UNLIMITED

313438

(3)

(2)

AD-A258 222

DEFENCE RESEARCH AGENCY
MALVERN

MEMORANDUM No. 4669

CODE VECTOR DENSITY IN TOPOGRAPHIC MAPPINGS

Author: S P Luttrell

92-30085

DEFENCE RESEARCH AGENCY,
MALVERN,
WORCS.

MEMORANDUM No. 4669

UNLIMITED

CONDITIONS OF RELEASE

0134072 313438

************************** DRIC U

************************** DRIC Y

Reports quoted are not necessarily available to members of the public or to commercial organisations.

Defence Research Agency

Malvern

Memorandum 4669

Code vector density in topographic mappings

Stephen P Luttrell

Image Processing Research Section
Defence Research Agency
St Andrews Rd, MALVERN, Worcestershire, WR14 3PS
Janet: luttrell@uk.mod.hermes
Internet:luttrell%hermes.mod.uk@relay.mod.uk

6 August 1992

Abstract

In this memorandum we present an informally argued derivation of the properties of topographic vector quantisers in the limit of a large codebook size. In particular, we prove that the code vector density does *not* depend on one's choice of neighbourhood function, provided that we use the minimum distortion (rather than the nearest neighbour) encoding prescription. This result suggests that widespread use of the nearest neighbour prescription in topographic mapping networks is fundamentally misguided. It would be advisable to remember that the nearest neighbour prescription is *assumed* not *derived*, so its adherents must accept defeat gracefully.

THIS PAGE IS INTENTIONALLY LEFT BLANK

Stephen P Luttrell, 6 August 1992

Table of Contents

THIS PAGE IS INTENTIONALLY LEFT BLANK

Stephen P Luttrell, 6 August 1992

1. Introduction

In this memorandum we concern ourselves with some interesting theoretical properties of unsupervised adaptive networks (specifically, topographic mappings [1]). This is part of the outcome of a programme of research whose purpose is to develop a more rigorous theoretical underpinning for topographic mappings than has hitherto appeared in the literature.

The asymptotic properties of topographic mappings have been the focus of some recent research [2,3,4]. The derivations in [4] concentrated on standard topographic mappings [1], whereas [3] concentrated on a slight variation of this approach which used a vector quantiser model [5,6] to develop topographic-like mappings. For convenience, we call such a model a topographic vector quantiser, to emphasise its relationship to both topographic mappings and to vector quantisers. Both of these studies were limited to the mapping of a one dimensional input space to a one dimensional output space.

In [4] an asymptotic density of weights (i.e. one dimensional weight vectors) $\rho \propto P^{\alpha}$ with $\alpha=((2w+1)^2/3)/((w+1)^2+w^2)$ emerged, where w described the half-width of a symmetric topographic neighbourhood function. In [3] the much simpler result $\alpha=1/3$ emerged, assuming a symmetric, monotonically decreasing topographic neighbourhood function. This $\alpha=1/3$ result is the same as is obtained in a standard scalar quantiser [7].

In this note we derive a vector generalisation of the scalar results that we presented in [3], which can be applied to the problem of mapping an N dimensional input space to an n dimensional output space ($n \leq N$). The result $\rho \propto P^{N/(N+2)}$ (which reduces to $P^{1/3}$ in the one dimensional $N=1$ case, as expected) is already known to hold for a standard vector quantiser [8], and the critical question is whether this also applies to our variant of topographic mappings, as it did in the one dimensional case. It turns out that it does, but the derivation is rather involved.

2. Outline

Our derivation rests on three critical steps.

1. We assume that we can *ignore second order effects* such as curvature of the surface in which the code vectors sit. This assumption may be satisfied by choosing a topographic neighbourhood function whose mass is concentrated in the neighbourhood of a single value, and then ensuring that the number of code vectors is sufficiently large that the surface formed by the code vectors is locally smooth. This allows us to approximate the expression for the average Euclidean distortion as the sum of a pair of covariance matrices (or inertia tensors): the covariance of each quantisation cell, plus the covariance of the cells in its topographic neighbourhood.
2. We assume that the topographic neighbourhood function is *translation invariant*, to show that the relationship between these two types of covariance matrix does not depend on one's location, apart from trivial rotation factors. Thus we express the Euclidean distortion solely in terms of the covariance of each quantisation cell.

