On-line lot-sizing with perceptrons

DOI:
10.6100/IR499938

Document status and date:
Published: 01/01/1997

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

BETA

Institute for
Business Engineering
and Technology Application

On-line lot-sizing with perceptrons

H.P. Stehouwer

On-line lot-sizing with perceptrons

CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Stehouwer, Herman Pieter

On-line lot-sizing with perceptrons / by Herman Pieter Stehouwer. -
Eindhoven : Eindhoven University of Technology, 1997.
Proefschrift. - ISBN 90-386-0561-7
NUGI 811
Subject headings: neural networks / production planning
1991 Mathematics Subject Classification: 90B99, 68T05, 62H30

Printed by Ponsen & Looijen, Wageningen

On-line lot-sizing with perceptrons

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de Rector Magnificus, prof.dr. M. Rem, voor een commissie aangewezen door het College voor Promoties in het openbaar te verdedigen op dinsdag 28 oktober 1997 om 16.00 uur

door

Herman Pieter Stehouwer

geboren te Wageningen

Dit proefschrift is goedgekeurd door de promotoren:

prof.dr. J. Wessels
en
prof.dr. E.H.L. Aarts

Het onderzoek dat tot dit proefschrift heeft geleid werd mede mogelijk gemaakt door de Stichting Wiskunde Onderzoek Nederland met een subsidie van de Nederlandse Organisatie voor Wetenschappelijk Onderzoek.

to Angelique

Preface

Several people have contributed to this thesis in one way or another. I would like to start by expressing my gratitude to them.

First of all, I want to thank my supervisors Jaap Wessels and Emile Aarts. Emile taught me the essentials of doing research. His editorial abilities have improved the clarity of my work considerably. I want to thank Jaap for giving me the opportunity to find my own direction in research. His constructive criticism resulted in many improvements.

There are many others that contributed to this thesis directly or indirectly. I want to thank my colleagues, especially my office mates, who created a pleasant working atmosphere. Special thanks go to Rolf Suurmond, who carried out his M.Sc. work on a subject related to my own research. I want to thank him for the stimulating discussions we had. I am grateful to the members of 'Het Genotschap Epicurus' for providing several ideas for the appended theses. I want to thank my parents for all the opportunities they offered me.

Finally, I want to thank Angelique for her endless love and support.

Peter Stehouwer Boxtel, September 1997

Contents

1

Introduction

Lot-sizing is the timing and sizing of production quantities that satisfy the demand for a product such that production resources are used efficiently. Research on lot-sizing has been focussed to a large extent on the analysis of off-line models with a finite time horizon. For an overview of this literature we refer to the work of Bahl, Ritzman & Gupta [1987] and Aggarwal & Park [1993]. Off-line models assume that all information about problem parameters is given in advance. However, in many practical settings, demand information comes in gradually and a sequence of decisions has to be made over an indefinite time horizon. In such cases, off-line models are often inadequate and the lot-sizing has to be done *on-line*. Such situations arise for instance in material requirements systems and hierarchical planning systems [Silver & Peterson, 1985]. An obvious approach is to consider *on-line lot-sizing* from a probabilistic point of view by modeling the demand process as a random process and choosing a suitable optimality criterion. Although mathematically very attractive, such an approach is practically of little use, since it is usually already technically complicated for relatively simple models of the demand process. Most attention in on-line lot-sizing has therefore been paid to relatively simple heuristics. These heuristics often have varying performance characteristics which typically depend on the particular on-line lot-sizing problem at hand. As for now, no on-line lot-sizing approach exists that shows a satisfactory performance and is generally applicable.

Recent advances in the design and manufacturing of integrated circuits have brought the construction of parallel computers, consisting of thousands of individual processing units, within our reach. A direct consequence of these technological advances is the growing interest in computational models that support the exploitation of massive parallelism. *Connectionist models* [Feldman & Ballard, 1982] are computational models that are inspired by an analogy with the neural network of human brains. The corresponding parallel computers are called *neural networks* and the field of research *neural computing*. For an overview of this field we refer to the textbook by Hertz, Krogh & Palmer [1991]. Besides the purely technical benefit of parallel computation, many models in neural computing have human-like capabilities such as *association* and *learning*, which are essential in areas such as speech and image processing [Kohonen, 1988]. The largest potential of neural computing is in areas where no efficient solution strategies exist, in which modeling of the decision process is difficult, or in which problems are characterized by incomplete data. In recent years neural computing has emerged as a practical technology with successful applications in many fields [Wong, Bodnovich & Yakup, 1997]. Moreover, several of the methods, which were once justified by vague appeals to their neuron-like qualities, can now be given a solid statistical foundation [Bishop, 1995]. The majority of these results are concerned with pattern recognition and use neural network models with a feed-forward network topology such as the *multi-layered perceptron*.

The essential feature of a neural network approach to on-line lot-sizing is that it is an *implicit modeling approach*, which means that we look for a tuning of the parameters of the neural network such that it mimics a sensible input-output behavior, rather than that we try to explicitly model the underlying demand process. One may expect that such an approach can lead to useful results, since the on-line lot-sizing problem is too complicated for the traditional explicit modeling approaches and, moreover, it is relatively easy to determine what would have been the optimal lot sizes afterwards. The latter feature enables the construction of examples of *on-line* lot-sizing situations and their corresponding *off-line* optimal lot sizes required for the tuning of the parameters of the neural network. This approach combines the implicit model-building skills of neural networks with traditional off-line analysis. Being the most successful and most widely studied neural network model, the multi-layered perceptron is the most obvious candidate to be investigated. In this thesis we investigate the potential of multi-layered perceptrons for on-line lot-sizing.

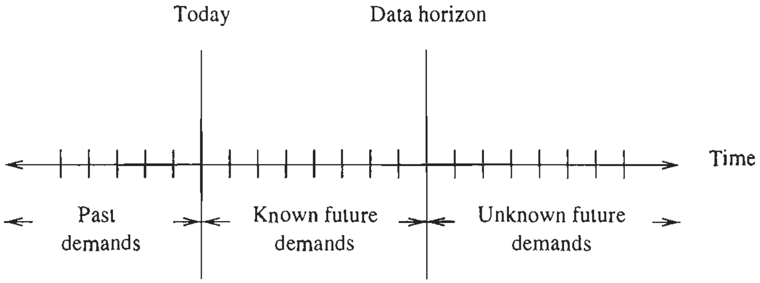


Figure 1.1. *On-line model of demand information.*

1.1 On-line lot-sizing

In on-line models it is often assumed that decisions are made without any knowledge of future problem data. For example, in the case of on-line bin packing, items arrive sequentially and whenever an item has arrived it must immediately be assigned to a bin. We refer to Karp [1992] for an overview of the literature in this area. In lot-sizing it is more realistic to adopt the following intermediate view between the two extremes constituted by off-line models (complete information about the future) and on-line models (no future information). It is assumed that the demand occurs during discrete time intervals called *periods* and is given for a fixed number of periods into the future. These periods are called the *data horizon.* Figure 1.1 shows the corresponding on-line model of demand information and distinguishes between demand information regarding the past, the known future, and the unknown future.

Important differences between off-line and on-line models for lot-sizing are in the problem formulation and the definition of optimality. Since all demands are given in advance, off-line lot-sizing problems can be formulated as optimization problems, where the goal is to find lot sizes such that demand is satisfied at minimal cost. However, in on-line problems the impact of a lot size decision depends in general on the unknown future demands, making a problem formulation conceptually more difficult.

To anticipate on the continuously changing knowledge of the future, on-line lot-sizing is usually done on a *rolling-horizon* basis, which can be described as follows. At the beginning of each period, one determines the lot sizes for a number of subsequent periods starting with the first period. Only the first of these lots becomes firm, the rest remains tentative. After this first decision has been implemented, the data horizon is updated and the procedure repeated. Therefore we may concentrate on the determination of the first lot size.

According to the specific model assumptions for the unknown future demands, we distinguish between three types of approach for on-line lot-sizing, i.e., *myopic* approaches, *explicit modeling* approaches, and *implicit modeling* approaches. Below we briefly discuss this taxonomy and give some examples.

Myopic approaches. In a myopic approach, nothing is assumed about the unknown future demands. The most obvious myopic approach is to optimize over the data horizon and to implement the first lot size. A typical result in this area is due to Lee & Denardo [1986], who give a worst-case error bound for this approach. Many myopic approaches involve the minimization of some local objective. For example in the heuristic proposed by Silver & Meal [1973], the first lot size is taken equal to the cumulative demand for the first k periods, where k is chosen such that the average cost per period is minimal. Several myopic approaches were proposed in the literature which are reviewed in Chapter 2.

Explicit modeling approaches. In an explicit modeling approach, the unknown future demands are explicitly modeled by assuming they are realizations of some random process, possibly with unknown parameter values. These parameters may characterize for example the noise part of the demand process or some systematic trend and could be estimated from past demand data. Given such an explicit model there are different possibilities.

One option that is commonly used in practice is to use an explicit model of the demand process to forecast some of the unknown future demand values. These forecasts are then incorporated in an off-line lot-sizing procedure [Silver & Peterson, 1985]. The performance of such an approach strongly depends on the quality of the forecasts and the sensitivity of the off-line lot-sizing procedure for forecast errors. If, moreover, it is assumed that the unknown future demands are independent realizations of a random variable, on-line lot-sizing problems can be formulated as Markov decision problems and can be solved as such [Tijms, 1994]. The results of this type of research are mathematically attractive but practically of little use, because of the rather restrictive structure which is required for the demand process and the relatively complicated estimation and optimization procedures required [Dellaert & Melo, 1995].

Finally, we mention the explicit modeling approach by Lee, Kramer & Hwang [1991], who model the unknown future demands as fuzzy sets [Zadeh, 1965] and solve the on-line lot-sizing problem as a fuzzy optimization problem [Delgado, Kacprzyk, Verdegay & Vila, 1994].

Implicit modeling approaches. In an implicit modeling approach, one takes a parameterized black-box in which the number of adaptive parameters can be increased in a systematic way. This black-box represents a very general class of functional forms and can be made increasingly general by increasing the number of adaptive parameters. Given such a black-box, one tries to find parameter values such that the device shows a sensible input-output behavior on at least a representative set of examples of on-line lot-sizing situations and their corresponding off-line optimal lot sizes.

An implicit modeling approach for the on-line version of the well-known lot-sizing problem introduced by Wagner & Whitin [1958] was proposed by Zwietering, Van Kraaij, Aarts & Wessels [1991]. To the best of our knowledge this is the only literature on the use of neural networks for on-line lot-sizing problems. We refer to the work of Corsten & May [1996] and Stehouwer, Aarts & Wessels [1994] for discussions of the potential for using neural networks for production planning and control applications.

Problem statement. Advantages of myopic approaches are the absence of demand history requirements and their often straightforward implementation. Unfortunately only worst-case performance guarantees can be given, since no assumptions are imposed on the unknown future demands. Some of these approaches can be arbitrarily bad [Vachani, 1992]. Although many myopic approaches were proposed in the literature, there is still no myopic approach that is *robust* in the sense that it yields a good performance, irrespective of the model parameters. These deficiencies could be overcome by adopting the mathematically attractive Markov decision formulation. However, this is only feasible if the nature of the demand process is well understood and not subject to change, which is hardly the case in practical situations. For these reasons there is a substantial gap between theory and practice and, up to now, there is no overall satisfactory solution approach for on-line lot-sizing problems.

The intention of this thesis is to investigate the potential of implicit modeling by multi-layered perceptrons for on-line lot-sizing problems. Implicit modeling approaches have the advantage that they do not require any prior understanding of the demand process and therefore have potential practical value. Implicit modeling approaches, however, do require a relevant demand history. An often heard argument against the use of neural networks is the black-box character of the obtained model. We object to that by remarking that most neural network approaches are statistically well-founded and from that point of view not essentially different from statistical methods [Bishop, 1995]. We aim at obtaining approaches that are robust, i.e., have good performance characteristics that are relatively insensitive to the model parameters. Under the condition that sufficient learning examples can be constructed, multi-layered perceptrons have the potential of providing such an approach.

1.2 Neural networks

A neural network consists of a network of elementary nodes that are linked through weighted connections. These nodes represent computational units which are capable of performing a simple computation. The result of this computation gives the output of the corresponding node. Moreover, the output of a node is used as an input for the nodes to which it is linked through an outgoing connection. The *network topology* of a neural network is determined by the number of nodes and the way they are connected. The neural network model under consideration is the *multi-layered perceptron*, in which the units are arranged in layers with connections between subsequent layers only. Multi-layered perceptrons are discussed in great detail in Chapter 4. For more details on other neural network models, we refer to the textbooks by Aarts & Korst [1989], Hecht-Nielsen [1990], Hertz, Krogh & Palmer [1991], and Kosko [1992].

The main tasks in the application of a neural network model to a certain problem consist of the determination of a network topology and connection weights such that the network solves the problem. To accomplish these tasks one can choose between two approaches, i.e., *network construction* or *learning*. These approaches are discussed next.

Network construction. In network construction, the network topology and the connection weights are derived directly from the problem formulation and are kept constant during the network execution. This embeds certain information into the network by design, which is reproduced during operation. Network construction is only applicable if the problem can be modeled and analyzed properly.

For instance, several network construction approaches were proposed for solving combinatorial optimization problems [Papadimitriou & Steiglitz, 1982]. In these approaches the combinatorial optimization problem is formulated in terms of a cost function which is to be minimized by the neural network; for examples, see the work of Looi [1992] and Zwietering [1994]. The main motivation for using such an approach is the potential speed-up from massively parallel computation. Until so far, however, the results obtained using neural network approaches for combinatorial optimization problems are disappointing. Their most important deficiency is a poor scalability to real-life instances [Foo & Takefuji, 1995; Zwietering, 1994].

Learning. In learning, the network topology and the connection weights are iteratively adjusted until the neural network performs the task accurately. At each iteration, an input is presented to the network and, according to the network outputs, the weights are adjusted. If, with the inputs, desired outputs are supplied and the weights are adjusted such that the difference between network outputs and desired outputs is minimized in some sense, the learning is called *supervised*. Such combinations of inputs and desired outputs are called *learning examples*. There are two other types of learning, called *unsupervised learning* and *reinforcement learning* [Hertz, Krogh & Palmer, 1991]. In unsupervised learning no correct outputs are supplied, which can be useful in data analysis. In reinforcement learning only information is supplied whether the network outputs are good or bad, which is mainly used for control applications.

Supervised learning has become very popular due to the discovery of suitable learning algorithms like the back-propagation algorithm [Rumelhart, McClelland & Williams, 1986], and is especially useful in case modeling or analysis is difficult. There exist many successful applications which include for instance forecasting, process monitoring, fault detection, and quality control [Dagli, 1994; Maren, Harston & Pap, 1990; Weigend & Gershenfeld, 1994; Wong, Bodnovich & Yakup, 1997; Zhang & Huang, 1995].

1.3 Towards a solution approach

The nature of an on-line lot-sizing problem is characterized by the following two components.

1. A *combinatorial* component involving the timing and sizing of the production quantities.
2. An *uncertainty* component representing the incomplete demand information.

In some sense these components are conflicting, since the combinatorial component involves detailed puzzling and benefits by complete demand information, which is contradicted by the uncertainty component. The essential problem is to somehow understand the demand process and to exploit this knowledge in the lot-sizing by anticipating on the formally unknown future. Note that the nature of an on-line lot-sizing problem typically changes with the length of the data horizon. It is likely that the combinatorial component becomes increasingly important if the data horizon is enlarged.

According to the treatment of the two components, we distinguish between two types of approach for on-line lot-sizing, i.e., *monolithic* approaches and *hierarchical* approaches. In a monolithic approach, both components are treated integrally. The Markov decision approach for on-line lot-sizing of Dellaert & Melo [1995] is a typical example of a monolithic approach. Hierarchical approaches first deal with the uncertainty component before the combinatorial puzzle is solved. A typical example of a hierarchical approach is to forecast some of the unknown future demand values which are then incorporated in an off-line lot-sizing procedure. Next we discuss the applicability of multi-layered perceptrons for on-line lot-sizing in this context.

Possibilities and limitations. The possibilities and limitations of multi-layered perceptrons for on-line lot-sizing can be discussed by distinguishing between *large data horizons* and *small data horizons*.

In case the data horizon is large, there is small demand uncertainty, and the essential problem lies in the combinatorial component. In such cases on-line lot-sizing problems can be viewed as combinatorial optimization problems. Zwietering, Aarts & Wessels [1991] and Zwietering [1994] showed that, in theory, multi-layered perceptrons can be constructed that solve each instance of a combinatorial optimization problem. However, for most problems the minimal required size of the network is already exponential in the number of inputs, which makes the approach impractical [Aarts, Stehouwer, Wessels & Zwietering, 1995]. Fortunately, there already exist excellent approaches for off-line lot-sizing problems based on the more traditional techniques [Aggarwal & Park, 1993; Federgruen & Tzur, 1991; Wagelmans, Van Hoesel & Koolen, 1992].

In case the data horizon is small, there is large demand uncertainty, the combinatorial component is less important, and the essential difficulty lies in the uncertainty component. We recall that emergent features of multi-layered perceptrons are their supervised learning and generalization capabilities, which enable implicit model-building on the basis of learning examples. Since model-building is the essential difficulty in the uncertainty component, multi-layered perceptrons have potential as a solution approach for on-line lot-sizing problems in which there is significant demand uncertainty.

From the discussion of these two cases, it is to be expected that there is in general a trade-off between combinatorial complexity and uncertainty. Furthermore, a monolithic approach based on multi-layered perceptrons seems only then appropriate if the data horizon is small. In a natural way the question arises if it is possible to develop a hierarchical approach which, for the uncertainty component, exploits the strong points of multi-layered perceptrons and, for the combinatorial component, builds upon the numerous results and techniques from off-line lot-sizing. In this way the best of both fields would be combined. We investigate hierarchical approaches for on-line lot-sizing problems based on supervised learning with multi-layered perceptrons.

Prerequisites. A necessary condition for the successful application of supervised learning is the availability of a representative set of learning examples. Therefore, a first prerequisite is the availability of a relevant demand history. From these past demands, such a set can be constructed in different ways. One option is to let a human expert judge situations in which lot-sizing decisions have been made. Another option is to calculate the optimal lot sizes afterwards, which is only possible if the lot-sizing model is well-defined. We study well-defined lot-sizing models and adopt the latter option. For that reason, a second prerequisite is an algorithm for the off-line calculation of learning examples. For the derivation of such algorithms we adopt the theory concerning planning and forecast horizons initiated by Wagner & Whitin [1958] and Lundin & Morton [1975]. If a forecast horizon can be found, optimal decisions for some periods can be obtained with limited information about future demands and cost parameters, even for infinite-horizon problems. To find such forecast horizons we employ a *forward algorithm*, which solves off-line finite horizon problems of increasing horizon length.

Variable-horizon policies. Motivated by the theory on planning and forecast horizons, we concentrate on the class of hierarchical approaches for on-line lot-sizing called *variable-horizon policies* . In such an approach, the lot sizes are determined by repeatedly optimizing over a variable *optimization horizon* which is determined by a *horizon-selection rule* on the basis of the available demand information. The optimization part computes the timing and sizing of the lot sizes and can be solved using dynamic programming techniques. The horizon-selection rule accounts for the uncertainty component of the on-line lot-sizing problem and uses as input the known future demands to return an optimization horizon within the data horizon. Such tasks, in which one of a finite number of possibilities has to be chosen on the basis of some feature vector, are usually called *classification* tasks. Through many successful applications, multi-layered perceptrons have shown to have excellent classification skills [Bishop, 1995; Pao, 1989; Ripley, 1994]. We address the problem of finding an *optimal* horizon-selection rule by formulating it as a *classification problem* and adopting common objectives from statistical classification like for instance the maximization of the expected classification rate. For these objectives it is easy to give explicit expressions for the optimal horizon-selection rules. We use supervised learning with multi-layered perceptrons to estimate the unknown parameters of these expressions and derive horizon-selection rules from the developed multi-layered perceptrons. The thus obtained hierarchical approaches combine the classification skills of multi-layered perceptrons with traditional off-line analyses.

1.4 Thesis outline

In Chapter 2, we formulate the on-line single-item lot-sizing problem with an arbitrary cost structure. We introduce a class of solution strategies for this problem, called variable-horizon policies, in which the lot sizes are determined by repeatedly optimizing over a variable optimization horizon. A horizon-selection rule chooses the optimization horizon given the available demand information. Furthermore, we introduce the corresponding off-line problem and derive forward algorithms to be used for the calculation of learning examples in Chapters 5, 6, and 7. These forward algorithms are only partly generic and require some cost-structure specific analysis.

In Chapter 3 we introduce three elementary cost structures which serve as a test bed for the evaluation of our ideas and techniques in Chapter 6 and Chapter 7. For these cost structures we give the cost-structure specific analysis that is required for the generic algorithms developed in Chapter 2.

Chapter 4 introduces the multi-layered perceptron and discusses its use in statistical classification. We discuss the mapping capabilities of multi-layered perceptrons for different response functions. Furthermore, we address supervised learning in a statistical perspective and discuss the subject of generalization.

In Chapter 5 we formulate the problem of finding an optimal horizon-selection rule as a classification problem, which we analyze in a statistical framework. We analyze two objectives, i.e., maximization of expected classification rate and minimization of expected excess cost. For these objectives we give explicit expressions for the optimal horizon-selection rules. Supervised learning with multi-layered perceptrons is used to estimate the unknown parameters of these expressions. Next we derive so-called MLP-based horizon-selection rules from the developed multi-layered perceptrons.

Chapter 6 studies the generalization capabilities of the MLP-based horizon-selection rules for an on-line lot-sizing problem with Wagner-Whitin cost structure. We discuss necessary conditions for good generalization and investigate the effect of the length of the data horizon on the generalization capabilities. Furthermore, we introduce K-nearest-neighbors, an alternative statistical approach for classification. Application of this approach yields two alternative horizon-selection rules which are used as a reference in our empirical studies.

In Chapter 7, we investigate the on-line lot-sizing performance of the variable-horizon policies constituted by the MLP-based horizon-selection rules proposed in Chapter 5 by means of an extensive empirical comparison with a benchmark of variable-horizon policies. The performance evaluation is done on a rolling-horizon basis for the three cost structures introduced in Chapter 3, for different combinations of demand processes and data horizon lengths. Preliminary results for these cost structures were presented in Stehouwer, Aarts & Wessels [1995] and Stehouwer, Aarts & Wessels [1996].

In Chapter 8, we conclude this thesis with a discussion of the obtained results. Moreover, we give some suggestions for future research.

2

Single-item lot-sizing

The intention of this chapter is twofold. First, it introduces the *on-line* single-item lot-sizing problem. Second, it develops algorithms that are used for the calculation of learning examples in later chapters. In general, such learning examples consist of *on-line* lot-sizing decision situations and their corresponding *off-line* optimal decisions. For that reason we also introduce and analyze the *off-line* single-item lot-sizing problem. Both the on-line problem and the off-line problem are generic in the sense that they are formulated in terms of arbitrary holding and production cost functions. Nevertheless, most algorithms developed in this chapter are only partly generic and their application typically requires some cost-structure specific analysis. In Chapter 3 we give this analysis for three cost structures.

The chapter is organized as follows. In Section 2.1, both the on-line and the off-line single-item lot-sizing model are introduced. We address the n-period problem in Section 2.2. This problem occurs as a subproblem in the solution approaches for both the on-line and the off-line lot-sizing problem. In Section 2.3 we analyze the off-line problem. The on-line problem is addressed in Section 2.4. Furthermore, we introduce the class of variable-horizon policies. Section 2.5 develops a forward algorithm for off-line simple planning horizon detection. Finally, in Section 2.6 and Section 2.7, some more forward algorithms are derived to be used for the calculation of learning examples.

2.1 Models for single-item lot-sizing

This section introduces the off-line model and the on-line model. Both models have the same basis, called the *basic model*, which is described first.

2.1.1 The basic model

Consider the case in which production has to be planned for a single commodity for which demand occurs during an infinite number of discrete time periods labeled $1, 2, \ldots$. Let d_t denote the demand in period t. It is assumed that all period demands are real-valued and non-negative. Let X_t and I_t denote the amount of production in period t and the inventory level at the end of period t, respectively. X_t is called the *lot size* for period t. For I_t we have

$$I_t = I_0 + \sum_{s=1}^{t} X_s - \sum_{s=1}^{t} d_s, \quad t = 1, 2, \ldots, \tag{2.1}$$

where I_0 denotes the initial inventory level. Furthermore, it is required that all demands must be satisfied on time and the lot sizes are nonnegative. Hence

$$X_t \geq 0 \quad \text{and} \quad I_t \geq 0, \quad t = 1, 2, \ldots . \tag{2.2}$$

The model includes production and holding cost. It is assumed that all cost functions are independent of time and are given in advance. Let $P : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ denote the cost function related to production and let $H : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ denote the cost function related to carrying inventory from one period to the next. It is further assumed that both P and H are strictly increasing. Therefore, there is no benefit in producing more than necessary. The pair (H, P) is called the *cost structure.*

By specifying the cost structure, different single-item lot-sizing models can be defined. Besides single-source models like the Wagner-Whitin model, in which there is only one way to satisfy demand, also multiple-source models can be defined. For instance, we may produce in-house or buy from outside suppliers. The difference between models with a single source and those with multiple sources is only in the production cost function P; see also Chapter 3.

Below we introduce the off-line model and the on-line model. These models are both build upon the basic model. The difference between the two models lies in their assumptions concerning demand data availability. In the off-line model it is assumed that there is complete information about future demands, whereas in the on-line model it is assumed that there is only partial information about future demands. For both models we give a problem formulation.

2.1.2 The off-line model

In this subsection we introduce the off-line model and we define the off-line lot-sizing problem. *Off-line* means that all information about future demands is given in advance. Our model encompasses an infinite number of periods. Therefore we extend the standard finite-horizon formulation of the problem along the lines of Lundin & Morton [1975]. First we give the standard finite-horizon formulation, which is defined as follows.

Definition 2.1. A vector $(X_1, \ldots, X_n)$ that satisfies (2.2) is called a *production plan* for the periods $1, \ldots, n$. The *cost* of a production plan $(X_1, \ldots, X_n)$ is given by

$$\sum_{t=1}^{n} [\ P(X_t) + H(I_t)\ ]. \tag{2.3}$$

The problem of finding a minimal cost production plan for the periods $1, \ldots, n$ is called the *n-period problem*; the corresponding minimal cost is denoted by $f(n)$. □

For reasons of convenience, a production plan for the periods $1, \ldots, n$ is called a *n-period plan* and a minimal cost n-period plan is called an *optimal* n-period plan. Furthermore, a production plan $(X_{u+1}, \ldots, X_v)$ for the periods $u + 1, \ldots, v$ is denoted by $\mathbf{X}_{uv}$ and $\mathbf{X}_{0v}$ is abbreviated to $\mathbf{X}_v$.

Next we turn to the infinite-horizon formulation. For many lot-sizing problems, one observes that the initial portion of an optimal production plan only depends on the demand information for a limited set of nearby periods. This gives rise to the following infinite-horizon optimality criterion.

Definition 2.2. Let $\mathbf{X}_t$ be a production plan. Then $f(n \mid \mathbf{X}_t)$ denotes the cost of an optimal n-period plan constrained to follow $\mathbf{X}_t$ for the periods $1, \ldots, t$. A production plan $\mathbf{X}_t$ is called *infinite-horizon optimal*, if there exits an integer n with $n \geq t$ such that

$$f(N \mid \mathbf{X}_t) = f(N) \quad \text{for all } N \geq n,$$

irrespective of demands in periods $n + 1, n + 2, \ldots$. We call t a *planning horizon* and n a *forecast horizon.* □

An obvious formulation of the *off-line problem* is to find infinite-horizon optimal lot sizes $X_1, X_2, \ldots$. For practical reasons, however, we concentrate on the determination of the first or first few infinite-horizon optimal lot sizes. It is easy to see that this is without loss-off generality, since by repeatedly solving instances of the off-line problem, the infinite-horizon optimal lot sizes $X_1, X_2, \ldots$ can be determined one by one. Stated formally, the off-line problem is to find an infinite-horizon optimal t-period plan $\mathbf{X}_t$ for some $t \in \mathbb{N}$, given I_0 and the demands $d_1, d_2, \ldots$. The off-line problem is further analyzed in Section 2.3.

We remark that the existence of infinite-horizon optimal lot sizes can in general not be assured. In fact, for specific cost structures, it is often possible to construct demand sequences for which no infinite-horizon optimal lot sizes exist [Bean, Smith & Yano, 1987]. Nevertheless, it is fairly safe to state that any reasonable problem is more likely to have infinite-horizon optimal lot sizes than not [Lundin, 1973; Lundin & Morton, 1975; Morton, 1981]. For conditions on the existence of planning horizons we refer to the work of Bean & Smith [1984] and Bean & Smith [1993].

2.1.3 The on-line model

We suppose that the realization of the demand in period $t + m$ becomes known at the end of period t. In this way the demands are always known for m periods into the future; these periods we call the *data horizon*. The integer m is referred to as the *length* of the data horizon. In analogy to the definition of the off-line problem we can define the *on-line problem* as to find an infinite-horizon optimal t-period plan $\mathbf{X}_t$ for some $t \in \mathbb{N}$, given I_0 and the demands $d_1, \ldots, d_m$. However, such infinite-horizon optimal production plans may depend on the demand in periods beyond the data horizon and therefore, in general, cannot be computed on the basis of the available demand information. We call an algorithm that only uses the available demand information to calculate a production plan an *m-policy*.

Definition 2.3. An algorithm for selecting $X_1, X_2, \ldots$ is called an *m-policy*, if (2.2) is satisfied and the choice of X_t for all $t = 1, 2 \ldots$ depends only on I_0 and d_s for $s < t + m$. □

In this thesis we aim at deriving m-policies that are optimal in some sense. Possible optimality criteria are to minimize some worst-case error bound on the cost, or, as is common in the literature on on-line algorithms, to introduce *competitiveness* [Karp, 1992]. Our definition of *on-line optimality* is closely related to our solution approach and is therefore given in Chapter 5. The on-line problem is further analyzed in Section 2.4.

2.2 The n-period problem

This section addresses the n-period problem introduced in Definition 2.1. We have to solve this problem, because it occurs as a subproblem in our solution approaches for both the on-line and the off-line problem. As a starting point we take the shortest path formulation due to Wagner & Whitin [1958]. This formulation is generalized to handle arbitrary cost structure (H, P). The corresponding shortest path algorithm is only partly generic and therefore its application to a particular cost structure requires cost-structure specific analysis. For three cost structures we give this analysis in Chapter 3.

Definition 2.4. Let $\mathbf{X}_n$ be a production plan. A vector $\mathbf{X}_{uv}$ with $0 \leq u < v \leq n$ is called a *subplan* of $\mathbf{X}_n$, if $I_u = I_v = 0$ and $I_s > 0$ for all $s = u+1, \dots, v-1$. A period t is called a *production period* if $X_t > 0$. If $I_t = 0$ we say that there is a *regeneration point* at the end of period t. □

Using these definitions the following property can be stated.

Proposition 2.1. [Wagner & Whitin, 1958]. *Suppose there exists an optimal n-period plan with a regeneration point at the end of period t with $0 < t < n$. Then the n-period plan that is obtained by independently finding optimal production plans for the first t periods and the last $n-t$ periods with $I_t = 0$, is optimal.*
Proof sketch. Production cost depends only on the amount produced in a particular period. Furthermore, since $I_t = 0$, the inventory holding cost associated with the last $n-t$ periods depends only on the lot sizes for these periods. □

Property 2.1 is known as the *inventory decomposition property* and is independent of the cost structure. In the sequel it is assumed that $I_0 = 0$. Since P is strictly increasing, there is no benefit in producing more than necessary, and all optimal n-period plans satisfy $I_n = 0$. From $I_0 = I_n = 0$ it follows that any optimal n-period plan can be decomposed into one or more subplans. From Proposition 2.1 it follows that, given the regeneration points, one can determine an optimal n-period plan by finding optimal subplans for each pair of consecutive regeneration points. Unfortunately, such optimal regeneration points are not known in advance. Nevertheless, the best combination of regeneration points can be selected, if optimal subplans are known for all possible pairs of regeneration points.

Let $c(u, v)$ denote the cost of an optimal production plan for the periods $u+1, \dots, v$ given that $I_u = I_v = 0$ and $I_t > 0$ for $t = u+1, \dots, v-1$. In other words, $c(u, v)$ represents the cost of an optimal subplan for the periods $u+1, \dots, v$. Let the possible regeneration points $0, \dots, n$ be represented in a network as nodes and let $c(u, v)$ represents the cost of traversing the arc from node u to node v. Then, since $I_0 = I_n = 0$ holds, each optimal n-period plan corresponds with a path from node 0 to node n. Since backlogging is prohibited, the network is acyclic. Hence, we can formulate the n-period problem as a shortest path problem in an acyclic network, which, given the arc costs $c(u, v)$ with $0 \leq u < v \leq n$, we can easily solve in $O(n^2)$ time using the forward recursion

$$\begin{cases} f(0) &= 0 \\ f(t) &= \min\{f(s) + c(s,t) \mid 0 \leq s < t\}, \quad 1 \leq t \leq n. \end{cases} \tag{2.4}$$

We refer to this underlying network as the *regeneration graph*. The usefulness of this recursion depends strongly on the complexity of computing the $n(n+1)/2$ arc costs and their corresponding subplans. In general, computing these arc costs and their corresponding subplans can be as difficult as the original problem and depends on the specific cost structure. Fortunately, for many interesting cost structures, the structure of optimal production (sub)plans have nice properties, which enable the arc costs to be calculated in polynomial time. This holds, for instance, in case the cost functions P and H are respectively fixed plus linear and linear [Wagner & Whitin, 1958], both concave [Love, 1973; Zangwill, 1968], or both piecewise concave [Swoveland, 1975]; see also the work of Aggarwal & Park [1993], Bitran & Yanasse [1982], and Florian, Lenstra & Rinnooy Kan [1980].

2.3 The off-line problem

The off-line problem was formulated as to find an infinite-horizon optimal t-period plan $\mathbf{X}_t$ for some $t \in \mathbb{N}$, given the demands $d_1, d_2, \ldots$. This means that we have to find a t-period plan $\mathbf{X}_t$ and an integer n such that $f(N \mid \mathbf{X}_t) = f(N)$ for all $N \geq n$ and irrespective of demands in periods $n+1, n+2, \ldots$. We called t a planning horizon and n a forecast horizon.

One easily verifies that it is equivalent to formulate the off-line problem as to determine an infinite-horizon optimal *ending condition* $I_t = I$, which is defined as follows.

Definition 2.5. We call the ending condition $I_t = I$ infinite-horizon optimal, if there exits an integer n with $n \geq t$, such that for all $N \geq n$ and irrespective of demands in periods $n+1, n+2, \ldots$ there exists an optimal N-period plan $\mathbf{X}_N$ with $I_t = I$. □

If ending condition $I_t = I$ is infinite-horizon optimal, then we can determine an infinite-horizon production plan $\mathbf{X}_t$ by solving the t-period problem with the constraint $I_t = I$; the integer t is a planning horizon.

The following result addresses the ending condition $I_t = 0$, which was called a regeneration point, and which plays an important role in many lot-sizing problems. Its proof is immediate from Proposition 2.1 and therefore omitted.

Proposition 2.2. *Let t and n be integers with $t \leq n$. Suppose that for all $N \geq n$ and irrespective of demands in periods $n+1, n+2, \ldots$ there exists an optimal solution to the N-period problem with a regeneration point at the end of period t. Then any optimal solution to the t-period problem is infinite-horizon optimal, t is a planning horizon, and n a forecast horizon.* □

2.3.1 Simple planning horizons

A planning horizon that satisfies the condition of Proposition 2.2 was called *simple* by Lundin & Morton [1975]. This gives rise to the following definitions.

Definition 2.6. The integer t is called a *simple planning horizon for forecast horizon n*, if for all $N \geq n$ and irrespective of demands in periods $n+1, n+2, \ldots$ there exists an optimal N-period plan with a regeneration point at the end of period t. The integer t is called a *simple planning horizon* if there exists an integer n such that t is a simple planning horizon for forecast horizon n. The integer n is called a *simple forecast horizon* if there exists a simple planning horizon for forecast horizon n. The smallest simple forecast horizon is called the *minimal* simple forecast horizon. □

Combining Proposition 2.1 and Definition 2.2 yields the following result.

Corollary 2.1. *Let t be a simple planning horizon. Then any optimal t-period plan is infinite-horizon optimal.* □

So, given a simple planning horizon t, the off-line problem *decomposes* into a t-period problem and a new off-line problem. In this way, off-line problems can be solved by repeatedly solving t-period problems, provided simple planning horizons can be found. We conclude that the existence of a simple planning horizon is a sufficient condition for the existence of an infinite-horizon optimal production plan. The necessity of this condition depends on the particular cost structure and is discussed next.

Concave cost structures. Using the following well-known results of Zangwill [1968] for single-item lot-sizing models with concave production and holding cost functions, we show that for such models the existence of a simple planning horizon is necessary and sufficient for the existence of an infinite-horizon optimal production plan; see also the work of Zangwill [1969] and Veinott [1969].

Proposition 2.3. [Zangwill, 1968]. *Suppose that both P and H are concave. Then for all optimal n-period plans $\mathbf{X}_n$ and $t = 1, \ldots, n$, we have*

(i) $I_{t-1} \times X_t = 0$, *and*

(ii) $X_t = 0$ *or* $X_t = D(t-1, k)$ *for some* $k \in \{t, \ldots, n\}$,

where $D(u, v)$ denotes the cumulative demand for the periods $u+1, \ldots, v$. □

Theorem 2.1. *Suppose that both P and H are concave. Then there exists an infinite-horizon optimal t-period plan with $t \in \mathbb{N}$ if and only if there exists a simple planning horizon.*

Proof. The 'only-if'-part is immediate from Proposition 2.2. We now prove the 'if'-part. Let $\mathbf{X}_t$ be an infinite-horizon optimal t-period plan and let n be the corresponding forecast horizon. In case $I_t = 0$, it is obvious that t is a simple planning horizon. What remains is the case $I_t > 0$. From Proposition 2.3 and the fact that n is a forecast horizon we infer that there exists a $k \in \{t+1, \ldots, n\}$ such that

$I_t = D(t, k)$ and $X_s = 0$ for all $s = t+1, \ldots, k$. It is easy to see that $\mathbf{X}_k$ is infinite-horizon optimal with $I_k = 0$, which implies that k is a simple planning horizon for forecast horizon n. This completes the proof. □

In the literature on concave cost models, simple planning horizons and forecast horizons are therefore often called planning horizons and forecast horizons, respectively [Bensoussan, Crouhy & Proth, 1983; Bensoussan & Proth, 1991]. Others completely focus on the underlying regeneration graph. Such a view was adopted by Federgruen & Tzur [1995], who discuss single-item lot-sizing models with concave production and holding cost functions within a generalized shortest path framework. They assume that the arc costs are given and that the arcs are indivisible actions. In their formulation a node n is called a forecast horizon if the shortest path from node 0 to node N goes through node t for all $N \geq n$ and irrespective of the arc costs $c(u, v)$ with $n \leq u < v$. Such a node t is referred to as a planning horizon.

General cost structures. For models with general cost functions, the existence of a simple planning horizon is in general not a necessary condition for the existence of an infinite-horizon optimal production plan, which is illustrated by the following example.

Example 2.1. Let us take the following cost structure

$$P(X) = \begin{cases} 0 & \text{if } X = 0 \\ 20 + X & \text{if } 0 < X \leq 10 \\ 20 + 10 + 3(X - 10) & \text{if } X > 10, \end{cases}$$

$$H(I) = I \quad \text{for all } I \geq 0,$$

which represents a simple overtime model with linear holding cost, setup cost 20, regular time production cost 1, overtime production cost 3, and a regular time production capacity of 10. Furthermore, we have an infinite horizon with demands $d_1 = 5$, $d_2 = 25$, and $d_t = 5$ for $t \geq 3$. Then for all t-period problems with t even, the first regeneration point in the unique optimal t-period plan is at the end of period 2. Furthermore for all t-period problems with t odd, the first regeneration point in the unique optimal t-period plan is at the end of period 3. It is impossible to conclude that a simple planning horizon prevails on basis of Proposition 2.2. Nevertheless, both the unique optimal 2-period plan as well as the unique optimal 3-period plan have the same first lot size $X_1 = 10$. So according to Definition 2.2 we have found an infinite-horizon optimal production plan, a planning horizon equal to 1, and a forecast horizon equal to 2. For more details on the overtime model we refer to Chapter 3. □

Although, as indicated by the above example, concentrating on simple planning horizons for general cost structures is at the risk of missing an infinite-horizon optimal production plan, we do concentrate on detecting simple planning horizons for the following reasons.

1. By looking only for simple planning horizons, we may concentrate on the underlying regeneration graph as we did for the n-period problem. Since this concept is generic, we can develop algorithms for general cost structures. Nevertheless, cost-structure specific analysis is still required for the computation of optimal subplans; see also Section 2.2.
2. In the extensive empirical study to be presented in Chapter 7, we computed over a million learning examples for both concave and non-concave cost functions. In all these cases a simple planning horizon could be detected. Based on these results, we conjecture that a small amount of variability in the demand process is sufficient for the existence of a simple planning horizon.
3. If a simple planning horizon can be detected, an attractive decomposition arises and infinite-horizon optimal lot sizes can be determined by solving the corresponding finite-horizon problem.

2.3.2 Forward algorithms

To find simple planning horizons, one commonly employs a *forward algorithm*. Such algorithms solve t-period problems for $t = 1, 2, \dots$ until a *stop criterion* indicates that a simple planning horizon has been found. It is obvious how to build a forward algorithm upon the forward recursion (2.4) for the n-period problem. Furthermore, if for $t = 1, 2, \dots$ we keep a record of the set of integers that occur as a regeneration point in any optimal t-period plan, at the kth iteration of the forward algorithm we can easily check if there exists an n and a t with $1 \leq t \leq n \leq k$ such that for $N = n, n+1, \dots, k$ there exists an optimal N-period plan with a regeneration point at the end of period t. Despite that, Definition 2.6 does not provide an appropriate stop criterion. What we need is a stop criterion that requires only a finite number of steps. Such a criterion is derived in Section 2.5.

The following notions can be used to express the efficiency of a forward algorithm with respect to the use of demand data.

Definition 2.7. Let the smallest integer n such that t is a simple planning horizon for forecast horizon n be denoted by $n^*(t)$. A forward algorithm that requires only $n^*(t)$ periods of demand information to discover a simple planning horizon t is called *perfect*. A forward algorithm is called *protective* if it can be guaranteed that it finds a simple planning horizon t under the condition that $n^*(t)$ is finite. □

Similar notions were introduced by Lundin & Morton [1975] for general planning and forecast horizons. Although there exist some examples of perfect forward algorithms for specific cost structures [Chand & Morton, 1986], such forward algorithms are in general hard to obtain.

2.4 The on-line problem

In analogy to the definition of the off-line problem we defined the *on-line problem* as to find an infinite-horizon optimal t-period plan $\mathbf{X}_t$ for some $t \in \mathbb{N}$, given I_0 and the demands $d_1, \ldots, d_m$. Since, in general, such infinite-horizon optimal production plans cannot be computed, we resort to algorithms that only use the available demand information, which we called m-policies. Many such algorithms are proposed in the literature, which we briefly review below.

The most obvious m-policy is to repeatedly optimize over the data horizon and to implement the first lot size. We refer to this approach as the *fixed-horizon policy*. Among others, Blackburn & Millen [1980] have pointed out that the cost performance of the fixed-horizon policy is poor for small data horizons. Furthermore, subsequent production plans can differ notably, introducing so-called *nervousness*.

Motivated by the theory on planning and forecast horizons, Chand [1982], Carlson, Beckman & Kropp [1982], and Federgruen & Tzur [1994] proposed m-policies for the Wagner-Whitin model in which the horizon over which is optimized is chosen dynamically. We call such policies *variable-horizon policies*. Chand [1982] uses a simple planning horizon result to construct a set of candidate simple planning horizons to chose from. A procedure similar to that of Silver & Meal [1973] is then used to choose an optimization horizon from this set. Federgruen & Tzur [1994] apply a forward algorithm to check if a simple planning horizon prevails within the data horizon. If this is the case, the corresponding finite-horizon problem is solved. In case no simple planning horizon is detected, they take an optimization horizon from a set of candidate simple planning horizons according to some worst-case error bound. Carlson, Beckman & Kropp [1982] investigated the use of forecasting to extend the data horizon for the Wagner-Whitin cost structure. Their approach was to forecast demand for as many periods in the future as necessary for a forward algorithm to detect a planning horizon.

The remaining part of the literature addresses the development of relatively simple heuristics that exhibit less nervousness than the fixed-horizon policy and perform well on the average. Most of these heuristics were designed for the special case of the Wagner-Whitin model with constant cost and determine the first lot size by aggregating a number of subsequent demands. These heuristics are also called *aggregation heuristics* [Silver & Peterson, 1985]. This number is determined by locally minimizing some reasonable objective function like for example least cost per unit time [Silver & Meal, 1973] or least cost per unit product [Gorham, 1968]. Other examples of such heuristics can be found in Bahl, Ritzman & Gupta [1987] and Silver & Peterson [1985].

Research on the performance of m-policies has mainly focussed on worst-case analysis and empirical testing. Worst-case results are given by Axsäter [1982], Axsäter [1985], Bitran, Magnanti & Yanasse [1984], Chen, Hearn & Lee [1995], Lee & Denardo [1986], Vachani [1992]. For empirical testing and a comparison of different m-policies in a rolling-horizon environment we refer to the work of Baker [1977], Baker [1989], Berry [1972], Blackburn & Millen [1980], Blackburn & Millen [1985], Chand [1982], Carlson, Beckman & Kropp [1982], Ritchie & Tsado [1986], and Zoller & Robrade [1988]. From these studies it turns out that non of these policies dominates under all cost and demand conditions.

2.4.1 Problem analysis

If we concentrate on simple planning horizons, as we did for the off-line problem, the on-line problem involves the determination of a simple planning horizon t with $t \in \{1, \dots, m\}$. Given such a simple planning horizon t the infinite-horizon optimal t-period plan is obtained by solving the t-period problem. Depending on I_0 and the known future demands $d_1, \dots, d_m$ and the formally unknown future demands $d_{m+1}, d_{m+2}, \dots$, we distinguish between the following four cases.

1. A simple planning horizon t for minimal forecast horizon n exists, such that $t \leq n \leq m$. Any perfect forward algorithm is able to detect these horizons in exactly n iterations.
2. A simple planning horizon t for minimal forecast horizon n exists, such that $t \leq m < n$. No forward algorithm is able to detect these horizons in m or less iterations.
3. A simple planning horizon t for minimal forecast horizon n exists, such that $m < t \leq n$. No forward algorithm is able to detect these horizons in m or less iterations.
4. No simple planning horizon exists.

Only in the first case, infinite-horizon optimal lot sizes can be determined with certainty. In the other three cases this is impossible, and one has to choose the lot sizes heuristically. Which of the four cases occurs depends on m, the cost structure, and the demand characteristics.

In the literature we found a number of studies, which empirically investigate the relation between on the one hand the cost structure and the demand characteristics and on the other hand the expected value of the planning horizon and the corresponding minimal forecast horizon. Lundin [1973] describes some experiments concerning prevalence and values of (simple) planning and forecast horizons for the Wagner-Whitin model with constant setup cost. In 97.5% of all cases a forecast horizon was found within 70 periods. His main conclusion is that the higher the average demand or the demand variability, the smaller the expected forecast horizon. Furthermore, he showed that if demand has an upward trend it is more likely to find forecast horizons than when demand has a downward trend. Federgruen & Tzur [1994] give similar conclusions for concave cost models with time varying cost structure; see also the work of Lundin & Morton [1975] and Morton [1981].

