Globally convergent decomposition algorithm for risk parity problem in portfolio selection

A.Cassioli, G.Cocchi, F.D'Amato, M.Sciandrone

February 11, 2016

1 Introduction

2 Preliminary background

Let us consider the following optimization problem:

$$\min_{x,y} \quad f(x,y) \tag{1a}$$

$$\text{s.t.} \quad l \leq x \leq u \tag{1b}$$

$$\mathbf{1}^T x = 1 \tag{1c}$$

where $x \in \mathbb{R}^n$, $y \in \mathbb{R}^k$, f continuously differentiable, $l, u \in \mathbb{R}^n$ with $l < u$ and $\mathbf{1} \in \mathbb{R}^n$ is all composed by ones.
Now we define the feasible set $\mathcal{F}$ of Problem (1):

$$\mathcal{F} = \{(x,y) \in \mathbb{R}^{n+k} : \mathbf{1}^T x = 1, l \leq x \leq u\}. \tag{2}$$

Since the constraints of Problem (1) respect constraints qualification conditions, a feasible point (x, y) is a stationary point, if the Karush-Kuhn-Tucker (KKT) conditions are satisfied.
Let $L(x, y, \lambda, \mu, \gamma)$ the Lagrangian function associated to Problem (1) then we can write KKT conditions.

Proposition 2.1 (Optimality conditions (Necessary)) *Let $(x^*, y^*) \in \mathbb{R}^{n+k}$, with $(x^*, y^*) \in \mathcal{F}$, a local optimum for Problem (1). Then there exist three multipliers $\lambda^* \in \mathbb{R}^n$, $\mu^* \in \mathbb{R}^n, \gamma^* \in \mathbb{R}$ such that:*

$$\begin{aligned}
&\nabla_x L(x^*, y^* \lambda^*, \mu^*, \gamma^*) = \nabla_x F(x^*, y^*) + \lambda^* - \mu^* + \gamma^* = 0 \\
&\nabla_x L(x^*, y^* \lambda^*, \mu^*, \gamma^*) = \nabla_y F(x^*, y^*) = 0 \\
&\lambda_i^*(l - x_i^*) = 0, \ \forall i \\
&\mu_i^*(x_i^* - u) = 0, \ \forall i \\
&\lambda^*, \mu^* \geq 0
\end{aligned} \tag{3}$$

From the first condition we have:

$$\nabla_x f(x^*, y^*) - \lambda^* + \mu^* + \gamma^* = 0 \tag{4}$$

Then there are three possible cases:

$$\frac{\partial F(x^*, y^*)}{dx_i} = \begin{cases} -\mu_i^* - \gamma & x_i^* = u \\ -\gamma + \lambda_i^* & x_i^* = l \\ -\gamma & l < x_i^* < u \end{cases} \tag{5}$$

Then if $x_j^* > l$:

$$\frac{\partial f(x^*)}{dx_j} \leq \frac{\partial f(x^*)}{dx_i}, \forall i \tag{6}$$

Let $(x, y) \in \mathcal{F}$, we define a set of all feasible direction in (x, y):

$$\mathcal{D}(x, y) = \{d \in \mathbb{R}^{n+k} : \mathbf{1}^T d_x = 0, d_i \geq 0 \ \forall i \in L(x), d_i \leq 0 \ \forall i \in U(x)\} \tag{7}$$

where:

$$\begin{aligned} L(x) &= \{i : \ x_i = l\} \\ U(x) &= \{i : \ x_i = u\} \end{aligned} \tag{8}$$

2.1 Set of sparse feasible directions

Because of we will describe our decomposition method with respect to x and y, we have to pay attention only on the x variable to find a feasible descent direction w.r.t. x. In our case we want to build a set of sparse feasible direction in order to justify our decomposition approach.
Let $(x, y) \in \mathcal{F}$ non stationary w.r.t. x, then it's easy to see that

$$L(x) \neq \{1, \ldots, n\} \tag{9}$$

hence $\exists i$ such that $x_i > 0$ and $j \neq i$ such that:

$$\frac{\partial f(x^*)}{dx_i} > \frac{\partial f(x^*)}{dx_j}, \tag{10}$$

Now we define a direction $d^{i,j} \in \mathbb{R}^n$ with only two non-zero components such that:

$$d_h^{i,j} = \begin{cases} 1, & h = j \\ -1, & h = i \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

