Chapter

Algorithms for LQR via Static Output Feedback for Discrete-Time LTI Systems

Yossi Peretz

Abstract

Randomized and deterministic algorithms for the problem of LQR optimal control via static-output-feedback (SOF) for discrete-time systems are suggested in this chapter. The randomized algorithm is based on a recently introduced randomized optimization method named the Ray-Shooting Method that efficiently solves the global minimization problem of continuous functions over compact non-convex unconnected regions. The randomized algorithm presented here has a proof of convergence in probability to the global optimum. The suggested deterministic algorithm is based on the gradient method and thus can be proved to converge to local optimum only. A comparison between the algorithms is provided as well as the performance of the hybrid algorithm.

Keywords: control systems, optimal control, discrete-time systems, state-space models, NP-hard control problems, randomized algorithms, deterministic algorithms

1. Introduction

The application of static-output-feedbacks (SOFs) for linear-quadratic regulators (LQR) is very attractive, since they are cheap and reliable and their implementation is simple and direct, because their components has direct physical interpretation in terms of sensors amplification rates and actuator activation power. Moreover, the long-term memory of dynamic feedbacks is useless for systems subject to random disturbances, to fast dynamic loadings or to random bursts and impulses, and the application of state feedbacks is not always possible due to unavailability of full-state measurements (see, e.g., [1]). Also, the use of SOF avoids the need to reconstruct the state by Kalman filter or by any other state reconstructor.

On the other hand, in practical applications, the entries of the needed SOFs are bounded, and since the problem of SOFs with interval constrained entries is NP-hard (see [2, 3]), one cannot expect the existence of a deterministic efficient (i.e., polynomial-time) algorithm to solve the problem. Randomized algorithms are thus natural solutions to the problem. The probabilistic and randomized methods for the constrained SOF problem and robust stabilization via SOFs (among other hard problems) are discussed in [4–7]. For a survey of the SOF problem see [8], and for a recent survey of the robust SOF problem see [9].

The Ray-Shooting Method was recently introduced in [10], where it was used to derive the Ray-Shooting (RS) randomized algorithm for the minimal-gain SOF problem, with regional pole assignment, where the region can be non-convex and unconnected. The Ray-Shooting Method was successfully applied recently also to the following hard complexity control problems for continuous-time systems:

- Structured and structured-sparse SOFs (see [11])
- LQR via SOF for continuous-time LTI systems (see [12])
- LQR optimal, H_∞-optimal, and H_2-optimal proportional-integral-differential (PID) controllers (see [13])
- Robust control via SOF (see [14])

The contribution of the research presented in the current chapter is as follows:

1. The randomized algorithm presented here (which we call the RS algorithm) is based on the Ray-Shooting Method (see [10]), which opposed to smooth optimization methods, and has the potential of finding a global optimum of continuous functions over compact non-convex and unconnected regions.
2. The RS algorithm has a proof of convergence in probability and explicit complexity.
3. Experience with the algorithm shows good quality of controllers (in terms of reduction of the LQR functional value with relatively small controller norms), high percent of success, and good run-time, for real-life systems. Thus, the suggested practical algorithm efficiently solves the problem of LQR via SOF for discrete-time systems.
4. The RS algorithm does not need to solve any Riccati or quadratic matrix equations (QMES) and thus can be applied to large systems.
5. The RS algorithm is one of the few, dealing with the problem of LQR via SOF for discrete-time systems.
6. A deterministic algorithm for the problem that generalizes the algorithm of Moerder and Calise [15], for discrete-time systems, is given (we call it the MC algorithm). The MC algorithm has a proof of convergence to a local optimum only, and it needs other algorithms for computing initial stabilizing SOF.
7. A comparison between the RS and the MC algorithms, as well as the performance of the hybrid algorithm, for real-life systems, is provided.

The reminder of the chapter is organized as follows:

In Section 2 we formulate the problem and give some useful lemmas (without a proof). In Section 3, we introduce the randomized algorithm for the problem of LQR via SOF for discrete-time LTI systems. Section 4 is devoted to the deterministic algorithm for the problem. In Section 5, we give the results of the algorithms for some real-life systems. Finally, in Section 6 we conclude with some remarks.

2. Preliminaries

Let a discrete-time system be given by

$$\begin{cases} x_{k+1} = Ax_k + Bu_k, k = 0, 1, \ldots \\ y_k = Cx_k \end{cases} \tag{1}$$

where $A \in \mathbb{R}^{p\times p}, B \in \mathbb{R}^{p\times q}$, $C \in \mathbb{R}^{r\times p}$, and $x_0 \in \mathbb{R}^p$. Let the LQR cost functional be defined by

$$J(x_0, u) := \sum_{k=0}^{\infty} \left(x_k^T Q x_k + u_k^T R u_k\right), \tag{2}$$

where $Q > 0$ and $R > 0$. Let $u_k = -Ky_k$ be the SOF, and let $A_{c\ell}(K) := A - BKC$ denote the closed-loop matrix. Let $\mathbb{D}$ denote the open unit disk, let $0 < \alpha < 1$, and let $\mathbb{D}_\alpha$ denote the set of all $z \in \mathbb{D}$ with $|z| < 1 - \alpha$ (where $|z|$ is the absolute value of z). For a square matrix Z, we denote by $\sigma(Z)$ its spectrum. For any rectangular matrix Z, we denote by Z^+ its Moore-Penrose pseudo-inverse. By $\|Z\|_F = trace\left(Z^T Z\right)^{\frac{1}{2}}$ we denote the Frobenius norm of Z, and by $\|Z\| = \left(\max\left(\sigma\left(Z^T Z\right)\right)\right)^{\frac{1}{2}}$ we denote the spectral norm of Z. By L_Z and R_Z, we denote the (left and right) orthogonal projections $I - Z^+ Z$ and $I - ZZ^+$ on the spaces $Ker\ (Z)$ and $Ker\ (Z^+)$, respectively. For a topological space $\mathcal{X}$ and a subset $\mathcal{U} \subset \mathcal{X}$, we denote by $int\ (\mathcal{U})$ the interior of $\mathcal{U}$, i.e., the largest open set included in $\mathcal{U}$. By $\overline{\mathcal{U}}$ we denote the closure of $\mathcal{U}$, i.e., the smallest closed set containing $\mathcal{U}$, and by $\partial\mathcal{U} = \overline{\mathcal{U}} -\ int\ (\mathcal{U})$ we denote the boundary of $\mathcal{U}$. Let $\mathcal{S}^{q\times r}$ denote the set of all matrices $K \in \mathbb{R}^{q\times r}$ such that $\sigma(A_{c\ell}) \subset \mathbb{D}$ (i.e., stable in the discrete-time sense), and let $\mathcal{S}^{q\times r}$ denote the set of all matrices $K \in \mathbb{R}^{q\times r}$ such that $\sigma(A_{c\ell}) \subset \mathbb{D}_\alpha$. If the last is nonempty, we say that $A_{c\ell}$ is α-stable and we call α the degree of stability. Let $K \in \mathcal{S}_\alpha^{q\times r}$ be given. Substitution of the SOF $u_k = -Ky_k = -KCx_k$ into (2) yields:

$$J(x_0, K) = \sum_{k=0}^{\infty} x_k^T \left(Q + C^T K^T R K C\right) x_k. \tag{3}$$

Since $Q + C^T K^T RKC > 0$ and $A_{c\ell}(K)$ is stable, it follows that the Stein equation

$$P - A_{c\ell}(K)^T P A_{c\ell}(K) = Q + C^T K^T RKC \tag{4}$$

has a unique solution $P > 0$, given by

$$P(K) = mat\left(\left(I_p \otimes I_p - A_{c\ell}(K)^T \otimes A_{c\ell}(K)^T\right)^{-1} \cdot vec\left(Q + C^T K^T RKC\right)\right). \tag{5}$$

Substitution of (4) into (3) and using $x_k = A_{c\ell}(K)^k x_0$ with the fact that $A_{c\ell}(K)$ is stable leads to

$$\begin{aligned} J(x_0, K) &= \sum_{k=0}^{\infty} x_k^T \left(P - A_{c\ell}(K)^T P A_{c\ell}(K)\right) x_k \\ &= \sum_{k=0}^{\infty} x_0^T A_{c\ell}(K)^{Tk} \left(P - A_{c\ell}(K)^T P A_{c\ell}(K)\right) A_{c\ell}(K)^k x_0 \\ &= x_0^T P(K) x_0 = \left\| P(K)^{\frac{1}{2}} x_0 \right\|^2. \end{aligned}$$

Thus, when x_0 is known, we search for $K \in \mathcal{S}_{\alpha}^{q\times r}$ that minimizes the functional

$$J(x_0, K) = x_0^T P(K) x_0. \tag{6}$$

Let

$$\sigma_{max}(K) \coloneqq \max\left(\sigma(P(K))\right). \tag{7}$$

Now, since $\frac{J(x_0,K)}{\|x_0\|^2} \le \sigma_{max}(K)$ for any $x_0 \neq 0$, with equality in the worst case, therefore

$$sup_{x_0\neq 0}\left(\frac{J(x_0,K)}{\|x_0\|^2}\right) = sup_{x_0\neq 0}\left(\frac{\left\|P(K)^{\frac{1}{2}}x_0\right\|^2}{\|x_0\|^2}\right) = \left\|P(K)^{\frac{1}{2}}\right\|^2 = \|P(K)\| = \sigma_{max}(K).$$

Thus, when x_0 is unknown, we search for $K \in \mathcal{S}_{\alpha}^{q\times r}$, such that $\sigma_{max}(K) = \|P(K)\|$ is minimal. Note that if λ is an eigenvalue of $A_{c\ell}(K)$ and v is a corresponding eigenvector, then (4) yields $1 - |\lambda|^2 = \frac{v^*\left(Q + C^T K^T R K C\right)v}{v^* P(K) v} \ge \frac{v^* Q v}{v^* P(K) v} > 0$. Therefore, $|\lambda|^2 \le 1 - \frac{v^* Q v}{v^* P(K) v} < 1$, and thus, minimizing $\sigma_{max}(K)$ results in eigenvalues that are getting closer to the boundary of $\mathbb{D}$. Since α, the degree of stability, is important to get satisfactory decay rate of the state to 0, and for disturbance rejection, we allow the user of the algorithms to determine α. Note that too high value of α might result in nonexistence of any SOF for the system or in complicating the search for a starting SOF. Higher values of α result in higher values of the optimal value of the LQR functional, i.e., higher energy consumption for decaying the disturbance x_0 to 0.

The functionals $J(x_0, K)$ and $\sigma_{max}(K)$ are generally not convex since their domain of definition $\mathcal{S}_{\alpha}^{q\times r}$ (and therefore $\mathcal{S}_{\alpha}^{q\times r}$) is generally non-convex. Necessary conditions for optimality for continuous-time systems were given as three QMEs in [15–18]. Necessary and sufficient conditions for optimality for continuous-time systems, based on linear matrix inequalities (LMI), were given in [19–21]. However, algorithms based on these formulations are generally not guaranteed to converge, seemingly because of the non-convexity of the coupled matrix equations or inequalities, and when they converge, it is to a local optimum only.

In the sequel, we will use the following lemmas, given here without proofs.

