{c}}_i(t)$, and the training vectors are divided up into the partitions, $P_i(t)$. The number of vectors falling in $P_i(t)$ will be denoted $N_i(t)$ and their centroid will be denoted $\mathbf{c}_i(t)$. The actual training step consists of finding a

Paper approved by E. Ayanoglu, the Editor for Communication Theory and Coding Application of the IEEE Communications Society. Manuscript received March 11, 1995; revised June 27, 1995.

The authors are with the Department of Electrical and Electronic Engineering, University of Melbourne, Parkville, 3052, Victoria, Australia

Publisher Item Identifier S 0090-6778(96)03154-6.

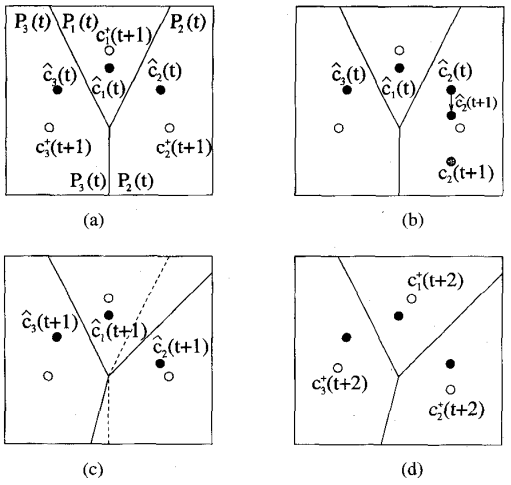


Fig. 1. Stages in a single update of vector 2. (a) Notation. (b) Approximation $\hat{\mathbf{c}}_2(t+1)$ moves toward sample centroid $\mathbf{c}_2(t+1)$. (c) Partitions move to reflect the changed approximations. (d) True centroids move to reflect the changed partitions.

new approximation $\hat{\mathbf{c}}_i(t+1)$ to the true centroid, $\mathbf{c}_i^+(t+1)$, of the previous partition. Note that although each step aims to approximate $\mathbf{c}_i^+(t)$, there is no fixed "optimum" value of $\hat{\mathbf{c}}_i$. Rather, it is required that $\hat{\mathbf{c}}_i(t)$ be well matched to $P_i(t)$. Let the change in optimal centroid between the tth and $t+1$st batches be denoted by $\mathbf{n}_i(t)$. Let the error in approximating $\mathbf{c}_i^+(t)$ by $\hat{\mathbf{c}}_i(t)$ be $\epsilon_i(t)$ and let the sampling error in $\mathbf{c}_i(t)$ be $\mathbf{e}_i(t)$, so

$$\mathbf{n}_i(t) = \mathbf{c}_i^+(t) - \mathbf{c}_i^+(t-1) \tag{1}$$

$$\epsilon_i(t) = \hat{\mathbf{c}}_i(t) - \mathbf{c}_i^+(t) \tag{2}$$

$$\mathbf{e}_i(t) = \mathbf{c}_i(t) - \mathbf{c}_i^+(t). \tag{3}$$

Then the training algorithm is given by

$$\hat{\mathbf{c}}_i(t) = a_i(t)\hat{\mathbf{c}}_i(t-1) + b_i(t)\mathbf{c}_i(t) \tag{4}$$

where $a_i(t)$ and $b_i(t)$ are parameters to be chosen and $b_i(t)$ is called the step length for the ith codevector in the tth batch.

Clearly, the above training includes the generalized Lloyd update [2] as a special case with $a_i(t) = 0$ and $b_i(t) = 1$.

III. Optimal Step Length

In this section, the above algorithm is treated as a problem of estimating the true value of $\mathbf{c}_i^+(t)$ from the noisy observations $\hat{\mathbf{c}}_i(t-1)$ and $\mathbf{c}_i(t)$. The aim is to find $a_i(t)$ and $b_i(t)$, such that $\hat{\mathbf{c}}_i(t)$ is the MVUE of $\mathbf{c}_i^+(t)$.