In these studies the main focus is usually at minimal forecast horizons and their corresponding (simple) planning horizons. From these studies we conclude that these simple planning horizons tend to be very small, indicating a very high potential for variable-horizon policies. Next we discuss these policies in more detail.

2.4.2 Variable-horizon policies

Figure 2.1 gives a template representing a general variable-horizon policy. The lot-sizes are determined by repeatedly optimizing over a chosen *optimization horizon* and implementing the lot sizes of the first subplan. Note that the inventory level at the end of Step 4. is zero. This template can be viewed as a generalization of the type of variable-horizon policies proposed by Chand [1982], Carlson, Beckman & Kropp [1982], and Federgruen & Tzur [1994] for the Wagner-Whitin model in which only the first lot size was implemented. To see this we refer to Proposition 2.3. This result implies that in case of concave cost functions, the first lot size of an optimal production plan equals the demand for an integer number of periods. So, for the Wagner-Whitin model, implementing the first subplan and implementing the first lot size are completely equivalent.

We call a rule that determines an optimization horizon given the demands within the data horizon a *horizon-selection rule.* Such a rule can be written as a mapping $g : \mathbb{R}^m \rightarrow \{1, 2, \ldots, m\}$ of the demands $d_1, \ldots, d_m$. Given a horizon-selection rule, the corresponding variable-horizon policy is completely determined. Below, we show that most m-policies that were proposed for the Wagner-Whitin model are in fact variable-horizon policies or can be simply adjusted to become one without loss of cost performance.

First, let us consider the fixed-horizon policy. It is easy to see that this policy is a variable-horizon policy in which the optimization horizon is fixed.

Second, we concentrate on the class of aggregation heuristics. In such heuristics, one produces the cumulative demand for the first k periods in period 1 with $k \in \{1, 2, \ldots, m\}$. The value of k is chosen on the basis of the available demand information. For instance one can take that k that minimizes the average cost per period. We adjust the aggregation heuristics as follows. It is easy to see that by

```
1. Given $I_0 = 0$ and demands within data horizon $d_1, \ldots, d_m$.
2. Choose an optimization horizon $t \in \{1, 2, \ldots, m\}$.
3. Determine an optimal $t$-period plan $\mathbf{X}_t$.
4. Implement the first subplan $(X_1, \ldots, X_s)$ of $\mathbf{X}_t$.
5. Update the data horizon, renumber the periods, and goto 1.
```

Figure 2.1. *A variable-horizon policy.*

choosing k, the inventory level becomes zero at the end of period k. Given such a regeneration point, by Proposition 2.1, it is optimal to solve the k-period problem independently. So we may use k as an optimization horizon and implement the first subplan without loss of cost performance. Although the computation of an optimal k-period plan for the Wagner-Whitin cost structure and the addition of k numbers both require $O(k)$ basic operations, it takes more time to compute the optimal k-period plan [Wagelmans, Van Hoesel & Koolen, 1992]. The advantage is that the adjusted aggregation policies can be applied to on-line problems with arbitrary cost structure.

2.4.3 A benchmark of variable-horizon policies

This subsection presents four variable-horizon policies, which are used as a reference in our empirical studies. We describe these variable-horizon policies by their horizon-selection rules. It is assumed that we are at the beginning of period 1 and that we have to choose an optimization horizon $t \in \{1, 2, \ldots, m\}$, given $I_0 = 0$ and the demands $d_1, d_2, \ldots, d_m$.

Economic order quantity policies. Consider the discrete time version of the infinite-horizon lot-sizing model with constant demand rate D of Harris [1913]. Then there is no demand uncertainty and it is mathematically optimal to use the same lot size each time a replenishment is made [Hax & Candea, 1984; Silver & Peterson, 1985]. Furthermore, because demand is deterministic and no shortages are allowed, it is clear that each replenishment is made when inventory is exactly zero. The optimal production policy is to produce a fixed quantity at each setup, which can be derived as follows. The total relevant cost per time unit corresponding to a fixed production quantity of Q are given by

$$\mathrm{TRC}(Q) \;=\; \frac{P(Q)D}{Q} + H(\bar{I}), \tag{2.5}$$

where $\bar{I}$ denotes the average inventory level. Examining the function $\mathrm{TRC}(Q)$ and its derivatives yields the value of Q that minimizes $\mathrm{TRC}(Q)$. This value is called the *economic order quantity* and is denoted by Q^*. For the lot-sizing problem with constant demand it is optimal to set up production every $n^* = Q^*/D$ time units and to produce Q^* units product at each setup. n^* is called the *order cycle*. Note that n^* constitutes a planning horizon.

Our models differ from the above model in that demand is in general subject to variation. Despite this difference, using a fixed economic order quantity is a simple and effective way for dealing with the on-line problem in situations where the demand rate is approximately constant [Silver & Peterson, 1985]. m-policies based on the economic order quantity proposed in the literature either use Q^* or n^*. We propose the following variable-horizon policy based on the order cycle. The selected optimization horizon is equal to the order cycle n^* rounded to the nearest integer greater than zero. For D we take the average demand level calculated over the data horizon, which is recalculated every time new information comes in. Note that, by using the average demand rate, time variability is simply ignored.

The least cost per unit product policy. The least cost per unit product policy is based on the aggregation heuristic described by Gorham [1968]. It selects the smallest optimization horizon t that minimizes the total cost per unit product. In other words it determines the smallest $t \in \{1, 2, \ldots, m\}$ such that

$$\frac{f(t)}{D(t)} = \min_{s \in \{1,\ldots,m\}} \frac{f(s)}{D(s)}, \tag{2.6}$$

where $D(v)$ denotes the cumulative demand covering periods $1, \ldots, v$. We remark that in the original procedure, as proposed by Gorham [1968], the first (local) minimum was chosen.

The least cost per unit time policy. The least cost per unit time policy is based on the aggregation heuristic described by Silver & Meal [1973]. It selects the smallest optimization horizon t that minimizes the total cost per unit time. In other words it determines the smallest $t \in \{1, 2, \ldots, m\}$ such that

$$\frac{f(t)}{t} = \min_{s \in \{1,\ldots,m\}} \frac{f(s)}{s}. \tag{2.7}$$

We remark that in the original procedure, as proposed by Silver & Meal [1973], the first (local) minimum was chosen.

The fixed-horizon policy. The fixed-horizon policy chooses t equal to m.

2.4.4 Discussion and outlook

All proposed benchmark policies are generic in the sense that they can be applied to arbitrary cost structures. However, policies based on the economic order quantity require cost-structure specific analysis. In Chapter 3 this analysis is given for the three cost structures under consideration.

Chapter 5 addresses the problem of finding an *optimal* horizon-selection rule. Since the problem of selecting an appropriate optimization horizon on the basis of the demands $d_1, \ldots, d_m$ is essentially a *classification problem*, in that chapter, we adopt common objectives from statistical classification like for instance maximization of the expected classification rate. For these objectives we give explicit expressions for the optimal horizon-selection rules. Supervised learning with multi-layered perceptrons is used to estimate the unknown parameters of these expressions and we derive approximative horizon-selection rules from the developed multi-layered perceptrons.

In the remainder of this chapter we derive forward algorithms for the off-line computation of learning examples. These algorithms are used in our experiments in Chapter 6 and Chapter 7.

2.5 Off-line simple planning horizon detection

The first planning horizon results are due to Wagner & Whitin [1958], who derived a forward algorithm and a stop criterion for the Wagner-Whitin model. Stronger results have been presented by Zabel [1964] and Eppen, Gould & Pashigian [1969]; however, these Wagner-Whitin-type planning horizon results are often unsatisfactory, because they only provide sufficient conditions for a simple planning horizon to occur. In fact, Lundin & Morton [1975] showed that such horizons only exist for a narrow range of cost parameter values.

2.5.1 Regeneration sets

Building on the work of Zabel [1964], Lundin [1973] and Lundin & Morton [1975] developed a more general theory of simple planning horizons and other stop criteria around the concept of *regeneration sets*. Given such a regeneration set they provide forward algorithms and give sufficient conditions for simple planning horizons to occur; see also the work of Chand [1979] and Morton [1981]. Below we introduce regeneration sets, which are used as a starting point for the derivation of a suitable stop criterion.

Definition 2.8. Let $\mathcal{S}$ be a finite set of integers and let $\underline{\mathcal{S}}$ and $\overline{\mathcal{S}}$ denote its minimal and maximal element, respectively. Then $\mathcal{S}$ is called a *regeneration set*, if for all $n \geq \overline{\mathcal{S}}$ and irrespective of demands in periods $\overline{\mathcal{S}} + 1, \overline{\mathcal{S}} + 2, \ldots$ there exists an optimal solution to the n-period problem with a regeneration point in $\mathcal{S}$. □

A regeneration set $\mathcal{S}$ is defined in such a way that any n-period problem, with $n \geq \overline{\mathcal{S}}$, has an optimal solution with a regeneration point in $\mathcal{S}$. Lundin & Morton [1975] proposed different strategies for deriving stop criteria from such regeneration sets. For instance, if for all elements t in a regeneration set, there exists an optimal t-

period plan with the same first regeneration point, then a simple planning horizon has been discovered.

Let the *length* of a subplan $\mathbf{X}_{uv}$ be the number of periods it covers, i.e., $v - u$. The next result is straightforward from Definition 2.4.

Proposition 2.4. *Suppose a finite upper bound M exists on the length of a subplan in an optimal n-period plan that is independent of n. Then any set of M consecutive periods constitutes a regeneration set.* □

Note that, if a finite upper bound M can be derived that satisfies the requirement of Proposition 2.4, then we have regeneration sets for any instance of the off-line problem.

Next we derive a stop criterion based on such regeneration sets. Let $\mathcal{L}_t$ denote the set of those s that minimize the right-hand side of (2.4). Then $\mathcal{L}_t$ represents the set of periods that occur as the last regeneration point but one in an optimal t-period plan. The set of periods that occur as a regeneration point in an optimal t-period plan is denoted by $\mathcal{R}_t$ and is defined by the forward recursion

$$\begin{cases} \mathcal{R}_0 &= \emptyset \\ \mathcal{R}_{t+1} &= \{t+1\} \cup \bigcup_{s \in \mathcal{L}_{t+1}} \mathcal{R}_s, \quad t = 0, 1, \ldots . \end{cases} \tag{2.8}$$

Given that such an upper bound M exists, the following theorem provides a necessary and sufficient condition for a simple planning horizon to occur. This condition is suitable for use as a stop criterion in a forward algorithm.

Theorem 2.2. *Suppose a finite upper bound M exists on the length of a subplan in an optimal n-period plan that is independent of n. Let*

$$\mathcal{S}_k(n) = \mathcal{R}_n \cap \mathcal{R}_{n+1} \cap \cdots \cap \mathcal{R}_{n+k-1}.$$

Then t is a simple planning horizon for forecast horizon n if and only if $t \in \mathcal{S}_M(n)$.

Proof. The 'only-if'-part follows directly from the definition of simple planning horizon. The 'if'-part we prove by showing that, if $t \in \mathcal{S}_M(n)$, we have $t \in \mathcal{R}_N$ for all $N \geq n$. We distinguish between two cases. In case $n \leq N < n + M$ this is obvious, since $t \in \mathcal{S}_M(n)$. What remains is the case $N \geq n + M$. Take such an N. Proposition 2.4 implies that $\{n, n+1, \ldots, n+M-1\}$ constitutes a regeneration set. So there exists a $k \in \{n, n+1, \ldots, n+M-1\}$ such that $k \in \mathcal{R}_N$. Take such a k, then $t \in \mathcal{S}_M(n)$ implies that $t \in \mathcal{R}_k$. The proof of the theorem is completed by using that $\mathcal{R}_k \subseteq \mathcal{R}_N$. □

2.5.2 A forward algorithm

The forward algorithm for the detection of simple planning horizons corresponding with Theorem 2.2 is presented in pseudo-code in Figure 2.2. For reasons of convenience we left out the code for the calculation of $f(n)$ and $\mathcal{R}_n$. On termination of

```
proc ForwardAlgorithm
var
  $k$, $n$ : int;
  $\mathcal{S}$ : set of int;
begin
  $k, n := 1, 1$;
  $\mathcal{S} := \mathcal{R}_1$; $\{\mathcal{S} = \mathcal{S}_1(1)\}$
  while $k < M$ do
    $\mathcal{S} := \mathcal{S} \cap \mathcal{R}_{n+1}$;
    if $S = \emptyset$ then
      $\mathcal{S}, k := \mathcal{R}_{n+1}, 1$;
      while $S \cap \mathcal{R}_{n-k+1} \neq \emptyset$ do $\mathcal{S}, k := \mathcal{S} \cap \mathcal{R}_{n-k+1}, k+1$ od
    else $k := k + 1$ fi
    $n := n + 1$; $\{\mathcal{S} = \mathcal{S}_k(n-k+1) \neq \emptyset\}$
  od $\{k = M\}$
end
```

Figure 2.2. *The forward algorithm in pseudo-code.*

the procedure t is a simple planning horizon for forecast horizon $(n - M + 1)$ for all $t \in \mathcal{S}_M(n - M + 1)$. Notice that Theorem 2.2 does not guarantee the existence of simple planning horizons; therefore, the termination of the forward algorithm cannot be assured. However, if a simple planning horizon for some forecast horizon exists, it is going to be found by the algorithm. So the forward algorithm is *protective*. Furthermore, if a simple forecast horizon is found by the algorithm it is the *minimal* simple forecast horizon. The forward algorithm is not perfect since it uses demand information for the periods $1, \ldots, n$ to detect a simple planning horizon for minimal forecast horizon $(n - M + 1)$.

2.5.3 Implementation issues

The forward algorithm has been implemented in the object-oriented programming language C++. We defined a class BasicClass, which contains generic features like for instance the shortest path recursion. For a specific type of cost structure, a new class has to be defined that inherits from BasicClass. For this new class, we only have to add two additional features, i.e., (i) an algorithm that calculates optimal subplans and their costs and (ii) an upper bound M on the length of a subplan. In order to possibly speedup the forward algorithm we incorporate the upper bound M in the shortest path recursion.

The forward algorithm frequently adds and merges ordered sets of integers. We used the efficient implementation of ordered sets by splay trees due to Sleator & Tarjan [1985] from the libg++-library. The amortized complexity of adding and merging are $O(\log n)$ and $O(n \log n)$ in the number of nodes n, respectively.

2.6 Off-line excess cost calculation

In this section we derive an algorithm that determines the excess cost incurred when decomposing the off-line problem at some period t and solving the t-period problem independently. In this section we derive such an algorithm. Such an algorithm is needed in our experiments in Chapter 6 and Chapter 7.

Let again $f(u, v)$ denote the cost of an optimal $(v-u)$-period plan for the periods $u+1, \ldots, v$. Then a recursive definition of $f(u, v)$ is straightforward from (2.4). We define $\Delta(p, n)$ as

$$\Delta(p, n) = f(p) + f(p, n) - f(n). \tag{2.9}$$

In words, $\Delta(p, n)$ denotes the excess cost over the cost of an optimal n-period plan incurred when independently finding optimal production plans for the first p periods and the last $n-p$ periods with $I_p = 0$. Furthermore, let $\Delta(p)$ be defined as

$$\Delta(p) = \lim_{n \to \infty} \Delta(p, n). \tag{2.10}$$

In words, $\Delta(p)$ denotes the excess cost of decomposing the off-line problem at period p. Remark that the limit value $\Delta(p)$ may not exist. Let $\mathcal{R}_{u,v}$ denote the set of integers that occur as a regeneration point in any optimal $(v-u)$-period plan for the periods $u+1, \ldots, v$. Then a recursive definition of $\mathcal{R}_{u,v}$ is straightforward from (2.8). The following result can be used as a stop criterion in a forward algorithm.

Theorem 2.3. *Suppose a finite upper bound M exists on the length of a subplan in an optimal n-period plan that is independent of n. Given is an integer p. Let*

$$\mathcal{S}_k(p, n) = \mathcal{S}_k(n) \cap \mathcal{R}_{p,n} \cap \mathcal{R}_{p,n+1} \cap \cdots \cap \mathcal{R}_{p,n+k-1}.$$

Then for all $t \in \mathcal{S}_M(p, n)$ we have

(i) $\Delta(p, N) = f(p) + f(p, t) - f(t)$ for all $N \geq n$, and

(ii) $\Delta(p) = f(p) + f(p, t) - f(t)$.

Proof. According to Theorem 2.2, all $t \in \mathcal{S}_M(p, n)$ are simple planning horizons. Completely analogously, it can be shown that all $t \in \mathcal{S}_M(p, n)$ are simple planning horizons for the instance of the off-line problem starting in period $p+1$. So we have $f(p, N) = f(p, t) + f(t, N)$ and $f(N) = f(t) + f(t, N)$ for all $t \in \mathcal{S}_M(p, n)$ and $N \geq n$. From this we infer that $f(p, t) - f(t) = f(p, N) - f(N)$ for all $t \in \mathcal{S}_M(p, n)$ and $N \geq n$, which, after some straightforward calculations, completes the proof. □

A major disadvantage of using Theorem 2.3 as a stop criterion in a forward algorithm is the large amount of memory that is required for storing the integer sets $\mathcal{R}_{p,s}$ and $\mathcal{R}_s$. Besides, it is often sufficient to know the excess cost within a certain prespecified tolerance. The following result provides a stop criterion that is less memory demanding. Moreover, in a natural way, this criterion allows for the inclusion of a tolerance parameter.

Theorem 2.4. *Suppose a finite upper bound M exists on the length of a subplan in an optimal n-period plan that is independent of n. Given is an integer p. For all $t \geq p$ we define*

$$\begin{aligned} L_k(p,t) &= \min_{t \leq s < t+k} \Delta(p,s) \\ U_k(p,t) &= \max_{t \leq s < t+k} \Delta(p,s). \end{aligned}$$

Then we have

(i) $L_M(p,t) \leq \Delta(p,n) \leq U_M(p,t)$ *for all* t, n *with* $n \geq t \geq p$,

(ii) $L_M(p,t) \leq \Delta(p) \leq U_M(p,t)$ *for all* $t \geq p$,

(iii) $L_M(p,t+1) \geq L_M(p,t)$ *for all* $t \geq p$,

(iv) $U_M(p,t+1) \leq U_M(p,t)$ *for all* $t \geq p$, *and*

(v) $U_M(p,t+1) - L_M(p,t+1) \leq U_M(p,t) - L_M(p,t)$ *for all* $t \geq p$.

Proof. Since in case $t \leq n < t + M$ the proof of part (i) is straightforward, we concentrate on the case $n \geq t + M$. Take such an n. From Proposition 2.4 we know that $\{t, t+1, \ldots, t+M-1\}$ is a regeneration set. Using the definition of regeneration set, there exist k, l with $t \leq k, l < t + M$, such that $f(n) = f(k) + f(k,n)$ and $f(p,n) = f(p,l) + f(l,n)$. Using $f(p,n) \leq f(p,k) + f(k,n)$ we derive $\Delta(p,n) \leq \Delta(p,k)$. Similarly, using $f(n) \leq f(l) + f(l,n)$, we derive $\Delta(p,n) \geq \Delta(p,l)$. Combining these two inequalities and the definition of $L_M(p,t)$ and $U_M(p,t)$ yields $L_M(p,t) \leq \Delta(p,l) \leq \Delta(p,n) \leq \Delta(p,k) \leq U_M(p,t)$, which completes the proof of part (i). Part (ii) is a direct consequence of part (i). Using part (i) we have $L_M(p,t) \leq \Delta(p,t+M) \leq U_M(p,t)$ for all t with $t \geq p$, which implies parts (iii) and (iv). Part (v) follows immediately from parts (iii) and (iv). □

Given such an upper bound M, we can calculate $\Delta(p)$ within a tolerance β with $\beta \geq 0$, by using $U_M(p,t) - L_M(p,t) \leq \beta$ as a stop criterion in a forward algorithm. Although $U_M(p,t) - L_M(p,t)$ is decreasing in t, the termination of the forward algorithm cannot be assured. Since the implementation of this algorithm is straightforward, its details are omitted.

2.7 Off-line optimal optimization horizon selection

This section derives forward algorithms that facilitate the off-line computation of the best possible optimization horizon. To that end we generalize the simple planning and forecast horizon framework developed in Section 2.5 by adopting a different notion of optimality, called *k-optimality*, which is defined as follows.

Definition 2.9. Let $f_k(n)$ denote the cost of a optimal n-period plan consisting only of subplans of length less than or equal to k. The corresponding optimal production plans are called *k-optimal.* □

It is easy to see that an inventory decomposition property as Proposition 2.1 can be derived for k-optimal production plans as well. Therefore, we formulate the problem of finding k-optimal production plan as a shortest path problem on the regeneration graph. The difference is that only arcs with a length less than or equal to k have to be considered. The corresponding recursion is given by

$$\left\{\begin{array}{lcl} f_k(0) & = & 0 \\ f_k(t) & = & \min\{f_k(s) + c(s,t) \mid s \geq 0,\ t-k \leq s < t\}, \quad t \geq 1. \end{array}\right. \tag{2.11}$$

Note that the set of paths with arc lengths less than or equal to j is a subset of the set of paths with arc lengths less than or equal to i. The following results are therefore immediate.

Proposition 2.5. *$f_i(t) \leq f_j(t)$ for all $i \geq j \geq 1$ and $t \geq 0$.* □

Proposition 2.6. *$f_k(t) \geq f(t)$ for all $k, t \geq 0$.* □

Proposition 2.7. *Suppose a finite upper bound M exists on the length of a subplan in an optimal n-period plan that is independent of n. Then $f_k(t) = f(t)$ for all $k \geq M$ and $t \geq 0$.* □

2.7.1 k-optimal simple planning horizons

Below, the simple planning and forecast horizon framework developed in Section 2.5 is straightforwardly generalized to k-optimality. Most results are given without proof, because similar results have already been proved in Section 2.5.

Definition 2.10. The integer t is called a *k-optimal simple planning horizon for forecast horizon n*, if for all $N \geq n$ and irrespective of demands in periods $n+1, n+2, \ldots$ there exists a k-optimal N-period plan with a regeneration point at the end of period t. The integer t is called a *k-optimal simple planning horizon* if there exists an integer n such that t is a k-optimal simple planning horizon for forecast horizon n. The integer n is called a *k-optimal simple forecast horizon* if there exists a k-optimal simple planning horizon for forecast horizon n. The smallest k-optimal simple forecast horizon is called the *minimal* k-optimal simple forecast horizon. □

Definition 2.11. Let t be a k-optimal simple planning horizon. Then any k-optimal t-period plan is called *infinite-horizon k-optimal.* □

Definition 2.12. Let $\mathcal{S}$ be a finite set of integers and let $\underline{\mathcal{S}}$ and $\overline{\mathcal{S}}$ denote its minimal and maximal element, respectively. Then $\mathcal{S}$ is called a *k-optimal regeneration set*, if for all $n \geq \overline{\mathcal{S}}$ and irrespective of demands in periods $\overline{\mathcal{S}}+1, \overline{\mathcal{S}}+2, \ldots$ there exists a k-optimal n-period plan with a regeneration point in $\mathcal{S}$. □

Proposition 2.8. *Any set of k consecutive periods constitutes a k-optimal regeneration set.* □

The following result is a direct consequence of this.

Corollary 2.2. *Suppose there exists a k-optimal simple planning horizon. Then there exists a k-optimal simple planning horizon t with $t \in \{1, \ldots, k\}$.* □

Let $\mathcal{L}_t^k$ denote the set of those s that minimize the right-hand side of (2.11). Then $\mathcal{L}_t^k$ represents the set of periods that occur as the last regeneration point but one in a k-optimal t-period plan. The set of periods that occur as a regeneration point in an k-optimal t-period plan is denoted by $\mathcal{R}_t^k$ and is recursively defined by

$$\begin{cases} \mathcal{R}_0^k &= \emptyset \\ \mathcal{R}_t^k &= \{t\} \cup \bigcup_{s \in \mathcal{L}_t^k} \mathcal{R}_s^k, \quad t \geq 1. \end{cases} \tag{2.12}$$

Proposition 2.9. *Let $\mathcal{S}_l^k(n) = \mathcal{R}_n^k \cap \mathcal{R}_{n+1}^k \cap \cdots \cap \mathcal{R}_{n+l-1}^k$. Then t is a k-optimal simple planning horizon for forecast horizon n, if and only if $t \in \mathcal{S}_k^k(n)$.* □

2.7.2 Off-line k-optimal simple planning horizon detection

We conclude that k-optimal simple planning horizons can be detected by using the forward algorithm derived in Section 2.5 for the detection of simple planning horizons with $M = k$. Note that this only works because, in our implementation, we included the bound M in the forward recursion (2.4); see Section 2.5.3.

2.7.3 Off-line k-optimal excess cost calculation

Let $f_k(u, v)$ denote the cost of an k-optimal $(v - u)$-period plan for the periods $u + 1, \ldots, v$. Then a recursive definition of $f_k(u, v)$ is straightforward from (2.11). We define $\Delta_k(p, n)$ as

$$\Delta_k(p, n) = f_k(p) + f_k(p, n) - f_k(n). \tag{2.13}$$

In words, $\Delta_k(p, n)$ denotes the excess cost over the cost of an k-optimal n-period plan incurred when independently finding k-optimal production plans for the first p

periods and the last $n - p$ periods with $I_p = 0$. Furthermore, let $\Delta_k(p)$ be defined as

$$\Delta_k(p) = \lim_{n \to \infty} \Delta_k(p, n). \tag{2.14}$$

We remark that this limit value may not exist. Given such an upper bound M, we can calculate $\Delta(p)$ within a tolerance β with $\beta \geq 0$ by using $U_M(p, t) - L_M(p, t) \leq \beta$ as a stop criterion in a forward algorithm. Although $U_M(p, t) - L_M(p, t)$ is decreasing in t, the termination of the forward algorithm cannot be guaranteed. Since the implementation of this algorithm is straightforward, its details are omitted.

We conclude that $\Delta_k(p)$ can be calculated by using the forward algorithm derived in Section 2.6 with $M = k$. Note that this only works because, in our implementation, we included the bound M in the forward recursion (2.4); see Section 2.5.3.

2.7.4 Implementation issue

Although an additional finite upper bound N on the length of a subplan in an optimal n-period plan that is independent of n is no longer required, in case such a bound N is available and if $N < k$, we can speedup the forward algorithm by taking $M = N$.

3

Some elementary cost structures

The framework for single-item lot-sizing presented in the previous chapter was formulated in terms of an arbitrary cost structure. For a particular cost structure, the framework presupposes three features. First, optimal subplans and their costs can be computed efficiently. Second, there exist an upper bound on the length of a subplan in an optimal production plan. Third, we have an expression for the economic order quantity. This chapter derives these cost structure specific features for three elementary cost structures, which, in the remainder of this thesis, serve as a test bed for the evaluation of our ideas and techniques.

The chapter is organized as follows. Section 3.1 addresses the Wagner-Whitin cost structure. The corresponding lot-size model is a single-source model in the sense that there is only one way to satisfy demand. Two-source models with overtime and purchasing are addressed in Section 3.2 and Section 3.3, respectively.

3.1 The Wagner-Whitin cost structure

The first cost structure originates from the single-source model described by Wagner & Whitin [1958]. Demand is only satisfied through in-house production. The production cost function P is fixed plus linear (concave) and is given by

$$P(X) = \begin{cases} 0 & \text{if } X = 0 \\ S + pX & \text{if } X > 0, \end{cases} \tag{3.1}$$

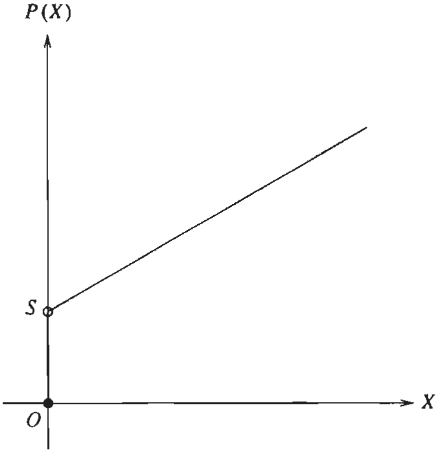


Figure 3.1. *Single-source production cost function.*

where p denotes the production cost per unit of product and S denotes the setup cost. The holding cost function is linear and is given by

$$H(I) = hI \quad \text{for all } I \geq 0, \tag{3.2}$$

where h denotes the holding cost per unit of product per period. We assume that $p, h > 0$ and $S \geq 0$, so that both P and H are strictly increasing. We refer to this cost structure as the *Wagner-Whitin cost structure.*

3.1.1 Characterization of optimal subplans

Since both H and P are concave, we may apply Proposition 2.3 to obtain the following result.

Corollary 3.1. *Let* $\mathbf{X}_{uv}$ *be a subplan of an optimal n-period plan* $\mathbf{X}_n$. *Then* $X_{u+1} = D(u, v)$. □

Using this result it is easy to see that the cost of an optimal subplan, i.e., the cost of an arc in the underlying shortest path problem in the regeneration graph, is given by

$$c(u, v) = P(D(u, v)) + \sum_{t=u+1}^{v-1} H(D(t, v)).$$

These arc costs can be computed recursively as was shown by Evans [1985]. In his algorithm, the computation of all arc costs requires $O(n^2)$ basic operations (additions, multiplications, and comparisons). Using the forward recursion (2.4), we obtain an $O(n^2)$ algorithm for the n-period problem with Wagner-Whitin cost structure. Recently, a number of authors developed algorithms for the n-period problem with Wagner-Whitin cost structure requiring only $O(n)$ basic operations [Aggarwal & Park, 1993; Federgruen & Tzur, 1991; Wagelmans, Van Hoesel & Koolen, 1992]. These algorithms exploit the special structure of the model even more.

3.1.2 Bounds on the length of an optimal subplan

Throughout this chapter we use the notation $\mathbf{e}_i$, which denotes an n-component vector with a one in the ith position, and zeros elsewhere. Furthermore, let $\lfloor x \rfloor$ denote the largest integer smaller than or equal to x.

Theorem 3.1. *Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then*

$$v - u \leq \left\lfloor \frac{S}{hd_v} \right\rfloor + 1 < \infty.$$

Proof. Let $\delta = \min\{X_{u+1}, \min_{u<t<v} I_t\}$. Then, since $\mathbf{X}_{uv}$ is a subplan, $\delta > 0$. Let $\mathbf{X}'_n = \mathbf{X}_n - \delta\mathbf{e}_{u+1} + \delta\mathbf{e}_v$. One easily verifies that $\mathbf{X}'_n$ is a n-period plan. Since $\mathbf{X}_n$ is optimal, the cost of $\mathbf{X}'_n$ must be greater than or equal to the cost of $\mathbf{X}_n$. Subtracting the equal cost components yields the inequality

$$\begin{aligned} v - u - 1 &\leq \frac{P(\delta) - (P(X_{u+1}) - P(X_{u+1} - \delta))}{h\delta} \\ &\leq \frac{S}{h\delta}. \end{aligned}$$

Corollary 3.1 implies that $X_{u+1} = D(u, v)$ and therefore $I_t = D(t, v)$ for all $t = u+1, \ldots, v-1$ and $\delta = d_v$. The proof of the first inequality is completed by using that $v - u$ is integer. Since $\mathbf{X}_{uv}$ is a subplan, $d_v > 0$. Together with $h > 0$, this proves the finiteness of $\lfloor S/(hd_v) \rfloor + 1$. □

Corollary 3.2. *Suppose a lower bound $d_L > 0$ on positive demand in a period exists. Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then*

$$v - u \leq \left\lfloor \frac{S}{hd_L} \right\rfloor + 1 < \infty.$$

□

For instance when demand is integer valued the upper bound on the length of a subplan of an optimal production plan is equal to $\lfloor S/h \rfloor + 1$. Note that such an upper bound may drastically reduce the number of arcs to be considered in the underlying shortest path problem. We use these bounds in the forward algorithm developed in Chapter 2.

3.1.3 Economic order quantity

The purpose of this subsection is to derive a horizon-selection rule based on the economic order quantity as proposed in Section 2.4.3. The variable-horizon policy constituted by this rule is used as a reference in our empirical studies. Next we analyze the discrete time version of the infinite-horizon lot-sizing model with constant demand described in Section 2.4.3.

Let D be the demand rate in units per period. It is easy to see that the average inventory level $\bar{I}$ is equal to $Q/2$. The total relevant cost per period, defined by (2.5), become

$$\mathrm{TRC}(Q) = \frac{hQ}{2} + \frac{DS}{Q} + pD.$$

Examining $\mathrm{TRC}(Q)$ yields the economic order quantity Q^* and the corresponding order cycle n^* given by

$$Q^* = n^* D = \sqrt{\frac{2DS}{h}}. \tag{3.3}$$

Although Harris [1913] was the first who derived this formula, it is widely known as *Wilson's lot size formula*. Wilson was a consultant who used such a formula in his work on inventory management in many companies [Hax & Candea, 1984]. For details on this analysis we refer to the textbooks by Hax & Candea [1984] and Silver & Peterson [1985].

The horizon-selection rule selects the optimization horizon equal to the order cycle n^* rounded to the nearest integer greater than zero. For D we take the average demand level calculated over the data horizon, which is recalculated every time new information comes in.

3.1.4 A worst-case result

In Section 2.6 we defined $\Delta(p, n)$, which denotes the excess cost over the cost of an optimal n-period plan, incurred when independently finding optimal production plans for the first p periods and the last $n - p$ periods with $I_p = 0$. The following worst-case result is due to Bitran, Magnanti & Yanasse [1984].

Proposition 3.1. *$\Delta(p, n) \leq S$ for all $p = 1, \ldots, n$.* □

This result can be generalized to k-optimality as follows.

Proposition 3.2. *$\Delta_k(p, n) \leq S$ for all $p = 1, \ldots, n$ and $k \in \mathbb{N}$.* □

Note that these results also hold for the limit values $\Delta(p)$ and $\Delta_k(p)$, provided these limit values exist.

3.2 A cost structure with overtime

The second cost structure originates from the two-source model mentioned by Jagannathan & Rao [1973]. Demand is satisfied either by production during normal

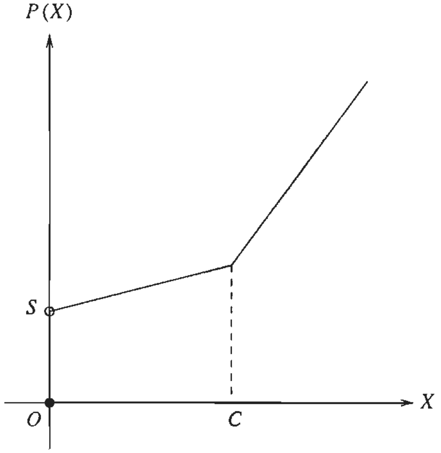


Figure 3.2. *Two-source production cost function with overtime.*

time or by production during overtime. The production cost function P is piecewise concave and is given by

$$P(X) = \begin{cases} 0 & \text{if } X = 0 \\ S + pX & \text{if } 0 < X \leq C \\ S + pC + q(X - C) & \text{if } X > C, \end{cases} \tag{3.4}$$

where C denotes the regular time production capacity, S denotes the setup cost, p denotes the regular time production cost per unit product, and q denotes the overtime production cost per unit product. The inventory cost function H is linear and is given by (3.2). We assume that $q > p > 0$. Furthermore, we assume that $h, C > 0$ and $S \geq 0$, so that P and H are both strictly increasing. In the analysis, the difference in cost per unit product between overtime production and regular time production plays an important role. For notational reasons we define $r = q - p$. We refer to r as the *overtime premium*. Jagannathan & Rao [1973] and Dixon [1980] analyzed similar cost structures with additional bounds on overtime production and inventory. Baker, Dixon, Magazine & Silver [1978] and Dixon, Elder, Rand & Silver [1983] considered overtime models with backlogging.

3.2.1 Properties of optimal production plans

First consider the following general result for single-item lot-sizing models with arbitrary piece-wise concave cost functions due to Swoveland [1975].

Proposition 3.3. [Swoveland, 1975]. *Suppose that P is concave on each of k intervals $[q_{i-1}, q_i]$ with $i = 1, \ldots, k$, and H is concave on each of l intervals $[b_{i-1}, b_i]$ with $i = 1, \ldots, l$. Let $\mathcal{P} = \{q_0, \ldots, q_k\}$ and $\mathcal{H} = \{b_0, \ldots, b_l\}$. Then there exists* *an optimal n-period plan with the property that between successive periods s and t with $1 \leq s < t \leq n$ and $I_s, I_t \in \mathcal{H}$ there is at most one period u with $s < u \leq t$ and $X_u \notin \mathcal{P}$.* □

We now return to the specific production cost function with overtime that is subject of this section. Using the formulation of Proposition 3.3, we have $k = 2$, $q_0 = 0$, $q_1 = C$, and $q_2 = \infty$ for the production cost function P and $l = 1$, $b_0 = 0$, and $b_1 = \infty$ for the holding cost function H. Applying Proposition 3.3 yields the following result.

Corollary 3.3. *There exists an optimal n-period plan with the property that each subplan contains at most one production period with a lot size that is unequal to the capacity of regular production C.* □

Theorem 3.2. *Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then $X_b \leq X_d$ for all production periods b and d with $u + 1 \leq b < d \leq v$.*

Proof. We concentrate on the case that $\mathbf{X}_{uv}$ contains two or more production periods, otherwise the proof is trivial. Suppose, on the contrary, that there exist production periods b and d with $u + 1 \leq b < d \leq v$ such that $X_b > X_d$. We show that such a production plan cannot be optimal. Let δ be defined by

$$\delta = \begin{cases} \min\{X_b, \min_{b \leq t < d} I_t, C - X_d\} & \text{if } X_d < X_b \leq C \\ \min\{\min_{b \leq t < d} I_t, C - X_d\} & \text{if } X_d < C \leq X_b \\ \min\{X_b - C, \min_{b \leq t < d} I_t\} & \text{if } C \leq X_d < X_b. \end{cases}$$

From X_b and X_d being production periods and $\mathbf{X}_{uv}$ being a subplan of $\mathbf{X}_n$, we infer that $\delta > 0$. Let $\mathbf{X}'_n = \mathbf{X}_n - \delta \mathbf{e}_b + \delta \mathbf{e}_d$. One easily verifies that $\mathbf{X}'_n$ is a n-period plan and has lower cost. This contradicts with $\mathbf{X}_n$ being optimal. □

3.2.2 Characterization of optimal subplans

Definition 3.1. A subplan $\mathbf{X}_{uv}$ of a n-period plan $\mathbf{X}_n$ is called *well-formed* if

(i) At most one production period t with $u + 1 \leq t \leq v$ exists such that $X_t \neq C$, and

(ii) $X_b \leq X_d$ for all production periods b and d with $u + 1 \leq b < d \leq v$.

□

The following result is straightforward from Corollary 3.3 and Theorem 3.2.

Corollary 3.4. *There exists an optimal n-period plan that consists only of well-formed subplans.* □

To facilitate a characterization of optimal well-formed subplans, we introduce the *cumulative demand axis* as described by Chung & Lin [1988]. Instead of giving

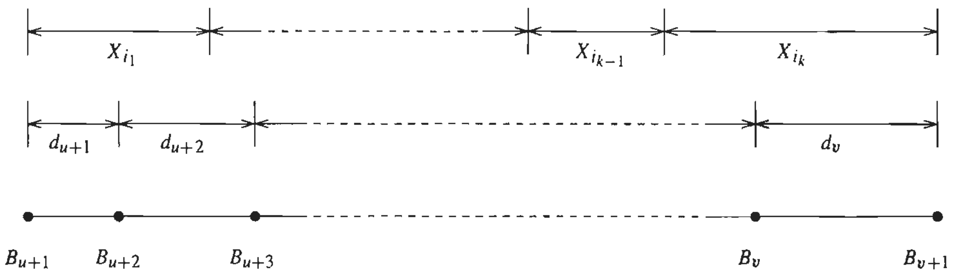


Figure 3.3. *Cumulative demand axis.*

each period an equal length on a time axis, each period is represented by an interval of length proportional to the demand in that period, and demand is spread uniformly over a period. The origin is used to indicate the beginning of period 1. We then mark the points $B_1 = 0$ and $B_t = D(0, t-1)$ for $t = 2, \ldots, n+1$. Each point B_t refers to the end of period $t-1$ and the beginning of period t, hence the interval from B_t to B_{t+1} represents the demand in period t. In Figure 3.3 this concept is visualized for a subplan $\mathbf{X}_{uv}$ with k production periods $i_1 < i_2 < \ldots < i_k$. Using this cumulative demand axis, it becomes clear that production in period i_k is used to meet the demand from $B_{v+1} - X_{i_k}$ to B_{v+1}. Production in period i_{k-1} is used to meet the demand from $B_{v+1} - X_{i_k} - X_{i_{k-1}}$ to $B_{v+1} - X_{i_k}$, and so on. In a subplan the production in each period can only be used to meet present or future demand and inventory must be positive, therefore we require $B_{v+1} - X_{i_k} > B_{i_k}$, $B_{v+1} - X_{i_k} - X_{i_{k-1}} > B_{i_{k-1}}$, and so on.

Theorem 3.3. *Let $\mathbf{X}_{uv}$ be a well-formed subplan of an optimal n-period plan $\mathbf{X}_n$. Suppose that $\mathbf{X}_{uv}$ has k production periods $i_1 < i_2 < \ldots < i_k$. Then we have*

(i) $i_1 = u + 1$,

(ii) If $k = 1$, *then* $X_{u+1} = D(u, v)$, *and*

(iii) If $k > 1$, *then the lot sizes* $X_{i_1}, \ldots, X_{i_k}$ *are given by*

$$(X_{i_1}, \ldots, X_{i_k}) = \begin{cases} (\beta, C, \ldots, C) & \text{if } 0 < \beta \leq C \\ (C, \ldots, C, \beta) & \text{if } \beta > C, \end{cases}$$

where $\beta = D(u, v) - (k-1)C$. *The timing of the lot sizes* $i_1, \ldots, i_k$ *is given by* $i_s = j_s$ *for* $s = 2, \ldots, k$, *where* j_s *is defined for* $s = k, k-1, \ldots, 1$ *by the backward recursion*

$$j_s = \max\{j \mid u+1 \leq j < j_{s+1},\ B_{v+1} - \sum_{m=s}^{k} X_{i_m} > B_j\},$$

with boundary condition $j_{k+1} = v + 1$.

Proof. Since $\mathbf{X}_{uv}$ is a subplan, parts (i) and (ii) are obvious. The remainder of this proof concentrates on part (iii). Subplan $\mathbf{X}_{uv}$ contains k production periods. Since $\mathbf{X}_{uv}$ is well-formed, at least $k-1$ of these production periods have a lot size of C. The lot size in the remaining production period is equal to $\beta = d_{uv} - (k-1)C$. Since $\mathbf{X}_{uv}$ is well-formed, there are two possibilities for the timing of β, i.e., in case $0 < \beta \leq C$ it must be in the first production period and in case $\beta > C$ it must be in the last production period. Using the definition of cumulative demand axis and since $\mathbf{X}_{uv}$ is a subplan, $i_s \leq j_s$ for $s = 2, \ldots, k$. Suppose there exists an index s with $2 \leq s \leq k$ such that $i_s < j_s$. Let t be the largest such index. So $i_t < j_t$, and $i_s = j_s$ for all $t < s \leq k$. We define $\mathbf{X}'_n = \mathbf{X}_n - X_{i_t}\mathbf{e}_{i_t} + X_{i_t}\mathbf{e}_{j_t}$. From the definition of j_t it follows that $\mathbf{X}'_n$ is a n-period plan and one easily verifies that $\mathbf{X}'_n$ has lower cost. This contradicts with $\mathbf{X}_n$ being optimal and we conclude that $i_s = j_s$ for all $s = 2, \ldots, k$. □

Theorem 3.3 implies that given the number of production periods of a well-formed subplan for the periods $u+1, \ldots, v$, computing the optimal lot sizes $X_{u+1}, \ldots, X_v$ requires $O(v-u)$ basic operations. Since an optimal n-period plan exists that consists only of well-formed subplans, it is sufficient to enumerate over all $(v-u)$ possible values for k and to choose a subplan with lowest cost. In this way, an optimal subplan can be obtained using $O((v-u)^2)$ basic operations, and the computation of all arc costs requires $O(n^4)$ basic operations. Using the forward recursion (2.4), we obtain an algorithm for the n-period problem with the overtime cost structure that requires $O(n^4)$ basic operations.

3.2.3 Bounds on the length of an optimal subplan

Lemma 3.1. *Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then*

$$d - b \leq \left\lfloor \frac{r}{h} \right\rfloor < \infty$$

for all production periods b and d with $u+1 \leq b < d \leq v$.

Proof. We concentrate on the case that $\mathbf{X}_{uv}$ contains two or more production periods, otherwise the proof is trivial. Let δ be defined by $\delta = \min\{X_b, \min_{b \leq t < d} I_t\}$. From the definition of subplan and from b being a production period it is obvious that $\delta > 0$. Let $\mathbf{X}'_n = \mathbf{X}_n - \delta\mathbf{e}_b + \delta\mathbf{e}_d$. One easily verifies that $\mathbf{X}'_n$ is a n-period plan. Since $\mathbf{X}_n$ is optimal, the cost of $\mathbf{X}'_n$ must be greater than or equal to the cost of $\mathbf{X}_n$. Subtracting the equal cost components gives the inequality

$$\begin{aligned}(d-b) &\leq \frac{(P(X_d+\delta) - P(X_d)) - (P(X_b) - P(X_b - \delta))}{h\delta} \\ &\leq \frac{q-p}{h},\end{aligned}$$

which, together with $d-b$ being integer, completes the proof of the first inequality. The finiteness of $\lfloor r/h \rfloor + 1$ is obvious from $h > 0$. □

Lemma 3.2. *Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then there exists a last production period d with $u < d \leq v$ and*

$$v - d \leq \left\lfloor \frac{S}{h \min\{C, d_v\}} \right\rfloor < \infty.$$

Proof. From $\mathbf{X}_{uv}$ being a subplan it is obvious that a last production period d exists. If $d = v$ we have $v - d = 0 < \infty$ and the result is obvious. We concentrate on the case $d < v$. Let $\delta = \min\{X_d, \min_{d \leq t < v} I_t\}$. Then, since $\mathbf{X}_{uv}$ is a subplan, $\gamma > 0$. Because d is the last production period, $I_t = D(t, v)$ for all $t = d, \ldots, v-1$, and therefore $\gamma = \min\{X_d, I_{v-1}\} = \min\{X_d, d_v\}$. Below, we show that γ is bounded from below. Therefore consider the following two cases. In case $d = u + 1$, it is obvious that $X_d = D(u, v) \geq d_v$. In case $u + 1 < d < v$, Corollary 3.3, Theorem 3.2, and $X_{u+1} > 0$, imply that $X_d \geq C$. So $\gamma \geq \min\{C, d_v\}$. Let $\mathbf{X}'_n = \mathbf{X}_n - \delta \mathbf{e}_d + \delta \mathbf{e}_v$. One easily verifies that $\mathbf{X}'_n$ is a n-period plan. Since $\mathbf{X}_n$ is optimal, the cost of $\mathbf{X}'_n$ must be greater than or equal to the cost of $\mathbf{X}_n$. Subtracting the equal cost components yields the inequality

$$v - d \leq \frac{P(\delta) - (P(X_d) - P(X_d - \delta))}{h\delta}.$$

By distinguishing between different cases for X_d and δ we have

$$\frac{v-d}{h\delta} \leq \begin{cases} S & \text{if } X_d \leq C \\ S - r(X_d - C) & \text{if } X_d > C,\ \ X_d - \delta \leq C, \text{ and } \delta \leq C \\ S - r(X_d - \delta) & \text{if } X_d > C,\ \ X_d - \delta \leq C, \text{ and } \delta > C \\ S - r\delta & \text{if } X_d > C,\ \ X_d - \delta > C, \text{ and } \delta \leq C \\ S - rC & \text{if } X_d > C,\ \ X_d - \delta > C, \text{ and } \delta > C \end{cases}$$
$$\leq S.$$

Using that $\delta \geq \min\{C, d_v\}$ yields $v - d \leq S/(h \min\{C, d_v\})$. The proof of the first inequality is completed by using that $v - d$ is integer. Since $\mathbf{X}_{uv}$ is a subplan, $d_v > 0$. Together with $h > 0$ and $C > 0$ this proves the finiteness of $\lfloor S/(h \min\{C, d_v\}) \rfloor$.
□

Theorem 3.4. *Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then*

$$v - u \leq \left\lfloor \frac{r}{h} \right\rfloor + \left\lfloor \frac{S}{h \min\{C, d_v\}} \right\rfloor + 1 < \infty.$$

Proof. Theorem 3.3 implies that period $u + 1$ is the first production period. Let period d be the last production period. Then $u + 1 \leq d \leq v$. Applying Lemma 3.1

yields $d - u \leq \lfloor r/h \rfloor + 1$. The proof is completed by combining this inequality with the equality of Lemma 3.2. □

The results of Theorem 3.4 are easily verified by substituting $p = q$ and $C = \infty$ to obtain Theorem 3.1.

