AMO–Advanced Modeling and Optimization, Volume 20, Number 2, 2018

On the Solution of the Eigenvalue Assignment Problem by Derivative–Free Optimization Methods

El-Sayed M.E. Mostafa and Fatma F.S. Omar

Dept. of Mathematics and Computer Sci., Faculty of Science, Alexandria University, Moharam Bek 21511, Alexandria, Egypt (emostafa@alex-sci.edu.eg)

Dept. of Mathematics and Computer Sci., Faculty of Science, Alexandria University, Moharam Bek 21511, Alexandria, Egypt (ffsaidm@yahoo.com)

Abstract

In this article, we consider the output feedback eigenvalue assignment problem for continuous and discrete–time control systems. This problem is formulated as unconstrained matrix optimization problems and tackled by the Nelder–Mead simplex method and particle swarm optimization method. The two methods are extended to compute the approximate solutions of the eigenvalue assignment problem for the particular cases of decentralized and periodic control systems. The performance of the methods is demonstrated numerically on several test problems from the benchmark collection [22] as well as other test examples from the system and control literature.

AMS subject classification: 93D15, 93B55, 93B60, 90C56

Key words: the eigenvalue assignment problem, output feedback design, Nelder-Mead simplex method, particle swarm optimization

1 Introduction

In this article, we consider the following unconstrained optimization problem

$$\min_{K \in \mathbb{R}^{p \times r}} \; f(K), \tag{1}$$

where $f : \mathbb{R}^{p \times r} \to \mathbb{R}$ is generally non-convex and non-smooth, and K is a matrix variable representing the unknown. As an example of the optimization problem (1) consider the

output feedback eigenvalue assignment problem (EAP) for continuous and discrete–time systems. The EAP is an important problem in systems and control literature; see e.g. the two surveys [5, 32].

A related problem to the EAP is the optimal output feedback design problem; see e.g. the survey [32], which consists of finding an output feedback gain matrix variable that minimizes a quadratic cost function in such a way that this output feedback gain leads to an asymptotically stable closed loop control system. In fact, both problems have been formulated as optimizations problems and have been tackled by various optimization methods; see e.g. [5, 8, 26, 27, 32] and the references therein. As one solves the optimal output feedback design problem the optimization method typically requires an initial stabilizing output feedback gain. At this stage it appears the need of the numerical solver of the EAP to provide such a stabilizing output feedback controller. In addition, the EAP has been of great importance in many applications in engineering and finance. In optimal control literature there are numerous research articles that have been written on different forms of the EAP for discrete or continuous–time systems among them [1, 3, 5, 6, 9, 10, 11, 14, 15, 16, 17, 18, 20, 21, 24, 25, 27, 30, 31, 34, 36, 38]. The problem in its simplest form was first addressed by Wonham [36] in 1967. Since then, huge number of publications have been proposed to tackle different formulations of this problem. In 1976 Moore [25] has given a characterization for the class of all closed loop eigenvectors of the state feedback problem. One of the first to address EAP by output feedback was Davison [11] and was extended by Davison and Chatterjee [9] and Sridhar and Lindhor [30]. Moreover, Fu [14] has proven that the EAP is NP–hard.

Recently, Mostafa et al. [26, 27] tackled the EAP by gradient-based optimization methods. Optimization techniques have been also used for eigenvalue assignment via output feedback as demonstrated in [8]. Due to the nature of this problem of non-smoothness, where the objective function is not everywhere differentiable, gradient-based methods for example face difficulties to converge to a local solution. In particular, Newton's method with trust region [38] fails to converge with problems having repeated eigenvalues at the solution. This clearly limits algorithms that require derivatives of the objective function to a class of problems where the desired eigenvalues are distinct. This issue motivates us to use derivative-free optimization methods to tackle this problem. Derivative-free methods have been appeared since the 1950s (see [23]). The attractive side of these methods is to avoid calculating gradients. In fact, derivative-free methods are quite effective in solving several types of difficult optimization problems. We focus in this work on the Nelder–Mead (NM) simplex method and particle swarm optimization (PSO) method; see e.g. [12] as a recent survey.

This article is organized as follows. The next section introduces the formulations of the EAP problem for discrete and continuous–time systems together with some basic definitions. In Section 3 we introduce NM simplex method for computing an approximate solution to the considered minimization problems. In Section 4 we apply the PSO method to tackle the same problems. In Section 5 the two methods are modified to approximately solve the EAP for decentralized continuous–time systems. Section 6 extends the two methods to find an approximate solution to the optimization problem corresponding to the EAP for periodic discrete–time systems. In Section 7 we demonstrate the performance of the methods on several test examples from the literature. Then we end with a conclusion.

Notations: The eigenvalues of a matrix $M \in \mathbb{R}^{m\times m}$ are denoted by $\lambda_i(M),\ i = 1, \ldots, m$.

The Greek letter $\rho(M)$ denotes the spectral radius of a square matrix M. Sometimes and for the sake of simplicity we skip the arguments of the considered functions, e.g., we use v to denote $vec(K)$ that stretches the matrix variable K into long column vector v.

2 Problem formulation and basic definitions

Consider the linear time-invariant control system with the following state space realization

$$\delta x = Ax + Bu, \quad x(0) = x_0, \quad y = C\,x, \tag{2}$$

where δ is an operator indicating the time derivative d/dt for continuous–time systems and a forward unit time shift for discrete–time systems. The vectors $x \in \mathbb{R}^m$, $u \in \mathbb{R}^p$, and $y \in \mathbb{R}^r$ are the state, the control input, and the measured output vectors, respectively. Moreover, $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{m\times p}$ and $C \in \mathbb{R}^{r\times m}$ are given constant matrices. We consider the control law $u = Ky$ to close the above control system which implies that

$$\delta x = (A + BKC)x := A(K)x, \quad x(0) = x_0, \tag{3}$$

where $A(K) = A + BKC$ is the closed-loop system matrix and $K \in \mathbb{R}^{p\times r}$ is the output feedback gain matrix which represents the unknown.

Let us consider in the following some basic definitions that will be used throughout the work.

Definition 2.1 *The spectral radius of a matrix $A \in \mathbb{C}^{m\times m}$ with eigenvalues $\lambda_1, \ldots, \lambda_m$ is defined as*

$$\rho(A) = \max\ \{\ |\lambda_i|\ :\ i \in \{1, 2, \ldots, m\}\}.$$

Definition 2.2 *The spectral abscissa of a matrix $A \in \mathbb{C}^{m\times m}$ having eigenvalues $\lambda_1, \ldots, \lambda_m$ is defined as*

$$\mu(A) = \max\{\mathrm{Re}(\lambda_i)\ :\ i \in \{1, \ldots, m\}\}. \tag{4}$$

It is well-known fact that the closed-loop system (3) of the continuous–time case is asymptotically stable if and only if $\mu(A(K)) < 0$ while the asymptotic stability for the discrete–time system requires $\rho(A(K)) < 1$.

The eigenvalue assignment problem is to find an output feedback gain matrix K such that the closed-loop system is in satisfactory stage by shifting the controllable eigenvalues to desirable locations in the complex plane. In particular, for discrete–time systems we require the spectral radius of the closed-loop system matrix $A(K)$ to be strictly within the open unit desk. In other words, let A, B, and C be given constant matrices. The attempt is to find $K \in \mathbb{R}^{p\times r}$ that solves the following optimization problem:

$$\min\ f(K) = \rho(A + BKC) \quad \text{s.t.} \qquad g(K) = \rho(A + BKC) - (1 - \tau) \le 0, \tag{5}$$

where $\tau \in (0, 1)$ is a given constant and $f : \mathbb{R}^{p\times r} \to \mathbb{R}_+$ is generally nonconvex and nonsmooth function.

The inequality constraint of the problem (5) represents a cut to the objective function, where the major task is to find a feasible point K to the problem (5). In fact this problem can be regarded as the following unconstrained minimization problem:

$$\min \ f(K) = \rho(A + BKC) \tag{6}$$

while we make sure of fulfilling the associated inequality constraint during the iterations of the proposed method. Obviously we can stop the unconstrained minimization solver as soon as the objective function becomes strictly less than one. However, we can run the method until it achieves a satisfactory stability margin as indicated by the inequality constraint. From a computational point of view, the unconstrained minimization solver when applied to the problem (6) it typically reduces the objective function where the imposed cut is fulfilled after finite number of iterations. Therefore, the focus is given to approximately solve the unconstrained minimization problem (6).

On the other side, the EAP for continuous–time control systems is stated as follows. Let A, B, and C be given constant matrices and let $\tilde{\lambda}_1, \tilde{\lambda}_2, \ldots, \tilde{\lambda}_m \in \mathbb{C}$ be given desired eigenvalues. The EAP is to find a matrix variable $K \in \mathbb{R}^{p\times r}$ such that

$$\lambda_i(A + BKC) = \tilde{\lambda}_i, \quad i = 1, 2, \ldots, m.$$

The EAP for continuous–time systems can be equivalently rewritten as: Find $K \in \mathbb{R}^{p\times r}$ that solves the following nonlinear least–squares problem; see [26, 38]:

$$\min \quad \hat{f}(K) = \frac{1}{2}\sum_{i=1}^{m}(\lambda_i(A(K)) - \tilde{\lambda}_i)^*(\lambda_i(A(K)) - \tilde{\lambda}_i), \tag{7}$$

where the superscript $*$ denotes the complex conjugate.

