Markdown Optimization via Approximate Dynamic Programming

Özlem COŞGUN[†]
Fatih University, Department of Industrial Engineering,
Istanbul, 34500, Turkey

Ufuk KULA
Sakarya University, Department of Industrial Engineering,
Sakarya, Turkey

Cengiz KAHRAMAN
Istanbul Technical University, Department of Industrial Engineering,
Istanbul, Turkey

Received 10 March 2012; accepted 4 September 2012

Abstract

We consider the markdown optimization problem faced by the leading apparel retail chain. Because of substitution among products the markdown policy of one product affects the sales of other products. Therefore, markdown policies for product groups having a significant crossprice elasticity among each other should be jointly determined. Since the state space of the problem is very huge, we use Approximate Dynamic Programming. Finally, we provide insights on the behavior of how each product price affects the markdown policy.

Keywords: Approximate Dynamic Programming, Multinomial logit model, Cross-price elasticity, Markdown optimization, Pricing

1. Introduction

Textile industry is an important part of Turkish economy. One of the factors that will have an impact on competitiveness of our textile industry in global markets is its ability to use information technology based decision support and management systems.

Apparel industry is one of the main driver of Turkish textile industry. Given the fact that apparel manufacturers tend to become retailers at the same time, (e.g. L.C. Waikiki, Mavi Jeans etc.) the performance of our apparel industry will depend also on how well these companies are competing against the global retailers. Success of national apparel retailers will depend on whether they can make good use of the data generated by information technologies in their decision making processes. Retail systems cover two main areas of management, namely supply chain and revenue management, where critical decisions are made. Determining the most appropriate prices over time to maximize profits is one essential component of revenue management, whereas capacity control is the other. In retail revenue management, the pricing control is the main tool to manage the demand.

The ever increasingly shortening selling seasons for fashion (apparel) products pressure the firms to eliminate or minimize distressed inventories to maximize revenues. One of the frequently used mechanisms to achieve this goal is markdowns. Markdowns are permanent price reductions used to clear inventory before products become obsolete. To the best of our knowledge, prior work on markdown optimization has focused only on single-product markdowns assuming that each product has an independent demand process. However, we believe that it is crucially important to consider the correlations and the interactions among products in markdown optimization. Thus, the main focus of our study is to develop methodologies for multi-product markdowns. Since products typically exhibit substitution, complementarity effects, and cross price elasticity, markdown policies for these products should be jointly determined.

Markdowns in apparel retailing industry were started to use firstly in 1950s in USA but then they weren't used broadly. After 1960s, their frequencies and discount rates increased. 6% average discount rate in 1967 increased to 28% in 1997 in apparel stores[1]. It is expected to increase for the coming years since 1) consumers can reach the stores in different places easily 2) the outlet store number increases 3) consumers want large product assortment. Parallel to this increase, interest in markdown optimization problems was accelerated.

These prior studies on markdown optimization have focused only on single product markdowns. However, consideration of multi-product optimization may lead to significant revenue

[†] Özlem Coşgun, ozlem_ince@hotmail.com, Fatih University, Department of Industrial Engineering, Istanbul, 34500, Turkey

increase since there may be significant crossprice elasticities among products. Cross-price elasticity measures the percent change in one product's demand when the price of an other products is increased. It may be either positive or negative depending on whether the demand of a product increases or decreases when the price of an other product goes up. If the cross-price elasticity between products A and B is positive, it means that product A is a substitute of product B, since an increase in product B's price causes a demand increase in product A's demand. On the other hand, if the cross-price elasticity between products A and B is negative, then products A and B complement each other since a decrease in product A demand occurs due to an increase in product B price. According to a study made in USA, 12% of the retailers use markdown optimization and 53% of the retailers plan to use such a system in 2 years[2]. The properties of the products such a markdown application applied are unrenewable orders in the same season because of long manufacturing lead times and decreasing product value. Because of this property we select end of season apparel products that are unrenewable. This problem is more rampant in the fashion and electronic retail industry where products get outdated quickly. The aim in markdown optimization is to minimize the inventory levels at the end of the season to maximize the revenue by decreasing prices.

The first studies about dynamic pricing are in marketing area in the literature. They aimed which dynamic pricing strategies are used in which conditions but they didn't consider operational dynamic pricing policies that will be applied in practice. In retail industry, Lazear studied dynamic pricing problem firstly[3]. In this study, N customers come to the store according to a known distribution and reservation price is known, and they appraise the product a low price with p probability. Lazear shows the effects of reservation price on price[3]. Pashigian adapts the model in Lazear and offers analytical and experimental results about the reason of markdowns is increasing product assortment[4]. Since the aim of these studies is understanding pricing strategies briefly, they are far away from being a decision tool used in pricing. All models in the literature focus on the single product markdowns and they didn't consider the correlations and substitution effects between the products. Rajan et al. analyse the optimal inventory levels and policies under deterministic demand while product value decreases by the time[5]. Another study observes the markdown under deterministic demand is Smith and Achabal's study[6]. They determine the optimal inventory levels and optimal prices by developing a nonlinear mathematical model. Demand depends on price, inventory levels and seasonal variations. Gallego and Van Ryzin develop a continous time optimal pricing model in which demand is described by a Poisson process or is deterministic and they determine the optimal prices, moreover they develop a heuristic for discrete prices[7]. Feng and Gallego develop a continous time Markov Process formulation with stochastic demand that determines the optimal timing and duration of a single price reduction[8]. Feng and Xiao adapt the problem in Feng and Gallego to the more than two prices[9]. Bitran et al. consider the one product markdown problem in more than one store and model it by using dynamic programming, but in practice, since the state space is large, the solutions of these problems are impossible by using classical dynamic programming. Because of this, they develop a heuristic and test with the retailing sector real data[10]. Mantrala and Rao developed a stochastic dynamic-programming model-based decision-support system, specifically to help retail-store buyers of fashion goods decide on optimal merchandise order quantities and markdown prices. This decision support system uses point of sales data to determine optimal initial inventory levels and prices[11]. However they didn't consider the correlations and substitution effects among the products, they only handle one product case and determine its initial inventory level. Su develops a model of dynamic pricing but their model captures both markups and markdowns for a single product. Since the customer population is heterogenous, time to buy for the customers is important. So he finds different policies for each cosumer group[12]. Reiner and Natter develop markdown pricing strategies on Austrian mobile phone market. They consider different markdown strategies on two different consumer groups[13]. Elmaghraby and Keskinocak analyze the optimal design of a markdown pricing mechanism with preannounced prices and their suitability in the presence of strategic buyers with multiunit demands[14]. All these studies don't involve the substitution effects among the products. The studies involve more than one product in revenue management literature consider the products that share the same resource in production or delivery of the products. However, we know that it is crucially important to consider the correlations and substitution effects between the products in markdown optimization. The substitution possibilities in retailing can be classified into three groups[15]. We consider the fourth one, price-based substitution: Consumer comes to the store to buy a certain product that she needs but she sees a substitute product that is cheaper than that product and decides to purchase the cheaper one. To the best of our knowledge, such a price-based substitution has not been studied. Therefore the contribution of this paper is to consider the price-based substitution for a given substitution product group for multiple periods.

