Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 12, No. 1, pp. 55–85, January 2016

NONLINEAR CONJUGATE GRADIENT METHODS FOR THE OUTPUT FEEDBACK POLE ASSIGNMENT PROBLEM

El-Sayed M.E. Mostafa, Mohamed A. Tawhid* and Eman R. Elwan

Abstract: The nonlinear conjugate gradient methods are very effective iterative methods for solving large-scale optimization problems. Indeed, these methods are widely used to obtain the numerical solution of the optimal control problems arising in various applications such as engineering and finance, especially for solving large-scale problems. In this article, we proposed three nonlinear conjugate gradient methods for solving the output and state feedback pole assignment problems. Also, we established the global convergence of the proposed algorithms under standard assumptions. Moreover, we extended these methods in order to tackle the output feedback pole assignment problem for decentralized control systems. Furthermore, we showed the performance of the proposed methods by giving numerical examples.

Key words: *the pole assignment problem, output feedback control, nonlinear conjugate gradient methods.*

Mathematics Subject Classification: *49N35, 49N10, 93D52, 93D22, 65K05.*

1 Introduction

In this paper, we consider the following unconstrained matrix optimization problem:

$$\min_{K\in\mathbb{R}^{p\times r}} f(K), \tag{1.1}$$

where $f:\mathbb{R}^{p\times r}\to\mathbb{R}$ and K is a matrix variable of appropriate dimension. As an example of the matrix optimization problem in (1.1), we consider the output and state feedback pole assignment problems for continuous-time systems. Both problems are common in systems and control literature; see e.g. the survey [5].

Our aim of this work is to analyze and study different types of nonlinear CG method for finding the local solution of the considered matrix optimization problem in (1.1). In this article, we apply and study three nonlinear conjugate gradient methods: the descent nonlinear CG methods (NCG), Yu and Wei's gradient method [54] (VLS+CG) and the modified Polak-Ribiére-Polyak CG method (MPRP-CG) [28] for solving the considered optimization problem. These methods are modified to tackle the problem structure. We focus on two formulations of the pole assignment problem (PAP) that are generally stated as the matrix optimization problem (1.1). We consider the output feedback pole assignment problem and the pole assignment with robustness measured in terms of the spectral condition number of the closed-loop eigenvector matrix.

*The research of the 2nd author is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

Nonlinear conjugate gradient methods are widely studied and comprise a class of unconstrained optimization algorithms which are characterized by low memory requirements and strong global convergence properties. These methods are descent direction methods which means that after choosing a starting point $K_0 \in \mathbb{R}^{p\times r}$ these methods generate a sequence of iterates $\{K_k\} \subset \mathbb{R}^{p\times r}$ according to the following relation:

$$K_{k+1} = K_k + \alpha_k \Delta K_k, \tag{1.2}$$

where the step-size $\alpha_k > 0$ satisfies the line search rule and ΔK_k is a descent direction for f at K_k.

Most of the CG methods update the directions ΔK_k by the following relation:

$$\Delta K_k = -g_k + \beta_k \Delta K_{k-1}, \qquad \Delta K_1 = -g_1, \tag{1.3}$$

where $g_k = g(K_k)$ is the gradient of f at K_k and β_k is a parameter that differs from one CG method to the other.

This article is organized as follows. In the next section, we introduce some basic concepts and definitions which are needed in the subsequent analysis. In Section 3, we introduce the output feedback pole assignment problem. In Section 4, we compute the required derivative of the objective function. In Section 5, we describe a conjugate gradient method in order to find the local solution of the output feedback pole assignment problem. In section 6, we establish global convergence results for the conjugate gradient method under appropriate conditions. In Section 7, we extend the proposed CG methods to tackle the output feedback PAP for decentralized control systems. In Section 8, we propose the state feedback PAP with robustness measure in terms of the spectral condition number of the closed-loop eigenvector matrix. In section 9, we test the proposed methods on various test problems from the literature and present the numerical results. We end the paper with some concluding remarks and future research.

Notations: For vectors the symbol $\|\cdot\|$ is the 2–norm, while for matrices $\|\cdot\|$ denotes the Frobenius norm defined by $\|M\| = \sqrt{\langle M, M\rangle}$, where $\langle\cdot,\cdot\rangle$ is the inner product given by $\langle M_1, M_2\rangle = Tr\,(M_2^* M_1)$, and $Tr\,(\cdot)$ is the trace operator. The symbol I_n denotes the identity matrix of order n. The eigenvalues of a matrix $M \in \mathbb{R}^{n\times n}$ are denoted by $\lambda_i(M)$, $i = 1,\ldots,n$ and $\Lambda(M) := \mathrm{diag}(\lambda_1,\ldots,\lambda_n)$ is a diagonal matrix with eigenvalues on its main diagonal. We use $\kappa \in \mathbb{R}^m$ to denote the vector obtained by the *vec*–operator that stretches the matrix variable $K \in \mathbb{R}^{p\times r}$ into a long column vector $\kappa \in \mathbb{R}^m$, where $m = p \cdot r$. Sometimes and for the sake of simplicity we skip the arguments of the considered functions, e.g. we use f_k to denote $f(\kappa_k)$ which also means $f(K_k)$.