Lemma 2.1. We have:

1. The equation $AX = B$ has solutions if and only if $AA^+B = B$ or equivalently, if and only if $R_A B = 0$. In this case, the set of all solutions is given by

$$X = A^+B + L_A Z,$$

where Z is arbitrary.

2. The equation $XA = B$ has solutions if and only if $BA^+A = B$ or equivalently, if and only if $BL_A = 0$. In this case, the set of all solutions is given by

$$X = BA^+ + ZR_A,$$

where Z is arbitrary.

Lemma 2.2. We have:

1. Let A, B, X be matrices with sizes $p \times q, r \times p, q \times r$, respectively. Then

$$\frac{\partial}{\partial X} trace(AXB) = A^T B^T.$$

2. Let A, B, C, X be matrices with sizes $p \times q, r \times r, q \times p, r \times q$, respectively. Then

$$\frac{\partial}{\partial X} trace\left(AX^T BXC\right) = B^T XA^T C^T + BXCA.$$

3. The randomized Ray-Shooting Method-based algorithm

The Ray-Shooting Method works as follows, for general function minimization: let $f(x) \geq 0$ be a continuous function defined over some compact set $X \subset \mathbb{R}^n$. Let $\epsilon > 0$ be given and assume that we want to compute $x_* \in X$ such that $y_* := f(x_*) = \min_{x \in X} f(x)$ up to ϵ, i.e., to find (x, y) in the set $\mathcal{S}(\epsilon) = \{(x, y) \mid x \in X, f(x_*) \leq y = f(x) \leq f(x_*) + \epsilon.\}$. Let $x_0 \in \mathcal{X}$ be given, let $y_0 := f(x_0)$ and let $\mathcal{S}^{(0)} = \left\{(x, y) \mid x \in X, f(x) \leq y \leq y_0\right\}$ denote the search space, which is a subset of the epigraph of f. Let $\mathcal{D}^{(0)} = \left\{(x, y) \,\middle|\, x \in \mathcal{X}, 0 \leq y \leq y_0\right\}$ denote the cylinder enclosed between $\mathcal{X}$ and the level y_0. Let $\mathcal{L}^{(0)} = \{(x, y) \mid x \in \mathcal{X}, y = 0 \text{ or } x \in \partial\mathcal{X}, 0 \leq y \leq y_0.\}$. Let $z_0 := (x_0, y_0)$ and note that $z_0 \in \mathcal{S}^{(0)}$. Then, we choose w_0 in $\mathcal{L}^{(0)}$ randomly, according to some distribution, and we define the ray as $z(t) := (1-t)z_0 + tw_0, 0 \leq t \leq 1$. We scan the ray and choose the largest $0 \leq t_0 \leq 1$ such that $(1-t_0)z_0 + t_0 w_0 \in \mathcal{S}^{(0)}$ (actually, we scan the ray from $t = 1$ in equal-spaced points and take the first t for which this happens). We define $z_1 := (1-t_0)z_0 + t_0 w_0$ and update sets $\mathcal{S}^{(0)}$, $\mathcal{D}^{(0)}$, and $\mathcal{L}^{(0)}$ by replacing y_0 with y_1, where $(x_1, y_1) = z_1$. Let $\mathcal{S}^{(1)}$, $\mathcal{D}^{(1)}$, and $\mathcal{L}^{(1)}$ denote the updated sets. We continue the process similarly from $z_1 \in \mathcal{S}^{(1)}$, and we define a sequence $z_n \in \mathcal{S}^{(n)}, n = 0, 1, \ldots$. Note that $\mathcal{S}(\epsilon) \subset \mathcal{S}^{(n+1)} \subset \mathcal{S}^{(n)}$ for any $n = 0, 1, \ldots$, unless we have $z_n \in \mathcal{S}(\epsilon)$ for some n (in which the process is ceased). One can show that the sequence $\{z_n\}_{n=0}^{\infty}$ converges (in probability) to a point in $\mathcal{S}(\varepsilon)$. Note that shooting rays from the points of local minimum have positive probability to hit $\mathcal{S}(\varepsilon)$ (under the following mild assumption), because any global minimum is visible from any local minimum. Moreover, for a given level of certainty, we hit $\mathcal{S}(\varepsilon)$ in a finite number of iterations (see Remark 3.1 below). Practically, we may stop the algorithm if no improvement is detected within a window of 20% of the allowed number of iterations. The function need not be smooth or even continuous. It only needs to be well defined and measurable over the compact domain $\mathcal{X}$, and $\mathcal{S}(\varepsilon)$ should have non-negligible measure (i.e., should have some positive volume). Obviously, global minimum points belong to the boundary of the search space $\mathcal{S}^{(0)}$, and actually such points are where the distance between the compact sets $\mathcal{X} \times \{0\}$ and $\mathcal{S}^{(0)}$ in $\mathbb{R}^{n+1}$ is accepted. This is essential for the efficiency of the Ray-Shooting Method, although we raised the search space dimension from n to $n + 1$.

In order to apply the Ray-Shooting Method for the LQR via SOF problem, we need the following definitions: assume that $K^{(0)} \in int\ (\mathcal{S}_\alpha)$ was found by the RS algorithm (see [10]) or by any other method (see [22–24]). Let $h > 0$ and let $\mathcal{U}^{(0)}$ be a unit vector (actually a matrix, but we consider here the space of matrices as a normed vector space) with respect to the Frobenius norm, i.e., $\|\mathcal{U}^{(0)}\|_F = 1$. Let $L^{(0)} = K^{(0)} + h \cdot \mathcal{U}^{(0)}$ and let $\mathcal{L}$ be the hyperplane defined by $L^{(0)} + V$, where $\langle V, \mathcal{U}^{(0)} \rangle_F = 0$. Here $\mathcal{L}$ is the tangent space at $L^{(0)}$ to the closed ball $\overline{\mathbb{B}(K^{(0)}, h)}$ centered at $K^{(0)}$ with radius h, with respect to the Frobenius norm on $\mathbb{R}^{q \times r}$. Let $r_\infty > 0$ and let $\mathcal{R}_\infty$ denote the set of all $F \in \mathcal{L}$, such that $\|F - L^{(0)}\|_F \le r_\infty$. Let $\mathcal{R}_\infty(\epsilon) = R_\infty + \overline{\mathbb{B}(0, \epsilon)}$, where $\overline{\mathbb{B}(0, \epsilon)}$ denotes the closed ball centered at 0 with radius ϵ $(0 < \epsilon \le \frac{1}{2})$. Let $\mathcal{D}^{(0)} = chull(K^{(0)}, \mathcal{R}_\infty(\epsilon))$ denote the convex hull of the vertex $K^{(0)}$ with the basis $\mathcal{R}_\infty(\epsilon)$. Let $\mathcal{S}_\alpha^{(0)} = \mathcal{S}_\alpha \cap \mathcal{D}^{(0)}$ and note that $\mathcal{S}_\alpha^{(0)}$ is compact (but generally not convex). We wish to minimize the continuous function $\sigma_{max}(K)$ (or the continuous function $J(x_0, K)$, when x_0 is known) over the compact set $\mathcal{S}_\alpha \cap \overline{\mathbb{B}(K^{(0)}, h)}$. Let K_* denote a point in $\mathcal{S}_\alpha \cap \overline{\mathbb{B}(K^{(0)}, h)}$ where the minimum of $\sigma_{max}(K)$ is accepted. Obviously, $K_* \in \mathcal{D}^{(0)}$, for some direction $U^{(0)}$ from $K^{(0)}$.

The Ray-Shooting Algorithm 1 for the LQR via SOF problem, works as follows: we start with a point $K^{(0)} \in int\ (\mathcal{S}_\alpha)$, found by the RS algorithm (see [10]). Assuming that $K_* \in \mathcal{D}^{(0)}$, the inner loop $(j = 1, \dots, n)$ uses the Ray-Shooting Method in order to find an approximation of the global minimum of the function $\sigma_{max}(K)$ over $\mathcal{S}_\alpha^{(0)}$ —the portion of $\mathcal{S}_\alpha$ bounded in the cone $D^{(0)}$. The proof of convergence in probability of the inner loop and its complexity (under the above-mentioned assumption) can be found in [10] (see also [11]). In the inner loop, we choose a search direction by choosing a point F in $\mathcal{R}_\infty(\epsilon)$—the base of the cone $\mathcal{D}^{(0)}$. Next, in the most inner loop $(k = 0, \dots, s)$, we scan the ray $K(t) := (1 - t)K^{(0)} + tF$ and record the best controller on it. Repeating this sufficiently many times, we reach K_* (or an ϵ neighborhood of it) with high probability, under the assumption that $K_* \in \mathcal{D}^{(0)}$ (see Remark 3.1).

The reasoning of the Ray-Shooting Method is that sampling the whole search space will lead to the probabilistic method that is doomed to the "curse of dimensionality," which the method tries to avoid. This is achieved by slicing the search space into covering cones (m is the number of cones allowed), because any point in the cone is visible from its vertex. At each cone we shoot rays (n is the number of rays per cone) from its node toward its basis, where each ray is sampled from its head toward its tail, while updating the best point found so far. Note that the global minimum of $\sigma_{\max}(K)$ over any compact subset of $\mathcal{S}_\alpha$ is achieved on the boundary of the related portion of the epigraph of $\sigma_{\max}(K)$. Therefore, we can break the most inner loop; in the first moment, we find an improvement in $\sigma_{\max}(K)$. This bypasses the need to sample the whole search space (although we raise by 1 the search space dimension) and explains the efficiency of the Ray-Shooting Method in finding global optimum. Another advantage of the Ray-Shooting Method which is specific to the problem of LQR via SOF is that the search is concentrated to the parameter space (the qr-dimension space where the K rests) and not to the certificate space (the p^2-dimension space where the Lyapunov matrices P rests). Thus, the method avoids the need to solve any Riccati, LMI, and BMI equations, which might make crucial difference for large-scale systems (i.e., where $p^2 > > qr$).

Algorithm 1. The Ray-Shooting Algorithm for LQR via SOF for discrete-time systems.

Require: *An algorithm for deciding α-stability, an algorithm for computing* $\sigma_{\max}(K)$ *and algorithms for general linear algebra operations.*

Input: $0 < \epsilon \leq \frac{1}{2}, 0 < \alpha < 1, h > 0, r_\infty > 0$, integers: $m, n, s > 0$, controllable pairs (A, B) and $\left(A^T, C^T\right)$, matrices $Q > 0, R > 0$ and $K^{(0)} \in int\ (\mathcal{S}_\alpha)$.

