On the application of an Augmented Lagrangian algorithm to some portfolio problems *

E. G. Birgin† J. M. Martínez ‡

February 24, 2015§

Abstract

Algencan is a freely available piece of software that aims to solve smooth large-scale constrained optimization problems. When applied to specific problems, obtaining a good performance in terms of efficacy and efficiency may depend on careful choices of options and parameters. In the present paper the application of Algencan to four portfolio optimization problems is discussed and numerical results are presented and evaluated.

Key words: Constrained optimization, Augmented Lagrangian, Portfolios, Generalized Order-Value Optimization, Conditional Value-at-Risk.

1 Introduction

The Augmented Lagrangian (AL) method is a consolidated technique for solving constrained optimization problems. At each (outer) iteration this method solves (approximately) a subproblem where the objective function plus a term that penalizes the constraints is minimized, perhaps subject to simple constraints. The distinctive characteristic of AL with respect to penalty methods is that in AL the constraints are penalized with respect to suitable shifted values, that are updated between outer iterations. This procedure generally avoids the employment of large penalty parameters, preserving well-conditioning of the subproblems. See [7] and references therein.

In [2] and [7] a particular Augmented Lagrangian algorithm, called Algencan, is described. The basic convergence theory was given in [2] and, in the last ten years, it has been updated many times. Algencan is based on the Powell-Hestenes-Rockafellar (PHR) Augmented Lagrangian approach [12, 19, 20], as well as Lancelot [9] and other AL algorithms. The general problem to which Algencan may be applied is the following:

$$\text{Minimize } f(x) \text{ subject to } h(x) = 0,\ g(x) \leq 0,\ \text{and } \ell \leq x \leq u, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}, h : \mathbb{R}^n \to \mathbb{R}^m$, and $g : \mathbb{R}^n \to \mathbb{R}^p$ are smooth and $\ell, u \in \mathbb{R}^n$ are lower and upper bounds on the variables, respectively. At each outer iteration k, Algencan addresses the subproblem

$$\text{Minimize } f(x) + \frac{\rho_k}{2}\left[\left\|h(x) + \frac{\lambda^k}{\rho_k}\right\|_2^2 + \left\|\left(g(x) + \frac{\mu^k}{\rho_k}\right)_+\right\|_2^2\right] \text{ subject to } \ell \leq x \leq u. \tag{2}$$

*This work was supported by PRONEX-CNPq/FAPERJ E-26/111.449/2010-APQ1, FAPESP (grants 2010/10133-0, 2013/03447-6, 2013/05475-7, and 2013/07375-0), and CNPq (grants 309517/2014-1 and 303750/2014-6).

†Department of Computer Science, Institute of Mathematics and Statistics, University of São Paulo, São Paulo, SP, Brazil. e-mail: egbirgin@ime.usp.br

‡Department of Applied Mathematics, Institute of Mathematics, Statistics, and Scientific Computing, State University of Campinas, Campinas, SP, Brazil. e-mail: martinez@ime.unicamp.br

§Revision made on July 24, 2015.

In (2), $\rho_k > 0$ is a penalty parameter, $\lambda^k \in \mathbb{R}^m$ and $\mu^k \in \mathbb{R}^p_+$ are the vectors of approximate Lagrange multipliers associated with equality and inequality constraints, respectively, and v_+ denotes the vector whose components are $\max\{0, v_j\}$, where $v_1, v_2, \ldots$ are the components of v. At each iteration, λ^k, μ^k, and ρ_k are conveniently updated, according to the behavior of the method in the last subproblem.

Essentially, there are no restrictions on the number of variables and constraints of (1). This versatility has stimulated the application of Algencan to the solution of practical problems in the last 10 years [7]. Algencan is a free software whose potentialities deserve to be known by the community of engineers, economists, and other users. Its ability to handle very large problems is well known, it is relatively easy to install and use, and, so, it is interesting to report its performance in practical problems. The interesting question is whether Algencan can solve these problems in time compatible with the user's necessities. Global optimality is not guaranteed but convergence to KKT points under very weak constraint qualifications is, a property that is shared by very few nonlinear programming solvers. As far as we know, no global optimization solver is able to find global solutions of general large-scale nonlinear programming problems in reasonable computer time. It is worthwhile to mention, however, that a global version of Algencan has been described in [7], based in a method introduced in [6] and further analyzed in [8]. Unfortunately, as in every sophisticated optimization software, the performance of Algencan in practical situations is user-dependent. Although any user can solve successfully many problems using "default parameters", in many cases experienced users can obtain results astonishingly more efficiently than casual ones, both in terms of robustness and speed. Moreover, in real–life applications the formulation of the problems is not separated from the algorithmic decisions, and the applied mathematician should be involved in both processes. In this paper we will consider some industrial financial problems to which these paradigms apply.

