CONTRIBUTIONS TO EFFICIENT VECTOR QUANTIZATION AND FREQUENCY ASSIGNMENT DESIGN AND IMPLEMENTATION

by

Kemal Demirciler

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

August 2003

Dedication

To my parents, Serpin and Onay,

for their unconditional love and support, and for selfless devotion to the welfare

of their children.

To my sister Naciye, for her sacrifice and love.

Acknowledgements

It is my pleasure to express my gratitude to my advisor Prof. Antonio Ortega for an excellent supervision with fruitful discussions and suggestions. His insights and insistence to look deeper into the problems not only increased the quality of my work, but also made it more interesting and fun to work on.

I thank Prof. Keith M. Chugg, Prof. Chrysostomos L. Nikias, Prof. C. –C. Jay Kuo and Prof. Ashish Goel for serving on my qualifying committee, and for their very useful feedback. Prof. Keith M. Chugg and Prof. Roger Zimmermann served on my dissertation committee for which I greatly thank them for their valuable time. I also thank Prof. Zhen Zhang and Prof. Keith M. Chugg for their feedback in the derivation of the performance bound in the Appendix, and I thank Prof. Ashish Goel for introducing me to the channel allocation problem.

I would also like to thank my research group members, especially, Krisda Lengwehasatit, Raghavendra Singh, Zhourong Miao, Sang-Yong Lee, Phoom Sagetong, Hua Xie and Naveen Srinivasamurthy for their friendship and for their occasional but essential assistance.

My deepest gratitude goes to my family for their support, encouragement and love for which I am indebt forever.

Obviously, without adequate financial support this work would not have been possible. I gratefully acknowledge the Powell Foundation and the Katherine Harumi Contant Foundation for the generous support throughout my graduate studies at USC. I thank Assistant Dean Margery Berti, Dean Leonard Silverman and Dean Chrysostomos L. Nikias for their belief in me and for recommending me for these prestigious fellowships. It is my sincere hope that I have been able to rise to their expectations. I also gratefully acknowledge the continuous support of the "Swim with Mike – Physically Challenged Athletes Foundation." It has been an honor to be a member of the "Swim with Mike" family, and to get to know Mr. Ron Orr and Mr. Mike Nyholt who are the driving force of the foundation.

Table of Contents

List of Tables

List of Figures

Abstract

Recently, Deterministic Annealing (DA) has become very popular for a wide variety of optimization problems. However, its computational complexity is very high. We investigated ways of reducing the complexity of DA by designing low complexity distributions to mimic the optimal Gibbs distribution. We also derived the theoretical performance loss for using the simple distributions instead of the optimal Gibbs distribution, and used the derived result to obtain optimal annealing schedules for the non-optimal distributions.

We applied the reduced complexity DA algorithms to the vector quantizer (VQ) design problem and to the channel/frequency allocation problem (FAP). In VQ design, compared to the generalized Lloyd algorithm (GLA) and a high performance stochastic relaxation algorithm (SR-D), the proposed algorithms significantly improved the quality of the final codebooks both with and without codebook initialization. Compared to the standard DA they reduced the computational complexity over a factor of 100 with negligible performance difference. In FAP the proposed algorithms are highly competitive with the presently available best assignment techniques on real-life GSM frequency planning scenarios.

In the last part of the dissertation we introduced a novel constrained vector quantizer (VQ), which we called Seg-VQ. As an extension of the transform coding

framework, in our approach the codevectors are constrained to be located on a series of line segments in the multidimensional space. The advantages of Seg-VQ are twofold: first, the encoding complexity is proportional to the number of segments rather than to the number of codevectors, and second, it can efficiently exploit the correlations in sources such as images. At high dimensions (8x8 blocks) we use multi-stage Seg-VQ where the input block is projected into a series of segments in order to be quantized. We proposed two different systems using multiple stages: in the first one we designed fixed codevectors constrained to be on the segments, and in the second one the segments are uniformly quantized depending on the required rate making it more robust for rate adaptation. The latter system is optimal for high rate quantization.

Chapter 1

Introduction

1.1 Optimization Problems

A wide variety of application problems in engineering, decision sciences and operations research can be posed as optimization problems. Such applications include digital signal processing, process control, database design, neural networks, resource allocation, VLSI design and strategic planning to name a few. Optimal solutions in these applications mean better implementation, faster execution, lower costs, and robust operation under changing conditions. Hence, there is a perpetual impetus for research for efficient optimization techniques [76, 55, 93, 107, 43, 9, 74, 82, 42, 88, 108, 61, 77].

In a general minimization variant of an optimization problem we are given a set S and a function $f : S \to R$, and asked to find an $s \in S$ for which $f(s) \leq f(s)$ for all $s \in S$. The set S is the domain of feasible solutions and f is the *objective* (or *cost*) function. Such an $s \in S$ is called the *global minimizer* of f over S, and the

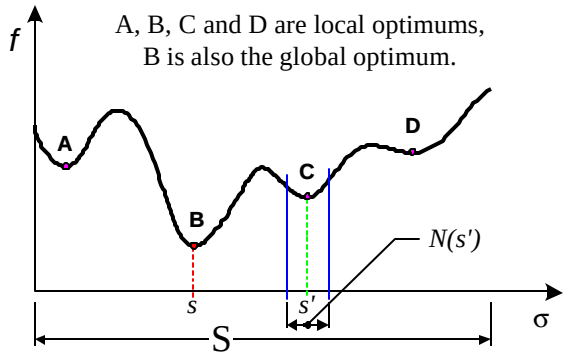


Figure 1.1: A 1-dimensional Euclidean optimization (minimization) problem.

corresponding function value, $f(s)$, is the *global minimum*. The mathematical formulation of this problem is as follows:

$$\min_{s} f(s) \quad \text{subject to } s \in S. \tag{1.1}$$

A solution $s' \in S$ is called a *local minimizer* if $f(s') \le f(s')$ for all s' in the neighborhood of s', $N(s')$, where the corresponding function value is a *local minimum*. In Figure 1.1 points *A*, *B*, *C* and *D* are all locally optimal, but only *B* is globally optimal. Note that a minimization problem can be transformed into a maximization problem since $\max f(s) = -\min(-f(s))$. In the following, unless otherwise stated, optimizing a problem will refer to minimizing its objective function.

Optimization problems are classified into two as *continuous* and *discrete* problems. A problem is continuous if the variables (unknowns) take on continuous real values, e.g., S in (1.1) would take on real values. Conversely, in a discrete optimization problem the variables take on discrete values, e.g., S would have

integer values. Both continuous and discrete problems are further classified into *constrained* optimization and *unconstrained* optimization problems based on the presence or absence of constraints. Depending on the form of the constraint functions used on S in (1.1), the constrained continuous problems are further classified into *linear* and *nonlinear*, where most real world continuous domain application problems fall into the class of nonlinear optimization problems [108, 85]. In this thesis we will be considering applications that employ discrete optimization methods.

1.1.1 Discrete Optimization Problems

When the feasible set S in (1.1) consists of discrete values the problem is called a *discrete optimization* problem. Discrete optimization is a field of study in combinatorics and discrete optimization is synonymously called *combinatorial optimization* [97]. Discrete optimization problems can be expressed in an *integer programming* (IP) formulation,

$$\begin{aligned} &\min_{\mathbf{s}} f(\mathbf{s}) \\ &\text{subject to} \quad h(\mathbf{s}) = 0 \\ &\qquad\qquad\ \ g(\mathbf{s}) \le 0 \\ &\qquad\qquad\ \ \mathbf{s} \in \mathbb{Z}^n \end{aligned} \tag{1.2}$$

where $\mathbb{Z}^n$ is the set of n-dimensional integer vectors. As in the continuous optimization problems, we classify the discrete problems according to the existence of constraints and their computational complexity. When there are equality and/or

inequality constraints, $h(\mathbf{s}) = 0$ and $g(\mathbf{s}) \leq 0$, the problem is called *constrained* optimization problem, and with those absent it is called an *unconstrained* optimization problem. Note again that problems with simple bounds on $\mathbf{s}$ are also classified as unconstrained.

Discrete constrained optimization problems are well studied in computer science and operations research areas. Based on their computational complexity, these problems are classified into class P or NP. The class P problems are those that can be solved by a polynomial-time algorithm [97, 27]. This means that the complexity of the problem grows as a polynomial in the number of variables of the problem. For the class NP (non-deterministic polynomial time) problems, it is not required that every instance of a problem can be solved in polynomial time, but simply that a given candidate solution (called a *certificate*) can be checked in polynomial time for its validity. Since every solvable problem is also certifiable, then P is a subset of NP, $P \subseteq NP$. To find out if this inclusion is proper or not is an important open problem in mathematics. A problem L to which all problems in NP polynomially reduce is called NP-hard, and if L itself is also in NP, then the problem L is said to be NP-complete [27, 45]. Such problems do not have polynomial time algorithms and their complexity grows exponentially in the size of the problem. In other words, the only way we know to solve these problems optimally is by making use of algorithms that run in exponential time. Many real-world application problems fall in the class of NP-hard problems: drilling of printed circuit boards (PCBs) [101], VLSI-chip fabrication [71], VLSI circuit design

and simulation [98], computer wiring and clustering of data arrays [79], X–ray crystallography [12], control of robots [101], genetic engineering [48], production planning, project resource management [28], CAD problems [73], machine scheduling [107], generalized Lloyd quantization [46], frequency allocation problem (FAP) [56], etc.

Indeed, application problems of practical interest fall in the category of hard optimization problems, both in the continuous domain and discrete domain alike.

1.2 Solving Discrete (Combinatorial) Optimization Problems

Given that many problems of practical relevance are computationally intractable (NP *-hard*), in general, it is infeasible to try to compute the optimum solution for these problems, simply because approaches for solving such problems exactly are all based on implicit enumeration of all feasible solutions and this takes an enormous amount of machine time even with very powerful computers. For example, consider the Traveling Salesman Problem (TSP) [88, 101, 79], a well known combinatorial optimization problem, which is NP -hard to solve but which can be easily stated: given a set of n cities and the geographical distances among them, the traveling salesman has to find the shortest tour in which he visits all the cities exactly once and returns to his starting city. In Figure 1.2, three tours are shown for a simple instance of a 9-city TSP. The tours *A* and *B* are feasible solutions whereas tour *C* is not a

feasible solution. Of the two feasible solutions tour *B* is shorter than tour *A*. For a 61 city – Traveling Salesman Problem (TSP) there are more possible solutions than the approximate number of particles in the universe, 10^{80} (there are $(n-1)!/2$ possible solutions for an n-city TSP; with $n=61$, $(60!/2) \cong 40\times10^{80}$).

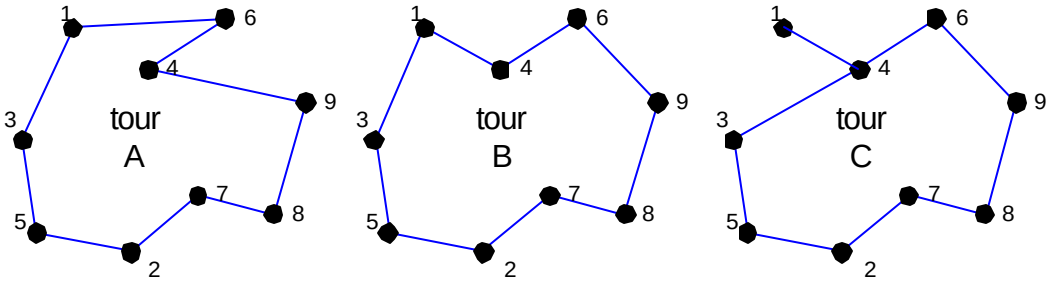


Figure 1.2: Three tours are shown for a TSP of 9 cities. Tours A and B are feasible, whereas, tour C is infeasible.

Even with a computer that can perform one billion additions per second, an exhaustive search would require about 7.7×10^{66} years to find the best tour (using 61 additions per tour). And since we do not have efficient algorithms to solve such computationally intractable problems, we have to look for sub-optimal solutions that can be obtained within realistic running times. The two main classes for treating these problems are *integer programming* (IP) methods and *(stochastic) local search* methods.

1.2.1 Integer Programming

An integer programming algorithm makes use of efficient upper and lower bounds on a problem's feasible search space (domain) to continually narrow down the possible solutions. The algorithm is stopped either when an incumbent solution is considered to be satisfactory, or when the running time has exceeded a predetermined limit. On large domains one cannot tell if an additional small amount of running time would improve on the incumbent solution significantly, or if an additional large amount of time would not improve the solution at all. The classic integer programming tool is the *branch-and-bound* (BB) method [97]. This is a divide-and-conquer method which tries to solve the problem by splitting it into smaller and smaller problems. The problem is split into subproblems such that the union of the feasible solutions of the subproblems give the solution of the actual problem. Subproblems are further divided into smaller subproblems until optimal solutions can be computed. Application of branch-and-bound algorithms is highly problem dependent [88], and they have been applied to various combinatorial optimization problems [40, 84, 5] with various success. The traveling salesman problem appears to be well suited for this approach where much progress has been made in solving large problems to optimality [5]. It is stated in [88] that still much work is needed to apply BB efficiently to other problems of practical interest.

1.2.2 Local Search

Local search methods find near-optimal solutions to hard optimization problems quickly and efficiently. They do not guarantee to find the optimum solutions nor do they guarantee solutions within a certain range of the optimum. Nevertheless, they are of great importance since they have become the standard way of obtaining high quality solutions for large combinatorial problems of practical interest. Hence, a vast amount of research is directed in this area to improve on existing search methods, to devise new ones, and to provide better understanding of them [82, 42, 88, 108, 61, 77]. In local search methods a move is made from an incumbent solution to another one within its neighborhood if the move results in a reduction of the cost. The neighborhood of a solution is generated by applying some suitably defined local change on the solution. The size of the neighborhood is a tradeoff between the aim of obtaining a good improvement each time a new solution is selected and the aim of limiting the computational cost. These methods start the algorithm with an initial solution that is selected either randomly or by a heuristic. It is apparent that a monotone improvement from one solution to the next results in a locally optimal solution dictated by the initial solution. To circumvent this drawback of getting stuck in a local minima, local search methods of interest allow random (stochastic) moves that may result in temporary decreases in performance, but eventually result in a better solution at the end of the entire search process. The most prominent stochastic search methods are Simulated Annealing [70, 16, 36, 62], Tabu Search [51, 52] and Genetic Algorithms [60, 53].

1.2.2.1 Simulated Annealing

The origin of simulated annealing (SA) [70, 16, 36, 62] rests in the physical process of annealing. Annealing is a thermal process for obtaining stable, low-energy state of a metal in a heat bath. First, the temperature of the heat bath is increased until the metal melts, where the particles of the metal arrange themselves randomly. And then, the temperature is carefully decreased allowing the molten metal to attain its lowest possible energy corresponding to the temperature. This process is continued until the metal completely cools, at which state the particles are arranged in a lattice with minimum energy.

By analogy to the physical annealing process, an optimization problem is "cooled in simulations" to find a low cost solution. The search is implemented by a Markov process which stochastically samples the solution space S of the problem. The optimization problem is characterized by an objective (cost or energy) function $E : S \to \mathbb{R}$, where $s \in S$ denotes the admissible solutions (or states) to the problem. The acceptance of a new state is determined by the Metropolis [89] algorithm, in which new states of decreased costs (increasing performance) are always accepted, and states with increased costs are accepted with a probability depending on the cost difference and the annealing temperature, T,

$$p\left(s^{\,current} \to s^{\,new}\right) = \begin{cases} 1 & \text{if } \Delta E = E\left(s^{\,new}\right) - E\left(s^{\,current}\right) \le 0, \\ e^{-\Delta E / T} & \text{otherwise.} \end{cases} \tag{1.3}$$

A Markov process with a transition probability (1.3) converges to an equilibrium probability distribution [44],

$$p(s) = \frac{e^{-\frac{1}{T}E(s)}}{\sum_{s' \in S} e^{-\frac{1}{T}E(s')}} \tag{1.4}$$

which is known as the Gibbs distribution.

The simulated annealing algorithm starts with an initial random state, $s^{initial} = s^{(0)}$ and an initial high temperature $T^{initial} = T^{(0)} \gg E\left(s^{(0)}\right)$ at iteration zero. At an iteration t, acceptance of a new state in the neighborhood of the current state, $s^{(t)}$ is determined by (1.3). Note that at the beginning of the algorithm when the temperature is high, $T \gg \Delta E$, the probability of accepting a cost-increasing state is close to one. The temperature schedule is a monotonically decreasing sequence where $\lim_{t \to \infty} T^{(t)} = 0$, and in the limiting case of zero temperature the algorithm only accepts cost decreasing state moves. The choice of the temperature schedule is an important factor affecting the SA performance. Geman and Geman [47] have shown that the global minimum can be achieved if the schedule obeys

$$T \propto \frac{1}{\log t}. \tag{1.5}$$

In practice, this schedule is very slow and Kirkpatrick [70] suggests using $T^{(t)} = r \cdot T^{(t-1)}$, where r is a positive constant less than one. The initial temperature, $T^{(0)}$ and the constant temperature reduction factor, r are application dependent and are determined experimentally [117].

1.2.2.2 Tabu Search

The tabu search (TS) [51, 52] method aims at avoiding suboptimal local minima as did the simulated annealing method. Hence, TS allows moves from a state s to neighbor state s' of higher cost (lower performance) with the hope that this will eventually lead to a better solution at the end of the search process. Unlike SA, the acceptance criterion in TS is to always choose the state in the neighborhood of the incumbent state, $N\left(s^{current}\right)$ with the lowest cost:

$$s^{new} = \underset{s'}{\operatorname{argmin}}\left\{E(s'): s' \in N\left(s^{current}\right)\right\} \tag{1.6}$$

where $E(s)$ is the cost of state s. The essential feature of the tabu search is the use of memory. To prevent the search from getting into infinite loops, a state that has recently been visited is included in a *tabu list*, Υ. The states in the tabu list are not allowed to be considered as candidate states in the neighborhood of the future states. The size of the tabu list, which is essentially the memory of the system, is an important parameter since a large memory increases the performance of the algorithm at the expense of higher complexity. Note that an infinite memory tabu search, with the whole solution domain as the neighborhood for any s, performs an exhaustive search of the solution space and returns the optimal solution. When the tabu list is full, to open up space for new tabu states the oldest members of the list are released. A released state becomes available as a candidate state. There are two stopping criterions for the tabu search algorithm; the algorithm is stopped either

when all neighborhood states of an incumbent state (current state) are in the tabu list, $N\left(s^{current}\right) \subseteq \Upsilon$, or if a predetermined number of moves are made without improvement. There are various extensions to the basic TS algorithm: for example, using multiple tabu lists of variable sizes [110], or concentrating the search on another location of the search space if the solution cannot be improved for a predefined number of iterations [7].

1.2.2.3 Genetic Algorithms

Genetic algorithms (GA) [60, 53] are randomized local search methods inspired by natural selection in biology [88, 41]. A genetic algorithm is characterized by maintaining a population of states (solutions) that evolves through a series of generations. The initial population, $\Pi^{(0)}$ is randomly selected from the domain of possible states, $\Pi^{(0)} \subset S$, where the size of the population from one generation to the next is kept constant, $|\Pi| = K$. In each iteration, t a new generation is derived by a three phase process: *evaluation of fitness*, *selection* and *generation of new individuals*.

In the first phase, *evaluation of fitness*, the quality (or fitness) of each state in the current population, $s \in \Pi^{(t)}$ is evaluated by a suitably defined function. This function, $f(s)$ called the fitness function of state s, is usually functionally related to the cost function of the optimization problem considered, $f(s) = g\left(E(s)\right)$. For

example, it can be the inverse of the cost function, $f(s) = \left[E(s)\right]^{-1}$. The *selection* phase corresponds to probabilistically selecting the states in the population, $\Pi^{(t)}$ according to their *relative* fitness. The selected states survive and pass into the new population while the unselected ones are discarded. The probability of selecting a state s is obtained by,

$$p_{\Pi}(s) = \frac{f(s)}{\sum_{s' \in \Pi} f(s')} \tag{1.7}$$

The probabilities provide the relative fitness for each state compared to the other states in the population; the better (low cost) states have a higher probability of surviving. Finally, in the third phase, *new individuals are generated* to replace the discarded ones during the selection phase. These are created either by recombining a pair of states (called *crossover*) or by modifying a single state (called *mutation*). To enable crossover and mutation, the states are encoded by fixed length binary strings. Hence, the crossover operation takes two states, s_1 and s_2, selects a random integer i uniformly from the set $\{1,2,\cdots,|s_1|\}$ (where $|s_1| = |s_2|$ because of fixed length encoding), and concatenates the first i bits of s_1 and last $|s_2|-1$ bits of s_2 to obtain a new state, s_3 for the new population (obviously, $s_3 \in S$). Symmetrically, the first i bits of s_2 and last $|s_1|-1$ bits of s_1 can be concatenated to obtain another new state, s_4. Mutation is usually performed by randomly flipping a bit of a state s to obtain the new state s'. At the end of this phase, the generation is derived and a new population is ready. The new generations are derived from the old ones in each

iteration until there is no improvement of the best solution (state) for a predetermined number of consecutive generations.

We see that in the development of the above algorithms natural phenomena (from physics and biology) have been an inspiration, and analogies drawn from these natural events are used to derive search methods that have become standard methods in solving computationally hard optimization problems. Out of these three methods, simulated annealing is the one with a relatively better defined theoretical framework and practically the most applied one [50, 69, 1, 2, 33, 64, 36, 57, 111, 38]. Tabu search and genetic algorithms have possibilities of further development [25].

1.3 Deterministic Annealing

Although simulated annealing is a general optimization framework applicable to various problems, its major drawback is its slowness (1.5). A deterministic variant of the simulated annealing, *deterministic annealing* (DA) was first suggested by Rose et al. [104]. Unlike stochastic moves made on the given energy surface (function), the deterministic annealing method can be viewed as incorporating the randomness into the energy (objective) function of the problem [104, 106, 103].

The main idea behind deterministic annealing (DA) is to use an effective energy function parameterized by a temperature term T, where at high temperatures the effective energy function is a smoothed (convexed) approximation of the actual energy function (i.e., the original energy function of the optimization problem), and

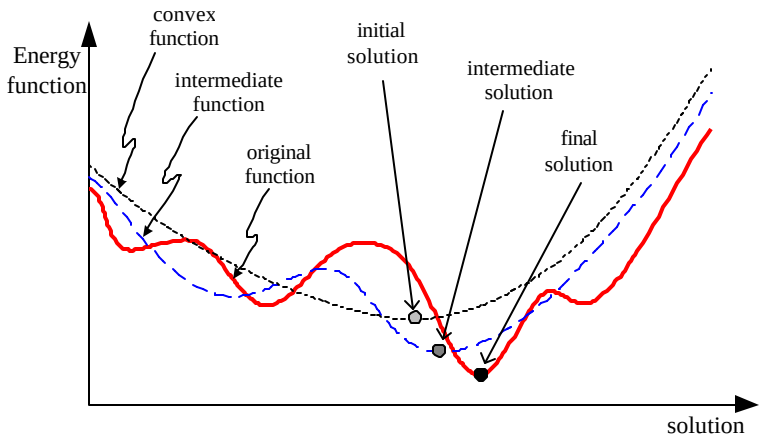


Figure 1.3: Three snap-shots showing the convergence of the DA. The initial convex function is obtained at the initial infinite temperature. This function has one minimum which is the global minimum. By gradually lowering the temperature the global minimum at the corresponding temperature is traced. When temperature reaches zero the original function is recovered and the final global solution is obtained.

as $T \to 0$ the original optimization problem is recovered. Hence, starting at the global optimum of the effective function, which is convex at the initial high temperature (theoretically $T \to \infty$), the DA method minimizes the effective energy function at each temperature to find the global minimum, and tracks the global optimum as the temperature is reduced gradually. Three snap-shots illustrating the convergence of the DA are shown in Figure 1.3. In the following subsections we will explain the main aspects of the deterministic annealing framework for solving combinatorial optimization problems.

1.3.1 Problem Setup

Assume that we have n entities (e.g., cities in TSP), $A=\{a_1,a_2,\cdots,a_n\}$ each of which can have K different assignments (labels), $B=\{b_1,b_2,\cdots,b_K\}$. The problem is to find the minimum cost assignment set (or state). Note that for this combinatorial problem there are K^n possible states. We define a state of the system by a vector, $\mathbf{a}=(a_1,a_2,\cdots,a_n)$, where each a_i corresponds to the assignment of the i^{th} entity a_i: $a_i=b_j \Rightarrow b_j$ is assigned to a_i, $a_i \leftarrow b_j$. Let $E(\mathbf{a})=E(a_1,a_2,\cdots,a_n)$ be the energy (cost) of the system when it is in state $\mathbf{a}=(a_1,a_2,\cdots,a_n)$. If we define $g(a_i,b_j)$ to be the cost of assigning b_j to a_i, where for a given state $\mathbf{a}$, $g(a_i,b_j)\equiv g(a_i,a(a_i))$, then the energy function is,

$$E(\mathbf{a})=\sum_{i=1}^{n} g(a_i,a(a_i)) \tag{1.8}$$

and the goal is to find,

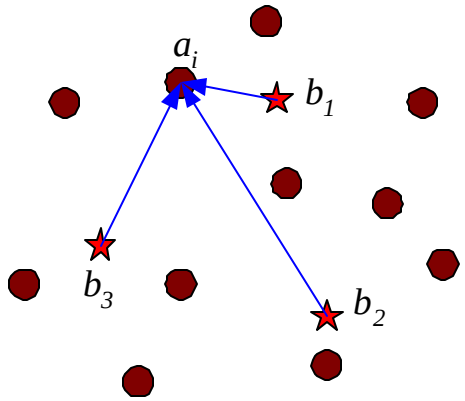
$$\begin{aligned}\mathbf{a}^* &= \underset{\mathbf{a}}{\operatorname{argmin}}\, E(\mathbf{a}) \\ &= \underset{\mathbf{a}}{\operatorname{argmin}} \sum_{i=1}^{n} g(a_i,a(a_i))\end{aligned} \tag{1.9}$$

Instead of tackling the problem as hard assignments in terms of states, in DA to each a_i, *all* possible labels, $\{b_1,b_2,\cdots,b_K\}$ get assigned with a certain probability,

$$\{p(b_1|a_i),p(b_2|a_i),\cdots,p(b_K|a_i)\}, \tag{1.10}$$

where

$$\sum_{j=1}^{K} p(b_j|a_i)=1 \quad \forall i. \tag{1.11}$$



Assuming, the closer b_j to a_i the more reliable it is:

$$p(b_1|a_i) > p(b_3|a_i) > p(b_2|a_i)$$

such that:

$$p(b_1|a_i) + p(b_2|a_i) + p(b_3|a_i) = 1$$

Figure 1.4: An example of soft assignment where each b_j gets assigned to each a_i in probability. In this example, the closer b_j to a_i the more reliable it is, hence the higher its association probability (soft assignment value).

We call these probabilities *soft assignments* or *soft information*. In communications engineering the definition of soft information is "a reliability measure over the sample space of the investigated random variable" (see section 1.3.2). In the above case the investigated random variable is $b \in B = \{b_1, b_2, \cdots, b_K\}$. A simple example of soft assignment is depicted in Figure 1.4, where the soft assignment value is inversely proportional to the distance between a_i and each b_j.

1.3.2 Soft Information Communications Example

Soft information is used in various communications engineering applications to provide a reliability measure on the possible signal choices [20, 22, 10]. For example, consider a communication system using binary signaling (e.g., two signal classes: $s_0 = -1$ and $s_1 = 1$) and transmission over an additive white Gaussian noise

(AWGN) channel. Let the observed channel output be y. Then, using Bayes' theorem, we can express the *a posteriori probability* (APP) of an event s_m conditioned on the observation y as follows,

$$p(s_m|y) = \frac{p(y|s_m)\,p(s_m)}{p(y)} = \frac{p(y|s_m)\,p(s_m)}{\sum_n p(y|s_n)\,p(s_n)}$$

where, $p(s_m)$ is the a priori probability of the m^{th} signal class, and $p(y|s_m)$ is the probability density function (pdf) of the received signal y conditioned on s_m. Assuming equal a priori probabilities the APP can be expressed as,

$$p(s_m|y) = \frac{p(y|s_m)}{\sum_n p(y|s_n)}.$$

Considering the additive white Gaussian noise with variance s^2, the pdf of y conditioned on s_m is,

$$p(y|s_m) = \frac{1}{s\sqrt{2p}} e^{\frac{1}{2s^2}\|y-s_m\|^2}$$

The channel decoder computes the APPs as follows,

$$p(s_m|y) = \frac{e^{-\frac{\|y-s_m\|^2}{2s^2}}}{\sum_{n=0}^{1} e^{-\frac{\|y-s_n\|^2}{2s^2}}} \qquad m = 0,1 \tag{1.12}$$

where $\sum_m p(s_m|y) = 1$. The APPs provided by the channel decoder are called the *soft information*, and such channel decoder is referred to as the *soft channel decoder*. The reliability of a decision depends on the relative magnitudes of the APPs. A hard

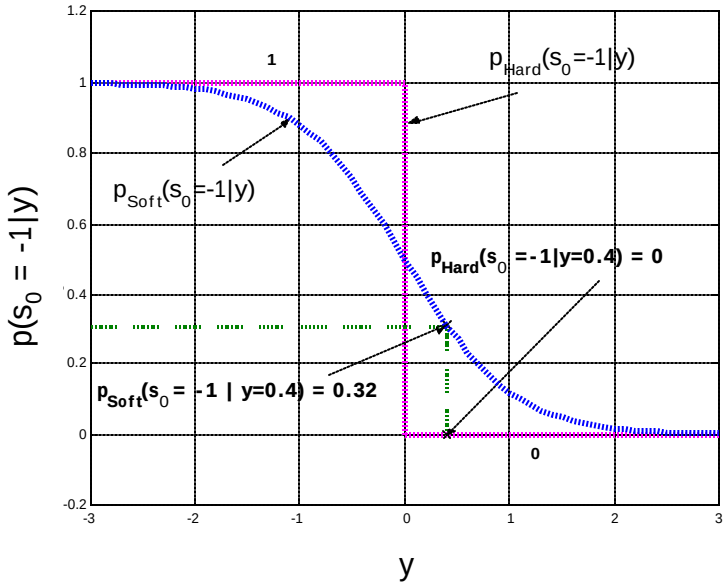


Figure 1.5: Comparing hard channel and soft channel decoding.

channel decoder would threshold the channel output, declaring $s_0 = -1$ was sent if $y < 0$, otherwise declaring $s_1 = 1$ was sent. Therefore, for a hard channel decoder,

$$p_{Hard}\left(s_0 = -1 \middle| y\right) = \begin{cases} 1, & y < 0 \\ 0, & y \geq 0 \end{cases} \tag{1.13}$$

Figure 1.5 illustrates the distinction between the outputs of a soft channel decoder and a hard channel decoder [22]. The soft output probability plot is obtained with (1.12) for $m = 0$, and the other with (1.13). In both soft and hard decoding cases, $p\left(s_1 = 1 \middle| y\right) = 1 - p\left(s_0 = -1 \middle| y\right)$. Note that hard channel decoding provides a hard decision based on which side of the threshold ($y_{threshold} = 0$) the channel output y is, and it does not give any further information. For example, for $y = 0.4$, a hard decoder would declare $s_1 = 1$ was sent. However, soft channel decoding provides

each hard decision, as well as the reliability of each decision. The highest soft value (APP) denotes the hard decision; that is, s_i is chosen such that, $i = \arg\max_i p(s_i | y)$. And the relative magnitude of $p(s_i | y)$ with other soft values (for the given channel output y) denotes the reliability of that decision. Thus, for $y = 0.4$, a soft decoder would also declare $s_1 = 1$ was sent, but it would also state that the reliability of declaring "$s_1 = 1$ was sent" is 68%, since $\max\left(p(s_0 | y), p(s_1 | y)\right)$

$= \max(0.32, 0.68) = 0.68$.

Similar to the definition of the soft channel decoder's soft output, the soft information in (1.10) gives the degree of reliability of *all* possible assignments to each entity, a_i. Note that an iterative algorithm utilizing soft information can be stopped at any time and the best computed hard assignments so far for each a_i can be obtained by thresholding the soft assignments, $a_i \leftarrow b^* = \arg\max_{b_j} p(b_j | a_i)$.

1.3.3 The Effective Energy Function

Recall that the main principle behind the DA method is to start at the global minimum of an effective energy function parameterized by a temperature term (this function is convex at the initial high temperature), minimize the effective energy function at each temperature to find the global minimum, and track the global optimum as the temperature is gradually reduced. In section 1.3.1 we have mentioned that in DA each entity a_i is associated in probability with all the labels,

$\{b_1, b_2, \cdots, b_K\}$. Hence, the state of the system is given by the probability distribution for associating entities with labels. In this section we will define what the effective energy function is and derive the optimal probability distribution for associating the entities with labels that minimizes the effective energy function. The derivation follows a similar approach to that of [103].

The expected cost (energy) function of the system defined over the probability assignments $\left\{ p\left(b_j \middle| a_i \right) \right\}$ is,

$$\mathbb{E}\{E\} = \sum_i \sum_j g\left(a_i, b_j \right) p\left(b_j \middle| a_i \right), \tag{1.14}$$

where the summation is over all entities and all labels. Note that at the limit, when the soft assignments are hard and each a_i gets assigned a unique $b \in B$ with probability one,

$$p\left(b_j \middle| a_i \right) = \begin{cases} 1, & \text{if } a_i \leftarrow b_j \\ 0, & \text{otherwise,} \end{cases} \tag{1.15}$$

the equation (1.14) becomes identical to (1.8) for the given hard assignments which form the state **a**. The assignment probabilities incorporate a level of randomness into the system, and the level of this randomness can be measured by the Shannon entropy,

$$\begin{aligned} H\left(a, b\right) &= -\sum_i \sum_j p\left(a_i, b_j \right) \log p\left(a_i, b_j \right) \\ &= H\left(a\right) + H\left(b \middle| a\right) \end{aligned} \tag{1.16}$$

where $H(a) = -\sum_i p(a_i) \log p(a_i)$ is independent of the assignments; it can be thought as the problem input entropy. Hence, the system randomness is expressed by the conditional entropy,

$$H(b|a) = -\frac{1}{n} \sum_i \sum_j p(b_j|a_i) \log p(b_j|a_i)$$

where $p(a_i) = 1/n$. Since $p(a)$ is a constant term common to all, it is irrelevant to the optimization, and we will use the following as the expression for the randomness of the system,

$$H'(b|a) = -\sum_i \sum_j p(b_j|a_i) \log p(b_j|a_i) \tag{1.17}$$

Therefore, the optimization problem is recast as *the minimization of the expected cost subject to a given level of randomness measured by the Shannon entropy:* $\text{minimize } \mathbb{E}\{E\} \text{ subject to } H = H_0$. This can be conveniently formulated as the minimization of the Lagrangian [106, 103],

$$F = \mathbb{E}\{E\} - T \cdot H \tag{1.18}$$

where T is the Lagrange multiplier, H is the conditional entropy given by (1.17) and $\mathbb{E}\{E\}$ is given by (1.14). The Lagrangian F in (1.18) is the principal component in the DA methodology and it is the *effective energy function* that is optimized (minimized) through annealing. It is called the *free energy* of the system. The name comes from statistical mechanics where a similar term that characterizes the thermodynamic potential of a system is called the Helmholtz free energy [103]. The Lagrange multiplier T is in analogy the temperature of the system, which in

(1.18) governs the level of randomness. The DA method seeks the minimum of the free energy at each temperature, and tracks the minimum while gradually lowering the temperature. To obtain the optimal distribution that minimizes the free energy F at a given temperature T, we differentiate (1.18) w.r.t. $p(b_j|a_i)$, equate to zero and solve for $p(b_j|a_i)$. Writing (1.18) explicitly, we have,

$$F = \sum_i \sum_j g(a_i, b_j)\, p(b_j|a_i) + T \sum_i \sum_j p(b_j|a_i) \log p(b_j|a_i). \tag{1.19}$$

Then,

$$\frac{\partial F(p)}{\partial p(b_j|a_i)} = g(a_i, b_j) + T\left[\log p(b_j|a_i) + 1\right] = 0$$

$$\Rightarrow \quad g(a_i, b_j) + T\left(1 + \log p(b_j|a_i)\right) = 0$$

$$\Rightarrow \quad \log p(b_j|a_i) = -\frac{g(a_i, b_j)}{T} - 1$$

$$\Rightarrow \quad p(b_j|a_i) = \mathrm{e}^{-1-\frac{1}{T} g(a_i, b_j)}$$

Since $p(b_j|a_i)$ is a probability mass function, $\sum_k p(b_k|a_i) = 1$ then,

$$p(b_j|a_i) = \frac{\mathrm{e}^{-\frac{1}{T} g(a_i, b_j)}}{\sum_k \mathrm{e}^{-\frac{1}{T} g(a_i, b_k)}}. \tag{1.20}$$

Hence, the optimal distribution that minimizes F at a given T is the Gibbs distribution (1.20).

Note that at infinite temperature the association probabilities (1.20) are uniform distributions, $p(b_j|a_i) = 1/K \ \forall j$ where $K = |B|$, which means that each assignment choice is equally associated with an entity a_i. This corresponds to maximum

softness and to maximum system entropy, $p(b_j|a_i) = 1/K \Leftrightarrow H(b|a) = \log K$. As the temperature is lowered the associations become more discriminating, and at $T = 0$ they become hard at which point the system entropy is zero, $p(b_j|a_i) = \{0, 1\}$ $\Leftrightarrow H(b|a) = 0$.

The cost function $g(a_i, b_j)$ is application dependent and its form is defined explicitly for each application problem. Its value in each iteration depends on the value assumed by the label b_j and the assignment probabilities.

Therefore, the practical DA algorithm is as follows:
starting at a high temperature, minimize F at each temperature iteratively by,

1) fixing costs, and updating the association probabilities $\{p(b_j|a_i)\}$;

2) fixing association probabilities, and updating costs, $\{g(a_i, b_j)\}$,

and lowering the temperature gradually.

The deterministic annealing method has been applied to clustering, classification and vector quantization problems [104, 106, 102, 105, 102, 92, 91, 59] where it outperformed the standard methods by a significant margin.

1.4 Contributions of the Research

The main principle of the DA method is that each entity is associated in probability with each assignment choice and these assignments are updated in each iteration. Hence, the computational complexity of the DA algorithm grows with the

size of the possible choices. On the other hand, in some applications the structure of the problem is such that a subset of the choices are less likely to be chosen compared to the others. This gets reflected in the soft information values; less likely choices have low reliability values compared to the high likely choices. And the smaller the soft value of a choice, the lesser its contribution towards an optimal final solution will be. However, the computational cost for all soft associations, having a high soft value or a low soft value, is the same. In this thesis, we study reducing the number of choices for each entity by identifying the subset of choices that are less likely for each entity, and show that setting their soft assignments to zero without computing results in large savings in the overall computation with negligible performance difference. Depending on the application, entities for which a subset of the choices can be suppressed can be identified either before the optimization starts, or as the iterations progress, and in some cases both before the optimization starts and as the optimization progresses.

In the standard deterministic annealing (DA) algorithm, the soft associations are computed using the Gibbs distribution which is the optimal distribution. However, the Gibbs distribution is a function of exponentials and therefore its computational complexity is high. If we recall that these are computed for each soft association update, for each entity, in each iteration, then it becomes clear that usage of a simpler distribution function can result in large computational savings. We have designed simplified soft information measures such that, while they are simple enough to facilitate fast computation of the soft associations, they can also closely mimic the

Gibbs distribution's functionality to keep the sacrifice in performance to a minimum. Nevertheless, using simplified soft measures instead of the optimal one is expected to result in some loss of performance. Hence, we have also derived the theoretical performance loss at a given system entropy due to using the simplified soft measures instead of the optimal Gibbs measure. Further, we have used this result to derive the optimal temperature reduction schedules for the simple soft measures, given the temperature schedule for the optimal Gibbs measure.

In Chapter 2, we apply our reduced complexity DA algorithms to the well studied vector quantizer (VQ) design. Compared to standard DA under the same temperature reduction schedule, we show that in exchange of a negligible performance difference our algorithms reduce the computational complexity by over a factor of 100. We also compare our algorithms to the traditionally used GLA and a high performing stochastic relaxation (SR) algorithm called SR-D, which is regarded by some researchers as a benchmark for near optimal performing quantizer design. We provide experimental evidence showing the superiority of our algorithms over these two algorithms. In this chapter, we also propose a stochastic extension to the reduced complexity DA algorithm. This extended algorithm contains a stochastic step similar to the one in SR-D. We show that the stochastic step improves the performance of the reduced complexity DA algorithms with diminishing benefits as the performance approaches the optimal. We also investigate the effect of codebook initialization on GLA, SR-D and the reduced complexity DA algorithms and show that, while GLA and SR-D receive major benefit from this initialization (PNN) at the

expense of increased computational complexity, the reduced complexity DA algorithms are able to attain the same performance without the need of initialization. Hence, the proposed algorithms are not sensitive to the choice of the initial codebook and outperform codebook initialized GLA and SR-D algorithms.

In Chapter 3, the channel/frequency assignment problem (FAP), which is a NP-hard combinatorial optimization problem applied to fixed channel allocation in mobile communication networks, is considered. The FAP is an important problem for today's wireless service providers, which has traditionally been tackled by graph theoretic approaches and more recently with also search methods like simulated annealing, tabu search, and genetic algorithms. However, deterministic annealing has not been applied to FAP before. Depending on the particular network the understanding of frequency assignment varies, but there are two main optimization flavors, one of them is minimizing the spectrum used to meet prescribed interference constraints, and the other is minimizing the total interference that results from a prescribed spectrum allocation. In this thesis we consider the latter (referred to as minimum interference-FAP (MI-FAP)) as our optimization problem, and apply the deterministic annealing together with our proposed complexity reductions. For MI-FAP a variant of simulated annealing has been reported to provide the best known results on some real-world scenarios (the reported work has not yet been released to the public domain at the time of this writing, the results are announced in a Ph.D. thesis published in 2001 [37]). We compare our algorithms with the standard simulated annealing, and provide experimental evidence that our algorithms result in

50% - 80% reduction in total interference in 20% - 50% less running time. Sometimes, together with minimizing the total interference in the network one may also wish to satisfy a certain separation for co-channel nodes (nodes using the same channel, which cause high interference on each other). For these cases we propose a channel blocking algorithm, which is an extension of the DA algorithm. We show that the proposed algorithm effectively reduces the number of co-channel separation violations with a trade-off of small increase in total network interference compared to without using the channel blocking. For example, for a test problem where there were 533 co-channel constraints (number of pairs of nodes with separation less than the required separation for co-channel assignment) to be satisfied, the blocking algorithm reduced the number of violations from an average of 6.6 to 0.9 after blocking with 0.12% increase in average total interference compared to without using channel blocking. We also test the proposed reduced complexity DA algorithm for MI-FAP on a realistic GSM frequency planning scenario obtained from the COST 259 project [37], and show that its performance is highly competitive with the presently best assignment techniques.