The MVUE was chosen over other "optimal" estimators (such as the maximum-likelihood estimator) because it is simple and well matched to the MSE criterion. If $p(\mathbf{x})$ denotes the probability density function of the source, then by (2) the expected distortion for a given reconstruction vector, $\hat{\mathbf{c}}_i$, is

$$\int_{\mathbf{x}\in P_i} p(\mathbf{x})(\mathbf{x}-\hat{\mathbf{c}}_i)^2\,d\mathbf{x} = \int_{\mathbf{x}\in P_i} p(\mathbf{x})(\mathbf{x}-\mathbf{c}_i^+)^2\,d\mathbf{x} + 2\epsilon_i \cdot \int_{\mathbf{x}\in P_i} p(\mathbf{x})(\mathbf{x}-\mathbf{c}_i^+)\,d\mathbf{x} + {\epsilon_i}^2 \int_{\mathbf{x}\in P_i} p(\mathbf{x})\,d\mathbf{x}.$$

But the first term is independent of the estimate, while the second term is zero by the definition of $\mathbf{c}_i^+$. Thus, the estimator which minimizes ${\epsilon_i}^2$ will also minimize the MSE. Since the expected value of ${\epsilon_i}^2$ is equal to the variance if $\hat{\mathbf{c}}_i$ is unbiased, the MVUE gives the optimal step length in the case of the MSE.

It will be assumed that all of the uncertainties $\mathbf{n}_i(t)$, $\mathbf{e}_i(t)$, and $\epsilon_i(t-1)$ have zero mean and are uncorrelated with each other. Now by (1)–(3)

$$\begin{aligned}\hat{\mathbf{c}}_i(t) &= a_i(t)\hat{\mathbf{c}}_i(t-1) + b_i(t)\mathbf{c}_i(t)\\ &= a_i(t)\big(\epsilon_i(t-1) + \mathbf{c}_i^+(t) - \mathbf{n}_i(t)\big)\\ &\quad + b_i(t)\big(\mathbf{c}_i^+(t) + \mathbf{e}_i(t)\big)\\ &= (a_i(t) + b_i(t))\mathbf{c}_i^+(t)\\ &\quad + a_i(t)(\epsilon_i(t-1) - \mathbf{n}_i(t)) + b_i(t)\mathbf{e}_i(t).\end{aligned}$$

Clearly, this is unbiased if $a_i(t) = 1 - b_i(t)$, since $\epsilon_i(t-1)$, $\mathbf{n}_i(t)$, and $\mathbf{e}_i(t)$ all have zero mean and are uncorrelated with $a_i(t)$ and $b_i(t)$. Hence, it remains to find $b_i(t)$ which minimizes the variance of $\hat{\mathbf{c}}_i(t)$, or equivalently that of $\epsilon_i(t) = a_i(t)(\epsilon_i(t-1) - \mathbf{n}_i(t)) + b_i(t)\mathbf{e}_i(t)$. This variance is given by

$$\sigma^2_{\epsilon_i(t)} = (1-b_i(t))^2\left(\sigma^2_{\epsilon_i(t-1)} + \sigma^2_{\mathbf{n}_i(t)}\right) + b_i(t)^2\sigma^2_{\mathbf{e}_i(t)}$$

where σ_x^2 denotes the variance of a random variable x. This follows from the assumption of uncorrelation. Setting the derivative of $\sigma^2_{\epsilon_i(t)}$ with respect to $b_i(t)$ to zero gives the optimal step length

$$b_i(t) = \frac{\sigma^2_{\epsilon_i(t-1)} + \sigma^2_{\mathbf{n}_i(t)}}{\sigma^2_{\epsilon_i(t-1)} + \sigma^2_{\mathbf{n}_i(t)} + \sigma^2_{\mathbf{e}_i(t)}} \tag{5}$$

for the MVUE, giving a variance of

$$\begin{aligned}\sigma^2_{\epsilon_i(t)} &= \frac{\sigma^2_{\mathbf{e}_i(t)}\left(\sigma^2_{\epsilon_i(t-1)} + \sigma^2_{\mathbf{n}_i(t)}\right)}{\sigma^2_{\epsilon_i(t-1)} + \sigma^2_{\mathbf{n}_i(t)} + \sigma^2_{\mathbf{e}_i(t)}}\\ &= \sigma^2_{\mathbf{e}_i(t)} b_i(t).\end{aligned} \tag{6}$$

Equation (5) is the exact optimal step length (in the sense of the MVUE) without any approximations, but for it to be of any use, estimates for the required variances must be available. Below are some special cases where the variances are known or may be approximated. An important case not considered below is that of a VQ which adapts to a slowly varying nonstationary source. In this case, $\mathbf{n}_i(t)$ must consider both changes in $\hat{\mathbf{c}}_i(t)$ and changes due to the changed distribution. Another case is when the structure of the VQ changes during training, such as when the size of the codebook grows due to splitting [2], [4]. Both of these are the subject of further investigation by the authors.

