Operations Research Forum (2023) 4:73
https://doi.org/10.1007/s43069-023-00257-w

RESEARCH

A Low-Cost Alternating Projection Approach for a Continuous Formulation of Convex and Cardinality Constrained Optimization

N. Krejić[1] · E. H. M. Krulikovski[2] · M. Raydan[2]

Received: 21 April 2023 / Accepted: 11 September 2023 / Published online: 3 October 2023

Abstract

We consider convex constrained optimization problems that also include a cardinality constraint. In general, optimization problems with cardinality constraints are difficult mathematical programs which are usually solved by global techniques from discrete optimization. We assume that the region defined by the convex constraints can be written as the intersection of a finite collection of convex sets, such that it is easy and inexpensive to project onto each one of them (e.g., boxes, hyper-planes, or half-spaces). Taking advantage of a recently developed continuous reformulation that relaxes the cardinality constraint, we propose a specialized penalty gradient projection scheme combined with alternating projection ideas to compute a solution candidate for these problems, i.e., a local (possibly non-global) solution. To illustrate the proposed algorithm, we focus on the standard mean-variance portfolio optimization problem for which we can only invest in a preestablished limited number of assets. For these portfolio problems with cardinality constraints, we present a numerical study on a variety of data sets involving real-world capital market indices from major stock markets. In many cases, we observe that the proposed scheme converges to the global solution. On those data sets, we illustrate the practical performance of the proposed scheme to produce the effective frontiers for different values of the limited number of allowed assets.

Keywords Cardinality constraints · Portfolio optimization · Efficient frontier · Projected gradient methods · Dykstra's algorithm

AMS Subject Classification 90C30 · 65K05 · 91G10 · 91G15

This article is part of the Topical Collection on *Operations Research in Applied Energy, Environment, Climate Change & Sustainability*

Extended author information available on the last page of the article

1 Introduction

We are interested in convex constrained optimization problems with an additional cardinality constraint. In other words, we are interested in finding sparse solutions of those optimization problems, i.e., solutions with a limited number of nonzero elements, as required in many areas including image and signal processing, mathematical statistics, machine learning, portfolio optimization problems, among others. One effective way to ensure the sparsity of the obtained solution is imposing a cardinality constraint where the number of nonzero elements of the solution is bounded in advance.

To be precise, let us consider the following constrained optimization problem:

$$\min_{x} f(x) \quad \text{subject to} \quad x \in \Omega \text{ and } \|x\|_0 \leq \alpha, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable, $1 \leq \alpha < n$ is a given natural number, Ω is a convex subset of $\mathbb{R}^n$ (that will change depending on the considered application), and the L_0 (quasi) norm $\|x\|_0$ denotes the number of nonzero components of x. The sparsity constraint $\|x\|_0 \leq \alpha$ is also called the cardinality constraint. Of course, we will assume that $\alpha < n$ since otherwise the cardinality constraint could be discarded.

The main difference between problem (1) and a standard convex constrained optimization problem is that the cardinality constraint, despite of the notation, is not a norm, nor continuous neither convex. Because of the non-tractability of the so-called zero norm $\|x\|_0$, the 1-norm $\|x\|_1$ has also been frequently considered to develop good approximate algorithms. Clearly, to impose a required level of sparsity, the use of the zero norm in (1) is much more effective.

Optimization problems with cardinality constraints are (strongly) NP-hard problems [5, 16], which can be solved by global techniques from discrete or combinatorial optimization (see, e.g., [4, 14, 23]). However, in a more general setting, a continuous reformulation has been recently proposed and analyzed in [12] to deal with this difficult cardinality constraint. The main idea is to address the continuous counterpart of problem (1):

$$\begin{array}{ll} \min\limits_{x,y} & f(x) \\ \text{subject to:} & x \in \Omega, \\ & e^\top y \geq n - \alpha, \\ & x_i y_i = 0, \text{ for all } 1 \leq i \leq n, \\ & 0 \leq y_i \leq 1, \text{ for all } 1 \leq i \leq n, \end{array} \tag{2}$$

where $e \in \mathbb{R}^n$ denotes the vector of ones. We note that the last n constraints denote a simple box in the auxiliary variable vector $y \in \mathbb{R}^n$. A more difficult reformulation substitutes the simple box by a set of binary constraints given by either $y_i = 0$ or $y_i = 1$ for all i. In that case, the problem is an integer programming problem (much harder to solve) for which there are several algorithmic ideas already developed (see, e.g., [4, 5, 14, 19, 36]). In here, we will focus on the continuous formulation (2) that will play a key role in our algorithmic proposal. For additional theoretical properties that include the equivalence between the original version (1) and the continuous relaxed version (2), see [12, 25, 28, 29].

As a consequence of the so-called Hadamard constraint ($x \circ y = 0$, i.e., $x_i y_i = 0$ for all i), the formulation (2) is a nonconvex problem, even when the original cardinality constrained problem (except for the cardinality constraint of course) was convex. Thus, one can in general not expect to obtain global minima. But if one is, for example, interested in obtaining local solutions or good starting points for a global method, this continuous formulation (2) can be useful.

In this work, we will pay special attention to those problems for which the set Ω is the intersection of a finite collection of convex sets, in such a way that it is very easy to project onto each one of them. In that case, the main idea is to take advantage of the fact that two of the constraints in (2), namely, $e^\top y \geq n - \alpha$ and $0 \leq y_i \leq 1$ for all i, are also “easy-to-project" convex sets, and so an alternating projection scheme can be conveniently applied to project onto the intersection of all the involved constraints in (2), except for the Hadamard constraint. For computing a solution candidate of the continuous formulation (2), we can then use a suitable low-cost convex constrained scheme, such as gradient-type methods in which the objective function includes $f(x)$ plus a suitable penalization term that guarantees that the Hadamard constraint is also satisfied at the solution. In Section 2, we will describe and analyze a general penalty method to satisfy the Hadamard constraint that appears in the relaxed formulation (2). In Section 3, we will describe a suitable alternating projection scheme as well as a suitable low-cost gradient-type projection method that can be combined with the penalty method of Section 2. We close Section 3 showing the combined algorithm that represents the main contribution of our work. Concerning some specific applications, in Section 4, we will consider in detail the standard mean-variance limited diversified portfolio selection problem (see, e.g., [13–15, 19, 21, 23]). In Section 5, we will present a numerical study to illustrate the computational performance of the proposed scheme on a variety of data sets involving real-world capital market indices from major stock markets. For each considered data set, we will focus our attention on the efficient frontier produced for different values of the limited number of allowed assets. In Section 6, we will present some final comments and perspectives.

2 A Penalization Strategy for the Hadamard Constraint

Let us consider again the continuous formulation (2), and let us focus our attention on the Hadamard constraint $x \circ y = 0$ (i.e., $x_i y_i = 0$ for all i). This particular constraint is the only one that does not define a convex set. The others define convex sets in which it is easy to project, as discussed in the previous section. To see that the set of vectors $(x, y) \in \mathbb{R}^{2n}$ such that $x \circ y = 0$ do not form a convex set, it is enough to consider the two 2-dimensional pairs: $(x_1, y_1) = (1, 0, 0, 1)$ and $(x_2, y_2) = (0, 1, 1, 0)$. Both pairs are clearly in that set, but the convex combination: $\frac{1}{2}(x_1, y_1) + \frac{1}{2}(x_2, y_2) = \frac{1}{2}e$, which is not in that set.

A classical and straightforward approach to force the Hadamard condition at the solution, while keeping the feasible set of our problem as the intersection of a finite collection of easy convex sets, is to add a penalization term $\tau h(x,y)$ to the objective function and consider instead the following formulation:

$$\begin{array}{ll} \min\limits_{x,y} & f(x) + \tau h(x,y) \\ \text{subject to:} & x \in \Omega, \\ & e^\top y \geq n - \alpha, \\ & 0 \leq y_i \leq 1, \ \text{for all } 1 \leq i \leq n, \end{array} \tag{3}$$

where $\tau > 0$ is a penalization parameter that needs to be properly chosen, and the function $h : \mathbb{R}^{2n} \to \mathbb{R}$ is continuously differentiable and chosen to satisfy the following two properties: $h(x,y) \geq 0$ for all feasible vectors x and y, and $h(x,y) = 0$ if and only if $x \circ y = 0$. Clearly, the function $h(x, y)$ is crucial and should be conveniently chosen depending on the considered application. However, a default option that satisfies all the required properties is given by $h(x,y) = \sum_{1 \leq i \leq n} x_i^2 y_i^2$.

Applying now a penalty scheme, problem (3) can be reduced to a sequence of convex constrained problems of the following form:

$$\min_{x,y} \quad f(x) + \tau_k h(x,y), \qquad \text{subject to} \qquad (x,y) \in \widehat{\Omega}, \tag{4}$$

where $\tau_k > 0$ is the penalty parameter that increases at every k to penalize the Hadamard-constraint violation, and the closed convex set $\widehat{\Omega}$ is given by

$$\widehat{\Omega} = \{(x,y) \in \mathbb{R}^{2n} \, : \, x \in \Omega, \ \ e^\top y \geq n - \alpha, \ \ 0 \leq y_i \leq 1, \ i = 1, \ldots, n\}.$$

Under some mild assumptions and some specific choice of the sequence $\{\tau_k\}$, it can be established that the sequence of solutions of problem (4) converges to a solution of (2) (see, e.g., [22] and [30, Secc. 12.1]). Let us assume that problem (2) attains global minimizers. Since f is a continuous function, it is enough to assume that one of the closed and convex sets involved in the definition of Ω in (2) is bounded. In here, for the sake of completeness, we summarize the convergence properties of the proposed penalty scheme (4).