Note that, optimization methods that require derivatives of the objective function when applied to the problem (7) face difficulty because the eigenvalues of $A + BKC$ may not be differentiable everywhere. Consequently, algorithms that require derivatives of the objective function are only appropriate for problems where the desired eigenvalues are distinct. In this work we will apply the Nelder–Mead and particle swarm optimization methods to tackle the two problems (6) and (7). In addition, we extend the two methods to tackle the EAP for decentralized and periodic control systems.

3 Nelder–Mead method for EAP

Nelder-Mead (NM) simplex method [28] is one of the classical direct search methods. This method was successful in solving various practical optimization problems. It is based on comparing the objective function of the minimization problem on a finite set of vertices. For an optimization problem with n variables the NM method searches for a minimizer of the objective function by evaluating the objective on a set of $n + 1$ vertices. The method continuously forms new simplexes by replacing the vertex having the worst objective value with a new one. The new vertex is generated by reflection, expansion, and contraction operations. Each iteration of the NM method begins with a non-degenerate simplex defined by the given vertices and the associated values of the objective function.

In our case, let us first focus on the minimization problem (6). For simple treatment with the NM method and its vertices let us regard the $p \times r$ matrix space as a space of $\mathbb{R}^{p\times r}$ vectors, where the matrix variable $K \in \mathbb{R}^{p\times r}$ is stretched into a long column vector $v = vec(K) \in \mathbb{R}^n$ where $n = p \cdot r$. The initialization of the method is done by choosing an initial guess K_0, where we assume that $\rho(A(K_0)) > 1$. Then we set $v_0 = vec(K_0)$ and generate the remaining vertices of the starting simplex as

$$v_i = v_0 + \hat{\sigma}\, e_i, \qquad i = 1, 2, \ldots, n, \tag{8}$$

where $\hat{\sigma} > 0$ is a given constant and e_i is the i^{th} column of the $n \times n$ identity matrix.

Let us be at iteration k and assume that the vertices $\{v_{i,k} \ : \ i \in \{0, \ldots, n\}\}$ are sorted according to the objective function values as

$$f(K_{0,k}) \leq f(K_{1,k}) \leq \ldots \leq f(K_{n,k}), \tag{9}$$

where as we compute the objective function we reshape the vectors $v_{i,k}$'s back into the corresponding matrices $K_{i,k}$'s. We define the set of vertices

$$\mathcal{S}(k) = \{v_{0,k}, v_{1,k}, \ldots, v_{n,k}\}, \tag{10}$$

and denote $v_k^b = vec(K_k^b)$ and $v_k^w = vec(K_k^w)$ the best and worst vertices of the set $\mathcal{S}(k)$ such that

$$f(K_k^b) = \min_{i=0,\ldots,n} f(K_{i,k}), \ f(K_k^w) = \max_{i=0,\ldots,n} f(K_{i,k}). \tag{11}$$

Moreover, let $\bar{v}_k$ be the centroid of the face of the simplex calculated by the vertices of the simplex $\mathcal{S}(k)$ except v_k^w. Hence,

$$\bar{v}_k = \frac{1}{n}\left(\sum_{i=0}^{n} v_{i,k} - v_k^w\right). \tag{12}$$

If v_k^w is the vertex corresponding to the largest value of the objective function among the vertices of a simplex, one expects that the vertex v_k^{re} obtained by reflecting the vertex v_k^w in the opposite face to have the smallest value. If this is the case, then we might construct a new simplex by rejecting the vertex v_k^w from the simplex and inserting the new vertex v_k^{re}. In other words, the NM method seeks to replace v_k^w by a vertex with a lower objective function value. The simplex is updated in five different ways during an iteration. Except in the case of a shrink, the worst vertex of the simplex at iteration k is replaced by one of the reflection, expansion, or contraction points, each being associated with a scalar parameter α (reflection), β (expansion), γ (contraction), and δ (shrink). The values of these parameters are such that $\alpha > 0$, $\beta > 1$, $0 < \gamma < 1$ and $0 < \delta < 1$. The overall algorithm for computing an approximate solution to the problem (6) is illustrated in the following lines.

Algorithm 3.1 (Nelder–Mead method for solving Problem (6))

1. Initialization: Let A, B, C be given constant matrices, and let $\alpha > 0, \beta > 1$, $0 < \gamma < 1$ and $0 < \delta$, $\texttt{Tol}\rho < 1$ be given constants. Choose $K_0 \in \mathbb{R}^{p\times r}$ then reshape it as a column vector $v_0 \in \mathbb{R}^n$ to be one of the initial simplex vertices. Then generate the remaining n vertices as explained above and compute $f(K_{i,0}),\ i = 0, \ldots, n$. Arrange the $n+1$ vertices so that (9) holds. Identify v_0^b, v_0^w, and compute $f(K_0^b)$. If $f(K_0^b) < \texttt{Tol}\rho$, stop; otherwise set $k \leftarrow 0$ and go to next step.

2. While $f(K_k^b) \geq \texttt{Tol}\rho$ and $k < k_{\max}$, do

   (i) Compute $\bar{v}_{k+1}$ using (12), set $v_{k+1}^{re} = \bar{v}_{k+1} + \alpha(\bar{v}_{k+1} - v_{k+1}^w)$, reshape v_{k+1}^{re} as K_{k+1}^{re} and compute $f_{k+1}^{re} = f(K_{k+1}^{re})$.

   (ii) (Reflection step) If $f(K_{k+1}^b) \leq f_{k+1}^{re} < f(K_{k+1}^w)$, set $v_{k+1}^w := v_{k+1}^{re}$; and go to step (vii).

   (iii) (Expansion step) If $f_{k+1}^{re} < f(K_{k+1}^b)$ then compute $v_{k+1}^e = \bar{v}_{k+1} + \beta(\bar{v}_{k+1} - v_{k+1}^w)$, reshape v_{k+1}^e as K_{k+1}^e and compute $f_{k+1}^e = f(K_{k+1}^e)$. If $f_{k+1}^e < f_{k+1}^{re}$, set $v_{k+1}^w := v_{k+1}^e$, otherwise $v_{k+1}^w := v_{k+1}^{re}$ and go to step (vii).

   (iv) (Outside contraction step) If $f(K_{i,k+1}) \leq f_{k+1}^{re} < f(K_{k+1}^w)\ \forall\ i = 0, \ldots, n,\ i \neq w$, then compute $v_{k+1}^{oc} = \bar{v}_{k+1} + \gamma(v^{re} - \bar{v}_{k+1})$, reshape v_{k+1}^{oc} as K_{k+1}^{oc} and compute $f_{k+1}^{oc} = f(K_{k+1}^{oc})$. If $f_{k+1}^{oc} \leq f(K_{k+1}^{re})$, set $v_{k+1}^w := v_{k+1}^{oc}$ and go to step (vii) otherwise go to step (vi).

   (v) (Inside contraction step) If $f_{k+1}^{re} \geq f(K_{k+1}^w)$, then compute $v_{k+1}^{ic} = \bar{v}_{k+1} - \gamma(v^{re} - \bar{v}_{k+1})$ reshape v_{k+1}^{ic} as K_{k+1}^{ic} and compute $f_{k+1}^{ic} = f(K_{k+1}^{ic})$. If $f_{k+1}^{ic} \leq f(K_{k+1}^w)$, set $v_{k+1}^w := v_{k+1}^{ic}$ and go to step (vii) otherwise go to step (vi).

   (vi) (Shrinking step) Set $v_{i,k+1} = v_{k+1}^b + \delta(v_{i,k+1} - v_{k+1}^b)$, reshape $v_{i,k+1}$ as $K_{i,k+1}$ and compute $f(K_{i,k+1})$ for all $i = 0, \ldots, n,\ i \neq b$.

   (vii) Arrange the $n+1$ vertices so that (9) holds and identify v_{k+1}^b and v_{k+1}^w.

   (viii) Reshape v_{k+1}^b as K_{k+1}^b and compute $f(K_{k+1}^b)$. If $f(K_{k+1}^b) < \texttt{Tol}\rho$, stop; otherwise set $k \leftarrow k+1$ and go to step (i).

   End (do)

Remark 3.1 *As explained in the introduction, numerical methods that seek optimal solution of the static output feedback design problem require an initial stabilizing output feedback gain matrix, i.e. $\hat{K}$ such that $\rho(A(\hat{K})) < 1$. Such a feedback controller can be provided by one of the considered methods.*

Therefore, it is reasonable to stop the NM method as soon as $f(K_k^b) \leq \texttt{Tol}\rho$, where $\texttt{Tol}\rho \in (0,1)$ is a small number representing the stability margin.

