A RANDOMIZED METHOD FOR SOLVING SEMIDEFINITE PROGRAMS

B. T. Polyak P. S. Shcherbakov

Trapeznikov Institute of Control Science,
Russian Academy of Sciences, Moscow, Russia

Abstract: Proposed is a novel iterative method for solving semidefinite programs. It exploits the ideology of cutting hyperplane through the center of mass of a convex body. To estimate the center of mass, we use a random walk technique known as the Hit-and-Run algorithm. The results of numerical simulations are compared to those obtained with presently available approaches. Robust versions of the method are considered, where the coefficient matrices contain norm-bounded uncertainties.

Keywords: linear matrix inequalities, optimization, random walk, center of mass.

1. INTRODUCTION AND OVERALL SCHEME OF THE METHOD

In this section we formulate the problem under consideration and present a brief schematic description of the main ideas underlying our method.

Considered is the standard semidefinite program (SDP) of the form

$$\min c^{\mathsf{T}} x \quad \text{s.t.} \quad A(x) \doteq A_0 + \sum_{i=1}^{n} x_i A_i \;\leq\; 0, \quad (1)$$

where $c \in \mathbb{R}^n$ and $A_i \in \mathbb{R}^{m\times m}$, $i = 0, \ldots, n$, are known symmetric matrices; the notation $A \leq 0$ stands for negative semidefiniteness of the matrix A. The constraint inequality in (1) is called a *linear matrix inequality* (LMI), and the convex set

$$D_{feas} = \{x \in \mathbb{R}^n : A(x) \leq 0\}$$

is referred to as the *feasible domain* of this LMI.

This problem is known to be one of the key problems in the theory of linear matrix inequalities (Boyd *et al.*, 1994). It has numerous applications in various fields of system theory and control, and at present there exist efficient solution techniques based on interior-point methods; e.g., see (Nesterov and Nemirovskii, 1994).

Inspired by the recent results in the rapidly developing area of randomized methods in control system analysis and design (Tempo *et al.*, 2005), we propose a novel approach to solving problem (1), which is based on totally different ideas. The iterative method that we developed leans on *random walks*, estimation of the *center of mass* of a convex set—the feasible domain D_{feas}, and uses the new notion of *boundary oracle* for LMIs.

The cornerstone components of our approach are

(a) the cutting hyperplane ideology, which is used to compute iteratively a sequence of embedded subdomains $D_{k+1} \subset D_k \subset D_{feas}$ and monotonically decrease the value of the objective function $f(x) = c^{\mathsf{T}} x$;

(b) the so-called Hit-and-Run (HR) algorithm for estimating the center of mass of D_k required in item **(a)** above;

(c) a boundary oracle which is needed for implementation of the HR-algorithm.

Specifically, let $D_k \subset D_{feas}$ be the domain obtained at the kth step of the iterative method

under consideration. For simplicity, it is assumed that D_{feas} is bounded in order to guarantee the boundedness of D_k. Using HR-algorithm, we generate N_{hr} random points distributed approximately uniformly on D_k and adopt their average x^k as an estimate of the center of mass of D_k. The point x^k might as well be taken to compute the current estimate $f^k = c^{\mathsf{T}}x^k$ of the objective function. Next, the hyperplane

$$H_k = \{x \in \mathbb{R}^n\colon\ c^{\mathsf{T}}(x - x^k) = 0\}$$

is drawn to cut off the "idle" portion of D_k thus reducing it to

$$D_{k+1} = \{x \in \mathbb{R}^n : x \in D_k,\ c^{\mathsf{T}}x \le c^{\mathsf{T}}x^k\} \subset D_k,$$

and the process is repeated with the set D_{k+1}. In other words, the convex set D_{k+1} is bounded by the LMI constraints in (1) and the half-space $\{x \in \mathbb{R}^n : c^{\mathsf{T}}(x - x^k) \le 0\}$ defined by the hyperplane H_k. Schematically, the behavior of the method is represented in Fig. 1 for the two-dimensional case where the vector c is taken in the form $c = (0,\ 1)^{\mathsf{T}}$.