Theorem 1 *If for all k*, $\tau_{k+1} > \tau_k > 0$ *and* (x_k, y_k) *is a global solution of* (4), *then*

$$\begin{aligned} f(x_k) + \tau_k h(x_k, y_k) &\leq f(x_{k+1}) + \tau_{k+1} h(x_{k+1}, y_{k+1}) \\ h(x_{k+1}, y_{k+1}) &\leq h(x_k, y_k) \\ f(x_k) &\leq f(x_{k+1}). \end{aligned}$$

Moreover, if $\bar{x}$ *is a global solution of problem* (2), *then for all k*

$$f(x_k) \ \leq \ f(x_k) + \tau_k h(x_k, y_k) \ \leq \ f(\bar{x}) \ .$$

Finally, if $\tau_k \to \infty$ *and* $\{(x_k, y_k)\}$ *is the sequence of global minimizers obtained by solving* (4), *then any limit point of* $\{(x_k, y_k)\}$ *is a global minimizer of* (2).

Remark 1 In the proof of the last statement of Theorem 1 (see, e.g., [30, Secc. 12.1]), the requirement of $\tau_k \to \infty$ is used only to guarantee that the term $h(x_k, y_k) \to 0$

when $k \to \infty$, i.e., to guarantee that $x_k \circ y_k \to 0$. In order to guarantee the convergence result, what is important is that the Hadamard product itself goes to zero even if $0 < \tau_k < \infty$ for all k. This fact will play a key role in our numerical study (Section 5).

We would like to close this section with a pertinent result [28, Theorem 4] that establishes a one-to-one correspondence between minimizers of problems (1) and (2), whenever the obtained solution $\bar{x}$ satisfies the cardinality constraint with equality, i.e., $\|\bar{x}\|_0 = \alpha$.

Theorem 2 *Let* $(\bar{x}, \bar{y})$ *be a local minimizer of the relaxed problem* (2). *Then,* $\|\bar{x}\|_0 = \alpha$ *if and only if* $\bar{y}$ *is unique, that is, if there exist exactly one* $\bar{y}$ *such that* $(\bar{x}, \bar{y})$ *is a local minimizer of* (2). *In this case, the components of* $\bar{y}$ *are binary* (*i.e.,* $\bar{y}_i = 0$ *or* $\bar{y}_i = 1$ *for all* $1 \le i \le n$) *and* $\bar{x}$ *is a local minimizer of* (1).

3 Dykstra's Method and the SPG Method

For every k, a low-cost projected gradient method can be used to compute a solution candidate of the optimization problem (4). For a given vector $\tilde{x} \in \mathbb{R}^{2n}$, a convenient tool for finding the required projections onto $\widehat{\Omega}$ is Dykstra's alternating projection algorithm [11], that projects in a clever way onto the convex sets, say $\Omega_1, \dots, \Omega_p$, individually to complete a cycle which is repeated iteratively, and as any other iterative method, it can be stopped prematurely.

In Dykstra's method, it is assumed that the projections onto each of the individual sets Ω_i are relatively simple to compute, e.g., boxes, spheres, subspaces, half-spaces, and hyperplanes. The algorithm has been adapted and used for solving a huge amount of different applications and has been combined with several techniques in optimization, including outer approximation strategies for solving nonlinear constraint problems (see, e.g., [2, 17, 33]). For a review on Dykstra's method, its properties and applications, as well as many other alternating projection schemes, see, e.g., [18, 20].

Dykstra's algorithm generates two sequences: the iterates $\{x_\ell^i\}$ and the increments $\{I_\ell^i\}$. These sequences are defined by the following recursive formulae:

$$\begin{aligned} x_\ell^0 &= x_{\ell-1}^p \\ x_\ell^i &= P_{\Omega_i}(x_\ell^{i-1} - I_{\ell-1}^i) \quad i = 1, 2, \dots, p, \\ I_\ell^i &= x_\ell^i - (x_\ell^{i-1} - I_{\ell-1}^i) \quad i = 1, 2, \dots, p, \end{aligned} \tag{5}$$

for $\ell \in \mathbb{Z}^+$ with initial values $x_0^p = \tilde{x}$ and $I_0^i = 0$ for $i = 1, 2, \dots, p$.

The sequence of increments play a fundamental role in the convergence of the sequence $\{x_\ell^i\}$ to the unique optimal solution $x^* = P_{\widehat{\Omega}}(\tilde{x})$. Boyle and Dykstra [11] established the key convergence theorem associated with algorithm (5), i.e., that for any $i = 1, 2, \dots, p$ and any given $\tilde{x}$, the sequence $\{x_\ell^i\}$ generated by (5) converges to $x^* = P_{\widehat{\Omega}}(\tilde{x})$ (i.e., $\|x_\ell^i - x^*\| \to 0$ as $\ell \to \infty$). Concerning the rate of convergence, it is well-known that Dykstra's algorithm exhibits a linear rate of convergence in the polyhedral case [18, 20], which is the case in all problems considered here (see Section 5). Finally, the stopping criterion associated with Dykstra's algorithm is a delicate issue. A discussion about this topic and the development of some robust stopping criteria are fully described in [10]. Based on that, in here, we will stop the iterations when

$$\sum_{i=1}^{p} \|I^i_{\ell-1} - I^i_{\ell}\|^2 \leq \varepsilon, \tag{6}$$

where $\varepsilon > 0$ is a small given tolerance.

Since the gradient $\nabla f(x, y)$ of $f(x, y) = f(x) + \tau h(x, y)$ is available for each fixed $\tau > 0$, then Projected Gradient (PG) methods provide an interesting low-cost option for solving (4). They are simple and easy to code, and avoid the need for matrix factorizations (no Hessian matrix is used). There have been many different variations of the early PG methods. They all have the common property of maintaining feasibility of the iterates by frequently projecting trial steps on the feasible convex set. In particular, a well-established and effective scheme is the so-called Spectral Projected Gradient (SPG) method (see Birgin et al. [6–9]).

The SPG algorithm starts with $(x_0, y_0) \in \mathbb{R}^{2n}$ and moves at every iteration j along the internal projected gradient direction $d_j = P_{\widehat{\Omega}}((x_j, y_j) - \alpha_j \nabla f(x_j, y_j)) - (x_j, y_j)$, where $d_j \in \mathbb{R}^{2n}$ and α_j is the well-known spectral choice of step length (see [9]):

$$\alpha_j = \frac{\langle s_{j-1}, s_{j-1} \rangle}{\langle s_{j-1}, (\nabla f(x_j, y_j) - \nabla f(x_{j-1}, y_{j-1})) \rangle},$$

and $s_{j-1} = (x_j, y_j) - (x_{j-1}, y_{j-1})$. In the case of rejection of the first trial point, $(x_j, y_j) + d_j$, the next ones are computed along the same direction, i.e., $(x_+, y_+) = (x_j, y_j) + \lambda d_j$, using a nonmonotone line search to choose $0 < \lambda \leq 1$ such that the following condition holds

$$f(x_+, y_+) \leq \max_{0 \leq l \leq \min\{j, M-1\}} f(x_{k-l}, y_{k-l}) + \gamma \lambda \langle d_j, \nabla f(x_j, y_j) \rangle,$$

where $M \geq 1$ is a given integer and γ is a small positive number. Therefore, the projection onto $\widehat{\Omega}$ must be performed only once per iteration. More details can be found in [6] and [7]. In practice, $\gamma = 10^{-4}$ and a typical value for the nonmonotone parameter is $M = 10$, but the performance of the method may vary for variations of this parameter, and a fine tuning may be adequate for specific applications.

Another key feature of the SPG method is to accept the initial spectral step-length as often as possible while ensuring global convergence. For this reason, the SPG method employs a non-monotone line search that does not impose functional decrease at every iteration. The global convergence of the SPG method combined with Dykstra's algorithm to obtain the required projection per iteration can be found in [8, Section 3].

Summing up, our proposed combined algorithm is now described in detail.

Algorithm Penalty-SPG (PSPG)

S0 : Given $\tau_{-1} > 0$, and vectors x_{-1} and y_{-1}; set $k = 0$.

S1 : Compute $\tau_k > \tau_{k-1}$

S2 : Set $x_{k,0} = x_{k-1}$ and $y_{k,0} = y_{k-1}$, and from $(x_{k,0}, y_{k,0})$ apply the SPG method to (10), until

$$\|P_{\widehat{\Omega}}((x_{k,m_k}, y_{k,m_k}) - \nabla f(x_{k,m_k}, y_{k,m_k})) - (x_{k,m_k}, y_{k,m_k})\|_2 \leq tol_1$$

is satisfied at some iteration $m_k \geq 1$. Set $x_k = x_{k,m_k}$ and $y_k = y_{k,m_k}$.

S3 : If

$$h(x_k, y_k) \leq tol_2 \text{ and } |f(x_k) - f(x_{k-1})| \leq tol_2$$

then stop. Otherwise, set $k = k + 1$ and return to S1.