Output: $K \in \mathcal{S}_\alpha$ close as possible to K_*.

compute $P\left(K^{(0)}\right)$ as in (5)
$P^{(\text{best})} \leftarrow P\left(K^{(0)}\right)$
$\sigma_{max}^{(\text{best})} \leftarrow \max\left(\sigma\left(P^{(\text{best})}\right)\right)$
$v \leftarrow 1$
for $i = 1$ **to** m **do**
  choose $\mathcal{U}^{(0)}$ such that $\left\|\mathcal{U}^{(0)}\right\|_F = 1$, uniformly at random
  $L^{(0)} \leftarrow K^{(0)} + h \cdot \mathcal{U}^{(0)}$
  for $j = 1$ **to** n **do**
    choose $F \in \mathcal{R}_\infty(\epsilon)$, uniformly at random
    for $k = 0$ **downto** s **do**
      $t \leftarrow \frac{k}{s}$
      $K(t) \leftarrow (1 - t)K^{(0)} + tF$
      if $K(t) \in \mathcal{S}_\alpha$ **then**
        compute $P(K(t))$ as in (5)
        $\sigma_{max}(K(t)) \leftarrow \max(\sigma(P(K(t))))$
        if $\left(\sigma_{max}(K(t)) < \sigma_{max}^{(\text{best})}\right)$ **then**
          $K^{(\text{best})} \leftarrow K(t)$
          $P^{(\text{best})} \leftarrow P(K(t))$
          $\sigma_{max}^{(\text{best})} \leftarrow \sigma_{max}(K(t))$
        end if
      end if
    end for
  end for
  if $i > v \cdot \lceil e\sqrt{2\pi\, qr} \rceil$ **then**
    $K^{(0)} \leftarrow K^{(\text{best})}$
    $v \leftarrow v + 1$
  end for
end for
return $K^{(\text{best})}, P^{(\text{best})}, \sigma_{max}^{(\text{best})}$

Remark 3.1. In [12] it is shown that by taking $m = \lceil e \cdot \sqrt{2\pi qr} \rceil$ iterations in the outer loop, we have $K_* \in D^{(0)}$, for some direction $\mathcal{U}^{(0)}$, almost surely. Let $\mathcal{S}_\alpha^{(0)}(\epsilon)$ denote the set $\left\{K \in \mathcal{S}_\alpha^{(0)} \mid \sigma_{\max}(K) \leq \sigma_{\max}(K_*) + \epsilon\right\}$. Then, the total number of arithmetic operations of the RS algorithm that guarantees a probability of at least $1 - \beta$ to hit $\mathcal{S}_\alpha^{(0)}(\epsilon)$ is given by $O\left(\frac{|\ln(\beta)| h}{\epsilon}\left(\frac{r_\infty}{r_\epsilon}\right)^{q_0 r_0}\left(\max(q, r)^3 + p^6\right)\right)$, for systems with $q \leq q_0, r \leq r_0$ for fixed q_0, r_0, where r_ϵ is the radius of the basis of a cone with height ϵ that has the same volume as of $\mathcal{S}_\alpha^{(0)}(\epsilon)$; see [10–12]. This is a polynomial-time algorithm by restricting the input and by regarding $\left(\frac{r_\infty}{r_\epsilon}\right)$ as the size of the problem.

4. The deterministic algorithm

The deterministic algorithm we introduce here as Algorithm 2 (which we call the MC algorithm) generalizes the algorithm of Daniel D. Moerder and Anthony A. Calise (see [15]) to the case of discrete-time systems. To the best of our knowledge, this is the best algorithm for LQR via SOF published so far, in terms of rate of convergence (to local minimum).

Here, we wish to minimize the LQR functional

$$J(x_0, P) = x_0^T P x_0, \tag{8}$$

under the constraints

$$Y(K,P) := Q + C^T K^T RKC - P + A_{c\ell}(K)^T P A_{c\ell}(K) = 0, P > 0. \tag{9}$$

Since $Y^T = Y$, there exist orthogonal matrix $\mathcal{U}$ such that $\hat{Y} = \mathcal{U}^T Y \mathcal{U}$ is diagonal. Now, minimizing (8) under the constraints (9) is equivalent to minimizing

$$\mathcal{L}(K,P,S) = trace\left(x_0^T P x_0\right) + \sum_{i=1}^{p} \hat{S}_{i,i} \hat{Y}_{i,i},$$

under the constraint $P > 0$, where $\hat{S}_{i,i}$ are the Lagrange multipliers. We have

$$\begin{aligned}
\mathcal{L}(K,P,S) &= trace\left(x_0^T P x_0\right) + \sum_{i=1}^{p} \hat{S}_{i,i} \hat{Y}_{i,i} \\
&= trace\left(x_0^T P x_0\right) + trace\left(\hat{S}\hat{Y}\right) \\
&= trace\left(x_0^T P x_0\right) + trace\left(\hat{S}\mathcal{U}^T Y \mathcal{U}\right) \\
&= trace\left(x_0^T P x_0\right) + trace\left(\mathcal{U}\hat{S}\mathcal{U}^T Y\right) \\
&= trace\left(x_0^T P x_0\right) + trace(SY)
\end{aligned}$$

where $S = \mathcal{U}\hat{S}\mathcal{U}^T$. Note that $S^T = S$. Let the Lagrangian be defined by

$$\mathcal{L}(K,P,S) = trace\left(x_0^T P x_0\right) + trace(SY(K,P)), \tag{10}$$

for any K any $P > 0$ and any S such that $S^T = S$. The necessary conditions for optimality are $\frac{\partial \mathcal{L}}{\partial K} = 0$, $\frac{\partial \mathcal{L}}{\partial P} = 0$, and $\frac{\partial \mathcal{L}}{\partial S} = Y^T = Y = 0$.

Now, using Lemma 2.2, we have

$$\begin{aligned}
&\frac{\partial \mathcal{L}}{\partial P} = 0 \\
&\Leftrightarrow x_0 x_0^T - S^T + A_{c\ell} S^T A_{c\ell}^T = 0 \\
&\Leftrightarrow x_0 x_0^T - S + A_{c\ell} S A_{c\ell}^T = 0 \\
&\Leftrightarrow S - A_{c\ell} S A_{c\ell}^T = x_0 x_0^T \\
&\Leftrightarrow \left(I_p \otimes I_p - A_{c\ell} \otimes A_{c\ell}\right) vec(S) = vec\left(x_0 x_0^T\right) \\
&\Leftrightarrow S = mat\left(\left(I_p \otimes I_p - A_{c\ell} \otimes A_{c\ell}\right)^{-1} vec\left(x_0 x_0^T\right)\right),
\end{aligned}$$

where the last passage is affordable because $\sigma(A_{c\ell}) \subset \mathbb{D}$. Note that the last with the stability of $A_{c\ell}$ implies that $S \geq 0$.

We also have

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial K} &= \frac{\partial}{\partial K} trace(SY) \\ &= \frac{\partial}{\partial K} trace\left(S\left(Q + C^T K^T RKC - P + \left(A^T - C^T K^T B^T\right)P(A - BKC)\right)\right) \\ &= \frac{\partial}{\partial K} trace\left(SC^T K^T RKC - SA^T PBKC - SC^T K^T B^T PA + SC^T K^T B^T PBKC\right) \\ &= \frac{\partial}{\partial K} trace\left(SC^T K^T RKC - SA^T PBKC - A^T P^T BKCS^T + SC^T K^T B^T PBKC\right) \\ &= R^T KCS^T C^T + RKCSC^T - B^T P^T AS^T C^T - B^T PASC^T \\ &\quad + B^T P^T BKCS^T C^T + B^T PBKCSC^T \\ &= 2RKCSC^T - 2B^T PASC^T + 2B^T PBKCSC^T.\end{aligned}$$

Therefore,

$$\begin{aligned}&\frac{\partial \mathcal{L}}{\partial K} = 0 \\ &\Leftrightarrow RKCSC^T - B^T PASC^T + B^T PBKCSC^T = 0 \\ &\Leftrightarrow \left(R + B^T PB\right)KCSC^T = B^T PASC^T \\ &\Leftrightarrow KCSC^T = \left(R + B^T PB\right)^{-1} B^T PASC^T.\end{aligned}$$

Thus, if CSC^T is invertible, then

$$K = \left(R + B^T PB\right)^{-1} B^T PASC^T \left(CSC^T\right)^{-1}. \tag{11}$$

Otherwise, if

$$\left(R + B^T PB\right)^{-1} B^T PASC^T \cdot L_{CSC^T} = 0,$$

which is equivalent to

$$B^T PASC^T \cdot L_{CSC^T} = 0, \tag{12}$$

then

$$K = \left(R + B^T PB\right)^{-1} B^T PASC^T \left(CSC^T\right)^{+} + Z \cdot R_{CSC^T}, \tag{13}$$

where Z is arbitrary $q \times r$ matrix (and we may take $Z = 0$, unless some other constraints on K are needed). Note that if condition (12) does not happen, then $\frac{\partial \mathcal{L}}{\partial K} \neq 0$. We conclude with the following theorem:

Theorem 4.1. Assume that $\mathcal{L}(K, P, S)$ given by (10) is minimized locally at some point K_*, $P_* > 0$, and S_* such that $S_*^T = S_*$. Then

$$\begin{cases} K_* = \left(R + B^T P_* B\right)^{-1} B^T P_* AS_* C^T \left(CS_* C^T\right)^{+} + Z_* \cdot R_{CSC^T}, for\ some\ q \times r\ matrix\ Z_* \\ P_* = mat\left(\left(I_p \otimes I_p - A_{c\ell}(K_*)^T \otimes A_{c\ell}(K_*)^T\right)^{-1} \cdot vec\left(Q + C^T K_*^T RK_* C\right)\right) \\ S_* = mat\left(\left(I_p \otimes I_p - A_{c\ell}(K_*) \otimes A_{c\ell}(K_*)\right)^{-1} vec\left(x_0 x_0^T\right)\right), \end{cases} \tag{14}$$

where $A_{c\ell}(K_*) = A - BK_* C$.

Proof:

Since $\mathcal{L}(K_*, P_*, S_*)$ is minimal in some neighborhood of (K_*, P_*, S_*), it follows that $\frac{\partial \mathcal{L}}{\partial K}(K_*, P_*, S_*) = 0$, $\frac{\partial \mathcal{L}}{\partial P}(K_*, P_*, S_*) = 0$, and $\frac{\partial \mathcal{L}}{\partial S}(K_*, P_*, S_*) = Y^T(K_*, P_*) = Y(K_*, P_*) = 0$.

The condition $\frac{\partial \mathcal{L}}{\partial S}(K_*, P_*, S_*) = Y^T(K_*, P_*) = Y(K_*, P_*) = 0$ is just

$$P_* - A_{c\ell}(K_*)^T P_* A_{c\ell}(K_*) = Q + C^T K_*^T R K_* C$$

which with $P_* > 0$ and $Q > 0, R > 0$ implies that $A_{c\ell}(K_*) = A - BK_* C$ is stable. Now, since $\sigma(A_{c\ell}(K_*)) \subset \mathbb{D}$, it follows that $I_p \otimes I_p - A_{c\ell}(K_*)^T \otimes A_{c\ell}(K_*)^T$ is invertible, and therefore,

$$P_* = mat\left(\left(I_p \otimes I_p - A_{c\ell}(K_*)^T \otimes A_{c\ell}(K_*)^T\right)^{-1} \cdot vec\left(Q + C^T K_*^T R K_* C\right)\right).$$

Since $I_p \otimes I_p - A_{c\ell}(K_*) \otimes A_{c\ell}(K_*)$ is invertible, $\frac{\partial \mathcal{L}}{\partial P}(K_*, P_*, S_*) = 0$ implies that

$$S_* = mat\left(\left(I_p \otimes I_p - A_{c\ell}(K_*) \otimes A_{c\ell}(K_*)\right)^{-1} vec\left(x_0 x_0^T\right)\right).$$

Finally, $\frac{\partial \mathcal{L}}{\partial P}(K_*, P_*, S_*) = 0$ implies that $K_* C S_* C^T = \left(R + B^T P_* B\right)^{-1} B^T P_* A S_* C^T$, which in view of Lemma 2.1 implies $\left(R + B^T P_* B\right)^{-1} B^T P_* A S_* C^T \cdot L_{CS_* C^T} = 0$ and

$$K_* = \left(R + B^T P_* B\right)^{-1} B^T P_* A S_* C^T \left(C S_* C^T\right)^+ + Z_* \cdot R_{CS_* C^T},$$

where Z_* is some $q \times r$ matrix. ■

Note that the equations are coupled tightly, in the sense that P_* and S_* need K_*, while K_* needs P_* and S_*. Note also the cubic dependencies (that can be made quadratic by introducing new variables). These make the related QMEs non-convex and, therefore, hard to compute.