A. Full Batch Learning: The LBG Algorithm

Consider the case in which the VQ is being trained from a single finite training set which is presented in its entirety for each batch. In that case, the best estimate of the underlying distribution is simply the distribution of the training set. Thus, the sampling uncertainty in $\mathbf{c}_i(t)$ is zero, since the entire training set is used. This gives $\sigma^2_{\mathbf{e}_i(t)} = 0$ so (5) becomes

$b_i(t) = 1$. In other words, after each presentation, each codeword is simply replaced by the centroid of the training vectors in its partition. This is simply the LBG algorithm [2], which has long been known to be the optimal learning rule for full batch learning. This is a useful check on the validity of expression (5).

B. Incremental Training with Static Partitions

An earlier work [3, eq. (21)] indicated that for the nth update of $\hat{\mathbf{c}}_i$, the optimal value of $b_i(t)$ is $b_i(t) = 1/n$. The basis for stating this is that a necessary condition for a VQ to be optimal is that $\hat{\mathbf{c}}_i(t)$ be the arithmetic mean of all vectors in $P_i(t)$. If $b_i(t)$ has the above form, then the estimate will be the arithmetic mean of all those which have mapped to the ith vector during training. In the case of semibatch updates, the requirement generalizes to

$$\hat{\mathbf{c}}_i(t) = \frac{\sum_{\tau \le t} N_i(\tau)\mathbf{c}_i(\tau)}{\sum_{\tau \le t} N_i(\tau)}. \tag{7}$$

However, since the partitions change, these need not all still be in $P_i(t)$, but must merely have been in $P_i(\tau)$ for some $\tau \le t$.

This assumption, that $P_i(t) = P_i$ is independent of t, can be made explicit by setting $\mathbf{n}_i(t) = 0$ in the analysis of Section III. Thus, and by stationarity, all of the input vectors from a given partition will be independent and identically distributed (i.i.d.). When the input vector is in $P_i(t)$, $\sigma^2_{\mathbf{e}_i(t)} = \sigma^2_{\mathbf{e}_i}$ is independent of t. If the input vector in the tth batch is not in $P_i(t)$, then no information about $\mathbf{c}_i^+(t)$ is given, so $\sigma^2_{\mathbf{e}_i(t)}$ can be regarded as infinite, giving $b_i(t) = 0$. Let $b_i{}^n$ denote $b_i(t)$ the nth time $\hat{\mathbf{c}}_i$ is updated. The initial estimate of $\hat{\mathbf{c}}_i(0)$ before any updates contains no information and can be assumed to have infinite variance. Thus, the initial step length will be $b_i{}^1 = 1$ from (5) giving the variance after the first update to be $\sigma^2_{\mathbf{e}_i}$. From (5) and (6), it is clear that if the single training vector in batch t is in $P_i(t-1)$, then $b_i{}^n = b_i{}^{n-1}/(b_i{}^{n-1}+1)$; it then follows by induction on n that for all subsequent t, $b_i{}^n = 1/n$ and $\sigma^2_{\epsilon_i(t)} = \sigma^2_{\mathbf{e}_i}/n$.

C. Training from a Random Sequence

This section will consider training from a stream of samples, rather than a single batch. A small number of samples (possibly one) is collected and then presented as a batch. There are several possible reasons for doing this rather than complete batch updates. For example, not all of the training data may be available at the start of training, or there may be insufficient storage to store the entire set. This is the case in adaptive systems [5], [6]. Another reason is that the stochastic nature of incremental training can sometimes avoid minima which trap batch algorithms [7]. It may also be faster than fully incremental training, since the weights do not need to be updated as often.