2 Portfolio optimization problems

Let us consider a portfolio consisting of n assets. A matrix $\theta \in \mathbb{R}^{m\times n}$, obtained by simulation, reflects m scenarios for the variations of prices during some period of time. Thus, if x_j is the present value of z units of the asset j, then $\theta_{ij}x_j$ is the inflation-discounted prize of z units of the asset j one period later, under scenario i. The expected return of a portfolio $x \in \mathbb{R}^n$ is $(1/m)\sum_{i=1}^m \sum_{j=1}^n \theta_{ij}x_j = \sum_{j=1}^n \bar{\theta}_j x_j$, where $\bar{\theta}_j = (1/m)\sum_{i=1}^m \theta_{ij}$, for all $j = 1, \ldots, n$. Including the budget constraint $\sum_{j=1}^n x_j = 1$ and non-negativity constraints $x \geq 0$, the objective of an investor could be to maximize the expected return, i.e.

$$\text{Maximize } \sum_{j=1}^n \bar{\theta}_j x_j \text{ subject to } \sum_{j=1}^n x_j = 1 \text{ and } x \geq 0. \tag{3}$$

This problem is trivial. Defining $\bar{\theta}_j = \max\{\bar{\theta}_1, \ldots, \bar{\theta}_n\}$, an obvious solution is $x_j = 1$ and $x_\ell = 0$ for all $\ell \neq j$. However, this solution is not satisfactory because it does not take into account the risk of large variations of the prices of the winner asset. By this reason, the classical Markowitz model [16] minimizes the variance subject to some desired expected return and many variations of this risk-conscious approach have been presented in the literature.

Given $p \in \{0, 1, \ldots, m-1\}$ we will consider the basic problem of maximizing the average of the $m - p$ worst returns subject to constraints. This is a Generalized Order-Value Optimization (GOVO) problem in the sense of [18] and is equivalent to the problem of minimizing the Conditional Value-at-Risk (CVaR) measure using scenarios. In other words, assume that we take all the possible combinations of $m-p$ scenarios and we associate to each of these combinations the corresponding average return. Then we choose the combination of $m - p$ scenarios associated with the lowest average return. The GOVO problem that corresponds to Conditional Value-at-Risk (CVaR) consists of finding the decision for which the above mentioned lowest average return is the largest possible. Clearly, this is a generalization of the problem of maximizing the lowest return (which corresponds to the case $p = m - 1$). As a consequence

[18, 21], adding the budget constraint $\sum_{j=1}^{n} x_j = 1$, the solution comes from the Linear Programming (LP) problem

$$\underset{x,z_0,z}{\text{Minimize}}\ (m-p)z_0 + \sum_{i=1}^{m} z_i$$

subject to $x \in \Omega \equiv \{x \in \mathbb{R}^n \mid \sum_{j=1}^{n} x_j = 1 \text{ and } x \geq 0\}$, $z_i \geq 0$ and $z_i \geq -\sum_{j=1}^{n} \theta_{ij} x_j - z_0$, $i = 1, \ldots, m$. The inclusion of a linear constraint of the form $\sum_{j=1}^{n} \bar{\theta}_j x_j \geq r$ on the expected return, where $r > 0$ is a given constant, does not alter the nature of the problem, which remains to be an LP. To be more realistic we need to add other constraints to our problem. On the one hand, it makes no sense investments in some asset j smaller than a known quantity ξ_j. This means that, for all $j = 1, \ldots, n$, we must add a constraint that says that x_j does not belong to the open interval $(0, \xi_j)$. We do this by means of the constraint

$$(x_j - \xi_j/2)^2 \geq \xi_j^2/4,$$

or, equivalently,

$$\xi_j x_j - x_j^2 \leq 0, \tag{4}$$

for all $j = 1, \ldots, n$. On the other hand, the investor may wish to concentrate the investments in a small number q of assets. This leads to the constraint

$$\sum_{j=1}^{n} H(x_j) \leq q. \tag{5}$$

In (5), H is the Heaviside step function defined by $H(t) = 0$ if $t \leq 0$ and $H(t) = 1$ if $t > 0$. Therefore, the quantity $\sum_{j=1}^{n} H(x_j)$ in (5) represents the number of assets in which investments are done. Strictly speaking, this constraint does not correspond to the model (1) because the function H is discontinuous. For this reason, in practice, we will replace H with a smooth approximation as suggested in [17]. Adding the constraints (4) and (5) the problem becomes nonlinear and represents an interesting challenge for Algencan. Note that if in constraint (4) we consider $\xi_j = \xi \leq 1$ for all j then, in addition to the lower-bound on each x_j, it also implicitly imposes an upper bound $q = \lfloor 1/\xi \rfloor$ on the number of non-null assets x_j at the solution; while constraint (5) imposes this limit explicitly. In any case, if the number of non-null assets can be at most $q \geq 1$, each asset x_j has a lower bound ξ, and the sum of the x_j must be equal to one, then a solution with $\hat{q} \leq q$ non-null assets has the theoretical upper bound

$$\bar{\xi} = 1 - (\hat{q} - 1)\xi \tag{6}$$

on each x_j implicitly imposed as well. Summing up, the variants of the GOVO problem that we are going to tackle are:

Problem 1: The simplest problem commented above is the Linear Programming problem that consists of maximizing GOVO, without additional constraints. For given $\theta \in \mathbb{R}^{m \times n}$ and $p \in \{0, 1, \ldots, m-1\}$, we wish to solve the following problem:

$$\begin{array}{lll}
\text{Minimize}_{x,z_0,z} & \frac{(m-p)}{m} z_0 + \frac{1}{m} \sum_{i=1}^{m} z_i & \\
\text{subject to} & z_i \geq -\sum_{j=1}^{n} \theta_{ij} x_j - z_0, & i = 1, \ldots, m, \\
& z_i \geq 0, & i = 1, \ldots, m, \\
& \sum_{j=1}^{n} x_j = 1, & \\
& x \geq 0. &
\end{array}$$

Note that we aim to maximize the average of the $m - p$ worst returns. For example, if $p = 0.99m$ then $(m - p)/m = 0.01$ and the goal is to maximize the average of the 1% worst returns.