In this study we use point of sales data gathered from 250 stores in apparel retailing industry and analyse data by using SAS software package to determine the product groups that have substitution effects. Then the cross price elasticities of the products are observed for the same product group.
Consumers face trade-offs in their purchase decisions, since their income is limited and choices are numerous. In order to make choices, consumers must combine budget constraints (what they can afford), and preferences (what they would like to consume). A budget contraint, means what a consumer can purchase is constrained by income. The slope of the budget constraint measures the rate at which one consumer can trade off one good for another, and the relative prices of the two goods. Budget constraints are determined by both the income of the consumers, and the relative prices. If a consumer equally prefers two product bundles, then the consumer is indifferent between the two bundles. The consumer will get the same level of satisfaction (utility) from either bundles. Therefore consumer behaviors are important to analyse the substitution effects and they should be considered in the model. Customer behavior modeling has been gaining increasing attention in the operations management community[16]. We use Multinomial Logit Model which is one of the discrete choice models and used mostly in marketing literatures to estimate the substitute demand.
In the apparel industry since we can have many products that may substitute each other, the markdown policy of one product affects the sales of other products. Therefore, markdown policies for product groups should be jointly determined. But this makes the markdown optimization problem hard. Since the states will be multidimensional, state space will be large and as a result solving such a problem by classical dynamic programming methods will be impossible. To overcome this problem, Approximate Dynamic Programming is used. We use Sarsa Algorithm which is a policy iteration method.
In the next section, consumer choice model and demand estimation take place. Then we mention about the Approximate Dynamic Programming (ADP) model. In section 4, numerical examples follow this section and finally conclusion part is considered.

2. Model

In this section, we will first discuss the assumptions of the consumer choice model considered in this paper. Then, the consumer choice process will be considered in detail and the purchase probabilities will be derived for the MNL model. We assume that we have a homogenous consumer group that the characteristics (such as income level, age, etc.) are thought as similar. Their purchase behaviour can change only with respect to the different price levels. The data typically available for estimating the parameters of a demand model include the sales for each product-week, prices for related weeks and the inventory of the products are obtained from an apparel company. Since we observe point of sales data, we only know the consumers that made transactions and don’t know their characteristics. Hence, we think that the target consumer group is homogenous.

2.1. *Demand Estimation*

We analyse the sales data by using data mining algorithms to form the multi-product groups that are substitutable. Firstly, 14,558 units of different models are observed from the sales data. Association Mining Algorithm is used to form the multi-product groups, but before that basic products whose sales aren’t affected from the seasonality are extracted from the observation. Because we can not apply markdown to these products. After filtering the products, the products that markdown will be applied are decided by talking with the company officials. In this approximation, we perform the following steps:
(a) Products are ordered in descending order according to the number of sales and started from the highest number to form the groups.
(b) One product that has the highest positive relations with the other products in the same product group is chosen.
(c) For the same product, negative relations are chosen from the remaining products. Therefore, the product has both positive (complementarity) and negative (substitution) relations.
(d) The (b) and (c) steps are repeated until all rules are extracted.
Firstly, we find a subset of 55 products from 600 products that markdown can be applied and we decide on the products which are in the same group by the replication of the steps. The demand of each product in the selected product groups is estimated by using the regression model. At last, we solve both the deterministic model and stochastic model to compare the results by using the same demand estimation. The deterministic solution is found by using GAMS software and the obtained optimal demand is used as a sample path in stochastic model to make comparison.
Demand function D_{it} for each product i at time t depends on the price of the observed product, the other products’ prices in the same group and the time.

$$D_{it} = \beta_{0i} + \beta_{1i}\text{Price}_{it} + \sum_{k \neq i} \beta_{2k}\text{Price}_{kt} + \beta_{3i}t_i + \varepsilon_t \quad (1)$$

β is the coefficient of the attributes (prices, time) of product i. Error term ε is normally distributed with mean μ and standard deviation σ. Very often, a change in the price of one product leads to a change in the demand of another, is called cross-price elasticity. Cross price elasticity of demand estimation is modelled by using discrete choice models. We use Multinomial Logit Model (MNL) which is very widely used in practical applications[17].

2.1.1. *Multinomial Logit Model*

The Multinomial Logit (MNL) model is a utility-based model that is commonly used in economics and marketing literatures. Each customer visiting the store associates a utility u_j with each option $j \in J$. The utility is decomposed into two parts, the deterministic component of the utility v_j also known as representative utility and a random component ε_j.

$$u_j = v_j + \varepsilon_j \tag{2}$$

The random component is modeled as a Gumbel random variable. Also known as Double Exponential or Extreme value Type-I, it is characterized by the distribution $\Pr\{X \le \varepsilon\} = \exp(-\exp-(\varepsilon/\mu+\gamma))$ where γ is Euler's constant (0.57722). Its mean is zero, and variance is $\mu^2\pi^2/6$. A higher μ implies a higher degree of heterogeneity among the customers. The realizations of ε_j are independent across consumers. Therefore, while each consumer has the same expected utility for each product, realized utility may be different. This can be due to the heterogeneity of preferences across customers or unobservable factors in the utility of the product to the individual. An individual chooses the product with the highest utility among the set of available choices. Hence, the probability that an individual chooses product $j \in J$ at time t is

$$P_{jt} = \frac{e^{V_{jt}}}{\sum_{i \in J} e^{V_{it}}}. \tag{3}$$

where the deterministic part of utility v_{jt} is

$$v_{jt} = \beta_{0j} + \beta_{1j} Price_{jt} + \sum_{i \ne j} \beta_{2i} Price_{it} + \beta_{3j} t_j \tag{4}$$

We assumed that the utility of a product is expressed by its price, the others' prices and time. Generally, when the price of the product increases, its utility decreases and while the others' prices decrease, its utility increases since it is thought that the purchase probabilities increase according to the MNL Model. Then the logit probability becomes

$$P_{jt} = \frac{e^{\beta_{0j} + \beta_{1j} Price_{jt} + \sum_{n \ne j} \beta_{2n} Price_{nt} + \beta_{3j} t_j}}{\sum_{i \in J} e^{\beta_{0i} + \beta_{1i} Price_{it} + \sum_{n \ne i} \beta_{2n} \mathrm{Pr}ice_{nt} + \beta_{3i} t_i}} \tag{5}$$

One of our key objectives here is to test the sensitivities of "price" and "time" to observe the elasticities of the products. We can calculate the utility v_{jt} from sales data. Firstly, we compute pr_{jt} from the sales data which is the ratio of number of customers that bought product j to the number of the customers that bought any product in that multi-product group on week t. This can be considered as the demand share of product j over all products

$$pr_{jt} = \frac{D_{jt}}{\sum_i D_{it}} \tag{6}$$

The utility of a product (7) is expressed as the difference between the market share of that product and the average of all market $\overline{pr_t}$ [18]. That is if market share of a product is higher than the average market share of all products, that product will have more utility.

$$\begin{aligned} pr_{jt} - \overline{pr}_t = v_{jt} &= \beta_{0j} + \beta_{1j}(prc_{jt} - \overline{prc}) + \\ &\sum_{i \ne j} \beta_{2i}(prc_{it} - \overline{prc}) + \beta_{time_j} \end{aligned} \tag{7}$$

where prc denotes the price of the product in the sales data and $\overline{prc}$ denotes the average price of the multi-product group.

