A stochastic programming approach to cash management in banking

Jordi Castro
Dept. of Statistics and Operations Research
Universitat Politècnica de Catalunya
Pau Gargallo 5, 08028 Barcelona (Catalonia, Spain)
jordi.castro@upc.edu

Research Report DR 2004-14
December 2004
Revised August 2005, June 2007, October 2007

Report available from http://www-eio.upc.es/~jcastro

A stochastic programming approach to cash management in banking

Jordi Castro
Dept. of Statistics and Operations Research
Universitat Politècnica de Catalunya
Jordi Girona 1–3
08034 Barcelona, Catalonia, Spain
jordi.castro@upc.edu
http://www-eio.upc.es/~jcastro

Abstract

The treasurer of a bank is responsible for the cash management of several banking activities. In this work we focus on two of them: cash management in automatic teller machines (ATMs), and in the compensation of credit card transactions. In both cases a decision must be taken according to a future customers demand, which is uncertain. From historical data we can obtain a discrete probability distribution of this demand, which allows the application of stochastic programming techniques. We present stochastic programming models for each problem. Two short-term and one mid-term models are presented for ATMs. The short-term model with fixed costs results in an integer problem which is solved by a fast (i.e. linear running time) algorithm. The short-term model with fixed and staircase costs is solved through its MILP equivalent deterministic formulation. The mid-term model with fixed and staircase costs gives rise to a multistage stochastic problem, which is also solved by its MILP deterministic equivalent. The model for compensation of credit card transactions results in a closed form solution. The optimal solutions of those models are the best decisions to be taken by the bank, and provide the basis for a decision support system.

Keywords: stochastic programming, OR in banking, integer stochastic programming

1 Introduction

Cash—or liquidity—management is one of the main concerns of a bank. This class of problems appears in several of the usual banking activities. In this work we focus on two relevant ones: cash management in automatic teller machines

(ATMs)—which is similar to cash management in branches—, and in the compensation of credit card transactions. In both cases the bank must advance some amount of money. This amount must be determined according to a future unknown probabilistic demand. If the demand is higher or lower than the amount, the bank incurs some problem specific penalty costs. These decision problems result in nontrivial extensions of the classical newsvendor problem, which are addressed in this work through stochastic programming techniques. Good introductions to stochastic programming are Birge and Louveaux [1], Kall and Wallace [8]. Briefly, stochastic programming problems are optimization problems (either linear, integer or nonlinear) with one or several stochastic model parameters. Stochastic programming has previously been used for decision making under uncertainty, both for generic decision support methodologies [11], and in particular decision support systems for other liquidity management activities in banking [4].

Cash management in ATMs (and similarly in branches) is a well-known problem faced by any bank. The purpose is to decide the optimum amount of money that will be placed in the ATM to satisfy an uncertain demand. If the demand exceeds the amount in the ATM, the bank incurs in costs due to refilling tasks. It is worth to note that refilling is performed by external companies. Therefore, from the bank point of view, location of warehouses, costs associated to transportation, etc., are irrelevant. In practice, money transportation companies offer two policies. In the first one, the bank pays a significant fixed fee (e.g, 50€) for the refilling, independently of the amount, plus a small extra cost for each fraction of a certain amount of money transported (e.g., 0.59€ for each fraction of 10000€); in this option we basically deal with fixed costs. In the second one, the fixed fee is small (e.g., 20€) while the staircase costs are significant (e.g., 30€ for each fraction of 6000€). The above figures are close to the real ones used by some Spanish money transportation companies. It is thus possible to consider both models, one with only fixed cost—which is solved in linear time by a specialized algorithm introduced in this paper—, the other with both fixed and staircase costs. In addition, the paper analyzes two different situations: the short-term problem, with one single refilling; and the mid-term problem, whit more than one refilling, which forces a multistage stochastic problem. All the above considerations make this problem significantly different from the classical newsvendor problem. Several integrated decision support systems in the market [12, 13] implement optimization/simulation procedures for cash management in ATMs, but no details are given about the particular techniques used.

The Spanish credit card system is based on a daily compensation mechanism between banks and a central agent that collects all credit card transactions. Each bank daily maintains an account, where the central agent charges all the transactions from credit cards owned by this bank. The crucial decision for the bank is what amount to daily maintain in that account, given the uncertain overall sum to be charged. We will see that particular instances of this problem can be solved through inventory techniques with stochastic demands [14, Ch. 17]. For a general model, we must rely on other more versatile procedures, as stochastic programming.

As far as we know, stochastic programming has not previously been used for the two problems of this work, although it was in many other financial applications (few of them are described, for instance, in Dantzig and Infanger [2], Golub et al. [5], Gondzio and Kouwenberg [6], Kallberg et al. [7], and many more can be found in the books Ziemba and Mulvey [17], Zenios and Ziemba [16], and the surveys Kouwenberg and Zenios [9], Yu et al. [15] and the more than 150 references therein). The stochastic programming approach used in this work is clear, easy to implement, very efficient, and provides the optimal solution according to the future possible set of scenarios. The application of this procedure may improve current techniques used by banks, which are based on simulation, experience and trial and error. Even with the current low interest rates, which anyway have been successively increasing in the last months in the European Union, a better solution to the problem is relevant. For instance, the financial group "La Caixa" (one of the biggest of Spain, located in Barcelona) owns near 7000 ATM's and 5000 branches. Assuming that an average daily amount of l € is currently disposed at each ATM, and interest rates of $r\%$, a solution that reduces the daily amount per ATM in just a $p\%$ would result in an overall benefit of $0.7 \cdot r \cdot l \cdot p$ € per year. Using realistic values $l = 100000$, $r = 3$, an improvement of $p = 5$ results in 1.05 million € per year.

The paper is structured as follows. Section 2 outlines the general formulation of the stochastic optimization problems used in this work. Section 3 describes and solves the stochastic models for the cash management problem in ATMs. Section 4 presents and solves the stochastic formulation of the cash management problem in the compensation of credit card transactions.