Problem 2: With the same data as problem 1 and, in addition, given $r > 0$, we wish to solve the following problem:

$$\begin{array}{lll}
\text{Minimize}_{x,z_0,z} & \frac{(m-p)}{m} z_0 + \frac{1}{m} \sum_{i=1}^{m} z_i & \\
\text{subject to} & z_i \geq -\sum_{j=1}^{n} \theta_{ij} x_j - z_0, & i = 1, \ldots, m, \\
& z_i \geq 0, & i = 1, \ldots, m, \\
& \sum_{j=1}^{n} x_j = 1, & \\
& \sum_{j=1}^{n} \bar{\theta}_j x_j \geq r, & \\
& x \geq 0. &
\end{array}$$

Problem 3: With the same data as problem 2 and, in addition, given $\xi_j > 0$ for all $j = 1, \ldots, n$, we wish to solve the following problem:

$$\begin{array}{lll}
\text{Minimize}_{x,z_0,z} & \frac{(m-p)}{m} z_0 + \frac{1}{m} \sum_{i=1}^{m} z_i & \\
\text{subject to} & z_i \geq -\sum_{j=1}^{n} \theta_{ij} x_j - z_0, & i = 1, \ldots, m, \\
& z_i \geq 0, & i = 1, \ldots, m, \\
& \xi_j x_j - x_j^2 \leq 0, & j = 1, \ldots, n, \\
& \sum_{j=1}^{n} x_j = 1, & \\
& \sum_{j=1}^{n} \bar{\theta}_j x_j \geq r, & \\
& x \geq 0. &
\end{array}$$

Problem 4: With the same data as problem 3 and, in addition, given an integer positive constant q, we wish to solve the following problem:

$$\begin{array}{lll}
\text{Minimize}_{x,z_0,z} & \frac{(m-p)}{m} z_0 + \frac{1}{m} \sum_{i=1}^{m} z_i & \\
\text{subject to} & z_i \geq -\sum_{j=1}^{n} \theta_{ij} x_j - z_0, & i = 1, \ldots, m, \\
& z_i \geq 0, & i = 1, \ldots, m, \\
& \xi_j x_j - x_j^2 \leq 0, & j = 1, \ldots, n, \\
& \sum_{j=1}^{n} x_j = 1, & \\
& \sum_{j=1}^{n} \bar{\theta}_j x_j \geq r, & \\
& \sum_{j=1}^{n} H(x_j) \leq q, & \\
& x \geq 0. &
\end{array}$$

The two linear programming (LP) problems 1 and 2 are being included in this study in order to put the nonlinear ones (problems 3 and 4) in perspective with respect to the better known (linear) case. We are aware that it is possible to solve million-variables LPs, of course, depending of the structure. In the nonlinear case the situation is much more erratic. Even some two-variables nonlinear problems could be unsolvable. As far as we know problems of the form 3 and 4 have not been solved, or, at least, its solution has not been commented in the Optimization literature. In any case, Algencan has two nice characteristics that make it suitable for large problems: it is matrix-free, so that no matrix (even sparse) factorizations are used at all and it handles inequality constraints without using slack variables. The use of slack variables obviously increase the dimension of the problems and it may be prohibitive in critical cases.

3 Numerical experiments and discussion

We will report the results on a case consisting of $n = 1\,000$ assets and $m = 10\,000$ scenarios. This means, that, roughly speaking, problems 1 and 2 are LP problems with $\approx 11\,000$ variables and $\approx 10\,000$ inequality constraints (excluding lower bounds). Problems 3 and 4 are Nonlinear Programming problems with the same number of variables, $\approx 10\,000$ linear inequality constraints (excluding bounds), and $\approx 1\,000$ nonlinear inequality constraints.

Since we don't want to bore the reader with meaningless and endless tables and graphics, we will concentrate our report on this case. The matrix $\theta \in \mathbb{R}^{m\times n}$ will be generated using standard simulation and expert intervention to add catastrophic events. The objective function value at the solution of problem (3) is bounded above by $\bar{\Theta} = \max\{\bar{\theta}_1, \ldots, \bar{\theta}_n\}$, where $\bar{\theta}_j = (\sum_{i=1}^{m} \theta_{ij})/m$ for all j, and bounded below by 1. The first value corresponds to the decision of investing all the budget in the asset with highest expected return and 1 is the result of investing all the budget in an asset that corresponds to inflation-corrected currency. In general, the first decision violates the risk constraints, but reporting this value gives us some feeling about the possible success of the investment.