In Chapter 4 we introduce a novel constrained vector quantizer (VQ), which we call Seg-VQ. The vast majority of practical image coding systems used today are based on the transform coding paradigm, where image blocks are projected into a series of basis functions, and the expansion coefficients are subsequently quantized. As an extension of the transform coding framework, in our approach codevectors are constrained to be located on a series of line segments in the multidimensional space.

These segments are designed sequentially based on a training set. The advantages of Seg-VQ are twofold: first, the encoding complexity is proportional to the number of segments rather than to the number of codevectors, and second, it can efficiently exploit the directional preferences (correlations) in sources such as images. For image sources, at low dimensions (e.g., 4 by 4 blocks), with the same encoding complexity of TSVQ, Seg-VQ outperforms TSVQ by 0.5 dB at 0.4375 bpp achieving a performance close to the optimal fixed rate unconstrained VQ. At higher dimensions (e.g., 8 by 8 blocks) we use multi-stage Seg-VQ where the input block (as in transform coding) is projected into a series of segments in order to be quantized. We propose two different systems using multiple stages: In the first one the codevectors are designed with Lloyd-Max quantization and are constrained to be on the segments. And in the second one there are no fixed codevectors on the segments; the segments are uniformly quantized depending on the required rate making it more robust for rate adaptation. The latter system is optimal for high rate quantization.

Chapter 2

Reduced Complexity Deterministic Annealing for Vector Quantizer Design

2.1 Introduction

Vector quantization is a source coding technique that approximates blocks (or vectors) of input data by one of a finite number of pre-stored vectors in a codebook. The challenge is to find the set of vectors (or quantization levels) such that a given criterion for the total distortion between the actual source and the quantized source is as small as possible under a constraint on the overall rate [49]. Since distortion depends on the codebook design, vector quantizer design is a key optimization problem to determine the performance of a VQ-based system [66, 109, 24, 67].

The traditionally used VQ design approach is the generalized Lloyd algorithm (GLA) also referred to as the LBG algorithm [80]. The GLA is an extension of Lloyd's algorithm to VQ design, where the original Lloyd algorithm was proposed for scalar quantizer design [81]. There are two necessary conditions for a quantizer

to be locally optimal: the quantizer partition must be optimal for a given set of codevectors, and the set of codevectors must be optimal for the partition. For the mean squared error distortion criterion the first condition implies nearest neighbor (NN) quantization rule and the second condition implies that the codevectors are located at the centroid of their corresponding partition. For a given source the algorithm starts with an initial codebook (a set of codevectors), optimizes the partition by assigning the source vectors to the nearest codevector (this minimizes the distortion for the fixed codebook), then optimizes the codevectors for the partition by replacing each codevector by the centroid of its corresponding partition region. The alternation is repeated until convergence to a local minimum. Hence, the GLA is an iterative descent algorithm, where in each iteration an improvement in performance is achieved compared to the previous one. The GLA has the advantage that it converges to a final codebook relatively quickly, however, the resulting codebook is *locally optimal* since the algorithm gets trapped in a local minimum of the distortion (energy) surface to which the initial codebook is closest. Consequently, the performance of GLA can be poor compared to that of a globally optimal quantizer.

As discussed in Chapter 1 (Section 1.2.2), a powerful approach to reduce the sensitivity of the algorithm to the initial codebook is the introduction of *randomness*. Several randomized optimization techniques have been investigated in the past. In [100] such "random search" techniques are discussed, where the idea is to randomly perturb the system at each iteration and determine the resulting change in

performance. In some of its variations a perturbation is only accepted if the performance increases, otherwise it is rejected; and in other variations perturbations that decrease performance are also accepted under certain conditions. In general, if a random search technique allows temporary decreases in an objective function with nonzero probability, then the algorithm is in the class of *stochastic relaxation* (SR) [47, 117], or stochastic local search techniques.

An important SR technique is the simulated annealing (SA) [117]. As described in Chapter 1 the idea behind the SA method has its origin in the physical process of gradually cooling a molten metal to obtain stable crystal structures. For VQ design the starting state is an initial codebook. In each iteration a new codebook is generated in the neighborhood of the old one, and the new codebook is accepted or rejected according to the Metropolis algorithm [89]. Recall that if sufficient computational resources are devoted, the SA algorithm is guaranteed to yield globally optimal solutions [75]. However, recall also that to achieve the global minimum the temperature schedule should be $T \propto 1/\log t$ with t being the iteration number [47]. Such schedules are not realistic in practical applications.

In order to avoid the computational difficulties associated with SA, a *reduced complexity* quantizer design based on SR is proposed in [117]. This is a simplified version of the SA algorithm achieving similar or slightly better results in much less time under similar temperature reduction schedules. Basically, the reduced complexity SR algorithm is the generalized Lloyd algorithm appended with a stochastic perturbation step, where the perturbations can either be on the encoder

(SR-C) or the decoder (SR-D). Recently, another method that uses a similar randomized search technique is suggested in [94]. Although this technique has an average performance comparable to SR-D, it has a higher complexity. We will give a description of the SR-D technique in the following sections.

In the above approaches random search moves were allowed on the energy surface in order to give the system the ability to avoid local minima. Unlike these SR techniques, a deterministic annealing (DA) approach for optimal vector quantizer design puts the problem in a probabilistic framework, and deterministically optimizes the probabilistic objective function in each iteration [106]. As explained in Chapter 1, in DA there are no random moves on the energy (cost) surface. At high temperatures the energy surface is smoothed, so that the algorithm starts at the global minimum on the smoothed energy surface. And through a careful annealing schedule it traces the global minimum as the energy surface assumes its non-convex "rugged" form with the decreasing temperature. The Gibbs distribution is used to associate sample vectors with codevectors since it maximizes the entropy under the constraint of a given average distortion. Note that the sample vector - codevector associations are not one-to-one, but rather they are one-to-many. In other words, each sample vector is assigned to *all* codevectors in probability: the closer a codevector to a sample vector, the higher its probabilistic assignment to that sample vector. The DA method can construct high performance vector quantizers by avoiding local minima. However, calculation of the association probabilities for each sample vector with *all* the codevectors in each iteration, coupled with the high

computational cost of evaluating the Gibbs distribution, and the slowness of the annealing process, result in a large computational complexity that limits DA's utility for some practice applications.

In this chapter we propose a reduced complexity deterministic annealing approach for VQ design by using soft information processing with simplified assignment measures. We refer to this formulation as the *soft vector quantizer* (SVQ) design. The reduced complexity DA techniques are developed through the design of simple soft-measures that can mimic the effect of the Gibbs distribution used in the standard DA. Hence, while the designed soft-measures are simple enough to facilitate fast computation, they also keep the sacrifice in performance to a minimum by mimicking the Gibbs distribution's functionality. We have also derived the theoretical performance loss due to using a simplified measure instead of the optimal one, and further used the result to derive optimal annealing schedules for the proposed simple soft-measures. In contrast to the standard DA which starts with essentially a single codevector and increases the size of the codebook through iterations, in SVQ the design starts with the required number of codevectors and optimizes their locations through iterations. It is also observed and empirically shown that, when all codevectors are considered the importance of a codevector at a large distance from a given sample vector relative to the other codevectors (in terms of the amount of probability mass associated) decreases exponentially fast even at relatively high temperatures. Hence, major computational gains can be obtained with negligible performance degradation by considering for each sample only a *few*

codevectors, namely those nearest to the given sample vector. We present experimental evidence indicating that through these techniques significant performance gains are achieved by the SVQ algorithms over the traditionally used GLA and over SR-D, where the latter is widely thought to provide near-optimal performance. We also investigate the effect of PNN [39] codebook initialization on GLA, SR-D and SVQ algorithms and show that, while GLA and SR-D benefit significantly from this initialization, at the expense of increased computational complexity, the SVQ algorithms are able to attain the same performance without the need of initialization. Hence, the SVQ algorithms are not sensitive to the choice of the initial codebook and outperform codebook initialized GLA and SR-D algorithms. Compared to the standard DA, the results show drastic reductions in computational complexity with very small sacrifice in performance. It is also shown that appending the SR technique [117] to the SVQ algorithms result in further improvement in performance with decreasing benefits as the performance approaches the optimal, i.e., the better the performance of SVQ the smaller the benefit obtained from SR.

The rest of the chapter is organized as follows: in section 2.2 we summarize vector quantizer design by stochastic relaxation and deterministic annealing techniques. In section 2.3 we explain and formulate the proposed reduced complexity deterministic annealing algorithms. We present experimental results in section 2.4. Finally, section 2.5 concludes the chapter.

2.2 Two Frameworks for Vector Quantizer Design

2.2.1 Vector Quantizer Design by Stochastic Relaxation

Zeger et al. [117] state that the major disadvantages of using SA approach for VQ design are the complexity of computing distortion and the slowness in reaching the thermal equilibrium. Thus, they propose a simplified version of the complicated SA algorithm. The rules to follow at each iteration in their reduced complexity SR algorithm are as follows:

1) Every proposed perturbation is accepted.

2) Simultaneously either perturb all encoder parameters or all decoder parameters (but not both).

3) Perform a repartitioning and centroid computation.

The first rule eliminates the need to calculate distortion in each iteration, which results in great complexity reduction. The second rule speeds up the algorithm by making many changes at once, and the third necessitates a generalized Lloyd iteration at the end of every perturbation (iteration). Therefore, the advantages provided by these 3 rules over the SA approach are drastically reduced complexity and increased convergence. Moreover, the authors state that this algorithm achieves comparable (usually slightly better) results than the SA. Two versions of the algorithm, the encoder perturbation (called SR-C) and the decoder perturbation (called SR-D), are described in [56], where the experimental results demonstrate that SR-D performs better than SR-C. Below, we give a brief description of SR-D.

The SR-D algorithm is realized by perturbing every codevector c_i in the codebook $\mathbf{C}$ at each iteration m as follows,

$$c_i^{(m)} = c_i^{(m)} + z_i\left(T^{(m)}\right) \quad \forall c_i \in \mathbf{C} \tag{2.1}$$

where z is a uniformly distributed, zero-mean noise process, and $T^{(m)}$ is the temperature schedule that controls the noise added at the m^{th} iteration. Although other types of schedules were investigated in [56] the following was found to give the best performance:

$$T^{(m)} = s_c^2\left(1-\frac{m}{I}\right)^p \tag{2.2}$$

In (2.2) s_c^2 is the variance of the sample vector components, I defines the total number of iterations, and $p=3$ was found to give the best performance. The total number of iterations, I, determines the run-time and the performance of the SR-D algorithm. The higher the value of I, the more gradual the annealing process and the closer the result to the global optimum will be, at the expense of longer run-time. A good trade off value was found to be $I=200$. Experimental results for image coder design, speech coder design, and Gauss-Markov sources at different rates are reported by the authors, and significant improvements over GLA are obtained.

2.2.2 Vector Quantizer Design by Deterministic Annealing

In the deterministic annealing algorithm proposed by Rose et al. [106] the main principle is the application of a probabilistic hierarchical clustering process, where

each sample vector in the training set is associated to a cluster with a certain degree of membership. Each cluster is represented by a codevector. Thus, the distortion (energy) function to be minimized is an expected distortion function,

$$E\{D\} = \sum_{x}\sum_{j} P\left(x \in R_j\right) d\left(x, c_j\right), \tag{2.3}$$

where $d\left(x, c_j\right)$ is the distortion measure incurred in representing sample vector x by codevector c_j, and $P\left(x \in R_j\right)$ is the probability that x belongs to the cluster represented by c_j. As a distortion measure the squared distance distortion is used, $d\left(x, c_j\right) = \left\|x - c_j\right\|^2$. The probability distribution used to define the associations is the Gibbs distribution, which is the distribution that maximizes the entropy under the constraint (2.3) [106]:

$$P\left(x \in R_j\right) = \frac{e^{-b\, d\left(x\, c_j\right)}}{\sum_{i=0}^{|\Re|-1} e^{-b\, d\left(x, c_i\right)}}, \tag{2.4}$$

where $|\Re|$ is the cardinality of the cluster set. Notice that the distribution in (2.4) is a form of soft information. In other words, it gives a reliability value to assigning the sample vector x to cluster R_j over the sample space of the cluster set. The parameter b is a term that is inversely proportional to the temperature in the annealing process. Hence, at infinite temperature, which corresponds to $b = 0$, the probability associations are uniform: $P\left(x \in R_j\right) = 1/|\Re|, \forall x, j$. This means that, each sample vector x is equally assigned to all the clusters. As b gets large, i.e., the temperature is lowered, the probability assignments for a sample vector x start to

favor clusters closer to x; the closer a cluster representative c_j to x, the higher its probability assignment. In the limit $b \to \infty$, each sample vector gets assigned exactly to one cluster, namely the cluster whose representative codevector is closest to the sample vector. We refer to this as a *hard* assignment, as opposed to a *soft* assignment where a sample vector gets assigned to more than one representative.

The codevector locations are defined as the weighted average of the sample vectors, where the weights are the probability associations of the sample vectors to the specific codevector being considered:

$$c_j = \frac{\sum_x x P\left(x \in R_j\right)}{\sum_x P\left(x \in R_j\right)}. \tag{2.5}$$

Thus, at $b = 0$ (at infinite temperature) all cluster representatives are at the center of mass of the training set,

$$c_j = \frac{\frac{1}{|\Re|}\sum_x x}{\frac{1}{|\Re|}\sum_x 1} = \frac{1}{K}\sum x, \quad \forall j \tag{2.6}$$

where K is the number of sample vectors in the training set. Essentially, at $b = 0$ there is only one cluster (or Voronoi region), which is the whole set, and a single representative codevector at its center of mass. The hierarchical design algorithm in [106] starts the annealing process with the whole training set as one cluster at $b = 0$, gradually increases b, and re-optimizes by solving (2.5) at each b. As b is increased the probability associations start to get harder, and the system goes through

a sequence of splitting of the clusters at *phase transitions* until the required number of clusters (or codevectors) is reached. The main focus in [106] is the derivation of the critical values of b, denoted b_c, at which these phase transitions occur. These are the optimum splitting temperatures of the clusters and the authors show that in order to be able to attain the global minimum, the splitting of the clusters should be at these critical moments. Note that b does not control the size of the codebook; the system goes through a sequence of phase transitions until the required number of representatives is reached. During the annealing process whenever b reaches b_c for an existing cluster, that cluster splits into smaller clusters. In the limit $b \to \infty$, the associations become hard and each sample vector is associated with one representative as in the GLA algorithm.

The work by Rose et al. [106] provides the theoretical framework explaining how the DA approach avoids local minima, and that through a careful annealing process it can achieve the global minimum. However, for practical applications the algorithm proposed has some drawbacks: besides the added computational complexity that is required for keeping track of the critical temperatures, b_c for each cluster, the annealing of the temperature has to be very slow especially in the vicinity of b_c; and the association probabilities for each sample vector have to be calculated for *all* codevectors. Such complexity is not realistic in many applications. In the next section, we present and analyze reduced complexity techniques for VQ design that result in very significant computational gains with negligible performance

difference. In the sequel we will refer to the method explained in this section as the *standard* DA.

2.3 Reduced Complexity Deterministic Annealing

2.3.1 Introduction

In the proposed algorithms, called *soft vector quantizer* (SVQ) algorithms [32], we formulate the vector quantizer design problem in a probabilistic framework as in the standard DA. However, unlike standard DA each training vector is allowed to be associated in probability in some cases with a *subset* of the codevectors. These probability associations provide a reliability measure on the set of codevectors that the training vector can be mapped to. The soft associations are functions of the relative distances of the codevectors from the training vector and also on the annealing temperature if it is present in the soft assignment measure.

The cost of the computation of the Gibbs soft assignment in (2.4), which involves exponentials, is high; if we count each of the basic operations (addition, subtraction, multiplication and division) to take one floating point operation, *flop*, then an exponential computation takes 8 flops. And since soft assignments for *all* codevectors have to be updated for all sample vectors in every iteration in the standard DA, this results in a system of very high computational complexity. Recall that in (2.4) the term b determines the level of softness of the assignments, so it acts as a *softness control* factor (as b increases assignments get harder). In order to

reduce the computational complexity of the system, we would like to define and use a simpler distribution, preferably one that does not involve exponential terms and converges faster. Let us define a general simple distribution as:

$$p_0\left(c_i \middle| x_k\right) = \frac{m\left(x_k, c_i\right)}{\sum_j m\left(x_k, c_j\right)}, \tag{2.7}$$

where $m\left(x_k, c_i\right)$ is a simple function of its variables which is non-increasing as the distance between x_k and c_i increases. It can be interpreted as the *goodness of match of codevector* c_i *to sample vector* x_k. The denominator is the sum of the goodness of matches with respect to codevectors that we take to be the *relevant* N codevectors to x_k (when $N = |C|$ all of the codevectors are regarded as relevant). Therefore, the softness control of (2.7) is a function of N. Using a simple function, $m\left(x_k, c_i\right)$ coupled with $N \ll |C|$ can result in major computational gains at the expense of some reduction in performance.

For a given set of soft assignments, $p_0\left(c_i \middle| x_k\right), \forall i, \forall k$, the codevector locations can be computed as the weighted average of the sample vectors as in (2.5),

$$\begin{aligned} c_i^* &= \sum_k x_k\, p_0\left(x_k \middle| c_i\right) = \frac{\sum_k x_k\, p\left(x_k\right) p_0\left(c_i \middle| x_k\right)}{\sum_k p\left(x_k\right) p_0\left(c_i \middle| x_k\right)} \\ &= \frac{\sum_k x_k\, p_0\left(c_i \middle| x_k\right)}{\sum_k p_0\left(c_i \middle| x_k\right)} \end{aligned} \tag{2.8}$$

where sample vector probabilities are assumed to be uniform, $p\left(x_k\right) = 1/K$, and where K is the size of the training set. The general iterative framework for updating the

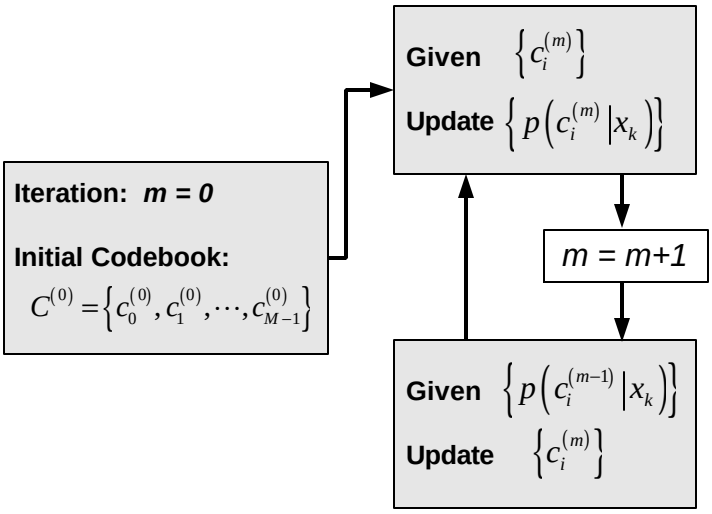


Figure 2.1: The iterative procedure showing the updating of the soft assignments and the codevectors.

soft assignments and codevector locations is shown in Figure 2.1. Note that this framework is independent of the type of soft assignment used; any type of soft assignment measure that can give soft assignment values for the set of codevectors for a given sample vector can be used in this framework.

Ideally, in any annealing algorithm the annealing temperature should start at a very high temperature (theoretically at infinite temperature) and gradually cool down to zero. However, as we have seen in the standard DA this results in a very slow convergence. In the proposed SVQs the temperature is not chosen to be infinite at the start, and we demonstrate that starting with a low temperature and with fixed (required) number of codevectors, it is possible to achieve near optimal performance. Starting with a low temperature means starting the algorithm with a non-convex energy surface. We show that introduction of controlled randomness into the iterations has the potential to improve the results due to the non-convexity of the

energy surface. The stochastic relaxation technique described in section 2.2.1 is an excellent candidate for this purpose because of its easy integration into an iterative algorithm and its low complexity.

2.3.2 Reduced Complexity Gibbs Distribution for VQ Design

In order to facilitate a practically usable algorithm and circumvent the slowness of the standard DA, we start the annealing process at temperatures low enough to reduce convergence time, but at the same time high enough for the principles of soft association to take effect. We know that as a result of soft association every sample vector x_k has a certain degree of belonging to *all* of the codevectors in the codebook. However, while it is computationally complex to take all the soft associations into account no matter how small they are, the effect of very small soft associations on (2.8) and on the converged codebook is negligible. Therefore, in practice, those association probabilities that are very close to zero can be set to zero. At this point, a logical approach would have been to define a threshold and set all the associations below this threshold to zero. But this would only save us computational cost in (2.8) , we would still need to calculate all the soft associations and compare them with the threshold. In order to further reduce the cost, we decided to select the N nearest codevectors from a given sample vector, compute the soft associations only for the closest N codevectors and set the other $|C| - N$ associations to zero. Note that we are not reducing the size of the codebook, that is fixed; what we do is, for each

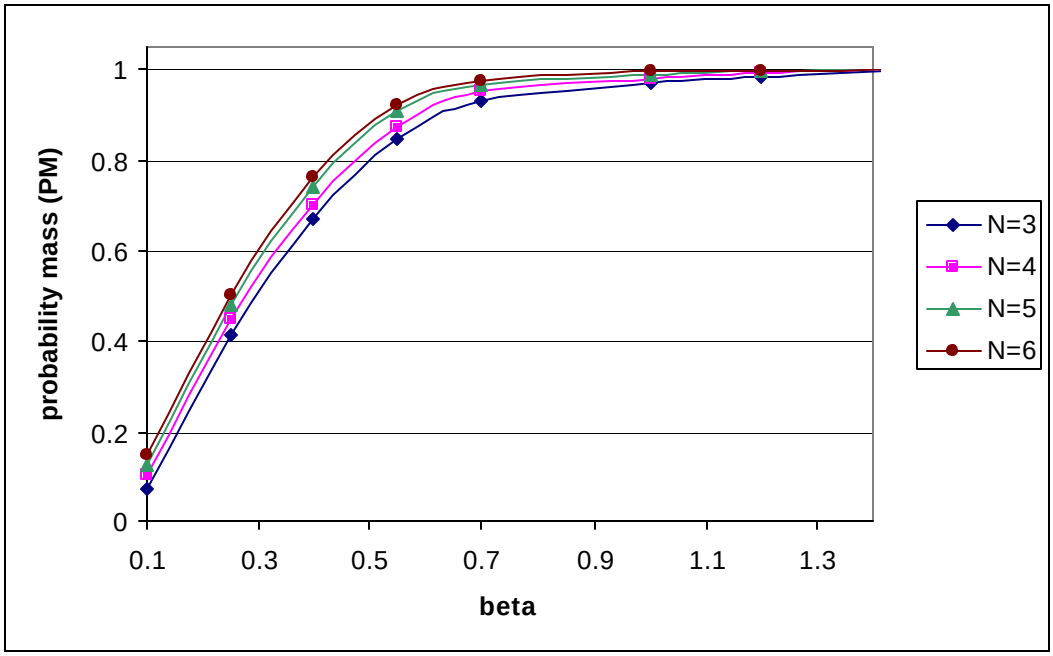


Figure 2.2: Probability mass contained in the nearest N codevectors from a randomly selected sample vector, at different temperature values. Codebook size = 128, vector dimension = 16, zero-mean, unit variance Gaussian source.

sample vector we only compute the soft information for the nearest N codevectors instead of the whole codebook. In this way, only the distances from a given sample vector to the codevectors need to be computed and the N nearest codevectors are determined. Denoting $\mathrm{N}(x_k, N)$ to be the nearest N codevectors from a given x_k, the soft information is computed by,

$$p(c_i \mid x_k) = \frac{e^{-b\, d(x_k, c_i)}}{\sum_{c_j \in \mathrm{N}(x_k, N)} e^{-b\, d(x_k, c_j)}}. \qquad (2.9)$$

where $d(x_k, c_i) = \|x_k - c_i\|^2$. We can assess from Figure 2.2 that taking all of the codevectors into consideration does not justify the required computational complexity. The figure shows the total probability mass contained in the nearest N

codevectors from a randomly chosen sample vector for increasing b (i.e., decreasing temperature) during the design of a size $|C| = 128$ codebook using the standard DA algorithm. Each point on the graph is found by summing the probability mass within the nearest N codevectors from each x_k, and averaging over all x_k,

$$PM(N) = \frac{1}{K}\sum_{k=0}^{K-1}\sum_{c_i \in N(x_k, N)} p(c_i | x_k), \tag{2.10}$$

where K again is the size of the training set. We can observe from the figure that even at low b (high temperature) $0 < b < 1.0$, a considerable amount of the probability mass is confined within a small number of codevectors compared to the size of the codebook. For example, in Figure 2.2, at $b \cong 0.25$ the nearest 3 codevectors account for more than 40% of the probability mass on average, and the rest 125 codevectors account for less than 60%. Experimentally we have found $N = 4$ to be a good trade-off value between performance and complexity. In other words, results obtained by setting $N = 4$ and with $N = |C|$ (i.e., using all the codevectors in the codebook) resulted in negligible performance difference, however, the computational savings are significant, especially for large codebooks (e.g., $|C| = 128, 256,$ *etc.*). A comparison of $N = 4$ and $N = |C| = 128$ using the same annealing schedule is given in Table 2.1. The loss in performance incurred by considering only the nearest 4 codevectors for each sample vector instead of the whole codebook is clearly negligible. In exchange for this negligible loss, a factor of

N	*Ave. SNR*	*Ave. CPU Time*
4	3.595 dB	136 sec.
128	3.598 dB	16483 sec.

Table 2.1: Average performance and running time comparison for $N = |C| = 128$ and $N = 4$. The source is uncorrelated Gaussian, the vector dimensions are 16, and soft information measure is reduced complexity Gibbs distribution. The results are averages over 20 experiments (details on experimental set-up are in Experimental Results section).

about 120 speed-up in running time is achieved which is a highly significant complexity reduction.

The proposed algorithm is shown in Figure 2.3, where the iterations starts with $N = 4$, $b = 0.3$ and an initial codebook C_0. Note that in the algorithm flowchart $t = 1/b$ is shown and the reduction factor $r = 0.995$ is used, $t^{(m)} = t^{(m-1)} \cdot r$. Note that $r = 0.995$ for t corresponds to an increment factor of $k = 1.005$ for b, $b^{(m)} = b^{(m-1)} \cdot k$. At each iteration, we gradually increase b (i.e., decrease temperature), update the soft information according to (2.9) and re-optimize the codevector locations using (2.8). We can then apply the codevector perturbation as explained in section 2.2.1 (also see [117]). As the temperature decreases the *softness* of the codevector associations also decreases; in other words, the closer codevectors to sample vector considered become increasingly important, while those that are away become less and less important. In the limit, when temperature approaches zero the algorithm becomes like the GLA and at each iteration all the probability mass for a given sample vector is assigned to the nearest codevector, or put differently, we reach the nearest neighbor condition.

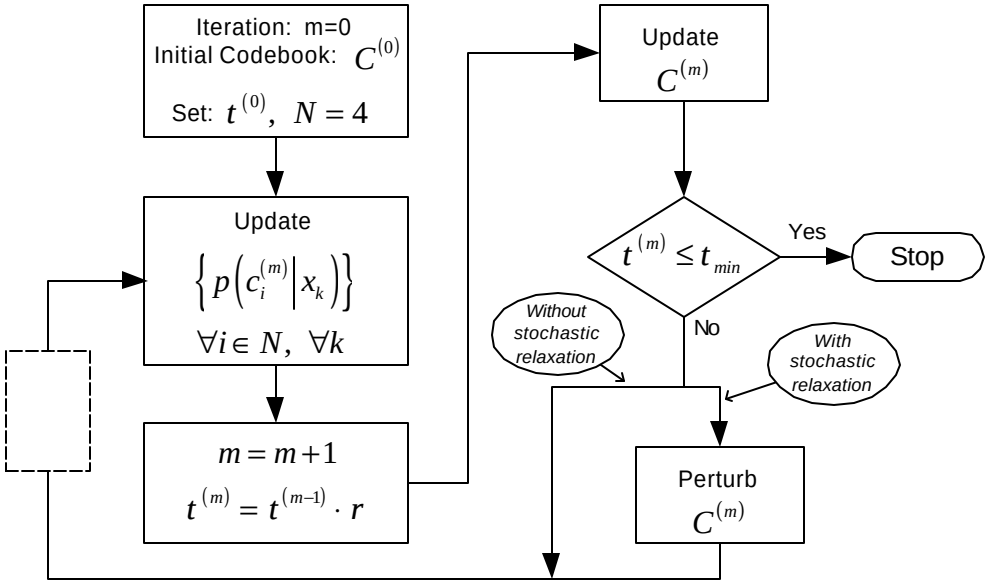


Figure 2.3: The flowchart for the reduced complexity Gibbs distribution algorithm.

In the above scheme we kept N fixed. When we need to design quantizers for very large codebook sizes (e.g., $|\mathrm{C}| = 512, 1024, \ldots$) we may find it useful to use a larger N (e.g., 10, 12, 15, …). However, we know that while the N^{th} furthest away codevector from a given sample vector plays an important role (has large probability mass) in the early iterations, its importance decreases in each iteration. As the temperature decreases (b increases) the probability mass is gradually transferred from the distant to the closer codevectors. Hence, after a while the N^{th} codevector will contain negligible mass and it can be discarded without any significant effect on the final performance. Gradually, the above will be the case for the $(N-1)^{th}$, $(N-2)^{th}$, … codevectors. In order to reduce the complexity with respect to the fixed N scheme, without affecting system performance, we can append the simple

algorithm shown in Figure 2.4 to the algorithm in Figure 2.3 (in the dashed block). As shown in Figure 2.4, whenever the average probability mass of the nearest $N-1$ codevectors, $PM(N-1)$ (2.10) exceed a certain mass p (typically $p=0.99$), N is reduced by one:

$$\textit{if} \quad PM(N-1)=\frac{1}{K}\sum_{k}\sum_{c_i\in \mathrm{N}(x_k,N-1)} p(c_i|x_k) > p, \quad \textit{then} \quad N=N-1. \qquad (2.11)$$

When N is large the cumulative effect of gradually decreasing the number of nearest neighbors to be taken into account results in considerable complexity reduction which was not possible in the fixed N scheme. However, it is important to note that in the case of small N (e.g., $N=4$) we may not have any computational gain or we may even increase the computational cost by using the gradual reduction scheme. This is a result of the fact that the small gain (from N being small) obtained by gradually decreasing N will be consumed by the computation of $PM(N-1)$.

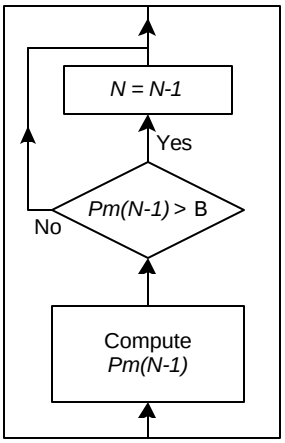


Figure 2.4: The scheme for gradual reduction of the number of nearest codevectors to be taken into account, N

Instead, when N is large enough, the reduction in computational cost obtained by reducing N surpasses the added cost of the computation of $PM(N-1)$.

2.3.3 Low Complexity Soft Information Measures for VQ Design

2.3.3.1 Single Triangular Membership Function

As previously stated, in order to reduce the computational complexity of the system we need to use in (2.7) a less complex distributions than the optimal Gibbs distribution. One of the simplest distributions that readily comes to mind is the "inverse Euclidean distance" distribution, in which, for a given sample vector x_k, the "importance" of the codevectors decrease with increasing distance from x_k.

"Inverse Euclidean distance" can be used as a soft information measure as follows,

$$p(c_i \mid x_k) = \frac{\frac{1}{d_i}}{\sum_{j=0}^{N-1} \frac{1}{d_j}} \tag{2.12}$$

The distances in (2.12) are the Euclidean norms between x_k and the codevectors (n is the vector dimension),

$$d_i = d(x_k, c_i) = \sqrt{(x_{k,0} - c_{i,0})^2 + (x_{k,1} - c_{i,1})^2 + \cdots + (x_{k,n-1} - c_{i,n-1})^2} \ .$$

The number of codevectors to be taken into consideration for each x_k can be determined by a circle centered on x_k with a radius R, where all codevectors closer than R to x_k constitute the N nearest codevectors. The radius R decreases from one iteration to the next, $R^{(m)} = R^{(m-1)} r$, where $0 < r < 1.0$.

Another soft information measure can be defined using a triangle function as shown in Figure 2.5 (we use absolute distances between a sample vector and each of the codevectors, for clarity of presentation both sides of the triangle is used in Figure

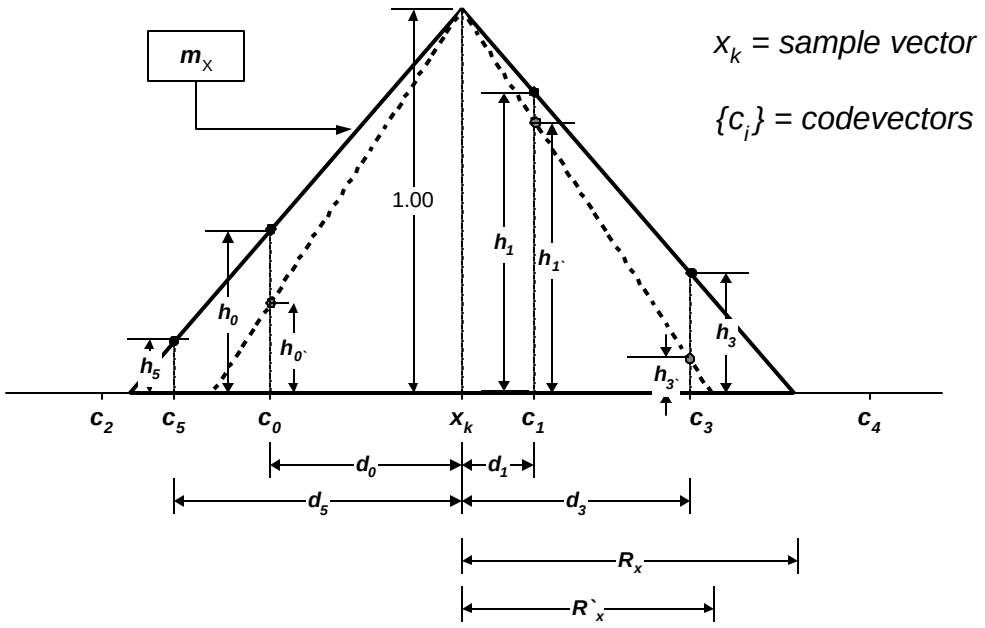


Figure 2.5: Triangular membership function used as a soft information measure. Codevectors within the spread of the function comprise the nearest N codevectors for the considered sample vector.

2.5). The function with height $h = 1$ and a spread R_x, will use all the codevectors within an Euclidean distance R_x of x_k, i.e., the N nearest codevectors. Using the fuzzy systems terminology, we can define this triangle function as the *membership function of* x_k and denote it by m_x. The soft associations are computed by using the *heights* of the membership function corresponding to the Euclidean distances of the codevectors from x_k,

$$p(c_i \mid x_k) = \frac{h_i}{\sum_{j=0}^{N-1} h_j}. \qquad (2.13)$$

The spread R_x decreases gradually in each iteration giving more and more importance to the nearer codevectors as the iterations increase. At the limit, when

only one codevector stays within the nearest neighbor set, i.e., $N=1$, the soft information measure becomes hard and all the probability mass gets assigned to the nearest codevector. Note that as the spread is decreased, for some sample vectors $N=1$ will be reached earlier than the others since the nearest codevector distance cannot be the same for each sample vector. As the spread continues to decrease, at some point for some sample vectors, $R_x < d(x_k, c_i) \quad \forall i$. In these cases, the algorithm assigns all the probability mass to the nearest codevector. The spread at the m^{th} iteration is controlled by a geometric schedule as in the Gibbs case:

$$R_x^{(m)} = R_x^{(m-1)} r \tag{2.14}$$

where r is the reduction factor, $0 < r < 1.0$. The soft information measure in (2.13) can be defined in terms of the spread, R_x and the distances, $d_i = d(x_k, c_i)$ using triangular similarities, where the height of the triangle is $h=1$:

$$\frac{h}{R_x} = \frac{h_i}{R_x - d_i}$$

$$\Rightarrow \quad h_i = \frac{h}{R_x}(R_x - d_i)$$

$$\Rightarrow \quad h_i = \frac{R_x - d_i}{R_x} \tag{2.15}$$

Therefore, (2.13) becomes,

$$p(c_i \mid x_k) = \frac{R_x - d_i}{\sum_{j=0}^{N-1}(R_x - d_j)}$$

$$= \frac{R_x - d_i}{NR_x - \sum_{j=0}^{N-1} d_j} = \frac{1 - \frac{d_i}{R_x}}{\sum_{j=0}^{N-1}\left(1 - \frac{d_j}{R_x}\right)} \tag{2.16}$$

We call $1-\frac{d_i}{R_x}$ as the "normalized distance." The soft information measure in (2.16) is a better one than the inverse Euclidean distance measure in (2.12), because this can mimic the effect of the temperature reduction in the Gibbs distribution much better. Consider the distances of the 6 codevectors, $\{c_0, c_1, c_2, c_3, c_4, c_5\}$ from a sample vector x_k shown in Figure 2.5. Using (2.12) with a radius $R = R_x$ the codevectors $\{c_0, c_1, c_3, c_5\}$ are taken as the set of nearest codevectors from x_k. When the radius is decreased to $R' = R'_x$ the set of nearest codevectors become $\{c_0, c_1, c_3\}$. But note that the distances $\{d_0, d_1, d_3\}$ are not affected at all by the decrease in the radius. With (2.12) the only time the soft assignments will change as the radius is decreased is when a codevector is left out of the circle of radius R. However, using (2.13) with the normalized distances, where $h_i = 1-\frac{d_i}{R_x}$, the *heights* (normalized distances) get affected by the reduction in the spread R_x as seen in the equivalent expression in (2.16). As the spread decreases the normalized distances approach zero assigning more weight to the closer codevectors. This is desired in order to approximate the effect of the temperature reduction in the Gibbs distribution; in other words, as the spread decreases the codevectors closer to x_k should increase their share of the soft assignment in conformity with their distances from x_k. Hence, the normalized distance defined soft information measure (2.16) is better than the Euclidean distance-defined measure (2.12) in terms of mimicking the Gibbs soft measure. This will be demonstrated by our experimental results. Note also that the computational cost of computing one soft assignment using (2.12) requires $5N + 4$ flops, whereas

using (2.16) it requires $N+7$ flops, counting addition, subtraction and multiplication as one flop and division as four flops (N is the number of codevectors taken into computation). Hence, for $N \geq 1$: $N+7 < 5N+4$, implying that (2.16) is also less costly than (2.12). Recalling that an exponential computation takes 8 times more than a basic operation (8 flops compared to one flop of operation time for a basic operation), then (2.9) takes $N(8+1+1)+4 = 10N+4$ flops, which is much larger than $N+7$. Therefore, the height-defined triangular soft information measure is a computationally less complex distribution than the Gibbs distribution.

2.3.3.2 Multi-Triangular Membership Function

The superiority of the Gibbs distribution comes from its constituent exponential functions, which take the Gaussian form for the squared-distance distortion. This allows the system to gracefully transfer the probability mass to the nearer codevectors provided that the temperature is lowered very gradually; hence, the system efficiently traces the global minimum until the temperature reaches zero. This being the case, we direct our attention to better approximate the effect of the exponential functions in our simplified soft information measure. To this end, we consider multiple membership functions, that is, a membership function not only for x_k, but for all the N nearest codevectors as shown in Figure 2.6. Compared to the case with a single membership function, multiple membership functions increase the effective region beyond the spread of m_x and transfer the probability mass to the

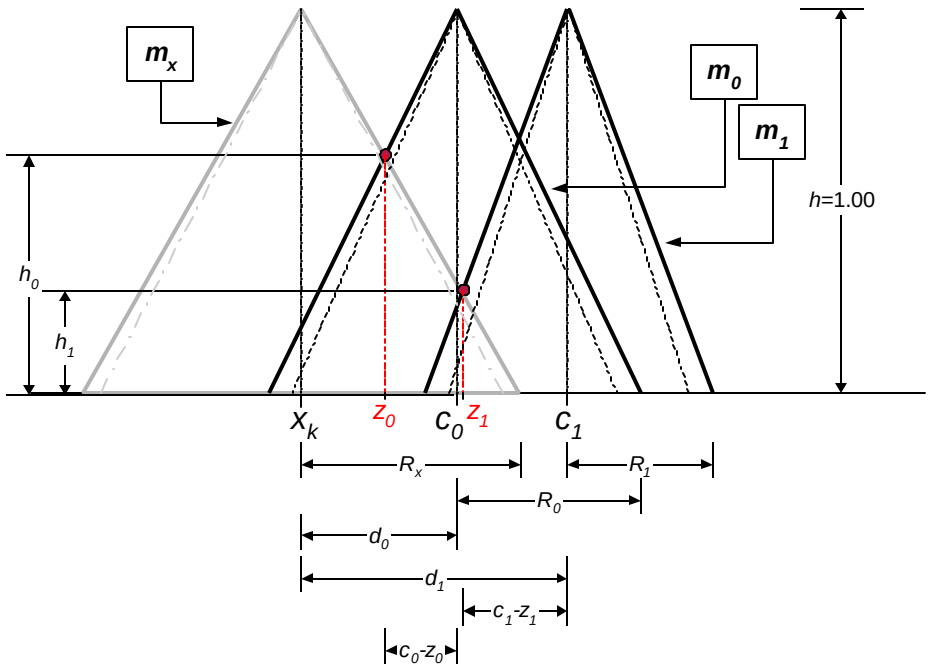


Figure 2.6: Multi-triangular membership function as a soft information measure. Codevectors within the cumulative spread of the multi-function comprise the nearest N codevectors for the considered sample vector.

nearer codevectors more gracefully as the spreads are decreased. The spread of m_x is R_x, and the spreads of the codevector membership functions m_i are R_i. Note that in this case the "heights" h_i are defined at the crossing points of m_i with m_x; their projections are denoted as z_i on the horizontal axis in the figure. Using the symmetry of the functions we can write the following two equations, (2.17) and (2.18):

$$\frac{h}{R_i} = \frac{h_i}{R_i - |c_i - z_i|} \quad \Rightarrow \quad h = \frac{R_i h_i}{R_i - |c_i - z_i|} \tag{2.17}$$

and

$$\frac{h}{R_x} = \frac{h_i}{R_x - |z_i - x_k|} \quad \Rightarrow \quad h = \frac{R_x h_i}{R_x - |z_i - x_k|} \tag{2.18}$$

To define h_i first we need to derive z_i. By equating (2.17) and (2.18) we get,

$$R_i\left(R_x - |z_i - x_k|\right) = R_x\left(R_i - |c_i - z_i|\right)$$
$$\Rightarrow \quad R_i z_i - R_i x_k = R_x c_i - R_x z_i$$

$$\Rightarrow \quad z_i = \frac{R_i x_k + R_x c_i}{R_x + R_i}. \tag{2.19}$$

Since our reference point for the Euclidean norms is x_k, we can set $x_k = 0$ and $c_i = d_i$:

$$\Rightarrow \quad z_i = \frac{R_x d_i}{R_x + R_i}. \tag{2.20}$$

Finally, substituting (2.20) into (2.17) or (2.18), with $h = 1$ we get h_i,

$$\Rightarrow \quad h_i = \frac{R_x + R_i - d_i}{R_x + R_i} = 1 - \frac{d_i}{R_x + R_i}. \tag{2.21}$$

The soft assignments are obtained by substituting (2.21) into (2.13),

$$p(c_i \mid x_k) = \frac{h_i}{\sum_{j=0}^{N-1} h_j}$$
$$= \frac{1 - \frac{d_i}{R_x + R_i}}{\sum_{j=0}^{N-1}\left(1 - \frac{d_j}{R_x + R_j}\right)} \tag{2.22}$$

Note that the spreads of the codevector membership functions, R_i are not equal. We have observed that the soft information measure in (2.22) better approximates the Gibbs measure if the spreads of the codevector membership functions, R_i decrease as

the distance of the codevectors from x_k increase, as opposed to having all R_i the same. We have used the following spread adjustment as a function of distance: Let the codevectors $\tilde{c}_0, \tilde{c}_1, \tilde{c}_2, \cdots$ be ordered in non-decreasing distances from x_k, $\tilde{d}_0, \tilde{d}_1, \tilde{d}_2, \cdots$. Normalize these distances with the distance of the nearest codevector to x_k, i.e., with $d_{\min} = \tilde{d}_0$:

$$\frac{\tilde{d}_0}{d_{\min}} \le \frac{\tilde{d}_1}{d_{\min}} \le \frac{\tilde{d}_2}{d_{\min}} \le \cdots \tag{2.23}$$

Let R be the initial spread from which all R_i are to be obtained as follows:

$$R_0 = \frac{R}{\tilde{d}_0 / d_{\min}} = R, \qquad R_1 = \frac{R}{\tilde{d}_1 / d_{\min}}, \qquad R_2 = \frac{R}{\tilde{d}_2 / d_{\min}}, \cdots$$

$$\Rightarrow \quad R_i = \frac{R d_{\min}}{\tilde{d}_i} \tag{2.24}$$

Note that in this case the annealing process is controlled by two spreads, namely by R_x and R. We can use geometric reduction schedules as before,

$$R^{(m)} = R^{(m-1)} r \quad \text{and} \quad R_x^{(m)} = R_x^{(m-1)} r \tag{2.25}$$

where $0 < r < 1.0$.

