On the Performance of Optimisation Networks

by

BRENTON S. COOPER, B.E. (HONS. I)

Thesis submitted for the degree of

Doctor of Philosophy

The University of Adelaide
Faculty of Engineering
Department of Electrical and Electronic Engineering

December, 1996

Table of Contents

Abstract

Optimisation networks are a family of approximate techniques for the solution of combinatorial optimisation problems. Typically an optimisation network comprises a large number of highly interconnected, simple processors which may be programmed to compute a solution to a variety of combinatorial optimisation problems. When implemented in hardware, optimisation networks like the Hopfield network and the mean field annealing algorithm are capable of producing solutions in a matter of milliseconds. The capability for real-time combinatorial optimisation is unique to optimisation networks.

As optimisation networks are not a mature technique, there remain several open questions relating to their use. One such issue, which has gone largely unnoticed in the literature, is that of solution quality. Previously, various annealing techniques have been developed to improve the quality of solutions obtained from an optimisation network. Our investigations of these annealing techniques confirm their ability to improve solution quality, but uncover the tendency to force the network toward invalid states. Consequently we develop a new, principled approach to annealing that improves solution quality while maintaining a valid network state.

Simple experiments reveal an alarming feature of the performance of optimisation networks: as the problem size increases, the quality of the solutions rapidly decreases. Such deteriorating performance would restrict optimisation networks to the solution of only small problems – a significant erosion of the niche market for optimisation networks. Our investigations show that the simple heuristics that optimisation networks employ to solve a combinatorial optimisation problem are ultimately responsible for the poor scaling to large problems. Furthermore, while alternative heuristics can be shown to far outperform those used by an optimisation network, many of them are not suitable for embedding into an optimisation network. However, after recognising that for any heuristic there is a trade-off between solution quality and computational effort, we developed a family of higher-order neural networks (HONNs) which embody that trade-off. It is shown that HONNs offer improved solution quality at the expense of extra complexity in the network. It is concluded that optimisation networks embody a simple heuristic approach to the solution of combinatorial optimisation problems, and as with any heuristic approach trade-offs must be made between solution quality and computational effort in order to meet demands on their performance.

Keywords: combinatorial optimisation, neural networks, Hopfield network, higher-order neural networks.

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma in any University or other tertiary institution, and to the best of the author's knowledge and belief contains no material previously published or written by another person, except where due reference has been made in the text.

Should the thesis be accepted for the award of the Degree, the author hereby consents to this copy, when deposited in the University Library, being made available for loan and photocopying.

Signature

Date 10/7/97

Acknowledgments

I gratefully acknowledge the support and encouragement of my supervisor, Professor Robert ("Bob") E. Bogner. I appreciate the freedom that he gave me to pursue my own ideas, and the invaluable advice that he offered as my work progressed.

My thanks go also to Dr Kenneth Pope for many helpful discussions, and for the assistance that he has so readily given throughout the course of my studies.

During my studies I was the recipient of a Telstra Research Laboratories (TRL) Postgraduate Fellowship. I would sincerely like to thank TRL for their support.

This work was supported by an Australian Postgraduate Award from the Australian Government. Support was also provided by the Cooperative Research Centre for Sensor Signal and Information Processing (CSSIP), The Levels, South Australia.

Finally, I would like to thank my parents for their support and encouragement, and for giving me the opportunity to achieve what I have.

B.S.C.

List of Figures

List of Tables

Publications

Cooper, B. S. (1994). Selected applications of neural networks in telecommunications systems – A review. *Australian Telecommunications Research Journal*, 28(2):9-29.

Cooper, B. S. (1995). Higher-order neural networks – Can they help us optimise? In *Proceedings of the Sixth Australian Conference on Neural Networks*, pages 29-32.

Cooper, B. S. (1995). Higher-order neural networks for combinatorial optimisation – Improving the scaling properties of the Hopfield network. In *Proceedings of the International Conference on Neural Networks*, volume 4, pages 1855-1860.

Cooper, B. S. (1996). A principled approach to annealing in optimisation networks. In Narasimhan, V. and Jain, L., editors, *Proceedings of the Australian and New Zealand Conference on Intelligent Information Systems*, pages 35-38.

Cooper, B. S. (1996). On penalty functions and annealing for optimisation networks. *Technical Report CSSIP TR 2/96, Cooperative Research Centre for Sensor Signal and Information Processing.* Also appears as *Technical Report SIG 1/96, Department of Electrical and Electronic Engineering, University of Adelaide.*

Cooper, B. S. (1996). A comparison of the number of stable points of optimisation networks. *Technical Report CSSIP TR 4/96, Cooperative Research Centre for Sensor Signal and Information Processing.* Also appears as *Technical Report SIG 2/96, Department of Electrical and Electronic Engineering, University of Adelaide.*

Glossary

General Notation

$\mathbf{A}, \mathbf{B}, \mathbf{C} \ldots$	Matrices are denoted by bold uppercase typeface
$\mathbf{a}, \mathbf{b}, \mathbf{c} \ldots$	Vectors are denoted by bold lowercase typeface
$[\mathbf{A}]_{i,j}$	Element i, j of matrix $\mathbf{A}$
b_i or $[\mathbf{b}]_i$	Element i of vector $\mathbf{b}$
$\|\|\mathbf{a}\|\|$	Euclidean norm of vector $\mathbf{a}$
$\mathcal{A}, \mathcal{B}, \mathcal{C} \ldots$	Sets are denoted by caligraphic typeface, except as noted below
$\|\|\mathcal{A}\|\|$	Dimensionality of set $\mathcal{A}$
$\mathbb{R}^N, \mathbb{I}^N$	The sets of real and imaginary N-dimensional vectors
$(a, b]$	The set of real numbers x: $a < x \leq b$
$\{a, b, c\}$	The set with elements a, b and c
$\langle a, b, c \rangle$	The unordered set with elements a, b and c
$[a, b, c]^T$	The column vector with elements a, b and c
$\int_{\mathcal{A}} \cdot \, d\mathbf{x}$	Shorthand for $\int \ldots \int_{\mathcal{A}} \cdot \, dx_1 \ldots dx_n$
$\Pr(\cdot)$	Probability operator
$\|x\|$	Absolute value of scalar x

Optimisation Networks

E^{lyap}	Lyapunov function
$\mathbf{T}$	Interconnection matrix
$\mathbf{b}$	External bias
η	Decay parameter
$\mathbf{v}$	Continuous state vector $v_i \in [0, 1] \ \forall i$
$\mathbf{u}$	Internal state vector
$\mathbf{s}$	Discrete state vector $s_i \in \{0, 1\} \ \forall i$
$g(u_i)$	Neuron transfer function
T^p	Pseudo-temperature
$\mathbf{T}^{cns}$	Interconnection matrix when $E^{lyap} = E^{cns}$ (page 31)
$\mathbf{b}^{cns}$	External bias when $E^{lyap} = E^{cns}$ (page 31)
$\mathbf{T}^{ann}$	Interconnection matrix when $E^{lyap} = E^{cns} + E^{ann}$ (page 42)
$\mathbf{b}^{ann}$	External bias when $E^{lyap} = E^{cns} + E^{ann}$ (page 42)
$\mathbf{T}^{mod}$	Interconnection matrix when $E^{lyap} = E^{cns} + E^{mod}$ (page 48)
$\mathbf{b}^{mod}$	External bias when $E^{lyap} = E^{cns} + E^{mod}$ (page 48)

Problem Mapping

E^{ann}	Standard hysteretic annealing function (page 39)
E^{mod}	Modified hysteretic annealing function (page 47)
γ	Annealing parameter (page 39)
E^{obj}	Objective function (page 24)
E^{cns}	Penalty function (page 24)
c	Weight on the penalty function (page 28)
$\hat{\mathbf{e}}$	Valid subspace offset vector (page 26)
$\mathbf{v}^{zs}$	Component of $\mathbf{v}$ in the zerosum subspace (page 26)
$\mathbf{v}^{1}$	Component of $\mathbf{v}$ in the direction of $\hat{\mathbf{e}}$ (page 28)
$\mathbf{v}^{inv}$	Component of $\mathbf{v}$ in the invalid subspace (page 28)
$\mathbf{T}^{zs}$	Zerosum subspace projection matrix (page 28)

Acronyms

ANN	Artificial Neural Network
CAM	Content Addressable Memory
HONN	Higher-Order Neural Network
HPP	Shortest Hamilton Path Problem
LK	Lin-Kernighan heuristic
MFA	Mean Field Annealing
NI	Node Insertion
NN	Nearest Neighbour
SA	Simulated Annealing
TSP	Travelling Salesman Problem
2-OPT	2-OPT heuristic for the TSP
3-OPT	3-OPT heuristic for the TSP
1D, 2D, 3D	One/Two/Three Dimension(al)
MATLAB®	Mathematical Simulation Environment

Specialist Terms

$0-1$ point	Vector where all elements are 0 or 1
valid solution	A $0-1$ point that satisfies the problem's constraints
invalid solution	A $0-1$ point that does not satisfy the problem's constraints
solution quality	Measure of the cost of a solution relative to the cost of the optimal solution

Chapter I

Introduction

1.1 Optimisation Networks

Optimisation networks belong to a broad class of systems known as artificial neural networks (ANNs), which have been developed to address the shortcomings of conventional computers. Conventional computers operate in a serial manner by performing a programmed sequence of instructions. They excel at numerically intensive tasks, but are significantly less effective at tasks which humans can do routinely, such as recognising objects and understanding speech. Given that the human brain performs so well on these complex tasks, there is considerable merit in attempting to develop an artificial neural network that mimics the computational processes of the brain.

Artificial neural networks (Hecht-Nielsen, 1989; Hertz et al., 1991; Zurada, 1992) generally comprise a large number of simple processors which are analogous to biological neurons. These processors are connected together via a dense network of interconnections, which perform a similar function to the synapses and axons in a human brain. ANNs exhibit a computational ability which arises not from the complexity of the individual processor, but from the high degree of connectivity between the processors. Just like the human brain, ANNs have been trained to perform a variety of tasks and have the ability to learn from experience. This model for computation is in stark contrast with conventional serial computers and consequently ANNs have been applied to many of the problems that have proven to be too difficult for conventional computers.

Optimisation networks are a subset of ANNs where the outputs of the individual processing units are fed-back, via the network of interconnections, to the inputs of the processors. By interconnecting the processors in this manner a nonlinear, dynamic system is produced, which under certain conditions is ideally suited to the solution of optimisation problems. Specifically, optimisation networks are used to solve combinatorial optimisation problems, the most famous example of which is the travelling salesman problem (TSP). The TSP may be simply stated as:

> *Given N cities in a plane, all of which must be visited only once, find the order in which to visit them, making sure to return to the initial city, so that the total distance travelled is minimised.*

The TSP, as with all combinatorial optimisation problems, is characterised by having a finite set, as opposed to a continuum, of possible solutions over which the objective function must be optimised. Rather than making the problem easier, it makes problems like the travelling salesman notoriously difficult. In fact, combinatorial optimisation problems are amongst the most difficult problems known to the mathematical community. To date, there is no known method for efficiently finding the optimal solution to a travelling salesman problem. This is not surprising when we consider that for a ten city travelling salesman problem, there are slightly under two hundred thousand tours to consider; yet if we increase the size of the problem to just twenty cities, there are approximately sixty million billion tours to evaluate in order to find the optimal solution.

Notwithstanding the complexity of combinatorial optimisation problems, experiments have repeatedly shown that optimisation networks can produce reasonably good solutions to problems like the TSP. The Hopfield network (Hopfield, 1984) is typical of most optimisation networks. Its ability to solve the TSP was first demonstrated by Hopfield and Tank (Hopfield and Tank, 1985). However, the initial results from the Hopfield network were far from convincing. The network regularly converged to states that did not satisfy the constraints of the problem. Furthermore, even when a valid solution was obtained, it was quite likely to be of poor quality. Despite these shortcomings, it was quite obvious that the Hopfield network was widely applicable and was a significant, new approach to the solution of combinatorial optimisation problems. Subsequently, a great deal of research into optimisation networks has seen many of their early limitations overcome.

One of the most significant factors to commend the use of optimisation networks is their amenability to implementation in fast, parallel hardware. Such hardware implementations could yield solutions in a matter of milliseconds, making the solution of real-time combinatorial optimisation problems an attainable goal. Moreover, as combinatorial optimisation problems arise in many areas of engineering and science, there is a significant array of suitable real-time problems: routing traffic in communications networks (Cooper, 1994), solving graph labelling problems for object recognition (Gee, 1993) and optimisation problems in low-level vision (Mjolsness et al., 1991) are just a few. Given the potential for optimisation networks to solve real-time problems and the many applications that would benefit, it appears that further research into optimisation networks is justified.

1.2 Themes and Contributions

To date, the vast majority of research into optimisation networks has concentrated on developing methods that improve the reliability with which networks converge to valid solutions. An additional thrust of the research has been to apply optimisation networks to more practical problems than the travelling salesman. While there have been some attempts at developing mechanisms to improve solution quality, there has been virtually no research investigating the existence of limits on the solution quality achievable with optimisation networks, or indeed the factors responsible for such limits, should they exist. The one theme that underpins all our investigations in this thesis is that of solution quality.

The main contributions of this thesis may be summarised as follows:

- We demonstrate the importance of annealing mechanisms, which aim to improve solution quality (Section 4.2). Furthermore we investigate the effect that annealing may have on the ability to obtain valid solutions (Section 4.3).
- As a result of our investigations we find it necessary to develop a new, principled approach to annealing that improves solution quality while maintaining a valid network state (Sections 4.4 and 4.5).
- We investigate the ability of optimisation networks to scale to large problem sizes, and the impact that problem size may have on solution quality (Sections 5.2 and 5.3).

- We establish that optimisation networks utilise simple heuristic methods for the solution of combinatorial optimisation problems (Sections 5.2.1 and 6.2). As with any heuristic method there is a trade-off between computational effort and solution quality.

- Following these investigations, we develop a family of alternative optimisation networks, which embody the trade-offs that can be made between solution quality and computational effort (Sections 5.4.2 and 6.3).

- We investigate the impact that the trade-off between computational effort and solution quality has on the performance of standard optimisation networks (Sections 7.3 and 7.4).

Throughout the thesis we have avoided large-scale simulations, as the underlying problems and the methods developed to counter them, are easily demonstrated on simple examples. While detailed mathematics is included where necessary, we have made as much use of simple intuitive arguments as possible. The broad outline of this thesis is as follows:

Chapter 2 presents an overview of various optimisation networks, including their method of operation, implementation in hardware and brief discussion of their *niche market.*

Chapter 3 discusses the process of *mapping* a problem onto an optimisation network. A rigorous analysis of the proposed mapping technique shows that a valid solution to problems such as the TSP can always be obtained from an optimisation network.

Chapter 4 investigates the use of *annealing* techniques to remove dependence on initial conditions and improve the quality of solutions. A potential conflict between the problem mapping and annealing process is first identified and then resolved with the development of the *modified hysteretic annealing* technique.

Chapter 5 demonstrates that optimisation networks employ simple heuristics to solve combinatorial optimisation problems. Experimental results show that the solution quality obtained from an optimisation network deteriorates quickly as the problem size increases. The poor scaling of optimisation networks to large problem sizes is directly attributable to the simple heuristics used. Various approaches to improving the scaling of optimisation networks are presented.

Chapter 6 develops a family of optimisation networks called higher-order neural networks (HONNs). HONNs make use of stronger heuristics in the solution of combinatorial optimisation problems and in many ways they embody the trade-off between computational effort and solution quality that may be made with any heuristic approach to optimisation.

Chapter 7 further investigates the computational effort versus solution quality trade-off by presenting a theoretical analysis of the performance of higher-order neural networks.

Chapter 8 presents the main conclusions of the thesis.

Appendices contain derivations and experimental details that are supplementary to the main text.

CHAPTER II

Optimisation Networks

Optimisation networks are a class of ANNs that may be applied to the solution of combinatorial optimisation problems. An optimisation networks comprises a large number of highly interconnected processing units. The outputs of the processing units are fed-back via the network of interconnections to their inputs. In doing so, a non-linear dynamic system is created that can be used to solve optimisation problems. It should be noted that optimisation networks do not attempt to implement any sort of cognitive function or indeed perform any learning: a fact that sets them apart from more conventional notions of what comprises an ANN.

In this chapter we present an overview of various optimisation networks and give a series of simple examples to demonstrate their use. We begin in Section 2.1 by briefly describing how combinatorial optimisation problems may be mapped into a form which is suitable for optimisation networks. Then in Section 2.2 we introduce the discrete Hopfield network and discuss its limitations when applied to combinatorial optimisation. The limitations of the discrete Hopfield network naturally lead to the development of simulated annealing and the mean field annealing algorithms, as detailed in Sections 2.3 and 2.4. To complete our coverage of optimisation networks, Section 2.5 introduces the continuous Hopfield network. All optimisation networks rely on hardware implementations to be competitive with alternative techniques for combinatorial optimisation. Accordingly, Section 2.6 presents a brief review of hardware implementations of optimisation networks. Finally, in Section 2.7 the niche market for optimisation networks is identified by contrasting their strengths and weaknesses with the performance of competing techniques for combinatorial optimisation.

2.1 Mapping combinatorial optimisation problems

Typically, combinatorial optimisation problems require the minimisation of an objective function, subject to a set of constraints, over a set of $0-1$ variables[1]. In order to solve such a problem with an optimisation network it is necessary to reformulate the problem. This process is called *mapping the problem* onto the network and is the subject of more detailed comment in Chapter 3. In many cases, the outcome of the problem mapping is to express the original optimisation problem as the minimisation of a single quadratic objective function over a set of $0-1$ variables *i.e.*

$$\begin{array}{rl} \text{minimise} & E(\mathbf{s}) = -\frac{1}{2}\mathbf{s}^T\mathbf{T}\mathbf{s} - \mathbf{s}^T\mathbf{b} \\ \text{where} & s_i \in \{0,1\}. \end{array} \tag{2.1}$$

Many of the problems that may be represented by such a mapping belong to the class of

[1] s_i is a $0-1$ variable if $s_i \in \{0,1\}$. The vector $\mathbf{s}$ is a $0-1$ point if $s_i \in \{0,1\}\,\forall i$.

$\mathcal{NP}$-hard problems. It is commonly accepted that it is impossible to find an algorithm that can, for every problem instance, produce the optimal solution to an $\mathcal{NP}$-hard problem in an amount of time bounded by a polynomial function of the problem size (Garey and Johnson, 1979; Papadimitirou and Steiglitz, 1982). Consequently, most algorithms abandon the search for an optimal solution, preferring to find a good, though not necessarily optimal solution in a reasonable amount of time. Such algorithms, may therefore be referred to as *approximate solution* techniques.

Optimisation networks are a class of approximate solution techniques that attempt to solve combinatorial optimisation problems by using the problem mapping given in equation (2.1). As we have previously stated, an optimisation network is a non-linear dynamic system. The state vector of the system is given by the outputs of the processing units in the network, and is constrained to lie within the unit hypercube. The network's dynamics, which determine how the state vector moves through the state space, are constructed so as to guide the state vector through a gradient descent on the objective function E. When the system converges to a stable state, and if measures have been taken to ensure that that state is a $0-1$ point, then the stable state may be interpreted as a solution to the optimisation problem of equation (2.1). Moreover, as the state was arrived at via a process of gradient descent on the objective function, it is expected that such stable states should represent good solutions to the optimisation problem. In the rest of this chapter, various optimisation networks are introduced and their utility at the solution of combinatorial optimisation problems is discussed.

2.2 Discrete Hopfield Network

The discrete state Hopfield neural network (Hopfield, 1982) is constructed by interconnecting a large number of simple processing units. The i^{th} processing unit, or *neuron*, is described by two variables: its internal state u_i and its output s_i. In the discrete state network each neuron may either be firing or non-firing, as represented by its output $s_i = \pm 1$. The strength of the synaptic connection from the output of neuron j to the input of neuron i is given by T_{ij}. Each neuron also has an external bias b_i applied at its input. The operation of the network depends on the choice of the update rule.

2.2.1 Deterministic Update

The Hopfield network, as it was originally introduced[2] (Hopfield, 1982), operates with a deterministic update rule

$$u_i = \sum_{j \neq i} T_{ij} s_j + b_i \tag{2.2}$$

$$s_i = \operatorname{sign}(u_i) \tag{2.3}$$

where $\operatorname{sign}(u_i) = \begin{cases} +1 & \text{if } u_i \geq 0 \\ -1 & \text{otherwise.} \end{cases}$

If the connections are symmetric *i.e.* $T_{ij} = T_{ji}$, and the neurons are updated asynchronously according to equations (2.2) and (2.3), then the network admits a Lyapunov function of the form (Hopfield, 1982)

[2]An equivalent, alternative formulation of the discrete state Hopfield network may use $s_i = \{0, 1\}$.

$$E^{lyap}(\mathbf{s}) = -\frac{1}{2}\sum_i \sum_{j \neq i} T_{ij} s_i s_j - \sum_i b_i s_i. \tag{2.4}$$

The existence of a Lyapunov function is significant, as it guarantees that as the neurons are updated, $E^{lyap}(\mathbf{s})$ is monotonically non-increasing (Hopfield, 1982). Since $E^{lyap}(\mathbf{s})$ is also bounded, state changes will continue until $\mathbf{s}$ reaches a local minimum of $E^{lyap}(\mathbf{s})$. Any such local minimum will be a stable state of the discrete Hopfield network.

Given the quadratic form of the Lyapunov function (2.4), it would at first seem that the discrete Hopfield network is well suited to the solution of optimisation problems posed in the form of equation (2.1). However, the utility of the discrete Hopfield network is dependent upon the relationship between the Lyapunov function and the state space. To understand this dependence, consider the schematic representation of the function E^{lyap} shown in Figure 2.1(a). As the network state $\mathbf{s}$ is updated according to equations (2.2) and (2.3), each change in state will result in a decrease in the Lyapunov function. Consequently, $\mathbf{s}$ moves towards the global minimum $\mathbf{s}^{min}$ and will eventually converge to it. Obviously, the discrete Hopfield network is well suited to finding the global minimum of this function. A contrasting scenario is shown in Figure 2.1(b). As any local minimum of E^{lyap} is a stable state, it is apparent that the network's ability to find the global minimum of this function is much reduced. Only if the initial state of the network lies in the basin of attraction for the global minimum will it be successful in finding the global minimum.

When mapping an optimisation problem into the form given by equation (2.1), the parameters $\mathbf{T}$ and $\mathbf{b}$ encode both the cost function and the constraints to be satisfied. It has previously been noted that the mapping process produces a Lyapunov function, or *energy landscape*, which is rich in structure with many local minima (Peterson and Anderson, 1988). Consequently, as illustrated by the simple example given in Figure 2.1(b), the discrete Hopfield network performs poorly.

2.2.2 Stochastic Update

Since the discrete Hopfield network, when operating with a deterministic update rule, is likely to become stuck at a local minimum of the Lyapunov function, it is desirable to incorporate a mechanism to facilitate escape from such local minima. To that end, a stochastic update rule has been suggested (Peretto, 1984),

$$\Pr(s_i = \pm 1) \;=\; g_\beta(\pm u_i) \;=\; \frac{1}{1+\exp(\mp \beta u_i)} \tag{2.5}$$

where u_i is defined by equation (2.2) and β is a parameter that controls the steepness of the transfer function $g_\beta(u_i)$ near $u_i = 0$. In the limit $\beta \to \infty$, $g_\beta(u_i)$ becomes a step function and the stochastic update rule reduces to the deterministic rule of equation (2.3). As β is decreased, this sharp threshold is softened in a stochastic manner. When the discrete Hopfield network operates with a stochastic rule it is known as the stochastic Hopfield network.

What is the meaning of this stochastic behaviour? The effect of the stochastic update rule may be loosely thought of as the injection of noise into the network, as modelled by thermal fluctuations. It is convenient to define the *pseudo-temperature* for the network as, $T^p \equiv \frac{1}{\beta}$. At high pseudo-temperature levels there is a large amount of noise in the network, and consequently transitions that violate the deterministic rule (2.3) are quite frequent. As the pseudo-temperature drops the amount of noise also drops, and

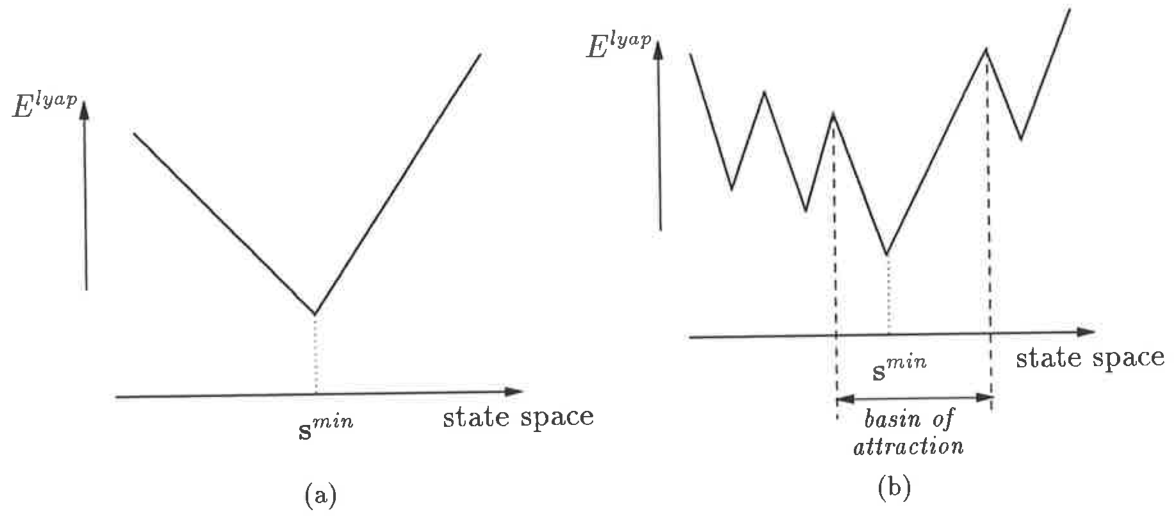


Figure 2.1: Schematic representation of two Lyapunov functions.

The ability of the discrete Hopfield network to find the global minimum of E^{lyap} is dependent on the relationship between E^{lyap} and the state space. These two Lyapunov functions provide contrasting examples for the performance of the discrete Hopfield network. (a) Convergence to the global minimum of this function is guaranteed. (b) The network will converge to the global minimum only if the initial state lies within the basin of attraction shown.

updates of the network state adhere more strictly to the deterministic rule. The stochastic update rule still moves the state vector from corner to corner of the hypercube, but each move is no longer guaranteed to decrease $E^{lyap}(\mathbf{s})$. In fact, any move which violates the deterministic update rule of equation (2.3) will result in an increase in $E^{lyap}(\mathbf{s})$. The injection of noise into the system dynamics has therefore made it possible to kick the network out of a local minimum of the energy function.

It is easily shown (Hopfield, 1982) that a change Δs_i in the state of neuron i produces a resultant change in the Lyapunov function given by $\Delta E^{lyap} = -u_i \Delta s_i$. Consider the situation where $s_i = -1$; we wish to evaluate the probability that an update of neuron i will select $s_i = +1$. From equation (2.5)

$$\Pr(s_i = +1) = \frac{1}{1 + \exp(\Delta E^{lyap}/2T^p)}. \tag{2.6}$$

Therefore, from equation (2.6) it can be seen that at high pseudo-temperatures T^p, the probability of accepting a transition that increases E^{lyap} is quite significant. Consequently, the system is able to wander over the state space with little regard to the underlying energy function. As the pseudo-temperature is decreased, uphill moves are less likely to be accepted and therefore it becomes increasingly difficult for the network to escape local minima. Finally, as $T^p \to 0$ all local minima of E^{lyap} are again stable states. Operating the stochastic Hopfield network at progressively lower pseudo-temperatures is in fact a useful optimisation technique. Networks operating in this manner are usually referred to as *Boltzmann machines* (Ackley et al., 1985; Aarts and Korst, 1989; Zissimopoulos et al., 1991).

2.3 Simulated Annealing

Simulated annealing (Kirkpatrick et al., 1983) is a general purpose optimisation technique that performs a discrete, stochastic search of the state space. The stochastic search process makes use of a pseudo-temperature parameter to determine the probability of accepting a transition that would result in an increase in the objective function. As the simulated annealing progresses the temperature is gradually lowered, until at zero temperature, uphill transitions are no longer accepted. Simulated annealing is a successful approach to optimisation, although it is well-renowned to have an unacceptably long running time. Indeed, in the limit of infinite computation time, simulated annealing is guaranteed to find the optimal solution (Aarts and Korst, 1989). As previously discussed, the combination of simulated annealing and a stochastic Hopfield neural network, produces a stochastic, parallel solution algorithm for combinatorial optimisation, commonly known as the Boltzmann machine (Ackley et al., 1985).

2.4 Mean Field Annealing

Mean field annealing (MFA) is a technique, closely related to the Boltzmann machine, that operates on the *average* statistics of the simulated annealing process. MFA provides a deterministic approximation to simulated annealing which gives greatly improved execution speed at the expense of guaranteed solution quality. MFA operates by computing a solution to the *saddle point equations*

$$u_i = \frac{1}{T^p}\left(\sum_j T_{ij} v_j + b_i\right) \tag{2.7}$$

$$v_i = \frac{1}{1+\exp(-u_i)} \tag{2.8}$$

at progressively lower temperatures. Equations (2.7) and (2.8) have been derived in Appendix A, where it is shown that a solution $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$ to the saddle point equations gives valuable information about a stochastic Hopfield network operating at the same pseudo-temperature T^p. In particular, the solution $\tilde{\mathbf{v}}$ to the saddle point equations gives the average value of the state of the stochastic Hopfield network *i.e.* $\tilde{\mathbf{v}} = \langle \mathbf{s} \rangle$. Moreover, at low temperatures it may be expected that $\tilde{\mathbf{v}}$ will closely approximate $\mathbf{s}^{min}$, the state that globally minimises $E(\mathbf{s})$.

It is shown in Appendix A that a solution $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$ to the saddle point equations is a local minimum of the *effective energy*

$$E'(\mathbf{u}, \mathbf{v}, T^p) = \frac{E(\mathbf{v})}{T^p} + \mathbf{u}^T\mathbf{v} - \sum_i \ln\left(\exp(u_i) + 1\right)$$

where $E(\mathbf{v})$ is given by equation (2.1). It is worthwhile to make several remarks about the effective energy. Firstly, it is important to note that the effective energy $E'(\mathbf{u}, \mathbf{v}, T^P)$ is smoother than $E(\mathbf{v})$ due to the presence of the $\sum_i \ln(\exp(u_i)+1)$ terms. Unfortunately, at low temperatures the rugged structure of the quadratic $\frac{E(\mathbf{v})}{T^p}$ term will dominate the smooth logarithmic terms and in addition to the global minimum at $(\bar{\mathbf{u}}, \bar{\mathbf{v}})$ the effective energy may have many sub-minima[3]. Therefore, it is likely that the solution to the saddle point equations will simply give values for $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$ that correspond to one of the sub-minima. In contrast, at high temperatures it is much less likely that the solution to the saddle point equations will correspond to a sub-minimum as the smooth logarithmic terms will dominate the effective energy. However, at such high temperatures $\tilde{\mathbf{v}} = \langle \mathbf{s} \rangle$ does not correspond to a single configuration of the underlying stochastic Hopfield network, but is an average over many states. Ultimately, when the temperature has risen to a sufficiently high level, there is only one minimum of the effective energy.

Therefore, in a manner similar to simulated annealing (Kirkpatrick et al., 1983), the saddle point equations are first solved at a high temperature. The temperature is then lowered and a new solution to the saddle point equations is computed. As the process is repeated at gradually decreasing temperatures, $\tilde{\mathbf{v}}$ should evolve from an average over many good configurations to the final desired value $\mathbf{s}^{min}$, while avoiding the sub-minima of the effective energy. The process of solving the saddle point equations at a series of progressively lower temperatures is known as *mean field annealing*.

The most common method for solving the saddle point equations (2.7) and (2.8) at each temperature is an iterative replacement procedure,

$$\mathbf{u}|_{t+1} = \mathbf{T}\mathbf{v}|_t + \mathbf{b} \tag{2.9}$$

$$v_i|_{t+1} = \frac{1}{1+\exp(-u_i|_{t+1}/T^p)} \tag{2.10}$$

The update rules (2.9) and (2.10) define the MFA algorithm[4]. It should be noted that there is no convergence guarantee for the MFA algorithm just described. In fact E may increase at any iteration. In practice MFA is suitable for many optimisation problems and exhibits rapid convergence to a solution of the saddle point equations.

[3]The term sub-minimum has been used to describe a local minimum which is not the global minimum.

[4]Alternative algorithms for solving the saddle point equations (2.7) and (2.8) have been proposed (Peterson and Anderson, 1988).

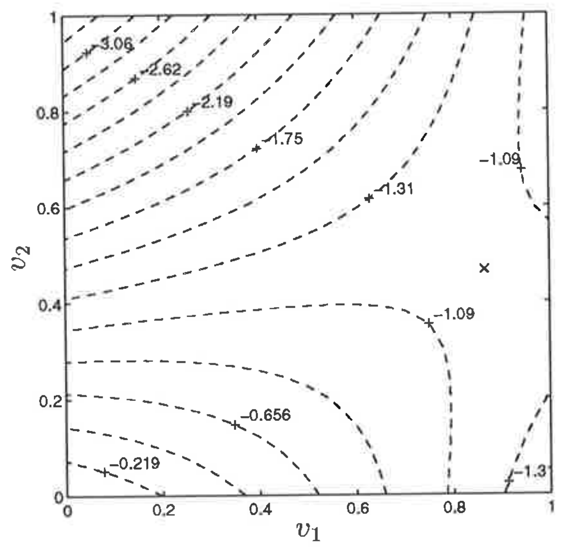


Figure 2.2: Contours of the quadratic objective function.

A contour plot for the function $E(\mathbf{v}) = -\frac{1}{2}(v_1^2+v_2^2)+4v_1v_2-v_1-3v_2$. *The global minimum of this function is* $E([0,\ 1]^T) = -3.5$. *Note the saddle point marked with an* $\times$ *at* $\mathbf{v} = [0.8667,\ 0.4667]^T$ *and the existence of the sub-minimum* $E([1,\ 0]^T) = -1.5$.

2.4.1 A Simple Example

As a simple example of the operation of the MFA algorithm, we shall use it to minimise the quadratic function

$$\begin{aligned} E(\mathbf{v}) &= -\frac{1}{2}\mathbf{v}^T\begin{bmatrix} 1 & -4 \\ -4 & 1 \end{bmatrix}\mathbf{v} - \mathbf{v}^T\begin{bmatrix} 1 \\ 3 \end{bmatrix} \\ &= -\frac{1}{2}(v_1^2+v_2^2) + 4v_1v_2 - v_1 - 3v_2\,. \end{aligned} \tag{2.11}$$

A contour plot of $E(\mathbf{v})$ is given in Figure 2.2. In order to use the MFA algorithm to minimise $E(\mathbf{v})$, we must set

$$\mathbf{T} = \begin{bmatrix} 1 & -4 \\ -4 & 1 \end{bmatrix}$$

and

$$\mathbf{b} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}.$$

The operation of the MFA algorithm is illustrated in Figure 2.3. The saddle point equations (2.7) and (2.8) are solved at a series of progressively lower temperatures until $v_i \in \{0,1\}$ when $T^p \to 0$. Rapid convergence of the update rules (2.9) and (2.10) was observed at each temperature T^p. Convergence to the global minimum of this particular objective function may always be obtained, regardless of the initial conditions, by utilising a gradual annealing schedule.

2.5 Continuous Hopfield Network

The continuous state Hopfield neural network (Hopfield, 1984) is constructed by interconnecting a large number of simple *analogue* processing units. The ith processing unit

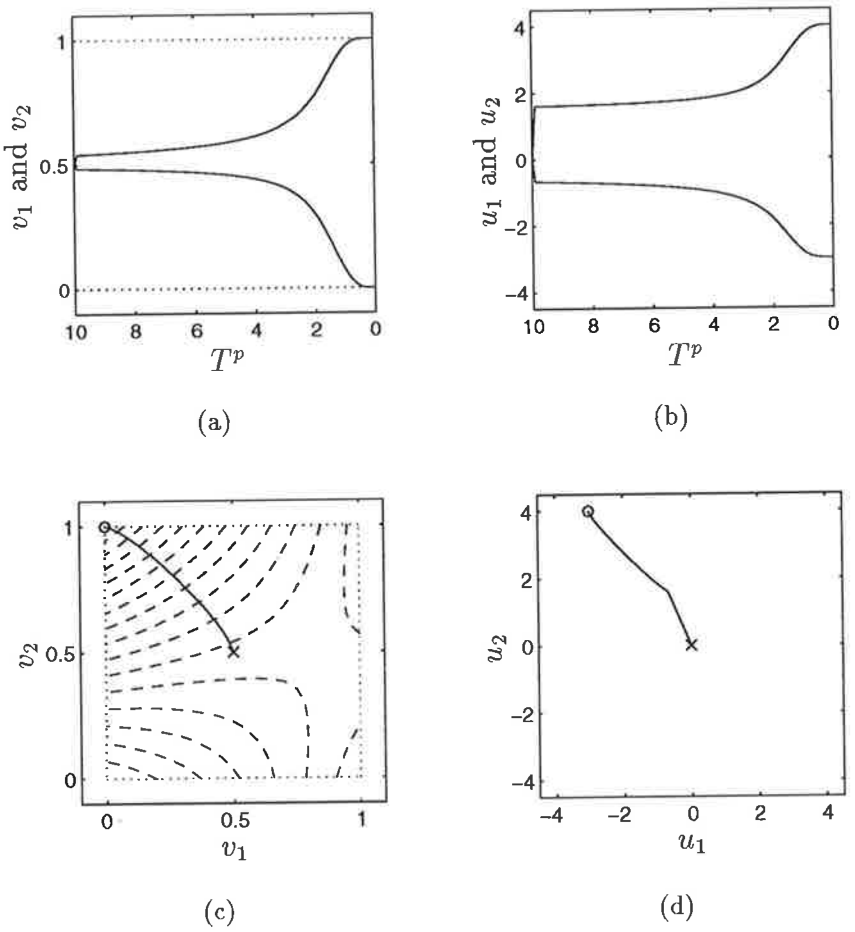


Figure 2.3: Operation of the MFA algorithm.

The plots show the operation of the MFA algorithm on the objective function E given in equation (2.11). As the temperature is lowered from 10 to 0.1, in equal decrements of 0.1, the saddle point equations are solved for $\mathbf{u}$ and $\mathbf{v}$. At each temperature the solutions were obtained using the MFA update rule, iterating until $||\Delta\mathbf{v}|| < 0.0001$. The network was initialised with $\mathbf{v} = [0.5, 0.5]^T$ and converged to the global minimum at $\mathbf{v} = [0, 1]^T$ as $T^p \to 0$. Figures (a) and (b) show the values of $\mathbf{u}$ and $\mathbf{v}$ obtained from the saddle point equations at each temperature. Figures (c) and (d) give the trajectories of $\mathbf{u}$ and $\mathbf{v}$ through the state space. The dashed lines shown in Figure (c) are contours of the objective function E.

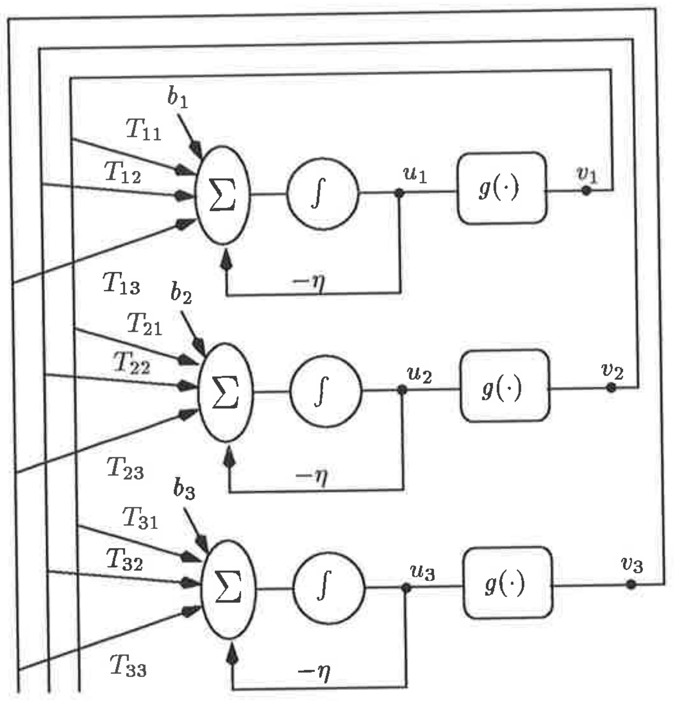


Figure 2.4: Continuous Hopfield network with 3 neurons.

The outputs v_j of the neurons are fed-back to the inputs, where they are weighted by the connection strengths T_{ij}. The summation of weighted outputs, external bias b_i and decay term $-\eta u_i$ is passed through an integrator to give the internal state u_i. The internal state passes through a transfer function to give the output $v_i = g(u_i)$.

is described by two variables: its internal state u_i and its output v_i. Unlike the binary outputs of the discrete state Hopfield network (see Section 2.2), the neuron outputs may assume a value $v_i \in [0,1]$. As before, the strength of the synaptic connection from the output of neuron j to the input of neuron i is given by T_{ij}. Each neuron also has an external bias b_i applied to the input. A schematic diagram of a continuous Hopfield network is given in Figure 2.4. The network forms a dynamic system described by the following equations:

$$\frac{du_i}{dt} = \sum_j T_{ij} v_j - \eta u_i + b_i \tag{2.12}$$

$$v_i = g(u_i)\,. \tag{2.13}$$

Here η is a decay parameter. The transfer function $g(\cdot)$ is a monotonically increasing function that restricts the neuron outputs to the range $v_i \in [0,1]$. The usual choice is the shifted hyperbolic tangent (sigmoid) function

$$g(u_i) = \frac{1}{1+\exp(-u_i/T^p)} \tag{2.14}$$

where T^p is a parameter controlling the slope of the transfer function in the linear region, and is termed the *pseudo-temperature.*

If the interconnections are symmetric *i.e.* $T_{ij} = T_{ji}$, then the network admits the Lyapunov function (Hopfield, 1984)

$$E^{lyap}(\mathbf{u}) = -\frac{1}{2}\sum_i\sum_j T_{ij}v_iv_j - \sum_i b_iv_i + \eta\sum_i\int_{0.5}^{v_i} g^{-1}(V)dV. \tag{2.15}$$

Under the assumption of symmetric interconnections and monotonically increasing transfer functions,

$$\begin{aligned}
\frac{dE^{lyap}}{dt} &= \sum_i \frac{\partial E^{lyap}}{\partial v_i}\frac{dv_i}{dt} \\
&= \sum_i \left(-\sum_j T_{ij}v_j + \eta u_i - b_i\right) g'(u_i)\frac{du_i}{dt} \\
&= -\sum_i g'(u_i)\left(\frac{du_i}{dt}\right)^2 \\
&\leq 0.
\end{aligned}$$

Since E^{lyap} is bounded below and the time derivative of the Lyapunov function is non-increasing, it is apparent *"that the time evolution of the system is a motion in state space that seeks out minima in E^{lyap} and comes to a stop at such points"* (Hopfield, 1984).

Relation Between Stable States of the Continuous and Discrete Hopfield Networks

The nature of the relationship that exists between the stable states of the continuous and discrete Hopfield networks is primarily determined by the pseudo-temperature T^p. The following discussion is based on the original work of Hopfield (Hopfield, 1984).

Consider only the first two terms of the Lyapunov function given by equation (2.15) for the continuous Hopfield network and assume that $T_{ii} = 0$. Then $E^{lyap} = E$ where

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} T_{ij} v_i v_j - \sum_i b_i v_i \, . \tag{2.16}$$

Obviously, E is equal to the Lyapunov function for the discrete Hopfield network, as given in equation (2.4). Typically, all extrema of E will lie at vertices[5] of the N-dimensional hypercube $0 \leq v_i \leq 1$. The discrete Hopfield network searches for minimal states at the vertices of the hypercube *i.e.* the network searches for a vertex that is lower in E than its adjacent vertices. Since E is a linear function of a single v_i along any edge of the hypercube, the extrema of E in the discrete space $v_i \in \{0,1\}$ are exactly the same vertices as the extrema of E defined over the continuous space $0 \leq v_i \leq 1$.

The integral term in the Lyapunov function of equation (2.15) for the continuous Hopfield network complicates the relationship somewhat. The integral is zero when $v_i = 1/2$ and is positive otherwise, becoming larger as v_i approaches the asymptotes at 0 and 1, as indicated in Figure 2.5. In the low pseudo-temperature limit $T^p \to 0$, the integral term becomes negligible and the stable states of the low pseudo-temperature continuous Hopfield network are exactly those of the discrete state Hopfield network.

For small non-zero T^p, the integral term gives a significant positive contribution to E^{lyap} near all surfaces of the hypercube, while it still contributes negligibly at the centre of the hypercube. Therefore, the maxima of the energy function remain at the vertices of the hypercube, but the minima are slightly displaced towards the interior. As the pseudo-temperature increases, each minimum moves further inward. Gradually the minima and saddle-points coalesce until, at very high pseudo-temperatures, the integral term in equation (2.15) dominates and the only minimum is at $v_i = 1/2 \; \forall i$. When the pseudo-temperature is suitably low so that there are many minima, each minimum is associated with a single minimum of the $T^p = 0$ case. Indeed as $T^p \to 0$ each minimum moves toward a particular vertex of the hypercube.

2.5.1 A Simple Example

To demonstrate the operation of the Hopfield network, we shall use it to minimise the quadratic function given by equation (2.11). For completeness, we restate the objective function as

$$E(\mathbf{v}) = -\frac{1}{2}\mathbf{v}^T \begin{bmatrix} 1 & -4 \\ -4 & 1 \end{bmatrix} \mathbf{v} - \mathbf{v}^T \begin{bmatrix} 1 \\ 3 \end{bmatrix} .$$

To use the Hopfield network to minimise E, it is necessary to set the interconnection matrix and external inputs as

$$\mathbf{T} = \begin{bmatrix} 1 & -4 \\ -4 & 1 \end{bmatrix}$$

and

$$\mathbf{b} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} .$$

The operation of the Hopfield network is illustrated in Figure 2.6. Annealing has not been used in this example. In order to equate the full Lyapunov function for the Hopfield network, as given by equation (2.15), with the objective function E it was necessary to set the decay parameter $\eta = 0$. Consequently, the $\mathbf{u}$ variables are unbounded and will

[5] In the unusual situation that T is positive or negative definite, it is possible for an extremum to exist in the interior of the hypercube.