3.1 Generation of scenarios

We considered the date range from January 14, 2013, to January 12, 2015, that, for the NYSE exchange, implies in 503 observable days. Companies were taken from the listing of the NYSE exchange available at `http://www.nasdaq.com/screening/company-list.aspx` that contains 3 300 companies (accessed on January 13, 2015). We considered the first 100 listed companies excluding those with less than 503 observations and those that contain the character ^ in its symbol. We took historical data (adjusted close value) from Yahoo! Finance using the web tool available at `http://finance.jasonstrimpel.com`, that is able to download multiple stock data series on one spreadsheet. We named this real data $\hat{D} \in \mathbb{R}^{503\times 100}$. To obtain a matrix $D \in \mathbb{R}^{503\times n}$ with $n = 1\,000$, we first computed $999 - 100$ additional columns as convex combinations of the 100 columns of $\hat{D}$. Then, we added a last column given by $(1, 1, \ldots, 1)^T \in \mathbb{R}^{503}$ corresponding to a risk-free asset. Using matrix D as a source, we computed the matrix of growing factors $G \in \mathbb{R}^{502\times n}$ whose elements g_{ij} are given by $g_{ij} = d_{i+1,j}/d_{ij}$ for $i = 1, \ldots, 502$ and $j = 1, \ldots, n$. Each row i of our scenarios matrix $\theta \in \mathbb{R}^{m\times n}$ with $m = 10\,000$ will represent a single scenario i that corresponds to applying 250 random growing factors to the unitary vector $e = (1, \ldots, 1) \in \mathbb{R}^n$ (note that 250 observable days corresponds to a calendar year). Finally, with probability of 0.1%, a random scenario was multiplied by a random number between 0.1 and 0.4 to simulate catastrophic events. For comparison purposes, it is worth noting that the asset with largest expected profit is asset $j = 21$ with $\bar{\Theta} = \bar{\theta}_{21} = 1.8601$. In this way we defined a problem with a considerable large number of assets most of them highly correlated, which could be hard for optimization purposes.

3.2 Parameters and subroutines for running Algencan

To solve problems with Algencan, we need to code subroutines to compute the objective function, the constraints, and their derivatives. Algencan deals explicitly with inequality constraints and, therefore, slack variables are not added to the model. When computing the gradient of the Lagrangian, only gradients of violated or nearly-active constraints are needed. For that reason, it is preferable, at least for the case of problems 1–4, which have a relatively large number of constraints, to provide the constraints (and their derivatives) individually. In Algencan, choices are: (a) to provide a subroutine that computes, at the same time, the objective function and *all* the constraints; or (b) to provide a subroutine that computes the objective function and other subroutine that, given j, computes the j-th constraint. By the reasons mentioned above, we chose (b) in this case. The Jacobian of the constraints of problem 1 has

the form

$$J = J(x, z_0, z) = \begin{pmatrix} -\theta & -e_m & -I \\ e_n^T & 0 & 0 \end{pmatrix} \in \mathbb{R}^{(m+1)\times(n+1+m)}, \tag{7}$$

where $e_m = (1, \ldots, 1)^T \in \mathbb{R}^m$, $e_n = (1, \ldots, 1)^T \in \mathbb{R}^n$, and $I \in \mathbb{R}^{m\times m}$ is the identity matrix. The matrix (7) has almost 90% of null elements. In the case of problems 1 and 2, the Hessian matrix of the Augmented Lagrangian function in (2) at iteration k is $\rho_k(J^T J)$. (In the case of the nonlinear problems 3 and 4 the Hessian matrix depends on the current point (x, z_0, z).) On the one hand, computing the product of this matrix times an arbitrary given vector is relatively cheap, due to the sparsity of J. On the other hand, computing and factorizing $\rho_k(J^T J)$ may be very time consuming (even storing it may be prohibited for large values of m). For that reason, it is preferable to use a truncated Newton approach instead of a Newtonian approach to solve the Augmented Lagrangian subproblems (2). By the same reason (avoiding matrix factorizations), the Algencan acceleration scheme, that tries to solve the KKT system by Newton's method, should not be activated. These requirements may be given to Algencan using the keywords TRUNCATED-NEWTON-LINE-SEARCH-INNER-SOLVER and SKIP-ACCELERATION-PROCESS. See [7] for details.

Algencan stopping criterion associated with success employs a tolerance $\varepsilon_{\text{feas}}$ for the sup-norm of the constraints and a tolerance ε_{opt} for the sup-norm of the projected gradient of the Lagrangian. Bound constraints are satisfied with zero-tolerance. We arbitrarily set $\varepsilon_{\text{feas}} = \varepsilon_{\text{opt}} = 10^{-8}$.

3.3 Results

All tests were conducted on a 3.4GHz Intel Core i5 with 8GB 1600 MHz DDR3 RAM memory and running OS X Yosemite operating system version 10.10.2 (14C109), Kernel Version: Darwin 14.1.0. Codes were compiled by the GFortran Fortran compiler of GCC (version 4.9.0) with the -O3 optimization directive enabled.