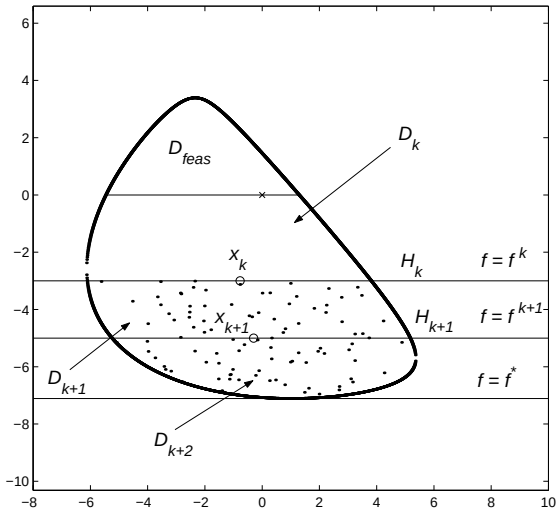

Fig. 1. Schematic representation of the method.

Under the assumption that x^k is a reasonably accurate estimate of the true center of mass, the lemma in (Radon, 1916) on the measures of symmetry of convex bodies is used to obtain an estimate on the *guaranteed rate* of decrease in the objective function. Thus, the method is expected to have a geometric convergence rate. Note that by no means do we intend to estimate the optimal point $x^* = \arg\min_{x \in D_{feas}} c^{\mathsf{T}}x$, but rather evaluate the optimal value f^* of the objective function.

2. CORNERSTONES

We now present the techniques and results underlying each of the items **(a)**–**(c)** above.

2.1 Cutting hyperplane

The implementation of item **(a)** and the guaranteed reduction of the value of the objective function is based on the following lemma.

Lemma 1 (Radon, 1916). *Let $D \subset \mathbb{R}^n$ be a convex bounded body and $g \in D$ be its center of mass. Denote by H an arbitrary $(n-1)$-dimensional hyperplane through g, and let H_1 and H_2 be the two hyperplanes supporting to D and parallel to H. Denote by*

$$r(H) \ \doteq\ \frac{\min\{\mathbf{dist}(H, H_1),\ \mathbf{dist}(H, H_2)\}}{\max\{\mathbf{dist}(H, H_1),\ \mathbf{dist}(H, H_2)\}}$$

the ratio of the distances from H to H_1 and H_2, respectively. Then

$$\min_H r(H) \ \ge\ \frac{1}{n}\,.$$

As applied to the setup in this paper, let H_1 and H denote the two hyperplanes through the two successive iterations x^k and x^{k+1} of the method, and let H_2 be the supporting hyperplane through the optimal point x^*. Assuming that the *exact* value of the center of mass is known, the following estimate is readily available:

$$f^{k+1} - f^* \ \le\ \varkappa(f^k - f^*), \quad \varkappa = \frac{n}{n+1}\,,$$

where f^* is the optimal value of the objective function and $f^k = c^{\mathsf{T}}x^k$ is the estimate obtained at the kth iteration.

Notably, to the best of our knowledge, this result has never been used in optimization, in contrast to similar results on the guaranteed *volumetric* reduction, which are typical to various modifications of the ellipsoid method.

2.2 Hit-and-Run algorithm

We now describe the Hit-and-Run algorithm, an efficient randomized technique exploited in this paper in the estimation of the center of mass of the sets D_k. This random walk algorithm originally proposed in (Smith, 1984) is simple to describe. It applies to a bounded convex body $D \in \mathbb{R}^n$ and returns a random point z having approximately uniform distribution on D. The arithmetic mean of this distribution is then adopted as an estimate of the center of mass.