Remark 4.1. When x_0 is unknown, it is customary to assume that x_0 is uniformly distributed on the unit sphere, which implies that $E\left[x_0 x_0^T\right] = I_p$, where $E[\bullet]$ is the expectation operator. Thus, changing the problem to that of minimizing $E[J(x_0, P)]$ amounts to replacing S_* with

$$E[S_*] = mat\left(\left(I_p \otimes I_p - A_{c\ell}(K_*) \otimes A_{c\ell}(K_*)\right)^{-1} vec\left(I_p\right)\right) > 0.$$

Therefore, there is change in Algorithm 2.

Remark 4.2. The convergence of Algorithm 2 to local minimum can be proved similarly to the proof appearing in [15], under the assumptions that $\mathcal{S}_\alpha^{q\times r}$ is nonempty and that $\left(Q^{\frac{1}{2}}, A\right)$ is detectable (here, we do not need this condition because of the assumption that $Q > 0$). The convergence can actually be proved for the more general problem that adds $\|K\|_F^2 = trace\left(K^T K\right)$ to the LQR functional, thus minimizing also the Frobenius norm of K. In this context, note that by adding $\|K\|^2 = \max\left(\sigma\left(K^T K\right)\right)$ to the LQR functional will lose the argument, and there will be a need to more general proof, because in the proof appearing in [15], the demand is for $\mathbf{C}^1$ smooth function of K, while $\|K\|^2 = \sigma_{\max}\left(K^T K\right)$ is continuous but not Lipschitz continuous. The RS algorithm can use any continuous function of K and can deal also with sparse SOFs for LQR and with regional pole-placement SOFs for LQR.

Example 4.1. In the following simple example, we illustrate the notions appearing in the definition of the RS algorithm, and we demonstrate the operation of the RS algorithm. Consider the unstable system

$$\begin{cases} x_{k+1} = \begin{bmatrix} 2 & 1 \\ 0 & -\frac{1}{2} \end{bmatrix} x_k + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u_k, k = 0, 1, \ldots \\ y_k = x_k, \end{cases}$$

where we look for SOF K stabilizing the system while reducing the LQR functional (2) with $Q = I, R = 1$. Let $K = [k_1 \;\; k_2]$ then

$$A_{c\ell}(K) = A - BK = \begin{bmatrix} 2 - k_1 & 1 - k_2 \\ -k_1 & -\frac{1}{2} - k_2 \end{bmatrix},$$

with characteristic polynomial $z^2 + z\left(k_1 + k_2 - \frac{3}{2}\right) + \frac{3}{2}k_1 - 2k_2 - 1$. Applying the Y. Bistritz stability criterion (see [25]), we have

$$\upsilon = Var\left\{\frac{5}{2}k_1 - k_2 - \frac{3}{2}, -\frac{3}{2}k_1 + 2k_2 + 2, \frac{1}{2}k_1 - 3k_2 + \frac{3}{2}\right\},$$

where υ is the number of sign variations in the set. According to the Bistritz criterion, the system is stable if and only if $\upsilon = 0$. We conclude that $\mathcal{S}$ is the set of all K such that $\frac{5}{2}k_1 - k_2 - \frac{3}{2} > 0$, $-\frac{3}{2}k_1 + 2k_2 + 2 > 0$, $\frac{1}{2}k_1 - 3k_2 + \frac{3}{2} > 0$ or $\frac{5}{2}k_1 - k_2 - \frac{3}{2} < 0$, $-\frac{3}{2}k_1 + 2k_2 + 2 < 0$, $\frac{1}{2}k_1 - 3k_2 + \frac{3}{2} < 0$, where the last branch is empty (which could have make the set non-convex). The set $\mathcal{S}$ appears in **Figure 1** as the blue region, where the golden star is the analytic global optimal solution $K_* = [1.09473459 \;\; 0.36138828]$ (computed by the related discrete algebraic Riccati equation).

Algorithm 2 The MC algorithm for LQR via SOF for discrete-time systems.

Require: *An algorithm for deciding α-stability, an algorithm for computing $\sigma_{\max}(K)$ and algorithms for general linear algebra operations.*

Input: $0 < \epsilon \leq \frac{1}{2}, 0 < \alpha < 1$, integers: $m, s > 0$, controllable pairs (A, B) and $\left(A^T, C^T\right)$, matrices $Q > 0, R > 0$ and $K^{(0)} \in int\ (\mathcal{S}_\alpha)$.

Output: $K \in \mathcal{S}_\alpha$ that locally minimizes $\sigma_{max}(K)$.

1. $j \leftarrow 0; A_0 \leftarrow A - BK_0C$
2. $P_0 \leftarrow mat\left(\left(I_p \otimes I_p - A_0^T \otimes A_0^T\right)^{-1} \cdot vec\left(Q + C^T K_0^T R K_0 C\right)\right)$
3. $S_0 \leftarrow mat\left(\left(I_p \otimes I_p - A_0 \otimes A_0\right)^{-1} \cdot vec\left(I_p\right)\right); \sigma_{max}(K_0) \leftarrow \max(\sigma(P_0))$
4. $\Delta K_0 \leftarrow \left(R + B^T P_0 B\right)^{-1} B^T P_0 A S_0 C^T (C S_0 C)^+ - K_0$
5. $flag \leftarrow 0$
6. **for** $k = 0$ **to** s **do**
7. $\quad t \leftarrow \frac{k}{s}$
8. $\quad K(t) \leftarrow (1 - t)K_0 + t\Delta K_0$
9. $\quad$ **if** $K(t) \in \mathcal{S}_\alpha$ **then**
10. $\quad\quad A(t) \leftarrow A - BK(t)C$

$P(t) \leftarrow mat\left(\left(I_p \otimes I_p - A(t)^T \otimes A(t)^T\right)^{-1} \cdot vec\left(Q + C^T K(t)^T R K(t) C\right)\right)$
$S(t) \leftarrow mat\left(\left(I_p \otimes I_p - A(t) \otimes A(t)\right)^{-1} \cdot vec\left(I_p\right)\right); \sigma_{max}(K(t)) \leftarrow \max\left(\sigma(P(t))\right)$
if $\sigma_{max}(K(t)) < \sigma_{max}(K_0)$ **then**
$K_1 \leftarrow K(t); A_1 \leftarrow A - BK_1C; P_1 \leftarrow P(t); S_1 \leftarrow S(t); \sigma_{max}(K_1) \leftarrow \sigma_{max}(K(t))$
$flag \leftarrow 1$
end if
end if
end for
if $flag == 1$ **then**
while $\left(\left|\sigma_{max}(K_{j+1}) - \sigma_{max}(K_j)\right| \geq \epsilon\right)$ and $(j < m)$ **do**
$\Delta K_j \leftarrow \left(R + B^T P_j B\right)^{-1} B^T P_j A S_j C^T \left(C S_j C^T\right)^{+} - K_j$
for $k = 0$ **to** s **do**
$t \leftarrow \frac{k}{s}$
$K(t) \leftarrow (1-t)K_j + t\Delta K_j$
if $K(t) \in \mathcal{S}_\alpha$ **then**
$A(t) \leftarrow A - BK(t)C$
$P(t) \leftarrow mat\left(\left(I_p \otimes I_p - A(t)^T \otimes A(t)^T\right)^{-1} \cdot vec\left(Q + C^T K(t)^T R K(t) C\right)\right)$
$S(t) \leftarrow mat\left(\left(I_p \otimes I_p - A(t) \otimes A(t)\right)^{-1} \cdot vec\left(I_p\right)\right); \sigma_{max}(K(t)) \leftarrow \max\left(\sigma(P(t))\right)$
if $\sigma_{max}(K(t)) < \sigma_{max}(K_j)$ **then**
$K_{j+1} \leftarrow K(t); A_{j+1} \leftarrow A - BK_{j+1}C; P_{j+1} \leftarrow P(t); S_{j+1} \leftarrow S(t); \sigma_{max}(K_{j+1}) \leftarrow \sigma_{max}(K(t))$
end if
end if
end for
$j \leftarrow j + 1$
end while
end if
return $K^{(\text{best})} \leftarrow K_j, P^{(\text{best})} \leftarrow P_j, \sigma_{max}^{(\text{best})} \leftarrow \sigma_{max}(K_j)$

In **Figure 1**, we can see how the RS algorithm works: we fix $\alpha = 10^{-3}, \epsilon = 10^{-16}, r_\infty = 2, h = 2$, and we set $m = 1, n = 5, s = 10$ for a single iteration, where the single cone is sampled along 5 rays and each ray is sampled 10 times. The sampled points are circled, where red circles indicate infeasible or non-improving points and the black circles indicate improving points. The green star point is the initial point $K^{(0)}$ found by the Ray-Shooting algorithm for minimal-norm SOF. The bold black circle is the boundary of the closed circle $\overline{\mathbb{B}\left(K^{(0)}, h\right)}$. We choose $\mathcal{U}^{(0)}$ randomly to define the search direction, and we set $L^{(0)} = K^{(0)} + h \cdot \mathcal{U}^{(0)}$ to be the point where the direction meets the boundary of the circle. L is the tangent line at $L^{(0)}$ to the circle, and $\mathcal{R}_\infty(\epsilon)$ is the $2r_\infty$ width segment on the line, inflated by ϵ. The search cone $\mathcal{D}^{(0)} = chull\left(K^{(0)}, R_\infty(\epsilon)\right)$ is the related black triangle. Here $\mathcal{S}_\alpha^{(0)} = \mathcal{S}_\alpha \cap \mathcal{D}^{(0)}$ is just the portion of the blue region inside the triangle, and we can see that the assumption that $K_* \in \mathcal{D}^{(0)}$ is in force. For the current problem $\lceil e\sqrt{2\pi\, qr} \rceil = 10$, and therefore, by making 10 iterations, K_* will be inside some triangle almost surely.

The algorithm chooses F in the basis of the triangle and defines $K(t)$ to be the ray from K_0 to F. The ray is sampled at 10 equally spaced points, and the best feasible point is recorded.

In **Figure 2**, we can see that 5 iterations suffice to include K_* in some triangle and to find improving points very close to K_*. In **Figure 3**, we can see that when we allow 20 iterations, after 10 iterations, the center $K^{(0)}$ is switched to the best point found so far (see lines $24-26$ in Algorithm 1). This is done in order to raise the probability to hit K_* or its ϵ-neighborhood, and as we can see, the final best point (green star) is very close to K_* (**Figure 4**).