2.1.2. *Substitution Effect*

When we develop an attribute of a product, its choice probability usually increases if it satisfies the consumers. Because of this, we consider the consumer choice behaviors. It is also important to consider their characteristics such as their income level, real necessity or the product characteristics such as their color, type, etc. But it will more complex to determine them when we consider the correlations between the products and it is very hard to collect data individually. To provide the simplicity, we assume that the consumers are homogenous and demands are affected only from the prices and the time. Moreover, we observe the price-based and time-based substitution.
Since choice probabilities are a function of observed variables, it is often useful to know the extent to which these probabilities change in response to a change in some observed factor. The change in the probability of purchasing alternative i given a change in an observed factor, price or time, entering the representative utility of that alternative (and holding the representative utility of other alternatives constant) is

$$\frac{\partial P_{it}}{\partial \text{Price}_{it}} = \frac{\partial (e^{v_{it}} / \sum_j e^{v_{jt}})}{\partial \text{Price}_{it}}$$
$$= \frac{e^{v_{it}}}{\sum e^{v_{jt}}} \frac{\partial v_{it}}{\partial \text{Price}_{it}} - \frac{e^{v_{it}}}{(\sum e^{v_{jt}})^2} e^{v_{it}} \frac{\partial v_{it}}{\partial \text{Price}_{it}}$$
$$= \frac{\partial u_{it}}{\partial \text{Price}_{it}} (P_{it} - P_{it}^2)$$
$$= \frac{\partial v_{it}}{\partial \text{Price}_{it}} P_{it}(1 - P_{it}) = \beta_{p,it} P_{it}(1 - P_{it}) = O_{it} \tag{8}$$

where P_{it} shows the logit probability of product *i* at time *t* and $\beta_{p,it}$ shows the price coeffcient of product *i*. This is known as own-elasticity. If one product has a price reduction, its choice probability increases with this probability. Then the final demand d_{it} becomes $d_{it} = \max(0, D_{it} + O_{it} D_{it})$ where D_{it} shows the original demand of product *i* at time *t*. This can be applied for the time parameter in the same manner which we called time-elasticity. Furthermore, one can also determine the extent to which the probability of choosing a particular alternative changes when an observed variable relating to another alternative changes. The probability of choosing alternative i changes as price of the product j increases as in (9)

$$\frac{\partial P_{it}}{\partial \text{Price}_{jt}} = \frac{\partial (e^{v_{it}} / \sum_k e^{v_{kt}})}{\partial \text{Price}_{jt}}$$
$$= -\frac{e^{v_{it}}}{(\sum e^{v_{jt}})^2} e^{v_{jt}} \frac{\partial v_{it}}{\partial \text{Price}_{jt}} \tag{9}$$
$$= -\frac{\partial v_{it}}{\partial \text{Price}_{jt}} P_{it} P_{jt} = -\beta_{p,jt} P_{it} P_{jt} = C_{it}$$

This is known as cross-elasticity. Then the final demand d_{it} becomes $d_{it} = \max(0, D_{it} + C_{it} D_{it})$. We call the cross elasticity as the substitution effect. Substitution is the demand increment in one product when the other's price decreases or the demand decrement when the other's price increases[19].

2.2. *Mathematical model*

To analyse and observe the results, we develop a deterministic model called posterior solution that we assume all demands are known. We model this problem by GAMS software package and solve it by using Dicopt non-lineer solver. Initial inventory levels, initial prices and demands are known. We construct a model that optimize the prices during the periods under the markdown constraint. According to the markdown constraint, the price of one period cannot be larger than the price of the previous period. We apply 3 discount rates (10%, 30% and 50%) to the product prices for each period or the price may not change. The aim is to decide on an optimal policy for each product.

The parameters of the model is as follows:

$i =$ product index,
$t =$ time index,
$T =$ planning horizon,
$\beta_i =$ coefficients of demand function for product *i,*
$IS_i =$ the initial inventory of the product *i,*
$IP_i =$ the initial price of product *i* ,
$WIS_{it} =$ the inventory of product *i* at the beginning of week *t,*
$h_{it} =$ unit holding cost of product *i* at week *t,*
$sv_i =$ salvage value of product *i,*
$disc(k) = k^{th}$ discount rate.
$M =$ very big number.

Furthermore, the decision variables of the model are;

$z =$ objective function value,
$p_{it} =$ the price of the product *i* at week *t,*
$S_{it} =$ sales of product *i* at week *t,*
$WFS_{it} =$ the on-hand inventory of product *i* at the end of week *t,*
$FS_i =$ the on-hand inventory of product *i* at the end of the period *T,*
$WD_{it} =$ the demand of product *i* at week *t,*
$D_{it} =$ the positive demand of product *i* at week *t,*
$r_{it} =$ binary variable that if the demand at week *t* is positive, it takes a value of 1, otherwise 0,
$f(i,t,k) =$ if k^{th} discount is applied for product *i* at week *t*, it takes a value of 1, otherwise 0,

The mathematical model of the problem is as follows:

$$\max_{p_{it}} \left\{ \sum_{i=1}^{n} \sum_{t=1}^{T} p_{it} S_{it} + \sum_{i=1}^{n} sv_i FS_i - \sum_{i=1}^{n} \sum_{t=1}^{T} h_{it} WFS_{it} \right\} \tag{10}$$

subject to

$$WD_{it} = \beta_{i0} + \beta_{i1} p_{it} + \sum_{i \neq j} \beta_{i2} p_{jt} + \beta_{i3} t, \ \forall i \tag{11}$$

$$WFS_{it} = WIS_{it} - S_{it}, \quad \forall i,t \tag{12}$$
$$WIS_{i1} = IS_i, \quad \forall i \tag{13}$$
$$D_{it} \le M * r_{it}, \quad \forall i,t \tag{14}$$
$$D_{it} - WD_{it} \le M * (1 - r_{it}), \quad \forall i,t \tag{15}$$

$$S_{it} \leq D_{it}, \qquad \forall i,t \tag{16}$$

$$WIS_{it} \geq S_{it}, \qquad \forall i,t \tag{17}$$

$$FS_i = IS_i - \sum_{t=1}^{T} S_{it}, \quad \forall i \tag{18}$$

$$WIS_{it+1} = WIS_{it} - S_{it}, \quad \forall i,t \tag{19}$$

$$p_{i,t} \leq p_{i,t-1} + M*(1-f(i,t,k)), \qquad \forall i,t \tag{20}$$

$$p_{i,t+1} \leq p_{it} * disc(k) + M*(1-f(i,t+1,k)), \\ \forall i,t,k \tag{21}$$

$$\sum_k f(i,t,k) = 1, \qquad \forall i,t \tag{22}$$

$$p_{iT} \geq sv_i, \quad \forall i \tag{23}$$

$$p_{i1} \leq IP_i, \quad \forall i \tag{24}$$

(10) shows the objective function that maximizes the total profit obtained by the difference between the revenue got from the sales and salvage revenue and the holding cost of the products. (11) shows the demand function where β values are known. (12) provides that weekly final stock is equal to initial stock of that period minus the sales of that period. (13) shows that weekly initial stock of the first period equals to the given initial stock of each product. (14) and (15) constraints provide the demand of each product are positive. (16) provides that the sales of the product for each week cannot be greater than the demand of that period. (17-19) constraints are related to stock constraints. (20) shows the markdown constraint. It provides that the price action of one period cannot be greater than the previous period and the prices should be one of the discounted prices (21). This is supported by the constraint (22) that shows we can do only one discount for each period. The price of the final period should not be less than the salvage value (23) and the price of the first period should not be greater than the initial prices of each product that are given (24).

2.3. *Approximate Dynamic Programming Model*

Multi-stage decision problems under uncertainty are abundant in process industries. Markov Decision Process (MDP) is a general mathematical formulation of such problems. Whereas stochastic programming and dynamic programming are the standard methods to solve MDPs, their unwieldy computational requirements limit their usefulness in real applications. Approximate dynamic programming (ADP) combines simulation and function approximation to alleviate the 'curse-of-dimensionality' associated with the traditional dynamic programming approach. The most important concept in ADP is path generation. Through the iterations, a lot of path is generated and the approximate values of the states are expected to be estimated by visiting them. This is important in terms of the convergence of the value function[20,21,22,23].

