数理解析研究所講究録
1114巻 1999年 67-73

Approximation of global optimal values of nonconvex programs using Successive Convex Relaxation Method

東京工業大学 数理・計算科学専攻　福田光浩 (Mituhiro Fukuda)
東京工業大学 数理・計算科学専攻　小島政和 (Masakazu Kojima)

1 Introduction

The computation of the global optima of nonconvex programs is a challenging issue both from the theoretical and practical aspects. Since 1998, Kojima and Tunçel have been proposing a new concept called Successive Convex Relaxation Method (SCRM) which allows to conceptually determine the global optima of Quadratic Optimization Problems (QOP) [5, 6]. Recently, Kojima, Matsumoto and Shida [4] observed that using an arithmetic transformation, the SCRM can also be applied to nonconvex programs. Let us consider an optimization problem of the form:

$$\begin{cases} \max & \boldsymbol{c}^T\boldsymbol{x} \\ s.t. & \boldsymbol{x} \in \mathcal{F} = \{\boldsymbol{x} \in I\!R^n : \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}\} \end{cases} \tag{1}$$

where $\boldsymbol{g} : I\!R^n \to I\!R^m$. The following condition is required for this problem.

Condition 1.1

(a) $\boldsymbol{g} \in \mathcal{C}^2$ *(or more weakly,* $\boldsymbol{g}$ *is peri-convex [4]);*

(b) $\mathcal{F}$ *is bounded.*

We observe that (1) includes a large class of smooth nonlinear nonconvex programs, as well as problems with integer constraints (imposing restriction like $x(x-1) = 0$ or $\sin(\pi x) = 0$).

This short note gives preliminary computational results of the SCRM which provides the global optimum value of (1), or at least an upper bound of this value. We chose some test problems from the literature [3, 1, 2] with small instances as a first experiment. With this study, the authors hope to gain some insights to develop more suitable algorithms for larger instances of nonlinear programs.

2 Successive Convex Relaxation Method

2.1 Transforming Nonlinear Programs into Quadratic Optimization Problems

Consider problem (1), and let us apply the transformation proposed in [4]. We will use the following notation. $I = \{1, 2, \cdots, m\}$, $C_I = \{i \in I : g_i(\cdot)$ is convex in $\mathcal{F}\}$, and

$N_I = I \backslash C_I$. For each $i \in N_I$, let $\sigma_i > 0$ be such that $g_i(\cdot) + \sigma_i || \cdot ||^2$ becomes convex at least in $\mathcal{F}$. This σ_i always exists under Condition 1.1 **(a)**. In practical matters, it is not easy to obtain σ_i for general nonconvex functions g_i's. Also, the magnitude of σ_i, in some sense, gives the nonconvexity of the function since it is the upper bound of the Euclidean norm of the Hessian matrix $\nabla^2 g_i(\cdot)$ in the prescribed feasible region.

Adding an artificial variable x_0 in (1), we can rewrite it as:

$$\begin{cases} \max & \boldsymbol{c}^T \boldsymbol{x} \\ s.t. & g_i(\boldsymbol{x}) \leq 0, & i \in C_I \\ & g_i(\boldsymbol{x}) + \sigma_i(||\boldsymbol{x}||^2 - x_0) \leq 0, & i \in N_I \\ & -||\boldsymbol{x}||^2 + x_0 \leq 0. \end{cases} \tag{2}$$

It is not difficult to see that the optimal value of (2) is always smaller then the optimal value of (1). In fact, it can be shown that solving (2), we obtain the global optimal solution of (1) (Theorem 2.1 [4]). If we call now $\mathcal{C}_1 \subseteq I\!R \times I\!R^n$ the set defined by the restrictions involving g_i's, *i.e.*,

$$\mathcal{C}_1 = \left\{ (x_0, \boldsymbol{x}) \in I\!R \times I\!R^n : \begin{array}{ll} g_i(\boldsymbol{x}) \leq 0, & i \in C_I \\ g_i(\boldsymbol{x}) + \sigma_i(||\boldsymbol{x}||^2 - \boldsymbol{x}_0) \leq 0, & i \in N_I \end{array} \right\},$$