3. We *define* a code vector density in terms of this covariance. This allows us to hold constant the total number of code vectors whilst we minimise the average Euclidean distortion with respect to the covariance of each quantisation cell.

Throughout our calculations we make extensive use of probabilities. We do this in order to simplify the notational problems that can arise when we quantities, such as "the set of all inputs that encode to give a particular code". Also, we make extensive use of Bayes' theorem to manipulate the probabilities into various forms, as required. We do not claim that the use of probabilities (plus Bayes' theorem) is a *necessary* part of our calculations, but it certainly makes them much easier to perform, because the notation does most of the work for us.

3. Covariance matrices

Firstly, define the average Euclidean distortion (or Lyapunov function) for encoding via $y(x)$, adding noise via $P(y'|y)$, and then decoding via $x'(y')$ as

$$D \equiv \int \mathrm{d}x\, \mathrm{P}(x) \int \mathrm{d}y'\, \mathrm{P}(y'|y(x)) \| x - x'(y') \|^2 \tag{1}$$

The integration over y' takes account of the various possible distortions that might be applied to the code y, and the integration over x averages over the various possible input vectors that might be presented. Although this definition of D uses an L_2 distortion metric, our results can easily be generalised to an L_r distortion metric, as we shall show later on.

Note that at this stage the *encoding and decoding functions* $y(x)$ and $x'(y')$ have not yet been specified; it is minimisation of D with respect to the choice of these functions that determines their actual form. We shall deal with $x'(y')$ immediately, and defer $y(x)$ until later on.

Define the average vectors

$$\begin{aligned} x(y) &\equiv \int \mathrm{d}x\, \mathrm{P}(x|y)\, x \\ x'(y') &\equiv \int \mathrm{d}x\, \mathrm{P}(x|y')\, x = \int \mathrm{d}y\, \mathrm{P}(y|y')\, x(y) \\ x_0(y) &\equiv \int dy'\, P(y'|y)\, x'(y') \end{aligned} \tag{2}$$

where we have implicitly used Bayes' theorem to construct the posterior probabilities $P(x|y)$, $P(x|y')$ and $P(y|y')$. Note that in Equation 2 we define $x'(y')$ so that it satisfies $\partial D/\partial x'(y')=0$, so henceforth we do not need to worry about minimisation of D with respect to $x'(y')$. We make the other two definitions for later use, and present them at this stage merely for convenience.

We wish to manipulate D into a form in which it is expressed as an integral over y-space, so that we can identify how much distortion is associated with each code vector. We therefore have to invert the order of the integrations in Equation 1. In order to do this we must express the encoding operation $y(x)$ using probability notation. Thus

$$D \equiv \int dx\, P(x) \int dy'\, P(y'|y) \int dy\, P(y|x) \| x - x'(y') \|^2 \tag{3}$$

where $P(y|x) \equiv \delta(y-y(x))$. Now we manipulate the probabilities using Bayes' theorem and the fact that $x \to y \to y'$ is a Markov chain, to obtain

$$P(x)P(y'|y)P(y|x) = P(x,y,y') = P(y)P(y'|y)P(x|y) \tag{4}$$

whence

$$\begin{aligned} D &= \int dy\, P(y) \int dy'\, P(y'|y) \int dx\, P(x|y) \left\| x - x'(y') \right\|^2 \\ &= \mathrm{trace}\left(\int dy\, P(y) \sigma(y) \right) \end{aligned} \tag{5}$$

where trace $zz^T = \|z\|^2$, and where we have defined the covariance matrix $\sigma(y)$ as

$$\sigma(y) \equiv \int dy'\, P(y'|y) \int dx\, P(x|y) (x - x'(y'))(x - x'(y'))^T \tag{6}$$

We have successfully arranged D as an integral over y-space, and we have "opened up" the norm to reveal the covariance matrix that is hidden within. Our use of a covariance matrix appears to be an unnecessary step at this point, but it turns out to be essential in order to define a code vector density later on. We therefore use (traces of) covariance matrices, rather than norms, throughout our calculations.

