94-74

Artificial Neural Networks for Modelling and Control of Nonlinear Systems

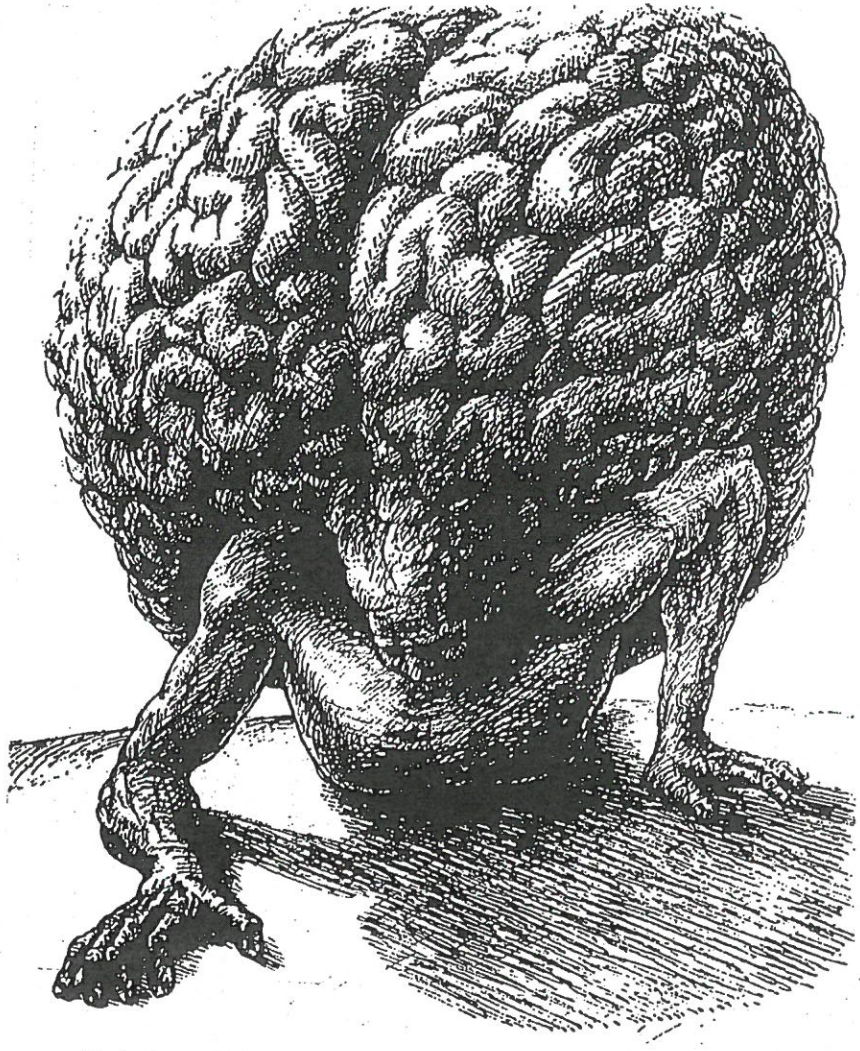
Wetenschappelijke Prijs SCK•CEN 1994

Johan Suykens and Bart De Moor

ESAT-SISTA Report 1994-74I

Katholieke Universiteit Leuven
Department of Electrical Engineering (ESAT)
Research Group SISTA
Systems, Identification, System Theory and Automation

Kardinaal Mercierlaan 94
B-3001 Leuven, Belgium
tel: 32/16/220931 fax: 32/16/221855
email: johan.suykens@esat.kuleuven.ac.be
bart.demoor@esat.kuleuven.ac.be



Designing artificial neural networks is not a 9-to-5 job.

Contents

1 Introduction and survey 7
- 1.1 A quick glance through this text . . . 8
- 1.2 Information technology and neural nets . . . 9
  - 1.2.1 The broad picture . . . 10
  - 1.2.2 Neural nets, innovation and valorisation . . . 13
  - 1.2.3 Possible interactions with the SCK/CEN . . . 13
- 1.3 Conclusions . . . 15
- 1.4 Acknowledgments . . . 15

2 Neural information processing systems 17
- 2.1 From biological to artificial neural networks . . . 17
- 2.2 Some history . . . 20
- 2.3 Artificial neural networks . . . 21
  - 2.3.1 Architectures . . . 21
  - 2.3.2 Training and learning . . . 23
- 2.4 Neural models and model-based solutions . . . 24
  - 2.4.1 Mathematical modelling and model-based control . . . 24
  - 2.4.2 Neural networks for modelling . . . 28
  - 2.4.3 Neural nets for control . . . 29
  - 2.4.4 Why are neural nets so useful in modelling and control . . . 30

3 Modelling and prediction by neural networks 33
- 3.1 Feedforward neural networks are universal approximators . . . 33
- 3.2 From linear to nonlinear dynamical systems . . . 37
- 3.3 Parametrization by ANNs and paradigms of learning . . . 38
- 3.4 Real life examples . . . 42
  - 3.4.1 Neural networks and complex systems . . . 42
  - 3.4.2 Identification of a glass furnace . . . 44

4 Neural networks and modern control theory 45
- 4.1 Simple neural nets that accomplish complex control tasks . . . 46
  - 4.1.1 The inverted pendulum . . . 46
  - 4.1.2 A double inverted pendulum . . . 50
  - 4.1.3 Why is this NOT an academic example ? . . . 50
- 4.2 Neural state space model framework for identification and control . . . 52
- 4.3 Stability criteria: NL_q theory . . . 55
- 4.4 Learning within NL_q theory . . . 58
- 4.5 Unifications in the theory of neural networks, systems and control . . . 60

Bibliography 62

Chapter 1

Introduction and survey

"There is, however, a much larger sense in which changes in knowledge are causing or contributing to enormous power shifts. The most important economic development of our lifetime has been the rise of a new system for creating wealth based no longer on muscle but on mind. Labor in the advanced economy consists of people acting on information and information acting on people"
-excerpted from *Power Shift*, A. Toffler, 1990, pp 8-9.

In this Chapter, we present a general overview of the scientific and technological contributions of this memorandum, which are concentrated around a new powerful paradigm in the engineering sciences, which is that of *artificial neural networks*.

It is sometimes said that the 21-st century will be the century of the *brains* and the *brain research*. Actually, the first signs, indications and results are already there. The past decade has seen a dramatic increase in interest in neural networks, both biological and artificial ones, and part of these developments will be covered in this award proposal.

Loosely speaking, neural networks are computing machines, which are 'inspired' by the fundamental operation modes of the human brain (or at least what is already known about it right now, which is still not that much!). Artificial neural networks are engineered computational systems modeled after or inspired by the learning abilities and parallelism of biological nervous systems. Within the last decade, spectacular progress has been made in the neuro-sciences and more and more becomes known about the properties of neurons and the complex molecules that govern and affect their behavior. However, this biomedical and psychological knowledge is not yet sufficient to understand, from the engineering point of view, how these networks of neurons and

interconnected systems of neurons really operate. What today is known as an artificial neural network is basically a very drastic engineering simplification of what is actually going on in our brains. One should therefore be careful with the 'hype' around and the fashionable 'chit-chat' about neural networks.
This is precisely what we try to do in our approach of the analysis and design of neural networks: ***To confront this new paradigm of artificial neural nets with the - let's call them - classical mathematical engineering disciplines such as linear and nonlinear system theory, linear and nonlinear control theory and mathematical programming (optimization) frameworks.***

This Chapter is organized as follows: In Section 1.1 we first give a general chapter-by-chapter overview of the global text, while in Section 1.2. we situate the main results of this work in the wider context of the socio-economic reality of our information-technology-based society, both in Europe and the World. In Section 1.3, we formulate the main conclusions. We also explore in some details the potentialities of neural nets for innovations and valorisation and in Section 1.2.3. we point out how our research on neural nets might be applicable to research activities in SCK/CEN.

1.1 A quick glance through this text

In the sequel of this Chapter, we will situate the neural net research within a broader context of the information technology based society of the (very near) future. We will treat in a nutshell the increasing impact of information technology on our social-economic environment and describe briefly the innovative role played by neural nets in this context.

In Chapter 2, we summarize some (well-known) facts about *neural information processing systems* to equip the reader with the necessary background material to understand Chapters 3 and 4, which describe our main scientific contributions. In this Chapter, we highlight the main constituting elements of neural networks, which are neurons, connections (synapses and synaptic weights) and the important differences between neural architectures and the classical Von Neumann computers. We will also briefly treat the historical development of the field.

In Chapter 3, we start by treating one of the key properties of artificial (feedforward multilayer) neural networks, namely their ability of *universal approximation*, which loosely speaking implies that they can approximate any multidimensional nonlinear function arbitrarily closely. We then treat one of our main results, which is the parametrization of nonlinear models by feedforward neural nets, integrated in a so-called *neural state space model*. It is then shown on several examples that have been described in the literature, and on data acquired on a real industrial system (a Philips

glass oven) that the class of *neural state space models* provides the best models in the sense of simulation and prediction.

In Chapter 4, we summarize our results that we have obtained so far in one of the most challenging open problems concerning neural net controllers, namely assessing and proving closed-loop (robust) stability and control.
First, we show how neural net controllers can accomplish complex control tasks. We use the inverted pendulum problem to make our points. Basically the idea is that even simple neural nets can already stabilize highly nonlinear plants and deliver control strategies that bring the system from one working point to another. One may not think too soon that this is a purely academic exercise. Many large-scale industrial plants are nicely controllable around a working point, at which their behavior is approximately linear. However, it also happens (as a matter of fact, in present day just-in-time production methodologies more and more frequently) that the process needs to switch from one working point to another. Nowadays this is basically done by human operators that have the required *fingerspitzengefühl* to successfully accomplish this difficult task. However, we claim that neural nets in the near future will be able to do this as well, and even better than a human operator because it will be possible to take into account precise quantitative side conditions, such as minimal energy and speed or accuracy.
Finally, we provide the reader with a summary on some of our very recent developments in neural net research, namely a new unifying framework which we have called NL_q-theory, in which, among other things, it is possible to analyse the stability of the input-output behavior of neural control systems.

We have been publishing most of the insights summarized in this memorandum in the literature (De Moor (1988), Suykens, De Moor et.al (1993a,1993b, 1993c, 1994a, 1994b, 1994c, 1994d, 1994e, 1994f, 1994g). We would refer the interested reader to those publications and a complete list of bibliographic references at the end of this memorandum for more details and examples.

1.2 Information technology and neural nets

First, we situate the broad socio-economic context, in which our world, the basics of which since the 19th century were based on labor and capital, is gradually shifting towards a knowledge-based society. Next we point out how neural nets may contribute (and actually are already doing!) to implement this new information technology based society. Finally, we make an attempt to demonstrate how neural nets might prove useful within the context of activities of the SCK/CEN.

1.2.1 The broad picture

After the microelectronics and integrated circuits era began in the late fifties, a phenomenal technology progress led to a spectacular and unprecedented evolution beyond all expectations. Microelectronics is now crucial to all information technology industries, such as the computer, consumer and communications industries. The steady decrease in integrated circuit cost per transistor has brought a stream of products to the market previously impossible to manufacture in a cost-effective way, if at all. Information processing system designers now have the disposal of fast and inexpensive computers or signal processors, leading to an explosion of applications in, *e.g.*, digital signal processing applications in telecommunications, video and audio, radar and sonar, process control and automation. Microelectronics is a challenging, but at the same time a very predictable technology. The available computing power is still steadily increasing, roughly doubling every two years. With the advent of multi-processor systems this increase is even more dramatic. The message is that in the future, systems designers will not have to worry so much about the available computing power. The question is, rather, whether it will be possible at all to take full profit from this evolution, as there is usually no point in solving the same problems at a speed which is doubled every two years.

The observation to be made in this proposal is that the most significant break-through can be achieved if we are prepared to drastically re-think our information processing problems, solutions and systems. Spurred by the significant advances in computer technology, information system research is to be directed towards developing *novel intelligent information processing systems*. Here 'intelligent' refers to the use of *advanced mathematical computations* and *new scientific paradigms* inspired by biological neural systems .

Following similar tendencies as described above, the last two decades or so, research and education in electrical engineering is no longer confined to what could be considered the classical field of electricity, electronics or power engineering.
Throughout the world, electrical engineering departments have broadened their scope and have diversified their interests to start up research in image processing and enhancement, algorithms, telecommunications, signal processing, control theory, neural networks, artificial intelligence, fuzzy logic etc ...[1].

The present proposal is an excellent illustration of these new directions and evolutions. Several major developments are actually married into one central engineering

[1] This is not only reflected in the research subjects of the Electrical Engineering Departments worldwide, but also in the courses that are taught by the staff of these departments. As an example, the basic courses in *Linear Algebra* in many places are given by people from EE Departments, despite the fact that this is a highly mathematical subject. Other courses include neural nets, computer-aided control system design, fuzzy systems, etc... all of which witness a dramatic shifting from the classical electrical engineering education to one that is more oriented towards information processing systems.

framework, namely that of *neural networks and model-based neural net controllers*.

Before we dive into the technicalities of this work, we would like to situate it into a wider context. Indeed, the growing diversification of the world of electronics is not only reflected in research and education (see above), but, maybe more importantly, it is a general trend in Europe and the World. We only present two figures to illustrate this observation [2]. From these figures, it is clear that the future of Europe is no longer

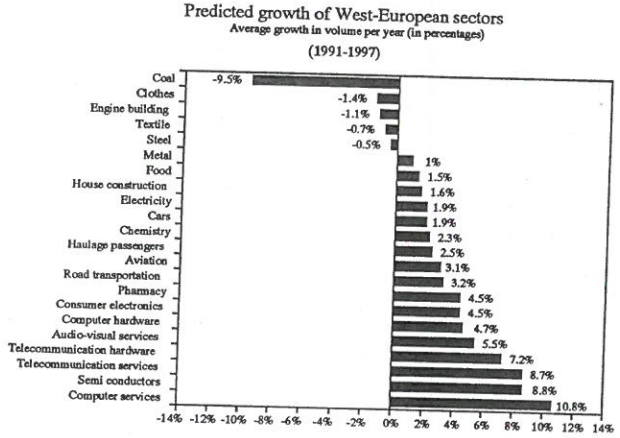


Figure 1.1 *Prediction of the growth of some Western European Economical Sectors. Clearly, the classical heavy industrial activities are on the decline, in favor of the field of* Information Technology. *At present, Europe is undergoing a major shift from classical industrial activities towards 'postmodern' computer and telecommunications applications.*

concentrated in the classical production sectors of the Third Industrial Revolution. Information technology has by now become one of the biggest sectors of the European economy and it is expected to become the largest by the end of the decade.
If we want to survive as one of the leading economical associations in the world, we should invest time, energy and money in our intelligence and creativity. Only in this way, we can maintain an economic growth which is badly needed from a socio-economic perspective and which in many ways is the best strategy to maintain a competitive

[2] These figures are borrowed and translated from *Investeren in de toekomst: Elektronika: Centraal in een vernieuwd industrie- en technologiebeleid.* (Fabrimetal, Vlaamse Gewest, IMEC, 1993)

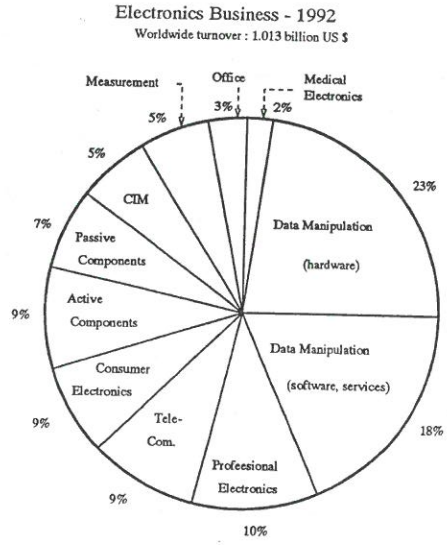


Figure 1.2 *The Electronics Business Worldwide. It is obvious that what is commonly called* Information Technology *occupies a most prominent place. The work that is proposed here is mainly related to Computer Integrated Manufacturing and Data Manipulation (software).*

position relative to the South-East Asian Tigers, Japan and the US [3].