The algorithm for the low complexity soft information measures, both for single membership function case and multiple membership function case is shown in Figure 2.7. The initial spreads used were $R_x^{(0)} = g \cdot s_X^2$ and $R^{(0)} = g \cdot s_X^2$, where s_X^2 is the variance of the training set components, and $g = 3.3$ was found to give the best performance. Larger values give similar results and are more complex. As the spread(s) decrease(s) the *softness* of the codevector associations decrease; in the

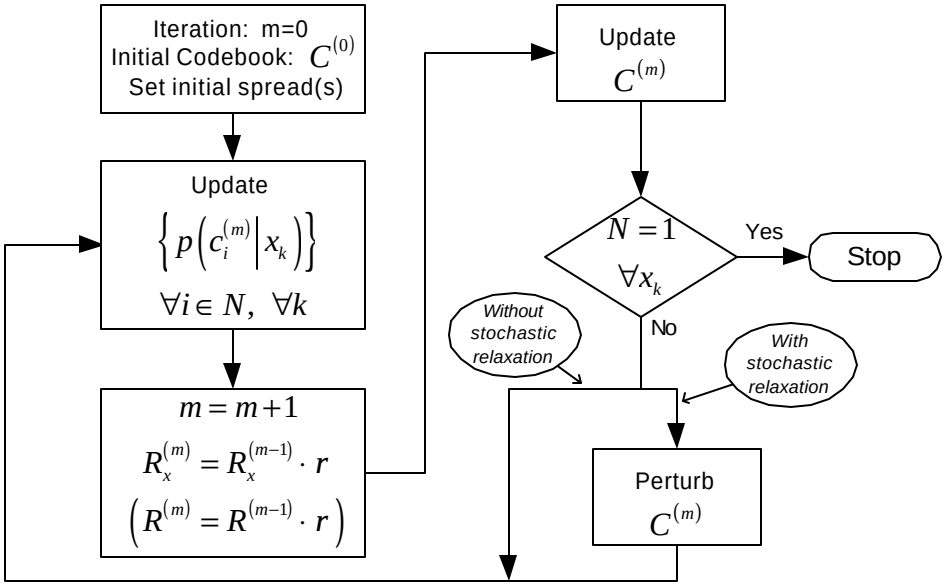


Figure 2.7: The flowchart for the low complexity soft information measure algorithm.

limit, when $N = 1$ for all sample vectors i.e., when all the probability mass for each sample vector x_k is assigned to the nearest codevector from x_k, the algorithm becomes like the GLA algorithm.

2.3.4 Optimal Temperature Schedule

In the previous section we have proposed two low complexity soft assignment measures, namely, the triangular soft information measure and the multi-triangular soft information measure as simplified ways of computing the soft assignments. Although these measures will significantly reduce the computational cost of the soft assignments compared to the Gibbs soft measure, this improvement in computational cost will come in exchange for some loss in performance since Gibbs is the optimal

soft measure. However, the loss in performance can be minimized if we can find temperature reduction schedules for the low complexity measures that can follow the Gibbs b schedule such that the "distance" between the two distributions is minimized. In other words, for a given codevector the probability mass assigned by the low complexity (non-optimal) measure and the optimal Gibbs measure is as small as possible. By definition this is the minimization of the L_1 distance between the two distributions [29],

$$\left\| p_0\left(c|x\right) - p_G\left(c|x\right) \right\|_1 = \sum_i \left| p_0\left(c_i|x\right) - p_G\left(c_i|x\right) \right|. \tag{2.26}$$

Let the low complexity measure be the triangle soft assignment measure, then we would like to find the spread reduction schedule R_x for a given Gibbs b schedule that minimizes (2.26). But note that minimizing (2.26) is equivalent to minimizing the relative entropy between $p_0\left(c|x\right)$ and $p_G\left(c|x\right)$, $D\left(p_0\left(c|x\right) \| p_G\left(c|x\right) \right)$, since we know from [29] that,

$$D\left(p_0\left(c|x\right) \| p_G\left(c|x\right) \right) \geq \frac{1}{2\ln 2} \left\| p_0\left(c|x\right) - p_G\left(c|x\right) \right\|_1^2 \tag{2.27}$$

with equality when $p_0 = p_G$. Although it is highly intuitive that in order to minimize the performance difference between a simplified soft-measure and the optimal soft-measure the relative entropy between them is minimized, in the Appendix it is shown that this is indeed the case. The error analysis in the appendix shows that at a given system entropy (softness) the performance loss in terms of distortion between two distributions (soft-measures) is a function of the relative entropy between them.

Hence, minimizing the relative entropy minimizes the distortion penalty paid for using a simplified soft-measure.

We will demonstrate that the relative entropy is approximately minimized when the variances of the two distributions, $p_0\left(c\middle|x\right)$ and $p_G\left(c\middle|x\right)$ are equal. The variances of $p_G\left(c\middle|x\right)$ and $p_0\left(c\middle|x\right)$, respectively, are (the lower limits of the integrals start from zero because we use absolute distances between sample vector and each of the codevectors):

$$\begin{aligned}\mathrm{var}_G &= \mathbb{E}_G\left\{z^2\right\} - \left(\mathbb{E}_G\left\{z\right\}\right)^2 \\ &= \frac{\int_{z=0}^{\infty} z^2 \mathrm{e}^{-bz^2}\,dz}{\int_{z=0}^{\infty} \mathrm{e}^{-bz^2}\,dz} - \left(\frac{\int_{z=0}^{\infty} z\,\mathrm{e}^{-bz^2}\,dz}{\int_{z=0}^{\infty} \mathrm{e}^{-bz^2}\,dz}\right)^2 = \frac{1}{2b} - \left(\sqrt{\frac{1}{p\cdot b}}\right)^2 \\ &= \frac{p-2}{2p\cdot b}\end{aligned} \tag{2.28}$$

$$\begin{aligned}\mathrm{var}_0 &= \mathbb{E}_0\left\{z^2\right\} - \left(\mathbb{E}_0\left\{z\right\}\right)^2 \\ &= \frac{\int_{z=0}^{R_x} z^2\left(R_x - z\right)dz}{\int_{z=0}^{R_x}\left(R_x - z\right)dz} - \left(\frac{\int_{z=0}^{R_x} z\left(R_x - z\right)dz}{\int_{z=0}^{R_x}\left(R_x - z\right)dz}\right)^2 = \frac{R_x^2}{6} - \left(\frac{R_x}{3}\right)^2 \\ &= \frac{R_x^2}{18}\end{aligned} \tag{2.29}$$

Equating (2.28) and (2.29), and solving for R_x, we get,

$$R_x = \sqrt{\frac{9\left(p-2\right)}{p\cdot b}}\,. \tag{2.30}$$

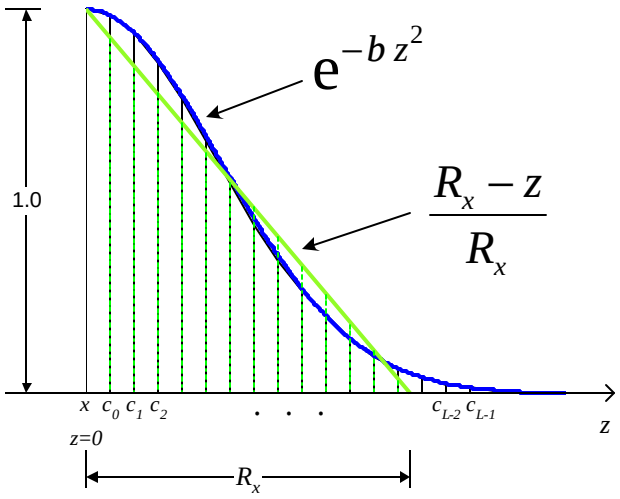


Figure 2.8: An instance of the Gibbs membership function with parameter β and an instance of the triangular membership function with parameter R_x is shown. There are L codevectors at increasing distances from sample vector *x*.

Hence, using (2.30) we can obtain a schedule for R_x given a schedule for b . We need to verify that the relationship in (2.30) minimizes the relative entropy.

We have used the set up in Figure 2.8 to show that for a given b for the Gibbs distribution, the spread R_x obtained by (2.30) for the triangle distribution minimizes the relative entropy. In the figure there are a set of L codevectors at increasing distances from a sample vector x. For each b in a set $\{b_1, b_2, \cdots, b_m\}$, where $0 < b_1 \ll 1$ and with small increments $b_k = b_{k-1} + \Delta b$, the soft Gibbs assignments of the codevectors are computed using the Gibbs soft information measure,

$$p(c_i|x) = \frac{e^{-b\,d(x,c_i)}}{\sum_{j=0}^{L-1} e^{-b\,d(x,c_j)}}.$$

Then, starting with a small value for the spread R_x, $R_x \ll 1$, the soft triangular assignments are computed for the codevectors using the triangular soft information measure,

$$p(c_i \mid x) = \frac{(R_x - d_i)/R_x}{\sum_{j=0}^{L-1}(R_x - d_j)/R_x} = \frac{R_x - d_i}{\sum_{j=0}^{L-1}(R_x - d_j)}.$$

Using the Gibbs and the triangular soft assignments obtained on the set of codevectors the relative entropy $D\left(p_0(c|x) \| p_G(c|x) \right)$ is computed. Without changing b, the spread R_x is incremented by a small value ΔR_x, $R_x + \Delta R_x$, the soft triangular assignments are computed for $R_x + \Delta R_x$, and the relative entropy between the Gibbs assignments and the new triangular assignments is computed. The process of incrementing R_x, computing the soft triangular assignments and computing the relative entropy is repeated until the value of R_x that minimizes the relative entropy $D(p_0 \| p_G)$ is found (the relative entropy $D(p_0 \| p_G)$ is a convex function of (p_0, p_G)). At this point, for the fixed b we have the spread R_x that minimizes the relative entropy. The b is incremented by a small value to $b + \Delta b$ and the Gibbs soft assignments are computed. Again, for the updated b the value of R_x that minimizes $D(p_0 \| p_G)$ is found through an exhaustive search, and the process is repeated. The resulting minimum relative entropy curve is shown in Figure 2.9 by the solid line. On the other hand, the dashed curve is obtained using the model (2.30) to get R_x for each b. We can see that the derived relation in (2.30) can

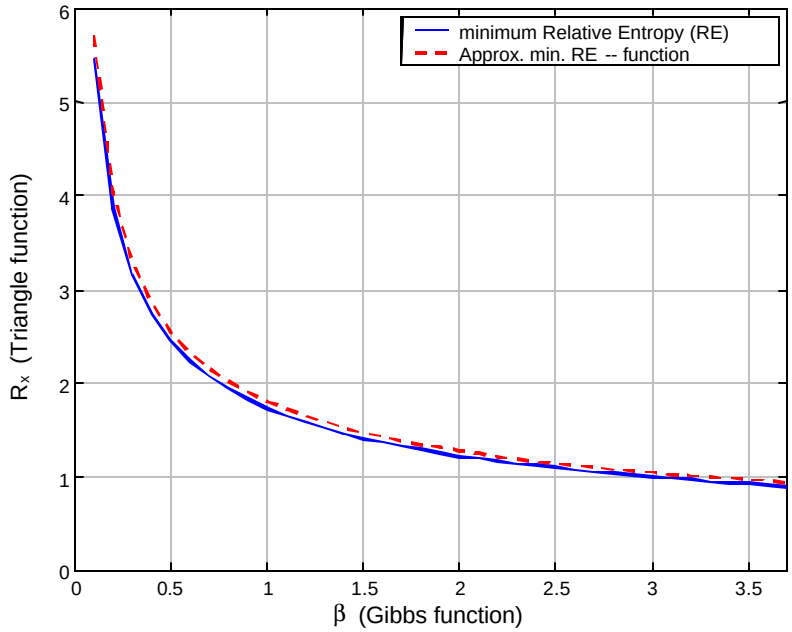


Figure 2.9: Plot showing the minimum relative entropy between the triangular soft measure and the Gibbs soft measure at various spread R_x and b pairs. The solid curve is obtained by sequentially searching increasing values of the spread R_x that gives the minimum relative entropy for a given value of b. While the dashed curve is obtained using the derived relationship between R_x and b to give the minimum relative entropy.

highly approximate the minimum relative entropy curve, and hence the best R_x schedule for a given b schedule.

The reduced complexity Gibbs algorithm and the low complexity soft measure algorithm for the triangular membership function using two different spread reduction schedules are used to design codebooks of size 128 and 256 (for details on experiments see the Experimental Results section). The results are shown in Table 2.2. Of the two schedules for the triangular soft information measure, the first one is

Codebook Size	Reduced Complexity Gibbs Soft Information Measure		Low Complexity Soft Information Measure – Triangular. Geometric spread reduction		Low Complexity Soft Information Measure – Triangular. Gibbs guided spread reduction	
	Ave. SNR	*Ave. CPU time*	*Ave. SNR*	*Ave. CPU time*	*Ave. SNR*	*Ave. CPU time*
128	3.595 dB	136 sec.	3.392 dB	91 sec.	3.411 dB	91 sec.
256	5.210 dB	329 sec.	4.919 dB	232 sec.	4.952 dB	232 sec.

Table 2.2: Comparing the geometric and Gibbs guided spread (temperature) reduction for the triangular membership function for the design of 128 and 256 sized codebooks for uncorrelated Gaussian source with vector dimensions 16.

the geometric spread reduction given in (2.14) with $r = 0.995$, and the second one is obtained using (2.30) and the Gibbs schedule (referred to as Gibbs guided spread reduction in the Table). Observe from the results in the Table 2.2 that the performance of the triangular soft information measure using the Gibbs guided spread reduction schedule outperformed the geometric spread reduction in same number of iterations (running time). Therefore, for a given b – schedule the relation in (2.30) provides a better R_x – schedule than a geometric reduction that is suggested in most annealing algorithms. Note that b – schedule is itself geometric, $b^{(m)} = b^{(m-1)} \cdot k$ where $k > 1.0$ ($k = 1.005$ is suggested in section 2.3.2) as used in most annealing algorithms. But since the Gibbs soft information measure is the optimal measure, following the b – schedule in a simple soft information measure that approximates the Gibbs measure, increases the simple soft information measure's performance as demonstrated above. Note also that to obtain the b –

schedule the Gibbs algorithm need not be run, it can be obtained using

$b^{(m)} = b^{(m-1)} \cdot k$, $k > 1.0$.

2.4 Experimental Results

We now present the results obtained when our algorithms were used to design codebooks of various sizes, with various sources. The results are compared with other algorithms of interest, namely, GLA, SR-D and standard DA. We first compared them without codebook initialization on Gauss-Markov sources. Then, since both GLA and SR-D are sensitive to the choice of the initial codebooks, we also compared the effect of initialization on these algorithms. We used the pairwise nearest neighbor (PNN) algorithm [39] to initialize the codebooks. In order to give a more comprehensive analysis using the PNN initialization we also compared these algorithms on human speech sampled at 8 kHz and on image sources from the USC image database. Our quoted execution times (CPU times) are based on those obtained with an *Intel PIII - 550 MHz* machine.

2.4.1 Without Codebook Initialization

The training sources we considered were two cases of first order Gauss-Markov sources, one with correlation coefficient $a_0 = 0.0$ (uncorrelated source) and the other with $a_0 = 0.9$ (correlated source). We blocked 16384 samples and 24576 samples,

into 1024 16-dimensional training vectors and 1024 24-dimensional training vectors, respectively; and designed codebooks of sizes 32, 64, 128 and 256 for both training sets.

We designed codebooks for the following algorithms where in the plots the appended "a" means without stochastic perturbation (e.g., in (1.) below, SVQ-Ga would mean without perturbation):

1. SVQ-G: Soft vector quantizer design using the reduced complexity Gibbs distribution as the soft measure.
2. SVQ-E: Soft vector quantizer design using the inverse Euclidean distance distribution as the soft measure.
3. SVQ-T: Soft vector quantizer design using the height-defined distribution with single triangular membership function as the soft measure.
4. SVQ-N: Soft vector quantizer design using the height-defined distribution with multiple triangular membership functions as the soft measure.
5. VQ-DA: Vector quantizer design using the standard deterministic annealing [106].
6. SR-D: Vector quantizer design using the reduced complexity decoder perturbation algorithm [117].
7. GLA: Vector quantizer design using the generalized Lloyd algorithm [80].

Algorithms 1 – 4 incorporate stochastic perturbation. When no perturbation is present the algorithm is abbreviation with an extension "a", e.g., in algorithm (1.) SVQ-Ga would mean without perturbation. For each case, except VQ-DA, the

average performances are computed for 20 different initial codebooks, where for each codebook design the same set of initial codebooks are used, allowing us to compare the average performances of the different algorithms. Recall that VQ-DA uses the center of mass of the training set as the initial codebook, so its performance with this initial condition is recorded. The performance measure used is signal-to-noise ratio (SNR), defined as: $SNR = 10 \cdot \log_{10}(P_s/D)$, where P_s is the signal power and D is the distortion per sample. The SR-D algorithm was run for 200 iterations as in [117], and the GLA was run until convergence.

The baseline performance for each source and rate $(rate = \log_2 |C|/dim, \text{in bits/sample})$ were computed by averaging the GLA results obtained from the 20 different initial conditions. The performances of the first 6 algorithms (listed above, both with and without perturbation) compared with the GLA performances are shown in Figures 2.10 – 2.13. In all cases, the reduced complexity DA algorithms (SVQ) achieved significant improvements over the traditionally used GLA and over SR-D, which is said to give near optimal results [117]. From the figures we observe that, the SVQ-G algorithm (reduced complexity Gibbs distribution) performed better than the other SVQ algorithms; however, the performance of SVQ-N is very competitive. Note the progression of performances of the low complexity soft information measures: the performance improves from the inverse Euclidean distance soft-measure (SVQ-E and SVQ-Ea) to the single-triangle function soft measure (SVQ-T and SVQ-Ta), and from the latter to the multi-triangle function soft measure (SVQ-N and SVQ-Na). For clarity of observation, these results are shown separately in Figure 2.14 for the

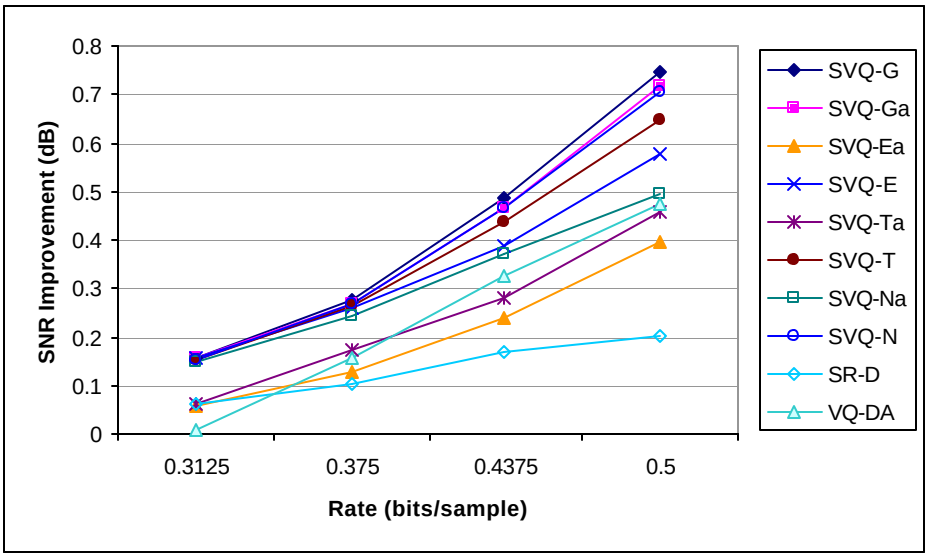


Figure 2.10: Improvements over GLA. Gaussian source. Vector dimension = 16 samples/vector.

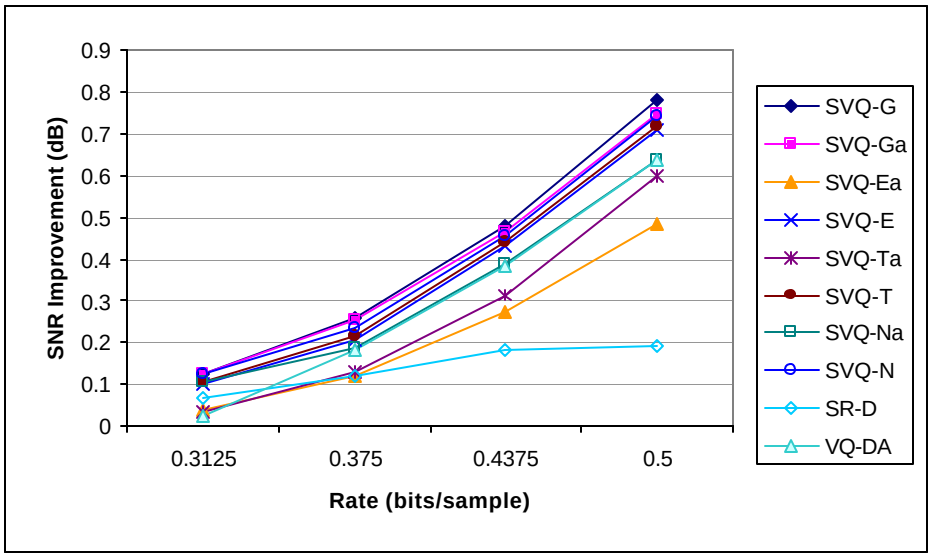


Figure 2.11: Improvements over GLA. Gauss-Markov source (a0=0.9). Vector dimension = 16 samples/vector.

Gauss-Markov source. This was an expected result since these algorithms were progressively designed to better approximate the optimal Gibbs distribution, hence the performance progressively approaches to the Gibbs performance. Note also the gain achieved by the stochastic relaxation (SR) in the SVQ algorithms (abbreviated with an extension "a", SVQ-*X*a) compared to non-stochastic cases (without extension "a", SVQ-*X*). The gain ranges from a high 0.2dB for SVQ-E (Figure 2.11) to a low 0.02 dB for SVQ-G algorithms. It should be noted that the better an algorithm performs *without* the SR, the lesser the additional gain achieved by the SR in the

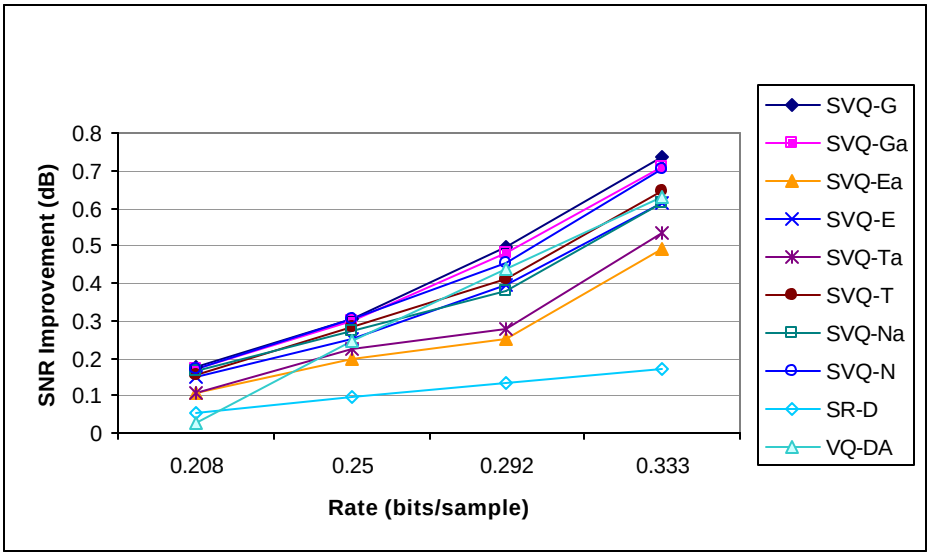


Figure 2.12: Improvements over GLA. Gaussian source. Vector dimension = 24 samples/vector.

SVQ algorithms. In other words, as an algorithm comes closer to the global optimum using the principles of soft information processing, it requires lesser and lesser help from the SR to attain an improved performance. In the limit, granting

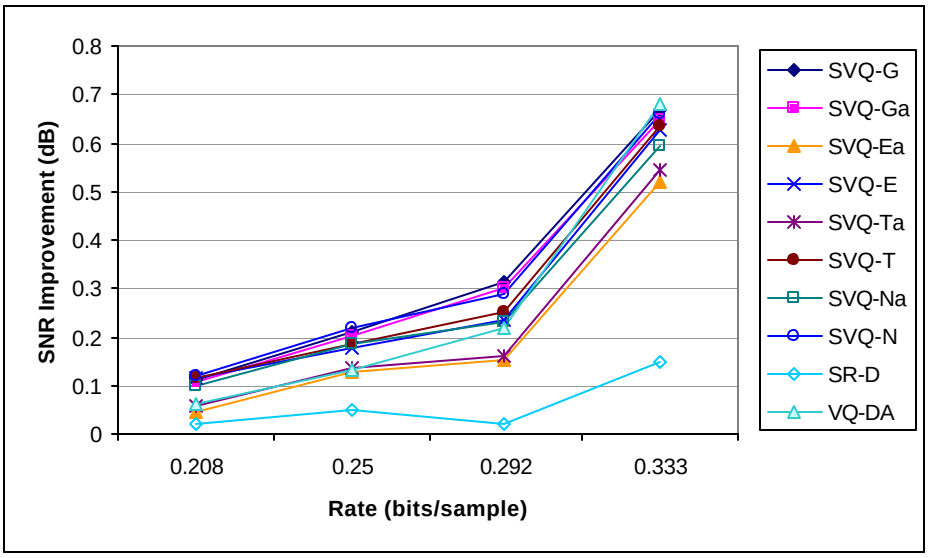


Figure 2.13: Improvements over GLA. Gauss-Markov source (a0=0.9). Vector dimension = 24 samples/vector.

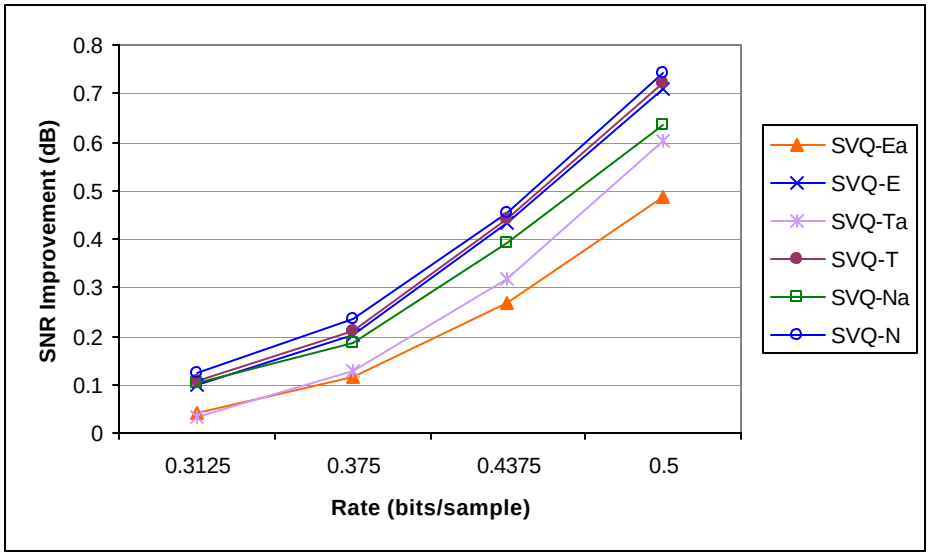


Figure 2.14: Improvements over GLA for low complexity information measures. Source is Gauss-Markov (a0=0.9). Vector dimension = 16 samples/vector.

enough computational resources for the full power of the soft information processing to be utilized, the global optimum can be reached without requiring any help from SR. But as the results demonstrate, for the reduced complexity DA approaches SR results in a performance improvement with negligible computational complexity.

The results for VQ-DA (standard DA) were obtained starting with all the sample vectors being equally associated with all the codevectors, which dictates an initial codebook where all the codevectors are at the center of mass of the training set. The simulations were conducted with a conservative annealing schedule which took over 120000 CPU seconds (about 24 hours) for the codebook of size $|C| = 256$ to converge. Recall that in VQ-DA the starting temperature is very high for the initial probability associations to be uniform, and the probability associations are computed to all codevectors for each sample vector, thus the algorithm executes very slowly especially for large codebooks. The figures show that, the performance of VQ-DA compared to reduced complexity DA algorithms is inferior in almost all cases considered (except for rate 0.333 in Figure 2.13). Moreover, the SVQ algorithms run much faster than VQ-DA, requiring 450 CPU seconds for $|C| = 256$ and 24 dimensional vectors. We are aware that, if enough computational resources are allocated, VQ-DA is expected to perform superbly as shown in [106]. However, the performance of the reduced complexity DA algorithms proved that for most practical applications the expected performance of VQ-DA do not justify its computational burden.

2.4.2 With Codebook Initialization

In addition to the Gauss-Markov sources used in the previous section we also considered sampled human speech and image sources to compare the GLA, SR-D and SVQ performances with PNN initialized codebooks. A segment of human speech was sampled at 8 kHz and partitioned into 2048 16-dimensional vectors, and we have designed five codebooks of sizes 16, 32, 64, 128 and 256. The image source was obtained by extracting 8192 16-dimensional vectors (corresponding to 4×4 blocks) from 512×512 monochrome training images from the USC image database with each pixel amplitude quantized to 8 bits. Four codebooks of sizes 32, 64, 128 and 256 were designed using this training set, and the performances of these codebooks were tested on "Lena" which was outside of the training set.

In Figure 2.15 we show the performances of the 4 codebooks on (uncorrelated) Gaussian source as improvement over the PNN initialized GLA. For clarity of presentation we have only included the SVQ-Ga performance from our proposed algorithms; the other SVQ algorithms behave comparatively the same with SVQ-Ga as in Figure 2.10. Note from the figure that the PNN initialization improves the GLA and SR-D algorithms, however the SVQ-Ga algorithm is not affected. This is a positive result for the SVQ algorithms for it shows that they can evade the local minimum dictated by the initial codebook, and hence are insensitive to the choice of the initial codebook. The PNN and its fast but sub-optimal version, fast-PNN require $O\left(K^3\right)$ and $O\left(K\log K\right)$ time, respectively, where K is the size of the training set [39]. The results presented in Figure 2.15 are obtained using the full search PNN

algorithm (with complexity $O\left(K^3\right)$) in order to get the best possible results with the GLA and the SR-D algorithms. The fast-PNN initialization would result in reduced performance; it is shown in [114] that the fast-PNN algorithm decreases the coding performance by 0.4 – 0.6dB for image sources compared to full search PNN. The

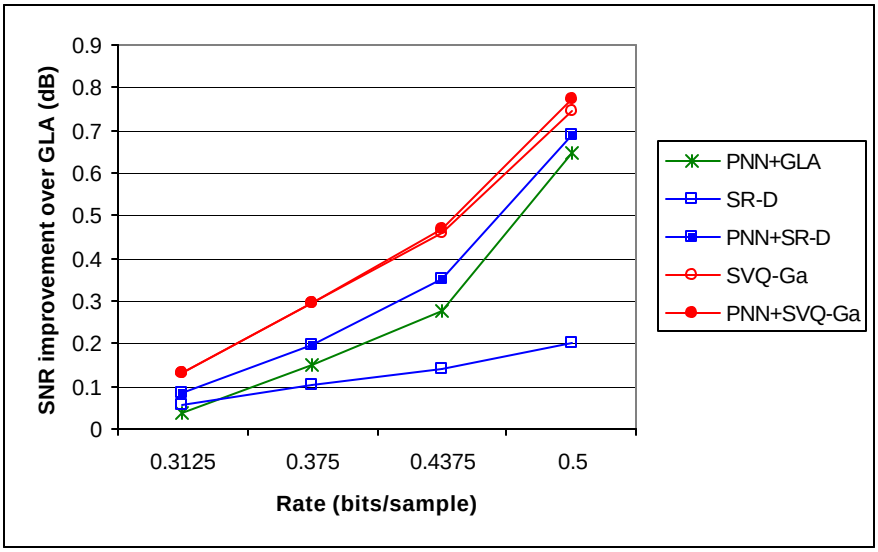


Figure 2.15: Shows the effect of PNN initialization as improvement over GLA. Source is Gaussian, vector dimension = 16 samples/vector.

SVQ algorithms outperformed both GLA and SR-D algorithms without the complexity of the initialization process, which gets computationally more impractical as the size of the training set increases. The running time for the generation of the PNN codebooks from a training set of 4096 16-dimensional vectors was 2374 CPU seconds, and the design of the size 256 codebooks for GLA, SR-D and SVQ-Ga algorithms on average were 44 CPU seconds, 366 CPU seconds and 1552 CPU seconds on the same machine, respectively. Therefore, with the PNN

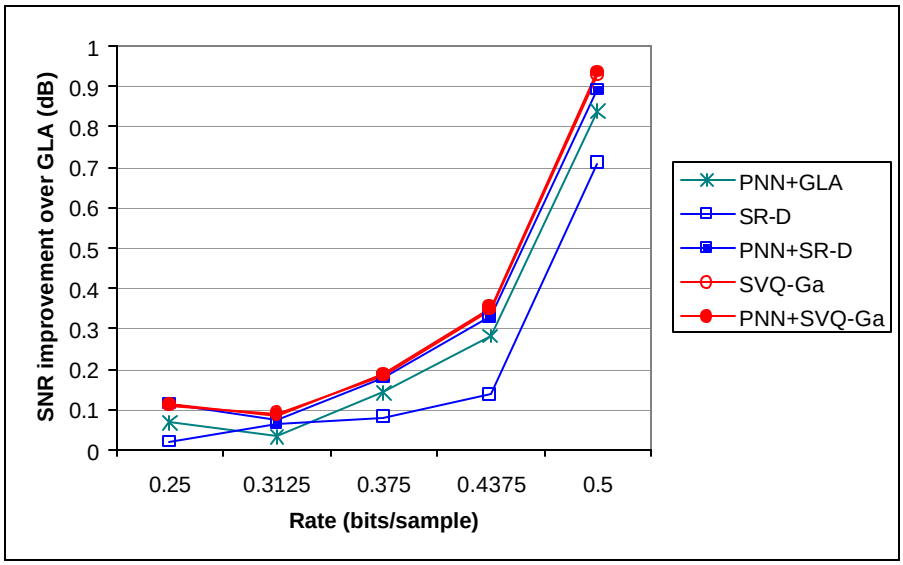


Figure 2.16: Improvements over GLA for speech source sampled at 8kH, vector dimensions = 16 samples/vector.

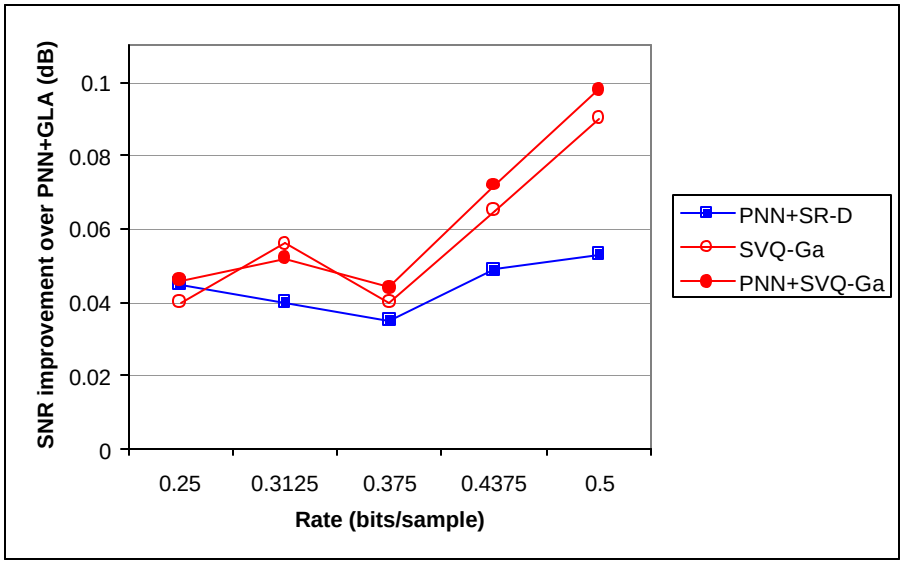


Figure 2.17: Improvements over PNN initialized GLA for speech source sampled at 8kH, vector dimensions = 16 samples/vector.

initialization the total running times for the GLA and the SR-D algorithms are higher than those for the SVQ-Ga algorithm. And since SVQ-Ga performs the same with and without the initialization, the SVQ-Ga algorithm outperforms GLA and SR-D with a lower running time.

The performance on the speech source using the three algorithms, GLA, SR-D and SVQ-Ga, with and without the codebook initialization is shown in Figures 2.16 and 2.17. In Figure 2.16 the performance improvement over GLA and in Figure 2.17 improvement over PNN initialized GLA are shown. Note that while the performance improvement of SVQ-Ga over GLA is large (0.95 dB at 0.5 bits/sample), compared with the PNN initialized GLA the improvement is rather modest. It is stated in [49] that for speech sources the higher the sampling rate, the less variable are the vector shapes for the same dimension and so the simpler the needed codebooks. The comparison made in [49] considers 8kHz (the sampling rate we used) as a high sampling rate. Hence, this means that, at high sampling rates, the expected performance increase over a good codebook with a better codebook is small for speech sources, explaining the small performance difference between PNN+GLA and SVQ-Ga. However, note also that the effect of the initialization is very small on the SVQ-Ga performance, whereas improvements of 0.85 dB and 0.2 dB are obtained at 0.5 bits/sample for GLA and SR-D, respectively, after initialization. Therefore, as in the Gaussian source, the SVQ-Ga renders the initialization unnecessary.

Finally, the results for the image source are shown in Figure 2.18 for the coding of the image source "Lena." As in the previous two source cases the SVQ-Ga performance is practically not sensitive to the initial codebook initialization. And it

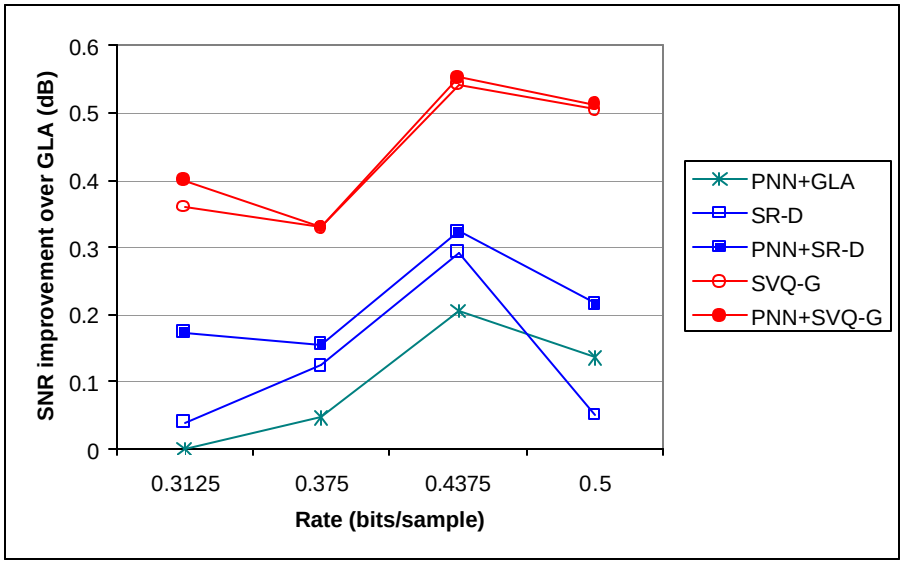


Figure 2.18: Improvement over GLA on image source "Lena." Vector dimensions = 16 pixels/vector (corresponding to 4x4 blocks).

outperformed the GLA and the SR-D algorithms by 0.3 – 0.4 dB and 0.2 – 0.3 dB, respectively, with both being initialized with PNN. Therefore, as in the Gaussian and the speech sources the SVQ-Ga outperformed the PNN+GLA and PNN+SR-D without the need of initialization.