We note that at any iteration $k \geq 1$, Step S2 of Algorithm PSPG starts from (x_{k-1}, y_{k-1}), which is the previous solution of (4), obtained using τ_{k-1}. We also note that to stop the SPG iterations, we monitor the value of $\|P_{\widehat{\Omega}}((x_k, y_k) - \nabla f(x_k, y_k)) - (x_k, y_k)\|_2$. It is worth recalling that if $\|P_{\widehat{\Omega}}((x, y) - \nabla f(x, y)) - (x, y)\|_2 = 0$, then $(x, y) \in \widehat{\Omega}$ is stationary for problem (4) (see, e.g., [6, 8]). Each SPG iteration uses Dykstra's alternating projection scheme to obtain the required projection onto $\widehat{\Omega}$, and this internal iterative process is stopped when (6) is satisfied.

4 Cardinality Constrained Optimal Portfolio Problem

Let the vector $v \in \mathbb{R}^n$ and the symmetric and positive semi-definite matrix $Q \equiv [\sigma_{ij}]_{i,j=1,\ldots,n} \in \mathbb{R}^{n\times n}$ be the given mean return vector and variance-covariance matrix of the n risky available assets, respectively. The entry σ_{ij} in Q is the covariance between assets i and j for $i, j = 1, \ldots, n$, $\sigma_{ii} = \sigma_i^2$ and $\sigma_{ij} = \sigma_{ji}$. As a consequence of the pioneering work of Markowitz [32], the mean-variance portfolio selection problem can be formulated as (1), where the objective function is given by

$$f(x) = \frac{1}{2} x^\top Q x, \tag{7}$$

and the convex set $\Omega = \{x \in \mathbb{R}^n : v^\top x \geq \rho,\ \ e^\top x = 1,\ \ 0 \leq x_i \leq u_i,\ i = 1, \ldots, n\}$, representing the constraints of minimum expected return level ρ, budget constraint ($e^\top x = \sum_{i=1}^n x_i = 1$ means that all available wealth will be invested), and lower ($x \geq 0$ excludes short sale) and upper bounds u_i for each x_i, respectively. Notice that the minimization of $f(x)$, involving the given covariance matrix Q, accounts for the minimization of the variance, while the return is expected to be at least ρ. Notice also that, as previously discussed, in this case, the set Ω is the intersection of three easy convex sets: a half-space, a hyperplane, and a box. The additional constraint in (1), $\|x\|_0 \leq \alpha$ for $0 < \alpha < n$, plays a key role here and indicates that among the n risky available options, we can only invest in at most α assets (cardinality constraint). The solution vector x denotes an investment portfolio, and each x_i represents the fraction held of each asset i. It should be mentioned that other inequality and/or equality constraints can be added to the problem, as they represent additional real-life constraints, e.g., transaction costs [3, 26].

Now, as discussed above, our main idea is to consider the continuous formulation (2) instead of the optimization problem (1). For the portfolio selection problem, we would end up with the following problem that involves the auxiliary vector y:

$$\begin{array}{ll} \min\limits_{x,y} & \frac{1}{2}x^\top Qx \\ \text{subject to:} & v^\top x \geq \rho, \\ & e^\top x = 1, \\ & 0 \leq x_i \leq u_i, \quad \text{for all} \;\; 1 \leq i \leq n, \\ & e^\top y \geq n - \alpha, \\ & x \circ y = 0, \\ & 0 \leq y_i \leq 1, \quad \text{for all} \;\; 1 \leq i \leq n, \end{array} \tag{8}$$

where the upper bound vector $u \in \mathbb{R}^n$ and $\rho > 0$ are given. Note that the vector y appears only in the last 3 constraints, and the vector x appears in the first three constraints but also in the (non-convex) Hadamard constraint: $x \circ y = 0$.

As discussed in Section 2, the best option to force the Hadamard condition at the solution while keeping the feasible set of our problem as the intersection of a finite collection of easy convex sets is to add the term $\tau h(x, y)$ to the objective function, where our convenient choice is $h(x, y) = x^\top y$:

$$f(x, y) = \frac{1}{2}x^\top Qx + \tau x^\top y, \tag{9}$$

where $\tau > 0$ is a penalization parameter that needs to be properly chosen as described in Section 2. Since the vectors x and y will be forced by the alternating projection scheme to have all their entries greater than or equal to zero, then $h(x, y) = x^\top y \geq 0$ for any feasible pair (x, y), and forcing $\tau x^\top y = 0$ is equivalent to forcing the Hadamard condition: $x_i y_i = 0$ for all i. Notice that setting $\tau = 0$ for solving (8) with $f(x, y)$ given by (9) minimizes the risk, independently of the Hadamard condition. On the other hand, if $\tau > 0$ is sufficiently large as compared to the size of Q, then the term $x^\top y$ must be zero at the solution. Hence, choosing $\tau > 0$ represents an explicit trade-off between the risk and the Hadamard condition.

Our algorithmic proposal consists in solving a sequence of penalized problems, as described in Section 2, using the SPG scheme and Dykstra's alternating projection method (that from now on will be denoted as the SPG method) to solve problem (8), without the complementarity constraint $x \circ y = 0$, and using the objective function given by (9). That is, for a sequence of increasing penalty terms $\tau_k > 0$, we will solve the following problems:

$$\begin{array}{ll} \min\limits_{x,y} & \frac{1}{2}x^\top Qx + \tau_k x^\top y \\ \text{subject to:} & v^\top x \geq \rho, \\ & e^\top x = 1, \\ & 0 \leq x_i \leq u_i, \text{ for all } 1 \leq i \leq n, \\ & e^\top y \geq n - \alpha, \\ & 0 \leq y_i \leq 1, \text{ for all } 1 \leq i \leq n. \end{array} \tag{10}$$

Since the function $h(x, y) = x^\top y$ satisfies the properties mentioned in Section 2, if we choose the sequence of parameters $\{\tau_k\}$ such that $h(x_k, y_k)$ goes to zero when k goes to infinity, then Theorem 1 guarantees the convergence of the proposed scheme.

Before showing some computational results in our next section, let us recall that the gradient and the Hessian of the objective function f at every pair (x, y) are given by

$$\nabla f(x, y) = \begin{pmatrix} Qx + \tau_k y \\ \tau x \end{pmatrix} \text{ and } \nabla^2 f(x, y) = \begin{pmatrix} Q & \tau_k I \\ \tau_k I & 0 \end{pmatrix}.$$

Notice that, for any $\tau_k > 0$, $\nabla^2 f(x, y)$ is symmetric and indefinite.

5 Computational Results

To add understanding and illustrate the advantages of our proposed combined scheme, we present the results of some numerical experiments on an academic simple problem ($n = 6$) and also on some data sets involving real-world capital market indices from major stock markets. All the experiments were performed using Matlab R2022 with double precision on an Intel® Quad-Core i7-1165G7 at 4.70 GHz with 16GB of RAM memory, using Windows 10 Pro with 64 Bits.

For our experiments, we use Algorithm PSPG described in Section 3, setting $x_{-1} = (1/n)e$, $y_{-1} = 0$, $tol_1 = 10^{-6}$, and $tol_2 = 10^{-8}$. We recall that for the portfolio problems $h(x_k, y_k) = x_k^\top y_k$. The value of $\|P_{\hat{\Omega}}((x_k, y_k) - \nabla f(x_k, y_k)) - (x_k, y_k)\|_2$ will be denoted as the pgnorm at iteration k (see the tables below). Concerning the nonmonotone line search strategy used by the SPG method, we set $\gamma = 10^{-4}$ and $M = 10$. Dykstra's alternating projection scheme is stopped when (6) is satisfied with $\varepsilon = 10^{-8}$.

To explore the behavior of Algorithm PSPG, we will vary the minimum expected return parameter $\rho > 0$ and the cardinality constraint positive integer $1 \leq \alpha < n$. In all cases, we set the upper bound vector $u = e$, where e is the vector of ones. Of course, for certain combinations of all those parameters, the problem might be infeasible. We will discuss possible choices of these parameters to guarantee that the feasible region of problem (10) is not empty.

To keep a balanced trade-off between the risk and the Hadamard condition, it is convenient to choose the initial parameter $\tau_{-1} > 0$ of the same order of magnitude of the largest eigenvalue of Q. For that, we proceed as follows: set $z = Qe$ and $\tau_{-1} = z^\top Qz/(z^\top z)$, i.e., a Rayleigh-quotient of Q with a suitable vector z, which produces a good estimate of $\lambda_{\max}(Q)$. This choice worked well for the vast majority of the test examples. According to Remark 1, to observe convergence, we need to drive the inner product $x_k^\top y_k$ down to zero. For that, we increase the penalization parameter as follows:

$$\tau_{k+1} = \delta_{k+1}\tau_k \quad \text{where} \quad \delta_{k+1} = \delta_k + \frac{(n-\alpha)\rho}{n} \frac{|v^\top x_{k+1}|}{\sqrt{x_{k+1}^\top Q x_{k+1}}} \quad \text{and} \quad \delta_{-1} = 1. \tag{11}$$

We note that in practice, this formula increases the penalty parameter in a controlled way taking into account the ratio between the absolute value of the current return $|v^\top x_{k+1}|$ and the current risk $\sqrt{x_{k+1}^\top Q x_{k+1}}$. In all the reported experiments, the controlled sequence $\{\tau_k\}$ given by (11) was enough to guarantee that the Hadamard product goes down to zero.