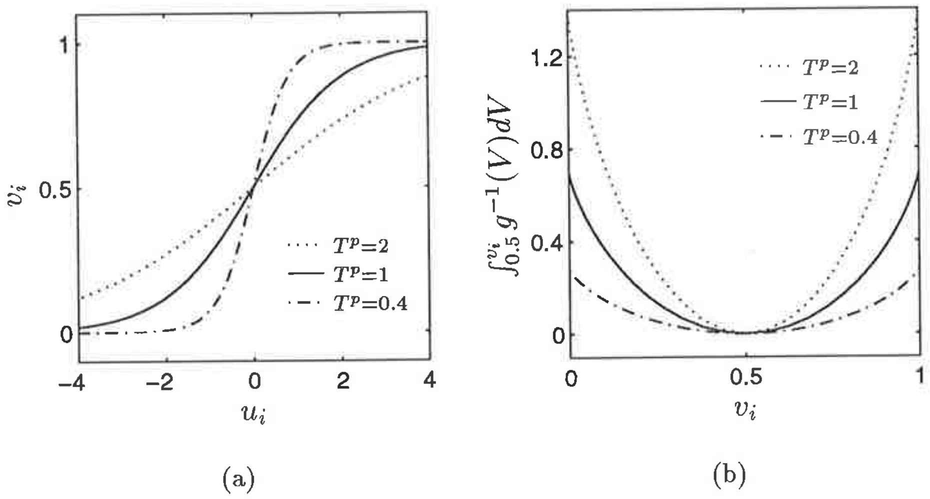


Figure 2.5: Plots of the transfer function and integral term in E^{lyap}.

(a) The sigmoidal input-output transfer function $v_i = g(u_i)$ for various values of T^p. As $T^p \rightarrow 0$ the transfer function approaches the step function used in the discrete Hopfield network. (b) The contribution of the integral term $\int_{0.5}^{v_i} g^{-1}(V)dV$ to the Lyapunov function in equation (2.15) for various values of T^p. As $T^p \rightarrow 0$ the integral term decreases in magnitude and E^{lyap} for the continuous Hopfield network approaches that of the discrete network i.e. equation (2.4).

continue to grow in magnitude. This is only practical when simulating the network on a digital computer. For a hardware implementation, η is set to a small non-zero value and the quadratic objective function may be approximated by the full Lyapunov function for the network only if very high gain transfer functions are used. In that case the contribution to the Lyapunov function from the integral term in equation (2.15) will be negligible.

As the Hopfield network operates by performing gradient descent, the stable point at which it arrives is dependent upon the initial conditions. Although the global minimum was found in the example given in this section, it is apparent that initial conditions exist that would lead to the sub-minimum at $\mathbf{v} = [1, 0]^T$. The process of annealing, as discussed in Section 2.5.2, is intended to prevent such behaviour and will be the subject of further discussion in Chapter 4.

2.5.2 Annealing Techniques

An important property of the Lyapunov function E^{lyap} is that it is smoother than the quadratic E due to the presence of the $\int_{0.5}^{v_i} g^{-1}(V)dV$ terms[6]. At low pseudo-temperatures, when the minima of the Lyapunov function approach the vertices of the hypercube, the rugged structure of the quadratic E will dominate the smooth integral terms, and in addition to the global minimum at $\bar{\mathbf{v}}$ the Lyapunov function will have many sub-minima. As the operation of the Hopfield network is a motion in state space that seeks out minima of the Lyapunov function, it is likely that the stable point reached by the network will be a sub-minimum. However, at sufficiently high pseudo-temperatures this is less likely, as the smooth integral terms dominate and only the deepest minimum of E will be evident in the Lyapunov function. Unfortunately, at high pseudo-temperatures the minima of the Lyapunov function have been displaced from the vertices of the hypercube towards the interior and are no longer interpretable solutions of the combinatorial problem. In order to avoid the sub-minima of the Lyapunov function whilst also ensuring that the final stable point reached by the network is suitably near a $0-1$ point, it is usual to employ some sort of *annealing* mechanism.

Temperature Annealing

Temperature annealing (Hopfield and Tank, 1985) involves gradually increasing the gain of the transfer functions (decreasing the pseudo-temperature T^p) while integrating the dynamic equation (2.12). Such a computation is analogous to that performed by the MFA algorithm, where a solution to the saddle point equations is computed at a series of progressively lower temperatures. The relationship between MFA and the Hopfield network may be further explored by considering an Euler approximation to the Hopfield dynamics in equation (2.12) with time step Δt,

$$\mathbf{u}(t+\Delta t) = \mathbf{u}(t) + \Delta t\left(-\eta\mathbf{u}(t) + \mathbf{T}\mathbf{v}(t) + \mathbf{b}\right).$$

Furthermore, if we choose $\eta = 1$ and $\Delta t = 1$ the Euler approximation reduces to the MFA algorithm update rule (2.9) (Peterson and Anderson, 1988; Van den Bout and Miller, 1990). Although MFA is essentially a discrete algorithm it is obviously closely aligned with the temperature annealed Hopfield network.

[6]Note that a similar statement has been made about the effective energy in the MFA algorithm. The Lyapunov function for the continuous Hopfield network is in fact closely related to the effective energy for the MFA algorithm.

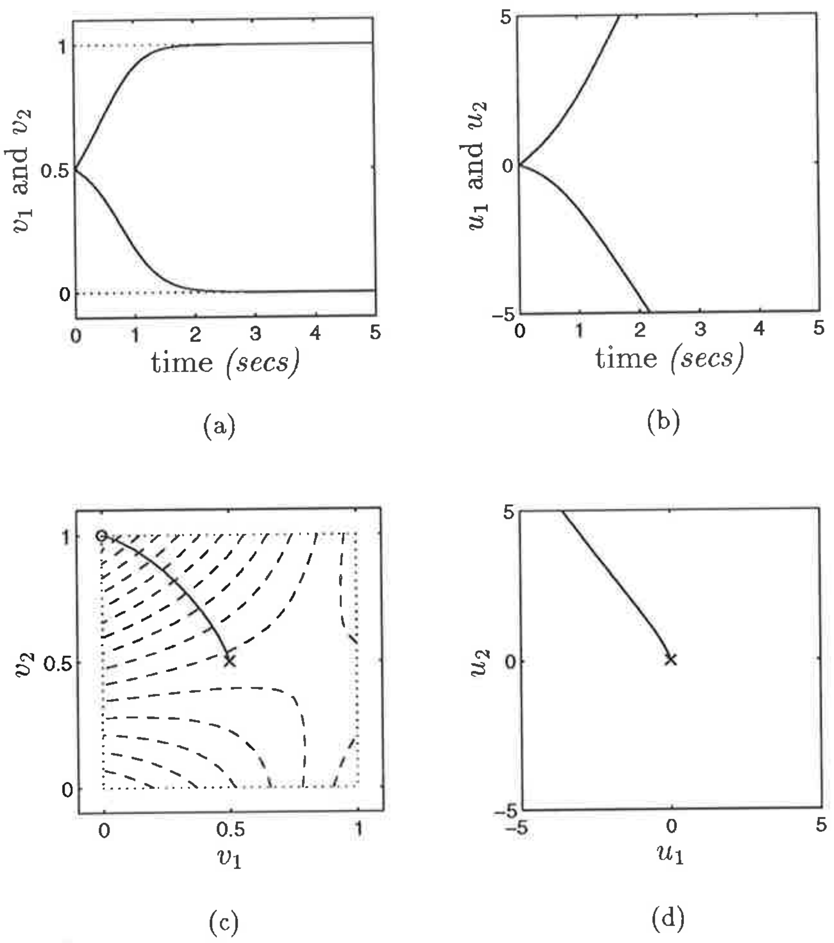


Figure 2.6: Operation of a Hopfield network.

The plots show the operation of a Hopfield network on the quadratic function E in equation (2.11). To ensure that $E^{lyap} = E$, the parameters $\eta = 0$ and $T^p = 1$ are used. Integration of the Hopfield dynamic equations yields the traces shown in (a) and (b). Since there is no decay term, the **u** *variables are unbounded. The gradient descent nature of the Hopfield dynamics is evident in (c) where the dashed lines are contours of E^{lyap}. The network is initialised at $\mathbf{v} = [0.5,\ 0.5]^T$ and converges to $\mathbf{v} = [0,\ 1]^T$, which is the the global minimum of the Lyapunov function within the unit square. The Hopfield dynamics, given by equations (2.12) and (2.13), were integrated using the Euler method with a constant step-size $\Delta t = 0.01$.*

Hysteretic Annealing

Hysteretic annealing (Eberhardt et al., 1991), matrix graduated non-convexity (Aiyer, 1991) and convex relaxation (Ogier and Beyer, 1990) are all closely related and we shall collectively refer to them as hysteretic annealing. Hysteretic annealing is an alternative approach to temperature annealing. When employing hysteretic annealing the pseudo-temperature parameter T^p is held constant and the decay term η is commonly set to zero, in which case the Lyapunov function in equation (2.15) reduces to the quadratic E in equation (2.16). In its most widely applicable form[7] (Gee, 1993) hysteretic annealing involves adding a term

$$E^{ann} = -\frac{1}{2}\gamma \sum_i (v_i - 0.5)^2 \tag{2.17}$$

to the Lyapunov function E^{lyap} of the Hopfield network. The function E^{ann} is either convex or concave, depending on the sign of the annealing parameter γ. Initially, γ is set to a large negative value, in which case E^{lyap} is convex and $\mathbf{v}$ converges to a point inside the hypercube. Subsequently, γ is gradually increased allowing more of the rugged structure of E to be evident in the Lyapunov function. Eventually E^{lyap} becomes concave and the network will converge to a vertex of the hypercube. Hysteretic annealing will be considered in more detail in Chapter 4. Hysteretic annealing may be implemented by modifying the dynamic equation (2.12), *viz.*

$$\frac{du_i}{dt} = \sum_j T_{ij} v_j - \eta u_i + b_i + \gamma (v_i - 0.5) \; .$$

2.6 Hardware Implementation

When compared to alternative methods for the solution of combinatorial optimisation problems, the one significant advantage of optimisation networks is their ability to approximately solve problems in milliseconds. Such extremely fast solution times may only be achieved by a direct implementation in hardware.

An analogue circuit implementation of the continuous Hopfield network is shown in Figure 2.7. Such an analogue circuit would relax to a stable state within a few time constants of the circuit components. Summing currents at the input to the operational amplifiers gives the circuit dynamics as (Hopfield, 1984)

$$C_i \frac{du_i}{dt} = \sum_j T_{ij} v_j + b_i - \frac{u_i}{r_i} \tag{2.18}$$

where

$$T_{ij} = \frac{1}{R^{\text{ex}}_{ij}} - \frac{1}{R^{\text{in}}_{ij}} \quad \text{and} \quad \frac{1}{r_i} = \sum_j \left(\frac{1}{R^{\text{ex}}_{ij}} + \frac{1}{R^{\text{in}}_{ij}} \right) + \frac{1}{R_i} \, .$$

The sigmoidal transfer function $v_i = g(u_i)$ is implemented by the operational amplifiers shown in Figure 2.7. Inclusion of both inverted and non-inverted outputs from the operational amplifier allows both inhibitory and excitatory connections to be made in the network. For an inhibitory interconnection ($T_{ij} < 0$) the inverted amplifier output for neuron j is connected to the input of neuron i by a resistor R^{in}_{ij}. An excitatory interconnection ($T_{ij} > 0$) is established by connecting R^{ex}_{ij} to the non-inverted amplifier

[7] It will be shown in Chapter 4 that the hysteretic annealing term E^{ann} given by equation (2.17) is not particularly well suited to the solution of many problems.

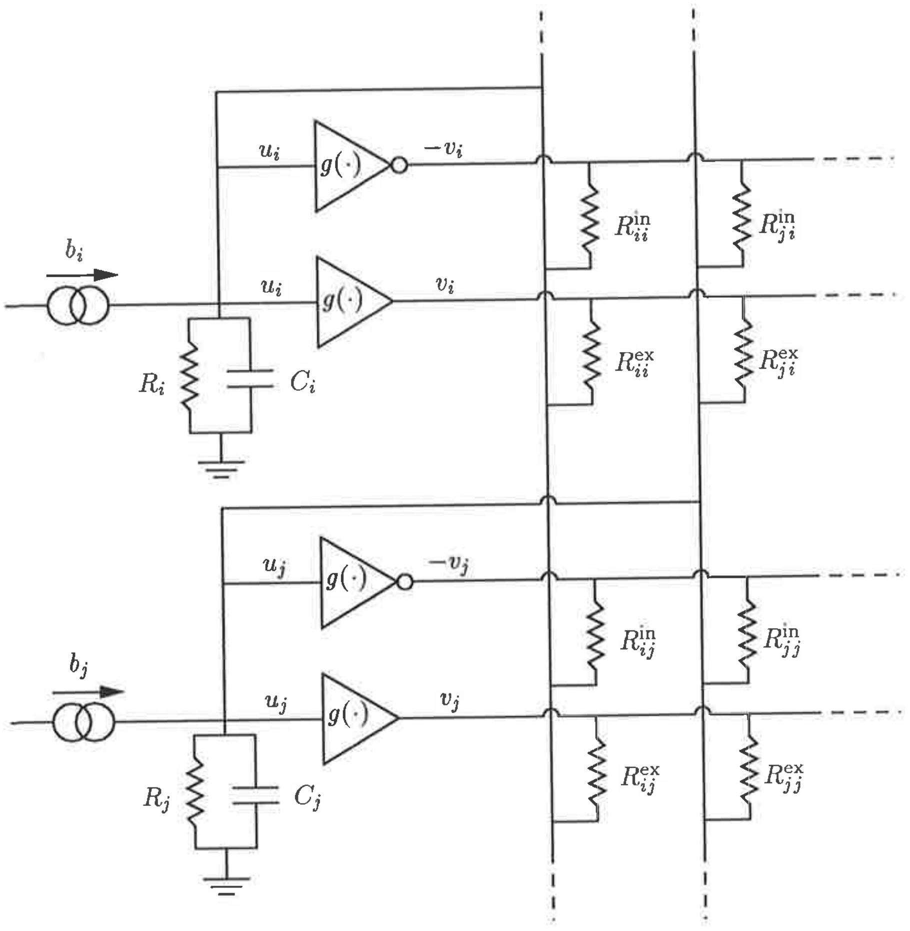


Figure 2.7: Analogue hardware implementation of a Hopfield network (Gee, 1993).

The operational amplifiers in the above circuit implement the sigmoidal transfer function $v_i = g(u_i)$. Inclusion of inverted and non-inverted amplifier outputs allows both inhibitory and excitatory connections to be implemented. The external bias b_i for each processing element is provided by a current source. Applying Kirchoff's current law at the input to the operational amplifiers reveals that the circuit dynamics are equivalent to those of a Hopfield network.

output. By comparison of equations (2.18) and (2.12), it is apparent that the circuit has dynamics equivalent to those of a Hopfield network with $\eta = 1/r_i$. The speed of operation is governed by the time constant of the circuit, as determined by the values of the circuit's capacitors and resistors. Annealing may be accomplished by slowly varying the gain of the amplifiers (Lee and Sheu, 1993; Bang et al., 1995).

Whilst circuits such as that in Figure 2.7 are a straightforward implementation of the Hopfield dynamics, they bear little resemblance to practical implementations. When undertaking analogue hardware design, consideration of non-idealities such as propagation delay (Smith and Portmann, 1989) and process variations (Johnson et al., 1995) heavily influence the implementation. Moreover, the programmable resistors required for the circuit of Figure 2.7 are difficult to implement in VLSI. Consequently, the operational amplifier – resistor circuit has been replaced in favour of MOSFET analogue amplifiers where the interconnection strengths may be stored as voltages (Eberhardt et al., 1992). Alternatively, analogue neural networks can be implemented using pulse stream VLSI (Hamilton et al., 1992; Johnson et al., 1995), where the advantages of both analogue and digital VLSI techniques may be exploited.

2.7 The Niche Market

As there already exists a wide variety of solution methods for combinatorial optimisation problems, it is important to understand the comparative advantages and disadvantages of optimisation networks in order to define their *niche market*. The existing algorithms may be broadly classified as either a construction heuristic, improvement heuristic or an exact method:

Construction Heuristics provide a one-shot solution algorithm *i.e.* they determine a solution according to some construction rule, but do not attempt to improve upon this solution. Typical examples of construction heuristics for the TSP are the nearest neighbour algorithm, insertion heuristics and heuristics based on spanning trees (Reinelt, 1994; Johnson, 1990). An average excess over the optimal solution cost, or alternatively a tightly computed lower bound, of 21% for the nearest neighbour algorithm, 14% for the insertion heuristics and 19% for the heuristics based on spanning trees have been reported for various Euclidean TSPs (Reinelt, 1994).

Improvement Heuristics take as their starting point the solution produced by a construction heuristic, and then attempt to improve upon this solution by iterating some type of basic moves. Examples of improvement heuristics for the TSP are node and edge insertion, and the k-opt heuristics of Lin and Kernighan (Lin, 1965; Lin and Kernighan, 1973). An average excess over the optimal solution cost, or alternatively a tightly computed lower bound, of 8.2% for the node and edge insertion algorithm, 8.4% for the 2-OPT algorithm, 3.6% for the 3-OPT heuristic and 1.3% for the full Lin-Kernighan heuristic have been reported for various Euclidean TSPs (Reinelt, 1994). More recently, simulated annealing (Kirkpatrick et al., 1983) and genetic algorithms (Goldberg, 1989; Fogel, 1994) have proved popular for combinatorial optimisation. While both techniques are indeed capable of producing good solutions they do require a considerably long running time.

Exact Methods such as the branch and bound algorithm (Lawler and Wood, 1966) or the more successful branch and cut algorithms (Grotschel and Holland, 1991; Padberg and Rinaldi, 1991) are capable of finding the exact solution to combinatorial optimisation problems such as the TSP. However, this is a time consuming process and it is often necessary to terminate such algorithms before optimality is reached.

Given the classification of existing techniques into these three broad categories, the trade-off that is possible between solution quality and computation time is made apparent in Figure 2.8. Obviously the precise execution times and solution qualities are problem dependent, but Figure 2.8 remains representative of the relative performance of these solution techniques. While construction heuristics may produce solutions of only moderate quality, they can execute on modern computer workstations in a matter of seconds. Improvement heuristics are capable of producing far better solution quality but typically require many minutes of computation time to do so. Exact methods are capable of solving combinatorial optimisation problems to optimality, but to do so they require very long run times. In practice they are often terminated after several hours of execution time, without having necessarily attained the optimal solution.

It is apparent from the vast literature available on the application of optimisation networks that they provide a solution quality that is worse than most other techniques, being comparable only with that of construction heuristics. Where optimisation networks have the edge is that they are inherently suited for implementation in fast, parallel hardware. Such hardware implementations give rise to solution times of the order of milliseconds.

However, optimisation networks have a significant disadvantage in that they are forced to adopt inefficient problem representations: for example, to solve an N-city TSP it is necessary to use a network of N^2 processors. Consequently, on a conventional digital computer, the time required for simulation prohibits optimisation networks from application to a TSP of more than 100 cities. However, as previously explained, the great advantage of optimisation networks is the quick solution times achieved when implemented in fast, parallel hardware and so their performance in simulation is only of secondary importance. Unfortunately, the inefficient problem representations lead to poor space complexity for hardware implementations.

It would appear from our discussion that the *niche market* for optimisation networks is moderate-sized problems of hundreds to tens of thousands of variables, where solution quality is not the overriding concern, but where execution time is of paramount importance. However, in the course of this thesis we will find it necessary to revise the definition of the niche market for optimisation networks — or at the very least, to acknowledge some previously unconsidered factors which are relevant to the definition of the niche market for optimisation networks.

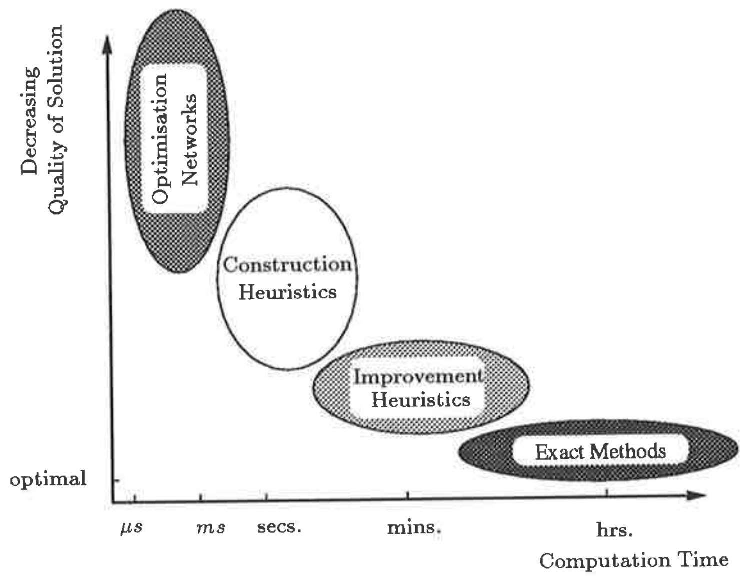


Figure 2.8: Comparison of solution methods for combinatorial optimisation.

The graph gives a representative comparison of the quality of solution attainable from, and computation time required for, several classes of algorithms. Existing algorithms have been classified as either a construction heuristic, improvement heuristic or an exact method. The remaining class is optimisation networks, which, at the expense of solution quality, are clearly the fastest algorithms.

Chapter III

The Problem Mapping

In this chapter we demonstrate how combinatorial optimisation problems can be *mapped* onto optimisation networks. Given that optimisation networks are well suited to the minimisation of a quadratic function over a set of $0-1$ variables, the goal of a successful mapping is to express the problem in exactly this form. Unfortunately, combinatorial optimisation problems are more naturally expressed as a quadratic $0-1$ programming problem: minimisation of a quadratic objective function, subject to a set of linear constraints, over a set of $0-1$ variables. While it is straightforward to associate the quadratic objective function with the Lyapunov function for an optimisation network, it is not readily apparent how to ensure that the constraints are satisfied. The answer is to recast the constraints as quadratic *penalty functions* which attain their minimum value only when the constraints are satisfied. Consequently, problem mapping involves the formulation of suitable penalty functions.

The mapping process has been the area of most intense research in the field of optimisation networks and so we begin in Section 3.1 by surveying the landmarks in this research. For the rest of the chapter we take as our starting point the valid subspace approach, which is the current state-of-the-art mapping technique. As a concrete example of the mapping process, the travelling salesman problem will be developed and the necessary problem representation is given in Section 3.2. When developing concepts in this chapter particular attention is paid to the travelling salesman problem. However, the concepts presented are widely applicable and may be used in any application of optimisation networks. To elucidate our preferred mapping technique, the valid subspace for a simple problem is developed in Section 3.3 and then generalised to the TSP. In Section 3.4 we show that once the valid subspace is defined, it is then a simple process to construct a penalty function which correctly constrains the network state to lie in the valid subspace. To gain more than just a geometrical insight into the action of the penalty function, we examine the penalty function to determine how the individual constraints of the TSP are enforced. In Section 3.5 we undertake a linearised analysis of the network dynamics to confirm that the chosen penalty function will ensure that the problem's constraints are satisfied. Finally, in Section 3.6 we show how to map some common problems onto an optimisation network.

3.1 Background

As was shown in the previous chapter, the operation of an optimisation network, such as the continuous Hopfield network, is to seek out minima in the Lyapunov function[1]

$$E^{lyap} = -\frac{1}{2}\mathbf{v}^T\mathbf{T}\mathbf{v} - \mathbf{v}^T\mathbf{b}. \tag{3.1}$$

Consequently, the Hopfield network provides a method for minimising functions given by equation (3.1), whilst limiting the output of the network to remain within the unit hypercube. Moreover, as the $0-1$ points lie at the vertices of the unit hypercube, the Hopfield network is well suited to the minimisation of a quadratic function over a set of $0-1$ variables.

Combinatorial optimisation problems, such as the travelling salesman problem, are naturally expressed as quadratic $0-1$ programming problems: where it is required to minimise a quadratic objective function E^{obj} over a set of $0-1$ variables, subject to a set of linear constraints. When mapping a combinatorial problem onto an optimisation network it would seem reasonable to set $E^{lyap} = E^{obj}$. However, such a strategy does not account for the problem's constraints. To do so, the linear constraints are recast as quadratic penalty functions E^{cns} which attain their minimum value whenever the constraints are satisfied. The problem may then be mapped onto the optimisation network by setting

$$E^{lyap} = E^{obj} + E^{cns},$$

and then identifying the necessary network parameters $\mathbf{T}$ and $\mathbf{b}$. The exact form of the penalty functions has been the topic of a great deal of empirical research.

An optimisation network is said to have obtained a *valid solution* if it converges to a $0-1$ point that satisfies the problem's constraints and is therefore an interpretable solution to the combinatorial problem. An *invalid solution* is obtained if the network converges to a $0-1$ point that does not satisfy the problem's constraints. The goal of a successful problem mapping is to ensure that the network state $\mathbf{v}$ remains valid (*i.e.* $\mathbf{v}$ always satisfies the problem's constraints) throughout the operation of the network. As will be shown in Chapter 4, when a successful problem mapping is combined with an appropriate annealing technique, convergence to a valid solution can be guaranteed.

One of the earliest criticisms of optimisation networks was for their inability to reliably produce valid solutions (Wilson and Pawley, 1988; Kamgar-Parsi and Kamgar-Parsi, 1990). Primarily, this was caused by the use of incorrectly formulated penalty functions (Aiyer, 1991). Indeed, it has been shown (Aiyer et al., 1990) that the penalty functions employed in the original mapping of the TSP onto the Hopfield network (Hopfield and Tank, 1985) have a tendency to force the network towards invalid solutions. To improve the performance of optimisation networks, a great deal of research concentrated on the search for a successful problem mapping. The result was a large array of *ad hoc* formulations for the penalty functions (Brandt et al., 1988; Protzel et al., 1993; Abe, 1993; Abe and Gee, 1995; Matsuda, 1995), usually employing a separate term for each constraint. Such ad hoc approaches were empirically motivated and achieved greatly varying levels of success.

The identification of successful problem mappings was placed on a solid theoretical foundation with the introduction of the *valid subspace* approach. The valid subspace approach defines a rigorous method for mapping a general combinatorial optimisation

[1]Note that the decay term η has been set to zero.

problem onto an optimisation network (Gee, 1993; Gee et al., 1993; Gee and Prager, 1994; Gee and Prager, 1995). Critical to the success of this approach, is the observation that any set of feasible linear constraints defines a bounded polyhedron, termed the valid subspace, in which the network state must lie if it is to be valid. By identifying the valid subspace, a penalty function can be defined that correctly constrains the network state to be valid.

One of the primary motivations for developing a valid subspace approach was the belief that ad hoc approaches suffered from a multiplicity of terms which tended to frustrate each other, and therefore could not guarantee a valid solution (Gee, 1993). In contrast, the valid subspace approach defines a single penalty function for the problem mapping. Such penalty functions, while easily understood from a geometric perspective, gave little insight into how individual constraints were enforced. In the course of presenting and verifying the validity of the valid subspace approach, we will show that the penalty function so defined actually consists of several terms, each designed to enforce a single constraint. In that sense the valid subspace approach is not at all dissimilar to the previous ad hoc approaches. The main advantage of the valid subspace approach is that it is a principled, general technique for problem mapping.

3.2 The Travelling Salesman Problem

As it is a popular benchmark for combinatorial optimisation, we will use the travelling salesman problem to demonstrate the mapping process. Restating the problem:

The travelling salesman problem (TSP) is to visit each of N cities once and only once, and return to the initial city having travelled the least possible distance.

This problem is $\mathcal{NP}$-complete (Lawler et al., 1985). To solve this problem with an optimisation network, an $N \times N$ array of neurons is used where the output of the neuron v_{xi} in row x and column i is one if city x is to be visited in the i^{th} position of the tour, and zero otherwise.

With this problem representation, the TSP may be formulated as the minimisation of an objective function over a set of $0-1$ variables, subject to a set of linear constraints. Denoting the distance between city x and city y as d_{xy}, the TSP is expressed as follows,

$$\text{minimise} \quad E^{obj} = \frac{1}{2}\sum_{x,y=1}^{N}\sum_{i=1}^{N} v_{xi}d_{xy}(v_{y,i-1}+v_{y,i+1}) \tag{3.2}$$

$$\text{subject to} \quad \sum_{x=1}^{N} v_{xi} = 1 \quad \forall\, i \in \{1,\dots N\} \tag{3.3}$$

$$\text{and} \quad \sum_{i=1}^{N} v_{xi} = 1 \quad \forall\, x \in \{1,\dots N\} \tag{3.4}$$

$$\text{where} \quad v_{xi} \in [0,1]. \tag{3.5}$$

Note that all indices are evaluated modulo N. As a solution to the TSP is being calculated, the variables of the optimisation network may take on values in the range $v_{xi} \in [0,1]$. However, for the final state of the optimisation network to be interpreted as a solution to the combinatorial optimisation problem, it is required that $v_{xi} \in \{0,1\}$. For the state of the network to represent a valid solution, the column and row sums of the array of neurons must equal one, as written in equations (3.3) and (3.4). Given a valid $0-1$ solution to the TSP, the objective function E^{obj} evaluates the length of the tour represented.

The operation of an optimisation network is to seek out minima in the Lyapunov function. As discussed in Section 3.1, to solve the TSP with an optimisation network it is essential that minima of the Lyapunov function correspond not only to short tours but also to valid representations of a tour. To achieve this the Lyapunov function is constructed as

$$E^{lyap} = E^{obj} + E^{cns},$$

where E^{obj} is the objective function for the TSP (3.2) and E^{cns} is a penalty function that is minimised when the constraints given in equations (3.3) and (3.4) are satisfied. In the rest of this chapter we will focus on the valid subspace approach to problem mapping. For details of other less successful problem mappings the reader is referred to the literature.

3.3 The Valid Subspace

The valid subspace approach to problem mapping relies on the identification of the *valid subspace*: a bounded region in which the problem's constraints are satisfied. For any set of feasible linear constraints it is possible to define a bounded polytope, such that if a $0-1$ point lies within the polytope then it must be a valid solution (Gee, 1993; Gee and Prager, 1994). This bounded polytope is in fact the valid subspace. This principle will be demonstrated on a simple 3-dimensional example and then generalised to the TSP.

3.3.1 *Simple 3-dimensional example*

Consider a problem with three variables, where for a vector to be valid it must meet the constraint

$$\sum_{i=1}^{3} v_i = 1 \qquad \text{where } v_i \in [0,1].$$

The shaded plane in Figure 3.1 shows the *valid subspace*, the set of all points that satisfy the constraints. Obviously, the only valid $0-1$ points are $\{[1,0,0]^T, [0,1,0]^T, [0,0,1]^T\}$, which lie at the vertices of the valid subspace, but there are many other points inside the unit cube which are valid.

The vector $\hat{\mathbf{e}} = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]^T$ is valid as the sum of its elements is equal to one. Consequently the tip of $\hat{\mathbf{e}}$ lies in the valid subspace. It is apparent from Figure 3.1 that any other valid vector $\mathbf{v}$ may be reached from $\hat{\mathbf{e}}$ by the addition of a component $\mathbf{v}^{zs}$,

$$\mathbf{v} = \mathbf{v}^{zs} + \hat{\mathbf{e}}.$$

As $\hat{\mathbf{e}}$ already satisfies the constraints, it is essential that the sum of the elements of $\mathbf{v}^{zs}$ equals zero. Therefore, $\mathbf{v}^{zs}$ is said to lie in the *zerosum subspace*. The vector $\hat{\mathbf{e}}$ is orthogonal to the zerosum subspace, since $\mathbf{v}^{zs} \cdot \hat{\mathbf{e}} = 0$.

3.3.2 *The TSP*

To simplify the discussion it is assumed that the output of the $N \times N$ array of neurons that is used to solve the TSP has been reshaped into a vector. When dealing with this vector it is to be understood that the terms row sum and column sum refer to the sums of elements in the rows and columns of the underlying matrix structure.

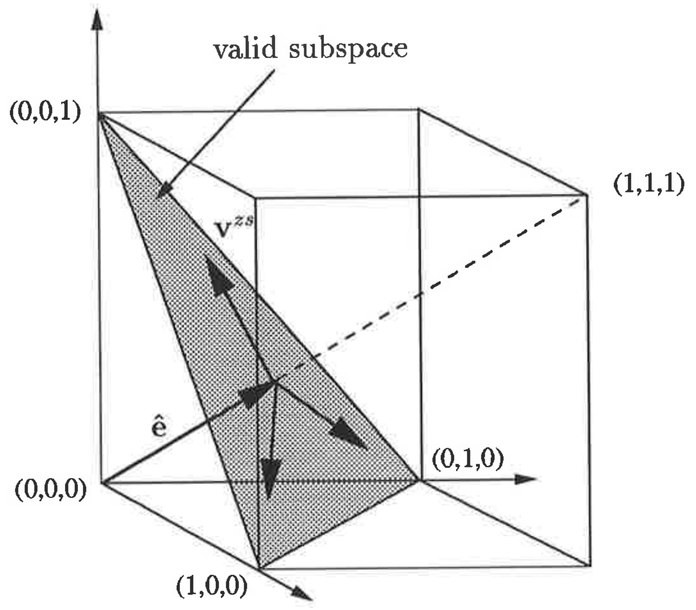


Figure 3.1: The valid subspace for a simple 3-D example (Aiyer, 1991).

The shaded plane shows the valid subspace for the constraint that $\sum_{i=1}^{3} v_i = 1$. *The only valid* $0-1$ *points are* $\{[1,0,0]^T, [0,1,0]^T, [0,0,1]^T\}$, *but there any many points inside the unit cube which satisfy the constraint. Any point which lies on the valid subspace can be expressed as* $\mathbf{v} = \mathbf{v}^{zs} + \hat{\mathbf{e}}$.

Now, unlike the simple 3-dimensional example, the constraints (3.3) and (3.4) for the TSP require that all the row and column sums of a valid vector equal one. However, it is still possible to define a valid subspace for the TSP. Integral to the definition of the valid subspace is the *zerosum subspace*, which in the case of the TSP, is the set of all vectors that have row and column sums equal to zero. Any vector $\mathbf{v}$, regardless of whether it is valid or invalid, may be decomposed into three mutually orthogonal components (Gee, 1993)

$$\begin{aligned} \mathbf{v} &= \mathbf{v}^{zs} + \mathbf{v}^{1} + \mathbf{v}^{inv} \\ &= \mathbf{T}^{zs}\mathbf{v} + (\mathbf{v}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}} + \mathbf{v}^{inv} \end{aligned} \tag{3.6}$$

where $\mathbf{v}^{zs} = \mathbf{T}^{zs}\mathbf{v}$ is the component of $\mathbf{v}$ that lies in the zerosum subspace, $\mathbf{T}^{zs}$ is a suitably defined projection matrix, $\mathbf{v}^{1} = (\mathbf{v}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}$ is a component in the direction of $\hat{\mathbf{e}} = \frac{1}{N}[1,1,\ldots,1]^T$ and $\mathbf{v}^{inv}$ is a component in the *invalid subspace*.

If a vector $\mathbf{v}$ does not satisfy the constraints (3.3) and (3.4), it is invalid and must therefore have a non-zero component $\mathbf{v}^{inv}$ in the invalid subspace. Conversely, if a vector is valid, there can be no component in the invalid subspace. Additionally, as any component in the zerosum subspace does not contribute to the row and column sum, it is necessary for the magnitude of $\mathbf{v}^1$ to be such that the column and row sums equal one. With these considerations it is apparent that any vector in the valid subspace can be written as

$$\begin{aligned} \mathbf{v} &= \mathbf{v}^{zs} + \hat{\mathbf{e}} \\ &= \mathbf{T}^{zs}\mathbf{v} + \hat{\mathbf{e}}. \end{aligned}$$

Obviously, $\mathbf{v}^1 = \hat{\mathbf{e}}$ satisfies the requirement for the row and column sum of a valid vector to be equal to one, while a component $\mathbf{v}^{zs}$ will not alter the row and column sum. Interestingly, all the vertices of the valid subspace coincide with valid $0-1$ points (Gee and Prager, 1994).

3.4 Enforcing Validity

Now that the valid subspace for the TSP has been defined, it is a simple matter to construct an appropriate penalty function. By appealing to simple qualitative arguments, we will show in this section that the chosen penalty function does indeed enforce the problem's constraints.

To ensure that $\mathbf{v}$ remains valid, we wish to encourage $\mathbf{v}$ to lie in the valid subspace during the operation of the network. This can be achieved by the use of an appropriate penalty function (Gee, 1993; Gee and Prager, 1994),

$$E^{cns} = \frac{1}{2}c\|\mathbf{v} - (\mathbf{T}^{zs}\mathbf{v} + \hat{\mathbf{e}})\|^2 \tag{3.7}$$

where c is a positive constant[2]. When $\mathbf{v}$ lies on the valid subspace, E^{cns} is zero, while E^{cns} grows rapidly as $\mathbf{v}$ strays further away from the valid subspace. To map a particular problem onto the network, the Lyapunov function is set as

$$E^{lyap} = E^{obj} + E^{cns}$$

[2]Note that c is similar to a Lagrange multiplier.

and in the limit of large c, E^{cns} will dominate over E^{obj} so that the gradient descent nature of the network will effectively pin $\mathbf{v}$ to the valid subspace. Furthermore, as long as $\mathbf{v}$ remains in the valid subspace, the penalty term does not interfere with the objective function, so that $E^{lyap} = E^{obj}$. This concept is illustrated in Figure 3.2.

The penalty function (3.7) is easily understood from a polyhedral perspective, but it is less than clear how it enforces the row and column sum to be equal to one. One of the primary motivations for developing the valid subspace approach was that penalty functions were previously constructed in an *ad hoc* manner, employing one term for each constraint, and it was suggested that such a multiplicity of terms tended to frustrate each other (Gee, 1993). While the following analysis will confirm that the valid subspace approach is indeed correct, it will further show that the penalty function (3.7) is itself constructed from several terms which individually enforce constraints. The remainder of this section is devoted to reformulating the penalty function (3.7) to understand how it enforces the constraints for the TSP.

Expanding equation (3.7) we obtain[3]

$$\begin{aligned} E^{cns} &= -\frac{c}{2}\mathbf{v}^T(\mathbf{T}^{zs} - \mathbf{I})\mathbf{v} - c\mathbf{v}^T\hat{\mathbf{e}} + \frac{c}{2}\hat{\mathbf{e}}^T\hat{\mathbf{e}} \\ &= -\frac{c}{2}\sum_{\rho,\sigma=1}^{N^2} v_\rho v_\sigma[\mathbf{T}^{zs} - \mathbf{I}]_{\rho\sigma} - \frac{c}{N}\sum_{\rho=1}^{N^2} v_\rho + \frac{c}{2}. \end{aligned}$$

Now to evaluate E^{cns} in terms of v_{xi}, the output of the neuron in row x and column i of the neuron array, we must make explicit the transformation between the $N \times N$ array of neurons and its vector representation,

$$v_\rho = v_{xi} \qquad \text{where } \rho = (x-1)N + i \tag{3.8}$$

and,

$$v_\sigma = v_{yj} \qquad \text{where } \sigma = (y-1)N + j \tag{3.9}$$

for $x, i, y, j \in \{1, \ldots N\}$ and $\rho, \sigma \in \{1, \ldots N^2\}$. With $[\mathbf{A}]_{u,v}$ denoting the u, v element of a matrix $\mathbf{A}$, the projection matrix $\mathbf{T}^{zs}$ may be defined as (Aiyer, 1991; Gee et al., 1993)

$$[\mathbf{T}^{zs}]_{\rho\sigma} = \left(\delta_{xy} - \frac{1}{N}\right)\left(\delta_{ij} - \frac{1}{N}\right) \tag{3.10}$$

and the identity matrix is given by

$$[\mathbf{I}]_{\rho\sigma} = \delta_{xy}\delta_{ij}$$

where the relationships between the variables ρ, σ and x, y, i, j are described by equations (3.8) and (3.9). Making substitutions into the expression for E^{cns} we obtain

$$\begin{aligned} E^{cns} &= -\frac{c}{2}\sum_{xi}\sum_{yj}\left(\left(\delta_{xy} - \frac{1}{N}\right)\left(\delta_{ij} - \frac{1}{N}\right) - \delta_{xy}\delta_{ij}\right) v_{xi}v_{yj} - \frac{c}{N}\sum_{xi} v_{xi} + \frac{c}{2} \\ &= \frac{c}{2}\left(\frac{1}{N}\sum_{xy}\sum_{i} v_{xi}v_{yi} + \frac{1}{N}\sum_{x}\sum_{ij} v_{xi}v_{xj} - \frac{1}{N^2}\sum_{xy}\sum_{ij} v_{xi}v_{yj}\right) - \frac{c}{N}\sum_{xi} v_{xi} + \frac{c}{2} \end{aligned}$$

[3]Note that $\mathbf{T}^{zs}$ is symmetric and since $\mathbf{T}^{zs}$ is a projection matrix, $\mathbf{T}^{zs}\mathbf{T}^{zs} = \mathbf{T}^{zs}$. Also, $\hat{\mathbf{e}}$ lies in the nullspace of $\mathbf{T}^{zs}$, so $\mathbf{T}^{zs}\hat{\mathbf{e}} = \mathbf{0}$, where $\mathbf{0}$ is the zero vector.

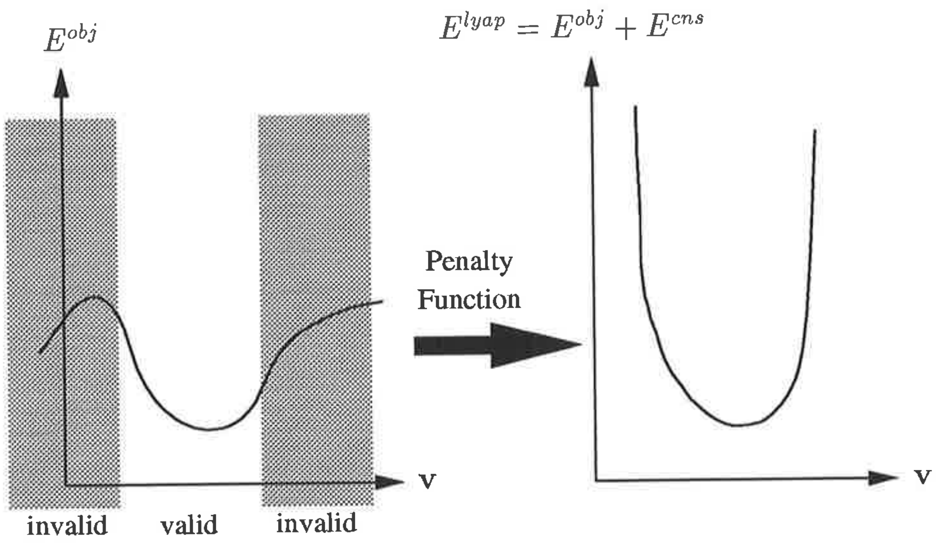


Figure 3.2: The valid subspace problem mapping in one dimension (Gee, 1993).

A simple optimisation problem where it is required to minimise the function E^{obj} over a limited range of validity. To map the problem onto an optimisation network it is necessary to set $E^{lyap} = E^{obj} + E^{cns}$, where E^{cns} is a suitably defined penalty function. With the valid subspace mapping the penalty function is zero over the valid subspace and grows rapidly as **v** *moves away from the valid subspace. Consequently* **v** *is encouraged to lie in the valid subspace.*

where we have used the convention that all summations are from 1 to N unless explicitly shown otherwise. To make further simplifications it should be noted that

$$\frac{1}{N}\sum_i\left(\sum_x v_{xi}-1\right)^2 = \frac{1}{N}\sum_i\sum_{xy} v_{xi}v_{yi} - \frac{2}{N}\sum_{xi} v_{xi}+1 \tag{3.11}$$

and

$$\frac{1}{N}\sum_x\left(\sum_i v_{xi}-1\right)^2 = \frac{1}{N}\sum_x\sum_{ij} v_{xi}v_{xj} - \frac{2}{N}\sum_{xi} v_{xi}+1. \tag{3.12}$$

Utilising equations (3.11) and (3.12), the penalty function E^{cns} may be reduced to

$$\begin{aligned} E^{cns} &= \frac{c}{2N}\sum_i\left(\sum_x v_{xi}-1\right)^2 + \frac{c}{2N}\sum_x\left(\sum_i v_{xi}-1\right)^2 \\ &\quad -\frac{c}{2}\left(\frac{2}{N}\sum_{xi} v_{xi} - \frac{1}{N^2}\sum_{xy}\sum_{ij} v_{xi}v_{yj} - 1\right) \\ &= \frac{c}{2N}\sum_i\left(\sum_x v_{xi}-1\right)^2 + \frac{c}{2N}\sum_x\left(\sum_i v_{xi}-1\right)^2 - \frac{c}{2N^2}\left(\sum_{xi} v_{xi}-N\right)^2 \end{aligned} \tag{3.13}$$

With the penalty function E^{cns} expressed as in equation (3.13), it is easy to see how the constraints for the TSP are enforced. The first two terms are zero when the column and row sums are equal to one, and positive otherwise. This clearly only penalises invalid states. These exact terms have been used in many mappings of the TSP onto an optimisation network (Brandt et al., 1988; Protzel et al., 1993; Abe, 1993; Abe and Gee, 1995; Matsuda, 1995). In fact, these two terms alone are sufficient to guarantee that **v** remains in the valid subspace (Aiyer, 1991). The final term is at first counter-intuitive. The minus sign indicates that it favours states where the sum of all elements is not equal to N. It is only the factor $1/N^2$ that prevents the third term from overpowering the constraining effects of the row and column sum terms[4]. Interestingly, the third term has a similar effect to the *global term* in Hopfield's original mapping of the TSP (Hopfield and Tank, 1985), where parameters were chosen to favour more than N neurons being switched on.