In the following lines we list minor changes made on Algorithm 3.1 to tackle the minimization problem (7) of the EAP for continuous–time systems.

- The initialization step: Similar to Algorithm 3.1, except we replace the objective function f by $\hat{f}$ as defined in the problem (7). Moreover, let $\tilde{\lambda}_1, \tilde{\lambda}_2, \ldots, \tilde{\lambda}_m \in \mathbb{C}$ be given desired eigenvalues. Replace the constant $\texttt{Tol}\rho$ by a given constant $\texttt{AverFun} \in (0,1)$ for the stopping condition.

- Replace the convergence criterion 'While $f(K_k^b) \geq \texttt{Tol}\rho$ and $k < k_{\max}$, do' by 'While $\sum_{i=0}^n \hat{f}(K_{i,k})/(n+1) \geq \texttt{AverFun}$ and $k < k_{\max}$, do', where $\hat{f}(K)$ is given by (7).

- Within the main loop the steps (i)–(vii) are the same except we replace f by $\hat{f}$ and the stopping condition becomes: 'If $\sum_{i=0}^n \hat{f}(K_{i,k+1})/(n+1) \leq \texttt{AverFun}$, stop'.

Remark 3.2 *The stopping criterion for NM method when solving the minimization problem (7) is such that the average objective function value over the current simplex is required to be less than a prescribed tolerance $\texttt{AverFun} \in (0,1)$.*

4 Particle swarm optimization algorithm for EAP

The particle swarm optimization (PSO) method is an evolutionary algorithm that can solve difficult optimization problems; see e.g. the recent survey [12]. This method was proposed in 1995 by Kennedy and Eberhart, see [12], and recently has gained increasing popularity with its performance in solving various optimization problems. PSO algorithm is a direct search method that is based on population. It has been inspired by the behavior of natural groups, e.g. birds, fish or bees swarm. The idea of a PSO method is to find at each iteration the best available solution by adjusting the moving vector of each particle according to its best solution it has achieved so far (cognition aspect) and the global best (social aspect) positions of particles in the entire swarm.

Let $v_i = vec(K_i) \in \mathbb{R}^n,\ i = 0, 1, \ldots, n$ be the current vertices representing the population (swarm) of size $n+1$ and dimension n. Note that the size of the swarm is not necessarily more than the unknowns vector by one. The change of the particles positions at iteration k are represented by the velocities $V_i(k)$, $i = 0, 1, \ldots, n$.

Populations of particles modify their positions based on the best positions visited earlier by themselves and other particles. All particles have fitness values that are evaluated by the fitness function of values $F_i = f(K_i)$, $i = 0, 1, \ldots, n$ to be optimized. In every iteration k, each particle is updated using specific two best values, namely, the particle best position $v^b(k)$, which is the best solution so far calculated by the current population, and the global best $g^b(k)$, which is the best value obtained so far by any particle in the population (among all particles) until the kth iteration.

The velocity of the i^{th} particle $\{V_i(k)\}_{i=0}^n$ is given as $[V_{i,1}(k), \cdots, V_{i,n}(k)]$ and is updated by

$$V_{i,j}(k+1) = \omega V_{i,j}(k) + c_1\, rand_1(v^b_{i,j}(k) - v_{i,j}(k)) + c_2\, rand_2(g^b_j(k) - v_{i,j}(k)),\ \forall\ i, j, \quad (13)$$

where c_1 and c_2 are given constants, $rand_1$ and $rand_2$ are uniform random numbers within $[0, 1]$. Moreover, ω is called the inertia weight which is updated as

$$\omega(k) = \omega_{\max} - \frac{k}{k_{\max}}(\omega_{max} - \omega_{min}), \quad (14)$$

where $\omega_{min}, \omega_{max} > 0$ are given constants and $k_{\max}$ is the iterations limit.

A new i^{th} particle position $v_{i,j}(k+1)$ is calculated by the previous particle position and its achieved velocity vector, based on (13):

$$v_{i,j}(k+1) = v_{i,j}(k) + V_{i,j}(k+1), \quad \forall\ i,\ j. \quad (15)$$

The initial velocities might be chosen as $V_{i,j}(0) = 0.1 \times v_{i,j}(0), \quad \forall\ i,\ j.$

The PSO algorithm is stated in the following lines.

Algorithm 4.1 (PSO method for solving Problem (6))

1. (Initialization) Let A, B, C be given constant matrices, let $\omega_{\min} \in (0, 1)$, $\omega_{\max} \in (0, 1)$, $c_1 \in (0, 2)$, $c_2 \in (0, 2)$, $\texttt{Tol}\rho \in (0, 1)$ be given constants. Choose a starting swarm position $v_i(0) \in \mathbb{R}^n$ and set $V_i(0) = 0.1 \times v_i(0)$, $i = 0, 1 \ldots, n$. Calculate the fitness of the particles $F_i(0) = f(K_i(0))$, $i = 0, 1, \ldots, n$. Then identify the index of the best particle $v^b(0)$ and $g^b(0)$. Reshape $g^b(0)$ as $K_{g^b}(0)$ and compute the objective function $F_{g^b}(0) = f(K_{g^b}(0))$. If $F_{g^b}(0) < \texttt{Tol}\rho$, stop; otherwise set $k \leftarrow 0$ and go to next step.

2. While $F_{g^b}(k) \geq \texttt{Tol}\rho$ and $k < k_{\max}$, do

   (i) Compute $\omega(k)$ using (14).

   (ii) Update the velocity $V_i(k+1)$ and the position $v_i(k+1)$ of particles using (13) and (15), respectively.

   (iii) Reshape the positions $v_i(k+1)$ into the matrices $K_i(k+1)$ and compute the fitness $F_i^{k+1} = f(K_i(k+1)),\ i = 0, 1, \ldots, n$ and identify the index of the best particle $v^{b1}(k+1)$.

   (iv) Update $v^b(k+1)$ for all the population $i = 0, 1, \ldots, n$:

   If $F_i(k+1) < F_i(k)$ set $v_i^b(k+1) = v_i(k+1)$
   else set $v_i^b(k+1) = v_i^b(k)$.

   (v) Update $g^b(k+1)$ of the population:

   If $F_{b1}(k+1) < F_b(k)$ set $g^b(k+1) = v^{b1}(k+1)$ and $v^b(k+1) = v^{b1}(k+1)$
   else $g^b(k+1) = g^b(k)$.

   (vi) Reshape $g^b(k+1)$ as the matrix $K_{g^b}(k+1)$ and compute the corresponding fitness function $F_{g^b}(k+1) = f(K_{g^b}(k+1))$. If $F_{g^b}(k+1) < \texttt{Tol}\rho$, stop; otherwise set $k \leftarrow k+1$ and go to step (i).

   End (do)

In the following lines we list the main changes made on the PSO algorithm 4.1 to tackle the minimization problem (7) representing the EAP for continuous–time systems.

- The initialization step: It is similar to Algorithm 4.1 except we replace the objective function f by $\hat{f}$ assuming that $\tilde{\lambda}_1, \tilde{\lambda}_2, \ldots, \tilde{\lambda}_m \in \mathbb{C}$ are given desired eigenvalues.

- Replace the convergence criterion 'While $f(K^b(k)) \geq \texttt{Tol}\rho$ and $k < k_{\max}$, do' by 'While $\sum_{i=0}^{n} \hat{f}(K_i(k))/(n+1) \geq$ `AverFun` and $k < k_{\max}$, do'.

- Within the main loop the steps (i)–(vii) are the same as in Algorithm 4.1 except we replace f by $\hat{f}$ as defined in (7) and the stopping condition is changed into: 'If $\sum_{i=0}^{n} \hat{f}(K_i(k+1))/(n+1) \leq$ `AverFun`, stop'.

5 The EAP for decentralized systems

Decentralized control systems appear in different models such as in electric power systems, communication networks, large space structures, robotic systems, economic systems and traffic networks, etc, see e.g. [29, 33] and the references therein. These systems are characterized as large-scale which are therefore composed of lower order subsystems. In the following we consider the continuous–time linear time–invariant decentralized system with $\tilde{n}$ control stations:

$$\dot{x}(t) = Ax(t) + \sum_{i=1}^{\tilde{n}} B_i u_i(t), \quad x(0) = x_0, \quad y_i(t) = C_i x(t), \tag{16}$$

where $x \in \mathbb{R}^m$, $u_i \in \mathbb{R}^{p_i}$, and $y_i \in \mathbb{R}^{r_i}$ are the state, the control input, and the measured output vectors, respectively. Moreover $A \in \mathbb{R}^{m\times m}$, $B_i \in \mathbb{R}^{m\times p_i}, C_i \in \mathbb{R}^{r_i\times m}$, $i = 1, ..., \tilde{n}$ are given constant matrices.