Corollary 3.5. *Suppose a lower bound $d_L > 0$ on positive demand in a period exists. Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then*

$$v - u \leq \left\lfloor \frac{r}{h} \right\rfloor + \left\lfloor \frac{S}{h \min\{C, d_L\}} \right\rfloor + 1 < \infty.$$

□

Corollary 3.4 provides us with an upper bound on the length of an individual arc on a shortest path, i.e., a subplan $\mathbf{X}_{uv}$ can be part of an optimal plan only if $v - u$ is less than or equal to this bound. Using this upper bound in our dynamic programming algorithm may reduce the number of computations drastically. Besides this computational profit, we use these upper bounds as a stop criterion for our forward algorithm.

3.2.4 Economic order quantity

Again we derive a horizon-selection rule based on the economic order quantity as proposed in Section 2.4.3. The corresponding variable-horizon policy is used as a reference in our empirical studies. Next we analyze the discrete time version of the infinite-horizon lot-sizing model with constant demand described in Section 2.4.3.

Let D be the demand rate in units per period. It is easy to see that the average inventory level $\bar{I}$ is equal to $Q/2$. The total relevant cost per period, defined by (2.5), become

$$\begin{aligned} \mathrm{TRC}(Q) &= \frac{P(Q)\,D}{Q} + H(Q/2) \\ &= \frac{hQ}{2} + \begin{cases} \frac{DS}{Q} + pD & \text{if } 0 < Q \leq C \\ \frac{D(S-rC)}{Q} + \dot{q}D & \text{if } Q > C. \end{cases} \end{aligned}$$

Carefully examining TRC(Q) yields the economic order quantity Q^* and the corresponding order cycle n^*. Let r_1, r_2 be defined by

$$r_1 = \sqrt{\frac{2DS}{h}},$$

and

$$r_2 = \sqrt{\frac{2D(S - rC)}{h}}.$$

Then

$$Q^* = n^* D = \begin{cases} r_1 & \text{if } 0 < r_1 \leq C \\ r_2 & \text{if } S > rC, r_2 > C, \text{ and TRC}(r_2) < \text{TRC}(C) \\ C & \text{otherwise.} \end{cases} \tag{3.5}$$

The horizon-selection rule selects the optimization horizon equal to the order cycle n^* rounded to the nearest integer greater than zero. For D we take the average demand level calculated over the data horizon, which is recalculated every time new information comes in.

3.3 A cost structure with purchasing

The third cost structure is based on the following two-source model. Demand is satisfied either by in-house production or by purchasing from an outside supplier. Suppose that the in-house production capacity is equal to C, and let p, q denote the in-house production cost per unit product and the purchasing cost per unit product, respectively. There is a fixed setup cost S for in-house production in a certain period; there are no fixed charges for purchasing. So the in-house production cost is fixed plus linear (concave) as in the Wagner-Whitin cost structure and the purchasing cost is linear. Such models are also called models with stockouts or models with lost-sales and were analyzed by Sandbothe & Thompson [1990] as concave cost network flow problems.

As was indicated by Chen, Hearn & Lee [1994] the difference between single-source models and multiple-source models can be captured in the production cost function. To make any sense, q must be greater than p, which implies that a break-even point B exists between producing and purchasing. This break-even point is found by solving $S + pB = qB$ and is given by $B = S/(q - p)$. For the same reason we assume that $B < C$. Using this result, the two-source model is captured by the following piecewise linear production cost function

$$P(X) = \begin{cases} qX & \text{if } 0 \leq X \leq B \\ S + pX & \text{if } B < X \leq C \\ S + pC + q(X - C) & \text{if } X > C. \end{cases} \tag{3.6}$$

The inventory cost function H is linear and is given by (3.2). We assume that $q > p > 0$, $h, C > 0$, and $S \geq 0$, so that P and H are both strictly increasing. In the analysis, the difference in cost per unit product between purchasing and in-house production is important. For notational reasons we define $r = q - p$. We refer to r as the *purchase premium*.

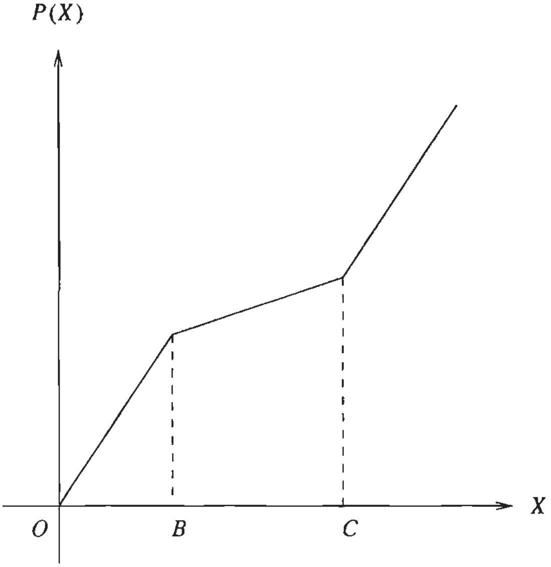


Figure 3.4. *Two-source production cost function with purchasing.*

3.3.1 Properties of optimal production plans

Using the formulation of Proposition 3.3, we could choose $k = 3$, $q_0 = 0$, $q_1 = B$, $q_2 = C$, and $q_3 = \infty$ for the production cost function P. But, since P is concave on $[0, C]$, we may also take $k = 2$, $q_0 = 0$, $q_1 = C$, and $q_2 = \infty$. For the holding cost function H, we take $l = 1$, $b_0 = 0$, and $b_1 = \infty$. Applying Proposition 3.3 yields the following result.

Corollary 3.6. *There exists an optimal n-period plan with the property that each subplan contains at most one production period with a lot size that is unequal to the capacity of in-house production C.* □

Given a production plan **X**. Let $R(X_t)$, $S(X_t)$ denote the amount of in-house production in period t, and the amount of product purchased in period t, respectively. These quantities can be calculated from X_t by

$$R(X_t) = \begin{cases} 0 & \text{if } 0 \leq X_t \leq B \\ X_t & \text{if } B < X_t \leq C \\ C & \text{if } X_t > C \end{cases}$$

and

$$S(X_t) = X_t - R(X_t).$$

Using this notation we can state the following two results due to Sandbothe & Thompson [1990].

Proposition 3.4. *There exists an optimal n-period plan* $\mathbf{X}_n$ *such that* $I_t S(X_t) = 0$ *for all t with* $1 \leq t \leq n$. □

Proposition 3.5. *There exists an optimal n-period plan* $\mathbf{X}_n$ *such that* $I_{t-1}(C - R(X_t))R(X_t) = 0$ *for all t with* $1 \leq t \leq n$. □

3.3.2 Characterization of optimal subplans

Definition 3.2. A subplan $\mathbf{X}_{uv}$ of a n-period plan $\mathbf{X}_n$ is called *well-formed* if

(i) At most one production period t with $u+1 \leq t \leq v$ exists such that $X_t \neq C$,

(ii) $S(X_t) = 0$ for $t = u+1, \ldots, v-1$, and

(iii) $R(X_t) \in \{0, C\}$ for $t = u+2, \ldots, v$.

□

The following result is straightforward from Corollary 3.6, Proposition 3.4, and Proposition 3.5.

Corollary 3.7. *There exists an optimal n-period plan that consists only of well-formed subplans.* □

Using the cumulative demand axis defined in Section 3.2.2 we can now characterize optimal well-formed subplans. The proof of the following result is similar to that of Theorem 3.5 and therefore omitted.

Theorem 3.5. *Let* $\mathbf{X}_{uv}$ *be a well-formed subplan of an optimal n-period plan* $\mathbf{X}_n$. *Suppose that* $\mathbf{X}_{uv}$ *has k production periods* $i_1 < i_2 < \ldots < i_k$. *Then we have*

(i) $i_1 = u + 1$,

(ii) If $k = 1$, *then* $X_{u+1} = D(u, v)$, *and*

(iii) If $k > 1$, *then the lot sizes* $X_{i_1}, \ldots, X_{i_k}$ *are given by*

$$(X_{i_1}, \ldots, X_{i_k}) = \begin{cases} (\beta, C, \ldots, C) & \text{if } B < \beta \leq C \\ (C, \ldots, C, \beta) & \text{otherwise,} \end{cases}$$

where $\beta = D(u, v) - (k-1)C$. *The timing of the lot sizes* $i_2, \ldots, i_k$ *is given by* $i_s = j_s$ *for* $s = 2, \ldots, k$, *where* j_s *is defined for* $s = k, k-1, \ldots, 1$ *by the backward recursion*

$$j_s = \max\{j \mid u+1 \leq j < j_{s+1},\ B_{v+1} - \sum_{m=s}^{k} X_{i_m} > B_j\},$$

with boundary condition $j_{k+1} = v + 1$.

$\mathbf{X}_n$ being optimal. We conclude that $i_s = j_s$ for all $s = 2, \ldots, k$. □

Theorem 3.5 implies that, given the number of production periods of a well-formed subplan for the periods $u+1, \ldots, v$, computing the optimal timing and sizing of the lot sizes requires $O(v-u)$ basic operations. Since an optimal n-period plan exists that consists only of well-formed subplans, it is sufficient to enumerate over all $(v-u)$ possible values for k and to choose a subplan with lowest cost. In this way, an optimal subplan can be obtained using $O((v-u)^2)$ basic operations, and the computation of all arc costs requires $O(n^4)$ basic operations. Using the forward recursion (2.4), we obtain an algorithm for the n-period problem with the purchase cost structure that requires $O(n^4)$ basic operations.

3.3.3 Bounds on the length of an optimal subplan

The following result provides us with an upper bound on the length of a subplan of an optimal n-period plan.

Theorem 3.6. *Let $\mathbf{X}_{uv}$ be a subplan of an optimal n-period plan $\mathbf{X}_n$. Then*

$$v - u \leq \left\lfloor \frac{r}{h} \right\rfloor + 1 < \infty.$$

Proof. Let $\delta = \min\{X_{u+1}, \min_{u+1 \leq t < v} I_t\}$. Then, from the definition of subplan, it is obvious that $\delta > 0$. Furthermore, let $\mathbf{X}'_n = \mathbf{X}_n - \delta\mathbf{e}_{u+1} + \delta\mathbf{e}_v$. One easily verifies that $\mathbf{X}'_n$ is a n-period plan. Since $\mathbf{X}_n$ is optimal, the cost of $\mathbf{X}'_n$ must be greater than or equal to the cost of $\mathbf{X}_n$. Subtracting the equal cost components gives the inequality

$$\begin{aligned} v - (u+1) &\leq \frac{(P(X_v + \delta) - P(X_v)) - (P(X_{u+1}) - P(X_{u+1} - \delta))}{h\delta} \\ &\leq \frac{q-p}{h}, \end{aligned}$$

which, together with $v-u$ being integer, completes the proof of the first inequality. The finiteness of $\lfloor r/h \rfloor + 1$ is obvious from $h > 0$. □

3.3.4 Economic order quantity

In this subsection we derive a horizon-selection rule based on the economic order quantity as proposed in Section 2.4.3. The corresponding variable-horizon policy is used as a reference in our empirical studies. Next we analyze the discrete time version of the infinite-horizon lot-sizing model with constant demand described in Section 2.4.3.

Let D be the demand rate in units per period. The analysis is different from the analysis for the two former models. Suppose we use a fixed lot size of Q. Then it consists of an in-house production part $R(Q)$ and a purchase part $S(Q)$.

Proposition 3.4 implies that it is not optimal to keep purchased goods in inventory. So either $Q = S(Q) = D$, and demand is satisfied directly by purchasing, or $Q = R(Q)$, and demand is satisfied by setting up production. In case $Q = S(Q)$, i.e., $0 \leq Q \leq B$, there is no inventory and the total relevant cost per period are given by

$$\mathrm{TRC}(Q) = qD,$$

which is independent of Q. In case $Q = R(Q)$, that is, $B < Q \leq C$, the average inventory level $\bar{I}$ is equal to $Q/2$ and the total relevant cost per period are given by

$$\mathrm{TRC}(Q) = \frac{hQ}{2} + \frac{DS}{Q} + pD.$$

Let

$$r_1 = \sqrt{\frac{2DS}{h}}$$

and

$$Q_1 = \begin{cases} B & \text{if } r_1 \leq B \\ r_1 & \text{if } B < r_1 \leq C \\ C & \text{if } r_1 > C. \end{cases}$$

Then, in case $\mathrm{TRC}(Q_1) > qD$, no economic order quantity exists and all demand is directly satisfied by purchasing from the outside supplier. Otherwise, in case $\mathrm{TRC}(Q_1) \leq qD$ we have an economic order quantity

$$Q^* = n^* D = Q_1. \tag{3.7}$$

The horizon-selection rule selects the optimization horizon equal to the order cycle n^* rounded to the nearest integer greater than zero. For D we take the average demand level calculated over the data horizon, which is recalculated every time new information comes in.

4

Multi-layered perceptrons for statistical classification

A multi-layered perceptron consists of a layered network of elementary nodes that are linked through weighted connections. These nodes represent computational units, which are capable of performing a simple computation that consists of a summation of the weighted inputs of the node, followed by the addition of a constant called the bias weight, and the application of a *response function.* The result of the computation of a unit gives the output of the corresponding node. The nodes are arranged in layers with connections between the inputs of the network and the nodes in the first layer and between subsequent layers only.

A history of Perceptrons. Early work on multi-layered perceptrons dates back to McCulloch & Pitts [1943], who studied a simple mathematical model for the behavior of a single neuron in a biological nervous system consisting of a simple processing unit. Single-layered networks of such units were studied by Widrow & Hoff [1960] under the name *adalines* and by Rosenblatt [1958] and Rosenblatt [1962] who called them *perceptrons.* Multi-layered perceptrons can be viewed as an extension of these single-layered networks. Rosenblatt [1962] showed that perceptrons can be used for adaptive pattern classification, by introducing a learning algorithm called the perceptron convergence procedure and by proving his famous perceptron convergence theorem. This theorem states that the perceptron convergence procedure finds the connection weights of a single-layered perceptron that solves a given pattern classification problem if such a solution exists. Among others, Minsky & Papert [1969] demonstrated the limitations of single-layered perceptrons by showing that they can only classify sets that are linearly separable. Minsky & Papert [1969] suggested the use of multi-layered perceptrons to overcome these difficulties. However, due to the lack of a convergence procedure for this type of networks and the convincing argument of Minsky & Papert [1969] for single-layered perceptrons, interest in perceptrons dropped to a modest level. In the last decade, multi-layered perceptrons regained interest due to the discovery of the *back-propagation algorithm*, which enabled an efficient evaluation of derivatives in multi-layered perceptrons. This algorithm is the backbone of many learning algorithms. Although similar ideas had been developed earlier by Werbos [1974] and Parker [1985], it was the paper by Rumelhart, McClelland & Williams [1986] that introduced the back-propagation algorithm to a broader audience.

In recent years multi-layered perceptrons have emerged as a useful neural network model with applications in many fields. The majority of these applications are in pattern recognition and classification. For examples of such applications we refer to the work of Huang & Lippmann [1988], Maren, Harston & Pap [1990], Michie, Spiegelhalter & Taylor [1994], and Ripley [1994].

The remainder of this chapter is outlined as follows. In Section 4.1 we introduce multi-layered perceptrons and we show that multi-layered perceptrons can be viewed as mappings. Their network mapping capabilities are discussed in Section 4.2. In Section 4.3 we address the supervised learning problem, i.e., the problem of constructing a multi-layered perceptron on the basis of learning examples. Supervised learning in the presence of noise or uncertainty is analyzed in Section 4.4. Section 4.5 is devoted to the subject of generalization. Finally, in Section 4.6, we introduce statistical classification and discuss the use of multi-layered perceptrons in this area.

4.1 Network mappings

In a multi-layered perceptron the inputs of units in the first layer correspond to the *inputs* of the network, while the inputs of the units in a higher layer are the outputs of the units in the preceding layer. The outputs of the units in the highest layer determine the *outputs* of the network, and this is called the *output layer*. Units that are not output units are called *hidden units*, and the corresponding layers are called *hidden layers*. The *topology* of a multi-layered perceptron is determined by the number of inputs, the number of layers, and the number of units per layer.

Let the term m-layered perceptron (mLP) refer to a multi-layered perceptron with m layers of computational units or, equivalently, m layers of weights. Below we define the mapping represented by a general mLP with $n^{(0)}$ inputs and $n^{(l)}$ units in layer l for $l = 1, \ldots, m$. Let x_i denote the ith input and let $y_j^{(l)}$ denote the output of unit j in layer l. We assume that within layer l each unit has the same response function given by $f^{(l)}(\cdot)$.

The output of the jth unit in the first layer is obtained by first computing a weighted linear combination of the $n^{(0)}$ inputs, and adding a bias weight, which yields

$$a_j^{(1)} = \sum_{i=1}^{n^{(0)}} w_{ji}^{(1)} x_i + w_{j0}^{(1)}, \tag{4.1}$$

where $w_{ji}^{(1)}$ denotes the weight of the connection between input i and first layer unit j and $w_{j0}^{(1)}$ denotes the *bias weight* of first layer unit j. Sometimes $-w_{j0}^{(1)}$ is called a *threshold*. The bias weight can be modeled as an ordinary weight by adding an extra input with fixed value $x_0 = 1$. Rewriting (4.1) then yields

$$a_j^{(1)} = \sum_{i=0}^{n^{(0)}} w_{ji}^{(1)} x_i. \tag{4.2}$$

The output of first layer unit j is then obtained by applying the response function $f^{(1)}(\cdot)$ and is given by

$$y_j^{(1)} = f^{(1)}(a_j^{(1)}). \tag{4.3}$$

The output of the kth unit in layer $l + 1$ is obtained by first computing a weighted linear combination of the n^l inputs from layer l, and adding a bias weight, which yields

$$a_k^{(l+1)} = \sum_{j=1}^{n^{(l)}} w_{kj}^{(l+1)} y_j^{(l)} + w_{k0}^{(l+1)}, \tag{4.4}$$

where $w_{kj}^{(l+1)}$ denotes the weight of the connection between unit j in layer l and unit k in layer $l + 1$, and $w_{k0}^{(l+1)}$ denotes the bias weight of unit k in layer $l + 1$. Again, this bias weight can be modeled as an ordinary weight by adding an extra unit 0 in layer l with a fixed output value $y_0^{(l)} = 1$. Rewriting (4.4), then yields

$$a_k^{(l+1)} = \sum_{j=0}^{n^{(l)}} w_{kj}^{(l+1)} y_j^{(l)}. \tag{4.5}$$

The output of unit k in layer $l+1$ is then obtained by applying the response function $f^{(l+1)}(\cdot)$ and is given by

$$y_k^{(l+1)} = f^{(l+1)}(a_k^{(l+1)}). \tag{4.6}$$

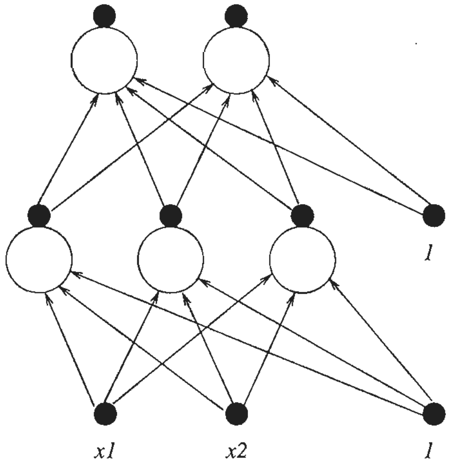


Figure 4.1. *An example of a 2LP with two inputs, three hidden units, and two output units. The bias weights are shown as weights from extra inputs having a fixed value* 1.

We refer to the above recursive computation of outputs from inputs as *forward propagation.* An example of a 2LP is shown in Figure 4.1. This network has 2 inputs, 3 hidden units, and 2 output units. Combining (4.1), (4.4), and (4.6), we obtain an explicit expression for the complete mapping represented by this 2LP. This expression is given by

$$y_k^{(2)} = f^{(2)}\left(w_{k0}^{(2)} + \sum_{j=1}^{3} w_{kj}^{(2)} f^{(1)}\left(w_{j0}^{(1)} + \sum_{i=1}^{2} w_{ji}^{(1)} x_i\right)\right) \quad \text{for } k = 1, 2. \tag{4.7}$$

From this example it is clear that a multi-layered perceptron can be viewed as a mapping $f : \mathbb{R}^M \to \mathbb{R}^K$, where M denotes the number of inputs and K the number of output units. Mapping f is called the *network mapping.* In the next section we address the question what network mappings can be realized by multi-layered perceptrons, i.e., we investigate the *network mapping capabilities* of multi-layered perceptrons.

4.2 Network mapping capabilities

In recent years the capabilities of multi-layered perceptrons to realize mappings have been investigated by many authors. Network mapping capabilities of multi-layered perceptrons can be subdivided into *exact capabilities* and *approximation capabilities.* Exact capabilities of multi-layered perceptrons are determined by the set of mappings that can be realized as a network mapping. Approximation capabilities of multi-layered perceptrons are determined by the extend to which they can approximate arbitrary mappings.

Below, we discuss some results on the network mapping capabilities for the type of multi-layered perceptrons under consideration. We concentrate on multi-layered perceptrons with one output unit. The corresponding results for multi-layered perceptrons with more than one output unit can be easily deduced from this simplified case as was shown by Hornik, Stinchcombe & White [1989]. We introduce the notation (α, β)-mLP to denote an m-LP with response function α for the hidden units and response function β for the output units. For reasons of convenience (α, α)-mLP is abbreviated to α-mLP.

First, we discuss the exact capabilities of θ-mLPs, where $\theta : \mathbb{R} \to \{0, 1\}$ denotes the *hard-limiting response function* given by

$$\theta(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases} \tag{4.8}$$

Second, we discuss the approximation capabilities of θ-mLPs and σ-mLPs, where $\sigma : \mathbb{R} \to [0, 1]$ denotes the *logistic sigmoid response function* given by

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \tag{4.9}$$

The term sigmoid refers to "S-shaped". The logistic sigmoid response function can be seen as a differentiable approximation of the hard-limiting response function of (4.8). Finally, we discuss the approximation capabilities of (θ, λ)-mLPs and (σ, λ)-mLPs, where $\lambda : \mathbb{R} \to \mathbb{R}$ denotes the *linear response function* given by

$$\lambda(x) = x. \tag{4.10}$$

4.2.1 Exact capabilities

θ-mLPs can be seen as *classification* devices that classify input vectors to one of a finite number of *classes* and the exact capabilities of θ-mLPs can be studied by considering their classification capabilities. For our discussion of the classification capabilities of θ-mLPs we follow the work of Zwietering [1994]. A subset $V \subseteq \mathbb{R}^M$ is said to be *classified* by a θ-mLP with M inputs and one output unit, if its network mapping $f : \mathbb{R}^M \to \{0, 1\}$ satisfies

$$f(x) = \begin{cases} 1 & \text{if } x \in V \\ 0 & \text{if } x \notin V. \end{cases}$$

The *decision region* $\mathcal{J}(f)$ of this θ-mLP is defined by

$$\mathcal{J}(f) = \{x \in \mathbb{R}^M \mid f(x) = 1\}.$$

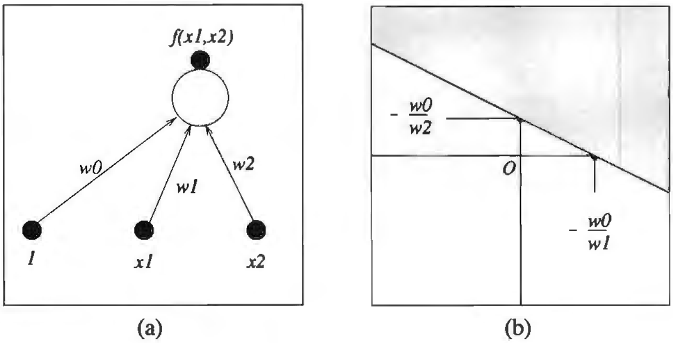


Figure 4.2. *(a) An example of a θ-1LP with two inputs and one output unit. The network mapping $f : \mathbb{R}^2 \to \{0, 1\}$ is given by $f(x_1, x_2) = \theta(w_1x_1 + w_2x_2 + w_0)$. (b) The shaded region corresponds to the decision region of the θ-1LP given by $\mathcal{J}(f) = \{(x_1, x_2) \in \mathbb{R}^2 \mid w_1x_1 + w_2x_2 + w_0 \geq 0\}$.*

Thus V is classified by a θ-mLP with network mapping f if the decision region $\mathcal{J}(f)$ equals V. In this way, input vectors $\mathbf{x} \in \mathbb{R}^M$ belong to one of two classes, i.e., either they are in V or they are not in V. To illustrate this, in Figure 4.2 we give an example of a θ-1LP with two inputs and one output unit. One easily verifies that its decision region is a closed affine half space in $\mathbb{R}^2$. The two classes are separated by a line in $\mathbb{R}^2$, called the *decision boundary.* The orientation of this decision boundary and its distance to the origin can be changed by adjusting the weights w_1 and w_2 and the bias weight w_0.

In order to specify the classification capabilities of θ-mLPs we need to define the following collections of subsets of $\mathbb{R}^M$. Let V^* denote the complement of V.

Definition 4.1. The collection of closed affine halfspaces H_M, the collection of polyhedra P_M, the collection of open and closed affine halfspaces $\tilde{H}_M$, the collection of pseudo polyhedra $\tilde{P}_M$, and the collection of finite unions of pseudo polyhedra $\tilde{U}_M$ are defined by

$$\begin{aligned}
H_M &= \{V \subseteq \mathbb{R}^M \mid \exists \mathbf{a} \in \mathbb{R}^M \backslash \{\mathbf{0}\}\ \exists b \in \mathbb{R} \ :\ V = \{\mathbf{x} \in \mathbb{R}^M \mid \mathbf{a} \cdot \mathbf{x} + b \geq 0\}\},\\
P_M &= \{V \subseteq \mathbb{R}^M \mid V = \textstyle\bigcap_{i=1}^K W_i,\ W_i \in H_M,\ K \in \mathbb{N}_0\},\\
\tilde{H}_M &= \{V \subseteq \mathbb{R}^M \mid V \in H_M \vee V^* \in H_M\},\\
\tilde{P}_M &= \{V \subseteq \mathbb{R}^M \mid V = \textstyle\bigcap_{i=1}^K W_i,\ W_i \in \tilde{H}_M,\ K \in \mathbb{N}_0\}, \text{ and}\\
\tilde{U}_M &= \{V \subseteq \mathbb{R}^M \mid V = \textstyle\bigcup_{i=1}^L V_i,\ V_i \in \tilde{P}_M,\ L \in \mathbb{N}_0\},
\end{aligned}$$

respectively. □

A polyhedron $V \in P_M$ is the intersection of a finite collection of closed affine halfspaces. Therefore, all its bounds, usually called *faces*, belong to the set. A pseudo polyhedron $V \in \tilde{P}_M$ is the intersection of a finite collection of closed or open affine halfspaces and can have faces belonging to the set and faces belonging to its complement V^*. The collection $\tilde{U}_M$ can be viewed as the collection of all subsets of $\mathbb{R}^M$ that have a finite number of piece-wise linear bounds. The following result specifies the classification capabilities of θ-mLPs.

Theorem 4.1. [Zwietering, 1994]. *Let $C_{m,M}$ denote the collection of subsets of $\mathbb{R}^M$ that can be classified with a θ-mLP with M inputs and one output unit. Then*

(i) $C_{1,M} = H_M \cup \{\emptyset, \mathbb{R}^M\}$,

(ii) $\tilde{P}_M \subset C_{2,M} \subset \tilde{U}_M$, *and*

(iii) $C_{m,M} = \tilde{U}_M$ *for* $m \geq 3$.

□

These results appeared in the papers by Zwietering, Aarts & Wessels [1991] and Zwietering, Aarts & Wessels [1992]. Gibson & Cowan [1990] and Gibson [1993] obtained related results. Zwietering [1994] derived a detailed characterization of $C_{2,M}$ by giving necessary and sufficient conditions for a subset to be classifiable with a θ-2LP. For more details and examples of subsets that can or cannot be classified by a θ-2LP we refer to the work of Zwietering, Aarts & Wessels [1992], Zwietering [1994], and Gibson & Cowan [1990].

4.2.2 Approximation capabilities

In the discussion of the approximation capabilities of multi-layered perceptrons we distinguish between the task of *classification*, in which input vectors have to be assigned to classes and the task of *regression*, in which continuous variables have to be predicted given input vectors. Below we discuss the approximation capabilities of θ-mLPs and σ-mLPs for classification and the approximation capabilities of (θ, λ)-mLPs, (σ, λ)-mLPs, and σ-mLPs for regression.

Classification. Lippmann [1987] showed that θ-mLPs, with $m \geq 3$, can approximate any decision boundaries with arbitrary accuracy, provided the number of hidden units is sufficiently large. Despite these capabilities the practical use of θ-mLPs for classification is limited. The main reason for this is that the hard-limiting response function is inappropriate for use in learning algorithms; see also Section 4.3. In such cases σ-mLPs can be used for classification by rounding the outputs to the nearest integer or by implementing a winner-takes-it-all mechanism. Bishop [1995] shows that σ-mLPs with $m \geq 2$ can approximate any decision boundary with arbitrary accuracy, provided the number of hidden units is sufficiently large.

Regression. Among others, Hornik, Stinchcombe & White [1989] showed that (θ, λ)-mLPs with $m \geq 2$ can approximate arbitrarily well any continuous function, provided the number of hidden units is sufficiently large. The authors also show that the same result holds for (σ, λ)-mLPs with $m \geq 2$; see also the work of Barron [1991], Cybenko [1989], Funahashi [1989], and Hornik [1991].

The regression capabilities of σ-mLPs can be characterized using the above results for (σ, λ)-mLPs. One easily verifies that a unit with a logistic sigmoid response function can approximate a unit with a linear response function with arbitrary accuracy by rescaling its incoming and outgoing weights [Bishop, 1995]. As a result of that, σ-mLPs, with $m \geq 2$, can approximate arbitrarily well any continuous function with range [0, 1], provided the number of hidden units is sufficiently large. We remark that, by choosing a suitable transformation, any bounded continuous function can be transformed to a function with range [0, 1].

Numerous alternative response functions exist with similar approximation capabilities. Hornik [1991] stressed that it is the multi-layered feed-forward topology that gives multi-layered perceptrons the potential of being *universal approximators*, rather than the specific choice of the response function, by showing that any continuous, bounded, and non-constant response function is sufficient. An extensive treatment of this area is beyond the scope of this thesis; elaborate discussions are provided by Ellacott [1994], Light [1992], Mason & Parks [1992], and Zwietering [1994].

A final remark addresses the profit of using mLPs with $m > 2$, considering the fact that we can approximate any mapping with arbitrary accuracy with a 2LP. One possibility is that by using more hidden layers we obtain more efficient approximation in the sense that the same accuracy is obtained with fewer weights. So far, however, there are hardly any results on this subject.

4.3 Supervised learning

From the results presented in the previous section we conclude that multi-layered perceptrons have quite impressive network mapping capabilities, but no construction methods were provided for finding appropriate network topologies. When applying multi-layered perceptrons to a certain task one has to choose a suitable network topology and weights such that the network performs the task accurately. If the underlying task is well-understood and can be analyzed properly, these parameter values can be derived directly from the problem formulation. Usually, however, this is not the case and the only available information consists of examples of the desired input-output behavior. In such cases one can apply supervised learning techniques. In this section we concentrate on the problem of determining suitable weights for a multi-layered perceptron with a fixed network topology on the basis of a finite set of examples. The problem of choosing a suitable network topology is addressed in Section 4.5 in the context of generalization.

4.3.1 Problem formulation

Given is a multi-layered perceptron with M inputs and K outputs and a finite set $\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_N\}$ of *learning examples*, where $\mathbf{s}_n = (\mathbf{x}_n, \mathbf{t}_n)$ with $n = 1, \ldots, N$, $\mathbf{x}_n \in \mathbb{R}^M$ represents an input vector for the multi-layered perceptron, and $\mathbf{t}_n \in \mathbb{R}^K$ represents the corresponding desired output or *target vector*. We refer to $\mathcal{S}$ as the *learning set*. The network is supposed to be already completely specified apart from the weights. For reasons of convenience we group all weights in the network to form a single weight vector $\mathbf{w}$. Let $\mathbf{y}(\mathbf{x}_n; \mathbf{w})$ denote the output vector of the multi-layered perceptron with weight vector $\mathbf{w}$ on input of input vector $\mathbf{x}_n$. Then the *supervised learning problem* is defined as to find a weight vector $\mathbf{w}$ such that the difference between the target vector $\mathbf{t}_n$ and the output vector $\mathbf{y}(\mathbf{x}_n; \mathbf{w})$ is minimal for all $n = 1, \ldots, N$. Usually, the difference is measured by an appropriate error function. Let $E = E(\mathbf{w})$ denote the *total error function*, which can be written as a sum, over all examples in the learning set, of the error $E_n(\mathbf{w})$ of each individual example, i.e.,

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}), \tag{4.11}$$

where $n = 1, \ldots, N$ labels the examples in the learning set. Many different error functions have been proposed in the literature and the selection of an appropriate one is usually problem-dependent [Bishop, 1995; Xu, Klasa & Yuille, 1992]. The most commonly used error function is the *sum-of-squares error function* given by

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}), \tag{4.12}$$

where

$$E_n(\mathbf{w}) = \| \mathbf{y}(\mathbf{x}_n; \mathbf{w}) - \mathbf{t}_n \|^2 = \sum_{k=1}^{K} (y_k(\mathbf{x}_n; \mathbf{w}) - t_{nk})^2,$$

and where $\| \cdot \|$ denotes the Euclidean norm.

Remark that, although the capabilities of multi-layered perceptrons to reproduce target vectors given input vectors may be useful in itself, usually the purpose is to *generalize*, i.e., to reproduce target vectors given input vectors that are outside the learning set. Generalization is subject of Section 4.5.

4.3.2 Solution approaches

Supervised learning involves the search for a weight vector $\mathbf{w}$ such that the total error function E is minimal. If the chosen response functions are differentiable, supervised learning can be viewed as the unconstrained minimization of a differentiable function of many variables. Such problems have been widely studied, and many of the conventional approaches in this area can directly be applied to supervised learning with multi-layered perceptrons; see for example the standard textbook on global optimization techniques by Fletcher [1987].

In the simplest case of a λ-1LP with the sum-of-squares error function (4.12), E is a convex function of $\mathbf{w}$ having a single minimum. This minimum can be found by solving a set of coupled linear equations [Bishop, 1995]. For more general networks, in particular those with more than one layer, E is typically a non-linear function of $\mathbf{w}$ and local minima may exist. As a consequence of this, it is in general impossible to find closed-form solutions for these minima. Instead, supervised learning approaches, which are called *learning algorithms*, typically minimize E by an iterative procedure in which the weights are adjusted in a sequence of steps until some stop criterion is met. At each such step we can distinguish between two distinct stages. In the first stage, the derivatives of E with respect to the individual weights must be evaluated at the current values of the weights. Until the discovery of the *back-propagation algorithm*, the computationally efficient evaluation of these derivatives was considered as a major problem. In the second stage, these derivatives are used to compute the weight adjustments. The most commonly used weight adjustment schemes involve some kind of gradient descent.

Although there are various learning algorithms that show a good performance on a wide range of applications, they all typically require problem-specific tuning, making a sound comparison of the different learning algorithms cumbersome. Therefore, no single best universal learning algorithm can be designated. In the sequel we discuss the back-propagation algorithm and gradient-descent based weight adjustment schemes. Furthermore, we discuss weight initialization and stop criteria. Elaborate overviews of the literature on learning algorithms are provided by Bishop [1995], Hertz, Krogh & Palmer [1991], Xu, Klasa & Yuille [1992].

4.3.3 Error back-propagation

Below we give a derivation of the back-propagation algorithm. We use the notation introduced in Section 4.1 for a general mLP. It is assumed that we are given an arbitrary fixed network topology, with arbitrary continuous, differentiable response functions, and an arbitrary differentiable error function.

We recall that the back-propagation algorithm is a procedure for the evaluation of the derivatives of the total error function E with respect to the weights $w_{ji}^{(l)}$. Using

(4.11) these derivatives can be expressed as sums over the learning examples of the derivatives of the error functions E_n. Consequently the learning examples $(\mathbf{x}_n, \mathbf{t}_n)$ may be considered one at a time. Thus we may concentrate on the calculation of the derivative of E_n with respect to weight $w_{ji}^{(l)}$. We suppose that we have supplied the input vector $\mathbf{x}_n$ to the network and calculated the outputs of all units by successive application of (4.3) and (4.6). First we note that E_n depends on $w_{ji}^{(l)}$ only through $a_j^{(l)}$, which yields

$$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = \frac{\partial E_n}{\partial a_j^{(l)}} \frac{\partial a_j^{(l)}}{\partial w_{ji}^{(l)}}. \tag{4.13}$$

Let $\delta_j^{(l)}$ be defined as

$$\delta_j^{(l)} = \frac{\partial E_n}{\partial a_j^{(l)}}. \tag{4.14}$$

For notational reasons we define $y_i^{(0)} = x_{ni}$. Using this notation and (4.1) and (4.4) it follows that

$$\frac{\partial a_j^{(l)}}{\partial w_{ji}^{(l)}} = y_i^{(l-1)}. \tag{4.15}$$

Substituting (4.14) and (4.15) into (4.13) yields

$$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} y_i^{(l-1)}. \tag{4.16}$$

The calculation of the $\delta_j^{(l)}$'s can be executed recursively as follows. Starting at the output layer, for output unit k in layer m we have

$$\delta_k^{(m)} = \frac{\partial E_n}{\partial a_k^{(m)}} = \frac{\partial E_n}{\partial y_k^{(m)}} \frac{\partial y_k^{(m)}}{\partial a_k^{(m)}}, \tag{4.17}$$

which, using (4.6), can be simplified to

$$\delta_k^{(m)} = f'^{(m)}(a_k^{(m)}) \frac{\partial E_n}{\partial y_k^{(m)}}, \tag{4.18}$$

where $f'^{(m)}$ denotes the derivative of the response function $f^{(m)}$. For hidden unit j in layer l with $1 \leq l < m$, we note that E_n only depends on $a_j^{(l)}$ through $a_k^{(l+1)}$ for $k = 1, \ldots, n^{(l+1)}$, which yields

$$\delta_j^{(l)} = \frac{\partial E_n}{\partial a_j^{(l)}} = \sum_{k=1}^{n^{(l+1)}} \frac{\partial E_n}{\partial a_k^{(l+1)}} \frac{\partial a_k^{(l+1)}}{\partial a_j^{(l)}}. \tag{4.19}$$

1. Apply input vector $\mathbf{x}_n$ to the network and use forward propagation to find all outputs $y_j^{(l)}$ of units in the network.
2. Evaluate the errors $\delta_k^{(m)}$ for the output units using (4.17).
3. Back-propagate the errors $\delta_k^{(m)}$ using the error back-propagation rule (4.20) to obtain $\delta_j^{(l)}$ for each hidden unit.
4. Use (4.16) to evaluate the derivatives.

Figure 4.3. *The back-propagation algorithm.*

This can be simplified using (4.4) and (4.14) to what is known as the *error back-propagation rule* given by

$$\delta_j^{(l)} = f'^{(l)}(a_j^{(l)}) \sum_{k=1}^{n^{(l+1)}} \delta_k^{(l+1)} w_{kj}^{(l+1)}. \tag{4.20}$$

With this rule we can recursively calculate the errors of units in the hidden layers from the errors of units in the output layer, where the errors are represented by the $\delta_j^{(l)}$'s. Remark that this holds for any network with feed-forward topology. The back-propagation algorithm for evaluating the derivatives of E_n with respect the weight $w_{ji}^{(l)}$ is summarized in four steps in Figure 4.3.

The derivative of the total error function E with respect to the weight $w_{ji}^{(l)}$ can be determined by summing the derivatives of E_n for all learning examples which results in

$$\frac{\partial E}{\partial w_{ji}^{(l)}} = \sum_{n=1}^{N} \frac{\partial E_n}{\partial w_{ji}^{(l)}}. \tag{4.21}$$

We end our discussion of the back-propagation algorithm with a remark on its computational efficiency. Let W denote the total number of weights. Then one can easily verify that applying the above steps to evaluate the W derivatives of E_n requires $O(W)$ operations. Note that this is quite efficient, since evaluation of the derivatives using their explicit formulas using forward propagation would require $O(W^2)$ operations. The evaluation of the derivatives of E thus requires $O(NW)$ operations.

The above derivation of the back-propagation algorithm allows for general forms of the error function, the response functions and the network topology. In our experiments in Chapter 6 and Chapter 7 we use a sum-of-squares error function. Moreover, in each unit, we use the logistic sigmoid response function, which has the convenient property that its derivative can be expressed in terms of the response function itself as

$$\sigma'(x) = \sigma(x)(1 - \sigma(x)). \tag{4.22}$$

This enables an efficient software implementation of the calculation of the derivative. For derivations of the back-propagation algorithm for particular combinations of error functions and response functions we refer to the textbooks by Bishop [1995], Hertz, Krogh & Palmer [1991] and Rumelhart, McClelland & Williams [1986].

4.3.4 Weight adjustment schemes

In this subsection we discuss gradient-descent based weight adjustment schemes for the minimization of E with respect to the weight vector $\mathbf{w}$. First, we introduce two basic weight adjustment schemes called *batch learning* and *sequential learning* and discuss their convergence properties. Since local minima may exist, we distinguish between *local convergence*, i.e., convergence to a local minimum, and *global convergence*, i.e., convergence to a global minimum. After that, we extend the two basic schemes by adding a *momentum term.*

Batch learning. In batch learning, we start with an initial guess for $\mathbf{w}$ and we update the weights repeatedly in the direction in which the total error function decreases most rapidly, i.e., in the direction of the negative gradient. The corresponding weight adjustment scheme equals

$$\Delta\mathbf{w} = -\eta\nabla_{\mathbf{w}}E = -\eta\sum_{n=1}^{N}\nabla_{\mathbf{w}}E_n, \tag{4.23}$$

where $\mathbf{w}$ denotes the weight vector, $\nabla_{\mathbf{w}}E_n$ denotes the gradient of E_n with respect to $\mathbf{w}$, and $\eta \in \mathbb{R}^+$ is called the *learning rate.* Note that the weights are adjusted after all learning examples in the learning set have been presented to the network.

Under suitable conditions, the batch learning weight update scheme converges to a local minimum of E. The value for η can be fairly critical, since a too small value results in slow convergence, whereas a too large value results in divergent oscillations. Furthermore, we remark that once trapped in a local minimum there is no escape and global convergence cannot be assured.

Sequential learning. In sequential learning, the weights are adjusted in the direction of the negative gradient of the error function for one learning example at a time. The corresponding weight adjustment scheme equals

$$\Delta\mathbf{w} = -\eta\nabla_{\mathbf{w}}E_n, \tag{4.24}$$

where the learning examples in the learning set are selected randomly or considered in sequence.

Convergence results for sequential learning require that the learning rate is made to decrease at each iteration according to some *cooling schedule*, given by η_t with $t = 1, 2, \ldots$. It has been frequently mentioned in the literature that the advantage of sequential learning over batch learning is that, due to its random behavior, sequential learning can escape from local minima. Furthermore, an analogy between sequential learning and simulated annealing [Kirkpatrick, Gelatt & Vecchi, 1983], where the temperature is controlled by the learning rate η_t, has been mentioned [Bottou, 1991]. Heskes, Slijpen & Kappen [1993] elaborate on the analogy with simulated annealing by showing that the fastest possible cooling schedule for η_t that guarantees convergence to a global minimum is exponentially slow. In practice, however, a constant value of η_t generally leads to much better results, although the guarantee of global convergence is lost.

Several authors claimed that sequential learning often yields the best results, especially for complex problems with many local minima [Barnard, 1992; Heskes & Wiegerinck, 1996]. Schiffmann, Joost & Werner [1992] compare the sequential version of gradient descent using a fixed learning rate with a large number of learning algorithms on a complex real world benchmark. They make a distinction between global weight adjustment schemes, in which the same learning rate is used for all weights, and local weight adjustment schemes, in which each weight has its own learning rate that is typically adapted during execution. In this study, sequential learning with a fixed learning rate was the best global weight adjustment schemes; only some of the local weight adjustment schemes gave slightly better results.

We end our discussiôn with some additional advantages of sequential learning over batch learning. Since weights are adjusted after every presentation, sequential learning is memory efficient. Moreover, in case of large redundant learning sets, sequential learning runs faster [Bishop, 1995; Ellacott, 1994]. Finally, sequential learning is naturally suitable for parallel implementation.

Momentum. For both batch learning and sequential learning a value of η that is too large may result in divergent oscillations. Conversely, if η is too small, the computation times may become prohibitive. The optimal value of η typically changes during the search. One commonly used remedy is to add a momentum term, which adds a weighted average of the previous gradients to the current gradient. To illustrate its effect we consider batch learning with momentum, which is given by

$$\Delta \mathbf{w}(t) = -\eta \nabla_{\mathbf{w}} E \,|_{\mathbf{w}(t)} + \mu \Delta \mathbf{w}(t-1), \tag{4.25}$$

where t refers to the tth weight adjustment. This is equivalent to

$$\Delta \mathbf{w}(t) = -\eta \sum_{s=0}^{t} \mu^s \nabla_{\mathbf{w}} E \,|_{\mathbf{w}(t-s)}, \tag{4.26}$$

and can be seen as applying exponential smoothing to (4.23). In the case that the subsequent gradients are approximately the same we have

$$\Delta \mathbf{w} = -\eta \nabla_{\mathbf{w}} E\{1 + \mu + \mu^2 + \dots\} = -\frac{\eta}{1-\mu} \nabla_{\mathbf{w}} E, \tag{4.27}$$

resulting in an increase of the effective learning rate from η to $\eta/(1-\mu)$. It is obvious that μ must be chosen such that $0 \leq \mu < 1$. On the other hand, if the subsequent gradients oscillate, successive fluctuations cancel to obtain a long term trend with an effective learning rate close to η Bishop [1995]. The corresponding weight adjustment scheme for sequential learning with momentum is as follows. After presentation of the learning example labeled n, the weights are adjusted according to

$$\Delta \mathbf{w}(t) = -\eta \nabla_{\mathbf{w}} E_n \,|_{\mathbf{w}(t)} + \mu \Delta \mathbf{w}(t-1). \tag{4.28}$$

An extra effect of using a momentum term for sequential learning is that one obtains an approximation of the total gradient. We refer to the work of Wiegerinck, Komoda & Heskes [1994] for convergence results of sequential learning with momentum.

4.3.5 Stop criteria

Possible stop criteria are to stop after (*i*) a fixed number of iterations, (*ii*) a fixed amount of computation time, or (*iii*) the error on the learning set has dropped below some prespecified level. A disadvantage of these criteria is that they ignore the network's generalization capabilities. Stop criteria that account for generalization capabilities are discussed in Section 4.5.