In order to estimate the optimal step length, some estimate must be found for $\sigma^2_{\mathbf{n}_i(t)}$, the variance of the amount by which the true centroid changes between batches, and $\sigma^2_{\mathbf{e}_i(t)}$, the sampling variance of the centroid of the training sample. Since $\mathbf{c}_i(t)$ is the sum of $\mathbf{c}_i^+(t)$, which is not a random variable, and $\mathbf{e}_i(t)$, which is, the variance of $\mathbf{e}_i(t)$ is equal to that of

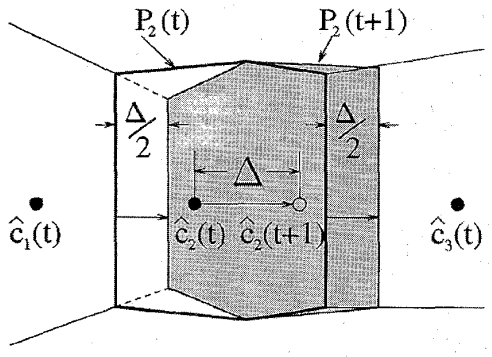


Fig. 2. Change in a partition when a reconstruction vector changes. When the approximation $\hat{\mathbf{c}}_2(t)$ moves by Δ, the Voronoi region is translated by $\Delta/2$ in the same direction and distorted slightly.

$\mathbf{c}_i(t)$. Since $\mathbf{c}_i(t)$ is simply the mean of $N_i(t)$ i.i.d. vectors, this sampling variance is given by $\sigma^2_{\mathbf{e}_i(t)} = \sigma^2_{\mathbf{e}_i}/N_i(t)$, where $\sigma^2_{\mathbf{e}_i}$ denotes the variance of a *single* training sample from the region $P_i(t)$, which is assumed to be roughly independent of t.

Since the boundaries of $P_i(t)$ are half way between $\hat{\mathbf{c}}_i(t)$ and the adjacent centroid estimates, it will be assumed that the change in *true* centroid of $P_i(t)$, $\mathbf{c}_i^+$, is half of the change in the *approximate* centroid, $\hat{\mathbf{c}}_i$ (see Fig. 2). This approximation assumes that the probability density is locally approximately constant and ignores changes in $\hat{\mathbf{c}}_j$ for $j \neq i$, but gives sufficiently good results in practice. In general, assume that the change in true centroid is $1/k$ times the change in approximate centroid, for some $k > 1$ rather than necessarily $k = 2$, so

$$\begin{aligned}\mathbf{n}_i(t+1) &\approx \frac{1}{k}(\hat{\mathbf{c}}_i(t) - \hat{\mathbf{c}}_i(t-1)) \\ &= \frac{1}{k} b_i(t)(\mathbf{c}_i(t) - \hat{\mathbf{c}}_i(t-1)) \\ &= \frac{1}{k} b_i(t)(\mathbf{e}_i(t) + \mathbf{n}_i(t) - \epsilon_i(t-1))\end{aligned}$$

giving

$$\sigma^2_{\mathbf{n}_i(t+1)} = \left(\frac{b_i(t)}{k}\right)^2 \left(\sigma^2_{\mathbf{e}_i(t)} + \sigma^2_{\mathbf{n}_i(t)} + \sigma^2_{\epsilon_i(t-1)}\right).$$

Unfortunately, this expression leads to a complicated joint recurrence relation between $\sigma^2_{\mathbf{n}_i(t)}$ and $\sigma^2_{\epsilon_i(t)}$. If it is assumed that $\sigma^2_{\mathbf{n}_i(t)}$ is negligible compared to $\sigma^2_{\mathbf{e}_i}$ and $\sigma^2_{\epsilon_i(t-1)}$, and that $\sigma^2_{\epsilon_i(t-1)} \approx \sigma^2_{\epsilon_i(t)}$, then a more tractable problem results, with

$$\sigma^2_{\mathbf{n}_i(t+1)} \approx \left(\frac{b_i(t)}{k}\right)^2 \left(\frac{\sigma^2_{\mathbf{e}_i}}{N_i(t)} + \sigma^2_{\epsilon_i(t)}\right).$$

Writing b^+ for $b_i(t)$, b for $b_i(t-1)$, N^+ for $N_i(t)$, and N for $N_i(t-1)$, and using the fact that $\sigma^2_{\epsilon_i(t)} = b_i(t)\sigma^2_{\mathbf{e}_i(t)} = b_i(t)\sigma^2_{\mathbf{e}_i}/N_i(t)$ from (6)

$$\sigma^2_{\mathbf{n}_i(t)} = b^2(1+b)\sigma^2_{\mathbf{e}_i}/k^2N$$

so the optimal step length of (5) becomes

$$\begin{aligned}b^+ &= \frac{b/N + b^2(1+b)/k^2N}{b/N + b^2(1+b)/k^2N + 1/N^+} \\ &= \frac{AN^+}{AN^+ + k^2N}\end{aligned} \tag{8}$$