This system is closed using the following control law

$$u_i(t) = K_i\, y_i(t), \qquad i = 1, ..., \tilde{n}, \tag{17}$$

which yields:

$$\dot{x}(t) = (A + \sum_{i=1}^{\tilde{n}} B_i K_i C_i)x(t) = A_{cl}(K_1, K_2, \ldots, K_{\tilde{n}})x(t), \qquad x(0) = x_0, \tag{18}$$

where $A_{cl}(\cdot) = (A + \sum_{i=1}^{\tilde{n}} B_i K_i C_i)$ and $K_i \in \mathbb{R}^{p_i\times r_i}$ are the output feedback gain matrices.

By introducing the following augmented matrices it is straightforward to rewrite the decentralized control system (16)–(17) in the original structure (2) and (3):

$$B = [B_1 \ \ldots \ B_{\tilde{n}}], \quad C = \left[C_1^T \ \ldots \ C_{\tilde{n}}^T\right]^T. \tag{19}$$

The corresponding closed-loop system matrix is $A_{cl}(K) = A + BKC$, where the output feedback gain matrix K takes the block-diagonal structure

$$K = \mathrm{diag}(K_1, \ldots, K_{\tilde{n}}). \tag{20}$$

The EAP for the decentralized control system (16)-(17) is stated as follows. Let $\tilde{\lambda_1}$, $\tilde{\lambda_2}$, $\ldots$, $\tilde{\lambda_m} \in \mathbb{C}$ be given desired eigenvalues, which are assumed to be closed under conjugation. The goal is to find output feedback matrices $K_1, K_2, \ldots, K_{\tilde{n}}$ such that

$$\lambda_i(A_{cl}(K_1, K_2, \ldots, K_{\tilde{n}})) = \tilde{\lambda}_i, \qquad i = 1, \ldots, m. \tag{21}$$

Similar to the problem (7) the EAP for decentralized systems can be stated as the following nonlinear least-squares problem

$$\min_{K_1,\ldots,K_{\tilde{n}}} \hat{f}(K_1, \ldots, K_{\tilde{n}})) = \frac{1}{2}\sum_{i=1}^{m}(\lambda_i(A_{cl}(\cdot)) - \tilde{\lambda}_i)^*(\lambda_i(A_{cl}(\cdot)) - \tilde{\lambda}_i), \tag{22}$$

where the superscript $*$ denotes the complex conjugate. Therefore, the attempt is to find K_1, $\ldots$, $K_{\tilde{n}}$ that assign the eigenvalues of the closed-loop system matrix $A_{cl}(\cdot)$ as close as possible to the desired vector $\tilde{\lambda} \in \mathbb{C}^m$ of desired eigenvalues.

Clearly, we can apply the NM simplex method or PSO method to compute an approximate solution to the minimization problem (22) similar to the treatment used with the problem (7).

6 The EAP for linear periodic systems

Periodic time control systems have been studied recently in several research articles in particular for the stabilization of systems of walking and hopping robots, see among others [2, 4, 7, 13, 37]. Consider the following linear discrete–time periodic control system

$$x(t+1) = A(t)x(t) + B(t)u(t), \quad y(t) = C(t)x(t), \quad t = 0, 1, 2, \ldots, \tag{23}$$

where $x \in \mathbb{R}^m$, $u \in \mathbb{R}^p$, and $y \in \mathbb{R}^r$ are the state, the control input, and the measured output vectors, respectively. Moreover, $A(\cdot) \in \mathbb{R}^{m\times m}$, $B(\cdot) \in \mathbb{R}^{m\times p}$ and $C(\cdot) \in \mathbb{R}^{r\times m}$ are given d–periodic matrices, i.e.,

$$A(t+d) = A(t), \quad B(t+d) = B(t), \quad C(t+d) = C(t), \quad \forall \ t, \quad d \geq 1. \tag{24}$$

Let us also consider the following control law to close the system (23)–(24):

$$u(t) = K(t)y(t), \tag{25}$$

where $K(t+d) = K(t)$ for all t and $d \geq 1$. This yields

$$x(t+1) = (A(t) + B(t)K(t)C(t))\, x(t) := A_{cl}(K(t))x(t),$$

where

$$\begin{aligned} A_{cl}(K(t)) &= (A(t) + B(t)K(t)C(t)), \\ A_{cl}(K(t+d)) &= A_{cl}(K(t)), \quad t = 0, 1, 2, \ldots. \end{aligned}$$

Note that, the control system (23) is time–variant. However, it is straightforward to restate it as a linear time–invariant system. Consequently, the two methods of NM and PSO can be extended to tackle the EAP for periodic systems. The formulation of the given system as a linear time–invariant one is described in the following lemma; see [37, Theorem 1] for a similar result.

Lemma 6.1 *Consider the linear periodic discrete–time system* (23)–(24) *together with the control law* (25)*. This system is equivalent to the following augmented linear time–invariant system:*

$$\begin{aligned} \bar{x}(\bar{t}+1) &= \bar{A}(\bar{t})\bar{x}(\bar{t}) + \bar{B}(\bar{t})\bar{u}(\bar{t}), \quad {\bar{x}_0}^T(\bar{t}) = [0, 0, \ldots, x_0], \\ \bar{y}(\bar{t}) &= \bar{C}(\bar{t})\bar{x}(\bar{t}), \quad \bar{t} = 0, 1, \ldots, \end{aligned} \tag{26}$$

with the output feedback control law:

$$\bar{u}(\bar{t}) = \bar{K}\bar{y}(\bar{t}) \tag{27}$$

where

$$\bar{x}(\bar{t}) = \begin{bmatrix} x(td) \\ x(td+1) \\ \vdots \\ x(td+d-1) \end{bmatrix}, \bar{u}(\bar{t}) = \begin{bmatrix} u(td) \\ u(td+1) \\ \vdots \\ u(td+d-1) \end{bmatrix}, \bar{y}(\bar{t}) = \begin{bmatrix} y(td) \\ y(td+1) \\ \vdots \\ y(td+d-1) \end{bmatrix}$$

$$\bar{A} = \begin{bmatrix} 0 & \cdots & 0 & A(0) \\ 0 & \cdots & 0 & A(1)A(0) \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & A(d-1)\ldots A(1)A(0) \end{bmatrix},$$

$$\bar{B} = \begin{bmatrix} B(0) & 0 & \cdots & 0 \\ A(1)B(0) & B(1) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ A(d-1)\ldots A(1)B(0) & A(d-1)\ldots A(2)B(1) & \cdots & B(d-1) \end{bmatrix},$$

where $\bar{A} \in \mathbb{R}^{dm \times dm}$, $\bar{B} \in \mathbb{R}^{dm \times dp}$, *and*

$$\bar{C} = \begin{bmatrix} 0 & \cdots & 0 & C(0) \\ 0 & \cdots & 0 & C(1)A_{cl}(0) \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & C(d-1)A_{cl}(d-2)\dots A_{cl}(0) \end{bmatrix},$$
$$\bar{K} = \operatorname{diag}(K(0), K(1), \dots, K(d-1)).$$

Proof. See [37, Theorem 1]. □

The drawback of considering this approach to tackle the EAP for periodic systems is the increase of the size of the associated system matrices. However, in the rest of this section we consider an alternative approach.

Lemma 6.2 *Consider the linear periodic discrete–time system* (23)–(24) *together with the control law* (25)*. This system is equivalently rewritten in the following closed form:*

$$\begin{aligned} x(t+d) &= A_{cl}(K(t+d-1))A_{cl}(K(t+d-2))\dots A_{cl}(K(0))x(t) \\ &= \psi(K(0), K(1), \dots, K(t+d-1))x(t), \end{aligned}$$

where

$$\psi(K(0), K(1), \dots, K(t+d-1)) = A_{cl}(K(t+d-1))A_{cl}(K(t+d-2))\dots A_{cl}(K(t)) \quad (28)$$

is the closed-loop monodromy matrix of the system. It follows that

$$x(t+\tilde{t}d) = \psi(A_{cl}(K(t)))x(t+(\tilde{t}-1)d),$$

where ψ *is given by* (28)*. In particular, let us denote* $x(\tilde{t}d) = x(\tilde{t})$*. If* $t = 0$*, then this yields the time-invariant discrete–time system*

$$x(\tilde{t}+1) = \psi(K(0), K(1), \dots, K(d-1))x(\tilde{t}),$$

where

$$\psi(K(0), K(1), \dots, K(d-1)) = A_{cl}(K(d-1))A_{cl}(K(d-2))\dots A_{cl}(K(0)). \quad (29)$$

Proof. See [4, Sec. 2.4]. □

The eigenvalues of the monodromy matrix are called characteristic multipliers. It is known that the control system (23)–(25) is asymptotically stable if and only if all eigenvalues of its monodromy matrix ψ lie in the open unit disk, see e.g. [4, Proposition 3.3].