4.3.6 Weight initialization

Weight initialization has received relatively little attention in the literature. The most commonly employed weight initializing procedure is to choose small random values. Random values are used to avoid problems due to symmetry in weight space [Rumelhart, McClelland & Williams, 1986]. For the logistic sigmoidal response function, large absolute values of the initial weights results in small values of derivatives of the response function, which leads to small values of the gradient and consequently a flat error surface. If, on the other hand, the initial weights are too small, the logistic sigmoid response function becomes approximately linear, which may slow down learning [Bishop, 1995]. Unfortunately, there does not exist a clear definition of small, and fine-tuning is needed for a particular problem at hand. Recently, a number of alternative weight initialization procedures have been proposed, which typically use prior, problem-specific knowledge. Smyth [1992] used the decision boundaries of a K-nearest-neighbors classifier [Duda & Hart, 1973] as initialization of the first hidden layer weights. Wessels & Barnard [1992] developed a procedure in which the weights are initialized such that the hidden unit decision boundaries are uniformly oriented in feature space. Denoeux & Lengellé [1993] used prototype vectors.

4.4 Learning in a statistical perspective

As we have stressed before, the real purpose of supervised learning is to model the process underlying the learning examples, rather than to memorize the set of learning examples. In this way, on input of an input vector outside the learning set, the best possible prediction for the corresponding output vector can be made. It is important to note that such a process may be subject to noise or may be inherently stochastic. In this section we discuss supervised learning by means of sum-of-squares error minimization in a statistical perspective.

Following Bishop [1995], we model the process that generates the learning examples by a random variable pair $(\mathbf{X}, \mathbf{T})$ defined on $\Omega_{\mathbf{X}} \times \Omega_{\mathbf{T}}$, where $\Omega_{\mathbf{X}} \subseteq \mathbb{R}^M$ denotes the input space and $\Omega_{\mathbf{T}} \subseteq \mathbb{R}^K$ denotes the *target space*. The process can then be characterized by the joint probability density function of $\mathbf{X}$ and $\mathbf{T}$ denoted by $f_{\mathbf{X},\mathbf{T}}$.

Let $\{(\mathbf{U}_n, \mathbf{V}_n)\}_{n=1}^{\infty}$ be a sequence of independent and identically distributed random learning examples with common distribution $f_{\mathbf{X},\mathbf{T}}$. Given is an $\{\alpha, \beta\}$-mLP with M inputs and K outputs completely specified apart from its weights $\mathbf{w}$. Let $\mathbf{y}(\mathbf{x}; \mathbf{w})$ denote the output vector of the network on input of input vector $\mathbf{x} \in \Omega_{\mathbf{X}}$. We introduce the sequence of random variables $\{\alpha_n\}_{n=1}^{\infty}$, where α_n denotes the sum-of-squares error between $\mathbf{y}(\mathbf{U}_n; \mathbf{w})$ and $\mathbf{V}_n$ given by

$$\alpha_n = \| \mathbf{y}(\mathbf{U}_n; \mathbf{w}) - \mathbf{V}_n \|^2 = \sum_{k=1}^{K} (y_k(\mathbf{U}_n; \mathbf{w}) - V_{nk})^2, \tag{4.29}$$

where $y_k(\mathbf{U}_n; \mathbf{w})$ and V_{nk} denote the kth component of $\mathbf{y}(\mathbf{U}_n; \mathbf{w})$ and $\mathbf{V}_n$, respectively. It is obvious that random variables α_n with $n = 1, 2, \ldots$ are independent and identically distributed. Finally, we introduce the sequence of random variables $\{\epsilon_N\}_{N=1}^{\infty}$, where ϵ_N denotes the mean sum-of-squares error of the N learning examples labeled $n = 1, \ldots, N$ given by

$$\epsilon_N = \frac{1}{N} \sum_{n=1}^{N} \alpha_n. \tag{4.30}$$

The following result is obtained by applying the the strong law of large numbers [Feller, 1968].

Theorem 4.2. *Suppose $E\{\| \mathbf{y}(\mathbf{X}; \mathbf{w}) - \mathbf{T} \|^2\} < \infty$. Then, with probability 1,*

$$\lim_{N \to \infty} \epsilon_N = \epsilon,$$

where

$$\epsilon = \iint \| \mathbf{y}(\mathbf{x}; \mathbf{w}) - \mathbf{t} \|^2 f_{\mathbf{X},\mathbf{T}}(\mathbf{x}, \mathbf{t}) d\mathbf{t} d\mathbf{x}. \tag{4.31}$$

□

Using the identity $f_{\mathbf{X},\mathbf{T}}(\mathbf{x}, \mathbf{t}) = f_{\mathbf{T}}(\mathbf{t} \mid \mathbf{X} = \mathbf{x}) f_{\mathbf{X}}(\mathbf{x})$ we rewrite (4.31) as

$$\begin{aligned} \epsilon = \int \| \mathbf{y}(\mathbf{x}; \mathbf{w}) - \mathrm{E}\{\mathbf{T} \mid \mathbf{X} = \mathbf{x}\} \|^2 f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x} + \\ \int \mathrm{var}\{\mathbf{T} \mid \mathbf{X} = \mathbf{x}\} f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}, \end{aligned} \tag{4.32}$$

where $f_{\mathbf{T}}(\mathbf{t} \mid \mathbf{X} = \mathbf{x})$ denotes the conditional probability density function of $\mathbf{T}$ given $\mathbf{X}$ and $\mathrm{E}\{\cdot \mid \cdot\}$ and $\mathrm{var}\{\cdot \mid \cdot\}$ denote the conditional expectation and the conditional variance, respectively. Note that the second term of (4.32) is independent of $\mathbf{w}$ and can be neglected when minimizing ϵ with respect to $\mathbf{w}$. Furthermore, since the integrand in the first term of (4.32) is non-negative, a global minimum of ϵ with respect to $\mathbf{w}$ is attained when

$$\mathbf{y}(\mathbf{x}; \mathbf{w}) = \mathrm{E}\{\mathbf{T} \mid \mathbf{X} = \mathbf{x}\} = \int \mathbf{t} f_{\mathbf{T}}(\mathbf{t} \mid \mathbf{X} = \mathbf{x}) d\mathbf{t}. \tag{4.33}$$

For the individual network outputs $y_k(\mathbf{x}; \mathbf{w})$ this corresponds to

$$y_k(\mathbf{x}; \mathbf{w}) = \mathrm{E}\{T_k \mid \mathbf{X} = \mathbf{x}\} = \int t_k f_{T_k}(t_k \mid \mathbf{X} = \mathbf{x}) dt_k. \tag{4.34}$$

This important result states that in the limit as the size N of the learning set goes to infinity, the network mapping corresponding with a minimum of ϵ is given by the conditional expectations of the targets. We notice that this only holds under the condition that (i) $\mathrm{E}\{\| \mathbf{y}(\mathbf{X}; \mathbf{w}) - \mathbf{T} \|^2\} < \infty$ and (ii) the $\{\alpha, \beta\}$-mLP has sufficient network mapping capabilities such that there exists a choice $\mathbf{w}$ which makes the first term in (4.32) sufficiently small. On easily verifies that condition (i) is satisfied, if β is bounded and $\mathrm{E}\{T_k\}$ and $\mathrm{E}\{T_k^2\}$ are both finite for $k = 1, \ldots, K$.

For obvious reasons we call the mapping $\mathbf{x} \to \mathrm{E}\{T \mid \mathbf{X} = \mathbf{x}\}$ the *target mapping*, i.e., the mapping that has to be learned by the multi-layered perceptron. In practice, we minimize the sum-of-squares error function (4.12) on a finite learning set of realizations of $(\mathbf{X}, \mathbf{T})$. In that case, the network outputs corresponding with a minimum of error become *approximations* of the conditional expectations of the targets. In other words the network mapping corresponding with a minimum of error becomes an approximation of the target mapping. For this approximation to be good, the learning set must be sufficiently large as to approximate an infinite learning set. The problem of determining a suitable network topology and a learning set that is sufficiently large is discussed in the next section in the context of generalization.

A final remark is that Theorem 4.2 is independent of our choice of feed-forward network topology, or even of using multi-layered perceptrons at all. It only requires that the representation for the mapping $y(\mathbf{x}; \mathbf{w})$ is such that there exists a choice for $\mathbf{w}$ which makes the first term in (4.32) sufficiently small. The contribution of multi-layered perceptrons is that they provide a practical framework for finding such a representation.

4.5 Generalization

In the context of supervised learning with multi-layered perceptrons, generalization refers to the prediction of target vectors for *new* input vectors, i.e., for input vectors that are not in the learning set. Good generalization is in general not simply evident and there are a number of conditions that must be satisfied, in order to achieve good generalization. These can be divided into problem specific and model specific conditions. This section is organized as follows. In Section 4.5.1 we discuss problem specific and model specific conditions for good generalization. The problem of selecting an optimal network topology is addressed in Section 4.5.2. In Section 4.5.3, stop criteria for learning that take generalization capabilities into account are discussed. Finally, in Section 4.5.4, we discuss committees of networks.

4.5.1 Condition for good generalization

Below we discuss two classes of conditions that enable generalization in multi-layered perceptrons.

Problem specific conditions. A first necessary condition for good generalization is that the target mapping is in some sense smooth, i.e., a small change in the inputs should, most of the time, produce a small change in the outputs. Problems that do not satisfy this condition are learning the input-output behavior of pseudo-random number generators and data encryption algorithms, for instance.

A second necessary condition for good generalization is that the learning set is a sufficiently large and representative subset of what statisticians use to call the *population*, i.e., all examples that you want to generalize to. The importance of this condition becomes clear when we distinguish between two types of generalization: *interpolation* and *extrapolation*. Interpolation applies to input vectors that are surrounded by learning examples that are close in some sense; everything else is extrapolation. In particular, input vectors that lie outside the subspace spanned by the learning examples require extrapolation. Interpolation can often be done reliably, but extrapolation is usually unreliable [Barnard & Wessels, 1992; Haley & Soloway, 1992]. Hence it is important to have sufficient learning examples to avoid the need for extrapolation. In some cases preprocessing may be necessary to ensure that the problem only requires interpolation. Problems that inherently require extrapolation are inappropriate for an approach based on supervised learning with multi-layered perceptrons.

Model specific conditions. In the previous sections we focussed on the minimization of some error function E on a finite learning set by choosing appropriate values for the weights in a multi-layered perceptron of fixed topology. What remains is the problem of determining a network topology that is optimal with respect to generalization capabilities. We saw that the suitability of a certain network topology depends on its network mapping capabilities; see Section 4.2. In Section 4.4 we noted that, in case of a learning set that is sufficiently large, a necessary condition for good generalization is the requirement that the network has sufficient network mapping capabilities to represent the target mapping.

In practice, however, there is usually only limited data available so that the optimal network topology is determined by the particular learning set at hand. In such cases, we typically seek for the network topology that achieves the *best* generalization. Remark that the network topology with the best possible generalization with respect to a certain learning set may have poor generalization capabilities. The dependency between the optimal network topology and the learning set can be characterized by the so-called *bias-variance trade-off*, which we now briefly discuss. A theoretical treatment of this subject is provided by Bishop [1995]. Let f be the network mapping of a multi-layered perceptron obtained by applying some learning algorithm with a finite learning set $\mathcal{S}$. Then f can be viewed as a function of $\mathcal{S}$, i.e., $f = f(\mathcal{S})$, and the *bias* measures the extent to which the average of the network mapping over all possible learning sets $\mathcal{S}$ differs from the target mapping. The *variance* measures the sensitivity of the network mapping to the particular choice of learning set. Bias and variance are complementary quantities, and the best generalization capabilities are typically obtained as a compromise between the conflicting requirements of low bias and small variance. Networks with too little network mapping capabilities typically smooth out some of the underlying structure (high bias), while networks with too much network mapping capabilities over-fit the learning examples (high variance).

4.5.2 Learning and generalization

Next we address the problem of determining a multi-layered perceptron with the best possible generalization capabilities with respect to a given set of learning examples. Various approaches have been presented in the literature for this problem.

The most commonly used approach to select an optimal network topology for a given learning set is to execute the learning algorithm for different fixed network topologies and to select the topology with the best generalization capabilities. To overcome *over-fitting*, one typically starts with a simple network and one increases the number of hidden units.

An alternative approach is to let the learning algorithm adapt the topology of the network during execution. Possibilities are to start with a small network and to add units or to start with a large network and to prune units [Fahlman & Lebiere, 1990].

Both approaches require a method to assess the generalization capabilities of a network. Depending on the available number of learning examples one of the following two methods can be used. In the *hold out method* the generalization capabilities of a network are assessed using an independent set of examples. The total error on this set is called the *generalization error*. In practice, the availability of learning examples may be limited and we may not be able to afford the luxury of keeping aside such a set of examples. In such cases we can adopt the method of *cross-validation*, where we divide the available set of examples into a number of distinct segments, develop a network using the data of all but one of these segments, assess the generalization capability of the network on the remaining segment, and repeat the process for each possible choice for the segment that is omitted from the learning process. The results are then averaged. The disadvantage of such an approach is the increase in computational effort.

4.5.3 Alternative stop criteria

In Section 4.3.5 we discussed stop criteria for learning algorithms. These criteria completely ignored the network's generalization capabilities. Next we discuss two alternative stop criteria in which, during execution of the learning algorithm, the generalization error is monitored on an independent set of examples. The monitoring can be done by using the hold out method or by using cross-validation.

In the first stop criterion, called *premature stopping*, the learning algorithm is stopped when the generalization error first starts to increase. This criterion is based on the observation that during a typical execution of a learning algorithm, the generalization error often shows a decrease at first, followed by an increase as the network start to over-fit the learning set. In the second stop criterion the learning algorithm is executed for a fixed number of iterations while always storing a copy of the network with the lowest generalization error.

The major advantage of using such stop criteria is the absence of over-fitting hazard. Consequently, the generalization capabilities of thus obtained networks are independent of the choice of network topology, provided the network mapping capabilities are sufficient.

Remark that, in this way, there are typically three independent sets of examples involved, i.e., a *learning set* used for weight adjustments, a *validation set* to monitor the generalization error during execution of the learning algorithm in order to pick the best network, and a *test set* to asses the generalization capabilities of the final network.

4.5.4 Committees of networks

In practical gradient-based learning algorithms global convergence is not guaranteed, hence the final network might be sensitive to the initial values of the weights. For that reason it is common practice to develop several networks with different weight initializations. The problem now is to decide which of these networks we are actually going to use. One possibility is to keep the network with the lowest error on the validation set and simply discard the remaining networks. Drawbacks of this technique are the effort waste for developing the discarded networks and the danger of the selected network being biased to the validation set. These drawbacks can be overcome by grouping the networks together to form a *committee* and let the committee determine the final output [Perrone & Cooper, 1993]. This determination can be done by majority voting or by averaging all network outputs, for instance. The advantage of such a committee decision is that significant improvements can be obtained with little extra computational requirements. Often the generalization capabilities of such a committee are even better than the best individual network. Bishop [1995] showed that some reduction of generalization error is to be expected, due to the reduced variance which results from the averaging over different networks.

4.6 Statistical classification

The task of classification occurs in a wide range of information processing problems of great practical significance, from automatic reading of postcodes to medical diagnosis. In general, a *classification problem* is concerned with the construction of a *classification rule* that assigns *objects* to pre-defined *classes* on the basis of observed *features*. The most general and commonly used framework in which solutions to classification problems are formulated is a statistical one. The corresponding field of research is called *statistical classification* or *statistical pattern recognition.* It is a well established field with a long history. In recent years it has been demonstrated that multi-layered perceptrons can be viewed as an extension of conventional techniques in statistical classification; see for example the textbooks by Duda & Hart [1973] and Ripley [1996]. Based on the numerous comparisons of the performance of neural network classifiers with the performance of conventional classifiers, we may conclude that multi-layered perceptrons often provide a practical solution approach with a performance that is competitive with the best traditional approaches [Huang & Lippmann, 1988; Michie, Spiegelhalter & Taylor, 1994]. In the remainder of this section we discuss multi-layered perceptrons in the context of statistical classification.

4.6.1 Preliminaries

Suppose we have to assign objects to one of K pre-defined classes on the basis of M observed real-valued features. Let $\Omega \subseteq \mathbb{R}^M$ denote the space of feature vectors and let $\mathcal{L} = \{1, \ldots, K\}$ denote the set of *class labels*. We model the relation between objects and their corresponding class label by a pair of random variables $(\mathbf{X}, Y)$ defined on $\Omega \times \mathcal{L}$. This relation can be characterized by the joint probability density function of $\mathbf{X}$ and Y denoted by $f_{\mathbf{X},Y}$. The corresponding classification problem can now be formulated as to find a classification rule $g : \Omega \rightarrow \mathcal{L}$ that optimizes some appropriate objective. After introducing some basic concepts, we discuss two such objectives, i.e., *maximization of expected classification rate* and *minimization of expected cost.*

4.6.2 Some basic concepts

In statistical classification it is often convenient to write the joint probability density function in the form

$$f_{\mathbf{X},Y}(\mathbf{x}, l) = \mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\} f_{\mathbf{X}}(\mathbf{x}), \tag{4.35}$$

where $\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}$ and $f_{\mathbf{X}}$ denote the conditional probability that an object belongs to class l given feature vector $\mathbf{x}$ and the probability density function of $\mathbf{X}$, respectively. The quantity $\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}$ is called the *posterior probability*, since it gives the probability that an object belongs to class l after the feature vector $\mathbf{x}$ is observed. Similarly, the joint probability density function can be written in the form

$$f_{\mathbf{X},Y}(\mathbf{x}, l) = f_{\mathbf{X}}(\mathbf{x} \mid Y = l)\mathrm{P}\{Y = l\}, \tag{4.36}$$

where $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)$ and $\mathrm{P}\{Y = l\}$ denote the conditional probability density function of $\mathbf{X}$ given $Y = l$ and the probability that an object belonging to class l, respectively. The quantity $\mathrm{P}\{Y = l\}$ is called the *prior probability*, since it gives the probability that an object belongs to class l before its feature vector is observed. Combining these two expressions for the joint probability density function we obtain

$$\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\} = \frac{f_{\mathbf{X}}(\mathbf{x} \mid Y = l)\mathrm{P}\{Y = l\}}{f_{\mathbf{X}}(\mathbf{x})}, \tag{4.37}$$

which is called *Bayes' formula*. It is named after Rev. Thomas Bayes, an 18th century mathematician. Bayes' calculations were published in 1763, two years after his death [Bayes, 1763].

Bayes' formula allows the posterior probabilities to be expressed in terms of prior probabilities $\mathrm{P}\{Y = l\}$ and conditional probability density functions $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)$, where $f_{\mathbf{X}}(\mathbf{x})$ can be written in the form

$$f_{\mathbf{X}}(\mathbf{x}) = \sum_{l \in \mathcal{L}} f_{\mathbf{X}}(\mathbf{x} \mid Y = l)\mathrm{P}\{Y = l\}. \tag{4.38}$$

4.6.3 Maximization of expected classification rate

In the present part we consider the problem of finding a classification rule $g : \Omega \rightarrow \mathcal{L}$ that maximizes the *expected classification rate* $r(g)$ defined by

$$r(g) = \int \mathrm{P}\{Y = g(\mathbf{x}) \mid \mathbf{X} = \mathbf{x}\} f_{\mathbf{X}}(\mathbf{x})d\mathbf{x}. \tag{4.39}$$

If $g(\mathbf{x})$ is chosen such that $\mathrm{P}\{Y = g(\mathbf{x}) \mid \mathbf{X} = \mathbf{x}\}$ is maximal for every $\mathbf{x}$, then $r(g)$ is maximal. This justifies the following statement of the *rate-optimal* classification rule. To maximize the expected classification rate, select that class l for which $\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}$ is maximal. The resulting expected classification rate is called the *Bayes rate* and is the best performance that can be achieved.

In practical classification applications the posterior probabilities are usually unknown and the rate-optimal classification rule cannot be applied. In such cases, one can use available examples of objects, their feature vectors, and corresponding classes to estimate the posterior probabilities, which can be used to classify objects. In this way we obtain *approximative* classification rules. One approach is to estimate the conditional probability density functions $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)$ and the prior probabilities $\mathrm{P}\{Y = l\}$ separately and then combine them using Bayes‘ theorem to give posterior probabilities. An alternative approach is to estimate the posterior probabilities directly. Below, we show that,with a suitable representation of the learning examples, the outputs of a multi-layered perceptron obtained by minimizing the sum-of-squares error on a finite set of learning examples can be interpreted as approximations to posterior probabilities.

Given is an $\{\alpha, \beta\}$-mLP with M inputs, one per input dimension,and K output units, one per class. The network is supposed to be already completely specified apart from the weights $\mathbf{w}$. Learning examples are constructed as follows. For a pair $(\mathbf{x}, l)$ of an object feature vector and its corresponding class label we take the input vector equal to $\mathbf{x}$. The corresponding target vector is defined by $\mathbf{t} = \mathbf{e}_l$, where $\mathbf{e}_l$ denotes a K component vector with a one in the lth position, and zeros elsewhere.

We model the relation between input vectors and target vectors by the pair of random variables $(\mathbf{X}, \mathbf{T})$ defined on $\Omega \times \{0, 1\}^K$, where $\mathbf{T} = (T_1, \ldots, T_K)$ is given by

$$T_k = \begin{cases} 1 & \text{if } Y = k \\ 0 & \text{otherwise} \end{cases} \quad \text{for } k = 1, \ldots, K. \tag{4.40}$$

We apply Theorem 4.2 to obtain that in the limit as the size of the learning set goes to infinity, the network mapping corresponding with a minimum of ϵ is given by the conditional expectations of the targets. Using $\mathrm{E}\{T_k \mid \mathbf{X} = \mathbf{x}\} = \mathrm{P}\{Y = k \mid \mathbf{X} = \mathbf{x}\}$ this yields that a global minimum of ϵ with respect to $\mathbf{w}$ is attained when

$$y_k(\mathbf{x}; \mathbf{w}) = \mathrm{P}\{Y = k \mid \mathbf{X} = \mathbf{x}\} \quad \text{for all } k \in \mathcal{L}. \tag{4.41}$$

This only holds under the condition that (i) the $\{\alpha, \beta\}$-mLP has sufficient network mapping capabilities such that there exists a choice $\mathbf{w}$ which makes the first term in (4.32) sufficiently small and (ii) $\mathrm{E}\{\| \mathbf{y}(\mathbf{X}; \mathbf{w}) - \mathbf{T} \|^2\} < \infty$. One easily verifies that $\mathrm{E}\{T_k\}$ and $\mathrm{E}\{T_k^2\}$ are both finite for $k = 1, \ldots, K$. A sufficient condition for (ii) to hold is that output unit response function β is bounded. A suitable choice is to use σ-mLPs.

For minimizing the sum-of-squares error (4.12) on a finite learning set we conclude that on input of object feature vector $\mathbf{x}$, the network output values can be interpreted as estimates for the posterior probabilities $\mathrm{P}\{Y = k \mid \mathbf{X} = \mathbf{x}\}$. Similar results can be found in the papers by Ruck, Rogers, Kabrisky, Oxley & Suter [1990] and Yeung [1993].

4.6.4 Class overlap

It is in general impossible to determine the class of an object on the basis of a feature vector with certainty. For instance because the feature vector simply contains too little information about the object. In the previous subsection we saw that in such cases it is rate optimal to select the class l for which the posterior probability $\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}$ is maximal. According to (4.37) this corresponds to selecting the class l for which $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)\mathrm{P}\{Y = l\}$ is maximal, where we use that $f_{\mathbf{X}}(\mathbf{x})$ can be viewed as a normalization factor. We recall that $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)$ denotes the probability density function of $\mathbf{X}$ given that the object belongs to class l. From equality (4.38) we know that the overall probability density function $f_{\mathbf{X}}(\mathbf{x})$ can be written as a sum of $|\mathcal{L}|$ components $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)\mathrm{P}\{Y = l\}$, one for each class $l \in \mathcal{L}$. Since the classification problem is completely characterized by these components, the nature of a classification problem depends on the amount by which these components overlap. We refer to this phenomenon as *class overlap*.

The amount of overlap between two probability density functions is usually called the *overlapping coefficient* and refers to the area under the two probability density functions simultaneously. The overlapping coefficient is defined as follows.

Definition 4.2. [Bradley, 1985]. Let $\mathbf{U}$ and $\mathbf{V}$ be random variables on $\mathbb{R}^n$ with corresponding probability density functions $f_{\mathbf{U}}$ and $f_{\mathbf{V}}$. Then the overlapping coefficient $\lambda_{\mathbf{U},\mathbf{V}}$ is defined by

$$\lambda_{\mathbf{U},\mathbf{V}} = \int \min\{f_{\mathbf{U}}(\mathbf{x}), f_{\mathbf{V}}(\mathbf{x})\}d\mathbf{x}.$$

□

Below we derive a measure for the amount of overlap for the general classification problem with the classification rate objective. There are two differences with the ordinary overlapping coefficient. The first difference is that there can be more than two probability density function involved. The second difference is that the probability density functions are weighted by their corresponding prior probabilities. The corresponding overlapping coefficient is denoted by $\lambda_{\mathbf{X},Y}$. For the two-class case we have

$$\lambda_{\mathbf{X},Y} = \int \min\{f_{\mathbf{X}}(\mathbf{x} \mid Y = 1)\mathrm{P}\{Y = 1\}, f_{\mathbf{X}}(\mathbf{x} \mid Y = 2)\mathrm{P}\{Y = 2\}\}d\mathbf{x},$$

which, using Bayes' rule, can be rewritten as

$$\begin{aligned}\lambda_{\mathbf{X},Y} &= \int \min\{\mathrm{P}\{Y = 1 \mid \mathbf{X} = \mathbf{x}\}, \mathrm{P}\{Y = 2 \mid \mathbf{X} = \mathbf{x}\}\}f_{\mathbf{X}}(\mathbf{x})d\mathbf{x} \\ &= 1 - \int \max\{1 - \mathrm{P}\{Y = 1 \mid \mathbf{X} = \mathbf{x}\}, 1 - \mathrm{P}\{Y = 2 \mid \mathbf{X} = \mathbf{x}\}\}f_{\mathbf{X}}(\mathbf{x})d\mathbf{x}.\end{aligned}$$

Using the identity $\mathrm{P}\{Y = 1 \mid \mathbf{X} = \mathbf{x}\} + \mathrm{P}\{Y = 2 \mid \mathbf{X} = \mathbf{x}\} = 1$, we derive

$$\begin{aligned}\lambda_{\mathbf{X},Y} &= 1 - \int \max\{\mathrm{P}\{Y = 1 \mid \mathbf{X} = \mathbf{x}\}, \mathrm{P}\{Y = 2 \mid \mathbf{X} = \mathbf{x}\}\}f_{\mathbf{X}}(\mathbf{x})d\mathbf{x} \\ &= 1 - \int \mathrm{P}\{Y = g^*(x) \mid \mathbf{X} = \mathbf{x}\}f_{\mathbf{X}}(\mathbf{x})dx \\ &= 1 - r(g^*),\end{aligned}$$

where $r(\cdot)$ denotes the expected classification rate as defined by (4.39) and g^* denotes the rate-optimal classification rule. For the general classification problem we define the overlapping coefficient by

$$\lambda_{\mathbf{X},Y} = 1 - r(g^*). \tag{4.42}$$

So the overlapping coefficient $\lambda_{\mathbf{X},Y}$ corresponds with the misclassification rate of the Bayes' optimal classification rule. For example, in case of non-overlapping components we have $r(g^*) = 1$ and $\lambda_{\mathbf{X},Y} = 0$. In practical classification applications, $r(g^*)$ usually cannot be calculated exactly. Nevertheless we know that $r(g) \leq r(g^*)$ for any classification rules g, which can be rewritten as $\lambda_{\mathbf{X},Y} \leq 1 - r(\rho)$. So the expected misclassification rate $1 - r(g)$ of any horizon-selection rule g provides an upper bound for $\lambda_{\mathbf{X},Y}$. Note that we can estimate $1 - r(g)$ by calculating the misclassification rate of g on an independent set of learning examples.

4.6.5 Minimization of expected cost

In some applications, maximization of the expected classification rate is an inappropriate objective. Distinct examples of such applications can be found in medical diagnosis, where it is often necessary to discriminate between the different types of erroneous classifications [Low & Webb, 1990]. A simple approach to obtaining such discrimination is to introduce a *cost matrix* with elements $c_{l,k}$ denoting the cost incurred for classifying an object that belongs to class $l \in \mathcal{L}$ to class $k \in \mathcal{L}$. However it depends on the application if such a cost matrix is sufficient. A more general approach is to introduce a cost function $c : \Omega \times \mathcal{L} \times \mathcal{L} \to \mathbb{R}$, where $c(\mathbf{x}, l, k)$ denotes the cost incurred for classifying an object with feature vector $\mathbf{x}$ that belongs to class $l \in \mathcal{L}$ to class $k \in \mathcal{L}$. Below we work out this approach. Since $\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}$ is the probability that the true class label of $\mathbf{x}$ is l, the expected cost associated with assigning an object with feature vector $\mathbf{x}$ to class k is

$$\mathrm{E}\{c(\mathbf{X}, Y, k) \mid \mathbf{X} = \mathbf{x}\} = \sum_{l \in \mathcal{L}} c(\mathbf{x}, l, k)\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}. \tag{4.43}$$

For notational reasons we abbreviate the conditional expectation $\mathrm{E}\{c(\mathbf{X}, Y, k) \mid \mathbf{X} = \mathbf{x}\}$ to $c(k \mid \mathbf{x})$. We can now consider the problem of finding a classification rule $g : \Omega \to \mathcal{L}$ that minimizes the *overall expected cost* $c(g)$ defined by

$$c(g) = \int c(g(\mathbf{x}) \mid \mathbf{x}) f_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}. \tag{4.44}$$

If $g(\mathbf{x})$ is chosen such that $c(g(\mathbf{x}) \mid \mathbf{x})$ is minimal for every $\mathbf{x}$, then the overall cost is minimal. This justifies the following statement of the *cost-optimal* classification rule. To minimize the overall expected cost, we compute the conditional expectations $c(k \mid \mathbf{x})$ for all $k \in \mathcal{L}$ and select class k for which $c(k \mid \mathbf{x})$ is minimal. The resulting minimum overall cost is called the *Bayes cost* and is the best performance that can be achieved. One easily verifies that maximizing the expected classification rate reduces to a special case of minimizing the overall expected cost by taking $c(\mathbf{x}, l, k)$ equal to

$$c(\mathbf{x}, l, k) = \begin{cases} 1 & \text{if } l \neq k \\ 0 & \text{otherwise.} \end{cases} \tag{4.45}$$

Note that in this case there is no cost distinction, since all errors have equal cost. The notion class overlap, as defined in the previous section, can be easily extended to the more general cost objective by defining $\lambda_{\mathbf{X},Y} = c(g^*)$, where g^* denotes the cost-optimal classification rule.

In practical classification applications the conditional expectations are usually unknown and the cost-optimal classification rule cannot be applied. In such cases, one can use available examples of objects, their feature vectors, and corresponding classes to estimate the conditional expectations, which can used to classify objects. In this way we obtain *approximative* classification rules. One approach is to estimate the posterior probabilities and use (4.43) to give the conditional expectations. An alternative approach is to estimate the conditional expectations directly. Below we show that, with a suitable representation of the learning examples, the outputs of a multi-layered perceptron obtained by minimizing the sum-of-squares error on a finite set of learning examples can be interpreted as approximations to the conditional expectations.

Given is an $\{\alpha, \beta\}$-mLP with M inputs, one per input dimension,and K output units, one per class. The network is supposed to be already completely specified apart from the weights $\mathbf{w}$. Learning examples are constructed as follows. For a pair $(\mathbf{x}, l)$ of an object feature vector and its corresponding class label we take the input vector equal to $\mathbf{x}$. The corresponding target vector is defined by $\mathbf{t} = (c(\mathbf{x}, l, 1), \ldots, c(\mathbf{x}, l, K))$.

We model the relation between input vectors and target vectors by the pair of random variables $(\mathbf{X}, \mathbf{T})$ defined on $\Omega \times \mathbb{R}^K$, where $\mathbf{T} = (T_1, \ldots, T_K)$ is given by

$$T_k = c(\mathbf{X}, Y, k) \quad \text{for } k = 1, \ldots, K. \tag{4.46}$$

We apply Theorem 4.2 to obtain that in the limit as the size of the learning set goes to infinity, the network mapping corresponding with a minimum of ϵ is given by the conditional expectations of the targets. Using $\mathrm{E}\{T_k \mid \mathbf{X} = \mathbf{x}\} = c(k \mid \mathbf{x})$ this yields that a global minimum of ϵ with respect to $\mathbf{w}$ is attained when

$$y_k(\mathbf{x}; \mathbf{w}) = c(k \mid \mathbf{x}) \quad \text{for all } k \in \mathcal{L}. \tag{4.47}$$

This only holds under the condition that (i) the $\{\alpha, \beta\}$-mLP has sufficient network mapping capabilities such that there exists a choice $\mathbf{w}$ which makes the first term in (4.32) sufficiently small and (ii) $\mathrm{E}\{\| \mathbf{y}(\mathbf{X}; \mathbf{w}) - \mathbf{T} \|^2\} < \infty$. A sufficient condition for (ii) to hold is that both output unit response function β and cost function $c(\mathbf{x}, l, k)$ are bounded. In that case a suitable choice is to scale the targets between 0 and 1 and to use σ-mLPs.

For minimizing the sum-of-squares-error on a finite learning set we conclude that on input of object feature vector $\mathbf{x}$, the network output values can be interpreted as estimates for the conditional expectations $c(k \mid \mathbf{x})$.

4.6.6 Discussion

The accuracy of the thus obtained estimates of posterior probabilities and conditional expectations depends on a number of conditions. To begin with, good generalization is only possible if the learning set is sufficiently large and representative and if the multi-layered perceptron has sufficient network mapping capabilities; see Section 4.5. Furthermore, the minimization of sum-of-squares-error as a function of the weights must be done appropriately. If these conditions are satisfied we expect the estimates to be accurate.

Note that the accuracy of these estimates becomes less important when used in a classification rule since, in that case, only correct classification counts. Depending on the chosen objective, such a classification rule classifies correctly if the estimated value of the posterior probability of the correct class is maximal, or if the estimated value of the conditional expectation of the correct class is minimal. This observation has been an inspiration for authors to derive alternatives for the traditional sum-of-squares error function that are tailored for use in a classification rules [Hampshire & Pearlmutter, 1991]; see also the work of Bishop [1995], Bridle [1990], and Weigend [1993]

5

MLP-based horizon-selection rules

In Chapter 2 we introduced a generic single-item lot-sizing model, and we studied three different problem formulations. We briefly discuss these problem formulations below.

First, we addressed the off-line finite-horizon problem, which was solved by dynamic programming. This problem was called the n-period problem.

Second, we studied the off-line infinite-horizon problem which we called the off-line problem. In many settings, infinite-horizon optimal lot sizes depend on limited future demand information only. The theory of planning and forecast horizons addresses this subject. We studied so-called simple planning horizons and showed that, given a simple planning horizon t, the off-line problem decomposes into the t-period problem and the remaining off-line problem. In this way, the off-line problem can be solved by repeatedly solving off-line finite-horizon problems, provided simple planning horizons can be found. We derived a forward algorithm for the detection of simple planning horizons.

Third, we addressed the on-line infinite-horizon problem which we called the on-line problem. As long as simple planning horizons can be calculated on-line, the above decomposition applies and optimal off-line solutions can be obtained. In general, however, simple planning horizons cannot be calculated on-line. For that reason, many heuristics have been proposed in the literature for on-line lot-sizing problems. However, most of these heuristics were tailored to the Wagner-Whitin cost structure. In Section 2.4.2 a class of heuristics for the generic on-line problem is introduced, called *variable-horizon policies*. In a variable-horizon policy the lot sizes are determined by repeatedly solving finite-horizon problems that are split off from the on-line problem. This splitting is done by a *horizon-selection rule*, which determines the number of periods over which is optimized on the basis of the available demand information. Section 2.4.3 showed that most of the existing heuristics for the Wagner-Whitin cost structure can be adapted to become variable-horizon policies without loss of cost performance.

This chapter addresses the problem of finding optimal horizon-selection rules. We formulate the problem of finding an optimal horizon-selection rule as a classification problem, which we analyze using the results and techniques from statistical classification presented in Section 4.6. We consider two objectives, i.e., maximization of expected classification rate and minimization of expected excess cost. For these objectives we give explicit expressions for the optimal horizon-selection rules. Supervised learning with multi-layered perceptrons can be used to estimate the unknown parameters of these expressions. In this way we obtain approximative horizon-selection rules, called *MLP-based* horizon-selection rules. In Chapter 6 we investigate the generalization capabilities of the MLP-based horizon selection rules. The on-line lot-sizing performance of the variable-horizon policies constituted by these rules is investigated in Chapter 7.

5.1 Optimal off-line versus optimal on-line

In Section 2.7 we showed that *optimal off-line* optimization horizons correspond to m-optimal simple planning horizons. Furthermore, we developed a forward algorithm to detect such m-optimal simple planning horizons. The use of this algorithm for on-line lot-sizing is limited, since such horizons can only be calculated on-line if an m-optimal simple planning horizon s for forecast horizon n exists, such that $s \leq n \leq m$. Furthermore, a protective forward algorithm must be available that is able to calculate s and n using no more than m periods of demand information. A perfect forward algorithm would be an example of such an algorithm; see also Section 2.3. In any other case, optimal off-line optimization horizons depend on demand realizations beyond the data horizon and can therefore not be computed on-line. Note that different demand realizations beyond the data horizon may lead to different optimal off-line optimization horizons. What we need is a definition of what is meant by an *optimal on-line* optimization horizon. To this end we make the following two important observations.

1. The task of selecting optimization horizons given available demand information can be seen as a classification task, where the objects to be classified

correspond to decision situations, the feature vectors represent available demand information, and the classes correspond to optimal off-line optimization horizons. From this view-point, horizon-selection rules correspond to classification rules and the problem of finding an optimal horizon-selection rule can be seen as a classification problem. A possible objective would then be to find a horizon-selection rule that maximizes the expected classification rate.

2. Although we do not know the exact relation between available demand information and the optimal off-line optimization horizon, we can calculate examples of this relation from demand history using the forward algorithm developed in Chapter 2. With such examples at hand it is possible to apply techniques from statistical classification to the abovementioned classification problem.

In the remainder of this chapter we exploit these observations by formulating the problem of finding an optimal horizon-selection rule as a classification problem and by adopting an appropriate objective. We analyze two such objectives, i.e., the *classification rate objective* and the *excess cost objective.*

5.2 Classification rate objective

A horizon-selection rule for the generic on-line problem with cost structure (H, P) and data horizon length m can be represented as a mapping $g : \Omega_m \to \mathcal{L}_m$, where $\Omega_m \subseteq \mathrm{IR}^m$ denotes the space of possible demand vectors and $\mathcal{L}_m = \{1, \ldots, m\}$ denotes the set of possible optimization horizons. Using the terminology of Section 4.6, we can state that Ω_m represents the space of feature vectors, and $\mathcal{L}_m$ represents the set of class labels. Before we can formulate the problem of finding an optimal horizon-selection rule, we have to model the relation between demand vectors and their corresponding optimal off-line optimization horizons.

5.2.1 Preliminaries

We recall that optimal off-line optimization horizons correspond to m-optimal simple planning horizons. Therefore, in the sequel, we use the term m-optimal simple planning horizon.

We assume that m-optimal simple planning horizons always exist and therefore can be calculated off-line. Note that the non-existence of m-optimal simple planning horizons could be modeled by the introduction of an extra class and extending $\mathcal{L}_m$ with an extra class label. This, however, goes beyond the scope of this thesis and is mainly of theoretical interest; see also Section 2.5 which discussed the existence of planning horizons.

Since more than one m-optimal simple planning horizon may exist, we have to break ties in some way. If the forward algorithm developed in Chapter 2 terminates, it returns the minimal m-optimal simple forecast horizon and all corresponding m-optimal simple planning horizons. To break ties, we concentrate on the smallest m-optimal simple planning horizon, which for reasons of convenience is called the *minimal* m-optimal simple planning horizon.

Since m-optimal simple planning horizons can in general not be computed on-line, they can in general not be expressed as a function of the m demands within the data horizon. For that reason, we model the relation between demand vectors and their corresponding minimal m-optimal simple planning horizon by the pair of random variables $(\mathbf{X}_m, Y_m)$ defined on $\Omega_m \times \mathcal{L}_m$. This relation can be characterized by the joint probability density function of $\mathbf{X}_m$ and Y_m denoted by $f_{\mathbf{X}_m.Y_m}$.

5.2.2 Problem formulation and analysis

We now formulate the problem of finding an optimal horizon-selection rule for the generic on-line problem with cost structure (H, P) and data horizon length m as to find a mapping $g : \Omega_m \rightarrow \mathcal{L}_m$ that maximizes the expected classification rate $r(g)$ given by

$$r(g) = \int \mathrm{P}\{Y_m = g(\mathbf{x}) \mid \mathbf{X}_m = \mathbf{x}\} f_{\mathbf{X}_m}(\mathbf{x}) d\mathbf{x}, \tag{5.1}$$

where $\mathrm{P}\{Y_m = l \mid \mathbf{X}_m = \mathbf{x}\}$ denotes the posterior probability that $l \in \mathcal{L}_m$ is the minimal m-optimal simple planning horizon given the demand vector $\mathbf{x} \in \Omega_m$ and $f_{\mathbf{X}_m}$ denotes the probability density function of $\mathbf{X}_m$. We refer to this objective as the *classification rate objective*.

If $g(\mathbf{x})$ is chosen such that $\mathrm{P}\{Y_m = g(\mathbf{x}) \mid \mathbf{X}_m = \mathbf{x}\}$ is maximal for every $\mathbf{x}$, then $r(g)$ is maximal. This justifies the definition of the *rate-optimal* horizon-selection rule ρ_m^* given by

$$\rho_m^*(\mathbf{x}) = \arg\max_{l \in \mathcal{L}_m} \mathrm{P}\{Y_m = l \mid \mathbf{X}_m = \mathbf{x}\}. \tag{5.2}$$

The expected classification rate $r(\rho_m^*)$ of the rate-optimal horizon-selection rule is called the *Bayes rate*. Note that, in this way, we implicitly define an *optimal on-line* optimization horizon to be an optimization horizon that has the highest posterior probability of being an m-optimal simple planning horizon.

5.2.3 Properties of the rate-optimal horizon-selection rule

At first we conjectured that the Bayes rate $r(\rho_m^*)$ is increasing in m for $m \geq 1$, because the amount of relevant demand information increases. However, looking at the trivial case $m = 1$, it is easy to see that, in general, this is invalid. For the case

$m = 1$ there is only one class, i.e., one possible off-line optimal optimization horizon, and therefore $r(\rho_1^*) = 1$. The reason for this is that the characteristics of the classification problem, represented by the joint probability density function $f_{\mathbf{X}_m, Y_m}$, change with m. Below, we present some results concerning these characteristics.

Proposition 2.7 showed that, under the condition that a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n, the notions optimality and k-optimality are equivalent for $k \geq M$. It is easy to see that, under the same condition, equivalence holds for the notions simple planning horizon and k-optimal simple planning horizon. The following result is then immediate.

Proposition 5.1. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then the random variables $Y_M, Y_{M+1}, \ldots$ are identically distributed. The corresponding distribution equals the distribution of minimal simple planning horizons.* □

Note that, if a bound M exists that satisfies the requirement of Proposition 5.1, random variables Y_k with $k = 1, \ldots, M - 1$ are in general not identically distributed, the number of relevant classes may change with k, and optimality is in general not equivalent to k-optimality for $k = 1, \ldots, M - 1$. Next we show that if a bound M exists that satisfies the requirement of Proposition 5.1, the Bayes rate $r(\rho_m^*)$ increases with m for $m \geq M$.

Theorem 5.1. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then*

$$r(\rho_M^*) \leq r(\rho_{M+1}^*) \leq \ldots .$$

Proof. Take some $m \geq M$. Consider the horizon-selection rule $g : \Omega_{m+1} \rightarrow \mathcal{L}_{m+1}$ that for each $\mathbf{x} \in \Omega_{m+1}$, applies ρ_m^* to $(x_1, \ldots, x_m)$, discards x_{m+1}, and returns $\rho_m^*(x_1, \ldots, x_m)$. Let $\mathbf{x} = (x_1, \ldots, x_{m+1})$ and $\tilde{\mathbf{x}} = (x_1, \ldots, x_m)$. Then, using Proposition 5.1, we obtain

$$P\{Y_{m+1} = g(\mathbf{x}) \mid \mathbf{X}_{m+1} = \mathbf{x}\} = P\{Y_m = \rho_m^*(\tilde{\mathbf{x}}) \mid \mathbf{X}_m = \tilde{\mathbf{x}}\},$$

and one easily verifies that $r(g) = r(\rho_m^*)$, which, using $r(g) \leq r(\rho_{m+1}^*)$, completes the proof. □

In the same way we did for m-optimal simple planning horizons, the relation between demand vectors and their corresponding (off-line) minimal m-optimal simple forecast horizons can be modeled by a pair of random variables $(\mathbf{X}_m, Z_m)$ defined on $\Omega_m \times \mathbb{N}$ and characterized by a joint probability density function $f_{\mathbf{X}_m, Z_m}$. The following result uses this model to derive a lower bound on the Bayes rate.

Theorem 5.2. *Suppose a perfect forward algorithm exists for the detection of m-optimal simple planning horizons. Then we have*

$$r(\rho_m^*) \geq P\{Z_m \leq m\}.$$

Proof. Consider the following horizon-selection rule denoted by g. We apply the perfect forward algorithm. If the algorithm terminates within the data horizon, it returns the minimal m-optimal simple planning horizon found by the algorithm. If the algorithm does not terminate, some optimization horizon is returned randomly. The expected classification rate $r(g)$ of this horizon-selection rule is greater than or equal to $\text{P}\{Z_m \leq m\}$, the probability that the m-optimal minimal forecast horizon is smaller than or equal to the data horizon. Since ρ_m^* is rate-optimal, we have $r(\rho_m^*) \geq r(g)$. This completes the proof. □

Unfortunately, we do not have a perfect algorithm for the generic lot-sizing model available. What we do have available is a protective forward algorithm. This algorithm can be used to generate a set of examples of the relationship between demand vectors and m-optimal minimal simple forecast horizons. From such a set of examples we can estimate the probabilities $\text{P}\{Z_m \leq m\}$. We remark that perfect forward algorithms do exist for special cases of the generic lot-sizing model. For example, Chand & Morton [1986] derived a perfect forward algorithm for the Wagner-Whitin cost structure. The next result concerns the characteristics of Z_m as a function of m and can be proven using similar arguments as in Proposition 5.1.

Proposition 5.2. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then the random variables $Z_M, Z_{M+1}, \ldots$ are identically distributed. The corresponding distribution equals the distribution of minimal simple forecast horizons.*
□

The protective forward algorithm developed in Chapter 2 for the calculation of simple planning horizons for forecast horizons uses some finite upper bound M on the length of a subplan in an optimal production plan. To detect a simple planning horizon t for forecast horizon n, the forward algorithm requires a data horizon of $n + M - 1$ periods. The following lemma is based on this observation. Its proof is analogous to that of Theorem 5.2 and therefore omitted.

Lemma 5.1. *Suppose a finite upper bound M exists on the length of a subplan in* *an optimal production plan for the n-period problem that is independent of n. Then*

$$r(\rho_m^*) \geq P\{Z_m \leq m - M + 1\}.$$

□

This lemma is used in the following proof of convergence of the Bayes rate. Furthermore, in this proof, we use the assumption that m-optimal simple planning horizons always exist.