Proposition 2.2 *Let (x, y) a feasible point for Problem (1). Then the direction $d^{i,j}$ is feasible and descent direction in x.*

Proof For the feasibility it is enough to see that $\mathbf{1}^T d^{i,j} = 1 - 1 = 0$. Then we can apply sufficient conditions for descent direction in x, such that:

$$\nabla_x f(x,y)^T d^{i,j} = \frac{\partial f(x)}{dx_j} - \frac{\partial f(x)}{dx_i} < 0;$$

As always, we should choose the steepest descent direction composed by only two non-zero components. This can be done computing the *Most Violating Pair* (i,j) such that $x_i > 0$ and:

$$(i,j) \in \arg\min_{l,m} \left\{ \frac{\partial f(x^*)}{dx_l} - \frac{\partial f(x^*)}{dx_m} \right\} \tag{12}$$

If one doesn't want to use decomposition methods, he can define a direction $d_{xy} \in \mathbb{R}^{n+k}$ such that:

$$d_{xy} = \{d^{i,j}, -\nabla_y f(x,y)\} \tag{13}$$

2.2 Armijo-Type Line Search Algorithm

In this section, we briefly describe the well-known Armijo-type line search along a feasible descent direction. The procedure will be used in the decomposition method presented in the next section. Let $d^k \in \mathcal{D}(x_k)$ $x^k \in \mathcal{F}$. In particular we choose $d^k = d_k^{i,j}$ with MVP $(i(k), j(k))$. We denote by Δ_k the maximum feasible step along d^k.
It is easy to see that:

$$\Delta_k = \min\{x^k_{j(k)} - l, u - x^k_{i(k)}\}$$

Then at iteration $k+1$ we have:

$$x^{k+1}_{j(k)} = \begin{cases} l & \alpha_k = x^k_{j(k)} - l \\ x^k_{j(k)} - u + x^k_{i(k)} & \alpha_k = u - x^k_{i(k)} \end{cases}$$

and:

$$x^{k+1}_{i(k)} = \begin{cases} x^k_{j(k)} - l + x^k_{i(k)} & \alpha_k = x^k_{j(k)} - l \\ u & \alpha_k = u - x^k_{i(k)} \end{cases}$$

Algorithm 1: Armijo-Type Line Search

Data: Given $\alpha > 0$, $\delta \in (0,1)$, $\gamma \in (0, 1/2)$ and the initial stepsize $\Delta^{(k)} = \min\{x^k_{j(k)} - l, u - x^k_{i(k)}\}$

1 Set $\alpha = \Delta^{(k)}$
2 **while** $f(x^k, y^k) + \alpha d^k) > f(x^k, y^k) + \gamma\alpha\nabla_x f(x^k, y^k)^T d^k$ **do**
3   Set $\alpha = \delta\alpha$

2.3 Exact Line Search

When we move along the direction d^{i^*,j^*}, defined in (11), we modify only 2 variables (x_{i^*}, x_{j^*}) leaving the others unchanged. Thus, we can see our $f(x, y)$ as a function of two components, i.e. we can rewrite Problem (1) as

$$
\begin{aligned}
\min_{x_{i^*}, x_{j^*}} \quad & f(x_{i^*}, x_{j^*}) && \text{(14a)}\\
\text{s.t.} \quad & l_{i^*} \leq x_{i^*} \leq u_{i^*} && \text{(14b)}\\
& l_{j^*} \leq x_{j^*} \leq u_{j^*} && \text{(14c)}\\
& x_{i^*} + x_{j^*} = \underbrace{1 - \sum_{h \neq i^*, j^*} x_h}_{c} && \text{(14d)}
\end{aligned}
$$

Thanks to the last constraint, we can substitute $x_{i^*} = c - x_{j^*}$ and then we obtain

$$
\begin{aligned}
\min_{x_{j^*}} \quad & f(x_{j^*}) && \text{(15a)}\\
\text{s.t.} \quad & x_{i^*} = c - x_{j^*} && \text{(15b)}\\
& ll_{j^*} = \max\{l_{j^*}, c - u_{i^*}\} \leq x_{j^*} \leq \min\{u_{j^*}, c - l_{i^*}\} = uu_{j^*} && \text{(15c)}
\end{aligned}
$$