For the control system (23)–(24) with the control law (25) the periodic eigenvalue assignment problem is to find matrices $K(0), K(1), \dots, K(d-1) \in \mathbb{R}^{p \times r}$ that solve the following minimization problem:

$$\begin{aligned} \min \quad & \tilde{f}(K(0), K(1), \dots, K(d-1)) = \rho(\psi(\cdot)) \\ \text{s.t.} \quad & \tilde{g}(K(0), K(1), \dots, K(d-1)) = \rho(\psi(\cdot)) - 1 + \tau \leq 0, \end{aligned} \quad (30)$$

where $\psi(\cdot)$ is as defined in (29) and $\tau \in (0,1)$ is a given constant. Similar to the treatment in the previous sections and instead of considering the constrained problem (30) we rather consider the following unconstrained minimization problem:

$$\min \ \tilde{f}(K(0), K(1), \ldots, K(d-1)) = \rho(\psi(\cdot)), \tag{31}$$

where $\psi(\cdot)$ is given by (29). The proposed NM and PSO methods are stopped as soon as the achieved $K(0), K(1), \ldots, K(d-1)$ are feasible with respect to the inequality constraint of the problem (30).

In the following we modify both of Algorithms 3.1 and 4.1 to tackle the problem (31) under the following changes. First, the changes made on Algorithm 3.1 are the following:

- The initialization step: It is similar to that given in Algorithm 3.1 except for the given data matrices we assume that $A(l),\ B(l),\ C(l),\ l = 0, 1, \ldots, d-1$ be given constant matrices. Choose $K_0(l) \in \mathbb{R}^{p\times r}$, reshape them as the column vectors $v_0(l) \in \mathbb{R}^{dn},\ l = 0, 1, \ldots, d-1$, and set $v_0 = [(v_0(0))^T, \ldots, (v_0(d-1))^T]^T$ to be one of the initial simplex vertices. Generate the remaining $d \times n$ vertices as explained in (8) and compute $\tilde{f}(K_0(0), \ldots, K_0(d-1))$. Arrange the $d \times n + 1$ vertices so that (9) holds. Identify v_0^b, v_0^w and compute $\tilde{f}(K_0^b(0), \ldots, K_0^b(d-1))$. If $\tilde{f}(K_0^b(0), \ldots, K_0^b(d-1)) < \texttt{Tol}\rho$, stop; otherwise set $k \leftarrow 0$ and go to the next step.

- Convergence criterion: It is changed into 'While $\tilde{f}(K_k^b(0), \ldots, K_k^b(d-1)) \geq \texttt{Tol}\rho$ and $k < k_{max}$, do'.

- Within the main loop keep steps (i) to (vii) as in Algorithm 3.1 except the objective function f is replaced by $\tilde{f}$ as defined in (31).

The following changes are made on Algorithm 4.1 to tackle the EAP (31) for periodic systems:

- Initialization: Let $A(l),\ B(l),\ C(l)$ be given constant matrices. Choose $K_0(l),\ l = 0, 1, \ldots, d-1$, reshape them as the long column vectors $v_i^0 \in \mathbb{R}^{dn}, i = 0, 1, \ldots, nd$, and set $v^0 = [(v_0^0(0))^T, \ldots, (v_{dn}^0(d-1))^T]^T$ to be one of the initial simplex vertices. Compute the fitness function of the particles

$$\tilde{F}_i^0 = \tilde{f}(K_0^0(0), \ldots, K_{dn}^0(d-1)), \quad \forall i$$

  and find the index of the best particle v_b^0 and $g_b^0(l)$. Reshape $g_b^0(l)$ as $K_{g_b^0}(l) \forall l$ and compute the objective function $\tilde{F}_{g_b^0} = \tilde{f}(K_{g_b^0}^0(0), \ldots, K_{g_b^0}^0(d-1))$. If $\tilde{F}_{g_b^0}(0) < \texttt{Tol}\rho$, stop; otherwise set $k \leftarrow 0$ and go to the next step.

- Convergence criterion: It becomes 'While $\tilde{F}_{g_b^k} \geq \texttt{Tol}\rho$ and $k < k_{max}$, do'.

- Within the main loop let steps (i) to (vi) the same as in Algorithm 4.1 except the objective function f is replaced by $\tilde{f}$.

7 Numerical results

In this Section, we have developed a Matlab implementation of the NM and PSO methods to tackle the minimization problems (6) and (7) as well as the problems (22) and (31) that correspond to the EAP for decentralized and periodic systems. Seven test problems are introduced in detail to show the performance of the two methods to assign the eigenvalues of the considered control systems in the desired region in the complex plane. In particular, for discrete–time systems the focus is given to achieve stabilizing output feedback controllers to the considered systems with satisfactory stability margin. For continuous–time systems it is desirable for the eigenvalues of the closed–loop system matrix $A(K)$ to be in the negative side of the complex plane. Therefore, we choose the desired eigenvalues as $\tilde{\lambda}_i = \lambda_i(A) - \mu(A) - s,\ i = 1, \ldots, m$, where is the spectral abscissa of the matrix A as defined in (4) and s is the imposed shift in the negative side of the complex plane.

In addition, the two methods are compared with respect to number of iterations and CPU time on wide range of test problems. All computations are carried out on Laptop with 1.8 Ghz Core Duo CPU and 2048 MB RAM.

Some of the considered test problems are chosen from the benchmark collection COMPleib [22], while other test problems are collected from different sources of the system and control literature. Note that the test problems from the benchmark [22] are for continuous–time systems. The function `c2d` from the control system toolbox of Matlab is considered to convert continuous–time models into its discrete–time counterparts and to provide the constant data matrices A, B, and C. The starting point K_0 is often generated randomly.

In our experiment we used the following values for the stopping conditions of the two algorithms

$$\texttt{AverFun} = 10^{-4},\ \texttt{MaxIter} = 500,\ \texttt{Tol}\rho = 0.98.$$

The considered parameters of the NM method have been set as follows

$$\hat{\sigma} = 0.5,\ \alpha = 1,\ \beta = 2,\ \gamma = 0.5,\ \delta = 0.5,$$

and the following values were assigned to the parameters of the PSO method

$$c_1 = 2 * rand(1),\ c_2 = 2 * rand(1),\ \omega_{\min} = 0.4,\ \omega_{\max} = 0.9.$$

7.1 The EAP for discrete–time systems

The following two examples describe in detail the performance of the NM and PSO methods on the minimization problem (6). Again, in each example the goal is to determine an output feedback control gain for the underlying control system that represents an approximate solution of the optimization problem (6).

Example 7.1 This test problem represents a third–order control system with one control input and two measured outputs ([22, NN1]). The data matrices for the corresponding discrete–time system are the following

$$A = \begin{bmatrix} 1.0000 & 0.1022 & 0.0051 \\ 0 & 1.0657 & 0.1022 \\ 0 & 1.3284 & 1.0657 \end{bmatrix}, B = \begin{bmatrix} 0.0002 \\ 0.0051 \\ 0.1022 \end{bmatrix}, C^T = \begin{bmatrix} 0 & -1 \\ 5 & -1 \\ -1 & 0 \end{bmatrix}.$$

The spectral radius of the system matrix A is 1.4341. The two methods of NM and PSO require 27 and 19 iterations, respectively, to achieve the following stabilizing output feedback controllers K_{fin}^{NM} and K_{fin}^{PSO}. The starting and achieved output feedback gain matrices are the following

$$\begin{aligned} K_0 &= \begin{bmatrix} 2.9080 & 0.8252 \end{bmatrix}, \\ K_{\text{fin}}^{NM} &= \begin{bmatrix} 8.3266 & 86.2021 \end{bmatrix}, K_{\text{fin}}^{PSO} = \begin{bmatrix} 5.9707 & 67.8677 \end{bmatrix}. \end{aligned}$$

The values of objective function at the initial and achieved final K are, respectively,

$$f(K_0) = 1.4711, \quad f(K_{\text{fin}}^{NM}) = 0.8432, \quad f(K_{\text{fin}}^{PSO}) = 0.9259.$$

Example 7.2 This test problem is obtained from the benchmark collection [22, HE1], which represents the longitudinal motion of a helicopter. The corresponding discrete–time system has the following data matrices:

$$A = \begin{bmatrix} 0.9964 & 0.0026 & -0.0004 & -0.0460 \\ 0.0045 & 0.9037 & -0.0188 & -0.3834 \\ 0.0098 & 0.0339 & 0.9383 & 0.1302 \\ 0.0005 & 0.0017 & 0.0968 & 1.0067 \end{bmatrix}, B = \begin{bmatrix} 0.0445 & 0.0167 \\ 0.3407 & -0.7249 \\ -0.5278 & 0.4214 \\ -0.0268 & 0.0215 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}.$$

The spectral radius of the system matrix A is 1.0280. Starting from the same K_0 the two methods of NM and PSO require 15 and 7 iterations, respectively, to achieve stabilizing output feedback controllers with the least possible objective function value. The initial and achieved output feedback gain matrices are the following

$$K_0 = \begin{bmatrix} 1.1811 \\ -0.7776 \end{bmatrix}, K_{\text{fin}}^{NM} = \begin{bmatrix} 0.5559 \\ 2.5693 \end{bmatrix}, K_{\text{fin}}^{PSO} = \begin{bmatrix} 0.6205 \\ 2.8399 \end{bmatrix}.$$

The objective function at the starting and achieved final output feedback gain is of values

$$f(K_0) = 1.9435, \quad f(K_{\text{fin}}^{NM}) = 0.9794, \quad f(K_{\text{fin}}^{PSO}) = 0.9791.$$

In Tables 1 and 2 we report the performance of the NM and PSO methods on the minimization problem (6). Table 1 shows the performance of the two methods on test problems from the benchmark collection [22], while Table 2 shows the performance of the two methods on test problems from different sources from the literature. Moreover, Table 3 shows the average number of iterations and average CPU time for the two methods on the considered set of test problems of the former tables. As can be seen the PSO method outperforms the NM simplex method with respect to CPU time and number of iterations.