Theorem 5.3. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then*

$$\lim_{m \to \infty} r(\rho_m^*) = 1.$$

Proof. Let $m > M$. Using Lemma 5.1 and Proposition 5.2 we obtain $\mathrm{P}\{Z_M \leq m - M + 1\} \leq r(\rho_m^*) \leq 1$. From the assumption that m-optimal simple planning horizons always exist it is evident that m-optimal simple forecast horizons always exist and thus can be calculated off-line. This implies that $\lim_{k \to \infty} \mathrm{P}\{Z_M \leq k\} = 1$. The proof is now completed by letting $m \to \infty$ and using elementary calculus. □

5.2.4 An MLP-based horizon-selection rule

The rate-optimal horizon-selection rule $r(\rho_m^*)$ requires explicit knowledge of the posterior probabilities $\mathrm{P}\{Y_m = l \mid \mathbf{X}_m = \mathbf{x}\}$. Unfortunately these quantities are usually unknown and this rule cannot be directly applied. Below we show that with a suitable representation of the learning examples, on input of demand vector $\mathbf{x}$ the outputs of a multi-layered perceptron obtained by minimizing the sum-of-squares error on a finite set of learning examples can be interpreted as estimates of the posterior probabilities $\mathrm{P}\{Y_m = l \mid \mathbf{X}_m = \mathbf{x}\}$. By substituting these estimated posterior probabilities into (5.2), we obtain an MLP-based horizon-selection rule.

Computing learning examples. We now describe how we construct a learning example from demand history. To that end we consider the generic on-line problem with cost structure (H, P) and data horizon length m. Without loss of generality, we consider some sequence of periods in demand history, labeled $1, 2, \ldots$. Furthermore, we suppose that the inventory level at the beginning of period 1 equals zero. We apply the forward algorithm developed in Section 2.3 for the detection of simple planning horizons with parameter M set to m. As was shown in Section 2.7, on termination, this algorithm returns a minimal simple m-optimal planning horizon s for forecast horizon n. As in Section 4.6.3, a learning example $(\mathbf{x}, \mathbf{t})$ is constructed as follows. For the input vector we take $\mathbf{x} = (d_1, \ldots, d_m)$. The corresponding target vector is defined by $\mathbf{t} = \mathbf{e}_s$, where $\mathbf{e}_s$ denotes an m component vector with a one in the sth position, and zeros elsewhere. Thus constructed target vectors we call *zero-one target vectors.*

Multi-layered perceptrons. Given is a multi-layered perceptron with m inputs, m output units and sigmoidal response functions σ that is completely specified apart from the weights $\mathbf{w}$. The following statement is justified by the results of Section 4.6.3. After minimization of the sum-of-squares error function on a finite set of learning examples with zero-one target vectors, on input of demand vector $\mathbf{x}$, the network output values $y_k(\mathbf{x}; \mathbf{w})$ can be interpreted as estimates for the posterior probabilities $\mathrm{P}\{Y_m = k \mid \mathbf{X}_m = \mathbf{x}\}$. The mapping $\rho_m^{\mathsf{MLP}} : \Omega_m \to \mathcal{L}_m$ given by

$$\rho_m^{\mathsf{MLP}}(\mathbf{x}) = \arg\max_{k \in \mathcal{L}_m} y_k(\mathbf{x}; \mathbf{w}), \tag{5.3}$$

can thus be seen as an *approximative* horizon-selection rule with respect to the classification rate objective.

5.3 Excess cost objective

In Section 4.6.5 we generalized the classification rate objective by introducing a cost function $c(\mathbf{x}, l, k)$, which denotes the cost incurred for classifying a class l object with feature vector $\mathbf{x}$ to class k. This enabled discrimination in terms of cost between the different types of erroneous classifications. For our purposes, such discrimination may be very important, since the developed horizon-selection rule is incorporated in a variable-horizon policy, and the performance of a variable-horizon policy is evaluated in terms of production and holding cost rather than in terms of classification rate. Before we can formulate the problem of finding an optimal horizon-selection rule, we have to define the cost of choosing an inappropriate optimization horizon.

5.3.1 Preliminaries

In Section 2.6 we introduced $\Delta(p)$, which denotes the excess cost (over infinite-horizon optimality) of decomposing the off-line problem at period p. Based on Theorem 2.3 we developed a forward algorithm for the calculation of $\Delta(p)$. This algorithm requires knowledge of a finite upper bound M on the length of a subplan in an optimal production plan. Although such bounds were derived in Chapter 3 for the cost structures under consideration, we prefer to use use the excess cost over infinite-horizon m-optimality because (i) no bounds are required and (ii) no variable-horizon policy with a data horizon of length m can do better than infinite-horizon m-optimality. In Section 2.6 we generalized $\Delta(p)$ towards k-optimality to obtain $\Delta_k(p)$, which denotes the excess cost (over infinite-horizon k-optimality) of decomposing the off-line problem at period p. We showed that the forward algorithm for calculating $\Delta(p)$ can be used to calculate $\Delta_k(p)$ by choosing $M = k$.

Given a variable-horizon policy for the on-line problem with cost structure (H, P) and data horizon length m. Suppose we execute one iteration of this policy and we select an optimization horizon $p \in \mathcal{L}_m$. Then, by definition, we have excess cost of $\Delta_m(p)$. These excess cost can in general not be calculated on-line. Consequently, $\Delta_m(p)$ can in general not be written as a function of the m demands within the data horizon $d_1, \ldots, d_m$ and a cost function $c(\mathbf{x}, k, l)$ as proposed in Section 4.6.5 does not exist.

For this reason we model the relation between demand vectors and the excess cost (over infinite-horizon m-optimality) of decomposing the off-line problem at period p by the pairs of random variables $(\mathbf{X}_m, C_m(p))$ for $p = 1, 2, \ldots,$ which are defined on $\Omega_m \times \mathbb{R}$. This relation can be characterized by the joint probability density function of $\mathbf{X}_m$ and $C_m(p)$ denoted by $f_{\mathbf{X}_m, C_m(p)}$. For notational reasons we introduce $\mathbf{C}_m = (C_m(1), \ldots, C_m(m))$ and the joint probability density function $f_{\mathbf{X}_m, \mathbf{C}_m}$ of $\mathbf{X}_m$ and $\mathbf{C}_m$.

Remark that, for similar reasons an m-optimal simple planning horizon may not exist, it may occur that $\Delta_m(p)$ does not exist. As for the classification rate objective, we assume that the cost excesses $\Delta_m(p)$ can always be calculated off-line. Note that, in practice we can always use the bounds $L(p, t)$ and $U(p, t)$ from Theorem 2.3 to estimate $\Delta_m(p)$.

5.3.2 Problem formulation and analysis

The problem of finding an optimal horizon-selection rule for the generic on-line problem with cost structure (H, P) and data horizon length m is now formulated as to find a mapping $g : \Omega_m \to \mathcal{L}_m$ that minimizes the overall expected excess cost $c(g)$ defined by

$$c(g) = \int \mathrm{E}\{C_m(g(\mathbf{x})) \mid \mathbf{X}_m = \mathbf{x}\} f_{\mathbf{X}_m}(\mathbf{x}) d\mathbf{x}, \tag{5.4}$$

where $\mathrm{E}\{C_m(k) \mid \mathbf{X}_m = \mathbf{x}\}$ denotes the conditional expectation of the excess cost when decomposing the off-line problem at period k given the demand vector $\mathbf{x} \in \Omega_m$. We refer to this objective as the *excess cost objective*.

If $g(\mathbf{x})$ is chosen such that $\mathrm{E}\{C_m(g(\mathbf{x})) \mid \mathbf{X}_m = \mathbf{x}\}$ is minimal for every $\mathbf{x}$, then $c(g)$ is minimal. This justifies the definition of the *cost-optimal* horizon-selection rule τ_m^* given by

$$\tau_m^*(\mathbf{x}) = \arg\min_{l \in \mathcal{L}_m} \mathrm{E}\{C_m(l) \mid \mathbf{X}_m = \mathbf{x}\}. \tag{5.5}$$

The expected excess cost $c(\tau_m^*)$ of the cost-optimal horizon-selection rule is called the *Bayes cost*. Note that, in this way, we implicitly define an *optimal on-line* optimization horizon to be an optimization horizon with lowest conditional expected excess cost.

5.3.3 Properties of the cost-optimal horizon-selection rule

At first we conjectured that the Bayes cost $c(\rho_m^*)$ is decreasing in m for $m \geq 1$, because the amount of relevant demand information increases. However, looking at the trivial case $m = 1$, it is easy to see that, in general, this is invalid. For the case $m = 1$ there is only one class and therefore $c(\rho_1^*) = 0$. The reason for this is that the characteristics of the classification problem, represented by the joint probability density function $f_{\mathbf{X}_m, \mathbf{C}_m}$, change with m. Below, we present some results concerning the characteristics of the classification problem as a function of m.

Proposition 2.7 showed that, under the condition that a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n, the notions optimality and m-optimality are equivalent for $m \geq M$. It is easy to see that under the same condition, equivalence holds for $\Delta(p)$ and $\Delta_m(p)$. The following result is then immediate.

Proposition 5.3. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then for all $p = 1, 2, \ldots$ the random variables $C_M(p)$, $C_{M+1}(p)$, ... are identically distributed. The corresponding distribution equals the distribution of excess cost (over infinite-horizon M-optimality) when decomposing the off-line problem at period p.* □

Note that, if a bound M exists that satisfies the requirement of Proposition 5.3, random variables $C_m(p)$ with $m = 1, \ldots, M - 1$ and $p = 1, 2, \ldots$ are in general not identically distributed, the number of relevant classes may change with m, and optimality is in general not equivalent to m-optimality for $m = 1, \ldots, M - 1$. Next we show that if a bound M exists that satisfies the requirement of Proposition 5.3, the Bayes cost $c(\rho_m^*)$ decreases with m for $m \geq M$. Since this result is similar to that of Theorem 5.1 its proof is omitted.

Theorem 5.4. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then*

$$c(\tau_M^*) \geq c(\tau_{M+1}^*) \geq \ldots .$$

□

The following conjecture is inspired by
Theorem 5.3 for the classification rate objective.

Conjecture 5.1. $\lim_{m \to \infty} c(\tau_m^*) = 0.$ □

5.3.4 An MLP-based horizon-selection rule

The cost-optimal horizon-selection rule τ_m^* requires explicit knowledge of the conditional expectations $\mathrm{E}\{C_m(l) \mid \mathbf{X}_m = \mathbf{x}\}$. Unfortunately these quantities are usually unknown and this rule cannot be directly applied. Below we show that with a suitable representation of the learning examples, on input of demand vector $\mathbf{x}$ the outputs of a multi-layered perceptron obtained by minimizing the sum-of-squares error on a finite set of learning examples can be interpreted as estimates of the conditional expectations $\mathrm{E}\{C_m(l) \mid \mathbf{X}_m = \mathbf{x}\}$. By substituting these estimated posterior probabilities into (5.5), we obtain an MLP-based horizon-selection rule.

Computing learning examples. We now describe how we construct a learning example from demand history. To that end we consider the generic on-line problem with cost structure (H, P) and data horizon length m. Without loss of generality, we consider some sequence of periods in demand history, labeled $1, 2, \ldots$. Furthermore, we suppose that the inventory level at the beginning of period 1 equals zero. We use the forward algorithm developed in Section 2.6 for the calculation of $\Delta(p)$ with the parameter M set to m. The algorithm is applied for $p = 1, \ldots, m$ to obtain excess cost $\Delta_m(1), \ldots, \Delta_m(m)$; see Section 2.6. A learning example $(\mathbf{x}, \mathbf{t})$ can now be constructed as follows: for the input vector $\mathbf{x}$ we take $\mathbf{x} = (d_1, \ldots, d_m)$ and for the target vector $\mathbf{t}$ we take $\mathbf{t} = (\Delta_m(1), \ldots, \Delta_m(m))$. Thus constructed target vectors we call *cost target vectors*.

Multi-layered perceptrons. Given is a multi-layered perceptron with m inputs and m output units. The network is supposed to be already completely specified apart from the weights $\mathbf{w}$. The relation between input vectors and target vectors was already modeled by the pair of random variables $(\mathbf{X}_m, \mathbf{C}_m)$. We apply Theorem 4.2 to obtain that, in the limit as the size of the learning set goes to infinity, the network mapping corresponding with a minimum of ϵ is given by the conditional expectations of the targets. Furthermore, a global minimum of ϵ with respect to $\mathbf{w}$ is attained when

$$y_k(\mathbf{x}; \mathbf{w}) = \mathrm{E}\{C_m(k) \mid \mathbf{X}_m = \mathbf{x}\} \quad \text{for all } k \in \mathcal{L}_m. \tag{5.6}$$

This only holds under the condition that (i) the multi-layered perceptron has sufficient network mapping capabilities such that there exists a choice $\mathbf{w}$ which makes the first term in (4.32) sufficiently small and (ii) $\mathrm{E}\{\| \mathbf{y}(\mathbf{X}_m; \mathbf{w}) - \mathbf{C}_m \|^2\} < \infty$. A sufficient condition for (ii) to hold is that both the output unit response functions and the excess cost functions $\Delta_m(p)$ for $p = 1, \ldots, m$ are bounded. In that case a suitable choice is to scale the targets between 0 and 1 and to use multi-layered perceptrons with logistic sigmoid response functions, for instance. We remark that this was proven for the Wagner-Whitin cost structure in Proposition 3.2.

For minimizing the sum-of-squares-error function on a finite set of learning examples with cost target vectors we conclude that, on input of demand vector $\mathbf{x}$, the network output values approximate the conditional expectations $\mathrm{E}\{C_m(k) \mid \mathbf{X}_m = \mathbf{x}\}$. The mapping $\tau_m^{\mathrm{MLP}} : \Omega_m \rightarrow \mathcal{L}_m$ given by

$$\tau_m^{\mathrm{MLP}}(\mathbf{x}) = \arg\min_{k \in \mathcal{L}_m} y_k(\mathbf{x}; \mathbf{w}) \tag{5.7}$$

can thus be seen as an *approximative* horizon-selection rule with respect to the excess cost objective.

5.4 Discussion

This chapter addressed the problem of finding optimal horizon-selection rules for the on-line problem with cost structure (H, P) and data horizon length m. To that end we formulated the problem of finding an optimal horizon-selection rule as a classification problem. We considered two objectives: the classification rate objective and the excess cost objective. For each of these objectives we derived and analyzed an explicit expression for the optimal horizon-selection rule. Unfortunately these optimal horizon-selection rules cannot be applied, because they are expressed in unknown quantities like for example posterior probabilities. It was shown that multi-layered perceptrons can be used to estimate these unknown quantities on the basis of learning examples. In this way we obtained MLP-based horizon-selection rules.

Implementation. The ideas and results in this chapter are completely independent of the cost structure (H, P). The only exception is found in the implementation of the procedure for the calculation of learning examples. Learning examples are computed using the forward algorithms developed in Chapter 2. To implement these forward algorithms for a particular cost structure, an algorithm for calculation of optimal subplans for that cost structure is needed. In Chapter 3 such algorithms were derived for three different cost structures. For these cost structures we implemented procedures for the construction of learning examples. In these procedures we always calculate both zero-one target vectors and cost target vectors. These vectors are combined to form *combined learning examples* of the form $(\mathbf{x}, \mathbf{t}, \mathbf{c})$, where $\mathbf{x}$ denotes an input vector, $\mathbf{t}$ a zero-one target vector, and $\mathbf{c}$ a cost targets vector. Advantages of using combined learning examples are a reduced data storage and the possibility of computing, for example, the cost effectiveness of an approach based on zero-one target vectors.

Performance evaluation. The performance of the MLP-based horizon-selection rules can be evaluated on an independent set of learning examples. For instance, in case of the classification rate objective, one can calculate the classification rate on an independent set of learning examples with zero-one target vectors. Such selection of optimization horizons for demand sequences that are outside the learning set is called *generalization*. The generalization capabilities of the MLP-based horizon-selection rules depend on the extent to which the conditions for good generalization discussed in Section 4.5 are satisfied. Stated briefly, the learning set must be sufficiently large and representative, and the multi-layered perceptron must have sufficient network mapping capabilities. For instance, if these conditions are satisfied in case of the classification rate objective, it is likely that ρ_m^{MLP} has a near-optimal classification rate, i.e., close to the Bayes rate. On the other hand, if one or more of these conditions are violated the classification rate may be far from optimal. In Chapter 6 we analyze the generalization capabilities of the horizon-selection rules ρ_m^{MLP} and τ_m^{MLP}. The on-line lot-sizing performance of their corresponding variable-horizon policies is evaluated in Chapter 7.

6

Generalization capabilities of MLP-based horizon-selection rules

The implementation of the horizon-selection rules for the on-line lot-sizing problem proposed in the previous chapter involves the construction of multi-layered perceptrons on the basis of learning examples using supervised learning. As every method that uses examples of input-output behavior to model a process, supervised learning is closely related to the subject of generalization. In that context, generalization refers to the prediction of outputs for *new* inputs, i.e., prediction of outputs for inputs that are not used for modeling. Good generalization is in general not simply evident and generalization capabilities are, among other factors, determined by the smoothness of the target mapping and the representativeness of the learning set.

This chapter studies the generalization capabilities of the MLP-based horizon-selection rules for on-line lot-sizing problems with Wagner-Whitin cost structure. Particularly, it investigates the effect of the length of the data horizon and the type of learning examples on the generalization capabilities. These investigations are based on a set of experiments.

The remainder of this chapter is outlined as follows. Section 6.1 discusses necessary conditions for good generalization. The experimental setup is given in Section 6.2. As a reference, in Section 6.3, we adopt the K-nearest-neighbors technique, which can be used for classification. Such K-nearest-neighbors classifiers have shown reasonable generalization capabilities in a large variety of tasks [Michie, Spiegelhalter & Taylor, 1994]. Application of the K-nearest-neighbors technique to the two objectives introduced in Chapter 5 yields two horizon-selection rules. Section 6.4 and Section 6.5 are devoted to generalization with zero-one target vectors and generalization with cost target vectors, respectively. Finally, in Section 6.6, we formulate the main conclusions of this chapter.

6.1 Generalization

In Chapter 5 we formulated the problem of finding an optimal horizon-selection rule as a classification problem and analyzed two objectives, i.e., the classification rate objective and the excess cost objective. Optimal horizon-selection rules were derived for these objectives, which contained some unknown quantities, e.g., posterior probabilities. It was shown that these quantities can be estimated using a multi-layered perceptron constructed on the basis of learning examples. In this way we obtained two MLP-based horizon-selection rules, one for each objective.

Depending on the objective, the notion of generalization has a different meaning. In case of the classification rate objective, multi-layered perceptrons were used to estimate posterior probabilities and generalization refers to the prediction of these probabilities for new demand vectors. In case of the excess cost objective, multi-layered perceptrons were used to estimate conditional expectations of excess cost and generalization refers to the prediction of these expectations for new demand vectors. Section 4.5 gave necessary conditions for good generalization and a distinction was made between problem specific conditions and model specific conditions for good generalization. Below, we discuss these conditions.

6.1.1 Problem specific conditions

In Section 4.5, two problem specific necessary conditions for good generalization were given. First, the target mapping underlying the learning examples must be sufficiently smooth. Second, the learning set must be sufficiently large and representative. In this thesis we consider artificial demand processes with known stationary distributions. For the construction of learning examples, demands are drawn from these distributions. Therefore, representativeness of a learning set is guaranteed under the condition that the set is sufficiently large. The required number of learning examples is determined by the smoothness of the target mapping. Next we give the target mappings for the two objectives in case of a generic on-line problem with cost structure (H, P) and data horizon length m. Furthermore, we discuss their smoothness.

Classification rate objective. Let us denote the target mapping underlying the learning examples with zero-one target vectors by $\phi_m : \Omega_m \to [0, 1]^m$. Then ϕ_m is given by

$$\phi_m(\mathbf{x}) = (\phi_{m,1}(\mathbf{x}), \ldots, \phi_{m,m}(\mathbf{x})), \tag{6.1}$$

where $\phi_{m,k}(\mathbf{x}) = \mathrm{P}\{Y_m = k \mid \mathbf{X}_m = \mathbf{x}\}$ for $k = 1, \ldots, m$. In Section 5.2.4 it was shown that the network mapping of a multi-layered perceptron obtained by minimizing the sum-of-squares error function on a finite set of learning examples with zero-one target vectors approximates the target mapping ϕ_m; see also Section 4.6.3. The generalization capabilities of such a network are partly determined by the smoothness of ϕ_m. Since ϕ_m is in general unknown, it is impossible to analyze the smoothness of ϕ_m for a given m. But we can give the following general result on the smoothness of ϕ_m as a function of m, which is obvious from Proposition 5.1 and our assumption that m-optimal simple planning horizons always exist.

Proposition 6.1. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. For $\mathbf{x} \in \mathbb{R}^\infty$ and for $m = 1, 2, \ldots$ we define $\mathbf{x}_m = (x_1, \ldots, x_m)$. Then for all $\mathbf{x} \in \mathbb{R}^\infty$ there exist integers $l \in \mathcal{L}_M$ and $N \geq M$ such that*

(i) $\phi_{M.l}(\mathbf{x}_M) \leq \phi_{M+1.l}(\mathbf{x}_{M+1}) \leq \ldots,$

(ii) $\phi_{M.k}(\mathbf{x}_M) \geq \phi_{M+1.k}(\mathbf{x}_{M+1}) \geq \ldots$ *for all $k \in \mathcal{L}_M \setminus \{l\}$, and*

(iii) $\phi_{m.k}(\mathbf{x}_m) = \begin{cases} 1 & \text{if } k = l \\ 0 & \text{otherwise} \end{cases}$ *for all $m \geq N$ and $k \in \mathcal{L}_M$.*

□

Next we use these results to discuss the smoothness of ϕ_m as a function of the data horizon length m. Assume that the assumption of Proposition 6.1 holds and such a bound M exists. We start by considering the case that m is sufficiently large. From (iii) we infer that the functions $\phi_{m.k}$ with $k \in \mathcal{L}_m$ are of the form $\phi_{m.k} : \Omega_m \to \{0, 1\}$. Using the terminology introduced in Section 4.2.1, these functions can be viewed as *classification* functions. The corresponding decision regions are given by

$$\mathcal{J}(\phi_{m.k}) = \{\mathbf{x} \in \Omega_m \mid \phi_{m.k}(\mathbf{x}) = 1\} \quad \text{for } k = 1, \ldots, m. \tag{6.2}$$

At each decision boundary, the target mapping ϕ_m is discontinuous. The more decision boundaries, the more discontinuities, and the less smooth ϕ_m.

We now decrease m. For m sufficiently small, the discontinuities at the decision boundaries vanish. Using (i) and (ii) we infer that when further decreasing m, the smoothness of ϕ_m at the decision boundaries increases until $m = M$. Proposition 6.1 does not address the case $m < M$.

Excess cost objective. Let us denote the target mapping underlying the learning examples with cost target vectors by $\psi_m : \Omega_m \to \mathbb{R}^m$. Then ψ_m is given by

$$\psi_m(\mathbf{x}) = (\psi_{m,1}(\mathbf{x}), \ldots, \psi_{m,m}(\mathbf{x})), \tag{6.3}$$

where $\psi_{m,k}(\mathbf{x}) = \mathrm{E}\{C_m(k) \mid \mathbf{X}_m = \mathbf{x}\}$ for $k = 1, \ldots, m$. In Section 5.3.4 it was shown that, the network mapping of a multi-layered perceptron obtained by minimizing the sum-of-squares error function on a finite set of learning examples with cost target vectors approximates the target mapping ψ_m. The generalization capabilities of this network are partly determined by the smoothness of ψ_m. Since ψ_m is partly determined by the cost structure (H, P), at this point nothing can be stated about its smoothness.

6.1.2 Model specific conditions

Data driven model building approaches, like supervised learning, typically have some parameter which controls the level of *model flexibility*. A necessary condition for good generalization is that the value of this parameter is chosen such that the level of model flexibility is optimal with respect to the learning set. Section 4.5.1 discussed this subject by means of the bias-variance trade-off. Too much model flexibility leads to over-fitting; too little model flexibility smoothes out some of the underlying structure. The model flexibility of a multi-layered perceptron is determined by its network mapping capabilities which are controlled by its topology. Remark that in most learning algorithms convergence to a global minimum of the sum-of-squares error function E with respect to the weights $\mathbf{w}$ is not assured. A final condition for good generalization with multi-layered perceptrons is that the weights are chosen such that E is sufficiently small.

6.1.3 Expectations

From the above results we suspect that the conditions for good generalization for learning with zero-one target vectors deteriorate with m. Furthermore, we suspect that for a given data horizon length m the target mapping ψ_m is smoother than the target mapping ϕ_m. In the remainder of this chapter this is further investigated.

6.1.4 Generalization assessment

Generalization assessment is done by means of the *hold out method* introduced in Section 4.5, in which generalization capabilities are measured on an independent test set. With respect to generalization capabilities we make a distinction between the multi-layered perceptron and its corresponding horizon-selection rule. The generalization capabilities of the multi-layered perceptron is measured by the sum-of-squares error on an independent test set. This is an obvious choice since it was obtained by minimizing the sum-of-squares error on a finite learning set. The generalization capabilities of the corresponding horizon-selection rule are measured differently. From that viewpoint, generalization can be seen as the selection of optimization horizons for new demand vectors. Obvious choices to measure generalization capabilities are then to calculate the classification rate on an independent test set in case of the classification rate objective, and to calculate the average excess cost on an independent test in case of the excess cost objective.

6.2 Experimental setup

In this chapter we consider an on-line lot-sizing problem with a Wagner-Whitin cost structure (holding cost $h = 1$, setup cost $S = 200$, and production cost $p = 1$) and demands that are uniformly distributed on $[0, 200]$. The length m of the data horizon is varied between 2 and 10.

For the calculation of learning examples we generate a 10,000 period demand sequence by independently drawing from a uniform distribution on $[0, 200]$. For each value of m, we generate three sets of 2,500 combined learning examples from this sequence, i.e., a learning set, a validation set for monitoring the generalization capabilities during execution of the learning algorithm, and a test set for assessing the generalization capabilities of a network afterwards. By taking the learning set rather large we want to accomplish that it approximates an infinite learning set.

Note that these combined learning examples contain both zero-one target vectors and cost target vectors as described in Section 5.4. The procedure for the computation of zero-one target vectors and cost target vectors was described in Sections 5.2.4 and 5.3.4, respectively. These procedures use the forward algorithms developed in Chapter 2 in which the Wagner-Whitin cost structure specific features derived in Section 3.1 are included.

6.2.1 Preprocessing

We include a preprocessing step in which all elements of input and target vectors are scaled between 0 and 1. Such scaling has the advantage that all target mappings are of the form $g : [0, 1] \to [0, 1]$, such that we can use multi-layered perceptrons with sigmoidal units in all cases. Scaling has the additional advantage that it allows us to take identical weight initialization procedures and identical values of the learning parameters. This may reduce the amount of time spent on parameter tuning drastically. Furthermore, in this way, the generalization performance among different target vector types and data horizon lengths can be compared fairly.

Next we describe the scaling procedure in detail. For the input vectors, we exploit our foreknowledge of the demand distribution and divide each input by 200, the maximal possible value of demand in a period. The zero-one target vectors obviously need no further scaling. The scaling of the cost target vectors is less straightforward. We use the worst-case result presented in Proposition 3.2 for the Wagner-Whitin cost structure. This result states that $\Delta_m(p) \leq S$, where S denotes the setup cost and $\Delta_m(p)$ denotes the excess cost (over infinite-horizon m-optimality) of decomposing the off-line problem at period p; see also Section 2.7 and Section 5.3. Consequently ψ_m is bounded and the appropriate scaling is obtained by dividing the elements of all cost target vectors by $S = 200$. The only remaining difference after preprocessing between learning with zero-one target vectors and learning with cost target vectors is that the network horizon-selection is determined by the output unit with maximum response and the output unit with minimum response, respectively. This difference is removed by applying the transformation $1 - x$ to the elements of the scaled cost target vectors.

6.2.2 Learning algorithm

After some experimentation with different settings of the parameters of the learning algorithm, we have chosen the following setting. The initial values of the weights are drawn from a uniform distribution on $[-1, 1]$. We use the sequential version of gradient descent with momentum term using the sum-of-squares error function (4.12) with learning rate $\eta = 0.1$ and momentum term $\mu = 0.9$. To develop a network, the learning algorithm is executed for 625,000 iterations. During execution, the sum-of-squares error is monitored on the validation set and the network with the lowest generalization error is kept. For more details on the learning algorithm we refer to Section 4.3.

6.2.3 Network topology selection

For the construction of the horizon-selection rules ρ_m^{MLP} and τ_m^{MLP}, we use multi-layered perceptrons with m inputs and m output units as implied by the structure of the learning examples. Furthermore, we take logistic sigmoid response functions. To determine a suitable network topology for each value of m, we investigate five network topologies of increasing network mapping capabilities consisting of networks with 0, m, $m + 2$, $m + 4$, and $m + 6$ hidden units, respectively. All network topologies have one hidden layer, except for the 0 hidden unit topology, which has no hidden layer. We develop ten networks for each of the $5 \times 9 = 45$ different combinations of data horizon lengths and network topologies. These ten networks are combined to form a committee by averaging over network outputs. For each committee we compute the sum-of-squares error on the validation set. Both the horizon-selection rules ρ_m^{MLP} and τ_m^{MLP} are obtained by selecting the committee with the lowest sum-of-squares error on the validation set.

6.2.4 Learning curves

The development of the sum-of-squares error on the learning set and the validation set during execution of the learning algorithm is called the *learning curve*. We investigate the learning curves for learning with both zero-one target vectors and cost target vectors. To enable a comparison between curves for different data horizon lengths, we monitor the average sum-of-squares error per output unit. This quantity is obtained by dividing the sum-of-squares error on each of the sets by m and by averaging over the ten multi-layered perceptrons with best network topology. In Figure 6.1 we plot this quantity for three values of m. In all cases the sum-of-squares error on the learning set decreases in the number of iterations. In some cases we observe that the sum-of-squares error on the validation set initially decreases, but starts to increase after some number of iterations. This effect, known as *over-fitting*, only occurs for zero-one target values, where it increases with m. Most networks converged within 200, 000 iterations. Comparing the learning curves for zero-one and cost target vectors with equal m, we observe that for the latter, the curves are smoother, have smaller variation, and show no over-fitting.

6.3 K-nearest-neighbors

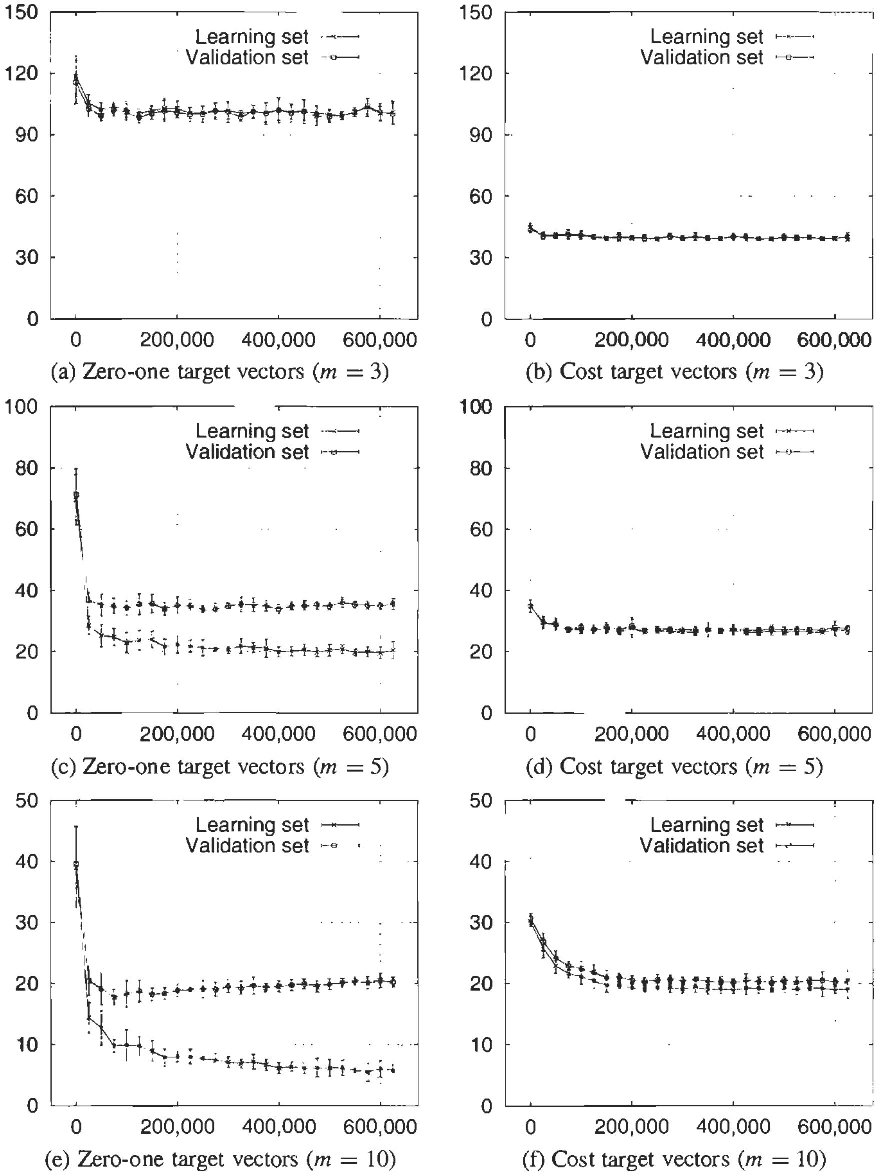
An alternative supervised learning technique from statistical classification is K-nearest-neighbors. First, we describe this technique in general and show that it can be used for estimation of posterior probabilities. Next we apply the K-nearest-neighbors technique to obtain approximate horizon-selection rules for the classification rate objective and the excess cost objective, respectively. Throughout this chapter, these *KNN-based* horizon-selection rules are used as a reference for the MLP-based horizon-selection rules.

6.3.1 K-nearest-neighbors estimation

Consider the general classification problem defined in Section 4.6, where the objective is to maximize the expected classification rate. Given are an integer K and a set of N examples of objects, characterized by their feature vectors $\mathbf{x} \in \Omega$ and their class labels $l \in \mathcal{L}$, and distributed according to $f_{\mathbf{X},Y}$. We call this set the *learning set*. Let N_l denote the number of class l learning examples.

Consider some new object with feature vector $\mathbf{x} \in \Omega$. We draw a hypersphere around $\mathbf{x}$ exactly enclosing K learning examples. Let the volume of the hypersphere be denoted by $V_K(\mathbf{x})$ and let $Q_K(l, \mathbf{x})$ denote the number of class l learning examples contained in the hypersphere. Then the conditional probability density $f_{\mathbf{X}}(\mathbf{x} \mid Y = l)$ can be estimated by

$$\frac{Q_K(l, \mathbf{x})}{N_l V_K(\mathbf{x})},$$



(a) Zero-one target vectors ($m = 3$)

(b) Cost target vectors ($m = 3$)

(c) Zero-one target vectors ($m = 5$)

(d) Cost target vectors ($m = 5$)

(e) Zero-one target vectors ($m = 10$)

(f) Cost target vectors ($m = 10$)

Figure 6.1. *Learning curves for different target vectors. The horizontal axis denotes the number of iterations of the learning algorithm. The vertical axis denotes the averages of the sum-of-squares error per output unit of the ten multi-layered perceptrons with optimal network topology with respect to the learning set.*

the probability density $f_{\mathbf{X}}(\mathbf{x})$ can be estimated by

$$\frac{K}{N V_K(\mathbf{x})},$$

and the prior probability $\mathrm{P}\{Y = l\}$ can be estimated by

$$\frac{N_l}{N}.$$

By substituting these estimates into Bayes' formula (4.37) we obtain an estimate for the posterior probabilities $\mathrm{P}\{Y = l \mid \mathbf{X} = \mathbf{x}\}$ given by

$$\frac{Q_K(l, \mathbf{x})}{K}.$$

The classification rule that for a given $\mathbf{x}$ assigns the class for which $Q_K(l, \mathbf{x})/K$ is largest is called the *K-nearest-neighbors classification rule.* For asymptotic results on the convergence of this rule to the Bayes rate when N goes to infinity we refer to the textbook by Duda & Hart [1973]. For finite N only negative results exist. It can, for instance, be shown that the convergence can be arbitrarily slow requiring a number of learning examples that grows exponentially with the input dimension Duda & Hart [1973].

6.3.2 Parameter selection and generalization

An important issue is the problem of selecting K. We remark that there is a relation between selecting an appropriate network topology in the context of multi-layered perceptrons and selecting an appropriate value for K. Both parameters control the level of flexibility of the model with respect to the data. Therefore the dependency between the optimal value for K and the learning set can also be characterized by the bias-variance trade-off as discussed in Section 4.5.1. If K is too large, some of the underlying structure is smoothed out (high bias). If K is too small, the obtained model is very sensitive to the learning set leading to over-fitting (high variance). Usually, K is determined as follows: calculate the performance of the classification rule on an independent validation set for a number of values of K and choose the one with the best performance on the validation set, i.e., the best generalization capabilities.

6.3.3 KNN-based horizon-selection rules

Similar to the development of the MLP-based horizon-selection rules in Chapter 5 we can apply the K-nearest-neighbors technique to estimate posterior probabilities and conditional expectations. Below we derive horizon-selection rules for both the classification rate objective and the excess cost objective.

Classification rate objective. Suppose we have a set of N learning examples with zero-one target vectors. Let K be some integer value with $K \leq N$. Given a demand vector $\mathbf{x} \in \Omega_m$, we can estimate the posterior probabilities $\mathrm{P}\{Y_m = l \mid \mathbf{X}_m = \mathbf{x}\}$ by

$$\frac{Q_K(l, \mathbf{x})}{K},$$

where $Q_K(l, \mathbf{x})$ denotes the number of class l learning examples contained in the hypersphere around $\mathbf{x}$ exactly enclosing K learning examples. The mapping $\rho_m^{\mathrm{KNN}} : \Omega_m \rightarrow \mathcal{L}_m$ given by

$$\rho_m^{\mathrm{KNN}}(\mathbf{x}) = \arg\max_{l \in \mathcal{L}_m} \frac{Q_K(l, \mathbf{x})}{K} \tag{6.4}$$

can thus be seen as an approximative horizon-selection rule with respect to the classification rate objective.

Excess cost objective. Suppose we have a set of N learning examples with cost target vectors denoted by $(\mathbf{x}^{(i)}, \mathbf{c}^{(i)})$ for $i = 1, \ldots, N$. Let K be some integer value with $K \leq N$. Given a demand vector $\mathbf{x} \in \Omega_m$, we can estimate the conditional expectation $\mathrm{E}\{C_m(l) \mid \mathbf{X}_m = \mathbf{x}\}$ by

$$\frac{1}{K} \sum_{i \in \mathcal{J}_K(\mathbf{x})} c_l^{(i)},$$

where $\mathcal{J}_K(\mathbf{x})$ denotes the set of indices of the learning examples contained in the hypersphere around $\mathbf{x}$ exactly enclosing K learning examples. The mapping $\tau_m^{\mathrm{KNN}} : \Omega_m \rightarrow \mathcal{L}_m$ given by

$$\tau_m^{\mathrm{KNN}}(\mathbf{x}) = \arg\min_{l \in \mathcal{L}_m} \frac{1}{K} \sum_{i \in \mathcal{J}_K(\mathbf{x})} c_l^{(i)} \tag{6.5}$$

can thus be seen as an approximative horizon-selection rule with respect to the excess cost objective.

Parameter selection. A suitable value for K is determined as follows. For the classification rate objective we compute the classification rate of ρ_m^{KNN} on the validation set for $K = 1, \ldots, 20$. This is done using the zero-one target vectors of the combined learning examples. We choose that value of K with the highest classification rate. For the excess cost objective we compute the average excess cost of τ_m^{KNN} on the validation set for $K = 1, \ldots, 20$. This is done using the cost target vectors of the combined learning examples. We choose that value of K with the lowest average excess cost.

m	ρ_m^{MLP}			ρ_m^{KNN}
	avg	ind	com	
2	75.32	75.68	75.08	74.00
3	81.72	81.92	81.16	80.76
4	88.95	89.88	89.72	88.04
5	91.87	92.28	92.44	89.52
6	92.15	92.68	93.08	87.40
7	91.32	92.28	92.76	85.60
8	91.24	91.92	92.80	84.60
9	91.10	92.12	92.24	84.16
10	91.28	91.56	92.36	82.40

Table 6.1. *Classification rate in percentages on test set for horizon-selection rules ρ_m^{MLP} and ρ_m^{KNN}. The entries for ρ_m^{MLP} represent averages of ten networks (avg), best individual networks (ind), and committees (com).*

6.4 Generalization with zero-one target vectors

This section addresses generalization with zero-one target vectors and is organized as follows. Section 6.4.1 assesses the generalization capabilities of the horizon-selection rules ρ_m^{MLP} and ρ_m^{KNN}. Class overlap is studied in Section 6.4.2. Sections 6.4.3 and 6.4.4 investigate model specific and problem specific conditions for good generalization, respectively. Finally, Section 6.4.5 gives some conclusions.

6.4.1 Generalization assessment

We assess the generalization capabilities of the horizon-selection rules ρ_m^{MLP} and ρ_m^{KNN} for data horizon lengths $m = 2, \ldots, 10$, by calculating the corresponding classification rates on the independent test set. Table 6.1 presents these results.

The classification rate for ρ_m^{MLP} increases from 75% for $m = 2$ to 93% for $m = 6$. For $m \geq 7$ the classification rate slowly deteriorates to 92% for $m = 10$. The overall performance of the committees is better than the average performance of the individual networks. In most cases, a committee performs even better than the best individual network.

The classification rate for ρ_m^{KNN} increases from 74% for $m = 2$ to 89% for $m = 5$. For $m \geq 6$ the classification rate rapidly deteriorates to 82% for $m = 10$.

Before discussing these results we remark that we do not know the Bayes rates $r(\rho_m^*)$ for $m = 2, \ldots, 10$. Therefore it is difficult to see what is good generalization and what is not. This problem is partly overcome by using the KNN-based horizon-selection rule as a reference. It is clear that the overall performance of ρ_m^{MLP} is better than that of ρ_m^{KNN}. For larger values of m the classification rates for both ρ_m^{MLP} and ρ_m^{KNN} fall short of our expectations. Apparently, one or more of the necessary condition for good generalization have been violated. Furthermore, it seems that the amount of violation increases with m and that the K-nearest-neighbors technique is more sensitive to this violation than multi-layered perceptrons.

6.4.2 Class overlap

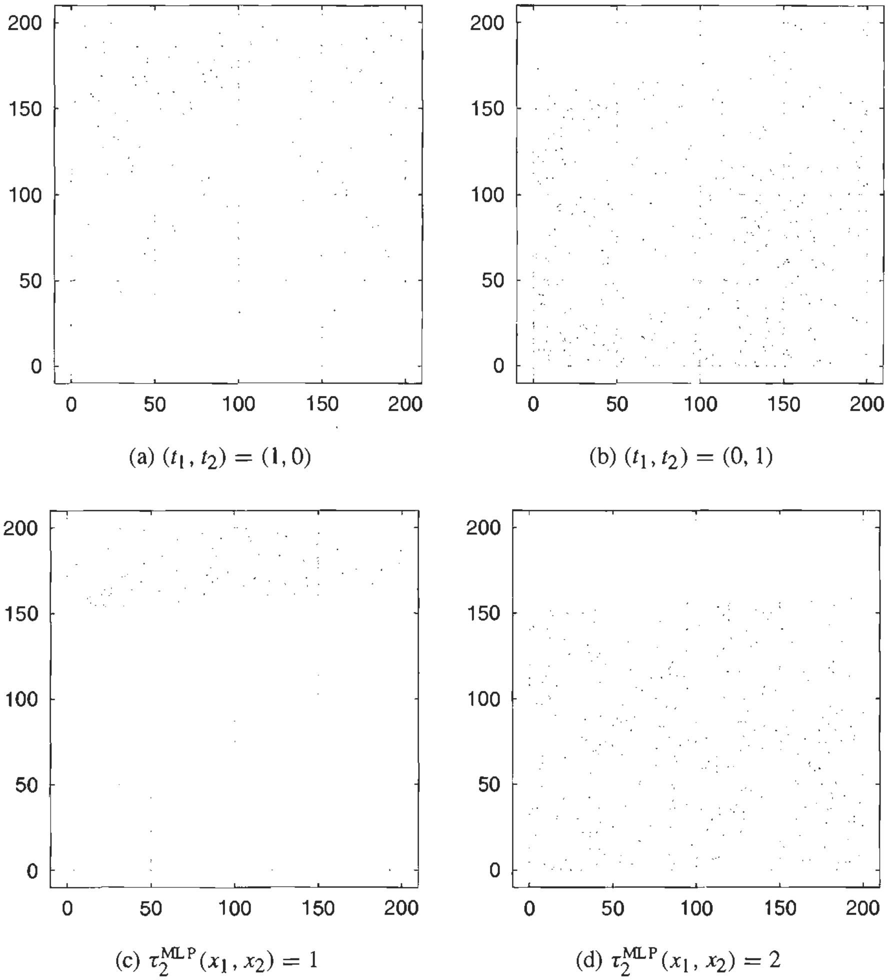
Section 4.6.4 discussed the phenomenon class overlap, which strongly effects the characteristics of a classification problem. In this subsection we investigate the presence and the amount of class overlap for different values of m. The presence of class overlap is evident from Figure 6.2, which gives a graphical representation of the 7,500 learning examples generated for the case $m = 2$. The two axes represent demands in subsequent periods. This figure was obtained by plotting the input vectors of the learning examples with 2-optimal simple planning horizon equal to 1 or 2 in (a) or (b), respectively. As an illustration we give a graphical representation of the horizon-selection rule constituted by the best committee for the case $m = 2$ in (c) and (d). To understand the origin of this overlap, we briefly summarize the procedure used to compute these learning examples.

Each learning example is obtained by applying a forward algorithm to a sequence of periods in demand history, labeled $1, 2, \ldots$. On termination, the forward algorithm returned a minimal m-optimal simple planning horizon s for some forecast horizon n. The input vector of the learning example was obtained by taking the first m demands $d_1, \ldots, d_m$. The remaining demands $d_{m+1}, \ldots, d_n$, necessary for determining s, were discarded. It is this discarding of demand information that "introduces" class overlap in the learning examples. We remark that this discarding of demand information is a consequence rather than a source of class overlap, since, if such class overlap exists, it is inherently present in the underlying classification problem.

In Section 4.6.4 we derived an expression for the amount of class overlap in a classification problem called the overlapping coefficient. Let λ_m denote the overlapping coefficient for the classification problem with data horizon length m. Then, by definition, we have $\lambda_m = 1 - r(\rho_m^*)$, where $r(\rho_m^*)$ denotes the Bayes rate. We can estimate $r(\rho_m^*)$ in the following two ways.

The first way is based on the observation that for any horizon-selection rule ρ we have $r(\rho) \leq r(\rho_m^*)$. So the expected classification rate $r(\rho)$ of any horizon-selection rule ρ provides a lower bound for the Bayes rate. Just as we did when assessing generalization capabilities, we can estimate $r(\rho)$ by calculating the classification rate of ρ on the independent test set.