3.5 Eigenvector Analysis

So far we have relied on rather simple, but compelling arguments to see that the penalty functions defined by the valid subspace approach do indeed restrict the network state **v** to be valid. In this section we present a rigorous analysis that formalises much of the preceding intuitive argument.

If an optimisation network for the TSP is configured without the objective terms in the Lyapunov function, so that

$$\begin{aligned} E^{lyap} &= E^{cns} \\ &= -\frac{1}{2}\mathbf{v}^T\mathbf{T}^{cns}\mathbf{v} - \mathbf{v}^T\mathbf{b}^{cns} + \text{constant} \end{aligned} \tag{3.14}$$

[4]A deeper analysis, following the procedure presented in Appendix C.1, would reveal that the presence of the third term in E^{cns} acts to make equal the eigenvalues of the connection matrix in the invalid subspace and in the direction of $\hat{\mathbf{e}}$. In doing so, equal significance has been placed on the action of the penalty functions in both these directions.

we could appeal to the gradient descent arguments presented in Section 3.4 to see that $\mathbf{v}$ would be restricted to the valid subspace as desired. However, we will obtain further insight into the operation of a successful problem mapping by analysing the linearised dynamics of the network. While this analysis only slightly adds to our understanding of the penalty functions, it will form the basis for understanding the problems with annealing schemes, as presented in Chapter 4. To undertake a linearised analysis it is necessary to determine the eigenvectors and eigenvalues of the connection matrix $\mathbf{T}^{cns}$.

To do so, it is first necessary to determine an expression for the connection matrix $\mathbf{T}^{cns}$. By substituting equation (3.13) into equation (3.14) the connection matrix and external bias may be determined as

$$\begin{aligned} [\mathbf{T}^{cns}]_{xi,yj} &= -\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2} && (3.15) \\ \mathbf{b}^{cns} &= c\hat{\mathbf{e}}. && (3.16) \end{aligned}$$

As shown in Appendix C.1, the connection matrix $\mathbf{T}^{cns}$ given by equation (3.15) has the following eigenvalues:

$\lambda_1 = -c$
The corresponding eigenvector is $\hat{\mathbf{e}}$.

$\lambda_2 = 0$
This is a degenerate eigenvalue with the corresponding eigenspace being the zerosum subspace.

$\lambda_3 = -c$
This is also a degenerate eigenvalue with the corresponding eigenspace being the invalid subspace.

3.5.1 Analysis of the Dynamics of the Network

To further understand how the penalty functions restrict the network state to lie in the valid subspace, it is necessary to analyse the linearised dynamics of the network. The operation of the network is described by the nonlinear equation

$$\dot{\mathbf{u}} = \mathbf{T}^{cns}\mathbf{v} + \mathbf{b}^{cns}$$

where $v_i = g(u_i)$. In order to perform an eigenvector analysis, we must linearise the transfer functions $g(u_i)$ about an arbitrary point $\mathbf{v}^{(0)} = g(\mathbf{u}^{(0)})$. Linearised analysis about the point $\mathbf{v}^{(0)}$ gives sufficient information to approximate the network dynamics at this point. However, as the network dynamics cause $\mathbf{v}$ to move away from $\mathbf{v}^{(0)}$ the linearised analysis will become less accurate. To accurately approximate the network it then becomes necessary to linearise the dynamics about the current state $\mathbf{v}$. Linearisation of the transfer functions gives

$$\mathbf{v} = \mathbf{K}\mathbf{u} + \mathbf{d}$$

where $[\mathbf{K}]_{ij} = \begin{cases} v_i^{(0)}(1 - v_i^{(0)})/T^p & i = j \\ 0 & i \neq j \end{cases}$ and $d_i = v_i^{(0)} - u_i^{(0)}v_i^{(0)}(1 - v_i^{(0)})/T^p$. Therefore $\dot{\mathbf{v}} = \mathbf{K}\dot{\mathbf{u}}$ and the linearised network dynamics are given by

$$\dot{\mathbf{v}} = \mathbf{K}\left(\mathbf{T}^{cns}\mathbf{v} + \mathbf{b}^{cns}\right).$$

Using equation (3.6) to decompose $\mathbf{v}$ into its components,

$$\dot{\mathbf{v}} = \mathbf{K}(\lambda_1\mathbf{v}^1 + \lambda_2\mathbf{v}^{zs} + \lambda_3\mathbf{v}^{inv} + \mathbf{b}^{cns}). \tag{3.17}$$

Substituting $\mathbf{b}^{cns} = c\hat{\mathbf{e}}$, $\mathbf{v}^1 = (\mathbf{v}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}$ and explicit values for the eigenvalues gives

$$\begin{aligned}\dot{\mathbf{v}} &= \mathbf{K}\left\{(\lambda_1(\mathbf{v}\cdot\hat{\mathbf{e}}) + c)\hat{\mathbf{e}} + \lambda_2\mathbf{v}^{zs} + \lambda_3\mathbf{v}^{inv}\right\} \\ &= \mathbf{K}\left\{(c - c(\mathbf{v}\cdot\hat{\mathbf{e}}))\hat{\mathbf{e}} - c\mathbf{v}^{inv}\right\}.\end{aligned} \tag{3.18}$$

Obviously, when the network has converged $\dot{\mathbf{v}} = \mathbf{0}$, where $\mathbf{0}$ is the zero vector, and the linearised dynamics (3.18) for the network may now be examined to determine how the network pins $\mathbf{v}$ to the valid subspace.

Firstly, the component of $\dot{\mathbf{v}}$ in the invalid subspace is $-c\mathbf{K}\mathbf{v}^{inv}$. Since $\mathbf{K}$ is a diagonal matrix with non-negative elements, the component of $\mathbf{v}$ in the invalid subspace will decrease in magnitude, leading to $\|\mathbf{v}^{inv}\| = 0$.

Secondly, the net component of $\dot{\mathbf{v}}$ in the direction of $\hat{\mathbf{e}}$, will be $\mathbf{K}(c - c(\mathbf{v}\cdot\hat{\mathbf{e}}))\hat{\mathbf{e}}$. Therefore, the component in the direction of $\hat{\mathbf{e}}$ will move to a hyperplane where

$$\begin{aligned}c - c(\mathbf{v}\cdot\hat{\mathbf{e}}) &= 0 \\ \mathbf{v}\cdot\hat{\mathbf{e}} &= 1 \\ \Rightarrow \mathbf{v}^1 &= \hat{\mathbf{e}},\end{aligned}$$

as required for $\mathbf{v}$ to be valid. Obviously, with $\mathbf{v}^1 = \hat{\mathbf{e}}$ and $\|\mathbf{v}^{inv}\| = 0$ the network state has converged to the valid subspace. Moreover, in doing so there has been no influence on the component $\mathbf{v}^{zs}$ in the zerosum subspace, which is the component in a valid solution which determines the particular tour to be travelled in the TSP. Naturally, this should be expected, since we have considered only the penalty functions in this analysis and therefore do not want to influence the choice of a tour in the TSP. Only the distance terms in the objective function should influence the component $\mathbf{v}^{zs}$.

3.5.2 Analysis of the Lyapunov function

Further to the analysis of the linearised dynamics it is possible to reformulate the Lyapunov function in terms of the eigenvalues of the connection matrix. In doing so, it is clearly shown how the individual components of the network state $\mathbf{v}$ are affected by the energy minimisation nature of the network. By decomposing $\mathbf{v}$ into components as in equation (3.6), and omitting the uninteresting constant, the Lyapunov function (3.14) becomes

$$E^{lyap} = -\frac{1}{2}\left(\lambda_1\|\mathbf{v}^1\|^2 + \lambda_2\|\mathbf{v}^{zs}\|^2 + \lambda_3\|\mathbf{v}^{inv}\|^2\right) - \mathbf{v}^T\mathbf{b}^{cns}. \tag{3.19}$$

Furthermore, explicitly substituting for the eigenvalues and noting from equation (3.16) that $\mathbf{b}^{cns} = c\hat{\mathbf{e}}$, gives

$$E^{lyap} = \frac{c}{2}\|\mathbf{v}^1\|^2 + \frac{c}{2}\|\mathbf{v}^{inv}\|^2 - c\hat{\mathbf{e}}^T\mathbf{v}.$$

Since $\mathbf{v}^1 = (\mathbf{v}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}$ where $\hat{\mathbf{e}}$ is a unit vector, the norm of $\mathbf{v}^1$ is given by $\|\mathbf{v}^1\| = \mathbf{v}\cdot\hat{\mathbf{e}}$. Therefore

$$E^{lyap} = \frac{c}{2}\left\{(\|\mathbf{v}^1\| - 1)^2 + \|\mathbf{v}^{inv}\|^2 - 1\right\}.$$

As an optimisation network seeks out minima in the Lyapunov function, and since c is a positive constant, we would expect the network to converge to a state where

$$\|\mathbf{v}^1\| = 1 \qquad i.e.\ \mathbf{v}^1 = \hat{\mathbf{e}}$$

and

$$\|\mathbf{v}^{inv}\| = 0.$$

In doing so, the network will have converged to the valid subspace. Moreover, there has been no influence on the component $\mathbf{v}^{zs}$ in the zerosum subspace, which is to be expected for the reasons outlined in Section 3.5.1.

3.6 Mapping Some Common Problems

Optimisation networks have been used to solve many problems apart from the TSP: shortest Hamilton path, graph labelling and the assignment problem are among the more common. In this section we show how to map each of these problems onto an optimisation network. Firstly, it is necessary to give a concise problem statement and then define a suitable problem representation. To complete the mapping, an appropriate objective function is defined and the problem constraints are detailed.

Shortest Hamilton Path

The Euclidean, shortest Hamilton path problem (HPP) is simply stated as:

Given N cities in a plane, all of which must be visited once only, find the order in which to visit them such that the total distance travelled is minimised.

The HPP is quite similar to the travelling salesman problem: the only difference being that a Hamilton path does not return to the city at which it started. To solve this problem with an optimisation network, an $N \times N$ array of neurons is used, where the output of the neuron in row x and column i is one if city x is to be visited in the i^{th} position of the path, and zero otherwise. Denoting the distance between city x and city y as d_{xy} the objective function for the HPP is given by,

$$E^{obj} = \frac{1}{2} \sum_{x,y=1}^{N} \sum_{i=1}^{N} v_{xi} d_{xy} (v_{y,i-1} + v_{y,i+1}). \tag{3.20}$$

Note that, unlike the TSP, the indices are not evaluated modulo N and so E^{obj} calculates the length of an *open* tour. For the network state $\mathbf{v}$ to be a valid solution to the HPP, it is required that precisely one neuron be on in each row and column of the array. Accordingly, the constraints are exactly as given for the TSP in equations (3.3) and (3.4), and the appropriate penalty function is given by equation (3.13).

Graph Labelling

The graph labelling problem is central to several invariant pattern recognition systems (Mjolsness et al., 1991; Gee et al., 1993). The graph labelling problem is simply stated as:

Given two graphs $G_{\mathcal{P}}$ and $G_{\mathcal{Q}}$, find the relabelling of the nodes in graph $G_{\mathcal{P}}$ such that the relabelled graph best matches graph $G_{\mathcal{Q}}$.

We assume that both graphs have the same number of nodes N, and let the edge weight matrix for $G_{\mathcal{P}}$ be $\mathbf{P}$ and that of $G_{\mathcal{Q}}$ be $\mathbf{Q}$. Then the graph labelling problem is to relabel the nodes in $G_{\mathcal{P}}$, thereby giving a reordered edge weight matrix $\mathbf{P}'$, such that $\mathbf{P}'$ best matches $\mathbf{Q}$.

Once again to solve this problem with an optimisation network an $N \times N$ array of neurons is used, where the neuron in row i and column j has output denoted by v_{ij}. If $v_{ij} = 1$ then node i in $G_{\mathcal{P}}$ is matched with node j in $G_{\mathcal{Q}}$, and if $v_{ij} = 0$ such a labelling has not been made. A quantity which measures the dissimilarity of the relabelled $G_{\mathcal{P}}$ and the original $G_{\mathcal{Q}}$ is (Mjolsness et al., 1991)

$$E^{obj} = -2 \sum_{i,j=1}^{N} \sum_{x,y=1}^{N} Q_{ix} v_{ij} P_{jy} v_{xy}.$$

For a valid graph labelling, each node in $G_{\mathcal{P}}$ must be matched with a single node in $G_{\mathcal{Q}}$. Consequently, for $\mathbf{v}$ to represent a valid labelling, only one neuron must be on in each column an row of the array. The constraints are therefore the same as the TSP and a suitable penalty function is given by equation (3.13), with the appropriate change of indices.

Assignment Problem

The assignment problem has application in certain resource allocation problems (Brandt et al., 1988; Eberhardt et al., 1991; Tagliarini et al., 1991; Protzel et al., 1993), where it is desired to find the least costly one-to-one assignment or match between the elements of two lists. The lists may be thought of as resources $\mathcal{R} = \{A, B, C, \dots\}$ and consumers $\mathcal{C} = \{1, 2, 3 \dots\}$. We assume that each list contains a total of N elements. A one-to-one assignment means that each resource in the set $\mathcal{R}$ has to be assigned to exactly one consumer in the set $\mathcal{C}$. The cost p_{Xi} for every possible pairing between resource X and consumer i is given. The objective is to minimise the cost of the assignment of resources to consumers.

A suitable neural representation is a two dimensional array of neurons where the neuron in row X and column i has output denoted by v_{Xi}. If $v_{Xi} = 1$ then resource X has been assigned to consumer i, and if $v_{Xi} = 0$ the assignment of resource X to consumer i has not been made. A suitable objective function which gives the overall cost of a valid one-to-one assignment of resources to consumers, is

$$E^{obj} = \sum_{X=1}^{N} \sum_{i=1}^{N} v_{Xi} p_{Xi}.$$

Note that unlike the previous problems that we have mapped onto optimisation networks, the objective function for the assignment problem is linear: it is an example of linear $0-1$ programming and can be solved to optimality by many techniques including optimisation networks. For a valid one-to-one assignment, only one resource may be assigned to each consumer. Conversely, only one consumer can be assigned each resource. Therefore the constraints are given by,

$$\begin{aligned} \sum_{X=1}^{N} v_{Xi} &= 1 \qquad \forall\, i \in \{1, \dots N\} \\ \sum_{i=1}^{N} v_{Xi} &= 1 \qquad \forall\, X \in \{1, \dots N\}. \end{aligned}$$

These are the same constraints as given for the TSP, and so a suitable penalty function is given by equation (3.13) (again with the appropriate change of indices).

3.6.1 *Are they suitable?*

The standard optimisation networks, such as the continuous Hopfield network and the MFA algorithm, are not suited to the solution of every combinatorial optimisation problem. It is the aim of this section to highlight some of the factors which preclude a problem from being efficiently solved by an optimisation network. More detailed information may be obtained from the relevant literature.

It is interesting to note that the valid subspace is identical for the TSP, the shortest Hamilton path problem, graph labelling and the assignment problem. For these problems the valid subspace is in fact an *integral* polytope (Gee and Prager, 1994), as all the vertices of the polytope are valid $0-1$ points. Moreover, it has been shown that when the valid subspace is an integral polytope, a valid $0-1$ solution can always be found by employing the valid subspace mapping in conjunction with an annealing algorithm (Gee, 1993). Given the nature of the descent process that led to the solution, it is reasonable to expect that the solution will be quite good. Furthermore, as we show in Chapter 4, the use of an annealing algorithm enhances the expected quality of solution. Consequently, optimisation networks are well suited to solving these problems.

However, there do exist many problems whose constraints do not define an integral polytope and as such standard optimisation networks are not well suited to their solution. One important example is the knapsack problem:

> *Given a knapsack of capacity C and a set of N items each with size x_i and usefulness y_i, decide which items to put into the knapsack so as to obtain the maximum usefulness from the load, without overfilling the knapsack.*

The constraints for the knapsack problem define a non-integral polytope (*i.e.* not every vertex of the valid subspace is a valid $0-1$ point) and as such, the valid subspace mapping cannot guarantee that a valid $0-1$ solution will be found (Beyer and Ogier, 1991; Gee and Prager, 1994). Standard optimisation networks are not well suited to these problems. Modified networks, such as the tabu network (Gee, 1993; Gee and Prager, 1994), which are capable of searching the vertices of the valid subspace, are better suited to solution of combinatorial problems over non-integral polytopes.

3.7 Chapter Summary

This chapter presented a comprehensive investigation of problem mapping. While optimisation networks are well suited to the minimisation of a quadratic function E^{lyap} over a set of $0-1$ variables, most combinatorial optimisation problems are best expressed as quadratic $0-1$ programming, where it is required to minimise a quadratic objective function E^{obj}, subject to a set of linear constraints. When solving a combinatorial problem with an optimisation network, a successful problem mapping ensures that the solution obtained not only minimises the objective function, but that it also satisfies the problem's constraints. To do so the problem's constraints are reformulated as quadratic penalty functions E^{cns}, which penalise invalid states and attain their minimal value when the constraints are satisfied. The problem is then mapped onto the optimisation network by setting the Lyapunov function $E^{lyap} = E^{obj} + E^{cns}$.

It should be noted that many of the discouraging results attributed to optimisation networks are in fact a consequence of incorrect problem mappings. Only recently has problem mapping been placed onto a solid foundation with the development of the valid subspace approach (Gee, 1993; Gee et al., 1993). Such an approach recognises that any set of feasible linear constraints defines a bounded polyhedron, termed the valid subspace, in which $\mathbf{v}$ must lie if it is to be valid. Moreover, any vector which lies in the valid subspace may be written as

$$\mathbf{v} = \mathbf{T}^{zs}\mathbf{v} + \hat{\mathbf{e}}.$$

Consequently, an appropriate penalty function is easily defined as

$$E^{cns} = \frac{1}{2}c\|\mathbf{v} - (\mathbf{T}^{zs}\mathbf{v} + \hat{\mathbf{e}})\|^2.$$

A process of gradient descent on the penalty function naturally encourages $\mathbf{v}$ to lie on the valid subspace. Much of the motivation for developing the valid subspace approach arose from the belief that ad hoc approaches, typically employing one term in the penalty function for each constraint, suffered from a multiplicity of terms which tended to frustrate each other and could not guarantee a valid solution. However, the analysis presented in this chapter revealed that E^{cns} does itself consist of several terms, each designed to enforce a single constraint. In that sense, the valid subspace approach is not at all dissimilar to the previous ad hoc approaches.

To gain a deeper insight into the mechanics of the valid subspace approach to problem mapping, we undertook an eigenvector analysis of the linearised network dynamics. While this analysis is an original contribution, and interesting in its own right, an additional benefit is that it forms the basis for understanding the fundamental problems with annealing schemes, as presented in Chapter 4.

Finally, it was shown how to map the shortest Hamilton path problem, graph labelling and the assignment problem onto an optimisation network using the valid subspace mapping. The chapter concluded with a brief discussion to highlight that optimisation networks are not suitable for the solution of every combinatorial optimisation problem. However, optimisation networks are well suited to the solution of combinatorial problems whose constraints define an integral polytope *i.e.* all vertices of the valid subspace are valid $0-1$ points. While not all combinatorial problems satisfy this requirement, it is fortunate that many do.

Chapter IV

Annealing Techniques

It was shown in the previous chapter that by using a suitable problem mapping an optimisation network can always obtain a valid solution to a problem such as the TSP. Furthermore, given the nature of the descent process leading to such a solution, it could even be expected that the solution will be of an acceptable quality. Unfortunately, optimisation networks, as proposed in their original form, tend to converge to poor, high cost solutions. While the validity of the solutions may now be guaranteed by the problem mapping, the quality of the solutions remains unsatisfactory.

In this chapter we investigate *hysteretic annealing* which is a technique for improving the quality of the solutions found by an optimisation network. Hysteretic annealing seeks to guide the network state towards good solutions by continuously modifying the network's Lyapunov function. In doing so, annealing influences not only the number but also the position of attractors in the state space. While the annealed optimisation network still operates in the same gradient descent manner as the original network, it aims to give the network a better chance of converging to the global minimum of the objective function.

We begin in Section 4.1 by introducing the details of the hysteretic annealing technique. The ability of annealing to improve solution quality is demonstrated in Section 4.2, where hysteretic annealing is employed in a simple optimisation problem in two dimensions. While such a simple, illustrative example does not address the theoretical foundations of annealing, it serves to demonstrate the concept of hysteretic annealing and is sufficient for our purposes. In Section 4.3 we present an eigenvector analysis of the effect that annealing has on the linearised dynamics of an optimisation network. In doing so, it is shown that hysteretic annealing is not well formulated as it conflicts with the action of the problem mapping and may well lead to invalid solutions. Consequently, in Section 4.4 we develop a new, principled approach to hysteretic annealing that retains the ability to improve solution quality while ensuring that a valid solution can still be guaranteed. A further eigenvector analysis is presented in Section 4.5 to verify the correct operation of the modified hysteretic annealing approach. Finally, in Section 4.6 the performance of the standard and modified annealing algorithms are compared on travelling salesman problems of varying sizes.

4.1 Hysteretic Annealing

As discussed in Section 2.5.2, an annealing mechanism is used in an optimisation network to discourage the network from converging to a local minimum of the Lyapunov function. Such a local minimum may well represent a poor, high cost solution to the optimisation problem. By successfully avoiding local minima in the Lyapunov function, annealing can give remarkable improvements in solution quality. Various annealing mechanisms have been proposed: including convex relaxation (Ogier and Beyer, 1990), hysteretic annealing (Eberhardt et al., 1991) and matrix-graduated non-convexity (Aiyer, 1991). These techniques are all closely related and we shall collectively refer to them as hysteretic annealing.

Hysteretic annealing has been introduced in Section 2.5.2, but for completeness we shall again describe the technique. Hysteretic annealing involves adding the term

$$E^{ann} = -\frac{\gamma}{2}\sum_i (v_i - 0.5)^2$$

to the Lyapunov function E^{lyap} of the Hopfield network. It should be noted that when employing hysteretic annealing, the decay term η is commonly set to zero, and so the Lyapunov function for the network reduces to the quadratic function

$$E = -\frac{1}{2}\mathbf{v}^T\mathbf{T}\mathbf{v} - \mathbf{v}^T\mathbf{b}.$$

The function E^{ann} is either convex or concave, depending on the sign of the annealing parameter γ. Initially, γ is set to a large negative value, in which case $E^{lyap} = E + E^{ann}$ is convex and $\mathbf{v}$ converges to a point inside the hypercube. Subsequently, γ is gradually increased, and the deepest minima of the quadratic function E become evident in the Lyapunov function. As γ increases further, more minima become evident in the Lyapunov function. Eventually E^{lyap} becomes concave and $\mathbf{v}$ is pushed toward the boundary of the hypercube, aiding convergence to a $0-1$ point. While annealing is unable to *free* $\mathbf{v}$ from local minima in the Lyapunov function, it is hoped that local minima will be avoided by the annealing process and that the probability of converging to the globally optimal solution is increased. The ability of hysteretic annealing to improve solution quality will be demonstrated in Section 4.2.

Originally, annealing techniques were supported only by their experimentally perceived benefits, but recent theoretical analysis has done much to validate their use (Ohta et al., 1993; Abe and Gee, 1995; Tomikawa and Nakayama, 1995). Further to the ability to improve solution quality, hysteretic annealing offers important benefits for a hardware implementation. Hysteretic annealing allows explicit control over the dynamics of the network: in particular, the ability to control the time required to settle into a solution. Consequently, by using annealing techniques the sensitivity to variations in time constants of the various processors in the network is reduced. Such sensitivity to process variations could otherwise result in oscillations that render the network inoperative (Smith and Portmann, 1989). Hysteretic annealing can easily be incorporated into hardware realisations of optimisation networks (Lee and Sheu, 1993).

4.2 Why Use Annealing?

The ability of hysteretic annealing to guide $\mathbf{v}$ towards good solutions is best understood with the aid of a simple example. Consider the following problem:

$$\begin{array}{ll} \text{minimise} & E = -\frac{1}{2}\left(v_1^2 + v_2^2\right) + 4v_1v_2 - v_1 - 3v_2 \\ \text{subject to} & v_i \in \{0, 1\}. \end{array}$$

A contour plot of E is given in Figure 4.1. Note, that in addition to the global minimum at $\mathbf{v} = [0,\ 1]^T$ a sub-minimum exists at $\mathbf{v} = [1,\ 0]^T$. As optimisation networks operate by

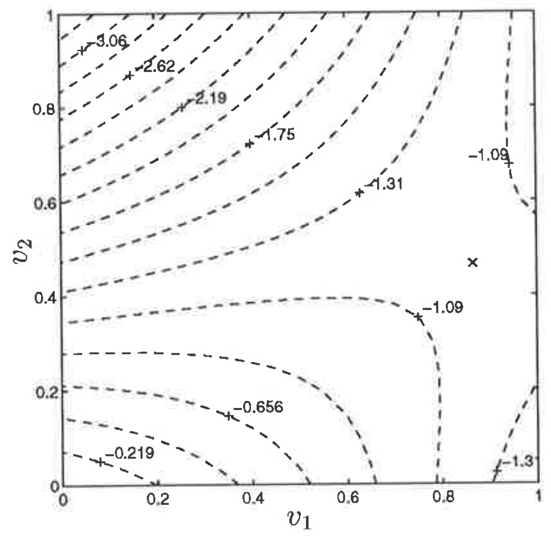


Figure 4.1: Contours of the quadratic objective function E.

The global minimum is $E([0,\ 1]^T) = -3.5$. *Note the saddle point at* $\mathbf{v} = [0.8667,\ 0.4667]^T$, *which has been marked with an* $\times$, *and the existence of the sub-minimum* $E([1,\ 0]^T) = -1.5$.

a process of gradient descent, it is clear that if $E^{lyap} = E$ then the final solution would depend on the initial conditions. Furthermore, if random initial conditions were used, there would be a high probability of converging to the sub-minimum at $\mathbf{v} = [1, 0]^T$.

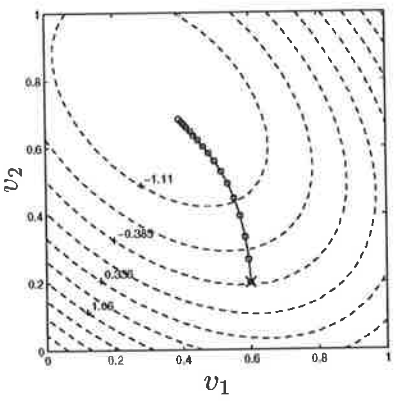
Hysteretic annealing may be used to overcome this limitation of optimisation networks by setting $E^{lyap} = E + E^{ann}$. The operation of the annealed optimisation network is shown in Figure 4.2. With γ set to a large negative value only one minimum exists in the Lyapunov function and all initial conditions lead to a state inside the unit square, as shown in Figure 4.2(a). As γ is slowly increased, the deepest minimum of E is the first to influence the dynamics and consequently $\mathbf{v}$ moves towards the global minimum at $\mathbf{v} = [0, 1]^T$ and eventually converges to it. Notice that if γ were to become sufficiently positive, then all vertices of the unit square would become local minima of E^{lyap} and the annealing is then effectively driving $\mathbf{v}$ to the nearest vertex. We see that annealing has influenced both the number and position of attractors in the Lyapunov function, and in doing so has improved the quality of solution obtained from the network.

4.3 Eigenvector Analysis

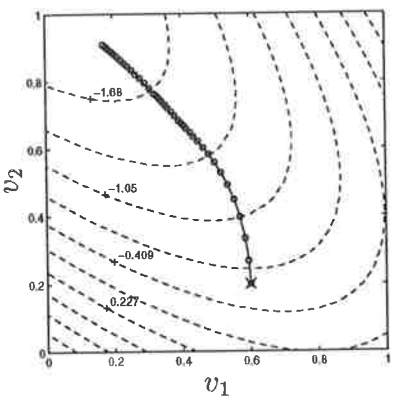
As demonstrated by the simple example in the previous section, hysteretic annealing is capable of improving the quality of solutions obtained from an optimisation network. However, the effect of annealing on the success of the problem mapping remains unknown. Can an optimisation network running under the valid subspace mapping guarantee that a valid solution will be found, even when hysteretic annealing is used? When hysteretic annealing is employed, does the network state remain pinned to the valid subspace?

To answer these questions it is necessary to analyse the linearised dynamics of an optimisation network. Accordingly, we examine an optimisation network used to solve the TSP, and arrange for the Lyapunov function to be

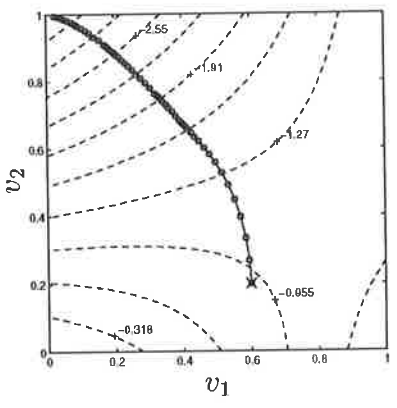
$$E^{lyap} = E^{cns} + E^{ann}$$



(a) $\gamma = -10$.



(b) $\gamma = -6$.



(c) $\gamma = 0$.

Figure 4.2: Snapshots of the operation of an annealed optimisation network.

The network is given the initial conditions $\mathbf{v} = [0.6,\ 0.2]^T$ *and the trajectory of* $\mathbf{v}$ *is clearly marked. The contour plots are of the Lyapunov function* $E^{lyap} = E + E^{ann}$, *given the current value of the annealing parameter* γ. *The annealing is started with* $\gamma = -10$ *causing* $\mathbf{v}$ *to converge to a point inside the hypercube as seen in (a). As* γ *is gradually increased,* $\mathbf{v}$ *moves toward the global minima at* $\mathbf{v} = [0, 1]^T$, *eventually converging to it as seen in (c). The gradient descent nature of the optimisation network and the ability of annealing to move* $\mathbf{v}$ *towards good solutions is clearly demonstrated. The network is simulated by integrating the Hopfield dynamics with the Euler method and a constant step-size* $\Delta t = 0.01$. *The annealing parameter* γ *is held constant until* $\|\Delta \mathbf{v}\| < 0.0001$, *at which time* γ *is increased by* $\Delta\gamma = 1$.

$$= -\frac{1}{2}\mathbf{v}^T\mathbf{T}^{ann}\mathbf{v} - \mathbf{v}^T\mathbf{b}^{ann} + \text{constant} \tag{4.1}$$

where the penalty term E^{cns} is given by equation (3.13), and the annealing term is given by

$$E^{ann} = -\frac{\gamma}{2}\sum_{xi}(v_{xi} - 0.5)^2. \tag{4.2}$$

The distance terms E^{obj} have been omitted to obtain a clearer insight into the mechanics of hysteretic annealing. Upon substitution, the Lyapunov function becomes

$$\begin{aligned} E^{lyap} &= \frac{c}{2N}\sum_i\left(\sum_x v_{xi} - 1\right)^2 + \frac{c}{2N}\sum_x\left(\sum_i v_{xi} - 1\right)^2 \\ &\quad -\frac{c}{2N^2}\left(\sum_{xi} v_{xi} - N\right)^2 - \frac{\gamma}{2}\sum_{xi}\left(v_{xi} - \frac{1}{2}\right)^2. \end{aligned} \tag{4.3}$$

By comparing equations (4.1) and (4.3) the connection matrix and external bias may be determined as

$$[\mathbf{T}^{ann}]_{xi,yj} = -\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2} + \gamma\delta_{xy}\delta_{ij} \tag{4.4}$$

$$\mathbf{b}^{ann} = \left(c - \frac{N\gamma}{2}\right)\hat{\mathbf{e}}. \tag{4.5}$$

In order to undertake the analysis of the linearised network dynamics it is necessary to determine the eigenvectors and eigenvalues of the connection matrix $\mathbf{T}^{ann}$. As shown in Appendix C.2, the connection matrix $\mathbf{T}^{ann}$ has the following eigenvalues:

$\lambda_1 = -c + \gamma$
The corresponding eigenvector is $\hat{\mathbf{e}}$.

$\lambda_2 = \gamma$
This is a degenerate eigenvalue with the corresponding eigenspace being the zerosum subspace.

$\lambda_3 = -c + \gamma$
This is also a degenerate eigenvalue with the corresponding eigenspace being the invalid subspace.

4.3.1 *Analysis of the Dynamics of the Network*

Now that the eigenvalues and eigenvectors of the connection matrix $\mathbf{T}^{ann}$ have been determined we may proceed with the analysis of the network's dynamics. By analogy with equation (3.17), the linearised dynamics may be expressed in terms of the eigenvectors of the connection matrix as

$$\dot{\mathbf{v}} = \mathbf{K}(\lambda_1\mathbf{v}^1 + \lambda_2\mathbf{v}^{zs} + \lambda_3\mathbf{v}^{inv} + \mathbf{b}^{ann}).$$

Substituting for $\mathbf{b}^{ann}$ from equation (4.5), and explicitly substituting the eigenvalues gives

$$\dot{\mathbf{v}} = \mathbf{K}\left\{(-c+\gamma)\mathbf{v}^1 + \gamma\mathbf{v}^{zs} + (-c+\gamma)\mathbf{v}^{inv} + \left(c - \frac{N\gamma}{2}\right)\hat{\mathbf{e}}\right\}.$$

Since $\mathbf{v}^1 = (\mathbf{v}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}$ this may be further simplified to

$$\dot{\mathbf{v}} = \mathbf{K}\left\{(c-\gamma)(1-(\mathbf{v}\cdot\hat{\mathbf{e}}))\hat{\mathbf{e}} - \frac{(N-2)\gamma}{2}\hat{\mathbf{e}} - (c-\gamma)\mathbf{v}^{inv} + \gamma\mathbf{v}^{zs}\right\}. \tag{4.6}$$

The effect of hysteretic annealing can now be easily explained. Firstly, the component of $\dot{\mathbf{v}}$ in the invalid subspace is $-(c-\gamma)\mathbf{K}\mathbf{v}^{inv}$. Since $\mathbf{K}$ is a diagonal matrix with non-negative elements, $\mathbf{v}^{inv}$ will decrease in magnitude if $\gamma < c$. As the annealing parameter γ is increased, the tendency for $\|\mathbf{v}^{inv}\| \to 0$ will be lessened. Indeed, if the annealing parameter is increased to such a point that $\gamma > c$, then $\mathbf{v}^{inv}$ will increase in magnitude, but such a situation can be avoided by properly setting the weight c on the penalty functions.

The effect of hysteretic annealing in the valid subspace is most easily understood with reference to Figure 4.3. E^{ann} is spherically symmetrical about the point $(0.5, 0.5, \ldots)$ and will therefore push $\mathbf{v}$ radially towards or away from the centre of the hypercube, depending on the sign of γ. With γ positive, annealing pushes any point on the valid subspace radially away from $(0.5, 0.5, \ldots)$, as indicated by the vector $\dot{\mathbf{v}}$, which may then be decomposed into a component $\dot{\mathbf{v}}^{zs}$ in the zerosum subspace and a component $\dot{\mathbf{v}}^1$ in the direction of $\hat{\mathbf{e}}$.

In order to improve solution quality, annealing must control the component of $\mathbf{v}$ in the zerosum subspace, as it is exactly this component which determines the particular tour to be travelled in the TSP and hence determines the solution quality. The component $\dot{\mathbf{v}}^{zs}$ of the dynamics, which has been introduced by annealing, is necessary to achieve such control. When considered in conjunction with the objective function E^{obj} and a process of gradually increasing γ, $\dot{\mathbf{v}}^{zs}$ will guide $\mathbf{v}$ towards good solutions (Ohta et al., 1993). If γ is positive, $\dot{\mathbf{v}}^{zs}$ aids convergence to a $0-1$ point.

The component of $\dot{\mathbf{v}}$ in the direction of $\hat{\mathbf{e}}$ comes under the influence of two forces, as seen from equation (4.6). Firstly, if $\gamma < c$ the term $(c-\gamma)(1-(\mathbf{v}\cdot\hat{\mathbf{e}}))\hat{\mathbf{e}}$ will move $\mathbf{v}$ towards the hyperplane where

$$\begin{aligned} 1-(\mathbf{v}\cdot\hat{\mathbf{e}}) &= 0 \\ \mathbf{v}\cdot\hat{\mathbf{e}} &= 1 \\ \Rightarrow \mathbf{v}^1 &= \hat{\mathbf{e}}. \end{aligned}$$

as required for $\mathbf{v}$ to be valid. However, as γ is increased during the annealing process, the tendency to confine $\mathbf{v}$ to the hyperplane where $\mathbf{v}^1 = \hat{\mathbf{e}}$ is reduced. The second influence is the component $-\frac{N-2}{2}\gamma\hat{\mathbf{e}}$, which will tend to increase the magnitude of the component $\mathbf{v}^1$ if γ is negative, and decrease it for positive γ. The net result is that the component of $\mathbf{v}$ in the direction of $\hat{\mathbf{e}}$ is not correctly constrained to make $\mathbf{v}$ lie on the valid subspace.

We have shown that while annealing has introduced the necessary components into the dynamics to allow the solution quality to be improved, it has also introduced a conflict between the penalty functions, which attempt to restrict $\mathbf{v}$ to the valid subspace, and the annealing which is attempting to push $\mathbf{v}$ off the valid subspace.

4.3.2 Analysis of the Lyapunov function

Further to the analysis of the dynamics, it is possible to express the Lyapunov function as given by equation (4.1), in terms of the eigenvalues of the connection matrix. By decomposing $\mathbf{v}$ into components as in equation (3.6), and omitting the uninteresting

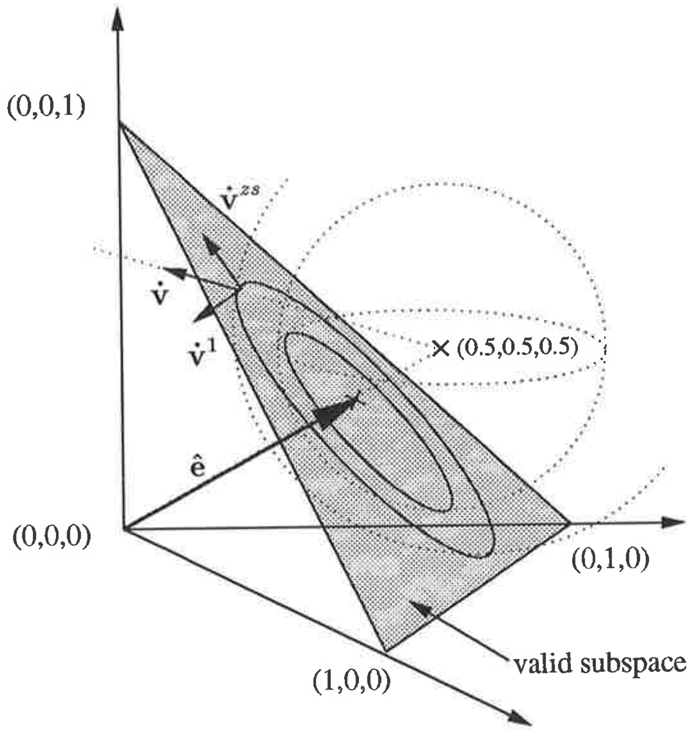


Figure 4.3: Effect of hysteretic annealing on the dynamics in the valid subspace.

The simple 3-dimensional example given in Section 3.3 has been chosen to illustrate the effect of hysteretic annealing. E^{ann} *is spherically symmetrical about the point* $(0.5, 0.5, \ldots)$ *and with* $\gamma > 0$ *will therefore push* $\mathbf{v}$ *radially away from the centre of the hypercube, as indicated by* $\dot{\mathbf{v}}$. $\dot{\mathbf{v}}$ *may be decomposed into a component* $\dot{\mathbf{v}}^{zs}$ *in the zerosum subspace and a component* $\dot{\mathbf{v}}^{1}$ *in the direction of* $\hat{\mathbf{e}}$. *The component* $\dot{\mathbf{v}}^{1}$ *tends to force* $\mathbf{v}$ *off the valid subspace, in conflict with the penalty terms which are attempting to confine* $\mathbf{v}$ *to the valid subspace.*

constant the Lyapunov function becomes

$$E^{lyap} = -\frac{1}{2}\left(\lambda_1\|\mathbf{v}^1\|^2 + \lambda_2\|\mathbf{v}^{zs}\|^2 + \lambda_3\|\mathbf{v}^{inv}\|^2\right) - \mathbf{v}^T\mathbf{b}^{ann}.$$

Substituting for the eigenvalues and using equation (4.5) for $\mathbf{b}^{ann}$,

$$E^{lyap} = -\frac{1}{2}\left\{(-c+\gamma)\left(\|\mathbf{v}^1\|^2 + \|\mathbf{v}^{inv}\|^2\right) + \gamma\|\mathbf{v}^{zs}\|^2\right\} - \left(c - \frac{N\gamma}{2}\right)\hat{\mathbf{e}}^T\mathbf{v}.$$

Further substituting $\hat{\mathbf{e}}^T\mathbf{v} = \|\mathbf{v}^1\|$ and simplifying, gives

$$E^{lyap} = \frac{c-\gamma}{2}\left\{(\|\mathbf{v}^1\| - 1)^2 + \|\mathbf{v}^{inv}\|^2 - 1\right\} + \frac{(N-2)\gamma}{2}\|\mathbf{v}^1\| - \frac{\gamma}{2}\|\mathbf{v}^{zs}\|^2. \quad (4.7)$$

As an optimisation network will seek out minima in the Lyapunov function, it is worthwhile establishing the conditions on the various components of $\mathbf{v}$ that correspond to a minimum of E^{lyap}. To minimise E^{lyap} it is necessary for,

1. $\|\mathbf{v}^{inv}\| \to 0$.
2. If $\gamma < 0$ then $\|\mathbf{v}^{zs}\| \to 0$, else if $\gamma > 0$ then $\|\mathbf{v}^{zs}\| \to \infty$, although it should be noted that $\mathbf{v}^{zs}$ cannot continue to grow indefinitely as $\mathbf{v}$ is restricted to the unit hypercube.
3. $\mathbf{v}^1$ must be set so that $\dfrac{\partial E^{lyap}}{\partial\|\mathbf{v}^1\|} = 0$, *i.e.*

$$\begin{aligned} (c-\gamma)(\|\mathbf{v}^1\| - 1) + \frac{(N-2)\gamma}{2} &= 0 \\ \therefore \quad \|\mathbf{v}^1\| = 1 - \frac{(N-2)\gamma}{2(c-\gamma)}. & \end{aligned} \quad (4.8)$$

Given that $\mathbf{v}$ will converge to a local minimum of the Lyapunov function, further evidence of conflict between the penalty functions and annealing is found by examining the third condition from above. For $\mathbf{v}$ to be valid $\mathbf{v}^1 = \hat{\mathbf{e}}$ *i.e.* $\|\mathbf{v}^1\| = 1$; but an annealed optimisation network with $\gamma < 0$ will converge to a state where $\|\mathbf{v}^1\| > 1$ and $\mathbf{v}$ lies above the valid subspace. Similarly for $\gamma > 0$, $\|\mathbf{v}^1\| < 1$ and $\mathbf{v}$ lies beneath the valid subspace. The magnitude of $\mathbf{v}^1$ corresponding to a minimum of the Lyapunov function, as given by equation (4.8), is shown in Figure 4.4 for various values of γ and N.

While hysteretic annealing may be used to improve the quality of solutions obtained by an optimisation network, this section has shown that it also removes $\mathbf{v}$ from the valid subspace. At worst, this could lead to invalid solutions, a situation that has until now been avoided by careful selection of parameters. It is also apparent that having pushed $\mathbf{v}$ from the valid subspace, the operation of the optimisation network is no longer principled, as we are attempting to minimise E^{obj} with an invalid representation for $\mathbf{v}$. Consequently, it is necessary to reappraise our approach to annealing.

4.4 Modified Hysteretic Annealing

Although hysteretic annealing can lead to improved solution quality, the previous section showed that it invalidates the use of an optimisation network as it forces $\mathbf{v}$ off the

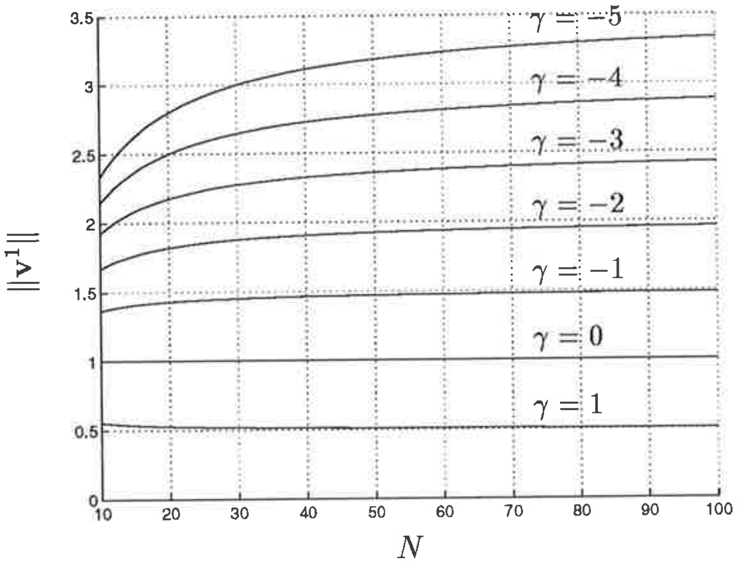


Figure 4.4: The magnitude of $\mathbf{v}^1$ necessary to minimise E^{lyap}.

When $E^{lyap} = E^{cns} + E^{ann}$, the magnitude of $\mathbf{v}^1$ necessary to minimise E^{lyap} is given by equation (4.8). The plots show $\|\mathbf{v}^1\|$ for various values of the problem size N and annealing parameter γ. Note that for $\mathbf{v}$ to be valid $\|\mathbf{v}^1\| = 1$, and so it is obvious that annealing has forced $\mathbf{v}$ off the valid subspace. The weight on the penalty functions was set so $c = N$, which is consistent with suggestions in (Abe and Gee, 1995).

valid subspace. Consequently, the optimisation network is attempting to minimise E^{obj} without a valid representation for $\mathbf{v}$. Therefore, it is necessary to reformulate annealing so that the ability to guide $\mathbf{v}$ toward good solutions is retained, whilst ensuring that $\mathbf{v}$ will remain on the valid subspace.