2 Basic Concepts and Definitions

A function $\phi : \mathbb{R}^m \to \mathbb{R}^m$ is is said to be locally Lipschitz continuous at $\kappa \in \mathbb{R}^m$ with a constant $L > 0$ if and only if there exists a number $\tau > 0$ such that

$$\|\phi(\kappa_1) - \phi(\kappa_2)\| \le L\|\kappa_1 - \kappa_2\|, \quad \forall \quad \kappa_1, \kappa_2 \in \mathcal{B}_\tau(\kappa),$$

where $\mathcal{B}_\tau(\kappa)$ is an open ball with center at κ and radius τ.

We regard the $p\times r$ matrix space as a space of $\mathbb{R}^{p\cdot r}$, where the matrix variable $K \in \mathbb{R}^{p\times r}$ is stretched into a long column vector $\kappa \in \mathbb{R}^m$ and $m = p \cdot r$. Let $\psi : \mathbb{R}^m \to \mathbb{R}^n$ be a locally Lipschitz continuous function. Radmacher's theorem; see e.g. [7], implies that the

mapping ψ is differentiable almost everywhere. Let $\mathcal{D}_\psi$ be the set of all κ at which ψ is differentiable and let $\nabla\psi^T$ be its Jacobian whenever it exists. Moreover, let

$$\partial_B \psi(\kappa) = \left\{ M \in \mathbb{R}^{n\times m} \ : \ M = \lim_{\kappa_k \to \kappa} \nabla\psi(\kappa_k)^T, \ \kappa_k \in D_\psi \right\}.$$

Clarke's subgradient of ψ at κ, denoted by $\partial\psi(\kappa)$, is the convex hull of all such $\partial_B \psi(\kappa)$; see [7].

Theorem 2.1. *A Lipschitz continuous function on a bounded set Ω is bounded.*

Proof. Suppose that a function f is Lipschitz continuous on Ω. Choose a point $y \in \Omega$, then for any other point $x \in \Omega$ we have:

$$||f(x) - f(y)|| \leq L_f ||x - y||,$$

where $L_f > 0$ is Lipschitz constant. Also, we have

$$||f(x) - f(y)|| \geq | \ ||f(x)|| - ||f(y)|| \ |,$$

suppose that $||f(x)|| \geq ||f(y)||$, then we can write:

$$||f(x)|| \leq ||f(y)|| + L_f ||x - y||.$$

Even though we do not know $||f(y)||$, we do know that it is finite. Let

$$||f(y)|| + L_f ||x - y|| \leq \tau.$$

This shows that $||f(x)||$ is bounded by the constant τ. □

Let $\mathcal{D} \subseteq \mathbb{R}^m$ be the set of all κ at which f is differentiable, which is an open set. Therefore, we replace $\mathcal{D}$ by the following level set:

$$\Omega(\kappa_0) := \{\kappa \in \mathcal{D} \ : \ f(\kappa) \leq f(\kappa_0)\}, \tag{2.1}$$

where $\kappa_0 \in \mathbb{R}^m$ is given and this set is assumed to be bounded.

3 The Output Feedback Pole Assignment Problem

The pole assignment problem of linear control systems has been an important issue for decades. The pole assignment via output feedback is more complex than the pole assignment via state feedback or dynamic feedback. Various approaches from linear system theory, complex function theory and algebraic geometry are used to explore this problem and the results are not complete until now; see the survey paper [5] for more details. The pole assignment problem is also known in the system and control literature as the 'pole placement' and 'pole shifting'; see, e.g., the survey [48] and subsequent references among them [1, 3, 5, 6, 8, 19, 20, 21, 32, 37, 43, 53].

The PAP can be stated as follows. Let $A_c : \mathbb{R}^{p\times r} \to \mathbb{R}^{n\times n}$ be a continuously differentiable function defined by $A_c(K) = A + BKC$, where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times p}, C \in \mathbb{R}^{r\times n}$ are given constant matrices and let $\hat{\lambda}_1, \ldots, \hat{\lambda}_n \in \mathbb{C}$ be given desired eigenvalues, which are closed under conjugation. The PAP is to find a matrix $K \in \mathbb{R}^{p\times r}$ such that

$$\lambda_i(A_c(K)) = \hat{\lambda}_i, \qquad i = 1, \ldots, n. \tag{3.1}$$

In particular, the output feedback PAP, see e.g. [1, 5, 6, 8, 20, 21, 32, 37, 53] is stated as follows. Consider the linear time-invariant control system with the following state space realization:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \quad x(0) = x_0, \\ y(t) &= Cx(t), \end{aligned} \tag{3.2}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}^p$, and $y(t) \in \mathbb{R}^r$ are the state, the control input, and the measured output vectors, respectively. This linear control system is often closed by the control law $u(t) = Ky(t)$ as

$$\dot{x}(t) = (A + BKC)x(t) := A_c(K)x(t), \quad x(0) = x_0,$$

where $K \in \mathbb{R}^{p\times r}$ is the output feedback gain matrix, which represents the unknown. For the above linear control system it is desired to find K that assigns by using the control law $u(t) = Ky(t)$ the poles of the closed–loop system matrix $A_c(K)$ in a pre-specified region in the complex plane.