Specifically, an initial point z^0 in the interior of D is selected, and let z^j be the point obtained at the jth step of the algorithm. A random direction y is generated (say, in the form $\xi/\|\xi\|$, where ξ is a Gaussian random vector with zero mean and identity covariance matrix, and $\|\cdot\|$ is the euclidean vector norm). The 1D-line $z^j + \lambda y$ is considered

and the points $\underline{z}^j$ and $\overline{z}^i$ of its intersection with the boundary of D are computed. The next-step point z^{j+1} is then generated randomly uniformly on the chord $[\underline{z}^j, \overline{z}^j]$.

In (Smith, 1984; Lovász, 1999) it has been shown that the sequence of random vectors $\{z^i\}_1^{N_{hr}}$ generated in such a way forms a discrete Markov chain having the property of uniform ergodicity. In other words, the distribution of the random vector z^i tends to the limiting uniform distribution on D with geometric rate; this property is referred to as *fast mixing*. The mixing rate depends on the shape of the set D and on the "position" of the initial point. The best results are obtained if the distribution of z^0 is close to the uniform (the so-called *warm start* of the HR-algorithm), and the set D is *isotropic*, i.e., it has "equal dimensions" along all directions.

There exist other random walk techniques; the HR-algorithm is chosen here because it is simple to implement, produces practically reasonable approximations to the uniform distribution, requires minimum a priori information about the set D, and can also be applied to nonconvex sets under mild extra assumptions.

It should be noted that the HR-algorithm has been first applied to solving convex optimization problems in (Bertsimas and Vempala, 2004), however the overall method in that paper essentially differs from the one proposed here.

2.3 Boundary oracle

To perform the HR-algorithm over D_k, we need to efficiently compute the intersections of a 1D-line and the boundary of D_k, which is accomplished via use of the semidefinite boundary oracle developed in (Polyak and Shcherbakov, 2006). The core of this oracle is the lemma below.

Lemma 2 (Polyak and Shcherbakov, 2006). *Let $A < 0$ and $B = B^{\mathsf{T}}$, then the minimal and the maximal values of the parameter $\lambda \in \mathbb{R}$ retaining the negative definiteness of the matrix $A+\lambda B$ are given by*

$$\underline{\lambda} \;=\; \begin{cases} \max\limits_{\lambda_i<0} \lambda_i, & \\ \\ -\infty, \qquad \textit{if all} \quad \lambda_i > 0; \end{cases} \tag{2}$$

and

$$\overline{\lambda} \;=\; \begin{cases} \min\limits_{\lambda_i>0} \lambda_i, & \\ \\ +\infty, \qquad \textit{if all} \quad \lambda_i < 0, \end{cases} \tag{3}$$

where λ_i are the generalized eigenvalues of the pair of matrices A and $-B$, i.e., $Ae_i = -\lambda_i B e_i$.

With this lemma, the desired endpoints of the chord $[\underline{z}^j, \overline{z}^j]$ are computed as $\underline{z}^j = z^j + \underline{\lambda} y$ and $\overline{z}^j = z^j + \overline{\lambda} y$.

In the setup of this paper, assume that $z^j \in D_{feas}$ and $y \in \mathbb{R}^n$ is a (random) direction. We have

$$A(z^j + \lambda y) \;=\; A(z^j) + \lambda \sum_{i=1}^{n} y_i A_i \;\doteq\; A + \lambda B,$$

and using Lemma 2, the desired intersection points of the line and the boundary of D_{feas} are given by $\underline{z}^j = z^j + \underline{\lambda} y$ and $\overline{z}^j = z^j + \overline{\lambda} y$.

To compute the points of intersection with the boundary of the set D_k, the additional linear condition $c^{\mathsf{T}}(x - x^{k-1}) \le 0$ defining D_k is to be taken into account, which is straightforward to implement.

As seen from the aforesaid, this operation should be frequently performed in the process of iterations. The basis of this operation is finding the eigenvalues of matrices, which is efficiently implemented in MATLAB as the `eig` routine; hence, the boundary oracle for LMIs is accurate and "cheap" enough for matrices of quite large dimensions, which is confirmed by numerical simulations.