In the second way we use the lower bound on the Bayes rate provided by The-



(a) $(t_1, t_2) = (1, 0)$

(b) $(t_1, t_2) = (0, 1)$

(c) $\tau_2^{\mathrm{MLP}}(x_1, x_2) = 1$

(d) $\tau_2^{\mathrm{MLP}}(x_1, x_2) = 2$

Figure 6.2. *Graphical representation of 7,500 learning examples with zero-one target vectors for the case $m = 2$. For each learning example $((x_1, x_2), (t_1, t_2))$, the input vector (x_1, x_2) is plot in (a), if $(t_1, t_2) = (1, 0)$, or in (b), if $(t_1, t_2) = (0, 1)$. (c) and (d) show the behavior on these examples of the best committee τ_2^{MLP}. Input vector (x_1, x_2) is plot in (c), if $\tau_2^{\mathrm{MLP}}(x_1, x_2) = 1$, or in (d), if $\tau_2^{\mathrm{MLP}}(x_1, x_2) = 2$.*

orem 5.2. We recall that $\mathrm{P}\{Z_m \leq m\}$ denotes the probability that a minimal m-optimal simple forecast horizon prevails within the data horizon. In Theorem 5.2 it was shown that, under the condition that a perfect forward algorithm exists, $r(\rho_m^*) \geq \mathrm{P}\{Z_m \leq m\}$. We can use this bound, because Chand & Morton [1986] derived such a perfect forward algorithm for the Wagner-Whitin cost structure. For each m with $2 \leq m \leq 10$, we estimate $\mathrm{P}\{Z_m \leq m\}$ by the fraction $\mathrm{LB}(m)$ of the 7,500 available learning examples (for which we stored the corresponding minimal m-optimal simple forecast horizons) that has a minimal m-optimal simple forecast horizon smaller than or equal to m. In Section 5.2.3 it was shown that in case $m = 1$ we have $r(\rho_1^*) = \mathrm{P}\{Z_1 \leq 1\} = 1$ and therefore $\mathrm{LB}(1) = 1$.

Let $\mathrm{MLP}(m)$ denote the classification rate on the test set of the best MLP-based horizon-selection rule we found for data horizon length m. Let $\mathrm{KNN}(m)$ denote the classification rate on the test set of the best KNN-based horizon-selection rule we found for data horizon length m. In case $m = 1$, the problem of selecting an optimal optimization horizon is trivial and therefore it is obvious to define MLP(1)=1 and KNN(1)=1.

Figure 6.3 shows the values of $\mathrm{MLP}(m)$, $\mathrm{KNN}(m)$, and $\mathrm{LB}(m)$, for $1 \leq m \leq 10$. The best estimate of $r(\rho_m^*)$ as function of m is now obtained by taking the maximum of $\mathrm{MLP}(m)$, $\mathrm{KNN}(m)$, and $\mathrm{LB}(m)$. The difference between $\mathrm{MLP}(m)$ and $\mathrm{LB}(m)$ for $m > 6$ is an indication of the optimality gap, which for instance could be closed by adding more learning examples.

6.4.3 Model specific conditions

A necessary condition for good generalization is that the level of model flexibility is optimal with respect to the learning set. This is certainly not a sufficient condition for good generalization, since the learning set may be too small or not representative, for instance. To investigate this condition we plot generalization capabilities as a function of the model flexibility in Figure 6.4 for different data horizon lengths m. The classification rate of ρ_m^{MLP} on the test set as function of the number of hidden units is plot in Figure 6.4(a) and the classification rate of ρ_m^{KNN} on the test set as function of K is plot in Figure 6.4(b). Next we discuss these plots.

Multi-layered perceptrons. We recall that during execution of the learning algorithm the sum-of-squares error on the validation set (generalization error) is monitored and a copy of the network with the lowest generalization error is kept. After termination of the algorithm this copy is restored. The advantage of this approach is that over-fitting due to a too small learning set or a too large model flexibility is overcome. From the learning curves (c) and (e) in Figure 6.1 the risk of over-fitting is clear. Consequently, the generalization capabilities as function of the number of hidden units have the following typical shape. Let N_m^* denote the smallest number

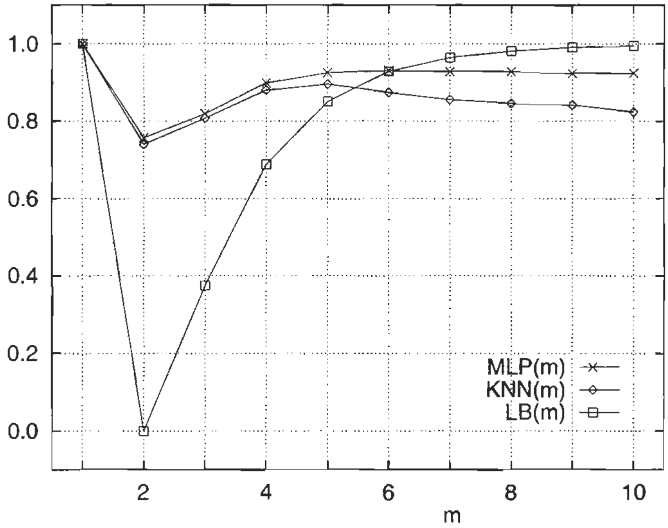
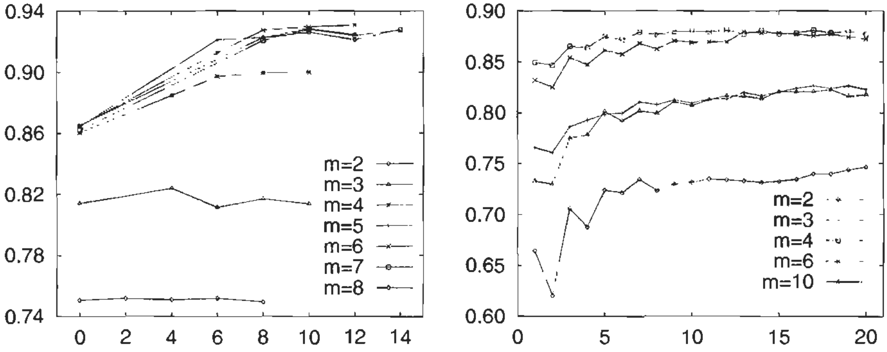


Figure 6.3. *Classification rate on test set as a function of m for best MLP-based horizon-selection rule and best KNN-based horizon-selection rule. Fraction LB(m) of the 7,500 available learning examples with a minimal m-optimal simple forecast horizon smaller than or equal to m.*



(a) Multi-layered perceptrons

(b) K-nearest-neighbors

Figure 6.4. *Generalization capabilities as function of the model flexibility. (a) Classification rate of ρ_m^{MLP} as function of the number of hidden units. (b) Classification rate of ρ_m^{KNN} as function of K.*

of hidden units for which the best possible generalization is achieved with data horizon length m. Suppose we increase the number of hidden units n starting at $n = 0$. Then for $n < N_m^*$ the generalization capabilities increase with n. For $n \geq N_m^*$ the generalization capabilities are approximately constant. For the different values of m, we observe such a shape in Figure 6.4(a). For example in the case $m = 6$ we have $N^* = 12$. We observe that N_m^* increases for $2 \leq m \leq 6$. For $m > 6$ we have $N_m^* \approx 10$.

K-nearest-neighbors. In Figure 6.4(b) the generalization capabilities of the horizon selection rule ρ_m^{KNN} are plot for different values of K and m. We observe that for small values of K the model flexibility is too high for the given learning set. The optimal value of K typically lies in the interval [10, 20].

Conclusion. From Figure 6.4 we infer that investigating more hidden units or larger values of K will most probably not result in significant improvements, upon which we conclude that the model specific conditions for good generalization are satisfied with respect to the learning sets. So the disappointing generalization capabilities for $m > 6$ cannot be explained in this way. As a consequence of that, one or more of the problem specific conditions for good generalization must be violated. This also explains the over-fitting during learning observed in the learning curves (c) and (e) in Figure 6.1. From the increase of N_m^* for $2 \leq m \leq 6$ we conclude that the *functional complexity* of the target mapping ϕ_m increases with m in the sense that more network mapping capabilities are required.

6.4.4 Problem specific conditions

The target mapping ϕ_m is completely characterized by the posterior probabilities $\mathrm{P}\{Y_m = k \mid \mathbf{X}_m = \mathbf{x}\}$ with $k \in \mathcal{L}_m$, or using Bayes' rule, by the prior probabilities $\mathrm{P}\{Y_m = k\}$ and the conditional probability density functions $f_{\mathbf{X}_m}(\mathbf{x} \mid Y_m = k)$ with $k \in \mathcal{L}_m$. The results presented in Section 5.2.3 imply that the characteristics of Y_m, given by the prior probabilities $\mathrm{P}\{Y_m = k\}$, change with the data horizon length m. Consequently, the characteristics of ϕ_m change with m, and we conjecture that, with these characteristics, the problem specific conditions for good generalization change. Below we investigate this conjecture.

Prior probabilities. We recall that the prior probability $\mathrm{P}\{Y_m = l\}$ denotes the probability that the minimal m-optimal simple planning horizon is equal to l. For each value of m we estimate $\mathrm{P}\{Y_m = l\}$ by the fraction $\alpha_m(l)$ with $l \in \mathcal{L}_m$, where $\alpha_m(l)$ denotes the fraction of the 7,500 available learning examples with data horizon length m that have zero-one target vector $\mathbf{e}_l$. These estimates are presented in percentages in Table 6.2. The entries for $m > 7$ are omitted. Note that the order cycle, as obtained by substituting the average demand level 100 in (3.3), equals the

m	$\alpha_m(1)$	$\alpha_m(2)$	$\alpha_m(3)$	$\alpha_m(4)$	$\alpha_m(5)$	$\alpha_m(6)$	$\alpha_m(7)$
2	29.13	70.87	-	-	-	-	-
3	20.45	55.09	24.45	-	-	-	-
4	19.24	53.17	22.53	5.05	-	-	-
5	19.09	53.00	22.33	4.84	0.73	-	-
6	19.05	52.97	22.32	4.84	0.73	0.08	-
7	19.05	52.97	22.32	4.84	0.73	0.08	0.00

Table 6.2. *Estimates $\alpha_m(l)$ of the prior probabilities $P\{Y_m = l\}$.*

optimization horizon with the highest prior probability. We observe that for $l \leq 6$ the fractions $\alpha_m(l)$ as function of m converge to fixed values. This behavior can be explained as follows.

Proposition 5.1 implies that, if a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n, random variables $Y_M, Y_{M+1}, \ldots$ are identically distributed. So, if such an upper bound M exists, the prior probabilities become independent of m for $m \geq M$. The observed behavior of $\alpha_m(l)$ as function of m can thus be explained by the existence of such a bound $M = 6$. However, it is unlikely that such a bound really exists. To illustrate this, we refer to the proof of Corollary 3.1 in which we derived a bound on the length of a subplan in an optimal production plan for the Wagner-Whitin cost structure. Since our demand is uniformly distributed on [0, 200], positive demand in a period can be arbitrarily small, and we can construct examples of n-period problems with arbitrarily large optimal subplans. Nevertheless, as can be concluded from the fractions $\alpha_m(l)$, the probability that a subplan in an optimal production plan has a length greater than 6 is very small. In fact of all 7,500 learning examples there was no one having an m-optimal simple planning horizon greater than 6. Consequently, the *empirical distributions* of Y_m for $m \geq 6$, which are represented by the fractions $\alpha_m(l)$, are identical. In our analysis of generalization with zero-one target vectors it thus makes sense to distinguish between the cases $2 \leq m < 6$, in which the empirical distribution of Y_m changes with m, and $m \geq 6$ in which the empirical distributions of Y_m are identical.

Curse of dimensionality. To understand the deteriorating classification rates for $m \geq 6$ as observed in Table 6.1 and Figure 6.3 we combine the outcome of the investigation of the prior probabilities with the results of Chapter 5.

The analysis of the prior probabilities implies that the zero-one target vectors of the combined learning examples are independent of m for $m \geq 6$. From this we deduce that the deteriorating classification rates for $m \geq 6$ must be caused by the increased input vector dimensionality. At first sight this looks very counterintuitive, since including more relevant demand information should enable better horizon-selection. However, we conjecture that a phenomenon occurred that is related to what Bellmann [1961] calls the "curse of dimensionality". In the context of generalization, the curse of dimensionality can be understood as follows. Suppose we have some finite learning set. Then increasing the dimensionality of the input space by adding new features rapidly leads to the point where the data is very sparse and the learning examples provide a very poor representation of the target mapping. In such a case, good generalization is only possible if the target mapping is a smooth function of the input vectors, such that it is possible to infer the target values at intermediate points, where no data is available, by interpolation.

Our conjecture is confirmed by Proposition 6.1, in which it was shown that, under the condition that a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n, the smoothness of the target mapping ϕ_m decreases with m for $m \geq M$.

Apparently, the K-nearest-neighbors technique is more sensitive to the smoothness of ϕ_m than supervised learning with multi-layered perceptrons. To further investigate this, we plot the classification rate of ρ_m^{KNN} as function of the learning set size for $m = 5$ and $m = 10$ in Figure 6.5. We take $K = 20$. In case $m = 5$, approximately ten times as many learning examples are needed to obtain a classification rate that is comparable with the classification rate of ρ_m^{MLP}. In case $m = 10$, using twenty times as many learning examples is not nearly sufficient. The number of learning examples required for good generalization with ρ_m^{KNN} seems to grow exponentially with m. We conclude that ρ_m^{MLP} uses the available learning examples more efficiently than ρ_m^{KNN}.

It would be very interesting to investigate the effect of the number of learning examples on the generalization capabilities of the MLP-based horizon-selection rules, but doing this thoroughly would take an enormous amount of computing time. Some preliminary steps in this direction were made by Zwietering, Van Kraaij, Aarts & Wessels [1991].

6.4.5 Discussion

We investigated the generalization capabilities of the MLP-based horizon-selection rules for the classification rate objective as a function of the data horizon length. It turned out that, when we enlarge the data horizon, these generalization capabilities diminish. By analyzing the necessary conditions for good generalization, we were able to point out the non-smoothness of the target mapping as the main source of problems when learning with zero-one target vectors. We end this section with a summary of the conclusions on the basis of the following two typical cases.

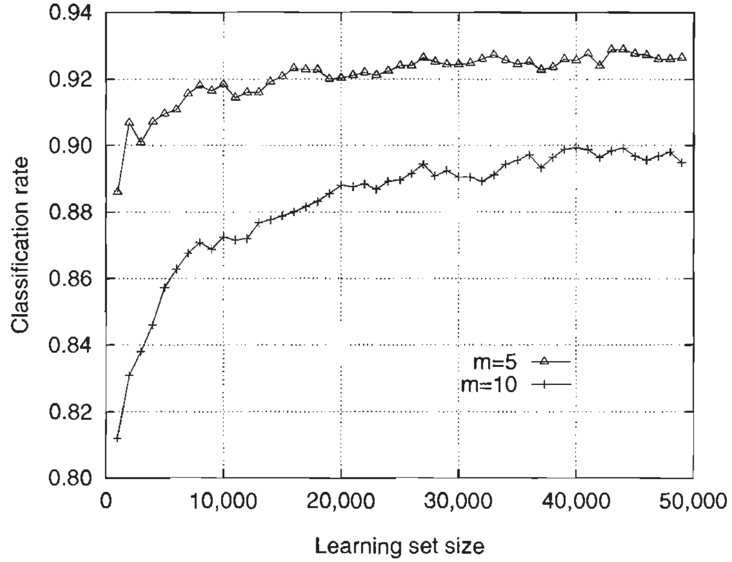


Figure 6.5. *Classification rate of horizon-selection rule ρ_m^{KNN} with $K = 20$ as function of the learning set size for $m = 5$ and $m = 10$.*

Large data horizons. In the case m is large, there is no class overlap and the target mapping can be viewed as a binary classification function $\phi_m : \Omega_m \to \{0, 1\}^m$. At each decision boundary ϕ_m is discontinuous. The degree of smoothness is determined by the number of decision boundaries and the more decision boundaries the less smooth ϕ_m. The functional complexity of the target mapping is typically high in the sense that many hidden units are required. In this ill-conditioned case, good generalization is difficult and requires lots of learning examples. An example is the case $m = 10$, in which 2,500 learning examples turned out to be insufficient for good generalization.

Small data horizons. Due to the presence of class overlap, in the case of small m, there are no discontinuities in the target mapping and most of the underlying structure is smoothed out. As a result of that, the functional complexity of the target mapping is typically low in the sense that only few hidden units are required. In this case, good generalization is possible with relatively few learning examples. An example is the case $m = 2$, in which 2,500 learning examples was more than sufficient for good generalization.

When we enlarge the data horizon from small to large, the smoothness of ϕ_m decreases, the functional complexity of ϕ_m increases, and generalization becomes more difficult requiring more learning examples.

m	τ_m^{MLP}			τ_m^{KNN}
	avg	ind	com	
2	8.79	8.64	8.75	9.55
3	5.20	5.17	5.21	5.59
4	2.14	2.04	1.91	2.05
5	1.01	0.90	0.84	1.48
6	0.81	0.65	0.80	1.17
7	0.55	0.48	0.53	0.81
8	0.39	0.33	0.37	0.64
9	0.29	0.23	0.27	0.58
10	0.28	0.22	0.19	0.54

Table 6.3. *Average excess cost on test set for horizon-selection rules τ_m^{MLP} and τ_m^{KNN}. The entries for τ_m^{MLP} represent averages of ten networks (avg), best individual networks (ind), and committees (com).*

6.5 Generalization with cost target vectors

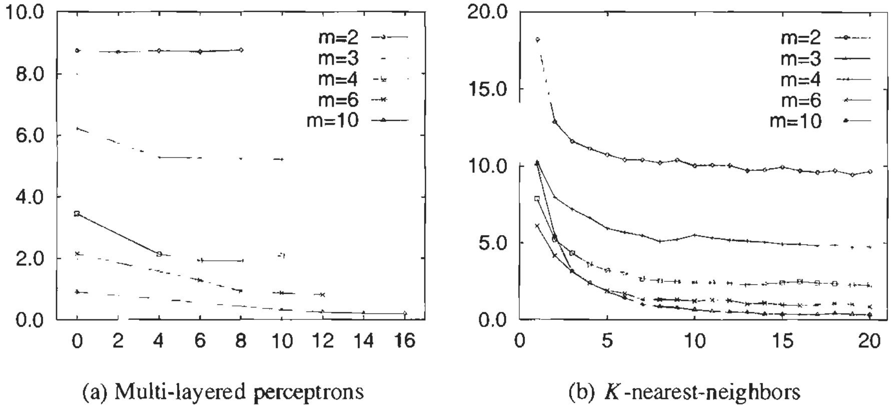
This section addresses generalization with cost target vectors; its organization is as follows. Section 6.5.1 assesses the generalization capabilities of the horizon-selection rules τ_m^{MLP} and τ_m^{KNN}. Model specific conditions and problem specific conditions for good generalization are studied in Sections 6.5.2 and 6.5.3, respectively. Finally, Section 6.5.5 gives some conclusions.

6.5.1 Generalization assessment

We assess the generalization capabilities of the horizon-selection rules τ_m^{MLP} and τ_m^{KNN} for data horizon lengths $m = 2, \ldots, 10$, by calculating the corresponding average excess cost on the independent test set. Table 6.3 presents these results. The average excess costs for both τ_m^{MLP} and τ_m^{KNN} are strictly decreasing in m. The overall performance of the committees is better than the average performance of the individual networks. The performance of the committees compares well with the performance of the best individual networks. The overall performance of the MLP-based horizon-selection rules is superior to that of the KNN-based horizon-selection rules. In the remainder of this section we further analyze these results.

6.5.2 Model specific conditions

We now investigate the model specific conditions for good generalization for the different approaches with respect to the learning set. Therefore, in Figure 6.6, we plot the generalization capabilities as a function of the model flexibility. For dif-



(a) Multi-layered perceptrons

(b) K-nearest-neighbors

Figure 6.6. *Average excess cost as function of the model flexibility. The horizontal axis represents the number of hidden units for multi-layered perceptrons and the value of K for K-nearest-neighbors.*

ferent data horizon lengths m, we plot the average excess cost of τ_m^{MLP} on the test set as function of the number of hidden units in (a) and the average excess cost of τ_m^{KNN} on the test set as function of K in (b). Looking at Figure 6.6 we conclude that the model specific conditions for good generalization are satisfied. The optimal number of hidden units N_m^* increases strictly with m indicating an increase of the functional complexity of the target mapping ψ_m. For K-nearest-neighbors, we observe that small values of K lead to poor generalization due to a too high model flexibility. The generalization capabilities increases smoothly with K. The results in Figure 6.6 suggest that slightly better generalization capabilities can be obtained by using more hidden units or higher values of K.

6.5.3 Problem specific conditions

The target mapping ψ_m is completely characterized by the conditional expectation of the excess cost $\mathrm{E}\{C_m(k) \mid \mathbf{X}_m = \mathbf{x}\}$ with $k \in \mathcal{L}_m$. From the results presented in Section 5.3.3, we know that the characteristics of random variables $C_m(k)$ with $k \in \mathcal{L}_m$ change with the data horizon length m. Consequently, the characteristics of ψ_m change with m. The effect of this change on the problem specific conditions for good generalization is determined by the cost structure (H, P). Below we examine the characteristics of random variables $C_m(k)$ by estimating their expectations $\mathrm{E}\{C_m(k)\}$.

The analogon of the prior probability $\mathrm{P}\{Y_m = k\}$ for the excess cost objective is the prior expectation $\mathrm{E}\{C_m(k)\}$. In words, $\mathrm{E}\{C_m(k)\}$ denotes the expected excess cost of decomposing the off-line problem at period k. For each m we esti-

m	$\beta_m(1)$	$\beta_m(2)$	$\beta_m(3)$	$\beta_m(4)$	$\beta_m(5)$	$\beta_m(6)$	$\beta_m(7)$
2	58.45	17.21	-	-	-	-	-
3	66.52	33.34	42.11	-	-	-	-
4	67.74	35.79	45.79	42.45	-	-	-
5	67.89	36.08	46.23	43.03	43.81	-	-
6	67.90	36.11	46.28	43.10	43.90	43.68	-
7	67.90	36.11	46.28	43.10	43.90	43.68	43.79

Table 6.4. *Estimates $\beta_m(k)$ of the prior expectations $E\{C_m(k)\}$.*

mate the prior expectations $\mathrm{E}\{C_m(k)\}$ with $k \in \mathcal{L}_m$ by averaging over the 7, 500 available cost target vectors. We denote the estimate of $\mathrm{E}\{C_m(k)\}$ by $\beta_m(k)$. Table 6.4 presents these estimates. The entries for $m > 7$ were omitted. Note that the order cycle, as obtained by substituting the average demand level 100 in (3.3), corresponds with the optimization horizon with the lowest prior expectation. We observe that $\beta_m(k)$ as function of m converges to a fixed value for $k \in \mathcal{L}_m$. This behavior can be explained as follows.

Proposition 5.3 implies that, if an upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n, random variables $C_M(k)$, $C_{M+1}(k)$, ... are identically distributed. So, if such an upper bound M exists, the prior expectations become independent of m for $m \geq M$. The observed behavior of $\beta_m(k)$ as function of m can thus be explained by the existence of such a bound $M = 6$. However, it is unlikely that such a bound really exists. Using similar arguments as put forward in Section 6.4.4, in our analysis, we distinguish between the cases $2 \leq m < 6$, in which the empirical distribution of $C_m(k)$ changes with m, and $m \geq 6$ in which the empirical distributions of $C_m(k)$ are identical.

6.5.4 Class overlap

The analogon of the overlapping coefficient λ_m for the excess cost objective with data horizon length m equals the Bayes cost $c(\tau_m^*)$. The following estimate is based on the observation that for any horizon-selection rule τ we have $c(\tau) \geq c(\tau_m^*)$. So the expected excess cost $c(\tau)$ of any horizon-selection rule τ provides an upper bound for the Bayes cost and can be estimated by calculating the average excess cost of τ on the independent test set. Let UB(m) denote the average excess cost on the test set of the best horizon-selection rule we found for data horizon length m. In case $m = 1$, the problem of selecting an optimal optimization horizon is trivial and therefore it is obvious to define UB(1)=0. In Figure 6.7 we plot the values of UB(m) for $1 \leq m \leq 10$. The observed behavior of UB(m) as a function of m supports Theorem 5.4 and Conjecture 5.1. These results state that $c(\tau_m^*)$ is

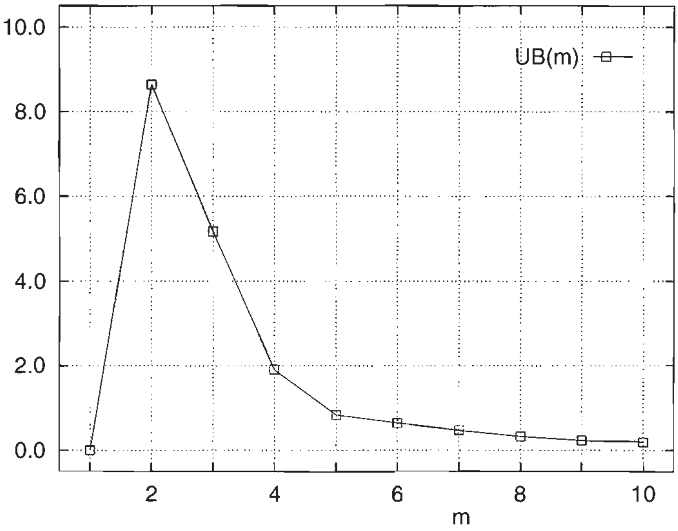


Figure 6.7. *Average excess cost on test set for best horizon-selection rule.*

decreasing in m for $m \geq 6$ and $\lim_{m\to\infty} c(\tau_m^*) = 0$ under the condition that an upper bound $M = 6$ exists.

6.5.5 Discussion

We studied the generalization capabilities of the MLP-based horizon-selection rule for the excess cost objective as a function of the data horizon length m. It turned out that good generalization is possible with cost target vectors for all m. We analyzed the necessary conditions for good generalization and observed that the number of hidden units required for good generalization increases strictly with m, indicating an increase of the functional complexity of the target mapping ψ_m. Despite this increase in functional complexity, there was no noticeable effect on the conditions for good generalization. In all cases 2,500 learning examples was sufficient for good generalization.

6.6 Conclusion

This chapter studied the generalization capabilities of the two MLP-based horizon-selection rules proposed in Chapter 5 for an on-line lot-sizing problem with Wagner-Whitin cost structure. Particularly, the effect of the length of the data horizon and the type of learning examples on the generalization capabilities were investigated. As a reference we developed two alternative horizon-selection rules based on K-nearest-neighbors estimation. Below, we give some conclusions.

MLP-based versus KNN-based. We conclude that, in case of generalization with zero-one target vectors, the use of multi-layered perceptrons is preferred to the K-nearest-neighbors technique; not only with respect to its generalization capabilities, but also with respect to its more efficient use of the available learning examples. In case of generalization with cost targets, multi-layered perceptrons were always slightly better.

If we compare the computational efficiency of supervised learning with multi-layered perceptrons and the K-nearest-neighbors technique, a distinction can be made between (i) the amount of effort that has to be put in before a good horizon-selection rule is obtained and (ii) the on-line processing speed. With respect to (i), both multi-layered perceptrons and K-nearest-neighbors are rather time consuming, since they require fine-tuning of the parameter controlling the model flexibility. The time to develop a multi-layered perceptron can be decreased by using improved learning algorithms and parallel processing. With respect to (ii), MLP-based horizon selection rules are very fast on-line compared to KNN-based horizon-selection rules. The reason therefore is that, in a KNN-based horizon-selection rule, the distance from the give demand vector to all learning examples in the learning set must be calculated. However, there also exist fast implementations of K-nearest-neighbors algorithms [Fukunaga & Narendra, 1975].

Zero-one targets versus cost targets. A possible disadvantage of using zero-one target vectors is that, although multiple simple planning horizons may exist within the data horizon, we only code the *minimal* one. In this way crucial information may be lost. One option is to allow multiple ones in the target vectors, however, in this way still no cost information is incorporated in the targets. These disadvantages can be overcome by using cost target vectors.

Generalization with zero-one target vectors turned out to be more difficult for large data horizons. The main source of difficulties is the smoothness of the corresponding target mapping. Due to the better conditions for good generalization, we expect that learning with cost target vectors requires less learning examples and can therefore be done faster.

Discussion. Although the above conclusions are in favor of using an MLP-based horizon-selection rule with cost target vectors, a fair comparison between the different approaches and the different types of learning examples cannot be made yet. This is because generalization capabilities are measured differently for the two objectives. For instance, a classification rate of 0.9 cannot be compared with an average excess cost of 5. One option is to calculate the average excess cost for all horizon-selection rules. But the most appropriate way of comparing the different horizon-selection rules is on the basis of the performance characteristics of their corresponding variable-horizon policies. In the next chapter we investigate these characteristics by means of an extensive empirical study.

7

Performance of variable-horizon policies for on-line lot-sizing

In Chapter 2 we introduced variable-horizon policies as a solution approach for on-line lot-sizing problems. Such policies determine the lot-sizes by repeatedly optimizing over some optimization horizon and implementing the lot sizes of the first subplan. A variable-horizon policy is completely determined by its horizon-selection rule, which determines the optimization horizon given the demands within the data horizon. In Chapter 5 we derived two horizon-selection rules based on supervised learning with multi-layered perceptrons. The purpose of this chapter is to investigate the performance characteristics of the variable-horizon policies constituted by these horizon-selection rules. Moreover, our intention is to give conclusions and recommendations with respect to the use of these variable-horizon policies. To that end we perform an extensive empirical study in which the MLP-based variable-horizon policies are compared with a benchmark of alternative variable-horizon policies on a rolling-horizon basis. This study focuses on the three cost structures introduced and analyzed in Chapter 3.

The chapter is outlined as follows. In Section 7.1 we discuss some related literature. Section 7.2 gives an outline of the experiments. In Section 7.3 we discuss policy performance evaluation. Section 7.4 investigates the potential of variable-horizon policies. In Section 7.5 we present the empirical results. Finally, in Section 7.6 we give some conclusions.

7.1 Related literature

The literature shows several studies that address the performance of variablehorizon policies on a rolling-horizon basis. These studies address, without exception, the Wagner-Whitin cost structure. Our design of experiments is inspired by the studies of Baker [1977], Blackburn & Millen [1980], Carlson, Beckman & Kropp [1982], and Zwietering, Van Kraaij, Aarts & Wessels [1991]. Below we discuss the main conclusions of these studies.

Baker [1977] investigated the impact of the length of the data horizon, the order cycle, and the demand characteristics on the performance of the fixed-horizon policy. He concludes that for a good performance of this policy the data horizon must be at least as large as the order cycle.

This work was extended by Carlson, Beckman & Kropp [1982], who included demand forecasting. They compare the fixed-horizon policy with three variable-horizon policies, each consisting of a simple demand forecasting technique and a forward algorithm. Demand is forecasted for as many periods as needed by the forward algorithm to detect a planning horizon. They conclude that forecasting becomes more beneficial as the variation in demand increases. Furthermore, they conclude that any reasonable extension of demand beyond the data horizon is feasible. They recommend, to let the length of the data horizon plus the number of forecasted demands exceed the order cycle.

Blackburn & Millen [1980] designed a set of experiments to investigate the impact of the length of the data horizon, the order cycle, and the variation in demand on the performance of the fixed-horizon policy and three different aggregation heuristics. It appears that, independent of demand characteristics, the performance of the aggregation heuristics becomes poorer as the variation in demand increases, whereas for data horizons larger than two order cycles, increasing variation in demand tends to improve the performance of the fixed-horizon policy. For reasons of cost effectiveness, they recommend the use of one of the aggregation heuristics when the information about future demand is limited. Typically, there is a data horizon length for which the fixed-horizon policy starts to dominate the aggregation heuristics. This value decreases when the demand variability increases.

Zwietering, Van Kraaij, Aarts & Wessels [1991] performed some experiments with multi-layered perceptrons for the rolling horizon version of the single-item lot-sizing model with Wagner-Whitin cost structure. The authors showed that multi-layered perceptrons can outperform the fixed-horizon policy and the aggregation heuristic of Silver & Meal [1973].

7.2 Outline of the experiments

This section outlines our experimental setup. The experiments evaluate the performance of a number of different variable-horizon policies for instances of three types of on-line lot-sizing problems. We vary the length of the data horizon between 2 and 10. The organization of this section is as follows. First, we characterize the instances under consideration by summarizing the corresponding demand processes and cost structures. Next, we describe the details of the variable-horizon policies we consoder. Finally, we describe how we evaluate policy performance.

7.2.1 Demand processes

We investigate the effect of the characteristics of the demand process on the performance of variable-horizon policies by considering three types of demand processes, i.e., uniformly distributed demand, Erlang distributed demand, and seasonal demand. These demand processes are stationary, i.e., their characteristics are constant in time.

Variability is on of the characteristic of the demand process that strongly effects the performance of a variable-horizon policy [Blackburn & Millen, 1980; Carlson, Beckman & Kropp, 1982]. The variability of a positive random variable X is measured by its *squared coefficient of variation* c_X^2 defined by

$$c_X^2 = \frac{\mathrm{var}\{X\}}{\mathrm{E}^2\{X\}}. \tag{7.1}$$

Moreover, there is a relationship between the variability of the demand process and the distribution of minimal planning and forecast horizons; see the work of Lundin [1973], Federgruen & Tzur [1994], Lundin & Morton [1975], and Morton [1981]. Since the MLP-based horizon-selection rules proposed in Chapter 5 depend on the distribution of minimal planning and forecast horizons, we expect that the performance of the variable-horizon policies constituted by these horizon-selection rules depends on the variability of the demand process. For these reasons, we consider a number of coefficients of variation. Below we describe these demand processes in more detail.

Uniformly distributed demand. Suppose the demand in period t is uniformly distributed with mean μ and range R, i.e., the demand d_t is uniformly distributed on $[\mu - \frac{1}{2}R, \mu + \frac{1}{2}R]$. Then the corresponding squared coefficient of variation is given by

$$c_{d_t}^2 = \frac{R^2}{12\mu^2}. \tag{7.2}$$

We use uniformly distributed demands with mean $\mu = 100$ and ranges $R = 75$, 150, and 200. The corresponding squared coefficients of variation are equal to

$c^2_{d_t} = 0.05$, 0.19, and 0.33, respectively. For notational reasons we label the three uniform distributions as U75, U150, and U200.

Erlang distributed demand. Suppose the demand in period t is Erlang-k distributed with parameter λ. Then the mean and the squared coefficient of variation are given by

$$\mathrm{E}\{d_t\} = \frac{k}{\lambda}, \tag{7.3}$$

$$c^2_{d_t} = \frac{1}{k}. \tag{7.4}$$

We generate Erlang demands with (approximately) equal mean and variances as the abovementioned uniform distributions by substituting $\mathrm{E}\{d_t\} = \mu$ and (7.2) into (7.3) and (7.4) to obtain

$$k = \max\left\{1, \left\lfloor \frac{12\mu^2}{R^2} + \frac{1}{2} \right\rfloor\right\}, \tag{7.5}$$

$$\lambda = \frac{12k\mu}{R^2}. \tag{7.6}$$

We label the three Erlang distributions as E75, E150, and E200.

Seasonal demand. We generate seasonal demands using the formula

$$d_t = (\mu - \frac{1}{2}R)\sin\left(\frac{2\pi t}{T}\right) + u_t, \tag{7.7}$$

where T denotes the *cycle length*, and u_t is a uniformly distributed random variable with mean $\mu = 100$ and range $R = 75$. It is obvious that

$$\mathrm{E}\{d_t\} = \mu + (\mu - \frac{1}{2}R)\sin\left(\frac{2\pi t}{T}\right). \tag{7.8}$$

Let $D_{t,t+T}$ denote the cumulative demand from period t up to period $t+T-1$, i.e., the demand during a cycle. Then one easily verifies that $\mathrm{E}\{D_{t,t+T}\} = \mu T$ and the expected demand per period during a cycle is μ. We use seasonal demand processes with cycle lengths $T = 3$ and 6. These are labeled as S3 and S6, respectively.

7.2.2 Cost structures

We study four cost structures, i.e., two Wagner-Whitin cost structures, one cost structure with overtime, and one cost structure with purchasing. All cost structures have the linear holding cost function (3.2). We normalize the holding cost h to 1. The production cost functions for the Wagner-Whitin cost structure, the cost structure with overtime, and the cost structure with purchasing are defined by (3.1), (3.4), and (3.6), respectively. The Wagner-Whitin cost structures were selected to yield order cycles n^* of 2 and 4 periods as determined by (3.3) with $D = \mu = 100$. To obtain the appropriate order cycles, S was set equal to 200 and 800, respectively. For the two-source models with overtime and purchasing, constructing reasonable cost structures is less straightforward. After some experimentation with these models, we choose two cost structures that showed a reasonable number of lot sizes requiring overtime or purchasing; see also the work of Dixon, Elder, Rand & Silver [1983] and Suurmond [1996]. For the cost structure with overtime we set the overtime premium r, the setup cost S, and the regular time production capacity C equal to 1, 425, and 200, respectively. The corresponding order cycle, as determined by (3.5) with $D = \mu = 100$, is equal to 2. For the cost structure with purchasing we take the purchase premium r, the setup cost S, and the in-house production capacity C equal to 10, 100, and 125, respectively. The corresponding order cycle, as determined by (3.7) with $D = \mu = 100$, is equal to 1.

7.2.3 Learning examples

For all $8 \times 4 \times 9 = 288$ combinations of demand processes, cost structures, and data horizon lengths, we generate two sets of 2,500 combined learning examples, i.e., one learning set and one validation set for monitoring the generalization capability during execution of the learning algorithm. These combined learning examples contain both zero-one target vectors and cost target vectors as described in Section 5.4. We remark that we generated demand for as many periods as necessary to compute 5,000 learning examples, which is approximately 5,100 periods in all cases. The procedures for the computation of zero-one target vectors and cost target vectors have been described in Sections 5.2.4 and 5.3.4, respectively. These procedures use the forward algorithms developed in Chapter 2 in which the cost structure specific features derived in Chapter 3 have been included.

There are two reasons for taking the same number of learning examples in all 288 cases. First, although we expect that in most cases the number of learning examples required for good generalization is smaller than 2,500, it takes too much time to investigate the required minimal number of learning examples for all 288 cases; see also Chapter 6. Second, we want to maintain as much as possible the same conditions for the 288 experiments. Note that, in practice, the availability of learning examples may be limited and one may be unable to afford the luxury of keeping aside a validation set. In such cases cross-validation can be used; see also Section 4.5.

7.2.4 MLP-based variable-horizon policies

The procedure to obtain the variable-horizon policies constituted by the MLP-based horizon-selection rules ρ_m^{MLP} and τ_m^{MLP} developed in Chapter 5 is, except for some minor differences, identical for each of the 288 combinations of different demand processes, cost structures, and data horizon lengths. Below, we describe this procedure and point out these differences.

Preprocessing. A preprocessing step is included in which all elements of the input and target vectors are scaled between 0 and 1. Next, we remove the difference in interpretation of network outputs between zero-one target vectors and cost target vectors by applying the transformation $x \rightarrow 1 - x$ to the elements of the scaled cost target vectors. In this way, all target mappings are mappings of the form $g : [0, 1] \rightarrow [0, 1]$. This enables the use of sigmoidal units, identical weight initialization procedures, and identical learning parameter values in all cases; see also Section 6.2.

Next we describe the scaling procedure. For the scaling of the input vectors for uniformly distributed and seasonal demand, we use the property that these demand processes are bounded. We divide the input vectors for uniformly distributed demand by $\mu + \frac{1}{2}R$, and the input vectors for seasonal demand by 2μ. Erlang distributed demand is unbounded. In that case we scale by dividing the input vectors by the largest observed demand realization. The zero-one target vectors obviously need no further scaling. The scaling of the cost target vectors is less straightforward. For the Wagner-Whitin cost structure we use the worst-case result presented in Proposition 3.2. This result states that $\Delta_m(p) \leq S$, where S denotes the setup cost and $\Delta_m(p)$ denotes the excess cost (over infinite-horizon m-optimality) of decomposing the off-line problem at period p; see also Section 2.7 and Section 5.3. Consequently, the appropriate scaling is obtained by dividing the elements of all cost target vectors by S. We were unable to derive such results for the other cost structures and therefore scale the elements of the cost target vector by the largest observed value of $\Delta_m(p)$.

Learning algorithm. The initial values of the weights of the multi-layered perceptron are drawn from a uniform distribution on $[-1, 1]$. We use the sequential version of gradient descent with momentum term using the sum-of-squares error function (4.12) with learning rate $\eta = 0.1$ and momentum term $\mu = 0.9$. The learning algorithm applies 2,500,000 iterations for each multi-layered perceptron. During execution the sum-of-squares error is monitored on the validation set and the network with the lowest error is kept.

Network topology selection. Because of the size of our experimental setup, manual tuning of the neural network topology is simply infeasible. Therefore, in our experiments, we use a predetermined set of neural network topologies. The risk of such an approach is that we may end up with a suboptimal policy, since for instance adding more hidden units would yield a better performance. On the other hand, it is interesting to see what results can be obtained in this way, since in practice there is often no time for fine tuning. Next we describe the network topologies we consider in our experiments.

Suppose we have a data horizon of length m. Then we use multi-layered perceptrons with m inputs and m output units as implied by the structure of the learning examples. Furthermore, we take logistic sigmoid response functions (4.9). To determine a suitable network topology, we investigate five network topologies with increasing network mapping capabilities, i.e., with increasing number of hidden units. We expect that two-source models require more hidden units than single-source models which we take into account as follows. For the single-source models we investigate network topologies with 0, $m, m+2$, $m+4$, and $m+6$ hidden units; for the two-source models we investigate network topologies with 0, m, $m+3$, $m+6$, and $m+9$ hidden units. Except for the topology with zero hidden units, which has no hidden layer, all network topologies have one hidden layer. For each of the five different network topologies we develop ten networks. These ten networks are combined to form a committee by averaging over network outputs. For each committee we compute the sum-of-squares error on the validation set. Both the horizon-selection rules ρ_m^{MLP} and τ_m^{MLP} are obtained by selecting the committee with the lowest sum-of-squares error.

7.2.5 KNN-based variable-horizon policies

Next we describe the procedure to obtain the variable-horizon policies constituted by the KNN-based horizon-selection rules ρ_m^{KNN} and τ_m^{KNN}; see Section 6.3. Suppose we have a data horizon of length m. Horizon-selection rule ρ_m^{KNN} is obtained by computing the classification rate on the validation set for $K = 1, \ldots, 20$, and by choosing the value of K with the highest classification rate. This is done using the zero-one target vectors of the combined learning examples. Horizon-selection rule τ_m^{KNN} is obtained by computing the average excess cost on the validation for $K = 1, \ldots, 20$, and by choosing the value of K with the lowest average excess cost. This is done using the cost target vectors of the combined learning examples. Note that for both horizon-selection rules the procedure for determining K is taken independent of any cost structure, demand process, or data horizon length.

7.2.6 More variable-horizon policies

This subsection summarizes the more conventional variable-horizon policies that were proposed in Section 2.4.3 and that are used as a reference throughout this chapter. Furthermore, we introduce two variable-horizon policies that include forecasting and are based on the work of Carlson, Beckman & Kropp [1982]. For reasons of convenience we distinguish between policies with forecasting and policies without forecasting.

Myopic policies. In Section 2.4.3 we derived the following four variable-horizon policies from well-known heuristics for the Wagner-Whitin cost structure.

1. Economic order quantity policies (EOQ).
2. The least cost per unit time policy (PUT).
3. The least cost per unit product policy (PUP).
4. The fixed-horizon policy (FIX).

We remark that EOQ was derived for the three cost structures under consideration in Chapter 3.

Policies with forecasting. Carlson, Beckman & Kropp [1982] investigated the use of forecasting to extend the data horizon for the Wagner-Whitin cost structure. Their approach was to forecast demand for as many periods in the future as necessary for a forward algorithm to detect a planning horizon. They investigated several forecasting techniques ranging from relatively simple techniques, like extension of last period's demand, to more sophisticated techniques, like exponential smoothing with trend and seasonality. From their experiments it appears that a simple 5-period moving average works as well as the more sophisticated forecasting techniques. Consequently, they concluded that any reasonable extension of the data horizon will do as well as the more accurate ones.

Based on the work of Carlson, Beckman & Kropp [1982] we can use a forecasting technique to extend the data horizon in combination with our forward algorithm to detect an m-optimal simple planning horizon. In this way we derive two variable-horizon policies based on the following two simple forecasting techniques.

1. Take the average demand $\mu = 100$ as a forecast for the unknown future demands.
2. Use a 5-period moving average to forecast the unknown future demands.

We denote the corresponding policies by AVG and MVG, respectively. Both forecasting techniques generate a demand sequence with low variability. For such cases, we found that the forward algorithm often needs a large amount of demand information to stop. For that reason we terminate the forward algorithm after a maximum of 100 iterations. If the forward algorithm terminates within 100 iterations, the optimization horizon is taken equal to the m-optimal simple planning horizon. In case no m-optimal simple planning horizon is found, we select the smallest element from the m-optimal regeneration set $\mathcal{S}_m^m(100)$; see also Theorem 2.9. AVG can also be obtained by exploiting that the demand beyond the data horizon is constant. This was shown by Van Nunen & Wessels [1978], who used the infinite-horizon lot-sizing model with constant demand to derive an infinite-horizon optimal ending condition in case of a Wagner-Whitin cost structure; see Definition 2.5. Advantage of this approach is that, since no forward algorithm is required, it is computationally less demanding.

7.3 Policy performance evaluation

The performance of m-policies is usually evaluated empirically by applying them to different instances of on-line problems with a finite number of periods, called the *problem horizon.* A typical set-up is to consider 50 replications of 48-period problems [Baker, 1977; Carlson, Beckman & Kropp, 1982]. Such experiments are strongly effected by so-called *end-of-horizon effects* caused by the truncation of the infinite horizon to a finite horizon [Blackburn & Millen, 1980]. To minimize these effects in our experiments we use instances with large problem horizons.

Suppose we apply some variable-horizon policy π to an instance of an on-line lot-sizing problem with a data horizon of length m and a problem horizon of length n. Then, when the end of the problem horizon is reached, we have obtained lot-sizes for k periods of demand for some k with $n - m < k \leq n$; the remaining $n - k$ lot sizes are determined by choosing the final optimization horizon equal to $n - k$. We denote the corresponding total sum of production and holding cost by $C_m^\pi(n)$. The performance of π can be measured in a number of different ways. The most obvious way is to measure the *absolute* performance $C_m^\pi(n)$. Unfortunately, this is inappropriate for our purposes, because it does not allow for a comparison of the performance of variable-horizon policies with different data horizon lengths, problem horizon lengths, cost structures, and demand processes. To that end, a *relative* performance measure is required, in which the performance of π is expressed relative to some reference value. This section introduces two relative performance measures, i.e., *off-line performance* and *on-line performance.*

7.3.1 Off-line performance

We speak of *off-line performance* if we take the cost $f(n)$ of an optimal n-period plan as a reference. The corresponding off-line performance measure is the deviation from off-line optimality $\gamma_m^\pi(n)$, defined by

$$\gamma_m^\pi(n) = \frac{C_m^\pi(n)}{f(n)}, \tag{7.9}$$

which can be computed using the recursion (2.4). We assume that $f(n) > 0$. It then is obvious that $\gamma_m^\pi(n) \geq 1$.