We may split $\sigma(y)$ into a sum of more primitive pieces as follows.

1. Use the identity $(x-x'(y')) \equiv (x-x(y))-(x'(y')-x(y))$ to obtain a pair of terms $(x-x(y))(x-x(y))^T$ and $(x'(y')-x(y))(x'(y')-x(y))^T$. Note that the cross term vanishes when x is integrated.
2. Use the identity $(x'(y')-x(y)) \equiv (x'(y')-x_0(y))-(x(y)-x_0(y))$ to replace the second of these terms by another pair of terms $(x'(y')-x_0(y))(x'(y')-x_0(y))^T$ and $(x(y)-x_0(y))(x(y)-x_0(y))^T$. Note that that the cross term vanishes when is y' integrated.
3. Define some covariance matrices.

$$\begin{aligned} \sigma_0(y) &\equiv \int dx\, P(x|y)(x - x(y))(x - x(y))^T \\ \sigma_1(y) &\equiv \int dy'\, P(y'|y)(x'(y') - x_0(y))(x'(y') - x_0(y))^T \\ \sigma_2(y) &\equiv (x(y) - x_0(y))(x(y) - x_0(y))^T \end{aligned} \tag{7}$$

to yield finally

$$\sigma(y) = \sigma_0(y) + \sigma_1(y) + \sigma_2(y) \tag{8}$$

We may interpret $\sigma_0(y)$, $\sigma_1(y)$ and $\sigma_2(y)$ as follows.

Consider $\sigma_0(y)$ first of all. $x(y)$ is the centroid of x (given y), which we obtain by integrating (using as a "measure" the posterior probability $P(x|y)$) over the continuum of inputs x. The vector $x-x(y)$ is the displacement of x relative to this centroid, so $\sigma_0(y)$ is the covariance of x (given y).

Now consider $\sigma_1(y)$. In this case we integrate over y', not over x as we did for $\sigma_0(y)$. Because y actually corresponds to a discrete index in the codebook, this integral is a sum (using as a "measure" the probability $P(y'|y)$) over code indices. In this case $x_0(y)$ is the centroid of $x'(y')$ (given y), and the vector $x'(y') - x(y)$ is the displacement of $x'(y')$ relative to this centroid, so $\sigma_1(y)$ is the covariance of $x'(y')$ (given y).

Finally consider $\sigma_2(y)$. $x(y)$ and $x_0(y)$ are the centroids used to define $\sigma_0(y)$ and $\sigma_1(y)$, respectively. $\sigma_2(y)$ is the dyadic constructed from the displacement of these centroids from each other.

We should point out at this stage that our calculations are *exact* thus far. In order to make any further progress we need to make approximations, which we shall discuss in detail.

4. Approximations

In this Section we shall introduce several approximations that are essential to the analysis of the leading order properties of topographic vector quantisers. The overall goal of this Section is to develop an approximate expression for the covariance matrix $\sigma(y)$ in the form $\sigma(y)=M(y)\sigma_0(y)M^T(y)$, where $M(y)$ is a transformation, in order to express the distortion D in terms of $\sigma_0(y)$ alone, rather than $\sigma_0(y)+\sigma_1(y)$.

In Section 4.1 we introduce a very useful approximation for the encoding function $y(x)$, which allows us to replace minimum distortion encoding prescription by an approximately equivalent nearest neighbour encoding prescription. Note that this is *not* simply a naive replacement of minimum distortion by nearest neighbour. In Section 4.2 we use this approximation for $y(x)$ to write an approximate expression for the covariance matrix $\sigma(y)$, which is amenable to further analysis, unlike the exact expression. In Section 4.3 we introduce a local lattice, and translation invariance $P(y'|y) = P(y'-y)$, *in order to approximate* the positions of the $x'(y')$ in each neighbourhood, which allows us to interrelate the two components of $\sigma(y)$.