2 The stochastic programming problem

In the two-stage stochastic programming problem a set of first-stage decisions $x \in \mathbb{R}^{n_1}$ must be taken before the values of some stochastic parameters, dependent of random variable ξ, are known. The cost of first-stage decisions is represented by $c(x)$. Once ξ is known, we must adjust another set of second-stage variables $y \in \mathbb{R}^{n_2}$ to minimize the costs we incurred by our choice of x and the particular value of ξ; the function of second-stage costs is $Q(x, \xi)$. We look for the decisions x that, in average, according to the distribution of ξ, provide the minimum costs. The general model (with fixed recourse) is

$$\begin{array}{rl} \min\limits_{x} & c(x) + \mathcal{Q}(x) \\ \text{subject to} & x \in X \subset \mathbb{R}^{n_1}, \end{array} \tag{1}$$

where

$$\mathcal{Q}(x) = E_\xi[Q(x, \xi)] \quad \text{and} \quad \begin{array}{rl} Q(x, \xi) = \min\limits_{y} & q(y; \xi) \\ \text{subject to} & Wy = h(\xi) - T(\xi)x \\ & y \in Y \subset \mathbb{R}^{n_2}. \end{array} \tag{2}$$

$c(x)$ and $x \in X$ in (1) are usually a linear function cx and a convex polyhedron defined by $X = \{x : Ax = b, x \geq 0\}$, respectively. Integrality constraints $x \in Z$,

either for a subset or for all the first-stage variables, may also appear in (1). $\mathcal{Q}(x)$, known as the recourse function, is the future average cost of our first-stage decisions x, for all scenario (i.e., for all realization of ξ). $q(y;\xi)$ and $y \in Y$ in (2) are usually a linear function $q(\xi)y$ and nonnegativity constraints $y \geq 0$, respectively. Integrality constraints $y \in Z$ can also appear in $y \in Y$, either for a subset or for all the second-stage variables; indeed we will need them in the cash management problem for ATMs. The stochastic parameters of this general formulation are $q(\xi)$, $h(\xi)$ and $T(\xi)$. In the models of next sections only $h(\xi)$ is stochastic, and is defined as $h(\xi) = \xi$, ξ being the customers demand of money, either from the ATM or through the credit card.

For some particular problems we can obtain a closed form solution for $Q(x,\xi) = q(y^*;\xi)$, y^* being the optimum second-stage decisions. In these cases we may be able to "compute" $\mathcal{Q}(x) = E_\xi[Q(x,\xi)]$, either evaluating the expectation (e.g., for discrete distributions or some low dimensional random variable) or approximating it (e.g., for multidimensional continuous variables) (see [1, Ch.9] for details). This allows the solution of (1) only in terms of the first-stage decisions. We will apply this strategy in the models of Subsection 3.1 and Section 4.

In general, however, no closed form exists for $Q(x,\xi)$ and we are forced to solve the extensive form or deterministic equivalent of the stochastic problem. For this purpose we consider ξ is a discrete random variable of s values $\xi_1, \ldots, \xi_s$ with probabilities $p_1, \ldots, p_s$. Each particular value $\xi_i, i = 1, \ldots, s$ is usually known as a scenario. Replicating for each scenario the second-stage variables (i.e, $y_i, i = 1, \ldots, s$) and constraints, and combining problems (1) and (2), we obtain the following (probably large) problem:

$$\begin{array}{rll} \min\limits_{x,y_i} & c(x) + \sum\limits_{i=1}^{s} p_i q(y_i;\xi_i) & \\ \text{subject to} & x \in X & \\ & \left.\begin{array}{l} Wy_i = h(\xi_i) - T(\xi_i)x \\ y_i \in Y \end{array}\right\} & i = 1, \ldots, s. \end{array} \tag{3}$$

Problem (3) can be solved with standard linear, integer or nonlinear programming algorithms if it is of moderate size; otherwise we need to apply specialized procedures that exploit the particular problem structure [10, 1, Ch. 5–8]. The model of Subsection 3.2 will be solved by a state-of-the-art MILP package.

Multistage stochastic problems generalize the above two-stage framework, involving a sequence of observation-decision events. For k stages, they can be formulated as

$$\begin{array}{rl} \min\limits_{x_i} & c_1(x_1) + E_{\xi^2}\left[\min c_2(x_2(\xi^2);\xi^2) + \cdots + E_{\xi^k}\left[\min c_k(x_k(\xi^k);\xi^k)\right]\cdots\right] \\ \text{subject to} & W_1 x_1 = h_1 \\ & W_2 x_2(\xi^2) + T_1(\xi^2)x_1 = h_2(\xi^2) \\ & W_i x_i(\xi^i) + T_{i-1}(\xi^i)x_{i-1}(\xi^{i-1}) = h_i(\xi^i) \quad i = 3, \ldots, k \\ & x_1 \geq 0, x_i(\xi^i) \geq 0 \quad i = 2, \ldots, k, \end{array} \tag{4}$$

where $W_i, i = 1, \ldots, k$ are known $m_i \times n_i$ matrices, h_1 is a known vector in $\mathbb{R}^{m_1}$, $h_i, i = 2, \ldots, m_i$ are random vectors in $\mathbb{R}^{m_i}$, $T_i, i = 1, \ldots, k-1$ are $m_i \times (n_i - 1)$ random matrices, $x_i \in \mathbb{R}^{n_i}, i = 1, \ldots, k$ is the vector of decisions at stage i (such that $x_i, i > 1$, depends of previous random events), and ξ^i means the history of random events up to stage i. In that formulation we have implicitly restricted our decisions to depend on past data, and they must be the same at stage i for scenarios with a common history up to stage $i-1$. In a deterministic equivalent formulation we must include explicit nonanticipativity constraints to model such behaviour (see [1] for details). The model of Subsection 3.3 will be solved by a three-stage stochastic model.

3 Cash management in ATMs (and branches)

The objective is to find the amount of cash to be disposed in an ATM for a certain period of time (e.g., one week-end, one week) using a stochastic programming approach. The same model can basically be also used for cash management in branches. We need the following information:

- Historical data $m_1, m_2, \ldots, m_t$ of the overall amount of money due to transactions in the ATM or branch, during last t periods (e.g., days). m_i can be either positive (i.e., money injected into the ATM or branch) or negative (i.e., money extracted from the ATM or branch). In practice we should differentiate depending on the type of period, e.g., workable, weekend, holidays, etc. We assume our data corresponds to the same type of periods. Banks record all the transactions performed in an ATM (and branch), and thus the historical data m_i is exhaustive. It is known that the empirical cumulative distribution function (cdf) is the maximum likelihood estimate of the real cdf [3, p. 310]. Because of the efficiency of the procedure described below, we will consider the empirical probability distribution ξ of values ξ_i and probabilities $p_i, i = 1, \ldots, s, \sum_{i=1}^{s} p_i = 1$, population data. That means that for all $i \in \{1, \ldots, t\}$ we guarantee that some scenario $j \in \{1, \ldots, s\}$ satisfies $\xi_j = m_i$. Usual techniques in stochastic programming for estimating sampling distributions, as bootstrapping, can thus be avoided.