Consider the mapping $\check{f} : \mathbb{R}^{p\times r} \to \mathbb{C}^n$ defined as

$$\check{f}(K) = \begin{bmatrix} \lambda_1(A_c(K)) - \hat{\lambda}_1 \\ \vdots \\ \lambda_n(A_c(K)) - \hat{\lambda}_n. \end{bmatrix}. \tag{3.3}$$

The PAP is to find $K \in \mathbb{R}^{p\times r}$ by solving the following system of nonlinear equations

$$\check{f}(K) = 0. \tag{3.4}$$

The PAP can be regarded as an inverse eigenvalue problem, see e.g. [52]. The authors in [15] and [30] introduced another approach for solving PAP in which the pole assignment problem is stated as a least squares optimization problem. This approach has several advantages; for example, it converts the exact pole assignment problem into an optimization problem which can be solved numerically. Also, even though exact pole assignment by output feedback may not be feasible, it can always provide a reasonable alternative which is optimal in the sense of the least squares of the difference between system poles and desired poles. Furthermore, the corresponding eigenstructure is obtained automatically if the pole assignment problem is exactly solved.

Recently, Yang and Orsi [53] proposed Newton's method with trust region for determining the local solution of a least–squares problem and Mostafa et al. [32] extended their work to quasi-Newton and inexact Newton methods. This formulation has the advantage of not requiring any restrictions on the dimensions of the problem matrices A, B and C, where a local solution can be obtained for non-solvable problems.

The relaxed output feedback PAP takes the form:

$$\min_{K\in\mathbb{R}^{p\times r}} \; f(K) := \frac{1}{2}||\check{f}(K)||^2 = \frac{1}{2}\sum_{i=1}^{n}(\lambda_i(A_c(K)) - \hat{\lambda}_i)^*(\lambda_i(A_c(K)) - \hat{\lambda}_i), \tag{3.5}$$

where $\check{f}(K)$ is defined in (3.3) and the superscript $*$ denotes the complex conjugate transpose; see [32, 53]. Consequently, the PAP is relaxed to find K that assigns the poles of the closed-loop system matrix $A_c(K)$ to be as close as possible to the vector $\hat{\lambda} \in \mathbb{C}^n$ of desired eigenvalues.

Recently, Fu [10] has shown that the PAP is NP–hard. Moreover, for the solvability of the PAP, Kiritsis [21] gave a sufficient condition for an arbitrary pole assignment, $p \cdot r \geq n$, i.e., the number of unknowns is greater than or equal to the number of the assigned poles. Despite the enormous efforts have been given for solving this problem, however, there is a lack of robust numerical algorithms.

In our work, systems with a symmetric state space realization, i.e., systems with state space matrices satisfying $A = A^T$, $C = B^T$, occur in various contexts, e.g., RC-networks [30]. For symmetric systems, the symmetric PAP is defined as follows: Find a matrix K such that

$$\lambda_i(A_c(K)) = \lambda_i(A + BKB^T) = \hat{\lambda}_i, \qquad i = 1, \ldots, n. \tag{3.6}$$

It is well known that the eigenvalues of a real symmetric matrix are not everywhere differentiable. A classical result of Ky Fan states that each eigenvalue of a symmetric matrix is the difference of two convex functions, which implies that the eigenvalues are semismooth functions, see e.g. [47]. This result motivates us to employ the theory of semismoothness to establish our results.

The PAP is considered as a semi-smooth and nonconvex problem. For related work on nonconvex and nonsmooth/semismooth problems we refer the interested reader to, e.g. [44, 45, 47] and the references therein.

4 Derivative of the Objective Function

Rademacher's theorem says that a loaclly Lipschitz continuous function is differentiable almost everywhere. The eigenvalues of $A_c(K)$ in (3.5) is not in general differentiable at a point K where $A_c(K)$ has multiple eigenvalues. Therefore, let us impose the following assumption on eigenvalues of A_c.

Assumption 4.1. *Assume that $A_c(K)$ has no multiple eigenvalues for all $K \in \mathbb{R}^{p\times r}$.*

According to [18, Theorem 5.4, pp 111] and the description given below, the following lemma gives the differentiability of the eigenvalues function $\Lambda(A_c(K))$.

Lemma 4.2. *Let the matrix valued function $A_c : \mathbb{R}^{p\times r} \to \mathbb{R}^{n\times n}$ be differentiable on an open subset $D \subseteq \mathbb{R}^{p\times r}$ and let $A_c(K)$ have distinct eigenvalues for all K. Then $\Lambda(A_c(K))$ is differentiable on D.*

Let $A_c(K)$ be diagonalizable and let $Q(A_c) \in \mathbb{C}^{n\times n}$ be a matrix whose columns are the eigenvectors of $A_c(K)$, i.e., Q and $\Lambda(A_c)$ satisfy

$$A_c(K)Q = Q\Lambda. \tag{4.1}$$

The columns of Q clearly satisfy:

$$q_i^T q_i = 1, \quad \frac{\partial q_i^T}{\partial \kappa_j} q_i = 0, \quad i = 1, \ldots, n,\ j = 1, \ldots, m,$$

which together with (4.1) imply that

$$\frac{\partial \lambda_i(A_c(K))}{\partial \kappa_j} = q_i^T \frac{\partial A_c(K)}{\partial \kappa_j} q_i, \quad i = 1, \ldots, n, \quad j = 1, \ldots, m. \tag{4.2}$$

The following lemma provides the first–order derivatives of the objective function $f(K)$ required by the CG methods.