4.1. Nearest neighbour encoding

The $y(x)$ that minimises the Euclidean distortion D is given by

$$y(x) = \underset{y}{\arg\min} \int dy'\, P(y'|y)\left\|x - x'(y')\right\|^2 \tag{9}$$

where "arg min y ..." means "the value of y that minimises ...". Because the expression that is minimised is $\int dy'\, P(y'|y)\left\|x - x'(y')\right\|^2$, rather than $\|x-x'(y)\|^2$, this type of $y(x)$ is called a "minimum distortion" encoding prescription, rather than a "nearest neighbour" encoding prescription. Note that these two prescriptions are the same when the neighbourhood function collapses to zero size, i.e. $P(y'|y)=\delta(y'-y)$.

By using the identity $(x-x'(y'))\equiv(x-x_0(y))-(x'(y')-x_0(y))$, and noting that the cross term vanishes when is y' integrated, we may rearrange $y(x)$ into the form

$$\begin{aligned} y(x) &= \underset{y}{\arg\min}\left(\left\|x - x_0(y)\right\|^2 + \int dy'\, P(y'|y)\left\|x'(y') - x_0(y)\right\|^2\right) \\ &= \underset{y}{\arg\min}\left(\left\|x - x_0(y)\right\|^2 + \operatorname{trace}\sigma_1(y)\right) \end{aligned} \tag{10}$$

Let us assume that the trace $\sigma_1(y)$ term is a slowly varying function of y compared to the $\|x-x_0(y)\|^2$ term, then we may approximate $y(x)$ as

$$y(x) \approx \underset{y}{\arg\min} \|x - x_0(y)\|^2 \tag{11}$$

which has the form of a "nearest neighbour" encoding prescription. Therefore the $x_0(y)$ act as the effective code vectors in a "nearest neighbour" encoding prescription that is approximately equivalent to the ideal "minimum distortion" encoding scheme.

We have deduced that the effect of the neighbourhood function $P(y'|y)$ can be accounted for by replacing "minimum distortion" encoding by an approximately equivalent "nearest neighbour" encoding prescription, which is *not* simply obtained by ignoring the effect of $P(y'|y)$ altogether, as one might have naively assumed.

We shall make extensive use of this equivalence, because "nearest neighbour" encoding is conceptually simpler than "minimum distortion" encoding.

4.2. Covariance matrix

The contribution of the $\sigma_2(y)$ term to the Euclidean distortion D is proportional to $\|x(y)-x_0(y)\|^2$. Therefore minimising D tends to make the vectors $x(y)$ and $x_0(y)$ become similar to each other, so we shall henceforth assume

$$x(y) \approx x_0(y) \tag{12}$$

This approximation says that the centroid $x(y)$ of the quantisation cell attached to y is approximately coincident with the effective code vector $x_0(y)$ that defines the quantisation cell boundaries via a "nearest neighbour" prescription.

We may use this relationship to approximate the contributions to $\sigma(y)$ as

$$\begin{aligned} \sigma_0(y) &\approx \int dx\, P(x|y)(x - x_0(y))(x - x_0(y))^T \\ \sigma_1(y) &= \int dy'\, P(y'|y)(x'(y') - x_0(y))(x'(y') - x_0(y))^T \\ \sigma_2(y) &\approx 0 \end{aligned} \tag{13}$$

where we repeat the exact equation for $\sigma_1(y)$, for convenience.

We use the $x_0(y)$ to implement a "nearest neighbour" encoding scheme, so $\sigma(y)$ reduces approximately to a sum $\sigma_0(y)+\sigma_1(y)$ of two contributions which we may readily interpret:

(a) $\sigma_0(y)$ is the covariance matrix of the "quantisation cell" attatched to y, and centred on $x_0(y)$.

(b) $\sigma_1(y)$ is the covariance matrix of the vectors $x'(y')$ having mass $P(y'|y)$, whose centroid is $x_0(y)$.

This simplification of the interpretation of $\sigma(y)$ is possible only because of the assumptions that we have made, which are essentially all to do with assuming that properties vary slowly on the scale of a quantisation cell.

4.3. Lattice of code vectors

There are two approximations that we make in order to simplify our analysis of the code vectors $x'(y')$.