2.5 Conclusion

In this Chapter we have designed reduced/low complexity methods for deterministic annealing (DA) for the vector quantizer design problem, which we

named *soft vector quantizer* (SVQ) design algorithms. The proposed low complexity soft measures are used as the soft association probabilities in the probabilistic framework of the DA to reduce the computational cost, as compared to the optimal Gibbs soft measure used in the standard DA. Although the simple soft measures significantly reduce the computational complexity of the system, this improvement comes at a price since these soft measures are not the optimal distributions. Hence, we have also derived the theoretical performance loss for using a simplified measure instead of the optimal measure, and used the result to derive optimal annealing schedules for the proposed simple soft-measures. We have demonstrated that using the derived optimal schedule for the low complexity soft measures increases the quality of the final codebook, as compared to using a geometric reduction schedule which is usually suggested in the annealing algorithms. We have also shown that the low complexity DA methods benefit from the stochastic relaxation techniques with decreasing benefits as the performance approaches the optimal.

We have demonstrated the effectiveness of our low/reduced complexity DA (SVQ) algorithms by designing codebooks for a variety of sources, namely Gauss-Markov, speech and image, at different rates. In each case, the proposed SVQ algorithms significantly improved the quality of the final codebooks, as compared to the traditionally used GLA and compared to the SR-D algorithm, where the latter is accepted as a benchmark reference by some researchers as a VQ design technique that performs near-optimally. We have also investigated the effect of codebook initialization on GLA, SR-D and SVQ algorithms and showed that, while GLA and

SR-D benefit from this initialization (PNN) at the expense of increased computational complexity, the SVQ algorithms are able to attain the same performance without the need for a special initialization. Hence, the SVQ algorithms are not sensitive to the choice of the initial codebook and outperform codebook initialized GLA and SR-D algorithms. Compared to the standard DA, the computational complexity of the SVQ algorithms is shown to be drastically reduced. Using the same annealing temperature the SVQ algorithms run by over a factor of 100 faster than the standard DA algorithm with negligible performance difference. We believe that the proposed algorithms, with their significantly higher performance over the widely used GLA and SR-D, and with their low computational complexity with negligible performance difference compared to the standard DA, have proved themselves to be important VQ design techniques.

Chapter 3

Deterministic Annealing for Frequency Assignment Problem

3.1 Introduction

The introduction of mobile communication systems has had a tremendous impact on our everyday lives, but at the same time the ever growing number of wireless/mobile users has made the optimal usage of the limited radio spectrum a highly important problem. The scarcity of commercially available spectrum requires that the frequencies be reused within a network, where the main limiting factor is then the level of interference. When two transmitters use the same frequency or frequencies close to each other their signals may interfere. The level of interference depends on many factors such as the power of the signals, distance between the transmitters, the direction in which signals are transmitted, environment, etc. On the one hand frequency reuse in a wireless network is a necessity due to spectrum scarcity, but on the other hand reuse may lead to quality loss in communication links.

Therefore, a high performance network can only be achieved by efficient allocation of the limited frequency spectrum to the transmitters. The assignment of frequencies in such a way that the interference is avoided, or if this is not possible, minimized, is called the Frequency Assignment Problem (FAP).

The radio spectrum (bandwidth) $[f_{\min}, f_{\max}]$ available to a wireless service provider is partitioned into a set of disjoint channels, all with the same bandwidth w. The channels are usually numbered by a sequence of integers, $\{0,1,\cdots, K-1\}$, where $K = (f_{\min} - f_{\max})/w$. On each available channel a transmitter and a receiver can communicate. For bi-directional traffic two channels are needed, one for each direction. In fact, when a wireless service provider purchases the spectrum $[f_{\min}, f_{\max}]$, it often gets a paired spectrum, $[g_{\min}, g_{\max}]$ of equal bandwidth that is well separated from the first one, $g_{\min} \gg f_{\max}$. The second spectrum is also partitioned into K disjoint channels, $\{s, s+1, \cdots, s+K-1\}$, and while the forward connection uses a channel from the set $\{0,1,\cdots, K-1\}$, the backward connection uses a channel that is shifted s channels up. Note that since $g_{\min} \gg f_{\max}$, then $s \gg K$ which prevents any interference between the forward and the backward channels. Consequently, the channel assignment problem considers only one directional channel set, e.g., the forward channels, $\{0,1,\cdots, K-1\}$. Technologies such as Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA) and Code Division Multiple Access (CDMA) make it possible to use each

channel for a limited number of callers simultaneously. In the FAP literature the allocation of channels is divided into three categories [72]: Fixed Channel Assignment (FCA), Dynamic Channel Assignment (DCA), and Hybrid Channel Assignment (HCA) schemes. In FCA, a channel is permanently allocated to each transceiver for its exclusive use. In this scheme it is not allowed to change the assignment on-line to satisfy mobile call demands for wireless connections. Instead, in DCA there is no fixed assignment between channels and transceivers; all channels are placed in a pool and they are assigned to calls as the need arises provided that certain interference constraints are satisfied. Finally, in HCA schemes a combination of FCA and DCA is implemented to obtain a better overall performance in the network. In these schemes a number of channels are assigned beforehand and the rest can be used for on-line assignment upon request. It can be proved that DCA schemes perform better than FCA schemes under light and non-uniform call traffic, and that under uniform and heavy traffic FCA schemes outperform the DCA schemes [65]. Note also that FCA performance provides a bound on the DCA performance, because FCA is designed to provide the optimal service when all the network resources are being used. A survey of FCA, DCA, and HCA schemes can be found in [68].

In this thesis we will be focusing on FCA because it is a widely used channel allocation scheme, which also provides bounds for the DCA schemes. Thus, our goal will be to find the optimal fixed assignment of channels to transceivers. The FCA scheme can be designed based on four different criteria, leading to four

problems [72]: the *minimum order* frequency assignment problem (MO-FAP), the *minimum span* frequency assignment problem (MS-FAP), the *minimum blocking* frequency assignment problem (MB-FAP), and the *minimum interference* frequency assignment problem (MI-FAP).

In MO-FAP we have to assign channels in such a way that no unacceptable interference occurs, and the total number of distinct channels that are used is minimized. The MO-FAP was the first channel assignment problem discussed in the literature. Metzger [90] was the first to recognize that it is a direct equivalent of the *graph-coloring problem* (a problem that belongs to the class of NP-complete problems), and Hale [56] was the first to formalize the frequency assignment problem as a graph-coloring problem. In MS-FAP, the task is to assign channels in such a way that no unacceptable interference occurs, and the difference between the maximum and minimum used frequency, the span, is minimized. In MB-FAP the goal is to assign channels such that no unacceptable interference occurs as in the previous cases, and the overall blocking probability of the network is minimized, which requires the traffic demand of the nodes to be taken into account. And finally, MI-FAP is the minimization of the total sum of interference levels in the network. That is, we have to assign the channels from a limited number of available channels in such a way that the total sum of the interference in the network is minimized. Note that MO-FAP is not the only model that is related to the graph-coloring problem. All FAP models, in one way or another are generalizations of the coloring of nodes in an undirected graph [72]. Hence, all FAP models belong to the class of

NP-complete problems, which means that there does not exist an algorithm that solves the problem in time polynomial in the length of the input, unless $P = NP$ [45].

In MO-FAP and MS-FAP the objectives are, respectively, to minimize the number of used channels and to minimize the span of used channels. However, in practice the wireless communication service providers license a fixed frequency bandwidth for long-term periods without the possibility of extending or reducing the bandwidth. Therefore, minimizing the number of used frequencies or minimizing the span of the used frequencies do not satisfy the implementation of today's frequency plans (minimization of the order and the span were popular objectives in 1970s and 1980s when frequencies were bought per unit at high prices [72]). Nowadays, the more realistic problem facing the operators (wireless communication service providers) is finding an assignment of the available frequency bands to various stations such that the incurred interference does not exceed a certain threshold, in order to guarantee high quality communication links to the mobile users. Therefore, the problem that we chose to address is the minimization of the total interference in a given network with fixed resources, i.e., the MI-FAP.

In this chapter we will model and apply the *deterministic annealing* (DA) method as a novel approach to solve the *minimum interference* frequency assignment problem (MI-FAP). The DA approach has been used to obtain near optimal solutions for various difficult combinatorial optimization problems by using soft information processing with exact and simplified reliability measures as introduced

in Chapter 2 [104, 106, 103, 32]. We will present experimental evidence indicating that significant performance gains are achieved by the DA method over the simulated annealing (SA) method. Note that an algorithm based on SA is reported to provide the best known performances for realistic frequency planning scenarios [37]. Besides performing better than the SA, experimental results will also show that the DA method converges faster than SA.

The rest of the chapter is organized as follows: in section 3.2 the relation between FAP and graph coloring is briefly explained, we then describe the shortcomings of the graph theoretic approaches that formulate FAP as a binary constraint satisfaction problem. The deterministic annealing algorithm for FAP is formulated and the algorithms are described in section 3.3. In the experimental results section we will be comparing the performance of our DA algorithms against the simulated annealing (SA) algorithm for FAP [36, 99, 58], using the test problems generated by the vertex saturation (VS) method (a test problem generator) [118]. In section 3.4 the experimental set-up and the experimental results comparing the performances of SA and DA to the optimal solutions obtained by the VS are presented. We have also tested our proposed DA algorithm on a realistic frequency planning scenario obtained from the COST 259 project [37], and compared its performance with other methods that are reported to provide the best performance on a collection of real life scenarios. Finally, section 3.5 concludes the chapter.

3.2 Graph Theoretic Approach

3.2.1 Graph Coloring and FAP

The coloring of a graph $G=(V,E)$, where V is the set of nodes in the graph and E is the set of edges (connected pairs of nodes) in the graph, is defined as the assignment of colors to the nodes $v_i \in V$ such that no two nodes connected by an edge will have the same color. Colors are defined to be a set of non-negative integers. A coloring that assigns c colors to graph G is termed c-coloring. The chromatic number of a graph G is the minimum number $c(G)$ for which $c(G)$-coloring exists for G. Hence, a graph is c-colorable, if $c(G) \le c$. Also, if we denote $w(G)$ as the largest clique size in G, where a clique is a complete subgraph of the graph G which means that all the nodes in the subgraph are connected with all the other nodes in the subgraph, then from the definition of the chromatic number we can deduce that $c(G) \ge w(G)$. Using the above description, the equivalence of co-channel frequency assignment problem and the graph-coloring problem is apparent. For the co-channel FAP, define:

- the nodes $v_i \in V$ of graph G to be the set of all transmitters in the network.
- an edge (v_i, v_j) exists in G if the transmitters v_i and v_j cannot use the same channel (i.e., v_i and v_j are co-channel constrained) $\forall v_i, v_j \in V$.

Let $y(v)$ to denote the channel assigned to v by the assignment rule y, $C=\{0,1,\cdots,K-1\}$ to be the channel set, $y(v)\in C$, then the co-channel FAP (the minimum span model) is equivalent to simple graph coloring:

$$\begin{aligned}&\text{instance } G=(V,E)\\&\text{find } y:\ V\to\mathbb{Z}_+\\&\quad\text{such that, }\ (v_i,v_j)\in E\ \Leftrightarrow\ y(v_i)\neq y(v_j)\\&\quad\text{and }\ |C|\text{ is minimized.}\end{aligned}\tag{3.1}$$

We can see that in the above formulation the constraints place a restriction on the assignment of the channels to pairs of transmitters/nodes, hence the edge set E in G define binary constraints between the pairs of nodes. Similarly, co-channel and first adjacent-channel FAP would be (where the edge set E is split into two, E_0 and E_1; nodes connected by the edges in E_0 are co-channel constrained, and the edges in E_1 are adjacent-channel constrained):

$$\begin{aligned}&\text{instance } G=(V,\,E_0,E_1)\\&\text{find } y:\ V\to\mathbb{Z}_+\\&\quad\text{such that, }\ (v_i,v_j)\in E_0\ \Leftrightarrow\ y(v_i)\neq y(v_j),\\&\qquad\qquad(v_i,v_j)\in E_1\ \Leftrightarrow\ |y(v_i)-y(v_j)|>1\\&\quad\text{and, }\ |C|\text{ is minimized.}\end{aligned}\tag{3.2}$$

In the same manner, higher order adjacent channel FAPs that pertain to minimizing the number of channels used (MS-FAP) can be defined:

$$\begin{aligned}&\text{instance } G=(V,E_0,E_1,\cdots,E_l)\\&\text{find } y:\ V\to\mathbb{Z}_+\\&\quad\text{such that, }\ (v_i,v_j)\in E_k\ \Leftrightarrow\ |y(v_i)-y(v_j)|>k\qquad\forall k\in\{0,1,\cdots,l\}\\&\quad\text{and, }\ |C|\text{ is minimized.}\end{aligned}\tag{3.3}$$

Note that the above problems are MS-FAP, hence the goal is to minimize the number of channels used. In the above formulations, (3.1), (3.2) and (3.3), a single channel is assigned to each transmitter and solutions that do not satisfy the constraints are infeasible solutions. Using the graph theoretic approach the interference minimization model of FAP, MI-FAP can be formulated as follows,

$$\begin{aligned}
&\text{instance } G=(V, E_0, E_1, \cdots, E_l) \\
&\text{find } y: \quad V \to \mathbb{Z}_+ \\
&\quad \text{such that,} \\
&\quad \sum_{k=0}^{l} \sum_{(v_i,v_j)\in E_k} \sum_{y(v_i)\in C} \sum_{y(v_j)\in C} c_{|y(v_i)-y(v_j)|}(v_i,v_j)\cdot d\left(\left|y(v_i)-y(v_j)\right| \le k\right) \\
&\quad \text{is minimized.}
\end{aligned} \tag{3.4}$$

In (3.4) $d(x)$ is the Kronecker delta function which is one if the condition x is true and zero otherwise, and $c_{|y(v)-y(w)|}(v,w)$ is defined as the penalty incurred for assigning channels $y(v)$ and $y(w)$ to nodes v and w, respectively. Note that $c_{|y(v)-y(w)|}(v,w)$ can either be a self assigned penalty or it can be the actual interference caused on the network when node v is assigned channel $y(v)$ and node w is assigned channel $y(w)$. Throughout this chapter we assume them to be actual interference values. In (3.4) a single channel is assigned to each transmitter and a feasible solution is the one that satisfies all channel separation requirements between the pairs of nodes of the graph, as in (3.1) - (3.3), which gives zero total interference. However, if no zero total interference solution can be found, then the goal is to minimize the total interference.

We have already mentioned that all FAP models belong to the class of NP-complete problems. We briefly give the computational complexity proof for the MI-FAP variant of the FAP models as follows [13, 37]:

> Given a set of channels $C=\{1,2,\cdots,K\}$ and a graph $G=(V, E_0, E_1, \cdots, E_{K-1})$, the *decision problem* for MI-FAP decides whether there is a node-channel assignment set of total cost no more than $q \in \mathbb{Q}_+$. If we are given K, the graph $G=(V, E_0, E_1, \cdots, E_{K-1})$, cost q and a certificate assignment set, we can verify in polynomial time if the total cost of the assignments is q or not; hence MI-FAP $\in NP$. It is also shown in [37] that the NP-hard problem *graph c-colorability* can be reduced to MI-FAP in polynomial time making MI-FAP also NP-hard. Hence, for every $q \in \mathbb{Q}_+$, MI-FAP $\in NP$-complete.

Moreover, not only that solving MI-FAP is NP-hard, but finding solutions that are guaranteed to be close to optimal (i.e., with a guaranteed upper bound on the total cost) is also NP-hard [37].

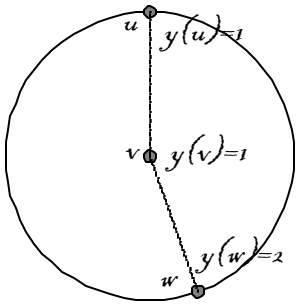
3.2.2 Binary Constraints

The need to keep interference below an acceptable level or minimize it requires that nearby transceivers use channels which are widely separated in the available spectrum. This means that the closer the transceivers are to each other, the more separated their assigned channels should be. The most widely used way of modeling

the physical-distance – channel-separation requirements is by using binary constraints on the pairs of transceivers (as in the formulations (3.1) to (3.4)) by attaching a minimum spatial separation for each possible channel separation between them. So, for K possible channels, $C = \{0,1,\cdots,K-1\}$,

$$d(v_i, v_j) < r_k \quad \Rightarrow \quad |y(v_i) - y(v_j)| > k, \quad \forall k = 0,1,\cdots,K-1 \tag{3.5}$$

where $d(v_i, v_j)$ is the Euclidean separation between any two nodes v_i and v_j, and $\{r_k\}_{k=0}^{K-1}$ are the minimum separation distances depending on the channel separation k, with $r_0 > r_1 > \cdots > r_{K-1}$. Through this model, the FAP is cast using an undirected graph, called an *interference graph* or a *constraint graph*, in which the edges of the graph represent the binary constraints. With this formulation it is closely associated with the well-studied graph coloring problem as mentioned earlier, and hence, traditionally, graph theoretic approaches have been used to solve FAP problems [56, 47, 90, 8, 116, 14, 15, 87, 115, 13, 86]. However, a number of recent studies have raised concerns about the adequacy of the binary constrains, stating that in any network the signal-to-interference ratio (SIR) at any point depends on the *cumulative* effect of signals received from *all* the transceivers, therefore, the use of binary constraints to model FAP is too restrictive [35, 96, 6, 34, 63]. In [35] it is argued that FAP should be formulated as a cost function optimization, and in [96, 6, 34, 63] the argument is that better assignments can be obtained with higher order constraints. By higher order constraints what is meant is that, instead of considering two transceivers at a time in assigning channels to them (binary constraint) as in (3.5),



$|y(v)-y(u)|=0$, v and u are co-channel nodes

$|y(v)-y(w)|=1$, v and w are adjacent-channel nodes

Figure 3.1: Two nodes, u and w at the same distance from node v. Node u is co-channel with v, and node w is adj-channel with node v.

more than two (the higher the better) transceivers are considered simultaneously. Although the merit of this argument is not extensively tested (only very small structured problems are considered in [96, 6, 34, 63]), there are some other drawbacks in using graph theoretic approaches. Next we illustrate these with a simple example.

By intuition we know that between two transceivers /nodes, the co-channel interference is larger than the adjacent channel interference. The nodes u and w in Figure 3.1 are at same distance from v, where $y(v)=f_1$, $y(u)=f_1$, $y(w)=f_2$. Using the penalty function in (3.4), with $c_{|y(v_i)-y(v_j)|}(v_i,v_j)$ being the penalty for interference between the two nodes v_i and v_j with channel assignments $y(v_i)$ and $y(v_j)$, respectively. The co-channel $(y(v_i)=y(v_j))$ and adjacent channel $(|y(v_i)-y(v_j)|=1)$ interferences are $c_0(v_i,v_j)$ and $c_1(v_i,v_j)$, respectively. Then, in the situation depicted in Figure 3.1 the interference penalties would be

$c_0(v,u) > c_1(v,w)$. Higher order channel interferences can also be defined, $c_0(v_i,v_j) > c_1(v_i,v_j) > c_2(v_i,v_j) > \cdots > c_k(v_i,v_j) > \cdots$. Note that these are still binary relationships between pairs of nodes, where a k^{th} order channel interference $c_k(v_i,v_j)$ means, the interference between nodes v_i and v_j that are k channels apart, $|y(v_i) - y(v_j)| = k$. In accordance with the model in (3.5) a minimum channel separation between potentially interfering nodes is imposed. In this model, it is perfectly legal to assign a channel $f_i \in C$ to a node w at a distance $r_0 + e$ from another node v, where $y(v) = f_i$, and where e is a very small positive number, $0 < e \ll 1$, because they are separated more than the minimum distance r_0 for co-channel assignment. However, by the same token, it would be illegal to assign f_i to

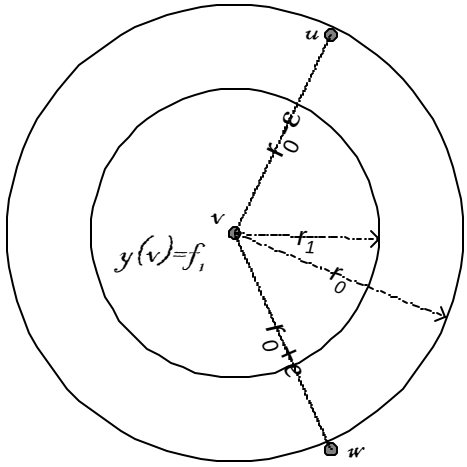
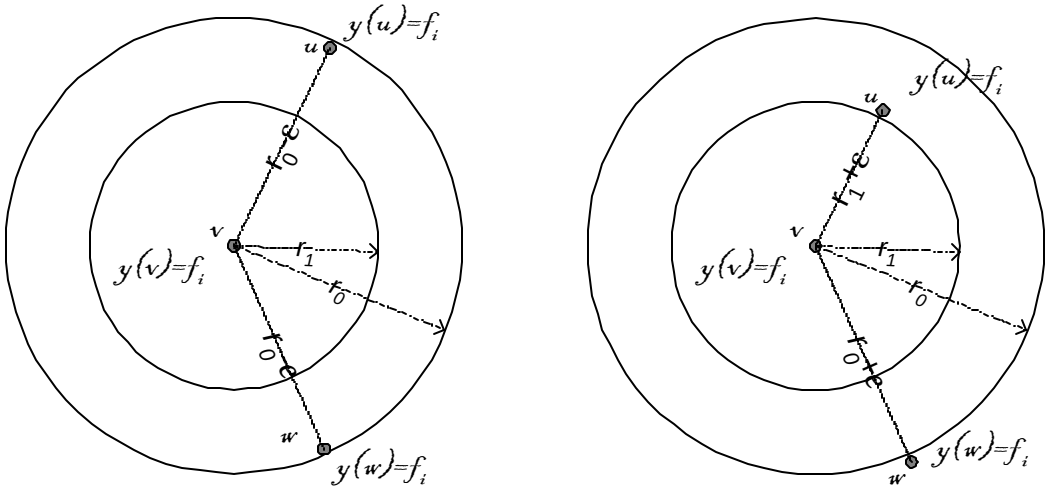


Figure 3.2: Node *w* satisfies co-channel binary separation with node *v*, but node *u* does not.

a node u at distance $r_0 - e$ from v since they are not separated by more than the minimum co-channel distance, r_0 as shown in Figure 3.2. No matter how small e is this would constitute an infeasible solution. But, we know that for very small e the interferences caused by the two assignments will be practically the same. Nevertheless, the model described above would make a strict decision and consider the assignment, $y(w) = f_i$, admissible, but would never choose the assignment, $y(u) = f_i$.



$d(v,u) \not> r_0, \quad |y(v) - y(u)| = 0$
$d(v,w) \not> r_0, \quad |y(v) - y(w)| = 0$
2 assignment violations

(a)

$d(v,u) \not> r_0, \quad |y(v) - y(u)| = 0$
$d(v,w) > r_0, \quad |y(v) - y(w)| = 0$
1 assignment violation

(b)

Figure 3.3: In (a) both nodes *w* and *u* do not satisfy co-channel binary separation with node *v*. In (b) node *w* satisfies the co-channel binary separation with node *v*, but node *u* does not.

Therefore, a method that treats the problem based on hard (binary) decisions for each pair of nodes, i.e., such that the set of admissible assignments is pre-set given the distance between nodes, may not be well suited to address MI-FAP. We require methods that allow gradual differences in interferences caused by all the nodes to be taken into account. In other words, the received signal quality depends on the *cumulative* effect of the signals received from all the nodes. Furthermore, if in a given problem a channel assignment fulfilling all the constraints does not exist or cannot be found, the described model cannot judiciously distinguish between two infeasible solutions. The best it can do is to count the number of constraint violations and choose the solution with the least number of violations. But this does not guarantee the best possible solution in terms of minimizing the total interference, as can be seen in the situation depicted in Figure 3.3 and explained in the following example.

Example 1:

To be able to better asses the situation, consider the *interference power* between two nodes v and u, operating at channels $y(v)$ and $y(u)$, respectively. We will use a simple model from [78] (later in the Experimental Results section we will use a more comprehensive model from [114]):

$$P\left(v, y(v); u, y(u)\right) = \frac{p_t \cdot h\left(\left|y(v) - y(u)\right|\right)}{d(v,u)^g}, \tag{3.6}$$

where p_t = transmission power at the nodes.

$h(\cdot)$ = channel difference response, positive and decreasing,

$h(0)=1, \quad h(0)>h(1)>\cdots>h(k)>\cdots$

$h(a)=\mathrm{e}^{-a} \quad \forall a \geq 0$, is a valid function.

$d(v,u)$ = Euclidean distance between nodes v and u.

g = exponent describing signal attenuation, usually between 2 and 6.

Note that except the terms containing v, u and the channel separation $|y(v)-y(u)|$, the rest of the terms in equation (3.6) are constants. Therefore, we will use the previous notation we had introduced for interference, i.e., $c_{|y(v)-y(u)|}(v,u)$, in equation (3.6). Turning back to Figure 3.3(a) (2 violation case), the interference powers at node v from nodes u and w:

$$c_0(v,u)=\frac{p_t \cdot h(0)}{d(v,u)^g}=\frac{p_t}{(r_0-e)^g} \tag{3.7}$$

$$c_0(v,w)=\frac{p_t \cdot h(0)}{d(v,w)^g}=\frac{p_t}{(r_0-e)^g} \tag{3.8}$$

From Figure 3.3(b) (1 violation case),

$$c_0'(v,u)=\frac{p_t \cdot h(0)}{d'(v,u)^g}=\frac{p_t}{(r_1+e)^g} \tag{3.9}$$

$$c_0'(v,w)=\frac{p_t \cdot h(0)}{d'(v,w)^g}=\frac{p_t}{(r_0+e)^g} \tag{3.10}$$

Defining $z = c_0(v,u)+c_0(v,w)$ and $z' = c_0'(v,u)+c_0'(v,w)$, after simplifications we have,

$$\frac{z'}{z} = \frac{(r_0 - e)^g}{2(r_1 + e)^g} + \frac{(r_0 - e)^g}{2(r_0 + e)^g}$$

$$\lim_{e \to 0} \frac{z'}{z} = \frac{r_0^g}{2r_1^g} + \frac{r_0^g}{2r_0^g} = \frac{r_0^g + r_1^g}{2r_1^g} \qquad (3.11)$$

and since, $r_1 < r_0 \quad \Rightarrow \quad \frac{z'}{z} > 1$

end Example 1.

Hence, the one-violation case (Figure 3.3(b)) results in higher interference than the two-violation case (Figure 3.3a), which shows that the solution with the least number of violations does not guarantee the least possible total interference. Therefore, formulating MI-FAP in graph theoretic terms as a constraint satisfaction problem (CSP) presents two drawbacks:

- the use of hard (binary) decisions, whether to allow or not to allow two nodes to use the same channel (zero channel separation) or higher order separated channels, does not take gradual interference differences into account, and
- when a feasible solution does not exist, it is difficult within a CSP formulation to pick the best infeasible solution.

In order to address these drawbacks, MI-FAP should be formulated as a *cost function optimization problem* rather than a CSP: given a list of nodes $v_i \in V$, $i = 1,2,\cdots,n$, a set of channels $C = \{f_0, f_1, \cdots, f_{K-1}\}$ as an interval of non-negative integers, the node-channel assignment interference cost function F is:

$$F = \sum_{v \in V} \sum_{f \in C} \sum_{w \in V} \sum_{f' \in C} c_{|f-f'|}(v,w) \cdot s_{wf'} \cdot s_{vf} \qquad (3.12)$$

where $s_{vf} = \begin{cases} 1, \text{ if channel } f \text{ is used at node } v, \\ 0, \text{ otherwise.} \end{cases}$

$c_{|f-f'|}(v, w) =$ interference incurred by assigning f to v and f' to w.

The aim is to find the optimal node-channel assignments, s^*_{vf}, such that the total interference cost is minimized. Note the difference in the formulations of MI-FAP in (3.4) and (3.12). The formulation in (3.4) aims to minimize the total interference by satisfying the binary constraints. So, if the assigned channels to a pair of nodes do not violate the required channel separation between those two nodes, then the cost incurred from this assignment is zero. This neglects individual interferences below a certain level, although the cumulative effect of many sub-threshold interferences from multiple nodes may result in a non-negligible overall interference on a certain node. Note that the optimal solution cost in (3.4) is zero, which is achieved when all the constraints are satisfied. This shows that, if there are more than one optimal assignment sets (zero cost solutions), they cannot be distinguished even though the total interferences in these optimal solutions may vary greatly. Instead, in (3.12), each node-channel assignment is evaluated by taking into account the cumulative effect of the interferences from all the surrounding nodes. This formulation is more suitable for the objective of MI-FAP.

3.3 Deterministic Annealing Solution for FAP

3.3.1 Problem Formulation and Algorithm

The *deterministic annealing* (DA) approach puts the channel assignment problem in a probabilistic framework, and optimizes the probabilistic objective function in each iteration. The node-channel assignments in DA are not one-to-one, they are one-to-many; each node is assigned to *all* the available channels with a given probability $p(f_i|v)$, where $\sum_i p(f_i|v)=1,\ \forall v \in V$. As stated in the previous chapters this is called a *soft association*, and the probabilities are called *soft information* (or *soft assignments*) since they give the reliability measures of assigning the channels to a node. Using the node-channel soft associations rather than the hard (one-to-one) assignments as in (3.12), the cost function to be minimized becomes an expected (probabilistic) cost function:

$$E\{F\}=\sum_{v\in V}\sum_{f\in C}\sum_{w\in V}\sum_{f'\in C} c_{|f-f'|}(v,w)\,p(f'|w)\,p(f|v) \tag{3.13}$$

Defining,

$$j(v,f)=\sum_{w\in V}\sum_{f'\in C} c_{|f-f'|}(v,w)\,p(f'|w), \tag{3.14}$$

we can express (3.13) more compactly,

$$E\{F\}=\sum_{v\in V}\sum_{f\in C} j(v,f)\,p(f|v). \tag{3.15}$$

$j(v,f)$ is the total expected interference incurred on node v from all soft assignments of the nodes $w\in V\setminus v$ and the channels $f'\in C$, when channel f is

assigned to node v, Figure 3.4. Hence, the total expected interference on the (v, f) pair averaged over all $f \in C$ and $v \in V$ gives the total expected interference cost in (3.15).

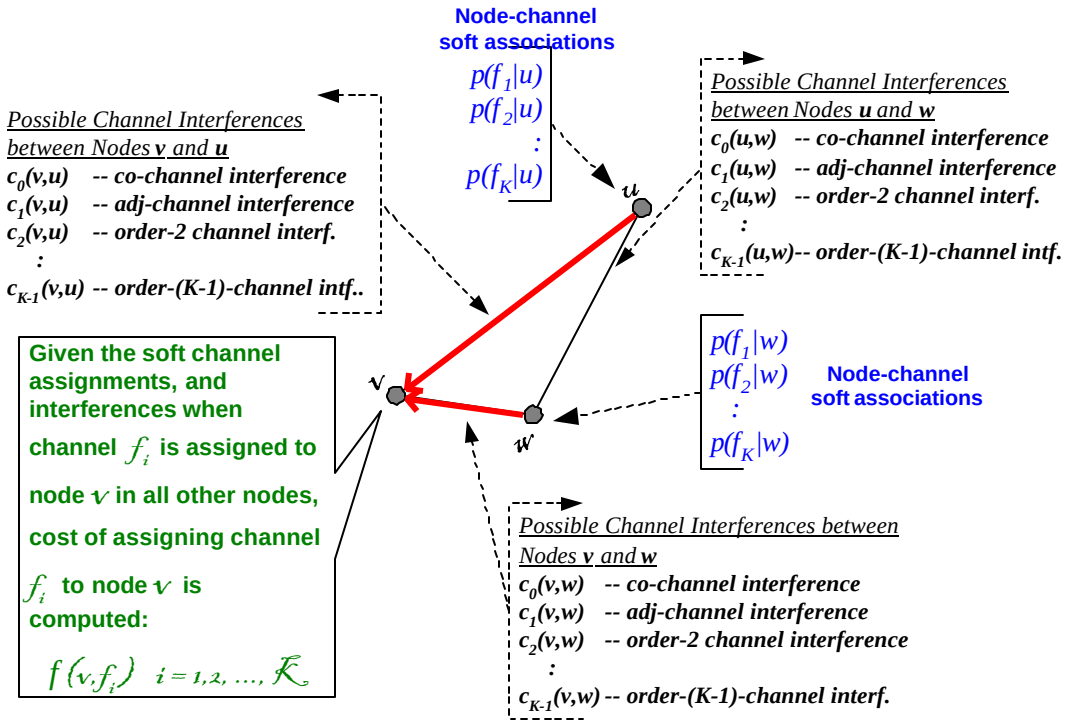


Figure 3.4: Computation of the cost of assigning channel f_i to node v.

The most powerful aspect of the DA method is that the optimization starts with all possible node-channel assignments being equally likely, i.e., $p(f_i|v) = 1/K \quad \forall i$, where K is the number of channels, $K = |C|$. Hence, at the beginning the entropy of the system is maximum, $H = \log K$, and the system has the highest uncertainty in its soft assignments. This shows that the DA procedure is completely unbiased and does not favor any of the channels for any node. The above also implies that it does

not require an initial starting configuration (a hard node-channel assignment set). The latter point is always a big concern in any iterative optimization algorithm that requires a starting configuration since the quality of the final solution is highly sensitive to this choice; there is no such concern in the DA optimization. The iterations start with this unbiased, soft state, $p(f_i|v) = 1/K \quad \forall f_i \in C, \forall v \in V$, and through iterations the softness (and thus the entropy) is gradually reduced until the hard channel assignment for each node is reached. Our goal is for that final hard state the assignments to be optimal, in the sense of minimizing the cost function. If we define the optimal node-channel assignments as $y^*(v), \forall v \in V$, where $y^*(v) \in C$, then,

$$p^*(f_i|v) = \begin{cases} 1, & \text{if} \quad f_i = y^*(v) \\ 0, & \text{otherwise} \end{cases} \tag{3.16}$$

and,

$$\min F = \sum_{v \in V} \sum_{f \in C} \sum_{w \in V} \sum_{f' \in C} c_{|f-f'|}(v,w) \;\; p^*(f'|w) \;\; p^*(f|v). \tag{3.17}$$

We see that in this probabilistic framework we need a distribution (soft information measure) that is defined over the set of all channel assignments, $\sum_i p(f_i|v) = 1 \;\; \forall v$, and also that assigns higher probability to assignments of lower cost. This distribution should also be parameterized by a *softness control* factor, such that, as the softness is decreased the distribution should become more discriminating by concentrating most of the probability in a smaller subset of low cost assignments, and in the limit should reach hard node-channel assignments as shown in (3.16). The expected interference $j(v, f)$ defined in (3.14) is the cost of

assigning f to v. We use two mechanisms to obtain the distribution model, $p(f|v)$, as explained in Chapter 2 (see also [32]):

1. *Gibbs membership function*,
2. *Triangular membership function*.

1. Gibbs membership function:

For FAP, of all possible probability distributions that yield a given expected total interference cost, (3.15) and satisfy $\sum_i p(f_i|v)=1$, the Gibbs distribution is optimal in the sense that it maximizes the entropy,

$$p(f_i|v)=\frac{e^{-b\cdot j(v,f_i)}}{\sum_{j=0}^{K-1} e^{-b\cdot j(v,f_j)}}. \tag{3.18}$$

Recall that b in (3.18) is the softness control factor, it determines the amount of discrimination among the possible channel assignments.

2. Triangular membership function:

The soft assignment values, (3.19) are obtained using the heights corresponding to the costs, where the heights are obtained from the membership triangle as explained in Chapter 2,

$$p(f_i|v) = \frac{R-j(v,f_i)}{KR-\sum_{j=0}^{K-1} j(v,f_j)}. \tag{3.19}$$

Recall that the spread, R of the triangle is the softness control factor of the soft assignments.

It should be noted that the DA formulation addresses the previously stated graph theoretic drawbacks for MI-FAP, namely,

- DA does not formulate the problem as a binary constraint satisfaction problem; it aims to find the global minimum which is more suitable to the objective of MI-FAP.
- In determining the cost of the node-channel assignments, DA takes into account interferences from all nodes with all possible assignments, weighted with assignment probabilities. Therefore, small differences in interferences which can have considerable effect when summed up, are not ignored.
- When there does not exist a feasible solution, DA is able to judiciously pick the best infeasible solution by aiming to find the global minimum.

The DA algorithm is shown in Figure 3.5 for the triangular distribution case. At iteration $t = 0$, the initial soft information values are uniformly distributed, $p^{(0)}(f_i|v) = 1/K \quad \forall i,$ and for each node $v \in V$ (the superscript in parenthesis is the iteration number), the initial spread $R^{(0)}$ in the triangular distribution, which ideally should ideally be infinite is chosen to be large enough so that the deviation from $1/K$ is less than 10^{-6}, i.e., $\left| p^{(0)}(f|v) - 1/K \right| < 10^{-6} \quad \forall f$ and $\forall v$. We use the same starting criterion in the Gibbs distribution in order to choose a non-zero, but small enough, $b^{(0)}$ (recall that in the triangular distribution softness decreases with decreasing R, and in the Gibbs distribution it decreases with increasing b). The iterations are

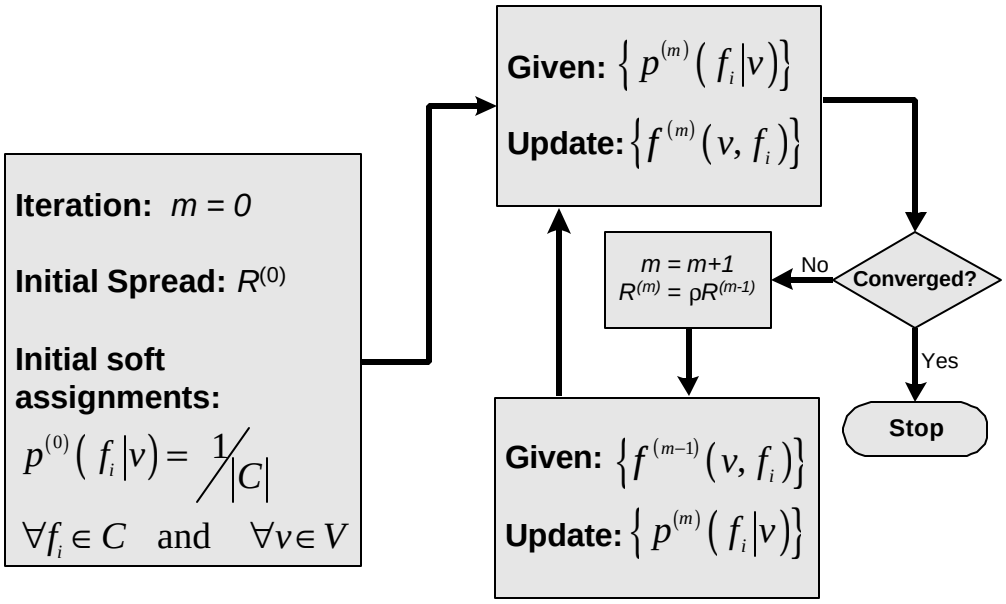


Figure 3.5: The iterative procedure showing updating of the soft assignments and the costs. The iterations are repeated until convergence. Convergence is reached when all the soft associations become hard.

repeated until convergence, where in each iteration t, the spread is decreased by a factor of r_T in the case of the triangular distribution, $R^{(t)} = r_T \cdot R^{(t-1)}$, $0 < r_T < 1$, and in the case of the Gibbs distribution b is increased by a factor of r_G, $b^{(t)} = r_G \cdot b^{(t-1)}$, $r_G > 1$. The algorithm converges when all the soft associations become hard as in (3.16). The values of r_T and r_G control the convergence rate in the triangular and the Gibbs cases, respectively. Clearly, the further away they are from 1 (r_T from lower and r_G from upper), the faster the convergence is, but with accompanying decrease in the quality of the final solution. On the other hand, if they are too close to 1, convergence takes too long without noticeable improvement.

3.3.2 Channel Blocking Algorithm

Since the co-channel interference is the highest cause of interference on any node, we may also wish to satisfy a certain separation for co-channel nodes together with minimizing the total interference in the network. In the DA algorithm when a node v starts to converge towards a channel $\bar{f}$ (by 'starting to converge' we mean $p(\bar{f}|v) > 0.5$ because most of the probability mass is on this assignment), $y(v) = \bar{f}$, then it is safe to say that with high probability the assignment of node v will fully

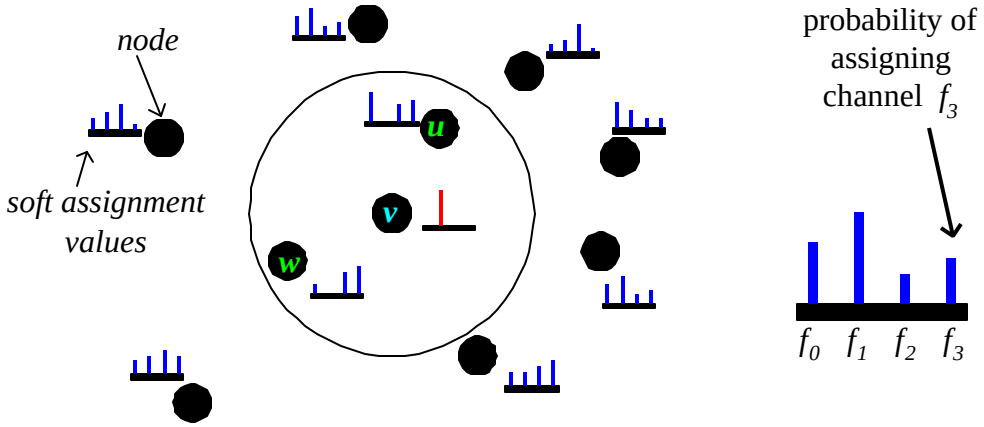


Figure 3.6: Node v has converged to channel f_1. For all the nodes within the neighborhood of node v, nodes u and w, channel f_1 is blocked by setting the assignment probabilities of nodes u and w to channel f_1 to zero.

converge to channel $\bar{f}$. Therefore, we propose as approximation that, when a node starts to converge to a channel $\bar{f}$ we will assume that it has converged by setting $p(\bar{f}|v) = 1.0$ and $p(\tilde{f}|v) = 0.0 \quad \forall \tilde{f} \in C \setminus \bar{f}$, and block the channel $\bar{f}$ for all the nodes w within the neighborhood of node v by setting their assignment probabilities

for channel $\bar{f}$ to zero, $p(\bar{f}|w) = 0.0$ as shown in Figure 3.6 (note that when w is in the neighborhood of v, then v is in the neighborhood of w).