The present work is a modest contribution to this larger program.

[3] A recent resolution of the European Parliament (January 1994, proposed by the commission on Economical and Monetarian Affairs and Industrial Policy) confirms the threat of an increasing dependency of the European electronics business from that of the US and Japan. The resolution proposes to strengthen the technological and financial basis of the European industry.

1.2.2 Neural nets, innovation and valorisation

Already today, there are many companies producing products based on neural net technology. The number of possible applications is infinitely large [4] ranging from speech and pattern recognition and classification in factory quality control (e.g. by the company Global Holonetics (USA)), via associative memory applications in for instance electroencephalogram analysis, loan risk assessors and mortgage-insurance underwriting (companies Nestor (USA), Adaptive Decision System (USA)), airline seat allocation (company BehavHeuristics (USA)), over explosives detectors to typical engineering problems like performance and robustness control of highly nonlinear chemical plants (company AI Ware (USA)) or robotic automation systems.

Specifically in this engineering context, the research on neural networks for modelling and control, when valorised within and applied to real industrial processes and systems, will potentially result in products and production strategies that *are cleaner, safer and faster, are more user and cost efficient, allow for just-in-time and 'zero-defect' production, have narrower tolerances, are more flexible and responsive to user's specifications, have an improved performance, etc ....*

The crucial new ingredient here is that systems can be optimized when a neural model is available, which allows to incorporate and exploit a priori information, ultimately leading to improved performance. Neural models also lead to Rapid Prototyping Methodologies, either on a real life prototype plant or within a *virtual engineering* setting, in which the real plant is replaced by a(n) (approximate) neural model which is then used in numerical simulations.

Both software and hardware ('neural-ware') products are becoming available with a gradual shift and increasing sophistication from pure software environments (such as e.g. the mathematical environment Matlab with its Neural Net Toolbox) via DSP (Digital Signal Processing) Boards (Hecht Nielsen Neurocomputer Corporation ANZA Basic Netware Package, NEC's Neuro-7 (PC-9800), Fujitsu's NEUROSIM/L (PC + MB86232-Board) over VLSI Boards (Neural semiconductors, Hitachi, Adaptive Solutions) to real VLSI designs (Neuro-chips) such as e.g. the Siemens SYNAPSE 1, which is a general purpose neuro-computer [5].

1.2.3 Possible interactions with the SCK/CEN

While as such up to this date we have no direct interaction nor experience with the scientific and technological activities of the SCK/CEN, it might be worthwhile to try

[4] The following examples come from Scientific American, November 1988, p.96.

[5] Ramacher U. et al.: *Synapse 1: Ein General-Purpose Neurocomputer*, Zentralabteilung Forschung und Entwicklung, Wiesbaden February 1993.

to explore possible connections and potential applications of the insights and neural net methodologies of this memorandum with the SCK/CEN research interests and activities. An non-exhaustive screening of some documents [6] has resulted in the following (non-complete) list of potential applications of neural nets within the SCK/CEN:

Waste and dismantling: - Assessment and processing of data from dismantling projects after nuclear reactor shut down, concerning management, bookkeeping, simulation and prediction of doses, costs, time and generated waste.

- Intelligent control of hydraulic remote telemanipulators for precise positioning of dismantling tools.
- Control and regulation of computer-aided tele-operations which are complex control problems necessary for maintenance, repair and dismantling, often consisting of repetitive tasks. Of particular importance here are mixed control-modes, in which computers (neural nets?) and human operators share responsibilities;
- Integrated feedback-based control of sensors (e.g. vision, tactile sensing), neural sensor fusion, actuators and tele-operated robots in a (sometimes unknown) nuclear environment.
- Investigation of correlations between certain isotopes in waste packages (that are weakly detectable) and key radionuclides in large excess. Neural nets might be successful here as high resolution separation algorithms starting from measurements.
- Nonlinear data processing, for instance processing of measurements of trace elements in solutions.

Waste and disposal: - Neural models of migration of radionuclides and diffusion of gases in geological repositories, starting from in situ experiments and field test generated measurements.

- Modelling of the diffusion of radionuclides through cement and using neural net technology in the design, evaluation and optimization of nondestructive analysis systems.
- Assessment and prediction of the long-term behavior and performance of geological disposal sites with respect to long-lived radioactive wastes.

Fuel and radiation protection - Data acquisition and control (e.g. temperature) of nuclear reactors using neural net technology.

[6] We have consulted the following sources: The 1993 Scientific Report, Nuclear Research Centre, Mol, Belgium; 1993 Jaarverslag SCK/CEN, Studiecentrum voor Kernenergie Mol, België; The 1992 Annual Report; The belgian deep repository project, brochure SCK/CEN; HADES, brochure SCK/CEN; Publications and open file reports on Characterization of conditioned radioactive wastes and disposal of radioactive wastes into the boom clay, SCK/SCK, December 1993;

- Modelling, monitoring and control of radioactivity levels in air, water an soil in normal and accidental (real-time decision support systems) situations.
- Correlate the mechanisms of radionuclide transfer in ecosystems to inputs such as seasonality, ageing, source characteristics, atmospheric conditions, etc....

It goes without saying that there must be many more potential applications in which the neural net technology could contribute significantly.

1.3 Conclusions

The following telegram style conclusions apply to the present memorandum:

- The twentyfirst century will be the one of the brain research. Both biological and neural networks will be in the spotlights.
- Neural networks might contribute significantly to our information technology based society of tomorrow. They will potentially result in new products ('neuralware', innovations through valorization of research results).
- The main results of our work are in the design and analysis of *model-based neural controllers*, in which both the model of the system to be controlled, as the controller are neural networks. The NL_q-framework that is summarized in Chapter 4 is an original unifying theory which allows to analyse the closed-loop stability of neural controlled systems and assess the robustness of closed-loop properties (such as stability and performance with respect to certain performance measures).
- There is no doubt that many of these insights are results are directly applicable within the research activities and interests of the SCK/CEN.

1.4 Acknowledgments

First of all, we would like to thank our 'common' promotor, Prof.Dr.Ir. Joos Vandewalle who, in 1987 decided to put the light on green for starting neural net research in Leuven. These efforts have resulted in numerous publications and several PhD theses. We would like to thank all these contributors, such as Lieven Vandenberghe, Tan Shaohua, Jianbin Hao, Dirk Thierens for their relentless efforts. The Research Council of the K.U.Leuven has stimulated the research on Neural Nets by granting us a Concerted Research Action [7]. All of these efforts have culminated in an *Interdisciplinary Center for Neural Networks*, in which mathematicians, physicists, engineers, physicians and other medical researchers, contribute and meet each other on a regular basis [8]. We would like to thank all the researchers involved for the stimulating intellectual environment.
We would also like to thank our friends at Stanford University, Berkeley (CA,USA), Cambridge University (UK), Linköping (Sweden) and Louvain-la-Neuve for the many stimulating discussions and invitations.
Finally, coming back home, we thank our friends and colleagues in Leuven. In particular, we thank (soon-to-be Dr.) Ir. Peter Van Overschee for providing us with the Latex style files used in printing this document and (soon-to-be Dr.) Ir. Ann Dierckx for providing us with information about the SCK/CEN and for several useful discussions.

JS, BDM.
Leuven, November 1994.

[7] Geconcerteerde Onderzoeksaktie 1990-1994: *Applicable Neural Networks*, GOA Vlaamse Gemeenschap, toegekend door Onderzoeksraad Leuven, Voorzitter Prof.Dr. Yvan Bruynseraede.

[8] Similar initiatives exist at other universities, such as the Center of Adaptive Systems at Boston University, where staff members are typically trained in at least three of the four fields: computer science, mathematics, psychology and neuro-biology (the so-called 3/4 Rule).

Chapter 2

Neural information processing systems

Cogito, ergo sum.
René Descartes (1596-1650).

In this chapter, we present a brief introduction on artificial neural networks, in order to situate our work within a general context. In Section 2.1. we start from biological networks while in Section 2.2. we give a brief discussion on the history of the subject. Architectures and learning modes are discussed together with application areas in Section 2.3. Some basic differences between neural and traditional computers are mentioned. Finally, in Section 2.4., we focus on the use of neural nets for modelling, prediction and control which will constitute the central themes of Chapters 3 and 4.

2.1 From biological to artificial neural networks

Artificial neural networks (ANNs) form a class of systems that is inspired by biological neural networks. They consists of a number of simple processing elements, called neurons, that are connected to each other. In most cases one or more layers of neurons are considered that are connected in a feedforward or recurrent way (Zurada (1992), Grossberg S. (1988), Lippmann (1987)). The strength of the interconnection is quantified by means of interconnection weights. Basic features of neural computers are that they work massively parallel, the weights have to be learned from a set of examples and can be adapted. The main differences between traditional Von Neumann type computers and neural computers are presented in Table 2.1.

Although ANNs can perform human brain-like tasks such as object and pattern recognition, speech recognition or associative memory, there is still a huge gap between the two. One of the limitations of neural computers up till now is the limited number of interconnections between neurons that can be implemented on chip, compared to the human brain. A striking example that illustrates the efficiency of biological neural networks with respect to today's computers, was given by Hopfield, who compared a pigeon with a SUN4/110 workstation in performing a face recognition task: if the SUN were as efficient as a pigeon it would have finished the task in 0.01 sec (see Table 2.2).

Nevertheless, although we are still far away from mimicing the human brain, from an engineering point of view, it is certainly a fruitful step to let us inspire by biology. Indeed ANNs have provided good solutions to many problems in various fields: examples include classification problems, vision, speech, signal processing, time series prediction, modelling and control, robotics, optimization, expert systems and financial applications (Hecht-Nielsen (1988b), Widrow *et al.* (1994)) (see Fig.2.1).

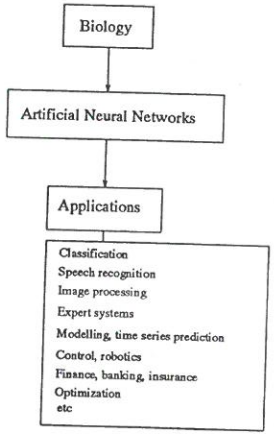


Figure 2.1: *The main source of inspiration for artificial neural networks is formed by the biological ones. There is basically no limit to the number of potential applications. In many domains neural nets have proved to be very useful when it comes to modelling and control of complex dynamic behavior.*

Classical computers	Neural computers
processes symbols and numbers	processes patterns
mathematical logic and logic circuits	static or dynamic nonlinear systems
sequential	by nature parallel
programming	training
software	learning of weights
high-precision arithmetic	low-precision, nonlinear mapping
ROM, RAM, binary memories	interconnecting weights of continuous value

Table 2.1: *Comparison between a classical Von Neumann digital computer and neural computers. A Von Neumann computer has a serial and digital architecture, and consists of unmodifiable connections and memory banks. The brain seems to be a computer that changes when it learns (by training), it appears to store and process information in the same places (which a classical computer does not do), its elements are analog instead of digital, and it is comparatively fault-tolerant. Moreover, there appears to be a premium on solutions that are fast and approximate instead of those that are slow but exact!*

Pigeon	Workstation SUN4/110
10^{12} synapses	0.5×10^8 transistors
clock 100 Hz	clock 24 MHz
recognition time of 1 sec	hours
10^{14} synapse-cycles	$> 4 \times 10^{19}$ synapse-cycles

Table 2.2: *Comparison between a pigeon and a workstation in recognizing the face of a person (Remark: synapse-cycles = number of synapses × clock frequency × time for accomplishing the task).*

2.2 Some history

In trying to understand the emergence of a new scientific discipline, it is often useful to enumerate its historical milestones, which is done in the following table. References can be found in the extensive bibliography at the end of this memorandum.

- 1942 [Mc Culloch and Pitts]: mathematical model of neurons, but no realizations.
- 1949 [Hebb]: first learning rule. One can memorise an object by adapting the weights.
- 1958 [Rosenblatt]: book on perceptrons, a machine that is capable of learning how to classify information by adapting the weights. Start of machine learning algorithms.
- 1960-1962 [Widrow & Hoff]: adalines and LMS rule
- 1969 [Minsky & Papert]: show theoretical limits of perceptrons as general computers.
- 23 years of hibernation but some 'stubborn' individuals (such as Grossberg, Amari, Fukushima, Kohonen, Taylor,...) continue doing research in neural nets.
- 1982 [Hopfield]: shows with energy functions that neural networks are capable of solving a large number of problems of the traveling salesman type. Re-emergence of the field.
- 1982 [Kohonen]: describes self-organizing map.
- 1986 [Rumelhart]: rediscovers backpropagation.
- 1987 [Minsky]: knowledge systems evolve in blocks of specialised agents rather than as a homogeneous network (criticism in his book 'The Society of Mind').
- 1991: reaction of the editor of the IEEE Transactions on Neural Networks in order to avoid a new hibernation.
- Present: Blossoming international research community active in neural nets, as witnessed by the existence of several international neural net societies and many international conferences. Theoretical and practical progress is continuously being made.

One might state that the evolution of the field of neural networks is characterized by a number of ups and downs. The 23 years period of hibernation is due to the fact that only networks consisting of one single layer were studied (perceptrons). Such models were unable e.g. to learn the well known EXOR problem, which is not a difficult problem in itself but cannot be represented by a single layer.

It was however a great breakthrough to introduce an additional hidden layer together with the backpropagation algorithm as a learning rule. It then became possible to learn difficult problems (Rumelhart *et al.* (1986), Werbos (1990)). Furthermore it was theoretically shown that such networks can represent any continuous nonlinear mapping (Cybenko (1989), Funahashi (1989), Hornik *et al.* (1989)) [1]. These important facts caused an explosion in artificial neural networks research and applications, which has now grown into a mature and blossoming engineering discipline.

2.3 Artificial neural networks

2.3.1 Architectures

An overview of the best known artificial neural network architectures is given in Table 2.3 (see Hecht-Nielsen (1988b)).