With the insight gained in Section 4.3 we can see that a correctly formulated approach to hysteretic annealing must have three essential features,

1. It should introduce a non-zero eigenvalue in the zerosum subspace that is controlled by the value of the annealing parameter γ. To encourage good solutions, annealing should be able to influence the component $\mathbf{v}^{zs}$ of a valid solution $\mathbf{v}$, as it is this component which decides the particular tour to be travelled in the TSP.

2. As the penalty terms alone correctly constrain $\mathbf{v}$ to be valid, annealing should not interfere with the operation of the penalty terms. Therefore, hysteretic annealing should not modify the eigenvalues of the connection matrix associated with the penalty terms, in the invalid subspace and in the direction of $\hat{\mathbf{e}}$.

3. The relative values of E^{lyap} when evaluated at valid $0-1$ points should not be altered. As the Lyapunov function at a valid $0-1$ point reflects the cost of the solution, the relative ordering of these points should be maintained.

As a candidate for a correctly formulated approach to annealing, consider the modified hysteretic annealing function given by

$$\begin{aligned} E^{mod} &= -\frac{\gamma}{2}\mathbf{v}^{zs\,T}\mathbf{I}\mathbf{v}^{zs} \\ &= -\frac{\gamma}{2}\mathbf{v}^{T}\mathbf{T}^{zs}\mathbf{v} \end{aligned} \tag{4.9}$$

where the substitution $\mathbf{v}^{zs} = \mathbf{T}^{zs}\mathbf{v}$ has been made[1]. Obviously such a function will allow annealing to influence the component of $\mathbf{v}$ in the zerosum subspace. Additionally, it will not conflict with the penalty functions, as it has no effect on the components of $\mathbf{v}$ in the invalid subspace and in the direction of $\hat{\mathbf{e}}$. Thirdly, from equation (4.9) it can be seen that $E^{mod} = -\frac{\gamma}{2}\|\mathbf{v}^{zs}\|^2$ and since for all valid $0-1$ points $\|\mathbf{v}^{zs}\|$ is identical, E^{mod} maintains the relative ordering of valid $0-1$ points. Therefore, the modified hysteretic annealing function satisfies the criteria for a correctly formulated approach to annealing and deserves further detailed investigation.

To clearly illustrate the utility of the modified annealing function, it is necessary to obtain an alternative expression for E^{mod}. Substituting equation (3.10) for the projection matrix $\mathbf{T}^{zs}$and expanding gives

$$\begin{aligned} E^{mod} &= -\frac{1}{2}\gamma\sum_{xi}\sum_{yj}\left(\delta_{xy}-\frac{1}{N}\right)\left(\delta_{ij}-\frac{1}{N}\right)v_{xi}v_{yj} \\ &= -\frac{1}{2}\gamma\left(\sum_{xi}v_{xi}^2-\frac{1}{N}\sum_{x}\sum_{ij}v_{xi}v_{xj}-\frac{1}{N}\sum_{i}\sum_{xy}v_{xi}v_{yi}+\frac{1}{N^2}\sum_{xi}\sum_{yj}v_{xi}v_{yj}\right). \end{aligned}$$

Using equations (3.11) and (3.12) this simplifies to

$$E^{mod} = \frac{\gamma}{2N}\sum_{x}\left(\sum_{i}v_{xi}-1\right)^2+\frac{\gamma}{2N}\sum_{i}\left(\sum_{x}v_{xi}-1\right)^2$$

[1]Once again, note that $\mathbf{T}^{zs}$ is symmetric and since $\mathbf{T}^{zs}$ is a projection matrix, $\mathbf{T}^{zs}\mathbf{T}^{zs} = \mathbf{T}^{zs}$.

$$
\begin{aligned}
& -\frac{\gamma}{2}\left(\frac{1}{N^2}\sum_{xi}\sum_{yj} v_{xi}v_{yj} - \frac{2}{N}\sum_{xi} v_{xi} + 1\right) - \frac{\gamma}{2}\left(\sum_{xi} v_{xi}^2 - \frac{2}{N}\sum_{xi} v_{xi} + 1\right) \\
= & \frac{\gamma}{2N}\sum_{x}\left(\sum_{i} v_{xi} - 1\right)^2 + \frac{\gamma}{2N}\sum_{i}\left(\sum_{x} v_{xi} - 1\right)^2 \\
& -\frac{\gamma}{2N^2}\left(\sum_{xi} v_{xi} - N\right)^2 - \frac{\gamma}{2}\sum_{xi}\left(v_{xi} - \frac{1}{N}\right)^2 . \qquad (4.10)
\end{aligned}
$$

Noting the similarity between the first three terms of the modified hysteretic annealing and the penalty functions in equations (3.13) and (3.7), allows E^{mod} to be written

$$
E^{mod} = \frac{1}{2}\gamma\|\mathbf{v} - (\mathbf{T}^{zs}\mathbf{v} + \hat{\mathbf{e}})\|^2 - \frac{\gamma}{2}\sum_{xi}\left(v_{xi} - \frac{1}{N}\right)^2 . \qquad (4.11)
$$

The function of the modified annealing term is now transparent. The first term in equation (4.11) is similar to the penalty term E^{cns}, where the weight c has been replaced by the annealing parameter γ, and serves to restrict $\mathbf{v}$ to the valid subspace. The second term in equation (4.11) is similar to the original hysteretic annealing term, but is now spherically symmetrical about $\hat{\mathbf{e}}$, which lies in the centre of the valid subspace. This is the only term in E^{mod} which influences the component of $\mathbf{v}$ in the zerosum subspace. It makes sense to have the term which controls $\mathbf{v}^{zs}$ centred in the valid subspace, rather than offset from the valid subspace, as it is in the original formulation of hysteretic annealing.

For any point $\mathbf{v}$ that does not lie on the valid subspace, there is a delicate balance between the penalty term in E^{mod}, which forces $\mathbf{v}$ back to the valid subspace and the expansive influence of the second term, which is attempting to move $\mathbf{v}$ further from the valid subspace. The net effect is that only $\mathbf{v}^{zs}$ is influenced by the modified annealing, leaving the components $\mathbf{v}^{inv}$ and $\mathbf{v}^1$ to be determined by the penalty functions E^{cns}.

4.5 Eigenvector Analysis

When the modified hysteretic annealing is employed, does the network state remain pinned to the valid subspace? Does modified hysteretic annealing incorporate the necessary components into the dynamics to improve solution quality? To answer these questions we will once again analyse the linearised dynamics of an optimisation network used to solve the TSP. Consider an optimisation network with the Lyapunov function

$$
\begin{aligned}
E^{lyap} &= E^{cns} + E^{mod} \\
&= -\frac{1}{2}\mathbf{v}^T\mathbf{T}^{mod}\mathbf{v} - \mathbf{v}^T\mathbf{b}^{mod} + \text{constant}
\end{aligned}
$$

where the penalty term E^{cns} is given by equation (3.13), and the annealing term is given by equation (4.10). Upon substitution, the Lyapunov function becomes

$$
\begin{aligned}
E^{lyap} = & \frac{(c+\gamma)}{2N}\sum_{i}\left(\sum_{x} v_{xi} - 1\right)^2 + \frac{(c+\gamma)}{2N}\sum_{x}\left(\sum_{i} v_{xi} - 1\right)^2 \\
& -\frac{(c+\gamma)}{2N^2}\left(\sum_{xi} v_{xi} - N\right)^2 - \frac{\gamma}{2}\sum_{xi}\left(v_{xi} - \frac{1}{N}\right)^2 .
\end{aligned}
$$

By analogy with equations (4.1), (4.3), (4.4) and (4.5) the connection matrix and external bias may be determined as

$$\begin{aligned}
[\mathbf{T}^{mod}]_{xi,yj} &= -\frac{(c+\gamma)}{N}\delta_{ij} - \frac{(c+\gamma)}{N}\delta_{xy} + \frac{(c+\gamma)}{N^2} + \gamma\delta_{xy}\delta_{ij} && (4.12)\\
\mathbf{b}^{mod} &= c\hat{\mathbf{e}}. && (4.13)
\end{aligned}$$

In order to analyse the linearised dynamics it is necessary to determine the eigenvectors and eigenvalues of the connection matrix $\mathbf{T}^{mod}$. As shown in Appendix C.3 the connection matrix $\mathbf{T}^{mod}$ has the following eigenvalues:

$\lambda_1 = -c$
The corresponding eigenvector is $\hat{\mathbf{e}}$.

$\lambda_2 = \gamma$
This is a degenerate eigenvalue with the corresponding eigenspace being the zerosum subspace.

$\lambda_3 = -c$
This is also a degenerate eigenvalue with the corresponding eigenspace being the invalid subspace.

By analogy with equation (3.17), the linearised dynamics may be expressed in terms of the eigenvectors of the connection matrix as

$$\dot{\mathbf{v}} = \mathbf{K}(\lambda_1\mathbf{v}^1 + \lambda_2\mathbf{v}^{zs} + \lambda_3\mathbf{v}^{inv} + \mathbf{b}^{mod}).$$

Substituting for $\mathbf{b}^{mod}$ from equation (4.13), and explicitly substituting the eigenvalues gives

$$\dot{\mathbf{v}} = \mathbf{K}\left\{-c\mathbf{v}^1 + \gamma\mathbf{v}^{zs} - c\mathbf{v}^{inv} + c\hat{\mathbf{e}}\right\}.$$

Further substituting $\mathbf{v}^1 = (\mathbf{v}\cdot\hat{\mathbf{e}})\hat{\mathbf{e}}$ gives

$$\dot{\mathbf{v}} = \mathbf{K}\left\{c(1-(\mathbf{v}\cdot\hat{\mathbf{e}}))\hat{\mathbf{e}} - c\mathbf{v}^{inv} + \gamma\mathbf{v}^{zs}\right\}. \qquad (4.14)$$

When the network has converged $\dot{\mathbf{v}} = \mathbf{0}$, where $\mathbf{0}$ is the zero vector and the success of the modified annealing approach may be determined by examining the linearised dynamics given by equation (4.14). Firstly, it should be noted that the component of $\dot{\mathbf{v}}$ in the invalid subspace is $-c\mathbf{K}\mathbf{v}^{inv}$. Since $\mathbf{K}$ is a diagonal matrix with non-negative elements, the component of $\mathbf{v}$ in the invalid subspace will decrease in magnitude, leading to $\|\mathbf{v}^{inv}\| = 0$.

Secondly, the net component of $\dot{\mathbf{v}}$ in the direction of $\hat{\mathbf{e}}$, will be $\mathbf{K}(c - c(\mathbf{v}\cdot\hat{\mathbf{e}}))\hat{\mathbf{e}}$. Therefore, the component in the direction of $\hat{\mathbf{e}}$ will move to a hyperplane where

$$\begin{aligned}
c - c(\mathbf{v}\cdot\hat{\mathbf{e}}) &= 0\\
\mathbf{v}\cdot\hat{\mathbf{e}} &= 1\\
\Rightarrow \mathbf{v}^1 &= \hat{\mathbf{e}},
\end{aligned}$$

as required for $\mathbf{v}$ to be valid. Obviously, with $\mathbf{v}^1 = \hat{\mathbf{e}}$ and $\|\mathbf{v}^{inv}\| = 0$ the network state has converged to the valid subspace. While annealing now influences $\mathbf{v}$ only in the zerosum subspace, the penalty functions control $\mathbf{v}$ only in the invalid subspace and in the direction of $\hat{\mathbf{e}}$. The actions of annealing and the penalty functions have been effectively decoupled into separate subspaces, thereby resolving the apparent conflict between hysteretic annealing and the penalty functions that was identified in Section 4.3.

The effect of modified hysteretic annealing in the valid subspace is illustrated in Figure 4.5. With reference to equation (4.10), the penalty-like terms in E^{mod} are zero on the valid subspace, while the remaining term in E^{mod} is spherically symmetrical about $\hat{\mathbf{e}}$ and will push $\mathbf{v}$ radially away or towards $\hat{\mathbf{e}}$ depending on the sign of γ. For positive γ the effect of E^{mod} on a valid $\mathbf{v}$ is shown as $\dot{\mathbf{v}}$. Obviously $\dot{\mathbf{v}} = \dot{\mathbf{v}}^{zs}$ as it lies wholly within the zerosum subspace. The effect of modified annealing is simply to guide $\mathbf{v}$ through the valid subspace by influencing the component $\mathbf{v}^{zs}$. Unlike the original hysteretic annealing, there is no tendency to push $\mathbf{v}$ from the valid subspace, avoiding any potential conflict with the penalty functions. As discussed in Section 4.3, the component of the dynamics $\dot{\mathbf{v}}^{zs}$ which has been introduced by annealing, when considered in conjunction with the objective function and a process of gradually increasing γ, will guide $\mathbf{v}$ towards good solutions.

The modified approach to hysteretic annealing proposed in equation (4.11) has all the essential features of a correctly formulated annealing mechanism. It retains the ability to guide $\mathbf{v}$ toward good solutions, by introducing a non-zero eigenvalue in the zerosum subspace. It also avoids any conflict with the penalty functions by effectively decoupling the actions of annealing and the penalty functions into different subspaces. As a result, $\mathbf{v}$ will remain in the valid subspace throughout the annealing process, which is a vital requirement for any optimisation network to operate correctly.

4.6 Simulations

To evaluate the performance of the standard and modified annealing algorithms, both algorithms were simulated on several Euclidean travelling salesman problems. The problems considered had 10, 30 and 50 cities placed randomly using a uniform distribution over the unit square. The optimised step-size technique as reported in (Abe, 1996) was used in all experiments. While more detailed information on the optimised step-size technique may be found in Appendix B, it is sufficient to note that such an algorithm holds the annealing parameter constant at γ_0 for the first t_d time steps, and then increments the annealing parameter by an amount $\Delta\gamma$ at successive time steps. In all simulations reported here, the penalty weight c is set so that

$$c = c_0 N \max_{x,y,z}(d_{xy} + d_{xz}),$$

as suggested by results in (Abe and Gee, 1995). It should be noted that all simulations converged to valid solutions, as a result of setting c_0 to an appropriately large value. The initial conditions are given by $v_{xi} = 1/N + 0.01 * \texttt{rand}$ where rand is a random value in the range $[-0.5, 0.5]$.

The results for the standard and modified annealing algorithms are shown in Tables 4.1, 4.2 and 4.3 and may be compared to the results from several well-known heuristics operating on the same set of problems, as given in Table 4.4. As a measure of the quality of solutions obtained from the optimisation networks, the mean percentage above the minimal tour length has been calculated. The minimal length tour is defined as the shortest tour found by any of the heuristic methods. Additionally, the number of times the optimisation networks found the minimal length tour is also recorded. It is apparent that optimisation networks give a similar quality of solution to the nearest neighbour techniques. This is consistent with the general trade-off between time and quality of

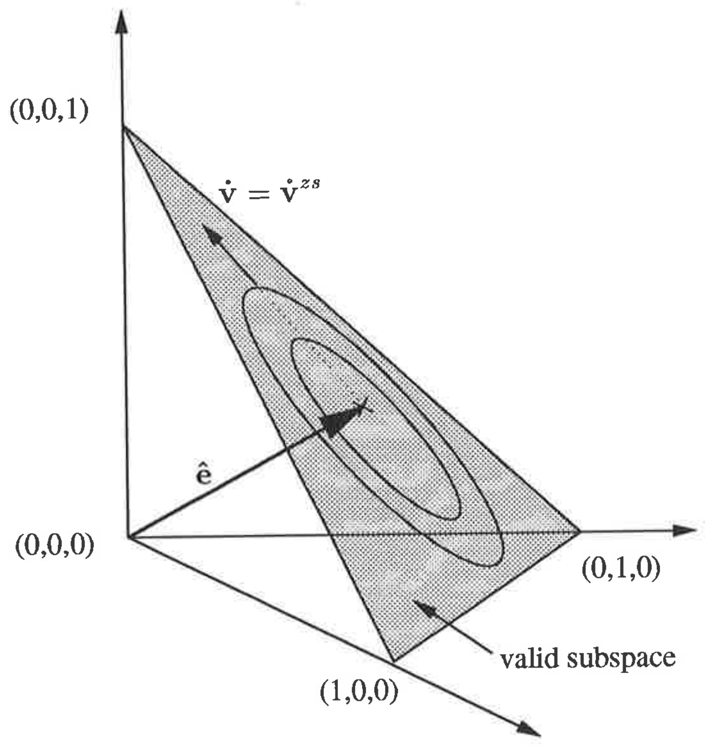


Figure 4.5: Action of modified annealing in the valid subspace.

The simple 3-D example given in Section 3.3 has been chosen to illustrate the effect of the modified hysteretic annealing in the valid subspace. With the annealing parameter $\gamma > 0$, the modified annealing will push a point in the valid subspace radially away from $\hat{\mathbf{e}}$. It should be noted that $\dot{\mathbf{v}}$ lies wholly within the valid subspace. Consequently, modified hysteretic annealing serves only to move $\mathbf{v}$ through the valid subspace and does not conflict with the penalty terms which are attempting to confine $\mathbf{v}$ to the valid subspace.

	Standard		Modified	
γ_0	mean %	no. min.	mean %	no. min.
-1	**0.47**	95	**0.47**	95
-1.5	0.39	95	**0.21**†	97
-2	**1.74**	82	1.91	82
-2.5	1.78	85	**1.64**	83
-3	1.49	83	**1.22**	88

Table 4.1: Simulation results for the 10-city problem.

The table shows the mean percentage above the minimal tour length of 2.9933 *(as found by heuristic methods - see Table 4.4), and the number of minimal length tours for various initial values of* γ_0 *(100 trials,* $\Delta\gamma = 0.005$, $c_0 = 2$, $t_d = 200$*).* †*Corresponds to a mean tour length of* 2.9996.

solution, as faster solution methods generally produce results of worse quality than more sophisticated techniques.

Several issues must be considered when comparing the standard and modified approaches to annealing. Firstly, the analysis of the preceding sections has shown that both annealing algorithms provide the ability to guide $\mathbf{v}$ towards good solutions. However, while modified annealing cannot cause invalid solutions, standard hysteretic annealing may potentially cause the network to converge to invalid solutions by removing $\mathbf{v}$ from the valid subspace. In our simulations, we have avoided this scenario by carefully selecting the weight on the penalty functions, thereby allowing the annealing methods to be compared on the quality of solution alone.

Once the potential for invalid solutions is removed, we may compare the annealing algorithms based solely on their quality of solution. It should be remembered that while the modified annealing approach does not remove $\mathbf{v}$ from the valid subspace, standard annealing does and so is attempting to minimise E^{obj} with an invalid representation for $\mathbf{v}$. While both the standard and modified annealing algorithms achieve similar results, as given in Tables 4.1, 4.2 and 4.3, it is clear that the modified annealing algorithm has a slightly superior quality of solution. The similarity between the two algorithms is not unexpected, as both possess the ability to guide $\mathbf{v}$ towards good solutions. Additionally, the distinction between the standard and modified annealing algorithms is blurred by the action of the objective function E^{obj} which itself tends to push $\mathbf{v}$ from the valid subspace.

The results presented in this section support modified annealing as being superior to the standard hysteretic annealing algorithm. Not only does the modified algorithm remove the possibility of annealing causing invalid solutions, it has been demonstrated to produce a better quality of solution.

It should be noted that the results for the optimisation networks may be improved by employing a more gradual annealing schedule. Alternatively, improvements may be made by reducing the weight c on the penalty functions (Abe, 1993), possibly at the expense of guaranteed convergence to a valid solution. It is also interesting to note the obvious deterioration in solution quality as the problem size increases. For the 10-city problem the best results from an optimisation network gave a mean tour length just 0.21% above that found by heuristic methods, while for the 50-city problem the corresponding result is 19.99%. While such a degrading quality of solution with increasing problem size is typical for optimisation networks, it has received little attention in the literature (Cooper, 1995b) and will therefore be considered in detail in Chapter 5.

	Standard		Modified	
γ_0	mean %	no. min.	mean %	no. min.
-1	13.73	7	**12.39**	11
-2	7.98	16	**7.91**†	22
-3	**8.66**	9	8.75	11
-4	**9.14**	4	12.42	2
-5	14.24	17	**11.14**	37
-6	14.81	21	**14.03**	32

Table 4.2: Simulation results for the 30-city problem.

The table shows the mean percentage above the minimal tour length of 4.4714 *(as found by heuristic methods - see Table 4.4), and the number of minimal length tours for various initial values of* γ_0 *(100 trials,* $\Delta\gamma = 0.005$, $c_0 = 2.5$, $t_d = 1500$*). †Corresponds to a mean tour length of* 4.8251.

	Standard	Modified
γ_0	mean %	mean %
-2	22.65	**21.84**
-3	21.43	**20.24**
-4	21.89	**19.99**†
-5	21.08	**20.17**
-6	23.87	**22.53**

Table 4.3: Simulation results for the 50-city problem.

The table shows the mean percentage above the minimal tour length of 5.8286 *(as found by heuristic methods - see Table 4.4) for various initial values of* γ_0 *(100 trials,* $\Delta\gamma = 0.005$, $c_0 = 4$, $t_d = 3000$*). No tours were found with length equal to the best tour found by heuristic methods. †Corresponds to a mean tour length of* 6.9938.

	10-city		30-city		50-city	
	mean %	min.	mean %	min.	mean %	min.
NN	10.93	2.9933	14.58	4.6694	20.51	6.2193
NI	0.00	2.9933	3.64	4.4714	7.66	5.8853
2-OPT	0.00	2.9933	2.01	4.4714	5.16	5.9124
LK	0.00	2.9933	0.56	4.4714	1.88	5.8286

Table 4.4: Simulation results for various heuristics on 10, 30 and 50-city problems.

The table shows simulation results for the nearest neighbour (NN), node insertion (NI), 2-OPT and Lin and Kernighan (LK) heuristics on all problem sizes. All heuristics are implemented as in (Reinelt, 1994). For each method and problem, the mean percentage above the minimal tour length (over 50 trials) and the minimum tour length found by that method are given. The minimal tour length found by any heuristic method is 2.9933 *(10-city),* 4.4714 *(30-city) and* 5.8286 *(50-city).*

4.7 Chapter Summary

This chapter presented an investigation of hysteretic annealing which is a technique used to improve the quality of solutions obtained from an optimisation network. Without hysteretic annealing an optimisation network is prone to become stuck in a local minimum of the objective function. Such a local minimum may represent a poor, high cost solution to the problem at hand.

Hysteretic annealing seeks to guide the network state towards good solutions by continuously modifying the network's Lyapunov function. Standard hysteretic annealing involves the addition of an extra term

$$E^{ann} = -\frac{\gamma}{2}\sum_{xi}(v_{xi} - 0.5)^2$$

to the Lyapunov function for the network. As shown in this chapter, a linearised analysis of the network dynamics reveals that while annealing has introduced the necessary components into the dynamics to allow solution quality to be improved, it also acts to push $\mathbf{v}$ away from the valid subspace. Consequently, there is a fundamental conflict between the penalty functions, which attempt to restrict $\mathbf{v}$ to the valid subspace, and standard annealing which forces $\mathbf{v}$ off the valid subspace. At worst this could lead to invalid solutions, a situation that has until now been avoided by careful selection of parameters. Moreover, by removing $\mathbf{v}$ from the valid subspace the operation of the network is invalidated as it now seeks to minimise an objective function without a valid representation for $\mathbf{v}$.

To overcome the limitations of the standard hysteretic annealing algorithm, it was necessary to develop a new approach. A correctly formulated approach to hysteretic annealing must satisfy three requirements:

1. Annealing must introduce a non-zero eigenvalue in the zerosum subspace that is controlled by the annealing parameter γ. This enables annealing to guide $\mathbf{v}$ towards good solutions.

2. As the penalty terms alone correctly constrain $\mathbf{v}$ to be valid, annealing should not interfere with the operation of the penalty terms.

3. The relative values of E^{lyap} when evaluated at valid $0-1$ points should not be altered.

Consideration of these requirements led to the development of the modified hysteretic annealing function, which for the TSP is given by

$$E^{mod} = \frac{1}{2}\gamma\|\mathbf{v} - (\mathbf{T}^{zs}\mathbf{v} + \hat{\mathbf{e}})\|^2 - \frac{\gamma}{2}\sum_{xi}\left(v_{xi} - \frac{1}{N}\right)^2.$$

The modified approach to hysteretic annealing has all the essential features of a correctly formulated annealing mechanism. It retains the ability to guide $\mathbf{v}$ toward good solutions, by introducing a non-zero eigenvalue in the zerosum subspace. It also avoids any conflict with the penalty functions by effectively decoupling the actions of annealing and the penalty functions into different subspaces. Consequently, $\mathbf{v}$ will remain in the valid subspace throughout the annealing process, which is vital for the correct operation of an optimisation network.

Simulations on a range of travelling salesman problems support the modified annealing approach as being preferable to the standard annealing algorithm. Not only does the modified annealing remove the possibility of causing invalid solutions, but it has been demonstrated to produce a slightly superior quality of solution.

CHAPTER V

The Issue of Scaling

To this point in the thesis we have examined an approach to problem mapping that can guarantee that valid solutions will be found for a variety of combinatorial optimisation problems. In addition, we have developed annealing techniques which help to guide an optimisation network to good quality solutions, while retaining the ability to ensure valid solutions. Therefore, it would appear that optimisation networks are ready for application.

However, we must look more closely at the performance of optimisation networks for there awaits another challenge. The many empirical studies of optimisation networks solving combinatorial optimisation problems to be found in the literature display an alarming feature: as the problem size increases, the quality of the solutions found by the network rapidly decreases. These empirical studies lead to the belief that optimisation networks *scale poorly* to large problem sizes. While this fact has been acknowledged by few researchers, it has serious ramifications for the already tight niche market of optimisation networks and must therefore be addressed.

In this chapter we uncover the reasons for the poor scaling of optimisation networks to large problems and investigate several approaches to overcoming the problem. Of particular importance to our investigations is the discovery that optimisation networks operate by *embedding a heuristic* into the dynamics of the network. Moreover, it will be shown that the heuristics used by optimisation networks are ultimately responsible for their poor scaling. To improve the performance of optimisation networks it is necessary to replace or modify the heuristics that they use.

We begin in Section 5.1 by demonstrating the poor scaling properties of optimisation networks when used to solve the travelling salesman problem. In Section 5.2 we use the Ising spin problem, which is a simple graph 2-colouring, to show that optimisation networks use simple heuristics to solve a problem. Furthermore, we show that the poor performance of optimisation networks on large problems can be attributed to the tendency of such simple heuristics to encourage the formation of small, locally optimal segments in the solution. As the key to the poor performance of optimisation networks is the heuristic that they employ, in Section 5.3 we contrast the performance of two discrete heuristics for solving the TSP. While these heuristics are more closely aligned with simulated annealing than optimisation networks, they offer valuable insights into the poor scaling of optimisation networks to large problem sizes. Finally, in Section 5.4 we propose two alternative methods for improving the performance of optimisation networks.

5.1 Scaling with Problem Size

While the optimisation network literature is heavily populated with experimental results for combinatorial problems such as the TSP, many of which add weight to the conjecture

	Mean Solution Quality (%)				
γ_0	10-city	20-city	30-city	40-city	50-city
-1	0.00	1.26	9.61	10.72	19.24
-2	0.00	**0.27**	6.61	11.50	15.88
-3	0.00	0.51	**3.97**	11.21	**13.61**
-4	0.00	1.41	7.74	9.79	13.62
-5	0.00	1.44	10.61	**9.50**	14.66
-6	0.00	1.44	9.17	9.92	15.27

Table 5.1: Performance of the Hopfield network on the travelling salesman problem.

For each TSP and initial value of the annealing parameter γ_0, the Hopfield network is run 50 times. The mean cost of the tours found by the Hopfield network is expressed as a percentage above the cost of the best tour found by 50 trials of the Lin-Kernighan heuristic. The Lin-Kernighan heuristic was implemented as given in (Reinelt, 1994). Parameters for the Hopfield network simulations were $c_0 = 2$, $\Delta\gamma = 0.001$, t_d = 1000(10-city), 1500(20-city), 2000(30-city), 2500(40-city), 3000(50-city).

that optimisation networks scale poorly *e.g.* (Abe and Gee, 1995; Abe, 1996), their purpose has not been to investigate the scaling properties of optimisation networks. To demonstrate the poor scaling of optimisation networks we have simulated the Hopfield network on a series of progressively larger Euclidean travelling salesman problems.

The problems considered had 10, 20, 30, 40 and 50 cities placed randomly using a uniform distribution over the unit square. Once again, the objective function E^{obj} for the travelling salesman problem is given by equation (3.2) and the penalty function E^{cns} is given by equation (3.13). For each problem, the weight c on the penalty function is set so that

$$c = c_0 N \max_{x,y,z}(d_{xy} + d_{xz}),$$

as suggested by the most recent results presented in the literature (Abe and Gee, 1995). By setting c_0 to an appropriately large value all simulations were made to converge to valid solutions. The initial conditions are given by $v_{xi} = 0.5 + \alpha * \texttt{rand}$ where `rand` is a random value in the range $[-0.5, 0.5]$ and $\alpha = 0.0001$. Annealing was accomplished with the modified hysteretic annealing technique described in Section 4.4 and the integration was performed using the optimised step-size technique. While detailed information on the optimised step-size technique can be found in Appendix B, it is sufficient to note that such a technique holds the annealing parameter constant at γ_0 for the first t_d time steps and then increments the annealing parameter by an amount $\Delta\gamma$ at successive time steps.

The results of the experiment are displayed in Table 5.1 and Figure 5.1, where the performance of the Hopfield network is given relative to the best solution found by the Lin-Kernighan (LK) heuristic. On the 10-city problem the Hopfield network always produces the solution found by the LK heuristic. For the 30-city problem the mean length of the tour found by a Hopfield network is 3.97% above the best solution from the LK heuristic and for the 50-city problem this figure has risen to 13.61%. For the results in this experiment we see that as the problem size increases, the quality of the solutions found by an optimisation network decreases.

In Section 2.7 a comparison between optimisation networks and alternative solution

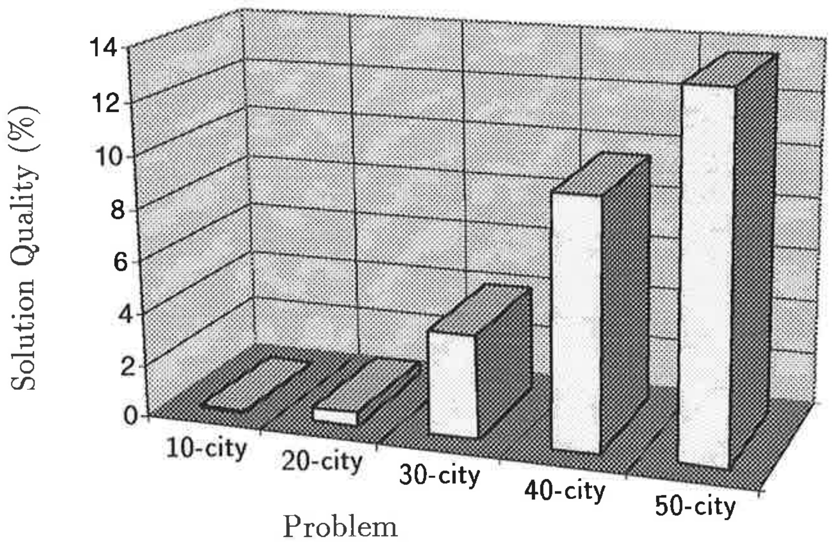


Figure 5.1: Performance of the Hopfield network on the travelling salesman problem.

The data for this graph is obtained from Table 5.1. For each problem the best mean solution quality obtained by the Hopfield network is plotted. The quality of the solutions found by the Hopfield network quickly deteriorates as the size of the problem increases.

methods for combinatorial optimisation problems was undertaken in order to establish the niche market for optimisation networks. It was concluded that even though optimisation networks generally produce solutions of lower quality than other methods, their great advantage is the very fast solution times achievable when implemented in hardware. In light of the evidence presented in this section it is necessary to consider a further element in the comparison: problem size. Extensive experimental studies (Reinelt, 1994) of heuristics such as the nearest neighbour algorithm, 2-OPT, 3-OPT and the full Lin-Kernighan algorithm show that the quality of solution achieved by these heuristics is nearly independent of the problem size. In contrast, the results of this section demonstrate that the problem size is a significant influence on the solution quality obtained from an optimisation network. While the space complexity of a hardware implementation already limits optimisation networks to moderate-sized problems, the poor scaling of optimisation networks is a significant limitation which further restricts the niche market for optimisation networks. Ideally, the performance of optimisation networks would be independent of problem size. In the rest of this chapter we investigate the reasons for the poor scaling of optimisation networks to large problems and suggest possible remedies.

5.2 Segmentation

The key to understanding the poor scaling of optimisation networks to large problem sizes was alluded to by Wilson and Pawley when they noted that the Hopfield network typically produces solutions which are *"composed of several segments, each of which is locally a good (solution), but which are joined together in such a way as to make a bad (solution) overall"* (Wilson and Pawley, 1988). In this section we will show that as the problem size increases, a growing level of segmentation of the solution results in ever-decreasing solution quality. As a vehicle for demonstrating the consequences of segmentation we have chosen the Ising spin problem, which is a simple graph 2-colouring. Although the Ising spin problem has obvious global minima, it serves to demonstrate the limitations of the Hopfield network and thereby explain the poor scaling of optimisation networks.

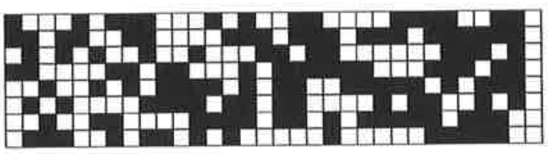
5.2.1 The Ising Spin Problem

The Ising spin model consists of N elements arranged into a $m \times n$ rectangular grid. Each element must be assigned the state `black` or `white`. Any two elements are neighbours if they share an edge. The problem is to assign a state to every element such that the number of neighbouring elements with the same state is minimised. For the Ising spin problem, the global minimum corresponds to a situation where each element has the opposite state from all of its neighbours, giving a checkerboard pattern of activity across the grid. Figure 5.2 shows the global minimum and a random state of an 8×32 Ising spin model.

Hopfield Network Mapping

In order to solve the Ising spin problem with the Hopfield network, each element i in the Ising model is assigned two neurons[1], with outputs $v_i^{(w)}$ and $v_i^{(b)}$ respectively. The neuron

[1]The Ising spin problem is a restricted version of the general graph N-colouring problem, which itself may be mapped onto the Hopfield network by using N neurons to represent each node in the graph.


(a) Random state.

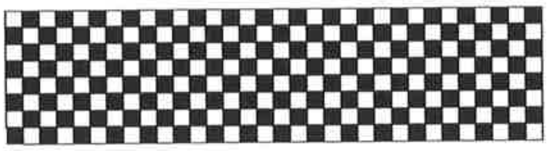

(b) Global minimum.

Figure 5.2: States of an Ising spin model.

An 8×32 Ising spin model is shown with (a) random element states and (b) in the global minimum state. The reverse checkerboard pattern, obtained by flipping the state of each element is also a global minimum.

outputs are continuous and may vary over the interval $[0, 1]$. If $v_i^{(b)} = 1$ and $v_i^{(w)} = 0$ then element i has been assigned the state **black**. Similarly, if $v_i^{(w)} = 1$ and $v_i^{(b)} = 0$ then element i has been assigned the state **white**. With this problem representation, the Ising spin problem may be expressed as follows:

$$\text{minimise} \quad E^{obj} = \frac{1}{4}\sum_{i=1}^{N}\sum_{j=1}^{N} C_{ij}\left(v_i^{(w)}v_j^{(w)} + v_i^{(b)}v_j^{(b)} - v_i^{(w)}v_j^{(b)} - v_i^{(b)}v_j^{(w)}\right) + E_0 \quad (5.1)$$

$$\text{subject to} \quad v_i^{(w)} + v_i^{(b)} = 1 \quad \forall\, i \in \{1, \dots N\} \quad (5.2)$$

where C_{ij} is 1 if elements i and j are neighbours, and 0 otherwise. The objective function for the Ising spin problem is given in equation (5.1) and with the constant term given by $E_0 = \frac{1}{2}(m(n-1) + n(m-1))$ the optimal checkerboard solution to the Ising spin problem has a cost $E^{obj} = 0$. Note that careful consideration of equation (5.1) would reveal that for every neighbouring pair of elements that share the same state, the cost of the solution is increased by one.

To map this problem onto an optimisation network we set $E^{lyap} = E^{obj} + E^{cns}$, where the penalty function is given by

$$E^{cns} = \frac{c}{4}\sum_{i=1}^{N}\left(v_i^{(w)} + v_i^{(b)} - 1\right)^2 \quad (5.3)$$

in accordance with the valid subspace problem mapping. The procedure for determining E^{cns} may be found in Appendix D. The penalty function E^{cns} is zero when the constraints for the Ising spin problem are satisfied and is positive otherwise. With c set to a large positive constant, the penalty function E^{cns} will encourage the network state to lie on the valid subspace.

Once again it is necessary to employ an annealing technique to encourage the formation of good solutions and aid convergence to a $0-1$ point. In Appendix D, it is shown that an appropriate modified hysteretic annealing function for the Ising spin problem is given by

$$E^{mod} = -\frac{\gamma}{4}\sum_{i=1}^{N}\left(v_i^{(w)} - v_i^{(b)}\right)^2. \tag{5.4}$$

For the purposes of our simulations the annealing parameter is given by $\gamma = (t/\tau)^2 + \gamma_0$ where τ is set to a positive constant and γ_0 is a small negative constant. Increasing the absolute values of both τ and γ_0 will improve the solution quality at the expense of longer running time. Modified hysteretic annealing operates in the usual manner by adding E^{mod} to the Lyapunov function for the Hopfield network *viz.* $E^{lyap} = E^{obj} + E^{cns} + E^{mod}$.

Now that the problem representation and Lyapunov function have been determined, the dynamic equation for the internal state $u_i^{(w)}$ of neuron $v_i^{(w)}$ can be calculated as

$$\begin{aligned}\frac{du_i^{(w)}}{dt} &= -\frac{\partial E^{lyap}}{\partial v_i^{(w)}} \\ &= \frac{1}{2}\sum_{j=1}^{N} C_{ij}\left(v_j^{(b)} - v_j^{(w)}\right) - \frac{c}{2}\left(v_i^{(w)} + v_i^{(b)} - 1\right) + \frac{\gamma}{2}\left(v_i^{(w)} - v_i^{(b)}\right). \end{aligned} \tag{5.5}$$

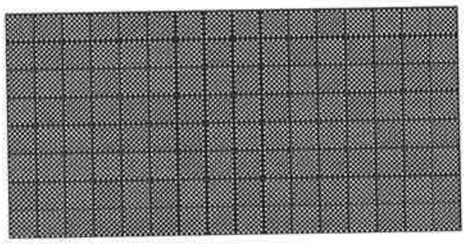
A dynamic equation for $u_i^{(b)}$ may be determined in similar fashion. Considering only the terms arising in equation (5.5) from the objective function, the dynamics of the Hopfield network may be simply explained: if the neighbours of element i are likely to be assigned the state `black` (*i.e.* $v_j^{(b)} > v_j^{(w)}$) then the assignment of state `white` to element i is encouraged ($du_i^{(w)}/dt > 0$). Conversely, if the neighbours of element i are likely to be assigned the state `white` (*i.e.* $v_j^{(b)} < v_j^{(w)}$) then the assignment of state `white` to element i is discouraged ($du_i^{(w)}/dt < 0$). Therefore, the combined effect of the dynamics for the neurons associated with element i is to encourage the formation of a locally optimal segment encompassing element i and its four immediate neighbours.

The tendency of the Hopfield network to encourage the formation of locally optimal segments encompassing an element and its four immediate neighbours may be viewed from an algorithmic standpoint as a *rule* or *heuristic* that has been inserted into the dynamics of the optimisation network. The success of the Hopfield network approach to the Ising spin problem will ultimately depend upon the utility of such a heuristic.

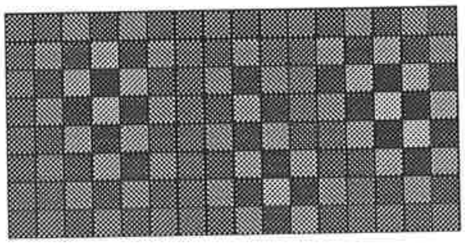
Simulation Results

For all experiments reported below the parameters were set as follows: $c = 200$, $\tau = 100$, $\gamma = 0$ and the gain of the transfer function in the Hopfield network was set to $T^p = 1$. Initially, the internal states of the neurons were randomly distributed on the interval $[-0.005, 0.005]$. Integration of the network dynamics given by equation (5.5) was performed using the function `ode15s.m` available in MATLAB®. As previously stated, the quality of solutions may be improved by employing a more gradual annealing schedule, corresponding to larger absolute values for τ and γ. However, while the solution quality may be improved, the qualitative results reported here will still be evident, but at a larger problem size.

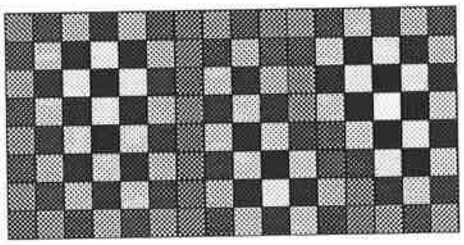
The operation of a Hopfield network on an 8×16 Ising spin problem is shown in Figure 5.3. As the operation of the network progresses, several regions in the Ising spin model evolve independently. Each of these regions is a *seed-point*, which due to the actions of the heuristic embedded into the network, will reinforce and expand in diameter. Gradually the elements of the Ising model converge to states representing



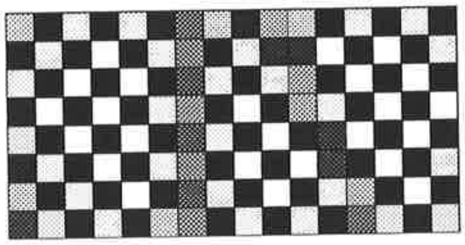
(a) $t = 0.1301$.



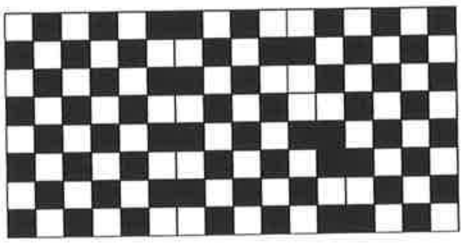
(b) $t = 2.8506$.



(c) $t = 3.7106$.



(d) $t = 5.6474$.

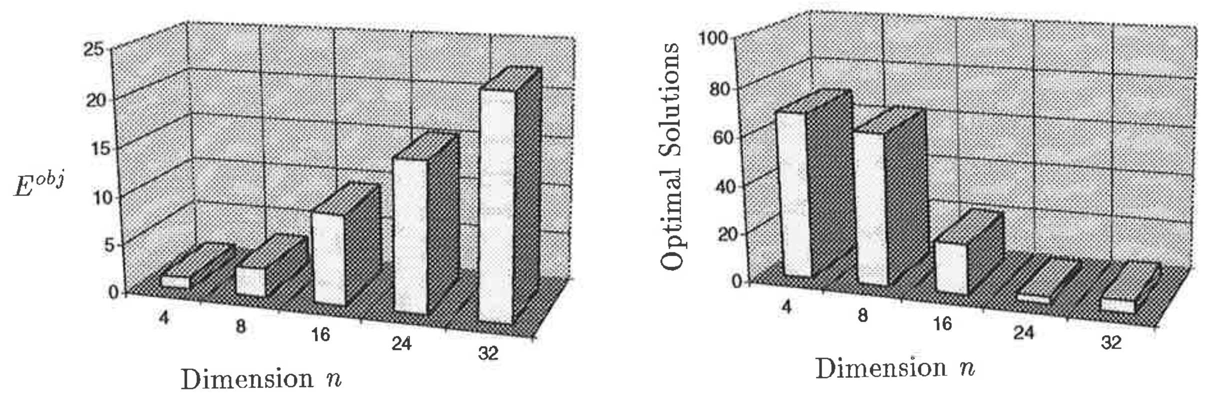


(e) $t = 81.2304$.

white
black

Figure 5.3: The Hopfield network solving an Ising spin problem.

Time evolution of the state of a Hopfield network as it solves an 8×16 *Ising spin problem. The elements of the Ising model have been rendered according to the difference in outputs of their associated neuron pair (i.e.* $v_i^{(w)} - v_i^{(b)}$*). The final solution has a cost* $E^{obj} = 18$*. Note the formation of multiple seed-points.*



(a) Mean Cost of Solutions. (b) Number of Optimal Solutions.

Figure 5.4: Performance of the Hopfield network on the Ising spin problem.

The performance of the Hopfield network as the problem size n is increased for the $8 \times n$ Ising spin problem. For a given problem size n, the Hopfield network was simulated 100 times.

`black` or `white`. As the heuristic used by the Hopfield network encourages the formation of only a small, locally optimal region, several seed-points can evolve simultaneously, as no one seed-point has enough influence to annihilate the others. Unfortunately, in this instance the seed-points correspond to opposing checkerboard patterns and the inevitable result is segmentation of the final solution as shown in Figure 5.3(e). Similar results have been noted for the travelling salesman problem, where disjoint segments of the tour are seen to evolve from several seed points, creating the need to introduce a sub-optimal cross-over into the tour (Van den Bout and Miller, 1989).

With a clear understanding of the heuristic used by the Hopfield network to solve the Ising spin problem, it is interesting to examine the performance of the network on a variety of problem sizes. Accordingly, we have trialed the network on instances of an $8 \times n$ Ising spin problem with increasing horizontal dimension n. The results of the experiment are shown in Figure 5.4, where the poor scaling of the Hopfield network is again apparent. For an 8×4 Ising spin problem the average cost of the solution is approximately $E^{obj} = 1$, implying that the average solution had only one segment. In contrast, for an 8×32 problem the average cost of the solution is over $E^{obj} = 24$, implying that the average solution contained four segments.