1. We make our first approximation in order to relate $\sigma_1(y)$ to $\sigma_0(y)$ in a simple way. Thus we develop a first order Taylor expansion of $x'(y')$ about the point $y'=y'_0$

$$x'(y') = x'(y'_0) + (y' - y'_0)^T . \frac{\partial x'(y'_0)}{\partial y'_0} + \cdots \tag{14}$$

   The term proportional to $(y'-y'_0)$ generates a linear variation of $x'(y')$ with y', which corresponds to a uniform lattice of vectors $x'(y')$, with y' indexing the lattice points and $x'(y')$ locating the lattice points in x-space.

   The omitted higher order terms would introduce non-uniformities into this lattice. There are two basic types of contribution: non-uniform stretching of the lattice, and lattice curvature. These two contributions arise from components of the second derivative $\partial^2 x'(y'_0)/\partial y'_0 \partial y'_0$ which lie parallel or perpendicular to the surface in which $\partial x'(y'_0)/\partial y'_0$ lies, respectively. We shall assume that the properties of $x'(y')$ vary sufficiently slowly on the scale of a topographic neighbourhood that we may ignore these higher order corrections.

2. Now we make our second approximation in order to ensure that the relationship between $\sigma_1(y)$ to $\sigma_0(y)$ has a simple dependence on y. We assume that $P(y'|y)$ is translation invariant, so that

$$P(y'|y) = P(y' - y) \tag{15}$$

Equation 2 then becomes

$$x_0(y) \equiv \int dy' \, P(y' - y) x'(y') \tag{16}$$

which ensures that the lattice of $x_0(y)$ is locally uniform, because it is a *convolution* of a fixed kernel with a locally uniform lattice of $x'(y')$. This requires that the $x_0(y)$ and the $x'(y')$ lattices are locally identical, modulo a spatial translation. Furthermore, this uniformity implies that the quantisation cells (implied by the $x_0(y)$) are themselves locally identical in shape.

Under these conditions, the decomposition $\sigma(y)$ can be expressed in the form

$$\sigma(y) \approx M(y)\sigma_0(y)M(y)^T \tag{17}$$

where $M(y)$ is a matrix that transforms $\sigma_0(y)$ into $\sigma_0(y)+\sigma_1(y)$. Assuming $P(y'|y) = P(y'-y)$, $\sigma_1(y)$ is determined *uniquely* by $\sigma_0(y)$, so $M(y)$ must have the approximate form

$$\begin{aligned} M(y) &\approx R(y) S R^T(y) \\ M(y)^{-1} &\approx R(y) S^{-1} R^T(y) \end{aligned} \tag{18}$$

where $R(y)$ is an orthogonal matrix (i.e. a rotation) which satisfies $R(y)R^T(y)=1$, and where S is a diagonal matrix (i.e. a scaling). Note that $R(y)$ depends on y because it must rotate

$\sigma_0(y)$ into a standard orientation (i.e. remove the peculiar local orientation of the lattice of $x_0(y)$ vectors) before applying the invariant scaling S (i.e. transform $\sigma_0(y)$ into $\sigma_0(y)+\sigma_1(y)$).

This factorisation of $\sigma(y)$ is important, because it allows us to simplify $\det(\sigma(y))$ thus

$$\det(\sigma(y)) \approx (\det S)^2 \det\sigma_0(y) \tag{19}$$

This result simply states that the "volume" of the full covariance $\sigma(y)$ is a constant factor times the "volume" of the covariance $\sigma_0(y)$ of a single quantisation cell. This result emerges in leading order because we assumed $P(y'|y) = P(y'-y)$.

5. Minimise the distortion

We now minimise the Euclidean distortion D with respect to variations of the components of $\sigma(y)$, whilst holding constant the total number K of code vectors. In Section 5.1 we use the covariance matrix $\sigma_0(y)$ to define a code vector density, which is, after all, the quantity of interest. In Section 5.2 we minimise D with respect to the values of the components of $\sigma_0(y)$ whilst holding constant the total number of code vectors, and in Section 5.3 we present the result for the optimal code vector density. In Section 5.4 we demonstrate how our approach specialises to the case of minimising the Euclidean distortion in a standard vector quantiser.