Year	Network	Inventor	Primary application
1957	Perceptron	Rosenblatt	character recognition
1960	Madaline	Widrow	adaptive modems, adaptive equalizers
1969	Cerebellatron	Mar, Albus	control motor action of robotic arms
1974	Backpropagation network	Werbos, Parker, Rumelhart	speech synthesis, adaptive control, banking
1977	Brain state in a box	Anderson	knowledge extraction, associative memory
1978	Neocognitron	Fukushima	character recognition
1978	Adaptive Resonance Theory	Carpenter, Grossberg	pattern recognition
1980	Self-organizing map	Kohonen	unsupervised learning
1982	Hopfield net	Hopfield	associative memory, optimization
1985	Bidirectional assoc. mem.	Kosko	associative memory
1985	Boltzmann machine	Hinton, Sejnowsky, Szu	pattern recognition for images, sonar, radar
1986	Counterpropagation	Hecht-Nielsen	image compression, statistical analysis
1988	Cellular neural nets	Chua, Yang	image processing

Table 2.3: *The best known artificial neural network architectures together with their year of introduction, their inventor/discoverer and their most important or typical applications.*

The most popular ANN architecture is certainly the multilayer *feedforward* neural network or backpropagation network[2]. In the sequel we will mainly consider these types of networks, together with *recurrent* neural networks, because these are most relevant for applications to modelling, prediction and control. In Fig.2.2 a multilayer *feedforward*

[1] We will come back to this so-called universal approximation property in Chapter 3.

[2] The term backpropagation network is misleading in the sense that 'backpropagation' is in fact one possible learning algorithm for a multilayer feedforward neural network and not an architecture in itself. However backpropagation was the first algorithm to train multilayer neural nets from a set of input-output patterns.

neural network with two hidden layers is presented (Zurada (1992)). One single neuron at a certain layer is described by a Mc Culloch-Pitts model

$$y = \sigma(\sum_i w_i x_i + \beta) ,$$

where y is the output of the neuron, which is equal to a weighted sum of incoming signals x_i (coming from other neurons) and a bias term β (or threshold), fed through the activation function $\sigma(.)$, which has typically a saturation-like characteristic (see some examples in Figure 3.1.). These relatively simple nonlinear processing elements are then arranged in layers and interconnected to each other. Hence the nonlinear mapping from the input to the output of the network is parametrized by the interconnection weights and the bias elements. In case the neural net consists of layers that are fed back to each other, the network is called *recurrent*. The neural net becomes then a nonlinear dynamical system instead of a static nonlinear mapping.

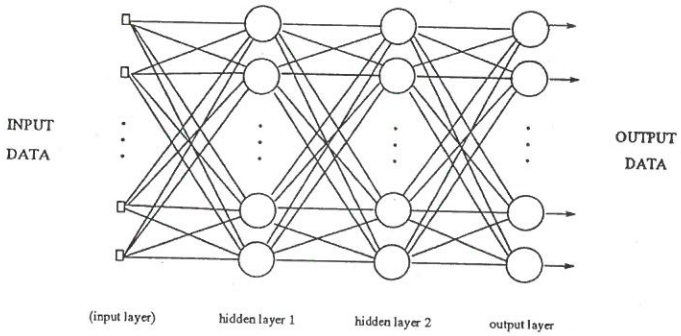


Figure 2.2: *Multilayer feedforward neural network with two hidden layers. One recognizes the input and output layer, which receive and send out signals to the external world. The layers consist of neurons, which are basic processing units that operate on the incoming signals and send out one output signal to the next layer. The incoming signals are weighted with the so-called synaptic weights (the numbers w_{ij} in the McCulloch-Pitts model), summed and sent through a nonlinearity (see below). The 'knowledge' or 'memory' of the neural network sits in the interconnection weights. 'Training' a neural network means that in one way or another, one is fixing the interconnection weights (using a training algorithm) in such a way that the observed input-output patterns are matched as closely as possible.*

2.3.2 Training and learning

Among the learning rules for ANNs one has basically two different modes: *supervised* and *unsupervised* learning (see Fig.2.3) (Zurada (1992)). In supervised learning an external 'teacher' gives information to the network about the correspondance of given input patterns to desired output patterns. The network learns by adapting its weights until the actual output of the network is as close as possible to the desired output, according to some distance measure. In unsupervised learning on the other hand there is no external teacher. The network adapts its weights based upon the actual output of the network only. In between those two extremes is reinforcement learning or learning with a critic, which informs the network only about the fact whether the outcome of a trial is good or bad.

Learning rule	Mode
Hebbian	U
Perceptron	S
Backpropagation	S
Self-organizing map	U

Table 2.4: *Some popular learning rules and their learning mode: supervised (S) or unsupervised learning (U).*

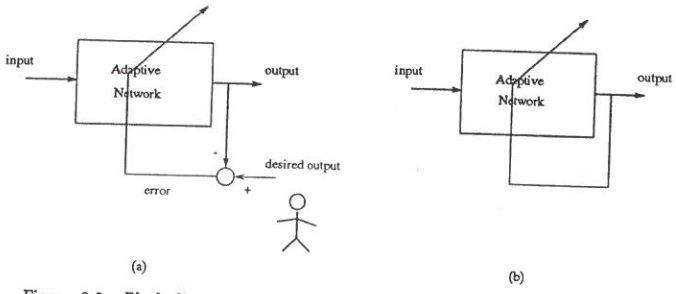


Figure 2.3: *Block diagram that illustrates the basic difference between: (a) supervised learning and (b) unsupervised learning. In supervised learning there is a teacher, somebody who tells the neural net how certain given inputs have to match certain desired outputs. This information will influence the way in which the synaptic (interconnection) weights are adapted. In unsupervised learning there is no external teacher. Instead, there is a certain feedback mechanism that adapts the weights.*

2.4 Neural models and model-based solutions

In this section, we first treat the necessity of having mathematical models to come up with solutions to engineering problems that are better than their non-model-based counter parts. (At the same time, we point out the important difference between what an engineer considers to be 'modelling' and the point of view of a mathematician/physicist). Next, we briefly announce the next two Chapters, in which we will treat in more or less detail how neural nets can be used in obtaining mathematical models and designing controllers.

2.4.1 Mathematical modelling and model-based control

A dynamic system can conceptually be described as in Fig.2.4, which covers almost all physical, economical, biological, industrial, etc ...systems.

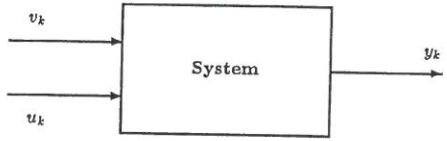


Figure 2.4. *A dynamic system with deterministic inputs u_k, outputs y_k and disturbances v_k (see below). All arrows represent vector signals and k is the discrete time index. The user can control u_k but not v_k. In some applications, either u_k, v_k or w_k can be missing. The measured output signals provide useful information about the unknown system.*

One could distinguish between mental, intuitive or verbal models, or graphically oriented approaches such as graphs and tables, but we will mainly be interested in mathematical models in general, and neural net models in particular. Such models are described as differential (continuous time) or difference (discrete time) equations. They describe the dynamic behavior of a system as a function of time. Mathematical models exist in all scientific disciplines, and, as a matter of fact, form the heart of scientific research itself. They are used for simulation, operator training, analysis, monitoring, fault detection, prediction, optimization, control system designs, quality control, *etc.* Most typically, models are highly useful in those situations in which experimenting with the real system is too expensive (e.g. involves too many people), too dangerous (nuclear plants), too difficult or merely impossible (experimenting with satellites in deep space). Last but not least, mathematical models are used for *control* and *feedback*, which, by the way, is

one of the major engineering inventions.
The underlying motivation of our work is the claim that model-based solutions are superior in performance and robustness compared to plain heuristic approaches. This fundamental insight is not new, and has been at the heart of modern control theory ever since the path breaking work of Kalman at the beginning of the sixties.
What is of interest to us here, is how such a mathematical model can be obtained in the first place using neural nets (Chapter 3) and then, in Chapter 4, how a controller (based on neural nets) can be designed and its stability be guaranteed.
It is only the last decade or so that modeling of industrial processes and the design of model-based controllers has become a serious mathematical engineering discipline, in which we hope to have made some important contributions.

Basically, there are two main roads to construct a mathematical model of a dynamic system. Physicists will be interested in models (physical laws) that carefully explain the underlying essential mechanisms of observed phenomena and that are not *falsified* by the available experiments. The necessary mathematical equipment is that of *partial differential equations*. This is the analytic approach, which rigorously develops the model from *first principles*.
For engineers however, this framework is often much too involved to be really *useful*. The reason is that engineers are not really interested in the exact model *as such*, but more in the potential engineering applications of models. In this perspective, a mathematical model is only one step in the global design of a system. The quality of a model is dictated by the ultimate goal it serves. *Model uncertainty is allowed as long as the robustness of the overall system is ensured.* Engineers -in contrast with mathematical physicists- are prepared to *trade-off model complexity versus accuracy*. A complex model will lead to a complex design, while a simplistic model will deteriorate overall performance and robustness of the final implementation. As an example, the best model for simulation (for instance a set of partial differential equations which accurately models the system's behavior) is not the best one for control, because, as a generic property of control system design, the complexity of the controller and the degree of difficulty associated with its implementation, are proportional to the model complexity.

Therefore engineers will typically use *system identification* techniques to build their models. This is the field of modeling dynamical systems from experimental data: Experiments are performed on a system, a certain parameterized model class is predefined by the user (for instance a neural network) and suitable numerical values are assigned to the parameters (the interconnection weights) so as to fit as closely as possible the recorded data. In this sense, system identification is the dynamic extension of curve fitting. Finally there is a validation step, in which the model is tried out on experimental data which were not used in the system identification experiment.

In Chapter 3, we describe an industrial process which perfectly illustrates the fundamentally different point of view between the two modeling approaches. The glass-tube manufacturing process described there could in principle be characterized completely using the laws of physics (in this case the laws that govern the behavior of solidifying glass). But, not only would this be a formidable task -if practically possible at all-, but even if there was such a model, it would be impossible to derive an appropriate control action to regulate the system, because of the complexity of the model. However, in Chapter 3, it will be shown how a relatively simple neural state space model beats other models obtained by e.g. linear techniques.

The message is that system identification provides a meaningful engineering alternative to physical modeling. Compared to models obtained from physics, system identification models have a limited validity and working range and in some cases have no direct physical meaning. But, they are relatively easy to obtain and use and even more importantly, these models are simple enough to make model-based control system design mathematically (and also practically) tractable. Of course, there are still problems such as the choice of an appropriate model structure, the fact that many systems are time-varying and often largely underestimated measurement problems (appropriate sensors, sampling times, filters, outlier detection, etc...).
Let us conclude this section by saying that system identification and control system design, being typical engineering disciplines, borrow many of their concepts and techniques from other mathematical and engineering fields such as optimization, numerical analysis, linear algebra, complex function theory, statistics, sensors and physical devices, experimental design, software engineering, etc ... and therefore they form in many respects an interdisciplinary activity. This is illustrated in Fig.2.5.

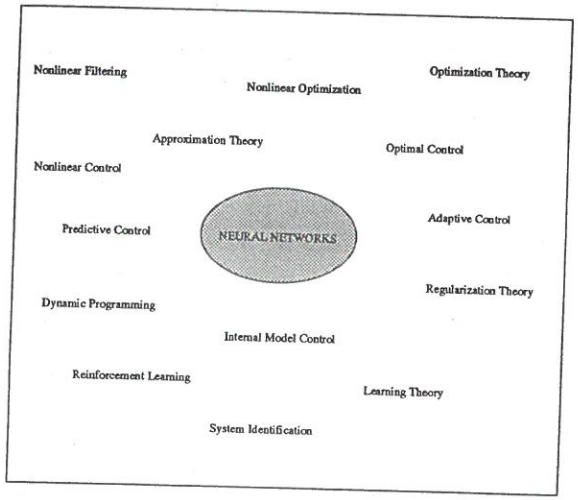


Figure 2.5: *The many mathematical engineering disciplines that are related to neural networks in modelling and control. Sometimes, it is far too easily assumed that neural nets could offer magic solutions to difficult problems. One should be careful with this type of 'hype' talk. Instead, we still know relatively little about how they really work. In our research, we have been trying to use results from other well-established mathematical engineering fields to analyse and design neural net models and controllers. These techniques and insights will be extensively discussed in Chapters 3 and 4.*

2.4.2 Neural networks for modelling

The fact that ANNs are nonlinear makes them more powerful compared to classical linear models for modelling complex behaviour, e.g. systems operating around many equilibria, chaotic behaviour, hysteresis effects, saturation effects etc. Neural nets are used here as black box models in order to represent the behaviour. The models are learned based on a training set of input-output data, measured on a plant (Fig.2.6). The difference between the actual outputs of the model and the desired output values of the training set is minimized. This procedure will be discussed in Chapter 3 in more detail.

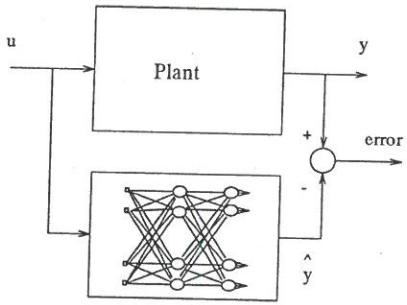


Figure 2.6: *Nonlinear system identification using neural networks. Inputs are applied to the real plant as well as to the neural net model. The outputs of the real plant are then compared to the ones generated by the model. Based on some measure that quantifies the magnitude of the error (for instance the sum of squares of the errors over a certain time horizon), the interconnection (synaptic) weights of the neural net are then adapted/modified in order to minimize the deviation between the real and modelled outputs.*

2.4.3 Neural nets for control

In control applications neural nets are used in order to parametrize controllers. Such neural controllers can be either modelbased or not. This is illustrated by the well known classics in neural control of 'broomstick balancing' Barto *et al.* (1983) and 'backing up a truck' (Nguyen & Widrow (1990)). In Barto *et al.* (1983) the broomstick balancing problem was solved by means of reinforcement learning. A critic rewards or punishes the neural controller in order to stabilize the pole through a number of trials without having a model for the plant. The example of backing up a truck on the other hand is model based. First a neural net emulator is derived for the truck dynamics. Then a controller is learned based upon this emulator model, which is finally applied to the real system. After these early results in neural control, many methods emerged such as neural optimal control, neural adaptive control, internal model control and predictive control, reinforcement learning, feedback linearization etc. (Miller *et al.* (1990), Parisini & Zoppoli (1991), Saerens *et al.* (1993), Hunt *et al.* (1992), Sutton *et al.* (1992), Suykens *et al.* (1994a), Sanner & Slotine (1992), Liu & Chen (1993)). Some control schemes are presented in Fig.2.7.

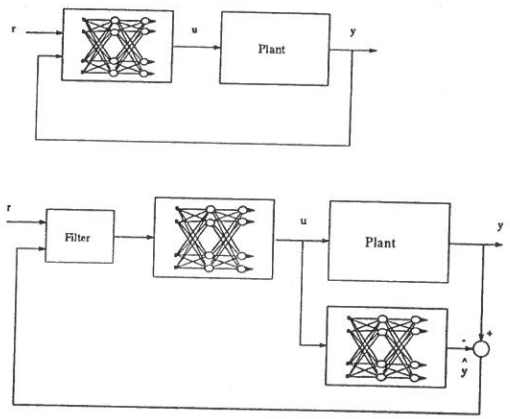


Figure 2.7: *Some neural control schemes: (a) neural adaptive control scheme, the adaptive control scheme can be either indirect (controller based upon an emulator for the plant) or direct (without emulator); (b) internal model control scheme.*

2.4.4 Why are neural nets so useful in modelling and control

The use of multilayer feedforward neural nets for modelling and control is motivated by the following list of attractive features (Hunt *et al.* (1992)):

1. ANNs are universal approximators:
   It is mathematically proven that any continuous nonlinear function can be approximated arbitrarily well over a compact interval by a multilayer neural network that consists of one or more hidden layers.
2. Parallel distributed processing:
   The network has a highly parallel structure and consists of many processing elements with a very simple structure, which is interesting from the viewpoint of implementation.
3. Hardware implementation:
   Dedicated hardware is possible, resulting in additional speed.
4. Learning and adaptation:
   The intelligence of neural networks comes from their generalization ability with respect to fresh, unknown data. On-line adaptation of the weights is possible.
5. Multivariable systems:
   ANNs have naturally many inputs and outputs, which makes it easy to model multivariable systems.