In a first experiment, we aim to solve problem 1. The problem has $n = 1\,000$ assets and $m = 10\,000$ scenarios. Therefore, it has 11 001 variables and 10 001 constraints (excluding non-negativity constraints). The values of p employed in the experiments were $9\,995, 9\,990, 9\,950, 9\,900$, and 0. We considered different possibilities for choosing the initial point x for the optimization process. Alternatives for the initial portfolio are: (a) to invest all the budget on the risk-free asset or (b) to invest all the budget on the asset with the largest expected return. Since the latter is a solution to the case $p = 0$, that was our choice. So, for the problem at hand, the initial approximation corresponds to $x_{21} = 1$ and $x_j = 0$ for all $j \neq 21$. Setting the values of z_0 and z to obtain a feasible initial guess (x, z_0, z) is very simple (for any given portfolio). Let $s_i = \sum_{j=1}^n \theta_{ij} x_j$ be the expected return under scenario i for $i = 1, \ldots, m$. We set $z_0 = s_{i_{(m-p)}}$, where $s_{i_{(m-p)}}$ is the $(m-p)$-th smallest value between $s_1, s_2, \ldots, s_m$, and then set $z_i = \max\{0, -s_i - z_0\}$ for $i = 1, \ldots, m$.

Table 1 shows the results. In the table, "Time" is the CPU time in seconds and GOVO is the expected return of the worst $m-p$ scenarios at the solution obtained. (Recall that this objective function corresponds to the simulation-based CVaR after a trivial transformation.) The return of the worst $(m-p+1)$–th scenario (which corresponds to the Value-at-Risk (VaR)) [13] is denoted by OVO(VaR) and the expected return considering all scenarios is denoted by E(R). These quantities are also reported for completeness. Recall that the quantity being maximized is the average of the w% worst possible returns, where $w = 100(m-p)/m$. In the extreme case, when $p = 0$, we have that $w = 100$ and, hence, GOVO coincides with the expected return. In this case, the expected return that is maximized and, as it should be, the obtained solution says that all the capital should be invested on the asset with the largest expected return, i.e. $j = 21$ that has $\bar{\theta}_{21} = 1.8601$. Note that, for the instance with $p = 0$, even starting from the solution, Algencan required a little less than 3 minutes to find the solution. This is due to the fact that Algencan did not receive the optimal Lagrange multipliers (dual solution) that would allow the method to verify that the initial point was a solution. Moreover, penalty-like methods

usually lose the possible feasibility of a given initial guess. Setting by hand a relatively large value of the penalty parameter ρ_1 associated with the first Augmented Lagrangian subproblem may be a remedy for this inconvenient. This strategy, known as short-cut or hot-start, may be useful in some situations but harmful in other cases. Since the instance with $p = 0$ was included for consistency checking only and starting from the solution is not the usual case, we decided to preserve the default setting of Algencan for the initial penalty parameter.

p	Time	OVO(VaR)	GOVO	E(R)
9999	4.17	0.2559	0.2559	1.3670
9995	16.26	0.4475	0.3438	1.8010
9990	57.06	0.6793	0.4851	1.8297
9950	136.38	1.0644	0.8626	1.7109
9900	43.68	1.1204	0.9823	1.7028
0	165.63	5.8097	1.8601	1.8601

Table 1: Description of the solutions to problem 1 for different settings of parameter p.

Problem 2 is similar to problem 1 but includes a (linear) constraint that adds a minimum requirement in the expected return of the portfolio. We already know that the asset with the largest expected return is asset $j = 21$ with $\bar{\theta} = 1.8601$. Requiring a fraction of this return appears to be a reasonable choice. We consider two possibilities: requiring an expected return of at least 90% of the maximum expected return and the same with 95%. Table 2 shows the results. As expected, in the cases in which the solution to problem 1 satisfies the additional constraint, this constraint has no effect and the solutions to problem 1 and 2 are the same. With the 90%-requirement, only the solution to problem 1 with $p = 9999$ does not satisfies the requirement. In this case, satisfying the new constraint implies in a reduction of GOVO from 0.2559 to 0.2412. In the same instance, with the 95%-requirement, GOVO goes down to 0.23. In all other instances (with $p \in \{9\,995, 9\,9990, 9\,9950, 9\,900, 0\}$) solutions to problem 1 and to problem 2 with the 90%-requirement in the expected return are the same. With the more restrictive constraints (expected return of at least 95% of the maximum expected return), solutions to problem 2 with $p = 9950$ and 9900 do not satisfy the requirement. In both cases the new constraint produces e reduction in the GOVO from 0.8626 to 0.8606 and from 0.9823 to 0.9771, respectively. Reductions in GOVO appear to be modest considering the increase between 5 and 6 percentage points in the expected return, completely justifying the introduction of the new constraint.

$\text{E(R)} \geq r \equiv 0.9\,\bar{\Theta} = 1.6741$				
p	Time	OVO(VaR)	GOVO	E(R)
9999	2.23	0.2412	0.2412	**1.6741**
9995	16.60	0.4475	0.3438	1.8010
9990	57.95	0.6793	0.4851	1.8297
9950	130.76	1.0644	0.8626	1.7109
9900	43.84	1.1204	0.9823	1.7028
0	166.89	5.8097	1.8601	1.8601

$\text{E(R)} \geq r \equiv 0.95\,\bar{\Theta} = 1.7671$				
p	Time	OVO(VaR)	GOVO	E(R)
9999	3.36	0.2300	0.2300	**1.7671**
9995	15.80	0.4475	0.3438	1.8010
9990	59.85	0.6793	0.4851	1.8297
9950	137.62	1.0551	0.8606	**1.7671**
9900	68.10	1.1212	0.9771	**1.7671**
0	165.81	5.8097	1.8601	1.8601

Table 2: Description of the solutions to problem 2 for different settings of parameter p. A less restrictive and a more restrictive constraint in the expected return are being considered.