(2) becomes a QOP, once $\mathcal{C}_1$ is convex. Therefore, we are actually in the original framework of the SCRM. In order to keep the notation, we redefine the variable $(x_0, \boldsymbol{x}) \in I\!R^{n+1}$ as $\boldsymbol{x}$, and $(0, \boldsymbol{c}) \in I\!R^{n+1}$ as $\boldsymbol{c}$, and we suppose $||\boldsymbol{c}|| = 1$ without loss of generality. We want to solve now the QOP

$$\begin{cases} \max & \boldsymbol{c}^T \boldsymbol{x} \\ s.t. & \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + 2\boldsymbol{q}^T \boldsymbol{x} + \gamma \leq 0, \\ & \boldsymbol{x} \in \mathcal{C}_1 \end{cases} \tag{3}$$

where $\boldsymbol{Q} = \begin{pmatrix} 0 & \boldsymbol{0}^T \\ \boldsymbol{0} & -\boldsymbol{I} \end{pmatrix}$, $\boldsymbol{q}^T = (\frac{1}{2}, \boldsymbol{0}^T)$, and $\gamma = 0$.

This problem is nonconvex since $\boldsymbol{Q}$ is negative semi-definite. However, we can rewrite the quadratic function as $\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + 2\boldsymbol{q}^T \boldsymbol{x} + \gamma = \boldsymbol{Q} \bullet \boldsymbol{X} + 2\boldsymbol{q}^T \boldsymbol{x} + \gamma$, with $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T$, where $\bullet$ is the inner-product in the space of $(n+1) \times (n+1)$-symmetric matrices ($\boldsymbol{A} \bullet \boldsymbol{B} = \Sigma_{i,j=1}^{n+1} a_{ij} b_{ij}$). If we disregard the equality $\boldsymbol{X} = \boldsymbol{x}\boldsymbol{x}^T$ and just suppose that $\boldsymbol{X}$ is symmetric, we have a linear relaxation of (3). The SCRM with linear relaxation version is based on this fact, and it can be shown that theoretical, a successive relaxation of this kind leads to the global optimal value of (3).

2.2 Successive Convex Relaxation Method

Since the theoretical details of the SCRM can be found in [5, 6], and its implementation issues in [7], we just attempt to consider the minimum necessary definitions and concepts to understand the algorithm here. Following the original notation in [6], we define two vector sets in $I\!R^{n+1}$. Given $\theta \in (0, \pi/2]$, let

$$\boldsymbol{D}_0 = \{\pm \boldsymbol{e}_1, \pm \boldsymbol{e}_2, \cdots, \pm \boldsymbol{e}_{n+1}\} \quad \text{and} \quad \boldsymbol{D}_1(\theta) = \{\boldsymbol{b}_i(\theta), \bar{\boldsymbol{b}}_i(\theta), \boldsymbol{c}, \quad 1 \leq i \leq n+1\}, \tag{4}$$

where $\{\boldsymbol{e}_i\}_{i=1}^{n+1}$ is the canonical basis in $\mathbb{R}^{n+1}$, $\boldsymbol{b}_i(\theta) = \dfrac{\boldsymbol{c}\cos\theta + \boldsymbol{e}_i \sin\theta}{||\boldsymbol{c}\cos\theta + \boldsymbol{e}_i \sin\theta||}$, and $\bar{\boldsymbol{b}}_i(\theta) = \dfrac{\boldsymbol{c}\cos\theta - \boldsymbol{e}_i \sin\theta}{||\boldsymbol{c}\cos\theta - \boldsymbol{e}_i \sin\theta||}$.