In our problem, we consider a class of multistage problems called the markdown optimization problem faced by the retailing industry. This apparel company provides many products for men, women and children so that it has a wide range of products. It is important to know which product is substitute of another in a multi-product group. The aim is to find optimal prices of these substitutable products under markdown constraints for all inventory levels. We observe the product groups that have more than one product. Since they have correlations among them, the markdown policy of one product affects the other product. We try to determine the optimal markdown policy for each state. System state S_t is defined with inventory level of each product which is denoted by s_{jt} for product *j* at time *t* and the decision given in the previous period which is denoted by a_{jt} because of the markdown constraint. The action or decision is the discounted prices applied to the products. According to the markdown constraint, the action given at time *t* cannot be higher than action given in the previous period.

$$S_t = (s_{1t}, s_{2t}, ..., s_{kt}, a_{1,t-1}, a_{2,t-1}, ..., a_{k,t-1}) \tag{25}$$

Since the ADP algorithm runs iteratively and the demand is random, the system can be in the same or different inventory levels in one period. If the same state is visited in one iteration, it is important to know the action given at time *t*-1 in the previous iterations due to the markdown constraint. That is, the decision at time *t* will be given according to the action given at time $t-1$. In this case the possible action set $A(S_t)$ of the S_t state is,

$$A(S_t) = \{a_{1t}, a_{2t}, ..., a_{kt} : a_{1t} \leq a_{1,t-1}, \\ a_{2t} \leq a_{2,t-1}, ..., a_{k,t-1}\} \tag{26}$$

if the product group has *k* products. If the product group has 4 products and each has 5000 units and we have 5 actions, the number of states will be $5000^4 \times 5^4 = 25^4 \times 10^7$ units. When the number of products and their inventory levels increase, this size becomes larger. This is the reason of why we cannot use classical dynamic programming.

2.3.1. *Solution Methodology: Sarsa Algorithm*

We use Sarsa algorithm which is an on-policy method. For an on-policy method we must estimate action-value function $Q^{\pi}(s,a)$ for the current behavior policy π and for all states *s* and actions a. We consider transitions from state-action pair to state-action pair and learn the value of state-action pairs. During the estimation of value function V^{π}

under policy π, the state values are learned from transitions between states, but in Sarsa these values are learned from transitions between state-action pairs. As in all on-policy methods, we continually estimate Q^{π} for the behavior policy π, and at the same time π changes toward greediness with respect to Q^{π}. Sarsa converges with probability 1 to an optimal policy when action-value function as long as all state-action pairs are visited infinitely and the policy converges in the limit to the greedy policy[20,24].

The a_t^* optimal price decision of state S_t is the price that maximizes the action-value function $Q(S_t, a_t)$ over all iterations (27).

$$a_t^* = \arg\max_{a_t}(Q(S_t, a_t)) \tag{27}$$

Then π policy is defined as $\pi(S_t) = [a_{1t}^*, a_{2t}^*, ..., a_{kt}^*]$ for each state S_t. Let consider the system is in state S_t. When a new action a_t is taken and demand D_t is appeared, system state S_t transites to the S_{t+1} state. This transition function is defined as in (28).

$$S_{t+1} = (S_t, a_t, D_t(a_t)) \tag{28}$$

As stated before, demand is a function of price action and time. In this model, any unsatisfied demand from one week is not allowed to be passed on to the next period. Demand is simply "lost" if not fulfilled in the same time period. As a result, the total demand for each week is simply the new demand in that week, which is exogenous information. When demand comes, the inventory level of product *j*, s_{jt}, changes into $s_{j,t+1} = \max(0, s_{jt} - D_{jt})$. If the optimal action given at time *t* is a_t^*, the system state is then defined as $S_{t+1} = (s_{1,t+1}, s_{2,t+1}, ..., s_{k,t+1}, a_{1t}^*, a_{2t}^*, ..., a_{kt}^*)$. The aim of this stochastic optimization problem is to determine the optimal policy given in (29)

$$\max_{\pi} E\left[\sum_{t=1}^{T} Q^{\pi}(S_t, a_t^{\pi})\right] \tag{29}$$

The action value function $Q(S_t, a_t)$ is formed from two parts: immediate reward $r_t(S_t, a_t)$ and discounted cost-to-go value based on policy defined as in (30).

$$Q(S_t, a_t) = r_t(S_t, a_t) + \gamma Q(S_{t+1}, a_{t+1}^{\pi}) \tag{30}$$

The action value function $Q(S_t, a_t)$ is then updated (31) when the same states are visited through the iterations.

$$Q(S_t, a_t) \leftarrow Q(S_t, a_t) + \alpha\left[r_t + \gamma Q(S_{t+1}, a_{t+1}^{\pi}) - Q(S_t, a_t)\right] \tag{31}$$

where $\alpha^n \in [0,1]$ is the stepsize at iteration n under the standard assumptions $\sum_n \alpha^n = \infty$ and $\sum_n (\alpha^n)^2 < \infty$. The reward function $r_t(S_t, a_t)$ is the amount obtained from the difference of the sales and the inventory holding cost.

$$r_t(S_t, a_t) = \sum_{i \in J} \max(0, D_{it}) * a_{it} - \max(0, s_{it} - D_{it}) * h_{it} \tag{32}$$

where h_{it} denotes the holding cost of product *i* at time *t*.

Steps of the Sarsa algorithm is given in Fig. 1[20]. The problem here is we should perform too many iterations to visit all states sufficiently, but it takes too long time. In practice, the computational burden of looping over all states in backward dynamic programming has been replaced with the statistical problem of estimating the value of many states. It is not enough to know the value of being in states that we actually visit. If we are making good decisions, we have to have good estimates of the value of states that we might visit. This may be far smaller than the entire set of states, but it can still be an extremely large number. There is a vast array of statistical techniques that can be used to approximate the value function. We use aggregation technique which is one of the most popular techniques used in the literature.

2.3.2. *Aggregation*

We consider aggregation as a process that we combine some states which have closer inventory levels. Let take the same example that product group has 4 products and each has 5000 units and we have 5 actions. If we aggregate the states by 100 units, the number of states decreases to $(5000/100)^4.5^4 = 250^4$ units from $25^4 \times 10^7$. This means that we cluster the states whose inventory levels have 1-100 units put in one group, 101-200 units put in other group, etc. Therefore the probability of visiting the same states increases and we get more truely results. However, aggregating the states by 50 units instead of 100 units will generate different results. Because the visiting number of the same states in aggregation by 50 units can be less than the other and this will affect all results. To decrease the coefficient of variation and get more reliable results we apply 'mixed aggregation' that combines the different aggregation levels.