The results of the algorithm for 1, 5 and 20 iterations are the following. Note that $\sigma_{\max}(K_*) = 5.9551$, and note the "huge variations" the function $\sigma_{max}(K)$ has.

For $m = 1$ we had

$$
\begin{aligned}
&K^{(0)} = [0.58739333 \quad -0.15823016], \sigma_{\max}^{(0)} = 25.7307, \\
&\mathbf{RS}: K^{(best)} = [1.17786349 \quad 0.35034398], \sigma_{\max}^{(best)} = 6.1391, \\
&\mathbf{MC}: K^{(best)} = [0.58739333 \quad -0.15823016], \sigma_{\max}^{(best)} = 25.7307, \\
&\mathbf{RS}+\mathbf{MC}: K^{(best)} = [1.05244278 \quad 0.31681948], \sigma_{\max}^{(best)} = 6.0001.
\end{aligned}
$$

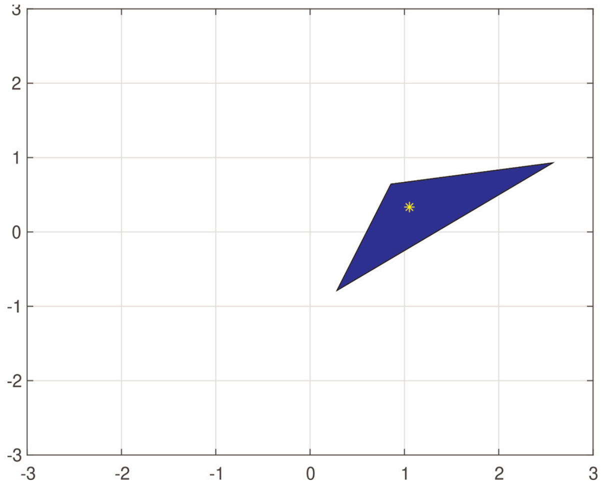

Figure 1.
The stability region S.

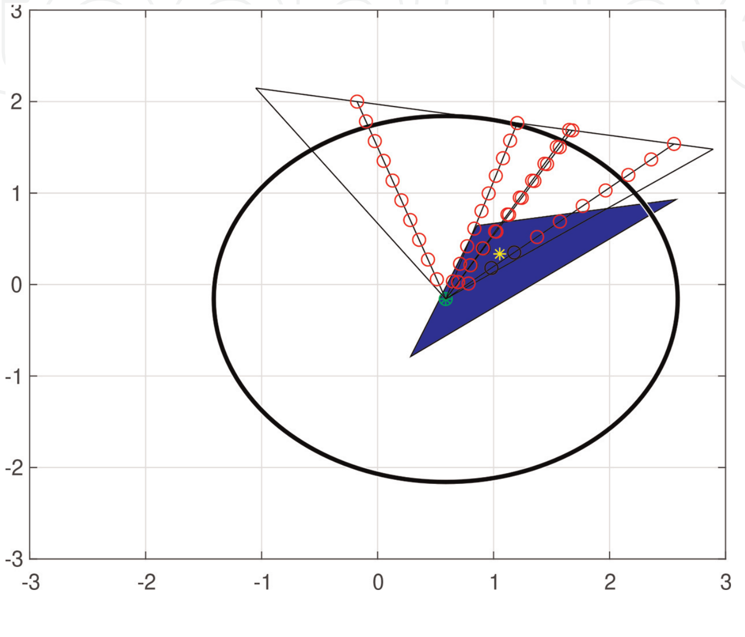

Figure 2.
Single iteration of the RS algorithm.

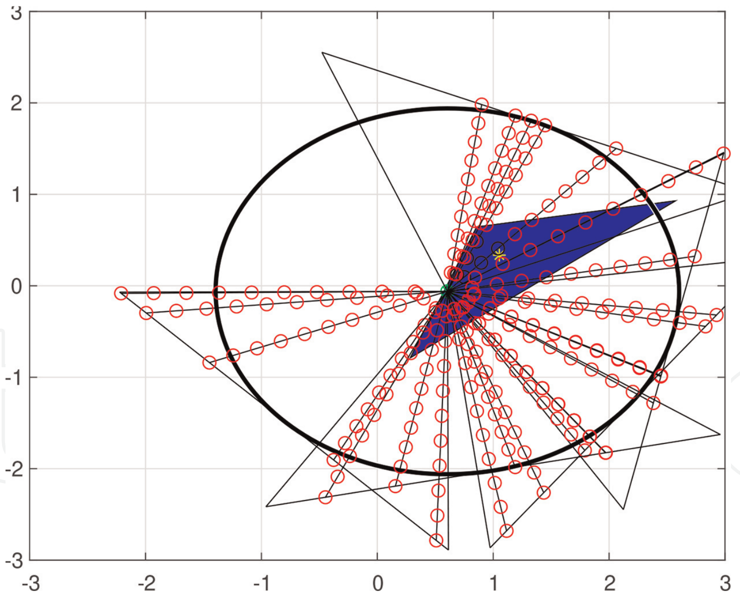

Figure 3.
Five iterations of the RS algorithm.

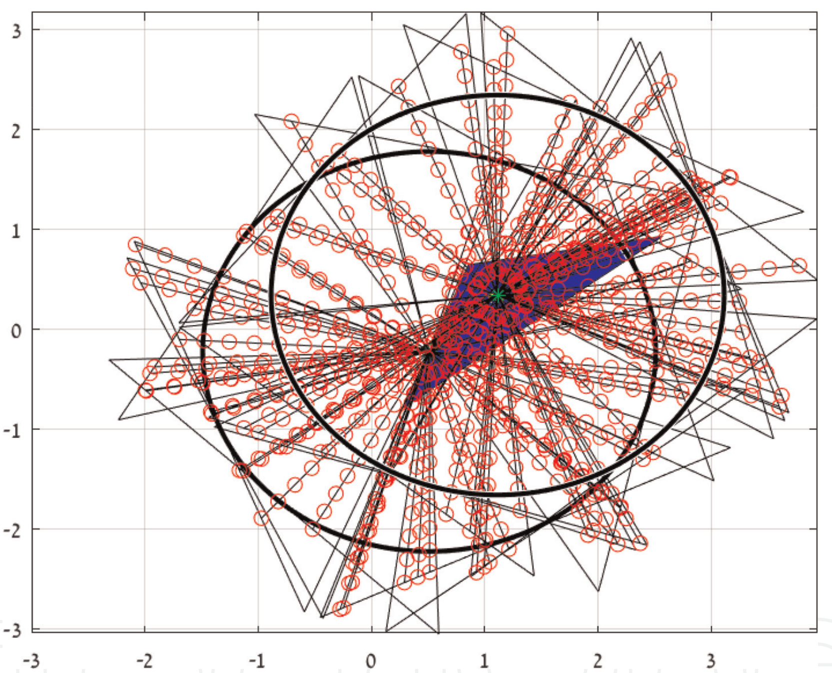

Figure 4.
Twenty iterations of the RS algorithm.

Note that in this case, the *MC* algorithm makes no improvement, while the RS and RS + MC have very close values to the global optimal value, with slightly better value for the RS + MC, over the RS algorithm.

For $m = 5$ we had

$$
\begin{aligned}
&K^{(0)} = [0.60478870 \quad -0.06023828], \sigma_{\max}^{(0)} = 36.4583,\\
&\mathbf{RS} : K^{(best)} = [1.04166520 \quad 0.40826562], \sigma_{\max}^{(best)} = 6.1655,\\
&\mathbf{MC} : K^{(best)} = [0.60478870 \quad -0.06023828], \sigma_{\max}^{(best)} = 36.4583,\\
&\mathbf{RS} + \mathbf{MC} : K^{(best)} = [1.04166520 \quad 0.40826562], \sigma_{\max}^{(best)} = 6.16557843.
\end{aligned}
$$

For $m = 20$ we had

$$K^{(0)} = [0.51029365 \quad -0.22521376], \sigma_{\max}^{(0)} = 3198.8196,$$

$$\mathbf{RS}: K^{(best)} = [1.11453066 \quad 0.33955607], \sigma_{\max}^{(best)} = 5.9893,$$

$$\mathbf{MC}: K^{(best)} = [0.51029365 \quad -0.22521376], \sigma_{\max}^{(best)} = 3198.8196,$$

$$\mathbf{RS} + \mathbf{MC}: K^{(best)} = [1.11453066 \quad 0.33955607], \sigma_{\max}^{(best)} = 5.9893.$$

In **Figure 5**, the initial condition response of the open-loop system is given. One can see the unstable mode related to the unstable eigenvalue 2. In **Figures 6–8**, the

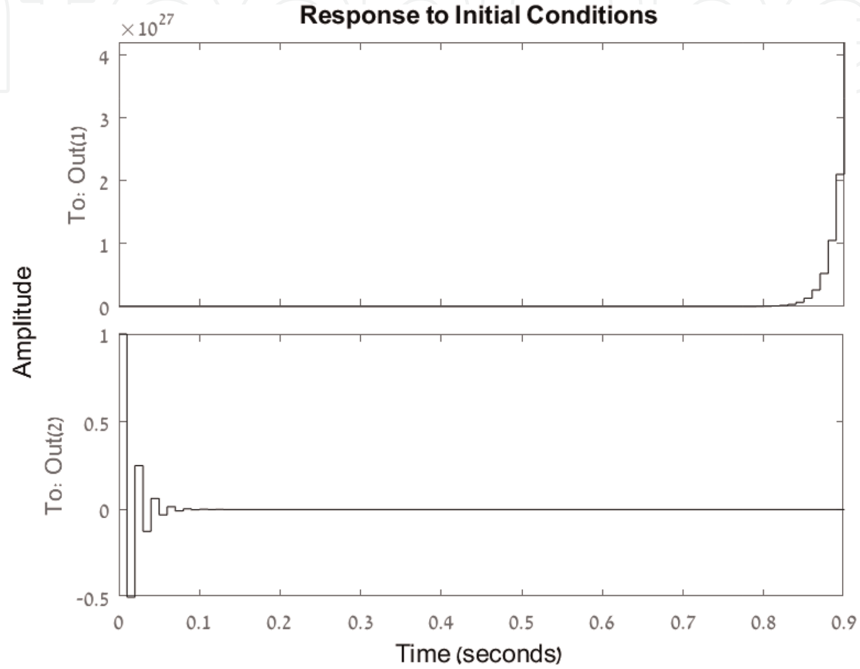

Figure 5.
The initial condition response of the open-loop system.