Lemma 4.3. *Let $A_c(K)$ satisfy Assumption 4.1 and let $\Lambda(A_c(K))$ and $Q \in \mathbb{C}^{n\times n}$ be as defined above. Then gradient of the objective function f has components of the form:*

$$\frac{\partial f(K)}{\partial K_{kl}} = \mathrm{Re}\Big\{ \sum_{i=1}^{n} \Big(\frac{\partial \lambda_i(A_c(K))}{\partial K_{kl}}\Big)^{*} (\lambda_i(A_c) - \hat{\lambda}_i)\Big\},\ k = 1,\ldots,p,\ l = 1,\ldots,r, \tag{4.3}$$

where

$$\frac{\partial \lambda_i(A_c(K))}{\partial K_{kl}} = q_i^T B E_{kl} C\, q_i, \quad i = 1,\ldots,n, \tag{4.4}$$

where q_i is the ith column of Q and $E_{kl} \in \mathbb{R}^{p\times r}$ is a matrix with zero entries except a value of one at the (k,l)–position.

Proof (see [32, Lemma 3.2]). The derivatives of $A_c(K)$ with respect to the entries of K are determined as follows:

$$\frac{\partial A_c(K)}{\partial K_{kl}} = B\frac{\partial K}{\partial K_{kl}}C = BE_{kl}C,\ k = 1,\ldots,p,\ l = 1,\ldots,r.$$

From (4.2) the derivatives of the ith component $\lambda_i(A_c(K))$ with respect the (k,l)–entry of K are:

$$\frac{\partial \lambda_i(A_c(K))}{\partial K_{kl}} = q_i^T \frac{\partial A_c(K)}{\partial K_{kl}} q_i = q_i^T B E_{kl} C\, q_i,\ k = 1,\ldots,p,\ l = 1,\ldots,r. \tag{4.5}$$

The derivative of $f(K)$ with respect to the (k,l)–component of K is

$$\frac{\partial f(K)}{\partial K_{kl}} = \mathrm{Re}\Big\{ \sum_{i=1}^{n} \Big(\frac{\partial \lambda_i(A_c(K))}{\partial K_{kl}}\Big)^{*} (\lambda_i(A_c(K)) - \hat{\lambda}_i)\Big\},\ k = 1,\ldots,p,\ l = 1,\ldots,r,$$

where $\partial\lambda_i(A_c(K))/\partial K_{kl}$ is given by (4.5). □

The gradient of f stretched as a long column vector of the form:

$$g(\kappa) = \mathrm{Re}\Big\{G(\kappa)\,(\lambda(A_c(K)) - \hat{\lambda})\Big\}, \tag{4.6}$$

where $G(\kappa) \in \mathbb{R}^{m\times n}$ is the Jacobian matrix of $\check{f}(K)$ which is given by

$$G(\kappa) = \begin{bmatrix} (q_1^T B E_{11} C q_1) & \ldots & (q_n^T B E_{11} C q_n) \\ \vdots & \ddots & \vdots \\ (q_1^T B E_{pr} C q_1) & \ldots & (q_n^T B E_{pr} C q_n) \end{bmatrix},$$

and the $i := (k,l)$th component of $g(\kappa)$ takes the form:

$$g_i(\kappa) = \mathrm{Re}\Big\{q_1^T B E_{kl} C q_1(\lambda_1(A_c) - \hat{\lambda}_1) + \cdots + q_n^T B E_{kl} C q_n(\lambda_n(A_c) - \hat{\lambda}_n)\Big\}, \tag{4.7}$$

where $i = 1,\ldots,m$.

The next lemma shows that the gradient g of the objective function is locally Lipschitz continuous.

Lemma 4.4. *Let A_c satisfy Assumption 4.1 and assume that $\Lambda(A_c(\kappa))$ is locally Lipschitz continuous in some neighborhood of the level set (2.1). Further suppose that $\|\kappa_1\| \neq \|\kappa_2\|$ for all $\kappa_1,\ \kappa_2 \in \mathcal{D}$. Then g is locally Lipschitz continuous in some neighborhood of (2.1).*

Proof. From (4.6) and for any two vectors $\kappa_1,\ \kappa_2 \in \mathcal{D}$ we have

$$||g(\kappa_1) - g(\kappa_2)|| = ||G(\kappa_1)(\lambda(A_c(\kappa_1)) - \hat{\lambda}) - G(\kappa_2)(\lambda(A_c(\kappa_2)) - \hat{\lambda})||$$

For simplicity, we write $\lambda_j(\kappa)$ and $\Lambda(\kappa)$ instead of $\lambda_j(A_c(\kappa))$ and $\Lambda(A_c(\kappa))$, respectively. Then:

$$\begin{aligned}
||g(\kappa_1) - g(\kappa_2)|| &\leq ||G(\kappa_1)\lambda(\kappa_1) - G(\kappa_2)\lambda(\kappa_2)|| + ||(G(\kappa_1) - G(\kappa_2))\hat{\lambda})|| \\
&\leq ||G(\kappa_1)\lambda(\kappa_1) - G(\kappa_2)\lambda(\kappa_1) + G(\kappa_2)\lambda(\kappa_1) - G(\kappa_2)\lambda(\kappa_2)|| \\
&+ ||(G(\kappa_1) - G(\kappa_2))\hat{\lambda})|| \\
&\leq ||G(\kappa_1) - G(\kappa_2)||\ ||\lambda(\kappa_1)|| + ||G(\kappa_2)||\ ||\lambda(\kappa_1) - \lambda(\kappa_2)|| \\
&+ ||G(\kappa_1) - G(\kappa_2)||\ ||\hat{\lambda}|| \\
&\leq ||G(\kappa_1) - G(\kappa_2)||\ (||\lambda(\kappa_1)|| + ||\hat{\lambda}||) + ||G(\kappa_2)||\ ||\lambda(\kappa_1) - \lambda(\kappa_2)|| \\
&\leq (||G(\kappa_1)|| + ||G(\kappa_2)||)\ (||\lambda(\kappa_1)|| + ||\hat{\lambda}||) + ||G(\kappa_2)||\ ||\lambda(\kappa_1) - \lambda(\kappa_2)||.
\end{aligned}$$

From the definition of $\Lambda(\kappa)$ we have

$$||\Lambda(\kappa_1) - \Lambda(\kappa_2)||^2 = \sum_{j=1}^{n} |\lambda_j(\kappa_1) - \lambda_j(\kappa_2)|^2 = ||\lambda(\kappa_1) - \lambda(\kappa_2)||^2,$$

and since $\Lambda(\kappa)$ is locally Lipschitz continuous, then $\forall \kappa_1,\ \kappa_2 \in \mathcal{D}$ there exists a constant $L > 0$ such that

$$||\lambda(\kappa_1) - \lambda(\kappa_2)|| \leq L||\kappa_1 - \kappa_2||.$$

Moreover, from Theorem 2.1 and the boundedness of the level set $\Omega(\kappa_0)$ we can deduce that there exists a constant $M_1 > 0$ such that:

$$||\lambda(\kappa)|| \leq M_1 \qquad \forall \kappa \in \Omega(\kappa_0).$$

The norm of $G(\kappa)$ is given by

$$||G(\kappa)||^2 = \sum_{k=1}^{p} \sum_{l=1}^{r} \sum_{j=1}^{n} |q_j^T B E_{kl} C q_j|^2,$$

where q_i's are normalized eigenvectors, which implies that $G(\kappa)$ is bounded, i.e. there exists a constant $M_2 > 0$ such that:

$$\|G(\kappa)\| \leq M_2 \qquad \forall \kappa \in \mathcal{D}.$$

Consequently,

$$\begin{aligned}
||g(\kappa_1) - g(\kappa_2)|| &\leq M_2 L\ ||\kappa_1 - \kappa_2|| + 2M_2(M_1 + M_3) \\
&\leq M_2 L\ ||\kappa_1 - \kappa_2|| + 2M_2(M_1 + M_3)\frac{||\kappa_1 - \kappa_2||}{||\kappa_1 - \kappa_2||} \\
&\leq M_2 L\ ||\kappa_1 - \kappa_2|| + 2M_2(M_1 + M_3)\frac{||\kappa_1 - \kappa_2||}{|\ ||\kappa_1|| - ||\kappa_2||\ |}
\end{aligned}$$

where $M_3 := ||\hat{\lambda}||$. By assumption $||\kappa_1|| \neq ||\kappa_2||\ \forall \kappa_1,\ \kappa_2 \in \mathcal{D}$. Then there exists a constant $M_4 > 0$ such that

$$\frac{1}{|\ ||\kappa_1|| - ||\kappa_2||\ |} \leq M_4.$$

This implies that

$$||g(\kappa_1) - g(\kappa_2)|| \leq \tilde{L}||\kappa_1 - \kappa_2||,$$

where $\tilde{L} = M_2 L + 2M_2(M_1 + M_3)M_4$. □

5 Nonlinear CG Method for the PAP

In this section the focus is to analyze and study three types of nonlinear CG method for calculating the local solution of the PAP problem (3.5).

For a given starting point $\kappa_0 \in \mathbb{R}^m$, the considered nonlinear CG methods generate a sequence of the form:

$$\kappa_{k+1} = \kappa_k + \alpha_k \delta\kappa_k, \tag{5.1}$$

such that $\{\kappa_k\} \subset \mathbb{R}^m$, where $\delta\kappa_k \in \mathbb{R}^m$ is a descent direction vector for f at κ_k and $\alpha_k > 0$ is the step-size. The new search direction $\delta\kappa_k$ for the nonlinear CG method is given by the following relation:

$$\delta\kappa_k = -g_k + \beta_k \delta\kappa_{k-1}, \qquad \delta\kappa_1 = -g_1, \tag{5.2}$$

where $g_k = g(\kappa_k)$ is the gradient of f at κ_k and β_k is a parameter. Now we recall a popular inexact line search condition, Wolfe conditions, see e.g. [33], in order to update a suitable step-size α_k for the new iterate (5.1):

$$f(\kappa_k + \alpha_k \delta\kappa_k) \leq f(\kappa_k) + \gamma\, \alpha_k g_k^T \delta\kappa_k \tag{5.3}$$

$$g(\kappa_k + \alpha_k \delta\kappa_k)^T \delta\kappa_k \geq \hat{\gamma}\, g_k^T \delta\kappa_k, \tag{5.4}$$

where $0 < \gamma < \hat{\gamma} < 1$. Moreover, the strong Wolfe condition replaces (5.4) by the following condition

$$\left| g(\kappa_k + \alpha_k \delta\kappa_k)^T \delta\kappa_k \right| \leq \hat{\gamma}\, \left| g_k^T \delta\kappa_k \right|. \tag{5.5}$$