- $l \geq 0$, $u > 0$ ($l < u$): minimum and maximum allowed amount of money in the ATM or branch. For an ATM u is its technical capacity. l can be a reserve fixed by law, mainly for branches. Values 0 and ∞ can be used if these bounds are not applicable.

- $c \geq 0$: Cost per € in the ATM or bank branch for the time horizon considered in the problem. This cost is associated with interest rates, and it may also include insurance costs for the money disposed in the ATM.

- $k \geq 0$: Fixed cost for the refilling or extraction of money if the ATM or branch gets out the allowed bounds l and u. This cost is independent of

the amount of money refilled. It is used in the models of Subsections 3.1, 3.2 and 3.3.

- $k_v \geq 0, v \geq 0$: cost for each fraction of v€ disposed in the ATM or branch. This cost defines the staircase costs of models of Subsections 3.2 and 3.3, and it is used in conjuction with the fixed cost defined by k above.

We analyze three realistic situations. The first two correspond to a short-term period of few days, e.g., one week-end, or some short period of holidays. In that case, at most one single refilling may be necessary. Depending of the policy considered by the money transportation company, we have two different models: the first one only considers fixed costs k associated to refilling, whereas the second also uses the staircase costs associated to k_v. An $O(s)$ algorithm is presented for the fixed cost problem, whereas the other is solved formulating the deterministic equivalent. The third model applies to a mid-term period, e.g., one week. Current practice in banking is to refill twice per week, since one single refill is not enough for the existing demand. This results in a multistage optimization problem, which is again solved by its deterministic equivalent.

3.1 The short-term problem with fixed costs

We define $x \in \mathbb{R}$ as the amount of money to be placed in the ATM or branch. Indeed we should impose $x \in Z$, but as shown below we will always obtain an integer solution. This is the first-stage decision to be optimized in our two-stage stochastic program. For the second-stage we define a binary variable $z \in \{0, 1\}$; z is 1 if the ATM has to be refilled, otherwise is 0. We also need auxiliary second-stage variables $y^+ \in \mathbb{R}$ and $y^- \in \mathbb{R}$. The final formulation is

$$\begin{array}{rl} \min\limits_{x} & cx + E_\xi[Q(x, \xi)] \\ \text{subject to} & l \leq x \leq u \end{array} \tag{5}$$

where

$$\begin{array}{rll} Q(x, \xi) = \min\limits_{z, y^+, y^-} & kz & \\ \text{subject to} & x + \xi + y^+ \geq l & \\ & x + \xi - y^- \leq u & \\ & y^+ \leq Mz & \\ & y^- \leq Mz & \\ & z \in \{0, 1\},\ y^+ \geq 0,\ y^- \geq 0, & \end{array} \tag{6}$$

M being a large enough value (e.g., $M = \max_i\{|\xi_i|, i = 1, \ldots, s\}$). Constraints of (6) force that the amount placed in the ATM plus the future demand is between bounds ($l \leq x + \xi \leq u$). If this can not be satisfied then we need a positive value for the auxiliary variables y^+ or y^-, and thus z must be 1, inducing refilling costs.

The deterministic equivalent of the above stochastic formulation is obtained by replicating the second-stage variables for each scenario (i.e, z_i, y_i^+, y_i^-, $i =$

$1, \ldots, s$), and the constraints where they intervene. It gives rise to the following MILP problem.

$$\begin{array}{ll} \min\limits_{x, z_i, y_i^+, y_i^-} & cx + k \sum\limits_{i=1}^{s} p_i z_i \\ \text{subject to} & l \le x \le u \\ & \left.\begin{array}{l} x + \xi_i + y_i^+ \ge l \\ x + \xi_i - y_i^- \le u \\ y_i^+ \le M z_i \\ y_i^- \le M z_i \\ z_i \in \{0,1\},\ y_i^+ \ge 0,\ y_i^- \ge 0 \end{array}\right\} i = 1, \ldots, s. \end{array} \tag{7}$$

(7) involves s binary variables. If s is large it can require too many iterations for a general MILP algorithm. However we show below that, exploiting the model structure, the equivalent stochastic problem (5)–(6) can be solved through a specialized procedure of running time $O(s)$.

3.1.1 Solving the integer stochastic problem problem in linear time

Assume the values $\xi_i, i = 1, \ldots, s$ are sorted in ascendent order (i.e., $\xi_i < \xi_{i+1}$). Note that sorting does not need to be performed in practice, since we can maintain a sorted array of ξ_i values, which is updated, e.g., daily, after a new m_i value is known. Since the probabilities $p_i, i = 1, \ldots, s$ have to be adjusted after each new value, the overall running time of this preprocessing is $O(s)$.

Then the optimal solution of (6) is

$$Q(x, \xi) = \begin{cases} k & \text{if} \quad (x + \xi < l) \text{ or } (x + \xi > u) \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Since conditions $x + \xi < l$ and $x + \xi > u$ in (8) are mutually exclusive (because $l < u$), the recourse function can be computed as follows:

$$\mathcal{Q}(x) = E_\xi[Q(x, \xi)] = \sum_{i=1}^{s} p_i Q(x, \xi_i) = \sum_{i=1}^{i_l(x)} p_i k + \sum_{i=i_u(x)}^{s} p_i k = kP(x), \tag{9}$$

where

$$P(x) = \sum_{\substack{i \in \{1, \ldots, i_l(x)\} \cup \\ \{i_u(x), \ldots, s\}}} p_i \tag{10}$$

and

$$i_l(x) = \max\{i : 1 \le i \le s, \xi_i < l - x\}, \quad i_u(x) = \min\{i : 1 \le i \le s, \xi_i > u - x\}. \tag{11}$$

Next proposition shows that indeed $i_l(x) \ne i_u(x)$ for any x.