5.1. Code vector density

The code vector density $\rho(x)$ is defined as the number of quantisation cells per unit volume of x-space, which is the reciprocal of the volume of the quantisation cells in the locality of x. This volume may easily be determined from the covariance matrix $\sigma_0(y)$, to yield the following definition of code vector density

$$\rho(x) \equiv \left(\det\sigma_0(y(x))\right)^{-1/2} \tag{20}$$

The total number of code vectors K is the integral over all x of the code vector density, so

$$\begin{aligned} K &= \int dx\, \rho(x) \\ &= \int dx \left(\det\sigma_0(y(x))\right)^{-1/2} \\ &\approx \det S \int dx\, \det\sigma(y(x)) \end{aligned} \tag{21}$$

where we used the factorisation of $\sigma(y)$ given in Equation 17. This expression for K is non-trivial, because it required a lot of work in Section 4 to derive Equation 17. Indeed, this was the main purpose of Section 4.

5.2. Stationary distortion

Finally, we have all of the basic results that we need in order to minimise the Euclidean distortion D with respect to the values of the components of $\sigma(y)$, whilst holding constant the total number K of code vectors.

We may impose the constraint by introducing a Lagrange multiplier λ and locating the stationary point of

$$\begin{aligned} D(\lambda) &\equiv D + \lambda K \\ &\approx \int dx \left(P(x)\, \mathrm{trace}\, \sigma(y(x)) + \lambda \det S \det \sigma(y(x)) \right) \end{aligned} \tag{22}$$

Functionally differentiate $D(\lambda)$ with respect to the components $\sigma_{ij}(y)$ of the covariance matrix $\sigma(y)$, using the results

$$\frac{\delta\, \mathrm{trace}\, \sigma(w)}{\delta \sigma_{ij}(y)} = \delta_{ij}\, \delta(w - y) \tag{23}$$

$$\frac{\delta \det \sigma(w)}{\delta \sigma_{ij}(y)} = \sigma(w)^{-1}_{ji} \det \sigma(w)\, \delta(w - y) \tag{24}$$

where $\sigma(w)^{-1}_{ji}$ is the (j,i)-th component of $\sigma(w)^{-1}$. This yields

$$\frac{\delta D(\lambda)}{\delta \sigma_{ij}(y)} = \int dx\, \delta(y - y(x)) \left(P(x) \delta_{ij} - \frac{\lambda \det S}{2} \sigma(y(x))^{-1}_{ji} \det \sigma(y(x)) \right) \tag{25}$$

If we assume that $P(x)$ varies slowly as x ranges over those values that satisfy $y=y(x)$, then we can satisfy the condition $\delta D(\lambda)/\delta\sigma_{ij}(y)=0$ by choosing

$$\sigma_{ij}(y(x)) \propto P(x)^{-2/(N+2)} \delta_{ij} \tag{26}$$

It is important to note that the x-dependence of this result does not depend on the details of the topographic neighbourhood function $P(y'|y)$, other than its assumed translation invariance $P(y'|y)=P(y'-y)$, and the slow variation assumptions that we made earlier.

Note that the result in Equation 26 can easily be generalised to an L_r distortion metric, by replacing the trace $\sigma(y)$ by $(\mathrm{trace}\ \sigma(y))^{r/2}$, to obtain eventually $\sigma_{ij}(y(x)) \propto P(x)^{-2/(N+r)} \delta_{ij}$.

5.3. Stationary code vector density

Finally, we may use the factorisation of $\sigma(y)$ in Equation 17 to write the covariance of each quantisation cell $\sigma_0(x)$ as

$$\sigma_0(x) \propto P(x)^{-2/(N+2)} R(y(x)) S^{-2} R(y(x))^T \tag{27}$$

and, using the definition of code vector density in Equation 20, $\rho(x)$ as

$$\rho(x) \propto P(x)^{N/(N+2)} \tag{28}$$

which easily generalises to $\rho(x) \propto P(x)^{N/(N+r)}$ for an L_r distortion metric.

The result in Equation 28 reveals that the code vector density does *not* depend on the form of the topographic neighbourhood function. However, note that we made the following assumptions in order to derive this result:

1. $P(y'|y)=P(y'-y)$, so that local translation invariant solutions can develop.