Problems 1 and 2 are linear programming (LP) problems. As a basis for comparison, we run the eighteen LP problems in Tables 1 and 2 using GLPK [27], the GNU Linear Programming Kit (v4.55).

Problems were coded in the GNU MathProg language (that, at the present time, is a subset of the well-known AMPL modelling language [11]). In all cases, as expected, the same solutions were found. A few words regarding the used CPU time are in order. For solving the LP problems from Table 1 with $p > 0$, GLPK reported a CPU time of approximately 19 seconds. If the time to read the input file, to generate the model, to scale it, and to solve the problem are considered as a whole, the method takes approximately 45 seconds. For the case with $p = 0$, those times are 120 and 145 seconds approximately, respectively. For the problems in Table 2 with $p > 0$, the GLPK solving time is approximately 18 seconds, while the overall time (reading, generating, scaling, and solving) is approximately 75 seconds. For the problem in Table 2 with $p = 0$, those times are approximately 2 and 3 minutes, respectively. Summing up, considering that Algencan and the method used by GLPK (Simplex method in this case) are completely different methods in nature, that the construction of the initial point is completely different, and the many tolerances involved, it is a remarkable result that both methods perform very similar with respect to effectivity and efficiency.

In problem 3, a set of nonlinear constraints is added to problem 2. On the one hand, the new constraints inhibit to invest small percentages of the capital to any given asset. On the other hand, constraints can also be used to impose actual lower bounds on the minimum amount of assets that can be bought of a given company. In our experiments, we considered $\xi_j = \xi$ for all j. We focused on a particular instance with $p = 9\,900$ and $r = 0.95\ \bar{\Theta} = 1.7671$, and different settings for $\xi \in \{0.01, 0.02, \ldots, 0.10\}$. Table 3 shows the results. Two different initial points were considered: (a) starting from the portfolio in which the whole capital is invested on the asset with largest expected return and (b) the LP problem 2 is solved first and its solution is used as initial point. From Table 2, it can be seen that the CPU time for solving the LP problem is 68.10 seconds. This is the time that should be added to the CPU times at the right-hand-side of Table 3 in order to make them comparable with the ones in the left-hand-side of Table 3. In the table, column "# assets" reports the number of non-null x_j's at the obtained solution. The first observation is that, when starting from the LP solution, Algencan converges faster to a solution. For small values of ξ, i.e. $\xi \in \{0.01, 0.02, 0.03, 0.04\}$, Algencan found the same solution when starting from both initial points. For larger values of ξ, in all cases except $\xi = 0.08$, starting from the initial point (a) led to non-global minimizers. The optimal value of the solution found for the instance with $\xi = 0.01$ coincides with the one obtained for (the Linear Programming) problem 2 with the same values of p and r. This allows us to conclude that the global minimizers of the NLP problem 3 was found in this case. When the value of ξ increases, the feasible region is reduced and, therefore, a decrease on the optimal value of GOVO is expected. This behavior can be observed in the GOVO column in the right-hand-side of Table 3. The "slow and smooth" decrease of GOVO (and in the number of non-null x_j at the solution) when ξ increases leads us to believe that a global minimizer may be found for all the instances. The value of the largest x_j at the solutions found may be an interesting information. In the ten solutions (for varying ξ) reported on the right-hand side of Table 3 this value was always near 0.44 (ranging between 0.43 and 0.45). Although it may be seen as a relatively preference for a particular asset, this value never attained the theoretical upper bound $\bar{\xi}$ described in (6), that depends on ξ and the number of non-null assets $\hat{q}$ at the solution found (reported as "# assets" in the table) and was always larger than 0.60.

Problem 4 presents an additional constraint on the number of non-null assets. Note that the constraint added in problem 3 also imposes an upper bound on the number of non-null assets, but in a different way. The combination of both constraints roughly corresponds to a requirement like this: "I would like to invest my capital in no more than 5 assets with at least 10% of my capital in each of them". The new constraint presented in problem 4 is discontinuous and, for this reason, we replace it with the smooth approximation given by

$$H_\kappa(t) = \frac{\kappa t^2}{1 + \kappa t^2}, \tag{8}$$

where κ is a positive given constant. We perform numerical experiments for a set of instances with $p =$