Proposition 1 *If $i_l(x) = i$ for some $x, l \le x \le u$, then $i_u(x) \ne i$ for any $x, l \le x \le u$.*

Proof. From (11), $i_l(x) = i$ means that $x < l - \xi_i$; since $l \leq x$, then $\xi_i < 0$. Let's suppose that for some x, $i_u(x) = i$. From (11) and $u \geq x$, we have $u \geq x > u - \xi_i$, and then $\xi_i > 0$, which is a contradiction.□

The deterministic equivalent problem to be solved is

$$\min_{l \leq x \leq u} cx + kP(x). \tag{12}$$

To compute $cx + kP(x)$ we must know the $i_l(x)$ and $i_u(x)$ indexes for each interval of x values. From the definition (11), considering that $l \leq x \leq u$, and using an artificial value $\xi_{s+1} = \infty$, we have that

$$i_l(x) = i,\ 1 \leq i \leq s, \quad \text{if } x \in [\max\{l - \xi_{i+1}, l\}, \min\{l - \xi_i, u\})] = [a_i, b_i)]. \tag{13}$$

Abusing of notation, ")]" means that the right endpoint is either or not included in the interval: if $u < l - \xi_i$, then the interval is $[a_i, b_i]$, otherwise it is $[a_i, b_i)$. An interval such that $b_i < a_i$ means that $i_l(x)$ is never i, for any value x; those intervals are removed from consideration. Note that the intervals $[a_i, b_i)]$ are contiguous, i.e., $b_i = a_{i+1}$. From (11), indices $i_u(x)$ are defined similarly, using an artificial value $\xi_0 = -\infty$,:

$$i_u(x) = i,\ 1 \leq i \leq s, \quad \text{if } x \in [(\max\{u - \xi_i, l\}, \min\{u - \xi_{i-1}, u\}] = [(c_i, d_i]. \tag{14}$$

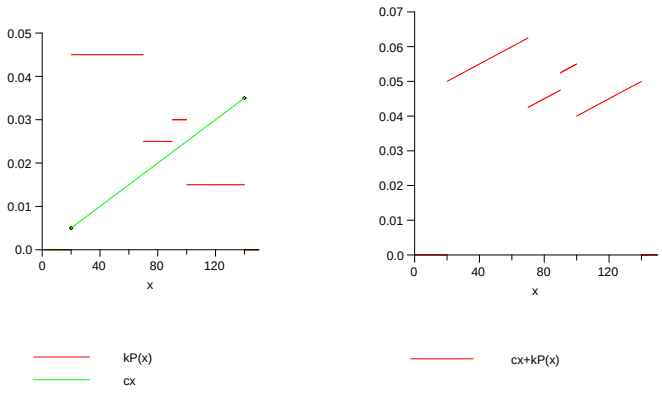
As before, "[(" means that the left endpoint is either included or not; it is if $l > u - \xi_i$, otherwise it is not. As for $i_l(x)$, an interval with $c_i < d_i$ means that $i_u(x)$ is never i, for any value x, and it is discarded. The intervals $[(c_i, d_i]$ are contiguous too. From the above intervals $[a_i, b_i)]$, $[(c_i, d_i)$, $P(x)$ can be computed as

$$P(x) = \sum_{i=1}^{i_l} p_i + \sum_{i=i_u}^{s} p_i, \quad \text{where}$$
$$i_l = \begin{cases} i & \text{if } \exists i : x \in [a_i, b_i)] \\ 0 & \text{otherwise} \end{cases}, \quad i_u = \begin{cases} i & \text{if } \exists i : x \in [(c_i, d_i] \\ s+1 & \text{otherwise} \end{cases}. \tag{15}$$

Consider for instance a small example for a short period of three days, with $s = 4$, $\xi = (-130, -80, -50, 50)$ with probabilities $p = (0.2, 0.3, 0.4, 0.1)$, $l = 20$, $u = 140$, $c = 0.00025$ (which corresponds approximately to an annual interest rate of 3%) and $k = 0.05$. ξ, l, u, k and x are expressed in 1000's €. Values k, l, u, and c are real ones, whereas ξ has been chosen for illustrative purposes (in a real situation, $s \gg 4$). The first three values of ξ represent an extraction of money from the ATM; the last scenario ξ_4 represents an injection. Using (13) and (14) we obtain the following $[a_i, b_i)]$ and $[(c_i, d_i]$ intervals:

$$i_l(x) = \begin{cases} 1 & \text{if} \quad x \in [100, 140] \\ 2 & \text{if} \quad x \in [70, 100) \\ 3 & \text{if} \quad x \in [20, 70) \\ 4 & \text{if} \quad x \in [20, -30) \quad \text{not possible} \end{cases}$$

Figure 1: a) cx and $kP(x)$ for the example. b) $cx + kP(x)$ for the example



a) b)

and

$$i_u(x) = \begin{cases} 1 & \text{if} \quad x \in (270, 140] \quad \text{not possible} \\ 2 & \text{if} \quad x \in (220, 140] \quad \text{not possible} \\ 3 & \text{if} \quad x \in (190, 140] \quad \text{not possible} \\ 4 & \text{if} \quad x \in (90, 140] \end{cases}$$

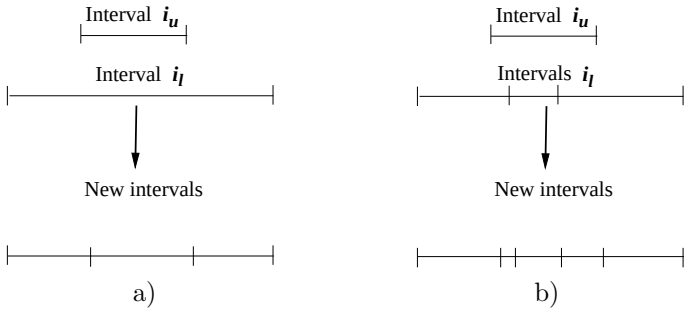
The resulting cx, $kP(x)$ and $cx + kP(x)$ functions of the example are shown in Figure 1. Note that the interval $(90, 140]$ for $i_u(x) = 4$ overlaps intervals $[70, 100)$ and $[100, 140]$ for $i_l(x) = 2$ and $i_l(x) = 1$, respectively, giving rise to the new ones $[70, 90]$, $(90, 100)$ and $[100, 140]$. As shown by next proposition the number of final intervals in $P(x)$ is at most $2s$. Abusing of notation and to simplify the cases in the proof, we will assume intervals for $i_l(x)$, $i_u(x)$ and those due to overlapping are of the form $[(a_i, b_i)]$, $[(c_i, d_i)]$ and $[(e_j, f_j)]$ (as before, "[(", ")]" means that the endpoint is either included or not).