2. $P(y'-y)$ is "local", so that smooth solutions can develop.

3. K is "large", so that local variations of $P(x)$ and higher order (curvature) corrections to the lattice of code vectors can be ignored.

5.4. Special case: vector quantiser

In the case of a standard vector quantiser (with a zero width topographic neighbourhood function), a simplified version of our approach can be used. The distortion D is then given by

$$\begin{aligned} D &\equiv \int dx\, P(x) \|x - x(y)\|^2 \\ &= \int dx\, P(x)\, \mathrm{trace}\, \sigma(y(x)) \end{aligned} \tag{29}$$

where $P(y'|y)=\delta(y'-y)$ and $\sigma(y)=\sigma_0(y)$ (i.e. we do not need to consider extra terms arising from a non-zero topographic neighbourhood). The code vector density $\rho(x)$ is then given by

$$\rho(x) \equiv \left(\det \sigma(y(x))\right)^{-1/2} \tag{30}$$

We could minimise the distortion with respect to the components $\sigma_{ij}(y)$ of the covariance matrix $\sigma(y)$, as we have already done. However, it is simpler to argue straight away that $\sigma(y)$ must be isotropic (i.e. $\sigma_{ij}(y)=s(y)\delta_{ij}$), because locally there are no special directions in x-space. Thus we can write

$$\begin{aligned} \mathrm{trace}\, \sigma(y) &= N\, s(y) \\ \det \sigma(y) &= s(y)^N \end{aligned} \tag{31}$$

and then minimise the distortion with respect to the scalar $s(y)$, or, equivalently, with respect to the code vector density $\rho(x)$ $(=s(y(x))^{-N/2})$ itself.

If we express this minimisation problem in terms of $\rho(x)$, rather than $s(y)$, it becomes

$$\frac{\delta}{\delta \rho(x)} \left(\int dx\, P(x) \rho(x)^{-2/N} + \lambda \int dx\, \rho(x) \right) = 0 \tag{32}$$

whose solution is $\rho(x) \propto P(x)^{N/(N+2)}$, as expected.

This result can easily be generalised to an L_r distortion metric by replacing the $\rho(x)^{-2/N}$ by $\rho(x)^{-r/N}$, to obtain $\rho(x) \propto P(x)^{N/(N+r)}$. This derivation of the optimum code vector density is simpler, and more intuitive, than the one presented in [8].

6. Conclusions

In this memorandum we have examined the properties of a special type of vector quantiser, called a topographic vector quantiser because of its similarity to a topographic mappings. The only difference between the respective training algorithms is that topographic mappings use a "nearest neighbour" encoding prescription, whereas topographic vector quantisers use a "minimum distortion" encoding prescription.

Code Vector Density

We have derived the leading order properties of the code vectors of topographic vector quantisers, and we have shown that the code vector density is insensitive to one's choice of topographic neighbourhood function, provided that it is of convolution type (i.e. the same for all code vectors). This result stands in stark contrast to the strong dependence of the code vector density on topographic neighbourhood function in a standard topographic mapping.

This result strongly suggests that minimum distortion encoding is a more theoretically respectable encoding prescription than nearest neighbour encoding. Indeed, minimum distortion is a *derived* prescription, whereas nearest neighbour is an *assumed* prescription, so the nearest neighbour prescription should be used with suspicion.

7. Notation

Define the basic vectors and functions.

$\boldsymbol{x}$	input vector
$\boldsymbol{y}$	code
$\boldsymbol{y}'$	distorted code
$\boldsymbol{y}(\boldsymbol{x})$	encoding function - transform from $\boldsymbol{x}$-space to $\boldsymbol{y}$-space
$\boldsymbol{x}'(\boldsymbol{y}')$	decoding function - transform from $\boldsymbol{y}'$-space to $\boldsymbol{x}$-space
N	the dimensionality of the input vector
D	the average Euclidean distortion between input and reconstruction
$\rho(\boldsymbol{x})$	the code vector density
K	the total number of code vectors
λ	a Lagrange multiplier

Define the basic densities.