Concerning the choice of the expected return, based on [13, 36], in order to consider feasible problems, we study the behavior of our combined scheme in an interval $[\rho_{\min}, \rho_{\max}]$ of possible values of the parameter ρ, which is obtained as follows. Let $\rho_{\min} = v^\top x_{\min}$ and $\rho_{\max} = v^\top x_{\max}$, where $x_{\min} = \arg\min_x \frac{1}{2}x^\top Qx + \tau x^\top y$ and $x_{\max} = \arg\max_x v^\top x - \tau x^\top y$, both of them subject to $e^\top x = 1$, $e^\top y \geq n - \alpha$, $0 \leq x_i \leq u_i$, and $0 \leq y_i \leq 1$, for all $1 \leq i \leq n$. These two auxiliary optimization problems are solved in advance, only once for each considered problem, using in turn the proposed Algorithm PSPG. For that, we fix the same parameters and we start from the same initial values indicated above. Once the interval $[\rho_{\min}, \rho_{\max}]$ has been obtained, to choose a suitable return ρ, we can proceed as follows. For a fixed $0 < \tilde{\epsilon} < 1$, if $\rho_{\min} + \tilde{\epsilon}(\rho_{\max} - \rho_{\min}) \geq 0$, we set $\rho = \rho_{\min} + \tilde{\epsilon}(\rho_{\max} - \rho_{\min})$, else if $|\rho| \leq v_{\max}$ we set $\rho = \tilde{\epsilon}|\rho|$, otherwise we set $\rho = \tilde{\epsilon}v_{\max}$. In here, $v_{\min} = \min\{v_1, \dots, v_n\}$ and $v_{\max} = \max\{v_1, \dots, v_n\}$.

For our first data set, we consider a simple portfolio problem with $n = 6$ available assets, denoted as *Simple-case* for which the mean return vector v and the covariance matrix Q are given by

$$v = (0.021 \;\; 0.04 \;\; -0.034 \;\; -0.028 \;\; -0.005 \;\; 0.006)^\top,$$

$$Q = \begin{bmatrix} 0.038 & 0.020 & 0.017 & 0.014 & 0.019 & 0.017 \\ 0.020 & 0.043 & 0.015 & 0.013 & 0.021 & 0.014 \\ 0.017 & 0.015 & 0.034 & 0.011 & 0.014 & 0.014 \\ 0.014 & 0.013 & 0.011 & 0.044 & 0.014 & 0.011 \\ 0.019 & 0.021 & 0.014 & 0.014 & 0.040 & 0.014 \\ 0.017 & 0.014 & 0.014 & 0.011 & 0.014 & 0.046 \end{bmatrix}.$$

We note that Q is symmetric and positive definite ($\lambda_{\min}(Q) = 1.79 \times 10^{-2}$ and $\lambda_{\max}(Q) = 1.17 \times 10^{-1}$). Notice that the assets three, four, and five have negative average returns. The purpose of this simple example is to demonstrate properties of the problem and the proposed algorithm in an easy-to follow fashion. For the other data sets, involving real-world capital market indices, we consider some larger problems obtained from Beasley's OR Library (http://people.brunel.ac.uk/~mastjjb/jeb/

info.html), built from weakly price data from March 1992 to September 1997, and that we will denote as Port1 (Hang Seng index with $n = 31$), Port2 (DAX index with $n = 85$), Port3 (FTSE 100 index with $n = 89$), Port4 (S&P 100 index with $n = 98$), and Port5 (Nikkei index with $n = 225$) (see also [1, 15, 24]).

The key properties, to be discussed and illustrated in the rest of this section, are the influence of the cardinality constraint to the feasible set in the risk-return plane, the efficient frontier, and the quality of the solution obtained by Algorithm PSPG. The feasible set is usually represented in the risk-return plane, presenting all possible combinations of assets that satisfy the constraints. In general, the feasible set for the classical problem without cardinality constraint has the so-called bullet shape. The efficient frontier is the set of optimal portfolios that offer the highest expected return for a defined level of risk or the lowest risk for a given level of expected return.

Introducing the cardinality constraints might complicate the feasible set in the sense that the set is shrinking as we will now show. Starting with the feasible interval for the expected return, we report in Table 1, $\rho_{\max} \le v_{\max}$ and $\rho_{\min} \ge v_{\min}$, for $\alpha = 5$ and for all the considered data sets.

Let us now take a closer look at the Simple-case. If we solve the original Markowitz problem [32] - the minimal variance portfolio, (i.e., $\min_x \frac{1}{2} x^\top Q x$ subject to $e^\top x = 1$) for the Simple-case problem, we obtain

$$\bar{x} = (0.0961, 0.1168, 0.2625, 0.2140, 0.1429, 0.1677)^\top,$$

risk $\sqrt{\bar{x}^\top Q \bar{x}} = 0.1379$, and expected return $v^\top \bar{x} = -0.0079$. Solving the same problem with the additional constraint $x \ge 0$, we get the same solution. Thus, the minimal variance portfolio is the same as the minimal variance portfolio without short sale. In Fig. 1, we present for the Simple-case problem, the return and risk for all 6 assets, the minimal variance portfolio, denoted by MVP, the classical Markowitz portfolio without short sale and the expected return constraint $v^\top x \ge \rho = 0.002$, denoted by MP, as well as the efficient frontier for different values of the cardinality constraint α. Clearly for $\alpha = 6$, i.e., without cardinality constraint, we get a classical convex efficient frontier, while for smaller α values, the curves are discontinuous and deformed (see, e.g., [15] for similar observations).

For the Simple-case problem, with $n = 6$ available assets, an approximation of the feasible set is shown in Fig. 2, which is obtained by running a simulation based on

Table 1 Return value with $\alpha = 5$ for all data sets

Problem	n	$v_{\min}$	$v_{\max}$	$\rho_{\min}$	$\rho_{\max}$
Simple case	6	−0.0340	0.0400	−0.0238	0.0373
Port1	31	5.64e-4	0.0435	0.0130	0.0435
Port2	85	−0.0160	0.0392	0.0099	0.0342
Port3	89	−0.0045	0.0328	0.0102	0.0268
Port4	98	−0.0079	0.0368	0.0077	0.0271
Port5	225	−0.0340	0.0159	−0.0060	−0.0060

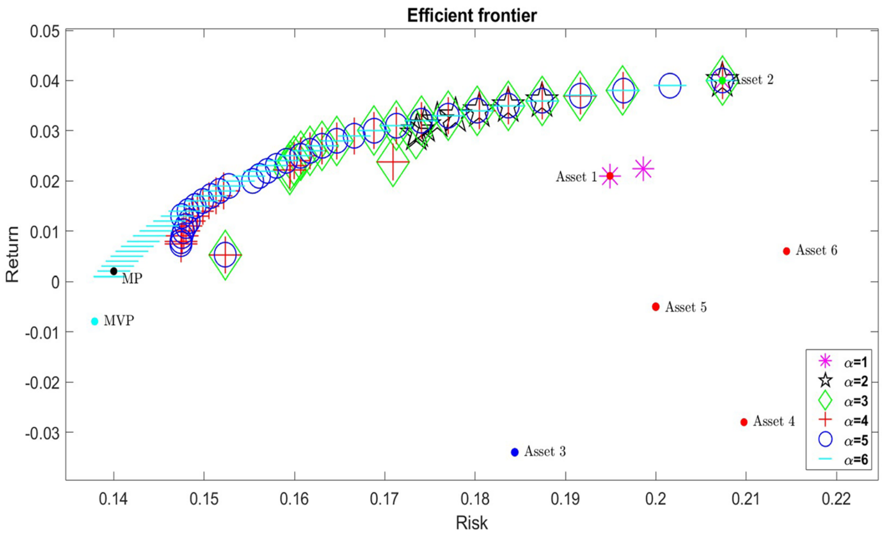

Fig. 1 Risk versus return, using Algorithm PSPG for the Simple-case problem

finite sampling. In our simulation, we pay more attention to the left side to observe the bullet shape. As a consequence, only a few scattered points are shown on the right side of the figure. We note that for a larger value of α, we get a larger area of the feasible set. We also note that the bullet shape is not affected by the cardinality constraint, but, as expected, the set is shrinking as the number of zero elements increases.

The same conclusions apply to the larger data sets coming from real assets. Below, in Fig. 3, we show the approximate feasible set for Port1. We note that once again, the area is shrinking when α decreases. We also note that the same is true for all considered cases.