Because the domain is $I = [ll_{j^*}, uu_{j^*}]$, if $f(x_{j^*})$ is continuous in I, then f has a minimum in I. If $f(x_{j^*})$ is differentiable in I we can compute $f'(x_{j^*})$. Let $R = \{r \mid f'(r) = 0, r \in I\}$ be set set of the real feasible roots of f'. Each $r \in R$ can be a local maximum, minimum or flex; if $R = \{\emptyset\}$, then the minimum of f is on the extreme points of I.
Let $r* = \arg\min_{r \in R} f(r)$, then the optimal step α^* along the direction d^{i^*,j^*} is

$$\alpha^* = x_{j^*} - r^* > 0 \tag{16}$$

TODO: problema di notazione, x_{j^*} rappresenta il valore della componente j^* del vettore x prima della ricerca di linea

3 A decomposition framework

The algorithm follows the Gauss-Seidel scheme with 2 blocks of variables (x and y). At each iteration, we optimize f w.r.t. one block of variables, considering the other block fixed.
The y-block of variables is unconstrained, so we can find

$$y^{(k+1)} = \arg\min_{y \in \mathbb{R}^m} f(x^{(k)}, y) \tag{17}$$

using the first-order necessary condition, i.e. *TODO: ma questo vale se f è convessa rispetto ad y*

$$\nabla_y f(x^{(k)}, y^{(k)}) = 0 \tag{18}$$

Algorithm 2: Decomposition Algorithm

Data: Given the initial feasible guess $(x^{(0)}, y^{(0)})$

Set $k = 0$

while *(not convergence)* **do**

  Compute $y^{(k+1)}$ as in (17)

  Compute $\nabla_x f(x^{(k)}, y^{(k+1)})$

  Choose indexes $i^*(k), j^*(k)$ using the Gauss-Southwell rules

  Choose a step $\alpha^{(k)}$ along the direction $d^{i^*(k),j^*(k)}$

  Set $x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{i^*(k),j^*(k)}$

  Set $k = k + 1$

4 Convergence analysis

5 A case study: Portfolio Selection

In this section, we show that the general framework presented above can be used to a specific class of portfolio selection problem, namely the Risk Parity portfolio selection. Let $x \in \mathbb{R}^n$ be the portfolio, where x_i represents the fraction of budget that is invested in the asset i, and let Q be the covariance matrix between the assets; in [**?**] is introduced the following least-square optimization problem:

$$\min_{x,\theta} \quad f(x, \theta) = \sum_{i=i}^{n} \left(x_i q_i^T \cdot x - \theta \right)^2 \tag{19a}$$

$$\text{s.t.} \quad l \leq x \leq u \tag{19b}$$

$$\mathbf{1}^T x = 1 \tag{19c}$$

where q_i is the i-th column of Q. Note that the problem is not necessary convex with respect to x but it is strictly convex and coercive with respect to θ. Infact we have:

$$\frac{\partial^2}{d^2\theta} f(x, \theta) = 2 \tag{20}$$

which is a sufficient condition for a strictly convex function. At each iteration k, we have

$$\theta^{(k+1)} = \arg\min_{\theta} f(x^{(k)}, \theta) \tag{21}$$

f is strictly convex with respect to θ, so we can find $\theta^{(k+1)}$ as a solution of

$$\nabla_\theta f(x^{(k)}, \theta) = 0 \tag{22}$$

That is

$$\theta^{(k+1)} = \frac{\sum_{i=1}^{n} x_i^{(k)} q_i^T \cdot x^{(k)}}{n} \tag{23}$$

n	G-S$_{(LS)}$	G-S$_{(OPT)}$	SNOPT
5	100.0%	100.0%	100.0%
10	100.0%	100.0%	100.0%
20	100.0%	100.0%	100.0%
50	99.0%	99.4%	99.5%
100	85.4%	87.0%	91.2%
200	40.0%	45.0%	52.0%
300	22.0%	24.0%	34.0%
500	0.0%	6.0%	3.0%
750	0.0%	2.0%	0.0%
1000	0.0%	0.0%	0.0%
1250	0.0%	0.0%	0.0%