Table 1: EAP for discrete–time systems: Performance of the NM and PSO methods on test problems from the benchmark collection [22].

	Problem size			NM		PSO	
Problem	n	p	r	No.it.	CPU	No.it.	CPU
AC1	5	3	3	7	0.34	13	0.09
AC3	5	2	4	5	0.20	3	0.08
AC4	4	1	2	33	0.50	103	0.11
AC5	4	2	2	36	0.53	20	0.08
AC6	7	2	4	7	0.27	4	0.08
AC7	9	1	2	98	1.22	117	0.09
AC8	9	1	5	12	0.37	2	0.08
AC9	10	1	5	116	1.36	49	0.20
AC11	5	2	4	52	0.78	28	0.11
AC12	4	3	4	21	0.45	3	0.05
AC15	4	2	3	6	0.30	7	0.08
AC16	4	2	4	6	0.23	5	0.08
AC17	4	2	2	2	0.25	3	0.06
HE1	4	2	1	15	0.36	7	0.08
HE2	4	2	2	5	0.23	5	0.06
HE3	8	4	6	298	3.54	15	0.11
HE4	8	4	6	115	1.23	6	0.09
HE5	8	4	2	498	4.96	131	0.23
REA1	4	2	3	8	0.27	11	0.06
REA2	4	2	2	2	0.20	5	0.09
REA3	12	1	3	70	0.98	105	0.14
REA4	8	1	1	1	0.14	3	0.06
DIS1	7	4	4	31	0.55	5	0.06
DIS2	3	2	2	4	0.23	4	0.05
DIS3	6	4	4	4	0.27	3	0.08
DIS4	6	4	6	108	1.26	13	0.11
DIS5	4	2	2	3	0.22	2	0.14
TG1	10	2	2	21	0.45	24	0.09
UWV	8	2	2	9	0.34	1	0.06
CSE1	20	2	10	147	1.86	19	0.19
CSE2	60	2	30	500	19.94	110	15.74
EB1	10	1	1	30	0.53	107	0.09
TF1	7	2	4	313	3.39	119	0.19
NN1	3	1	2	27	0.50	19	0.06
NN2	2	1	1	2	0.22	4	0.09
NN4	4	2	3	5	0.23	5	0.06
NN5	7	1	2	90	1.15	114	0.11
NN8	3	2	2	3	0.23	4	0.06
NN9	5	3	2	53	0.78	18	0.09
NN13	6	2	2	18	0.36	69	0.11
NN14	6	2	2	24	0.48	7	0.09
NN15	3	2	2	15	0.37	13	0.06
NN16	8	4	4	44	0.59	5	0.06
NN17	3	2	1	2	0.22	3	0.05
DLR1	10	2	2	22	0.45	24	0.06
WEC1	10	3	4	7	0.28	2	0.06
WEC2	10	3	4	8	0.33	3	0.08
WEC3	10	3	4	7	0.27	4	0.09

Table 2: EAP for discrete–time systems: Performance of the NM and PSO methods on test problems from different sources.

Problem	Problem size			NM		PSO	
	n	p	r	No.it.	CPU	No.it.	CPU
[16]	4	2	4	7	0.33	4	0.08
[3]	5	3	2	5	0.22	5	0.08
[1]	4	2	2	7	0.28	8	0.08
[6]	6	2	3	3	0.23	3	0.06
[35]	3	1	2	6	0.25	4	0.08
[19]	2	2	2	12	0.31	11	0.09
[17]	8	4	3	41	0.66	12	0.06
[16]	5	2	5	89	1.12	16	0.09
[10]	3	2	2	7	0.33	5	0.06
[31]	5	2	4	20	0.39	8	0.06
[21]	4	2	2	14	0.39	4	0.05
[21]	6	3	2	40	0.66	28	0.11
[21]	5	3	2	42	0.56	13	0.09
[30]	4	3	2	14	0.42	9	0.08
[30]	4	2	2	11	0.31	9	0.08
[30]	4	3	3	4	0.25	4	0.05

Table 3: EAP for discrete–time systems: Comparison between the NM and PSO methods on the considered test problems of Tables 1 and 2.

	NM	PSO
Average no. of iterations	50.5	23.26
Average CPU–time (sec.)	0.95	0.33

7.2 The EAP for continuous–time systems

The following two examples demonstrate the performance of the NM and PSO methods for finding an approximate solution to the EAP (7) for continuous–time systems.

Example 7.3 This test problem is from the benchmark collection[22, DIS2], which has three state variables, two input variables and two measured output variables and has the following data matrices:

$$A = \begin{bmatrix} -4 & 2 & 1 \\ 3 & -2 & 5 \\ -7 & 0 & 3 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, C^T = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}.$$

The system matrix A has eigenvalues -6.3509, 1.6755, 1.6755. The desired eigenvalues $\tilde{\lambda}_i$'s are such that the shift is $s = 0.3$ in the negative side of the complex plane, which are namely -8.3264, -0.3000, -0.3000. Starting from the same K_0 the two methods of NM and PSO require 68 and 197 iterations with CPU times 0.97 and 0.31, respectively, to achieve the following output feedback gain matrices

$$K_0 = \begin{bmatrix} 0.4094 & 0.4626 \\ 0.2385 & 0.7591 \end{bmatrix}, K_{\text{fin}}^{NM} = \begin{bmatrix} 4.1724 & 4.3972 \\ 0.9595 & -10.0893 \end{bmatrix},$$
$$K_{\text{fin}}^{PSO} = \begin{bmatrix} 2.4544 & 1.3151 \\ 0.8794 & -8.3759 \end{bmatrix}.$$

The corresponding objective function of the problem (7) at the starting and achieved final K is of values

$$\hat{f}(K_0) = 17.2279,\ \hat{f}(K_{\text{fin}}^{NM}) = 4.9595 \times 10^{-5},\ \hat{f}(K_{\text{fin}}^{PSO}) = 8.0159 \times 10^{-6}.$$

Example 7.4 This test problem is borrowed from [30, SL5] of fourth order control system which has the following data matrices

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & -4 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}, C = B^T.$$

The system matrix A has the eigenvalues $-4, -3, -2, 1$. The desired eigenvalues $\tilde{\lambda}_i$'s are chosen as -5.3, -4.3, -3.3, -0.3. Starting from the same K_0 the two methods of NM and PSO require 124 and 199 iterations with CPU times 1.67 and 0.61, respectively, to reach the least possible output feedback gains. The starting and the achieved final output feedback gain matrices are the following

$$K_0 = \begin{bmatrix} 0.6957 & 0.4736 & 0.2798 \\ 0.6279 & 0.9497 & 0.4470 \\ 0.4504 & 0.0835 & 0.5876 \end{bmatrix}, K_{\text{fin}}^{NM} = \begin{bmatrix} -2.1399 & 2.0307 & 2.4550 \\ 1.0197 & -0.9836 & 0.4033 \\ 0.2153 & 0.5881 & -2.8372 \end{bmatrix},$$
$$K_{\text{fin}}^{PSO} = \begin{bmatrix} -1.3415 & 0.8498 & 0.6204 \\ -0.4112 & -1.4770 & 0.7279 \\ 0.3956 & 0.2753 & -1.0107 \end{bmatrix}.$$

The objective function has the following values at the corresponding K

$$\hat{f}(K_0) = 37.9842,\ \hat{f}(K_{\text{fin}}^{NM}) = 4.4042 \times 10^{-5},\ \hat{f}(K_{\text{fin}}^{PSO}) = 3.5893 \times 10^{-6}.$$

Table 4: The EAP for continuous–time systems ($s = 0.1$): Performance of the NM and PSO methods on test problems from the benchmark collection [22] and other sources.