However, there is the possibility that when $p(\bar{f}|v) = 0.5 + e$, for some small e, $0 < e \ll 1.0$, there may be another channel $\bar{\bar{f}}$ with assignment probability very close to 0.5, $p(\bar{\bar{f}}|v) = 0.5 - e'$ for small e', $0 < e' \ll 1.0$, and for all other channel assignments $\tilde{f}$ in the channel set C, $\tilde{f} \in C \setminus \{\bar{f}, \bar{\bar{f}}\}$ the assignment probabilities are either zero or very close to zero. In such cases we cannot predict with high certainty that channel $\bar{f}$ will be assigned to node v at the end of the algorithm. To reduce the possible errors in these cases we set the threshold of convergence higher than $0.5 + e$. But we cannot increase it too high also, because we want to do these changes while the system has high entropy (high softness in associations) so that it can adopt to the changes. We have found out 0.55 to give good performance.

When we set the assignment probability for channel $\bar{f}$ to zero in node w, $p(\bar{f}|w) = 0.0$, we need to update the assignments for the other channels because we have reduced the probability mass by an amount of $p(\bar{f}|w)$ and we need to satisfy $\sum_i p(f_i|w) = 1.0$. Let $\bar{f} = f_m$ be the channel being blocked and, let the new assignments be $p'(f_0|w), p'(f_1|w), \cdots, p'(f_m|w) = 0, \cdots, p'(f_{K-1}|w)$. Let $p(f_m|w) = g_m$; then we need to distribute g_m among $p'(f_i|w)$ $\forall f_i \in C \setminus f_m$. The most

obvious way of obtaining the new assignments is by normalizing each old assignment (excluding $p(f_m|w)$) by $1-g_m$,

$$p'(f_i|w) = \frac{1}{1-g_m} p(f_i|w) \quad \forall f_i \in C \setminus f_m. \tag{3.20}$$

Note that the assignment in (3.20) is the maximum entropy assignment given the previous set of probabilities.

Therefore, the channel blocking algorithm is the same as the DA algorithm for FAP with the addition of the following step: whenever a node v 'starts to converge' to a channel $\overline{f}$ we set $p(\overline{f}|v) = 1.0$ and $p(\tilde{f}|v) = 0.0 \quad \forall \tilde{f} \in C \setminus \overline{f}$; block the channel $\overline{f}$ for the node(s) w within the neighborhood of node v by setting their assignment probabilities for channel $\overline{f}$ to zero, $p(\overline{f}|w) = 0.0$; update the probability assignments for node(s) w as in (3.20); and continue the iterations. In the Experimental Results section we will demonstrate that the above algorithm effectively reduces the number of co-channel violations with a trade-off of small increase in total network interference compared to without using the channel blocking.

3.4 Experimental Results

We compared the proposed algorithms on two sets of channel assignment problems. In the first case, we generated test problems using a problem generator,

Vertex Saturation, and in the second case we use a realistic frequency planning scenario from the COST 259 project [37].

3.4.1 Generated Test Problems

3.4.1.1 Experimental Setup

The model we have used consists of randomly placed nodes/transceivers on a square field of 25×25 units. The propagation model used is as follows:

i. All transceivers/nodes are assumed to have identical transmission powers and the radiation to be omni-directional.

ii. Free-space propagation loss is assumed to be the only source of signal power attenuation. The decay of signal power P_s with distance d is modeled by the inverse power law [114],

$$P_s = \frac{P_t}{d^g} \tag{3.21}$$

where P_t is the power of the transmitter, d is the distance between the transmitter and the receiver, and g is the fading factor (or propagation exponent) with values between 2 and 6 depending on the environment. We used $g = 4$ as in [6, 114, 54].

iii. The interfering signal power P_i from a transmitter using the same channel as the receiver, i.e., co-channel interference is,

$$P_i = \frac{P_t}{d^g}, \tag{3.22}$$

and from a transmitter using the m^{th} adjacent channel $(m = 1,2,\cdots)$ is,

$$P_i = \frac{P_t}{d^g} \cdot 10^{-a\,(1+\log_2 m)/10} \tag{3.23}$$

where a is the attenuation factor for adjacent channel interference measured in dB/octave. Note that as a is increased the adjacent channel interference decreases (with no effect on co-channel interference). It is shown in [114] that when a is close to 30 dB/octave all adjacent channel interferences can be neglected. We used $a = 15$ dB/octave as in [54] and as suggested in [72]. We also assumed that the total interference power at a receiver location from multiple interfering transmitters is the sum of the individual interferences from each transmitter.

iv. Two different models are used to define the cells (the receiver regions corresponding to each node),

- in the *Voronoi region model* the nearest node to a receiver location provides service to that location, hence the desired signal at each receiver location is from the nearest node,
- in the *best server model* the node achieving the highest signal-to-interference ratio (SIR) at a receiver location provides the service.

3.4.1.2 Vertex Saturation – Problem Generator

The test problems are generated by the Vertex Saturation (VS) algorithm with different sets of thresholds, (r_0, r_1). Vertex Saturation is a problem generator method developed by Zoellner and Beall [118]. It generates test problems with known optimal solutions because the test problems are constructed so that there are no constraint violations for the given co-channel and adjacent channel constraints, r_0 and r_1, and the given number of colors/channels, K. The procedure starts by generating a large number of random points on a 2-dimensional grid. These points are the candidate nodes. Each candidate point is considered in sequence and an attempt is made to color it with the smallest number of color from a set $C = \{0,1,\cdots, K-1\}$ such that both of the following hold:

1. The Euclidean distance between the current point and all previous points that are colored the same color is greater than some threshold r_0.
2. The Euclidean distance between the current point and all previous points that are colored with a color differing by one is greater than threshold r_1 where $r_1 < r_0$.

If there is no feasible color for a candidate, then the candidate is rejected and the next point is considered. If the point can be colored, then the candidate becomes a node and it is added to the constructed graph with the feasible color assigned to that node. The procedure continues until all candidates have been considered. A graph is considered to be saturated when candidates no longer can be added as nodes. The

constructed graph has a minimum span assignment that is exactly the colors selected in C, satisfying both co-channel and adjacent channel constraints. Note that our DA algorithms do not provide a solution that minimizes the span (MS-FAP), but we wanted to run our algorithms on these test problems to see how close they come to satisfying the constraints set by these problems, in addition to minimizing the total interference. Note also that a saturated graph satisfying the co-channel and adj-channel separations between all node pairs will result in a very competitive total interference value, which will be very close, if not equal to the optimum value.

In order to obtain reasonable thresholds, (r_0, r_1) we made use of the results in [114], where channel sizes and (r_0, r_1) thresholds are investigated for various SIRs. Although the work in [114] only considered regular hexagonal networks where the locations of interfering nodes are highly structured, here, we consider networks where the locations of the interfering nodes are random. We have used the following intuition: at a node v, adjacent channel interfering signal power from a distance r_1 from v has to be equal to co-channel interfering signal power from a distance r_0 away from v. The results in [114] show that equal contribution of co-channel interference and adj-channel interference is a reasonable choice. Using the power equation (3.22) for co-channel interference, and (3.23) for adjacent channel interference ($m = 1$) with $g = 4$ and $a = 15$, the ratio r_0/r_1 becomes 2.37:

$$\frac{P_i^{adj}}{P_i^{co}} = 1 \quad \Rightarrow \quad \frac{r_0^g}{10^{a/10} r_1^g} = 1 \quad \Rightarrow \quad \frac{r_0}{r_1} = \left(10^{a/10}\right)^{1/g}$$

$$\Rightarrow \quad \frac{r_0}{r_1} = 2.37$$

Twelve test problems are generated with various number of channels, K and thresholds, (r_0, r_1) using the VS method. In six of these test problems, *P1s*, *P2s*, … , *P6s*, the VS method is allowed to saturate the field with the maximum number of nodes possible given K and (r_0, r_1), and in the other six test problems, *P1*, *P2*, … , *P6*, the number of nodes is left below saturation. The details of these test problems are given in Table 3.1, where $|V|$ is the number of nodes and $|E|$ is the number of edges. The edges represent those pairs of nodes that have Euclidean distances less than r_0 between them. In other words, they are the ones that can cause r_0 or r_1

Problem	K	r_0	r_1	$\|V\|$	$\|E\|$	$\|E\|/\|V\|$	*density (%)*
P1	6	7.11	3.00	45	170	3.78	17.18
P1s	6	7.11	3.00	53	219	4.13	15.89
P2	6	9.48	4.00	28	104	3.71	27.51
P2s	6	9.48	4.00	35	149	4.26	25.04
P3	7	7.11	3.00	50	218	4.36	17.80
P3s	7	7.11	3.00	61	303	4.97	16.56
P4	7	9.48	4.00	32	140	4.38	28.23
P4s	7	9.48	4.00	40	199	4.98	25.38
P5	8	7.11	3.00	55	272	4.95	18.32
P5s	8	7.11	3.00	68	383	5.63	16.81
P6	8	9.48	4.00	36	181	5.03	28.73
P6s	8	9.48	4.00	46	261	5.67	25.22

Table 3.1: Specifications of the test problems.

constraint violations. Also included in the table is the ratio of the number of edges to the number of nodes, where for a given (K, r_0, r_1)-triple (recall that these triples generate the problem instances) the higher this ratio the more saturated the field is and the more difficult it is to satisfy the binary constraints set by the thresholds.

Finally, we also include the edge densities (i.e., number of edges relative to the maximum possible number of $|V| \cdot (|V|-1)/2$) in percentage. Most of the realistic frequency planning scenarios reported in [89] have graph densities in the range of 8% - 14%, hence the test problems we are using are at least as dense as, and sometimes more dense than, the realistic scenarios in [37].

3.4.1.3 Results

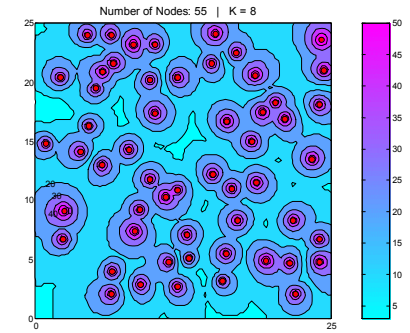
The DA, SA [36, 62] algorithms and the VS method are implemented using C. The computations are performed on an Intel Pentium III processor machine, operating at 550 MHz clock speed and equipped with 256 MB RAM. The convergence times for SA and DA are given in CPU seconds.

For each problem instance 10 experiments are performed and the average results displayed in Table 3.2. Recall that total interference results for VS are very close to optimal results, especially in the saturated problems. From the results in Table 3.2, comparing SA and DA we clearly notice the superiority of the DA over the SA approach, both in performance and in convergence time. In the saturated problem cases, (*P1s – P6s*) the total interference results obtained with DA (Gibbs case) are within 26% to 36% of the optimal, whereas, SA performance ranges between 175% and 380%. In the non-saturated cases, (*P1 – P6*) the total interference performances are 7% to 23% for the DA-Gibbs case and 180% to 428% for the SA. Although, the problem formulations for both DA and SA aimed at minimizing the total

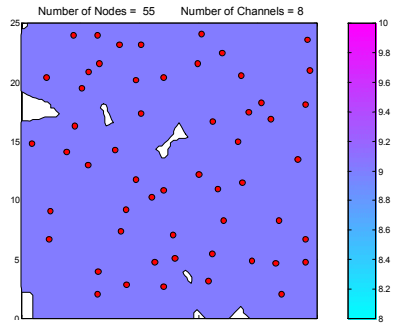
	VS	SIMULATED ANNEALING				DETERMINISTIC ANNEALING (TRIANGULAR DISTRIBUTION)				DETERMINISTIC ANNEALING (GIBBS DISTRIBUTION)			
Problem	*ave. total interfer.*	*ave. total interfer.*	*% diff. from VS ave. interfer.*	*ave. # viol.*	*ave. time (sec.)*	*ave. total interfer.*	*% diff. from VS ave. interfer.*	*ave. # viol.*	*ave. time (sec.)*	*ave. total interfer.*	*% diff. from VS ave. interfer.*	*ave. # viol.*	*ave. time (sec.)*
P1	0.03689	0.11151	202.28	26.4	300.95	0.04549	23.31	9.7	102.49	0.04358	18.13	8.1	175.66
P1s	0.05024	0.14687	192.34	32.7	464.05	0.06902	37.38	18.1	135.12	0.06643	32.23	15.7	220.06
P2	0.00709	0.03723	425.11	19.5	96.53	0.00895	26.23	7.3	56.30	0.00858	21.02	6.2	95.74
P2s	0.01121	0.04428	295.00	26.7	166.12	0.01500	33.81	12.9	84.35	0.01469	31.04	12.2	134.64
P3	0.04034	0.11348	181.31	28.5	511.76	0.05181	28.43	11.9	169.24	0.04793	18.82	9.5	292.49
P3s	0.05924	0.16572	179.74	40.5	878.85	0.08142	37.44	20.2	237.74	0.08073	36.28	20.0	391.90
P4	0.00839	0.04221	403.10	21.9	187.48	0.01078	28.49	8.4	101.19	0.01032	23.00	8.5	166.89
P4s	0.01297	0.06222	379.72	31.4	315.57	0.01784	37.55	17.3	147.78	0.01694	30.61	14.8	235.61
P5	0.04303	0.12199	183.50	32.3	791.02	0.05229	21.52	11.7	268.02	0.05122	19.03	10.5	449.28
P5s	0.06402	0.17565	174.37	43.4	1262.68	0.08788	37.27	23.5	382.92	0.08291	29.51	20.6	619.70
P6	0.00967	0.05108	428.23	25.7	281.97	0.01128	16.65	8.5	165.70	0.01033	6.82	6.6	256.48
P6s	0.01498	0.06804	354.21	38.5	510.26	0.02115	41.19	19.2	245.20	0.01896	26.57	16.0	392.01

Table 3.2: Results comparing the Deterministic Annealing with Gibbs and Triangular membership functions with the Simulated Annealing in terms of average total interference, average number of binary constraint violations and average running time.

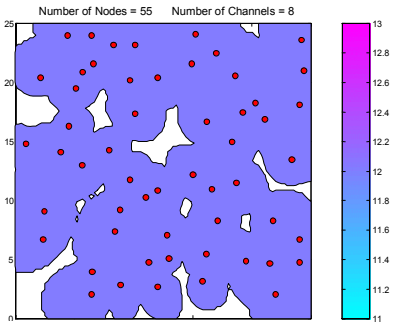
interference, we also computed the number of binary constraint violations in the solutions. We again see that the DA results show up to 4 times fewer violations than the SA results. Besides the number of violations, it is also important to look at the range of distances between the violating node pairs, i.e., how large the violating distances are. This is important because, as we have mentioned before, when considering binary constraints it does not matter how large the violation distance



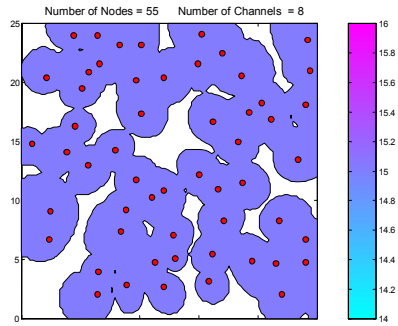
(a) Contour Plot (SA)



(b) Contour Plot, Threshold = 9dB (SA)



(c) Contour Plot, Threshold = 12dB (SA)



(d) Contour Plot, Threshold = 15dB (SA)

Figure 3.7 (a-d) : Contour plots of the field obtained by Simulated Annealing. In (a) full contour plot is shown, and in (b), (c) and (d) thresholded contour plots with thresholds 9dB, 12 dB and 15 dB, respectively, are shown.

between any two nodes v and u is: if it is ε, $r - d(v,u) = \varepsilon$ for very small ε, $0 < \varepsilon \ll 1$, or almost the whole constraint distance, $r - d(v,u) \approx r$ (where $r = r_0$ for co-channel nodes, and $r = r_1$ for adj-channel nodes), the violating node pair is counted as a violation without distinction. However, for all practical purposes, the *quality* (the smaller the violation distance the higher the quality) of the violations

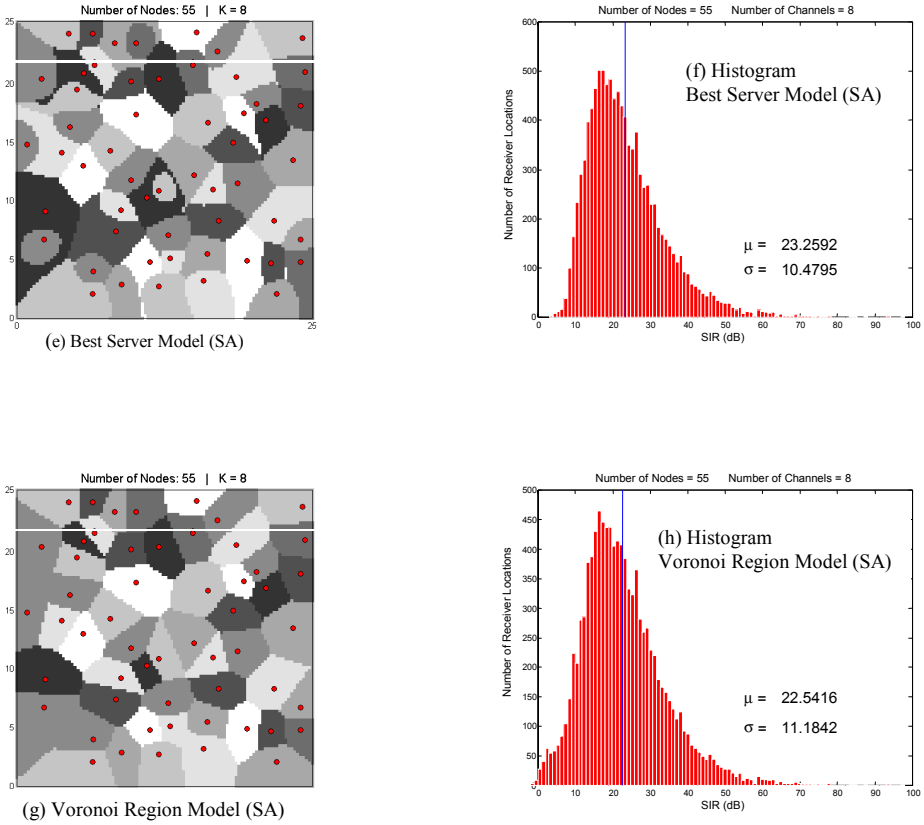
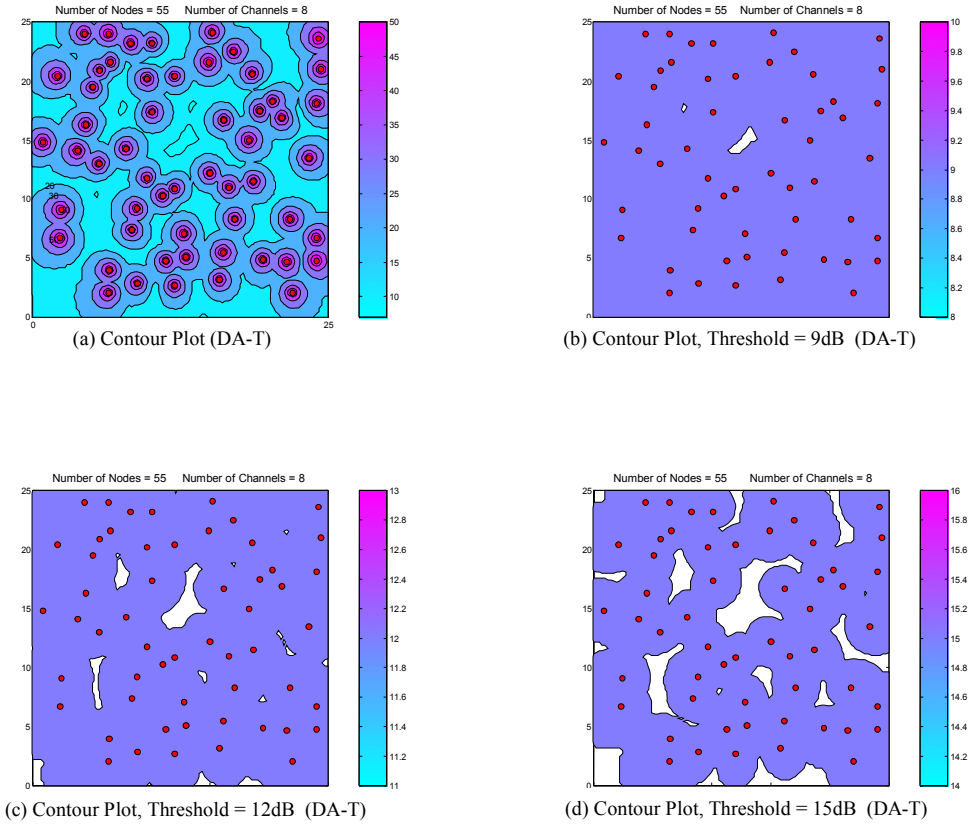


Figure 3.7 continued, (e-h) : Simulated Annealing; cells obtained by the Best Server Model and the Voronoi Region Model are shown in (e) and (g), respectively. And the signal-to-interference (SIR) histograms for each location on the field corresponding to the Best Server Model and the Voronoi Region Model are shown in (f) and (h), respectively.

makes big difference in terms of the accumulated interference, and hence, on the signal-to-interference ratio (SIR); the smaller the violation distances are, the smaller the incurred interferences and the larger the SIR will be. Thus, we computed the distances between the nodes that fall short of satisfying r_0 and r_1 in the DA and SA results. We found out that, on average, the DA violations were at $0.86 \cdot r$, i.e., 14%



(a) Contour Plot (DA-T)

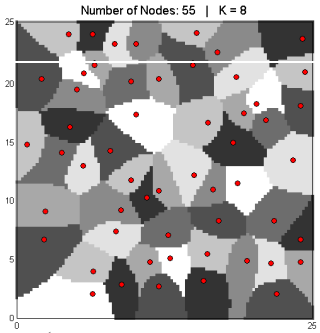
(b) Contour Plot, Threshold = 9dB (DA-T)

(c) Contour Plot, Threshold = 12dB (DA-T)

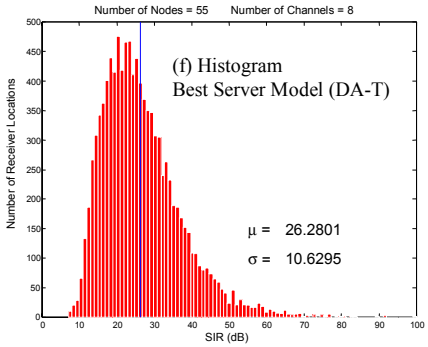
(d) Contour Plot, Threshold = 15dB (DA-T)

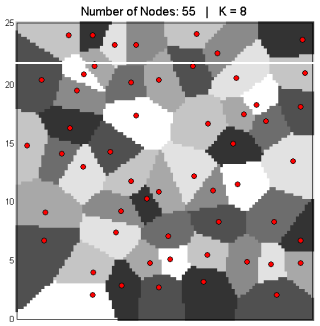
Figure 3.8 (a-d) : Contour plots of the field obtained by Deterministic Annealing Triangular case. In (a) full contour plot is shown, and in (b), (c) and (d) thresholded contour plots with thresholds 9dB, 12 dB and 15 dB, respectively, are shown.

less than the constraint distances ($r = r_0$ for co-channel nodes and $r = r_1$ for adj-channel nodes), whereas, this value for the SA violations was $0.73 \cdot r$, i.e., 27% less than the constraint distances. Therefore, not only do the DA solutions have fewer constraint violations than the SA solutions, but the "quality" of the violations is also higher. This also explains why the DA method outperforms the SA method by a



(e) Best Server Model (DA-T)





(g) Voronoi Region Model (DA-T)

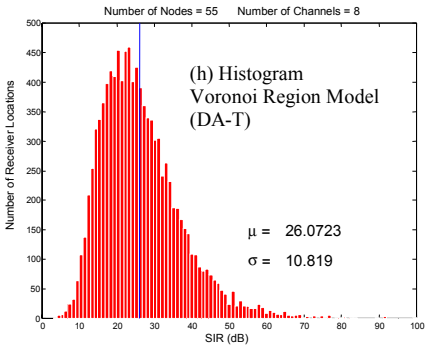
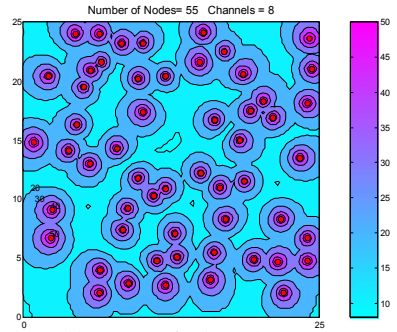
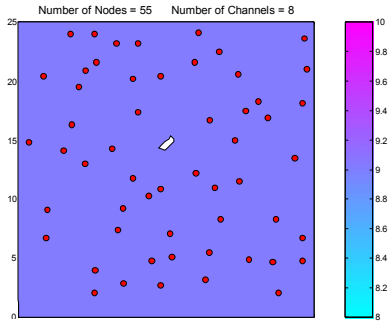


Figure 3.8 continued, (e-h) : Deterministic Annealing Triangular membership function; cells obtained by the Best Server Model and the Voronoi Region Model are shown in (e) and (g), respectively. And the signal-to-interference (SIR) histograms for each location on the field corresponding to the Best Server Model and the Voronoi Region Model are shown in (f) and (h), respectively.

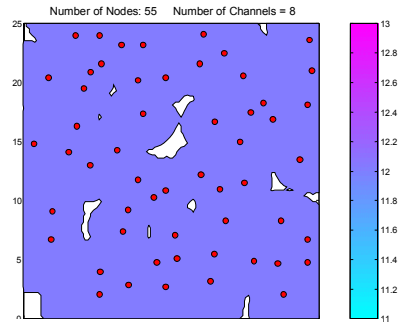
large margin in terms of total interference. When we compare the DA results obtained with the 2 different soft information measures, namely, the Gibbs distribution (DA-G) and the triangular distribution (DA-T), we see that DA-G performs better than DA-T both in total interference and number of violations, but at the expense of longer convergence time; DA-T converges about 40% faster than DA-G with 2% - 10% higher total interference. This was an expected result since Gibbs



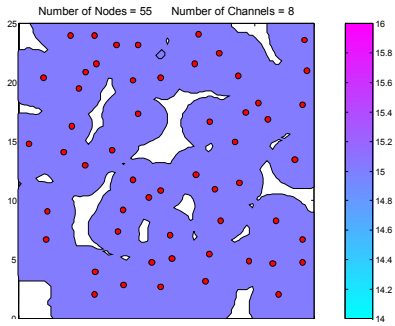
(a) Contour Plot (DA-G)



(b) Contour Plot, Threshold = 9dB (DA-G)



(c) Contour Plot, Threshold = 12dB (DA-G)

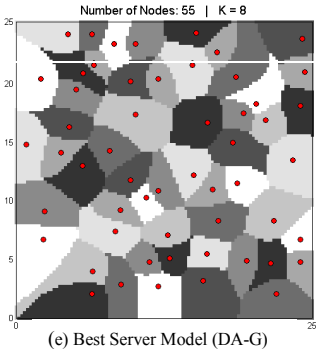


(d) Contour Plot, Threshold = 15dB (DA-G)

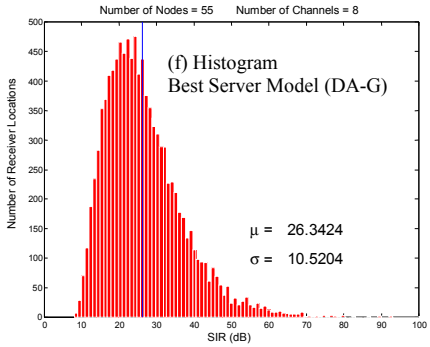
Figure 3.9 (a-d) : Contour plots of the field obtained by Deterministic Annealing Gibbs case. In (a) full contour plot is shown, and in (b), (c) and (d) thresholded contour plots with thresholds 9dB, 12 dB and 15 dB, respectively, are shown.

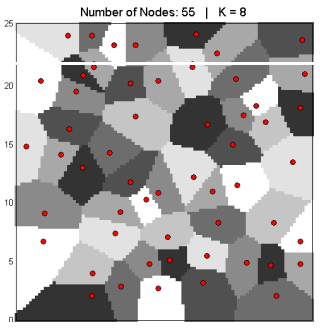
is the optimal distribution, but at the same time it has higher computational complexity.

In practice, interference plots are commonly used in network planning for visualization of the spatial distribution of interference/SIR. Figures 3.7 – 3.9 depict plots for one of the experimental configurations considered for problem *P5* for DA-Gibbs (Figure 3.9a-h), DA-Triangular (Figure 3.8a-h), and SA (Figure 3.7a-h),



(e) Best Server Model (DA-G)





(g) Voronoi Region Model (DA-G)

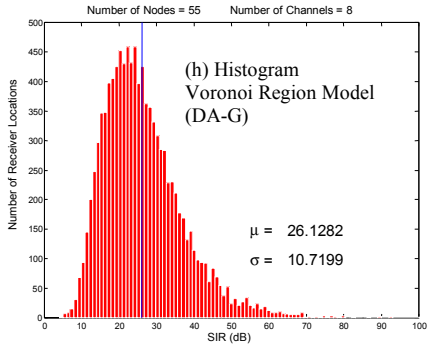


Figure 3.9 continued, (e-h) : Deterministic Annealing Gibbs membership function; cells obtained by the Best Server Model and the Voronoi Region Model are shown in (e) and (g), respectively. And the signal-to-interference (SIR) histograms for each location on the field corresponding to the Best Server Model and the Voronoi Region Model are shown in (f) and (h), respectively.

solutions. The full range SIR-contour plot (Figures 3.7a, 3.8a, 3.9a) corresponding to the best server model show the SIR in dB at each location on the field. The 9dB (Figures 3.7b, 3.8b, 3.9b), 12dB (Figures 3.7c, 3.8c, 3.9c) and 15dB (Figures 3.7d, 3.8d, 3.9d) thresholded contour plots depict the areas covered above the corresponding thresholds, also corresponding to the best server model. The cell maps show the field partitioned into service areas corresponding to both the best server model (Figures 3.7e, 3.8e, 3.9e) and the Voronoi region model (Figures 3.7g, 3.8g, 3.9g). The cells are color coded according to the assigned channel number, such that, cells with zero being the assigned channel are black, whereas the cells with $K-1$ being the assigned channel are white, and the cells with channels $1,2,\cdots,K-2$ are colored with shades of gray from black to white. Finally, the histograms for the best server model (Figures 3.7f, 3.8f, 3.9f) and the Voronoi region model (Figures 3.7h, 3.8h, 3.9h) show the number of locations at each SIR value when the field is sampled at 100×100 location (receiver) points.

We also computed the SIR coverage on the domain of the nodes (the field) for 9dB, 12dB and 15dB, and the results are shown in Table 3.3 (in [37] it is stated that for GSM networks a 9dB SIR threshold is acceptable, whereas in [72] a threshold range of 12dB to 15dB is suggested). As we can see from the Table and from the plots, the coverage is highest in the solutions produced by the DA-Gibbs case. The SA performance lags behind both of the DA-Gibbs and the DA-Triangular performances. Note from the histograms that the average SIR in the SA case is more than 3dB below that of DA averages. The service areas depicted in the cell maps,

	VERTEX SATURATION			SIMULATED ANNEALING			DETERMINISTIC ANNEALING (TRIANGULAR DISTRIBUTION)			DETERMINISTIC ANNEALING (GIBBS DISTRIBUTION)		
PROBLEM	*% coverage area above*			*% coverage area above*			*% coverage area above*			*% coverage area above*		
	9 dB	12 dB	15 dB	9 dB	12 dB	15 dB	9 dB	12 dB	15 dB	9 dB	12 dB	15 dB
P1	96.87	90.64	80.02	92.83	87.87	73.39	95.98	90.59	79.48	97.85	92.46	81.75
P3	98.46	94.85	85.11	95.96	87.92	77.27	98.64	94.56	85.66	98.73	95.20	86.15
P5	99.42	96.03	88.28	97.71	92.21	81.72	99.60	96.69	88.65	99.88	96.91	89.47

Table 3.3: Results comparing Deterministic Annealing Gibbs and Triangular cases, Simulated Annealing and Vertex Saturation in terms of percentage signal-to-interference coverage are above 9dB, 12dB and 15dB.

Voronoi and best server models, indicate further that SA did not produce good channel assignments. Looking at the assignments in the best server model we see that the solution generated cells within cells which definitely imply not good assignments and not good partitioning of the field. Looking at the Voronoi region model, this time we see that the assignments produce co-channel nodes having common borders (recall that the cells are color coded according to the assigned channel to them). Consequently, from both models' point of view, SA did not produce good solutions as the numerical results showed too. The partitionings resulted from the DA solutions, with the Gibbs and the triangular cases, show much better structure compared to the SA partitioning under both models, which were again in accordance with the numerical results.

When we compare DA and VS results in Table 3.3, we see that DA-Gibbs results in a better coverage than the VS approach in all the cases, and DA-Triangular results are comparable with the VS results. In other words, the DA algorithms produced as good as or better assignment solutions in terms of coverage area than the VS on the problems generated by the VS itself. This may be surprising since in terms of total interference VS results were better than the DA results in Table 3.2. But recall that the total sum of the interference results presented in Table 3.2 was between pairs of nodes, i.e., interference from one node position to another node position. And since VS assignments are such that r_0 and r_1 separations are satisfied between pairs of co-channel and adj-channel nodes, respectively, then the cumulative interference on a node from the other nodes was low, resulting in low total interference in the network. Hence, at the location of the nodes and in the very close vicinity of the nodes the SIR is high in VS solutions. However, as the receiver distance from the service providing node increases, the desired signal power decreases and the interference power, not only from the co-channel and adj-channel nodes, but from all of the nodes increase at the receiver's location. Hence, the gradual interference accumulation becomes significant, and algorithms such as DA, that take it into account have an advantage, and thus produce solutions that enjoy higher SIR coverage area.

Finally, in Table 3.4 the results of applying the proposed channel blocking algorithm to the DA – Gibbs case is presented. Recall that the aim of the channel blocking algorithm is to satisfy a given separation constraint while minimizing the

					DETERMINISTIC ANNEALING (GIBBS DISTRIBUTION)					
					without blocking algorithm			*with blocking algorithm*		
K	***r₀***	***|V|***	***|E|***	*density (%)*	*ave. total interfer.*	*ave. # of r_0 viol.*	*% viol. of total edges |E|*	*ave. total interfer.*	*ave. # of r_0 viol.*	*% viol. of total edges |E|*
8	7.11	55	272	18.32	0.0512	6.3	2.30	0.0543	1.1	0.40
10	7.11	65	389	18.70	0.0550	7.0	1.80	0.0567	1.5	0.39
12	7.11	75	533	19.21	0.0599	6.6	1.24	0.0600	0.9	0.17

Table 3.4: Results of the blocking algorithm applied to Deterministic Annealing Gibbs case for channel sets of sizes 8, 10 and 12.

total interference. The algorithm is applied for channels $K = 8,10,12$ and compared with the results where the blocking is not applied. Note that all three problems are almost at the same level of graph density for the given co-channel binary constraint, r_0, hence, the complexity of the problems is kept roughly fixed. Although the blocking did not result in 100% satisfaction of the binary constraints the violations were well below 0.5% after the blocking algorithm is applied, with only a small degradation in average total interference, as compared with the results obtained without blocking. For example, for $K = 12$, where there were 533 constraints (number of pairs of nodes with separation less than r_0) to be satisfied, the number of violations decreased from an average of 6.6 to 0.9 after blocking was used, with a very small increase in average total interference (only 0.12%.) We see a tendency that as the number of channels increases, the blocking algorithm performs better, although the density of constraints is about the same (18% - 19%). This can

be explained by the fact that blocking is done only on one channel, hence when the size of the channel set is large the degree of freedom in the assignment set after blocking stays relatively high compared to the cases when the number of channels is smaller. We can conclude from the results in Table 3.4 that the proposed blocking algorithm can effectively reduce the number of violations with a small trade-off in total interference, and the results get better with increasing number of channels and fixed graph density.

3.4.2 Realistic Frequency Planning Scenario

We have also tested our proposed algorithm on a realistic frequency planning scenario obtained from the COST 259 project [37]. Under this project various realistic GSM frequency planning scenarios are compiled to allow the comparison of different planning methods. Using the scenario named "K" from this set we have tested and compared the performance of our algorithm with the best known techniques that have been tested on the COST 259 project scenarios.

Scenario K is a GSM 1800 network with 92 sites, 264 cells and 267 transmitter/receiver (TRX) units. Fifty contiguous channels form the allowed spectrum. Each site is a collection of cells and each cell can have multiple TRX units as shown in Figure 3.10. The objective is to assign one channel to each TRX from the available 50 channels such that the total interference is minimized. There are also hard constraints that are not allowed to be violated. These are co-cell separation constraints and co-site separation constraints. The co-cell separation

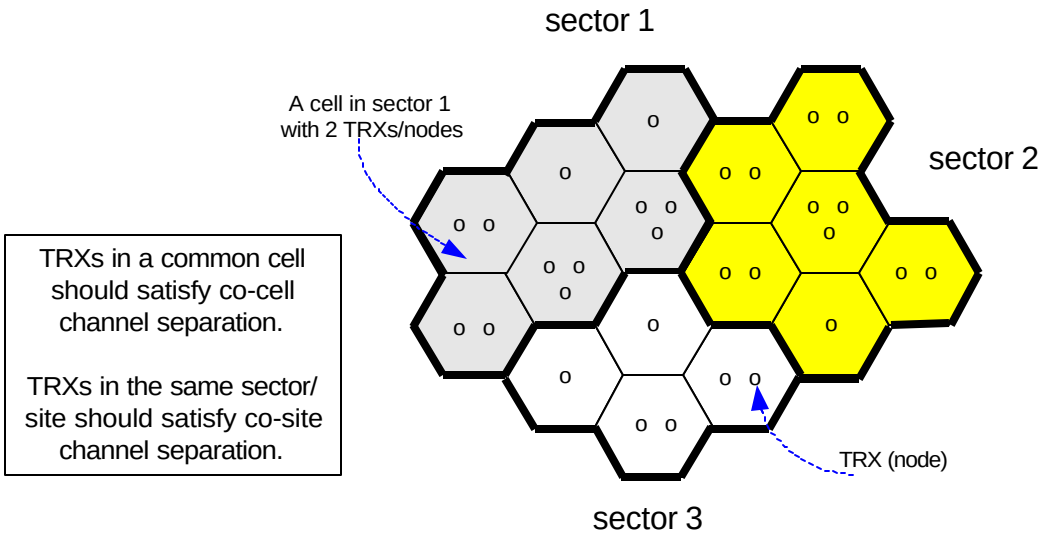


Figure 3.10: Co-site and co-cell channel separations. These are hard constraints and cannot be violated.

constraint requires that any two TRXs within one cell should be assigned channels that are separated by at least 4 channels. The co-site separation constraint requires that any two TRXs within one site in different cells should be assigned channels that are separated by at least 2 channels. From our design perspective each TRX is a node. The characteristics of the scenario is given in Table 3.5.

The results are given in Table 3.6. The methods T-Coloring+VDS, TS+T-Coloring+VDS and TS+DC5+VDS are channel assignment techniques developed in [37], and Threshold Accepting [58] is the algorithm that produces the best assignments on COST 259 scenarios. Among the frequency assignment techniques developed in [37], T-Coloring+VDS, TS+T-Coloring+VDS and TS+DC5+VDS, the performance increases in the given order while the running time increases in that order too. The running times are machine dependent and it is stated in [37] that all

$\|V\|$	Density (%)	Ave. degree	Max. degree	Max. clique	Number of co-channel edges	Number of adj-channel edges	Number of Channels
267	56.57	151	238	69	19111	996	50

Table 3.5: Characteristics of scenario "K" from COST 259 project [37].

three algorithms run within less than 30 seconds on an IBM ThinkPad with Intel PIII processor, 650 MHz clock speed and with 575 MB of RAM. It is also stated in [37] that Threshold Accepting requires an order of magnitude more running time than the other algorithms. On the other hand our proposed algorithm using Deterministic

Assignment Technique	Violations		Total interf.	Number of Edges with Interference above					
	Co-site	Co-cell		0.01	0.02	0.03	0.04	0.05	0.1
T-Coloring+VDS	0	0	1.38	41	14	6	5	1	1
TS+T-Coloring+VDS	0	0	1.25	40	2				
TS+DC5+VDS	0	0	0.82	21	5				
Threshold Accepting	0	0	0.45	6	1				
DA – FAP	0	0	0.95	19	2	2			

Table 3.6: Performance comparison of various frequency assignment techniques on scenario "K" from COST 259 project.

Annealing, denoted as DA – FAP in Table 3.6 has a running time of just over 60 seconds on a Dell, Intel PIII processor, 550 MHz clock speed and with 384 MB of RAM. While a proper comparison of running times is not possible, the results seem to indicate that the complexity of DA-FAP is significantly lower than that of Threshold Accepting technique. In terms of performance Threshold Accepting produces the best result. Our proposed algorithm, DA – FAP, although cannot outperform Threshold Accepting, nevertheless, it is competitive. Note that T-

Coloring+VDS, TS+T-Coloring+VDS, TS+DC5+VDS and Threshold Accepting are all specifically designed for channel assignment scenarios like the scenario "K." Hence, the competitive performance of our algorithm proves it to be a good candidate for further investigation to adapt it for real life channel assignment scenarios.

3.5 Conclusion

Deterministic annealing (DA) has been proposed as a novel approach for the interference minimization variant of the frequency/channel assignment problem (MI-FAP). The DA's concept of starting the process by equal importance to all possible assignments eliminates the requirement of a choice of initial configuration. Through a probabilistic iterative process the DA algorithm is capable of gathering the global information iteration by iteration (via the soft assignment values) resulting in high performance node-channel assignments. The experimental results show that the DA performance outperforms the simulated annealing (SA) performance by a large margin, and moreover, DA converges faster than the SA algorithm. The importance of this conclusion is increased by the announcement in [37] that a variant of the SA method, called Threshold Accepting, presently provided the best performance on a collection of realistic frequency planning scenarios. This method has not been made publicly available yet. However, comparison with the results of the Threshold Accepting algorithm (given in [37]) and with the algorithms developed in [37] show

that the DA algorithm is a competitive channel assignment method. Hence, the competitive performance of our algorithm proves it to be a good candidate for further investigation to adapt it for real life channel assignment scenarios. Recall that Threshold Accepting algorithm and the ones proposed in [37] are specifically designed for such real life scenarios. We believe that the performance of the DA algorithm applied to channel assignment problems presented in this chapter provide substantial evidence that the DA algorithm is an excellent candidate for the channel assignment problem.