As has been shown in this section, the optimisation network approach to solving a combinatorial optimisation problem is to embed a simple heuristic into the dynamics of the network. These simple heuristics encourage the formation of locally optimal segments and are evident as seed-points in the solution. Moreover, as only small, locally optimal regions are encouraged, there arises the possibility for several seed points to coexist. This is increasingly true as the size of the problem grows and the influence of one seed point on another lessens. With an increased number of seed-points there will be a corresponding increase in the segmentation of the final solution. As the problem size increases we see a greater degree of segmentation, which translates to a decrease in the solution quality.

5.3 A Comparison of Two Heuristics

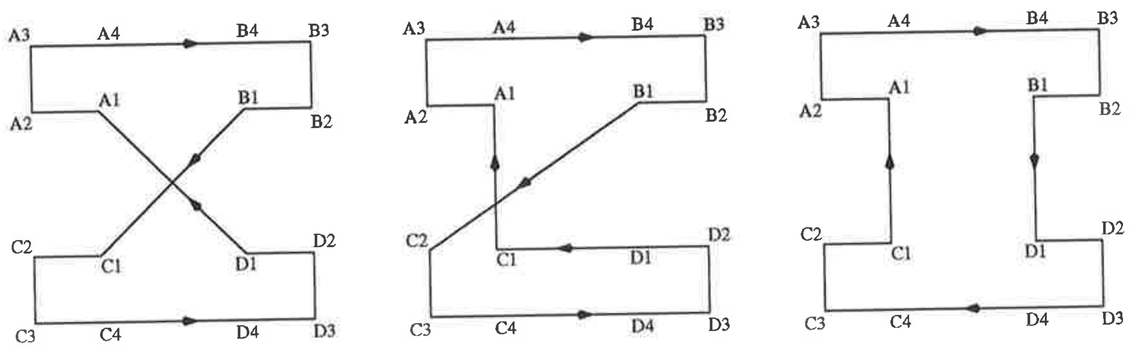
In the previous section we have seen that optimisation networks operate by embedding a heuristic into the dynamics of the network. The success of the network is dependent upon the utility of that heuristic for that problem. Moreover, the experiments conducted in the previous section on the Ising spin problem indicate that the heuristics employed by optimisation networks are responsible for the segmentation of solutions and ultimately for the poor scaling of optimisation networks to large problems. It is natural then to ask if there are heuristics that scale to large problem sizes better than the standard heuristics used by optimisation networks. One of the few researchers to address this issue is Lister (Lister, 1990; Lister, 1993; Lister, 1994), who has investigated the connection between the heuristic employed in an optimisation algorithm and characteristics of the energy landscape that may make the search successful.

In his work Lister considers two algorithms for the solution of the travelling salesman problem. While both algorithms are really instances of simulated annealing and may not truly be called optimisation networks, they do offer valuable insight into the poor scaling of optimisation networks to large problem sizes. In the rest of this section we summarise Lister's work on the scaling properties of two different approaches to the travelling salesman problem.

The first algorithm implements a node insertion heuristic that can shift a city from one position in the tour to another, in a single operation. The algorithm starts with an initial randomly chosen tour and at each step moves to another legal tour by applying a node insertion move. The choice of which node insertion move to make at each step of the algorithm is determined by applying rejection-less simulated annealing (Greene and Supowit, 1984). Such a technique makes a weighted random selection from the set of all possible node insertion moves, with the bias for each move calculated as a function of the resulting change in the tour length. By favouring node insertion moves that decrease the tour length, this algorithm can achieve good solutions to the travelling salesman problem.

To understand the operation of the node insertion algorithm, consider the travelling salesman problem shown in Figure 5.5. The problem consists of sixteen cities arranged into four clusters of four cities each. The tour shown in Figure 5.5(a) is typical of those found by optimisation networks, as it exhibits segmentation of a good solution, similar to that which we have shown to occur for the Ising spin problem and which others have noticed for the travelling salesman problem (Wilson and Pawley, 1988; Van den Bout and Miller, 1989). The tour consists of two locally optimal segments which have been connected by a sub-optimal crossing-over of the segments B1 $\rightarrow$ C1 and D1 $\rightarrow$ A1. The operation of the node insertion algorithm can be seen in Figure 5.5(b), where city C1 has been moved between cities A1 and D1. This node insertion move has removed the cross-over from Figure 5.5(a), but has introduced another cross-over and actually increased the length of the tour. To remove all cross-overs from the tour it is necessary to move every city from one side of the cross-over to the other. In this instance it would take a further six moves, and there would be no appreciable decrease in the length of the tour until the cross-over had been removed.

As shown by the simple example in Figure 5.5(a) and (b), to remove a sub-optimal cross-over from a tour with the node insertion heuristic, it takes a number of moves which is proportional to the number of cities on one side of the cross-over. Moreover, there is no appreciable decrease in the length of the tour until the cross-over has finally been removed. Consequently, the energy landscape defined by the node insertion algorithm



(a) length = 5.4142 (b) length = 5.6514 (c) length = 5.0000

Figure 5.5: Heuristics operating on a travelling salesman problem.

An initial tour for a 16-city travelling salesman problem is shown in (a). The tour shown in (b) is obtained from the original tour in (a) by using the node insertion heuristic to move city C1 between cities A1 and D1. The tour shown in (c) is obtained from the initial tour shown in (a) by reversing the direction of travel on the segment C1...D1.

can be seen to have large *flat spots.* These flat spots become larger as the problem size increases and the number of cities that must be moved to remove a cross-over in a tour increases. Experimental evidence (Lister, 1990; Lister, 1993) suggests that for small problem sizes there is a reasonable probability that a cross-over in the tour can be eliminated by node insertion, but as the problem size increases and the flat spots in the energy landscape become larger, the probability of removing a cross-over from the tour decreases.

The second algorithm considered by Lister combines the segment reversal heuristic with rejection-less simulated annealing. The result of the segment reversal heuristic can be seen in Figure 5.5(c), which shows the tour from Figure 5.5(a) after the direction of travel on the segment C1...D1 has been reversed. Obviously only one segment reversal is needed to remove the sub-optimal cross-over from the tour. Simulations performed by Lister (Lister, 1993) show that the segment reversal heuristic gives much better solutions to the TSP than the node insertion heuristic. A typical run of both algorithms on a 200 city TSP is shown in Figure 5.6 (Lister, 1993). While both algorithms maintain a random solution at a high temperature, as the temperature in the annealing process is decreased, the solution for the segment reversal heuristic exhibits good coarse structure. In contrast, the solution for the node insertion heuristic contains many long-range connections which are retained in the final solution. This example clearly demonstrates that even when operating on large problems, solutions found by the segment reversal heuristic do not exhibit any segmentation. Lister concludes that the segment reversal heuristic scales well to large problem size (Lister, 1993).

Why does segment reversal scale to large problem sizes better than node insertion? Considering that any node insertion move can be emulated by two operations of segment reversal, whereas an arbitrary segment reversal can take up to $N/2$ node insertion moves to accomplish - can we attribute the supremacy of segment reversal to it being in some sense a *stronger heuristic*? Lister has suggested that the explanation is not this simple. Rather, he argues that segment reversal defines an energy landscape that is "quasi-fractal" and that simulated annealing is well suited to finding the global minimum of a

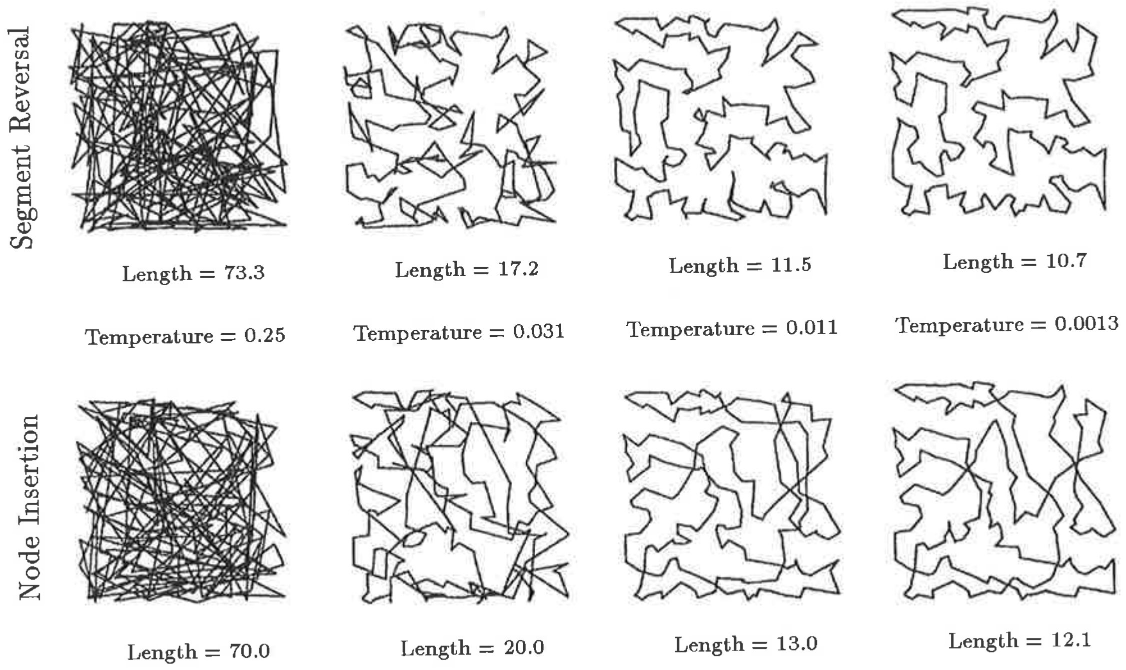


Figure 5.6: Operation of the two heuristics on a 200 city TSP (Lister, 1993).

Three intermediate solutions and the final solution from a run of simulated annealing using the segment reversal heuristic (top) and the node insertion heuristic (bottom). At high temperatures both algorithms have maintained random solutions. However, as the temperature falls the segment reversal heuristic quickly develops a good coarse structure for the final solution. In contrast, the node insertion heuristic continues to contain many long-range connections. The final solution produced by the node insertion heuristic exhibits the same poor structure that was noticed for optimisation networks by Wilson and Pawley (Wilson and Pawley, 1988).

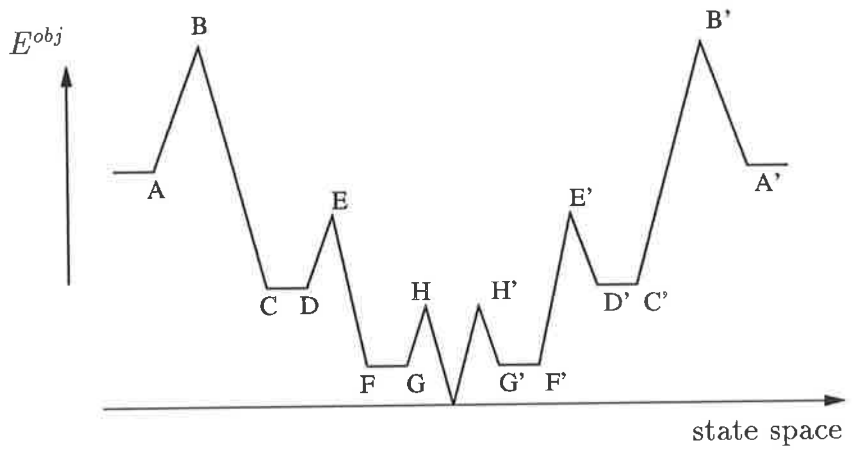


Figure 5.7: A fully-fractal landscape (Lister, 1993).

A fully-fractal energy landscape exhibits self-similar structure. It has been shown that simulated annealing is well suited to finding the minimum of a fully-fractal landscape (Sorkin, 1990).

quasi-fractal landscape (Lister, 1993; Lister, 1994; Lister, 1995). The utility of simulated annealing when operating on a *fully-fractal* energy landscape can be understood with reference to Figure 5.7. At very high temperatures, simulated annealing allows the state to wander freely over the entire state space. As the temperature decreases, the transition A→B→C is substantially more probable than the reverse transition, causing the solution to be effectively confined to the region between C and C′. Such *ergodicity breaking* effectively locks in the coarse structure of the final solution, in a sense implementing a *divide and conquer* approach to optimisation. Obviously, as the temperature drops further simulated annealing will readily converge to the global minimum. While segment reversal does not define a fully-fractal landscape there is quantitative support for the conjecture that the landscape is quasi-fractal (Kirkpatrick and Toulouse, 1985). Lister argues that the failure of node insertion to scale to large problems is a consequence of its landscape not being quasi-fractal.

Finally, it must be asked if the performance of discrete heuristics such as node insertion and segment reversal is relevant to analog optimisation networks? The answer lies in the rule that has been embedded into the dynamics of an optimisation network approach to the TSP. In Chapter 6 it will be shown that the node insertion heuristic discussed here is in fact a close discrete-state analogy to the rule embedded in an analogue optimisation network and as such, if flat spots in the energy landscape for the node insertion heuristic prevent the heuristic from working well on large problems, then it can be expected that similar problems will be encountered by the analogue optimisation network. However, while segment reversal overcomes the problems associated with the node insertion heuristic Lister readily admits that segment reversal is *"incompatible with the gradient descent component"* (Lister, 1994) of optimisation networks and therefore cannot be implemented as an analogue optimisation network.

5.4 Overcoming Segmentation

So far in this chapter we have established that optimisation networks work by embedding simple heuristics into the dynamics of the network. Moreover, it was shown that these heuristics are responsible for the poor performance of optimisation networks on large problems. Naturally we must ask if the performance of optimisation networks can be improved, by simply choosing a discrete heuristic that performs well and embedding it into the dynamics of the optimisation network? Unfortunately this is not always possible, as the heuristic we choose must be compatible with the gradient descent nature of an optimisation network. As we noted in the previous section, while segment reversal is indeed a good heuristic for the travelling salesman problem it is not possible to embed segment reversal into the dynamics of an optimisation network.

In this section we will investigate two different approaches that attempt to improve the performance of optimisation networks by modifying the heuristics used. Of particular concern is their applicability to a wide variety of problems.

5.4.1 Multi-scale Networks

To improve the performance of optimisation networks, it is necessary to somehow prevent the network from converging to solutions exhibiting locally optimal segments which have been poorly connected together. Perhaps the most obvious way to eliminate the segmentation problem is to implement a multi-scale approach by incorporating several layers into an optimisation network. The basic structure of a multi-scale network is shown in Figure 5.8. The lowest layer of a multi-scale network is similar to a standard optimisation network and represents the optimisation problem at the finest scale or resolution. At successively higher layers the optimisation problem is represented at coarser scales, requiring fewer neurons in each layer. *Top-down* connections between the layers would ensure that information about the state of the coarser scale layers flows down to the finer scale layers. At the same time, *bottom-up* connections ensure an information flow from the finer to the coarser scale layers. Such a multi-scale optimisation network implements a *divide and conquer* approach to optimisation. Lister (Lister, 1993; Lister, 1994; Lister, 1995) suggests that when an optimisation network implements a divide and conquer strategy, it will have an energy landscape that scales well to large problems.

In his work on simulated annealing, Lister (Lister, 1994) has proposed a discrete multi-scale heuristic for solving the Ising spin problem. The multi-scale heuristic constructs a hierarchy of layers, as shown in Figure 5.9, with each layer in the hierarchy representing the Ising spin problem at a different scale. In this case, the lowest level is, in isolation, a normal 4×16 Ising spin model. At level two there is a 2×8 network of elements, with each element being associated with several contiguous elements in level one. Levels three and four are constructed in a similar manner. When an element in level two or higher changes state, all of its associated level one elements change state. The change in energy for such a transition is given by the energy change at level one of the hierarchy. It should be noted that elements in level two and higher can adopt more than two states. The exact number of states equals the number of possible state configurations that their corresponding level one elements can adopt. For the hierarchy shown in Figure 5.9, level two elements have 2^4 states. When used in conjunction with simulated annealing, the multi-scale heuristic is a powerful mechanism for solving the Ising spin problem, which significantly outperforms simulated annealing operating directly on the Ising spin problem (*i.e.* directly on level one). While the multi-scale heuristic is not developed as an optimisation network, the structure of an analogous multi-scale network for solving the Ising spin problem is immediately obvious when comparing Figures 5.8 and 5.9.

This is not the first time that the concept of multi-scale networks has been proposed. Indeed, multi-scale optimisation networks have been developed and applied to graph

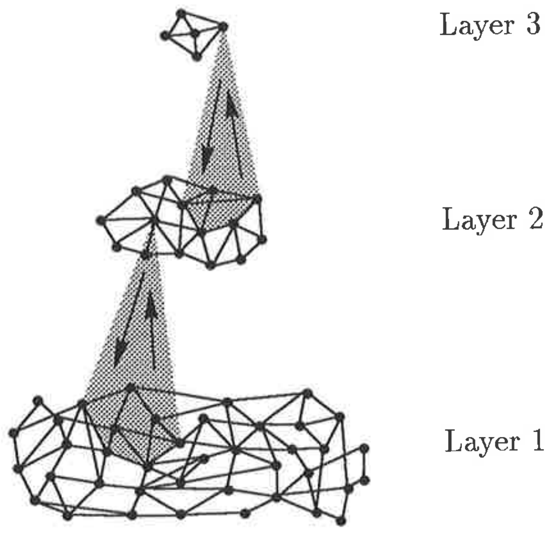


Figure 5.8: A multi-scale optimisation network.

A multi-scale optimisation network consists of several layers. Layer 1 represents the optimisation problem at its finest scale. At successively higher layers the problem representation becomes coarser, requiring fewer neurons in each layer. For clarity the interconnections between layers have not been explicitly shown. Instead we have indicated the typical extent of connections between layer 1 and a single neuron in layer 2 by shading. Both top-down and bottom-up interconnections exist. Similarly the connections between layer 2 and a neuron in layer 3 have been indicated by shading.

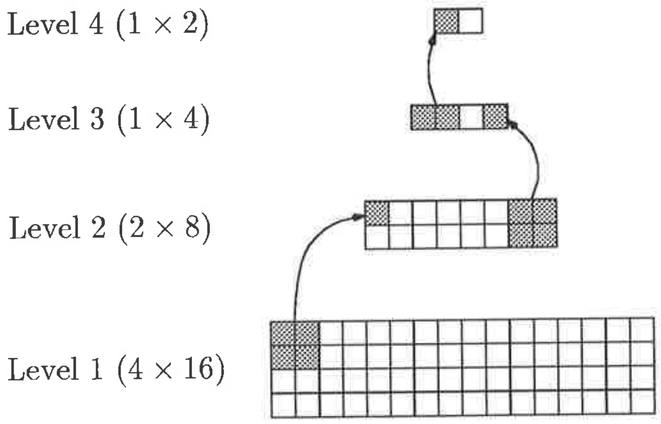


Figure 5.9: Multi-scale discrete heuristic for the Ising spin problem (Lister, 1994).

The discrete multi-scale heuristic for the Ising spin problem makes use of a hierarchy of layers. In isolation, level 1 is a normal 8×16 *Ising spin problem. Level 2 is constructed by associating each element with several contiguous elements from layer 1. Layers 3 and 4 are constructed in a similar manner. If an element in level 2 or higher changes state then all of its associated level 1 units also change state.*

labelling (Mjolsness et al., 1991), selected problems in low-level vision (Mjolsness et al., 1991; Tsioutsias and Mjolsness, 1996) and signal decomposition (Truyen and Cornelis, 1995). In the majority of these cases, the motivation for using a multi-scale optimisation network has been to speed up the rate of convergence by using smaller, approximate versions of the problem at coarser scales (Mjolsness et al., 1991). The emphasis has not been on the improvements in solution quality that a multi-scale approach to optimisation can offer.

To implement a multi-scale network, it is first necessary to identify a suitable *partitioning function* which will allow the optimisation problem to be represented at progressively coarser scales. This is clearly exhibited by the multi-scale heuristic for the Ising spin problem in Figure 5.9. In this case the partitioning function groups several contiguous elements at the lower level and represents them by a single element in the next highest level. Our knowledge of how segmentation occurs in the Ising spin problem naturally suggests this partitioning function. To understand why, consider the lowest level of the hierarchy for the multi-scale heuristic. As shown by the simulations in Section 5.2.1 it is common for the state of this level to be given by two locally optimal segments, each belonging to opposing checkerboard solutions to the Ising spin problem. With the partitioning function used in the multi-scale heuristic, one of these locally optimal segments can be changed into the opposing checkerboard solution by coordinated switching of several elements in the higher levels. In doing so, the globally optimal solution to the Ising spin problem has been obtained by removing the segmentation from the lowest level in the hierarchy.

In much the same way as we have done for the Ising spin problem, all applications of multi-scale networks to date, have exploited some knowledge about how segmentation effects the problem at hand. This knowledge is then used to construct a suitable partitioning function. Moreover, this knowledge is available prior to simulation of the multi-scale network and so can be built into the structure of the network. Unfortunately this is not true for all problems. Take for example a random travelling salesman problem. It is not immediately clear where in the final solution of the TSP segmentation will become evident. Of course, a suitable partitioning function may well be learnt through experience in repeatedly running the multi-scale network and optimising the partitioning function. However, such a process will be exceptionally time costly and will erode the one great advantage of optimisation networks – speed. It seems that even though multi-scale networks can significantly improve the performance of optimisation networks, they are limited to problems such as the Ising spin problem where a suitable partitioning function does not have to be learnt.

5.4.2 Extended Neighbourhood

In Section 5.2 it was shown that segmentation of solutions found by optimisation networks arises as a consequence of the dynamics encouraging the formation of small, locally optimal segments. As this is true for all applications of optimisation networks, a widely applicable approach to improving their performance is to extend the neighbourhood in which the network encourages the formation of locally optimal segments. This concept will be demonstrated on the familiar Ising spin problem.

Using the problem representation for the Ising spin problem given in Section 5.2.1, we will construct a dynamic equation for the internal state $u_i^{(w)}$ of neuron $v_i^{(w)}$, that will encourage the formation of larger locally optimal segments. To that end, consider the second-order dynamics[2] [3]

$$\frac{du_i^{(w)}}{dt} = \sum_{j=1}^{N} \sum_{\substack{k=1 \\ k \neq i}}^{N} C_{ij} C_{jk} \left(v_j^{(b)} v_k^{(w)} - v_j^{(w)} v_k^{(b)} \right) . \tag{5.6}$$

An equivalent dynamic equation for $u_i^{(b)}$ may be constructed. The second-order dynamics are easily interpreted; if the neighbours of element i are likely to be assigned the state `black`, and the neighbours of the neighbouring elements of i are likely to be assigned the state `white`, then the assignment of state `white` to element i is encouraged $(du_i^{(w)}/dt > 0)$. Conversely, if the neighbours of element i are likely to be assigned the state `white`, and the neighbours of the neighbouring elements of i are likely to be assigned the state `black`, then the assignment of state `white` to element i is discouraged $(du_i^{(w)}/dt < 0)$. The combined effect of the second-order dynamics for neurons associated with element i is to encourage the formation of a locally optimal segment encompassing element i, its four immediate neighbours and their immediate neighbours. The second-order dynamics represent a *much stronger heuristic* approach to solving the Ising spin problem than that which is implemented by the Hopfield network.

The concept of enlarging the neighbourhood in which locally optimal segments are encouraged may be further extended by including even higher-order terms in the dynamics *e.g.* the third-order dynamics[1] [2]

$$\frac{du_i^{(w)}}{dt} = \sum_{j=1}^{N} \sum_{\substack{k=1 \\ k \neq i}}^{N} \sum_{\substack{l=1 \\ l \neq j}}^{N} C_{ij} C_{jk} C_{kl} \left(v_j^{(b)} v_k^{(w)} v_l^{(b)} - v_j^{(w)} v_k^{(b)} v_l^{(w)} \right) . \tag{5.7}$$

Simulation Results

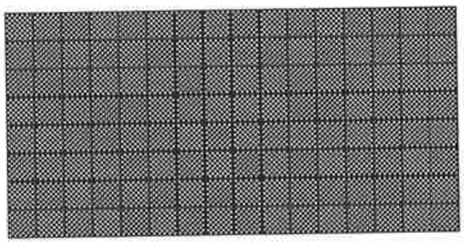
To demonstrate the benefit that can be obtained from extending the neighbourhood in which locally optimal segments are encouraged, we have simulated the higher-order dynamics given in equations (5.6) and (5.7) and compared the results with those obtained for the Hopfield network in Section 5.2.1. The simulation method and parameters were as given in Section 5.2.1.

A typical solution to the third-order dynamics of equation (5.7), when simulated on an 8×16 Ising spin problem, is shown in Figure 5.10. These results may be contrasted with the results obtained by a Hopfield network in Figure 5.3. Once again, the final solution is seen to evolve from seed-points which were formed in the early stages of convergence. However, as can be seen in Figures 5.10(b) and (c), the extended neighbourhood of the third-order dynamics allows only a single seed-point to form. Under the action of the dynamics, the seed-point reinforces and expands in diameter, with the elements of the Ising spin model converging to states representing `black` or `white`. In this run of the third-order dynamics one of the globally optimal checkerboard patterns was obtained.

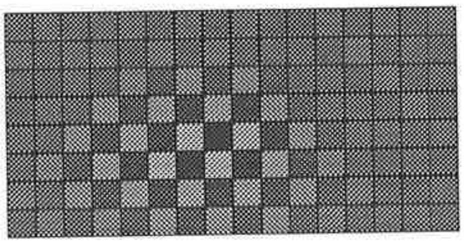
Our motivation for developing the higher-order dynamics was to extend the neighbourhood in which locally optimal segments are encouraged and in doing so to overcome

[2]For the sake of clarity, annealing and penalty terms have not been shown but must be included as in equation (5.5).

[3]The existence of a Lyapunov function for these dynamics is not guaranteed by the results of Chapter 6 as the necessary symmetry conditions are not satisfied. However, convergence has been attained in all experiments reported herein.

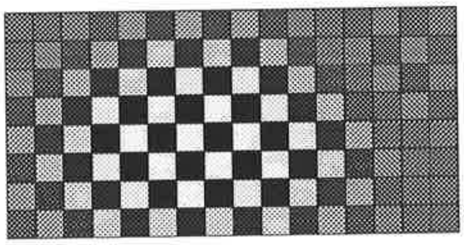


(a) $t = 0.0962$.

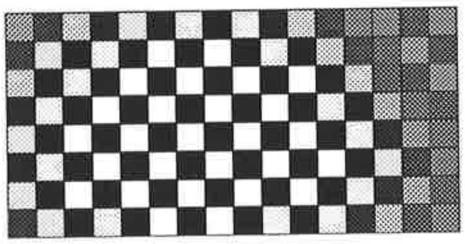


(b) $t = 0.3398$.

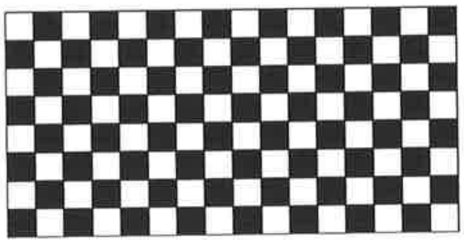
white
black



(c) $t = 0.4971$.



(d) $t = 0.7917$.



(e) $t = 2.5494$.

Figure 5.10: Solution of an Ising spin problem using third-order dynamics.

Time evolution of the third-order dynamics (equation (5.7)) used to solve an 8×16 *Ising spin problem. The elements of the Ising model have been rendered according to the difference in outputs of their associated neuron pair (i.e.* $v_i^{(w)} - v_i^{(b)}$*). The final solution is the global minimum with cost* $E^{obj} = 0$*. Note the formation of only a single seed-point.*

the problem of segmentation. If the higher-order dynamics have been successful in tackling the problem of segmentation, then they should also improve upon the scaling of the Hopfield network to large problem sizes. To investigate this claim, we have simulated the higher-order dynamics on instances of an $8 \times n$ Ising spin problem, and compared the results to those obtained with the Hopfield network in Section 5.2.1. The results of the experiment are shown in Figure 5.11. By obtaining more optimal solutions and achieving a lower mean cost over 100 trials at each problem size, it is quite obvious that the higher-order dynamics have improved upon the Hopfield network.

The superior performance of the higher-order dynamics may be attributed to the empirical observation that the basin of attraction for the globally optimal solution is enlarged as the order of the dynamics is increased. However, this observation must be tempered by the realisation that if a near-optimal solution is desired then as the problem size increases, it is necessary to introduce even higher-order dynamics so as to encourage the formation of even larger locally optimal segments. The need to introduce even higher-order dynamics exemplifies the trade-off between between computational effort and solution quality, that can be made with many heuristics.

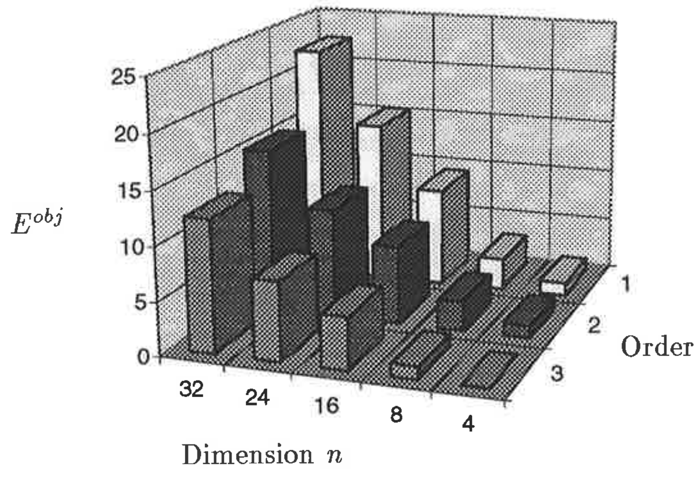
In Chapter 6 we will further develop the use of higher-order dynamics in optimisation networks by formalising the concept of higher-order neural networks (HONNs) for optimisation. In addition, it will be shown how to map the travelling salesman problem onto a HONN.

5.5 Chapter Summary

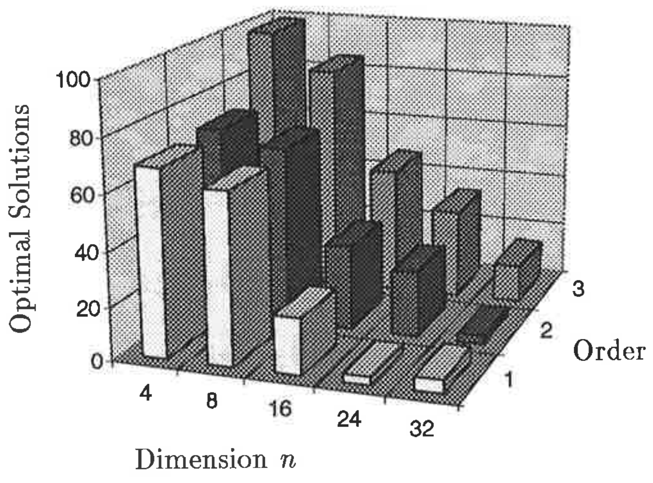
At the beginning of this chapter we used the travelling salesman problem to demonstrate that as the problem size increases, the quality of solutions found by an optimisation network rapidly decreases. Such poor scaling further restricts the already tight niche market for optimisation networks. The purpose of this chapter was to understand the causes for poor scaling, and if possible to suggest remedies.

It was revealed that optimisation networks operate by embedding simple heuristics into the dynamics of the network. Typically, these heuristics encourage the formation of small, locally optimal segments and their effect can be seen most clearly when used to solve the Ising spin problem, which is a simple graph 2-colouring problem. The heuristics used in the Ising spin problem allow several seed-points to evolve independently. As a result, the final solution exhibits several locally optimal segments which have been joined together so as to produce a poor solution. Moreover, as the problem size increases, even more locally optimal segments develop independently and consequently the solution quality deteriorates.

To improve the performance of optimisation networks it is necessary to replace or modify the heuristics used in their dynamics. An investigation into the node insertion and segment reversal heuristics for the travelling salesman problem suggested that the energy landscape of standard optimisation network approaches contained large flat spots. Moreover, the size of such flat spots increased with the problem size and made it increasingly difficult to find good solutions to the travelling salesman problem. In contrast, the segment reversal heuristic was shown to scale well to large problem sizes, but unfortunately it was not amenable to implementation as a heuristic to be used in optimisation networks.



(a) Mean Cost of Solutions.



(b) Number of Optimal Solutions.

Figure 5.11: Performance of higher-order dynamics on the Ising spin problem.

The performance of the higher-order dynamics is contrasted to that of the Hopfield network (1st order dynamics) on the $8 \times n$ Ising spin problem, with increasing horizontal dimension n. For a given problem size n, each network was simulated 100 times. As shown in (a) the mean cost of the solutions found decreases as the order of the dynamics used increases, and (b) the number of optimal solutions found increases as the order of the dynamics increases. Note that in (b) the direction of increasing magnitude has been reversed on some axes as compared to (a).

Although there do exist heuristics which outperform those used by optimisation networks, we cannot simply select one and then embed it into the dynamics of the optimisation network. Instead, we have proposed two new optimisation networks. The first is a multi-scale optimisation network, which incorporates several layers into an optimisation network. Each layer treats the problem at a different scale and so the whole multi-scale network implements a divide and conquer strategy to optimisation. Unfortunately, implementation of a multi-scale network relies on detailed apriori knowledge of how segmentation will be exhibited in the solution obtained from a standard optimisation network. While such knowledge is available for simple problems like the Ising spin problem, it is not readily available for other problems like the travelling salesman.

We have shown that the poor scaling of optimisation networks arises form the tendency of the heuristics to encourage small locally optimal segments in the solution. In view of this fact, the second approach to improving the performance of optimisation networks extends the neighbourhood in which the network encourages the formation of locally optimal segments by including higher-order terms in the network dynamics. The resulting network is more complex, but this can be seen as embodying the trade-off between computational effort and solution quality that can be made with most heuristics. Using higher-order dynamics greatly improved the performance of optimisation networks on the Ising spin problem.

Chapter VI

Higher-Order Neural Networks

We have now established that the Hopfield network does not scale well to large problem sizes due to its tendency to form solutions which consist of small, locally optimal segments. Moreover, the Hopfield network's dynamics embed simple heuristics which are responsible for this behaviour. We showed in Chapter 5 that to improve the performance of optimisation networks we can use higher-order terms in the network dynamics to extend the neighbourhood in which locally optimal segments are encouraged. In this chapter we will formalise this concept as that of higher-order neural networks (HONNs) for optimisation. HONNs are a new class of optimisation networks that may be applied to almost any combinatorial optimisation problem to which the Hopfield network has been applied. In this chapter we will show how to apply HONNs to the solution of the travelling salesman problem.

We begin in Section 6.1 by describing the model we have used for HONNs. As with other optimisation networks, HONNs operate by a process of gradient descent and are therefore suited to the solution of optimisation problems. A simple example is given in Section 6.1.1 where we show how a HONN may be used for minimising a cubic objective function. While our approach of using HONNs for the solution of combinatorial optimisation problems is novel, it is not the first application of higher-order recurrent neural networks that has been documented. Consequently, in Section 6.1.2 we present a brief survey of other applications of higher-order recurrent networks that may be found in the literature.

A major goal for this chapter is to demonstrate how to use a HONN for the solution of difficult combinatorial optimisation problems like the travelling salesman. However, before we apply HONNs to the solution of the TSP it is necessary to understand the approach used by the Hopfield network for solving the TSP. Consequently, in Section 6.2 we examine the dynamics of the Hopfield network to extract the heuristic used when solving the TSP. We then proceed to formulate a HONN approach to the TSP by extending the heuristic to encourage the formation of larger locally optimal segments. This process is described in Section 6.3. An experimental comparison of the Hopfield network and the HONN approach to the TSP is given in Section 6.4. Finally, in Section 6.5 we present a summary of the chapter.

6.1 The HONN model

In many ways a HONN may be viewed as an extension of the Hopfield network where the feedback vector is no longer restricted to be linear, but is allowed to be a polynomial function of the network output $\mathbf{v}$. As with the continuous state Hopfield network, the i^{th} processing unit is described by two variables: its internal state u_i and its output $v_i \in [0,1]$. In order to produce a feedback vector which is a polynomial function of the neuron outputs, a HONN uses multiplication units to form the products of outputs from various subsets of neurons, which are then fed-back to the input of the network. The HONN forms a dynamic system described by the following equations:

$$\frac{du_i}{dt} = \sum_{k=1,\, i\in\mathcal{I}_k}^{L} T_k \frac{m_i(k)}{v_i} \prod_{j\in\mathcal{I}_k} v_j^{m_j(k)} - \eta u_i \tag{6.1}$$

$$v_i = g(u_i) \tag{6.2}$$

where η is a decay parameter, $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots \mathcal{I}_L\}$ is a collection of L unordered subsets of the indices $\{1, 2, \ldots N\}$, T_k is the synaptic weight applied to the product of the outputs of neurons in the unordered subset $\mathcal{I}_k$ and $m_j(k)$ is the power to which the output of neuron j is raised when calculating the product of neuron outputs in subset $\mathcal{I}_k$. The transfer function $g(\cdot)$ is a monotonically increasing function that restricts the neuron outputs to the range $v_i \in [0, 1]$. As for the continuous Hopfield network, the usual choice is the shifted hyperbolic tangent function

$$g(u_i) = \frac{1}{1 + \exp(-u_i/T^p)}$$

where T^p is a parameter controlling the slope of the transfer function in the linear region, and is termed the pseudo-temperature. A cursory examination of the HONN dynamics given by equation (6.1) reveals that the feedback vector is now a polynomial function of the neuron output $\mathbf{v}$. The continuous Hopfield network is a restricted version of the HONN where the subsets $\mathcal{I}_k$ are given by all indices and all unordered pairs of indices from $\{1, 2, \ldots N\}$. In which case, if the $m_j(k)$ functions are suitably chosen the dynamic equation (6.1) will reduce to the Hopfield dynamics given by equation (2.12).

A HONN admits the Lyapunov function (Dembo et al., 1991; Poteryaiko, 1991)

$$E^{lyap}(\mathbf{u}) = -\sum_{k=1}^{L} T_k \prod_{j\in\mathcal{I}_k} v_j^{m_j(k)} + \eta \sum_i \int_{0.5}^{v_i} g^{-1}(V)dV. \tag{6.3}$$

It is easily shown that under the action of the network dynamics the Lyapunov function for the network is non-increasing, *viz.*

$$\begin{aligned}
\frac{dE^{lyap}}{dt} &= \sum_i \frac{\partial E^{lyap}}{\partial v_i}\frac{dv_i}{dt} \\
&= \sum_i \left(-\sum_{k=1,\, i\in\mathcal{I}_k}^{L} T_k \frac{m_i(k)}{v_i} \prod_{j\in\mathcal{I}_k} v_j^{m_j(k)} + \eta u_i \right) g'(u_i)\frac{du_i}{dt} \\
&= -\sum_i g'(u_i)\left(\frac{du_i}{dt}\right)^2 \\
&\leq 0.
\end{aligned}$$

Since E^{lyap} is bounded below and the time derivative of E^{lyap} is non-increasing, the HONN will seek out minima in the Lyapunov function and come to a stop at such points. Obviously HONNs operate in much the same manner as continuous Hopfield networks, with only the form of the Lyapunov function being different. While a HONN does necessitate the use of multiplication units and an increase in the number of connections in the network, it also allows more flexibility than is available with a Hopfield network.

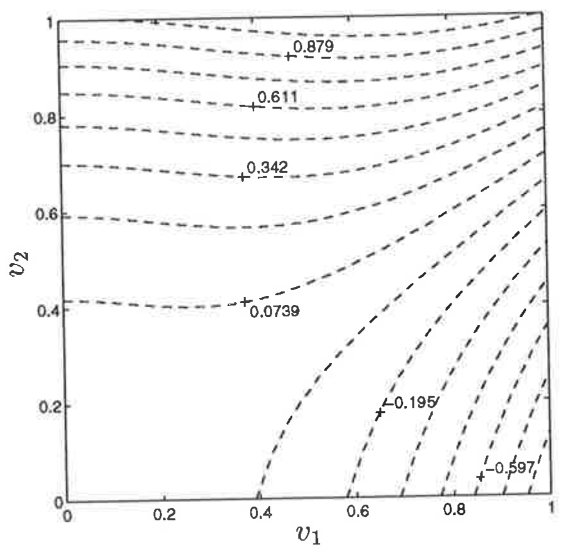


Figure 6.1: Contours of the 2-dimensional cubic objective function.

A contour plot for the function $E = v_1^2 v_2 - v_1^3 + v_2^3$. *The global minimum of this function is* $E([1,\ 0]^T) = -1$.

6.1.1 Simple Example

To demonstrate the operation of a HONN, we shall use a HONN to minimise the 2-dimensional cubic function given by

$$E = v_1^2 v_2 - v_1^3 + v_2^3. \tag{6.4}$$

A contour plot of the function E is shown in Figure 6.1. In order to minimise this function with a HONN it is necessary to equate the Lyapunov function of equation (6.3) with the function E given in equation (6.4). To achieve this we set $\eta = 0$, and choose the unordered sets as $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3\} = \{\langle 1, 2\rangle,\ \langle 1\rangle,\ \langle 2\rangle\}$. Other parameters are set as follows:

$$\begin{aligned} T_1, T_3 &= -1 \\ T_2 &= 1 \\ m_1 &= (2,\ 3,\ 0) \\ m_2 &= (1,\ 0,\ 3). \end{aligned}$$

By substituting these parameters into equation (6.3), it is easily verified that $E^{lyap} = E$. Further substitution of the parameters into equation (6.1) gives the network dynamics as

$$\begin{aligned} \dot{u}_1 &= -2v_1 v_2 + 3v_1^2 \\ \dot{u}_2 &= -v_1^2 - 3v_2^2. \end{aligned}$$

Note that these dynamics are a second-order system and so the entire network is referred to as a second-order HONN[1]. Operation of the HONN is shown in Figure 6.2. The

[1]A second-order HONN will have a third-order Lyapunov function. Throughout this thesis we have used the convention of naming a system according to the order of its dynamics and not the order of its Lyapunov function.

network is initialised at the centre of the unit square and converges to the global minimum of the Lyapunov function at $\mathbf{v} = [1,\, 0]^T$. Obviously the network is performing gradient descent upon the Lyapunov function.

6.1.2 Applications of HONNs

The concept of HONNs is not new. Indeed, recurrent networks that utilise higher-order connections have been applied to a variety of problems, usually in response to the shortcomings of first-order networks such as the Hopfield network. In this section we briefly present the applications of HONNs that have appeared in the literature.

Associative Memories

One of the earliest uses of the Hopfield network was as a form of associative or content addressable memory (CAM) (Hopfield, 1982; Hopfield, 1984). When a set of patterns have been stored into memory, a CAM will retrieve one of these patterns based upon the similarity of the stored patterns to the pattern which is presented at the input. When a corrupted version of one of the patterns is presented at the input, the CAM will retrieve the stored pattern which best matches the input. To utilise the Hopfield network as a CAM, the connection weights and external biases must be programmed so that the minima of the network's Lyapunov function correspond to the desired stored patterns. When operating as a CAM, the state of the Hopfield network is initialised with a corrupted pattern and the network dynamics will guide the network state to the nearest local minimum of the Lyapunov function. In doing so, it is expected that the network will have corrected the errors in the corrupted pattern. While it is now generally accepted that the Hopfield network has a low storage capacity, HONNs have been proposed as a means to improve the number of patterns that may be stored. Unlike the continuous state HONNs which were introduced in Section 6.1, most studies of HONNs as content addressable memories have used discrete state units (Lee et al., 1986; Chen et al., 1986; Psaltis and Park, 1986; Gardner, 1987; Abbott and Arian, 1987; Personnaz et al., 1987; Baldi and Venkatesh, 1987; Baldi, 1988; Baldi and Venkatesh, 1993; Karlholm, 1993; Chao et al., 1993).

Grammatical Inference

Grammatical inference is the task of learning to recognise temporal sequences (strings) generated by a set of rules known as a grammar (Hopcroft and Ullman, 1979). The language of a grammar is the set of all strings that can be generated by the grammar. To infer the grammar from a set of positive and negative example strings (*i.e.* strings that do or do not belong to the language) an inference engine is used to determine which rules belong to the grammar. A rule is of the form: {`input, current state`} → `next state`.

There has been considerable interest in the use of recurrent neural networks as inference engines for the task of grammatical inference. Recurrent neural networks seem to be well suited to implementing grammars as their dynamics make use of internal state information and naturally implement the state transitions that correspond to the rules of a grammar. By using suitable algorithms to learn the weights on the connections in the network, a recurrent neural network can infer the grammar based on a set of positive and negative example strings. While the recurrent networks used for grammatical inference are not exactly as described in Section 6.1, it is interesting to note that both theoretical and experimental comparisons show that second-order recurrent networks are better

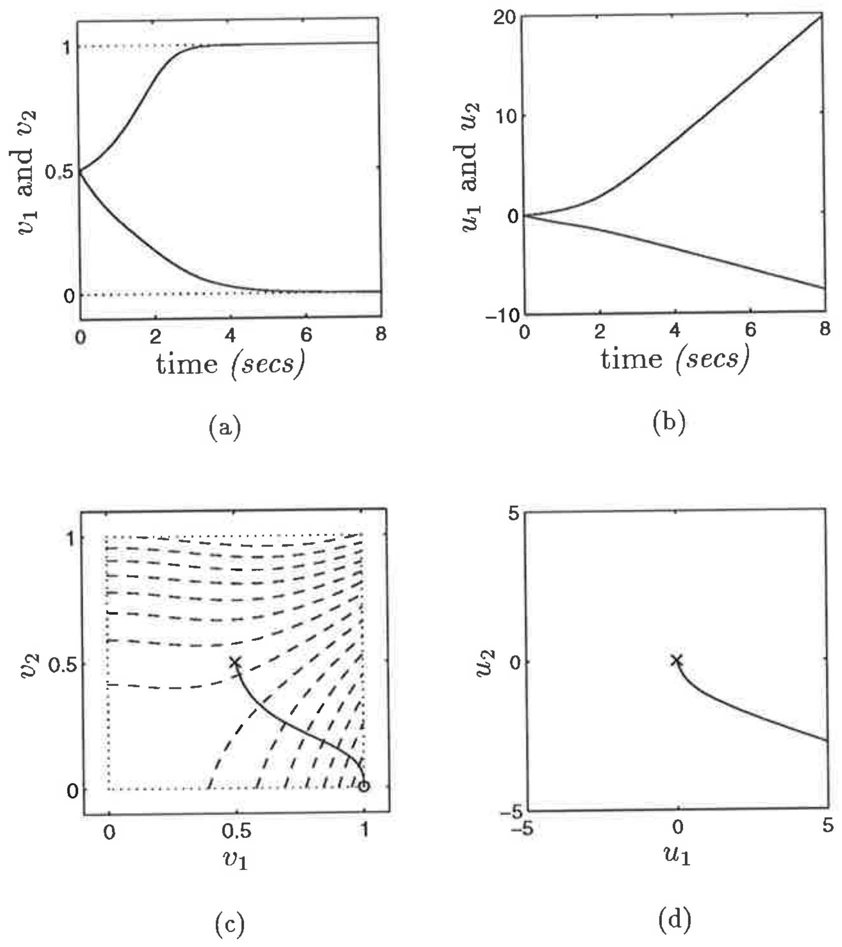


Figure 6.2: Operation of a HONN.