Problem	Problem size			NM		PSO	
	n	p	r	No.it	CPU	No.it	CPU
AC1	5	3	3	154	1.97	81	0.30
AC3	5	2	4	69	1.03	140	0.44
AC12	4	3	4	252	3.00	141	0.59
AC15	4	2	3	89	1.28	100	0.27
AC16	4	2	4	76	1.01	80	0.27
HE2	4	2	2	63	0.94	500	0.76
HE4	8	4	6	500	6.15	167	1.64
REA1	4	2	3	96	1.33	239	0.51
REA2	4	2	2	73	1.08	212	0.36
DIS2	3	2	2	55	0.89	81	0.17
DIS3	6	4	4	500	5.83	500	2.71
DIS4	6	4	6	497	5.65	178	1.50
NN2	2	1	1	5	0.31	106	0.11
NN4	4	2	3	43	0.69	44	0.14
NN8	3	2	2	35	0.67	49	0.17
NN15	3	2	2	40	0.64	109	0.23
[16]	4	2	4	101	1.56	72	0.23
[16]	5	2	5	249	2.82	90	0.33
[10]	3	2	2	165	2.06	500	0.72
[3]	5	3	2	240	2.82	169	0.48
[1]	4	2	2	114	1.59	500	0.73
[21]	4	2	2	399	4.84	500	0.72
[30]	4	3	2	159	2.28	500	0.98
[30]	4	2	2	103	1.45	500	0.76
[30]	4	3	3	110	1.51	92	0.33
[6]	6	2	3	235	2.70	500	1.19
[19]	2	2	2	41	0.70	65	0.14

Table 5: The EAP for continuous–time systems ($s = 0.1$): Comparison between the NM and PSO methods on the considered test problems of Table 4.

	NM	PSO
Average no. of iteratons	165.29	230.18
Average CPU-time (sec.)	2.10	0.62

Table 4 shows the performance of the NM and PSO methods on the minimization problem (7) with respect to test problems from the benchmark collection [22] as well as test problems from the literature. The desired eigenvalues $\tilde{\lambda}_i,\ i = 1, \ldots, m$ of the minimization problem (7) are chosen such that the eigenvalues of $A(K_{\text{fin}})$ are sufficiently shifted in the negative side of the complex plane as explained at the beginning of the section, where $s = 0.1$.

Table 5 shows the average number of iterations and average CPU time for the NM and PSO methods. As can be seen the PSO method outperforms the NM method with respect to CPU time. Although the NM method requires less average number of iterations, but the increase of the CPU time is due to the increase of the number of function evaluations per iteration in that method.

We also ran the two methods on the problem (7) after adding extra shift on the desired eigenvalues in the negative side of the complex plane, namely $s = 0.3$. Table 6 shows the

Table 6: The EAP for continuous–time systems ($s = 0.3$): Performance of NM and PSO on test problems from the benchmark collection [22] and other sources.

	Problem size			NM		PSO	
Problem	n	p	r	No.it	CPU	No.it	CPU
AC1	5	3	3	175	2.71	196	0.67
AC3	5	2	4	74	1.36	57	0.22
AC12	4	3	4	331	4.38	239	1.20
AC15	4	2	3	111	1.58	85	0.25
AC16	4	2	4	105	1.45	89	0.30
AC17	4	2	2	57	1.00	106	0.20
HE2	4	2	2	112	2.00	500	0.80
REA1	4	2	3	114	1.61	245	0.53
REA2	4	2	2	149	2.03	500	0.69
DIS2	3	2	2	68	0.97	197	0.31
NN4	4	2	3	82	1.26	115	0.30
NN8	3	2	2	53	0.87	29	0.09
NN15	3	2	2	107	1.40	62	0.14
[16]	4	2	4	120	1.76	75	0.25
[16]	5	2	5	416	6.36	356	1.23
[10]	3	2	2	227	2.67	500	0.72
[3]	5	3	2	275	4.01	500	1.20
[1]	4	2	2	187	2.82	500	0.73
[30]	4	3	2	193	2.39	500	0.97
[30]	4	2	2	119	1.78	500	0.78
[30]	4	3	3	124	1.67	199	0.61
[6]	6	2	3	264	4.12	491	1.17
[19]	2	2	2	43	0.75	255	0.39

Table 7: The EAP for continuous–time systems ($s = 0.3$): Comparison between NM and PSO on the test problems of Table 6.

	NM	PSO
Average number of iterations	152.43	273.73
Average CPU-time (sec.)	2.21	0.59

performance of the two methods corresponding to this case. Moreover, Table 7 shows the comparison between the two methods for these results.

7.3 The EAP for decentralized control systems

The following two examples quite show the applicability of the methods NM and PSO to tackle the EAP for decentralized control systems.

Example 7.5 This test problem appeared in [29] of a decentralized control system with two state variables and two control stations. The given data matrices are the following:

$$A = \begin{bmatrix} 8 & 1 \\ -8 & -2 \end{bmatrix}, B_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C_1^T = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, C_2^T = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

The system matrix A has the eigenvalues 7.1231, -1.1231. The desired eigenvalues $\tilde{\lambda}_i$'s are chosen as -0.3000, -8.5462. Starting from the same K_0 the two methods of NM and PSO require 28 and 48 iterations with CPU times 0.70 and 0.12, respectively, to achieve the

least possible values of the objective function of the problem (22). The starting and achieved output feedback gain matrices are

$$K_0 = \begin{bmatrix} 0.3727 & 0.0000 \\ 0.0000 & 0.9120 \end{bmatrix},$$
$$K_{\text{fin}}^{NM} = \begin{bmatrix} -7.4255 & 0.0000 \\ 0.0000 & -7.4247 \end{bmatrix}, K_{\text{fin}}^{PSO} = \begin{bmatrix} -7.4243 & 0.0000 \\ 0.0000 & -7.4198 \end{bmatrix},$$

where $\hat{f}(K_0) = 130.3228$ and the corresponding objective function values are $\hat{f}(K_{\text{fin}}^{NM}) = 8.4257\times10^{-6}$ and $\hat{f}(K_{\text{fin}}^{PSO}) = 1.7122\times10^{-5}$.

Example 7.6 This test problem appeared in [33] of a fifth order control system with two control stations and has the following data matrices:

$$A = \begin{bmatrix} -0.4 & 0.2 & 0.6 & 0.1 & -0.2 \\ 0 & -0.5 & 0 & 0 & 0.4 \\ 0 & 0 & -2 & 0 & 0.2 \\ 0.2 & 0.1 & 0.5 & -1.25 & 0 \\ 0.25 & 0 & -0.2 & 0.5 & -1 \end{bmatrix}, B_1 = \begin{bmatrix} 1 & -1 \\ 2 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, B_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ -2 \\ 1 \end{bmatrix},$$
$$C_1 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \end{bmatrix}, C_2 = \begin{bmatrix} 0 & 0 & 1 & -1 & 1 \end{bmatrix}.$$

The matrix A has the eigenvalues -0.2592, -1.4535, -1.8564, $-0.7904 \pm 0.1470\,\text{i}$, and the desired eigenvalues $\tilde{\lambda}_i$'s are chosen as -0.1000, -0.6312, -0.6312, -1.6972. Starting from the same K_0 the two methods of NM and PSO require 95 and 83 iterations with CPU times 1.40 and 0.27, respectively, to achieve the least possible value of the objective function of the problem (22). The starting and achieved output feedback gain matrices are, respectively,

$$K_0 = \begin{bmatrix} 0.0196 & 0.4243 & 0.0000 \\ 0.3309 & 0.2703 & 0.0000 \\ 0.0000 & 0.0000 & 0.1971 \end{bmatrix}, K_{\text{fin}}^{NM} = \begin{bmatrix} 0.0300 & 0.2233 & 0.0000 \\ 0.3778 & 0.4320 & 0.0000 \\ 0.0000 & 0.0000 & 0.4495 \end{bmatrix},$$
$$K_{\text{fin}}^{PSO} = \begin{bmatrix} 0.0097 & 0.2073 & 0.0000 \\ 0.3368 & 0.4080 & 0.0000 \\ 0.0000 & 0.0000 & 0.4476 \end{bmatrix},$$

where $\hat{f}(K_0) = 1.3410$ and the corresponding objective function values are

$$\hat{f}(K_{\text{fin}}^{NM}) = 4.9970\times10^{-5},\ \hat{f}(K_{\text{fin}}^{PSO}) = 9.5371\times10^{-5}.$$

7.4 The EAP for discrete–time periodic systems

We consider the following example to demonstrate the applicability of the NM and PSO methods for tackling the EAP for discrete–time periodic systems.