The channel blocking algorithm, proposed as an extension to the proposed DA algorithm for FAP, which aims to satisfy a given co-channel separation while minimizing the total interference satisfied the separations in over 99.50% of the edges in exchange of less than 6% increase in total interference compared to the results without blocking. The algorithm performed better with large channel sets satisfying 99.83% of the edges with a small increase in interference (0.12%).

Chapter 4

A Novel Constrained Vector Quantizer Design Based on Multiple Projections and Multiple Stages

4.1 Introduction

The vast majority of practical image coding systems used today are based on the transform coding paradigm, where image blocks are projected onto a series of basis functions, and the expansion coefficients are subsequently quantized. Vector quantization (VQ) techniques have been found to be of somewhat limited practical use for high quality image coding. Unconstrained VQ is limited to rather modest vector dimensions and codebook sizes for practical problems because of the encoding complexity [49]. Constrained VQ techniques (say, for example, tree-structured VQ, TSVQ) can be used for these high dimension sources but often do not make explicit use of special characteristics of the source data, such as the correlation present in typical images. Our motivation in this work comes from considering transform coding as a very efficient constrained VQ algorithm for correlated sources,

where the reconstructed signal is obtained as a linear combination of scaled vectors in the multidimensional space. Our goal in this chapter is also to use projections of the input onto multiple segments, but to tightly couple the selection of these segments with the quantizer design. In contrast, in transform coding the same transform is used for all inputs and the transform is designed without taking quantization into account. Moreover, the segments we select are not constrained to form orthogonal bases.

We propose a novel constrained VQ design algorithm, called Seg-VQ. The constraint requires that the codevectors be located on line segments, where the line segments are free to be anywhere in the source space. These line segments are obtained based on an iterative procedure, where initial segment values are obtained using principal component analysis. Then each input vector in the training set is assigned to the best candidate segment, and the segments are further refined based on all the training vectors that were assigned to each segment. These segments exploit the linear correlations in the source. An input vector is encoded by projecting it on to all the segments and choosing the closest one, as shown in Figure 4.1. The closest codevector on that segment is obtained from a look-up table, and therefore the encoding complexity depends primarily on the number of segments.

We have also designed an entropy constrained version of Seg-VQ, which we call Seg-ECVQ. In this case, once the segments are designed as in Seg-VQ the ECVQ algorithm [19] constrained to the segments is used to get Seg-ECVQ.

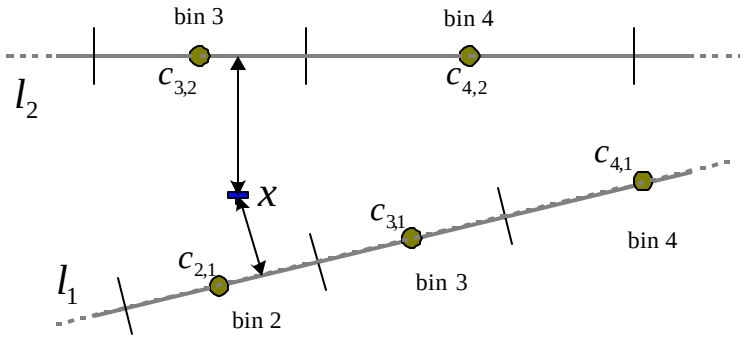


Figure 4.1: Encoding is done by finding the closest segment to *x* and quantizing it with the codevector of the bin it falls into. In this example closest segment is l_1 and projection of *x* falls into bin 2, so *x* is quantized to $c_{2,1}$.

Although Seg-VQ has a low encoding complexity its storage complexity is the same as that of VQ with unstructured codebooks. This prohibits using codes with large dimensions; for example, using $n = 64$ dimensional vectors (8×8 blocks) at $r = 0.5$ bits/pixel require a codebook size of $2^{nr} = 2^{32}$ which is clearly not practical. To circumvent this barrier we can use Seg-VQ in multiple stages in a transform coding flavor. We call this design Seg-MSVQ. In Seg-MSVQ, Seg-VQ is applied to the training set in the first stage and to the residual vectors in subsequent stages, so that specific codebooks are designed for each stage. The reproduction vector for a given input vector is the sum of the quantized representations in each stage. Lagrangian optimization is used to determine the number of stages to be used for each input vector. We keep encoding the input in each subsequent stage until the Lagrangian cost is no longer decreasing. Note that in the first stage we encode the input vector and from second stage onwards we encode the residue vector from the previous stage. Since variable number of stages are used for each input vector there

is an overhead required to indicate the number of stages used to the decoder (this overhead is entropy coded). The segment codevectors in each stage are also entropy coded. While the approach is similar to transform coding, in our system the segments are not constrained to form a basis. As in transform coding a variable number of projections is used for each input. Seg-MSVQ has both low encoding and storage complexity and shows promising performance at low rates.

The rest of the chapter is organized as follows: in section 4.2 we explain the segment based VQ design and present its algorithm (Seg-VQ). We also explain the entropy constrained design of Seg-VQ. In section 4.3 we present experimental results comparing Seg-VQ and its entropy constrained version with PNN [39] initialized GLA on Gauss-Markov and image sources. We extend the Seg-VQ design to multiple stages in section 4.4 (Seg-MSVQ), and in section 4.5 we modify Seg-MSVQ to be more robust to rate adaptation. We present possible extensions and future work in section 4.6. Finally, section 4.7 concludes the chapter.

4.2 Segment-Based VQ Design

4.2.1 Line Segment Based Voronoi Regions

A *line segment* in $\mathbb{R}^n$ is defined as $l(u) = m + u \cdot r$, where m and r are any two vectors in $\mathbb{R}^n$, and u is a bounded scalar $u_{\min} \le u \le u_{\max}$. Let the projection of a vector $x \in \mathbb{R}^n$, $x = (x_0, x_1, ..., x_{n-1})$, onto a line segment between c_i and c_j be b, all

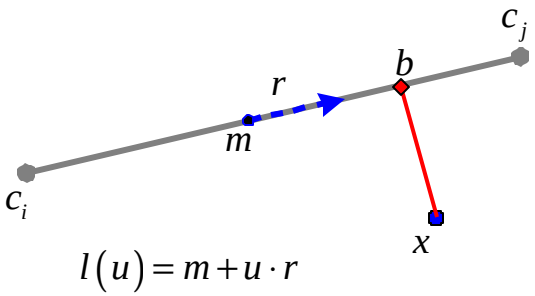


Figure 4.2: Perpendicular projection of a vector on to a line segment. The projection of vector x on the segment is b.

in $\mathbb{R}^n$ as shown in Figure 4.2. Then b is on the line segment defined by u, $b = c_i + u\left(c_j - c_i\right)$. Using the information that b is the perpendicular projection of x on the segment, we can find the scalar u as follows,

$$\begin{aligned} &\left(x - b\right)\cdot\left(c_j - c_i\right) = 0 \\ &\Rightarrow \left(x - c_i - u\left(c_j - c_i\right)\right)\cdot\left(c_j - c_i\right) = 0 \\ &\Rightarrow u = \frac{\sum_{l=0}^{n-1}\left(x_l - c_{i,l}\right)\left(c_{j,l} - c_{i,l}\right)}{\sum_{l=0}^{n-1}\left(c_{j,l} - c_{i,l}\right)^2} \end{aligned}$$

Instead of using the whole segment as a vector, $\left(c_j - c_i\right)$, we can use a unit 1 length vector r, $\|r\|^2 = 1$, along the segment and a vector m on the segment where vector r starts (see Figure 4.2). Then the perpendicular projection b of a vector x on the segment can be defined in terms of m and r,

$$u = \sum_{l=0}^{n-1}\left(x_l - m_l\right)r_l \,. \tag{4.1}$$

The minimum distance between a vector $x \in \mathbb{R}^n$ and a line segment l,

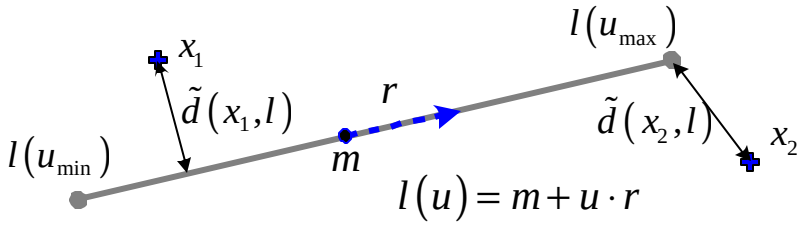


Figure 4.3. Line segment and minimum distances of vectors to it.

$\tilde{d}(x,l) = \min d(x,l)$, is the perpendicular distance from x to l if there is a perpendicular projection, otherwise it is the distance of x to the closer end of l as shown in Figure 4.3. Then, for an arbitrary set of L line segments $\{l_0, l_1, ..., l_{L-1}\}$, we define the line segment based Voronoi regions $V_0, V_1, ..., V_{L-1}$ of a training set T $\left(T = \bigcup_{s=0}^{L-1} V_s\right)$ corresponding to the L line segments as,

$$V_s = \left\{ x : \; s = \underset{j}{\operatorname{argmin}}\, \tilde{d}(x, l_j) \right\}, \quad s \in \{0, ..., L-1\}. \tag{4.2}$$

4.2.2 Optimal Subspace Decomposition

We know that the optimal subspace decomposition can be achieved by the Karhunen-Loeve Transformation (KLT) via the eigenvalue decomposition of the covariance matrix of the training set T, $R = \frac{1}{K} \sum_{x \in T} (x - m) \cdot (x - m)^T$, where $m = \frac{1}{K} \sum_{x \in T} x$ and K is the training set size. Hence, the optimal transformation to a k-dimensional subspace, $k < n$, with respect to minimizing the mean squared error

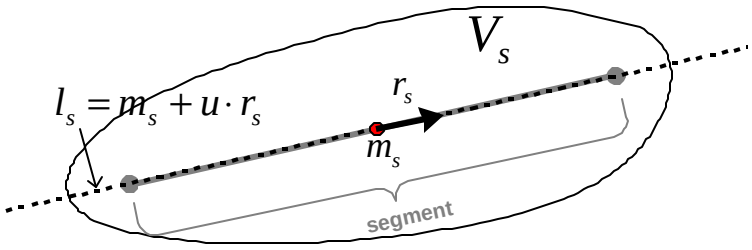


Figure 4.4. First principal component minimizing the mean squared error in V_s and a segment of it.

is the KLT. The $k \times n$ subspace transformation matrix W consists of k rows of the eigenvectors corresponding to the largest k eigenvalues of the covariance matrix R. Assuming that eigenvectors are orthonormal, the transformed vector is $\hat{x} = W^T W x$, and $\|x - \hat{x}\|^2$ is minimum with respect to any other k-dimensional subspace in $\mathbb{R}^n$. Therefore, for a given region $V_s \subset T$ the optimal line $(k = 1)$ $l_s = m_s + u r_s$ minimizing the mean squared error $\|x - \hat{x}\|^2$, where $x \in V_s$ and $\hat{x} \in l_s$, can be obtained by setting $m_s = \frac{1}{|V_s|} \sum_{x \in V_s} x$ and r_s as the unit direction vector in the direction of the eigenvector corresponding to the maximum eigenvalue of the covariance matrix R_s of the region V_s, $R_s = \frac{1}{|V_s|} \sum_{x \in V_s} (x - m_s) \cdot (x - m_s)^T$ as shown in Figure 4.4.

4.2.3 Optimal Location of Codevectors with Line Constraint

Using the subspace decomposition we get the optimal line l_s in V_s which minimizes the mean squared error between a vector $x \in V_s$ and its perpendicular

projection location $\hat{x}$ on l_s. This minimum error is achieved if each $\hat{x} \in l_s$ is used as a codevector. However, we are interested in minimizing the error with a smaller set of codevectors to be placed on l_s. For a given number p_s of codevectors, $C_s = \{c_{0,s}, c_{1,s}, ..., c_{p_s-1,s}\}$, we want to find their optimal locations on l_s such that the squared distortion in V_s,

$$D_s = \sum_{x \in V_s} d(x, c_{i,s}), \qquad i = \underset{j}{\operatorname{argmin}}\, d(x, c_{j,s}) \tag{4.3}$$

is minimized. Note that the two outermost codevectors will determine the segment of $l_s = l(u_s) = m_s + u_s r_s$. In other words, the scalar u_s will be bounded, $u_{\min,s} \le u_s \le u_{\max,s}$, and the two outermost codevectors will be at $l(u_{\min,s})$ and $l(u_{\max,s})$. The locations of the codevectors $\{c_{0,s}, c_{1,s}, ..., c_{p_s-1,s}\}$ are given by $u_s = \{u_{0,s}, u_{1,s}, ..., u_{p_s-1,s}\}$.

To find $C_s = \{c_{0,s}, c_{1,s}, ..., c_{p_s-1,s}\}$ in V_s we use GLA, iterating between the nearest-neighbor condition and line-constrained (in this case l_s) codevector locations. Starting with arbitrary locations for p_s codevectors on l_s, we obtain the Voronoi regions $V_{0,s}, V_{1,s}, ..., V_{p_s-1,s}$ using the nearest-neighbor condition. Then, in each region $V_{i,s}$ we want to find $c_{i,s}$, $i = 0,1,...,p_s - 1$, that minimizes the total distortion in $V_{i,s}$,

$$D_{i,s} = \sum_{x \in V_{i,s}} d(x, c_{i,s}) \qquad i = 0,1,...,p_s - 1.$$

Since $c_{i,s}$ is constrained to be on l_s given by $c_{i,s} = m_s + u_{i,s} r_s$, finding $u_{i,s}$

corresponds to finding $c_{i,s}$ and we can express the total distortion in $V_{i,s}$ as a function of $u_{i,s}$,

$$D(u_{i,s}) = \sum_{x \in V_{i,s}} d(x,\ m_s + u_{i,s} r_s).$$

Using the squared error distortion, $d(x, c_{i,s}) = \|x - c_{i,s}\|^2$ we solve for $\partial D(u_{i,s}) / \partial u_{i,s} = 0$ to get $u_{i,s}$,

$$\frac{\partial D(u_{i,s})}{\partial u_{i,s}} = \sum_{x \in V_{i,s}} \frac{\partial d(x,\ m_s + u_{i,s} r_s)}{\partial u_{i,s}} = 2 \sum_{x \in V_{i,s}} \left(u_{i,s} + r_s^T m_s - r_s^T x \right)$$

$$= 2 \left[|V_{i,s}| \left(u_{i,s} + r_s^T m_s \right) - \sum_{x \in V_{i,s}} r_s^T x \right] = 0$$

$$\Rightarrow \qquad u_{i,s} = \frac{1}{|V_{i,s}|} \left(\sum_{x \in V_{i,s}} r_s^T x \right) - r_s^T m_s. \qquad (4.4)$$

In each $V_{i,s}$, $i = 0,1,...,p_s - 1$, we compute $u_{i,s}$ using (4.4) and we get the corresponding codevector $c_{i,s} = m_s + u_{i,s} r_s$ on l_s. We then update the Voronoi regions $V_{0,s}, V_{1,s}, ..., V_{p_s - 1,s}$ using the codevectors $c_{0,s}, c_{1,s}, ..., c_{p_s - 1,s}$ and iterate until

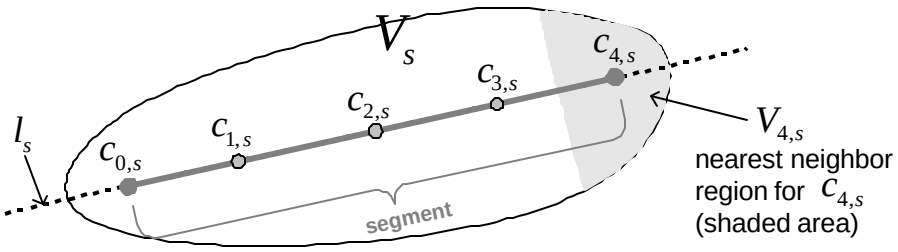


Figure 4.5: The codevectors designed on the first principal component of the region V_s. The two outermost codevectors define the segment.

convergence. After convergence the two outermost codevectors on l_s determine the segment as shown in Figure 4.5, and using (4.2) we get the segment based Voronoi region V_s.

4.2.4 Incremental Addition of Line Segments

In the previous sections we assumed that initially L regions or line segments are given. However, since we do not know the best placement locations for these L segments we start with one line segment and gradually increase the number of segments until some required number, $L = L_r$, or a required distortion threshold is reached. Initially, we start with $L = 1$ and the whole training set as the only region, $V_0 = T$, Figure 4.6 (a). The line $l_0(u)$ is the first principal component of T and we obtain the segment l_0 using $l_0(u)-$ constrained GLA as explained in Section 4.2.3. Then, we search for a location to add the next segment. To avoid obtaining segments representing too few vectors, we search for a training vector z ($z \in T$) such that when we obtain Voronoi regions V_0 and V_z corresponding to the segment l_0 and the vector z, respectively, the cardinality of region V_z is the maximum compared to all other $z \in T$, Figure 4.6 (b). This requires an exhaustive search of the training set and it is computationally complex. To reduce this complexity we can set a threshold, such that when the cardinality of V_z is above that threshold we stop the search. However, exhaustive search of the training set gives better results, and the results

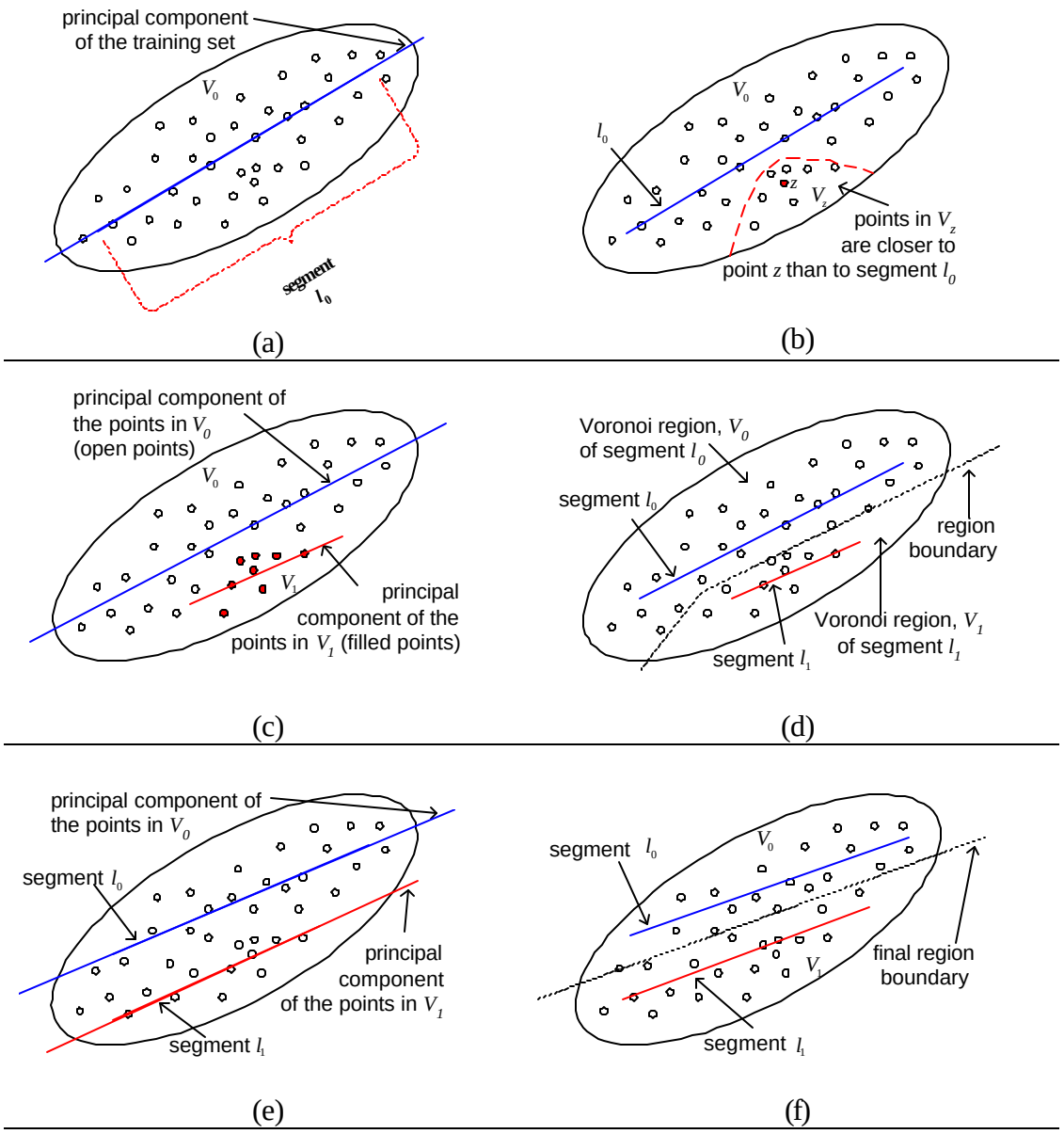


Figure 4.6: To sequentially add the segments, start with the first principal component of the whole input set T as one region V_0 in (a). The best location z for the second segment is such that when we obtain Voronoi regions V_0 and V_z, corresponding to the segment l_0 and the vector z, respectively, the cardinality of region V_z is maximum compared to all other z in T, (b). Set $V_1 = V_z$ as the new region and find the principal components of V_0 and V_1 in (c), and the segments in (d). Update regions V_0 and V_1 and obtain the principal components in (e), and then the segments of the updated regions in (f). Iterate between updating the regions and the segments until convergence.

presented in this chapter are based on exhaustive search. Once we find z and thus V_z, we set $V_1 = V_z$. As shown in Figure 4.6 (c) – (f), we repeat finding the optimal lines, then the optimal segments, and updating Voronoi regions V_0 and V_1 until convergence. We continue adding line segments in this fashion until $L = L_r$.

We stated that the two outermost of the p_s codevectors determine the line segment l_s, but we have not explained how p_s for each region V_s is determined. We found that good performance is obtained if we use $p_s = p = \lceil |C|/L_r \rceil \ \forall s$. Note that this heuristic assignment of equal number of codevectors to each segment is used only during the design of the line segments, where the two outermost codevectors determine the segment. Obviously, it is not necessarily optimal to use equal (or almost equal) number of codevectors on each segment in the final codebook. Therefore, once L_r segments are designed we resort to the popular resource allocation technique, the Lagrangian optimization for the allocation of the number of codevectors to each segment. Lagrangian optimization gives us the optimal number of codevectors to be assigned to each segment such that the total distortion is minimized subject to the codebook size [95]. This is explained next.

4.2.5 Optimal Allocation of Codevectors to Line Segments

Once L segments have been designed, we have to allocate the codevectors to the segments such that the total number of allocated codevectors is equal to the required

codebook size, $M = |C|$. Hence, the problem is to find the optimal number of codevectors, p_s to be allocated to each segment, l_s, $s = 0,1,..., L-1$, such that the total distortion, $D = \sum_{s=0}^{L-1} D_s(p_s)$ is minimized:

$$\min_{\{p\}} \sum_{s=0}^{L-1} D_s(p_s) \quad \text{such that} \quad \sum_{s=0}^{L-1} p_s = M \tag{4.5}$$

In (4.5) $D_s(p_s)$ is the distortion in region s with p_s codevectors allocated to segment l_s. We can put the constrained problem (4.5) in a Lagrangian framework and solve it as an unconstrained problem

$$J = \sum_{s=0}^{L-1} \left(D_s(p_s) + l \cdot p_s \right). \tag{4.6}$$

The non-negative Lagrangian multiplier, $l \geq 0$, allows us to select specific trade-off points between codebook size and distortion. The unconstrained optimization problem can be written as,

$$\min J = \min \sum_{s=0}^{L-1} \left(D_s(p_s) + l \cdot p_s \right) = \sum_{s=0}^{L-1} \min \left(D_s(p_s) + l \cdot p_s \right) \tag{4.7}$$

which means that the minimum can be computed independently for each region [95]. This is also called "constant slope optimization" because the minimum of Lagrangian cost J is obtained with the same l in each region (for details see [95]).

In order to solve (4.7) we need to compute the distortion $D_s(p_s)$ in each region, $s = 0,1,..., L-1$, with all possible codevector allocation sizes, $p_s = 1,2,..., M$. For eachp_s value we find the optimal locations of p_s codevectors constrained to the line

segment l_s as explained in Section 4.2.3, and then compute the resulting distortion, $D_s(p_s)$. The best allocation size p_s for a given l can be obtained by computing $D_s(p_s) + l \cdot p_s$ for $p_s = 1,2,...,M$, and choosing the allocation that gives the minimum, $p_s^* = \text{argmin}_{p_s}\left(D_s(p_s) + l \cdot p_s\right)$. This is done independently for each region. We use the bisection method to find the correct l that achieves the optimal solution for the required codebook size. In other words, we search for the l that gives the minimum Lagrangian cost with the total allocated codevectors $\sum_{s=0}^{L-1} p_s$ equal to the desired codebook size, M. The resulting codevector allocation minimizes the total distortion with the given quota of codevectors.

4.2.6 Seg-VQ Algorithm

Putting all the steps together we have the Seg-VQ algorithm:

0) Initialization: $L = 1$ and $V_0 = T$.

1) For L segment regions, iterate until convergence:

   a. for each V_s, $s = 0,1,..., L-1$, find the first principal component,

      $l_s(u) = m_s + u\, r_s$, $u \in \mathbb{R}$,

   b. for each line $l_s(u)$, $s = 0,1,..., L-1$, use line-constrained GLA to find

      the segment l_s,

c. obtain the segment-based Voronoi regions $V_0, V_1, ..., V_{L-1}$ corresponding to the segments $l_0, l_1, ..., l_{L-1}$.

2) If $L = L_r$ go to step 3), else, find the next segment region, V_L, set $L = L + 1$, and go to step 1).

3) Determine the number of codevectors, p_s to be allocated to each segment l_s, $s = 0, 1, ..., L-1$, using Lagrangian optimization.

4) For each line segment l_s, use line-constrained GLA to find the optimal locations of the p_s codevectors, and stop.

4.2.7 Entropy Constrained Seg-VQ Design

In the previous section we described a segment constrained codebook design for a given fixed codebook size. The index of each codevector can be represented in binary with $\left(\log_2 |C|\right)/n$ bits per sample, where n is the vector dimension. In an entropy constrained design each codevector length is ideally equal to its self-entropy, $-\log_2 p(c)$, where $p(c)$ is the probability of codevector c. The distortion criterion is modified to account for the codevector lengths, so that a given training vector x is mapped to the codevector c_m such that,

$$d(x, c_m) - l \log_2 p(c_m) \le d(x, c_k) - l \log_2 p(c_k) \quad \forall k \tag{4.8}$$

The average binary codevector length is then equal to the index entropy, $H = -\frac{1}{n}\sum_c p(c)\log p(c)$, up to the penalty introduced by the chosen entropy coder (say Huffman coding). The parameter $l \geq 0$ is the Lagrange multiplier whose value will determine the output entropy or the rate, H.

For the entropy and line-segment constrained design, we modify the Seg-VQ algorithm as follows: In step 0) we also require a rate target R to be defined. Steps 1) and 2) do not change, i.e., we obtain the L line segments as before. In step 3) we set a value for l, and instead of allocating a fixed number of codevectors to each segment, we allocate many codevectors on each line segment. In step 4), we replace the Euclidean distance criterion in the nearest neighbor rule with the one in (4.8), which takes the self-entropy into account. We obtain the line-constrained locations of the codevectors using (4.4). In step 3), we iterate between the nearest neighbor rule (4.8) and updating of the segment-constrained codevectors until convergence for the particular value of l set in step 3). This procedure achieves a codebook with codevectors constrained to the L segments and an encoder partition which is optimal for the rate associated with the value of l. For another rate, we go to step 3), set a different value of l, and step 4) produces a different encoder partition and a different codebook, but constrained to be on the same L segments. To reach a desired rate we use the bisection method on l which gradually approaches to the correct l that gives the target rate. Once this is reached, $H = R$, or approached for acceptably small $d > 0$, $|R - H| \leq d$, we stop. Hence, the entropy and segment constrained algorithm, Seg-ECVQ is:

0) Initialization: $L=1$ and $V_0=T$, and rate constraint R.

1) For L segment regions, iterate until convergence:

   a. for each V_s, $s=0,1,...,L-1$, find the first principal component, $l_s(u)=m_s+u\,r_s$, $u\in\mathbb{R}$,

   b. for each line $l_s(u)$, $s=0,1,...,L-1$, use line-constrained GLA to find the segment l_s,

   c. obtain the segment-based Voronoi regions $V_0,V_1,...,V_{L-1}$ corresponding to the segments $l_0,l_1,...,l_{L-1}$.

2) If $L=L_r$ go to step 3), else, find the next segment region, V_L, set $L=L+1$, and go to step 1).

3) Set a value for l, and allocate many codevectors to each segment, $s=0,1,...,L-1$.

4) For each segment, $s=0,1,...,L-1$, use entropy and segment constrained GLA to find the optimal codebook. If $|R-H|\le d$ stop, otherwise go to step 3).

4.3 Experimental Results

We have compared the Seg-VQ algorithm with PNN [39] initialized GLA [80] using two first order Gauss-Markov sources, one with correlation coefficient

$a_0 = 0.0$ (uncorrelated Gaussian source) and the other with $a_0 = 0.9$ (correlated source). We also experiment with an image source, where the 16-dimensional training vectors (corresponding to 4×4 blocks) were obtained from two 512×512 monochrome training images from the USC image database (each with pixel amplitude quantized to 8 bits) and the performance is tested on the image "Lena" which was outside of the training set. For the image source we have also compared the performance of Seg-VQ with the performance of TSVQ [49] at about the same encoding complexity. Recall that Seg-VQ's encoding complexity depends on the number of segments, hence it can be adjusted to match approximately that of TSVQ's. The entropy constrained version of Seg-VQ, Seg-ECVQ, algorithm is compared with the ECVQ algorithm [19] on the Gauss-Markov sources and the image source.

We observe from Table 4.1 that in the case of the uncorrelated Gaussian source the performance of Seg-VQ is well below PNN+GLA especially when the Seg-VQ codebook is constrained to a small number of segments. For example using $L = 4$ segments Seg-VQ is 1.49 dB below PNN+GLA at 0.375 bits/sample (bps) and 3.63 dB at 1.5 bps. However, the encoding complexity of Seg-VQ with $L = 4$ segments is 16 times less than PNN+GLA's full search complexity. With increasing segment size Seg-VQ's performance increases providing a trade-off between complexity and performance. But note that the performance does not increase linearly with the number of segments. The increase in performance gradually decreases with the increase in the number of segments because the resource, in this case the number of

Rate (bps)	PNN +GLA (dB)	TSVQ (dB)	Seg-VQ (dB)				Seg-VQ+GLA (dB)			
			L=4	*L=8*	*L=12*	*L=32*	*L=4*	*L=8*	*L=12*	*L=32*
0.375	2.47	2.15	0.98	1.49	1.95	2.31	2.32	2.31	2.38	2.38
1.5	8.24	7.53	4.61	6.38	7.37	7.99	7.98	8.03	8.02	8.13

Table 4.1: Gaussian source. Codebook size is 64. In Seg-VQ, L is the number of segments.

Rate (bps)	PNN +GLA (dB)	TSVQ (dB)	Seg-VQ (dB)				Seg-VQ+GLA (dB)			
			L=4	*L=8*	*L=12*	*L=32*	*L=4*	*L=8*	*L=12*	*L=32*
0.375	7.92	7.41	6.66	7.20	7.48	7.80	7.78	7.80	7.83	7.90
1.5	13.61	12.93	11.93	12.40	13.01	13.31	13.37	13.51	13.49	13.58

Table 4.2: Gauss-Markov source, correlation coefficient 0.9. Codebook size is 64. In Seg-VQ, L is the number of segments.

codewords, that the segments can share is fixed. Using $L = 32$ segments Seg-VQ achieves a performance that is only 0.2 dB – 0.25 dB below PNN+GLA with an encoding complexity that is half of PNN+GLA (full search of 64 codevectors). Note also that using Seg-VQ as an initialization for GLA, Seg-VQ+GLA, we obtain unstructured codebooks that can perform within 0.1 dB of PNN+GLA performance. Hence, Seg-VQ codebook can also be used as an initialization technique for GLA as an alternative to PNN initialization, avoiding the long PNN algorithm in exchange for a very small performance penalty. To encode an input vector TSVQ requires $2\log M$ distance computations, where $M = |C|$ is the codebook size, and Seg-VQ requires L distance computations. Therefore, Seg-VQ has the same encoding complexity as TSVQ when $L = 12$, since with $M = 64$, $2\log(64) = 12$, and we

observe from Table 1 that at the same complexity performance of Seg-VQ is about 0.15 dB below TSVQ for the uncorrelated source.

In correlated sources, Seg-VQ can perform better compared to uncorrelated sources because it can exploit the correlations in the source. In Table 4.2, using a small number of segments, $L = 4$ ($L \ll M$) it can perform 1.26 dB and 1.68 dB below PNN+GLA at 0.375 bps and 1.5 bps, respectively, compared to 1.49 dB and 3.63 dB at the same rates in uncorrelated source in Table 1. Therefore, this shows that in correlated sources, Seg-VQ can utilize its structure to exploit the directional preferences in the source to close the performance gap between itself and uncorrelated near optimal PNN+GLA. Comparing Seg-VQ with TSVQ at the same encoding complexity $(L = 12)$ in Table 4.2 we notice that Seg-VQ outperforms TSVQ by a small margin. Improvement of Seg-VQ over TSVQ going from an uncorrelated to a correlated source again shows Seg-VQ's ability of exploiting the directional preferences in the source to achieve higher performance codebooks than TSVQ. Therefore, at the same encoding complexity Seg-VQ can perform better than and has lower storage complexity than TSVQ in the case of correlated sources. The storage complexity of TSVQ is nearly double that of unstructured VQ; using binary tree for a codebook size of M the total storage requirement for TSVQ is $2(M-1)$ vectors compared to M for Seg-VQ. Similar to the performance in Table 4.1, the results in Table 4.2 for Seg-VQ+GLA indicate that Seg-VQ is a good alternative to PNN as a codebook initialization.

We have compared the Seg-VQ algorithm with PNN+GLA for performance and with TSVQ for encoding complexity for the image source too. The results shown in Table 4.3 indicate that Seg-VQ can perform very close to the unstructured PNN+GLA with very low encoding complexity. Note that the performance achieved using the Seg-VQ codebook as an initialization for GLA, Seg-VQ+GLA, is very

Rate (bpp)	PNN+GLA (dB)	TSVQ (dB)	Seg-VQ (dB)	Seg-VQ+GLA (dB)
0.4375	29.39	28.75	29.27 *(L=14)*	29.36
0.5	30.15	29.44	29.85 *(L=16)*	30.11

Table 4.3: Test image is "Lena," outside training set. The results are PSNR. Vector dimensions are 16 (4x4 blocks) in both cases. In Seg-VQ, *L* is the number of segments.

close to PNN+GLA and, Seg-VQ's very small performance difference from Seg-VQ+GLA means that Seg-VQ is able to achieve very high performance codebooks with low encoding complexity. At the same encoding complexity, Seg-VQ outperforms TSVQ 0.4 – 0.5 dB at the rates considered. This result, together with the results in Tables 4.1 and 4.2, show that the higher the correlation in the source, the higher the performance gain of Seg-VQ over TSVQ. The results also clearly demonstrate that Seg-VQ is able to exploit the directional preferences in the source to achieve high performance codebooks.

The results in Tables 4.4, 4.5 and 4.6 compare ECVQ and Seg-ECVQ performances for (uncorrelated) Gaussian, Gauss-Markov (correlation coefficient 0.9) and image source, respectively. We observe that, although Seg-ECVQ closes

Rate (bps)	ECVQ (dB)	Seg-ECVQ (dB)			
		L=4	*L=8*	*L=16*	*L=32*
0.375	3.82	0.99	1.50	2.05	2.41
1.5	9.52	4.63	6.47	7.94	8.33

Table 4.4: Gaussian source. Vector dimensions are 4 and 16. In Seg-VQ, *L* is the number of segments.

Rate (bps)	ECVQ (dB)	Seg-ECVQ (dB)			
		L=4	*L=8*	*L=16*	*L=32*
0.375	8.67	6.75	7.27	7.67	7.95
1.5	14.10	12.68	12.92	13.40	13.80

Table 4.5: Gauss-Markov source, correlation coefficient 0.9. Vector dimensions are 4 and 16. In Seg-VQ, *L* is the number of segments.

Rate (bpp)	ECVQ (dB)	Seg-ECVQ (dB)
0.4375	30.20 (retained |C| = 639)	29.56 *(L=14)* (retained |C| = 282)
0.5	30.76 (retained |C| = 1280)	29.95 *(L=16)* (retained |C| = 622)

Table 4.6: Test image is "Lena," outside training set. The results are PSNR. Vector dimensions are 16 (4x4 blocks) in both cases. In Seg-VQ, *L* is the number of segments.

the performance gap compared to ECVQ as the source correlation increases, it cannot outperform ECVQ. This is because, while ECVQ optimizes a VQ's performance minimizing the average distortion for a given rate on the entropy of the codewords, Seg-ECVQ optimizes Seg-VQ's performance for a given rate on the entropy of the codewords. However, the segment constraints in Seg-VQ limit the improvement that it can gain from the ECVQ design principle compared to the improvement that VQ gains. For example, in all the sources considered in Tables 4.4, 4.5 and 4.6 the effective Seg-ECVQ codebook sizes were always less than half

of the effective ECVQ codebook sizes. Hence, while the segment-based structure provides an advantage in correlated sources, the same structure reduces the gain in entropy constraint design. But, we should also bear in mind that we are comparing a constrained VQ to an unconstrained VQ; Seg-ECVQ's encoding complexity depends on the number of segments and ECVQ's is much higher depending on the codebook size. Furthermore, if we consider that ECVQ retains more than twice the number of codevectors than Seg-ECVQ, then we see that ECVQ's storage complexity is also higher than Seg-ECVQ's storage complexity. Hence, in the case of correlated sources Seg-ECVQ becomes an alternative to ECVQ where the choice between them provides a trade-off between performance and encoding – storage complexity.

4.4 Segment Constrained Multistage VQ

4.4.1 Motivation for Multiple Stages

Although Seg-VQ has a low encoding complexity its storage complexity is the same as that for an unstructured VQ codebook. This prohibits using codes with large dimensions; for example, using $n = 64$ dimensional vectors (8×8 image blocks) at rate $r = 0.5$ bits/pixel require a codebook size of $2^{nr} = 2^{32}$ which is not practical.

An alternative technique that has low storage complexity compared to unstructured VQ storage requirement is multistage VQ (MSVQ) [49]. The basic idea of MSVQ is to divide encoding into successive stages, where the first stage quantizes the input vector and after this each successive stage quantizes the error vector from

the previous stage. The quantized error vectors in successive stages provide successive refinement of the input vector, hence the reproduction vector is obtained by sequentially quantizing the residue vector in each stage. This multiple-stage processing results in a product codebook, $C^{(1)} \times C^{(2)} \times \cdots \times C^{(S)}$ that is the Cartesian product of each stage codebook, $C^{(i)}$ $i = 1,2,...,S$ stages. If the size of i^{th} codebook is $M^{(i)} = \left|C^{(i)}\right| = 2^{B_i}$, where $\sum_{i=1}^{S} B_i = n \cdot r$, then the product codebook size is $\prod_{i=1}^{S} 2^{B_i} = 2^{nr}$, and we see that the storage complexity of MSVQ is reduced from $\prod_{i=1}^{S} 2^{B_i} = 2^{nr}$ to $\sum_{i=1}^{S} 2^{B_i} = \sum_{i=1}^{S} M^{(i)}$.

In order to be able to use large dimensions and thus also have low storage complexity besides low encoding complexity, Seg-VQ is designed with multiple stages (Seg-MSVQ). In Seg-MSVQ with S stages, Seg-VQ design is used in each stage to obtain the stage codebook using the residue vectors from the previous stage. In other words the residue vector set generated in stage $k-1$ is the input set to stage k. This set is trained as in Seg-VQ to obtain the codebook in stage k. Then, the residue vector set in stage k is generated with the stage k codebook using the nearest segment encoding of the input set. Hence, note that the residue vector set generated in each stage is unique. The encoding complexity of Seg-MSVQ is $\sum_{i=1}^{S} L_i$ where L_i is the number of segments in stage i, and its storage complexity is the same as in MSVQ. The reproduction vector $\hat{x}$ for an input vector x is the sum of the stage codevectors obtained in each stage,

$$\hat{x} = c^{(1)} + c^{(2)} + ... + c^{(S)} = \sum_{i=1}^{S} c^{(i)} \tag{4.9}$$

where $c^{(i)}$ is the codevector used in stage i to quantize the input vector from the previous stage. The overall quantization error between x and its reproduction $\hat{x}$ after S stages is equal to the quantization error introduced in the last stage. To see this, let $R^{(i)}$ be the residue vector in stage i, where the input vector to stage i is the residue vector in stage $i-1$, $R^{(i-1)}$, and let $\hat{x}^{(k)}$ be the reproduction vector at stage k:

Stage	Residue	Reproduction
stage 1:	$R^{(1)} = x - c^{(1)}$	$\hat{x}^{(1)} = c^{(1)} = x - R^{(1)}.$
stage 2:	$R^{(2)} = R^{(1)} - c^{(2)}$	$\hat{x}^{(2)} = c^{(1)} + c^{(2)}$
		$= \left(x - R^{(1)}\right) + \left(R^{(1)} - R^{(2)}\right)$
		$= x - R^{(2)}.$
⋮	⋮	
stage S:	$R^{(S)} = R^{(S-1)} - c^{(S)}$	$\hat{x} = \hat{x}^{(S)} = c^{(1)} + c^{(2)} + ... + c^{(S)}$
		$= \left(x - R^{(1)}\right) + ... + \left(R^{(S-1)} - R^{(S)}\right)$
		$= x - R^{(S)}.$

(4.10)

Since the residue (error) vectors in stage k form the input vector set to stage $k+1$, the error variance in stage k is the source variance of stage $k+1$. So, the *SNR* in dB of the system becomes sum of the *SNRs* in each stage i, SNR_i [49],

$$SNR = \sum_{i=1}^{S} SNR_i \ . \tag{4.11}$$

Each stage generates an index corresponding to the codevector selected and these indexes are sent to the decoder. Decoding is done by a table look-up for each stage.

4.4.2 Multistage Seg-VQ Training

In the Seg-MSVQ system we assume that all stages are fixed rate vector quantizers, each stage is trained as a separate Seg-VQ using the residue set from the previous stage, and encoding is done by nearest segment encoding.

