Estimating Value at Risk by the Discrete Moment Problem

Gábor Rudolf

December 21, 2004

Introduction

The purpose of this paper is to present a method for estimating the value at risk for portfolios consisting of (not necessarily independent) assets with log-normal distribution. Our approach is based on the fact that while the actual distribution of the value of the portfolio might not be known its moments can still be computed. These moments can be used to derive a set of linear constraints on the distribution of a discretization of the portfolio using the discrete moment problem. This in turn provides a possibility to derive lower and upper bounds on the value at risk using linear programming.

1 Value at Risk

Let us consider a portfolio of the following form:

$$X = \sum_{i=1}^{m} X_i e^{Y_i}$$

where X_i are nonnegative numbers and $Y^T = (Y_1 \dots Y_m)$ has normal distribution with expectation $M^T = (M_1 \dots M_m)$ and covariance matrix C.

Remark. *In a typical application we might have* $Y_i = \sqrt{T}\sigma_i Z_i(T) + TM_i$ *for some future time* T *where* $\sigma_i = \sqrt{C(i,i)}$ *and* Z_i *is a standard Brownian motion process.*

Our goal is to find the value x for which $\mathrm{Prob}(X \geq x) = p$ for some given probability p (in a typical application we might have $p = 0.95$ or $p = 0.99$). In financial mathematics this p-quantile is referred to as **value at risk**.

In this paper we describe a method which attempts to solve the inverse problem, namely for a given value x we try to compute (or provide lower and upper bounds on) $\text{Prob}(X \geq x)$. We can then either compute this probability for a sufficiently large number of values of x to describe the distribution of X or perform binary search to find the value at risk for a given probability p.

2 The Discrete Moment Problem

In this section we describe the discrete moment problem, our main tool in obtaining bounds for value at risk.
Let us consider a discrete-valued random variable X with possible values $Z_0 < Z_2 < \cdots < Z_n$ and corresponding (unknown) probabilities $p_0, p_1, \ldots p_n$. Assume that we know the first k moments of X:

$$\mu_0 = E(X^0) = 1, \mu_1 = E(X^1), \mu_2 = E(X^2), \ldots \mu_k = E(X^k)$$

Then the following linear equalities must hold:

$$\sum_{i=0}^{n} p_i Z_i^0 = \mu_0$$

$$\sum_{i=0}^{n} p_i Z_i^1 = \mu_1$$

$$\sum_{i=0}^{n} p_i Z_i^2 = \mu_2$$

$$\vdots$$

$$\sum_{i=0}^{n} p_i Z_i^k = \mu_k$$

furthermore, $p_i \geq 0$ (for all i=0,...,n)

(Note that the first condition simply states $\Sigma_{0=1}^{n} p_i = 1$).

We can now obtain lower and upper bounds for any function of the (unknown) distribution $P^T = (p_0, p_2, \ldots p_n)$ by solving the problems

$$\text{Minimize/Maximize } f(P)$$

$$\text{s.t. } AP = \mu$$

$$P \geq 0$$

where $A(i,j) = Z_j^i$ (for $i = 0, \ldots, k,\ \ j = 0, \ldots, n$) and $\mu^T = (\mu_0, \mu_1, \ldots mu_k)$.

The following special case will prove to be useful for estimating value at risk: Let x be an arbitrary value; if $Z_{r-1} < x \leq Z_r$ then we can bound the probability $Prob(X \geq x)$ by solving the LPs S^r_{min} and S^r_{max}:

$$\text{Minimize/Maximize } \sum_{i=r}^{n} p_i$$

$$\text{s.t. } AP = \mu$$

$$P \geq 0$$

3 Estimating Value at Risk

In this section we briefly outline our basic approach to estimating value at risk using the discrete moment problem.
Let X be a portfolio as in **section 1**. We begin by computing the first k moments of X to obtain $\mu^T = (\mu_0, \mu_1, \ldots, \mu_k)$. Details of this computation are described in **section 4.3**. Next we discretize X by selecting a set of possible values $Z_0 < Z_2 < \cdots < Z_n$ (the details of the discretization are described in **section 4.4**). Now we can consider the LPs S^r_{min} and S^r_{max} for $r = 0, \ldots, n$ (as defined in the previous section) and treat their optimum values as lower and upper bounds, respectively, on the probabilities $Prob(X \geq Z_r)$. Note, however, that since we use the moments of the continuous variable X in the discrete moment problem the resulting LPs might not be feasible. It is also not clear whether the optimum values will indeed be lower and upper bounds on the actual probabilities.

4 Details of the Implementation

The current implementation uses a hybrid approach: we create our data(if necessary), compute the moments, discretize X and prepare the matrix A in **MATLAB** and create an **LP** file which we solve with **CPLEX**. This became necessary since **MATLAB**'s built-in LP solver could not solve our LPs. In the following we briefly describe each of the main parts of the implementation. These parts are combined in the script *VARmain.m*.