Example 7.7 This test problem appeared in [4] and has the following data matrices, where the period $d = 2$:

$$A(0) = \begin{bmatrix} 1.1052 & 0 \\ 0 & 1.1052 \end{bmatrix}, A(1) = \begin{bmatrix} 1.1052 & 0 \\ 0.1052 & 1 \end{bmatrix}, B(0) = \begin{bmatrix} 0 & 0 \\ 0 & 0.1052 \end{bmatrix},$$
$$B(1) = \begin{bmatrix} 0.1052 & 0 \\ 0.1052 & 0.1000 \end{bmatrix}, C(0)^T = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, C(1)^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

The spectral radius of the monodromy matrix $\psi(K(0), K(1))$ is 1.2214. Starting from the same $K_0(0)$ and $K_0(1)$ (given below) the NM and PSO methods require 10 and 6 iterations with 0.41 and 0.09 CPU-time, respectively, to achieve the least objective function value. The starting and achieved output feedback controllers are the following:

$$K_0(0) = \begin{bmatrix} 1.5772 \\ 2.2770 \end{bmatrix}, K_0(1) = \begin{bmatrix} -0.7162 \\ -0.7277 \end{bmatrix},$$
$$K_{\text{fin}}^{NM}(0) = \begin{bmatrix} 0.3482 \\ 3.4455 \end{bmatrix}, K_{\text{fin}}^{NM}(1) = \begin{bmatrix} -3.8925 \\ -2.0183 \end{bmatrix},$$
$$K_{\text{fin}}^{PSO}(0) = \begin{bmatrix} 1.1626 \\ 2.7766 \end{bmatrix}, K_{\text{fin}}^{PSO}(1) = \begin{bmatrix} -1.2036 \\ -0.7427 \end{bmatrix}.$$

The objective function value at starting point is $\tilde{f}(K_0(0), K_0(1)) = 1.2290$ and at the achieved final points is of values

$$\tilde{f}(K_{\text{fin}}^{NM}(0), K_{\text{fin}}^{NM}(1)) = 0.9705, \ \tilde{f}(K_{\text{fin}}^{PSO}(0), K_{\text{fin}}^{PSO}(1)) = 0.9697.$$

As can be seen both values lie in the open unit disk as required.

Conclusion

In this work the eigenvalue assignment problem is considered for discrete and continuous–time control systems. Two matrix optimization problems are considered corresponding the two cases. The first problem is an inequality constraint minimization problem where the objective function is included as a cut. However, this problem is treated as an unconstrained problem, while monitoring the fulfillment of the constraint during the minimization of the objective function. The second problem is a nonlinear least–squares problem, which is derived from the EAP for continuous–time systems.

Two derivative-free optimization techniques of the Nelder-Mead simplex method and particle swarm optimization methods are proposed to tackle the optimization problems. Both methods are extended to tackle the eigenvalue assignment problem for decentralized and periodic systems. The performance of the methods is demonstrated over wide range of test problems from the benchmark collection [22] and different test examples from the system and control literature. The two methods are successful in tackling the considered problems. In particular, the particle swarm optimization method relatively outperforms the Nelder-Mead simplex method.

Our future work is focused on the following two issues for improving the current implementation:

- study the performance of other derivative–free optimization methods for tackling the EAP.

- apply some variants of hybrid and three–term conjugate gradient algorithms for the output feedback EAP.

References

[1] A.T. Alexandridis and P.N. Paraskevopoulos. A new approach to eigenstructure assignment by output feedback, *IEEE Transactions on Automatic Control*, **41** (1996), 1046–1050.

[2] F.A. Aliev and V.B. Larin. Optimization problems for periodic systems, *International Applied Mechanics*, **45** (2009), 1162–4511.

[3] O. Bachelier, J. Bosche and D. Mehdi. On pole placement via eigenstructure assignment approach, *IEEE Transactions on Automatic Control*, **51** (2006), 1554–1558.

[4] S. Bittanti and P. Colaneri. Periodic systems filtering and control, *Springer*, 2009.

[5] C.I. Byrnes. *Pole placement by output feedback in three decades of mathematical system theory*, Lecture Notes in Control and Information Science 135, H. Nijmeijer and J.M. Schumacher, Eds., Springer, New York, 1989, 31–78.

[6] L. Carotenuto, C. Franze and P. Muraca. New computational procedure to the pole placement problem by static output feedback, *IEE Proceedings–Control Theory and Applications*, **148**:6 (2001), 466–471.

[7] P. Colaneri. Periodic control systems: Theoretical aspects, *Applied and Computational Mathematics*, **3**:2 (2004), 84-94.

[8] F.E. Curtis, T. Mitchell and M.L. Overton. A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles, *Optimization Methods and Software*, **32**:1 (2017), 148–181.

[9] E.J. Davison and R. Chatterjee. A note on pole assignment in linear systems with incomplete state feedback, *IEEE Transaction on Automatic Control*, **16** (1971), 98–99.

[10] E. Davison and S. Wang. On pole assignment in linear multivariable systems using output feedback, *IEEE Transaction on Automatic Control*, AC-206 (1975), 516–518.

[11] E.J. Davison. On pole assignment in linear systems with incomplete state feedback, *IEEE Transaction on Automatic Control*, **15** (1970), 348–351.

[12] K.-L. Du and M.N.S. Swamy. *Search and optimization by metaheuristics*, Birkhäuser, 2016.

[13] C. Farges, D. Peaucelle and D. Arzelier. Resilient static output feedback stabilization of linear periodic systems. *In: 5th IFAC Symposium on Robust Control Design. Toulouse*, (2006), 5–7.

[14] M. Fu. Pole placement via static output feedback is NP-hard, *IEEE Transactions on Automatic Control*, **49** (2004), 855–857.

[15] S.M. Karbassi. An algorithm for minimizing the norm of state feedback controllers in eigenvalue assignment, *Computers and Mathematics with Applications*, **41** (2001), 1317–1326.

[16] J. Kautsky, N.K. Nichols and P. Van Dooren. Robust pole assignment in linear feedback, *International Journal of Control*, **41** (1985), 1129–1155.

[17] H. Kimura. A further result on the problem of pole assignment by output feedback, *IEEE Transactions on Automatic Control*, **22** (1977), 458–463.

[18] K.H. Kiritsis. A necessary condition for pole assignment by constant output feedback, *Systems & Control Letters* **45** (2002), 317–320.

[19] H. Kwakernaak and R. Sivan. *Linear optimal control systems*, Wiley Interscience, 1972.

[20] K.H. Lee, J.H. Lee and W.H. Kwon. Sufficient LMI conditions for H_∞ output feedback stabilization of linear discrete–time systems, *IEEE Transactions on Automatic Control*, **51**:4 (2006), 675–680.

[21] T.H. Lee, Q.G. Wan and E.K. Koh. An iterative algorithm for pole placement by output feedback, *IEEE Transaction on Automatic Control*, **39**:3 (1994), 565–568.

[22] F. Leibfritz. COMPleib: constraint matrix optimization problem library. Version 1.1, Univ. Trier, Germany, 2005 `http:// www.compleib.de/`.

[23] R.M. Lewis. V. Torczon and M.W. Trosset. Direct search methods, then and now, *Journal of Computational and Applied Mathematics*, **124** (2000), 191–207.

[24] R.E. Mahony and U. Helmke. System assignment and pole placement for symmetric realizations, *Journal of Mathematical Systems, Estimation and Control*, **5**:2 (1995), 1–32.

[25] B.C. Moore. On the flexibility offered by state feedback in multivariable systems beyond closed loop eigenvalue assignment, *IEEE Transaction on Automatic Control*, **21** (1976), 689–692.

[26] E.M.E. Mostafa, M.A. Tawhid and E.R. Elwan. Inexact Newton and quasi–Newton methods for the output feedback pole assignment problem, *Computational and Applied Mathematics*, **33** (2014), 517-542.

[27] E.M.E. Mostafa, A.W. Aboutahoun and F.F.S. Omar. On the solution of the eigenvalue assignment problem for discrete–time systems, *Journal of Applied Mathematics*, **2017**, Article ID 7256769, 12 pages.

[28] J.A. Nelder and R. Mead. A simplex method for function minimization, *The Computer Journal* **7**:4 (1965), 308–313.

[29] S. Sojoudi, J.Lavaei, A. G. Aghdam. *Structurally constrained controllers: Analysis and synthesis*, Springer, New York, 2011.

[30] B. Sridhar and D.P. Lindorf. Pole placement with constraint gain output feedback, *International Journal of Control*, **18** (1973), 993–1003.

[31] V.L. Syrmos and F.L. Lewis. Output feedback eigenstructure assignment using two Sylvester equations, *IEEE Transactions on Automatic Control*, **38** (1993), 495–499.

[32] V.L. Syrmos, C.T. Abdallah, P. Dorato and K. Grigoriadis. Static output feedback–a survey, *Automatica*, **33** (1997), 125–137.

[33] M. Tarokh. A unified approach to exact, approximate, optimized and decentralized output feedback pole assignment, *International Journal of Control, Automation, and Systems*, **6**:6 (2008) 939–947.

[34] H.A. Tehrani and N. Ramroodi. Eigenvalue assignment of discrete–time linear systems with state and input time–delays, *AUT Journal of Modeling and Simulation*, **45**:2 (2013), 23–30.

[35] H. Wang and S. Dale. A fault detection method for unknown systems with unknown inputs and its application to hydraulic turbine monitoring, *International Journal of Control*, **57** (1993), 247–260.

[36] W. Wonham. On pole assignment in multi-input controllable linear systems, *IEEE Transaction on Automatic Control*, **12**:6 (1967), 660–650.

[37] Y. Yang. An efficient LQR design for discrete–time linear periodic system based on a novel lifting method, *Automatica*, **87** (2018), 383–388.

[38] K. Yang and R. Orsi. Static output feedback pole assignment via a trust region approach, *IEEE Transactions on Automatic Control*, **52** (2007), 2146–2150.