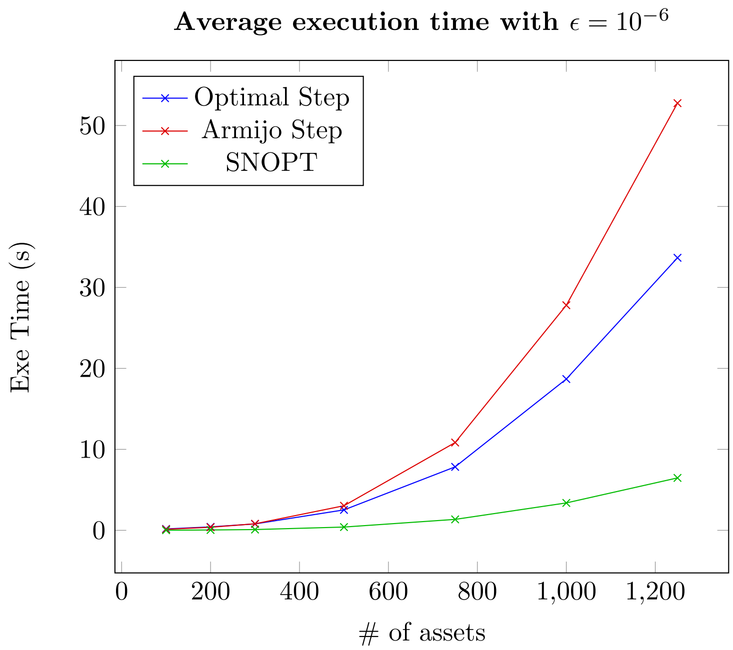
Table 1: $x_i^{(0)} = \frac{1}{n} \quad \forall i$

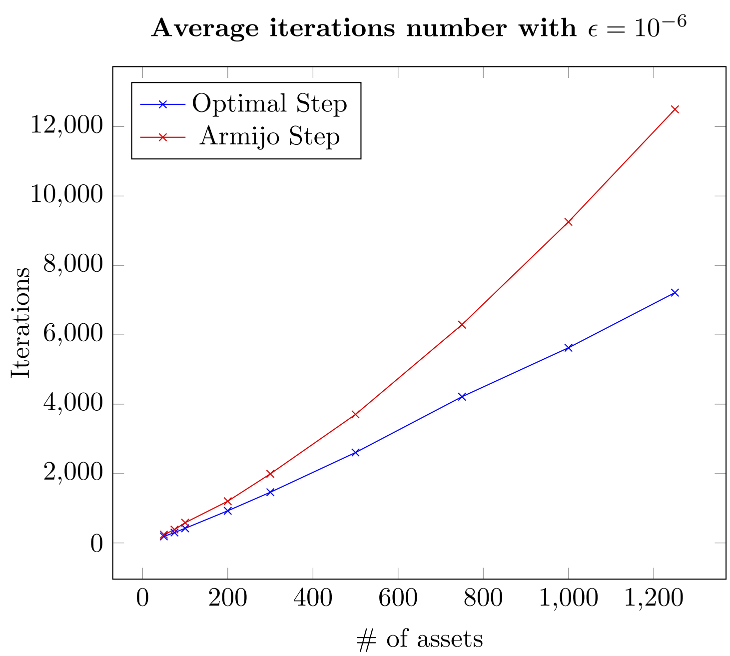
n	G-S$_{(LS)}$	G-S$_{(OPT)}$	SNOPT
5	100.0%	100.0%	100.0%
10	100.0%	100.0%	100.0%
20	99.8%	99.8%	99.8%
50	100.0%	99.8%	99.1%
100	100.0%	99.8%	96.6%
200	100.0%	99.6%	94.8%
300	100.0%	99.0%	92.0%
500	100.0%	99.0%	92.5%
750	99.0%	98.0%	86.5%
1000	96.0%	95.0%	76.0%
1250	90.0%	93.0%	76.5%