Proposition 2 *Let* $[(a_i, b_i)], i = 1, \ldots, s_l \leq s$ *be the intervals for* $i_l(x)$ *and* $[(c_i, d_i)], i = 1, \ldots, s_u \leq s$ *those for* $i_u(x)$*. The number of new intervals* $[(e_j, f_j)]$ *due to overlapping is, at most,* $2s$*.*

Proof. We assume the intervals are ordered, i.e., $a_{i+1} = b_i$, $c_{i+1} = d_i$. This assumption is guaranteed if intervals are computed by (13) and (14). The interval $[(c_i, d_i)]$ may overlap some $[(a_j, b_j)]$ intervals, resulting in new ones. There are four cases, listed below. The first two are the nontrivial ones, while the last two reduce to one of the first two cases.

a) The endpoints of $[(c_i, d_i)]$ belong to the same interval $[(a_j, b_j)]$, i.e., $a_j \leq c_i \leq d_i \leq b_j$. Figure 2.a) illustrates this situation. The resulting three intervals are $[(a_j, c_i)]$, $[(c_i, d_i)]$, and $[(d_i, b_j)]$, increasing by one the number of original two overlapping intervals.

Figure 2: Two nontrivial cases for overlapping intervals. a) Interval $[(c_i, d_i)]$ is contained in interval $[(a_j, b_j)]$; b) Interval $[(c_i, d_i)]$ starts and ends in two different intervals $[(a_j, b_j)]$ and $[(a_{j+l}, b_{j+l})]$.



b) The endpoints of $[(c_i, d_i)]$ belong to different intervals $[(a_j, b_j)]$ and $[(a_{j+l}, b_{j+l})]$, $l \geq 1$, i.e, $a_j \leq c_i \leq b_j \leq a_{j+l} \leq d_i \leq b_{j+l}$, as shown in Figure 2.b). The resulting $l + 3$ intervals are $[(a_j, c_i)]$, $[(c_i, b_j)]$, $[(a_{j+1}, b_{j+1})]$, $[(a_{j+2}, b_{j+2})]$,..., $[(a_{j+l}, d_i)]$, $[(d_i, b_{j+l})]$, increasing by one the number of original $l + 2$ overlapping intervals.

c) The left endpoint of $[(c_i, d_i)]$ is less than a_1, the leftmost point of all $[(a_i, b_i)]$ intervals, i.e., $c_i < a_1 \leq d_i$. We then create one new interval $[(c_i, a_1)]$, an apply to the remaining portion of $[(c_i, d_i)]$, namely $[(a_1, d_i)]$, one of cases a), b) or d).

d) The right endpoint of $[(c_i, d_i)]$ is greater than b_{s_l}, the rightmost point of all $[(a_i, b_i)]$ intervals, i.e., $c_i < b_{s_l} \leq d_i$. We then create one new interval $[(b_{s_l}, d_i)]$, and apply to the remaining portion of $[(c_i, d_i)]$, namely $[(c_i, b_{s_l})]$, one of cases a), b) or c).

Therefore, for any interval $[(c_i, d_i)]$ that overlaps with intervals $[(a_i, b_i)]$ one new one is created at most. Since, from Proposition 1 $s_l + s_u$, the number of intervals $[(a_i, b_i)]$ and $[(c_i, d_i)]$, is at most s, the number of new intervals created is at most $2s$. □

Since cx has always a positive slope, the minimum of $cx + P(x)$ is on the left point of one of the intervals due to overlapping. It is thus sufficient to evaluate $cx + P(x)$ in at most $2s$ points to obtain the optimal solution. It is worth to note that if $\xi_i, i = 1, \ldots, s$, l and u are integer values, then the left points of all the intervals will also be integer and the procedure will report an integer solution. Algorithm of Figure 3 shows the main steps of the procedure, which is a constructive proof of the following result:

Proposition 3 *The integer stochastic programming problem (5)–(6) for cash management in ATMs can be solved in polynomial time using algorithm of Figure 3. Moreover, the running time is $O(s)$, s being the number of scenarios.*

Proof. It is immediate from the discussion of previous paragraph that algorithm of Figure 3 solves (5)–(6). Moreover, the running time of steps 1–4 of algorithm of Figure 3 is $O(s)$, and then the overall procedure is $O(s)$. □

Figure 3: Procedure for the solution of problem (12)

Algorithm *Cash_Management_ATM*:
1 Compute intervals of x $[a_i, b_i)]$ $i = 1, \ldots s$ using (13)
2 Compute intervals of x $[(c_i, d_i]$ $i = 1, \ldots s$ using (14)
3 Obtain new intervals of x $[(e_i, f_i)]$ $i = 1, \ldots l \leq 2s$ due to overlapping
4 $i^* = \arg\min\{ce_i + kP(e_i), i = 1, \ldots, l\}$
5 **Return:** $x = e_{i^*}$
End_algorithm

Looking at Figure 1, the final number of intervals is four, and the optimal solution is $x = 100$ (100,000€ to be put in the ATM for the time period considered) —the left point of interval $[100, 140]$— with an expected cost of 0.04 (40€ for the period). Solving (7) for this example through a MILP solver the same solution is obtained.

3.2 The short-term problem with fixed and staircase costs

Extending formulation (5) we obtain the following model for this second situation:

$$\begin{array}{rl} \min\limits_{x} & cx + E_{\xi}[Q(x,\xi)] \\ \text{subject to} & l \leq x \leq u \end{array} \tag{16}$$

where

$$\begin{array}{rll} Q(x,\xi) = & \min\limits_{z,y^+,y^-,n_v} & kz + k_v n_v \\ & \text{subject to} & x + \xi + y^+ \geq l \\ & & x + \xi - y^- \leq u \\ & & y^+ \leq Mz \\ & & y^- \leq Mz \\ & & y^+ \leq n_v v \\ & & y^- \leq n_v v \\ & & z \in \{0,1\},\ n_v \in \mathbb{N},\ y^+ \geq 0,\ y^- \geq 0, \end{array} \tag{17}$$

n_v being the number of fractions of v€ disposed to or removed from the ATM.