$P(\boldsymbol{x})$	density of inputs
$P(y\vert\boldsymbol{x})$	density of codes (given that the input is known) - assumed to be $\delta(y-\boldsymbol{y}(\boldsymbol{x}))$
$P(y'\vert y)$	density of distorted codes (given that the code is known)

Define the derived densities (obtained using Bayes' theorem).

$P(\boldsymbol{x}\vert y)$	posterior density of inputs (given that the code is known)
$P(\boldsymbol{x}\vert y')$	posterior density of inputs (given that the distorted code is known)
$P(y\vert y')$	posterior density of codes (given that the distorted code is known)

Define the derived functions (obtained using the posterior densities, etc)

$\boldsymbol{x}(\boldsymbol{y})$	the centroid of $\boldsymbol{x}$ (given that $\boldsymbol{y}$ is known)
$\boldsymbol{x}_0(\boldsymbol{y})$	the centroid of $\boldsymbol{x}'(\boldsymbol{y}')$ (given that $\boldsymbol{y}$ is known)
$\sigma(\boldsymbol{y})$	the matrix whose average trace is the Euclidean distortion
$\sigma_0(\boldsymbol{y})$	the contribution to $\sigma(\boldsymbol{y})$ from a quantisation cell

$\sigma_1(y)$ the contribution to $\sigma(y)$ from the topographic neighbourhood of a cell

$M(y)$ the transformation which converts $\sigma_0(y)$ into $\sigma(y)$

$R(y)$ the "rotate" part of $M(y)$

S the "scale" part of $M(y)$

8. References

[1] Kohonen T, Self organisation and associative memory, Springer-Verlag, 1984

[2] Luttrell S P, Asymptotic code vector density in topographic vector quantisers, RSRE, 1990, Technical Report 4392

[3] Luttrell S P, Code vector density in topographic mappings: scalar case, IEEE *Trans. NN, 1991*, **2**, 427-436

[4] Ritter H, Asymptotic level density for a class of vector quantisation processes, IEEE Trans. NN, 1991, **2**, 173-175

[5] Luttrell S P, Derivation of a class of training algorithms, IEEE Trans. NN, 1990, **1**, 229-232

[6] Luttrell S P, Hierarchical self-organising networks, Proc. 1st IEE Conf. on Artificial Neural Networks, London, 1989, 2-6

[7] Lloyd S P, Least squares quantisation in PCM, IEEE Trans. IT, 1982, **28**, 129-137

[8] Zador P L, Asymptotic quantisation error of continuous signals and the quantisation dimension, IEEE Trans. IT, 1982, **28**, 139-149

INTENTIONALLY BLANK

REPORT DOCUMENTATION PAGE

DRIC Reference Number (if known) ....

Overall security classification of sheet ..........UNCLASSIFIED..
(As far as possible this sheet should contain only unclassified information. If it is necessary to enter classified information, the field concerned must be marked to indicate the classification, eg (R), (C) or (S).

Originators Reference/Report No.	Month	Year
MEMO 4669	AUGUST	1992

Originators Name and Location
DRA, ST ANDREWS ROAD
MALVERN, WORCS WR14 3PS

Monitoring Agency Name and Location

Title
CODE VECTOR DENSITY IN TOPOGRAPHIC MAPPINGS

Report Security Classification	Title Classification (U, R, C or S)
UNCLASSIFIED	U

Foreign Language Title (in the case of translations)

Conference Details

Agency Reference	Contract Number and Period

Project Number	Other References

Authors	Pagination and Ref
LUTTRELL, S P	14

Abstract
In this memorandum we present an informally argued derivation of the properties of topographic vector quantisers in the limit of a large codebook size. In particular, we prove that the code vector density does *not* depend on one's choice of neighbourhood function, provided that we use the minimum distortion (rather than the nearest neighbour) encoding prescription. This result suggests that widespread use of the nearest neighbour prescription in topographic mapping networks is fundamentally misguided. It would be advisable to remember that the nearest neighbour prescription is *assumed* not *derived*, so its adherents must accept defeat gracefully.

Abstract Classification (U, R, C or S)
U

Descriptors

Distribution Statement (Enter any limitations on the distribution of the document)
UNLIMITED

580/48

INTENTIONALLY BLANK