Theoretically speaking, we need in fact that $\boldsymbol{D}_1$ being a δ-net of a neighborhood $\boldsymbol{D}(\boldsymbol{c},\kappa) = \{\boldsymbol{v} \in \mathbb{R}^{n+1} : ||\boldsymbol{c}-\boldsymbol{v}|| \leq \kappa, \quad ||\boldsymbol{v}|| = 1\}$ of $\boldsymbol{c} \in \mathbb{R}^{n+1}$ for some $\kappa > 0$ [6]. Or more precisely, $\boldsymbol{D}_1$ is a δ-net of $\boldsymbol{D}(\boldsymbol{c},\kappa)$ if for all $\boldsymbol{v}' \in \boldsymbol{D}_1$, $||\boldsymbol{v}'|| = 1$, and if for all $\boldsymbol{v} \in \boldsymbol{D}(\boldsymbol{c},\kappa)$ there exists $\boldsymbol{v}' \in \boldsymbol{D}_1$ such that $||\boldsymbol{v} - \boldsymbol{v}'|| \leq \delta$. In this case, $\boldsymbol{D}_1$ will have necessary an exponential number of vectors, which will be impractical to be implemented. Since the above defined $\boldsymbol{D}_1(\theta)$ (4) is not the desired δ-net, we cannot expect a theoretical global convergence of the algorithm as mentioned in Theorem 2.2 below. On the other hand, we tried to compensate the lack of vectors in the neighborhood of vector $\boldsymbol{c}$ reconstructing $\boldsymbol{D}_1(\theta)$ several times for different values of θ's along the algorithm. In the following lines, we start to describe our algorithm. Let $\mathcal{C}$ a compact convex set, $\boldsymbol{u}, \boldsymbol{v} \in \boldsymbol{D}_0 \cup \boldsymbol{D}_1$, and consider

$$\begin{aligned} \alpha_{(\mathcal{C},\boldsymbol{v})} &= \max\{\boldsymbol{v}^T\boldsymbol{x} : \boldsymbol{x} \in \mathcal{C}\}, \\ r2sf(\boldsymbol{x};\mathcal{C},\boldsymbol{u},\boldsymbol{v}) &= -(\boldsymbol{u}^T\boldsymbol{x} - \alpha_{(\mathcal{C},\boldsymbol{u})})(\boldsymbol{v}^T\boldsymbol{x} - \alpha_{(\mathcal{C},\boldsymbol{v})}), \end{aligned}$$

and define the following set of quadratic functions

$$\begin{aligned} \mathcal{P}^2(\mathcal{C},\boldsymbol{D}_0) &= \{r2sf(\cdot;\mathcal{C},\boldsymbol{u},\boldsymbol{v}) : \boldsymbol{u},\boldsymbol{v} \in \boldsymbol{D}_0, \quad \boldsymbol{u} \neq \boldsymbol{v}\} \\ \mathcal{P}^2(\mathcal{C},\boldsymbol{D}_0,\boldsymbol{D}_1) &= \{r2sf(\cdot;\mathcal{C},\boldsymbol{u},\boldsymbol{v}) : \boldsymbol{u} \in \boldsymbol{D}_0, \quad \boldsymbol{v} \in \boldsymbol{D}_1\} \\ \mathcal{P}^F &= \{\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + 2\boldsymbol{q}^T\boldsymbol{x} + \gamma \text{ given in (3)}\}. \end{aligned}$$

In addition, let

$\mathcal{S}^{n+1}$	:	space of $(n+1)\times(n+1)$-symmetric matrices;
$\boldsymbol{E}_i$	:	matrix with all zeros except the (i,i) diagonal element with 1;
$qf(\cdot;\boldsymbol{Q},\boldsymbol{q},\gamma)$	:	quadratic functions in the form $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + 2\boldsymbol{q}^T\boldsymbol{x} + \gamma$;
ζ^*	:	the optimal value of problem (3).

We implemented the following version of the SCRM.

Algorithm 2.1

Step 0: *Let $\boldsymbol{D}_0$ and $\boldsymbol{D}_1(\theta)$ defined in (4); $\theta \in (0,\pi/2]$, $0 < \theta_{\min} < \theta$, $0 < \epsilon_0 << \epsilon$ and $\rho,\eta \in (0,1)$;*

Step 1: *Compute $\alpha_{(\mathcal{C}_1,\boldsymbol{v})} = \{\boldsymbol{v}^T\boldsymbol{x} : \boldsymbol{x} \in \mathcal{C}_1\}$, for $\boldsymbol{v} \in \boldsymbol{D}_0 \cup \boldsymbol{D}_1(\theta)$, and let $k = 1$ and $\zeta_0 = +\infty$ (notice that $\zeta_k = \{\boldsymbol{c}^T\boldsymbol{x} : \boldsymbol{x} \in \mathcal{C}_k\}$ was already computed since $\boldsymbol{c} \in \boldsymbol{D}_1(\theta)$);*

Step 2: *If $\left(\dfrac{\zeta_k - \zeta^*}{\max\{|\zeta_k,\epsilon_0\}} \leq \epsilon\right)$ or $(\theta \leq \theta_{\min})$, stop;*