Although the solution of (17) is given by

$$Q(x,\xi) = \begin{cases} k + k_v \left\lceil \dfrac{l-(x+\xi)}{v} \right\rceil & \text{if} \quad x+\xi < l \\ k + k_v \left\lceil \dfrac{(x+\xi)-u}{v} \right\rceil & \text{if} \quad x+\xi > u \\ 0 & \text{otherwise,} \end{cases} \tag{18}$$

the expression of $\mathcal{Q}(x)$ can not be easily computed, unlike in (9). We then have to resort to the following deterministic equivalent formulation

$$\begin{array}{ll} \min\limits_{x,z_i,y_i^+,y_i^-,n_{v_i}} & cx + k\sum\limits_{i=1}^{s} p_i(z_i + k_v n_{v_i}) \\ \text{subject to} & l \le x \le u \\ & \left.\begin{array}{l} x+\xi_i+y_i^+ \ge l \\ x+\xi_i-y_i^- \le u \\ y_i^+ \le M z_i \\ y_i^- \le M z_i \\ y_i^+ \le n_{v_i} v \\ y_i^- \le n_{v_i} v \\ z_i \in \{0,1\}, n_{v_i} \in \mathbb{N},\ y_i^+ \ge 0,\ y_i^- \ge 0 \end{array}\right\} i = 1,\ldots,s. \end{array} \tag{19}$$

Although (19) has s binary variables and s integer ones, in practice, it can be efficiently solved by state-of-the-art MILP solvers. Table 1 shows the computational results for some instances generated from the example used in Subsection 3.1. The realistic values considered, expressed in 1000's €, are $l = 20$, $u = 140$, $c = 0.00025$ (which corresponds approximately to an annual interest rate of 3%), $k = 0.02$, $k_v = 0.03$ and $v = 6$. First instance of 4 scenarios of demand of money corresponds to the example of Subsection 3.1. The scenarios for the other instances were randomly generated from the same theoretical distribution, a normal of $\mu = -65$ and $\sigma = 20$, which explains the same optimal value in all cases. We used a theoretical distribution for the demand because we have no access to such a confidential information. The runs were performed on a PC with one AMD Athlon 4400+ 64 bits dual core processor, using the AMPL modelling language and the solver CPLEX 9.1. For each intance, Table 1 shows the number of scenarios (column s), optimal first stage decision in 1000's € (x^*), number of overall simplex iterations (MIP iter.), the number of branch-and-bound nodes explored (B&B nodes) and the overall CPU time (CPU). Clearly, it is shown that, unlike the specialized algorithm of Subsection 3.1, the CPU times do not linearly increase. However, an optimal solution is provided in few seconds.

3.3 The multistage mid-term problem

For a mid-term planning (e.g. one week), models of Subsections 3.1 and 3.2 are not applicable, since they only consider a single refilling. Current technical

Table 1: Solution of (19) for pseudo-randomly generated instances

s	x^*	MIP iter.	B&B nodes	CPU
4	138	8	0	0.016
100	114	243	6	0.064
1000	114	7352	434	4.27
5000	114	35567	837	35.33

capacities of ATM's and customers demand of money force two or more refill operations, depending of the time horizon. In theory we can consider any number h of refill operations, giving raise to a $(h+1)$-stages stochastic problem. As for the model of Subsection 3.2, we will consider fixed and staircase costs for the refill operations. The general $(h+1)$-stages model can be formulated as an extension of (16)–(17):

$$\begin{aligned}
\min_{x} \quad & cx_1 + E_{\xi_2,\ldots,\xi_{h+1}}\left[\sum_{t=2}^{h} cx_t + \sum_{t=2}^{h+1}\left(kz_t + k_v(n^+_{v,t} + n^-_{v,t})\right)\right] \\
\text{subject to} \quad & \left.\begin{array}{l}
l \le x_{t-1} \le u \\
b_t = b_{t-1} + x_{t-1} + \xi_t + y^+_t - y^-_t \\
l \le b_t \le u \\
y^+_t \le M z_t \\
y^-_t \le M z_t \\
y^+_t \le n^+_{v,t} v \\
y^-_t \le n^-_{v,t} v \\
z_t \in \{0,1\},\ y^+_t \ge 0,\ y^-_t \ge 0 \\
n^+_{v,t} \in \mathbb{N},\ n^-_{v,t} \in \mathbb{N}
\end{array}\right\} t = 2,\ldots,h+1 \\
& b_1 = 0.
\end{aligned} \tag{20}$$

$\xi_t, t = 2,\ldots,h+1$ are the demands for the t-th stage. $b_t, t = 1,\ldots,h$ is the amount of money in the ATM at stage t prior to decision x_t (initially $b_1 = 0$, though any other value could be used). Unlike previous models, the amount of money refilled or extracted, y^+_t and y^-_t, is bounded by different $n^+_{v,t}, n^-_{v,t}$ variables. This allows the inclusion of both variables y^+_t and y^-_t in the money balance equations. If necessary, dependency of random variables between stages could be added to (20), i.e. $\xi_{t+1|(t,t-1,\ldots,1)}$.

The above problem can not be solved through a special algorithm, as we did in Subsection 3.1, and we are forced to use the deterministic equivalent formulation. Although in theory any number h of refill operations can be considered, we will restrict to a three-stage situation, i.e., $h = 2$. This is the current practice in Spain for weekly periods, where refill operations are scheduled by some banks for Tuesday and Friday. The deterministic equivalent of this three-stage model is given by

Table 2: Solution of (21) for pseudo-randomly generated instances

s_2	s_3	s	x^*	MIP iter.	B&B nodes	CPU
50	20	1000	133	1219	0	0.14
50	50	2500	133	2161	367	10.87
75	50	3750	121	18431	12638	316.99
100	50	5000	124	11426	6631	602.3[†]