Based on the stationarity of the demand processes under consideration we expect that $\gamma_m^\pi(n)$ converges to a limit value as n goes to infinity. We denote this

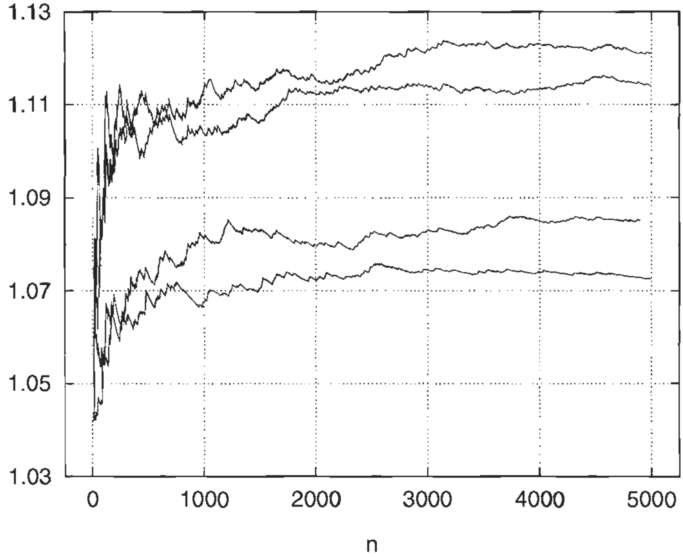


Figure 7.1. *Typical behavior of $\gamma_m^\pi(n)$ as a function of n for variable-horizon policies* PUT, AVG, FIX, *and* PUP, *using the Wagner-Whitin cost structure with order cycle 4, a data horizon of length 4, and uniformly distributed demands (U200).* PUT *performs best, followed by* AVG, FIX, *and* PUP.

limit value by γ_m^π. Figure 7.1 gives an example of the convergent behavior of $\gamma_m^\pi(n)$ for four variable-horizon policies as a function of n. In our experiments we assess policy performance by estimating these limit values. Apparently, good estimates of γ_m^π can be obtained by taking a problem horizon of 2,000-4,000 periods. In our experiments a problem horizon of 4,000 periods is used.

Discussion. The off-line performance characteristics of a variable-horizon policy π can be investigated by estimating γ_m^π for different data horizon lengths, demand processes, and cost structures. For instance, one may determine how much performance could be gained by enlarging the data horizon, e.g., by improving the information system. In this way we may determine the value of demand information for a given policy. Furthermore, the performance characteristics of different variable-horizon policies can be compared. For instance, if $\gamma_m^{\pi_1} < \gamma_m^{\pi_2}$ for two policies π_1 and π_2, then π_1 uses the available demand information *more efficient than* π_2 for data horizon length m. But, since it is unknown *how* efficient these policies use the available demand information, it is impossible to decide upon the off-line policy performance whether it is useful to search for better policies for a given data horizon length. For such issues, off-line optimality is inappropriate as a reference. For example, $\gamma_m^\pi = 4$ can be optimal for $m = 2$ in the sense that no variable-horizon policy exists that yields better results and $\gamma_m^\pi = 2$ can be not optimal for $m = 10$ in the sense that another policy exists that yields better results. We call this alternative notion of optimality *on-line optimality*, which is introduced next.

7.3.2 On-line performance

In analogy with the definition of off-line performance, it would be obvious to express the on-line performance of a variable-horizon policy π for an instance of an on-line lot-sizing problem with data horizon length m and problem horizon length n relative to the best performance of any m-policy for that instance by

$$\frac{C_m^\pi(n)}{\min\{C_m^{\pi'}(n) \mid \pi' \in \mathcal{P}_m\}}, \tag{7.10}$$

where $\mathcal{P}_m$ denotes the set of all possible m-policies. Nevertheless we express the on-line performance of π relative to the best performance of any variable-horizon policy, because the policies under consideration in this thesis are variable-horizon policies.

We recall that a variable-horizon policy is completely specified by its horizon-selection rule, which, for a data horizon length m, can be written as a function $g : \mathbb{R}^m \to \{1, \ldots, m\}$ of the m demands within the data horizon; see also Section 2.4.2. Let $\mathcal{H}_m$ denote the set of all possible horizon-selection rules that can be written as a function $g : \mathbb{R}^m \to \{1, \ldots, m\}$ of the m demands within the data horizon. For reasons of convenience we denote the variable-horizon policy corresponding with a horizon-selection rule g by $\pi(g)$. The on-line performance measure is the deviation $\phi_m^\pi(n)$ from the best performance of any variable-horizon policy, defined by

$$\phi_m^\pi(n) = \frac{C_m^\pi(n)}{\min\{C_m^{\pi(g)}(n) \mid g \in \mathcal{H}_m\}}. \tag{7.11}$$

Since we assumed that $f(n) > 0$, it is obvious that $\phi_m^\pi(n) \geq 1$.

Based on the stationarity of the demand processes under consideration we expect that $\phi_m^\pi(n)$ converges to a limit value as n goes to infinity. We denote this limit value by ϕ_m^π. The on-line performance characteristics of a variable-horizon policy π can be investigated by estimating ϕ_m^π for instances of on-line lot-sizing problems with different data horizon lengths m, demand processes, and cost structures. Based on these characteristics we can study the *robustness* of the different variable-horizon policies. A variable-horizon policy is called robust if it yields a good on-line performance, irrespective of the cost structure, the demand process, and the length of the data horizon.

Unfortunately, computing $\phi_m^\pi(n)$ for instances with sufficiently large values of n, as in the off-line case, will not work because the best performance of any arbitrary variable-horizon policy for an instance of an on-line lot-sizing problem with data horizon length m and problem horizon length n is in general unknown. Therefore, we next derive an upper and a lower bound on $\phi_m^\pi(n)$ that can be used to estimate ϕ_m^π.

An upper bound. Let $\overline{\phi}_m^\pi(n)$ denote the deviation from off-line m-optimality defined by

$$\overline{\phi}_m^\pi(n) = \frac{C_m^\pi(n)}{f_m(n)}. \tag{7.12}$$

The following result provides an upper bound on the on-line performance of a variable-horizon policy and is immediate from the fact that $C_m^\pi(n)$ is greater than or equal to the cost $f_m(n)$ of an m-optimal n-period plan.

Proposition 7.1. *$\phi_m^{\pi(g)}(n) \leq \overline{\phi}_m^{\pi(g)}(n)$ for all $g \in \mathcal{H}_m$.* □

Based on the stationarity of the demand processes under consideration, we again expect that $\overline{\phi}_m^\pi(n)$ converges to a limit value as n goes to infinity. We denote this limit value by $\overline{\phi}_m^\pi$. It is obvious that $\phi_m^\pi \leq \overline{\phi}_m^\pi$. The advantage is that $\overline{\phi}_m^\pi(n)$ can be easily computed using the recursion (2.11).

A lower bound. The following result follows directly from the fact that any variable-horizon policy provides an on-line performance lower bound.

Proposition 7.2. $\phi_m^{\pi(g)}(n) \geq \dfrac{C_m^{\pi(g)}(n)}{C_m^{\pi(g')}(n)}$ *for all $g, g' \in \mathcal{H}_m$.* □

We expect that these lower bounds converge when taking the limit of n to infinity.

In our experiments we typically compare a number of different variable-horizon policies. The above result can then be used as follows. Suppose we have N variable-horizon policies π_i, $i = 1, \ldots, N$. Then for all $i = 1, \ldots, N$ the best lower bound for $\phi_m^{\pi_i}(n)$ is given by

$$\underline{\phi}_m^{\pi_i}(n) = \frac{C_m^{\pi_i}(n)}{\min\{C_m^{\pi_j}(n) \mid j = 1, \ldots, N\}}. \tag{7.13}$$

Note that a value of 1.01 for this lower bound implies that the performance of π_i deviates 1% from the best of these N policies.

Discussion. Suppose we have a benchmark of N variable-horizon policies π_i with $i = 1, \ldots, N$. Then we can investigate the on-line performance characteristics of variable-horizon policy π_i by estimating the on-line performance bounds $\overline{\phi}_m^{\pi_i}$ and $\underline{\phi}_m^{\pi_i}$ for instances of on-line lot-sizing problems with different data horizon lengths, demand processes, and cost structures. Good estimates are obtained for large problem horizons. The smaller the gap between upper and lower bound, the better we can estimate the on-line performance $\phi_m^{\pi_i}$ for a particular instance. Using the upper bound as estimate has the advantage that we can *guarantee* that the on-line performance is not worse than $\overline{\phi}_m^{\pi_i}$. This is nice if the gap between the upper bound and the lower bound is small, but of less use if the gap is large. On the other hand, if one of the N variable-horizon policies under consideration has a near-optimal on-line performance, it would be appropriate to use the lower bound as an estimate for $\phi_m^{\pi_i}$. Preluding on the experiments, we only observed significant gaps for small values of the data horizon. Fortunately, it is especially in those cases that the conditions for good generalization are excellent, and near on-line optimal performance of our policies can be expected. For these reasons, the lower bounds are used as estimates for the on-line performance. For the sake of completeness, we also give the upper bounds.

7.3.3 Explanation of the tables

Unless stated otherwise, the tables in this chapter present performance characteristics in percentages deviation. We add a background gray color to each table entry, whose level of darkness scales with the entry. In this way we can easily recognize patterns in the performance characteristics. The lighter the background gray level of a table entry, the better its performance. Gray levels are in the range [0,1], where 0 and 1 denote black and white, respectively. Percentages deviation are mapped onto background gray levels using the function $s : \mathbb{R}^+ \to [0, 1]$, defined by

$$s(x) = \begin{cases} 1 - \frac{x}{5} & \text{if } 0 \leq x \leq 5 \\ 1 & \text{otherwise.} \end{cases} \tag{7.14}$$

In the range 0-5%, background gray levels change linearly as a function of the percentage deviation from white (0% deviation) to black (5% deviation). All table entries with a percentage deviation larger than five are assigned the background gray level black.

7.4 The potential of variable-horizon policies

Before discussing the performance characteristics of specific variable-horizon policies, we focus on their potential. Let $\alpha_m(n)$ be the deviation of m-optimality over optimality for the n-period problem defined by

$$\alpha_m(n) = \frac{f_m(n)}{f(n)}. \tag{7.15}$$

From Proposition 2.6 it follows that $\alpha_m(n) \geq 1$. Let $\pi = \pi(g)$ with $g \in \mathcal{H}_m$ be a variable-horizon policy. Then one easily verifies that $\alpha_m(n)$, $\overline{\phi}_m^{\pi}(n)$, and $\gamma_m^{\pi}(n)$ satisfy the relation

$$\gamma_m^\pi(n) = \alpha_m(n)\overline{\phi}_m^\pi(n), \tag{7.16}$$

which, using that $\alpha_m(n) \geq 1$ and $\overline{\phi}_m^\pi(n) \geq 1$, yields the following lower bound for the off-line performance of any variable-horizon policy.

Proposition 7.3. $\gamma_m^{\pi(g)}(n) \geq \alpha_m(n)$ *for all* $g \in \mathcal{H}_m$. □

Based on the stationarity of the demand processes under consideration we expect that $\alpha_m(n)$ converges to a limit value as n goes to infinity. We denote this limit value by α_m. The quantity α_m can be seen as an indication of the potential of variable-horizon policies with data horizon length m. For that reason we estimate α_m for the 288 considered combinations of cost structures, data horizon lengths, and demand processes, by applying the recursions (2.4) and (2.11) to the corresponding 4,000-period problem. The corresponding percentages deviation are presented in Table 7.1. The value of α_m decreases strictly with m to become equal to zero for sufficiently large values of m. This behavior can be explained using the following results that can be readily obtained from Proposition 2.7.

Corollary 7.1. *Suppose a finite upper bound M exists on the length of a subplan in an optimal production plan for the n-period problem that is independent of n. Then* $\alpha_m(t) = \alpha_M(t)$ *for all* $m \geq M$ *and* $t \geq 1$. *Furthermore,* $\alpha_m = \alpha_M$ *for all* $m \geq M$. □

Corollary 7.2. *For all* $l \geq m \geq 1$ *and* $t \geq 1$, $\alpha_l(t) \leq \alpha_m(t)$. *Furthermore,* $\alpha_l \leq \alpha_m$ *for all* $l \geq m \geq 1$. □

For instance, for the cost structure with purchasing, applying Theorem 3.6 yields a finite upper bound M with a value of 11.

Comparing the value of α_m for uniformly and Erlang distributed demand, we observe that the value of α_m does not depend on the type of demand distribution but it increases for increasing demand variability. For seasonal demand, the value of α_m increases with the cycle length. Note that the variability increases with the cycle length. Because different cost structures have different cost parameters, conclusions with respect to the characteristics of α_m as a function of the cost structure are less obvious. A commonly used technique to compare different cost structures is to use the corresponding order cycles, based on the average demand level, as a reference. The idea is that, in the constant demand case, different cost structures with equal order cycles can be considered equivalent; see also Section 2.4.3.

For instance, let us determine the minimal length of the data horizon such that the deviation is smaller than 1% for all demand processes. From Table 7.1, we obtain that this length equals 1.5 order cycles for both Wagner-Whitin cost structures,

		Demand process							
		Uniform			Erlang			Seasonal	
Cost structure	*m*	U75	U150	U200	E75	E150	E200	S3	S6
Wagner-Whitin h = 1 S = 200 n^* = 2	2	0.66	3.64	7.07	0.61	[illegible]	[illegible]	[illegible]	[illegible]
	3	0.00	0.22	0.91	0.00	0.08	0.34	0.00	0.71
	4	0.00	0.01	0.09	0.00	0.00	0.02	0.00	0.00
	5	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Wagner-Whitin h = 1 S = 800 n^* = 4	2	31.59	37.90	42.73	31.64	36.67	41.82	36.24	40.92
	3	6.75	10.90	14.00	6.83	10.32	13.02	5.40	14.33
	4	0.81	2.69	4.22	0.65	2.15	3.76	3.17	5.47
	5	0.01	0.52	1.03	0.00	0.35	0.81	0.06	1.40
	6	0.00	0.07	0.25	0.00	0.03	0.15	0.00	0.03
	7	0.00	0.01	0.08	0.00	0.01	0.02	0.00	0.00
	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Overtime h = 1 r = 1 S = 425 C = 200 n^* = 2	2	2.65	7.62	10.71	2.63	[illegible]	[illegible]	[illegible]	10.56
	3	0.00	0.70	1.65	0.01	0.41	0.99	0.07	2.00
	4	0.00	0.03	0.19	0.00	0.02	0.12	0.00	0.00
	5	0.00	0.00	0.01	0.00	0.00	0.01	0.00	0.00
	6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Purchase h = 1 r = 10 S = 100 C = 125 n^* = 1	2	1.17	10.99	[illegible]	1.86	13.81	16.54	17.39	23.88
	3	0.29	[illegible]	9.21	0.51	6.52	[illegible]	0.95	13.44
	4	0.09	2.11	4.92	0.11	3.18	[illegible]	0.30	2.32
	5	0.03	0.93	2.42	0.04	1.51	2.28	0.21	0.31
	6	0.01	0.46	1.04	0.00	0.89	1.16	0.03	0.06
	7	0.00	0.21	0.61	0.00	0.43	0.62	0.01	0.01
	8	0.00	0.08	0.26	0.00	0.16	0.24	0.01	0.00
	9	0.00	0.01	0.05	0.00	0.04	0.06	0.00	0.00
	10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 7.1. *Estimates of α_m in percentages deviation for different combinations of cost structures, data horizon lengths m, and demand processes.*

two order cycles for the cost structure with overtime, and seven order cycles for the cost structure with purchasing. These lengths can be viewed as an indication of the value of demand information for the corresponding on-line lot-sizing problem. Based on the relative proportions between these lengths, the on-line lot-sizing problem with purchasing requires relatively much demand information with respect to the order cycle in order to obtain a certain performance. From this viewpoint it can be considered as more difficult.

7.5 Empirical results

For all 2,880 combinations of variable-horizon policies, data horizon lengths, cost structures, and demand processes, we compute the off-line performance and the on-line performance bounds on the corresponding 4,000 period demand sequences. In view of the size of the setup we need to arrange the results conveniently. This is done by means of aggregation. We aggregate over the eight demand processes included in the setup by presenting the average and the worst-case results for the off-line performance. Furthermore we present average and the worst-case results for the on-line performance bounds.

The remainder of this section is organized as follows. We start with some preliminaries. After that, we discuss the performance characteristics of the variable-horizon policies for the two Wagner-Whitin cost structures, the cost structure with overtime, and the cost structure with purchasing. Finally, we give some conclusions.

7.5.1 Preliminaries

Before we start our discussion of the empirical results we introduce some notions that are central in the discussion of the performance characteristics.

Off-line performance. From the work of Baker [1977], Blackburn & Millen [1980], and Carlson, Beckman & Kropp [1982], we know that the off-line performance of variable-horizon policies like FIX, AVG, and MVG can be arbitrarily close to off-line optimality by taking the data horizon sufficiently large. This does not hold for the variable-horizon policies based on supervised learning, because in general the data horizon cannot be enlarged unlimited without running into the *curse of dimensionality*; see also Section 6.4.4. For these reasons, there exists a length of the data horizon for which FIX, AVG, and MVG begin to exhibit a consistently superior average and worst-case off-line performance to that of the MLP-based and the KNN-based variable-horizon policies. This data horizon length is called the *switch-over point*. The greatest potential for the latter policies is observed for data horizons with a length smaller than the switch-over point. For all cost structures we determine the switch-over point and we discuss the off-line performance characteristics of the different policies for data horizons smaller than this point.

The efficiency of a variable-horizon policy with respect to the available demand data can be investigated by determining the minimal length of the data horizon such that its worst-case off-line performance is smaller than 1%. Lundin & Morton [1975] determined these minimal lengths for the fixed-horizon policy. In their setup they considered a large number of instances of the on-line lot-sizing problem with Wagner-Whitin cost structure; the corresponding order cycles ranged from 1 up to 150. Their main conclusion was that using FIX with a data horizon of five order cycles provides solutions within 1% of an infinite-horizon optimal solution, irrespective of the cost parameter values. We investigate the *data efficiency* of the different variable-horizon policies and determine these minimal lengths for all cost structures under consideration.

On-line performance. A variable-horizon policy is said to be *robust* if, irrespective of the cost structure, the demand process, and the length of the data horizon, it yields a good on-line performance. To investigate the robustness of the different variable-horizon policies we use the on-line performance lower bound. Note that, unlike the off-line performance, it is fair to compare the on-line performance of different variable-horizon policies, not only for different demand processes, but also for different cost structures and data horizon lengths.

In Section 6.4.4 we discussed the phenomenon *curse of dimensionality*, which occurs when generalizing on the basis of zero-one target vectors, and which caused a deterioration in the classification rate of KNN- and MLP-based horizon-selection rules for increasing lengths of the data horizon. Especially the K-nearest-neighbors technique turned out to be very sensitive to this phenomenon, which is related to the smoothness of the target mapping. It is to be expected that the variable-horizon policies based on these horizon-selection rules also suffer from this phenomenon. The amount of deterioration is used as an indication of the complexity of the underlying classification problem and will be investigated.

7.5.2 Two Wagner-Whitin cost structures

Table 7.2 presents the off-line performance characteristics of ten variable-horizon policies for the Wagner-Whitin cost structures with order cycles 2 and 4, respectively. The corresponding on-line performance characteristics are given in Table 7.4 for order cycle 2 and in Table 7.5 for order cycle 4.

Off-line performance. Using Table 7.2, one easily verifies that the switch-over points for the Wagner-Whitin cost structures with order cycles $n^* = 2$ and $n^* = 4$ are obtained for data horizon lengths $4.5n^*$ and $3n^*$, respectively. For data horizons

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	6.81	6.81	7.15	6.90	6.81	6.73	5.88	5.88	5.92	5.72	Average
3	[illegible]	2.22	6.53	5.75	2.86	2.23	1.55	1.53	1.54	1.49	
4	1.49	1.15	5.30	6.64	1.49	1.02	0.55	0.55	0.59	0.52	
5	0.89	0.89	4.29	6.46	0.86	0.55	0.28	0.41	0.32	0.31	
6	0.37	0.78	4.05	6.75	0.37	0.28	0.19	0.37	0.18	0.18	
7	0.23	0.76	4.18	7.34	0.19	0.15	0.14	0.48	0.10	0.10	
8	0.09	0.72	4.03	7.44	0.09	0.08	0.13	0.54	0.05	0.06	
9	0.03	0.75	4.42	8.27	0.04	0.02	0.13	0.68	0.03	0.04	
10	0.01	0.77	4.51	8.52	0.01	0.01	0.14	0.79	0.02	0.03	
2	12.97	12.97	12.97	13.73	12.97	12.91	11.26	11.77	11.58	11.[illegible]	Worst case
3	5.75	3.69	11.12	16.57	4.21	3.18	2.73	2.97	2.65	3.01	
4	2.01	2.35	8.35	12.96	2.03	1.55	0.71	0.77	0.73	0.78	
5	1.76	2.00	7.11	11.64	1.43	1.02	0.41	0.52	0.41	0.45	
6	0.86	1.74	6.47	12.51	0.84	0.65	0.27	0.60	0.27	0.29	
7	0.83	1.73	6.02	12.78	0.50	0.40	0.23	0.88	0.18	0.21	
8	0.27	1.69	5.82	13.06	0.27	0.26	0.25	1.00	0.13	0.15	
9	0.07	1.73	5.07	12.77	0.11	0.07	0.26	1.16	0.09	0.09	
10	0.04	1.73	5.62	13.17	0.04	0.03	0.27	1.46	0.06	0.07	

(a) Order cycle 2

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	37.77	37.77	37.77	37.77	37.78	37.78	37.75	37.21	37.77	37.69	Average
3	11.46	11.31	11.48	11.46	16.65	17.50	11.32	11.40	11.26	11.21	
4	6.80	4.90	7.29	6.39	5.68	5.19	4.25	4.22	4.[illegible]	4.17	
5	7.31	2.67	7.81	6.29	2.38	2.38	1.69	1.74	1.70	1.70	
6	4.89	1.48	7.15	6.68	1.33	1.33	0.88	0.93	0.94	0.88	
7	2.40	1.13	6.08	6.98	0.86	0.91	0.54	0.75	0.65	0.67	
8	1.28	0.92	5.30	6.94	0.60	0.57	0.43	0.68	0.48	0.55	
9	1.21	0.79	5.03	6.79	0.49	0.39	0.35	0.66	0.38	0.41	
10	0.96	0.72	4.[illegible]	6.70	0.34	0.31	0.32	0.63	0.30	0.38	
2	43.08	43.08	43.08	43.08	43.14	43.14	43.08	43.08	43.08	43.08	Worst case
3	16.19	16.19	16.19	16.19	22.34	24.35	16.80	16.79	16.53	16.50	
4	11.04	8.64	12.12	10.27	6.57	8.07	6.31	6.74	6.70	6.64	
5	9.63	6.93	13.36	9.53	5.17	4.82	2.73	3.03	2.73	3.00	
6	5.66	2.72	13.31	10.31	1.99	2.01	1.35	1.50	1.38	1.38	
7	3.41	1.94	12.38	11.29	1.11	1.43	0.73	1.13	0.88	1.03	
8	1.85	1.88	11.14	10.84	0.70	0.62	0.55	1.05	0.61	0.65	
9	1.65	1.83	10.72	11.26	0.58	0.44	0.45	1.09	0.53	0.51	
10	1.31	1.73	10.58	10.89	0.45	0.42	0.53	1.06	0.36	0.46	

(b) Order cycle 4

Table 7.2. *Percentage deviation from off-line optimality for ten different variable-horizon policies, using the Wagner-Whitin cost structures with two different order cycles and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

					0-1 targets		Cost targets	
n^*	m-OPT	FIX	AVG	MVG	MLP	KNN	MLP	KNN
2	$1.5n^*$	$3n^*$	$3n^*$	$3n^*$	$2n^*$	$2n^*$	$2n^*$	$2n^*$
4	$1.5n^*$	$3n^*$	$2n^*$	$2n^*$	$1.75n^*$	∞	$1.75n^*$	$2n^*$

Table 7.3. *Minimal length of the data horizon such that the worst-case deviation from off-line optimality is smaller than 1% for Wagner-Whitin cost structures with order cycles $n^* = 2$ and $n^* = 4$.*

with lengths smaller than these switch-over points, MLP (0-1 targets), MLP (cost targets), and KNN (cost targets) provide an average and worst-case off-line performance that is consistently superior to that of the other variable-horizon policies included in the benchmark. The off-line performance characteristics of KNN (0-1 targets) deteriorate with the length of the data horizon.

From Table 7.2 we determine the minimal length of the data horizon such that the worst-case off-line performance is smaller than 1% for all 20 different combinations of Wagner-Whitin cost structures and variable-horizon policies. These minimal lengths are expressed in multiples of the order cycle in Table 7.3. The entries of the variable-horizon policies that do not yield such a performance are omitted. Furthermore, we include the minimal length of the data horizon such that the deviation of off-line m-optimality over off-line optimality is smaller than 1% worst case. These lengths are denoted by m-OPT and were determined in Section 7.4. Proposition 7.3 implies that for any on-line lot-sizing problem, a variable-horizon policy cannot obtain an off-line performance smaller than 1% with a data horizon smaller than m-OPT.

From Table 7.3 it appears that FIX, AVG and MVG require three order cycles of demand information to consistently yield a worst-case off-line performance that is smaller than 1%. Only two order cycles are needed for MLP (0-1 targets), MLP (cost targets), and KNN (cost targets). This is rather data efficient, because the smallest possible length of the data horizon for which a variable-horizon policy may exist that obtains such a performance (m-OPT) equals 1.5 order cycles.

On-line performance. Using Table 7.4 and Table 7.5, one easily verifies that the average gap between on-line performance upper and lower bounds decreases with m. The smaller this gap, the better we can estimate the average on-line performance. Significant gaps are only found for small data horizons, i.e., for data horizon lengths $m = 2, 3$. Due to excellent conditions for generalization, we expect that our policies are near on-line optimal in case of small data horizons. For these reasons, it makes sense to use the on-line performance lower bound to estimate the on-line performance.

From Table 7.4 and Table 7.5 it is clear that MLP (0-1 targets), MLP (cost tar-

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	1.06	1.06	1.36	1.14	1.06	0.99	0.19	0.19	0.23	0.04	Average
3	2.63	0.80	5.03	4.27	1.44	0.81	0.13	0.11	0.13	0.08	
4	1.01	0.67	4.70	6.14	1.01	0.54	0.07	0.07	0.11	0.04	
5	0.62	0.63	4.02	6.18	0.59	0.29	0.02	0.14	0.05	0.04	
6	0.21	0.62	3.89	6.58	0.21	0.12	0.03	0.21	0.02	0.02	
7	0.15	0.68	4.09	7.26	0.12	0.07	0.06	0.40	0.02	0.03	
8	0.05	0.68	3.98	7.39	0.05	0.03	0.08	0.49	0.01	0.02	
9	0.01	0.73	4.40	8.25	0.02	0.00	0.11	0.66	0.01	0.02	
10	0.00	0.76	4.50	8.57	0.00	0.00	0.13	0.78	0.01	0.02	
2	1.85	1.85	3.83	2.82	1.85	1.56	0.62	0.46	0.69	0.20	Worst case
3	[illegible]	2.35	8.25	7.72	2.15	1.90	0.42	0.33	0.35	0.35	
4	1.41	2.17	7.60	12.30	1.42	0.95	0.20	0.17	0.21	0.10	
5	1.46	1.87	6.82	11.61	1.10	0.63	0.11	0.28	0.11	0.09	
6	0.62	1.67	6.34	12.38	0.60	0.41	0.11	0.48	0.09	0.05	
7	0.70	1.70	5.97	12.72	0.37	0.27	0.18	0.83	0.07	0.08	
8	0.15	1.68	5.80	13.04	0.15	0.14	0.23	0.98	0.02	0.04	
9	0.03	1.73	5.66	12.75	0.05	0.01	0.25	1.15	0.02	0.04	
10	0.01	1.73	5.62	13.17	0.01	0.00	0.27	1.46	0.04	0.04	

(a) Lower bound

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	2.98	2.98	3.29	3.06	2.98	2.91	2.10	2.09	2.13	1.94	Average
3	3.79	1.93	6.22	5.45	2.58	1.94	1.26	1.24	1.26	1.21	
4	1.47	1.13	5.19	6.63	1.47	1.01	0.53	0.54	0.58	0.51	
5	0.89	0.89	4.29	6.46	0.86	0.55	0.28	0.41	0.32	0.31	
6	0.37	0.78	4.05	6.75	0.37	0.28	0.19	0.37	0.18	0.18	
7	0.23	0.76	4.18	7.34	0.19	0.15	0.14	0.48	0.10	0.10	
8	0.09	0.72	4.03	7.44	0.09	0.08	0.13	0.54	0.05	0.06	
9	0.03	0.75	4.42	8.27	0.04	0.02	0.13	0.68	0.03	0.04	
10	0.01	0.77	4.51	8.58	0.01	0.01	0.14	0.79	0.02	0.03	
2	5.51	5.51	5.84	6.21	5.50	5.45	3.90	4.39	4.21	4.11	Worst case
3	5.37	2.96	10.11	9.57	3.38	2.46	1.80	2.04	1.72	2.07	
4	2.01	2.35	8.24	12.86	2.03	1.55	0.68	0.74	0.71	0.69	
5	1.76	2.00	7.11	11.86	1.43	1.02	0.41	0.52	0.41	0.45	
6	0.86	1.74	6.47	12.51	0.84	0.65	0.27	0.60	0.27	0.29	
7	0.83	1.73	6.02	12.78	0.50	0.40	0.23	0.88	0.18	0.21	
8	0.27	1.69	5.82	13.06	0.27	0.26	0.25	1.00	0.13	0.15	
9	0.07	1.73	5.67	12.77	0.11	0.07	0.26	1.16	0.09	0.09	
10	0.04	1.73	5.62	13.17	0.04	0.03	0.27	1.46	0.06	0.07	

(b) Upper bound

Table 7.4. *Bounds on the percentage deviation from on-line optimality for ten different variable-horizon policies, using the Wagner-Whitin cost structure with order cycle 2 and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	0.06	0.06	0.06	0.06	0.07	0.07	0.04	0.01	0.06	0.00	Average
3	0.27	0.13	0.28	0.27			0.14	0.22	0.09	0.06	
4	2.58	0.77	3.04	[illegible]	1.51	1.04	0.15	0.11	0.12	0.07	
5	5.60	1.03	6.05	4.59	0.75	0.75	0.07	0.12	0.08	0.08	
6	4.02	0.64	6.26	5.80	0.50	0.49	0.04	0.10	0.11	0.04	
7	1.86	0.60	5.52	6.41	0.32	0.38	0.01	0.21	0.12	0.14	
8	0.87	0.50	4.87	6.49	0.19	0.16	0.01	0.26	0.06	0.13	
9	0.88	0.46	4.68	6.44	0.16	0.05	0.02	0.33	0.05	0.08	
10	0.69	0.46	4.53	6.42	0.07	0.04	0.05	0.37	0.03	0.11	
2	0.27	0.27	0.27	0.27	0.26	0.26	0.27	0.06	0.27	0.01	Worst case
3	1.28	0.58	1.38	1.28			0.53	0.71	0.29	0.28	
4	4.96	2.00	5.37	3.63	2.14	1.46	0.61	0.31	0.35	0.19	
5	6.94	4.32	10.60	6.96	2.61	2.27	0.26	0.29	0.28	0.27	
6	4.81	2.13	12.66	9.01	1.40	1.43	0.15	0.45	0.34	0.16	
7	3.09	1.63	12.24	10.49	0.81	1.12	0.08	0.39	0.33	0.30	
8	1.57	1.60	10.83	10.51	0.27	0.25	0.07	0.56	0.22	0.26	
9	1.38	1.57	10.43	10.92	0.26	0.11	0.14	0.78	0.20	0.18	
10	1.05	1.48	10.31	10.60	0.13	0.11	0.26	0.80	0.12	0.18	

(a) Lower bound

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	0.23	0.23	0.23	0.23	0.23	0.23	0.21	0.18	0.23	0.17	Average
3	1.12	0.98	1.13	1.12			0.99	1.07	0.94	0.91	
4	3.81	1.97	4.26	3.41	2.73	2.25	1.35	1.31	1.32	1.27	
5	6.75	2.13	7.74	5.92	1.84	1.85	1.16	1.20	1.17	1.17	
6	4.81	1.41	7.07	6.61	1.26	1.26	0.81	0.86	0.87	0.81	
7	2.38	1.12	6.07	6.96	0.84	0.90	0.53	0.73	0.64	0.65	
8	1.28	0.92	5.30	6.94	0.60	0.57	0.43	0.68	0.48	0.55	
9	1.21	0.79	5.03	6.79	0.49	0.39	0.35	0.66	0.38	0.41	
10	0.96	0.72	4.81	6.70	0.34	0.31	0.32	0.63	0.30	0.38	
2	0.55	0.55	0.55	0.55	0.55	0.55	0.55	0.33	0.55	0.33	Worst case
3	2.44	1.93	2.54	2.44	[illegible]	6.75	2.47	2.47	2.23	2.22	
4	6.55	3.00	7.58	5.81	4.19	3.33	2.10	2.42	2.10	2.24	
5	6.52	5.65	11.79	7.20	3.71	3.37	1.68	1.98	1.68	1.95	
6	[illegible]	2.69	13.27	10.00	1.95	1.98	1.08	1.27	1.23	1.11	
7	3.48	1.94	12.58	11.30	1.11	1.43	0.65	1.04	0.85	0.94	
8	1.85	1.88	11.14	[illegible]	0.70	0.62	0.55	1.05	0.61	0.65	
9	1.65	1.83	10.60	11.29	0.58	0.44	0.45	1.09	0.53	0.51	
10	1.31	1.73	10.58	10.89	0.45	0.42	0.53	1.06	0.36	0.46	

(b) Upper bound

Table 7.5. *Bounds on the percentage deviation from on-line optimality for ten different variable-horizon policies, using the Wagner-Whitin cost structure with order cycle 4 and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

gets), and KNN (cost targets) are robust, because their deviation from the on-line performance lower bound of the best variable-horizon policy is less than 0.3% on average and less than 0.7% worst case. Blackburn & Millen [1980] recommended the use of PUT for situations with small data horizons. Although the average deviation of PUT is less than 1%, it is not a robust policy, because its worst-case deviation is 4.32%.

In Table 7.4 and Table 7.5 we observe that the performance of KNN (0-1 targets) deteriorates with the length of the data horizon. To a lesser extend this also holds for MLP (0-1 targets). This deterioration is caused by the curse of dimensionality and the length of the data horizon for which KNN (0-1 targets) starts to deteriorate is approximately two order cycles.

7.5.3 The cost structure with overtime

Table 7.6 presents the off-line performance characteristics of ten variable-horizon policies for the cost structures with overtime. The corresponding on-line performance characteristics are given in Table 7.8.

Off-line performance. The off-line performance characteristics of the variable-horizon policies for the cost structures with overtime are similar to the off-line performance characteristics of the Wagner-Whitin cost structure with order cycle 2 presented in Table 7.2(a). The switch-over point is obtained for a data horizon of length $5n^*$. For data horizons smaller than $5n^*$ periods, MLP (0-1 targets), MLP (cost targets), and KNN (cost targets) dominate all other policies.

The data efficiency of the different variable-horizon policies is presented in Table 7.7. Both FIX and PUT require four order cycles to obtain a worst-case off-line performance smaller than 1%. The other policies that come within 1% need 2.5 order cycles of demand information, which is rather efficient, because a data horizon of at least two order cycles is necessary to obtain such a performance with a variable-horizon policy (m-OPT). The ratios between the minimal lengths in Table 7.7 are similar to those for the Wagner-Whitin cost structure in Table 7.3.

On-line performance. The on-line performance characteristics are similar to the on-line performance characteristics of the Wagner-Whitin cost structure with order cycle 2 presented in Table 7.4. Again we use the on-line performance lower bound as an estimate for the on-line performance. From Table 7.8 it is clear that only MLP (0-1 targets), MLP (cost targets), and KNN (cost targets) are robust, because their deviation from the on-line performance lower bound of the best variable-horizon policy is less than 0.2% on average and less than 0.4% worst case.

In Table 7.8 the effect of the curse of dimensionality on the performance of the variable-horizon policies based on supervised learning can be clearly observed.

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	8.38	8.43	8.41	8.57	8.37	[illegible]	[illegible]	[illegible]	8.38	8.29	Average
3	5.99	2.84	6.07	4.62	2.26	2.62	1.94	2.04	1.99	2.00	
4	3.05	1.24	5.47	6.21	1.11	1.20	0.79	0.85	0.83	0.84	
5	1.67	0.88	4.14	6.80	0.73	0.78	0.51	0.63	0.51	0.53	
6	0.93	0.69	3.78	7.30	0.40	0.55	0.35	0.50	0.34	0.34	
7	0.62	0.58	3.57	7.40	0.31	0.31	0.28	0.47	0.22	0.24	
8	0.39	0.53	3.41	7.64	0.19	0.25	0.27	0.52	0.16	0.19	
9	0.21	0.48	3.69	8.00	0.14	0.16	0.25	0.54	0.13	0.16	
10	0.14	0.49	4.22	8.58	0.07	0.08	0.26	0.65	0.11	0.11	
2	17.52	12.52	12.52	12.57	12.45	12.45	12.75	12.76	12.65	12.72	Worst case
3	7.99	6.42	11.30	7.41	3.81	4.50	3.49	3.48	3.56	[illegible]	
4	4.31	1.89	8.81	9.8[illegible]	1.70	1.67	1.39	1.47	1.46	1.43	
5	2.22	1.59	6.21	9.79	0.90	0.96	0.78	0.95	0.70	0.80	
6	1.08	1.23	6.10	10.69	0.56	0.74	0.47	0.84	0.43	0.48	
7	1.00	0.97	5.87	[illegible]	0.47	0.40	0.39	0.79	0.28	0.30	
8	0.50	0.96	5.52	10.97	0.30	0.34	0.47	0.99	0.24	0.26	
9	0.32	0.80	5.40	11.07	0.19	0.20	0.47	1.11	0.16	0.17	
10	0.18	0.82	5.41	11.17	0.09	0.09	0.43	1.25	0.13	0.14	

Table 7.6. *Percentage deviation from off-line optimality for ten different variable-horizon policies, using the cost structure with overtime and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

						0-1 targets		Cost targets	
n^*	m-OPT	FIX	PUT	AVG	MVG	MLP	KNN	MLP	KNN
2	$2n^*$	$4n^*$	$4n^*$	$2.5n^*$	$2.5n^*$	$2.5n^*$	∞	$2.5n^*$	$2.5n^*$

Table 7.7. *Minimal length of the data horizon such that the worst-case deviation from off-line optimality is smaller than 1% for the cost structure with overtime.*

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	0.12	0.15	0.15	0.11	0.12	0.15	0.09	0.07	0.12	0.03	Average
3	3.41	0.90	4.07	2.84	0.34	0.69	0.01	0.12	0.06	0.08	
4	2.25	0.46	4.00	5.40	0.33	0.42	0.01	0.07	0.04	0.06	
5	1.19	0.41	3.65	6.36	0.26	0.31	0.03	0.16	0.04	0.06	
6	0.62	0.38	3.46	6.97	0.09	0.25	0.05	0.19	0.03	0.03	
7	0.41	0.37	3.35	7.27	0.10	0.11	0.08	0.26	0.01	0.04	
8	0.25	0.38	3.26	7.48	0.05	0.10	0.12	0.38	0.02	0.05	
9	0.10	0.37	3.57	7.91	0.02	0.05	0.14	0.43	0.02	0.05	
10	0.07	0.42	4.14	9.51	0.00	0.01	0.19	0.58	0.04	0.04	
2	0.28	0.43	0.31	0.28	0.31	0.38	0.24	0.21	0.33	0.18	Worst case
3	4.88	3.96	10.36	6.41	0.81	1.01	0.05	0.21	0.23	0.14	
4	3.51	0.92	7.97	8.98	0.54	0.62	0.03	0.20	0.18	0.13	
5	1.90	1.01	5.66	9.21	0.42	0.36	0.11	0.31	0.13	0.13	
6	0.88	0.88	5.73	10.58	0.23	0.42	0.18	0.43	0.08	0.11	
7	0.77	0.75	5.64	10.24	0.25	0.20	0.17	0.58	0.05	0.07	
8	0.39	0.74	5.29	10.78	0.13	0.19	0.33	0.83	0.03	0.10	
9	0.22	0.66	5.26	10.97	0.06	0.07	0.38	1.02	0.05	0.07	
10	0.12	0.75	5.23	11.12	0.00	0.02	0.38	1.20	0.06	0.09	

(a) Lower bound

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	0.95	0.99	0.98	0.94	0.95	0.98	0.92	0.90	0.95	0.87	Average
3	4.63	2.09	5.31	4.06	1.53	1.88	1.20	1.30	1.25	1.26	
4	3.01	1.20	5.17	6.18	1.07	1.16	0.74	0.81	0.78	0.79	
5	1.67	0.88	4.14	6.86	0.73	0.78	0.50	0.63	0.51	0.53	
6	0.93	0.69	3.78	7.30	0.40	0.55	0.35	0.50	0.34	0.34	
7	0.62	0.58	3.57	7.49	0.31	0.31	0.28	0.47	0.22	0.24	
8	0.39	0.53	3.41	7.64	0.19	0.25	0.27	0.52	0.16	0.19	
9	0.21	0.48	3.69	8.04	0.14	0.16	0.25	0.54	0.13	0.16	
10	0.14	0.49	4.22	9.90	0.07	0.08	0.26	0.65	0.11	0.11	
2	1.82	2.13	2.01	1.77	1.83	2.07	1.85	1.82	1.94	1.79	Worst case
3	5.86	4.33	11.32	7.33	2.11	2.96	1.93	2.09	2.16	2.00	
4	4.31	1.77	8.11	9.83	1.58	1.55	1.27	1.35	1.34	1.31	
5	2.22	1.59	6.33	9.78	0.90	0.95	0.78	0.94	0.70	0.79	
6	1.08	1.23	6.01	10.89	0.56	0.74	0.47	0.84	0.43	0.48	
7	1.00	0.97	5.87	10.99	0.47	0.40	0.39	0.79	0.28	0.30	
8	0.50	0.96	5.57	10.93	0.30	0.34	0.47	0.99	0.24	0.26	
9	0.32	0.80	5.40	11.07	0.19	0.20	0.47	1.11	0.16	0.17	
10	0.18	0.82	5.31	11.17	0.09	0.09	0.43	1.25	0.13	0.14	

(b) Upper bound

Table 7.8. *Bounds on the percentage deviation from on-line optimality for ten different variable-horizon policies, using the cost structure with overtime and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	16.34	29.33	20.23	29.04	16.26	16.22	14.34	14.62	14.10	14.01	Average
3	8.26	29.52	12.22	28.60	8.22	8.07	6.38	6.63	6.58	6.17	
4	4.90	28.18	10.77	29.01	4.90	4.80	3.11	3.50	3.13	2.88	
5	2.49	27.81	10.11	29.31	2.49	2.42	1.82	2.48	2.17	1.46	
6	1.17	27.40	9.57	29.45	1.16	1.16	1.26	2.53	1.36	0.85	
7	0.69	27.17	9.29	29.44	0.71	0.68	1.17	2.94	1.10	0.57	
8	0.43	26.98	9.09	29.57	0.44	0.43	1.15	3.39	0.82	0.43	
9	0.18	26.81	8.82	29.68	0.19	0.19	1.16	3.94	0.71	0.31	
10	0.08	26.72	8.63	29.75	0.09	0.09	1.24	4.40	0.59	0.29	
2	26.11	44.82	33.60	44.19	26.15	26.19	24.48	24.45	24.64	24.27	Worst case
3	17.71	43.47	24.42	41.83	17.63	17.59	13.64	13.94	13.39	13.31	
4	8.93	42.53	20.29	43.38	8.94	8.93	6.11	6.79	6.47	5.99	
5	5.14	42.29	18.03	43.37	5.16	5.07	3.80	4.96	4.25	3.39	
6	3.12	41.66	17.08	43.48	3.23	3.21	2.49	4.66	2.62	1.99	
7	1.58	40.99	16.79	43.43	1.62	1.61	2.09	5.39	1.79	1.47	
8	0.98	40.72	16.64	44.27	1.03	1.05	2.40	6.29	1.49	1.01	
9	0.52	40.32	15.94	44.51	0.53	0.51	2.66	7.50	1.54	0.70	
10	0.33	40.19	14.98	44.26	0.34	0.33	3.03	8.45	1.32	0.77	

Table 7.9. *Percentage deviation from off-line optimality for ten different variable-horizon policies, using the cost structure with purchasing and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

The strongest deterioration as a function of the data horizon length occurs for KNN (0-1 targets) and to a lesser extend for MLP (0-1 targets). For the policies KNN (cost targets) and MLP (cost targets) we observe some deterioration, but it is an order of magnitude smaller than for the policies based on supervised learning with zero-one target vectors.

7.5.4 The cost structure with purchasing

Table 7.9 presents the off-line performance characteristics of ten variable-horizon policies for the cost structures with purchasing. The corresponding on-line performance characteristics are given in Table 7.10.

Examination of these tables leads to two remarkable observations. First, MLP (cost targets) is consistently outperformed by KNN (cost targets), indicating that the model specific conditions for good generalization are not satisfied. Apparently, the network topologies that were considered during the construction of the horizon-selection rule had too little functional complexity, i.e., too few hidden units; see Chapter 6. This is supported by the second observation, which is the relatively strong deterioration of KNN (0-1 targets) and MLP (0-1 targets), which indicates that the target mapping underlying the learning examples with zero-one target vectors

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	1.97	13.15	5.28	12.90	1.99	1.96	0.45	0.52	0.08	0.01	Average
3	1.92	20.90	5.56	20.94	1.89	1.75	0.21	0.44	0.39	0.01	
4	1.96	24.13	7.59	25.24	1.97	1.86	0.23	0.60	0.24	0.01	
5	1.02	25.89	8.49	27.36	1.02	0.95	0.36	1.01	0.71	0.01	
6	0.32	26.29	8.61	28.33	0.32	0.31	0.42	1.67	0.52	0.01	
7	0.18	26.40	8.73	28.75	0.19	0.17	0.66	2.41	0.59	0.06	
8	0.10	26.54	8.72	29.12	0.11	0.11	0.82	3.05	0.50	0.10	
9	0.04	26.67	8.66	29.48	0.05	0.05	1.02	3.79	0.57	0.17	
10	0.01	26.62	8.54	29.67	0.01	0.01	1.17	4.32	0.51	0.22	
2	7.32	30.70	13.30	19.90	7.12	7.35	2.37	1.63	0.30	0.03	Worst case
3	3.89	37.76	9.83	39.01	3.84	3.81	0.60	0.98	1.29	0.08	
4	5.75	37.37	16.79	39.66	5.83	5.87	0.76	0.99	0.45	0.04	
5	3.16	38.30	17.16	40.54	3.20	3.22	0.53	1.75	2.25	0.05	
6	1.11	38.89	16.78	41.01	1.20	1.19	0.57	2.61	1.05	0.07	
7	0.68	39.27	16.55	41.64	0.70	0.69	1.21	4.10	1.41	0.32	
8	0.51	39.58	16.46	42.89	0.52	0.48	1.43	5.16	1.33	0.34	
9	0.34	40.02	15.73	44.20	0.34	0.29	2.44	7.22	1.36	0.48	
10	0.07	40.09	14.90	44.18	0.08	0.08	2.95	8.37	1.20	0.70	

(a) Lower bound

m	Myopic				Forecasting		0-1 targets		Cost targets		
	FIX	PUT	PUP	EOQ	AVG	MVG	MLP	KNN	MLP	KNN	
2	2.82	14.11	6.16	11.86	2.83	2.80	1.29	1.35	0.91	0.84	Average
3	2.65	21.99	6.34	21.88	2.62	2.48	0.92	1.16	1.11	0.72	
4	2.67	25.23	8.35	26.16	2.68	2.58	0.93	1.30	0.94	0.70	
5	1.51	26.52	9.02	28.01	1.51	1.44	0.84	1.50	1.19	0.49	
6	0.70	26.79	9.05	28.84	0.70	0.69	0.80	2.06	0.90	0.39	
7	0.45	26.85	9.02	29.12	0.46	0.44	0.93	2.69	0.86	0.33	
8	0.33	26.81	8.99	29.44	0.34	0.34	1.05	3.30	0.73	0.33	
9	0.16	26.78	8.79	29.63	0.17	0.16	1.14	3.91	0.69	0.29	
10	0.08	26.72	8.68	29.78	0.09	0.09	1.24	4.40	0.59	0.29	
2	7.42	21.98	13.90	31.45	7.43	7.47	4.40	3.33	1.98	2.00	Worst case
3	4.95	38.41	11.07	39.65	4.95	4.61	1.76	2.23	2.32	1.83	
4	6.45	38.87	17.86	40.53	6.53	6.57	1.83	2.14	1.55	1.40	
5	3.64	38.95	17.66	41.19	3.68	3.70	1.35	2.62	2.68	0.95	
6	2.07	40.20	17.09	41.71	2.16	2.15	1.44	3.58	1.57	0.95	
7	0.96	40.14	16.72	42.56	0.98	0.98	1.48	4.79	1.60	0.84	
8	0.72	40.36	16.64	43.90	0.76	0.78	2.14	5.07	1.49	0.77	
9	0.52	40.26	15.99	44.44	0.53	0.47	2.61	7.15	1.54	0.65	
10	0.33	40.19	14.98	44.28	0.34	0.33	3.03	8.45	1.32	0.77	

(b) Upper bound

Table 7.10. *Bounds on the percentage deviation from on-line optimality for ten different variable-horizon policies, using the cost structure with purchasing and different data horizons m. The entries represent average and worst-case percentages of eight different demand processes.*

n^*	m-OPT	FIX	AVG	MVG	KNN
1	$7n^*$	$8n^*$	$8n^*$	$8n^*$	$8n^*$

Table 7.11. *Minimal length of the data horizon such that the worst-case deviation from off-line optimality is smaller than 1% for the cost structure with purchasing.*

has a relatively high functional complexity; see also Section 6.1.