9 900, $r = 0.95$ $\bar{\Theta} = 1.7671$, $\xi = 0.05$, and varying $q \in \{7, 6, 5, 4, 3\}$. For each instance, the initial point is the solution of the corresponding LP problem 2 (i.e. problem 2 with the same values for p and r). For each instance, a sequence of problems with increasing values of κ are solved. We start trying $\kappa = 1$. At the solution, the number of non-null assets is computed. If this number is larger that q then κ is increased by multiplying it by 2 and a new problem is solved. The initial point for the problem with this new κ is given by the solution to the problem with the previous value of κ. Table 4 shows the results. In many cases, the solution to a problem with a certain value of κ is also a solution to the same problem with κ replaced with 2κ. In these cases, Algencan verifies that the initial solution satisfies the stopping criterion, no iterations are done, the value of κ is doubled again and a new problem is solved. In the table, only the "subproblems" for which the initial point was not a solution are reported. A few comments are in order. The instance with $q = 7$ is reported by completeness since, as it can be seen in Table 3, the solution to the corresponding problem 3 already satisfies the new constraint on the maximum number of non-null x_j's. For the instances with $q = 6$ and $q = 5$, the same solutions that were found with $\xi = 0.06$ and $\xi = 0.07$, respectively, were recovered (note that we are considering $\xi = 0.05$). For the cases $q = 4$ and $q = 3$, different solutions were found. Note that the maximum value of κ, i.e. the number of subproblems that need to be solved, for each value of q is different. We also considered the short-cut strategy of starting from a large value of κ, like, for example, 2 048, but local non-global solutions were found for some instances with this strategy. Nevertheless, once again, the "slow and smooth" decrease of GOVO when q decreases leaves us to believe that global solutions were found in the five instances.

Same initial guess used for the LPs				
ξ	Time	OVO(VaR)	GOVO	# assets
0.01	89.06	1.1212	0.9771	8
0.02	43.06	1.1203	0.9770	8
0.03	51.18	1.1188	0.9768	8
0.04	49.18	1.1254	0.9765	7
0.05	287.54	1.1080	0.9691	4
0.06	64.25	1.1121	0.9731	6
0.07	604.67	1.1080	0.9691	4
0.08	59.12	1.1166	0.9735	5
0.09	130.62	1.0735	0.9414	3
0.10	522.74	1.1080	0.9691	4

LP's solution as initial guess				
ξ	Time	OVO(VaR)	GOVO	# assets
0.01	0.12	1.1212	0.9771	8
0.02	1.91	1.1203	0.9770	8
0.03	18.62	1.1188	0.9768	8
0.04	10.51	1.1254	0.9765	7
0.05	11.62	1.1228	**0.9757**	7
0.06	14.65	1.1237	**0.9752**	6
0.07	52.14	1.1162	**0.9735**	5
0.08	18.33	1.1166	0.9735	5
0.09	16.49	1.1158	**0.9733**	5
0.10	22.75	1.1153	**0.9730**	5

Table 3: Description of the solutions to problem 3 with $p = 9900$, $r = 0.95$ $\bar{\Theta} = 1.7671$, and different settings for parameter ξ. Two different initial guesses are considered: (a) the same initial guess used to solve problems 1 and 2 (i.e. invested the whole budget on the asset with largest expected return) and (b) LP problem 2 is solved first and its solution is used as initial guess for solving problem 3.

4 Final Remarks

Nonlinear constraints allow one to tackle many financial and economic models for which linear approaches are not practical [3]. Several free packages for nonlinear programming are publicly available and those based on the Augmented Lagrangian paradigm are generally able to attack problems with a large (and sometimes huge) number of variables and constraints. In particular, some options of Algencan handle inequality constraints without slack variables, which seems to be an advantage when many of these constraints are present in the model. Users should make a careful use of Algencan options in these cases, avoiding the employment of large-scale direct linear solvers and consequently appealing to truncated-

q	k	Time	# assets	OVO(VaR)	GOVO	E(R)
7	1	12.12	7	1.1228	0.9757	1.7671
6	1	11.82	7	1.1228	0.9757	1.7671
	2048	56.09	6	1.1239	0.9753	1.7671
5	1	11.66	7	1.1228	0.9757	1.7671
	512	112.91	6	1.1238	0.9753	1.7671
	1024	80.12	5	1.1162	0.9735	1.7671
4	1	12.12	7	1.1228	0.9757	1.7671
	256	82.89	6	1.1120	0.9720	1.7671
	512	232.13	4	1.1080	0.9691	1.7671
3	1	11.76	7	1.1228	0.9757	1.7671
	128	188.03	5	1.1022	0.9628	1.7956
	256	91.16	4	1.0845	0.9526	1.8139
	512	75.49	3	1.0825	0.9529	1.7671

Table 4: Description of the solutions to problem 4 with $p = 9900$, $r = 0.95$ $\bar{\Theta} = 1.7671$, $\xi = 0.05$, and different settings for parameter q.

Newton approaches. In this paper we explained how to proceed in those situations.

The Generalized Order-Value Optimization approach is useful to model situations that involve functions whose evaluation depends on order relations on some representation functional set. Sometimes GOVO problems may be solved by means of standard optimization solvers whereas, occasionally, GOVO needs *ad hoc* formulations [18]. In the cases considered in this paper GOVO is equivalent to approaching CVaR by means of scenarios generated by simulations. We decided to formulate our problems in terms of maximizing return instead of minimizing loss (as is standard in CVaR) because we believe that, in this way, the objective is more palpable for casual readers. Current literature includes several applications of CVaR for optimizing portfolios and other practical problems under different conditions [1, 4, 10, 14, 15, 22, 23, 24, 26]. In several of these cases it should be interesting to test the GOVO approach (see [5]).