TABLE I
RATIO OF MEAN TO STANDARD DEVIATION, AND CROSS-CORRELATION FOR QUANTITIES $\mathbf{n}_i(t)$, $\mathbf{e}_i(t)$, AND $\epsilon_i(t-1)$

$R(\mathbf{n}_i(t))$	$R(\mathbf{e}_i(t))$	$R(\epsilon_i(t))$	$\rho(\mathbf{n}_i(t), \mathbf{e}_i(t))$	$\rho(\mathbf{n}_i(t), \epsilon_i(t-1))$	$\rho(\mathbf{e}_i(t), \epsilon_i(t-1))$
-0.00380	-0.00739	-0.00201	-0.0180	-0.0718	0.00437

where

$$A = b^3 + b^2 + k^2 b.$$

D. Some Properties of the Optimal Step-Length Sequence

For constant $N_i(t)$, the expression in (8) becomes

$$b^+ = \frac{(b^2 + k^2)b + b^2}{(b^2 + k^2)(b+1)}. \tag{9}$$

If $b_i(0) \in (0,1)$, then $b_i(t) \to 0$ as $t \to \infty$ since $b_i(t)$ is monotonic decreasing and bounded below by zero and hence, convergent, and zero is the only stationary point.

In the limiting case of small $b_i(t)$ it is easy to show that $b_i(t) \sim k^2/n(k^2-1)$ if $k \neq 1$ and $b_i(t) \sim (3n)^{-1/3}$ otherwise, where n denotes the number of times that the ith vector has been updated.

Note that for large k (slowly moving partitions) the result reduces to that of [3], $b_i(t) = 1/n$. Also, the step length itself converges to zero, while the sum over all time is unbounded, ensuring that anomalies early in training are completely swamped by later training [8], [9].

IV. EXPERIMENTAL RESULTS

Many approximations were made in the above calculations, and while they were heuristically justified, it is useful to confirm the results with numerical simulations. In this section, results for uncorrelated two-dimensional unit variance Gaussian sources at a rate of one bit per sample (b/sample) are presented, along with some results for image vector quantization. The initial codebooks used were tightly centered around zero for the Gaussian sources, and tightly centered around the half-scale value of 128 for the image data. For comparison, results will be presented both for the step length given by (5) and that proposed in [3], which was derived as an optimal step length under stronger implicit assumptions than those of this paper.

A. Verifying Assumptions

Before examining the performance of the system, it is important to verify that $\mathbf{n}_i(t)$, $\mathbf{e}_i(t)$, and $\epsilon_i(t-1)$ all have zero mean and are uncorrelated as assumed in deriving (5). A variable x can be considered zero mean if $|R(x)| \ll 1$, where $R(x) = E[x]/\sqrt{Var[x]}$, and x and y can be considered uncorrelated if their correlation satisfies $|\rho(x,y)| \ll 1$. Table I shows these quantities, which can be seen to be approximately zero as required. These results are the average over t and i for 1000 batches, each of 128 vectors from a uniform distribution presented to a 64-level one-dimensional VQ.

B. Example: Incremental Training of Gaussian VQ

Fully incremental training with the step length proposed here and that proposed in [3] yields similar results, as can be seen in Fig. 3. In the early stages, the results are quite unreliable due to the random processes involved, but the proposed step length gives slightly superior performance in the later stages [Fig 1(b)]. Although the traces are quite close, toward the end of training the actual time taken for the proposed scheme to reach any given level, say MSE = 0.365, is almost half of that required using the step length presented in [3].