Let (g) shows the aggregation set index. Then the mixed aggregation value of state $S_t^{(g)}$ becomes as in (33) if we have two aggregation levels such as (g1) and (g2)

$$Q(S_t^{(g)}) = w^{(g_1)} Q(S_t^{(g_1)}) + w^{(g_2)} Q(S_t^{(g_2)}) \qquad (33)$$

where w denotes the weight applied to the estimate the value of being in state S_t at the g^{th} level of aggregation that are proportional to the mean square error (MSE) of $(g)^{th}$ aggregation sets that are calculated below[25]. Then the MSE of the single aggregation sets are updated.

$$\omega^{(g_j)}(S_t) = \frac{MSE^{(g_j)}(S_t)}{\sum_{y=1}^{2} MSE^{(g_y)}(S_t)} \qquad (34)$$

We measure our performance using MSE statistical measure. If we have an initial estimate of state S_t which is $Q^{n-1}(S_t)$, and the new value of the state $\hat{v}_n(S_t)$, we use a standard stochastic gradient (smoothing) expression of the form (35)

$$\begin{aligned} MSE^n(S_t) = (1-\alpha_{n-1}) MSE^{n-1}(S_t) + \\ \alpha_{n-1}(Q^{n-1}(S_t) - \hat{v}_n(S_t))^2 \end{aligned} \qquad (35)$$

α_{n-1} is a stepsize parameter at iteration n-1. We use McClain's formula to get the stepsize α given by

$$\alpha_n = \frac{\alpha_{n-1}}{1 + \alpha_{n-1} - \bar{\alpha}} \qquad (36)$$

where $\bar{\alpha}$ is a specified parameter.

```
Step 0: Determine $\pi$ policy arbitrarily
Step 1: Initialize $Q(s,a)$ for each $s$ and $a$
Step 2: Determine iteration number $n$ and period size $T$
Step 3: Do for $i = 1,2,...,m$
        Step 3a: determine initial state $s$
        Step 3b: choose an action $a$ ($\varepsilon$-greedy)
        Step 3c: do for $t = 1,2,...,T$
                Step 3c_1: for the chosen action $a$, observe the reward $r$ and the following
                           state $s'$
                Step 3c_2: choose actions $a'$ come from policy $\pi$ for each state $s'$
                Step 3c_3: calculate $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)]$
                Step 3c_4: $s \leftarrow s'; a \leftarrow a'$
                Step 3c_5: $\pi(s) = \arg\max_a [Q(s,a)]$
        Step 3d: if $i \le m$, go to step 3. Otherwise go to step 4.
Step 4: Return policy $\pi$.
```

Fig. 1. On-policy TD algorithm : Sarsa

3. Numerical Study

We consider two cases that have 2 and 3 products that have correlations among them in the multi-product group. One of the main goal is to observe the effect of elasticity on optimal policies, therefore we discuss and analyse the cross price elasticity and then time elasticity of the products. Time is important for the end of season products since the purchasing behavior may change in that period. Then we get insights how these observed effects affect the optimal policy[26].

We first analyse the convergence of the algorithm with the different number of iterations and decide on the iteration number for the analysis. For each case, we consider three subcases: First is the basic case with substitution and time effect are included, second is the case only time effect is included and finally the case is with only the substitution effect is included.

3.1. *Two product case*

In this case, we have 2 products with inventory levels of 4500 and 1700 units respectively. Initial prices of the products are 30 TL/unit and 20 TL/unit. Demands are obtained by the regression model that are as follows and the average demand functions used in numerical study are as below.

$$D_{1t}(p_{1t}, p_{2t}) = 950 - 19p_{1t} + 15p_{2t} - 25t$$
$$D_{2t}(p_{1t}, p_{2t}) = 700 + 10p_{1t} - 10p_{2t} - 15t$$

where D_{it} shows the demand of product i and p_{it} shows the price of product i at time t. Demand is a function of its price and the other product price and time. (-) price coefficients show the magnitude of own price elasticity of each product, and (+) price coefficients measure the magnitude of cross-price elasticity of each product. For example, the (-) coefficients are -19 and -10 for products 1 and 2, respectively. This means that one unit increase in product 1 price decreases average demand by 19 units whereas the decrease in product B average demand is just 10 units when its price is increased by a unit. On the other hand, the (+) coefficients are 15 for product 1 and 10 for product 2. This means that when product 2 price is decreased by one unit, average decrease in product 1 demand is 15 units and when product 1 price is decreased by one unit, the demand of product 2 is decreased by 10 unit.

In each period, four prices can be applied for each product, so totally we have $4\times4=16$ actions that will be observed. We apply 3 discount rates such as 10%, 30% and 50% or we may not apply any discount, that is, the price of the product remains same. Therefore, if we observe all levels of the inventory, we will have $4500\times1700\times16=12.24\times10^7$ different states.

Due to the very big number of states, sufficient number of iterations should be done to visit each state and get accurate results. Since it takes a long time, we aggregate the states according to its inventory levels such as by 50 units or 100 units. Therefore, our state number reduces to $90\times34\times16=48.96\times10^3$ if we aggregate by 50 units and $45\times17\times16=12.24\times10^3$ if we aggregate by 100 units. The number of iterations are important for the convergence of the algorithm. In Fig. 2-5, we see the convergence of the Sarsa algorithm for 1000, 2000, 3000 and 4000 iterations respectively.

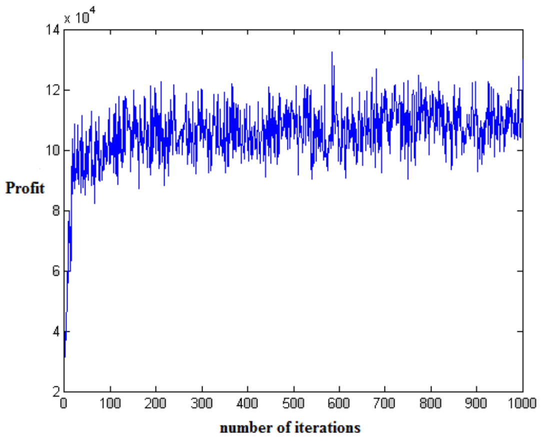

Fig. 2. Convergence for 1000 iterations

The system starts to converge after approximately 500 iterations for the four figures above. But when we compare the initial state's expected values and standard deviations in Table 1, although the expected values are approximately same, standard deviation coefficients get smaller due to the visiting number for the states increases by the iteration number. So we apply 3000 iterations for the following experiments.

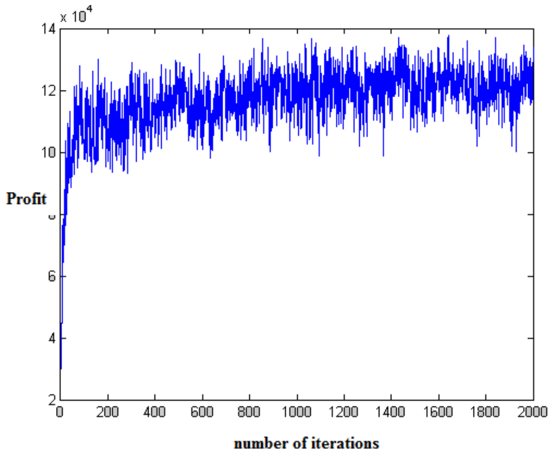

Fig. 3. Convergence for 2000 iterations

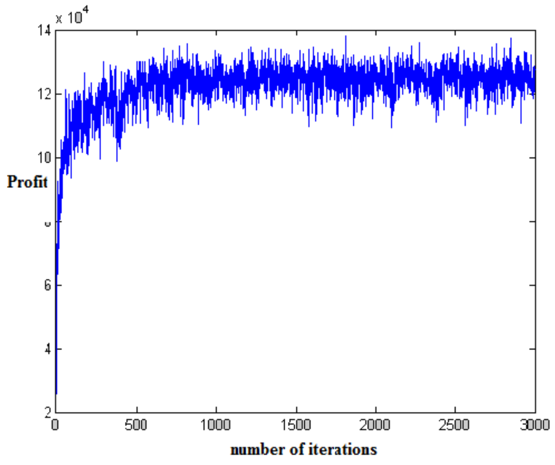

Fig. 4. Convergence for 3000 iterations

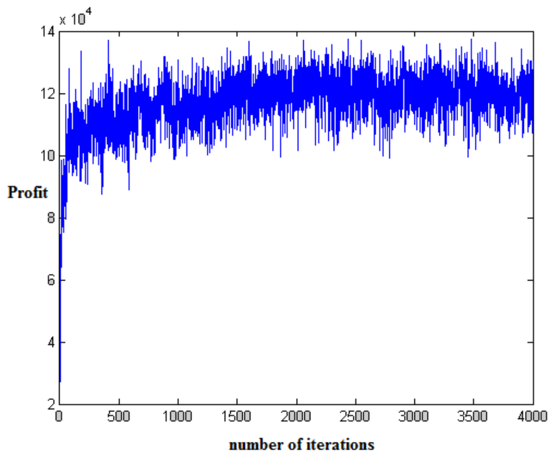

Fig. 5. Convergence for 4000 iterations

To see the differences between single aggregation and mixed aggregation, we perform the following analysis given in Table 2. Although the expected values are approximately same, the standard deviations decrease in mixed aggregation. Thereore we use mixed aggregation for the two-products numerical cases.