Generalizations. Together with the HR-algorithm, the semidefinite boundary oracle can be exploited in a wide range of problems. We briefly describe some of its applications to the problems where the sets D_{feas} are defined by other types of matrix constraints encountered in optimization and control.

Often, the LMI constraints in problem (1) are formulated in terms of a matrix (rather than vector) variable; e.g. in the form of the matrix Lyapunov inequality

$$C + A^{\mathsf{T}}X + XA \le 0, \qquad X > 0, \tag{4}$$

where $C = C^{\mathsf{T}}$ and A are given, $X = X^{\mathsf{T}} > 0$ is a matrix variable, and the goal is to minimize the function $\operatorname{tr} X$ subject to (4). Such a setup is typical to quadratic stabilization of dynamic systems.

We formulate the boundary oracle for this problem; i.e. assuming that (4) holds for some $X > 0$ and letting $Y = Y^{\mathsf{T}}$ be an $m \times m$ increment (direction), we find the minimum and maximum values of λ such that (4) is satisfied with $X + \lambda Y$. We have

$$\begin{aligned} C + A^{\mathsf{T}}(X + \lambda Y) + (X + \lambda Y)A = & \\ = \underbrace{C + A^{\mathsf{T}}X + XA}_{A_0} + \lambda \underbrace{(A^{\mathsf{T}}Y + YA)}_{A_1} & \\ \doteq A_0 + \lambda A_1, & \end{aligned} \tag{5}$$

where $A_0 \le 0$ and $A_1 = A_1^{\mathsf{T}}$. Hence, the problem reduces to the one considered in Lemma 2. The only difference is that we have to account for the constraint $X + \lambda Y > 0$, which is easy, since it is of the same form. Finally, let $\Lambda = [\underline{\lambda},\ \overline{\lambda}]$ be the segment of sign-definiteness in (5) and $\Lambda_1 = [\underline{\lambda}_1,\ \overline{\lambda}_1]$ be that of sign-definiteness for $X + \lambda Y$; then $\lambda \in \Lambda \cap \Lambda_1$.

Another type of constraints often encountered in control is quadratic in X; for example, they might have the form of the matrix Riccati inequality:

$$C + A^{\mathsf{T}}X + XA + XBB^{\mathsf{T}}X \le 0, \qquad X > 0.$$

Let it hold for an $X > 0$, and let $Y = Y^{\mathsf{T}}$ be an $m \times m$ increment; consider the matrix $X + \lambda Y$. Denoting $M(X)$ the matrix on the right-hand side of the quadratic inequality above, similarly to (5) we obtain

$$M(X + \lambda Y) = G + \lambda H + \lambda^2 F F^{\mathsf{T}},$$

where

$$\begin{aligned} G &= M(X) \le 0, \\ H &= XBB^{\mathsf{T}}Y + YBB^{\mathsf{T}}X = H^{\mathsf{T}}, \\ F &= YB. \end{aligned}$$

Using Schur lemma, the matrix inequality $G + \lambda H + \lambda^2 F F^{\mathsf{T}} \le 0$, which is quadratic in λ, reduces to the linear one

$$\begin{pmatrix} G + \lambda H & \lambda F \\ \lambda F^{\mathsf{T}} & -I \end{pmatrix} \le 0,$$

i.e.,

$$\begin{pmatrix} G & 0 \\ 0 & -I \end{pmatrix} + \lambda \begin{pmatrix} H & F \\ F^{\mathsf{T}} & 0 \end{pmatrix} \doteq A_0 + \lambda A_1 \le 0,$$

so that (at the expense of increasing dimension by a factor of two) we are within the old setup.

3. NUMERICAL EXPERIMENTS

Various modifications of the method leading to essential acceleration of the basic scheme were used in the numerical experiments. Most attention has been paid to the practical implementation of the HR-algorithm. In particular, since $D_{k+1} \subset D_k$, economy sampling schemes can be applied which use those of the HR-points generated at the previous iteration, which fall in the D_{k+1} (so-called *reuse* techniques; e.g., see (Chen *et al.*, 2004)). Other modifications include various types of averaging the HR-points, different from simple arithmetic mean; use of by-product boundary points; selecting the initial point, etc. These lead to a more accurate evaluation of the center of mass and speed up the realization of the HR-algorithm at every step of the method.