Although neural networks have potentially enormous abilities for modelling and control, the enthusiasm is a little bit tempered by a number of weak points

1. *Local optima:*
   Training the interconnection weights corresponds to solving nonlinear optimization problems in the unknown weights of the model or the controller. These problems have many local optima. From the viewpoint of optimization theory algorithms like backpropagation are local optimization schemes. Nevertheless in many applications it is not necessarily needed to find the global optimum. In cases where local optima form a problem, global optimization algorithms can be used such as e.g. simulated annealing (e.g. Gelfand & Mitter (1993), Szu (1987), Kirkpatrick *et al.* (1983)), genetic algorithms (e.g. Goldberg (1989)) or the recently introduced Fokker-Planck learning machine (Suykens & Vandewalle (1994g)). These algorithms have a higher probability to end up in the global optimum. The main drawback is that they are quite slow.

2. Number of hidden neurons:
   Given a training set of data, the generalization ability of the network after learning (which highly determines its intelligence) depends upon the chosen complexity of the network (the number of hidden neurons). If this number is chosen too high, one may obtain a neural network that will just memorize the presented input-output patterns of the training set, but the generalization ability will degrade. Although there already exist some first theoretical results and some rules of thumb about the choice of this complexity, much work remains to be done on this topic.

3. *Stability criteria:*
   The dynamical models and control systems that contain neural networks can be highly nonlinear. Hence a good understanding of input-output stability and internal stability is needed, because updating of interconnection weights in such systems may lead to instabilities, which might even be dangerous for certain systems. In Chapter 4 we will describe (sufficient) stability criteria for neural control systems, which was an open problem up till now.

Chapter 3

Modelling and prediction by neural networks

Ceci n'est pas une pipe. René Magritte.
A model is not the real system.
(...But it's good when it is close enough...)

In this Chapter neural network architectures and learning rules will be discussed for nonlinear system identification and nonlinear prediction, together with examples. It will be explained in what sense we have contributed to this area.
In Section 3.1., we analyze a very important feature of neural nets, which is there universal approximation property. In Section 3.2., we show how linear models for multivariable system can be the major inspiration to come up with so-called *neural state space models*. In Section 3.3., the nonlinear functions that appear in nonlinear state space models are replaced by feedforward neural nets. Finally, in Section 3.4., we discuss some real life examples. The most important one there is one to which we have contributed ourselves, namely the identification of the input-output characteristics part of a Philips glass oven.

3.1 Feedforward neural networks are universal approximators

Two types of feedforward networks will be considered here: the multilayer feedforward neural network and the radial basis function network

1. Multilayer feedforward neural networks:
The multilayer feedforward neural network shown in Fig.2.2 is described in matrix-

vector notation as

$$y = W\,\sigma(V_2\,\sigma(V_1\,x + \beta_1) + \beta_2).$$

Elementwise one has

$$y_i = \sum_{r=1}^{n_{h_2}} w_{ir}\,\sigma\Big(\sum_{s=1}^{n_{h_1}} v_{rs}^{(2)}\,\sigma\Big(\sum_{j=1}^{m} v_{sj}^{(1)} x_j + \beta_s^{(1)}\Big) + \beta_r^{(2)}\Big),\ i = 1, \ldots, n.$$

The input of the neural net is $x \in \mathbb{R}^m$, the output is $y \in \mathbb{R}^n$. The parameters of the neural network consist of the interconnection matrices $W \in \mathbb{R}^{n \times n_{h_2}}$, $V_2 \in \mathbb{R}^{n_{h_2} \times n_{h_1}}$, $V_1 \in \mathbb{R}^{n_{h_1} \times m}$ and the bias vectors $\beta_2 \in \mathbb{R}^{n_{h_2}}$, $\beta_1 \in \mathbb{R}^{n_{h_1}}$, where n_{h_1} and n_{h_2} are the number of hidden neurons of the first and second hidden layer respectively. A good choice for the activation function $\sigma(.)$ is the hyperbolic tangent function tanh(.) (see Fig.3.1), which is applied to each of the elements of its (vector) argument. In many cases only one hidden layer is used instead of two, because this is sufficient in order to approximate any continuous nonlinear function.

2. Radial basis function networks:
These networks are described as

$$y = \sum_{i=1}^{n_h} w_i\,h(\|x - c_i\|).$$

They consist of one hidden layer. The parameters of the network are the n_h centers $c_i \in \mathbb{R}^m$, where n_h is the number of hidden neurons and the vector of output weights $w \in \mathbb{R}^{n_h}$. An important difference with the previous architecture is in the choice of the activation function $h(.)$, which is in most cases a scalar Gaussian function of several parameters (see Fig.3.1).

For each of these architectures there exist general approximation Theorems and specialized learning algorithms.

The history of universal approximation Theorems for neural networks starts in fact in the year 1900, when Hilbert formulated a list of 23 challenging mathematical problems for the century to come. The famous 13th problem is the conjecture that there are analytic functions of three variables which cannot be represented as a finite superposition of continuous functions of only two variables. This conjecture was refuted by Kolmogorov and Arnold in 1957 who proved that any continuous function $f(x_1, \ldots, x_n)$ defined on the hypercube $[0,1]^n$ with $n \geq 2$ can be represented in the form

$$f(x) = \sum_{j=1}^{2n+1} \xi_j\Big(\sum_{i=1}^{n} \phi_{ij}(x_i)\Big)$$

where ξ_j, ϕ_{ij} are continuous functions and ϕ_{ij} is monotone. In 1965 Sprecher refined Kolmogorov's Theorem. The link with neural networks was first revealed by Hecht-Nielsen in 1987. More refined Theorems in Cybenko (1989), Funahashi (1989), Hornik *et al.* (1989) and Leshno *et al.* (1993) show that any continuous nonlinear function can be approximated arbitrarily well on a compact interval by a multilayer feedforward neural net with one or more hidden layers. Moreover it has been shown in Barron (1993) that neural networks need fewer parameters in the approximation than polynomial expansions. For radial basis function networks a universal approximation Theorem was given in Park & Sandberg (1991). Hence the approximation Theorems serve as a solid basis for the use of ANNs as parametrizations in modelling and control. However the Theorems are not constructive in the sense that they do not come up with a learning algorithm for the interconnection weights.

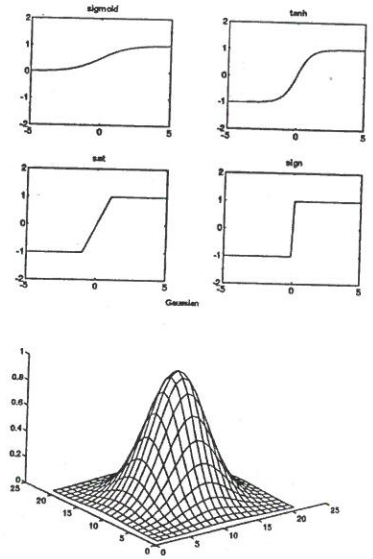


Figure 3.1: *Activation functions for feedforward neural networks: Top left is shown the 'classical' sigmoid, which is a smooth differentiable monotonically increasing function, ranging from 0 to 1. Right of it is shown the hyperbolic tangent function* tanh(.) *which is used in many cases. It ranges from -1 to 1 and is differentiable as well (differentiability is important for certain training algorithms!); On the second row left and right we see two idealizations of the two proceeding graphs. The saturation characteristic to the left still has a slope around 0, while the sign function depicted to the right maps negative numbers to -1 and positive ones to +1. (b) Gaussian activation function for radial basis function networks for a 2-dimensional input. When we would intersect this graph by a plane perpendicular to the horizontal plane, we would always see the characteristic of the Gaussian probability distribution, hence the name. Generalizations for vectors in more than 2 dimensions are perfectly well possible.*

3.2 From linear to nonlinear dynamical systems

Many models in the theory of *linear* system identification can be considered as special cases of the following generalized discrete time model structure

$$A(q)y_k = \frac{B(q)}{F(q)}u_k + \frac{C(q)}{D(q)}e_k \ ,$$

where $A(q)$,$B(q)$,$C(q)$,$D(q)$,$F(q)$ are (matrix) polynomials in the delay operator q ($qu_k = u_{k-1}$). (see Ljung (1987)). The input and output signal is u_k and y_k respectively. The signal e_k is a white noise exogenous input. The following commonly used models can be considered as special cases:

Used polynomials	Name of model structure
B	FIR (Finite Impulse Response)
AB	ARX
ABC	ARMAX
AC	ARMA (Auto Regressive Moving Average)
BF	OE (Output Error)
BFCD	BJ (Box-Jenkins)

where the models then are the following:

ARX : $A(q)y_k = B(q)u_k + e_k$
ARMAX : $A(q)y_k = B(q)u_k + C(q)e_k$
OE : $y_k = \frac{B(q)}{F(q)}u_k + e_k$
BJ : $y_k = \frac{B(q)}{F(q)}u_k + \frac{C(q)}{D(q)}e_k$

The step towards nonlinear modelling is made as follows:

NARX (Nonlinear ARX):

$$y_k = f(y_{k-1}, y_{k-2}, ..., y_{k-n_y}, u_{k-1}, u_{k-2}, ..., u_{k-n_u}) + e_k$$

NARMAX (Nonlinear ARMAX):

$$y_k = f(y_{k-1}, y_{k-2}, ..., y_{k-n_y}, u_{k-1}, u_{k-2}, ..., u_{k-n_u}, e_{k-1}, e_{k-2}, ..., e_{k-n_e}) + e_k$$

where n_y, n_u, n_e are the lags for the output, input and noise signal. The latter are all *input-output* models.

On the other hand one can also consider nonlinear *state space* models of the form

$$\begin{cases} x_{k+1} &= f(x_k, u_k, \varphi_k) \\ y_k &= g(x_k, u_k, \psi_k) \end{cases}$$

with x_k the state vector and φ_k, ψ_k are respectively process noise and measurement noise, assumed to be zero mean white Gaussian with covariance matrices

$$E\{\begin{bmatrix} \varphi_k \\ \psi_k \end{bmatrix} [\varphi_s^t \ \psi_s^t]\} = \begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \delta_{ks}.$$

A special case is the well-known linear system in state space form

$$\begin{cases} x_{k+1} &= Ax_k + Bu_k + \varphi_k \\ y_k &= Cx_k + Du_k + \psi_k, \end{cases}$$

to which we have made some contributions in another context (De Moor 1994a, De Moor 1994b, Van Overschee and De Moor, 1993, 1994) .

Neural networks are used then in the nonlinear dynamical model structures as parametrizations for the nonlinear functions. This makes sense because the neural nets are universal approximators. Hence system identification by means of neural networks aims at modelling general nonlinear systems, which is certainly more ambitious than estimating approximate linear models. It becomes possible then to model complex nonlinear behaviour such as chaos, hysteresis or saturation effects or combinations of several nonlinear phenomena.

3.3 Parametrization by ANNs and paradigms of learning

In order to fix the ideas let us consider a NARX model. A parametrization is denoted as

$$y_k = f(y_{k-1}, y_{k-2}, ..., y_{k-n_y}, u_{k-1}, u_{k-2}, ..., u_{k-n_u}; \theta) \ ,$$

where θ is a vector of unknown parameters. In the case of a parametrization by a multilayer feedforward neural network with one hidden layer this becomes

$$y_k = w^T \tanh(Vx_{k-1} + \beta) \ ,$$

where the input vector x_{k-1} of the neural network consists of y_{k-1} ,..., y_{k-n_y} , u_{k-1} ,..., u_{k-n_u} and θ is the parameter vector that contains the elements of w, V and β. For the

training of dynamical systems containing neural networks from input-output data, there exist two possible ways of looking at the problem: either one considers series-parallel models or parallel models (see Narendra & Parthasarathy (1990)). In series-parallel models the true outputs are used as input of the neural network model

$$\hat{y}_k = f(y_{k-1}, ..., y_{k-n_y}, u_{k-1}, ..., u_{k-n_u}; \theta) ,$$

whereas parallel models look like

$$\hat{y}_k = f(\hat{y}_{k-1}, ..., \hat{y}_{k-n_y}, u_{k-1}, ..., u_{k-n_u}; \theta) ,$$

where $\hat{y}_k$ is the output of the neural network at time k.

Given N data $\{u_k, y_k\}_{k=1}^N$, the objective in prediction error learning algorithms is then to minimize a cost function like

$$\min_\theta J(\theta) = \frac{1}{2N} \sum_{k=1}^{N} \epsilon_k^2 ,$$

where $\epsilon_k = y_k - \hat{y}_k(\theta)$ is the prediction error. Gradient based optimization techniques are used then to minimize the cost function. The difference between series-parallel and parallel mode is then in the way the gradient of the cost function is derived:

1. *Series-parallel mode:*
   In this case the expression of the gradient can be obtained by means of the *back-propagation* algorithm (for details and explanations of notations, we refer e.g. to Rumelhart *et al.* (1986), Werbos (1990)). A paradigm for adaptation of the weights is the generalized delta rule

$$\begin{cases} \Delta w_{ij}^l &= \eta \, \delta_{i,k}^l \, x_{j,k}^{l-1} \\ \delta_{i,k}^L &= (x_{i,k}^d - x_{i,k}^L) \, \sigma'(\xi_{i,k}^L) \\ \delta_{i,k}^l &= (\sum_{r=1}^{N_{l+1}} \delta_{r,k}^{l+1} w_{ri}^{l+1}) \, \sigma'(\xi_{i,k}^l), \qquad l = 1, ..., L-1 \end{cases}$$

   where the neural network is described as

$$x_{i,k}^{l+1} = \sigma(\xi_{i,k}^{l+1}), \qquad \xi_{i,k}^{l+1} = \sum_{j=1}^{N_l} w_{ij}^{l+1} x_{j,k}^l, \qquad l = 1, ..., L \ (L \text{ layers}).$$

   Here l is the layer index of the neural network ($l = 1, .., L$) and N_l is the number of neurons at layer l. w_{ij}^l denotes the ij-th neuron at layer l, η is the learning rate. More advanced learning algorithms are described in Chen *et al.* (1990a,b) and Billings *et al.* (1992).

2. *Parallel mode:*
   The original backpropagation algorithm cannot be applied. Instead *dynamic back-propagation* (or sensitivity method) must be used, which is a procedure for calculating gradients in dynamical systems, containing neural networks, introduced in Narendra & Parthasarathy (1990)(1991).