3.1.1. *Deterministic Model Analysis and Results*

The results of the deterministic model which is named as posterior solution is given in Table 3 for three cases that will be analysed by ADP. When we ignore the substitution effect (without susbtitution effect case) the policy is differ from the basic case (substitution and time effect included). The prices of product 2 firstly increased and then decreased more than the base case. The price of product 2 started to the week with 20 TL and stayed same until the sixth week, then decreased to 14 TL. Because when the substitution effect is ignored, demands of the products decrease. Since demand decreases, prices decrease. Then the revenue decreases to 97,030 TL from 104,877 TL which is the revenue of the basic case. Moreover, when the time effect (without time effect case) is ignored, demand increases since time has a negative effect on demand functions. Therefore the prices and demand increase, then the revenue also increases to 106,255 TL.

Table 1. Results for Convergence Analysis

				95% confidence interval	
Number of iterations	**Standard deviation**	**Expected value (TL)**	**Cv**	**Lower bound (TL)**	**Upper bound (TL)**
1000	10,218	106,502	0,0959	105,868	107,136
2000	8820	115,809	0,0762	115,422	116,196
3000	8671	117,345	0,0738	117,034	117,655
4000	7889	121,345	0,0650	121,100	121,590

Table 2. The analysis results for each aggregation level in two product case

				95% confidence interval	
Agg level	**Agg level by (units)**	**Standard deviation**	**Expect value (TL)**	**Lower bound (TL)**	**Upper bound (TL)**
single	50	8671	114,345	114,035	114,655
	100	9275	117,970	117,638	118,302
mixed	50	5718	127,660	127,455	127,865
	100	6220	117,258	117,035	117,481

Table 3. Optimal policies obtained by the posterior solution

			Week						
			1	**2**	**3**	**4**	**5**	**6-10**	**Revenue**
Basic case	Optimal policy	Prod1	30	15	15	15	15	15	104,877 TL
		Prod2	20	20	18	18	18	18	
Without substitution effect	Optimal policy	Prod1	30	15	15	15	15	15	97,030 TL
		Prod2	20	20	20	20	20	14	
Without time effect	Optimal policy	Prod1	30	15	15	15	15	15	106,255 TL
		Prod2	20	20	20	20	20	20	

3.1.2. *Stochastic Model Analysis and Results*

When we compare the posterior solution with stochastic model which is solved by Sarsa Algorithm, the following results are obtained. Again 3 subcases are examined. When demands are independent from each other, that is the substitution effect is ignored, average demand functions change as below.

$$D_{1t}(p_{1t}) = 950 - 19p_{1t} - 25t$$
$$D_{2t}(p_{1t}) = 700 - 10p_{2t} - 15t$$

Since the substitution affects the demands positively, when it is ignored, demand decrease. Moreover when we ignore the time effect, average demand functions change as follows.

$$D_{1t}(p_{1t}, p_{2t}) = 950 - 19p_{1t} + 15p_{2t}$$
$$D_{2t}(p_{1t}, p_{2t}) = 700 + 10p_{1t} - 10p_{2t}$$

Time affects the demand of the products negatively. Because consumers usually want to purchase products at the beginning of the season and this purchase behavior decreases to the end of the season. This is proved by the results of regression analysis as seen in demand functions. When we ignore the time effect, demands increase, then the system doesn't decrease the price to make more profit. The algorithms results are given in Table 4. According to the results in Table 4, similar results are obtained. In the basic case, there is no markdown for the product 1. For product 2, system starts with 18 TL and after week 2 10% discount is made. In case 2 (without substitution effect), since the sales of the products decrease, prices decrease. For the first product, 10% discount is made in week 2, 10% discount is made in week 4 and finally again 10% discount is made in week 5. Moroever the price of the second product decreases to 5.10 TL. Therefore the revenue decreases to 124,210 TL from 155,953 TL. In case without time effect, system starts to season with the highest prices, 30 TL and 20 TL, and makes only one discount during the season because it is not necessary to discount the prices due to the high demands since we have substitution effect and no time effect. Then, the revenue is the highest as in posterior solution.

3.2. *Three product case*

Similar analysis are done for the 3 product case but the run time of the algorithm increases very much. While the runtime for 2 product case takes about 30 minutes for 5000 iterations, runtime for 3 product case with the aggregation level of 50 units takes about 2.5 hours although the inventory level of each product decrease to 3000 units. Since the state space increases, we should perform many iterations to visit most of the states sufficiently.
When we run the algorithm for 5000 iterations, we get the convergence after about 500 iterations as seen in Fig. 6. Hence we apply 5000 iterations for the following analysis.

Table 4. Optimal policies obtained by Sarsa algorithm

			Week							**Revenue**	**Revenue of the posterior solution**
			1	**2**	**3**	**4**	**5**	**6**	**7-10**		
Basic Case	Optimal Policy	Prod1	30	30	30	30	30	30	30		
		Prod2	18	18	16.2	16.2	16.2	16.2	16.2	155,953TL	104,877TL
Without Substitution Effect	Optimal Policy	Prod1	30	27	27	24.3	21.87	21.87	21.87		
		Prod2	20	20	14	12.6	11.34	10.20	5.10	124,210TL	97,030TL
Without Time Effect	Optimal Policy	Prod1	30	30	27	27	27	27	27		
		Prod2	20	20	20	20	18	18	18	161,800TL	106,255TL

Each of the products have inventory levels of 3000 units. Initial prices of the products are 31 TL, 31 TL and 36 TL respectively. Expected demand functions are obtained by the regression model that are as follows:

$$D_{1t}(p_{1t}, p_{2t}, p_{3t}) = 400 - 10p_{1t} + 5p_{2t} + 11p_{3t} - 20t$$
$$D_{2t}(p_{1t}, p_{2t}, p_{3t}) = 950 - 20p_{1t} - 8p_{2t} + 17p_{3t} - 15t$$
$$D_{3t}(p_{1t}, p_{2t}, p_{3t}) = 800 - 12p_{1t} + 12p_{2t} - 9p_{3t} - 10t$$

Product 2 and 3 are substitute of product 1, product 3 is substitute of product 2 and product 2 is subtitute of product 3. In each period, again 4 prices can be applied for each product, so totally we have $4 \times 4 \times 4 = 64$ actions that will be observed. We apply 3 discount rates such as 10%, 30% and 50% or we may not apply any discount, that is, the price of the product remains same. If we observe all levels of the inventory, we will have $3000 \times 3000 \times 3000 \times 64 = 17.28 \times 10^{11}$ different states.