The plots show the operation of a HONN on the cubic function E given in equation (6.4). To ensure that $E^{lyap} = E$*, the parameters* $\eta = 0$ *and* $T^p = 1$ *are used. Integration of the HONN dynamic equations yields the traces shown in (a) and (b). Since there is no decay term, the* $\mathbf{u}$ *variables are unbounded. The gradient descent nature of the HONN dynamics is evident in (c) where the dashed lines are contours of* E^{lyap}*. The network is initialised at* $\mathbf{v} = [0.5,\ 0.5]^T$ *and converges to* $\mathbf{v} = [1,\ 0]^T$*, which is the the global minimum of the Lyapunov function within the unit square. The HONN dynamics were integrated using the Euler method and a constant step-size* $\Delta t = 0.01$.

suited to the task of grammatical inference than first-order networks (Goudreau et al., 1994; Miller and Giles, 1993). These studies suggest that second-order dynamics offer a more direct implementation of the rules of a grammar. Consequently, most approaches to grammatical inference with recurrent networks now advocate the use of higher-order connections (Giles et al., 1992; Watrous and Kuhn, 1992; Zeng et al., 1993; Horne and Giles, 1995; Giles and Omlin, 1993; Giles and Omlin, 1994).

Optimisation

Higher-order neural networks have been used for the solution of optimisation problems where the objective function could not be expressed as a quadratic function. For example the problems of line labelling (Salem and Young, 1991), block design (Bofill, 1993) and relational structure mapping (Miller and Zunde, 1992) require cubic or quartic objective functions and so must be solved with a HONN.

A very different standpoint on the use of HONNs for optimisation was suggested in Chapter 5 and underlies the rest of this thesis and various other published works (Cooper, 1995a; Cooper, 1995b). As optimisation networks operate by embedding simple heuristics into the network's dynamics, higher-order networks potentially have the ability to embed more sophisticated heuristics into the dynamics. While we are not compelled to use the higher-order connections in order to solve the problem, we do so in order to gain an improvement in the solution quality at the expense of extra network complexity.

Another approach to the use of higher-order neural networks was proposed by Xu and Tsai (Xu and Tsai, 1991). They propose a first-order network for the solution of the TSP and proceed to run that network many times. For each trial run, if the network converged to a poor solution, then higher-order terms were added to the network dynamics in order to prevent this solution from being obtained again. In essence, they are sculpting the Lyapunov function, using higher-order terms to add *"bumps"* in locations which help to prevent poor solutions from being obtained in the next trial.

Pattern Recognition

Although we are primarily interested in the applications of recurrent networks that utilise higher-order connections, it is worthwhile noting that there have been many applications of higher-order feed-forward networks. Specifically, feed-forward networks have been used to implement translation, rotation and scale invariant pattern recognition (Maxwell et al., 1986; Giles and Maxwell, 1987; Giles et al., 1988; Perantonis and Lisboa, 1992).

6.2 The Hopfield approach to the TSP

In Chapter 5 it was shown that a practical approach to improving the performance of optimisation networks was to extend the neighbourhood in which locally optimal solutions were encouraged by using higher-order terms in the network dynamics. Moreover, while this approach was demonstrated only on the relatively simple Ising spin problem we stated that increasing the order of the network dynamics was a widely applicable approach to improving solution quality. To justify this assertion we shall demonstrate how to apply HONNs to the solution of the travelling salesman problem.

However, before we can apply higher-order networks to the TSP it is necessary to ascertain the approach that the Hopfield network has employed in solving the TSP. In order to do this we must proceed, as we did with the Ising spin problem, by analysing the dynamics of the Hopfield network to determine the heuristic that has been embedded.

It is appropriate at this time to recapitulate the problem representation for the TSP as it was presented in Section 3.2. To solve a N city TSP, a $N \times N$ array of neurons is used where the output v_{xi} of the neuron in row x and column i is one if city x is to be visited in the i^{th} position of the tour and zero otherwise. To map the TSP onto the Hopfield network, we set the Lyapunov function to be

$$E^{lyap} = E^{obj} + E^{cns} + E^{mod}$$

where the objective function is given by equation (3.2), *viz.*

$$\begin{aligned} E^{obj} &= \frac{1}{2}\sum_{\substack{xy \\ x\neq y}}\sum_{i} v_{xi} d_{xy}(v_{y,i-1} + v_{y,i+1}) \\ &= \sum_{\substack{xy \\ x\neq y}}\sum_{i} d_{xy} v_{xi} v_{y,i+1}. \end{aligned}$$

Here we have once again made use of the convention that all summations are from 1 to N unless explicitly shown otherwise. Furthermore, the penalty function is given by equation (3.13), *viz.*

$$E^{cns} = \frac{c}{2N}\sum_{i}\left(\sum_{x} v_{xi} - 1\right)^2 + \frac{c}{2N}\sum_{x}\left(\sum_{i} v_{xi} - 1\right)^2 - \frac{c}{2N^2}\left(\sum_{xi} v_{xi} - N\right)^2$$

and the modified hysteretic annealing function is given by equation (4.10), *viz.*

$$\begin{aligned} E^{mod} &= \frac{\gamma}{2N}\sum_{x}\left(\sum_{i} v_{xi} - 1\right)^2 + \frac{\gamma}{2N}\sum_{i}\left(\sum_{x} v_{xi} - 1\right)^2 \\ &\quad -\frac{\gamma}{2N^2}\left(\sum_{xi} v_{xi} - N\right)^2 - \frac{\gamma}{2}\sum_{xi}\left(v_{xi} - \frac{1}{N}\right)^2. \end{aligned}$$

6.2.1 Extracting the heuristic from the Hopfield network

The first step in extracting the heuristic used by the Hopfield network for solving the TSP is to establish the dynamics of the network. The network dynamics are easily determined as follows:

$$\begin{aligned} \frac{du_{xi}}{dt} &= -\frac{\partial E^{lyap}}{\partial v_{xi}} \\ &= -\frac{(c+\gamma)}{N}\left(\sum_{y} v_{yi} - 1\right) - \frac{(c+\gamma)}{N}\left(\sum_{j} v_{xj} - 1\right) + \frac{(c+\gamma)}{N^2}\left(\sum_{yj} v_{yj} - N\right) \\ &\quad +\gamma\left(v_{xi} - \frac{1}{N}\right) - \sum_{\substack{y \\ y\neq x}} d_{xy}(v_{y,i+1} + v_{y,i-1}). \end{aligned} \tag{6.5}$$

Our analysis of the valid subspace mapping in Chapter 3 showed that the terms in the dynamic equation arising from the penalty function E^{cns} act only to restrict the network state $\mathbf{v}$ to lie on the valid subspace. Additionally, the analysis of the modified annealing technique presented in Chapter 4 showed that the terms arising in the dynamic equation from E^{mod} help only to guide $\mathbf{v}$ through the valid subspace. In no way do these terms represent a heuristic for minimising the distance travelled in the TSP, and so for the purpose of extracting a heuristic we may simply express the network dynamics as

$$\frac{du_{xi}}{dt} = \begin{pmatrix} \texttt{penalty and} \\ \texttt{annealing terms} \end{pmatrix} - \sum_{\substack{y \\ y \neq x}} d_{xy}(v_{y,i+1} + v_{y,i-1}). \tag{6.6}$$

Although it is not explicitly stated in the problem representation, there is considerable merit in interpreting $v_{y,i+1}$ as the probability that city y will occupy position $i+1$ of the tour. Indeed, our understanding of the network dynamics is greatly assisted by making such an interpretation of the network outputs. Consider the situation where it is probable that city y will occur in position $i+1$ of the tour *i.e.* $v_{y,i+1}$ is much larger than all other neuron outputs in column $i+1$ of the array. We can see from equation (6.6) that in such a situation u_{xi} will be decreased by an amount which is proportional to the distance between cities x and y. Moreover, a decrease in the internal state u_{xi} also results in a decrease in the neuron output $v_{xi} = g(u_{xi})$ and so the probability that city x occupies position i of the tour is decreased. However, it must be remembered that all neurons in column i of the array will have their internal state decreased by an amount proportional to the distance between city y and the city that they represent. Consequently, if the distance between cities x and y is small compared to the distance between city y and any other city, then the probability of city x occurring in position i of the tour is *comparatively encouraged.* A similar argument may be presented for the interaction between a neuron $v_{y,i-1}$ and the neurons in column i of the array.

Lister (Lister, 1993) suggests that equation (6.6) embeds an analogue version of the node insertion heuristic for the TSP where *"portions of cities are moved around"* in the tour. However, the preceding discussion of the network dynamics suggests an alternative interpretation: as equation (6.6) is acting to encourage the inclusion into the tour of *segments which connect two nearby cities*, we suggest that the network dynamics embed an analogue version of the nearest neighbour heuristic for the TSP (Reinelt, 1994). In its discrete form, the nearest neighbour heuristic builds a tour for the travelling salesman by starting at any city and then including the segment which visits the closest city which has not yet been included into the tour. The analogue version of the nearest neighbour heuristic embedded into equation (6.6) constructs a tour by using "portions" of segments between two cities. Interpretation of the network dynamics as an analogue nearest neighbour heuristic is further supported by the formation of seed-points as the network computes a solution to the TSP (Van den Bout and Miller, 1989). When the seed point first forms, it represents a small path that will be found in the final tour. As the seed point expands, the path is extended by adding a further segment to the end of the existing path. Obviously this is closely related to the manner in which the discrete nearest neighbour heuristic works. The difference between the two interpretations of the network dynamics is only of interest when we consider how to extend the heuristic for use in HONNs.

6.3 The HONN approach to the TSP

As was shown in Section 6.2 the Hopfield network approach to the TSP is an analogue nearest neighbour heuristic, whereby a tour is constructed by encouraging the formation of locally optimal segments between two cities. In this section we will show that a HONN may be used to extend the neighbourhood of these locally optimal segments to include not just two cities, but three or more cities.

As an objective function for the TSP when mapped onto a second-order HONN, consider

$$E^{obj} = \sum_{\substack{xyz \\ x\neq y\neq z}} \sum_{i} (d_{xy} + d_{yz}) v_{xi} v_{y,i+1} v_{z,i+2}. \tag{6.7}$$

When equation (6.7) is evaluated at a valid $0-1$ point it equates to twice the length of the tour represented by that point. In that respect, equation (6.7) is similar to the objective function used for the Hopfield network mapping of the TSP. However, it is important to note that equation (6.7) differs from the objective function for the Hopfield network at points inside the hypercube and consequently the basins of attraction for valid $0-1$ points have been altered. This observation will be the subject of more detailed comment in Chapter 7.

With the objective function given by equation (6.7), the Lyapunov function for the second-order HONN to solve the TSP is given by

$$E^{lyap} = E^{obj} + E^{cns} + E^{mod}.$$

Here the penalty function E^{cns} and the modified hysteretic annealing function E^{mod} are as given in Section 6.2. The network dynamics are determined as follows:

$$\begin{aligned}
\frac{du_{xi}}{dt} &= -\frac{\partial E^{lyap}}{\partial v_{xi}} \\
&= -\frac{(c+\gamma)}{N}\left(\sum_{y} v_{yi} - 1\right) - \frac{(c+\gamma)}{N}\left(\sum_{j} v_{xj} - 1\right) \\
&\quad + \frac{(c+\gamma)}{N^2}\left(\sum_{yj} v_{yj} - N\right) + \gamma\left(v_{xi} - \frac{1}{N}\right) \\
&\quad - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{xy} + d_{yz}) v_{y,i+1} v_{z,i+2} - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yx} + d_{xz}) v_{y,i-1} v_{z,i+1} \\
&\quad - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yz} + d_{zx}) v_{y,i-2} v_{z,i-1}.
\end{aligned} \tag{6.8}$$

Once again, we note that the terms arising in the dynamic equation from the penalty function E^{cns} act only to restrict $\mathbf{v}$ to lie on the valid subspace. Additionally, any term arising in equation (6.8) from the modified annealing function E^{mod} helps only to guide $\mathbf{v}$ through the valid subspace. As none of these terms act to minimise the distance travelled in the TSP, they should not be considered as part of the heuristic embedded into the dynamic equation of the HONN. Consequently, for the purpose of analysing the heuristic used by a HONN, we may simply express equation (6.8) as

$$\begin{aligned}
\frac{du_{xi}}{dt} &= \begin{pmatrix} \texttt{penalty and} \\ \texttt{annealing terms} \end{pmatrix} - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{xy} + d_{yz}) v_{y,i+1} v_{z,i+2} \\
&\quad - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yx} + d_{xz}) v_{y,i-1} v_{z,i+1} - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yz} + d_{zx}) v_{y,i-2} v_{z,i-1}.
\end{aligned} \tag{6.9}$$

In developing the heuristic which is embedded into the network dynamics of equation (6.9), we shall again interpret $v_{y,i+1}$ as the probability that city y will occur in position $i+1$ of the tour. Consider the situation where it is probable that city y will occupy position $i+1$ of the tour and city z will occupy position $i+2$ of the tour (*i.e.* $v_{y,i+1}$ is much larger than all other neuron outputs in column $i+1$ of the array and $v_{z,i+2}$ is much larger than all other neuron outputs in column $i+2$ of the array). In such a situation, the effect of the first summation in equation (6.9) is to decrease u_{xi} by an amount which is proportional to the length of the path connecting $x \rightarrow y \rightarrow z$. Moreover, a decrease in the internal state u_{xi} also results in a decrease in the neuron output $v_{xi} = g(u_{xi})$ and so the probability that city x occupies position i of the tour is decreased. However, it must be remembered that all neurons in column i of the array will have their internal state decreased by an amount which is proportional to the length of the path which connects the city they represent to city y and then to city z. Consequently, if the length of the path $x \rightarrow y \rightarrow z$ is small compared to the length of the path connecting any other city to city y and then to city z, then the probability of city x occuring in position i of the tour is *comparatively encouraged.*

A similar argument may be presented to explain the operation of the second and third summations in equation (6.9). These summation terms will encourage city x to occupy position i of the tour if city x corresponds to short paths $y \rightarrow x \rightarrow z$ and $y \rightarrow z \rightarrow x$ respectively.

The preceding analysis of the dynamics for a second-order HONN suggests that the network encourages the formation of locally optimal paths which connect three cities in the tour. In contrast, it was shown in Section 6.2 that the Hopfield network encouraged the formation of locally optimal paths (or segments) which connect only two cities in the tour. By extending the neighbourhood in which locally optimal segments are encouraged, the second-order HONN embeds a much stronger heuristic approach to the solution of the TSP than that which is used in the Hopfield network. In view of the results presented in Chapter 5 for the Ising spin problem, we expect that HONNs will give similar improvements in the solution quality for the TSP. Section 6.4 presents an experimental evaluation of the performance of the HONN approach to the solution of the TSP.

The concept of enlarging the neighbourhood in which locally optimal segments are encouraged may be further extended by considering even higher-order networks *e.g.* a third-order HONN with the objective function

$$E^{obj} = \sum_{\substack{xyzw \\ x \neq y \neq z \neq w}} \sum_{i} (d_{xy} + d_{yz} + d_{zw}) v_{xi} v_{y,i+1} v_{z,i+2} v_{w,i+3} \tag{6.10}$$

will act to encourage the formation of locally optimal paths which connect four cities in the tour. A $(N-1)^{th}$ order network would encourage the formation of locally optimal paths which connect all N cities in the tour. However, calculation of the Lyapunov function for a $(N-1)^{th}$ order network would explicitly calculate the cost of every possible tour. In that case, calculating the Lyapunov function involves the same computational effort as an enumeration technique which calculates the length of every possible tour and chooses the shortest. It is well known that as the number of possible tours for the TSP rises exponentially with the problem size, such enumeration techniques are impractical for all but the smallest problems. Consequently, a $(N-1)^{th}$ order HONN is also impractical for all but the smallest problems.

As the order of the HONN increases, the heuristic which is embedded into the network dynamics offers a progressively stronger approach to the solution of the TSP. In that sense, we may view HONNs as a family of heuristic approaches to the solution of the TSP where we may trade increased computational effort for improved solution quality by simply increasing the order of the network. Unlike most other heuristics for the

TSP, an increase in the computational effort required by a HONN does not mean a longer running time, but rather it results in an increased number of connections in the network. This is quite obvious when we consider that the number of interconnections necessary to implement a Hopfield network for the TSP is $O(N^3)$, while for a second-order HONN that number has increased to $O(N^4)$ and for a third-order HONN the number of interconnections is $O(N^5)$. As the order of the HONN increases, the limits of what may be feasibly implemented in hardware will very quickly be approached. Consequently, when considering a HONN approach to the TSP, one must have in mind a clear understanding of their performance. While we may be willing to trade complexity for improved solution quality, this can only be done up to a certain level, beyond which the complexity of the required network prohibits it from being implemented. It should be noted that such limitations are not unique to optimisation networks. Most other heuristics involve a trade-off between run time and solution quality, where as the demands on the solution quality increase the necessary run time will approach the impractical run times needed for a direct enumeration approach to the TSP.

6.3.1 An alternative approach

The HONN approach to the solution of the TSP that we have described was first proposed by Cooper (Cooper, 1995a). An alternative approach to the solution of the TSP with a higher-order recurrent network was recently proposed by Matsui (Matsui and Nakabayashi, 1995). While the higher-order recurrent network they use does not have a Lyapunov function, it is similar to our approach in that E^{obj} is a cubic function of the network state $\mathbf{v}$, *viz.*

$$E^{obj} = \sum_{\substack{xyz \\ x\neq y\neq z}} \sum_{i} (d_{xy} + d_{yz}) v_{xi} (v_{y,i+1} + v_{y,i-1})(v_{z,i+1} + v_{z,i-1}). \tag{6.11}$$

Equation (6.11) may be compared to the objective function which we have proposed for a second-order HONN, as given in equation (6.7). There are several significant differences between our HONN approach to the TSP and that suggested by Matsui and Nakabayashi.

- Firstly, it is not apparent that minimising the objective function given by equation (6.11) will lead to short tours. Although equation (6.11) does equate to four times the tour length at a valid $0-1$ point, its meaning at points inside the hypercube is not clear. Consider the term

  $$(d_{xy} + d_{xz}) v_{xi} v_{y,i+1} v_{z,i+1}$$

  which appears in the objective function of equation (6.11). This term measures the distance between a city x occuring in position i of the tour and *two* cities y and z occuring in position $i+1$ of the tour; but as we know, for the network state to represent a valid tour only *one* city can occupy each position in the tour. There is no sound basis for including this and other similar terms into the objective function for the TSP.

  In contrast, the only term in the objective function given by equation (6.7) for our approach to the TSP measures the length of a path visiting cities x, y and z in consecutive positions on the tour. Quite obviously, minimising the objective function that we have proposed will lead to short tours for the travelling salesman.

- The objective function given by equation (6.11) does not correspond to a Lyapunov function for the higher-order network that has been proposed in (Matsui and Nakabayashi, 1995). Consequently, the value of the objective function may increase at any iteration of their algorithm.
- Finally, the results of their simulations are questionable. They report that a Hopfield network solving a twenty city TSP obtained solutions which were in the vicinity of 90% longer than the optimal solution. This should be compared to the simulations presented in Section 5.1 where on a similar twenty city problem we showed the Hopfield network obtained solutions which were only 0.27% longer than the optimal. While this massive performance difference may well be attributable to the rigorous problem mapping and principled annealing mechanisms that we have used, it also casts doubt on any conclusions drawn from a comparison between their higher-order recurrent network and the Hopfield network when operating in such an environment.

6.4 Simulations

In this section we present an experimental comparison of the Hopfield network and HONN approaches to the TSP. For the purposes of this comparison we have solved 1000 instances of a 10-city Euclidean TSP, where, for each instance, the cities have been placed randomly inside the unit square and the optimal tour has been found by exhaustive search[2]. Each instance of the TSP was solved with a Hopfield network, a second-order HONN and a third-order HONN.

The network dynamics are given by equation (6.5) for the Hopfield network and equation (6.8) for the second-order HONN. The network dynamics for a third-order HONN, where the objective function is given by equation (6.10), can be determined by setting the Lyapunov function to $E^{lyap} = E^{obj} + E^{cns} + E^{mod}$ and evaluating the partial derivative $\partial E^{lyap}/\partial v_{xi}$. In all cases integration of the network dynamics was performed using the function `ode15s.m` available in MATLAB®.

For each of the optimisation networks used in this experiment we have employed modified hysteretic annealing. The annealing parameter was given by $\gamma = (t/\tau)^2 + \gamma_0$ where τ is a positive constant and γ_0 is a small negative constant. Increasing the absolute values of both τ and γ_0 will improve the solution quality at the expense of longer running time. For the experiments reported in this section, we set the network parameters as follows: $\tau = 100$, $\gamma_0 = -1$ and the gain of the transfer function was set to $T^p = 1$. For each instance of the TSP the weight c on the penalty function E^{cns} was set so that

$$c = c_0 N \psi \max_{x,y,z}(d_{xy} + d_{xz})$$

where $c_0 = 1$ and

$$\psi = \begin{cases} 1 & \text{for the Hopfield network} \\ 3 & \text{for the second-order HONN} \\ 6 & \text{for the third-order HONN.} \end{cases}$$

This method of setting the weight on the penalty function E^{cns} is consistent with results recently presented in the literature (Abe and Gee, 1995). For each simulation the initial conditions were given by $v_{xi} = 1/N + 0.01 * \texttt{rand}$ where `rand` is a random value in the range $[-0.5, 0.5]$.

[2]The problem database used for these simulations was developed and made available by A. H. Gee of the University of Cambridge, England.

Order	Mean % error	Number of Optimal tours
1 (Hopfield)	1.9983	477
2	1.8013	523
3	1.6497	525

Table 6.1: Simulation results for HONNs solving the TSP.

The performance of the Hopfield network and second and third-order HONNs on 1000 *instances of a* 10*-city TSP is shown. For each order of network, the mean percentage error and the number of optimal solutions found is reported. The percentage error is given by* $(L_{found} - L_{opt})/L_{opt} \times 100\%$ *where* L_{found} *is the length of the tour found by the optimisation network and* L_{opt} *is the length of the optimal tour which has been found for each TSP by exhaustive search.*

The results of these experiments can be seen in Table 6.1. While the performance of the three networks is quite similar, Table 6.1 shows that the mean percentage error decreases and the number of optimal solutions found increases as the order of the network is increased. While the advantages of increasing the order of the network are evident in these results, the difference between the Hopfield network and HONNs is not astounding. This should be expected as we know that the Hopfield network performs satisfactorily on problems as small as these 10-city TSPs. We would also expect that HONNs will significantly outperform the Hopfield network on larger problems, but we have not verified this by simulation as the ability of HONNs to scale to large problem sizes was adequately demonstrated on the Ising spin problem in Chapter 5. The main aim of this chapter, which we have quite clearly accomplished, was to demonstrate how to apply HONNs to the solution of difficult combinatorial optimisation problems like the travelling salesman.

6.5 Chapter Summary

Our investigations in previous chapters have suggested that higher-order terms in the dynamics of an optimisation network may be used to improve solution quality. Such higher-order terms allow more sophisticated heuristics to be embedded into the dynamics of an optimisation network. In this chapter we have formally introduced higher-order neural networks (HONNs) for optimisation. HONNs are a new class of optimisation techniques that may be applied to a wide variety of combinatorial optimisation problems.

HONNs are basically an extension of the Hopfield network where the feedback vector is no longer restricted to be linear, but is allowed to be a polynomial function of the network output $\mathbf{v}$. A HONN forms a dynamic system described by

$$\begin{aligned}\frac{du_i}{dt} &= \sum_{k=1,\, i\in\mathcal{I}_k}^{L} T_k \frac{m_i(k)}{v_i} \prod_{j\in\mathcal{I}_k} v_j^{m_j(k)} - \eta u_i \\ v_i &= g(u_i).\end{aligned}$$

Here η is a decay parameter, $\mathcal{I} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots \mathcal{I}_L\}$ is a collection of L unordered subsets of the indices $\{1, 2, \ldots N\}$, T_k is the synaptic weight applied to the product of the outputs of neurons in the unordered subset $\mathcal{I}_k$ and $m_j(k)$ is the power to which the output of neuron j is raised when calculating the product of neuron outputs in subset $\mathcal{I}_k$. The transfer function $g(\cdot)$ is a monotonically increasing function that restricts the neuron outputs to the range $v_i \in [0, 1]$. A HONN admits a Lyapunov function

$$E^{lyap}(\mathbf{u}) = -\sum_{k=1}^{L} T_k \prod_{j \in \mathcal{I}_k} v_j^{m_j(k)} + \eta \sum_i \int_{0.5}^{v_i} g^{-1}(V) dV.$$

As with the Hopfield network, HONNs operate by a process of gradient descent on the Lyapunov function. While the work reported in this thesis is amongst the first applications of HONNs to optimisation, HONNs have been applied as associative memories and have also been used to solve the task of grammatical inference.

The major goal of this chapter was to demonstrate the application of HONNs to difficult combinatorial optimisation problems like the TSP. In order to achieve this it was first necessary to ascertain the approach that the Hopfield network has employed in solving the TSP. When using the Hopfield network to solve the TSP, the objective function is given by

$$E^{obj} = \sum_{\substack{xy \\ x \neq y}} \sum_i d_{xy} v_{xi} v_{y,i+1}.$$

Our analysis revealed that an analogue version of the nearest neighbour heuristic is embedded into the dynamics of the Hopfield network. The Hopfield network effectively constructs a tour by encouraging the formation of locally optimal segments which join two cities. The HONN approach to the TSP was then formulated by extending the neighbourhood in which locally optimal segments are encouraged. For a second-order HONN approach to the TSP we suggested the objective function

$$E^{obj} = \sum_{\substack{xyz \\ x \neq y \neq z}} \sum_i (d_{xy} + d_{yz}) v_{xi} v_{y,i+1} v_{z,i+2}.$$

Analysis of the dynamics for the second-order HONN suggests that the network constructs a tour by encouraging the formation of locally optimal paths which connect three cities.

The concept of enlarging the neighbourhood in which locally optimal segments are encouraged may be further extended by considering even higher-order networks. As the order of the HONN increases, the heuristic which is embedded into the dynamics becomes a progressively stronger approach to the solution of the TSP. However, as the order of the network increases the number of connections necessary to implement the network increases exponentially. Consequently, we may view HONNs as a family of heuristic approaches to combinatorial optimisation where we may trade increased computational effort for improved solution quality. The trade-off between solution quality and computational effort is the subject of more detailed discussion in Chapter 7.

Chapter VII

Investigating the Quality Versus Computational Effort Trade-off

To this point in the thesis we have highlighted the need for improved heuristics for optimisation networks and subsequently developed the higher-order neural network approach to combinatorial optimisation. We have applied HONNs to the solution of the Ising spin and travelling salesman problems. While experimental evidence has supported our motivation for using HONNs, in this chapter we will develop a deeper understanding of HONNs by investigating the trade-off that can be made between network complexity and solution quality. An important outcome of this investigation is a clear explanation of the shortcomings of the Hopfield network approach to combinatorial optimisation.

We begin in Section 7.1 by considering the operation of an optimisation network as a dynamic system whose stable attractors can be interpreted as solutions to an optimisation problem. We discuss how the location and number of attractors in the system affects both the validity and quality of solutions found by an optimisation network. We proceed in Section 7.2 by establishing the conditions under which, for the TSP, a vertex of the valid subspace is a stable attractor for both the Hopfield network and HONNs. When the optimal solution to the TSP satisfies these conditions for a given network, we can obtain a measure of the expected solution quality by determining the number of competing attractors in that network. In Section 7.3 we present a numerical comparison of the number of stable attractors in various networks when operating on two example TSPs. The results of these comparisons support our assertion that HONNs are a family of solution techniques that embody a trade-off between solution quality and computational effort. In Section 7.4 we discuss the implications of our results for the performance of HONNs and show that they clearly explain the inadequacies of the Hopfield network approach to combinatorial optimisation. The impact of annealing on our quantitative results is also considered, along with a discussion of further work which may be undertaken. Finally in Section 7.5 we present a summary of the chapter.

7.1 Attractors and Basins of Attraction

Throughout this thesis we have considered optimisation networks simply as an algorithm for minimising a quadratic function of the network state. While such an algorithmic treatment of optimisation networks has afforded great insights into their operation and allowed us to formulate an approach to combinatorial optimisation, it is interesting to shift our emphasis slightly by considering the operation of an optimisation network as that of a dynamic system. Viewing an optimisation network as a dynamic system does not invalidate any of our previous analysis, for the network is unchanged, but it will motivate the analysis that we present in this chapter.

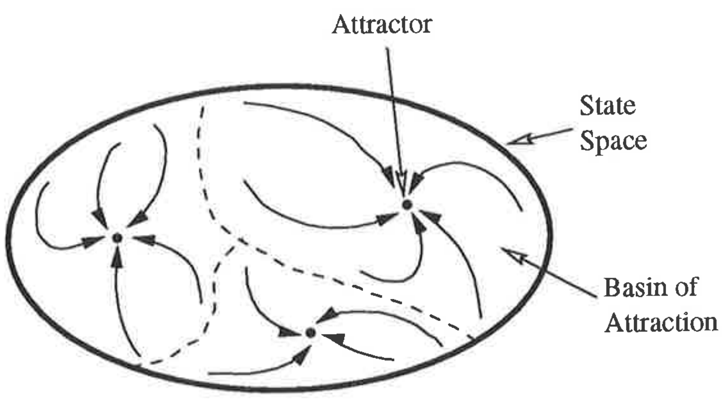


Figure 7.1: Optimisation networks are dynamic systems with stable attractors.

The Lyapunov function for an optimisation network defines an energy landscape over the entire state space. The minima of the Lyapunov function are attractors for the network dynamics. Each attractor has an associated basin of attraction which corresponds to a "valley" in the Lyapunov function for the network.

Quite clearly, an optimisation network is a dynamic system where the network output $\mathbf{v}$ moves about the state space under the action of the network equations. Moreover, our analysis of the Lyapunov function for the Hopfield network, as presented in Section 2.5, and for HONNs, as presented in Section 6.1, shows that $\mathbf{v}$ will converge to a stable attractor. Each attractor in the system corresponds to a minimum in the network's Lyapunov function and is therefore located at the bottom of a "valley" in the Lyapunov function. As the system dynamics perform a gradient descent on the Lyapunov function, the basin of attraction for each attractor is defined by the corresponding "valley" in the Lyapunov function. When utilising an optimisation network for the solution of a combinatorial optimisation problem the system is given some initial condition $\mathbf{v}_0$ for the state vector $\mathbf{v}$. The system dynamics will then move the state vector $\mathbf{v}$ toward the attractor at the bottom of the basin of attraction in which $\mathbf{v}_0$ is situated. When $\mathbf{v}$ has converged to that attractor it is interpreted as the solution to the combinatorial problem. A schematic representation of an optimisation network as a dynamic system is shown in Figure 7.1, where we can see that the entire state space is partitioned into various basins of attraction.

As the attractors of the dynamic system are interpreted as solutions to the optimisation problem, both the problem mapping and annealing techniques that we have used must impact upon the attractors in an optimisation network. In order to understand the nature of the impact of annealing and problem mapping, we begin by making several remarks with regard to how the location and number of attractors of the dynamic system affect the validity and quality of the solutions found by the network:

Location of attractors - Firstly, as the attractors of the system are interpreted as solutions to the optimisation problem, we can guarantee that no invalid solutions are found by ensuring that all attractors for the system lie in the valid subspace. Secondly, the attractors must be situated sufficiently close to a $0-1$ point so as to be interpretable solutions to the combinatorial problem.

Number of attractors - As the number of attractors increases, the valid subspace is partitioned into more basins of attraction. With more attractors in the valid subspace, the likelihood of placing the initial state $\mathbf{v}_0$ into the basin of attraction of the optimal solution is decreased. Consequently, we can expect fewer optimal solutions to be found.

Size of the basin of attraction - To obtain a good solution quality from the optimisation network it is desirable for good solutions to the problem to have large basins of attraction. Ideally we would like the optimal solution to the optimisation problem to be the only attractor in the valid subspace with a basin of attraction encompassing the entire state space. However, numerical results presented in Section 7.3 will show that this is impractical.

From the above discussion, we see that to guarantee valid solutions we must ensure that all attractors lie in the valid subspace. It should be remembered that the problem mapping discussed in Chapter 3 guaranteed that the network state would always remain valid by defining suitable penalty functions E^{cns} to be included in the network's Lyapunov function. Therefore, we can infer that the penalty functions E^{cns} guarantee valid solutions by ensuring that all attractors for the system lie in the valid subspace.

The impact of annealing on the location and number of attractors in an optimisation network is somewhat more complicated but our understanding is aided by considering the simple example of annealing presented in Section 4.2. With reference to that example, when the annealing parameter γ is set to a sufficiently large negative value there will be only one attractor in the valid subspace and the network state $\mathbf{v}$ will move towards it (see Figure 4.2(a)). As the annealing parameter is gradually increased more attractors appear in the system and consequently the valid subspace is further partitioned into separate basins of attraction (see Figure 4.2(c)). Furthermore, we know that as the annealing parameter is increased still further, annealing encourages convergence to a $0-1$ point. This is achieved by forcing the attractors to the vertices of the valid subspace. The key to the success of annealing in improving the solution quality is to ensure that the attractor which corresponds to the optimal solution is among the first attractors to be included into the system. Obviously, the more attractors that have been included into the system before the optimal solution becomes an attractor, the less likely it is that $\mathbf{v}$ will converge to the optimal solution.

7.2 Stability criteria for valid $0-1$ points

In this section we will establish the conditions under which an arbitrary solution to a TSP is an attractor for both the Hopfield network and the HONN.

The first step towards our goal is to establish the criteria under which a $0-1$ point is an attractor for the system dynamics. We begin by noting that when the annealing techniques described in Chapter 4 are used to improve solution quality, it is common practice to set the decay parameter η equal to zero. Consequently, if a $0-1$ point is to be an attractor the internal state $\mathbf{u}$ of the network will grow without bound at that point. Evidence of this is shown in the simple examples of the Hopfield network and the HONN presented in Sections 2.5.1 and 6.1.1[1]. We will consider a $0-1$ point $\mathbf{v}$ to be an attractor if that point is an asymptotically stable equilibrium point of the network

[1]Note that in these simple examples annealing was not employed. However, to ensure that $E^{lyap} = E^{obj}$ the decay parameter η was set to zero and we saw that in each case $\mathbf{u}$ grows without bound at the $0-1$ point to which the network converged.

dynamics, *i.e.*

$$\frac{dv_{xi}}{dt} \to 0 \qquad \forall\ x, i \in \{1, \ldots N\}. \tag{7.1}$$

Furthermore, as $v_{xi} \in [0,1]$ we require that $\frac{dv_{xi}}{dt} \to 0$ from above if $v_{xi} = 1$, and $\frac{dv_{xi}}{dt} \to 0$ from below if $v_{xi} = 0$. Note that since $v_{xi} = g(u_{xi})$ we have

$$\frac{dv_{xi}}{dt} = \frac{dg}{du_{xi}} \frac{du_{xi}}{dt}. \tag{7.2}$$

Here $\frac{dg}{du_{xi}}$ is the slope of the sigmoid function, which, although it is always positive, does approach zero as $u_{xi} \to \infty$ or $u_{xi} \to -\infty$. Therefore, from equations (7.1) and (7.2) we can deduce that a $0-1$ point $\mathbf{v}$ will be asymptotically stable if

$$\frac{du_{xi}}{dt} > 0 \ \text{ for } \ v_{xi} = 1 \tag{7.3}$$

and

$$\frac{du_{xi}}{dt} < 0 \ \text{ for } \ v_{xi} = 0. \tag{7.4}$$

Examination of equations (7.3) and (7.4) shows that when $\mathbf{v}$ is an asymptotically stable $0-1$ point $\mathbf{u}$ will grow without bound.

7.2.1 The Hopfield network

Now that we have established the criteria for which a $0-1$ point $\mathbf{v}$ is an asymptotically stable equilibrium point, we turn our attention to the problem mapping of the TSP for the Hopfield network. Once again, to map the TSP onto the Hopfield network we set the Lyapunov function to be

$$E^{lyap} = E^{obj} + E^{cns} + E^{mod}$$

where E^{obj} is the objective function for the TSP, E^{cns} is the penalty function defined by the valid subspace problem mapping and E^{mod} is the modified hysteretic annealing function. While complete details of the problem mapping may be found in Section 6.2, we take as our starting point the dynamic equation for the network which is given by equation (6.5) and which for completeness we restate as

$$\begin{aligned} \frac{du_{xi}}{dt} &= -\frac{(c+\gamma)}{N}\left(\sum_y v_{yi} - 1\right) - \frac{(c+\gamma)}{N}\left(\sum_j v_{xj} - 1\right) + \frac{(c+\gamma)}{N^2}\left(\sum_{yj} v_{yj} - N\right) \\ &\quad + \gamma\left(v_{xi} - \frac{1}{N}\right) - \sum_{\substack{y \\ y \neq x}} d_{xy}(v_{y,i+1} + v_{y,i-1}). \end{aligned} \tag{7.5}$$

Here c is the weight on the penalty function, γ is the annealing parameter and d_{xy} is the distance between cities x and y.

From the analysis presented in Chapter 3 we know that the penalty functions E^{cns} will correctly constrain $\mathbf{v}$ to lie on the valid subspace. Furthermore, we have established that annealing will force $\mathbf{v}$ to a vertex of the valid subspace, which in the case of the TSP will be a valid $0-1$ point (Gee and Prager, 1994). Consequently, we will confine our stability analysis to consider only the valid $0-1$ points. When $\mathbf{v}$ lies on the valid subspace the first three terms in equation (7.5) will equate to zero, in which case the dynamic equation reduces to

$$\frac{du_{xi}}{dt} = \gamma\left(v_{xi} - \frac{1}{N}\right) - \sum_{\substack{y \\ y \neq x}} d_{xy}(v_{y,i+1} + v_{y,i-1}). \tag{7.6}$$

Now for a valid $0-1$ point to be an asymptotically stable equilibrium point, equation (7.3) or equation (7.4) must be satisfied for each neuron v_{xi}. In order for equation (7.3) to be satisfied for each neuron in the network where $v_{xi} = 1$, we can deduce from equation (7.6) that

$$\gamma > \frac{N}{N-1} \max_{\overline{yx}, \overline{xz} \in \text{tour}(\mathbf{v})} (d_{yx} + d_{xz}). \tag{7.7}$$

Here the notation $\overline{yx}, \overline{xz} \in \text{tour}(\mathbf{v})$ indicates that the segments $\overline{yx}$ and $\overline{xz}$ occur in the tour described by vertex $\mathbf{v}$ of the valid subspace (alternatively y, x and z are consecutive cities in the tour described by $\mathbf{v}$). Similarly, for equation (7.4) to be satisfied for each neuron in the network where $v_{xi} = 0$, we can deduce from equation (7.6) that

$$\gamma > -N \min_{x \neq y} d_{xy}. \tag{7.8}$$

It is apparent from equations (7.7) and (7.8) that the annealing parameter γ determines the stability of a vertex of the valid subspace. When $\gamma < 0$, equation (7.7) cannot be satisfied and therefore no vertex of the valid subspace is stable. However, we should remember that during the course of the annealing process the annealing parameter is gradually increased, and when $\gamma > 0$ equation (7.8) is satisfied for all valid $\mathbf{v}$ and the stability of a vertex of the valid subspace is then determined solely by equation (7.7). Moreover, we see from equation (7.7) that the stability of a vertex of the valid subspace is determined not by the length of the tour it describes but by the maximum length of a segment joining three consecutive cities in that tour.

As the annealing parameter γ is increased, there will be a critical value $\gamma = \gamma_c$ where the first vertex of the valid subspace becomes stable. It is important to realise that the tour described by that vertex of the valid subspace is by no means guaranteed to be the optimal tour (*i.e.* the minimal length tour). Indeed, it is likely that the annealing parameter must be increased to a level $\gamma_{opt} > \gamma_c$ before the vertex of the valid subspace which describes the optimal tour is stable. Moreover, when $\gamma = \gamma_{opt}$ it is most likely that there will be many other vertices of the valid subspace which will also be stable, each of which corresponds to a sub-optimal tour. Each of these stable vertices of the valid subspace will have a basin of attraction and consequently the valid subspace will be partitioned into many different basins of attraction. If we were to place the network output $\mathbf{v}$ at a random position in the valid subspace then it is quite likely that the network will converge to a vertex of the valid subspace which corresponds to a sub-optimal tour. Obviously the method by which the Hopfield network attempts to solve the TSP does not provide the necessary discrimination between optimal and sub-optimal tours to guarantee that the network will reliably converge to the optimal solution.

7.2.2 HONNs

We will now consider the HONN approach to the TSP and establish the criteria under which a valid $0-1$ point is an asymptotically stable equilibrium point for the HONN.

Once again, to map the TSP onto an HONN we set the Lyapunov function to be

$$E^{lyap} = E^{obj} + E^{cns} + E^{mod}$$

where E^{cns} is the penalty function defined by the valid subspace problem mapping, E^{mod} is the modified hysteretic annealing function and E^{obj} is the objective function for the TSP. In this section we will consider a second-order HONN for the solution of the TSP and so the objective function is given by

$$E^{obj} = \sum_{\substack{xyz \\ x\neq y\neq z}} \sum_i (d_{xy} + d_{yz}) v_{xi} v_{y,i+1} v_{z,i+2}.$$

While complete details of the problem mapping may be found in Section 6.3, we will take as our starting point the dynamic equation for the network which is given by equation (6.8) and which for completeness we restate as

$$\begin{aligned}\frac{du_{xi}}{dt} &= -\frac{(c+\gamma)}{N}\left(\sum_y v_{yi} - 1\right) - \frac{(c+\gamma)}{N}\left(\sum_j v_{xj} - 1\right) \\ &\quad + \frac{(c+\gamma)}{N^2}\left(\sum_{yj} v_{yj} - N\right) + \gamma\left(v_{xi} - \frac{1}{N}\right) \\ &\quad - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{xy} + d_{yz}) v_{y,i+1} v_{z,i+2} - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yx} + d_{xz}) v_{y,i-1} v_{z,i+1} \\ &\quad - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yz} + d_{zx}) v_{y,i-2} v_{z,i-1}. \end{aligned} \qquad (7.9)$$

Here c is the weight on the penalty function, γ is the annealing parameter and d_{xy} is the distance between cities x and y.

Once again we will confine our analysis to an examination of the stability of vertices of the valid subspace. When $\mathbf{v}$ lies on the valid subspace the first three terms in equation (7.9) will equate to zero, in which case the dynamic equation reduces to

$$\begin{aligned}\frac{du_{xi}}{dt} &= \gamma\left(v_{xi} - \frac{1}{N}\right) - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{xy} + d_{yz}) v_{y,i+1} v_{z,i+2} \\ &\quad - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yx} + d_{xz}) v_{y,i-1} v_{z,i+1} - \sum_{\substack{yz \\ y\neq z\neq x}} (d_{yz} + d_{zx}) v_{y,i-2} v_{z,i-1}. \end{aligned} \qquad (7.10)$$

Note that a vertex $\mathbf{v}$ of the valid subspace is a valid $0-1$ point and therefore if $\mathbf{v}$ is to be an asymptotically stable equilibrium point equation (7.3) or equation (7.4) must be satisfied for each neuron v_{xi}. In order for equation (7.3) to be satisfied for each neuron in the network where $v_{xi} = 1$, we can deduce from equation (7.10) that

$$\gamma > \frac{N}{N-1} \max_{\overline{vw}, \overline{wx}, \overline{xy}, \overline{yz} \in \text{tour}(\mathbf{v})} (d_{vw} + 2d_{wx} + 2d_{xy} + d_{yz}). \qquad (7.11)$$

Here the notation $\overline{vw}, \overline{wx}, \overline{xy}, \overline{yz} \in \text{tour}(\mathbf{v})$ indicates that the segments $\overline{vw}, \overline{wx}, \overline{xy}$ and $\overline{yz}$ occur in the tour described by vertex $\mathbf{v}$ of the valid subspace (alternatively v, w, x, y and z are consecutive cities in the tour described by $\mathbf{v}$). Similarly, for equation (7.4) to be satisfied for each neuron in the network where $v_{xi} = 0$, we can deduce from equation (7.10) that

$$\gamma > -N \min_{\overline{xy} \in \text{tour}(\mathbf{v}),\, \overline{yz} \notin \text{tour}(\mathbf{v})} (d_{xy} + d_{yz})\,. \tag{7.12}$$

It is apparent from equations (7.11) and (7.12) that the stability of a vertex of the valid subspace is once again determined by the annealing parameter γ. Note that when $\gamma < 0$ equation (7.11) cannot be satisfied and therefore no vertex of the valid subspace can be stable. However, during the course of the annealing process the annealing parameter will be increased and when $\gamma > 0$ equation (7.12) is satisfied and the stability of any vertex of the valid subspace is then determined solely by equation (7.11). Furthermore, it is interesting to note from equation (7.11) that the stability of a vertex of the valid subspace is not determined by the length of the tour it describes, but instead is determined by a function of the intercity distances on a segment of the tour which passes through five consecutive cities. This is in clear contrast with the Hopfield network, where the stability of a vertex of the valid subspace was determined by the maximum length of a segment joining three consecutive cities in the tour.