One of the most crucial issues associated with the implementation of the HR-algorithm is worth mentioning in more detail. Typically, as the method approaches the optimum, the sets D_k become "skinny" in the direction c, and the isotropic property vanishes. As a result, the HR-algorithm exhibits bad mixing and tends to stick inside certain subdomains of D_k, because most of the randomly generated vectors ξ (see Section 2.2) tend to be directed "across" the D_k, not "along" the set. To avoid such effects, it is suggested to dilate the set D_k in order to make it isotropic. Such a dilation can be performed using certain linear transformation; e.g., see (Bertsimas and Vempala, 2004)). Specifically, having generated N_{hr} HR-points z^i in the set D_{k-1}, we compose the following *sampled covariance matrix*:

$$W = \frac{1}{N_{hr}-1} \sum_{i=1}^{N_{hr}} \big(z^i - \hat{z}\big)\big(z^i - \hat{z}\big)^{\mathsf{T}},\ \hat{z} = \frac{1}{N_{hr}} \sum_{i=1}^{N_{hr}} z^i.$$

This matrix reconstructs the shape of D_{k-1} from the available information, and the direction vector for the HR-algorithm at the next step is taken in the form $\eta = W^{1/2}\xi$, where $\xi \in \mathbb{R}^n$ is uniformly distributed on the surface of the unit hypersphere. In other words, since the shape of D_k is "similar" to that of D_{k-1}, the directions η are generated uniformly on the ellipsoid which approximates the skinny shape of D_k; this leads to a much better mixing and a more accurate evaluation of the center of mass even for highly shrunk domains. As a result, the method attains considerably higher accuracy.

In the experiments, the method demonstrated very stable performance over a number of randomly generated test problems with the dimensions of the A_i matrices as large as $m = 100$, and the dimensions of the design vector x as high as $n = 300$. As far as the widely used computer MATLAB-based realizations of interior-point methods (the `solvesdp` routine in SeDuMi Toolbox) are concerned, our method has showed comparable performance, sometimes exceeding the classical methods in accuracy.

To illustrate, we briefly describe some results of simulations.

The first set of experiments was conducted with SDP problems having dimensions $n = 300$, $m = 10$. Typically, the method produces 7 to 8 exact decimal digits for the function value after 15 iterations (averaging in the HR-algorithm was performed over $N_{hr} = 2,000$ points). The `solvesdp` routine typically exhibits slightly lower accuracy (6 to 7 digits), and sometimes it yields an infeasible point x^*, i.e. $\lambda_{\max}\big(A(x^*)\big) \approx 10^{-7} > 0$.

In the second set of experiments, we took $n = 10$ and $m = 100$ as the matrix dimension. Using only $N_{hr} = 200$ HR-points usually leads to 9 exact digits after 15 to 20 steps, and the estimated

convergence rate $\varkappa$ (see Section 2.1) is greatly exceeded.

Finally, we mention the "worst-case" geometry SDP problems, where the feasible domain has simplicial form. In that case, the estimate given by Lemma 1 is attained, which is confirmed by the experiments.

4. ROBUST VERSIONS

Among very important extensions of the method is its modification to the case where the matrices A_i are uncertain, $A_i = A_i^0 + \Delta_i$, and $\Delta_i = \Delta_i^{\mathsf{T}}$ are bounded in the spectral norm. Namely, the *robust* statement of problem (1) that we consider is to minimize $c^{\mathsf{T}}x$ over the *robustly feasible* domain

$$D_{feas}^{rob} = \big\{x \in \mathbb{R}^n\colon\ A(x,\Delta) \le 0\ \forall \text{ admissible } \Delta\big\},$$

where

$$A(x,\Delta) \doteq A_0^0 + \Delta_0 + \sum_{i=1}^n x_i(A_i^0 + \Delta_i),$$

and by admissibility of uncertainty Δ we mean

$$\|\Delta_i\| \le \varepsilon_i, \quad i = 1,\dots n,$$

for some specified $\varepsilon_i \ge 0$.