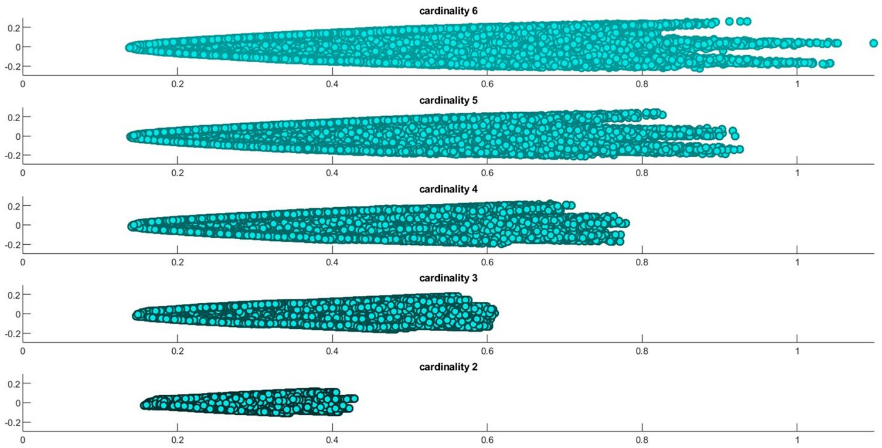

Fig. 2 Feasible set for the Simple case and $\alpha = 2, 3, 4, 5$, and 6

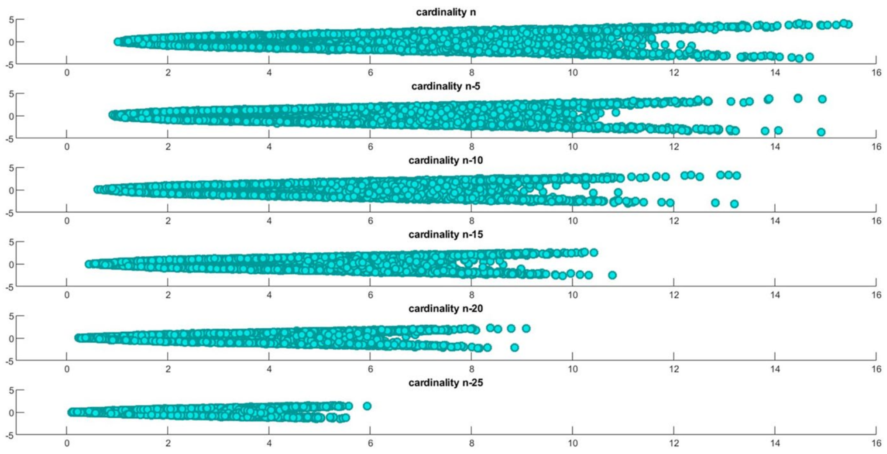

Fig. 3 Feasible set for Port1 and $\alpha = 6, 11, 16, 21, 26,$ and 31

The efficient frontier for Port1 is shown in Fig. 4. Again, we observe that the efficient frontier is deformed by the value of the cardinality constraint, and when $\alpha < n$, it is not a convex curve. For the sake of completeness, in the Appendix, we provide some tables with more detailed results, varying the cardinality constraints, for all considered data sets. We can observe in all figures and tables the effectiveness of our low-cost continuous approach (Algorithm PSPG).

Additionally, we compare our approach to IBM ILOG CPLEX Optimization Studio, Version: 22.1.0.0. CPLEX is a mixed integer quadratic programming (MIQP) solver. We note that for these problems, the solution provided by CPLEX is the globally optimal one up to the provided tolerances. The goal of comparison

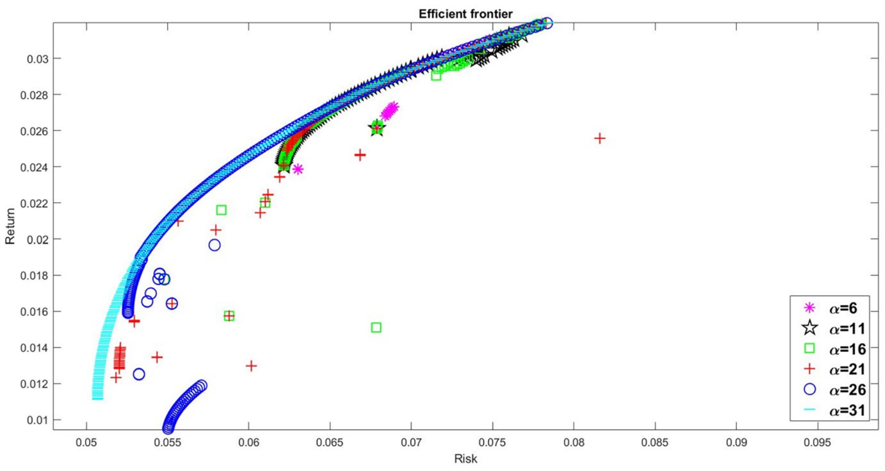

Fig. 4 Risk versus return, using Algorithm PSPG for Port1 and $\alpha = 6, 11, 16, 21, 26, 31$

is to investigate the quality of solutions obtained by PSPG and CPLEX in terms of risk and return. We also report CPU time, although CPLEX is implemented in a low-level language, and so it requires significantly less execution time than our high-level Matlab implementation. Hence, CPU time might be misleading. For solving the problems with CPLEX, we consider the following MIQP formulation, instead of (1):

$$\begin{array}{ll} \min\limits_{x,y} & \frac{1}{2}x^\top Qx \\ \text{subject to:} & v^\top x \geq \rho, \\ & e^\top x = 1, \\ & e^\top y \geq n - \alpha, \\ & 0 \leq x_i \leq 1, \quad \text{for all } 1 \leq i \leq n, \\ & x_i + y_i \leq 1, \quad \text{for all } 1 \leq i \leq n, \\ & y_i \in \{0, 1\}. \end{array}$$

Notice that in the above problem formulation, we do not have the Hadamard constrained, and instead, we have $x_i + y_i \leq 1$ followed by $y_i \in \{0, 1\}$. CPLEX is designed to work with linear constraints, and for $y_i = 0$ or $y_i = 1$, we get the same condition. It is worth mentioning that (1) can also be formulated as a convex mixed integer non-linear program (MINLP) (see, e.g., [27]). Therefore, using a convenient formulation, (1) can also be solved by branch-and-bound, e.g., using BARON [34] or SCIP [35], or by outer approximation strategies [2], e.g., using SHOT [31].

The details of tests for all considered data sets are presented in Tables 3, 4, 5, 6, 7, and 8 in the Appendix. One can easily see that PSPG produces solutions with slightly higher risk and significantly better return. In Table 5, we observe that CPLEX needs a very large number of iterations to solve the problem for $\alpha \leq 20$, which corresponds to the fact the PSPG needed a special value of τ_{-1} for these values of α and large values of penalty parameter τ. Thus, this behavior is associated with the data of Port2. In some other cases, reported in the tables in the Appendix, we can observe a rather large number of CPLEX iterations for small values of α, while PSPG solved the same problems with reasonably small values of the penalty parameters.

An interesting observation from the literature, and confirmed by our experiments, is the fact that the optimal portfolio without cardinality constraint is in fact sparse. In Table 2, we report the number of assets obtained by our algorithm and CPLEX which is in accordance with the results reported in [13, Figure 5] and [14, Section 5.2.2]. We can observe that the number of assets in the unconstrained mean-variance optimal portfolio for Port1 $\|x^*\|_0 \leq 12$, for Port4 $\|x^*\|_0 \leq 40$, and for Port5 $\|x^*\|_0 \leq 15$.

As noticed above, the feasible set of (8) belongs to the feasible set of (10). In addition, since the solution of (10) satisfies the Hadamard condition, we obtain that the solution is also a solution of (8). Then, by Theorem 2, we have that if (x^*, y^*) is a local minimizer of (10) satisfying $\|x^*\|_0 = \alpha$, then the components of y^* are binary, y^* is unique, and x^* is a local minimizer of (1). In fact, for the solutions reported in Tables 3, 4, and 6 in the Appendix, if $\|x^*\|_0 = \alpha$, we have that the components of y^* are binary. The solution may have non-binary entries in y^*; for instance, port1

Table 2 Performance of Algorithm PSPG for all cases when $n = \alpha$

Problem	$\alpha = n$	PSPG $\|x\|_0$	PSPG $v^\top x^*$	PSPG $\sqrt{(x^*)^\top Q x^*}$	CPLEX $\|x\|_0$	CPLEX $v^\top x^*$	CPLEX $\sqrt{(x^*)^\top Q x^*}$
Simple-case	6	6	0.0003	0.1394	6	0.0003	0.1394
Port1	31	12	0.0133	0.0509	12	0.0133	0.0509
Port2	85	24	0.0085	0.0234	25	0.0084	0.0234
Port3	89	34	0.0101	0.0282	34	0.0101	0.0282
Port4	98	38	0.0098	0.0223	38	0.0098	0.0223
Port5	225	12	0.0003	0.0349	12	0.0003	0.0349

with $\alpha = n = 31$, we have that y^* is binary; however, the cardinality constraint is not active $\|x^*\|_0 = 12$. Another interesting example is detected for Port3 with $\alpha = n = 89$ in which we obtain a binary y^* but $\|x^*\|_0 = 34$.

6 Conclusions and Final Remarks

Taking advantage of a recently developed continuous formulation, we have developed and analyzed a low-cost and effective computational scheme for finding a solution candidate of convex constrained optimization problems that also include a “hard-to-deal" cardinality constraint. As it appears in many applications, we assume that the region defined by the convex constraints can be written as the intersection of a finite collection of “easy to project" convex sets. Under this continuous formulation, to fulfill the cardinality constraint, the Hadamard condition $x \circ y = 0$ must be satisfied between the solution vector x and an auxiliary vector y. In our scheme, this condition is achieved by adding a non-negative penalty term $h(x, y)$ and using a classical penalization strategy. For each penalty subproblem, a convex constrained problem must be solved, which in our proposal is achieved by combining two low-cost computational schemes: the spectral projected gradient (SPG) method and Dykstra’s alternating projection method.