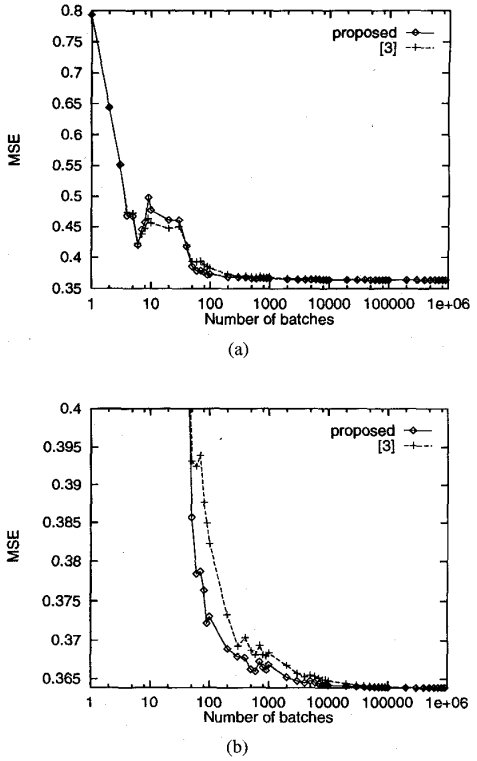


Fig. 3. Reconstruction error (outside training sequence) for a Gaussian codebook trained with fully incremental training; (a) entire sequence; (b) close-up of final stages.

C. Example: Semibatch Gaussian VQ

Greater improvements can be seen in the case of semibatch learning. When samples are taken 100 at a time, training proceeds as in Fig. 4. As well as results from the proposed optimal approach, this graph includes two sets of results from the technique presented in [3]. One is the simple minded use of the rule $b_i(t) = 1/n$, where n is the number of times that vector i has been updated. This is clearly not what was intended, but has been included for comparison. The other case is the degenerate case of (8) for large k. This corresponds to the case in which the partitions are assumed not to alter during training, and calculates $\hat{\mathbf{c}}_i(t)$ according to (7). These two rules give indistinguishable results.

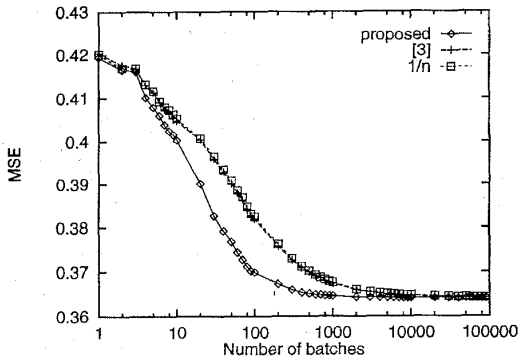


Fig. 4. Reconstruction error (outside training sequence) for a Gaussian codebook trained with batches of 100 random vectors.

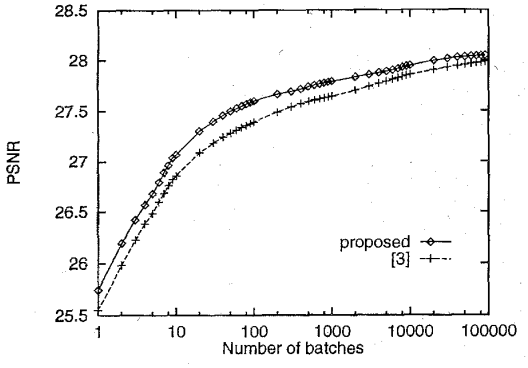


Fig. 5. Peak signal to noise ratio (PSNR) (within training set) for a 64-element codebook designed on 16 dimensional vectors from Lena.

All three schemes eventually produce approximately optimal quantizers, but training is faster when optimal updates are made.

The reason that the results in Fig. 4 show greater improvement than those in Fig. 3 may be understood by considering how the partitions move. In the fully incremental case, vectors fall into roughly the correct positions early in training, and so the effect of the vectors outside the final partitions is swamped by later training. When semibatch training is used, many training samples are mapped to the vectors before they have had a chance to find their correct positions. Thus, it is more important to take into account the effect of changing partitions.

E. Example: Image Data

In addition to the simple Gaussian example, the updates were tested on data from the popular Lena image. A 64-level vector quantizer using 4×4 pixel blocks [giving a bit rate 0.375 bits per pixel (bpp)] was trained on the 256×256 greyscale image, with initial weights clustered around the half-scale value of 128. Training vectors were presented in batches of one tenth of the entire training set (410 vectors per batch). As can be seen from the results in Fig. 5, the step length of the proposed scheme provides better performance in this case than those of (7), the direct generalization of [3], which ignores the change in partitions. In particular, the number of training steps required by the proposed scheme to reach a given performance level is noticeably less than that required using (7). This more than offsets the increased time per iteration to compute the step length, although it may not justify the added code complexity.