In the literature, there are only limited results on robust statements of the SDP problem, e.g., see (El-Ghaoui *et al.*, 1998; Ben-Tal and Nemirovski, 2002).

For such a setup, we developed the *robust semidefinite oracle* which makes efficient the robust version of the method proposed. This robust oracle is based on the lemma below on the radius of nonsingularity for symmetric matrices, cf. (Polyak, 2003).

Lemma 3 (Polyak and Shcherbakov, 2006). *For a nonsingular symmetric matrix $M \in \mathbb{S}^{m\times m}$, its symmetric radius of nonsingularity defined as*

$$\rho(M) \doteq \inf\{\|P\|\colon P \in \mathbb{S}^{m\times m},\ M+P \text{ is nonsingular}\}$$

is given by

$$\rho(M) \ =\ 1/\|M^{-1}\| \ =\ \min_i |\lambda_i(M)|.$$

Using this result, an efficient procedure described below can be devised for computing the intersection points of a 1D-line with the boundary of D_{feas}^{rob}, and the HR-algorithm can be performed over D_{feas}^{rob}.

Namely, let $x \in D_{feas}^{rob}$ be a robustly feasible point and $y \in \mathbb{R}^n$ be a direction. We consider the line $x + \lambda y$ and find the quantities $\underline{\lambda}^{rob}, \overline{\lambda}^{rob}$, the minimal and maximal values of λ retaining the negative definiteness of the matrix $A(x+\lambda y, \Delta)$ for all admissible Δ. We have

$$A(x+\lambda y,\Delta) \ =\ \hat{A}(\lambda) + \Delta(\lambda),$$

where

$$\hat{A}(\lambda) = A_0 + \sum_{i=1}^n (x_i + \lambda y_i)A_i,$$

$$\Delta(\lambda) = \Delta_0 + \sum_{i=1}^n (x_i + \lambda y_i)\Delta_i,$$

and according to Lemma 3, the matrix $\hat{A}(\lambda) + \Delta(\lambda)$ remains nonsingular (hence, negative definite) for all admissible perturbations Δ satisfying

$$\big\|\big(\hat{A}(\lambda)\big)^{-1}\big\| \ <\ \frac{1}{\|\Delta(\lambda)\|}\,.$$

Since the perturbations independently sweep their respective domains of uncertainty, the estimate

$$\begin{aligned}\|\Delta(\lambda)\| \ &\le\ \|\Delta_0\| + \sum_{i=1}^n |x_i + \lambda y_i|\,\|\Delta_i\| \\ &=\ \varepsilon_0 + \sum_{i=1}^n |x_i + \lambda y_i|\,\varepsilon_i\end{aligned}$$

is sharp. Therefore, by considering the two scalar functions

$$\begin{aligned}\varphi(\lambda) \ &=\ \Big\|\Big(A_0 + \sum_{i=1}^n (x_i+\lambda y_i)A_i\Big)^{-1}\Big\|; \\ \varepsilon(\lambda) \ &=\ \frac{1}{\varepsilon_0 + \sum\limits_{i=1}^n |x_i + \lambda y_i|\,\varepsilon_i}\,,\end{aligned} \tag{6}$$

the segment $[\underline{\lambda}^{rob},\ \overline{\lambda}^{rob}]$ of robust negative semidefiniteness of the family $A(x+\lambda y,\Delta)$ can be computed numerically as $\{\lambda\colon\ \varphi(\lambda) \le \varepsilon(\lambda)\}$.