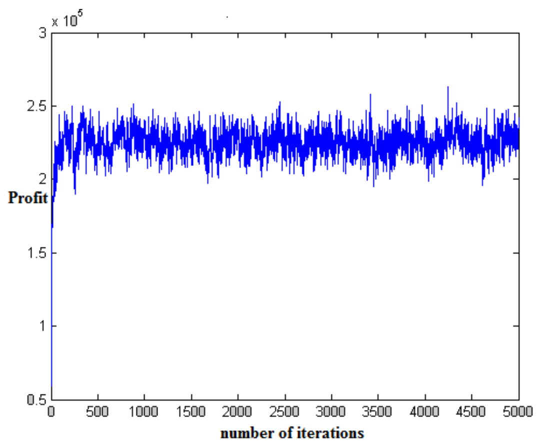

Fig. 6. Convergence for the three product case

Therefore, we apply the mixed aggregation level to take the advantage of visiting states sufficiently and get more reliable states.. The results show that when we apply the mixed aggregation level, the visiting number of states increase and the *cv* values decrease like in the 2 product case. Although the expected values of these aggregated levels are approximately same given in Table 5, the *standard error* value decreases to 6492 from 7357 for aggregation level by 100 units and this error value reduces to 7356 from 7422 for 50 units

Table 5. The analysis results for each aggregation level in three product case

				95% confidence interval	
Agg level	**Agg level by (units)**	**Standard deviation**	**Expected value (TL)**	**Lower bound (TL)**	**Upper bound (TL)**
Single	50	7422	230,907	230,701	231,113
	100	7357	224,319	224,115	224,523
Mixed	50	7356	230,391	230,187	230,595
	100	6492	218,227	218,047	218,407

aggregation. Furthermore, the 95% confidence interval is very reliable for these states. The optimal policy for mixed aggregation level with substitution case is obtained as in Table 6. The system starts with the initial prices of 27.9 TL, 31 TL and 36 TL respectively. In period 2, 10% discount decision is taken for each product, in period 3, 10% discount decision is taken for only product 2 and 3 and finally 10% discount is made in week 5. When the substitution effect is disapperared, the profit decreases to 112,890 TL from 230,200 TL since demands and also the prices decrease. Moreover, the discount decisions are given many times more than the previous case, because the system tries to clear on-hand inventories by decreasing the prices.

Table 6. Optimal policies obtained by Sarsa Algorithm

			Week							
			1	**2**	**3**	**4**	**5**	**6**	**7-10**	**Revenue**
Basic Case	Optimal policy	Prod1	27.9	25.11	25.11	22.59	22.59	22.59	22.59	230,200 TL
		Prod2	31	27.9	25.11	25.11	25.11	25.11	25.11	
		Prod3	36	32.4	29.16	29.16	26.24	26.24	26.24	
Without Substitution Effect	Optimal policy	Prod1	21.7	21.7	15.19	15.19	10.63	5.31	5.31	112,890 TL
		Prod2	21.7	21.7	21.7	21.7	19.53	13.67	13.67	
		Prod3	36	18	18	18	18	9	9	
Without Time Effect	Optimal policy	Prod1	27.9	27.9	25.11	25.11	25.11	25.11	25.11	250,081 TL
		Prod2	31	27.9	27.9	27.9	27.9	27.9	27.9	
		Prod3	36	32.4	32.4	32.4	32.4	32.4	32.4	

When time is not considered, the number of discounts decreases because we do not need to discount the prices when we have sufficient demands. The revenue levels increase to 250,081 TL from 230,200 TL.

4. Validation

To evaluate the algorithm results, we apply the algorithm to real data. One of the example data set is given in Table 7. These products are observed by linear regression models. The values of the parameters related to this linear regression model are given in Table 8 and their relations are shown in Fig. 7.

Table 7. Example multiproduct group

Product name_1	Product name_2	relation	Product code_1	Product code_2
PNT,CESTA	PNT,RELATE	-	393415	393440
PNT,CESTA	KK.TSH,B.Y.DUBAR	+	393415	400479

Table 8. Regression model of the example mutiproduct group

Model	**Constant term**	**393415_price**	**393440_price**	**400479_price**	**Sales period (week)**
393415	400	-10	5	11	-20
393440	950	-20	-8	17	-15
400479	800	-12	12	-9	-10

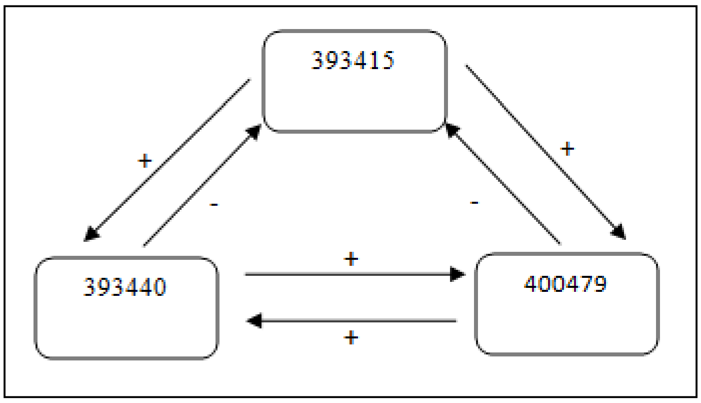

Fig. 7. Relationship between the products

The approximate optimal markdown policy of this multiproduct group is estimated by using ADP algorithm. The comparison of the policies are shown in Fig. 8-10. Retailer prices are higher than the estimated prices obtained from the ADP algorithm. Since demand changes with respect to the price, the revenue of the retailer is 440,960 TL while the estimated revenue by ADP is 516,925 TL. Therefore, ADP algorithm gives better solution than the other manually decided policies by the retailer.