From the viewpoint of optimization theory the backpropagation algorithm without momentum term is a steepest descent algorithm, which is slow. Faster algorithms are quasi-Newton methods, where curvature information of the surface is built up during optimization based on first order gradient information only, or conjugate gradient algorithms (van der Smagt (1994), Møller (1993), Fletcher (1987), Gill *et al.* (1981)). Also global optimization techniques can be applied such as simulated annealing (Gelfand & Mitter (1993), Szu (1987)) and genetic algorithms (Goldberg (1989)). Besides off-line estimation algorithms also on-line algorithms exist such as recursive prediction error algorithms (Chen *et al.* (1990a,b)) and extended Kalman filtering (Shah *et al.* (1992)). The quality of the obtained result will depend on the number of hidden neurons that is chosen in the architecture. A compromise has to be made here between fitting error and generalization error. As suggested by Akaike's information criterion one has to look for the minimal fitting error with the model as simple as possible (in terms of number of parameters) (Reed (1993), Sjöberg & Ljung (1992)). Hence there must be enough neurons in order to model the underlying nonlinear dynamics of the system, but not too much in order to avoid overfitting and overparametrization. A procedure which is often applied is to stop training once the error on the test data (data that are not used for the identification) starts increasing. Furthermore the obtained models have to be validated: higher order correlations test are discussed in Billings *et al.* (1992).

In our work parametrizations of nonlinear state space models by neural networks were investigated that take into account process noise and measurement noise (Suykens *et al.* (1994b)). A predictor of the form

$$\begin{cases} \hat{x}_{k+1} &= f(\hat{x}_k, u_k; \theta) + K(\theta)\epsilon_k \ ; \ \hat{x}_0 = x_0 \text{ given} \\ y_k &= g(\hat{x}_k, u_k; \theta) + \epsilon_k . \end{cases}$$

where K is the Kalman gain leads to so called neural state space models

$$\begin{cases} \hat{x}_{k+1} &= W_{AB} \tanh(V_A \hat{x}_k + V_B u_k + \beta_{AB}) + K\epsilon_k \\ y_k &= W_{CD} \tanh(V_C \hat{x}_k + V_D u_k + \beta_{CD}) + \epsilon_k \end{cases}$$

This predictor makes use of a direct parametrization of the Kalman gain K instead of exploiting a complicated dependence through a Riccati equation as in extended Kalman filtering (see Fig.3.2) and is quite similar to the well-known Kalman filter for linear systems (Ljung (1979))

$$\begin{cases} \hat{x}_{k+1} &= A\hat{x}_k + Bu_k + K\epsilon_k \\ y_k &= C\hat{x}_k + Du_k + \epsilon_k . \end{cases}$$

The gradient of the cost function is also calculated by means of the sensitivity method (Narendra & Parthasarathy (1991)). The gradient is generated as the output of another dynamical system which is related to the predictor. Hence a single augmented system can be considered that consists of the predictor and its gradient system. At a given iteration step the augmented system has to be simulated as many times as there are elements in the parameter vector θ (This makes the algorithm attractive for parallel training because these simulations can be distributed over the available number of processors).

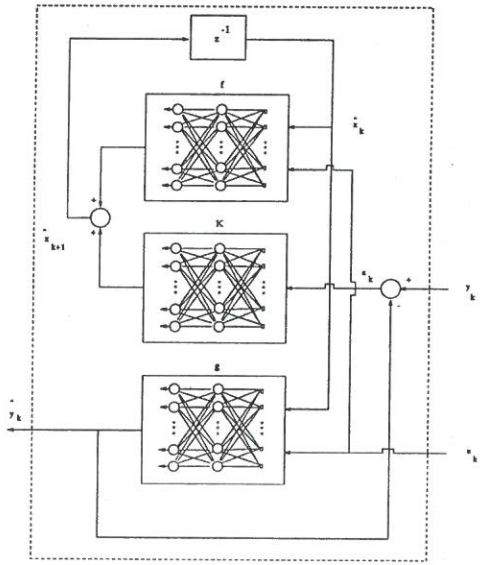


Figure 3.2: *Neural state space model: a filter introduced in Suykens* et al. *(1994b). The filter takes into account process noise and measurement noise and makes use of a direct parametrization of a Kalman gain. Efficient parallel training algorithms are developed based upon Narendra's dynamic backpropagation procedure. The functions f and g are parametrized by a feedforward neural net as is the Kalman gain K. Input and output measurements are processed through these neural networks and result in state vectors and predictions of the outputs.*

In all cases where multilayer feedforward neural nets are used as parametrizations one ends up with nonlinear optimization problems with many local optima. This problem is more or less avoided by the use of radial basis function networks. In a series-parallel model, the learning of the output weights and the centers of the Gaussian activation functions can be treated separately then in a first stage. The centers are placed first such that they cover the input space well. Once this is done the parametrization is linear and in the output weights only (Chen *et al.* (1991)). This means that linear regression can be applied in order to find the output weights. Furthermore radial basis function networks have a deep mathematical foundation in regularization theory (Poggio & Girosi (1990)). It can be shown that they are the optimal solution to the variational problem

$$\min_f H[f] = \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \|Pf\|^2$$

for a specific choice of constraint operator P (smoothness constraint). In this problem one looks for the best function f in order to approximate a set of data $S = \{(x_i, y_i) \in \mathbb{R}^n \times \mathbb{R} | i = 1, ..., N\}$. The theory is closely related to splines.

3.4 Real life examples

3.4.1 Neural networks and complex systems

In order to give some idea of the capabilities of neural networks for modelling and prediction we will present here some real life examples

- *The Santa Fe time series prediction contest:*
  From august 1991 onward, a set of time series was made generally available at the Santa Fe institute (Weigend & Gershenfeld (1993)). Participants in the competition were asked to submit forecasts of continuation of the data sets (that were withheld). The following data sets were selected

  1. A clean physics laboratory experiment with a NH_3 laser (low-dimensional deterministic chaos)
  2. Computer generated series (high-dimensional deterministic chaos)
  3. Currency exchange rate data (Swiss Franc - US Dollar)
  4. Astrophysical data from a variable white dwarf star
  5. Physiological data from a patient with sleep apnea
  6. J.S. Bach's last fugue from *Die Kunst der Fuge*

The results of the competition were presented at the NATO *Advanced Research Workshop*, spring 1992. The winner of the competition was a neural network model. On all data sets best results were obtained by neural network models. Also more classical methods such as a Wiener filter were applied to the data. However we have to mention that not *all* neural network models produced these excellent results, which shows that the right expertise is needed in order to produce good results (selection of a good architecture, learning rule and choice of complexity).

- *Load forecasting in Electric Power systems:*
  In this example the data consist of hourly temperature and load for the Seattle/Tacoma area in the period of 1 Nov 1988 till 30 Jan 1989 (Park *et al.* (1991)). Peak load, total load and hourly load were predicted based on the temperature by means of neural network models. Improvements up to 2 % were obtained with respect to the currently used techniques.

  At our research group in Leuven similar problems are studied for the simulation and prediction of the Belgian consumption of gas and electricity (project in cooperation with Electrabel). Best results that we obtained so far were by means of the neural state space models described in this memorandum.

- *Modelling of space shuttle main engine:*
  Although a complete nonlinear dynamic model is already available for simulation of the space shuttle main engine, its complexity is too high for use in real time applications. This is the reason why NASA is interested in neural nets, which are massively parallel and can be simulated fast as such, in order to emulate the engine's dynamics (Saravanan *et al.* (1993)). The system is multivariable and consists of 5 inputs, which is the rotary motion of five valves and 4 outputs, which are the chamber pressure, the mixture ratio and the speed of high pressure fuel and oxidizer turbine. The identification with neural nets was successful.

- *F-16 aircraft dynamic modeling:*
  The behaviour of the F-16 aircraft is characterized by significant nonlinearities due to kinematic and inertial couplings and aerodynamic nonlinearities. Both low and high angle of attack dynamics are taken into account in order to have full effect of nonlinearities. Neural networks were able here to emulate the dynamical behaviour in highly nonlinear operation mode (Youssef (1993)).

These are just a few of the many convincing real life applications of neural networks that have been reported in literature. In the following Section we will focuss on a specific example of the modelling of a glass furnace by means of neural nets.

3.4.2 Identification of a glass furnace

The system considered here is a glass feeder of Philips Eindhoven [1], consisting of three inputs (two heaters and one cooler), switched by pseudo-random binary noise and six outputs, which are temperatures in a cross-section of the oven. Although methods in linear system identification have already produced satisfactory results in the past, it is possible to improve on the quality of the models by using neural networks. Neural state space models (see Fig.3.2) were used here as black box models in order to model the input-output behaviour of the multivariable plant in an accurate way. For the training procedure linear identification results were adopted as starting points for neural state space models, a procedure described in Suykens *et al.* (1994b). In Fig.3.3 some of the input-output data are shown together with a prediction for one of the temperatures.

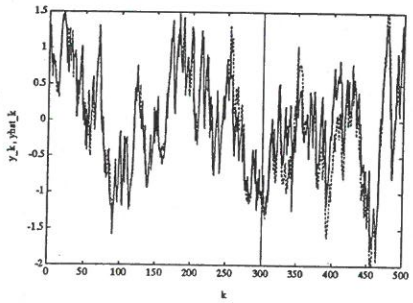


Figure 3.3: *System identification of a glass furnace by means of a neural state space model. Shown in the figure is one of the six temperatures with respect to time, measured at a cross section of the oven, which is influenced by two heating and one cooling input. The data before the vertical line were used for training, behind the vertical line a prediction of the model is shown on the 'fresh' data (data that were not used for identification) (Full line = measured data; dashed line = output of the neural state space model). The results are superior with respect to classical linear identification techniques. One should realize that this industrial system is really difficult if not impossible to model from first (physical) principles. The neural net model does not contain or reveal physical insight. Yet, it manages to identify perfectly well this system, in the sense that simulations and predictions of the behavior are very reliable.*

[1] Research Project with Philips, in which several other European universities were participating as well (Delft, Leicester, Eindhoven). Some other results can be seen in De Moor (1994b) and Van Overschee (1993).

Chapter 4

Neural networks and modern control theory

The control of severely nonlinear systems has for the most part escaped the attention of control theorists and practitioners.
Prof. Bernard Widrow, Stanford University.

In Chapter 2 we have already given a quick glance on methods in neural control. A complete description of all methods and applications in adaptive, optimal and reinforcement neural control would go beyond the scope of this work. Important work in this area is e.g. Miller *et al.* (1990), Parisini & Zoppoli (1991), Saerens *et al.* (1993), Hunt *et al.* (1992), Sutton *et al.* (1992), Sanner & Slotine (1992), Liu & Chen (1993). Instead we prefer to discuss some of our own contributions to this field (Suykens *et al.* (1993b)(1994a,c,d,e,f)) as follows: in Section 4.1. we show how relatively simple neural networks can accomplish rather complicated control tasks. The discussion here is not academic in the sense that the type of problems we have in mind occurs very frequently in industrial situations (switching from one working point of an industrial production process to another one). In Section 4.2. we derive *neural state space models* for identification and control and we describe their combination to come up with model-based neural controllers. In Section 4.3. we describe a new and original framework, which we have called NL_q-theory, which allows to analyse robust stability and robustness of other performances within the context of neural networks, making use of some well-understood techniques from convex optimization. Learning within this framework is described in Section 4.4. while in Section 4.5. we point out how these new framework has good potentials to provide a unifying insight in the interaction between systems, control and neural nets.

4.1 Simple neural nets that accomplish complex control tasks

4.1.1 The inverted pendulum

In the broomstick balancing problem (Barto *et al.* (1983)) it has been illustrated that neural controllers are able to learn and perform stabilization tasks. However it is well known that such problems can be solved by means of classical linear control strategies too. So, is there anything new with neural control? We will illustrate on the inverted pendulum system that the answer is definitely, yes (see Suykens *et al.* (1994a)). This can be understood as follows. The inverted pendulum system is described by a nonlinear state space model

$$\dot{x} = f(x) + g(x)u$$

where the state vector is $x = [x_p; \dot{x}_p; \theta; \dot{\theta}]$ with x_p the position of the cart and θ the angle of the pole and the input u is the force applied to the cart. Suppose we propose a static full state feedback controller, parametrized by a feedforward neural network with one hidden layer

$$u = \alpha \tanh(w^T \tanh(Vx))$$

and we would consider the control task of locally stabilizing the pole at its upright position ($x = 0$), then a sufficient condition for local stability at that point is given by

$$k^T = \alpha w^T V$$

where the vector k is related to a linear stabilizing controller $u = k^T x$ that is designed for the nonlinear system, linearized at $x = 0$. From the latter expression it becomes clear some degree of *freedom* is left for the choice of stabilizing interconnection weights w and V. This means that we can do more than just local stabilization around one equilibrium point. In Suykens *et al.* (1994a) it is shown how the extra degrees of freedom can be exploited in order to realize transitions between equilibrium points. For the inverted pendulum system this means that we can swing up the pole and locally stabilize it in its upright position. This is a typical example of non-local control, which cannot be solved by means of one single linear controller. In the linear controller design case one should have several linear models around many operating points together with a complicated switching strategy (e.g. fuzzy control) to switch between several linear controllers in order to solve the swinging up problem. In the neural control case one has one simple architecture which can be learned in an efficient way.

The way the controller is trained is by formulating an optimal control problem

$$\min_{\alpha,w,V} C(\alpha, w, V, f, g, x(0), \eta, \zeta, T)$$

where the cost function C is defined as

$$C = \eta(x(T)) + \int_{t=0}^{T} \zeta(x(t), u(t))dt$$

with constraints

1. Local stability at the target point
2. Closed loop system dynamics

Such problems can then be formulated as nonlinear optimization problems in the interconnection weights. The problem is solved by fast simulation of the closed loop system and calculation of the cost function from the simulation result. Both local optimization schemes such as quasi-Newton methods (Suykens *et al.* (1994a)) and global ones such as genetic algorithms (Thierens *et al.* (1993)) have been applied with success to this optimization problem. Best results were obtained by means of a Fokker-Planck learning machine recently proposed in Suykens & Vandewalle (1994g).
So far we have only considered the case of static state feedback. It is also shown that the swinging up problem can be accomplished by a nonlinear dynamic output feedback controller where only x_p and θ are fed back (instead of x) and the controller is a recurrent neural network. It was also found that the recurrent neural controller could generalize much better towards other initial states $x(0)$ that were proposed after learning (because of its internal memory state), in order to be capable still to accomplish the control task. The control schemes and simulation results for the swinging up problem are shown in Fig.4.1 and Fig.4.2.