V. Conclusion

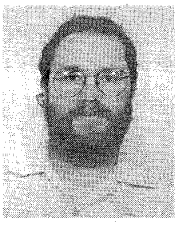
This paper has presented a new unified approach to selecting optimal step lengths for training vector quantizers. By treating updating as an estimation task, an expression, (5), has been derived for the optimal step length. Expression (5) is valid for the cases of batch, semibatch, and incremental training. As well as generalizing to the semibatch case, this expression has been seen to give results superior to those generated by a step-length sequence previously thought to be optimal [3]. An explicit expression, (8), has been found for an important special case—that of training from a random source.

Acknowledgment

The authors would like to thank Dr. Y. Hua for helpful comments on this paper. While performing this work, the first author was on a scholarship from the Australian Telecommunications and Electronics Research Board (ATERB).

References

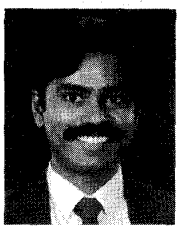
[1] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 129–137, Mar. 1982.
[2] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. 28, pp. 84–95, Jan. 1980.
[3] L. Wu and F. Fallside, "On the design of connectionist vector quantizers," *Computer, Speech, Language*, vol. 5, pp. 207–229, 1991.
[4] L. L. H. Andrew, "Neuron splitting for efficient feature map formation," in *Proc. Australia–New Zealand Conf. Intelligent Inform. Syst.'94*, Brisbane, Australia, 1994, pp. 10–13.
[5] C. N. Manikopoulos and G. E. Antoniou, "Adaptive encoding of a videoconference image sequence via neural networks," *J. Elec., Electron. Eng., Australia*, vol. 12, pp. 233–241, Sept. 1992.
[6] L. L. H. Andrew and M. Palaniswami, "A new adaptive image sequence coding scheme using Kohonen's SOFM," in *Proc. Int. Conf. Neural Networks*, Perth, Western Australia, 1995.
[7] H. Ritter and K. Schulten, "Kohonen's self-organizing maps: Exploring their computational capabilities," in *Proc. Int. Conf. Neural Networks*, San Diego, CA, 1988, pp. I-109–I-116.
[8] ——, "Convergence properties of Kohonen's topology conserving maps: Fluctuations, stability and dimension selection," *Biol. Cybern.*, vol. 60, pp. 59–71, 1988.
[9] M. T. Wasan, *Stochastic Approximation*. London: Cambridge University Press, 1969.



Lachlan L. H. Andrew was born in Melbourne, Australia, on September 21, 1970. He received the B.E. degree in electrical engineering and the B.Sc. degree in computer science from the University of Melbourne, Australia, in 1992 and 1993, respectively.

He is currently studying toward the Ph.D. at the University of Melbourne, Australia. His research interests include image coding, vector quantization, and neural networks.

Mr. Andrew was awarded an IEE undergraduate prize in 1992.



Marimuthu Palaniswami (S'84–M'85–SM'94) was born in Ayyampudur, India. He received the Master of Engineering Science degree from the University of Melbourne, Australia, in 1984, and the Ph.D. degree from the University of Newcastle, Australia, in 1987.

Since 1987, he has been with the University of Melbourne, Australia, where he is Associate Professor and a member of the Cooperative Research Centre for Sensor and Signal Processing. He has active collaborative links with a number of international Universities, including Florida International University, Indian Institute of Science, and Nanyang Technological University. He conducted research at the Communications Research Laboratory under the foreign specialist invitation program of the Ministry of Education, Japan. His research activities are supported by the Australian Research Council, Department of Technology and Commerce, National Australia Bank, Broken Hill Propriety Limited, Defence Science and Technology Organization, Integrated Control Systems Pty Ltd., and Signal Processing Associates Pty Ltd. His research interests are in adaptive control, neural networks, and signal processing. He has published over 100 conference and journal papers in these topics.

Dr. Palaniswami has served on the program committees for a number of conferences, including the IEEE Workshop on Emerging Technologies and Factory Automation, and the 3rd, 4th, and 5th Australian Conference(s) on Neural Networks. He was also the Technical Programme Chairman for the 1st IEEE Australia–New Zealand Conference on Intelligent Information Processing Systems, in 1993, the Technical Program Co-chairman for the IEEE International Conference on Neural Networks, in 1995, and is Chairman of the IEEE Special Interest Group on Neural Networks (Victoria).