5.1 Descent nonlinear CG method for the PAP

The PRP–CG method has been considered as one of the most efficient conjugate gradient methods in practical computation. It essentially performs a restart if a bad direction occurs. However, the PRP–CG method can cycle infinitely without approaching to any stationary point when the objective function is nonconvex, see e.g. [13]. Recently, Yu, Zhao and Wei [54] defined a new formula for β_k in order to avoid such a behavior as well as ensure the sufficient descent condition:

$$g_k^T \delta\kappa_k \leq -c||g_k||^2, \qquad c > 0, \tag{5.6}$$

for all $k \geq 0$ independent of the line search rule. The new search direction of this method is defined by

$$\delta\kappa_k = -g_k + \beta_k^N \delta\kappa_{k-1}, \qquad \delta\kappa_1 = -g_1, \tag{5.7}$$

and β_k^N is given by the following formula

$$\beta_k^N(\mu) = \begin{cases} \frac{||g_k||^2 - |g_k^T g_{k-1}|}{\mu |g_k^T \delta\kappa_{k-1}| + ||g_{k-1}||^2}, & \text{if } ||g_k||^2 \geq |g_k^T g_{k-1}|; \\ 0, & \text{otherwise} \end{cases} \tag{5.8}$$

where $\mu > 1$. Global convergence has been established for this method under the standard Wolfe conditions.

The following theorem shows that the search direction $\delta\kappa_k$ generated by (5.7) and (5.8) is a descent direction.

Theorem 5.1. *Let $\kappa \in \mathcal{D}$ generated by (5.1) and $\delta\kappa_k$ is calculated by (5.7), where $\beta_k = \beta_k^N$ is given by (5.8). Then for all $k \geq 1$*

$$g_k^T \delta\kappa_k \leq -\Big(1 - \frac{1}{\mu}\Big)||g_k||^2. \tag{5.9}$$

Proof (see also [54, Theorem 2.1]). For $k = 1$ we have

$$g_1^T \delta\kappa_1 = -||g_1||^2 < 0.$$

For $k \geq 2$ and $\mu > 1$ and using (5.7) gives

$$g_k^T \delta\kappa_k = -||g_k||^2 + \beta_k^N g_k^T \delta\kappa_{k-1}.$$

We obtain from the definition of β_k^N that

$$\beta_k^N \leq \frac{||g_k||^2}{\mu |g_k^T \delta\kappa_{k-1}|}.$$

Hence we obtain

$$g_k^T \delta\kappa_k = -||g_k||^2 + \beta_k^N g_k^T \delta\kappa_{k-1} \leq -||g_k||^2 + \frac{||g_k||^2}{\mu |g_k^T \delta\kappa_{k-1}|} |g_k^T \delta\kappa_{k-1}| \leq -\Big(1 - \frac{1}{\mu}\Big)||g_k||^2.$$

□

5.2 Descent VLS+CG method for the PAP

Another formula for updating β_k has been given in [54] and defined as the following:

$$\beta_k^{VLS+}(\mu) = \max(0, \beta_k^{VLS}), \tag{5.10}$$

where

$$\beta_k^{VLS} = \frac{||g_k||^2 - |g_k^T g_{k-1}|}{\mu |g_k^T \delta\kappa_{k-1}| - g_{k-1}^T \delta\kappa_{k-1}},$$

where $\mu \geq 1$. The formula $\beta_k^{VLS+}(\mu)$ possesses the same properties of the formula (5.8). The search direction for this method takes the following form:

$$\delta\kappa_k = -g_k + \beta_k^{VLS+} \delta\kappa_{k-1}, \qquad \delta\kappa_1 = -g_1. \tag{5.11}$$

The following theorem shows that the search direction $\delta\kappa_k$ generated by (5.11) together with (5.10) is a descent direction.

Theorem 5.2. *Let $\kappa \in \mathcal{D}$ be generated by (5.1) and $\delta\kappa_k$ is calculated by (5.11) where β_k^{VLS+} is given by (5.10). Then for all $k \geq 1$:*

$$g_k^T \delta\kappa_k \leq -\Big(1 - \frac{1}{\mu}\Big)||g_k||^2. \tag{5.12}$$

Proof. By using mathematical induction we have for $k = 1$ that

$$g_1^T \delta\kappa_1 = -||g_1||^2 < 0.$$

At $k - 1$ suppose that

$$g_{k-1}^T \delta\kappa_{k-1} \leq -c||g_{k-1}||^2, \tag{5.13}$$

where $c := (1 - \frac{1}{\mu})$ and $c > 0$. Form (5.11), we have

$$g_k^T \delta\kappa_k = -||g_k||^2 + \beta_k^{VLS+} g_k^T \delta\kappa_{k-1}.$$

If $\beta_k^{VLS+} = 0$, then $g_k^T \delta\kappa_k = -||g_k||^2 < 0$. Moreover, if $\beta_k^{VLS+} = \beta_k^{VLS}$ then we from the definition of β_k^{VLS} and (5.13) have

$$\beta_k^{VLS} \leq \frac{||g_k||^2}{\mu|g_k^T \delta\kappa_{k-1}| + c||g_{k-1}||^2}.$$