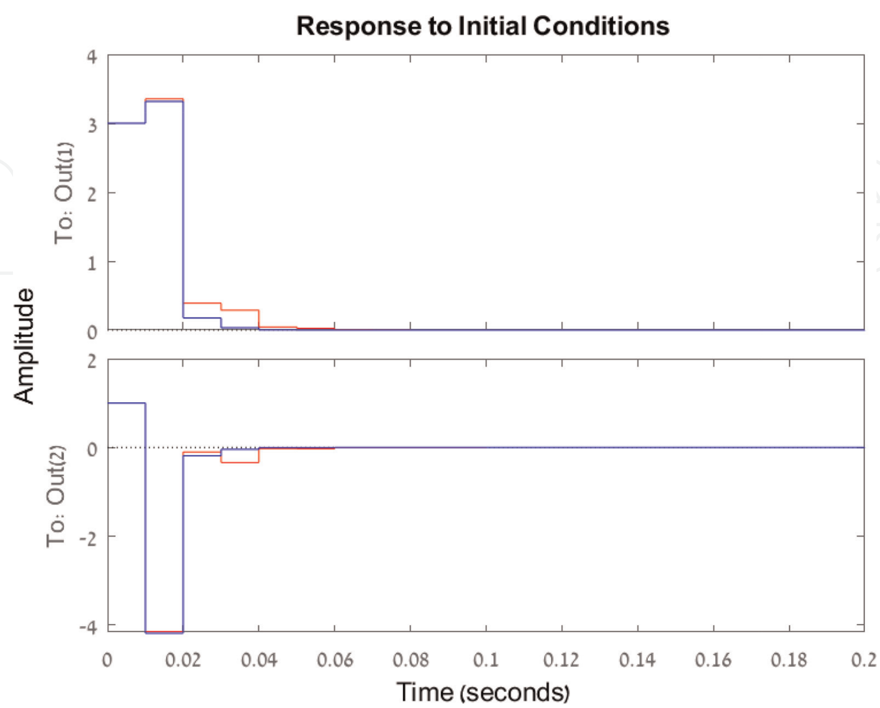

Figure 6.
The initial condition response of the closed-loop system with the SOF computed by the RS algorithm (blue) compared with the global optimal response (red).

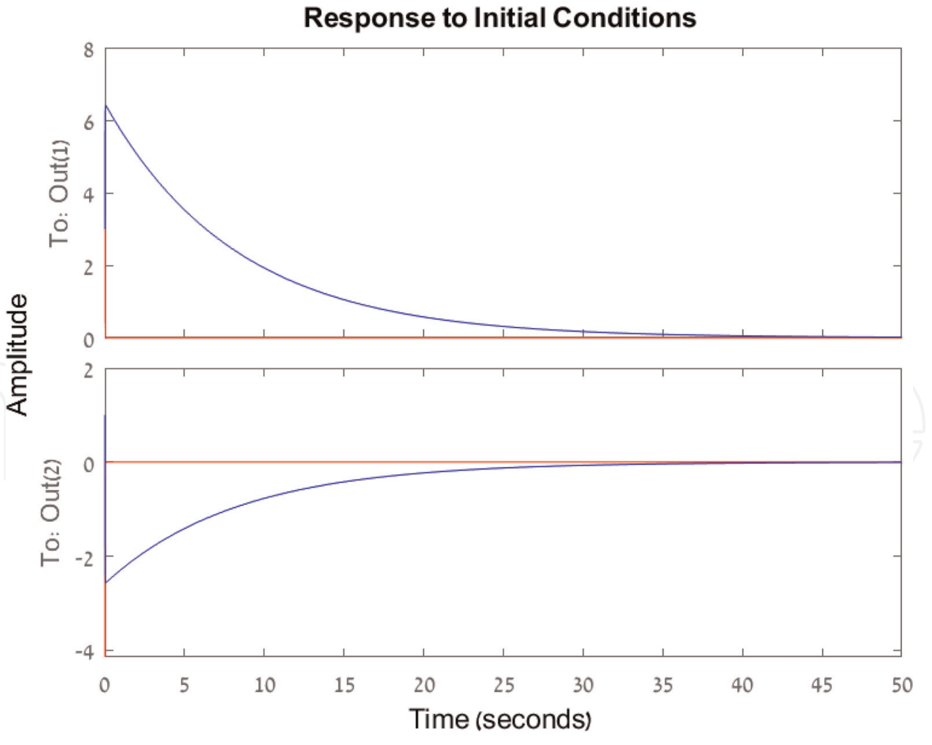

Figure 7.
The initial condition response of the closed-loop system with the SOF computed by the MC algorithm (blue) compared with the global optimal response (red).

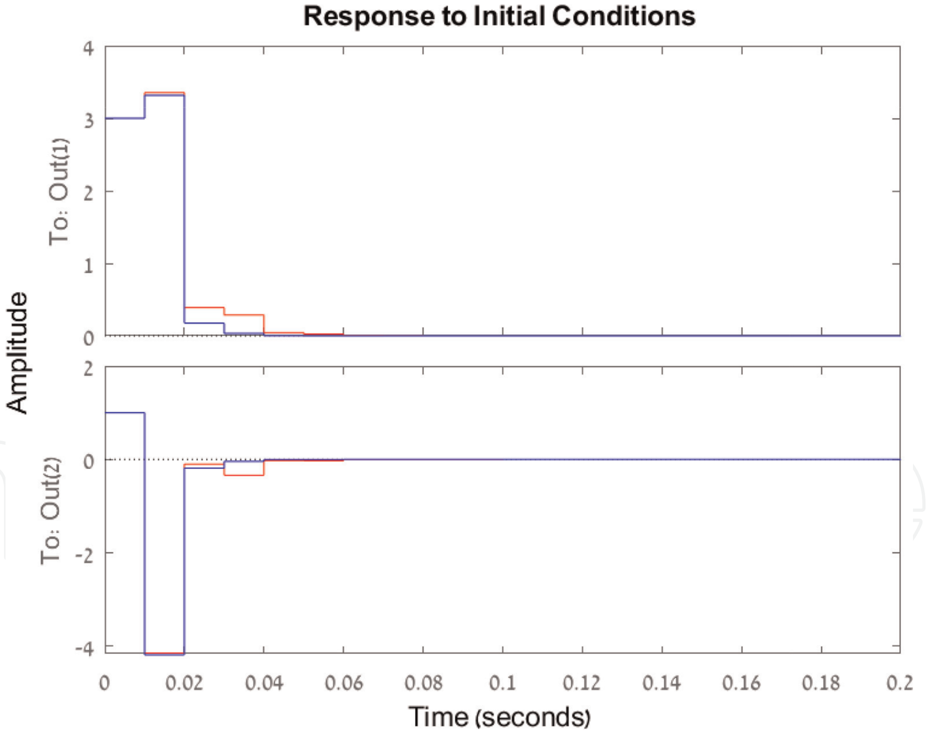

Figure 8.
The initial condition response of the closed-loop system with the SOF computed by the RS + MC algorithm (blue) compared with the global optimal response (red).

initial condition responses of the closed-loop systems with the SOFs for $m = 20$, with $x_0 = [3 \quad 1]^T$ and sampling time $T_s = 0.01$, are given. One can see that the responses of the closed-loop systems with the SOFs computed by RS and RS + MC are very close to the global optimal response, while the response of the closed-loop system with the SOF computed by the MC algorithm (actually with the initial SOF), although stable, is unacceptable.

5. Experiments

In the following experiments, we applied Algorithms 1 and 2, for systems taken from the libraries [26–28]. The systems given in these libraries are real-life continuous-time systems. In order to get related discrete-time systems, we sampled the systems using the Tustin method with sampling rate $T_s = 0.01[\sec]$. We took only the systems for which the RS algorithm succeeded in finding SOFs for the continuous-time systems (see [10], Table 8, p. 231). In order to initialize the MC Algorithm, we also used the RS algorithm to find a starting α-stabilizing SOF. In all the experiments, we used $m = 2\lceil e\sqrt{2\pi\, qr}\rceil, n = 100, s = 100; h = 100, r_\infty = 100$, $\epsilon = 10^{-16}$, for the RS algorithm; and $m = 200\lceil e\sqrt{2\pi\, qr}\rceil, s = 100$, for the MC Algorithm, in order to get the same number of total iterations and the same number $s = 100$ of iterations for the local search. We took $Q = I_p, R = I_q$ in all the cases.

The stability margin column of **Table 1** relates to $0 < \alpha < 1$ for which the absolute value of any eigenvalue of the closed loop is less than $1 - \alpha$. The values of α in **Table 1** relates to the largest $0 < \alpha < 1$ for which the RS algorithm succeeded in finding a starting SOF $K^{(0)}$. As we saw above, it is worth searching for a starting point $K^{(0)}$ that maximizes $0 < \alpha < 1$. This can be achieved efficiently by running a binary search on the $0 < \alpha < 1$ and using the RS algorithm as an oracle. Note that the RS CPU time appearing in the fourth column of **Table 1** relates to running the RS algorithm for known optimal value of $0 < \alpha < 1$. The RS algorithm is sufficiently fast also for this purpose, but other algorithms such as the HIFOO (see [24]) and

System	Size (p, q, r)	Stab. Mgn.	RS CPU time [sec]	$\sigma_{\max}^{(0)}$ for (A, B, C)	$\sigma_{\max}(F_*)$ for (A, B)
AC1	$(5, 3, 3)$	0.01	2.6226	$1.0701 \cdot 10^4$	$1.3073 \cdot 10^3$
AC5	$(4, 2, 2)$	0.001	1.5468	$1.5888 \cdot 10^9$	$8.4264 \cdot 10^7$
AC6	$(7, 2, 4)$	0.001	0.7094	$3.1767 \cdot 10^3$	$5.9783 \cdot 10^2$
AC11	$(5, 2, 4)$	0.01	1.0575	$1.2968 \cdot 10^4$	$5.8777 \cdot 10^2$
HE1	$(4, 2, 1)$	0.001	0.0872	$1.5040 \cdot 10^3$	$3.0013 \cdot 10^2$
HE3	$(8, 4, 6)$	0.001	2.6845	$5.4064 \cdot 10^6$	$6.1185 \cdot 10^4$
HE4	$(8, 4, 6)$	0.001	2.5633	$4.1660 \cdot 10^6$	$2.2992 \cdot 10^4$
ROC1	$(9, 2, 2)$	10^{-5}	0.5279	$1.5906 \cdot 10^7$	$1.1207 \cdot 10^5$
ROC4	$(9, 2, 2)$	10^{-5}	0.4677	$1.2273 \cdot 10^6$	$8.5460 \cdot 10^4$
DIS4	$(8, 4, 6)$	0.01	2.5074	$4.5133 \cdot 10^3$	$1.7556 \cdot 10^2$
DIS5	$(4, 2, 2)$	0.001	1.2187	$2.8686 \cdot 10^8$	$9.0756 \cdot 10^6$
TF1	$(7, 2, 4)$	10^{-4}	0.8011	$7.9884 \cdot 10^5$	$5.8134 \cdot 10^3$
NN5	$(7, 1, 2)$	10^{-4}	0.4138	$5.4066 \cdot 10^6$	$2.8789 \cdot 10^5$
NN13	$(6, 2, 2)$	0.01	0.4876	$7.8402 \cdot 10^2$	63.5366
NN16	$(8, 4, 4)$	10^{-4}	3.5530	$1.9688 \cdot 10^3$	$2.3327 \cdot 10^2$
NN17	$(3, 2, 1)$	0.001	0.0925	$3.2733 \cdot 10^4$	$3.1358 \cdot 10^2$

Table 1.
General information of the systems and initial values.

HINFSTRUCT (see [29]) can be applied in order to get a starting SOF. The advantage of the use of the RS is of finding starting SOF with relatively small norm.