The main drawback of nonlinear-programming formulations is that, in general, we cannot guarantee global optimality of the solution found. Algencan converges to global minimizers of the problem only under the assumption of global optimality at each subproblem. Therefore, global optimality can be established only by means of *ad hoc* analysis involving the comparison with similar problems and approximations, as we did in the problems considered in this paper.

Acknowledgements. The authors are thankful to the anonymous referees whose comments helped to improve the quality of this work.

References

[1] S. Alexander, T. F. Coleman, and Y. Li, Minimizing CVaR and VaR for a portfolio of derivatives, *Journal of Banking & Finance* 30, pp. 583–605, 2006.

[2] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, On Augmented Lagrangian methods with general lower-level constraints, *SIAM Journal on Optimization* 18, pp. 1286–1309, 2008.

[3] M. Bartholomew-Biggs, *Nonlinear Optimization with Financial Applications*, Springer Science & Business Media, 2006.

[4] M. Baweja and R. R. Saxena, Portfolio optimization with structured products under return constraint, *Yugoslav Journal of Operations Research* 25, pp. 221–232, 2015.

[5] E. G. Birgin, L. F. Bueno, N. Krejic, and J. M. Martínez, Low Order-Value approach for solving VaR-constrained optimization problems, *Journal of Global Optimization* 51, pp. 715–742, 2011.

[6] E. G. Birgin, C. A. Floudas, and J. M. Martínez, Global minimization using an Augmented Lagrangian method with variable lower-level constraints, *Mathematical Programming* 125, pp. 139–162, 2010.

[7] E. G. Birgin and J. M. Martínez, *Practical Augmented Lagrangian Methods for Constrained Optimization*, SIAM, Philadelphia, 2014.

[8] E. G. Birgin, J. M. Martínez, and L. F. Prudente, Augmented Lagrangians with possible infeasibility and finite termination for global nonlinear programming, *Journal of Global Optimization* 58, pp. 207–242, 2014.

[9] A. R. Conn, N. I. M. Gould, and Ph. L. Toint, *Lancelot: A Fortran Package for Large-Scale Nonlinear Optimization* (Release A), volume 17 of Springer Series in Computational Mathematics, Springer-Verlag, New York, 1992.

[10] Y. Elahi and I. A. Aziz, Mean-variance-CVaR model of multiportfolio optimization via linear weighted sum method, *Mathematical Problems in Engineering* 2014, Article ID 104064, 2014.

[11] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A modeling Language for Mathematical programming*, Boyd & Fraser Publishing Company, Danvers, MA, 2002.

[12] M. R. Hestenes, Multiplier and gradient methods, *Journal of Optimization Theory and Applications* 4, pp. 303–320, 1969.

[13] P. Jorion, *Value at Risk: The new benchmark for managing financial risk*, McGraw-Hill, New York, 2007.

[14] M. Kull, *Portfolio optimization for constrained shortfall risk: Implementation and IT Architecture considerations*, M.Sc. Thesis, ETH Zürich, 2014.

[15] C. Lim, H. D. Sherali, and S. Uryasev, Portfolio optimization by minimizing conditional value-at-risk via nondifferentiable optimization, *Computational Optimization and Applications* 46, pp. 391–415, 2010.

[16] H. Markowitz, Portfolio selection, *Journal of Finance* 7, pp. 77–91, 1952.

[17] J. M. Martínez, Minimization of discontinuous cost functions by smoothing, *Acta Applicandae Mathematicae* 71, pp. 245–260, 2002.

[18] J. M. Martínez, Generalized Order-Value Optimization, *TOP* 20, pp. 75–98, 2012.

[19] M. J. D. Powell, A method for nonlinear constraints in minimization problems, in *Optimization*, R. Fletcher (ed.), Academic Press, New York, NY, pp. 283–298, 1969.

[20] R. T. Rockafellar, Augmented Lagrange multiplier functions and duality in nonconvex programming, *SIAM Journal on Control and Optimization* 12, pp. 268–285, 1974.

[21] R. T. Rockafellar and S. Uryasev, Conditional value-at-risk for general loss distributions, *Journal of Banking & Finance* 26, pp. 1443–1471, 2002.

[22] A. Takeda, S. Fujiwara, and T. Kanamori, Extended robust support vector machine based on financial risk minimization, *Neural Computation* 26, pp. 2541–2569, 2014.

[23] A. Takeda and T. Kanamori, Using financial risk measures for analyzing generalization performance of machine learning models, *Neural Networks* 57, pp. 29–38, 2014.

[24] S. Uryasev, Conditional value-at-risk: optimization algorithms and applications, in *Proceedings of the IEEE/IAFE/INFORMS 2000 Conference on Computational Intelligence for Financial Engineering (CIFEr)*, pp. 49–57, 2000.

[25] Y. Wang, C. Dang, and S. Wang, Robust novelty detection via worst case CVaR minimization, *IEEE Transactions on Neural Networks and Learning Systems*, to appear (DOI: 10.1109/TNNLS.2014.2378270).

[26] S. Zhu and M. Fukushima, Worst-case conditional value-at-risk with application to robust portfolio management, *Operations Research* 57, pp. 1155–1168, 2010.

[27] GLPK – GNU Project – Free Software Foundation (FSF): `https://www.gnu.org/software/glpk/`