We predicted that the functional complexity of the target mappings of the two-source models would be higher than that of the single-source models. This was taken into account when predetermining the set of network topologies by taking more hidden units in these cases. However, as we may conclude know, still more hidden units are required. As a result of that we end up with suboptimal MLP-based variable-horizon policies. The off-line performance characteristics of KNN (cost targets) are an indication of the performance that can be obtained by adding more hidden units. For these reasons we only address KNN (cost targets) in our discussion of the performance characteristics.

Off-line performance. The switch-over point is obtained for data horizon length $8n^*$. For data horizons smaller than $8n^*$ periods, KNN (cost targets) dominates the other policies. In contrast with the other cost structures, FIX can be hardly improved by using a variable-horizon policy with forecasting.

The data efficiency of the different variable-horizon policies is presented in Table 7.11. The policies that obtain a worst-case off-line performance smaller than 1% all need eight order cycles of demand information. This is rather efficient, because any variable-horizon policy requires a data horizon of at least seven order cycles to obtain such a performance with a variable-horizon policy (m-OPT).

On-line performance. We use the on-line performance lower bound as an estimate for the on-line performance. From Table 7.10 it is clear that KNN (cost targets) is robust, because the deviation from the on-line performance lower bound of the best variable-horizon policy is less than 0.3% on average and less than 0.7% worst case. The other policies all have worst-case deviations up to 8%.

7.6 Conclusion

This chapter investigated the lot-sizing performance of the MLP-based variable-horizon policies by means of an extensive empirical study using a benchmark of different variable-horizon policies.

In Section 7.5.1 we postulated the existence of a switch-over point with respect to the performance of the variable-horizon policies based on supervised learning. For all cost structures we determined this switch-over point. We conclude that for data horizons with a length smaller than the switch-over point the MLP-based variable-horizon policies have superior performance characteristics. For data horizons with a length larger than this switch-over point, these policies are outperformed by FIX, AVG. and MVG. It appeared that the switch-over point is mainly determined by the cost structure, and, we can conclude from the experimental results that it is larger than the length of the data horizon needed by FIX to obtain a worst-case off-line performance smaller than 1%.

For data horizons with a length smaller than the switch-over point, we investigated both the data efficiency and the robustness of the different variable-horizon policies. We conclude that, irrespective of cost structure and demand process, the MLP-based variable-horizon policies yield excellent performance which is obtained using only little demand information.

We end this chapter with some conclusions concerning the applicability of MLP-based variable-horizon policies for on-line lot-sizing problems. To that end we distinguish between two typical cases, i.e., small data horizon and large data horizon. Note that the notions 'small' and 'large' are relative and depend on the cost structure. Such a distinction was also made in Chapter 1 and Chapter 6.

Small data horizons. In case the data horizon is small, demand uncertainty is large, and it is important to determine a good ending condition, i.e., a suitable optimization horizon. We investigated variable-horizon policies with three different types of horizon-selection rules. The first type of rule employed a simple myopic heuristic (FIX, PUT, PUP, EOQ). The second type of rule used a simple forecasting method as proposed by Carlson, Beckman & Kropp [1982] to extend the data horizon in combination with a forward algorithm (AVG, MVG). The third type of rule employed a horizon-selection rule based on supervised learning (MLP, KNN). From the results presented in this chapter we conclude that the MLP-based variable-horizon policies dominate all other variable-horizon policies through good performance characteristics, great data efficiency, and robustness.

Large data horizons. In case the data horizon is large, the demand uncertainty is small, and there is hardly any benefit from determining a suitable optimization horizon by using either forecasting, multi-layered perceptrons, or the K-nearest-neighbors technique. The best policy that can be used in this case is the fixed-horizon policy FIX.

8

Conclusion

In this thesis we investigated the potential of supervised learning with multi-layered perceptrons for on-line lot-sizing problems. The starting point of our study is a general single-item on-line lot-sizing problem. We propose a class of hierarchical solution approaches that we call variable-horizon policies. In such policies, lot sizes are determined by repeatedly optimizing over a variable optimization horizon that is chosen by some horizon-selection rule that takes the available demand information into account.

We formulated the problem of finding an optimal horizon-selection rule as a classification problem, which we analyzed in a statistical framework. We considered two objectives, i.e., maximization of expected classification rate and minimization of expected excess cost. For these objectives we can give explicit expressions for the optimal horizon-selection rules. Supervised learning with multi-layered perceptrons is used to estimate the unknown parameters of these expressions. Next we derived so-called MLP-based horizon-selection rules from the developed multi-layered perceptrons. To facilitate the off-line computation of learning examples, we developed forward algorithms.

We have analyzed the conditions for good generalization and their effect on the generalization capabilities of the MLP-based horizon-selection rules. Numerical results show that these conditions deteriorate if the number of known future demands increases. By means of an extensive empirical study, we compared the performance characteristics of the variable-horizon policies constituted by the MLP-based horizon-selection rules with those of a benchmark of variable-horizon policies. This study showed that, in situations with large demand uncertainty, the MLP-based variable-horizon policies dominate all other variable-horizon policies with respect to robustness, performance, and data efficiency. In situations with small demand uncertainty, using multi-layered perceptrons is not beneficial.

The remainder of this chapter is organized as follows. First, we formulate criteria that justify a supervised learning approach, Next, we comment on the models and techniques presented in this thesis. Finally, we give some suggestions for future research.

Criteria for supervised learning. We conclude that an approach based on supervised learning may contribute significantly to the performance of on-line lot-sizing systems if the following criteria are satisfied.

1. There is no efficient solution approach for the particular problem at hand. This may be, for instance, because modeling is (too) difficult or the problem is characterized by incomplete data.
2. There is sufficient relevant input data available to construct a sufficiently large set of learning examples. In most applications this will be plain historical data. In case there is only limited data available or historical data is no longer up to date or relevant, one option is to model relevant cases and construct learning examples for them.
3. It is possible to provide target data for input data in an efficient way. This can be by means of an algorithm or by means of a human expert.

Success, however, is not assured, since good generalization is only possible if the conditions for generalization are satisfied; see Section 4.5. For instance, the problem of learning the inverse of a one-way function used in user authentication [Tilborg, 1988] does satisfy the abovementioned criteria, but does not satisfy the conditions for good generalization. Furthermore, we conclude that choosing an appropriate *problem representation* is of the utmost importance; for instance, we refer to the differences in generalization capabilities between the approaches based on different target vectors observed in Chapter 6.

We stated before that the nature of an on-line lot-sizing problem is characterized by two components, i.e., a combinatorial component involving the timing and sizing of the production quantities, and an uncertainty component representing the incomplete demand information; see also Section 1.3. We developed a hierarchical approach which exploits the strong points of multi-layered perceptrons for the uncertainty component and which builds upon the numerous results and techniques from off-line lot-sizing for the combinatorial component. In this way we combine the best of both fields. We conclude that such a two-stage approach may contribute significantly to the performance of on-line lot-sizing systems if the learning problem of the first stage satisfies the above criteria and the combinatorial problem of the second stage can be computed efficiently.

General comments. In this thesis we focussed on on-line lot-sizing problems. Nevertheless, the ideas and techniques presented in this thesis are more general. This can be argued as follows. The backbone of our approach is the off-line computation of planning and forecast horizons by means of a forward algorithm with a stop criterion. In fact the techniques presented in this thesis can be directly applied to on-line versions of any problem that fits within the regeneration set framework [Lundin, 1973; Lundin & Morton, 1975]. Examples of such problems are cash balancing [Mensching, Garstka & Morton, 1978], capacity expansion [Rajagopalan, 1994; Udayabhanu & Morton, 1988], equipment replacement [Bylka, Sethi & Sorger, 1992; Sethi & Chand, 1979], and facility location [Bastian & Volkmer, 1992; Daskin, Hopp & Medina, 1992].

An important step in our approach is the observation that the problem of selecting an appropriate optimization horizon can be viewed as a classification problem. This viewpoint enabled a practicable definition of on-line optimality. It is important to note that other classification approaches than multi-layered perceptrons can be applied. Actually, this was illustrated by the application of the K-nearest-neighbors technique in Chapter 6 and Chapter 7.

An interesting subject, not addressed in this thesis, is the effect of the number of learning examples on the performance of the MLP-based variable-horizon policies. Van Kraaij [1991] did some preliminary experiments for an on-line lot-sizing problem with Wagner-Whitin cost structure. In these experiments, only 50 learning examples were sufficient to outperform the heuristic of Silver & Meal [1973]. More experiments are needed to reach convincing conclusions. We can conclude from our experiments that the number of learning examples required for good generalization increases with the length of the data horizon; see Chapter 6.

In this thesis we considered so-called *uncapacitated* lot-sizing models in the sense that the total production capacity in each period, which may stem from different sources, is infinite. In practice, however, the total amount of production in a period may be bounded. Additionally, there may be finite bounds on the amount of product kept in inventory from one period to the next. The latter extension is rather straightforward, since planning horizon results exist [Sandbothe & Thompson, 1993]. The former extension, however, is more difficult since no planning horizon results exist for capacitated problems. One option is to relax the capacity constraint and to introduce an uncapacitated model with penalty cost for exceeding capacity as we considered in this thesis. An additional problem with capacitated problems is that some of the possible optimization horizons may be infeasible.

This problem can be overcome by taking the feasible optimization horizon with the highest estimated posterior probability, for instance.

Suggestions for future research. An interesting subject for future research is to investigate if the ideas and techniques presented in this thesis can be applied to on-line versions of decision problems that do not fit in the regeneration set framework. A possible starting point is the infinite-horizon dynamic programming framework proposed by Morton [1979], who introducing the concept of T-regeneration sets, which applies to a reasonably general form of the dynamic programming problem. Another option is to proceed from the planning horizon framework introduced by Federgruen & Tzur [1995] and Federgruen & Tzur [1996], which handles the class of problems that can be formulated as shortest path problems in acyclic graphs.

Promising is the study of techniques for the incorporation of problem-specific knowledge into neural networks. Such knowledge can take a variety of forms, but usually consists of some general information about the form which the target mapping should take or some constraint which it should satisfy. This kind of knowledge is referred to as *prior knowledge*. Since any information that is build directly into the network reduces the complexity of the learning problem involved, this may lead to substantial improvements in data requirements, learning efficiency, and generalization. For examples we refer to the work of Barnard & Botha [1993], Joerding & Meador [1991], and Low & Webb [1990]; see also the textbooks by Bishop [1995] and Honavar [1994]. We mention two possibilities for incorporating prior knowledge when designing MLP-based horizon-selection rules. A first possibility occurs in the case that the demand process is seasonal with a cycle length larger than the data horizon. Then we can exploit this foreknowledge by including an extra demand lag such that the number of inputs equals the cycle length. A second possibility is the incorporation of prior knowledge about one or more of the relevant decision boundaries of the underlying classification problem. Such decision boundaries can for instance be derived for the on-line lot-sizing problems with overtime and purchasing by rewriting Theorem 3.3 and Theorem 3.5, respectively. These decision boundaries can be directly hardwired into the network or can be given as extra inputs to the network. Numerical experiments are needed to examine the impact of incorporating such knowledge on the generalization capabilities of the MLP-based horizon selection rules and the on-line lot-sizing performance of the corresponding variable-horizon policies.

Bibliography

Aarts, E.H.L., and J. Korst [1989], *Simulated Annealing and Boltzmann Machines*, John Wiley & Sons, New York.

Aarts, E.H.L., H.P. Stehouwer, J. Wessels, and P.J. Zwietering [1995], Neural networks for combinatorial optimization, in: J. Doležal and J. Fidler (eds.), *Optimization-based Computer-aided Modelling and Design, Proceedings of the 3-rd IFIP WG-7.6 Working Conference*, UTIA, Prague, 25–40.

Aggarwal, A., and J.K. Park [1993], Improved algorithms for economic lot size problems, *Operations Research* **41**, 549–571.

Axsäter, S. [1982], Worst case performance for lot sizing heuristics, *European Journal of Operational Research* **9**, 339–343.

Axsäter, S. [1985], Performance bounds for lot sizing heuristics, *Management Science* **31**, 634–640.

Bahl, H.C., L.P. Ritzman, and J.N.D. Gupta [1987], Determining lot sizes and resource requirements: a review, *Operations Research* **35**, 329–345.

Baker, K.R. [1977], An experimental study of the effectiveness of rolling schedules, *Decision Science* **8**, 19–27.

Baker, K.R. [1989], Lot-sizing procedures and a standard data set: a reconciliation of the literature, *Journal of Manufacturing and Operations Management* **2**, 199–221.

Baker, K.R., P.S. Dixon, M.J. Magazine, and E.A. Silver [1978], Computational complexity of the capacitated lot size problem, *Management Science* **24**, 1710–1720.

Barnard, E. [1992], Optimization for training neural nets, *IEEE Transactions on Neural Networks* **3**, 232–240.

Barnard, E., and E.C. Botha [1993], Backpropagation uses prior information efficiently, *IEEE Transactions on Neural Networks* **4**, 794–802.

Barnard, E., and L.F.A. Wessels [1992], Extrapolation and interpolation in neural network classifiers, *IEEE Control Systems Magazine* **12**, 50–53.

Barron, A.R. [1991], *Universal approximation bounds for superpositions of sigmoidal functions*, Technical Report 58, Department of Statistics, University of Illinois.

Bastian, M., and M. Volkmer [1992], A perfect forward procedure for a single facility dynamic location / relocation problem, *Operations Research Letters* **12**, 11–16.

BAYES, REV., T. [1763], An essay toward solving a problem in the doctrine of chances, *Philosophical Transactions of the Royal Society of London* **53**, 370–418.

BEAN, J.C., AND R.L. SMITH [1984], Conditions for the existence of planning horizons, *Mathematics of Operations Research* **9**, 391–401.

BEAN, J.C., AND R.L. SMITH [1993], Conditions for the discovery of solution horizons, *Mathematical Programming* **59**, 215–229.

BEAN, J.C., R.L. SMITH, AND C.A. YANO [1987], Forecast horizons for the discounted dynamic lot-size problem allowing speculative motive, *Naval Research Logistics* **34**, 761–774.

BELLMANN, R.E. [1961], *Adaptive Control Processės: a Guided Tour*, Princeton University press, Princeton, New Jersey.

BENSOUSSAN, A., M. CROUHY, AND J.-M. PROTH [1983], *Mathematical Theory of Production Planning*, Advanced Series in Management, North-Holland.

BENSOUSSAN, A., AND J.-M. PROTH [1991], A planning horizon algorithm for deterministic inventory management with piecewise linear concave cost, *Naval Research Logistics* **38**, 729–742.

BERRY, W.L. [1972], Lot sizing procedures for requirements planning systems: a framework for analysis, *Production and Inventory Management* **13**, 19–34.

BISHOP, C.M. [1995], *Neural Networks for Pattern Recognition*, Clarendon Press, Oxford.

BITRAN, G.R., T.L. MAGNANTI, AND H.H. YANASSE [1984], Approximation methods for the uncapacitated lot size problem, *Management Science* **30**, 1121–1140.

BITRAN, G.R., AND H.H. YANASSE [1982], Computational complexity of the capacitated lot size problem, *Management science* **28**, 1174–1186.

BLACKBURN, J.D., AND R.A. MILLEN [1980], Heuristic lot-sizing performance in a rolling-schedule environment, *Decision Sciences* **11**, 691–701.

BLACKBURN, J.D., AND R.A. MILLEN [1985], A methodology for predicting single-stage lot-sizing performance: analysis and experience, *Journal of Operations Management* **5**, 433–448.

BOTTOU, L. [1991], Stochastic gradient learning in neural networks, *Proceedings of the Fourth International Conference on Neural Networks and their Applications*, Nîmes, France, 687–696.

BRADLEY, E.L. [1985], Overlapping coefficient, *Encyclopedia of Statistical Sciences* **6**, John Wiley & Sons, New York.

BRIDLE, J.S. [1990], Probabilistic interpretation of feedforward classification network outputs, in: F. Fogelman-Souliè and J. Hérault (eds.), *Neurocomputing*, Springer-Verlag, Berlin, 223–236.

BYLKA, S., S. SETHI, AND G. SORGER [1992], Minimal forecast horizons in equipment replacement models with multiple technologies and general switching costs, *Naval Research Logistics* **39**, 487–507.

CARLSON, R.C., S.L. BECKMAN, AND D.H. KROPP [1982], The effectiveness of extending the horizon in rolling production scheduling, *Decision Sciences* **13**, 129–146.

CHAND, S. [1979], *Perfect Planning Horizon Procedures*, Ph.D. thesis, Graduate School of Industrial Administration, Carnegie-Mellon University.

CHAND, S. [1982], A note on dynamic lot sizing in a rolling-horizon environment, *Decision Sciences* **13**, 113–119.

CHAND, S., AND T.E. MORTON [1986], Minimal forecast horizon procedures for dynamic lot size models, *Naval Research Logistics Quarterly* **33**, 111–122.

CHEN, H.-D., D.W. HEARN, AND C.-Y. LEE [1994], A dynamic programming algorithm for dynamic lot size models with piecewise linear costs, *Journal of Global Optimization* **4**, 397–413.

CHEN, H.-D., D.W. HEARN, AND C.-Y. LEE [1995], Minimizing the error bound for the dynamic lot size model, *Operations Research Letters* **17**, 57–68.

CHUNG, C.-S., AND C.-H.M. LIN [1988], An $O(T^2)$ algorithm for the NI/G/NI/ND capacitated lot size problem, *Management Science* **34**, 420–426.

CORSTEN, H., AND C. MAY [1996], Artificial neural networks for supporting production planning and control, *Technovation* **16**, 67–76.

CYBENKO, G. [1989], Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals, and Systems* **2**, 304–314.

DAGLI, C.H. [1994], *Artificial Neural Networks for Intelligent Manufacturing*, Chapman & Hall, London.

DASKIN, M.S., W.J. HOPP, AND B. MEDINA [1992], Forecast horizons and dynamic facility location planning, *Annals of Operations Research* **40**, 125–151.

DELGADO, M., J. KACPRZYK, J.-L. VERDEGAY, AND M.A. VILA (eds.) [1994], *Fuzzy Optimization: Recent Advances*, Physica-Verlag, Heidelberg.

DELLAERT, N.P., AND M.T. MELO [1995], Heuristic procedures for a stochastic lot-sizing problem in make-to-order manufacturing, *Annals of Operations Research* **59**, 227–258.

DENOEUX, T., AND R. LENGELLÉ [1993], Initializing back-propagation networks with prototypes, *Neural Networks* **6**, 351–363.

DIXON, P.S. [1980], *Single-item lot-sizing with limited regular and overtime capacity*, Working paper, Department of Finance and Management Science, Saint Mary's University, Halifax, Canada.

DIXON, P.S., M.D. ELDER, G.K. RAND, AND E.A. SILVER [1983], A heuristic algorithm for determining lot sizes of an item subject to regular and overtime production capacities, *Journal of Operations Management* **3**, 121–130.

DUDA, R.O., AND P.E. HART [1973], *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York.

ELLACOTT, S.W. [1994], Aspects of the numerical analysis of neural networks, *Acta Numerica*, 145–203.

EPPEN, G.D., F.J. GOULD, AND B.P. PASHIGIAN [1969], Extensions of the planning horizon theorem in the the dynamic lot size model, *Management Science* **15**, 268–277.

EVANS, J.R. [1985], An efficient implementation of the Wagner-Whitin algorithm for dynamic lot-sizing, *Journal of Operations Management* **5**, 229–235.

FAHLMAN, S.E., AND C. LEBIERE [1990], *The cascade-correlation learning architecture*, Technical Report CMU-CS-90-100, Carnegie Mellon University, Pittsburg, PA.

FEDERGRUEN, A., AND M. TZUR [1991], A simple forward algorithm to solve general dynamic lot sizing models with n periods in $O(n \log n)$ or $O(n)$ time, *Management Science* **37**, 909–925.

FEDERGRUEN, A., AND M. TZUR [1994], Minimal forecast horizons and a new planning procedure for the general lot sizing model: nervousness revisited, *Operations Research* **42**, 456–468.

FEDERGRUEN, A., AND M. TZUR [1995], Fast solution and detection of minimal forecast horizons in dynamic programs with a single indicator of the future: applications to dynamic lot-sizing models, *Management Science* **41**, 874–893.

FEDERGRUEN, A., AND M. TZUR [1996], Detection of minimal forecast horizons in dynamic programs with multiple indicators of the future, *Naval Research Logistics* **43**, 169–189.

FELDMAN, J.A., AND D.H. BALLARD [1982], Connectionist models and their properties, *Cognitive science* **6**, 205–254.

FELLER, W. [1968], *An Introduction to Probability Theory and Its Applications, Volume I* (third ed.), John Wiley & Sons, New York.

FLETCHER, R. [1987], *Practical Methods of Optimization* (second ed.), John Wiley & Sons, New York.

FLORIAN, M., J.K. LENSTRA, AND A.H.G. RINNOOY KAN [1980], Deterministic production planning: algorithms and complexity, *Management Science* **26**, 669–679.

FOO, S.Y., AND H.S. TAKEFUJI [1995], Scaling properties of neural networks for job-shop scheduling, *Neurocomputing: an International Journal* **8**, 79–92.

FUKUNAGA, K., AND P.M. NARENDRA [1975], A branch and bound algorithm for computing k-nearest neighbors, *IEEE Transactions on Computers* **24**, 750–753.

FUNAHASHI, K. [1989], On the approximate realization of continuous mappings by neural networks, *Neural Networks* **2**, 183–192.

GIBSON, G.J. [1993], A combinatorial approach to understanding perceptron capabilities, *IEEE Transactions on Neural Networks* **4**, 989–992.

GIBSON, G.J., AND C.F.N. COWAN [1990], On the decision regions of multilayer perceptrons, *Proceedings of the IEEE* **78**, 1590–1594.

GORHAM, T. [1968], Dynamic order quantities, *Production and Inventory Management* **9**, 75–81.

HALEY, P.J., AND D. SOLOWAY [1992], Extrapolation limitations of multilayer feedforward neural networks, *Neural Networks: IEEE International Joint Conference* **IV**, 25–30.

HAMPSHIRE, J.B., AND B. PEARLMUTTER [1991], Equivalence proofs for multilayer perceptron classifiers and the Bayesian discriminant function, in: D.S. Touretzky, J.L. Elman, T.J. Sejnowski, and G.E. Hinton (eds.), *Connectionist Models: Proceedings of the 1990 Summer School*, Morgan Kaufmann Publishers, San Mateo, 159–172.

HARRIS, F.W. [1913], How many parts to make at once, *Factory: The Magazine of Management* **10**, 135–136.

HAX, A.C., AND D. CANDEA [1984], *Production and Inventory Management*, Prentice-Hall, Englewood Cliffs, New Jersey.

HECHT-NIELSEN, R. [1990], *Neurocomputing*, Addison-Wesley, New York.

HERTZ, J., A. KROGH, AND R.G. PALMER [1991], *Introduction to the Theory of Neural Computation*, Addison-Wesley, New York.

HESKES, T., E.T.P. SLIJPEN, AND B. KAPPEN [1993], Cooling schedules for learning in neural networks, *Physical Review E* **47**, 4457–4464.

HESKES, T., AND W. WIEGERINCK [1996], A theoretical comparison of batch-mode, on-line, cyclic, and almost-cyclic learning, *IEEE Transactions on Neural Networks* **7**, 919–925.

HONAVAR, V. [1994], *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration*, Chapter XXV, 615–644. Academic Press, London.

HORNIK, K. [1991], Functional approximation and learning in artificial neural networks, *Neural Networks* **4**, 251–257.

HORNIK, K., M. STINCHCOMBE, AND H. WHITE [1989], Multilayer feedforward networks are universal approximators, *Neural Networks* **2**, 359–366.

HUANG, W.Y., AND R.P. LIPPMANN [1988], Neural nets and traditional classifiers, in: D.Z. Anderson (ed.), *Neural Information Processing Systems*, American Institute of Physics, New York, 387–396.

JAGANNATHAN, R., AND M.R. RAO [1973], A class of deterministic production planning problems, *Management Science* **19**, 1295–1300.

JOERDING, W.H., AND J.L. MEADOR [1991], Encoding a priori information in feedforward networks, *Neural Networks* **4**, 847–856.

KARP, R.M. [1992], On-line algorithms versus off-line algorithms: how much is it worth to know the future?, in: J. van Leeuwen (ed.), *Algorithms, Software, Architecture*, Elsevier Science Publishers, 416–429.

KIRKPATRICK, S., C.D. GELATT, JR., AND M.P. VECCHI [1983], Optimization by simulated annealing, *Science* **220**, 671–680.

KOHONEN, T. [1988], *Self-organization and Associative Memory*, Springer-Verlag, Berlin.

KOSKO, B. [1992], *Neural Networks and Fuzzy Systems*, Prentice-Hall, Englewood Cliffs, New Jersey.

KRAAIJ, M.J.A.L. VAN [1991], The use of neural networks for lot-sizing, Master's thesis, Eindhoven University of Technology, (in Dutch).

LEE, C.-Y., AND E.V. DENARDO [1986], Rolling planning horizons: error bounds for the dynamic lot size model, *Mathematics of Operations Research* **11**, 423–432.

LEE, Y.Y., B.A. KRAMER, AND C.L. HWANG [1991], A comparative study of three lot-sizing methods for the case of fuzzy demand, *International Journal of Operations and Production Management* **11**, 72–80.

LIGHT, W. [1992], Ridge functions, sigmoidal functions and neural networks, in: E.W. Cheney, C.K. Chui, and L.L. Schumaker (eds.), *Approximation Theory VII*, Acadamic Press, Bostón.

LIPPMANN, R.P. [1987], An introduction to computing with neural nets, *IEEE ASSP Magazine* **4**, 4–22.

LOOI, C.-K. [1992], Neural network methods in combinatorial optimization, *Computers & Operations Research* **19**, 191–208.

LOVE, S.F. [1973], Bounded production and inventory models with piecewise concave costs, *Management Science* **20**, 313–318.

LOW, D., AND A.R. WEBB [1990], Exploiting prior knowledge in network optimization: an illustration from medical prognosis, *Network: Computation in Neural Systems* **1**, 299–323.

LUNDIN, R.A. [1973], *Planning Horizon Procedures for Production-Inventory Systems with Concave Costs*, Ph.D. thesis, University of Chicago.

LUNDIN, R.A., AND T.E. MORTON [1975], Planning horizons for the dynamic lot size model: Zabel vs. protective procedures and computational results, *Operations Research* **23**, 711–734.

MAREN, A.J., C.T. HARSTON, AND R.M. PAP [1990], *Handbook of Neural Network Computing Applications*, Academic Press, London.

MASON, J.C., AND P.C. PARKS [1992], Selection of neural network structures – some approximation theory guidelines, in: K. Warwick, G.W. Irwin, and K.J. Hunt (eds.), *Neural Networks for Control and Systems*, IEE Control Engineering Science 46, Peter Peregrinus, Letchworth, Chapter 8.

MCCULLOCH, W.S., AND W. PITTS [1943], A logical calculus of the ideas imminent in nervous activity, *Bulletin of Mathematics and Biophysics* **5**, 115–133.

MENSCHING, J., S. GARSTKA, AND T.E. MORTON [1978], Protective planning-horizon procedures for a deterministic cash balance problem, *Operations Research* **26**, 637–652.

MICHIE, D., D.J. SPIEGELHALTER, AND C.C. TAYLOR (eds.) [1994], *Machine Learning, Neural and Statistical Classification*, Ellis Horwood series in artificial intelligence, Ellis Horwood, London.

MINSKY, M., AND S. PAPERT [1969], *Perceptrons*, MIT Press, Cambridge, MA.

MORTON, T.E. [1979], Infinite-horizon dynamic programming – a planning horizon formulation, *Operations Research* **27**, 730–742.

MORTON, T.E. [1981], Forward algorithms for forward-thinking managers, in: R.L. Schultz (ed.), *Applications of Management Science: A Research Annual*, JAI Press, London, 1–55.

NUNEN, J.A.E.E. VAN, AND J. WESSELS [1978], Multi-item lot size determination and scheduling under capacity constraints, *European Journal of Operational Research* **2**, 36–41.

PAO, Y.H. [1989], *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Amsterdam.

PAPADIMITRIOU, C.H., AND K. STEIGLITZ [1982], *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, New York.

PARKER, D.B. [1985], *Learning logic*, Technical Report TR-47, MIT Center for Research in Computational Economics and Management Science, Cambridge, MA.

PERRONE, M.P., AND L.N. COOPER [1993], When networks disagree: ensemble methods for hybrid neural networks, in: R.J. Mammone (ed.), *Artificial Neural Networks for Speech and Vision*, Chapman & Hall, London, 126–142.

RAJAGOPALAN, S. [1994], Capacity expansion with alternative technology choices, *European Journal of Operational Research* **77**, 392–403.

RIPLEY, B.D. [1994], Neural networks and related methods for classification, *Journal of the Royal Statistical Society* **56**, 409–456.

RIPLEY, B.D. [1996], *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, MA.

RITCHIE, E., AND A.K. TSADO [1986], Review of lot-sizing techniques for deterministic time-varying demand, *Production and Inventory Management* **27**, 65–97.

ROSENBLATT, R. [1958], The perceptron: A probabilistic model for information storage and organization in the brain, *Psychological Review* **65**, 386–408.

ROSENBLATT, R. [1962], *Principles of Neurodynamics*, Spartan Books, New York.

RUCK, D.W., S.K. ROGERS, M. KABRISKY, M.E. OXLEY, AND B.W. SUTER [1990], The multilayer perceptron as an approximation to a Bayes optimal discriminant function, *IEEE Transactions on Neural Networks* **1**, 296–298.

RUMELHART, D.E., J.L. MCCLELLAND, AND R.J. WILLIAMS [1986], Learning internal representations by error propagation, in: D.E. Rumelhart and J.L. McClelland (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, MIT Press, Cambridge, 318–362.

SANDBOTHE, R.A., AND G.L. THOMPSON [1990], A forward algorithm for the capacitated lot size model with stockouts, *Operations Research* **20**, 474–486.

SANDBOTHE, R.A., AND G.L. THOMPSON [1993], Decision horizons for the capacitated lot size model with inventory bounds and stockouts, *Computers & Operations Research* **20**, 455–465.

SCHIFFMANN, W., M. JOOST, AND R. WERNER [1992], *Optimization of the Backpropagation Algorithm for Training Multilayer Perceptrons*, Technical report, Institute of Physics, University of Koblenz, Koblenz, Germany.

SETHI, S.P., AND S. CHAND [1979], Planning horizon procedures for machine replacement models, *Management Science* **25**, 140–151.

SILVER, E.A., AND J.C. MEAL [1973], A heuristic for selecting lot size quantities for the case of deterministic time-varying demand rate and discrete opportunities for replenishment, *Production and Inventory Management* **14**, 64–74.

SILVER, E.A., AND R. PETERSON [1985], *Decision Systems for Inventory Management and Production Planning*, John Wiley & Sons, New York.

SLEATOR, D.D., AND R.E. TARJAN [1985], Self-adjusting binary search trees, *Journal of the Association for Computing Machinery* **32**, 652–686.

SMYTH, S.G. [1992], Designing multilayer perceptrons from nearest-neighbor systems, *IEEE Transactions on Neural Networks* **3**, 329–333.

STEHOUWER, H.P., E.H.L. AARTS, AND J. WESSELS [1994], *On the applicability of neural networks for production planning under uncertainty*, Memorandum COSOR 94-32, Eindhoven University of Technology, Department of Mathematics and Computing Science.

STEHOUWER, H.P., E.H.L. AARTS, AND J. WESSELS [1995], Multi-layered perceptrons for on-line lot sizing, *Proceedings INRIA/IEEE Symposium on Emerging Technologies and Factory Automation, Vol. 3*, Paris, France, 279–287.

STEHOUWER, H.P., E.H.L. AARTS, AND J. WESSELS [1996], *Learning to detect planning horizons with multi-layered perceptrons: a case study for lot-sizing*, Memorandum COSOR 96-18, Eindhoven University of Technology, (Accepted for publication in: International Journal of Production Economics).

SUURMOND, R.T. [1996], Using multi-layered perceptrons for production planning: the capacitated lot-sizing model with stockouts, Master's thesis, Eindhoven University of Technology.

SWOVELAND, C. [1975], A deterministic multi-period production planning model with piecewise concave production and holding-backorder costs, *Management Science* **21**, 1007–1013.

TIJMS, H.C. [1994], *Stochastic Models: An Algorithmic Approach*, John Wiley & Sons, New York.

TILBORG, H.C.A. [1988], *An Introduction to Cryptology*, The Kluwer International Series in Egineering and Computing Science, Kluwer Academic Publishers, Dordrecht.

UDAYABHANU, V., AND T.E. MORTON [1988], Planning horizons for capacity expansion, *European Journal of Operational Research* **34**, 297–307.

VACHANI, R. [1992], Performance of heuristics for the uncapacitated lot-size problem, *Naval Research Logistics* **39**, 801–813.

VEINOTT, A.F. [1969], Minimum concave-cost solution of leontief substitution models of multi-facility inventory systems, *Operations Research* **17**, 262–291.

WAGELMANS, A., S. VAN HOESEL, AND A. KOOLEN [1992], Economic lot sizing: an $O(n \log n)$ algorithm that runs in linear time in the Wagner-Whitin case, *Operations Research* **40**, 145–156.

WAGNER, H.M., AND T. WHITIN [1958], Dynamic version of the economic lot size model, *Management Science* **5**, 89–96.

WEIGEND, A.S. [1993], Book review: Hertz, J.A., Krogh, A.S., and Palmer, R.G., Introduction to the Theory of Neural Computation, *Artificial Intelligence* **62**, 93–111.

WEIGEND, A.S., AND N.A. GERSHENFELD (eds.) [1994], *Time Series Prediction: Forecasting the Future and Understanding the Past*, Santa Fe Institute Studies in the Sciences of Complexity 15, Addison-Wesley, New York.

WERBOS, P.J. [1974], *Beyond Regression: New Tools for Prediction and Analysis in the Behavioural Sciences*, Ph.D. thesis, Harvard University, Boston.

WESSELS, L.F.A., AND E. BARNARD [1992], Avoiding false local minima by proper initialization of connections, *IEEE Transactions on Neural Networks* **3**, 899–905.

WIDROW, B., AND M.E. HOFF [1960], Adaptive switching circuits, *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record* **Part 4**, 96–104.

WIEGERINCK, W., A. KOMODA, AND T. HESKES [1994], Stochastic dynamics of learning with momentum in neural networks, *Journal of Physics A* **27**, 4425–4437.

WONG, B.K., T.A. BODNOVICH, AND S. YAKUP [1997], Neural network applications in business: a review and analysis of the literature (1988-95), *Decision Support Systems* **19**, 301–320.

XU, L., S. KLASA, AND A. YUILLE [1992], Recent advances on techniques of static feedforward networks with supervised learning, *International Journal of Neural Systems* **3**, 253–290.

YEUNG, D.-Y. [1993], Constructive neural networks as estimators of Bayesian discriminant functions, *Pattern Recognition* **26**, 189–204.

ZABEL, E. [1964], Some generalizations of an inventory planning horizon theorem, *Management Science* **10**, 465–471.

ZADEH, L.A. [1965], Fuzzy sets, *Information and Control* **8**, 338–353.

ZANGWILL, W.I. [1968], Minimum concave cost flows in certain networks, *Management Science* **14**, 429–450.

ZANGWILL, W.I. [1969], A backlogging model and a multi-echelon model of a dynamic lot size production system: a network approach, *Management science* **15**, 506–527.

ZHANG, H.-C., AND S.H. HUANG [1995], Applications of neural networks in manufacturing: a state-of-the-art survey, *International Journal of Production Research* **33**, 705–728.

ZOLLER, K., AND A. ROBRADE [1988], Efficient heuristics for dynamic lot sizing, *International Journal of Production Research* **26**, 249–265.

ZWIETERING, P.J. [1994], *The Complexity of Multi-Layered Perceptrons*, Ph.D. thesis, Eindhoven University of Technology.

ZWIETERING, P.J., E.H.L. AARTS, AND J. WESSELS [1991], The design and complexity of exact multi-layered perceptrons, *International Journal of Neural Systems* **2**, 185–199.

ZWIETERING, P.J., E.H.L. AARTS, AND J. WESSELS [1992], Exact classification with two-layered perceptrons, *International Journal of Neural Systems* **3**, 143–156.

ZWIETERING, P.J., M.J.A.L. VAN KRAAIJ, E.H.L. AARTS, AND J. WESSELS [1991], Neural networks and production planning, *Proceedings of the Fourth International Conference on Neural Networks and their Applications*, Nîmes, France, 529–542.

Author index

Subject index

Samenvatting

Dit proefschrift beschouwt situaties waarin de productie van één eindproduct moet worden gepland voor opeenvolgende perioden in de tijd. We nemen aan dat de vraag naar product een vast aantal perioden vooruit bekend is. Dit aantal wordt de *data horizon* genoemd. Er moet altijd aan de vraag naar product in een bepaalde periode worden voldaan. Er worden productie- en voorraadkosten beschouwd, en het is zaak een zo goedkoop mogelijk productieplan te vinden waarin op tijd aan alle vraag naar product wordt voldaan. Dit soort problemen worden ook wel *seriegroottebepalingsproblemen* genoemd. Gezien de manier waarop de vraag naar product bekend wordt spreken we van *on-line* seriegroottebepaling.

De klassiek wiskundige aanpak voor dit soort problemen is de onzekerheid in toekomstige vraag naar product te modelleren en het zo ontstane model te analyseren. Zulke analyses zijn vaak lastig en vergen kennis en begrip van het vraagproces. Mede hierdoor worden in de praktijk vaak eenvoudige heuristieken gebruikt waarvan de prestatie vaak te wensen overlaat. Dit onderzoek kijkt in hoeverre neurale netwerken voor verbetering kunnen zorgen. In het bijzonder kijken we naar het gebruik van meerlaags perceptrons.

In Hoofdstuk 2 formuleren we het *on-line* seriegroottebepalingsprobleem met een willekeurige kostenstructuur. Verder introduceren we een klasse van oplossingsstrategieën die we *variabele-horizon strategieën* noemen. Zulke strategieën bepalen de seriegroottes door herhaaldelijk te optimaliseren over een variabele horizon. Een horizon-selectie regel kiest zo'n horizon op basis van de beschikbare vraaggegevens. Bovendien worden *voorwaartse algoritmen* afgeleid die gebruikt worden voor het berekenen van leervoorbeelden. Deze algoritmen zijn gedeeltelijk generiek, zodat er voor toepassing op een specifieke kostenstructuur nog extra analyse nodig is. In Hoofdstuk 3 geven we deze analyse voor drie elementaire kostenstructuren. Deze kostenstructuren worden voor testdoeleinden gebruikt in de experimenten in Hoofdstuk 6 en Hoofdstuk 7.

Hoofdstuk 4 introduceert meerlaags perceptrons en bespreekt hun nut voor statistische classificatie. In het bijzonder kijken we naar de vermogens van meerlaags perceptrons om te leren en te generaliseren op basis van leervoorbeelden.

In Hoofdstuk 5 beschouwen we het probleem om een optimale horizon-selectie regel te vinden. Dit probleem kunnen we formuleren en analyseren als een classificatieprobleem. We beschouwen twee doelstellingen: maximalisatie van classificatiegraad en minimalisatie van verwachte kosten. Voor deze doelstellingen geven we expliciete uitdrukkingen voor de optimale horizon-selectie regels. Deze regels bevatten nog onbekende grootheden, zoals bijvoorbeeld a posteriori kansen. Meerlaags perceptrons worden gebruikt om deze grootheden te schatten. Door deze schattingen te substitueren in de expressies voor de optimale regels krijgen we op meerlaags perceptrons gebaseerde horizon-selectie regels.

Hoofdstuk 6 onderzoekt de generaliserende vermogens van de op meerlaags perceptrons gebaseerde horizon-selectie regels voor een on-line seriegroottebepalingsproblemen met Wagner-Whitin kostenstructuur. We bediscussiëren de noodzakelijke condities voor goede generalisatie en onderzoeken het effect van de lengte van de data horizons op deze vermogens. Het blijkt dat de condities voor goede generalisatie verslechteren als de hoeveelheid informatie over de toekomst toeneemt.

In Hoofdstuk 7 onderzoeken we de prestaties van de op meerlaags perceptrons gebaseerde horizon-selectie regels wanneer ze gebruikt worden in een variabele-horizon strategie. Dit doen we door middel van een omvangrijke empirische studie waarin de prestaties van deze strategieën worden vergeleken met die van andere strategieën. Deze studie laat zien dat, in situaties met grote vraagonzekerheid, de op meerlaags perceptrons gebaseerde strategieën beter presteren dan alle andere strategieën met betrekking tot robuustheid, data efficiëntie en kosten. Als er daarentegen weinig onzekerheid is met betrekking tot de toekomstige vraag naar product heeft het weinig zin meerlaags perceptrons te gebruiken.

Hoofdstuk 8 sluit het proefschrift af met een discussie van de bereikte resultaten en suggesties voor verder onderzoek.

Curriculum vitae

Peter Stehouwer was born on November 6, 1968 in Wageningen, the Netherlands. In 1987 he completed his secondary school education at the Christelijk Lyceum in Zeist and moved to Eindhoven to study computing science at the Eindhoven University of Technology. During his studies Peter worked freelance as a software engineer for Stichting Van Haren Pensioenfonds in Waalwijk and was employed as a teaching assistant at the Department of Industrial Engineering of the Eindhoven University of Technology. In February 1993 he graduated on the subject of using neural networks for the traveling salesman problem. His Master's thesis was written under the supervision of prof.dr. E.H.L. Aarts and ir. P.J. Zwietering.

After his graduation he started a Ph.D. research project at the Department of Mathematics and Computing Science of the Eindhoven University of Technology under the supervision of prof.dr. J. Wessels and prof.dr. E.H.L. Aarts. His project was entitled *Anticipative and adaptive planning with neural networks* and was financed by the Dutch Organization for Scientific Research (NWO). This thesis is the result of the research he performed. Peter is currently employed as a consultant at the Centre for Quantitative Methods CQM B.V. in Eindhoven.

Stellingen

behorende bij het proefschrift

On-line lot-sizing with perceptrons

van

H.P. Stehouwer

Stellingen

behorende bij het proefschrift

On-line lot-sizing with perceptrons

van

H.P. Stehouwer

I

Beschouw een handelsreizigersprobleem met stedenverzameling $S = \{s_1, \ldots, s_n\}$. Zij voor elk paar steden s_p, s_q hun onderlinge afstand gegeven door $\Delta(s_p, s_q)$. Beschouw tevens een graaf $\mathcal{G}$ met puntenverzameling $V = \{v_1, \ldots, v_n\}$, waarbij $\mathcal{G}$ een cykel is. Definieer voor elk paar punten v_s, v_t hun onderlinge afstand $\Delta_{\mathcal{G}}(v_s, v_t)$ als de lengte van het kortste pad van v_s naar v_t in $\mathcal{G}$. Een afbeelding $f : V \rightarrow S$ heet *topologie behoudend* als

$$\forall_{v_p, v_q, v_r \in V} : \Delta_{\mathcal{G}}(v_p, v_q) < \Delta_{\mathcal{G}}(v_p, v_r) \Rightarrow \Delta(f(v_p), f(v_q)) < \Delta(f(v_p), f(v_r)).$$

Zij $f : V \rightarrow S$ een bijectieve topologie behoudende afbeelding. Dan is de tour die $\mathcal{G}$ induceert door middel van f optimaal.

[1] H.P. Stehouwer (1993), Self organizing feature maps and the travelling salesman problem: a theoretical study, Master's thesis, Eindhoven University of Technology.

II

Bowman [1] gebruikt in zijn *Management Coefficients Theory* voorbeelden van het beslissingsgedrag van managers in het verleden ter verbetering van hun beslissingsgedrag in het heden. Neurale netwerken kunnen uitstekend gebruikt worden bij de implementatie van deze theorie in beslissingsondersteunende systemen.

[1] E.H. Bowman (1963), Consistency and optimality in managerial decision making, *Management Science* **9**, 310–321.

III

On-line beslissingsproblemen lenen zich voor een hybride aanpak op basis van neurale netwerken en deterministische technieken. Hierbij is het zaak de puzzlekwaliteiten van deterministische technieken te combineren met het vermogen van neurale netwerken om met onzekerheid om te gaan.

[1] H.P. Stehouwer (1997), dit proefschrift.

IV

In [1] wordt bewezen dat in veel leeralgoritmen voor*feedforward* netwerken het veranderen van de steilheid van de responsefunctie equivalent is aan het veranderen van de stapgrootte van het leeralgoritme en de initiële gewichten. Dit resultaat elimineert de noodzaak om de steilheid van de responsefunctie te bepalen.

[1] G. Thimm, P. Moerland en E. Fiesler (1996), The interchangeability of learning rate and gain in backpropagation neural networks, *Neural Computation* **8**, 451–460.

V

Het meerdere malen publiceren van exact hetzelfde artikel [1,2] kan een gunstige uitwerking hebben op het aantal keren dat er naar dit artikel verwezen wordt [3].

[1] V.S. Badami en C.M. Parks (1991), A classifier based approach to flow shop scheduling, *Computers and Industrial Engineering* **21**, 329–333.

[2] V.S. Badami en C.M. Parks (1991), A classifier based approach to flow shop scheduling, *Computers and Industrial Engineering* **21**, 401-405.

[3] C.H. Dagli (1994), *Artificial Neural Networks for Intelligent Manufacturing*, Chapman & Hall, London.

VI

De benaming milleniumprobleem voor het gegeven dat men in veel computerprogrammatuur slechts twee numerieke posities voor een jaartal heeft gereserveerd is onjuist. Het gaat om een eeuwprobleem.

VII

Het feit dat gedogen als juridisch fenomeen uitsluitend voorkomt in het Nederlandse recht zou de overheid te denken moeten geven.

VIII

Gezien de toenemende ongedisciplineerdheid van de weggebruikers zou het veiliger zijn om bij bewaakte spoorwegovergangen pas de spoorbomen te openen als de rode lichten gedoofd zijn.

IX

Een van de voorwaarden voor het slagen van een milieubeleid is de algemene bewustwording van het feit dat de aarde niet van de mens maar de mens van de aarde is.

X

De kans op vormfouten neemt toe met het “kaliber” van de misdadiger.

XI

Deze stelling is onwaar indien goedgekeurd door de rector.

XII

Een onderzoeker vindt het in zijn bovenkamer.

Eindhoven
University of Technology

Department of Mathematics
and Computing Science

BETA

Institute for
Business Engineering
and Technology Application