To illustrate the computational performance of our combined scheme, we have considered in detail the standard mean-variance limited diversified portfolio selection problem, which involves obtaining the proportion of the initial budget that should be allocated in a limited number of the available assets. For this specific application, we proposed a natural differentiable choice of the penalty term (given by $h(x, y) = x^\top y$) that must be driven to zero, which allowed us to develop a simple way of increasing the associated penalty parameter in a controlled and bounded way. In our numerical study, we have included a variety of data sets involving real-world capital market indices. For these data sets, we have produced the feasible sets and also the efficient frontier (a curve illustrating the tradeoff between risk and return) for different values of the limited number of allowed assets. In each case, we highlighted the differences that arise in the shape of this efficient frontiers as compared with the unconstrained efficient one. The presented numerical study includes comparison with CPLEX, a professional software for general mixed integer programming problems. The comparison is presented in terms of quality of solution (higher return, lower risk), and PSPG appears to be competitive.

In our modeling of the portfolio problem, we have bounded the proportion to be invested in each of the selected assets between 0 and 1. However, without altering our proposed scheme, stricter upper limits (less than 1) can be imposed on some particular assets. Clearly, this would require a more careful analysis of the feasible options for the expected return. Moreover, it could also be interesting from a portfolio point of view to allow negative entries in some of the proportions to be invested, and that can be accomplished by allowing negative values in the lower bounds of the solution vector. In that case, the penalization term to force the Hadamard condition needs to be chosen accordingly (e.g., $h(x, y) = \sum_{i=1}^{n}(x_i^2 y_i)$).

Appendix. Performance of Algorithm PSPG for All Data Sets

In Tables 3, 4, 5, 6, 7, and 8, we report the performance of PSPG and CPLEX, for several values of α, reporting the values of optimal portfolio return, risk, number of non-zero portfolio weights, number of iteration (Iter), and number of SPG iterations for PSPG, the CPU time (time) in seconds, the last value of τ, as well as the final value of the Hadamard product, and the total number of required function evaluations (fcnt). It is worth noticing that in all the results reported in these tables, the pgnorm at the obtained solution and the Hadamard products $(x^*)^\top y^*$ are strictly less than 10^{-6}, and hence, we did not report these values.

Table 3 Performance of PSPG and CPLEX for the Simple case

Algorithm	α	$v^\top x^*$	$\sqrt{(x^*)^\top Q x^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
PSPG	1	0.0400	0.2074	1	2	4	0.3708	0.117590	7	0.0018
	2	0.0293	0.1735	2	2	6	0.3133	0.117577	9	0.0016
	3	0.0053	0.1523	3	2	11	0.2786	0.117560	13	0.0017
	4	0.0053	0.1523	3	2	12	0.3211	0.117558	16	0.0017
	5	0.0053	0.1523	3	2	8	0.2799	0.117557	10	0.0012
	6	0.0003	0.1394	6	2	7	0.3001	0.117556	9	0.0003
CPLEX	1	0.0210	0.1949	1	22	-	0.09	-	-	0.0018
	2	0.0016	0.1612	2	19	-	0.05	-	-	0.0016
	3	0.0017	0.1483	3	19	-	0.03	-	-	0.0017
	4	0.0017	0.1414	4	19	-	0.05	-	-	0.0017
	5	0.0012	0.1414	5	19	-	0.06	-	-	0.0012
	6	0.0003	0.1394	6	13	-	0.02	-	-	0.0003

Table 4 Performance of Algorithm PSPG and CPLEX for problem Port1

Algorithm	α	$v^\top x^*$	$\sqrt{(x^*)^\top Qx^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
PSPG	1	0.0435	0.1382	1	2	4	0.8109	0.1475	6	0.0097
	2	0.0435	0.1382	1	2	3	0.3113	0.1476	5	0.0126
	3	0.0435	0.1382	1	2	3	0.3203	0.1477	5	0.0133
	4	0.0435	0.1382	1	2	3	0.2200	0.1476	5	0.0132
	5	0.0435	0.1382	1	2	3	0.2524	0.1476	5	0.0133
	10	0.0435	0.1382	1	2	4	0.2028	0.1475	6	0.0136
	15	0.0151	0.0678	2	2	17	0.6132	0.1473	23	0.0133
	20	0.0154	0.0530	5	2	17	0.3751	0.1473	19	0.0132
	30	0.0133	0.0509	11	2	13	0.2978	0.1471	15	0.0133
	31	0.0133	0.0509	12	2	12	0.3267	0.1471	14	0.0133
CPLEX	1	0.0233	0.0717	1	32	–	0.0900	–	- -	0.0097
	2	0.0126	0.0591	2	17	–	0.0300	–	–	0.0126
	3	0.0140	0.0544	3	17	–	0.0500	–	–	0.0133
	4	0.0132	0.0523	4	17	–	0.0300	–	–	0.0132
	5	0.0137	0.0516	1	17	–	0.0500	–	–	0.0133
	10	0.0136	0.0510	10	19	–	0.0600	–	–	0.0136
	15	0.0133	0.0509	12	13	–	0.0300	–	–	0.0133
	20	0.0132	0.0509	12	13	–	0.0200	–	–	0.0132
	30	0.0133	0.0509	12	13	–	0.0200	–	–	0.0133
	31	0.0133	0.0509	12	13	–	0.0200	–	–	0.0133

Table 5 Performance of Algorithm PSPG and CPLEX for problem Port2

Problem	α	$v^\top x^*$	$\sqrt{(x^*)^\top Qx^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
PSPG	1	0.0392	0.1065	1	2	8	0.6761	0.0976	28	0.0085
	2	0.0392	0.1065	1	2	5	0.3885	7.0147	19	0.0058
	3	0.0745	0.1327	2	2	10	0.5509	11.815	39	0.0079
	4	0.1045	0.1628	3	3	29	0.7973	12.012	155	0.0125
	5	0.0745	0.1327	2	2	10	0.5157	11.866	42	0.0163
	10	0.1267	0.2010	4	5	109	2.5451	42.404	383	0.0158
	15	0.1804	0.2954	7	3	58	1.4936	72.623	163	0.0161
	20	0.0745	0.1327	2	2	12	0.5987	26.025	69	0.0022
	25	0.0745	0.1327	2	2	12	2.1676	29.028	69	0.0024
	30	0.0745	0.1327	2	2	11	0.8536	39.052	61	0.0037
	35	0.0291	0.0428	5	2	11	0.2984	0.0977	18	0.0109
	40	0.0291	0.0428	5	2	11	0.2845	0.0977	18	0.0117
	45	0.0291	0.0428	5	2	13	0.2991	0.0976	25	0.0115

Table 6 (continued)

Problem	α	$v^\top x^*$	$\sqrt{(x^*)^\top Qx^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
	50	0.0291	0.0428	5	2	11	0.2731	0.0976	18	0.0115
	55	0.0225	0.0357	8	2	15	0.2739	0.0975	22	0.0110
	60	0.0186	0.0319	13	2	13	0.2580	0.0974	17	0.0110
	65	0.0190	0.0321	12	2	15	0.2748	0.0974	17	0.0111
	70	0.0110	0.0237	23	2	19	0.2579	0.0973	21	0.0110
	75	0.0111	0.0238	23	2	16	0.2120	0.0973	18	0.0111
	80	0.0103	0.0235	25	2	16	0.2108	0.0973	18	0.0103
	85	0.0085	0.0234	24	2	15	0.1876	0.0973	17	0.0070
CPLEX	1	0.0134	0.0477	1	504	–	0.14	–	–	0.0085
	2	0.0066	0.0331	2	5638	–	0.27	–	–	0.0058
	3	0.0084	0.0296	3	34024	–	0.53	–	–	0.0079
	4	0.0125	0.0289	4	13926	–	0.39	–	–	0.0125
	5	0.0163	0.0298	5	6982	–	0.28	–	–	0.0163
	10	0.0158	0.0263	10	2743	–	0.20	–	–	0.0158
	15	0.0161	0.0259	15	1485	–	0.23	–	–	0.0161
	20	0.0083	0.0234	20	75	–	0.20	–	–	0.0022
	25	0.0083	0.0234	25	13	–	0.02	–	–	0.0024
	30	0.0084	0.0234	25	13	–	0.02	–	–	0.0037
	35	0.0109	0.0236	24	14	–	0.02	–	–	0.0109
	40	0.0117	0.0238	24	14	–	0.03	–	–	0.0117
	45	0.0115	0.0238	24	14	–	0.03	–	–	0.0115
	50	0.0115	0.0238	24	14	–	0.02	–	–	0.0115
	55	0.0110	0.0237	24	14	–	0.02	–	–	0.0110
	60	0.0110	0.0237	24	14	–	0.02	–	–	0.0110
	65	0.0111	0.0237	24	14	–	0.02	–	–	0.0111
	70	0.0110	0.0237	24	14	–	0.03	–	–	0.0110
	75	0.0111	0.0237	24	14	–	0.05	–	–	0.0111
	80	0.0103	0.0235	26	14	–	0.03	–	–	0.0103
	85	0.0084	0.0234	25	13	–	0.03	–	–	0.0070