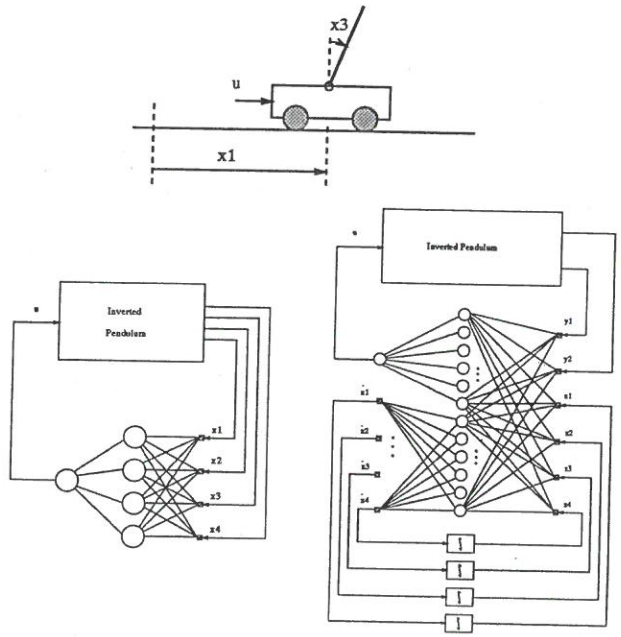


Figure 4.1: *Inverted pendulum system with feedforward and recurrent neural controllers in order to perform the swinging up problem together with local stabilization at the target equilibrium point (upright position). The inverted pendulum is really a little cart on which we have mounted a rigid pole which can move in a vertical plane. The position and velocity of the car can be controlled by applying forces to the right and the left. The ultimate goal is to balance the stick in its upright position. Note that there are two equilibrium positions, the one up which is unstable and the one down, which is stable. The control strategy shown to the left consists of a state feedback controller which manipulates the input (the force applied to the system). The controller to the left is itself a dynamical system (recurrent neural net) using two output measurements and manipulating the input of the system.*

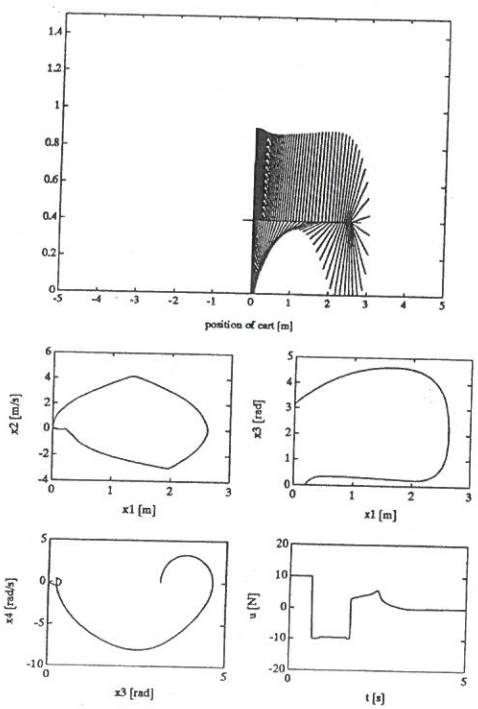


Figure 4.2: *Simulation results of the swinging up problem by means of a feedforward neural controller: The top figure shows the time evolution of the pole. First the car moves to the right and the pole is swinging behind. Then suddenly the car moves to the left, an action which swings the pole up. The position and velocity are then regulated in such a way that the pole ends up in the upright position as shown, at which it is then equilibrated by relatively small control actions. The other figures show the behavior of the closed loop system in the 4 dimensional state space and the last figure contains the control signal (the applied force) with respect to time.*

4.1.2 A double inverted pendulum

An even more complicated problem is the swinging up problem of a double inverted pendulum. A similar control strategy and learning algorithm was applied here (Suykens *et al.* (1993b)). In the static full state feedback case 6 hidden neurons were sufficient to accomplish the extremely complex control task in 6 dimensional state space (Fig.4.3). As a result it is quite surprising at first sight that such simple neural networks can perform very complicated control tasks. An explanation is that by introducing hidden layers in networks additional parameters and degrees of freedom are created. From the study it follows that the linear controller design techniques (such as PID, LQR, H_2, H_∞, etc.) can be used in order to constrain the choice of the interconnection weights of both feedforward and recurrent neural network controllers, in order to impose a suitable behaviour around the target point.

4.1.3 Why is this NOT an academic example ?

One may not think too soon that the single and double inverted pendulums are purely academic exercises. Many large-scale industrial plants are nicely controllable around a working point, at which their behavior is approximately linear. However, it also happens (as a matter of fact, in present day just-in-time production methodologies more and more frequently) that the process needs to switch from one working point to another. Nowadays this is basically done by human operators that have the required *fingerspitzengefühl* to succesfully accomplish this difficult task. However, we claim that neural nets in the near future will be able to do this as well, and even better than a human operator because it will be possible to take into account precise quantitative side conditions, such as minimal energy and speed or accuracy. As we have shown here, one can design nonlinear neural net controllers that switch a system from one working point to another one, by taking into account that, around the desired working point, the controller should behave as a desired linear controller (e.g. a minimum variance controller). In doing so, we combine the best of two worlds within one neural controller: Changing the working point and optimal performance and robustness around the working point (which can be achieved by linear control strategies).

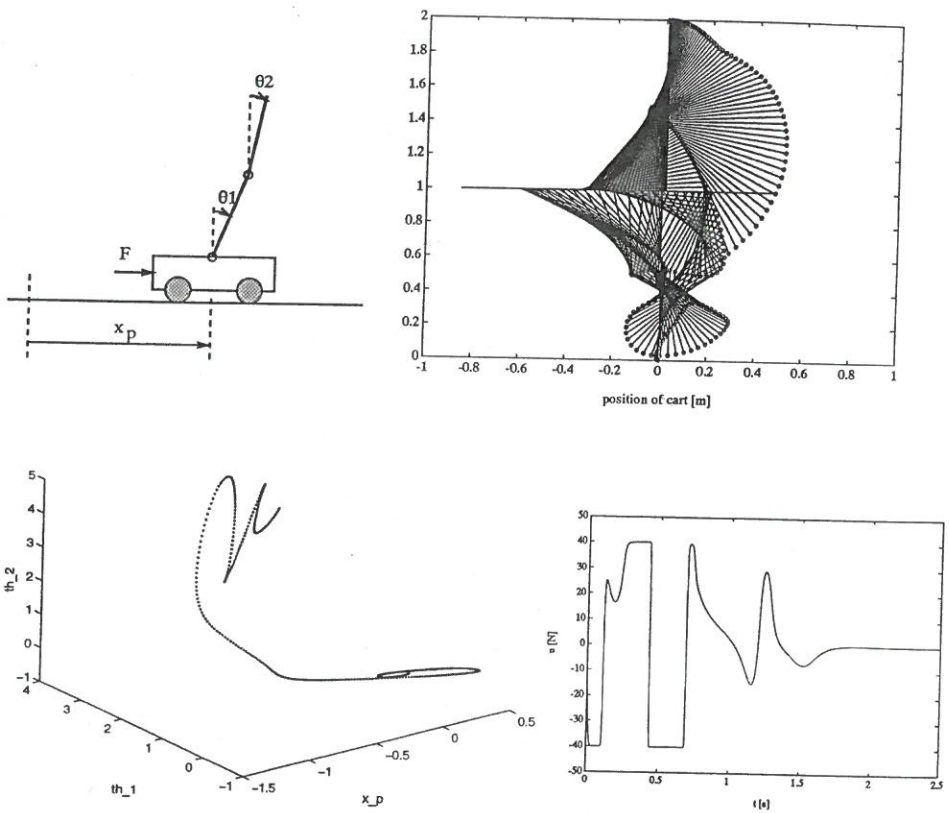


Figure 4.3: *Simulation result of neural control for the swinging up problem of a double inverted pendulum: system, time evolution of the poles, a three-dimensional view of the time trajectory of the vector $(x_p, \theta_1, \theta_2)$ on the six dimensional state space and the control signal with respect to time.*

4.2 Neural state space model framework for identification and control

We already explained the notion of neural state space models in Section 3.3 together with their learning rule. Now it turns out that these architectures in state space form are particularly attractive. We will now explain how stability criteria and conditions for performance and even robust performance can be obtained. This is important in order to train the neural controller based upon a neural network model, because it gives conditions in order to prevent the learning process from becoming unstable. Furthermore it provides us with a better understanding of the working principles, possibilities and limitations of neural networks.

The framework that is considered consists of nonlinear multivariable systems of the form

$$\begin{cases} \hat{x}_{k+1} &= f(\hat{x}_k, u_k, \epsilon_k; \theta_m) \\ y_k &= g(\hat{x}_k, u_k, \epsilon_k; \theta_m) \end{cases}$$

and nonlinear dynamic output feedback controllers of the form

$$\begin{cases} z_{k+1} &= h(z_k, y_k, d_k; \theta_c) \\ u_k &= s(z_k, y_k, d_k; \theta_c). \end{cases}$$

The plant model and controller are neural state space models and neural state space controllers, respectively parametrized by the neural networks θ_m and θ_c. The signals $\hat{x}_k$, z_k, u_k, y_k, d_k, ϵ_k are respectively the internal state of the model and controller, the input and output of the plant, the reference input and a white noise innovations input in order to model the influence of process noise and measurement noise. Examples of neural state space models that that can be considered within this framework are

$$\mathcal{M}_0(\theta_m): \begin{cases} \hat{x}_{k+1} &= A\hat{x}_k + Bu_k + K\epsilon_k \\ y_k &= C\hat{x}_k + Du_k + \epsilon_k \end{cases}$$

$$\mathcal{M}_1(\theta_m): \begin{cases} \hat{x}_{k+1} &= W_{AB}\tanh(V_A\hat{x}_k + V_Bu_k + \beta_{AB}) + K\epsilon_k \\ y_k &= C\hat{x}_k + Du_k + \epsilon_k \end{cases}$$

$$\mathcal{M}_2(\theta_m): \begin{cases} \hat{x}_{k+1} &= W_{AB}\tanh(V_A\hat{x}_k + V_Bu_k + \beta_{AB}) + K\epsilon_k \\ y_k &= W_{CD}\tanh(V_C\hat{x}_k + V_Du_k + \beta_{CD}) + \epsilon_k \end{cases}$$

and neural controllers

$$C_0(\theta_c): \begin{cases} z_{k+1} &= Ez_k + Fy_k + F_2 d_k \\ u_k &= Gz_k + Hy_k + H_2 d_k \end{cases}$$

$$C_1(\theta_c): \begin{cases} z_{k+1} &= Ez_k + Fy_k + F_2 d_k \\ u_k &= \tanh(Gz_k + Hy_k + H_2 d_k) \end{cases}$$

$$C_2(\theta_c): \begin{cases} z_{k+1} &= W_{EF}\tanh(V_E z_k + V_F y_k + V_{F_2} d_k + \beta_{EF}) \\ u_k &= W_{GH}\tanh(V_G z_k + V_H y_k + V_{H_2} d_k + \beta_{GH}) \end{cases}$$

$$C_3(\theta_c): \begin{cases} z_{k+1} &= W_{EF}\tanh(V_E z_k + V_F y_k + V_{F_2} d_k + \beta_{EF}) \\ u_k &= \tanh(W_{GH}\tanh(V_G z_k + V_H y_k + V_{H_2} d_k + \beta_{GH})). \end{cases}$$

More complex parametrizations with two hidden layers are discussed in Suykens *et al.* (1994f). The linear model $\mathcal{M}_0$ (the celebrated Kalman filter !) is interpreted as a simplified form of a neural state space model without a hidden layer. The same for the controller C_0 which is the well-known linear dynamic output feedback law. Recall that by means of $\mathcal{M}_2$ and C_3 any (continuous) nonlinear system and nonlinear dynamic output feedback law can be represented because of the universal approximation ability of neural networks. Hence the framework is very general. The neural control scheme is treated then in the framework of modern control theory (Boyd & Barratt (1991)): here the so-called *standard plant* form is considered (see Fig.4.4). The standard plant is a reorganized scheme with exogenous input w_k (consisting of the reference input and disturbance signals), regulated output e_k (consisting of $d_k - \hat{y}_k$ and possibly also other variables of interest), sensed output y_k and actuator input u_k. An *augmented plant* $\mathcal{S}_i$ is considered with inputs w_k, u_k and output e_k, y_k.

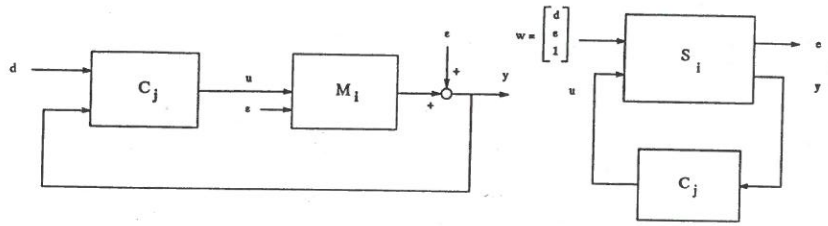


Figure 4.4: *Neural control scheme with neural state space model $\mathcal{M}_i$ and neural state space controller C_j and its representation in standard plant form with augmented plant $\mathcal{S}_i$, according to modern control theory.*

Within the framework neural state space control problems Ξ_j^i are investigated as several possible combinations of models $\mathcal{M}_i$ ($i \in \{0,1,2\}$) with controllers C_j ($j \in \{0,1,2,3\}$). An overview is given in the table below. The problem Ξ_0^0 is a studied in modern *linear* control theory (Boyd & Barratt (1991), Maciejowski (1989)).

	C_0	C_1	C_2	C_3
$\mathcal{M}_0$	Ξ_0^0	Ξ_1^0	-	-
$\mathcal{M}_1$	Ξ_0^1	Ξ_1^1	Ξ_2^1	Ξ_3^1
$\mathcal{M}_2$	Ξ_0^2	Ξ_1^2	Ξ_2^2	Ξ_3^2

This table shows the possible model-based control strategies that can be analysed within the framework we present in this memorandum. The rows represent the several possible models in increasing level of complexity (hidden layers). The columns represent the several possible neural controllers in increasing level of complexity.

4.3 Stability criteria: NL_q theory

The analysis of neural state space control systems is possible by means of so-called NL_q theory, a new theory introduced in Suykens *et al.* (1994f). This unifying framework offers broad perspectives with respect to the following issues:

- It provides a unification of all the possible control and modelling strategies presented in the previous section.
- It will allow to analyse and study the robustness of the closed-loop stability and of all kinds of performance criteria.
- It turns out that several known control strategies for linear and nonlinear systems reduces to special cases of NL_q-theory.

NL_q systems (or shortly NL_qs) are nonlinear systems of the form (related to a standard plant)

$$\begin{cases} p_{k+1} &= \Gamma_1(V_1\Gamma_2(V_2...\Gamma_q(V_q p_k + B_q w_k) + B_{q-1}w_k) + ...) + B_1 w_k) \\ e_k &= \Lambda_1(W_1\Lambda_2(W_2...\Lambda_q(W_q p_k + D_q w_k) + D_{q-1}w_k) + ...) + D_1 w_k) \end{cases}$$

with state vector $p_k \in \mathbb{R}^{n_p}$, input vector $w_k \in \mathbb{R}^{n_w}$ and output vector $e_k \in \mathbb{R}^{n_e}$. Here Γ_i, Λ_i $(i = 1, ..., q)$ are diagonal matrices with diagonal elements γ_j and λ_j depending on p_k and w_k and belonging in the interval $[0, 1]$ for all values of p_k and w_k. An equivalent representation for NL_qs is (see Fig.4.5)

$$\begin{bmatrix} p_{k+1} \\ e_k \end{bmatrix} = \prod_{i=1}^{q} \Omega_i(p_k, w_k) R_i \begin{bmatrix} p_k \\ w_k \end{bmatrix}$$

where Ω_i are diagonal matrices containing Γ_i and Λ_i and R_i is related to the constant matrices V_i, B_i, W_i and D_i. An important property is $\|\Omega_i\| \leq 1$ because $\|\Gamma_i\| \leq 1$ and $\|\Lambda_i\| \leq 1$ for all i.

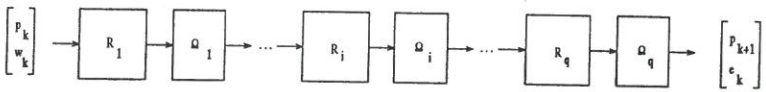


Figure 4.5: *NL_q system: matrices R_i $(i = 1, .., q)$ are constant and matrices Ω_i are diagonal with elements depending upon p_k, w_k but satisfy $\|\Omega_i\| \leq 1$ for all i. Remark the multilayer feature of the NL_q.*

It can be shown that all Ξ_j^i neural state space control problems are special cases of NL_qs. This is obtained by applying the trick of state augmentation. The fact $\|\Omega_i\| \leq 1$ is obeyed because of the choice of the activation function tanh(.) which is a nonlinearity belonging to the sector $[0, 1]$.