Hence

$$\beta_k^{VLS} \leq \frac{||g_k||^2}{\mu|g_k^T \delta\kappa_{k-1}|}.$$

Consequently

$$g_k^T \delta\kappa_k = -||g_k||^2 + \beta_k^{VLS} g_k^T \delta\kappa_{k-1} \leq -||g_k||^2 + \frac{||g_k||^2}{\mu|g_k^T \delta\kappa_{k-1}|}|g_k^T \delta\kappa_{k-1}| \leq -\Big(1 - \frac{1}{\mu}\Big)||g_k||^2.$$

□

5.3 Modified PRP–CG method for the PAP

In [28] the authors proposed a modification on the PRP method denoted by MPRP-CG method. The search direction of this method is defined as

$$\delta\kappa_k = -g_k + \beta_k^{MPRP} \delta\kappa_{k-1}, \qquad \delta\kappa_1 = -g_1, \tag{5.14}$$

in which the update parameter β_k^{MPRP} is given by:

$$\beta_k^{MPRP} = \begin{cases} \frac{||g_k||^2 - |g_k^T g_{k-1}|}{\max(0,\ g_k^T \delta\kappa_{k-1}) + ||g_{k-1}||^2}, & \text{if } ||g_k||^2 \geq |g_k^T g_{k-1}| \geq \bar{m}\,||g_k|| \\ 0, & \text{otherwise,} \end{cases} \tag{5.15}$$

where $\bar{m} \in (0, 1)$.

The MPRP–CG method generates sufficient descent directions with an inexact line search; see [28] for the proof.

Theorem 5.3. *Let $\kappa \in \mathcal{D}$ generated by (5.1) and $\delta\kappa_k$ be calculated by (5.14) with β_k given by (5.15). Then for all $k \geq 1$ it holds that*

$$g_k^T \delta\kappa_k \leq -||g_k||^2. \tag{5.16}$$

In order to find a local solution of the considered problem (3.5), we summarize the nonlinear CG algorithm as the follows.

Algorithm 5.1. Nonlinear CG method for the output feedback PAP]

0. Let $\kappa_0 \in \mathbb{R}^m$, $0 < \gamma < \hat{\gamma} < 1$ and $\epsilon \in (0, 1)$ be given. Moreover, let A, B, C be given constant matrices. Choose $\tilde{\lambda} \in \mathbb{C}^n$ and $\alpha_0 \in (0, 1]$. Compute $g(\kappa_0)$ and set $\delta\kappa_0 = -g(\kappa_0)$. If $||g(\kappa_0)|| \leq \epsilon$ or $f(\kappa_0) \leq \epsilon$ stop; otherwise set $k \leftarrow 0$ and go to the next step.

While $||g(\kappa_k)|| > \epsilon$ or $f(\kappa_k) > \epsilon$, do

1. Compute $\alpha_k > 0$ that satisfies Wolfe conditions (5.3)–(5.4), set $\kappa_{k+1} = \kappa_k + \alpha_k \delta\kappa_k$ and then calculate g_{k+1}.
2. If $\|g(\kappa_{k+1})\| \leq \epsilon$ or $f(\kappa_{k+1}) \leq \epsilon$ stop; otherwise go to the next step.
3. Calculate β_k by one of the formulas (5.8) or (5.10) or (5.15).
4. Obtain a new search direction $\delta\kappa_{k+1}$ by (5.2), set $k \leftarrow k+1$, and go to Step 1.

End (do)

Remark 5.4. From the problem structure and the definition of the objective function in (3.5), it is convenient to stop the CG method if $f(\kappa_k) \leq \epsilon$.

6 Convergence Analysis

In this section, the convergence analysis of Algorithm 5.1 is established. Next, assume that $g(\kappa_k) \neq 0$ for all k; otherwise a stationary point is found.

Assumption 6.1. *We assume the following throughout the paper.*

(a) *Boundedness: The level set $\Omega(\kappa_0)$ in (2.1) is bounded.*

(b) *Lipschitz continuity: In some neighborhood N of the level set (2.1) the gradient g of the objective function f is Lipschitz continuous.*

From Assumption 6.1 we can deduce that there exists a constant $M > 0$ such that

$$||g(\kappa)|| \leq M \qquad \forall \kappa \in \Omega(\kappa_0). \tag{6.1}$$

The following lemma, known as Zoutendijk's condition, is often used to prove global convergence properties of nonlinear CG methods.

Lemma 6.2 (see e.g. [13, Theorem 2.1])**.** *Let $\{\kappa_k\}$ be generated by Algorithm 5.1 and let Assumption 6.1 hold, then*

$$\sum_{k=1}^{\infty} \frac{(g_k^T \delta\kappa_k)^2}{||\delta\kappa_k||^2} < +\infty. \tag{6.2}$$

6.1 Convergence of the descent nonlinear CG method

The following result can be found in [54, Lemma 2.2] which is useful in our subsequent analysis.