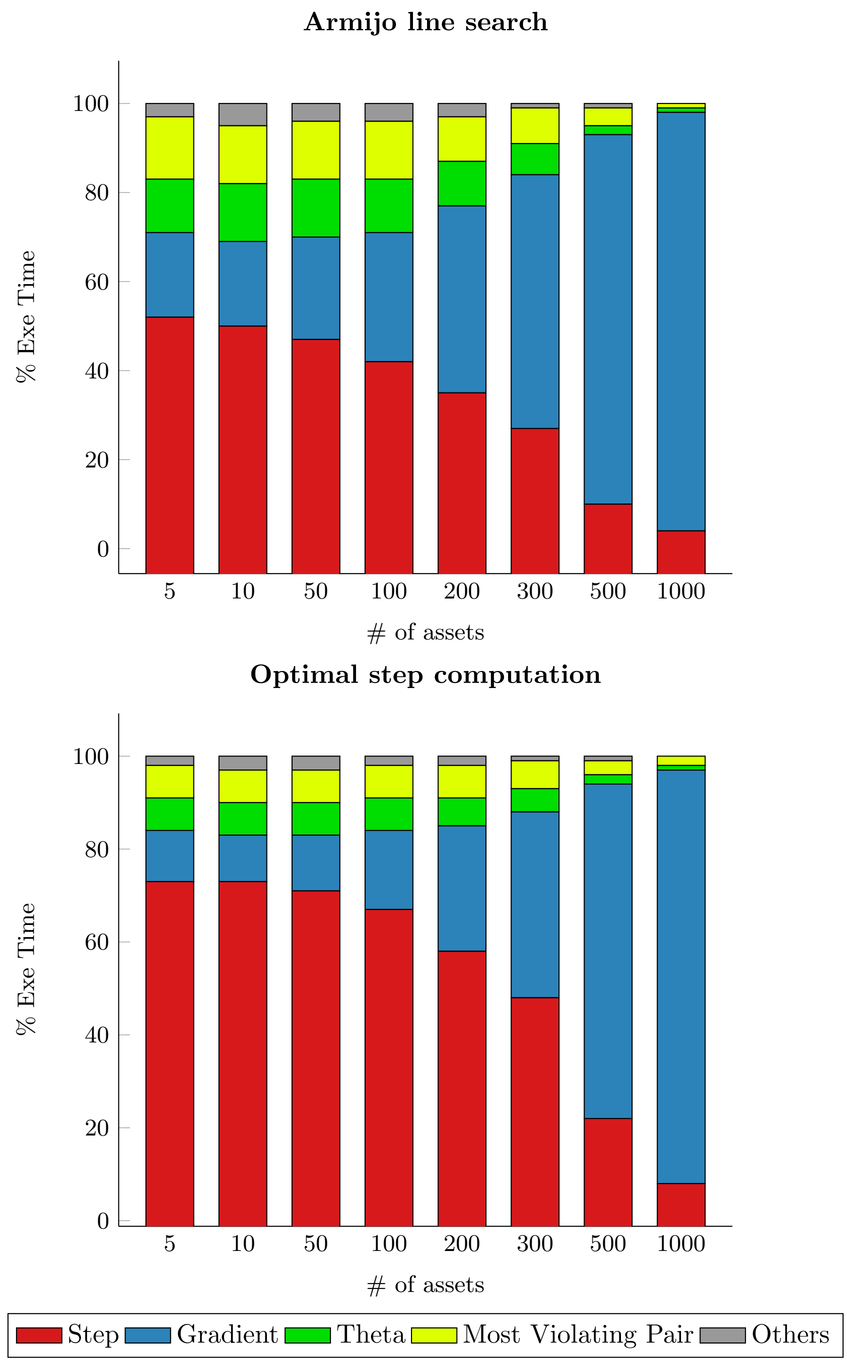
Table 2: $x^{(0)} = [1, 0, .., 0]$

6 Computational experiments

We compare the convergence properties of the algorithms in term of number of times that one algorithm converges to a stationary point that is the minimum point for the convex formulation.


Average execution time with $\epsilon = 10^{-6}$
Optimal Step
Armijo Step
SNOPT
Exe Time (s)
# of assets
0
10
20
30
40
50
200
400
600
800
1,000
1,200


Average iterations number with $\epsilon = 10^{-6}$
Optimal Step
Armijo Step
Iterations
# of assets
0
2,000
4,000
6,000
8,000
10,000
12,000
200
400
600
800
1,000
1,200


Armijo line search
% Exe Time
0
20
40
60
80
100
5
10
50
100
200
300
500
1000
# of assets
Optimal step computation
% Exe Time
0
20
40
60
80
100
5
10
50
100
200
300
500
1000
# of assets
Step
Gradient
Theta
Most Violating Pair
Others