Clearly, the inclusion $[\underline{\lambda}^{rob},\ \overline{\lambda}^{rob}] \subset [\underline{\lambda},\ \overline{\lambda}]$ is valid, where $\underline{\lambda}, \overline{\lambda}$ are the critical values of the parameter λ in the perturbation-free case (i.e., the minimal and maximal values of λ retaining the negative definiteness of $A(x+\lambda y, 0)$). Hence, checking the condition $\varphi(\lambda) \le \varepsilon(\lambda)$ should only be performed over the segment $[\underline{\lambda},\ \overline{\lambda}]$, and we arrive at the following *robust boundary oracle.*

Lemma 4. *Let $A(x,0) < 0$. For any $y \in \mathbb{R}^n$, the maximal and minimal values of λ retaining the negative definiteness of the matrix $A(x+\lambda y,\Delta)$ for all admissible perturbations Δ are given by the two solutions of the equation $\varphi(\lambda) = \varepsilon(\lambda)$ (6) on the interval $[\underline{\lambda},\ \overline{\lambda}]$ (2)–(3).*

It is this boundary oracle that constitutes component **(c)** of the method; the rest of the components remain the same as in the non-robust setting.

Experiments. Experiments conducted with the robust version of the method showed reasonable accuracy and rate of convergence for randomly generated coefficient matrices A_i. For example, in a typical low-dimensional problem with $n = 2$, $m = 3$, the method yields 5 to 6 exact decimal digits after 10 iterations.

The main numerical difference from the non-robust problem is that the nonlinear equation (6) has to be solved at every step of the HR-algorithm (the standard MATLAB routine `fsolve` was used). As a result, every step of the method requires 5 to 20 times as much cpu time as compared to the non-robust version (depending on the dimensions of the SDP problem).

5. CONCLUSION

The randomized optimization approach presented here is based on novel yet transparent ideas. Importantly, it admits generalizations to other convex optimization problems such as those specified by linear and quadratic algebraic inequalities, quadratic matrix inequalities, conic programming, etc.

Although we do not provide a rigorous *theoretical* justification of the method, preliminary numerical tests have shown its quite competitive performance in the original setup (1) and a reasonable behavior in the robust formulation. It is felt that a promising direction for future research would be development of alternative randomization routines different from HR, which might lead to *provable* versions of the method.

REFERENCES

Ben-Tal, A. and A. Nemirovski (2002). On tractable approximations of uncertain linear matrix inequalities affected by interval uncertainty. *SIAM J. Optimiz.* **12**(3), 811–833.

Bertsimas, D. and S. Vempala (2004). Solving convex programs by random walks. *J. ACM* **51**(4), 540–556.

Boyd, S. P., L. El Ghaoui, E. Feron and V. Balakrishnan (1994). *Linear Matrix Inequalities in System and Control Theory.* SIAM. Philadelphia.

Chen, X., K. Zhou and J. Aravena (2004). Fast construction of robustness degradation function. *SIAM J. Control Optimiz.* **46**(6), 1960–1971.

El-Ghaoui, L., F. Oustry and H. Lebret (1998). Robust solutions to uncertain semidefinite programs. *SIAM J. Optimiz.* **9**(1), 33–52.

Lovász, L. (1999). Hit-and-run mixes fast. *Math. Progr.* **86**, 443–461.

Nesterov, Yu. and A. Nemirovskii (1994). *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM. Philadelphia.

Polyak, B. T. (2003). Robust linear algebra and robust aperiodicity. In: *Directions in Mathematical Systems Theory and Optimization* (A. Rantzer and C. I. Byrnes, Eds.). pp. 249–260. Springer.

Polyak, B. T. and P. S. Shcherbakov (2006). The D-decomposition technique for linear matrix inequalities. *Automat. Remote Control* **67**(11), 159–174.

Radon, J. (1916). Über eine Erweiterung des Begriffs der konvexen Functionen, mit einer Anwendung auf die Theorei der konvexen Körper. *S.– B. Akad. Wiss. Wien* **125**, 241–258.

Smith, R. L. (1984). Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research* **32**(6), 1296–1308.

Tempo, R., G. Calafiore and F. Dabbene (2005). *Randomized Algorithms for Analysis and Control of Uncertain Systems.* Springer-Verlag. London.