Step 3: *If $\dfrac{\zeta_{k-1} - \zeta_k}{\max\{|\zeta_k|,1\}} \leq \rho$, reconstruct $\boldsymbol{D}_1(\theta)$ with $\theta := \eta\theta$;*

Step 4: *Compute $\alpha_{(\mathcal{C}_{k+1},\boldsymbol{v})} = \{\boldsymbol{v}^T\boldsymbol{x} : \boldsymbol{x} \in \mathcal{C}_{k+1}\}$ for $\boldsymbol{v} \in \boldsymbol{D}_0 \cup \boldsymbol{D}_1(\theta)$*

where

$$\mathcal{P}^k = \mathcal{P}^2(\mathcal{C}_k, \boldsymbol{D}_0) \cup \mathcal{P}^2(\mathcal{C}_k, \boldsymbol{D}_0, \boldsymbol{D}_1) \text{ and}$$

$$\mathcal{C}_{k+1} = \left\{ \boldsymbol{x} \in \mathcal{C}_1 : \begin{array}{ll} \exists \boldsymbol{X} \in \mathcal{S}^{n+1} \text{ such that} & \\ \boldsymbol{Q} \bullet \boldsymbol{X} + 2\boldsymbol{q}^T\boldsymbol{x} + \gamma \le 0, & \forall qf(\cdot; \boldsymbol{Q}, \boldsymbol{q}, \gamma) \in \mathcal{P}^F \cup \mathcal{P}^k \\ \boldsymbol{x}^T\boldsymbol{E}_i\boldsymbol{x} + \boldsymbol{E}_i \bullet \boldsymbol{X} \le 0, & 1 \le i \le n+1 \end{array} \right\};$$

Step 5: *Let $k = k + 1$, and go to Step 2;*

Note that we have to solve a convex programs to compute each $\alpha_{(\mathcal{C}_{k+1}, \boldsymbol{v})}$.

Algorithm 2.1 is the actually implemented algorithm. There are some slight modifications compared with the theoretical version [6] to improve the convergence. See also [7]. As mentioned previously, once $\boldsymbol{D}_1(\theta)$ was taken as a suitable δ-net of $\boldsymbol{D}(\boldsymbol{c}, \kappa)$ (which implies that Step 3 is unnecessary), we can talk about global convergence.

Theorem 2.2 *[6] Suppose that Condition 1.1 holds. Let $\kappa, \epsilon > 0$, and $\boldsymbol{D}_0$ given. Then, there exists a $\delta > 0$ such that if we take a δ-net $\boldsymbol{D}_1$ of $\boldsymbol{D}(\boldsymbol{c}, \kappa)$, then Algorithm 2.1 constructs sequences $\{\mathcal{C}_k\}_{k=1}^{+\infty}$ and $\{\zeta_k\}_{k=1}^{+\infty}$ such that:*

(a) $\mathcal{C} = \{\boldsymbol{c} \in \mathcal{C}_1 : \boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{q}^T\boldsymbol{x} + \gamma \le 0, \quad qf(\cdot; \boldsymbol{Q}, \boldsymbol{q}, \gamma) \in \mathcal{P}^F\} \subseteq \cdots \subseteq \mathcal{C}_{k+1} \subseteq \mathcal{C}_k$, *and* $\zeta^* \le \cdots \le \zeta_{k+1} \le \zeta_k, \quad k = 1, 2, \cdots;$

(b) *There exists an k such that $\zeta^* \le \zeta_k \le \zeta^* + \epsilon$.*

3 Computational Experiments

This section will provide preliminary computational experiments over some benchmark problems [3, 1, 2]. The program code was written in C++ with AMPL (version 19981109) interface in order to utilize the nonlinear program solver CONOPT (version 2.070B) for AMPL. The experiments were conducted at DEC Alpha (599 MHz, 1GB memory) under Digital UNIX.

Table 1 gives the parameters for Algorithm 2.1. We started to construct $\boldsymbol{D}_1(\theta)$ for $\theta = \frac{4}{9}\pi$ since we noticed that it is not always true that the algorithm converges for small θ's.