4.1 Initialization

Initialization is performed by the script *initialize.m* which contains all the parameter values, clears all the variables to be used and sets the number format (the various parameters will be discussed in the relevant sections).

4.2 Creating Input Data

The script *rdata.m* creates random input data on which the method can be tested. It begins by creating a covariance matrix C, then creates a random expectation vector M and random coefficients $X_1, \ldots, X_m$. Finally it rescales C to assure that the variances are between $5 - 15\%$ of the corresponding expectations. Since multiplying X by a positive number essentially doesn't change the problem, the values of X_i are created between 0 and 1. The range of the M_i values is determined by the parameters *aa* and *bb*.

4.3 Computing the moments

The script *moments.m* computes the first k moments (in our current implementation k is given by the parameter kk the maximum value of which is 9) using the following formula:

Proposition.

$$\mu_k = \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} \cdots \sum_{i_k=1}^{m} X_{i_1} X_{i_2} \ldots X_{i_k} e^{\sum_{j=1}^{k} (M_{i_j} + \frac{1}{2}\sigma_{i_j}^2) + \sum_{j<l} C(i_j, i_l)}$$

Proof. Recall that if Y is a normally distributed random variable with mean M and variance σ^2 then $E(e^Y) = e^{M + \frac{1}{2}\sigma^2}$. Now we have

$$\mu_k = E(X^k) = E\left(\left(\sum_{i=1}^{m} X_i e^{Y_i}\right)^k\right) = \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} \cdots \sum_{i_k=1}^{m} X_{i_1} X_{i_2} \ldots X_{i_k} E\left(e^{\sum_{j=1}^{k} Y_{i_j}}\right)$$

where $\sum_{j=1}^{k} Y_{i_j}$ is a normally distributed random variable with mean $\sum_{j=1}^{k} M_{i_j}$ and variance $\sum_{j=1}^{k} \sigma_{i_j}^2 + 2\sum_{j<l} C(i_j, i_l)$. The proposition immediately follows. $\square$

Note that for a portfolio of size m computing the k^{th} moment takes $O(m^k)$ time, which makes using higher moments with portfolios of large size difficult.

4.4 Discretization

The script *discretize.m* discretizes X in the following manner: it selects an upper bound U for the discretization (see below), then computes the matrix A corresponding to the possible values $Z_i = U \cdot \frac{i}{n}$ (for $i = 0, \ldots, n$) where the parameter n gives the number of intervals in our discretization. Based on the numerical tests it seems that selecting a large enough U is crucial in ensuring the feasibility of the resulting LP. The following (somewhat crude) approach proved effective in practice: we choose a tolerance parameter ε (denoted by *ee* in the script); let

$$U := \sum_{i=1}^{m} X_i e^{M_i + \sigma_i \Phi^{-1}(1-\frac{\varepsilon}{m})}$$

where Φ is the c.d.f. of the standard normal distribution.

Proposition. *Prob*$(X > U) \leq \varepsilon$

Proof. By the union bound we have:

$$\text{Prob}(X > U) = Prob(\sum_{i=1}^{m} X_i e^{Y_i} > \sum_{i=1}^{m} X_i e^{M_i + \sigma_i \Phi^{-1}(1-\frac{\varepsilon}{m})}) \leq$$

$$\leq \sum_{i=1}^{m} \text{Prob}(X_i e^{Y_i} > X_i e^{M_i + \sigma_i \Phi^{-1}(1-\frac{\varepsilon}{m})}) = \sum_{i=1}^{m} \text{Prob}(Y_i > M_i + \sigma_i \Phi^{-1}(1-\frac{\varepsilon}{m})) =$$

$$= \sum_{i=1}^{m} \text{Prob}(\frac{Y_i - M_i}{\sigma_i} > \Phi^{-1}(1-\frac{\varepsilon}{m})) = m \cdot \frac{\varepsilon}{m} = \varepsilon$$

□

4.5 Solving the LPs

The script *newwrite.m* outputs a file in **LP** format describing the following (trivial) problem:

$$\text{Minimize } \sum_{i=0}^{n} p_i$$

$$\text{s.t. } AP = \mu$$

$$P \geq 0$$

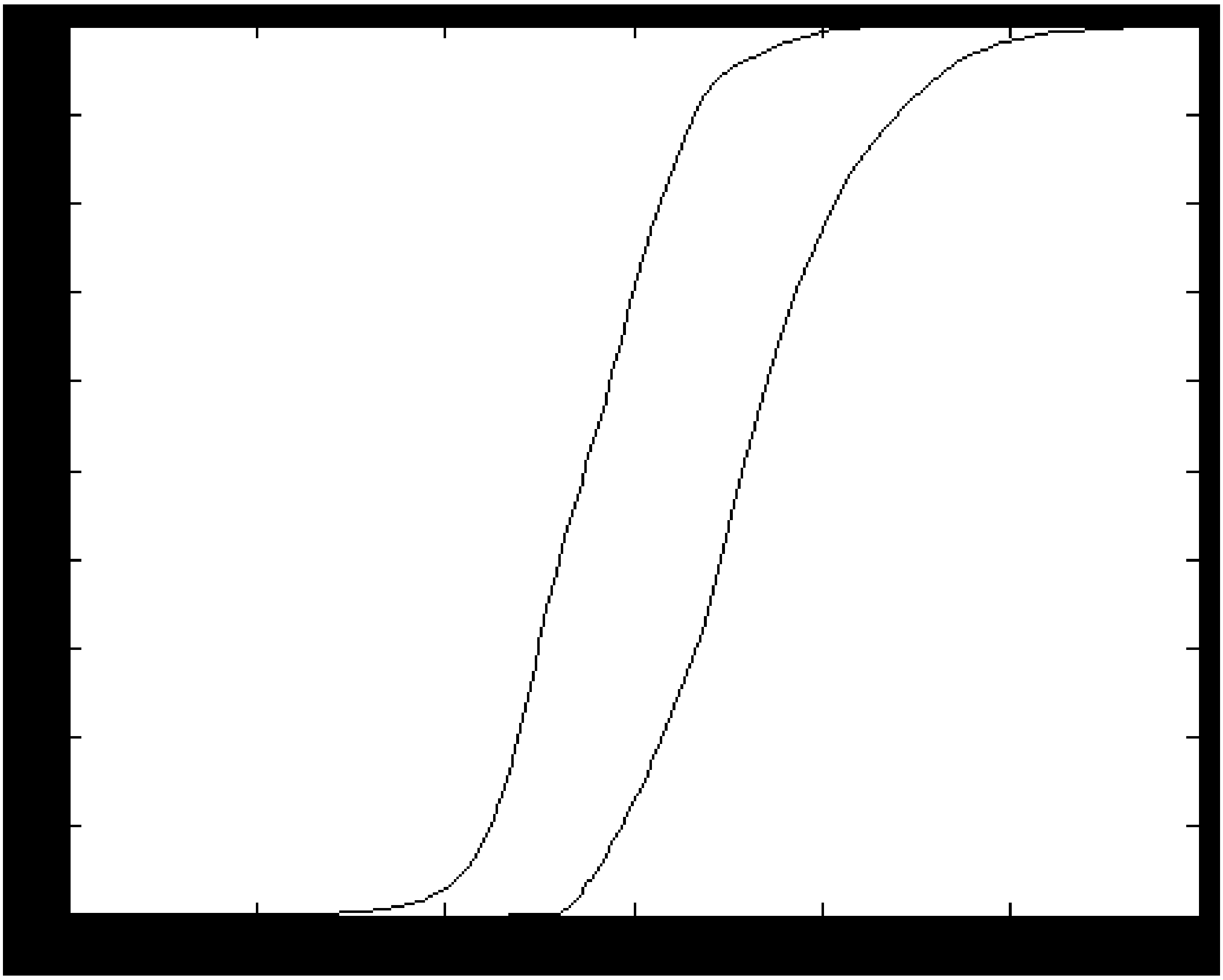
The name of the output file is given by the parameter *filename*. The objective function and objective sense can easily be changed manually. We use **CPLEX** to solve the resulting LPs.

5 Some Numerical Results

The following graph shows our lower and upper estimates of the distribution function of X using 7 moments. The randomly created data was:

$$M^T = \begin{pmatrix} 5.155117521407628e-001 & 3.339514799717586e-001 & 4.329065961067291e-001 \end{pmatrix}$$

$$C = \begin{pmatrix} 3.742356884934993e-002 & 3.357050885476298e-002 & 9.247477597603035e-003 \\ 3.357050885476298e-002 & 3.606031034053808e-002 & 2.556530841393126e-002 \\ 9.247477597603035e-003 & 2.556530841393126e-002 & 5.456203262464229e-002 \end{pmatrix}$$



The discretization parameters were $n = 10^4$, $\varepsilon = 10^{-5}$, $U = 7.739412171614991$.

6 Future Work

For the implementation it would be desirable to combine the two parts of our current hybrid approach, preferably into a **C++** code which would make manual changes unnecessary and would improve overall speed and performance. We should also examine the effects of using even higher order moments. It should be pointed out that while our implementation works well while the numbers in M (and therefore in C) are small, for greater values it runs into considerable numerical difficulties. There are several available methods to help this situation:

1. It is possible to use scaling techniques on the matrix A and the constants $X_1, \ldots, X_m$.

2. We could use some variation reduction techniques (as described in **Glasserman, Heidelberger, Shahabuddin**: *Variance Reduction Techniques for Estimating Value-at-Risk*, 2000 Management Science Vol.46, No.10.

3. Exploit the special structure of feasible bases in our LPs as suggested in **Prékopa**: *The Discrete Moment Problem and Linear Programming*, 1989 Discrete Applied Mathematics 27.

From the theoretical point of view it would be important to prove that when discretizing finely enough on a sufficiently large range the resulting LPs will be feasible and their optima can be used to bound the actual probabilities.