The codevectors are entropy coded in each stage separately using Huffman coding. Since it is unlikely that each codevector in a stage will be used with the same frequency, entropy coding reduces the average rate for the codevector indexes from $\log_2 M^{(i)}$ to their entropies in all stages $i = 1,2,...,S$. The entropy codes are obtained using the training set and they can be computed sequentially for each stage right after stage training is finished as shown in Figure 4.7. We will denote the length of an arbitrary entropy codeword in stage i as $b_c^{(i)}$. Note that the subscript 'c' in $b_c^{(i)}$ is used to distinguish between bit lengths for codevectors and bit lengths for stage codewords; later we will use subscript 's' for stage codeword lengths, $b_s^{(i)}$.

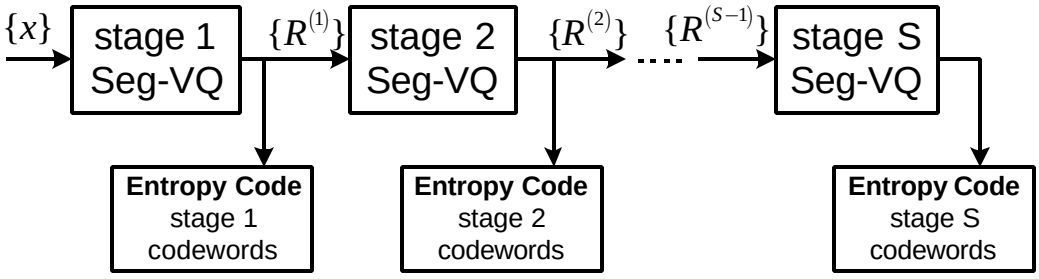


Figure 4.7: Multistage Seg-VQ. In each stage Seg-VQ algorithm is used to generate the stage codebook. After each stage the codevectors are entropy coded using Huffman copding.

Note that using all S (maximum stage number) stages to encode may not always give the best result. It is possible that in a stage k, $k < S$, the residue vector is the zero vector, $R^{(k)} = R^{(k-1)} - c^{(k)} = \mathbf{0}$, i.e., $R^{(k-1)} = c^{(k)}$. From (4.10) we see that in this situation we can perfectly reconstruct x with k stages, $\hat{x}^{(k)} = x$, and there is no need to proceed to the next stage. A more important situation to consider is when we have a rate budget. It is obvious that with each additional stage used the rate increases. So, at a stage k the decrease in distortion to be obtained by encoding with one more stage $(k+1)$ may not worth the increase in rate. In such situations we encode the input vector using stages 1 to k. Therefore, optimal encoding does not necessarily require all stages to be used for each input vector, both with and without a rate budget.

This means that we will use a variable number of stages to encode each vector, and so we need an index to specify the number of stages used for each vector. We

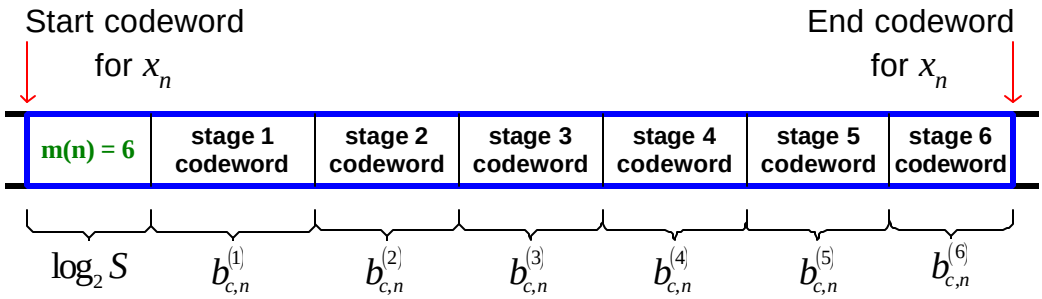


Figure 4.8: Codeword for x_n in the encoded bit stream. The index *m(n)* indicates how many stages are used to encode x_n. In this example the index *m* is not entropy coded. $b_{c,n}^{(i)}$ is the length of the stage *i* codeword for x_n.

can do this in a straightforward manner encoding the indexes into fixed length codewords requiring $\log_2 S$ bits (S is the number of stages). Let this index be m; m takes on values 1,2,…, S. This allows the decoder to know the boundaries of each vector in the bit sequence it receives. The situation is depicted in Figure 4.8.

To find the optimal number of stages $m(n)$ to encode each input vector x_n from an input set such that the total distortion D is minimized for a given rate constraint P, we have to solve

$$\min_{\{m(n)\}} \sum_n \left\| R_n^{(m(n))} \right\|^2 \quad \textit{such that} \quad \sum_n r_n^{(m(n))} \le \mathrm{P}. \tag{4.12}$$

Note that $\left\| R_n^{(m(n))} \right\|^2$ is the squared error incurred by encoding x_n up to stage $m(n)$, and $r_n^{(m(n))}$ is the number of bits required to encode x_n up to stage $m(n)$. We use the Lagrangian optimization to convert the constrained problem in (4.12) to an unconstraint problem parameterized by the Lagrange multiplier l,

$$\begin{aligned} J &= \mathrm{D} + l \cdot \mathrm{P} \\ &= \sum_n \left\| R_n^{(m(n))} \right\|^2 + l \cdot \sum_n r_n^{(m(n))} \\ &= \sum_n \left(\left\| R_n^{(m(n))} \right\|^2 + l \cdot r_n^{(m(n))} \right). \end{aligned} \tag{4.13}$$

To encode an input vector x_n let the codevector selected at stage k be $c_n^{(k)}$. Then the Lagrangian cost of encoding x_n up to stage k is,

$$J_n^{(k)} = \left\| R_n^{(k)} \right\|^2 + l \cdot r_n^{(k)}. \tag{4.14}$$

In (4.14) $R_n^{(k)} = R_n^{(k-1)} - c_n^{(k)}$ and $r_n^{(k)}$ is the number of bits used up to stage k,

$$r_n^{(k)} = \log_2 S + \sum_{i=1}^{k} b_{c,n}^{(i)} \tag{4.15}$$

where $b_{c,n}^{(i)}$ is the length of the codeword in bits corresponding to the codevector $c_n^{(i)}$. So, the cost function in (4.14) is,

$$J_n^{(k)} = \left\| R_n^{(k)} \right\|^2 + l \cdot \left(\log_2 S + \sum_{i=1}^{k} b_{c,n}^{(i)} \right). \tag{4.16}$$

Note that currently we use fixed length codewords, $\log_2 S$, to define the stage; later we will explain how to obtain entropy codes for them. The Lagrangian cost in (4.13) that we would like to minimize over $m(n)\ \forall n$ becomes,

$$\begin{aligned} \min_{\{m(n)\}} J(l) &= \min_{\{m(n)\}} \sum_n \left(\left\| R_n^{(m(n))} \right\|^2 + l \cdot \left(\log_2 S + \sum_{i=1}^{m(n)} b_{c,n}^{(i)} \right) \right) \\ &= \sum_n \min_{\{m(n)\}} \left(\left\| R_n^{(m(n))} \right\|^2 + l \cdot \left(\log_2 S + \sum_{i=1}^{m(n)} b_{c,n}^{(i)} \right) \right) \end{aligned} \tag{4.17}$$

which means that the minimum can be computed independently for each input vector for a given l. The algorithm is shown in Figure 4.9. For each input vector x_n, we compute the residue vector $R_n^{(1)}$ in the first stage and if $\left\| R_n^{(1)} \right\| = 0$ then the reproduction vector is $\hat{x}_n = c_n^{(1)}$, which in this case $\hat{x}_n = x_n$, and go to encode the next input vector, x_{n+1}. Otherwise, we compute the cost $J_n^{(1)}$ using (4.16) with $k = 1$ and go to stage 2. If the stage 2 cost is less than stage 1 cost, $J_n^{(1)} > J_n^{(2)}$, we go to stage 3, and continue until the cost is no longer decreasing. If the minimum cost is achieved for example at stage k, that is, $J_n^{(k)} \le J_n^{(k+1)}$, then the reproduction vector is

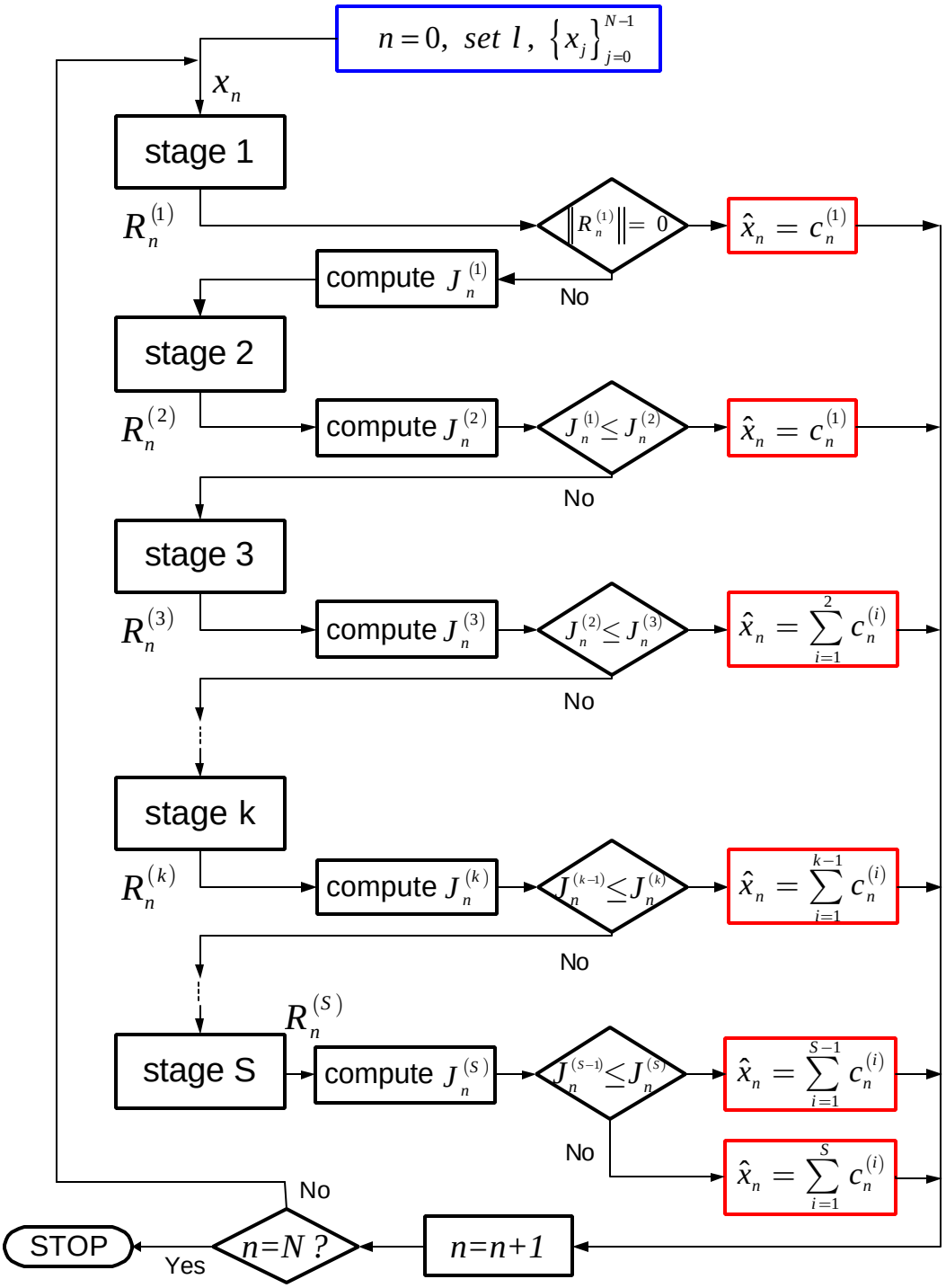


Figure 4.9: Multistage Seg-VQ encoding algorithm for a given rate. The Lagrangian cost function is used to determine the optimal number of stages to be used to encode each input vector for the given rate. 8 is the Lagrange multiplier.

$\hat{x}_n = \sum_{i=1}^{k} c_n^{(i)}$. Recall that in each stage i we use the nearest segment encoding to generate the codevector $c_n^{(i)} \ \forall n$; it is the codevector that the input vector to stage i is quantized with.

We have used fixed length codewords, $\log_2 S$, to define the stages. But, entropy coding reduces the average rate from $\log_2 S$ for stage indexes to their entropies. To obtain the entropy codes for each stage index i and thus their bit lengths $b_s^{(i)}$ (subscript 's' is to denote that the bit length is for a stage index), we start with $b_s^{(i)} = \log_2 S \ \forall i$ and iterate using the algorithm in Figure 4.10, where the block F is the algorithm in Figure 4.9. Define the stage probabilities with $p_s(i) \ i = 1,2,...,S$. So, $b_s^{(i)} = \log_2 S \ \forall i$ means that $p_s(i) = 1/S \ \forall i$ in the first iteration. Let $t(i)$ be the number input vectors that were encoded using stages 1 to i. At the end of each iteration we compute $p_s(i) = t(i)/N \ \forall i$, where N is the training set size, and use

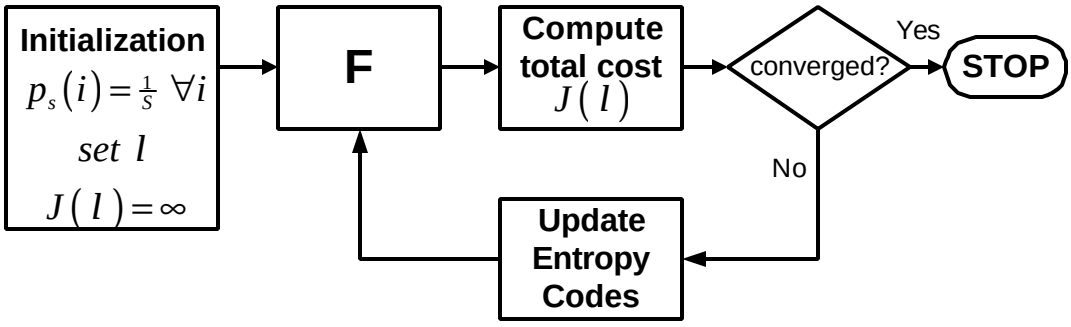


Figure 4.10: The algorithm to obtain the entropy codes for the stage indexes. The block F is the algoritm in Figure 4.9.

$b_s^{(i)} = -\log_2 p_s(i)$ in the next iteration in the cost function. The cost function at stage k becomes,

$$J_n^{(k)} = \left\| R_n^{(k)} \right\|^2 + l \cdot \left(-\log_2 p_s(k) + \sum_{i=1}^{k} b_{c,n}^{(i)} \right). \tag{4.18}$$

We iterate until probabilities be $p_s(i)$ $i = 1,2,...,S$ converge. After convergence we use $p_s(i)$ to get the entropy codes for the stages using Huffman coding, where $b_s^{(i)}$ are the length of the codes in bits. The obtained codes are optimal for the Lagrange multiplier l used which corresponds to a rate. If this rate is not our target rate we have to change l and repeat this process until the desired rate is reached. We have used the bisection method to search for the right l for the target rate.

To encode a source for a given rate budget we use the algorithm in Figure 4.9 with the Huffman tables generated. Let $b_s^{(i)}$ be the length of the binary codeword representing stage i, and let $b_{c,n}^{(i)}$ be the length of the binary codeword representing the codevector $c_n^{(i)}$ which is used to quantize the n^{th} input vector to stage i. Then, at each stage the Lagrangian cost of encoding vector n is,

$$\begin{aligned} J_n^{(k)} &= \left\| R_n^{(k)} \right\|^2 + l \cdot r_n^{(k)} \\ &= \left\| R_n^{(k)} \right\|^2 - l \cdot \left(b_s^{(k)} + \sum_{i=1}^{k} b_{c,n}^{(i)} \right). \end{aligned} \tag{4.19}$$

To encode each input vector x_n, we compute the residue vector $R_n^{(1)}$ in the first stage and if $\left\| R_n^{(1)} \right\| = 0$ then the reproduction vector is $\hat{x}_n = c_n^{(1)} = x_n$, and we go on to

encode the next input vector, x_{n+1}. Otherwise, we compute the cost $J_n^{(1)}$ using (4.19) and go to stage 2. We continue sequentially through the stages until the cost is no longer decreasing. If this happens for example at stage k, $J_n^{(k-1)} \leq J_n^{(k)}$, then the reproduction vector is $\hat{x}_n = \sum_{i=1}^{k} c_n^{(i)}$ and the number of bits used is $r_n^{(k)} = b_s^{(k)} + \sum_{i=1}^{k} b_{c,n}^{(i)}$. We search through l to reach the desired rate budget. In our experiments we used the bisection method to locate the correct l for the desired rate budget.

4.4.3 M−Algorithm for Encoding

In the sequential encoding with Seg-MSVQ explained above, in each stage the nearest-segment encoding is performed greedily without considering the upcoming stages. Hence, it overlooks at the possibility that a sub-optimal codevector choice in stage k (sub-optimality is with respect to nearest-segment encoding) may result in a smaller residue error in a stage $h \geq k+1$. For example, let $R^{(k)}$ be the residue vector in stage k corresponding to the nearest codevector on the nearest-segment, $c^{(k)}$, and let $\overline{R}^{(k)}$ be the residue vector corresponding to codevector $\overline{c}^{(k)}$ such that $\left\|R^{(k)}\right\| < \left\|\overline{R}^{(k)}\right\|$. Seg-VQ selects $c^{(k)}$. However, the residue vectors $R^{(k+1)}$ and $\overline{R}^{(k+1)}$ generated in stage $k+1$ corresponding to $R^{(k)}$ and $\overline{R}^{(k)}$, respectively, may be such that $\left\|R^{(k+1)}\right\| > \left\|\overline{R}^{(k+1)}\right\|$. And so, considering both stages k and $k+1$, $\overline{c}^{(k)}$ is a better

choice than $c^{(k)}$ in stage k. This means that the optimal codevector sequence for the stages requires an exhaustive search to be performed among $\prod_{i=1}^{S} M^{(i)}$ choices, which is the encoding complexity of an unstructured codebook of that size. Since this is not practical for large codebooks, the $\mathrm{M}-$ algorithm is used to improve the performance at the expense of controlled increase in complexity.

The basic idea of the $\mathrm{M}-$ algorithm [49] is to grow a tree of choices where each level of the tree corresponds to a stage and at each level no more than M paths are preserved as shown in Figure 4.11. The tree is constructed out to level p, where p is the largest integer for which $2^p \leq \mathrm{M}$, keeping all 2^p paths. At level (stage) $p+1$ the total number of paths reaches $2^{p+1} > \mathrm{M}$, however, only the "best" paths are saved as candidate codeword sequences. Then, at level $p+2$ all of the M retained paths are extended producing $2\mathrm{M}$ paths, and again only the "best" M of the $2\mathrm{M}$ paths are saved the others are pruned. This is repeated until the last level (stage) S. At this point we have M sequences of codewords from stage 1 to S and the best one is chosen.

Recall that Seg-VQ selects the nearest codevector on the closest segment in each stage. Therefore, at each node of the tree we select: *the nearest codevector on the closest segment* as the first choice and closer of {*the second nearest codevector on the closest segment, the nearest codevector on the second closest segment*} as the second choice. At each level/stage we keep the paths corresponding to the M

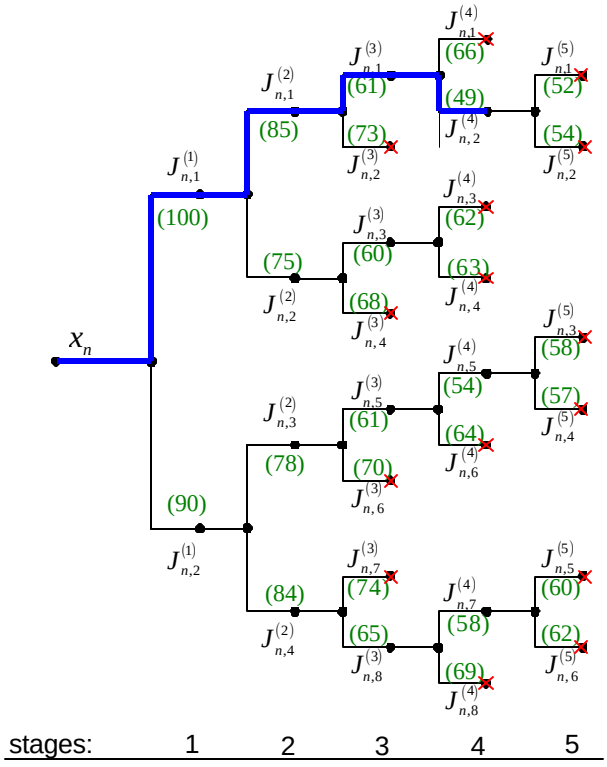


Figure 4.11: An example of encoding using the M-algorithm with M=4. At any stage no more then M paths are allowed; the larger cost paths are pruned. Also, a path is not grown if the cost function is no longer decreasing. In this example the cost at each node is shown in paranthesis. The best path has a cost of 49 at stage 4. Hence, the input vector is encoded using the codeword choices made along that path up to stage 4.

smallest cost, where the cost function is (4.19). A path is grown as long as it is one of the M best paths and the cost is decreasing. We select the minimum cost path at the last stage or if the costs start to increase before reaching the last stage, as shown in the example in Figure 4.11, we select the path up to the stage/level where the cost is minimum.

4.4.4 Experimental Results

We have trained the Seg-MSVQ codebooks using three 512×512 monochrome images from the USC image database (each with pixel amplitude quantized to 8 bits). We used 8×8 blocks corresponding to 64 dimensional vectors and the performance is tested on "Lena" which was outside of the training set. In each stage we trained the stage codebook using 64 codevectors and $L=15$ segments. The results for various stages and rates are shown in Table 4.7. Note that at very low rates (below 0.3 bpp) the results are better than standard JPEG. We observe from the Table that the results improve with increasing number of stages especially at high

Rate (bpp)	JPEG	Seg-MSVQ			
		8 stages	12 stages	16 stages	32 stages
0.20	28.47	29.14	29.48	29.50	29.49
0.25	30.12	30.01	30.39	30.41	30.42
0.30	31.82	30.63	31.11	31.14	31.16
0.40	33.45	31.43	32.14	32.27	32.33
0.50	34.60	31.84	32.84	33.10	33.25
0.75	36.45	--	33.70	34.40	34.92

Table 4.7: Test image is "Lena." The results are in PSNR. In all cases, each stage is designed with 64 codevectors and $L=15$ segments.

rates. With more stages the distortion is naturally expected to decrease, and since rate-distortion optimization is used to determine the optimal number of stages to be used, this results in reducing the distortion the most for the given rate. In other words, depending on the available rate, the Lagrangian optimization determines the number of stages to be used for each input vector. If there are many stages available, the higher order stages can only be used if the rate is high enough. A low rate will

not be enough for higher order stages to be used to encode the input vectors. This can be seen at rate 0.20 bpp in Table 4.7. We see an improvement going from 8 stage to 12 stage case. However, having more stages (16 and 32) does not further improve the performance, because 0.20 bpp does not provide the enough bit resources to go over 12 stages.

The Tables 4.8 and 4.9 show the results obtained using the $M-$ algorithm with $M=4$ and $M=8$ for the 16-stage and 32-stage Seg-MSVQ, respectively. The column "1 path" contains the results from Table 4.7 for comparison. We observe from Table 4.8 that starting from 0.25 bpp we see an improvement in using multiple

Rate (bpp)	M = 1 (single path) In each stage nearest codevector on the closest segment is chosen	M = 4 (paths)	M = 8 (paths)
0.20	29.50	29.51	29.53
0.25	30.41	30.64	30.65
0.30	31.14	31.48	31.51
0.40	32.27	32.79	32.85
0.50	33.10	33.70	33.82
0.75	34.40	35.24	35.30

Table 4.8: Test image is "Lena." Seg-MSVQ with 16 stages.

Rate (bpp)	M = 1 (single path) In each stage nearest codevector on the closest segment is chosen	M = 4 (paths)	M = 8 (paths)
0.20	29.49	29.53	29.54
0.25	30.42	30.62	30.60
0.30	31.16	31.45	31.45
0.40	32.33	32.83	32.86
0.50	33.25	33.81	33.91
0.75	34.92	35.71	35.82

Table 4.9: Test image is "Lena." Seg-MSVQ with 32 stages.

paths over the single path. And as the rate increases the improvement increases; for example, at 0.75 bpp the difference between $M=1$ and $M=4$ is 0.84 dB. Note that this is a large increase considering that the search is narrowed down to only 4 of the possible 2^{16} paths (codeword sequences). The result with $M=8$ at the same rate is only 0.06 dB better than the one with $M=4$. This shows that at rates below 1 bpp we can obtain large performance increase in exchange of a small increase in the search space, and thus, the encoding complexity. Observe that at low rates, for example 0.20 bpp, the improvement going from $M=1$ to $M=4$ (and $M=8$) is negligible. This is because when the rate budget is low, the Lagrangian-based encoding allocates a small number of encoding bits to each input vector, which leads to a small number of stages being used. Therefore, for all of the input vectors in all of the $M>1$ paths, the Lagrangian cost function reaches the minimum at early stages. In Table 4.9, where the results are compared with $M=1$ and $M>1$ for 32 stages, the improvements obtained with the $M-$ algorithm are similar to those in the case of 16 stages in Table 4.8. The only difference is that at high rates we get better results with 32 stages than with 16 stages for both $M=1$ and $M>1$.

4.5 Segment Constraint Multistage VQ with Uniform Quantization of the Segments

4.5.1 Motivation

In the previous sections we have designed systems using Lloyd-Max quantization on the segments. In this section we will consider a system that uses uniform quantization of the segments where the step sizes Δ, and thus the codevectors, will be adapted to the desired rate. The codevectors are at the center of the bins of size Δ and we use mid-rise quantizers on the segments. The reason for the latter is explained below. A system that has adjustable quantization bins rather than fixed bins is more flexible to rate changes.

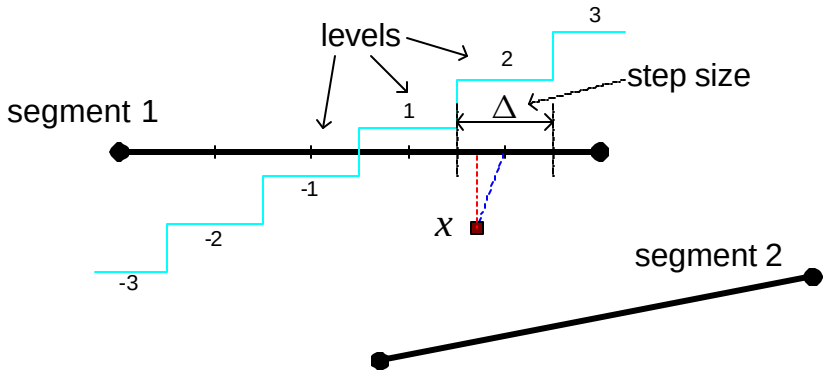


Figure 4.12: The vector x is closer to segment 1 than to segment 2. Its perpendicular projection falls in the bin of level 2. The bin (step) sizes are uniform. x is quantized to the center point of level 2. To be able to reproduce the quantized value of x the decoder needs to know the step size, the segment and the level number.

The system consists of stages with segments and step sizes for each stage. The segments in each stage are designed as in Seg-MSVQ. The encoding of an input vector x in stage k is done by projecting it onto the nearest segment, and quantizing it with the center of the bin it falls into as shown in Figure 4.12. This means that the codeword will need to specify the segment and the level used for quantization. Therefore we need to design codewords for the segment indexes and codewords for level indexes.

4.5.2 System Design

Since we are using the nearest segment there is a unique segment selection for each input vector in each stage. The popularity of the segments is not expected to be uniform and therefore we entropy code the segment indexes. Using the training set we count the number of times each segment is selected and we rank the segments according to their popularity in each stage. The most popular segment in each stage is given index 0, the second most popular segment is given index 1, etc. The frequencies of these indexes are used to entropy code them using Huffman coding. Hence, we have one segment Huffman table for all the stages. We could have obtained separate Huffman tables for each stage, however, we observed that the variation in the popularity of the segments in each stage is not large. To explain this consider a system with 3 stages having 2 segments in each stage, and let the training set size be 100. Then, we say that the variation in the popularity of the segments is

small if the cardinalities of the popular segment in the 3 stages are close to each other, for example, they are 71, 69 and 73. And the cardinalities of the second popular segment are also close to each other; 29, 31 and 27.

Obtaining the level codewords is not straightforward because the number of levels depends on the step sizes and step sizes depend on the rate. For large rates the step sizes are small compared to low rates. Therefore, first of all, we need to find the optimal step sizes for each stage for a given rate such that the total distortion is minimized. To simplify the problem we can think of limiting the number of possible step size choices to p. The initial problem with this approach is how to choose the p step size choices for each stage. However, note that even if we find a way of choosing p step sizes for each stage there is another problem. Each input vector will generate a residue vector for each step size choice, so in stage 1 we will have p different residue vectors for each input vector. This means that there will be p input sets for stage 2 corresponding to each choice in stage 1. At the end of stage 2 there will be p^2 residue sets, and the number of residue sets at the end of stage S will be p^S. Therefore, this approach requires that we find the best step size sequence from the set of p^S sequences, which would be a very complex task. It is apparent that the problem of this approach is the step size (or the quantization) dependency among the stages.

We can decouple the step size dependency among the stages by using the perpendicular projection vectors on the nearest segment in a stage as the input vector

set to the next stage. For example, in stage 1 let the perpendicular projection of x on the nearest segment be $y^{(1)}$, then the residue vector is $R^{(1)} = x - y^{(1)}$. Hence, for each x there is a unique residue vector, and so there is a unique residue vector set $\left\{R^{(1)}\right\}$ which will be the input set for stage 2 as shown in Figure 4.13 where $l^{(1)} = m^{(1)} + u \cdot r^{(1)}$ is the closest segment to x in stage 1. In this way there is a unique input vector set for each stage. The sub-optimality introduced by this approach is small for small step sizes (high rate), and it increases with increasing step sizes.

In order to solve for the optimal step sizes (and so for the optimal number of levels) for each stage for a given rate such that the total distortion is minimized, we need an expression for the total distortion. We will use the decoupling of the stages explained above to derive an upper bound on the distortion in terms of the number of levels. Note that the segments can be anywhere in the space with respect to the

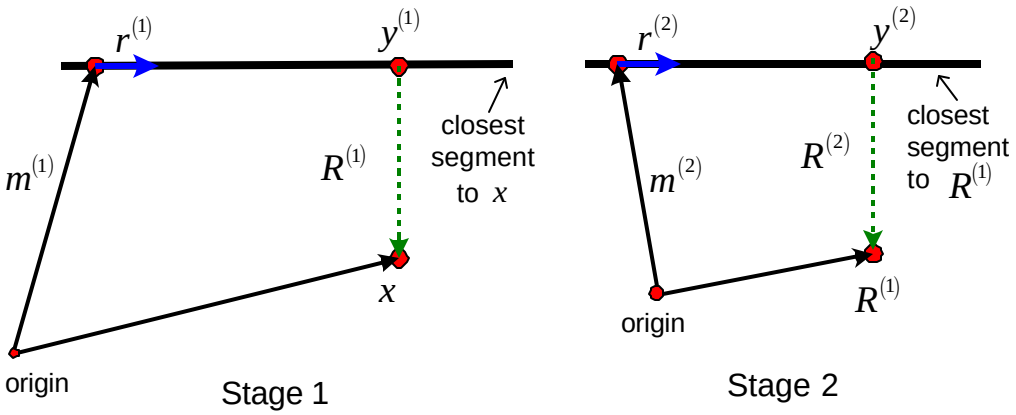


Figure 4.13: In each stage perpendicular projection of the input vectors on the nearest segments are used to generate the residue vectors.

origin, as in the 2D examples in Figure 4.13. In the derivation of the distortion bound we assume that in each stage the coordinate system is shifted by the mean vector, m, of the nearest segment to the input vector in consideration. This brings the vector r of the nearest segment to the origin. We can represent the shifted vectors as follows,

$$x' = x - m^{(1)} \qquad (stage\ 1)$$
$$y'^{(k)} = y^{(k)} - m^{(k)} \qquad (stage\ k)$$
$$R'^{(k-1)} = R^{(k-1)} - m^{(k)} \quad \left(stage\ k,\ R^{(k-1)} \text{ is the input vector to stage } k\right).$$

Note that shifting the coordinate system by the mean vector of the nearest segment has no effect on the derivation of the distortion bound, because the magnitudes of the residue vectors and the quantization errors do not change. For the sake of notational simplicity, in the following we will omit the dashed representation and assume that the coordinate system is shifted by the mean vector in each stage.

We can represent $y^{(1)}$ as,

$$y^{(1)} = \left\langle x, r^{(1)} \right\rangle \cdot r^{(1)} \tag{4.20}$$

where $\langle a,b \rangle$ is the inner product of vectors a and b. Since $R^{(1)}$ is orthogonal to $r^{(1)}$,

$$\|x\|^2 = \left|\left\langle x, r^{(1)} \right\rangle\right|^2 + \left\|R^{(1)}\right\|^2. \tag{4.21}$$

$R^{(1)}$ will be the input vector to stage 2. Let $y^{(2)}$ be the perpendicular projection of $R^{(1)}$ on the closest segment in stage 2, then,

$$R^{(1)} = y^{(2)} + R^{(2)}$$
$$y^{(2)} = \left\langle R^{(1)}, r^{(2)} \right\rangle \cdot r^{(2)}$$

$$\left\| R^{(1)} \right\|^2 = \left| \left\langle R^{(1)}, r^{(2)} \right\rangle \right|^2 + \left\| R^{(2)} \right\|^2 . \tag{4.22}$$

Hence,

$$\|x\|^2 = \left| \left\langle x, r^{(1)} \right\rangle \right|^2 + \left| \left\langle R^{(1)}, r^{(2)} \right\rangle \right|^2 + \left\| R^{(2)} \right\|^2 . \tag{4.23}$$

Note that this does not require that the segments be perpendicular. After k stages, letting $R^{(0)} = x$,

$$\|x\|^2 = \sum_{i=1}^{k} \left| \left\langle R^{(i-1)}, r^{(i)} \right\rangle \right|^2 + \left\| R^{(k)} \right\|^2 . \tag{4.24}$$

So far there is no quantization. Define $a^{(k)}$ as the inner product $\left\langle R^{(k-1)}, r^{(k)} \right\rangle$ at stage k, i.e., $a^{(k)}$ is a coefficient,

$$a^{(k)} = \left\langle R^{(k-1)}, r^{(k)} \right\rangle$$

$$\Rightarrow \qquad y^{(k)} = \left\langle R^{(k-1)}, r^{(k)} \right\rangle \cdot r^{(k)} = a^{(k)} r^{(k)} . \tag{4.25}$$

We can call $r^{(k)}$ as the chosen basis function, recall that $r^{(k)}$ lies on the segment. Let $x^{(k)}$ be the reconstruction of x up to stage k using perpendicular projections on the closest segments, then,

$$x^{(k)} = \sum_{i=1}^{k} y^{(i)}$$
$$= \sum_{i=1}^{k} \left\langle R^{(i-1)}, r^{(i)} \right\rangle \cdot r^{(i)} = \sum_{i=1}^{k} a^{(i)} r^{(i)} . \tag{4.26}$$

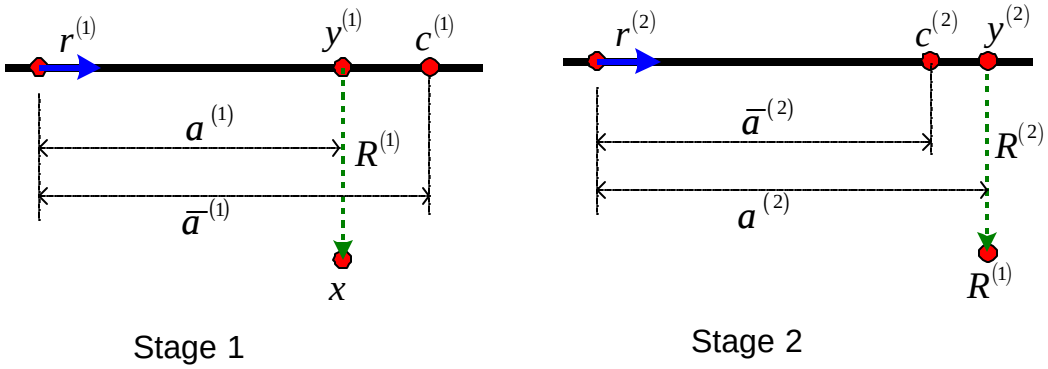


Figure 4.14: Both non-quantized and quantized coefficients are shown. In each stage the quantized coeffiecients will be used to reconstruct the input vectors.

But, we will be reconstructing the input vectors using quantized coefficients. Let the quantized coefficients be $\left\{\bar{a}^{(k)}\right\}$, and let the reproduction vector corresponding to the quantized coefficient be $c^{(k)} = \bar{a}^{(k)} r^{(k)}$ as shown in Figure 4.14. Then, the reconstructed vector at stage k using the quantized coefficients, $\bar{x}^{(k)}$ is,

$$\begin{aligned} \bar{x}^{(k)} &= c^{(1)} + c^{(2)} + \ldots + c^{(k)} = \sum_{i=1}^{k} c^{(i)} \\ &= \sum_{i=1}^{k} \bar{a}^{(i)} r^{(i)}. \end{aligned} \tag{4.27}$$

The squared error distortion between $x^{(k)}$ and $\bar{x}^{(k)}$ is,

$$\begin{aligned} d &= \left\| x^{(k)} - \bar{x}^{(k)} \right\|^2 = \left\| \sum_{i=1}^{k} a^{(i)} r^{(i)} - \sum_{i=1}^{k} \bar{a}^{(i)} r^{(i)} \right\|^2 \\ &= \left\| \sum_{i=1}^{k} \left(a^{(i)} - \bar{a}^{(i)} \right) r^{(i)} \right\|^2. \end{aligned} \tag{4.28}$$

Recall that $\left\| r^{(1)} \right\|^2 = 1 \quad \forall i$. Using triangular inequality we can upper bound the distortion,

$$
\begin{aligned}
d &= \left\| \sum_{i=1}^{k} \left(a^{(i)} - \bar{a}^{(i)} \right) r^{(i)} \right\|^2 \\
&\le \sum_{i=1}^{k} \left| \left(a^{(i)} - \bar{a}^{(i)} \right) \right|^2 \left\| r^{(i)} \right\|^2 \\
&= \sum_{i=1}^{k} \left| \left(a^{(i)} - \bar{a}^{(i)} \right) \right|^2 . \qquad (4.29)
\end{aligned}
$$

Therefore, the upper bound on the total distortion for all input vectors using all S stages is,

$$
D \le \frac{1}{N} \sum_{n=0}^{N-1} \sum_{i=1}^{S} \left| \left(a_n^{(i)} - \bar{a}_n^{(i)} \right) \right|^2 \qquad (4.30)
$$

where N is the training set size and, $a_n^{(i)}$ and $\bar{a}_n^{(i)}$ are the non-quantized and quantized coefficients of the n^{th} vector in stage i.

Recall that we were going to uniformly quantize the segments. Using a step size of Δ the quantization error lies in the interval $\left[-\frac{\Delta}{2}, \frac{\Delta}{2} \right]$. For small Δ we assume that the input is uniform in this interval, and thus the mean squared quantization error in the interval $\left[-\frac{\Delta}{2}, \frac{\Delta}{2} \right]$ is $\Delta^2/12$. Recall also that we were going to quantize the segments in each stage. Since the segments in a stage are not necessarily of same length we will use the average segment length in each stage. Let the average segment length in stage k be L_k and let m_k be the number of equal intervals on L_k. Then the step size Δ_k in stage k is,

$$
\Delta_k = \frac{L_k}{m_k} \qquad (4.31)
$$

and the uniform quantization error in stage k is,

$$\frac{\Delta_k^2}{12} = \frac{L_k^2}{12 \cdot m_k^2}. \tag{4.32}$$

Using (4.32) we can represent the upper bound on the total distortion in (4.30) as

$$\begin{aligned} D &\le \frac{1}{N} \sum_{n=0}^{N-1} \sum_{i=1}^{S} \left| \left(a_n^{(i)} - \bar{a}_n^{(i)} \right) \right|^2 \\ &= \frac{1}{N} \sum_{n=0}^{N-1} \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} \\ &= \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2}. \end{aligned} \tag{4.33}$$

Hence, we have an expression of the total distortion in terms of the number of levels m_i for each stage, $i = 1,...,S$. Since we know average segment lengths L_i we can compute Δ_i using (4.31) if we know m_i. Therefore, to obtain the optimal number of levels for each stage, $i = 1,...,S$, we need to solve for m_i such that the total distortion is minimized for a given bit rate. Let the total rate be H, which is the sum of the rates in each stage, where the rate in a stage k, H_k is a function of the number of levels, m_i, $H_k(m_k)$. We can solve this problem using the Lagrangian optimization using the distortion expression in (4.33),

$$\begin{aligned} J(l) &= D + l \cdot H \\ &= \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} + l \cdot \sum_{i=1}^{S} H_i(m_i). \end{aligned} \tag{4.34}$$

The level probabilities are assumed to be uniform which assumes fixed length codewords for ease of computation, and so (4.34) becomes,

$$J(l) = \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} + l \cdot \sum_{i=1}^{S} \log_2(m_i). \tag{4.35}$$

To find optimal $m_k \ \forall k$, at the given l, solve for $\left(\partial J(l)/\partial m_k\right) = 0$:

$$\frac{\partial J(l)}{\partial m_k} = -\frac{L_k^2}{6 \cdot m_k^3} + \frac{l}{m_k \ln 2} = 0 .$$

Finally, the optimal number of step sizes for a given rate budget dictated by l are,

$$m_k = L_k \sqrt{\frac{\ln 2}{6 \cdot l}} \quad \forall k , \tag{4.36}$$

and so the optimal step sizes are,

$$\Delta_k = \sqrt{\frac{6 \cdot l}{\ln 2}} \quad \forall k . \tag{4.37}$$

Therefore, the optimal step sizes are the same for all stages.