Let $\sigma_{max}(F)$ denote the functional (7) for the system (A, B, I_p), where $A - BF$ is stable, i.e., $F \in \mathcal{S}^{q \times p}$. Let $P(F)$ denote the Lyapunov matrix (5) for the system (A, B, I_p) with F as above. Let $\sigma_{max}(K)$ denote the functional (7) for the system (A, B, C) with $K \in \mathcal{S}^{q \times r}$ and related Lyapunov matrix $P = P(K)$ given by (5). Now, if $A - BKC$ is stable for some K, then $A - BF$ is stable for $F = KC$ (but there might exist F such that $A - BF$ is stable, which cannot be defined as KC for some $q \times r$ matrix K). Therefore,

$$\sigma_{max}(F_*) = \min_{F \in \mathcal{S}^{q \times p}} \sigma_{max}(F) \leq \min_{K \in \mathcal{S}_\alpha^{\{curr;q \times r} \cap \overline{\mathbb{B}\left(K^{(0)}, h\right)}} \sigma_{max}(K) = \sigma_{max}(K_*), \tag{15}$$

where F_* is an optimal LQR state-feedback (SF) controller for the system (A, B, I_p). We conclude that $\sigma_{max}(F_*) \leq \sigma_{max}(K_*) \leq \sigma_{max}\left(K^{(\text{best})}\right)$. Thus, $\sigma_{max}(F_*)$ is a lower bound for $\sigma_{max}\left(K^{(best)}\right)$ and can serve as a good estimator for it, in order to quantify the convergence of the algorithm to the global minimum (as is evidently seen from **Table 1** in many cases) and in order to stop the algorithm earlier, since $\sigma_{max}(F_*)$ can be calculated in advance. The lower bound appears in the last column of **Table 1**.

For all the systems, we had (A, B), $\left(A^T, C^T\right)$ controllable, except for ROC1 and ROC2. All the systems are unstable, except for AC6, AC15, and NN16 which are stable, but not α-stable, for α given in the stability margin column.

The experiments were executed on:
Computer: LAPTOP-GULIHG OV, ASUSeK COMPUTER, INC.
TUF GAMING FX504GM-FX80GM.
Processor: Intel(R) Core(TM) i7-8750H CPU@2.20GHz.
Platform: MATLAB, Version R2018b Win 64.

5.1 Conclusions from the experiments

Regarding the experimental results in **Table 2** and the comparison between the RS algorithm and the MC algorithm, we conclude:

1. The RS algorithm performs in magnitude better than the MC algorithm for the systems: AC1, AC11, HE1, HE4, ROC1, ROC4, TF1, and NN5.

2. The MC algorithm performs in magnitude better than the RS algorithm for the systems AC5 and DIS5.

3. The MC algorithm performs slightly better than the RS algorithm for systems HE3 and NN16.

Regarding the experimental results in **Table 2** and the performance of the RS + MC algorithm, we conclude:

1. The RS + MC algorithm performs better than each algorithm separately, for systems AC6, HE4, TF1, NN13, NN16, and NN17.

2. The RS + MC algorithm performs better than the RS algorithm for systems AC5, HE3, and DIS5.

3. The RS + MC algorithm performs exactly as the RS algorithm for systems AC1, AC11, DIS4, HE1, ROC1, ROC4, and NN5. This observation assesses the claim for convergence of the RS algorithm to global optimum.

System	$\sigma_{\max}^{(\text{best})}$ for (A, B, C) RS Algo.	$\sigma_{\max}^{(\text{best})}$ for (A, B, C) MC Algo.	$\sigma_{\max}^{(\text{best})}$ for (A, B, C) RS + MC Algo.	RS Algo. CPU time [sec]	MC Algo. CPU time [sec]	RS + MC Algo. CPU time [sec]
AC1	$1.9207 \cdot 10^3$	$1.0701 \cdot 10^4$	$1.9207 \cdot 10^3$	2.9843	0.0468	3.0312
AC5	$1.5888 \cdot 10^9$	$2.5905 \cdot 10^8$	$2.5905 \cdot 10^8$	2.0156	0.4062	2.2500
AC6	$6.1449 \cdot 10^2$	$6.5913 \cdot 10^2$	$6.1389 \cdot 10^2$	5.1250	0.2500	5.1718
AC11	$2.4234 \cdot 10^3$	$1.2968 \cdot 10^4$	$2.4234 \cdot 10^3$	2.9531	0.0468	3.0000
HE1	$9.1253 \cdot 10^2$	$1.0968 \cdot 10^3$	$9.1253 \cdot 10^2$	2.2812	0.0468	2.3437
HE3	$8.6808 \cdot 10^4$	$7.1816 \cdot 10^4$	$8.1737 \cdot 10^4$	13.4843	0.2343	13.7656
HE4	$5.2247 \cdot 10^4$	$1.1817 \cdot 10^6$	$3.1783 \cdot 10^4$	10.9687	0.0468	11.2656
ROC1	$6.6239 \cdot 10^5$	$1.5906 \cdot 10^7$	$6.6239 \cdot 10^5$	7.6250	0.1250	7.7500
ROC4	$5.9923 \cdot 10^5$	$1.2273 \cdot 10^6$	$5.9923 \cdot 10^5$	5.3750	0.0625	5.4218
DIS4	$1.7590 \cdot 10^2$	$2.0376 \cdot 10^2$	$1.7590 \cdot 10^2$	7.2187	0.1250	7.2656
DIS5	$2.8686 \cdot 10^8$	$3.2079 \cdot 10^7$	$3.2079 \cdot 10^7$	1.8593	0.1562	1.9687
TF1	$1.9289 \cdot 10^4$	$1.1230 \cdot 10^5$	$1.9270 \cdot 10^4$	5.0000	0.0625	5.0468
NN5	$9.6780 \cdot 10^5$	$1.3372 \cdot 10^6$	$9.6780 \cdot 10^5$	1.2968	0.1718	1.3593
NN13	$2.0521 \cdot 10^2$	$2.6553 \cdot 10^2$	$1.7953 \cdot 10^2$	2.3281	0.2656	2.4843
NN16	$6.4416 \cdot 10^2$	$6.0032 \cdot 10^2$	$6.0030 \cdot 10^2$	6.4062	0.5468	6.7343
NN17	$3.6805 \cdot 10^3$	$4.9674 \cdot 10^3$	$3.6787 \cdot 10^3$	0.9375	0.2812	1.1718

Table 2.
Experimental results.

4. The RS + MC algorithm performs exactly as the MC algorithm for systems AC5 and DIS5.

Regarding improvements over the starting point, we had:

1. The RS algorithm failed in finding any improvement over $\sigma_{\max}\left(K^{(0)}\right)$ for systems AC5 and DIS5.

2. The MC algorithm failed in finding any improvement over $\sigma_{\max}\left(K^{(0)}\right)$ for systems AC1, AC11, ROC1, and ROC4. This observation assesses the heuristic that it is better to start with a SOF that brings the poles of the closed loop as close as possible to the boundary of the disk $\mathbb{D}_\alpha$.

3. The RS + MC algorithm improved $\sigma_{\max}\left(K^{(0)}\right)$ in any case.

Regarding the assessment of convergence to a global minimum, we had the following results:

1. The RS algorithm and the RS + MC algorithm had very close values of $\sigma_{\max}\left(K^{(best)}\right)$ (or exactly the same value) which is very close to the lower bound, for systems AC1, AC6, HE1, HE3, HE4, ROC1, DIS4, NN5, and NN16.

2. The MC algorithm achieved a very close value of $\sigma_{\max}\left(K^{(best)}\right)$ to the lower bound, for the systems AC6, HE3, DIS5, NN5, and NN16.

As was expected, the MC algorithm seems to perform better locally, while the RS algorithm seems to perform better globally. Thus, practically, the best approach is to apply the RS algorithm in order to find a close neighborhood of a global minimum and then to apply the MC algorithm on the result, for the local optimization, as is evidently seen from the performance of the RS + MC algorithm.

5.2 Some specific results

Example 5.1. For the HE4 system with

$A = [A_1 A_2]$, where

$$A_1 = \begin{bmatrix} 0.99999985 & 0.00000014 & 0.00001538 & 0.00988556 \\ -0.00000082 & 0.99999927 & 0.00944714 & -0.00015179 \\ -0.00016276 & -0.00014358 & 0.89058607 & -0.02380208 \\ -0.00002661 & 0.00002887 & 0.00410983 & 0.98016538 \\ -0.00002930 & -0.00000576 & -0.01923117 & -0.00428369 \\ -0.32100350 & -0.00003189 & -0.00471073 & 0.02122199 \\ 0.00098904 & 0.32051910 & -0.02076100 & -0.00474084 \\ -0.01910442 & 0.01711196 & 0.00004146 & -0.00066123 \end{bmatrix},$$

$$A_2 = \begin{bmatrix} 0.00053188 & 0.00000088 & 0.00000089 & -0.00000005 \\ 0.00059005 & 0.00000508 & -0.00000451 & 0.00000034 \\ -0.00060285 & 0.00100715 & -0.00089937 & 0.00006730 \\ -0.00000054 & 0.00016706 & 0.00018078 & -0.00001158 \\ 0.99268256 & 0.00018120 & -0.00003706 & 0.00002045 \\ -0.00008474 & 0.99978709 & -0.00020742 & 0.00015757 \\ 0.00841925 & 0.00020153 & 0.99963047 & 0.00000293 \\ -0.00000005 & 0.00013964 & -0.00000914 & 0.99709909 \end{bmatrix},$$

$$B = \begin{bmatrix} -0.00000097 & 0.00002352 & -0.00000082 & -0.00000055 \\ 0.00000679 & 0.00000357 & -0.00013147 & -0.00000146 \\ 0.00117865 & 0.00072316 & -0.02601681 & -0.00016953 \\ -0.00035692 & 0.00470509 & 0.00008442 & -0.00000015 \\ 0.00302236 & 0.00013040 & -0.00468257 & -0.00205817 \\ 0.00286635 & -0.00539604 & -0.00009665 & 0.00000026 \\ -0.00018970 & 0.00014386 & -0.00517989 & 0.00234114 \\ -0.04813606 & -0.00000574 & -0.00000060 & -0.00000001 \end{bmatrix},$$

$$C = [C_1 C_2], \text{ where}$$

$$C_1 = \begin{bmatrix} -0.00000185 & 0.00001067 & -0.00071398 & 0.00083459 \\ 0.99999992 & 0.00000007 & 0.00000769 & 0.00494278 \\ -0.00000041 & 0.99999963 & 0.00472357 & -0.00007589 \\ -0.00001393 & -0.00000365 & -0.00972548 & -0.05509146 \\ -0.00008138 & -0.00007179 & 0.94529303 & -0.01190104 \\ -0.00001330 & 0.00001443 & 0.00205491 & 0.99008269 \end{bmatrix},$$

$$C_2 = \begin{bmatrix} 0.00022183 & 0.05942943 & 0.05327854 & -0.99534942 \\ 0.00026594 & 0.00000044 & 0.00000044 & -0.00000002 \\ 0.00029502 & 0.00000254 & -0.00000225 & 0.00000017 \\ 0.99634129 & 0.00008613 & -0.00002336 & 0.00001053 \\ -0.00030142 & 0.00050357 & -0.00044968 & 0.00003365 \\ -0.00000027 & 0.00008353 & 0.00009039 & -0.00000579 \end{bmatrix},$$