Table 6 Performance of Algorithm PSPG and CPLEX for problem Port3

Problem	α	$v^\top x^*$	$\sqrt{(x^*)^\top Q x^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
PSPG	1	0.0328	0.0779	1	2	7	1.0909	0.1133	16	0.0101
	2	0.0328	0.0779	1	2	3	1.0322	9.8615	11	0.0104
	3	0.0328	0.0779	1	2	4	0.6540	9.7468	12	0.0102
	4	0.0328	0.0779	1	2	3	0.5711	9.6557	10	0.0160
	5	0.0328	0.0779	1	2	5	0.7342	9.5319	17	0.0135
	10	0.0328	0.0779	1	3	56	4.5180	9.0391	402	0.0119
	15	0.0328	0.0779	1	3	55	1.3683	11.429	406	0.0117
	20	0.0104	0.0284	14	14	676	36.438	0.0003	885	0.0104
	25	0.0104	0.0284	14	11	530	28.613	0.0003	666	0.0104
	30	0.0104	0.0284	14	12	547	29.572	0.0003	695	0.0104
	35	0.0328	0.0779	1	2	4	0.5872	89.248	13	0.0107
	40	0.0104	0.0286	12	5	215	10.943	0.0005	282	0.0104
	45	0.0251	0.0464	4	3	51	1.9333	0.2019	59	0.0114
	50	0.0104	0.0285	13	5	194	3.2324	0.0005	252	0.0104
	55	0.0104	0.0285	13	4	151	2.6642	0.0005	183	0.0104
	60	0.0167	0.0321	19	5	183	6.3165	1.1546	440	0.0133
	65	0.0157	0.0310	21	8	341	5.8265	2.8615	652	0.0145
	70	0.0105	0.0333	10	2	29	0.9089	0.1129	31	0.0105
	75	0.0105	0.0308	11	2	20	0.6148	0.1129	22	0.0105
	80	0.0105	0.0295	16	2	16	0.6074	0.1129	18	0.0105
	85	0.0104	0.0286	23	2	26	0.6432	0.1129	28	0.0104
	89	0.0101	0.0282	34	2	18	0.5024	0.1129	20	0.0101
CPLEX	1	0.0151	0.0473	1	328	–	0.19	–	–	0.0101
	2	0.0117	0.0384	2	9537	–	0.42	–	–	0.0104
	3	0.0104	0.0346	3	133879	–	2.76	–	–	0.0102
	4	0.0160	0.0340	4	26021	–	0.58	–	–	0.0160
	5	0.0135	0.0314	5	125555	–	2.61	–	–	0.0135
	10	0.0119	0.0290	10	35025	–	1.08	–	–	0.0119
	15	0.0117	0.0286	15	4705	–	0.42	–	–	0.0117
	20	0.0104	0.0282	20	1102	–	0.44	–	–	0.0104
	25	0.0104	0.0282	24	909	–	0.48	–	–	0.0104
	30	0.0104	0.0282	28	545	–	0.50	–	–	0.0104
	35	0.0107	0.0282	32	14	–	0.02	–	–	0.0107
	40	0.0104	0.0282	33	13	–	0.02	–	–	0.0104
	45	0.0114	0.0283	30	13	–	0.03	–	–	0.0114
	50	0.0104	0.0282	33	13	–	0.03	–	–	0.0104
	55	0.0104	0.0282	33	13	–	0.03	–	–	0.0104
	60	0.0133	0.0289	27	13	–	0.03	–	–	0.0133
	65	0.0145	0.0294	28	14	–	0.05	–	–	0.0145
	70	0.0105	0.0282	33	13	–	0.02	–	–	0.0105

Table 6 (continued)

Problem	α	$v^{\top}x^*$	$\sqrt{(x^*)^{\top}Qx^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
	75	0.0105	0.0282	33	13	–	0.02	–	–	0.0105
	80	0.0105	0.0282	33	13	–	0.02	–	–	0.0105
	85	0.0104	0.0282	33	13	–	0.02	–	–	0.0104
	89	0.0101	0.0282	34	13	–	0.09	–	–	0.0101

Table 7 Performance of Algorithm PSPG and CPLEX for problem Port4

Problem	α	$v^{\top}x^*$	$\sqrt{(x^*)^{\top}Qx^*}$	$\|x^*\|_0$	Iter	Iter-SPG	Time	τ	fcnt	ρ
PSPG	1	0.0343	0.0983	1	2	11	1.0960	0.0891	32	0.0095
	2	0.0368	0.1084	1	2	5	0.2783	0.8903	18	0.0091
	3	0.0368	0.1084	1	2	5	0.2805	0.8899	18	0.0075
	4	0.0368	0.1084	1	2	26	0.4912	0.8908	74	0.0108
	5	0.0368	0.1084	1	2	5	0.2296	0.8905	18	0.0101
	10	0.0368	0.1084	1	2	4	0.2047	0.8890	11	0.0050
	15	0.0368	0.1084	1	2	5	0.2230	1.7778	17	0.0047
	20	0.0368	0.1084	1	2	4	0.3299	3.7049	12	0.0048
	25	0.0206	0.0365	10	3	72	0.9029	0.4027	118	0.0053
	30	0.0194	0.0346	14	3	107	2.0201	0.5036	194	0.0060
	35	0.0197	0.0349	13	3	74	0.9154	0.5033	133	0.0060
	40	0.0178	0.0319	20	3	96	1.7136	0.8921	144	0.0055
	45	0.0127	0.0371	4	3	51	0.9430	0.0202	54	0.0091
	50	0.0175	0.0410	3	3	51	0.9916	0.0202	67	0.0091
	55	0.0109	0.0292	6	2	24	0.4256	0.0101	34	0.0106
	60	0.0194	0.0346	14	6	213	3.8296	0.9062	417	0.0063
	65	0.0191	0.0339	16	20	979	17.236	1.1020	2148	0.0061
	70	0.0132	0.0344	5	2	20	0.3864	0.0301	28	0.0010
	75	0.0252	0.0481	7	13	561	14.115	0.9461	1052	0.0067
	80	0.0138	0.0365	5	2	27	2.1716	0.0888	35	0.0080
	85	0.0142	0.0364	6	2	17	1.5391	0.0888	25	0.0075
	90	0.0086	0.0250	14	2	23	1.1892	0.0888	25	0.0073
	95	0.0089	0.0231	18	2	28	1.1098	0.0888	30	0.0080
	98	0.0098	0.0223	38	2	20	0.9998	0.0888	22	0.0098
CPLEX	1	0.0115	0.0462	1	806	–	0.09	–	–	0.0095
	2	0.0095	0.0350	2	29245	–	0.48	–	–	0.0091
	3	0.0081	0.0300	3	506050	–	3.63	–	–	0.0075
	4	0.0108	0.0287	4	1750081	–	14.89	–	–	0.0108
	5	0.0101	0.0266	5	2497651	–	20.86	–	–	0.0101
	10	0.0070	0.0231	10	698137	–	6.38	–	–	0.0050
	15	0.0077	0.0223	15	8163	–	0.33	–	–	0.0047
	20	0.0075	0.0221	20	11669	–	0.33	–	–	0.0048

Table 7 (continued)

Problem	α	$v^{\top}x^{*}$	$\sqrt{(x^{*})^{\top}Qx^{*}}$	$\|x^{*}\|_{0}$	Iter	Iter-SPG	Time	τ	fcnt	ρ
	25	0.0074	0.0221	25	1062	–	0.19	–	–	0.0053
	30	0.0078	0.0221	28	540	–	0.17	–	–	0.0060
	35	0.0076	0.0221	35	74	–	0.09	–	–	0.0060
	40	0.0077	0.0220	38	14	–	0.02	–	–	0.0055
	45	0.0091	0.0222	39	14	–	0.02	–	–	0.0091
	50	0.0091	0.0222	39	14	–	0.00	–	–	0.0091
	55	0.0106	0.0226	35	13	–	0.01	–	–	0.0106
	60	0.0077	0.0220	38	14	–	0.03	–	–	0.0063
	65	0.0077	0.0220	38	14	–	0.02	–	–	0.0061
	70	0.0077	0.0220	38	14	–	0.03	–	–	0.0010
	75	0.0077	0.0220	38	14	–	0.06	–	–	0.0067
	80	0.0080	0.0220	38	14	–	0.02	–	–	0.0080
	85	0.0077	0.0220	38	14	–	0.01	–	–	0.0075
	90	0.0077	0.0220	38	14	–	0.02	–	–	0.0073
	95	0.0080	0.0220	38	14	–	0.02	–	–	0.0080
	98	0.0098	0.0223	38	14	–	0.03	–	–	0.0098