[†] stopped by CPU time limit, with a relative optimality gap of 0.006

$$
\begin{aligned}
\min_{x} \quad & \sum_{i=1}^{s} p_i \left(\sum_{t=2}^{3} \left(c x_{t-1_i} + k z_{t_i} + k_v (n^+_{v,t_i} + n^-_{v,t_i}) \right) \right) \\
\text{subject to} \quad & \\
& \left.
\begin{array}{l}
l \le x_{t-1_i} \le u \\
b_{t_i} = b_{t-1_i} + x_{t-1_i} + \xi_{t_i} + y^+_{t_i} - y^-_{t_i} \\
l \le b_{t_i} \le u \\
y^+_{t_i} \le M z_{t_i} \\
y^-_{t_i} \le M z_{t_i} \\
y^+_{t_i} \le n^+_{v,t_i} v \\
y^-_{t_i} \le n^-_{v,t_i} v \\
z_{t_i} \in \{0,1\},\ y^+_{t_i} \ge 0,\ y^-_{t_i} \ge 0 \\
n^+_{v,t_i} \in \mathbb{N},\ n^-_{v,t_i} \in \mathbb{N}
\end{array}
\right\} \quad \begin{array}{l} t = 2,3 \\ i = 1,\ldots,s \end{array} \\
& b_{1_i} = 0 \quad i = 1,\ldots,s \\
& \text{additional nonanticipativity constraints.}
\end{aligned}
\tag{21}
$$

As usual, nonanticipativity constraints force same decisions at stage t for scenarios with a common history up to stage $t-1$ (e.g., for x_1 they could be $x_{1_1} = x_{1_j}$ for all $s \ge j > 1$). Problem (21) has $2s$ binary, $4s$ integer and $8s$ continuous variables, such that $s = \sum_{i=1}^{s_2} s_{3_i}$, s_2 and s_{3_i} being the different values of ξ_2 and $\xi_{3|2_i}$.

Table 2 shows the computational results for some instances. As for Table 1, we used the realistic values (expressed in 1000's €) $l = 20$, $u = 140$, $c = 0.00025$ (related to an annual interest rate of 3%), $k = 0.02$, $k_v = 0.03$ and $v = 6$. The different values for ξ_2 and ξ_3 were randomly generated from the same two theoretical distributions, a normal of $\mu = -65$ and $\sigma = 20$ for ξ_2, and $\mu = -80$ and $\sigma = 15$ for ξ_3, since we have no access to real-world —thus confidential— values. The runs were performed on a PC with one AMD Athlon 4400+ 64 bits dual core processor, using the AMPL modelling language and the solver CPLEX 9.1. For each instance, Table 2 shows the number of values for ξ_2 (column s_2, which corresponds to the number of nodes of the second stage in the scenario tree), number of values for ξ_3 (column s_3), number of scenarios ($s = s_2 \cdot s_3$ in these runs), optimal first stage decision in 1000's € (x^*), number of overall simplex iterations (MIP iter.), the number of branch-and-bound nodes explored (B&B nodes) and the overall CPU time (CPU). Compared to those of

previous sections, this model is computationally more expensive. However it is still possible to obtain optimal or near-optimal solutions to small and mid-size problems in seconds or minutes of CPU.

4 Cash management in the compensation of credit card transactions

In the Spanish system there is a central agent that daily records all the banks credit card transactions and performs the payments from some accounts owned by these banks. Banks must guarantee a certain amount for such payments in their particular account. If the account becomes empty the central agent performs the remaining payments, lending money to the bank at a high interest. Otherwise, if the bank decides to maintain a large amount of money in the account, it causes an excess of overstock costs associated with interest rates. That problem can be seen as an inventory problem with stochastic demands, in particular as a newsvendor problem. However, we will consider a stochastic programming formulation, which is more general and can be easily accommodated to extra constraints and nonlinear cost functions. The information required for the model is:

- Historical data $m_1, m_2, \ldots, m_t$ of the daily credit card payments made by the bank customers during last t days. As in Section 3, from this data we generate the empirical probability distribution ξ of values ξ_i and probabilities $p_i, i = 1, \ldots, s$, which is the maximum likelihood estimate of the real cdf. We will also differentiate depending on the type of day.
- $l \geq 0$, $u > 0$ ($l < u$): minimum and maximum capacity of money in the account (set 0 and ∞ if not applicable).
- $c_1 \geq 0$: Cost per € in the account, due to interest rates.
- $c_2 > 0$: Cost per € lent by the central agent ($c_2 > c_1$).

Let x be the amount of money in the account, and let y^+ be the money lent by the central agent and y^- the residual money in the account after the payments. These are respectively the first and second-stage decisions. The formulation is

$$\begin{array}{rl} \min\limits_{x} & c_1 x + E_\xi[Q(x,\xi)] \\ \text{subject to} & l \leq x \leq u \end{array} \tag{22}$$

where

$$\begin{array}{rl} Q(x,\xi) = \min\limits_{y^+, y^-} & c_2 y^+ \\ \text{subject to} & x + y^+ - y^- = \xi \\ & y^+ \geq 0,\ y^- \geq 0. \end{array} \tag{23}$$

This is a two-stage stochastic problem with simple recourse. The solution of (23) is $y^{+^*} = \max\{\xi - x, 0\}$, $y^{-^*} = \max\{x - \xi, 0\}$. The recourse function can thus be easily computed as $\mathcal{Q}(x) = E_\xi[c_2 y^{+^*}]$. Alternatively, we can use a general expression of $\mathcal{Q}(x)$ for simple recourse problems where the only stochastic term is the right-hand side of the second-stage problem [1, p. 93]. Applying this expression of $\mathcal{Q}(x)$ in (22) we obtain

$$\begin{array}{rl} \min\limits_{x} & g(x) = c_1 x + c_2 \left(\bar{\xi} - x + F(x)x - \int_{\xi \leq x} \xi f(\xi) d\xi \right) \\ \text{subject to} & l \leq x \leq u, \end{array} \tag{24}$$

$f(\xi)$ and $\bar{\xi}$ being respectively the density function and the expected value of the daily payments. (24) is an optimization problem of one variable with simple bounds. If ξ is continuous, $g(x)$ is a convex continuous function of derivative

$$g'(x) = c_1 + c_2(F(x) - 1). \tag{25}$$

If bounds are inactive, the optimal solution satisfies $g'(x^*) = 0$. If ξ is discrete then $F(\xi)$ has discontinuities at points $\xi_i, i = 1, \ldots, s$, and $g(x)$ is convex piecewise linear and its subdifferential is

$$\partial g(x) = \left\{ \pi | c_1 + c_2(F(x) - 1) \leq \pi \leq c_1 + c_2(F^+(x) - 1) \right\}, \tag{26}$$

where $F^+(x) = \lim_{\xi \downarrow x} F(\xi)$. If bounds are inactive, a point x^* such that $0 \in \partial f(x^*)$ is optimum. Therefore, either for a continuous or discrete distribution, the solution is:

$$x^* = \begin{cases} l & \text{if } F(l) > 1 - \frac{c_1}{c_2} \geq 0 \\ u & \text{if } F(u) < 1 - \frac{c_1}{c_2} \leq 1 \\ F^{-1}(1 - \frac{c_1}{c_2}) & \text{otherwise.} \end{cases} \tag{27}$$