Despite the seemingly stricter stability criteria for a vertex of the valid subspace in a second-order HONN as compared to a Hopfield network, it is still possible that the first vertex of the valid subspace to become stable as the annealing parameter is increased will not correspond to the optimal solution to the TSP. Indeed, when the annealing parameter has been increased to a level $\gamma = \gamma_{opt}$ where equation (7.11) is satisfied for the vertex of the valid subspace which corresponds to the optimal tour, it is likely that many other vertices of the valid subspace will also be stable. However, we suggest that the number of vertices of the valid subspace which are stable when $\gamma = \gamma_{opt}$ will be greatly reduced in the second-order HONN as compared to the Hopfield network. Such a reduction in the number of stable vertices of the valid subspace is a consequence of the stricter stability criteria of equation (7.11) for the HONN as compared to equation (7.7) for the Hopfield network. Consequently, the valid subspace will be partitioned into fewer basins of attraction than was the case with the Hopfield network, and so we can expect the second-order HONN, when given a random starting point in the valid subspace, to be more likely to find the optimal tour.

As was shown in Section 6.3, third or even higher-order objective functions may be used in a HONN to solve the TSP. For each network, analogous conditions to those given in equations (7.11) and (7.12) may be derived for the asymptotic stability of a vertex of the valid subspace. In much the same way as we have suggested for the second-order HONN, we would expect that the stability criteria for even higher-order networks would provide a significantly greater level of discrimination between optimal and sub-optimal tours. Consequently, as the order of the network is increased there should be a reduction in the number of vertices of the valid subspace which are stable when the annealing parameter is set so that the vertex which corresponds to the optimal tour is only just stable. Furthermore, such a reduction in the number of stable vertices of the valid subspace results in the valid subspace being partitioned into fewer basins of attraction. With fewer basins of attraction in the valid subspace, the likelihood of the network converging from a random starting point to the optimal solution will increase.

7.3 Numerical comparison of the number of stable points

In the previous section we asserted that the number of vertices of the valid subspace which are stable when the optimal tour is only just stable will decrease as the order of the network is increased. In this section, we aim to justify that assertion by presenting a numerical comparison of the number of stable vertices of the valid subspace for HONNs of increasing order when operating on two example problems.

7.3.1 Procedure

To undertake a numerical comparison of the number of stable points for HONNs of increasing order, we must first find the optimal tour for each of our chosen problems. In turn, this will allow us to calculate the value of the annealing parameter γ_{opt} at which the optimal tour is only just stable. We stress that knowledge of the optimal tour is only necessary to facilitate a numerical comparison of the number of stable points for HONNs of increasing order. If we were to actually simulate the operation of these networks then we would use an annealing mechanism to encourage good solutions, in which case it is not necessary to calculate γ_{opt} as the annealing process initially sets γ to a large negative value and gradually increases it without reference to γ_{opt}. The first of our chosen problems is a 10-city Euclidean TSP where the cities have been placed randomly in the unit square according to a uniform distribution. In this case the optimal tour is easily found by exhaustive search. The final problem we have considered is `bayg29`, a 29 city Euclidean TSP which may be found in the widely available TSPLIB (Reinelt, 1991) database. The optimal solution for `bayg29` has been found by an exact solution method and is also found in the TSPLIB database.

Given the optimal solution to our chosen problems we must then determine, for each order of HONN that we wish to examine, the value of the annealing parameter γ_{opt} at which the optimal solution to the TSP is only just stable. For the Hopfield network γ_{opt} is calculated by evaluating the stability condition given in equation (7.7) for the vertex $\mathbf{v}$ of the valid subspace which represents the optimal solution. Similarly, γ_{opt} for the second-order HONN is calculated by evaluating equation (7.11) with $\mathbf{v}$ corresponding to the optimal tour. For higher-order networks it is a simple matter to determine analogous stability conditions to those given in equations (7.7) and (7.11). To determine the value of γ_{opt} for a HONN of a particular order the appropriate stability condition is evaluated for the vertex of the valid subspace which represents the optimal tour.

For the 10-city problem we want to examine the stability of every vertex of the valid subspace for HONNs of order 1 (*i.e.* a Hopfield network) through to 10. To accomplish this we proceed by selecting a particular HONN and then setting the annealing parameter at the appropriate value of γ_{opt} such that the optimal solution is only just stable. We then enumerate all possible tours, testing each tour to determine if it satisfies the stability condition for the current network (the stability condition is given by the analogy to equations (7.7) and (7.11) which is appropriate for the current HONN). This procedure is performed for each HONN that we wish to examine. The results for this experiment are given in Section 7.3.3.

The numerical comparison is performed in a similar fashion for the `bayg29` problem. The only significant difference is that we do not enumerate all possible tours as the number of tours makes it prohibitive to do so. Instead, we have generated a set of one hundred thousand near-optimal tours to examine. Starting from the optimal tour, the near-optimal tours were generated by performing a series of segment reversal and node insertion moves. To ensure that the tours generated were near-optimal, the maximum number of such moves performed in succession was forty, after which the sequence was repeated starting once again from the optimal solution. For the `bayg29` problem we examined HONNs of order 1 through to 16. The results for this experiment are presented in Section 7.3.3.

7.3.2 Quality Factors

Until now we have only been interested in the number of vertices of the valid subspace which are stable when the annealing parameter is set so that the vertex corresponding to the optimal tour is stable. While this will give some indication of the relative performance of HONNs of different orders, it is also interesting to investigate the quality of the stable points which correspond to sub-optimal tours. Consequently we define a measure of quality Q_1 which is simply the normalised mean length of all stable tours

$$Q_1(p) = \frac{\sum_{\mathbf{v} \in \mathcal{S}(p)} L(\mathbf{v})}{\|\mathcal{S}(p)\| L(\mathbf{v}_{opt})}.$$

Here $\mathcal{S}(p)$ is the set of vertices of the valid subspace which are stable for a HONN of order p when the annealing parameter is set so that the optimal tour is only just stable, $\|\mathcal{S}(p)\|$ is the number of vertices in the set $\mathcal{S}(p)$, $L(\mathbf{v})$ is the length of the tour described by vertex $\mathbf{v}$ of the valid subspace and $L(\mathbf{v}_{opt})$ is the length of the optimal tour.

By weighting the length of each tour equally, the quality measure Q_1 implicitly assumes that each stable vertex of the valid subspace is equally likely to be found by the network. However, with a random starting point in the valid subspace, the probability of converging to a particular stable vertex is directly related to the size of its basin of attraction. Therefore, an improved measure of quality is given by the normalised weighted mean length of all stable tours, where the weighting factor is a measure of the size of the appropriate basin of attraction. Unfortunately, it is an exceptionally difficult problem to exactly determine the size of a basin of attraction in a dynamic system. Instead we will weight each stable vertex by the factor

$$M(\mathbf{v}) = \sum_{xi} \left| \frac{du_{xi}}{dt} \right|$$

where $|\cdot|$ denotes the absolute value. Note that $M(\mathbf{v})$ is a measure of the total derivative of the internal state $\mathbf{u}$ at an asymptotically stable vertex $\mathbf{v}$. When a vertex $\mathbf{v}$ is only just stable the corresponding basin of attraction is small. Moreover, as the stability conditions given in equations (7.3) and (7.4) have only just been satisfied, the total derivative of the internal state $\mathbf{u}$ will also be small. As the annealing parameter is increased, the total derivative of the internal state $\mathbf{u}$ will also increase. We would also expect that as the annealing parameter is increased, the basin of attraction for vertex $\mathbf{v}$ will expand. Therefore, while $M(\mathbf{v})$ is by no means an exact measure of the size of the basin of attraction for vertex $\mathbf{v}$, we do expect that it is somehow related. With the weighting for each stable vertex given by $M(\mathbf{v})$, we define the measure of quality Q_2 as

$$Q_2(p) = \frac{\sum_{\mathbf{v} \in \mathcal{S}(p)} L(\mathbf{v}) M(\mathbf{v})}{\|\mathcal{S}(p)\| L(\mathbf{v}_{opt}) M(\mathbf{v}_{opt})}.$$

7.3.3 Results

In this section we present results for both the 10-city and `bayg29` experiments.

- The results for the 10-city experiment are shown in Table 7.1, where we see quite clearly that the number of tours which are stable when the optimal tour is only just stable decreases as the order of the network is increased. Note that the figures given in Table 7.1 refer to the number of stable tours, each of which will correspond to $2N$ vertices of the valid subspace due to the inherent degeneracy in representing a tour with a $N \times N$ array of neurons.

- Furthermore, the measure of quality given by Q_1 indicates that as the order of the network is increased, the mean length of the tours that remain stable is becoming smaller. This suggests that the quality of solution will improve as the order of the network increases.

- We also note that for each order of HONN Q_2 is greater than Q_1, which suggests that longer tours have been weighted more heavily than shorter tours. As the weighting factor $M(\mathbf{v})$ is related to the size of the the basin of attraction for a stable vertex $\mathbf{v}$ of the valid subspace, we suggest that the longer tours have larger basins of attraction.

The results for the `bayg29` problem are shown in Figure 7.2. Once again we see that the number of stable vertices decreases as the order of the network increases. While there are approximately 8000 stable tours for a HONN of order 1, there are only two stable tours (including the optimal tour) for a HONN of order 16. In addition, the quality measures Q_1 and Q_2 both suggest that the quality of solution will improve as the order of the network increases.

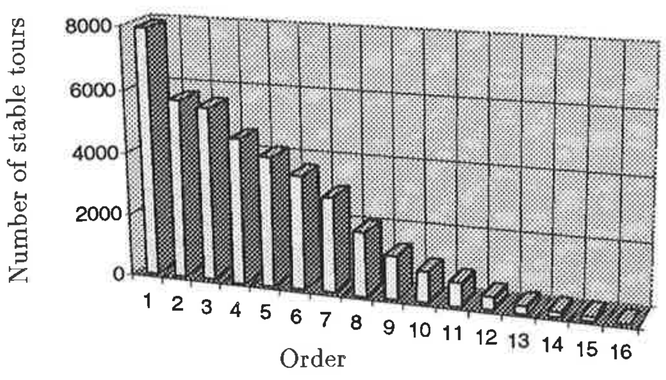
The results presented in this section support our assertion that the number of vertices of the valid subspace which are stable when the optimal tour is only just stable will decrease as the order of the network is increased. But what is the relevance of this fact to the operation of an optimisation network? By determining the number of vertices of the valid subspace which are stable, we have effectively measured the probability of finding the optimal tour when starting from a random point in the valid subspace. While this observation does give great insight into the operation of these networks, it is a valid observation only if the network is run with the annealing parameter held at a constant value of γ_{opt}. For a general problem we will not know the value of γ_{opt} and moreover, by holding γ constant we have discarded the benefits that may be gained by properly utilising annealing. This will be the subject of further comment in Section 7.4.

A further factor which is worthy of comment is that extremely high-order networks are required to ensure that the only stable vertices of the valid subspace correspond to the optimal tour. For example in the `bayg29` problem even a HONN of order sixteen has two stable tours. We know that the space complexity of a HONN increases exponentially with the order of the network and so it is certainly not feasible to build a HONN of order 16. However, the necessity for such extremely high-order networks in order to guarantee the optimal solution is consistent with the trade-off between computational effort (in this case space complexity of the HONN) and solution quality, that we have developed as a major theme for this thesis. While it is infeasible to build a HONN of extremely high-order to guarantee the optimal solution, the numerical comparisons we have performed do reveal the extent to which the Hopfield network has accepted a trade-off resulting in a simple network that will produce solutions of poor quality.

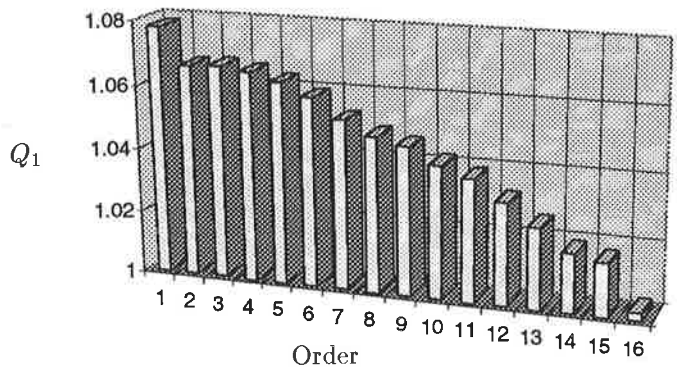
Order	Number of stable tours	Q_1	Q_2
1 (Hopfield)	233	1.2844	1.5455
2	11	1.1028	1.1909
3	2	1.0341	1.0623
4	2	1.0292	1.0532
5	1	1.0000	1.0000
6	1	1.0000	1.0000
7	1	1.0000	1.0000
8	1	1.0000	1.0000
9	1	1.0000	1.0000
10	1	1.0000	1.0000

Table 7.1: Numerical comparison of HONNs on the 10-city problem.

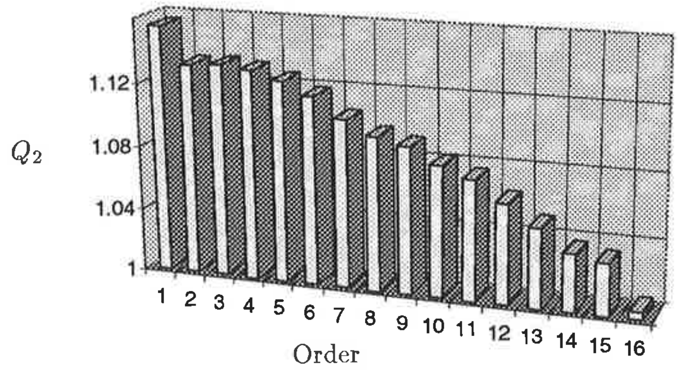
For each network, the annealing parameter γ has been set such that the vertex of the valid subspace corresponding to the optimal tour is only just stable. All possible tours are exhaustively searched to determine if the vertices of the valid subspace which correspond to each tour are stable. Due to the degeneracy inherent in representing a tour by an $N \times N$ array of neurons, the number of vertices of the valid subspace which are stable is given by $2N$ times the number of stable tours. Q_1 gives the normalised mean length of the stable tours and Q_2 gives the normalised weighted mean length of all stable tours.



(a) Number of stable tours.



(b) Measure of quality Q_1.



(c) Measure of quality Q_2.

Figure 7.2: Numerical comparison of HONNs on the bayg29 problem.

For each network the annealing parameter γ has been set such that the vertex of the valid subspace corresponding to the optimal tour is only just stable. The collection of 10^5 near-optimal tours is exhaustively searched to determine if the vertices of the valid subspace which correspond to each tour are stable. The number of distinct near-optimal tours whose corresponding vertices of the valid subspace are stable is shown, along with the measures of quality Q_1, which gives the normalised mean length of the stable tours, and Q_2, which gives the normalised weighted mean length of all stable tours.

7.4 Significance of the Results

7.4.1 The Quality Versus Computational Effort Trade-off

In this chapter we have shown that the number of vertices of the valid subspace which are stable when the optimal tour is only just stable will decrease as the order of the network is increased. Furthermore, the results of our experiments suggest that the quality of the stable solutions improved as the order of the network was increased. Both these facts are evidence of the trade-off that exists between solution quality and computational effort. HONNs are a family of solution techniques for combinatorial optimisation that embody a trade-off between solution quality and computational effort. By increasing the order of HONN we use to solve a problem, we can expect an improvement in the quality of the solutions found. However, we must pay for this improvement by providing more interconnections in the network.

Following the theme of a trade-off between solution quality and computational effort it is interesting to consider a $(N-1)^{th}$ order HONN for the solution of the TSP. By analogy with the HONN objective functions presented in Section 6.3, the objective function for a $(N-1)^{th}$ order network is given by

$$E^{obj} = \sum_{\substack{x_1,x_2,\ldots x_N \\ x_1\neq x_2\neq\ldots\neq x_N}} \sum_i (d_{x_1x_2} + d_{x_2x_3} + \ldots + d_{x_{N-1}x_N}) v_{x_1,i}v_{x_2,i+1}v_{x_3,i+2}\ldots v_{x_N,i+N-1}.$$

In a similar fashion to the analysis presented in Section 7.2, we can determine the conditions under which an arbitrary vertex of the valid subspace is asymptotically stable for the $(N-1)^{th}$ order network. The analysis presented in Section 7.2 showed that for a Hopfield network, the stability of a vertex $\mathbf{v}$ of the valid subspace is determined by a function of the distances between three consecutive cities in the tour described by $\mathbf{v}$. Similarly, for a second-order HONN, the stability of vertex $\mathbf{v}$ is described by a function of the distances between five consecutive cities in the tour described by $\mathbf{v}$. Therefore, we would expect that for a $(N-1)^{th}$ order HONN, the stability of vertex $\mathbf{v}$ will be determined by a function of the distances between $2N-1$ consecutive cities in the tour described by $\mathbf{v}$. By analogy with equations (7.7) and (7.11), we see that the condition under which the output of neuron $v_{x_1,i}$ is asymptotically stable when vertex $\mathbf{v}$ corresponds to the tour $x_1, x_2, x_3 \ldots x_N$ and $v_{x_1,i} = 1$ is given by[2] [3]

$$\begin{aligned}\gamma \;>\; & \frac{N}{N-1}\Big(d_{x_2x_3} + 2d_{x_3x_4} + \ldots + (N-2)d_{x_{N-1}x_N} + (N-1)d_{x_Nx_1} \\ & +(N-1)d_{x_1x_2} + (N-2)d_{x_2x_3} + (N-3)d_{x_3x_4} + \ldots d_{x_{N-1}x_N}\Big)\end{aligned}$$

which simplifies to

$$\gamma > N \times \texttt{tourlength} \tag{7.13}$$

where `tourlength` is the length of the tour described by vertex $\mathbf{v}$ of the valid subspace. Equation (7.13) shows that for a $(N-1)^{th}$ order network the stability of a vertex of the valid subspace is determined by the length of the tour which it describes. Consequently, as the annealing parameter γ is increased during the annealing process the first vertex

[2]Note that equations (7.7) and (7.11) show the conditions under which all neurons, whose output is equal to one, are asymptotically stable. Here we are only concerned with the stability of the output of the single neuron $v_{x_1,i}$, and so there is no need for the max operator.

[3]Note that all indices are evaluated modulo N *e.g.* $x_0 = x_N$, $x_{-1} = x_{N-1}$.

of the valid subspace to become stable must correspond to the optimal solution to the TSP. This is in stark contrast with the result for the Hopfield network, where the first vertex to become stable was by no means guaranteed to be the optimal solution.

A $(N-1)^{th}$ order HONN represents the *ultimate trade-off* of computational effort for solution quality. While we have just shown that the vertex of the valid subspace corresponding to the optimal tour will be the first vertex to become stable as the annealing parameter is increased, we must realise that a $(N-1)^{th}$ order HONN for the TSP requires a number of interconnections which is exponential in the size of the problem. In many ways a $(N-1)^{th}$ order HONN is similar to a direct enumeration approach to solving the TSP, where all tours are listed and the shortest one is chosen. While both methods are guaranteed to arrive at the optimal solution, they must both expend an amount of computational effort which is an exponential function of the problem size and are therefore impractical. For the enumeration approach the computational effort is seen as an exponentially long runtime, while for the HONN it takes the form of an exponentially large number of interconnections in the network.

Further to the theme of a trade-off between computational effort and solution quality we must consider the Hopfield network as the HONN which has obtained the simplest network architecture by trading a reduction in computational effort for a degradation in solution quality. Moreover, the Hopfield network has taken this trade-off to the maximum possible extent, and so we should not be surprised by the relatively poor solution quality achieved by the Hopfield network.

7.4.2 The Effect of Annealing

For the numerical comparisons presented in this chapter we have held the annealing parameter constant at a value γ_{opt} so that the vertex of the valid subspace corresponding to the optimal tour is stable. However, were we to actually simulate the operation of a HONN we would use hysteretic annealing to improve solution quality and so the annealing parameter would not be held constant. When using annealing γ is initially set to a large negative value, which results in the network state $\mathbf{v}$ converging to a point on the valid subspace. Subsequently γ is gradually increased. An important benefit of the annealing process is that it actually helps to guide $\mathbf{v}$ towards good solutions. Consequently, when the annealing parameter has reached a value $\gamma = \gamma_{opt}$, the network state is not at a random position in the valid subspace, rather we would expect $\mathbf{v}$ to be in the vicinity of a near-optimal solution. The overall effect is that the proper use of annealing allows much better solution quality to be achieved than is suggested by the numerical comparisons in this chapter.

When the Hopfield network was used to solve the `bayg29` problem in Section 7.3, we found that setting $\gamma = \gamma_{opt}$ caused there to be in excess of 7000 stable tours out of the 100,000 near-optimal tours that we considered. This would suggest that the chance of finding the optimal solution with a Hopfield network is extremely remote. However, we know from the results presented in Section 4.6 that the Hopfield network can regularly find the optimal solution to a similar 30-city Euclidean TSP. These two observations seem to be contradictory. However, the difference is explained when we realise that the experiments in Section 4.6 utilised annealing and so were able to guide the network state towards good solutions. In contrast the results presented in this section assume that the annealing parameter is fixed and that the network state is placed randomly into the valid subspace.

Furthermore, when in Section 7.3.3 we noted that extremely high-order networks were required so that the optimal solution to the TSP could be obtained, we had not considered the benefits that are gained from annealing. The ability of annealing to guide **v** towards good solutions means that the network can regularly converge to the optimal solution for networks of much lower order than previously suggested. However, the significance of the results in this chapter is not diminished by the effects of annealing. The number of stable vertices of the valid subspace when $\gamma = \gamma_{opt}$ remains as a worthwhile indicator of the trade-offs to be made between solution quality and computational effort.

7.4.3 *Further Work*

Underpinning much of the analysis in this chapter is the assumption that any decrease in the number of vertices of the valid subspace which are stable when the vertex corresponding to the optimal solution is only just stable will result in the valid subspace being partitioned into fewer basins of attraction. Furthermore, if there are fewer basins of attraction in the valid subspace then the probability of placing a random valid starting point into the basin of attraction of the optimal solution will be increased.

What we have so far neglected to consider is the possibility of stable attractors lying not at the vertices of the valid subspace but within this subspace. Should any such attractors exist then the valid subspace may well be partitioned into more basins of attraction than we have accounted for by considering only attractors at the vertices of the valid subspace. Consequently, our interpretation of the number of stable vertices of the valid subspace as a measure of the probability that the optimal solution will be found would no longer be well founded.

To determine if any attractors exist inside the valid subspace we must examine the time derivative of the Lyapunov function for a HONN solving the TSP. In Section 6.1 we showed that

$$\frac{dE^{lyap}}{dt} = -\sum_{xi} g'(u_{xi}) \left(\frac{du_{xi}}{dt}\right)^2 \leq 0.$$

For a point **v** inside the valid subspace to be an attractor we require that

$$\frac{du_{xi}}{dt} = 0 \qquad \forall\ x, i \in \{1, \dots N\}. \tag{7.14}$$

Here the time derivative of u_{xi} is given by the analogy to equation (7.5) which is appropriate for the HONN under examination. In addition, the value of the annealing parameter, which appears in the time derivative of u_{xi}, must be set at the value of γ_{opt} for the HONN under examination. While equation (7.14) establishes the conditions under which some point inside the valid subspace is an attractor for the system, we have been unable to determine if it is or is not possible for any point to satisfy these conditions. Consequently, while we await a method to determine if attractors do or do not exist inside the valid subspace we must content ourselves with the analysis presented in this chapter, despite its possible shortcomings.

7.5 Chapter Summary

In this chapter we have explored the trade-off between solution quality and computational effort that may be made when using a HONN approach to combinatorial optimisation. We began by emphasising that an optimisation network is a dynamic system where the stable attractors can be interpreted as solutions to an optimisation problem. The problem mapping and annealing techniques that we have developed in previous chapters determine the number and location of attractors in the system. The key to our investigation of solution quality is to understand that the more attractors that exist, the less likely it is that we will arrive at the attractor which corresponds to the optimal solution.

Since we know that annealing will eventually force all attractors for the system to the vertices of the valid subspace, our analysis has focused upon the stability of the vertices of the valid subspace. As we have previously noted, for the case of the TSP all vertices of the valid subspace correspond to valid tours. For the Hopfield network we established that an arbitrary vertex $\mathbf{v}$ of the valid subspace would only become stable when the annealing parameter had been increased to a level where

$$\gamma > \frac{N}{N-1} \max_{\overline{yx},\,\overline{xz}\,\in\,\mathrm{tour}(\mathbf{v})} (d_{yx} + d_{xz}).$$

Here the notation $\overline{yx}, \overline{xz} \in \mathrm{tour}(\mathbf{v})$ indicates that y, x and z are consecutive cities in the tour represented by $\mathbf{v}$. We see that the stability of a vertex of the valid subspace is not determined by the length of the tour it describes, rather it is determined by the maximum length of a segment joining three consecutive cities in that tour. Consequently, as the annealing parameter is increased during the annealing process, the first vertex to become stable is by no means guaranteed to be the optimal solution. Moreover, by the time the vertex of the valid subspace which corresponds to the optimal tour is stable, many other vertices may be stable and so the valid subspace may be partitioned into many basins of attraction. With many basins of attraction there is little chance of obtaining the optimal solution, and so the solution quality is expected to be quite poor.

A contrasting situation is presented by a second-order HONN for the solution of the TSP. In that case an arbitrary vertex $\mathbf{v}$ of the valid subspace will only become stable when the annealing parameter satisfies

$$\gamma > \frac{N}{N-1} \max_{\overline{vw},\,\overline{wx},\,\overline{xy},\,\overline{yz}\,\in\,\mathrm{tour}(\mathbf{v})} (d_{vw} + 2d_{wx} + 2d_{xy} + d_{yz}).$$

Here the notation $\overline{vw}, \overline{wx}, \overline{xy}, \overline{yz} \in \mathrm{tour}(\mathbf{v})$ indicates that v, w, x, y and z are consecutive cities in the tour represented by $\mathbf{v}$. Once again the stability of a vertex of the valid subspace is not determined by the length of the tour which it describes, but instead is determined by a function of the intercity distances on a segment of the tour which passes through five consecutive cities on the tour. The stability requirement for a vertex $\mathbf{v}$ in the second-order HONN provides a greater level of discrimination between optimal and sub-optimal tours than is the case with the Hopfield network. We suggested that as the order of the network is increased further, the number of vertices of the valid subspace which are stable when the optimal tour was only just stable should decrease.

This assertion was validated by a numerical comparison of the number of stable points in HONNs when operating on two example TSPs. The first problem that we investigated was a 10-city Euclidean TSP, where it was possible to exhaustively search all vertices of the valid subspace. When the annealing parameter was set so that the optimal tour was stable, we saw that for the Hopfield network there were over 230 other stable tours. In contrast, for a HONN of order 5 the optimal tour was the first tour to become stable. The second problem was `bayg29`, a 29 city Euclidean TSP where the size of the problem made it necessary to investigate the stability of a collection of near-optimal tours. Once again our results showed that the number of stable tours decreased rapidly as the order of the network was increased. Furthermore, as the order of the network increased, the mean length of the stable tours decreased, indicating that the solution quality was improving.

The results of our experiments motivated the observation that HONNs are a family of solution techniques for combinatorial optimisation that embody a trade-off between solution quality and computational effort. By increasing the order of the network used to solve a problem, we can expect an improvement in the quality of the solutions produced. However, we must pay for this improvement by providing more interconnections in the network. Further insight into the trade-off between solution quality and computational effort was gained by considering a $(N-1)^{th}$ order network for the solution of the TSP. It was shown that such a network can guarantee that the first vertex of the valid subspace to become stable will correspond to the optimal solution, but to do so the network requires a number of interconnections which is exponential in the size of the problem. In many ways a $(N-1)^{th}$ order HONN is similar to an enumeration technique for the solution of the TSP. Both methods are guaranteed to find the optimal solution to the problem, but to do so they require an impractical amount of computational effort.

Our investigations of the trade-off between solution quality and computational effort provide a clear explanation of the shortcomings of the Hopfield network approach to combinatorial optimisation. The Hopfield network must be considered as the simplest possible HONN where we have traded a reduction in network complexity for a corresponding decrease in the solution quality. When compared to the $(N-1)^{th}$ order HONN the Hopfield network lies at the opposite extreme of the solution quality versus computational effort trade-off, and so we should not be surprised by the relatively poor solution quality achieved by the Hopfield network.

Chapter VIII

Conclusions

In this thesis we have presented an examination of the performance of optimisation networks. Our main objective was to determine if there exist any factors which limit the solution quality that may be achieved with optimisation networks. Furthermore, our aim was to determine the reasons for any such limitations, and if possible to suggest remedies for them.

It has long been recognised that optimisation networks, in their original form, tend to produce poor, high cost solutions. Various annealing algorithms have been developed to improve the solution quality obtained from optimisation networks, and so we began our examination of the performance of optimisation networks by investigating the effects of these annealing algorithms. Our investigations confirmed that annealing has the ability to improve solution quality, but also uncovered the tendency of annealing to force the network toward invalid states. Consequently, we developed a new, principled approach to annealing that retained the ability to improve solution quality, while also ensuring that the network state remained valid.

Even when a correctly formulated approach to annealing is used, experimental evidence suggests that as the problem size increases, the solution quality obtained from an optimisation network will decrease. While the inability of optimisation networks to scale to large problem sizes has gone largely unnoticed in the literature, it represents a significant erosion of the niche market for optimisation networks. In order to discover the causes of such poor scaling to large problems, we showed that optimisation networks solve combinatorial problems by using simple heuristics which are embedded into the network dynamics. These heuristics encourage the formation of small, locally optimal segments in the solution. Moreover, as the size of the problem increases there is no corresponding increase in the size of the locally optimal segments which are formed by these heuristics. The simple heuristics are ultimately responsible for the poor scaling of optimisation networks. To improve the performance of optimisation networks it is necessary to replace or modify the heuristics that they use.

To improve upon the poor scaling of standard optimisation networks, HONNs were proposed as a means to extend the neighbourhood in which locally optimal segments are encouraged. By increasing the order of the network, a stronger heuristic may be embedded into the network dynamics and consequently the solution quality should be improved. However, as the order of the network is increased, the number of interconnections necessary to implement the network also increases. HONNs are a family of solution techniques for combinatorial optimisation that embody a trade-off between solution quality and computational effort. Further insight into the trade-off between solution quality and computational effort is gained by contrasting the performance of the Hopfield network with a $(N-1)^{th}$ order HONN when solving a N city TSP. While a $(N-1)^{th}$ order network can almost guarantee that the optimal solution will be found, it requires a number of interconnections which is exponential in the size of the problem. In contrast, the Hopfield network is a relatively simple network with $O(N^3)$ connections, but it cannot guarantee that the optimal solution will be found. The Hopfield network must be considered as the simplest possible HONN, where we have traded a reduction in network complexity for a corresponding decrease in solution quality. When compared to the $(N-1)^{th}$ order HONN, the Hopfield network lies at the opposite extreme of the trade-off between solution quality and computational effort, and so we should not be surprised by the relatively poor performance of the Hopfield network.

We conclude that optimisation networks embody a simple heuristic approach to the solution of combinatorial optimisation problems. As with any heuristic approach to optimisation, there are trade-offs to be made between solution quality and computational effort. HONNs embody that trade-off by allowing increased computational effort to be traded for improved solution quality by simply increasing the order of the network. If when applying optimisation networks to a particular problem it becomes necessary to trade increased complexity for improved solution quality, we must be aware that this can only be done up to a certain level, beyond which the complexity of the network prohibits it from being implemented.

The major contributions made by this thesis may be summarised as follows:

- We have performed a rigorous analysis of the effects of annealing techniques on the validity and quality of solutions obtained from an optimisation network (Section 4.3). Following this analysis, we proposed a principled approach to annealing which allows solution quality to be improved while maintaining a valid network state (Sections 4.4 and 4.5).

- We have explained the inability of standard optimisation networks to scale to large problem sizes (Sections 5.2 and 5.3). Furthermore, we have investigated two alternative approaches that attempt to improve the performance of optimisation networks by modifying the heuristics that they use (Section 5.4).

- Finally, we have introduced a new family of optimisation networks that embody a trade-off between solution quality and computational effort (Section 6.3). Our investigation of that trade-off provided a clear explanation of the shortcomings of the Hopfield network approach to combinatorial optimisation (Sections 7.3 and 7.4).

Appendix A

Derivation of Mean Field Annealing

Simulated Annealing (Kirkpatrick et al., 1983) is a powerful, general purpose optimisation technique, that has been widely applied since its introduction. The technique employs a controlled, stochastic search of the state space, but unfortunately it is sometimes unacceptably slow. Mean field annealing (MFA) is a deterministic approximation to simulated annealing, which sacrifices solution quality for execution speed. Essentially, the MFA algorithm computes a solution to a pair of temperature dependent, coupled, nonlinear equations which are termed the *saddle point equations.* This section presents a rigorous derivation of the saddle point equations and is based on the works of Aiyer (Aiyer, 1991) and Peterson (Peterson and Anderson, 1988). More practical implementation details of the MFA algorithm are presented in Section 2.4.

A.1 Mean Field Theory

When a stochastic Hopfield network operates at a constant non-zero temperature, its state will vary with time. When the network has reached *thermal equilibrium*, the state of the network will fluctuate about a constant average value. Mean field theory allows such *average* statistics of the network to be determined.

A fundamental result from physics is the Boltzmann-Gibbs distribution. It states that a system, such as the stochastic Hopfield network, when in thermal equilibrium will be found in state $\mathbf{s}$ with probability,

$$\Pr(\mathbf{s}) = \frac{1}{Z} \exp\left(\frac{-E(\mathbf{s})}{T^p}\right) \tag{A.1}$$

where the normalising factor

$$Z = \sum_{\{\mathbf{s}\}} \exp\left(\frac{-E(\mathbf{s})}{T^p}\right)$$

is called the partition function and the summation is over all possible states *i.e.* all 2^N combinations of $s_i = \{0, 1\}$ for $i \in \{1, \ldots N\}$. The energy function $E(\mathbf{s})$ is exactly the Lyapunov function for the stochastic Hopfield network, and is reproduced here for completeness

$$E(\mathbf{s}) = -\tfrac{1}{2}\mathbf{s}^T\mathbf{T}\mathbf{s} - \mathbf{b}^T\mathbf{s}. \tag{A.2}$$

In order to facilitate comparisons between MFA networks and the continuous Hopfield network, the activation levels of the units in the stochastic Hopfield network have been changed from $s_i = \pm 1$ to $s_i \in \{0, 1\}$.

In principle, the expected value of the energy may now be calculated by utilising equation (A.1),

$$\langle E(\mathbf{s})\rangle = \frac{1}{Z}\sum_{\{\mathbf{s}\}} E(\mathbf{s}) \exp\left(\frac{-E(\mathbf{s})}{T^p}\right). \tag{A.3}$$

In practice, the difficulty lies in calculating the partition function Z. However, once we have done so, most useful quantities may be derived from Z itself. For example, the expected value of the state vector can be determined as,

$$\begin{aligned} T^p\nabla_{\mathbf{b}} \ln(Z) &= T^p \frac{\nabla_{\mathbf{b}} Z}{Z} \\ &= T^p \frac{\sum_{\{\mathbf{s}\}} \frac{\mathbf{s}}{T^p} \exp\left(-\frac{E(\mathbf{s})}{T^p}\right)}{Z} \\ &= \sum_{\{\mathbf{s}\}} \mathbf{s}\,\Pr(\mathbf{s}) \\ \Rightarrow T^p\nabla_{\mathbf{b}} \ln(Z) &= \langle \mathbf{s}\rangle \end{aligned} \tag{A.4}$$

A.1.1 Simplification of the partition function Z

To obtain a more tractable expression for the partition function, we utilise the multi-dimensional delta function,

$$\int_{\mathbb{R}^N} \delta(\mathbf{s}-\mathbf{v}) f(\mathbf{v})\, d\mathbf{v} = f(\mathbf{s}).$$

Noting that the multi-dimensional delta function can be expressed as a complex exponential,

$$\delta(\mathbf{s}-\mathbf{v}) = C \int_{\mathbb{I}^N} \exp\left(\mathbf{u}^T(\mathbf{s}-\mathbf{v})\right)\, d\mathbf{u}$$

where C is a constant; the partition function Z may be written as the summation of a double integral, at the expense of introducing $2N$ new variables, $\mathbf{u}$ and $\mathbf{v}$:

$$\begin{aligned} Z &= C\sum_{\{\mathbf{s}\}} \int_{\mathbb{R}^N}\int_{\mathbb{I}^N} \exp\left(\mathbf{u}^T(\mathbf{s}-\mathbf{v})\right) \exp\left(-\frac{E(\mathbf{v})}{T^p}\right) d\mathbf{u}\, d\mathbf{v} \\ &= C\int_{\mathbb{R}^N}\int_{\mathbb{I}^N} \exp\left(-\frac{E(\mathbf{v})}{T^p} - \mathbf{u}^T\mathbf{v}\right) \sum_{\{\mathbf{s}\}} \exp\left(\sum_i u_i s_i\right) d\mathbf{u}\, d\mathbf{v}. \end{aligned}$$

The summation over all 2^N possible states may be simplified by noting that,

$$\sum_{\{\mathbf{s}\}} \exp\left(\sum_i u_i s_i\right) = \prod_i \left(\exp(u_i)+1\right) = \exp\left(\sum_i \ln\left(\exp(u_i)+1\right)\right).$$

With the *effective energy* $E'(\mathbf{u},\mathbf{v},T^P)$ defined as,

$$E'(\mathbf{u},\mathbf{v},T^p) = \frac{E(\mathbf{v})}{T^p} + \mathbf{u}^T\mathbf{v} - \sum_i \ln\left(\exp(u_i)+1\right) \tag{A.5}$$

the partition function may now be expressed as,

$$Z = C\int_{\mathbb{R}^N}\int_{\mathbb{I}^N} \exp\left(-E'(\mathbf{u},\mathbf{v},T^p)\right) d\mathbf{u}\, d\mathbf{v}\,.$$

A.1.2 Saddle Point Expansion

The partition function may be further analysed by performing a saddle point expansion of $E'(\mathbf{u}, \mathbf{v}, T^P)$. Firstly, we must make use of the *mean field approximation* by defining the average effective energy per neuron,

$$f(\mathbf{u}, \mathbf{v}, T^P) = \frac{E'(\mathbf{u}, \mathbf{v}, T^p)}{N}$$

thereby allowing the partition function to be written with the exponent proportional to N,

$$Z = C \int_{\mathbb{R}^N} \int_{\mathbb{I}^N} \exp\left(-Nf(\mathbf{u}, \mathbf{v}, T^p)\right) \, d\mathbf{u} \, d\mathbf{v} \, . \tag{A.6}$$

This expression still requires the evaluation of a double integral, but as the exponent in equation (A.6) is proportional to N, it is easily evaluated in the limit of large N. The bigger N is, the more the integral is dominated by contributions from where $f(\mathbf{u}, \mathbf{v}, T^p)$ is smallest. So we can approximate the integral by finding the value $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$ which minimises $f(\mathbf{u}, \mathbf{v}, T^p)$, and expanding the integrand around there. This is known as the *saddle point method* and in the limit of large N, yields (Hertz et al., 1991),

$$Z \approx C \exp(-Nf(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, T^p)) = C \exp(-E'(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, T^p)) \tag{A.7}$$

i.e. the partition function may be approximated by the value of the effective energy at the saddle point $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$.

At a saddle point, the following equations will be satisfied,

$$\frac{\partial E'}{\partial u_i} = 0 \tag{A.8}$$

$$\frac{\partial E'}{\partial v_i} = 0 \, . \tag{A.9}$$

Evaluating the partial derivatives at $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$ gives the *saddle point equations*,

$$\tilde{u}_i = \frac{1}{T^p}\left(\sum_j T_{ij}\tilde{v}_j + b_i\right) \tag{A.10}$$

$$\tilde{v}_i = \frac{1}{1+\exp(-\tilde{u}_i)} \, . \tag{A.11}$$

Now that the saddle point equations have been successfully derived, and if it is realised that the MFA algorithm presented in Section 2.4 merely computes a solution to the saddle point equations, then the question arises: why is the solution of the saddle point equations important?

The answer is that the solution to the saddle point equations gives a great deal of insight into the underlying stochastic Hopfield network. Utilising the MFA algorithm, a solution $(\tilde{\mathbf{u}}, \tilde{\mathbf{v}})$ to the saddle point equations may be obtained. Then, by making the appropriate substitutions from equations (A.7), (A.5) and (A.2), the expected value of the state vector at temperature T^p can be deduced from equation (A.4) as,

$$\begin{aligned} \langle \mathbf{s} \rangle &= T^p \nabla_{\mathbf{b}} \ln(Z) \\ &= T^p \frac{\nabla_{\mathbf{b}} Z}{Z} \\ &\approx -T^p \frac{C \exp\left(-E'(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, T^p)\right)\left(\nabla_{\mathbf{b}} E(\tilde{\mathbf{v}})/T^p\right)}{C \exp\left(-E'(\tilde{\mathbf{u}}, \tilde{\mathbf{v}}, T^p)\right)} \\ &\approx \tilde{\mathbf{v}} \end{aligned} \tag{A.12}$$

i.e. the expected value of the state vector of the stochastic Hopfield network operating at a temperature T^p is given by the solution $\tilde{\mathbf{v}}$ to the saddle point equations (A.10) and (A.11) at that temperature. Moreover, it is apparent from the Boltzmann-Gibbs distribution of equation (A.1) that as the temperature decreases, those states with lower energy $E(\mathbf{s})$ will predominate, and so at low temperatures it may be expected that the solution $\tilde{\mathbf{v}}$ to the saddle point equations will be a good approximation to $\mathbf{s}^{min}$, the global minimum of $E(\mathbf{s})$. Conversely, at high temperatures $\tilde{\mathbf{v}} = \langle \mathbf{s} \rangle$ does not correspond to a single configuration but is an average over many states.

In addition, the expected value of the energy (Lyapunov) function for the Hopfield network (equation (A.2)), may be determined as,

$$\begin{aligned}
\langle E(\mathbf{s}) \rangle &= -\frac{1}{2}\sum_i \sum_j T_{ij}\langle s_i s_j \rangle - \sum_i b_i \langle s_i \rangle \\
&\approx -\frac{1}{2}\sum_i \sum_j T_{ij}\langle s_i \rangle \langle s_j \rangle - \sum_i b_i \langle s_i \rangle \\
&\approx -\frac{1}{2}\sum_i \sum_j T_{ij}\tilde{v}_i \tilde{v}_j - \sum_i b_i \tilde{v}_i \\
&\approx E(\tilde{\mathbf{v}}) \qquad \text{(A.13)}
\end{aligned}$$

where the mean field approximation $\langle s_i s_j \rangle \approx \langle s_i \rangle \langle s_j \rangle$ has been used (Aiyer, 1991).

APPENDIX B

Simulation details for TSP experiments

When simulating an optimisation network for solving the TSP, we have utilised the optimised step-size technique (Abe, 1996). Such a technique, integrates the dynamic equation

$$\frac{d\mathbf{v}}{dt} = \mathbf{T}\mathbf{v} + \mathbf{b}, \quad v_i \in [0,1]$$

by setting the integration step-size at time t to be the minimum step-size which will make some component of $\mathbf{v}$ reach the surface of the unit hypercube. Obviously if $v_i = 0$ and $[\mathbf{T}\mathbf{v} + \mathbf{b}]_i \leq 0$, or if $v_i = 1$ and $[\mathbf{T}\mathbf{v} + \mathbf{b}]_i \geq 0$ at time t, then v_i is moving away from the unit hypercube and component i must be excluded from the calculation of the step-size. Therefore, let $\mathcal{I}$ be the set of integers where $v_i = 0$ and $[\mathbf{T}\mathbf{v} + \mathbf{b}]_i \leq 0$, or $v_i = 1$ and $[\mathbf{T}\mathbf{v} + \mathbf{b}]_i \geq 0$. Additionally let $\mathcal{N} = \{1, \ldots N\}$. Then for $i \in \mathcal{N} - \mathcal{I}$ calculate

$$t_i = \begin{cases} \dfrac{v_i}{[\mathbf{T}\mathbf{v} + \mathbf{b}]_i} & \text{for } [\mathbf{T}\mathbf{v} + \mathbf{b}]_i < 0 \\ \dfrac{v_i - 1}{[\mathbf{T}\mathbf{v} + \mathbf{b}]_i} & \text{for } [\mathbf{T}\mathbf{v} + \mathbf{b}]_i > 0 \end{cases}$$

where t_i is the step-size required for v_i to reach the surface of the unit hypercube. The integration step-size Δt at time t is then chosen as,

$$\Delta t = \min_{i \in \mathcal{N} - \mathcal{I}} t_i. \tag{B.1}$$

Since the determination of the step size is a deterministic process, it is possible that the algorithm will become stuck in a infinite loop during the integration. In such a situation the same components of $\mathbf{v}$ will continually be chosen to determine the step-size, and $\mathbf{v}$ will move between several constant states. Obviously this is not desirable and must be prevented. Consequently, several mechanisms have been developed to introduce randomness into the step-size determination process, thereby helping to prevent an infinite loop from occuring. While the precise details can be obtained from the original exposition (Abe, 1996), we will present a summary of the proposed methods.

1. Firstly, if the same component i is selected by equation (B.1) to determine the step-size in consecutive time steps, then the calculated step-size is modified by setting

$$\Delta t \leftarrow \Delta t \times \texttt{rand} \tag{B.2}$$

where `rand` is a random value uniformly distributed in the range $[0, 1]$.

2. If equation (B.2) does not succeed in preventing an infinite loop and the same component is selected by equation (B.1) to determine the step-size at three consecutive time steps, or if the same pairs of indices are selected by equation (B.1) three times in succession *e.g.* the sequence ...256256256..., then the calculated step-size is modified by

$$\Delta t \leftarrow \Delta t(1 + \beta_1 \times \texttt{rand})$$

   where β_1 is a small positive constant; typically $\beta_1 = 0.01$.

3. For all other time steps, the step-size is modified by setting

$$\Delta t \leftarrow \Delta t(1 + \beta_2 \times \texttt{rand})$$

   where β_2 is a small positive value; typically $\beta_2 = 0.0001$.