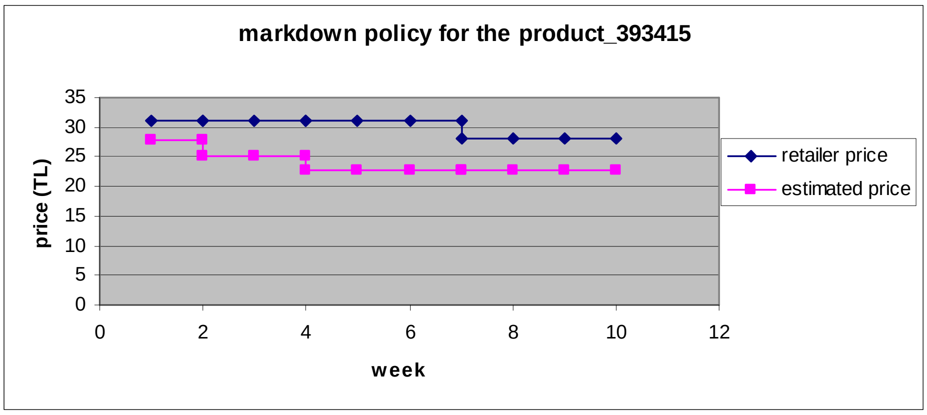

Fig. 8. Comparison between the retailer and estimated prices

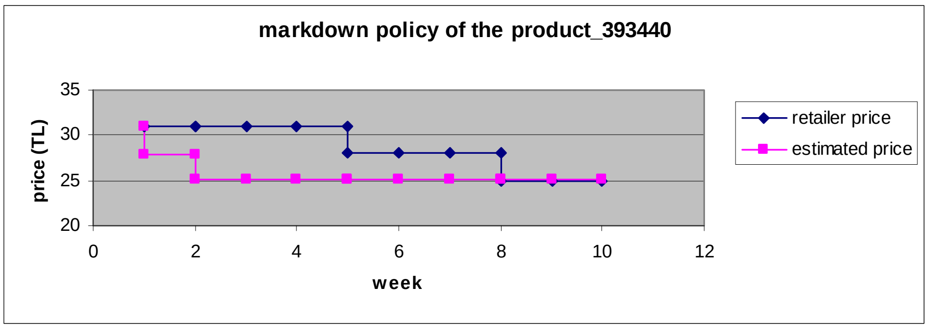

Fig. 9. Comparison between the retailer and estimated prices

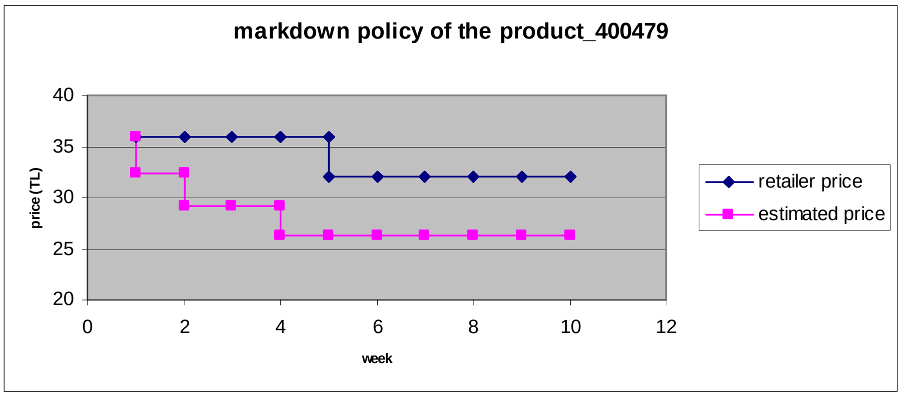

Fig. 10. Comparison between the retailer and estimated prices

5. Conclusion

We consider a markdown optimization problem faced by a retailing industry. There is no paper that investigate the substitution effects on optimal policies and timing of the markdowns if the products are correlated. This paper has demonstrated that the correlations between the products are very important to determine the optimal policy. We use a real sales data obtained from a retailing company. Since our state space is very large and multidimensional, we use Approximate Dynamic Programming. To encounter all states sufficiently, we apply aggregation technique.

We showed that substitution has positive effects on optimal policy of the products. When we consider the correlations between products, some demand of the products pass to the other products according to the changes in some attributes of the products. Then since demand increases, system decides to increase the prices of the product, therefore the profit increases too.
Another observed effect is time effect. We assume that demands decrease to the end of the season, so time has a negative effect on demands. When we don't consider the time effect, demands increase according to the basic case which has the time effect. When demands increase, again system decides to increase the prices of the product and then the profit increases. Therefore time has a negative effect on profits.

As a result, correlation between products is an important subject that should be taken into consideration. Because of substitution among products, the markdown policy of one product affects the sales of other products. Therefore, markdown policies for product groups having a significant crossprice elasticity among each other should be jointly determined. Otherwise, we can decide on wrong price policies and get smaller profits.

In this paper, we analyse only the substitution effect on markdown policies. In future research, we can observe the substitution and complementary effects together and get insights about how optimal policies will change. Moreover, another function approximation techniques such as Neural Network, SHAPE algorithm can be used and compared with the aggregation technique used in this paper.

Acknowledgement

The authors would like to thank the Turkish Scientific & Technological Research Council (TUBITAK), for supporting this research under Project No. 107M257.

References

1. Phillips, R. L., *Pricing and Revenue Optimization*, (Stanford Business Books, 2005).
2. Reda, S., Despite Early Positive Results, Retailers Haven't Jumped on Analytics Bandwagon, *Stores (KhiMetrics, Inc.)* 85(3),(2003), 203-238.
3. Lazear, E,P., Retail pricing and clearance sales, *The American Economic Review* 76 (1), (1986), 14-32.
4. Pashigian, B.P., Demand uncertainty and sales; A stıdy of markdown pricing, *American Economy Review* 78 (5), (1988), 936-953.
5. Rajan, A. and Rakesh, R. S., Dynamic pricing and ordering decisions by a monopolist, *Management Sci.* 38, (1992), 240–262.
6. Smith, S.A. and Achabal, D.D., Clearance Pricing and Inventory Policies for Retail Chains, *Management Sci.* 44 (3), (1998), 285-300.
7. Gallego, G. and van Ryzin, G., Optimal dynamic pricing of inventories with stochastic demand over finite horizons, *Management Sci.* 40, (1994), 999-1020.
8. Feng, Y. and G. Gallego, Optimal starting times for end-of-season sales and optimal stopping times for promotional fares, *Management Sci.* 41, (1995), 1371–1391.
9. Feng, Y. and Xiao, B., Integration of pricing and capacity allocation for perishable products,

European Journal of Operational Research 168, (2006), 17-34.

10. Bitran, G., Caldentey, R. and Mondschein, S.V., Coordinating Clearance Markdown Sales of Seasonal Products in Retail Chains, *Operations Research* 46, (1998), 609–624.
11. Mantrala, M. K. and Rao, S., A Decision-Support System that Helps Retailers Decide Order Quantities and Markdowns for Fashion Goods, *Interfaces* 31, (2001), 146-165.
12. Su, X., Intertemporal pricing with strategic customer behavior, *Management Sci.* 53 (5), (2007), 726-741.
13. Reiner, G. and Natter, M., An encompassing view on markdown pricing strategies: an analysis of the Austrian mobile phone market, *OR Spectrum* 29, (2007), 173- 192.
14. Elmaghraby, W. and Keskinocak, P., Dynamic Pricing in the Presence of Inventory Considerations: Research Overview, Current Practices, and Future Directions, *Management Sci.* 49 (10), (2003), 1287-1309.
15. Kök, A.G. and Fischer, M.L., Demand Estimation and Assortment Optimization Under Substitution: Methodology and Application, *Operations Research* 55 (6), (2007), 1001-1021.
16. Shen, Z.M. and, Su, X., Customer Behavior Modeling in Revenue Management and Auctions: A review and new research oppurtunities, *Production and Operations Management* 16 (6), (2007), 713-728.
17. Jain, D. C., Vilcassim, N. J. and Chintagunta, P. K., A Random Coefficients Logit Brand Choice Model Applied to Panel Data", *Journal of Business and Economic Statistics* 12, (1994), 317-328.
18. Cooper, L.G. and Nakanishi, M., *Market-share analysis: Evaluating competitive marketing effectiveness*, (Kluwer Academic Publishers, 1988).
19. Train, K, *Discrete Choice Methods*, (MIT Press, London, 2002).
20. Powell, W.B., *Approximate Dynamic Programming: Solving the curses of dimensionality*, (John Wiley & Sons, Inc. 2007).
21. Kaelbling, L.P., Littman M.L. and Moore, A.W., Reinforcement Learning: A review, Journal of Artificial Intelligence *Research* 4, (1996), 237-285.
22. Moriarty, D.E., Schultz, A.C. and Grefenstette, J.J., Evolutionary algorithms for reinforcement learning, *Journal of Artificial Intelligence Research* 11, (1999), 241-276.
23. Whatkins, C.J.C.H., Learning from Delayed Rewards, *PhD Thesis, (*1989).
24. Sutton, R.S. and Barto, A.G., *Reinforcement Learning: An introduction*, (The MIT Press, 1998).
25. Powell, W.B., Merging AI and OR to Solve High Dimensional Stochastic Optimization Problems Using Approximate Dynamic Programming, *Informs Journal on Computing* 22 (1), (2010), 2-17.
26. Coşgun, Ö., Kahraman, C. and Kula, U., Markdown Optimization Under Stochastic Demand, *PhD Thesis*, (2011).