Table 8 Performance of Algorithm PSPG and CPLEX for problem Port5

Problem	α	$v^{\top}x^{*}$	$\sqrt{(x^{*})^{\top}Qx^{*}}$	$\|x^{*}\|_{0}$	Iter	Iter-SPG	Time	τ	fcnt	ρ
PSPG	2	0.0161	0.1081	2	2	102	7.4534	0.1010	356	7.7209e-06
	3	0.0037	0.0538	3	20	1020	77.895	0.0100	1342	7.8605e-06
	4	0.0034	0.0500	3	20	1020	80.528	0.0100	1333	7.8656e-06
	5	0.0009	0.0388	4	12	612	43.506	0.0100	772	1.2051e-05
	10	0.0006	0.0355	7	4	204	16.183	0.0400	482	1.1788e-05
	15	0.0036	0.0377	8	5	255	19.572	0.2000	658	7.2419e-06
	20	0.0059	0.0399	9	11	561	39.421	0.9000	1633	5.9825e-06
	25	0.0058	0.0399	10	9	459	19.429	1.0000	1461	6.4813e-06
	30	0.0003	0.0349	10	8	325	21.612	0.0001	463	7.8618e-06
	225	0.0003	0.0349	12	2	10	1.4476	0.9051	12	1.1926e-05
CPLEX	2	0.0058	0.0439	2	1964	–	0.48	–	–	7.7209e-06
	3	0.0027	0.0391	3	1034	–	0.41	–	–	7.8605e-06
	4	0.0009	0.0367	4	386	–	0.61	–	–	7.8656e-06
	5	0.0003	0.0356	5	132	–	0.33	–	–	1.2051e-05
	10	0.0003	0.0349	10	19	–	0.27	–	–	1.1788e-05
	15	0.0003	0.0349	12	17	–	0.22	–	–	7.2419e-06
	20	0.0003	0.0349	12	17	–	0.11	–	–	5.9825e-06
	25	0.0003	0.0349	12	17	–	0.26	–	–	6.4813e-06
	30	0.0003	0.0349	12	17	–	0.09	–	–	7.8618e-06
	225	0.0003	0.0349	12	17	–	0.11	–	–	1.1926e-05

Acknowledgements We are thankful to the comments and suggestions of two anonymous referees, which helped us to improve the presentation of our work.

Author Contributions NK: study, conception, design, data collection, conceptualization, methodology, and formal analysis. EHMK: study, conception, design, data collection, conceptualization, methodology, and formal analysis. MR: study, conception, design, data collection, conceptualization, methodology, and formal analysis.

Funding Open access funding provided by FCT|FCCN (b-on). The first author was financially supported by the Serbian Ministry of Education, Science, and Technological Development and Serbian Academy of Science and Arts, grant no. F10. The second author was financially supported by Fundação para a Ciência e a Tecnologia (FCT) (Portuguese Foundation for Science and Technology) under the scope of the projects UIDB/MAT/00297/2020, UIDP/MAT/00297/2020 (Centro de Matemática e Aplicações), and UI/297/2020-5/2021. The third author was financially supported by Fundação para a Ciência e a Tecnologia (FCT) (Portuguese Foundation for Science and Technology) under the scope of the projects UIDB/MAT/00297/2020, UIDP/MAT/00297/2020 (Centro de Matemática e Aplicações).

Data Availability The data generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Code Availability The codes generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Ethics Approval Not applicable.

Consent to Participate Not applicable.

Consent for Publication All authors read and approved the submission of the final manuscript.

Conflict of Interest The authors declare no competing interests.

References

1. Beasley JE (1990) OR-Library: distributing test problems by electronic mail. J Oper Res Soc 41(11):1069–1072
2. Bernal DE, Peng Z, Kronqvist J, Grossmann IE (2022) Alternative regularizations for Outer-Approximation algorithms for convex MINLP. J Glob Optim 84:807–842

3. Bertsimas D, Darnell C, Soucy R (1999) Portfolio construction through mixed-integer programming at Grantham. Mayo, Van Otterloo and Company, Interfaces 29(1):49–66
4. Bertsimas D, Shioda R (2009) Algorithm for cardinality-constrained quadratic optimization. Comput Optim Appl 43:1–22
5. Bienstock D (1996) Computational study of a family of mixed-integer quadratic programming problems. Math Programming 74:121–140
6. Birgin EG, Martínez JM, Raydan M (2000) Nonmonotone spectral projected gradient methods on convex sets. SIAM J Optim 10:1196–1211
7. Birgin EG, Martínez JM, Raydan M (2001) Algorithm 813: SPG - software for convex-constrained optimization. ACM Trans Math Softw 27:340–349
8. Birgin EG, Martínez JM, Raydan M (2003) Inexact spectral projected gradient methods on convex sets. IMA J Numer Anal 23:539–559
9. Birgin EG, Martínez JM, Raydan M (2014) Spectral projected gradient methods: review and perspectives. J Stat Softw 60(3)
10. Birgin EG, Raydan M (2005) Robust stopping criteria for Dykstra's algorithm. SIAM J Sci Comput 26:1405–1414
11. Boyle JP, Dykstra L (1986) A method for finding projections onto the intersections of convex sets in Hilbert spaces. In: Dykstra R, Robertson T, Wright FT (eds) Advances in Order Restricted Statistical Inference. Lecture Notes in Statistics, 37: 28–47. Springer, New York
12. Burdakov OP, Kanzow C, Schwartz A (2016) Mathematical programs with cardinality constraints: reformulation by complementarity-type conditions and a regularization method. SIAM J Optim 26(1):397–425
13. Cesarone F, Scozzari A, Tardella F (2009) Efficient algorithms for mean-variance portfolio optimization with hard real-world constraints. Giornale dell'Istituto Italiano degli Attuari 72:37–56
14. Cesarone F, Scozzari A, Tardella F (2013) A new method for mean-variance portfolio optimization with cardinality constraints. Ann Oper Res 205:213–234
15. Chang TJ, Meade N, Beasley JE, Sharaiha YM (2000) Heuristics for cardinality constrained portfolio optimisation. Comput Oper Res 27(13):1271–1302
16. Chen Y, Ye Y, Wang M (2019) Approximation hardness for a class of sparse optimization problems. J Mach Learn Res 20(38):1–27
17. Combettes PL (2000) Strong convergence of block-iterative outer approximation methods for convex optimization. SIAM J Control Optim 38:538–565
18. Deutsch FR (2001) Best approximation in inner product spaces. Springer-Verlag, New York
19. Di Lorenzo D, Liuzzi G, Rinaldi F, Schoen F, Sciandrone M (2012) A concave optimization-based approach for sparse portfolio selection. Optim Methods Softw 27:983–1000
20. Escalante R, Raydan M (2011) Alternating projection methods. SIAM, Philadelphia
21. Fastrich B, Paterlini S, Winkler P (2015) Constructing optimal sparse portfolios using regularization methods. Comput Manag Sci 12(3):417–434
22. Fiacco AV, McCormick GP (1968) Nonlinear programming: sequential unconstrained minimization techniques. John Wiley and Sons, New York
23. Gao JJ, Li D (2013) Optimal cardinality constrained portfolio selection. Oper Res 61(3):745–761
24. Juszczuk P, Kaliszewski I, Miroforidis J, Podkopaev D (2022) Mean return - standard deviation efficient frontier approximation with low-cardinality portfolios in the presence of the risk-free asset. Int Trans Oper Res. https://doi.org/10.1111/itor.13121
25. Kanzow C, Raharja AB, Schwartz A (2021) Sequential optimality conditions for cardinality-constrained optimization problems with applications. Comput Optim Appl 80:185–211
26. Krejić N, Kumaresan M, Rožnjik A (2011) VaR optimal portfolio with transaction costs. Appl Math Comput 218(8):4626–4637
27. Kronqvist J, Bernal DE, Lundell A, Grossmann IE (2019) A review and comparison of solvers for convex MINLP. Optim Eng 20:397–455
28. Krulikovski EHM, Ribeiro AA, Sachine M (2021) On the weak stationarity conditions for mathematical programs with cardinality constraints: a unified approach. Appl Math Optim 84:3451–3473
29. Krulikovski EHM, Ribeiro AA, Sachine M (2022) A comparative study of sequential optimality conditions for mathematical programs with cardinality constraints. JOTA 192:1067–1083
30. Luenberger DG (1984) Linear and nonlinear programming. Addison-Wesley, Menlo Park, CA
31. Lundell A, Kronqvist J, Westerlund T (2017) SHOT - a global solver for convex MINLP in Wolfram Mathematica. Comput Aided Chem Eng 40:2137–2142
32. Markowitz H (1952) Portfolio selection. J Financ 7:77–91

33. Moreno J, Datta B, Raydan M (2009) A symmetry preserving alternating projection method for matrix model updating. Systems and Signal Processing 23:1784–1791
34. Sahinidis NV (1996) BARON: a general purpose global optimization software package. J Glob Optim 8:201–205
35. Vigerske S, Gleixner A (2018) SCIP: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. Optim Methods Softw 33(3):563–593
36. Zeng X, Sun X, Li D (2014) Improving the performance of MIQP solvers for quadratic programs with cardinality and minimum threshold constraints: a semidefinite program approach. INFORMS J Comput 26(4):690–703

Authors and Affiliations

N. Krejić[1] · E. H. M. Krulikovski[2] · M. Raydan[2]

✉ E. H. M. Krulikovski
e.krulikovski@fct.unl.pt

N. Krejić
natasak@uns.ac.rs

M. Raydan
m.raydan@fct.unl.pt

[1] Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 4, 21000 Novi Sad, Serbia

[2] Center for Mathematics and Applications (NovaMath), FCT NOVA, 2829-516 Caparica, Portugal