For a discrete distribution given by the sample $m_1, m_2, \ldots, m_t$, which is the usual case in practice, the quantile $F^{-1}(1 - \frac{c_1}{c_2})$ is computed by first sorting in ascendent order the above data, obtaining $m_{(1)}, m_{(2)}, \ldots, m_{(t)}$, and then finding the position $i = \left\lceil t(1 - \frac{c_1}{c_2}) - 1 \right\rceil$. The solution is

$$x^* = m_{(i+1)}. \tag{28}$$

If we allow interpolation then we can alternatively use

$$x^* = m_{(i)} + (t(1 - \frac{c_1}{c_2}) - i)(m_{(i+1)} - m_{(i)}). \tag{29}$$

In some numerical tests performed, the extensive form of (22–23) solved with a linear programming code provided the first solution (28). Note that, if bounds l and u are not active, the solution $F^{-1}(1 - \frac{c_1}{c_2})$ is equivalent to $F^{-1}(\frac{c_u}{c_o + c_u})$, which we would have obtained by considering an inventory problem with stochastic demands, and understock and overstock costs $c_u = c_2 - c_1$ and $c_o = c_1$, respectively [14, Ch. 17]. However, the stochastic programming approach is more versatile,

allowing lower and upper bounds l and u, and even the solution of generalizations of the problem. A realistic one would be to consider a convex or piecewise linear objective function in (23), i.e., the larger the value of y^+—money lent by the central agent—, the larger the interest rates or penalization applied. In this case we can solve the deterministic equivalent of the new stochastic model, which is a linear programming problem.

5 Conclusions

Several banking activities currently performed in some institutions simply by simulation, experience or trial and error, can effectively be optimized by stochastic programming techniques. In this work we focused on models for two particular activities: cash management in ATMs and in the compensation of credit card transactions. For basic models of both problems we provided very efficient procedures for computing the best decision ahead of an uncertain money demand. For extensions of the ATM model (i.e., short and mid-term problems with fixed and staircase costs) the MILP deterministic equivalent formulations were solved.

The models for cash management in ATMs and compensation of credit card transactions considered can be extended in several ways to fit the particular bank reality. For instance, the cash management models for ATMs could deal with situations where only some type of bills (e.g., 20€) are exhausted. In this case there is likely no need to refill the ATM, but the bank incurs costs due to dissatisfaction of customers, who are forced to get multiples of, e.g., 50€. Both problems can also be extended with nonlinear or piecewise linear cost objective functions in the second stage. For some of these extensions it can even be possible to obtain a specialized solution procedure, as those presented in this work. Otherwise, we must solve the extensive form of the problem, as we did for the two-stage and multistage cash management models with staircase costs. This could mean a larger, but hopefully yet reasonable, solution time. However, this is not a main drawback, since decisions in this context are not taken in real-time, but periodically.

6 Acknowledgments

The author is indebted to the staff of the Spanish division of a worldwide bank, who suggested and detailed the problems dealt with in this work. We have been asked to maintain confidential both the names of the persons and the bank. The author also thanks two anonymous referees whose suggestions clearly improved the presentation of the work. This work has been partially supported by the Spanish MEC grant MTM2006-05550.

References

[1] J.R. Birge, F. Louveaux. 1997. Introduction to Stochastic Programming, Springer, New York.

[2] G.B. Dantzig, G. Infanger. 1993. Multi-stage stochastic linear programs for portfolio optimization, Annals of Operations Research 45 59-76.

[3] B. Efron, R.J. Tibshirani. 1993. An Introductoin to the Bootstrap, Chapman & Hall, London.

[4] F. Gardin, R. Power, E. Martinelli. 1995. Liquidity management with fuzzy qualitative constraints, Decision Support Systems 15(2) 147–156.

[5] B. Golub, M. Holmer, R. McKendall, L. Pohlman, S.A. Zenios. 1995. Stochastic programming models for money management, European Journal of Operations Research, 85(2) 282–296.

[6] J. Gondzio, R. Kouwenberg. 2001. High performance computing for asset liability management, Operations Research 49(6) 879–891.

[7] J.G. Kallberg, R.W. White, W.T. Ziemba. 1982. Short term financial planning under uncertainty, Management Science, 28 670–682.

[8] P. Kall, S.W. Wallace. 1994. Stochastic Programming, Wiley, Chichester. Available electronically from `http://www.unizh.ch/ior/Pages/-Deutsch/Mitglieder/Kall/bib/ka-wal-94.pdf`.

[9] R. Kouwenberg, S.A. Zenios. 2006. Stochastic programming models for asset liability management, in: Stavros Zenios and Bill Ziemba (eds.), Handbook of Asset and Liability Management, series Handbooks in Finance, Elsevier, 253–303.

[10] W.K. Klein Haneveld, M.H. van der Vlerk. 1999. Stochastic integer programming: General models and algorithms, Annals of Operations Research 85 39–57.

[11] D.C. Novak, C.T. Ragsdale. 2003. A decision support methodology for stochastic multi-criteria linear programming using spreadsheets, Decision Support Systems 36(1) 99–116.

[12] Transoft Inc. 2004. OptiCa$h. `http://www.transoftinc.com/opticash.htm`.

[13] Wincor/Nixdorf. 2004. ProCash Analyzer. `http://www.-wincor-nixdorf.com/internet/us/Products/Software/Banking/-CashAnalyzer/index.html`.

[14] W.L. Winston. 1994. Operations Research. Applications and Algorithms, 3rd ed., Duxbury Press, Belmont.

[15] L.-Y. Yu, X.-D. Ji, S.Y. Wang. 2003. Stochastic programming models in financial optimization: A survey, Advanced Modeling and Optimization 5(1) 1–26.

[16] S. A. Zenios, W.T. Ziemba. 2006, 2007. Handbook of Asset and Liability Management, Vols. 1 and 2, series Handbooks in Finance, Elsevier.

[17] W.T. Ziemba, J.M. Mulvey. 1998. Worlwide asset and Liability Modelling, Cambridge University Press.