Table 1: Parameter for Algorithm 2.1

stopping criterion (relative error)	ϵ	0.0001
initial angle for $\boldsymbol{D}_1(\theta)$	θ	$\frac{4}{9}\pi$
rate for lack of improvement	ρ	0.001
decreasing rate of θ	η	0.3
minimal admissible angle	$\theta_{\min}$	$\frac{\pi}{180}$
computational zero	ϵ_0	1.0e-9

Table 2 gives the results of our experiments. The column "prob." gives the source and the problem number or subsection where the problem is described; the problem type is one of the follows:

quad : involving only quadratic or linear (objective or constraint) functions;
cub : involving cubic functions;
pow p : involving polynomials of degree at most p;
trig : involving trigonometric functions;
ln : involving logarithms;
minlp : mixed integer nonlinear program.

The columns "var.", "con." and "noncon." give the number of variables, the number or convex and nonconvex restrictions, respectively, when formulated as (1) (without considering the box constraints). We have added an extra variable for problems with nonlinear objective functions (the number with †); some equalities were transformed in two inequalities (each added constraint with ‡); a convex and a nonconvex restrictions were introduced to represent a variable $x \in \{0,1\}$ ($x(x-1) \leq 0$ and $-x(x-1) \leq 0$ — entries with $^+$). Problems with "m" had their original formulation changed. For instance, constraints like $\ln(x_1^2+1)$ where changed into $\ln(y)$ and $y = x_1^2+1$, which contributed greatly in the convergence (added variable or constraint with c). The relative errors were computed as in Step 2 of Algorithm 2.1. Finally, "subprob." are the number of convex programs we solved and "reconst." is the number of times we reconstructed $\boldsymbol{D}_1(\theta)$ (at most 4 times — see iteration numbers with $\star$).

Table 2: Preliminary numerical experiments for nonlinear programs

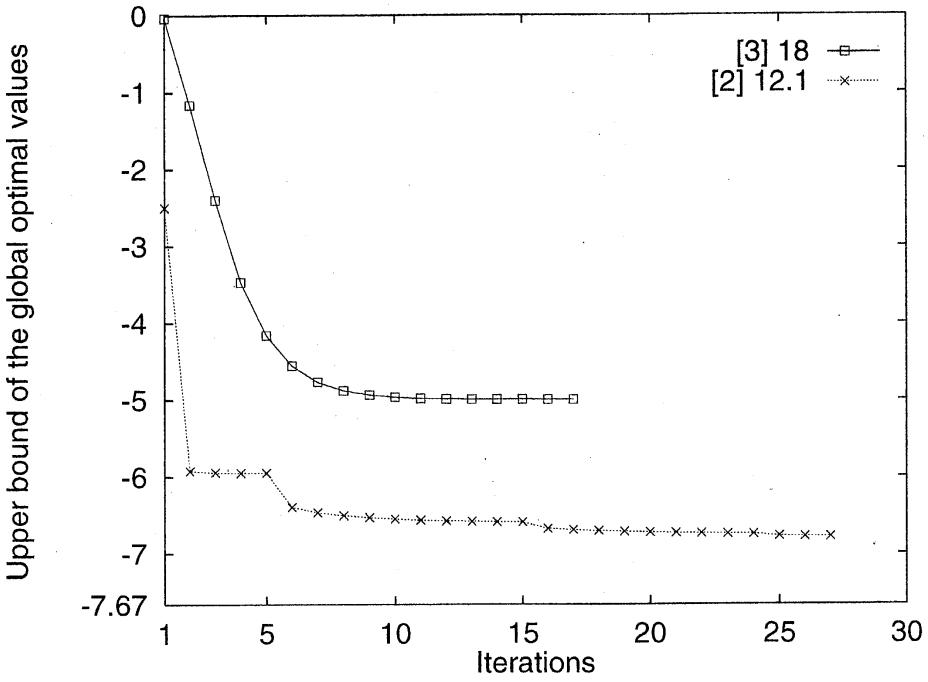
prob.	type	var.	con.	noncon.	iter.	CPU (s)	rel. error	subprob.	reconst.
[3] 5 m	trig	$3^{\dagger}$	0	$1^{\dagger}$	12	11.1	0.00006	181	1
[3] 6	quad	$3^{\dagger}$	$2^{\dagger\ddagger}$	1	1	0.0	0.00000	1	0
[3] 7 m	ln	$4^{\dagger c}$	2^{c}	2^{c}	4	4.9	0.00004	77	0
[3] 13	cub	$3^{\dagger}$	$1^{\dagger}$	1	68$\star$	40.8	0.09147	1021	4
[3] 18	quad	$3^{\dagger}$	$1^{\dagger}$	2	17	10.9	0.00006	256	2
[3] 23	quad	$3^{\dagger}$	$2^{\dagger}$	4	4	2.1	0.00000	61	0
[3] 30	quad	$4^{\dagger}$	$1^{\dagger}$	1	1	0.0	0.00000	1	0
[3] 31	quad	$4^{\dagger}$	$1^{\dagger}$	1	7	11.3	0.00003	134	1
[3] 42	quad	$4^{\dagger}$	$2^{\dagger\ddagger}$	1	1	0.0	0.00000	1	0
[3] 61	quad	$4^{\dagger}$	$3^{\dagger\ddagger\ddagger}$	2	1	0.0	0.00000	1	0
[1] 3.4	quad	3	2	1	86$\star$	80.1	0.25397	1463	4
[1] 4.6	pow 4	2	0	2	54$\star$	26.0	0.16088	703	4
[1] 4.7	pow 4	$3^{\dagger}$	$2^{\dagger\ddagger}$	1	1	0.0	0.00000	1	0
[2] 12.1	minlp	5	$4^{\ddagger+++}$	$3^{\ddagger\ddagger+++}$	27$\star$	149.1	0.11564	676	4
[2] 12.2	minlp	$4^{\dagger}$	$4^{\dagger+}$	2^{+}	2	1.5	-0.00184	39	0