It is shown that a sufficient condition for global asymptotic stability of the NL_q (which means here that $p = 0$ is the unique equilibrium point) in the autonomous case (zero external input) is to find diagonal matrices D_i such that

$$\|D_{tot} V_{tot} D_{tot}^{-1}\|_2^q = \beta_D < 1$$

holds where

- $$V_{tot} = \begin{bmatrix} 0 & V_2 & & 0 \\ & 0 & V_3 & \\ & & \ddots & \\ & & & 0 \quad V_q \\ V_1 & & & 0 \end{bmatrix}, V_i \in \mathbb{R}^{n_{h_i} \times n_{h_{i+1}}}, n_{h_1} = n_{h_{q+1}} = n_p$$
- $D_{tot} = \text{diag}\{D_2, D_3, ..., D_q, D_1\}$, $D_i \in \mathbb{R}^{n_{h_i} \times n_{h_i}}$ diagonal matrices with nonzero diagonal elements.

This is mathematically proven by means of a radially unbounded Lyapunov function. In Suykens *et al.* (1994f) 'sharper' criteria together with input-output stability criteria for NL_qs are derived. It is also investigated under what condition the NL_q system is *dissipative*[1], which means that it dissipates energy with respect to a certain supply rate. The stability criteria and conditions for dissipativity are closely related to modern control theories such as H_∞ control theory and μ theory in robust control (Packard & Doyle (1993)). In fact results from these theories are found back as special cases of NL_q theory. For instance, in the case of the linear control system Ξ_0^0 with $\Omega_1 = I$ $(q = 1)$ Theorems of NL_q theory reduce to well-known criteria of the H_∞ control problem with linear dynamic output feedback! Hence NL_q theory can be interpreted as an extension of modern control theories, for treating more complex systems such as interconnected systems or systems with multiple layers, like neural networks. In Section 4.5 we will discuss more on the unifying nature of NL_q theory for problems of neural networks, systems and control.

Up till now we considered NL_qs that are exact in the sense that there are no parametric uncertainties on the matrices V_i, W_i, B_i, D_i of the NL_q. However like it is possible in robust control theory [2] to study the influence of uncertainty upon nominal *linear models*, this is also possible within NL_q theory but upon nominal *nonlinear* models. This is done by considering perturbed NL_qs (see Fig.4.6). Again criteria for internal stability and input-output stability for the perturbed NL_q can be derived (Suykens *et al.*

[1] The notion of dissipativity as introduced by J.C. Willems in general system theory.

[2] In robust control theory one analyses the influence of parametric uncertainties and unmodelled dynamics upon the system's stability and performance or one tries to take into account these uncertainties when synthesizing linear controllers.

(1994d,f)). Furthermore it can be shown that Linear Fractional Transformations that are used for modelling uncertainty in μ theory are a special case of NL_qs for $q = 1$ (see Fig.4.7) and that the so-called 'state space upper bound test' in μ theory is a special case of one of the Theorems derived in NL_q theory!

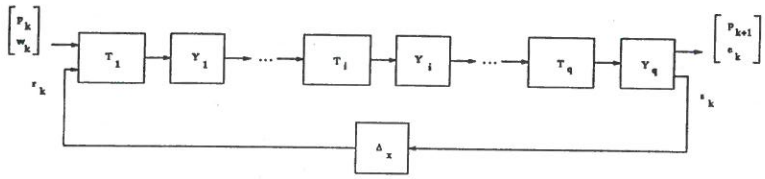


Figure 4.6: *Perturbed NL_q system: the original NL_q is perturbed by means of the feedback interconnection Δ_x in order to model uncertainties upon its interconnection weights and to study the influence of these uncertainties upon its stability and performance.*

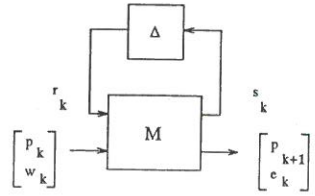


Figure 4.7: *Linear Fractional Transformation with Δ uncertainty block in modern robust control theory. This is a special case of a perturbed NL_q for $q = 1$.*

4.4 Learning within NL_q theory

In the previous Section we have shown how the stability and input-output behaviour of neural state space control systems can be analysed within NL_q theory. However the question remains, given a neural network model, how can we learn a neural controller by taking into account these results from NL_q theory ? The classical learning paradigm for neural controllers would be to apply Narendra's dynamic backpropagation (like this was done for learning of neural networks in parallel mode for system identification) in order to learn to track a specific reference input. However there are some pitfalls then: the learning process may become unstable and it is not guaranteed that a stabilizing neural controller will be found after learning which might be dangerous when other reference inputs are presented to the control system[3]. Using results from NL_q theory as constraint to dynamic backpropagation one might overcome the problem. This can be done as follows (see Suykens *et al.* (1994c,f)). The objective for tracking of a given reference input d_k is

$$\min_{\theta_c} J(\theta_c) = \frac{1}{2N} \sum_{k=1}^{N} \{[d_k - \hat{y}_k(\theta_c)]^T [d_k - \hat{y}_k(\theta_c)] + \lambda . u_k(\theta_c)^T u_k(\theta_c)\}$$

where θ_c is the unknown parameter vector of the neural controller and the parameter λ penalizes the control energy, needed to accomplish the control task. Narendra's sensitivity method can be used in order to compute the gradient of the cost function with respect to the parameters of the neural controller. A condition like

$$\|D_{tot} V_{tot} D_{tot}^{-1}\|_2^q < 1$$

from NL_q theory for ensuring closed loop stability of the control system has to be added then as a constraint to the optimization problem. Important to note here is that the stability criterion can be written as a Linear Matrix Inequality (LMI) [4]

$$V_{tot}^T D_{tot}^2 V_{tot} < D_{tot}^2$$

or otherwise stated $\lambda_{max}(V_{tot}^T D_{tot}^2 V_{tot} - D_{tot}^2) < 0$, where λ_{max} denotes the maximum eigenvalue of the matrix. Like in many problems arising in systems and control LMIs play also here a crucial role for neural control systems[5]. The recent book of Boyd *et al.* (1994) gives an excellent overview on LMIs. The fact that the criteria can be written as LMIs is important from a computational point of view, because finding a feasible point

[3] Up till now this was an often heard objection against neural control (before people were aware of NL_q theory).

[4] LMIs are expressions of the form $M(x) < 0$ with $M = M^T$ with unknown $x \in \mathbb{R}^n$ where M depends affinely on x: $M(x) = M_0 + \sum_{i=1}^{m} x_i M_i$ with $M_0 = M_0^T$, $M_i = M_i^T$.

[5] This affirms again the by now famous words of J.C. Willems: 'The basic importance of the LMI seems to be largely unappreciated. It would be interesting to see whether or not it can be exploited in computational algorithms, for example. See Willems (1971a,b)(1972).'

to LMIs is a convex problem, which means there is only one minimum and this minimum can be found in polynomial time. Efficient interior-point algorithms are available for solving these problems (Nesterov & Nemirovskii (1994)). Hence we finally end up with the following modification of the dynamic backpropagation algorithm in order to ensure closed loop stability

$$\min_{\theta_c, D_{tot}} J(\theta_c) \quad \text{such that} \quad M(\theta_c, D_{tot}) < 0$$

with $M(\theta_c, D_i) = V_{tot}(\theta_c)^T D_{tot}^2 V_{tot}(\theta_c) - D_{tot}^2$. This overall optimization problem is non-convex and non-differentiable. General methods for solving such methods are presented in Polak & Wardi (1982). We can however exploit the fact that for a fixed $\theta_c = \theta_c^0$ the constraint is convex. The modified dynamic backpropagation algorithm consists then of solving two nested optimization problems: the 'inner' convex feasibility problem

$$M(\theta_c^0, D_i) < 0$$

and the 'outer' optimization problem

$$\min_{\theta_c} J(\theta_c) \quad \text{such that} \quad \lambda_{max}[M(\theta_c)] < 0.$$

Of course it is not sure a priori that there exist a feasible point to the LMI. In case there is no feasible point one may relax the condition of global asymptotic stability of the closed loop system to local stability around the target point, which is again an LMI problem.

4.5 Unifications in the theory of neural networks, systems and control

In Section 4.3 we showed that all neural state space control problems can be written as NL_qs and we already mentioned that certain results from H_∞ theory and μ theory in robust control can be considered as special cases of NL_q theory. Besides these results we were able to prove that many more problems in neural networks, systems and control can be represented as special cases of NL_qs (Suykens *et al.* (1994f)). An overview is presented in Table 4.1.

1. *Control:*
   Both problems from classical and modern control theory such as the Lur'e problem (Vidyasagar (1993)) and Linear Fractional Transformations with real diagonal uncertainty block (Packard & Doyle (1993)) are NL_qs with $q = 1$. Also the commonly encountered problem of a linear control system with saturation of the actuator (Boyd & Barratt (1991)) is an NL_1. Neural state space control systems require $q \geq 1$ in general because of their multiple layers.

2. *Neural networks:*
   Many types of neural networks can be represented as NL_qs. The simplest one is the discrete time Hopfield network with synchronous updating which has $q = 1$. Another one is the discrete time Cellular Neural Network (CNN), introduced by Chua & Yang (1988), which has also $q = 1$. The more recently introduced Generalized Cellular Neural Network in Guzelis & Chua (1993) and Harrer (1993) contain several CNNs that are connected in a feedforward, recurrent or cascade way. These are NL_qs which may have $q \geq 1$. Another architecture, called Locally Recurrent Globally Feedforward (LRGF) neural networks, which is recently introduced by Tsoi & Back (1994) has $q = 1$. This architecture has certain advantages in order to model both short term and long term effects in time series, which is a difficult problem. Furthermore the LRGF neural network is in itself already a unification of other architectures !

3. *Signal processing:*
   Digital filters with overflow characteristic (Liu & Michel (1992)) are NL_1s.

Hence NL_qs play a unifying role in the analysis and design of many dynamical systems, that contain nonlinearities belonging to a sector $[0, K]$ $(K > 0)$, arising in many application areas.

NL_q system	q value	Application
Neural state space control systems	$q \geq 1$	modelling and control
Generalized CNNs	$q \geq 1$	signal processing
LFTs with real diagonal Δ block	$q = 1$	control
Lur'e problem	$q = 1$	control
Linear control scheme with saturation input	$q = 1$	control
Digital filters with overflow characteristic	$q = 1$	signal processing
Hopfield network, CNN	$q = 1$	signal processing, optimization
LRGF networks	$q = 1$	modelling

Table 4.1: *The many special cases of NL_qs (a concept introduced in Suykens* et al. *(1994f)) in neural networks, signal processing, modelling and control.*

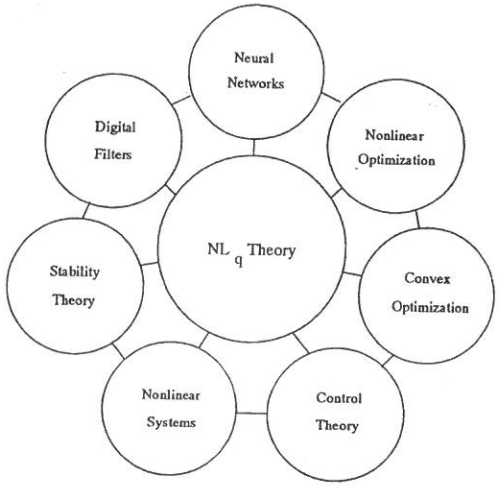


Figure 4.8: *NL_q theory and its relation to other mathematical engineering disciplines. This figure convincingly shows that the analysis and design of neural networks is NOT straightforward, in sharp contrast with the impression that one might get from the literature. In this memorandum, we have used results and facts from all of these disciplines to come up with new techniques for the analysis and design of neural net modelling and control strategies.*

Bibliography

Anderson B.D.O., Moore J.B. (1979). *Optimal filtering*, Prentice-Hall.

Barron A.R. (1993). Universal approximation bounds for superposition of a sigmoidal function, *IEEE Transactions on Information Theory*, Vol.39, No.3, pp.930-945.

Barto A.G., Sutton R.S., Anderson C.W. (1983). Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, No.5, pp.834-846.

Billings S., Jamaluddin H., Chen S. (1992). Properties of neural networks with applications to modelling non-linear dynamical systems, *International Journal of Control*, Vol.55, No.1, pp.193-224.

Boyd S., Barratt C. (1991). *Linear controller design, limits of performance*, Prentice-Hall.

Boyd S., El Ghaoui L., Feron E., Balakrishnan V. (1994). *Linear matrix inequalities in system and control theory*, SIAM (Studies in Applied Mathematics), Vol.15.

Broomhead D.S., Lowe D. (1988). Multivariable functional interpolation and adaptive networks, *Complex Systems*, 2, pp.321-355.

Bryson A.E., Ho Y.C. (1969). *Applied optimal control*, Waltham, MA: Blaisdel.

Casalino G., Parisini T., Zoppoli R. (1993). A neural optimal controller for a real space robot, *World Congress on Neural Networks '93, Portland*, Vol.III, pp.208-211.

Casdagli M. (1989). Nonlinear prediction of chaotic time series, *Physica D*, 35, pp.335-356.

Chakraborty K., Mehrotra K., Mohan C.K., Ranka S. (1992). Forecasting the behaviour of multivariate time series using neural networks, *Neural Networks*, Vol.5, pp.961-970.

Chen S., Billings S., Grant P. (1990a). Nonlinear system identification using neural networks, *International Journal of Control*, Vol.51, No.6, pp. 1191-1214.

Chen S., Cowan C., Billings S., Grant P. (1990b). Parallel recursive prediction error algorithm for training layered neural networks, *International Journal of Control*, Vol.51, No.6, pp.1215-1228.

Chen S., Cowan C., Grant P. (1991). Orthogonal least squares learning algorithm for radial basis function networks, *IEEE Transactions on Neural Networks*, Vol.2, No.2, pp.302-309.

Chua L.O., Yang L. (1988). Cellular neural networks: Theory, *IEEE Transactions on Circuits and Systems*, Vol. 35, No.10, pp.1257-1272.

Cybenko G. (1989). Approximations by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems*, 2, pp.183-192.

De Moor B. (1988). Mathematical concepts and techniques for modelling of static and dynamic systems. PhD Thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium.

De Moor B., Van Overschee P., Suykens J. (1991). Subspace algorithms for system identification and stochastic realization, *Recent Advances in Mathematical Theory of Systems, Control, Networks and Signal Processing, Proc. of the International Symposium MTNS-91, Kobe, Japan*, June 17-21, MITA Press, pp.589-595.

De Moor B., Peter Van Overschee (1994a). *Graphical User Interface Software for System Identification*. ESAT-SISTA Report 1994-06, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium

De Moor B.(1994b). *Numerical algorithms for state space subspace system identification*. Academia Analecta, Klasse der Wetenschappen, Koninklijke Akademie voor Wetenschappen, België.

Desoer C.A., Vidyasagar M. (1975). *Feedback systems: input-output properties*, New York: Academic Press.