Unlike most alternative algorithms for simulating the operation of an optimisation network *e.g.* (Abe, 1993), the optimised step-size algorithm does not examine the rate of convergence of $\mathbf{v}$ to control the annealing schedule. Instead, the optimised step-size algorithm holds the annealing parameter constant at γ_0 for the first t_d time steps and then increments the annealing parameter by an amount $\Delta\gamma$ at successive time steps until an upper limit γ_{max} is reached. The simulation is stopped when a valid solution is reached or the number of iterations exceeds a user defined value (typically of the order of ten thousand iterations). While the precise value of all parameters will depend on the problem to be solved, representative values for the problems solved in this thesis are: $\gamma_0 = -1$ to -6, $t_d = 2000$ and $\Delta\gamma = 0.005$.

Appendix C

Eigenvalues of Interconnection Matrices

C.1 Eigenvalues of $\mathbf{T}^{cns}$

When the Lyapunov function $E^{lyap} = E^{cns}$, the connection matrix for the optimisation network is given by equation (3.15), *i.e.*

$$[\mathbf{T}^{cns}]_{xi,yj} = -\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2}. \tag{C.1}$$

The eigenvalues and eigenvectors of the connection matrix $\mathbf{T}^{cns}$ are determined, using the method employed in (Aiyer et al., 1990), as follows.

C.1.1 Determination of λ_1

We determine λ_1 by showing that $\hat{\mathbf{e}} = \frac{1}{N}[1, 1, \ldots, 1]^T$ is a eigenvector of $\mathbf{T}^{cns}$ with an eigenvalue $\lambda_1 = -c$. Now

$$\begin{aligned} [\mathbf{T}^{cns}\hat{\mathbf{e}}]_{xi} &= \sum_y \sum_j \left(-\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2}\right)\frac{1}{N} \\ &= -\sum_y \frac{c}{N^2} - \sum_j \frac{c}{N^2} + \sum_y \sum_j \frac{c}{N^3} \quad = \quad -\frac{c}{N} \end{aligned} \tag{C.2}$$

i.e. $\mathbf{T}^{cns}\hat{\mathbf{e}} = -c\hat{\mathbf{e}}$. Therefore $\hat{\mathbf{e}}$ is an eigenvector of $\mathbf{T}^{cns}$ with a corresponding eigenvalue $\lambda_1 = -c$.

C.1.2 Determination of λ_2

We calculate λ_2 by showing that the zerosum subspace is an eigenspace of $\mathbf{T}^{cns}$ with a corresponding eigenvalue $\lambda_2 = 0$. As described in Section 3.3, every valid solution $\mathbf{v}$ to the TSP can be decomposed into a component $\mathbf{v}^{zs}$ in the zerosum subspace and a component $\hat{\mathbf{e}}$,

$$\mathbf{v} = \mathbf{v}^{zs} + \hat{\mathbf{e}}.$$

Therefore

$$\begin{aligned} \mathbf{T}^{cns}\mathbf{v}^{zs} &= \mathbf{T}^{cns}\mathbf{v} - \mathbf{T}^{cns}\hat{\mathbf{e}} \\ &= \mathbf{T}^{cns}\mathbf{v} - \lambda_1\hat{\mathbf{e}}. \end{aligned}$$

Now since $\mathbf{v}$ is valid, $\sum_y v_{yi} = \sum_j v_{xj} = 1$ and $\sum_y \sum_j v_{yj} = N$, allowing the following simplification,

$$[\mathbf{T}^{cns}\mathbf{v}]_{xi} = \sum_y \sum_j \left(-\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2}\right) v_{yj} = -\frac{c}{N}.$$

Upon substitution we obtain

$$[\mathbf{T}^{cns}\mathbf{v}^{zs}]_{xi} = -\frac{c}{N} - \left(\frac{-c}{N}\right) = 0.$$

Therefore there is a degenerate eigenvalue of $\lambda_2 = 0$ that corresponds to the zerosum subspace.

C.1.3 Determination of λ_3

To determine λ_3 it is shown that the invalid subspace is an eigenspace of $\mathbf{T}^{cns}$ with a corresponding eigenvalue $\lambda_3 = -c$.

Firstly, consider the matrix $\mathbf{S}$ which is the sum of outer products of all valid solutions $\mathbf{v}$ and is derived in Appendix C.4. Since the invalid subspace is mutually orthogonal to both the valid subspace and $\hat{\mathbf{e}}$, $\mathbf{S}\mathbf{v}^{inv} = \mathbf{0}$, where $\mathbf{0}$ is the vector of all zeros. Therefore the eigenvalue of $\mathbf{S}$ in the invalid subspace is zero.

Now by establishing an expression for $\mathbf{T}^{cns}$ in terms of $\mathbf{S}$, we shall determine the eigenvalues of $\mathbf{T}^{cns}$ in the invalid subspace. By considering the equations (C.1) and (C.9) the connection matrix $\mathbf{T}^{cns}$ may be written as

$$\mathbf{T}^{cns} = \frac{c}{N(N-2)!}\mathbf{S} + c(1-N)\hat{\mathbf{e}}\hat{\mathbf{e}}^T - c\mathbf{I}. \tag{C.3}$$

To verify equation (C.3) we shall evaluate the right hand side. Noting that $[\hat{\mathbf{e}}\hat{\mathbf{e}}^T]_{xi,yj} = 1/N^2$, $[\mathbf{I}]_{xi,yj} = \delta_{xy}\delta_{ij}$ and substituting equation (C.9) gives

$$\begin{aligned}
[\mathbf{T}^{cns}]_{xi,yj} &= \frac{c(N-1)}{N}\delta_{xy}\delta_{ij} + \frac{c}{N}(1-\delta_{xy})(1-\delta_{ij}) + \frac{c(1-N)}{N^2} - c\,\delta_{xy}\delta_{ij} \\
&= -\frac{c}{N}\delta_{xy} - \frac{c}{N}\delta_{ij} + \frac{c}{N^2}
\end{aligned}$$

which coincides with the expression for $\mathbf{T}^{cns}$ given by equation (C.1).

The eigenvalue of $\mathbf{T}^{cns}$ in the invalid subspace may now be determined by considering the individual terms in equation (C.3). Multiplication of $\mathbf{S}$ by $\frac{c}{N(N-2)!}$ scales the eigenvalues of $\mathbf{S}$, but the eigenvalue of $\mathbf{S}$ in the invalid subspace remains zero. Similarly, multiplication of $\hat{\mathbf{e}}\hat{\mathbf{e}}^T$ by a constant affects only the eigenvalue of the eigenvector in the direction of $\hat{\mathbf{e}}$. The remaining term, $-c\mathbf{I}$ introduces an eigenvalue of $-c$ in the invalid subspace. Therefore, for a vector $\mathbf{v}^{inv}$ which lies in the invalid subspace

$$\mathbf{T}^{cns}\mathbf{v}^{inv} = -c\mathbf{v}^{inv}$$

i.e. there is a degenerate eigenvalue $\lambda_3 = -c$ that corresponds to the invalid subspace.

C.2 Eigenvalues of $\mathbf{T}^{ann}$

For an optimisation network where the Lyapunov function $E^{lyap} = E^{cns} + E^{ann}$, the connection matrix is given by equation (4.4), *i.e.*

$$[\mathbf{T}^{ann}]_{xi,yj} = -\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2} + \gamma\delta_{xy}\delta_{ij}. \tag{C.4}$$

The eigenvalues and eigenvectors of the connection matrix $\mathbf{T}^{ann}$ are determined, using a method similar to that employed in (Aiyer et al., 1990), as follows.

C.2.1 Determination of λ_1

The eigenvalue λ_1 is calculated by showing that $\hat{\mathbf{e}} = \frac{1}{N}[1, 1, \ldots, 1]^T$ is an eigenvector of $\mathbf{T}^{ann}$ with a corresponding eigenvalue $\lambda_1 = -c + \gamma$. Now

$$\begin{aligned}
[\mathbf{T}^{ann}\hat{\mathbf{e}}]_{xi} &= \sum_y \sum_j \left(-\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2} + \gamma\delta_{xy}\delta_{ij} \right) \frac{1}{N} \\
&= -\sum_y \frac{c}{N^2} - \sum_j \frac{c}{N^2} + \sum_y \sum_j \frac{c}{N^3} + \frac{\gamma}{N} \\
&= (-c+\gamma)\frac{1}{N} \qquad \text{(C.5)}
\end{aligned}$$

i.e. $\mathbf{T}^{ann}\hat{\mathbf{e}} = (-c+\gamma)\hat{\mathbf{e}}$. Therefore $\hat{\mathbf{e}}$ is an eigenvector of $\mathbf{T}^{ann}$ with a corresponding eigenvalue $\lambda_1 = -c + \gamma$.

C.2.2 Determination of λ_2

The eigenvalue λ_2 is determined by showing that the zerosum subspace is an eigenspace of $\mathbf{T}^{ann}$ with a corresponding eigenvalue $\lambda_2 = \gamma$. As described in Section 3.3, every valid solution $\mathbf{v}$ to the TSP can be decomposed into a component $\mathbf{v}^{zs}$ in the zerosum subspace and a component $\hat{\mathbf{e}}$,

$$\mathbf{v} = \mathbf{v}^{zs} + \hat{\mathbf{e}}.$$

Therefore

$$\begin{aligned}
\mathbf{T}^{ann}\mathbf{v}^{zs} &= \mathbf{T}^{ann}\mathbf{v} - \mathbf{T}^{ann}\hat{\mathbf{e}} \\
&= \mathbf{T}^{ann}\mathbf{v} - \lambda_1\hat{\mathbf{e}}.
\end{aligned}$$

Now since $\mathbf{v}$ is valid, $\sum_y v_{yi} = \sum_j v_{xj} = 1$ and $\sum_y \sum_j v_{yj} = N$, allowing the following simplification,

$$\begin{aligned}
[\mathbf{T}^{ann}\mathbf{v}]_{xi} &= \sum_y \sum_j \left(-\frac{c}{N}\delta_{ij} - \frac{c}{N}\delta_{xy} + \frac{c}{N^2} + \gamma\delta_{xy}\delta_{ij} \right) v_{yj} \\
&= -\frac{c}{N}\sum_y v_{yi} - \frac{c}{N}\sum_j v_{xj} + \frac{c}{N^2}\sum_y \sum_j v_{yj} + \gamma v_{xi} \\
&= -\frac{c}{N} + \gamma v_{xi}.
\end{aligned}$$

Upon substitution we obtain

$$\begin{aligned}
[\mathbf{T}^{ann}\mathbf{v}^{zs}]_{xi} &= -\frac{c}{N} + \gamma v_{xi} - [\lambda_1\hat{\mathbf{e}}]_{xi} \\
&= \gamma\left(v_{xi} - \frac{1}{N}\right) \quad = \quad \gamma v^{zs}_{xi}.
\end{aligned}$$

Therefore there is a degenerate eigenvalue of $\lambda_2 = \gamma$ that corresponds to the zerosum subspace.

C.2.3 Determination of λ_3

To determine λ_3 we show that the invalid subspace is an eigenspace of $\mathbf{T}^{ann}$ with a corresponding eigenvalue $\lambda_3 = -c + \gamma$.

Firstly, consider the matrix $\mathbf{S}$ which is the sum of outer products of all valid solutions $\mathbf{v}$ and is derived in Appendix C.4. Since the invalid subspace is mutually orthogonal to both the valid subspace and $\hat{\mathbf{e}}$, $\mathbf{S}\mathbf{v}^{inv} = \mathbf{0}$, where $\mathbf{0}$ is the vector of all zeros. Therefore the eigenvalue of $\mathbf{S}$ in the invalid subspace is zero.

Now by establishing an expression for $\mathbf{T}^{ann}$ in terms of $\mathbf{S}$, we shall determine the eigenvalues of $\mathbf{T}^{ann}$ in the invalid subspace. By considering the equations (C.4) and (C.9) the connection matrix $\mathbf{T}^{ann}$ may be written as,

$$\mathbf{T}^{ann} = \frac{c}{N(N-2)!}\mathbf{S} + c(1-N)\hat{\mathbf{e}}\hat{\mathbf{e}}^T - c\mathbf{I} + \gamma\mathbf{I}. \qquad \text{(C.6)}$$

Equation (C.6) may be verified by following the same procedure as used in Appendix C.1.3. The details of this procedure are left to the reader. To determine the eigenvalue of $\mathbf{T}^{ann}$ in the invalid subspace, we must consider each term in equation (C.6). Multiplication of $\mathbf{S}$ by $\frac{c}{N(N-2)!}$ scales the eigenvalues of $\mathbf{S}$, but the eigenvalue of $\mathbf{S}$ in the invalid subspace remains zero. Similarly, multiplication of $\hat{\mathbf{e}}\hat{\mathbf{e}}^T$ by a constant affects only the eigenvalue of the eigenvector in the direction of $\hat{\mathbf{e}}$. The remaining term, $-c\mathbf{I} + \gamma\mathbf{I}$ introduces an eigenvalue of $-c + \gamma$ in the invalid subspace. Therefore, for a vector $\mathbf{v}^{inv}$ which lies in the invalid subspace

$$\mathbf{T}^{ann}\mathbf{v}^{inv} = (-c + \gamma)\mathbf{v}^{inv}$$

i.e. there is a degenerate eigenvalue $\lambda_3 = -c+\gamma$ that corresponds to the invalid subspace.

C.3 Eigenvalues of $\mathbf{T}^{mod}$

For an optimisation network where the Lyapunov function $E^{lyap} = E^{cns} + E^{mod}$, the connection matrix is given by equation (4.12), *i.e.*

$$[\mathbf{T}^{mod}]_{xi,yj} = -\frac{(c+\gamma)}{N}\delta_{ij} - \frac{(c+\gamma)}{N}\delta_{xy} + \frac{(c+\gamma)}{N^2} + \gamma\delta_{xy}\delta_{ij}. \qquad \text{(C.7)}$$

The eigenvalues and eigenvectors of the connection matrix $\mathbf{T}^{mod}$ are determined, using a method similar to that employed in (Aiyer et al., 1990), as follows.

C.3.1 Determination of λ_1

To determine λ_1 we show that $\hat{\mathbf{e}} = \frac{1}{N}[1, 1, \ldots, 1]^T$ is an eigenvector of $\mathbf{T}^{mod}$ with an eigenvalue $\lambda_1 = -c$. Now

$$\begin{aligned}\left[\mathbf{T}^{mod}\hat{\mathbf{e}}\right]_{xi} &= \sum_y\sum_j\left(-\frac{c+\gamma}{N}\delta_{ij} - \frac{c+\gamma}{N}\delta_{xy} + \frac{c+\gamma}{N^2} + \gamma\delta_{xy}\delta_{ij}\right)\frac{1}{N} \\ &= -\sum_y\frac{c+\gamma}{N} - \sum_j\frac{c+\gamma}{N} + \sum_y\sum_j\frac{c+\gamma}{N^2} + \gamma \quad = \quad -\frac{c}{N}\end{aligned}$$

i.e. $\mathbf{T}^{mod}\hat{\mathbf{e}} = -c\hat{\mathbf{e}}$. Therefore $\hat{\mathbf{e}}$ is an eigenvector of $\mathbf{T}^{mod}$ with a corresponding eigenvalue $\lambda_1 = -c$.

C.3.2 Determination of λ_2

The eigenvalue λ_2 is determined by showing that the zerosum subspace is an eigenspace of $\mathbf{T}^{mod}$ with a corresponding degenerate eigenvalue $\lambda_2 = \gamma$. As described in Section 3.3, every valid solution to the TSP can be decomposed into a component $\mathbf{v}^{zs}$ in the zerosum subspace and a component $\hat{\mathbf{e}}$,

$$\mathbf{v} = \mathbf{v}^{zs} + \hat{\mathbf{e}}.$$

Therefore

$$\begin{aligned}\mathbf{T}^{mod}\mathbf{v}^{zs} &= \mathbf{T}^{mod}\mathbf{v} - \mathbf{T}^{mod}\hat{\mathbf{e}} \\ &= \mathbf{T}^{mod}\mathbf{v} - \lambda_1\hat{\mathbf{e}}.\end{aligned}$$

Now since $\mathbf{v}$ is valid, $\sum_y v_{yi} = \sum_j v_{xj} = 1$ and $\sum_y \sum_j v_{yj} = N$, allowing the following simplification,

$$\begin{aligned}\left[\mathbf{T}^{mod}\mathbf{v}\right]_{xi} &= \sum_y \sum_j \left(-\frac{c+\gamma}{N}\delta_{ij} - \frac{c+\gamma}{N}\delta_{xy} + \frac{c+\gamma}{N^2} + \gamma\delta_{xy}\delta_{ij}\right) v_{yj} \\ &= -\frac{c+\gamma}{N}\sum_y v_{yi} - \frac{c+\gamma}{N}\sum_j v_{xj} + \frac{c+\gamma}{N^2}\sum_y \sum_j v_{yj} + \gamma v_{xi} \\ &= -\frac{c+\gamma}{N} + \gamma v_{xi}.\end{aligned}$$

Upon substitution we obtain

$$\begin{aligned}\left[\mathbf{T}^{mod}\mathbf{v}^{zs}\right]_{xi} &= -\frac{c+\gamma}{N} + \gamma v_{xi} - \left[\lambda_1\hat{\mathbf{e}}\right]_{xi} \\ &= \gamma\left(v_{xi} - \frac{1}{N}\right) \quad = \quad \gamma v_{xi}^{zs}\end{aligned}$$

Therefore there is a degenerate eigenvalue of $\lambda_2 = \gamma$ that corresponds to the zerosum subspace.

C.3.3 Determination of λ_3

To determine λ_3 we show that the invalid subspace is an eigenspace of $\mathbf{T}^{mod}$ with a corresponding eigenvalue $\lambda_3 = -c + \gamma$.

Firstly, consider the matrix $\mathbf{S}$ which is the sum of outer products of all valid solutions $\mathbf{v}$ and is derived in Appendix C.4. Since the invalid subspace is mutually orthogonal to both the valid subspace and $\hat{\mathbf{e}}$, $\mathbf{S}\mathbf{v}^{inv} = \mathbf{0}$, where $\mathbf{0}$ is the vector of all zeros. Therefore the eigenvalue of $\mathbf{S}$ in the invalid subspace is zero.

Now by establishing an expression for $\mathbf{T}^{mod}$ in terms of $\mathbf{S}$, we shall determine the eigenvalues of $\mathbf{T}^{mod}$ in the invalid subspace. By considering the equations (C.7) and (C.9) the connection matrix $\mathbf{T}^{mod}$ may be written as,

$$\mathbf{T}^{mod} = \frac{c+\gamma}{N(N-2)!}\mathbf{S} + (c+\gamma)(1-N)\hat{\mathbf{e}}\hat{\mathbf{e}}^T - c\mathbf{I}. \qquad \text{(C.8)}$$

Equation (C.8) may be verified by following the same procedure as used in Appendix C.1.3. The details of this procedure are left to the reader. The eigenvalue of $\mathbf{T}^{mod}$ in the invalid subspace can now be determined by considering the individual terms in equation (C.8). Multiplication of $\mathbf{S}$ by $\frac{c+\gamma}{N(N-2)!}$ scales the eigenvalues of $\mathbf{S}$, but the eigenvalue of $\mathbf{S}$ in the invalid subspace remains zero. Similarly, multiplication of $\hat{\mathbf{e}}\hat{\mathbf{e}}^T$ by a constant affects only the eigenvalue of the eigenvector in the direction of $\hat{\mathbf{e}}$. The remaining term, $-c\mathbf{I}$ introduces an eigenvalue of $-c$ in the invalid subspace. Therefore, for a vector $\mathbf{v}^{inv}$ which lies in the invalid subspace

$$\mathbf{T}^{mod}\mathbf{v}^{inv} = -c\mathbf{v}^{inv}$$

i.e. there is a degenerate eigenvalue $\lambda_3 = -c$ that corresponds to the invalid subspace.

C.4 Derivation of S

To assist in the determination of eigenvalues in the invalid subspace, it is necessary to first derive the matrix **S**, which is formed from the sum of outer products of all the valid solutions **v**. The derivation of **S** presented here follows that given in (Aiyer et al., 1990).

Let N be the number of cities in the TSP and $\mathbf{v}^{(a)}$ be a valid $0-1$ vector *i.e.* it satisfies the constraints (3.3) and (3.4) and $v_{xi} \in \{0,1\}$. Since **S** is formed from the sum of outer products of all valid solutions, **S** is given by

$$\begin{aligned} \mathbf{S} &= \sum_a \mathbf{v}^{(a)}\mathbf{v}^{(a)^T} \\ \Rightarrow S_{xi,yj} &= \sum_a v_{xi}^{(a)} v_{yj}^{(a)}. \end{aligned}$$

Now, consider the following cases :

- $i = j$ and $x \neq y$.
  In a valid solution it is not possible for $v_{xi} = 1$ and $v_{yi} = 1$ if $x \neq y$. Therefore $S_{xi,yi} = 0 \ \forall i,\, x \neq y$.

- $i \neq j$ and $x = y$.
  In a valid solution it is not possible for $v_{xi} = 1$ and $v_{xj} = 1$ if $i \neq j$. Therefore $S_{xi,xj} = 0 \ \forall x,\, i \neq j$.

- $i = j$ and $x = y$.
  There are $(N-1)!$ valid solutions $\mathbf{v}^{(a)}$ with city x fixed at position i of the tour. Therefore $S_{xi,xi} = (N-1)! \ \forall x, i$.

- $i \neq j$ and $x \neq y$.
  City x is fixed at position i and city y is fixed at position j of the tour. The remaining cities may be arranged in $(N-2)!$ possible permutations. Therefore $S_{xi,yj} = (N-2)! \ \forall x \neq y,\, i \neq j$.

So it is clear that the matrix **S** may be expressed as

$$[\mathbf{S}]_{xi,yj} = (N-1)!\ \delta_{xy}\delta_{ij} + (N-2)!\ (1-\delta_{xy})(1-\delta_{ij}). \qquad \text{(C.9)}$$

Appendix D

Valid Subspace Mapping for the Ising Spin problem

The valid subspace mapping (Gee, 1993; Gee et al., 1993; Gee and Prager, 1994) is the current state-of-the-art method for mapping a combinatorial optimisation problem onto an optimisation network. In this section we will show how to use the valid subspace approach to map the Ising spin problem onto an optimisation network. In addition, a modified hysteretic annealing function that encourages the formation of good solutions is derived for the Ising spin problem.

Before considering the Ising spin problem in detail, it is necessary to demonstrate how the valid subspace mapping may be applied to a general combinatorial optimisation problem (Gee, 1993). Many combinatorial optimisation problems may be expressed as the minimisation of a quadratic objective function, subject to a set of linear constraints:

$$\begin{array}{rl} \text{minimise} & E^{obj} = -\frac{1}{2}\mathbf{v}^T\mathbf{T}\mathbf{v} - \mathbf{v}^T\mathbf{b} \\ \text{subject to} & \mathbf{A}\mathbf{v} = \mathbf{y} \\ \text{where} & v_i \in [0,1]. \end{array}$$

To map such a problem onto an optimisation network, the valid subspace approach sets the Lyapunov function for the network to be $E^{lyap} = E^{obj} + E^{cns}$. The penalty function is given by

$$E^{cns} = \frac{1}{2}c\|\mathbf{v} - (\mathbf{T}^{val}\mathbf{v} + \mathbf{s})\|^2 \qquad \text{(D.1)}$$

where c is a positive constant, $\mathbf{T}^{val}$ is a projection matrix and $\mathbf{s}$ is an offset vector[1]. The projection matrix and offset vector are given by (Gee, 1993)

$$\mathbf{T}^{val} = \mathbf{I} - \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{A} \qquad \text{(D.2)}$$

$$\mathbf{s} = \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{y}. \qquad \text{(D.3)}$$

D.1 Deriving $\mathbf{T}^{val}$ and s for the Ising Spin problem

Before proceeding with the valid subspace mapping of the Ising spin problem it is necessary to restate the problem representation that was introduced in Section 5.2.1. In order to solve the Ising spin problem, each element in the Ising spin model is assigned two neurons with outputs $v_i^{(w)}$ and $v_i^{(b)}$ respectively. If $v_i^{(b)} = 1$ and $v_i^{(w)} = 0$ then element i

[1]In Chapter 3 when the valid subspace mapping was used to map the TSP onto an optimisation network, the projection matrix $\mathbf{T}^{val}$ was renamed $\mathbf{T}^{zs}$ and the offset vector $\mathbf{s}$ was renamed $\hat{\mathbf{e}}$.

has been assigned the state `black`. Similarly, if $v_i^{(w)} = 1$ and $v_i^{(b)} = 0$ then element i has been assigned the state `white`. The output of the network may be represented as the column vector,

$$\mathbf{v} = [v_1^{(w)}, v_1^{(b)}, v_2^{(w)}, v_2^{(b)}, \dots v_N^{(w)}, v_N^{(b)}]^T.$$

The constraints for the problem mapping were given in equation (5.2) and have been reproduced here for completeness,

$$v_i^{(w)} + v_i^{(b)} = 1 \quad \forall\, i \in \{1, \dots N\}.$$

These constraints may be written in vector notation as $\mathbf{A}\mathbf{v} = \mathbf{y}$ where $\mathbf{A}$ is the $N \times 2N$ matrix given by,

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 1 \end{bmatrix} \tag{D.4}$$

and $\mathbf{y}$ is the $N \times 1$ column vector with all entries equal to one.

Before the projection matrix and the offset vector for the valid subspace mapping of the Ising spin problem are determined, we shall evaluate some useful auxiliary expressions. By substituting from equation (D.4) it can be shown that

$$\mathbf{A}\mathbf{A}^T = 2\mathbf{I}$$

and therefore

$$\left(\mathbf{A}\mathbf{A}^T\right)^{-1} = \frac{1}{2}\mathbf{I}. \tag{D.5}$$

Also, the $2N \times 2N$ matrix $\mathbf{A}^T\mathbf{A}$ is given by

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 1 & 1 \end{bmatrix}. \tag{D.6}$$

Now the offset vector for the valid subspace mapping of the Ising spin problem can be determined by substituting from equations (D.4) and (D.5) into equation (D.3), *i.e.*

$$\begin{aligned} \mathbf{s} &= \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{y} \\ &= \frac{1}{2}\mathbf{A}^T\mathbf{y} \\ &= \frac{1}{2}\mathbf{o} \end{aligned} \tag{D.7}$$

where $\mathbf{o}$ is a $2N \times 1$ column vector with all entries equal to one. Similarly, the projection matrix for the Ising spin problem can be determined by substituting from equations (D.5) and (D.6) into equation (D.2), *i.e.*

$$\mathbf{T}^{val} = \mathbf{I} - \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{A}$$

$$
\begin{aligned}
&= \mathbf{I} - \frac{1}{2}\mathbf{A}^T\mathbf{A} \\
&= \begin{bmatrix}
0.5 & -0.5 & 0 & 0 & \cdots & 0 & 0 \\
-0.5 & 0.5 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0.5 & -0.5 & \cdots & 0 & 0 \\
0 & 0 & -0.5 & 0.5 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & 0 & \cdots & 0.5 & -0.5 \\
0 & 0 & 0 & 0 & \cdots & -0.5 & 0.5
\end{bmatrix}.
\end{aligned}
\tag{D.8}
$$

D.2 Evaluating the Penalty and Annealing Functions

Given the projection matrix and offset vector for the valid subspace mapping of the Ising spin problem, we can now proceed to evaluate the penalty function E^{cns}. Expanding equation (D.1) we obtain[2]

$$E^{cns} = -\frac{c}{2}\mathbf{v}^T(\mathbf{T}^{val} - \mathbf{I})\mathbf{v} - c\mathbf{v}^T\mathbf{s} + \frac{c}{2}\mathbf{s}^T\mathbf{s}.$$

By substituting equation (D.8) for $\mathbf{T}^{val}$ and equation (D.7) for $\mathbf{s}$, the penalty function simplifies to

$$
\begin{aligned}
E^{cns} &= \frac{c}{4}\sum_{i=1}^{N}\left(v_i^{(w)} + v_i^{(b)}\right)^2 - \frac{c}{2}\sum_{i=1}^{N}\left(v_i^{(w)} + v_i^{(b)}\right) + \frac{c}{4} \\
&= \frac{c}{4}\sum_{i=1}^{N}\left(v_i^{(w)} + v_i^{(b)} - 1\right)^2.
\end{aligned}
\tag{D.9}
$$

It is easily verified that the penalty function E^{cns} is zero only when the constraints for the Ising spin problem are satisfied. With c set to a large positive constant, the penalty function E^{cns} is positive for any point that does not satisfy the constraints for the Ising spin problem and a simple gradient descent argument reveals that the penalty function encourages the network state $\mathbf{v}$ to lie on the valid subspace.

To improve the quality of solutions found by an optimisation network approach to the Ising spin problem it is desirable to include some form of annealing. Modified hysteretic annealing is a principled approach to annealing, which allows the solution quality to be improved whilst maintaining the ability to guarantee convergence to a valid solution. While the principles motivating the modified hysteretic annealing technique may be found in Chapter 4, it is sufficient to know that the modified hysteretic annealing function, as given by equation (4.9), is[3]

$$E^{mod} = -\frac{\gamma}{2}\mathbf{v}^T\mathbf{T}^{val}\mathbf{v}.$$

Substitution of equation (D.8) for $\mathbf{T}^{val}$ gives the modified hysteretic annealing function for the Ising spin problem as

$$E^{mod} = -\frac{\gamma}{4}\sum_{i=1}^{N}\left(v_i^{(w)}\left(v_i^{(w)} - v_i^{(b)}\right) + v_i^{(b)}\left(v_i^{(b)} - v_i^{(w)}\right)\right)$$

[2]Note that $\mathbf{T}^{val}$ is symmetric and since $\mathbf{T}^{val}$ is a projection matrix , $\mathbf{T}^{val}\mathbf{T}^{val} = \mathbf{T}^{val}$. Also, s lies in the nullspace of $\mathbf{T}^{val}$, so $\mathbf{T}^{val}\mathbf{s} = \mathbf{0}$ where $\mathbf{0}$ is the zero vector.

[3]As previously stated the projection matrix for the TSP was named $\mathbf{T}^{zs}$ and so it is appropriate to make the substitution of $\mathbf{T}^{val}$ for $\mathbf{T}^{zs}$ in equation (4.9).

$$= \quad -\frac{\gamma}{4}\sum_{i=1}^{N}\left(v_i^{(w)} - v_i^{(b)}\right)^2 . \tag{D.10}$$

Bibliography

Aarts, E. and Korst, J. (1989). *Simulated Annealing and the Boltzmann Machine.* Wiley Interscience series in Discrete Mathematics and Optimisation. John Wiley and Sons.

Abbott, L. F. and Arian, Y. (1987). Storage capacity of generalised networks. *Physical Review A*, 36(10):5091–5094.

Abe, S. (1993). Global convergence and suppression of spurious states of the Hopfield neural networks. *IEEE Transactions on Circuits and Systems*, 40(4):246–257.

Abe, S. (1996). Convergence acceleration of the Hopfield network by optimising integration step sizes. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, 26(1).

Abe, S. and Gee, A. H. (1995). Global convergence of the Hopfield neural network with nonzero diagonal elements. *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, 42(1):39–45.

Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science*, 9:147–169.

Aiyer, S. V. B. (1991). *Solving Combinatorial Optimisation Problems using Neural Networks with Applications in Speech Recognition.* PhD thesis, Cambridge University.

Aiyer, S. V. B., Niranjan, M., and Fallside, F. (1990). A theoretical investigation into the performance of the Hopfield model. *IEEE Transactions on Neural Networks*, 1(2):204–215.

Baldi, P. (1988). Neural networks, orientations of the hypercube and algebraic threshold functions. *IEEE Transactions on Information Theory*, 34(3):523–530.

Baldi, P. and Venkatesh, S. S. (1987). Number of stable points for spin-glasses and neural networks of higher orders. *Physical Review Letters*, 58(9):913–916.

Baldi, P. and Venkatesh, S. S. (1993). Random interactions in higher-order neural networks. *IEEE Transactions on Information Theory*, 39(1):274–283.

Bang, S. H., Chen, O., Chang, J., and Sheu, B. J. (1995). Paralleled hardware annealing in multilevel Hopfield neural networks for optimal solutions. *IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing*, 42(1):46–49.

Beyer, D. A. and Ogier, R. G. (1991). Tabu learning: A neural network search method for solving nonconvex optimisation problems. In *Proceedings of the International Joint Conference on Neural Networks*, Singapore.

Bofill, P. (1993). Higher-order networks for the optimisation of block designs. In *Proceedings of the International Workshop on Artificial Neural Networks IWANN'93*, pages 114–118.

Brandt, R. D., Wang, Y., Laub, A., and Mitra, S. K. (1988). Alternative networks for solving the travelling salesman problem and the list-matching problem. In *Proceedings of the International Conference on Neural Networks*, volume 2, pages 333–340.

Chao, D. Y., Wang, D. T., and Hung, D. D. C. (1993). Convergence time and memory capacity of higher-order Hopfield associative memory with multi-valued neurons. *The Computer Journal*, 36(6):554–561.

Chen, H. H., Lee, Y. C., Sun, G. Z., Lee, H. Y., Maxwell, T., and Giles, C. L. (1986). High order correlation model for associative memory. In Denker, J., editor, *AIP Conference 151 - Neural Networks for Computing*, pages 86–99.

Cooper, B. S. (1994). Selected applications of neural networks in telecommunications systems – A review. *Australian Telecommunications Research Journal*, 28(2):9–29.

Cooper, B. S. (1995a). Higher order neural networks – Can they help us optimise? In *Proceedings of the Sixth Australian Conference on Neural Networks (ACNN'95)*, pages 29–32.

Cooper, B. S. (1995b). Higher order neural networks for combinatorial optimisation – Improving the scaling properties of the Hopfield network. In *Proceedings of the International Conference on Neural Networks*, volume 4, pages 1855–1860.

Dembo, A., Farotimi, O., and Kailath, T. (1991). High-order absolutely stable neural networks. *IEEE Transactions on Circuits and Systems*, 38(1):57–65.

Eberhardt, S. P., Daud, T., Kerns, D. A., Brown, T. X., and Thakoor, A. P. (1991). Competitive neural architecture for hardware solution to the assignment problem. *Neural Networks*, 4:431–442.

Eberhardt, S. P., Tawel, R., Brown, T. X., Daud, T., and Thakoor, A. P. (1992). Analog VLSI neural networks: Implementation issues and examples in optimisation and supervised learning. *IEEE Transactions on Industrial Electronics*, 39(6):552–564.

Fogel, D. B. (1994). An introduction to simulated evolutionary optimisation. *IEEE Transactions on Neural Networks*, 5(1):3–14.

Gardner, E. (1987). Multiconnected neural network models. *Journal of Physics A*, 20:3453–3464.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability - A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company.

Gee, A. H. (1993). *Problem Solving with Optimisation Networks.* PhD thesis, University of Cambridge.

Gee, A. H., Aiyer, S. V., and Prager, R. (1993). An analytical framework for optimising neural networks. *Neural Networks*, 6(1):79–97.

Gee, A. H. and Prager, R. (1995). Limitations of neural networks for solving travelling salesman problems. *IEEE Transactions on Neural Networks*, 6(1):280–282.

Gee, A. H. and Prager, R. W. (1994). Polyhedral combinatorics and neural networks. *Neural Computation*, 6(1):161–180.

Giles, C., Miller, C. B., Chen, D., Chen, H., Sun, G. Z., and Lee, Y. C. (1992). Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, 4:393–405.

Giles, C. L., Griffin, R. D., and Maxwell, T. (1988). Encoding geometric invariances in higher order neural networks. In Anderson, D. Z., editor, *Neural Information Processing Systems*, New York, N. Y. American Institute of Physics.

Giles, C. L. and Maxwell, T. (1987). Learning, invariance and generalisation in high-order networks. *Applied Optics*, 26(23):4972–4978.

Giles, C. L. and Omlin, C. W. (1993). Extraction, insertion and refinement of symbolic rules in dynamically driven recurrent neural networks. *Connection Science*, 5(3 & 4):307–337.

Giles, C. L. and Omlin, C. W. (1994). Pruning recurrent neural networks for improved performance. *IEEE Transactions on Neural Networks*, 5(5):848–851.

Goldberg, D. (1989). *Genetic Algorithms in Search, Optimisation and Machine Learning.* Addison-Wesley.

Goudreau, M. W., Giles, C. L., Chakradhar, S. T., and Chen, D. (1994). First-order versus second-order single-layer recurrent neural networks. *IEEE Transactions on Neural Networks*, 5(3):511–513.

Greene, J. and Supowit, K. (1984). Simulated annealing without rejected moves. In *Proceedings of the IEEE International Conference on Computer Design*, pages 658–663, New York.

Grotschel, M. and Holland, O. (1991). Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming*, 51:141–202.

Hamilton, A., Murray, A. F., Baxter, D. J., Churcher, S., Reekie, H. M., and Tarassenko, L. (1992). Integrated pulse stream neural networks: Results, issues and pointers. *IEEE Transactions on Neural Networks*, 3(3):385–393.

Hecht-Nielsen, R. (1989). *Neurocomputing.* Addison-Wesley.

Hertz, J., Krogh, A., and Palmer, R. (1991). *Introduction to the theory of neural computation.* Addison Wesley.

Hopcroft, J. and Ullman, J. (1979). *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79:2554–2558.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, 81:3088–3092.

Hopfield, J. J. and Tank, D. W. (1985). Neural computation of decisions in optimisation problems. *Biological Cybernetics*, 52:141–152.

Horne, B. G. and Giles, C. L. (1995). An experimental comparison of recurrent neural networks. In *Neural Information Processing Systems 7*.

Johnson, D. E., Marsland, J. S., and Eccleston, W. (1995). Neural network implementation using a single MOST per synapse. *IEEE Transactions on Neural Networks*, 6(4):1008–1011.

Johnson, D. S. (1990). Local optimisation and the travelling salesman problem. In Goos, G. and Hartmanis, J., editors, *Automata, Languages and Programming*, number 442 in Lecture Notes in Computer Science, pages 446–461. Springer.

Kamgar-Parsi, B. and Kamgar-Parsi, B. (1990). On problem solving with Hopfield neural networks. *Biological Cybernetics*, 62:415–423.

Karlholm, J. M. (1993). Associative memories with short-range higher order couplings. *Neural Networks*, 6:409–421.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimisation by simulated annealing. *Science*, 220(4598):671–680.

Kirkpatrick, S. and Toulouse, G. (1985). Configuration space analysis of travelling salesman problems. *Journal de Physique*, 46:1277–1292.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A., and Shmoys, P. B., editors (1985). *The travelling salesman problem: a guided tour of combinatorial optimisation.* John Wiley & Sons.

Lawler, E. L. and Wood, D. E. (1966). Branch-and-bound methods: A survey. *Operations Research*, 14:699–719.

Lee, B. and Sheu, B. J. (1993). Paralleled hardware annealing for optimal solutions on electronic neural networks. *IEEE Transactions on Neural Networks*, 4(4):588–599.

Lee, Y. C., Doolen, G., Chen, H. H., Sun, G. Z., Maxwell, T., Lee, H. Y., and Giles, C. L. (1986). Machine learning using a higher order correlation network. *Physica D*, 22:276–306.

Lin, S. (1965). Computer solutions of the travelling salesman problem. *The Bell System Technical Journal*, 44(10):2245–2269.

Lin, S. and Kernighan, B. W. (1973). An effective heuristic algorithm for the travelling salesman problem. *Operational Research*, 21(2):498–516.

Lister, R. (1990). Segment reversal and the travelling salesman problem. In *Proceedings of the International Joint Conference on Neural Networks*, volume 1, pages 424–427.

Lister, R. (1993). Annealing networks and fractal landscapes. In *Proceedings of the International Conference on Neural Networks*, volume 1, pages 257–262.

Lister, R. (1994). Simulated annealing: Quasi-fractals and quasi-failures. In Stonier, R. and Yu, X., editors, *Complex Systems: Proceedings of the Second National Conference on Complex Systems*, pages 369–376. IOS Press.

Lister, R. (1995). Fractal strategies for NN scaling. In Arbib, M., editor, *Handbook of Brain Theory and Neural Networks*. MIT Press.

Matsuda, S. (1995). Stability of solutions in Hopfield neural network. *Systems and Computers in Japan*, 26(5):67–78.

Matsui, N. and Nakabayashi, K. (1995). Minimum searching by extended Hopfield network. In *Proceedings of the International Conference on Neural Networks*, volume 5, pages 2648–2651.

Maxwell, T., Giles, C. L., Lee, Y. C., and Chen, H. H. (1986). Nonlinear dynamics of artificial neural systems. In Denker, J. S., editor, *AIP Conference 151 - Neural Networks for Computing*, pages 299–304.

Miller, C. B. and Giles, C. L. (1993). Experimental comparison of the effect of order in recurrent neural networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(4):849–872.

Miller, K. R. and Zunde, P. (1992). High-order attention shifting networks for relational structure matching. In *Proceedings of the International Joint Conference on Neural Networks IJCNN92-Baltimore*, volume 4, pages 499–506.

Mjolsness, E., Garrett, C. D., and Miranker, W. L. (1991). Multiscale optimisation in neural nets. *IEEE Transactions on Neural Networks*, 2(2):263–274.

Ogier, R. G. and Beyer, D. A. (1990). Neural network solution to the link scheduling problem using convex relaxation. In *Proceedings of the 1990 IEEE Global Telecommunications Conference*, pages 1371–1376.

Ohta, M., Anzai, Y., Yoneda, S., and Ogihara, A. (1993). A theoretical analysis of neural networks with nonzero diagonal elements. *IEICE Transactions Fundamentals*, E76-A(3):284–291.

Padberg, M. W. and Rinaldi, G. (1991). A branch and cut algorithm for the resolution of large-scale symmetric travelling salesman problems. *SIAM Review*, 33:60–100.

Papadimitirou, C. H. and Steiglitz, K. (1982). *Combinatorial Optimisation - Algorithms and Complexity.* Prentice Hall.

Perantonis, S. J. and Lisboa, P. J. G. (1992). Translation, rotation and scale invariant pattern recognition by high-order neural networks and moment classifiers. *IEEE Transactions on Neural Networks*, 3(2):241–251.

Peretto, P. (1984). Collective properties of neural networks: A statistical physics approach. *Biological Cybernetics*, 50:51–62.

Personnaz, L., Guyon, I., and Dreyfus, G. (1987). High-order neural networks: Information storage without errors. *Europhysics Letters*, 4(8):863–867.

Peterson, C. and Anderson, J. (1988). Neural networks and NP-complete optimisation problems: A performance study on the graph bisection problem. *Complex Systems*, 2:59–89.

Poteryaiko, I. (1991). On Lyapunov function for neural network models with multi-unit interaction. In *Proceedings of the International Conference on Artificial Neural Networks*, pages 21–23. Elsevier Science Publishers.

Protzel, P. W., Palumbo, D. L., and Arras, M. (1993). Performance and fault-tolerance of neural networks for optimisation. *IEEE Transactions on Neural Networks*, 4(4):600–614.

Psaltis, D. and Park, C. H. (1986). Nonlinear discriminant functions and associative memories. In Denker, J. S., editor, *AIP Conference 151 - Neural Networks for Computing*, pages 370–375.

Reinelt, G. (1991). TSPLIB - A travelling salesman problem library. *ORSA Journal on Computing*, 3:376–384. ftp://elib.zib-berlin.de/pub/mp-testdata/tsp/index.html.

Reinelt, G. (1994). *The Travelling Salesman - Computational Solutions for TSP Applications.* Lecture Notes in Computer Science. Springer-Verlag.

Salem, G. J. and Young, T. Y. (1991). Interpreting line drawings with higher order neural networks. In *Proceedings of the International Joint Conference on Neural Networks IJCNN'91-Seattle*, volume 1, pages 713–721.

Smith, M. J. and Portmann, C. L. (1989). Practical design and analysis of a simple "neural" optimisation circuit. *IEEE Transactions on Circuits and Systems*, 36(1):42–50.

Sorkin, G. (1990). Simulated annealing on fractals: Theoretical analysis and relevance for combinatorial optimisation. In Dally, W., editor, *Advanced Research in VLSI*, pages 331–351. MIT Press.

Tagliarini, G. A., Christ, J. F., and Page, E. (1991). Optimisation using neural networks. *IEEE transactions on computers*, 40(12):1347–1358.

Tomikawa, Y. and Nakayama, K. (1995). Convergence analysis of recurrent neural network with self-loops based on eigenvalues of a connection matrix. In *Proceedings of the International Conference on Neural Networks*, volume 5, pages 2642–2647.

Truyen, B. and Cornelis, J. (1995). Adiabatic layering: A new concept of hierarchical multi-scale optimisation. *Neural Networks*, 8(9):1373–1378.

Tsioutsias, D. and Mjolsness, E. (1996). A multiscale attentional framework for relaxation neural networks. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems 8*. MIT Press.

Van den Bout, D. E. and Miller, T. K. (1989). Improving the performance of the Hopfield-Tank neural network through normalisation and annealing. *Biological Cybernetics*, 62:129–139.

Van den Bout, D. E. and Miller, T. K. (1990). Graph partitioning using annealed neural networks. *IEEE Transactions on Neural Networks*, 1(2):192–203.

Watrous, R. L. and Kuhn, G. M. (1992). Induction of finite-state languages using second-order recurrent networks. *Neural Computation*, 4:406–414.

Wilson, V. and Pawley, G. (1988). On the stability of the travelling salesman algorithm of Hopfield and Tank. *Biological Cybernetics*, 58:63–70.

Xu, X. and Tsai, W. T. (1991). Effective neural algorithms for the travelling salesman problem. *Neural Networks*, 4(2):193–205.

Zeng, Z., Goodman, R. M., and Smyth, P. (1993). Learning finite state machines with self-clustering recurrent networks. *Neural Computation*, 5:976–990.

Zissimopoulos, V., Paschos, V., and Pekergin, F. (1991). On the approximation of NP-complete problems by using the Boltzmann machine method : The cases of some covering and packing problems. *IEEE Transactions on Computers*, 40(12):1413–1418.

Zurada, J. M. (1992). *Introduction to Artificial Neural Systems.* West Publishing Company.