with

$$\sigma(A) = \begin{Bmatrix} 0.638773652186517, 0.847768449750652 \\ 1.002501752901569, 0.960047795833900 \\ 0.990353602254223 \end{Bmatrix},$$

we had the following results:
by the RS algorithm for minimal-gain SOF (see [10])

$$\begin{cases} K^{(0)} = \left[K_1^{(0)} K_2^{(0)}\right], \text{ where} \\ K_1^{(0)} = \begin{bmatrix} -0.05281866 & 0.30558099 & -0.04123125 \\ -0.74370605 & -0.07272045 & 0.16180699 \\ -0.34989799 & -0.70937255 & -0.03438071 \\ 0.36682921 & -0.55329174 & -0.42930790 \end{bmatrix}, \\ K_2^{(0)} = \begin{bmatrix} -0.07894070 & -0.83106320 & 0.63513665 \\ -0.57049060 & 0.02944824 & 0.03985277 \\ 0.01822942 & -0.40097959 & -0.32739026 \\ -0.32035712 & 0.23550532 & 0.55239497 \end{bmatrix}, \\ \sigma\left(A - BK^{(0)}C\right) = \begin{Bmatrix} 0.88529956, 0.98303140, \\ 0.99890558 \pm 0.00709027i, \\ 0.99895917 \pm 0.00548801i, \\ 0.99648905, 0.99228590 \end{Bmatrix}, \\ \sigma_{\max}^{(0)} = 4.1660 \cdot 10^6, \left\|K^{(0)}\right\| = 1.1052, \end{cases}$$

by the RS Algorithm1

$$\left\{\begin{aligned}
K^{(best)} &= \left[K_1^{(best)} K_2^{(best)}\right], \text{where} \\
K_1^{(best)} &= \begin{bmatrix} 2.60115550 & 0.71670943 & 1.38518242 \\ -1.21472623 & 7.41955425 & -2.28748737 \\ 0.08032866 & -1.01678491 & -3.50944968 \\ 0.88639191 & -0.92895747 & 0.77107501 \end{bmatrix}, \\
K_2^{(best)} &= \begin{bmatrix} -0.43606038 & -0.52797919 & -1.91992615 \\ -0.37181168 & -0.49434738 & -1.97765275 \\ -0.43167652 & 2.40298411 & -0.14274690 \\ -1.34628983 & -2.73288946 & -0.19682963 \end{bmatrix}, \\
\sigma\left(A - BK^{(best)}C\right) &= \left\{\begin{array}{l} 0.86726666, 0.98939915, \\ 0.99600356 \pm 0.01886288i, \\ 0.99670041 \pm 0.00199237i, \\ 0.97791267 \pm 0.01292326i, \\ \end{array}\right\}, \\
\sigma_{\max}^{(best)} &= 5.2247 \cdot 10^4, \left\|K^{(best)}\right\| = 8.1964,
\end{aligned}\right.$$

by the MC Algorithm 2

$$\left\{\begin{aligned}
K^{(best)} &= \left[K_1^{(best)} K_2^{(best)}\right], \text{where} \\
K_1^{(best)} &= \begin{bmatrix} 0.16219293 & 0.22685502 & -1.41842788 \\ 0.10558055 & 0.04299252 & -1.12924145 \\ -0.01638169 & -0.08184317 & -0.14133958 \\ -0.03859594 & 0.07947463 & 0.12534983 \end{bmatrix}, \\
K_2^{(best)} &= \begin{bmatrix} -0.23992446 & -0.17536269 & -0.44875450 \\ -0.14481130 & -0.18150136 & -0.27764220 \\ -0.04123859 & 0.04014410 & 0.05631322 \\ -0.06158264 & 0.05048672 & 0.16208075 \end{bmatrix}, \\
\sigma\left(A - BK^{(best)}C\right) &= \left\{\begin{array}{l} 0.53091472, 0.95364142, \\ 0.98741833, 0.99897830, \\ 0.99717581 \pm 0.00564301i, \\ 0.95033459 \pm 0.08956374i, \\ \end{array}\right\}, \\
\sigma_{\max}^{(best)} &= 1.1817 \cdot 10^6, \left\|K^{(best)}\right\| = 195.3621,
\end{aligned}\right.$$

and by RS + MC Algorithms 1 and 2:

$$\begin{cases} K^{(best)} = \left[K_1^{(best)} K_2^{(best)}\right], \text{where} \\ K_1^{(best)} = \begin{bmatrix} 0.16101573 & 0.52768736 & -1.99112247 \\ -1.8453102 & 11.28218548 & -3.45382160 \\ 0.45368113 & -0.97316671 & -6.96467764 \\ 0.57106266 & -0.53606197 & -0.45209883 \end{bmatrix}, \\ K_2^{(best)} = \begin{bmatrix} -0.67588625 & -0.18705486 & -0.28941025 \\ -0.69074912 & -0.26655181 & 0.20004806 \\ -0.83280356 & 3.20877857 & -0.31860927 \\ -3.09450298 & -0.73194640 & -0.07799223 \end{bmatrix}, \\ \sigma\left(A - BK^{(best)}C\right) = \begin{Bmatrix} 0.98195937 \pm 0.03551664i, \\ 0.99264852 \pm 0.01953917i, \\ 0.99354901 \pm 0.00357529i, \\ 0.99828371, 0.98490431 \end{Bmatrix} \\ \sigma_{\max}^{(best)} = 3.1783 \cdot 10^4, \left\|K^{(best)}\right\| = 12.1029. \end{cases}$$

6. Concluding remarks

The Ray-Shooting Method is a powerful tool, since it practically solves the problem of LQR via SOF, for real-life discrete-time LTI systems. The proposed hybrid algorithm RS + MC has good performance in terms of run-time, in terms of the quality of controllers (by reducing the starting point LQR functional value and by reducing the controller norm) and in terms of the success rate in finding a starting point feasible with respect to the needed α-stability. The RS + MC algorithm has a proof of convergence in probability to a global minimum (as is evidently seen from the experiments). This enlarges the practicality and scope of the Ray-Shooting Method in solving hard complexity control problems, and we expect to receive more results in this direction.

Author details

Yossi Peretz
Computer Sciences Department, Lev Academic Center, Jerusalem College of Technology, Jerusalem, Israel

*Address all correspondence to: yosip@g.jct.ac.il

References

[1] Camino JF, Zampieri DE, Peres PLD. Design of a vehicular suspension controller by static output feedback. Proceedings of the American Control Conference; San Diego, California; June 1999

[2] Nemirovskii A. Several NP-hard problems arising in robust stability analysis. Mathematics of Control, Signals, and Systems. 1993;**6**:99-105

[3] Blondel V, Tsitsiklis JN. NP-hardness of some linear control design problems. SIAM Journal on Control and Optimization. 1997;**35**(6):2118-2127

[4] Vidyasagar M, Blondel VD. Probabilistic solutions to some NP-hard matrix problems. Automatica. 2001;**37**: 1397-1405

[5] Tempo R, Calafiore G, Dabbene F. Randomized Algorithms for Analysis and Control of Uncertain Systems. London: Springer-Verlag; 2005

[6] Tempo R, Ishii H. Monte Carlo and Las Vegas randomized algorithms for systems and control. European Journal of Control. 2007;**13**:189-203

[7] Arzelier D, Gryazina EN, Peaucelle D, Polyak BT. Mixed LMI/ randomized methods for static output feedback control. In: Proceedings of the American Control Conference. Baltimore, Maryland, USA: Institute of Electrical and Electronics Engineers (IEEE); 2010. pp. 4683-4688

[8] Syrmos VL, Abdallah C, Dorato P, Grigoradis K. Static output feedback: A survey. Automatica. 1997;**33**:125-137

[9] Sadabadi MS, Peaucelle D. From static output feedback to structured robust static output feedback: A survey. Annual Reviews in Control. 2016;**42**:11-26

[10] Peretz Y. A randomized approximation algorithm for the minimal-norm static-output-feedback problem. Automatica. 2016;**63**:221-234

[11] Peretz Y. On applications of the ray-shooting method for structured and structured-sparse static-output-feedbacks. International Journal of Systems Science. 2017;**48**(9):1902-1913

[12] Peretz Y. On application of the ray-shooting method for LQR via static-output-feedback. MDPI Algorithms Journal. 2018;**11**(1):1-13

[13] Peretz Y. A randomized algorithm for optimal PID controllers. MDPI Algorithms Journal. 2018;**11**(81):1-15

[14] Peretz Y, Moyal S, Merzbach O. A randomized algorithm for robust stabilization via static-output-feedback. In: IEEE International Conference on the Science of Electrical Engineering in Israel (ICSEE); December 12-14, 2018; Eilat, Israel

[15] Moerder D, Calise A. Convergence of numerical algorithm for calculating optimal output feedback gains. IEEE Transactions on Automatic Control. 1985;**AC-30**:900-903

[16] Johnson T, Athans M. On the design of optimal dynamic compensators for linear constant systems. IEEE Transactions on Automatic Control. 1970;**AC-15**:658-660

[17] Levine W, Athans M. On the determination of the optimal constant output feedback gains for linear multivariables systems. IEEE Transactions on Automatic Control. 1970;**AC-15**:44-48

[18] Levine W, Johnson TL, Athans M. Optimal limited state variable feedback controllers for linear systems. IEEE Transactions on Automatic Control. 1971;**AC-16**:785-793

[19] Peres PLD, Geromel J, de Souza S. Optimal H_2 control by output feedback. In: Proceedings of the 32nd IEEE conference on Decision and Control, San Antonio, Texas, USA. 1993. pp. 102-107

[20] Iwasaki T, Skelton RE. All controllers for the general H_∞ control problem: LMI existence conditions and state space formulas. Automatica. 1994; **30**:1307-1317

[21] Iwasaki T, Skelton RE. Linear quadratic suboptimal control with static output feedback. Systems & Control Letters. 1994;**23**(6):421-430

[22] Henrion D, Loefberg J, Kočvara M, Stingl M. Solving polynomial static output feedback problems with PENBMI. In: Proceedings of the joint IEEE Conference on Decision and Control and European Control Conference, Sevilla, Spain. 2005

[23] Yang K, Orsi R. Generalized pole placement via static output feedback: A methodology based on projections. Automatica. 2006;**42**:2143-2150

[24] Gumussoy S, Henrion D, Millstone M, Overton ML. Multiobjective robust control with HIFOO 2.0. Proceedings of the IFAC Symposium on Robust Control Design; Haifa, Israel. 2009

[25] Bistritz Y. Zero location with respect to the unit circle of discrete-time linear system polynomials. Proceedings of the IEEE. 1984;**72**(9):1131-1142

[26] Leibfritz F. COMPleib: Constrained matrix-optimization problem library—A collection of test examples for nonlinear semidefinite programs, control system design and related problems. Tech.-Report. 2003

[27] Leibfritz F, Lipinski W. COMPleib 1.0—User manual and quick reference. Tech.-Report. 2004

[28] Leibfritz F. Description of the benchmark examples in COMPleib 1.0. Tech.-Report. 2003

[29] Gahinet P, Apkarian P. Frequency-domain tuning of fixed-structure control systems. In: Proceedings of 2012 UKACC International Conference on Control; 3–5 September 2012; Cardiff, UK. IEEE. 2012. pp. 178-183