Notice that the SCRM obtained the global optimum value solving exactly one convex problem for some benchmark problems. However, we encountered 4 problem which the method did not converge to the global optima.

Figure 1 gives a typical behavior of the upper bound values along the major iterations of Algorithm 2.1. In particular, it shows the upper bounds of the global optimal value of [3] 18 (optimal value -5) and [2] 12.1 (optimal value -7.67).

As we observe, the convergence slows downs in the latter iterations. Therefore, it seems reasonable to stop the algorithm after the first 10 iterations if we want a rough

Figure 1: Upper bound of the optimal values for [3] 18 and [2] 12.1



approximation of the global optimal value, for instance, embedded in a branch-and-bound framework. Although it is still not clear, one of the reasons of non-convergence for some problems may be due to numerical errors.

4 Conclusion

This paper gives the first implementation of the SCRM using linear relaxation for non-convex programs based on the work of Kojima and Tunçel, and Kojima, Matsumoto and Shida. This method is quite powerful since it gives the upper bounds of global optimal values of smooth nonlinear programs, and in some cases, the global optimal values themselves. At a first glance, the method seems promising. Though the numerical experiments shows that we still need further work to deal with larger instances of difficult nonlinear programs.

References

[1] C. A. Floudas and P. M. Pardalos, *A collection of test problems for constrained global optimization algorithms*, Lecture Notes in Compute Science, (Springer-Verlag, Berlin, 1990).

[2] C. A. Floudas *et al.*, *Handbook of Test Problems in Local and Global Optimization* (Kluwer Academic Publishers, Dordrecht, 1999).

[3] W. Hock and K. Schittkowski, *Test examples for nonlinear programming code*, Lecture Notes in Economics and Mathematical Systems, (Springer-Verlag, Berlin, 1981).

[4] M. Kojima, T. Matsumoto and M. Shida, "Moderate nonconvexity = convexity + quadratic concavity," Research Report B-348, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Japan. March 1999, revised April 1999.

[5] M. Kojima and L. Tunçel, "Cones and matrices and successive convex relaxations of nonconvex sets". To appear in *SIAM Journal on Optimization.*

[6] M. Kojima and L. Tunçel, "Discretization and localization in successive convex relaxation methods for nonconvex quadratic optimization," Research Report B-341, Department of Mathematical and Computing Sciences, Tokyo Institute of Technology, Japan. Also issued as CORR98-34, Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Canada. July 1998, revised July 1999.

[7] A. Takeda, Y. Dai, M. Fukuda and M. Kojima, "Towards implementations of successive convex relaxation methods for nonconvex quadratic optimization problems". To appear in the *Proceedings of Approximation and Complexity in Numerical optimization: Continuous and Discrete Problems.*