Fletcher R. (1987). *Practical methods of optimization*, second edition, Chichester and New York: John Wiley and Sons.

Funahashi K.-I. (1989). On the Approximate Realization of continuous Mappings by Neural Networks, *Neural Networks*, Vol.2, pp.183-192.

Gelfand S.B., Mitter S.K. (1993). Metropolis-type annealing algorithms for global optimization in $\mathbb{R}^d$, *SIAM Journal on Control and Optimization*, Vol.31, No.1, pp.111-131.

Gill P.E., Murray W., Wright M.H. (1981). *Practical Optimization*, London: Academic Press.

Goldberg D.E. (1989). *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley, Reading, Mass.

Golub G.H., Van Loan C.F. (1989). *Matrix Computations*, Baltimore MD: Johns Hopkins University Press.

Goodwin G., Sin K. (1984). *Adaptive filtering, prediction and control*, Prentice-Hall.

Grossberg S. (1988). Nonlinear neural networks: principles, mechanisms, and architectures, *Neural Networks*, Vol.1, pp.17-61.

Guzelis C., Chua L.O. (1993). Stability analysis of generalized cellular neural networks, *International Journal of Circuit Theory and Applications*, Vol.21, pp.1-33.

Harrer H. (1993). Multiple layer discrete-time cellular neural networks using time-variant templates, *IEEE Transactions on Circuits and Systems - II*, Vol. 40, No.3, pp.191-199.

Hebb D.D. (1949). *The organization of behaviour*, John Wiley & Sons, New York.

Hecht-Nielsen R. (1987). Kolmogorov's mapping neural network existence theorem, *Proc. IEEE International Conference on Neural Networks.*

Hecht-Nielsen R. (1988a). Neurocomputing: picking the human brain, *IEEE Spectrum*, 25(3), pp.36-41.

Hecht-Nielsen R. (1988b). Neurocomputer applications, in *NATO ASI Series, Vol.F 41, Neural computers*, Eds. R. Eckmiller, Ch. v.d. Malsburg, Springer-Verlag.

Hill D.J., Moylan P.J. (1980). Connections between finite-gain and asymptotic stability, *IEEE Transactions on Automatic Control*, Vol.AC-25, No.5, pp.931-936.

Hill D.J., Moylan P.J. (1976). The stability of nonlinear dissipative systems, *IEEE Transactions on Automatic Control*, Vol.AC-21, pp.708-711.

Hinton G.E. (1989). Connectionist Learning Procedures, *Artificial Intelligence*, 40, pp.185-234.

Hopfield J.J. (1982). Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Sciences of the USA*, 79, April, pp.2554-2558.

Hornik K., Stinchcombe M., White H. (1989). Multilayer feedforward networks are universal approximators, *Neural Networks*, Vol.2, pp.359-366.

Hunt K.J., Sbarbaro D. (1991). Neural networks for nonlinear internal model control, *IEE Proceedings-D*, Vol.138, No.5, pp.431-438.

Hunt K.J., Sbarbaro D., Zbikowski R., Gawthrop P.J. (1992). Neural networks for control systems - a survey, *Automatica*, Vol. 28., No. 6, pp.1083-1112.

Kaszkurewicz E., Bhaya A. (1993). Robust stability and diagonal Liapunov functions, *SIAM Journal on Matrix Analysis and Applications*, Vol.14, No.2, pp.508-520.

Kirkpatrick S., Gelatt C.D., Vecchi M. (1983). Optimization by simulated annealing, *Science*, 220, pp.621-680.

Kohonen T. (1977). *Associative memory - a system-theoretical approach*, Springer-Verlag, Berlin.

Le Cun Y. (1988). A theoretical framework for back-propagation, *Proceedings of the 1988 Connectionist Models Summer School*, San Mateo, CA: Morgan Kaufmann, (Eds. D. Touretzky, G. Hinton, T. Sejnowski).

Leshno M., Lin V.Y., Pinkus A., Schocken S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function, *Neural Networks*, Vol.6, pp.861-867.

Lippmann R.P. (1987). Introduction to computing with neural nets, *IEEE ASSP Magazine*, April, pp.4-22.

Liu C.-C., Chen F.-C. (1993). Adaptive control of non-linear continuous-time systems using neural networks - general relative degree and MIMO cases, *International Journal of Control*, Vol.58, No.2, pp.317-335.

Liu D., Michel A.N. (1992). Asymptotic stability of discrete time systems with saturation nonlinearities with applications to digital filters, *IEEE Transactions on Circuits and Systems I*, Vol.39, No.10, pp.798-807.

Ljung L. (1987). *System Identification: Theory for the User*, Prentice-Hall.

Ljung L. (1979). Asymptotic behavior of the extended Kalman filter as a parameter estimator for linear systems, *IEEE Transactions on Automatic Control*, Vol.AC-24, No.1, pp.36-50.

Maciejowski J.M. (1989). *Multivariable feedback design*, Addison-Wesley.

Miller W.T., Sutton R.S., Werbos P.J. (1990). *Neural networks for control*, Cambridge, MA: MIT Press.

Møller M.F. (1993). A scaled conjugate gradient algorithm for fast supervised learning, *Neural Networks*, Vol.6, pp.525-533.

Narendra K.S., Parthasarathy K. (1990). Identification and control of dynamical systems using neural networks, *IEEE Transactions on Neural Networks*, Vol.1, No.1, pp. 4-27.

Narendra K.S., Parthasarathy K. (1991). Gradient methods for the optimization of dynamical systems containing neural networks, *IEEE Transactions on Neural Networks*, Vol.2, No.2, pp.252-262.

Nesterov Y., Nemirovskii A. (1994). *Interior point polynomial algorithms in convex programming*, SIAM (Studies in Applied Mathematics), Vol.13.

Nguyen D., Widrow B. (1990). Neural networks for self-learning control systems, *IEEE Control Systems Magazine*, 10(3), pp.18-23.

Packard A., Doyle J. (1993). The complex structured singular value, *Automatica*, Vol.29, No.1, pp.71-109.

Parisini T., Zoppoli R. (1991). Backpropagation for N-stage optimal control problems, *International Joint Conference on Neural Networks, Signapore '91*, pp.1518-1529.

Park D.C., El-Sharkawi M.A., Marks II R.J., Atlas L.E., Damborg M.J. (1991). Electric load forecasting using an artificial neural network, *IEEE Transactions on Power Systems*, Vol.6., No.2, pp.442-449.

Park J., Sandberg I.W. (1991). Universal approximation using Radial-Basis-Function networks, *Neural Computation*, 3, 246-257.

Poggio T., Girosi F. (1990). Networks for approximation and learning, *Proceedings of the IEEE*, Vol.78, No.9, pp.1481-1497.

Polak E., Wardi Y. (1982). Nondifferentiable optimization algorithm for designing control systems having singular value inequalities, *Automatica*, Vol.18, No. 3, pp.267-283.

Psaltis D., Sideris A., Yamamura A. (1988). A multilayered neural network controller, *IEEE Control Systems Magazine*, April, pp.17-21.

Reed R. (1993). Pruning algorithms - a survey, *IEEE Transactions on Neural Networks*, Vol.4, No.5, pp.740-747.

Ritter H., Martinetz T., Schulten K. (1992). *Neural computation and self-organizing maps: an introduction*, Addison-Wesley.

Rosenblatt R. (1959). *Principles of neurodynamics*, Spartan Books, New York.

Rumelhart D.E., Hinton G.E., Williams R.J. (1986). Learning representations by back-propagating errors, *Nature*, Vol.323, pp.533-536.

Saarinen S., Bramley R., Cybenko G. (1993). Ill-Conditioning in neural network training problems, *SIAM Journal on Scientific Computing*, Vol.14, No.3, pp.693-714.

Saerens M., A. Soquet (1991). Neural controller based on back-propagation algorithm, *IEE Proceedings-F*, Vol.138, No.1, pp.55-62.

Saerens M., Renders J.-M., H. Bersini H. (1993). Neural controllers based on backpropagation algorithm. In *IEEE-Press Book on Intelligent Control: Theory and Practice*, M. M. Gupta, N. K. Sinha (Eds.), IEEE Press.

Sanner R.M., Slotine J.-J. E. (1992). Gaussian networks for direct adaptive control, *IEEE Transactions on Neural Networks*, Vol.3, No.6, pp. 837-863.

Saravanan N., Duyar A., Guo T.-H., Merrill W.C. (1993). Modeling of the space shuttle main engine using feed-forward neural networks, *Proceedings of the American Control Conference*, San Francisco, California, pp.2897-2899.

Sbarbaro-Hofer D., Neumerkel D., Hunt K. (1993). Neural control of a steel rolling mill, *IEEE Control Systems*, June, pp.69-75.

Schiffmann W.H., Geffers H.W. (1993). Adaptive control of dynamic systems by back propagation networks, *Neural Networks*, Vol.6, pp.517-524.

Shah S., Palmieri F., Datum M. (1992). Optimal filtering algorithms for fast learning in feedforward neural networks, *Neural Networks*, Vol.5, pp.779-787.

Sjöberg J., Ljung L. (1992). Overtraining, regularization and searching for minimum in neural networks, *4th IFAC International Symposium on adaptive systems in control and signal processing*, ACASP 92, pp.669-674, Grenoble, France.

Slotine J.-J. E., Sanner R.M. (1993). Neural networks for adaptive control and recursive identification: a theoretical framework, In *Essays on control: Perspectives in the theory and its applications*, H.L. Trentelman, J.C. Willems (Eds.), Birkhäuser.

Sontag E.D. (1993). Neural networks for control, in *Essays on control: Perspectives in the theory and its applications*, H.L. Trentelman, J.C. Willems (Eds.), Birkhäuser.

Sutton R.S., Barto A., Williams R. (1992). Reinforcement learning is direct adaptive optimal control, *IEEE Control Systems*, April, pp.19-22.

Suykens J.A.K., De Moor B. (1993a). Nonlinear system identification using multilayer neural networks: some ideas for initial weights, number of hidden neurons and error criteria, *IFAC World Congress, Sydney Australia*, July 19-23, Vol. 3, pp. 49-52.

Suykens J.A.K., De Moor B., Vandewalle J. (1993b). Stabilizing neural controllers: a case study for swinging up a double inverted pendulum, *NOLTA 93 International Symposium on Nonlinear Theory and its Applications, Honolulu, Hawaii*, Dec., Vol.2, pp.411-414.

Suykens J.A.K., De Moor B., Vandewalle J. (1993c). Neural network models as linear systems with bounded uncertainty, applicable to robust controller design, *NOLTA 93 International Symposium on Nonlinear Theory and its Applications, Honolulu, Hawaii*, Dec., Vol.2, pp.419-422.

Suykens J.A.K., De Moor B., Vandewalle J. (1994a). Static and dynamic stabilizing neural controllers, applicable to transition between equilibrium points, *Neural Networks*, Vol.7, No.5, pp.819-831.

Suykens J.A.K., De Moor B., Vandewalle J. (1994b). Nonlinear system identification using neural state space models, applicable to robust control design, *International Journal of Control*, to appear.

Suykens J.A.K., De Moor B., Vandewalle J. (1994c). Stability criteria for neural control systems, K.U. Leuven, Department of Electrical Engineering, ESAT-SISTA, Technical report 94-62I, submitted to *European Control Conference 95, Roma, Italy.*

Suykens J.A.K., De Moor B., Vandewalle J. (1994d). NL_q theory and its extension to μ theory, K.U. Leuven, Department of Electrical Engineering, ESAT-SISTA, Technical report 94-63I, submitted to *European Control Conference 95, Roma, Italy.*

Suykens J.A.K., Vandewalle J. (1994e). Generalized Cellular Neural Networks represented in the NL_q framework, K.U. Leuven, Department of Electrical Engineering, ESAT-SISTA, Technical report 94-71I, submitted to *IEEE International Symposium on Circuits and Systems (ISCAS 95), Seattle, Washington, USA.*

Suykens J.A.K., De Moor B., Vandewalle J. (1994f). Neural state space model framework for identification and control with global asymptotic stability criteria, *K.U. Leuven, Department of Electrical Engineering, ESAT-SISTA, Technical report 94-58I*, pp.75.

Suykens J.A.K., Vandewalle J. (1994g). A Fokker-Planck Learning Machine for Global Optimization, K.U. Leuven, Department of Electrical Engineering, ESAT-SISTA, Technical report 94-57I, submitted to *Neural Networks.*

Szu H. (1987). Nonconvex optimization by fast simulated annealing, *Proceedings of the IEEE*, Vol.75, No.11, pp.1538-1540.

Thierens D., Suykens J., Vandewalle J., De Moor B. (1993). Genetic weight optimization of a feedforward neural network controller, *Artificial neural nets and genetic algorithms, Proceedings of the international conference in Innsbruck, Austria 1993*, (eds. R.F. Albrecht, C.R. Reeves, N.C. Steele), Springer-Verlag Wien New York.

Tsoi A.C., Back A.D. (1994). Locally recurrent globally feedforward networks: a critical review of architectures, *IEEE Transactions on Neural Networks*, Vol.5, No.2, pp. 229-239.

van der Schaft A.J. (1992). L_2-gain analysis of nonlinear systems and nonlinear state feedback $\mathcal{H}_\infty$ control, *IEEE Transactions on Automatic Control*, Vol.AC-37, pp.770-784.

van der Schaft A.J. (1993). Nonlinear state space $\mathcal{H}_\infty$ control theory, in *Essays on Control: perspectives in the theory and its applications*, Eds. H.L. Trentelman, J.C. Willems, Birkhäuser.

van der Smagt P.P. (1994). Minimisation methods for training feedforward neural networks, *Neural Networks*, Vol.7, No.1, pp.1-11.

Van Overschee P., De Moor B.(1993) *Subspace identification of a glass tube manufacturing process.* Proc. of the second European Control Conference, June 28-July 1, 1993, Groningen, The Netherlands, pp. 2338-2343

Van Overschee P., De Moor B. (1994). N4SID : Subspace Algorithms for the Identification of Combined Deterministic-Stochastic Systems, *Automatica*, Vol. 30, No. 1, pp. 75-93.

Vidyasagar M. (1993). *Nonlinear systems analysis*, Prentice-Hall.

Weigend A.S., Gershenfeld N.A. (1993). Results of the time series prediction competition at the Santa Fe institute, *IEEE International Conference on Neural Networks*, San Francisco, CA, pp. 1786-1793.

Werbos P. (1990). Backpropagation through time: what it does and how to do it, *Proceedings of the IEEE*, 78 (10), pp.1150-1560.

Widrow B., Rumelhart D.E., Lehr M.A. (1994). Neural networks: applications in industry, business and science, *Communications of the ACM*, March, Vol.7, No.3.

Willems J.C. (1971a). The generation of Lyapunov functions for input-output stable systems, *SIAM Journal Control*, 9(1):105-134.

Willems J.C. (1971b). Least squares stationary optimal control and the algebraic Riccati equation, *IEEE Transactions on Automatic Control*, AC-16(6):621-634.

Willems J.C. (1972). Dissipative dynamical systems I: General theory. II: Linear systems with quadratic supply rates. *Archive for Rational Mechanics and Analysis*, 45:321-343.

Youssef H.M. (1993). Comparison of neural networks in nonlinear system modeling, *World Congress on Neural Networks '93*, Portland, Vol.IV, pp.5-9.

Zurada J.M. (1992). *Introduction to Artificial Neural Systems*, West Publishing Company.