We train our system using the result in (4.37) for a given rate budget. In other words, we search for the correct l using (4.37) that gives us the desired rate. Although the result in (4.37) assumes that the level indexes will be of fixed length (this assumption means that in each stage the level probabilities are equal), in practice the frequency of occurrence of each level in a stage k using Δ_k will not be the same. More specifically, in each stage there are more vectors that are projected to the center of the segments than to the edges. Hence, entropy coding of the level indexes improves the result. To take advantage of the fact that projections are concentrated at the center we combined the negative and positive indexes of the same integer into one category. Hence, level indexes $\{-1,1\}$ form category 1, $\{-2,2\}$ form category 2, etc. A sign bit is used to differentiate between the two selections in a category. Therefore, to represent a codeword in a stage k we specify,

$$segment\ codeword + category codeword + signbit\ .$$

For the n^{th} input vector to stage k let the length of the chosen segment codeword be defined as $b_{l,n}^{(k)}$. Then the length of the codeword required to encode the n^{th} input vector in stage k using fixed length codewords for the categories is,

$$b_{c,n}^{(k)} = b_{l,n}^{(k)} + \log_2\left(\frac{m_k}{2}\right) + 1 \tag{4.38}$$

Note that if there are m_k levels, using 2 levels in each category gives $m_k/2$ categories. And since we are using mid-rise quantizers there is an even number of levels resulting in an integer number for the category size. Using (4.38) the total codeword length required to encode x_n up to stage k is,

$$\sum_{i=1}^{k} b_{c,n}^{(i)} = \sum_{i=1}^{k}\left(b_{l,n}^{(i)} + \log_2\left(\frac{m_i}{2}\right) + 1\right) \tag{4.39}$$

We want to compute the entropy codes for each category in each stage. Hence, we need the category probabilities. To obtain the category probabilities in each stage for a given l we start with uniform assignments and iterate until they converge. In other words, given l we find the number of levels using (4.36) and assign uniform probabilities to the corresponding categories: for example, if $m_k = 10$ then there are 5 categories, $\{-1,1\}$, $\{-2,2\}$, $\{-3,3\}$, $\{-4,4\}$ and $\{-5,5\}$, each with probability 0.2. In each iteration we keep track of the number of occurrences of the categories in each stage, compute their probabilities and generate their Huffman codes. We use these codes in the next iteration. We repeat this until convergence. After

convergence we have the entropy codes for the category indexes in each stage for the given l. Let the length of the category codeword that encoded the n^{th} input vector to stage k be $b_{t,n}^{(k)}$, where the subscript 't' is to indicate category, then, (4.39) becomes,

$$\sum_{i=1}^{k} b_{c,n}^{(i)} = \sum_{i=1}^{k} \left(b_{l,n}^{(i)} + b_{t,n}^{(i)} + 1 \right). \tag{4.40}$$

This is the total rate required to encode x_n up to stage k.

Therefore, after the system is trained for each stage we have a category Huffman table and one Huffman table for the segments. Note that the decoder also needs the index for stage k to be able to decode correctly. We can either use fixed length codewords for the stage indexes, $b_s^{(i)} = \log_2 S$ for S stages, or we can obtain their entropy codes while obtaining the entropy codes for the category indexes. We can do the latter by alternating between *updating the category index codewords with fixed stage index codewords*, and *updating the stage index codewords with fixed category index codewords*. This is repeated until convergence. In this way we obtain a Huffman table for the stage indexes too.

The encoding of an input set is done using the algorithm in Figure 4.9 where in each stage we project the input vector on the nearest segment, compute the level it falls into using the step size, compute the residue error, and compute the Lagrangian cost. The cost of encoding the n^{th} input vector up to stage k is,

$$
\begin{aligned}
J_n^{(k)} &= \left\| R_n^{(k)} \right\|^2 + l \cdot r_n^{(k)} \\
&= \left\| R_n^{(k)} \right\|^2 + l \cdot \left(b_s^{(k)} + \sum_{i=1}^{k} b_{c,n}^{(i)} \right) \\
&= \left\| R_n^{(k)} \right\|^2 + l \cdot \left(b_s^{(k)} + \sum_{i=1}^{k} \left(b_{l,n}^{(i)} + b_{t,n}^{(i)} + 1 \right) \right). \qquad (4.41)
\end{aligned}
$$

To encode the input set for a given rate budget we search for the correct l using the bisection method. We have named this system Seg-U-MSVQ, where 'U' is used to indicate the uniform quantization nature of this system.

4.5.3 Experimental Results

The trained Seg-U-MSVQ using $S = 16$ stages for a high rate, 1.2 bpp, and for a lower rate, 0.6 bpp. The training set used was the same one that was used to train the Seg-MSVQ in the previous section. We have tested Seg-U-MSVQ on the Lena image, which was outside the training set. The results are shown in Figure 4.15. The Seg-MSVQ plot is also shown as a reference. Note that when the system is trained for a high rate, 1.2 bpp (small step size) it performs better than Seg-MSVQ at the design rate 1.2 bpp and at mis-match rates down to about 0.45 bpp. On the other hand, when Seg-U-MSVQ is trained for a relatively lower rate, 0.6 bpp, then we see that its performance is below Seg-U-MSVQ optimized for 1.2 bpp at all rates. At low rates, we would have expected to see Seg-U-MSVQ optimized for 0.6 bpp to outperform Seg-U-MSVQ optimized for 1.2 bpp. However, this is not the case. The

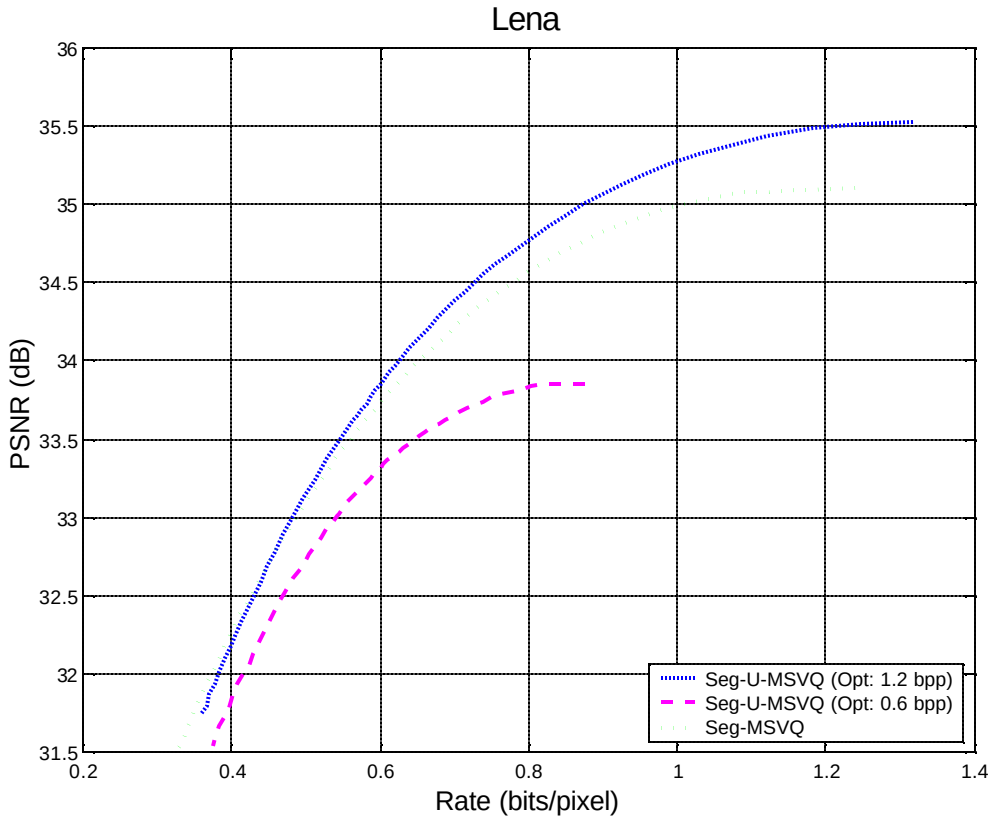


Figure 4.15: The Seg-U-MSVQ system optimized for a high rate, 1.2 bpp, and for a lower rate, 0.6 bpp. The performance is shown on image Lena.

reason for this is that the upper bound for the distortion is tight for small step sizes, and the step sizes that are derived are for fixed length codewords. At high rates the step sizes are small and the level probabilities are more uniform requiring codewords close to uniform length. Hence, the equation for the step sizes is optimal for high rate quantization, and the results confirm this.

4.6 Possible Extensions - Future Work

4.6.1 Hybrid Models

As the number of stags increase the residue vectors become more random and so the residue vector set become less correlated. We have seen that segment constraint VQ's performance decreases as the correlation in the source decreases. Therefore, we can use hybrid models where the lower order stages use Seg-VQ and higher order stages use, for example, either unstructured VQ or TSVQ. Using unstructured VQ keeps the storage complexity same as Seg-VQ while increasing the encoding complexity. And using TSVQ keeps the encoding complexity comparable to Seg-VQ increasing storage complexity. Therefore, there is a storage – encoding complexity tradeoff between these two hybrid models.

4.6.2 Joint Stage Quantizer Design in Seg-MSVQ

In multistage Seg-VQ, Seg-MSVQ, each stage is designed as if it is the last stage. In other words, each stage codebook is generated considering only the error due to the previous stages, i.e., the causal error. The anticausal error, the error due to the subsequent stages is ignored. A joint design approach that takes both the causal and anticausal errors into account will reduce the overall error. The joint design will increase the performance of Seg-MSVQ at a given rate compared to causal (or greedy) Seg-MSVQ performance.

4.6.3 Using Level Entropies in the Design of Optimal Step Sizes

We have designed the optimal step sizes for the uniform quantization of the segments assuming that the level probabilities will be uniform, and so the level indexes will have fixed codeword lengths. Specifically, we solved for the optimal number of levels, $m_i \;\; i = 1,2,...,S$, using the cost function in (4.35) which is shown here again for convenience,

$$\begin{aligned} J(l) &= \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} + l \cdot \sum_{i=1}^{S} H_i(m_i) \\ &= \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} + l \cdot \sum_{i=1}^{S} \log_2(m_i). \end{aligned} \tag{4.42}$$

The solution to (4.42) assumes that each level of the optimal number of levels, m_i in a stage i, will be represented with $\log_2 m_i$ bits. Which means that the probability of each of the m_i levels are assumed to be uniform: $p_i(l) = 1/m_i$ for all $l \in \{-m_i/2,...,-2,-1,\ 1,2,...,m_i/2\}$. However, the frequency of occurrence of each level is not necessarily the same. Therefore, the optimal number of levels, m_i $i = 1,2,...,S$, should be solved using,

$$\begin{aligned} J(l) &= \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} + l \cdot \sum_{i=1}^{S} H_i(m_i) \\ &= \sum_{i=1}^{S} \frac{L_i^2}{12 \cdot m_i^2} - l \cdot \sum_{i=1}^{S} \sum_{\substack{l=-m_i/2 \\ l \neq 0}}^{m_i/2} p_i(l) \log_2 p_i(l) \end{aligned} \tag{4.43}$$

which does not assume uniform distribution of the levels.

4.7 Conclusion

The vast majority of practical image coding systems used today are based on the transform coding paradigm, where image blocks are projected into a series of basis functions, and the expansion coefficients are subsequently quantized. In this chapter we introduced a novel constrained vector quantizer (VQ), which we called Seg-VQ. As an extension of the transform coding framework, in our approach codevectors are constrained to be located on a series of line segments in the multidimensional space. These segments are designed sequentially based on a training set. The advantages of Seg-VQ are twofold: first, the encoding complexity is proportional to the number of segments rather than to the number of codevectors, and second, it can efficiently exploit the directional preferences (correlations) in sources such as images. For image sources, at low dimensions (e.g., 4 by 4 blocks), with the same encoding complexity of TSVQ, Seg-VQ outperformed TSVQ by 0.5 dB at 0.4375 bpp achieving a performance close to the optimal fixed rate unconstrained VQ. At higher dimensions (e.g., 8 by 8 blocks) we use multi-stage Seg-VQ where the input block (as in transform coding) is projected into a series of segments in order to be quantized. We have proposed two different systems using multiple stages: In the first one we designed fixed codevectors constrained to be on the segments using Lloyd-Max quantization. And in the second one there are no fixed codevectors on the segments; the segments are uniformly quantized depending on the required rate

making it more robust for rate adaptation. The latter proposed system is optimal for high rate quantization.

Bibliography

[1] E. H. Aarts, J. Korst and P. J. van Laarhoven, "Simulated annealing," in *Local Search in Combinatorial Optimization*, (E. H. Aarts and J. K. Lenstra eds.), John Wiley and Sons, 1997.

[2] D. Abramson, "Constructing school timetables using simulated annealing: sequential and parallel algorithms," *Management Science*, 37(1), pp. 98-113, 1991.

[3] B. Alidaee, B. G. Kochenberger and A. Ahmadian, "0-1 quadratic programming approach for the optimal solution of two scheduling problems," *International Journal of Systems Science*, vol. 25, pp. 401-408, 1995.

[4] A. Anastasopoulos and K. M. Chugg, "Iterative Equalization/Decoding of TCM for Frequency-Selective Fading Channels," *Asilomar Conf. on Signals, Systems and Computers*, November 1997.

[5] D. Applegate, R. Bixby, V. Chvatal and W. Cook, "On the solution of traveling salesman problems," *Documenta Mathematica*, vol. Extra Volume ICM III, pp. 645-656, 1998.

[6] J. Bater, P. Jeavons and D. Cohen, "Are there optimal resuse distance constraints for FAPs with random Tx placements," Technical Report CSD-98-01, Royal Holloway University of London, 1998.

[7] R. Battiti and G. Tecchiolli, "The reactive tabu search," *ORSA Journal on Computing*, vol. 6, no. 2, pp. 126-140, 1994.

[8] I. Baybars, "Optimal Assignment of broadcasting frequencies," *European Journal of Operations Research*, Vol. 9, pp. 257-263, 1982.

[9] D. A. Behrens, J.P.Caulkins, G. Gernot Tragler, G. Feichtinger, "Optimal Control of Drug Epidemics: Prevent and Treat - But Not at the Same Time," *Management Science*, Vol. 46, No. 3, pp.333-347, 2000.

[10] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Soft-input soft-output modules for the construction and distributed iterative decoding of code networks," *European Trans. Commun*., March/April 1998.

[11] R. E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inform. Theory*, Vol. IT-18, pp. 460-473, July 1972.

[12] R. E. Bland and D. F. Shallcross, "Large traveling salesman problems arising from experiments in X-ray crystallography: A preliminary report on computation," *Operations Research Letters*, vol. 8, pp. 125-128, 1989.

[13] R. Borndörfer, A. Eisenblätter, M. Grötschel and A. Martin, "Frequency Assignment in Cellular Phone Networks," *Annals of Operations Research*, 76, pp. 73-93, 1998.

[14] D. Brelaz, "New methods to color the vertices of a graph," *Communications ACM*, Vol. 22, pp. 251-256, 1979.

[15] S. H. Cameron, "The solution of the graph coloring problem as a set covering problem," *IEEE Transactions on Electromagnetic Compatibility*, Vol. EMC-19, pp. 320-322, 1973.

[16] V. Cerny, "Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm," Journal of Optimization Theory and Applications, vol. 45, pp. 41-51, 1985.

[17] X. Chen and K. M. Chugg, "Near-optimal page detection for two-dimensional ISI/AWGN channels using concatenated modeling and iterative detection," *Proc. International Conf. Communications*, Atlanta, GA, 1998.

[18] E. Chong and S. H. Zak, *An Introduction to Optimization*, Wiley – Interscience Publication, John Wilet and Sons, Inc., 1996.

[19] P. A. Chou, T. Lookabaugh, and R. M. Gray, "Entropy-constrained vector quantization," *IEEE Trans. Acoust. Speech Signal Process.*, pp. 31-42, Jan. 1989.

[20] K M. Chugg and X. Chen, "Efficient Architectures for Soft-Output Algorithms," *Proc. International Conf. Communications*, 1998, Atlanta, GA, June 1998.

[21] K. M. Chugg, A. Anastasopoulos and X. Chen, *Iterative Detection: Adaptivity, Complexity Reduction, and Applications*, Kluwer Academic Press, 2001.

[22] K. M. Chugg, K. Demirciler and A. Ortega "Soft information for source channel coding and decoding," *Proc. of thirty-third International Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, October 1999.

[23] K. M. Chugg, X. Chen, A. Ortega and C-W. Chang, "An Iterative Algorithm for Two-Dimensional Digital Least Metric Problems with Applications to Digital Image Compression," *Proc. International Conf. Image Compression*, Chicago, October 1998.

[24] R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. Inform. Theory*, vol. 38, pp. 713-718, Mar. 1992.

[25] A. Colorni, M. Dorigo, F. Maffioli, V. Maniezzo, G. Righini and M. Trubian, "Heuristics from Nature for hard combinatorial optimization problems," *International transactions in Operational Research*, vol. 3, pp. 1-21, 1996.

[26] J. H. Conway and N. J. A. Sloane, "Soft decoding techniques for codes and lattices, including the Golay code and the Leech lattice," *IEEE Trans. Inform. Theory*, vol. 32, Jan. 1986.

[27] T. H. Cormen, C. E. Leiserson and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill, 1990.

[28] D. Corne and P. Ross, "Practical issues and recent advances in job- and open-shop scheduling," in *Evolutionary Algorithms in Engineering Applications*, (D. Dasgupta and Z. Michalewics, eds.), pp. 531-546, Springer, 1997.

[29] T. M, Cover and J. A Thomas, *Elements of Information Theory*, Wiley, 1991.

[30] I. Csiszár, "On the computation of rate distortion functions," *IEEE Trans. Inform. Theory*, IT-20, pp. 122-124, 1974.

[31] K. Demirciler and A. Ortega, "Image coding based on multiple projections and multistage vector quantization," accepted to *IEEE International Conference on Image Processing, 2003*.

[32] K. Demirciler and A. Ortega, "Reduced Complexity Deterministic Annealing for Vector Quantizer Design," *submitted to Trans. on Image Processing*.

[33] R. Diekmann, R Lüling and J. Simon, "Problem independent distributed simulated annealing and its applications," in *Applied Simulated Annealing*, R. V. Vidal (ed), *Lecture Notes in Economics and Mathematical Systems*, Springer, 1993.

[34] N. Dunkin, J. Bater, P. Jeavons and D. Cohen, "Towards high order constraint representations for the frequency assignment problem," Technical Report CSD-TR-98-05, Royal Holloway University of London, 1998.

[35] M. Duque-Anton and D. Kunz, "Channel assignment based on neural network algorithms," *Proceedings of DMR IV*, pp.541 – 549, 1990.

[36] M. Duque-Anton, D. Kunz and B. Ruber, "Channel assignment for Cellular Radio using Simulated Annealing," *IEEE Trans. on Vehicular Technology*, Vol. 42, No. 1, pp.14 – 21, Feb. 1993.

[37] A. Eisenblätter, "*Frequency assignment in GSM Networks: Models, Heuristics, and Lower Bounds*," Ph.D. thesis, Techniche Universität Berlin, 2001.

[38] A. E. El Gamal, L. A. Hemachandra, I. Shperling and V. K. Wei, "Using simulated annealing to design good codes," *IEEE Trans. Inform. Theory*, vol. 33, pp 116-123, Jan. 1987.

[39] W. H. Equitz, "A new vector quantization clustering algorithm," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 37, pp. 1568-1575, October 1989.

[40] M. Fischetti, J. J. S. Gonzalez and P. Toth, "A branch-and-cut algorithm for the symmetric generalized traveling salesman problem," *Operations Research*, vol. 45, pp. 378-394, 1997.

[41] L. J. Fogel, A. J. Owens and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, New York, John Wiley & Sons, 1966.

[42] J. Frank, "*Local Search For NP-Hard Problems*," Ph.D. thesis, U.C. Davis, 1997.

[43] J. Gallien, "*Optimization-Based Auctions and Stochastic Assembly Replenishment Policies for Industrial Procurement,*" Ph.D. thesis, MIT, 2000.

[44] C. W. Gardiner, *Handbook of Stochastic Methods*, Berlin, Springer, 1983.

[45] M. R. Garey and D. S. Johnson, *Computers and Intractability – A Guide to the Theory of NP-completeness*. W. H. Freeman and Company, 1979.

[46] M. Garey, D. S. Johnson and H. S. Witsenhausen, "The complexity of the generalized Lloyd – Max problem," *IEEE Trans. Inform. Theory*, vol. 28, pp. 255 – 266, March, 1982.

[47] S. Geman and D. Geman, "Sthocastic relaxation, Gibbs distribution, and Bayesian Restoration of Images" *IEEE Trans. Patt. Anal.Mach. Intel.*, Vol. 6, pp. 721 – 741, 1984.

[48] J. S. Gero, V. A. Kazakov and T. Schnier, "Genetic engineering and design problems," in *Evolutionary Algorithms in Engineering Applications*, (D. Dasgupta and Z. Michalewics, eds.), pp. 47-68, Springer, 1997.

[49] A. Gersho, R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer International Series in Engineering and Computer Science, 1992.

[50] K. A. Girodias, H.H. Barrett, R.L. Shoemaker, "Parallel simulated annealing for emission tomography," *Phys. Med. Biol*., vol. 36, no. 7, pp. 921-938, 1991.

[51] F. Glower, "Tabu Search – Part I," *ORSA Journal on Computing*, 1(3):190 – 206, 1989.

[52] F. Glower, "Tabu Search – Part II," *ORSA Journal on Computing*, 2(1):4 – 32, 1990.

[53] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison – Wesley Publishing Company, Inc., 1989.

[54] R. A. H. Gower and R. A. Leese, "The sensitivity of channel assignment to constraint specification," *Proceedings of IEEE International Symposium on Electromagnetic Compatibility*, Zurich, pp. 131-136, 1997.

[55] I. E. Grossmann, (ed.), *Global Optimization in Engineering Design*, Kluwer Academic Publishers, 1996.

[56] W. K. Hale, "Frequency assignment: Theory and Applications," *Proceedings of the IEEE,* Vol. 68, pp. 1497 - 1514, Dec. 1980.

[57] Z. He, C. Wu, J. Wang and C. Zhu, "A New Vector Quantization Algorithm Based on Simulated Annealing," *International Symposium on Speech, Image Processing and Neural Networks*, Hong Kong, 13-16 April, 1994.

[58] M. Hellebrandt and H. Heller, "A new heuristic method for frequency assignment," Technical Report TD(00)003, COST259, Valencia, Spain, 2000.

[59] T. Hofmann and J.M. Buhmann, "Pairwise data clustering by deterministic annealing," *IEEE Trans. Pattern Annal. Machine Intell.*, vol. 19, pp. 1-14, Jan. 1997.

[60] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[61] H. H. Hoos, "*Stochastic Local Search – Methods, Models, Applications*," Ph.D. thesis, Techniche Universität Darmstadt, 1998.

[62] S. Hurley and D. H. Smith, "Fixed spectrum frequency assignment using natural algorithms," *IEE Genetic Algorithms in Engineering Systems*, publication No. 414, Sept. 1995.

[63] P. Jeavons, N. Dunkin and J. Bater, "Why higher order constraints are necessary to model frequency assignment problem," *Proceedings of ECAI 98*, John Wiley & Sons Ltd., 1998.

[64] D. Johnson, C. Aragon, L. McGeoch and C. Schevon, "Optimization by Simulated Annealing: an Experimental Evaluation, Part I, Graph Partitioning," *Operations Research*, vol. 37, pp. 865-892, 1989.

[65] P. K. Johri, "An insight into dynamic channel assignment in cellular mobile communications systems," *European Journal of Operational Research*, 74, pp. 70-77, 1994.

[66] R. L. Joshi, H. Jafarkhani, J. H. Kasner, T. R. Fischer, N. Farvardin, M. W. Marcellin, and R. H. Bamberger, "Comparison of different methods of classification in subband coding of images," *IEEE Trans. Image Processing*, vol. 6, pp. 1473-1486, Nov. 1997.

[67] B. –H. Juang, W. Chou and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 5, pp. 257-265, May 1997.

[68] I. Katzela and M. Naghshineh, "Channel assignment schemes for cellular mobile telecommunication systems," *Personal Communications Magazine*, 1996.

[69] K. J. Kearfott and S.E. Hill, "Simulated annealing image reconstruction method for a pinhole apperture single photon emission computed tomograph (SPECT)," *IEEE Trans. Med. Imaging*, vol. 9, pp. 128-143, 1990.

[70] S. Kirkpatric, C.D. Gellat, Jr., and M.P. Vecchi, "Optimization by simulated annealing," *Science*, Vol. 220, pp. 671-680, 1983.

[71] B. Korte, "Applications of Combinatorial Optimization," *13th International Mathematical Programming Symposium*, Tokyo, 1988.

[72] A. M. C. A. Koster, "*Frequency Assignment - Models and Algorithms*," Ph.D. thesis, Maastricht University, 1999.

[73] J. Krarup and P. M. Pruzan, "Computer-aided layout design," *Mathematical Programming Study*, vol. 9, pp. 75-94, 1978.

[74] V. Kumar, "Approximating circular arc colouring and bandwidth allocation in all-optical ring networks," *Proc. of Approximation Algorithms for Combinatorial Optimization Problems* (APPROX '98), Aalborg, Denmark, July, 1998.

[75] P. J. M. van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, Boston, D. Reidel Publishing, 1987.

[76] J. G. Lauprete, "*Portfolio Risk Minimization under Departures from Normality,*" Ph.D. thesis, MIT, 2001.

[77] A. P. Leclerc, "*Efficient and Reliable Global Optimization*," Ph.D. thesis, Ohio State University, 1992.

[78] R. A. Leese, "Radio Spectrum," Report for telecommunications industry, *University of Oxford UK*, (http://www.maths.ox.ac.uk/~leese/).

[79] J. K. Lenstra and A. H. G. R. Kan, "Some simple applications of the traveling salesman problem," *Opl. Res.*, vol. 26, pp. 717–733, 1975.

[80] Y. Linde, A. Buzo and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, vol. COM-28, pp. 84-95, Jan. 1980.

[81] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. 28, pp.129-137, Mar. 1982.

[82] C. Loader, "*Local Search Algorithms for Geometric Object Recognition*," Ph.D. thesis, University of Western Australia,1995.

[83] N. J. C. Lous, P. A. H. Bours and H. C. A. van Tilborg, "On maximum likelihood soft-decision decoding of binary linear block codes," *IEEE Trans. Inform. Theory*, vol. 39, Jan. 1993.

[84] A. Lucena and J. E. Beasley, "A branch-and-cut algorithm for the Steiner problem in graphs," *Networks: An International Journal*, vol. 31, pp. 39-59, 1998.

[85] D. G. Luenberger, *Linear and Nonlinear Programming*, Addison – Wesley Publishing Company, 1984.

[86] E. Malesinska, "*Graph-Theoretical Models for Frequency Assignment Problem*," Ph.D. thesis, Techniche Universität Berlin, 1997.

[87] D. W. Matula and L.L. Beck, "Smallest-last ordering and clustering and graph coloring algorithms," *Journal of the ACM*, Vol. 30, pp. 417-427, 1983.

[88] P. Merz, "*Memetic Algorithms for Combinatorial Optimization Problems*," Ph.D. thesis, University of Seigen, 2000.

[89] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys*, Vol. 21, pp. 1087 – 1092, 1953.

[90] B. H. Metzger, "Spectrum management technique," *38th National ORSA Meeting*, Detroit, MI, 1970.

[91] D. Miller and K. Rose, "Combined source-channel ve4ctor quantization using deterministic annealing," *IEEE Trans. Commun.*, vol. 42, pp. 347-356, Feb. 1994.

[92] D. Miller, A. Rao, K. Rose and A. Gersho, "A global optimization technique for statistical classifier design," *IEEE Trans. Signal Processing*, vol. 44, pp 3108-3122, Dec. 1996.

[93] E. S. Mistakidis and G. E. Stavroulakis (eds.), *Nonconvex Optimization in Mechanics. Algorithms, Heuristics and Engineering Applications by the F.E.M.*, Kluwer Academic Publishers, 1998.

[94] U. Moller, M. Galicki, E. Baresova and H. Witte, "An efficient Vector Quantizer Providing Globally Optimal Solutions," *IEEE Trans. on Signal Processing*, vol. 46, No. 9, Sept. 1998.

[95] A. Ortega and K. Ramchandran, "Rate-Distortion Techniques in Image and Video Compression," *IEEE Signal Processing Magazine*, pp. 23-50, Vol. 15, No. 6, Nov. 1998.

[96] G. Ottosson and M. Carlsson, "Using global constraints for frequency allocation," Technical Report TR-97-07, ASTEC, 1997.

[97] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization*, Dover Publications, Inc., 1998.

[98] A. Pothen, "Graph partitioning algorithms with applications to scientific computing," *Tech. Rep*. TR-97-03, Old Dominion University, 1997.

[99] A. Quellmalz, A. Knälmann and B. Müller, "Efficient frequency assignment with simulated annealing," *IEE Antennas and Propagation Conference*, No. 407-2, pp. 301-304, 1995.

[100] L. A. Rastrigin, *Random Search in Optimization Problems for Multiparameter Systems*, Riga Latvia. "Zinatne" Publishing House, 1965.

[101] G. Reinelt, *The Travelling Salesman: Computational Solutions for TSP Applications*, vol. 840, Lecture Notes in Computer Science, Springer–Verlag, Berlin, Germany, 1994.

[102] A. Roa, D. Miller, K. Rose and A. Gersho, "Mixture of experts regression modeling by deterministic annealing," *IEEE Trans. Signal Processing*, vol. 45, pp 2811-2820, Nov. 1997.

[103] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proc. IEEE*, vol. 86, no. 11, Nov. 1998.

[104] K. Rose, E. Gurewitz and G. C. Fox, "Statistical mechanics and phase transitions in clustering," *Phys. Rev. Lett*. Vol. 65, no. 8, pp. 945-948, 1990.

[105] K. Rose, E. Gurewitz, and G. C. Fox, "Constrained clustering as an optimization method," *IEEE Trans. Pattern Annal. Machine Intell.*, vol. 15, pp. 785-794, Aug. 1993.

[106] K. Rose, E. Gurewitz, and G. C. Fox, "Vector Quantization by Deterministic Annealing," *Trans. on Inform. Theory*, vol. 38, no. 4, pp. 1249-1257, July 1992.

[107] M. Schaffter, "*Scheduling Jobs with Communication Delays: Complexity Results and Approximation Algorithms,"* Ph.D. thesis, Techniche Universität Berlin, 1996.

[108] Y. Shang, "*Global Search Methods for Solving Nonlinear Optimization Problems*," Ph.D. thesis, University of Illinois at Urbana-Champaign, 1997.

[109] E. Simoncelli and J. Portilla, "Texture characterization via joint statistics of wavelet coefficient magnitudes," *Proc. 5th IEEE Int. Conf. Image Processing*, vol. I, pp. 4-7, Chicago, IL, Oct. 1998.

[110] E. Taillard, "Tabu Search," in *Local Search in Combinatorial Optimization*, (E. H. L. Aarts and J. K. Lenstra, eds.), ch. 1, pp, 1 – 17, John Wiley and Sons, 1997.

[111] J. Vaisey and A. Gersho, "Simulated annealing and codebook design," pp. 1176-1179, Proc. IEEE Intl. Conf. On Acoustics, Speech, and Signal Processing (ICASSP), New York, April, 1988.

[112] A. Vardy and Y. Bèery, "Bit level soft-decision decoding of Reed-Solomon codes," *IEEE Trans. Comm. Theory*, vol. 37, no 3, May 1991.

[113] A. Vardy and Y. Bèery, "Maximum likelihood soft-decision decoding of BCH codes," *IEEE Trans. Inform. Theory*, vol. 40, Mar. 1994.

[114] S. –W. Wang and S. S. Rappaport, "Signal-to-interference calculations for balanced channel assignment patterns in cellular communications systems," *IEEE Trans. Communications*, vol. 37, pp. 1077-1087, 1989.

[115] D. D. Werra and Y. Gay, "Chromatic Scheduling and frequency assignment," *Discrete Applied Math.*, 49, 1-3, pp. 165 - 174, 1994.

[116] A. Wisse, "Mathematical model for the radio link frequency assignment problem," *EUCLID CALMA Project*, Delft Univ. of Technology, The Netherlands, 1994.

[117] K. Zeger, J. Vaisey and A. Gersho "Globally Optimal Vector quantizer design by Stochastic Relaxation," *IEEE Trans. on Signal Processing*, vol. 40, No.2, Feb. 1992.

[118] J. A. Zoellner and C. L. Beall, "A breakthrough in spectrum conserving frequency assignment thechnology," *IEEE Trans. on Electromahnetic Compatibility*, Vol. EMC-19, pp.313 – 319, 1977.

Appendix

In Chapter 2 we proposed two low complexity soft assignment measures, the triangular soft information measure and the multi-triangular soft information measure, as simplified ways of computing the soft assignments for the VQ design problem using deterministic annealing. Although these measures significantly reduce the computational cost of the soft assignments compared to the optimal Gibbs soft measure, this improvement in computational cost comes in exchange for some loss in performance since Gibbs is the optimal soft measure. In this section we derive the penalty paid in distortion for using the simplified soft measures instead of the optimal one at a given system entropy (softness).

For a given soft assignment measure (conditional probability), $p(c|x)$ we have the expected distortion,

$$D\left(p\left(c|x\right)\right)=\sum_{x}p(x)\sum_{c}p\left(c|x\right)d\left(x,c\right), \tag{1}$$

and the average mutual information,

$$I\left(X;C\right)=\sum_{x}p(x)\sum_{c}\ p\left(c|x\right)\log\frac{p\left(c|x\right)}{p(c)}. \tag{2}$$

Let P_I be the set of all *I-admissible* soft assignment measures,

$$P_I=\left\{p\left(c|x\right)\ :\ I\left(X;C\right)\le I\right\}, \tag{3}$$

and hence, for fixed I,

$$D(R)=\min_{p(c|x)\in P_I}D\left(p\left(c|x\right)\right), \tag{4}$$

where $R(D) = \min_{p(c|x) \in P_I} I(X;C)$. Now, let $p_G(c|x)$ and $p_0(c|x)$ be two different soft assignment measures, and assume that $p_G(c|x)$ is the optimal *I-admissible* soft assignment measure for some rate *I*, $p_G(c|x) \in P_I$, and the expected distortion corresponding to $p_G(c|x)$ is $D(p_G(c|x))$. Let the other soft assignment measure, $p_0(c|x)$ to be defined as,

$$p_0(c|x) = p_G(c|x) + \Delta p(c|x) \quad \forall c, x . \tag{5}$$

We require two conditions to be satisfied by (5):

$$\sum_c \Delta p(c|x) = 0 \qquad \forall x , \tag{6}$$

and

$$\Delta I = I_0(X;C) - I_G(X;C) = 0 . \tag{7}$$

The condition in (6) is required so that $p_0(c|x)$ in (5) is a valid pmf, $\sum_c p_0(c|x) = 1$, and the condition in (7), the difference in the mutual information to be zero, is required so that $p_0(c|x) \in P_I$. We would like to obtain the difference in the expected distortion, $\Delta D = D(p_0(c|x)) - D(p_G(c|x))$ subject to the conditions (6) and (7). The situation is depicted in Figure A.1, and a real simulation result is shown in Figure A.2.

We will start by expanding, $\Delta I = I_0(X;C) - I_G(X;C)$,

$$\Delta I = \sum_x p(x) \sum_c p_0(c|x) \log \frac{p_0(c|x)}{p_0(c)} - \sum_x p(x) \sum_c p_G(c|x) \log \frac{p_G(c|x)}{p_G(c)} . \tag{8}$$

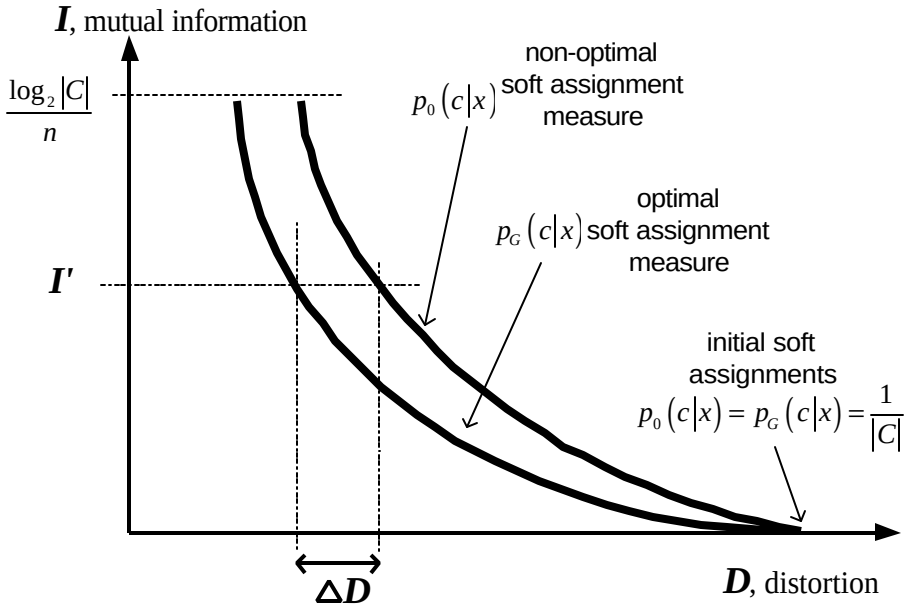


Figure A.1: Convergence of the optimal and a non-optimal soft assignment measures (distributions). Starting with equal, uniform soft assignments, the optimal soft assignment measure achieves a lower distortion than the non-optimal soft measure. At a given system entropy level, I' the difference in distortion is shown as ΔD. The term n is the vector dimension and |C| is the size of the codebook.

Substitute $p_G(c|x) = p_0(c|x) - \Delta p(c|x)$ from (5) into (8):

$$\Delta I = \sum_x p(x) \sum_c p_0(c|x) \log \frac{p_0(c|x)}{p_0(c)} - \sum_x p(x) \sum_c \left(p_0(c|x) - \Delta p(c|x) \right) \log \frac{p_G(c|x)}{p_G(c)}$$

$$= \sum_x p(x) \sum_c \Delta p(c|x) \log \frac{p_G(c|x)}{p_G(c)} + \sum_x p(x) \sum_c \left[p_0(c|x) \log \frac{p_0(c|x)}{p_G(c|x)} - p_0(c|x) \log \frac{p_0(c)}{p_G(c)} \right]$$

Simplifying the above expression, we get,

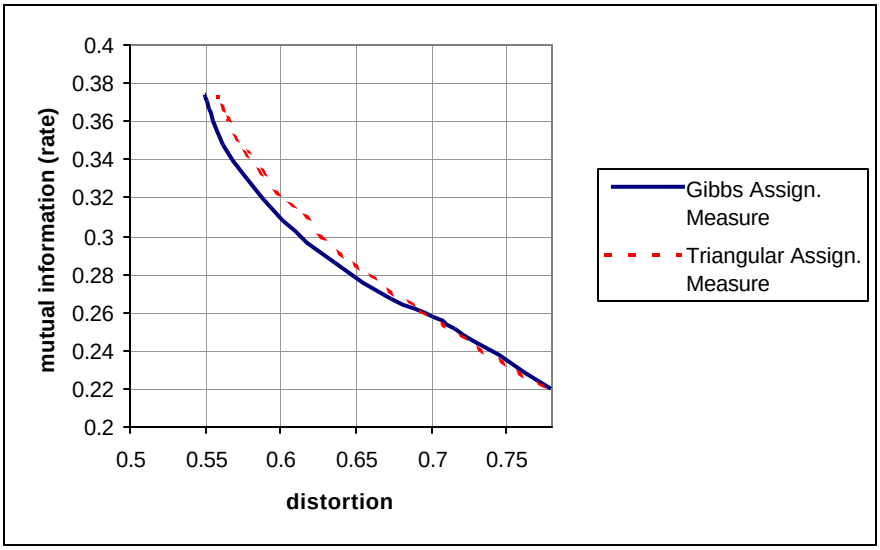


Figure A.2: Plot showing the convergence of two experiments, one with Gibbs soft assignment measure and the other with Triangular soft assignment measure, for the design of a codebook of size 64, vector dimension 16, and the source type zero-mean, unit variance Gaussian.

$$\Delta I = \sum_x p(x) \sum_c \Delta p(c|x) \log \frac{p_G(c|x)}{p_G(c)} + D\left(p_0(c|x) \| p_G(c|x)\right) - D\left(p_0(c) \| p_G(c)\right). \qquad (9)$$

The optimal distribution is the Gibbs distribution, $p_G(c|x) = \frac{e^{-b\,d(x,c)}}{\sum_{c'} e^{-b\,d(x,c')}}$, where

$p_G(c)$ is uniform. Let $p_G(c) = z$, a constant $\forall c$, and substitute the Gibbs distribution in the first term in (9),

$$\sum_x p(x) \sum_c \Delta p(c|x) \log \frac{p_G(c|x)}{p_G(c)}$$

$$= \sum_x p(x) \sum_c \Delta p(c|x) \log \left(\frac{1}{p_G(c)} \cdot \frac{e^{-b\,d(x,c)}}{\sum_{c'} e^{-b\,d(x,c')}} \right)$$

$$= \sum_x p(x) \sum_c \Delta p(c|x) \log \left(\frac{e^{-b\, d(x,c)}}{z \cdot \sum_{c'} e^{-b\, d(x,c')}} \right)$$

$$= \sum_x p(x) \sum_c \Delta p(c|x) \left(-\frac{b\, d(x,c)}{\ln 2} - \log \sum_{c'} z \cdot e^{-b\, d(x,c')} \right).$$

Therefore,

$$\sum_x p(x) \sum_c \Delta p(c|x) \log \frac{p_G(c|x)}{p_G(c)}$$

$$= \frac{-b}{\ln 2} \underbrace{\sum_x p(x) \sum_c \Delta p(c|x)\, d(x,c)}_{=\Delta D} - \sum_x p(x) \left(\log \sum_{c'} z \cdot e^{-b\, d(x,c')} \right) \underbrace{\sum_c \Delta p(c|x)}_{=0 \ \forall x} \qquad (10)$$

$$= \frac{-b}{\ln 2} \Delta D.$$

Substituting (10) into(9), and using the condition in (7) that $\Delta I = 0$, we get,

$$\Delta I = \frac{-b}{\ln 2} \Delta D + D\left(p_0(c|x) \| p_G(c|x) \right) - D\left(p_0(c) \| p_G(c) \right) = 0. \qquad (11)$$

Finally, the difference in the expected distortion is,

$$\Delta D = \frac{\ln 2}{b} \left[D\left(p_0(c|x) \| p_G(c|x) \right) - D\left(p_0(c) \| p_G(c) \right) \right]. \qquad (12)$$

Note that for large vector dimensions [106], $-D\left(p_0(c) \| p_G(c) \right) \cong D\left(p_G(c) \| p_0(c) \right)$.

Hence, for large dimensions the penalty paid in terms of distortion at a given system entropy for using the non-optimal soft assignment measure, $p_0(c|x)$ instead of the optimal one, $p_G(c|x)$, is,

$$\Delta D = \frac{\ln 2}{b} \left[D\left(p_G(c) \| p_0(c) \right) + D\left(p_0(c|x) \| p_G(c|x) \right) \right]. \qquad (13)$$

Note also that $p_G(c)$ is uniform and $p_0(c)$ depends on $p_0(c|x)$. Hence, minimizing the conditional relative entropy, $D\left(p_0(c|x)\middle\|p_G(c|x)\right)$ in (13) minimizes ΔD.