Lemma 6.3. *Let $\sigma > 0$ and b be given constants, and assume that $\{\xi_i\}$ is series of positive terms satisfying the following inequality for all k:*

$$\sum_{i=1}^{k} \xi_i \geq \sigma k + b.$$

Then

$$\sum_{i \geq 1} \xi_i^2 / i = +\infty \quad \text{and} \quad \sum_{k \geq 1} \frac{\xi_k^2}{\xi_1 + \cdots + \xi_k} = +\infty.$$

Under the condition (6.2) the following convergence theorem is established.

Theorem 6.4. *Suppose that $\{\kappa_k\}$ is generated by Algorithm 5.1 with β_k^N updated by (5.8). Assume further that Assumptions 6.1 and (6.2) hold, then we have*

$$\liminf_{k\to\infty} ||g_k|| = 0. \tag{6.3}$$

Proof. From (5.7) we have:

$$\delta\kappa_k + g_k = \beta_k^N \delta\kappa_{k-1}, \qquad k \geq 1. \tag{6.4}$$

By squaring both sides of (6.4) we have

$$||\delta\kappa_k||^2 = -||g_k||^2 - 2g_k^T\delta\kappa_k + (\beta_k^N)^2||\delta\kappa_{k-1}||^2. \tag{6.5}$$

Since $\beta_k^N \leq \frac{||g_k||^2}{||g_{k-1}||^2}$ then we obtain the following:

$$||\delta\kappa_k||^2 = -||g_k||^2 - 2g_k^T\delta\kappa_k + (\beta_k^N)^2||\delta\kappa_{k-1}||^2 \leq -||g_k||^2 - 2g_k^T\delta\kappa_k + \Big(\frac{||g_k||^2}{||g_{k-1}||^2}\Big)^2||\delta\kappa_{k-1}||^2$$

$$\leq -||g_k||^2 - 2g_k^T\delta\kappa_k + ||g_k||^4\frac{||\delta\kappa_{k-1}||^2}{||g_{k-1}||^4}.$$

Now, we can write

$$t_k \leq t_{k-1} - \frac{1}{||g_k||^2} + \frac{2r_k}{||g_k||^2}, \tag{6.6}$$

where

$$t_k = \frac{||\delta\kappa_k||^2}{||g_k||^4}, \quad r_k = -\frac{g_k^T\delta\kappa_k}{||g_k||^2}.$$

Initially we have $t_1 = \frac{1}{||g_1||^2}$ and $r_1 = 1$. By summing up the two sides of (6.6) over all indices yields:

$$t_k \leq -\sum_{i=1}^{k}\frac{1}{||g_i||^2} + 2\sum_{i=1}^{k}\frac{|r_i|}{||g_i||^2}. \tag{6.7}$$

Now, suppose that (6.3) does not hold. Then there exists a positive scalar τ such that for all $k \geq 1$

$$||g_k|| \geq \tau. \tag{6.8}$$

Hence, it follows from (6.7) and (6.1) that

$$t_k \leq -\frac{k}{M} + \frac{2}{\tau}\sum_{i=1}^{k}|r_i|, \tag{6.9}$$

which gives

$$t_k \leq \frac{2}{\tau}\sum_{i=1}^{k}|r_i|. \tag{6.10}$$

Using the fact that $t_k \geq 0$ then from (6.9) we obtain

$$\sum_{i=1}^{k}|r_i| \geq \frac{\tau k}{2M}. \tag{6.11}$$

Combining (6.11) with (6.10) and lemma 6.3 we get:

$$\sum_{k\geq 1}\frac{(g_k^T\delta\kappa_k)^2}{||\delta\kappa_k||^2}=\sum_{k\geq 1}\frac{r_k^2}{t_k}=\infty,$$

contradicting with Zoutendijk's condition (6.2). This implies that (6.3) holds and the proof is complete. □

6.2 Convergence of the method VLS+CG

The global convergence of the VLS+CG method is given in the following theorem. The proof will be omitted because it is similar to theorem 6.4.

Theorem 6.5. *Suppose that $\{\kappa_k\}$ is generated by Algorithm 5.1 with β_k^{VLS+} updated by (5.10). Assume further that Assumptions 6.1 and (6.2) hold, then we have*

$$\liminf_{k\to\infty}||g_k||=0.$$

6.3 Convergence of the method MPRP

The following theorem indicates the global convergence of the method MPRP; see [28, Theorem 3.5] for the proof.

Theorem 6.6. *Suppose that $\{\kappa_k\}$ is generated by Algorithm 5.1 with β_k^{MPRP} updated by (5.15). Assume further that Assumptions 6.1 and (6.2) hold, then we have*

$$\liminf_{k\to\infty}||g_k||=0.$$

7 Extending the CG Methods for Decentralized Control Systems

In engineering and finance there are many applications of large-scale control systems that are often composed of lower order subsystems using the decentralized technique, for example aerospace systems, computer network systems; see, e.g., [41] for details on decentralized control systems.

Consider the linear time-invariant decentralized control system with ν control stations:

$$\begin{aligned}\dot{x}(t)&=Ax(t)+\sum_{i=1}^{\nu}B_iu_i(t),\quad x(0)=x_0,\\ y_i(t)&=C_ix(t),\end{aligned}\tag{7.1}$$

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

The output feedback PAP for the decentralized system (7.1) is to find output feedback gain matrices $K_i\in\mathbb{R}^{p_i\times r_i}$ that assign by using the control law

$$u_i(t)=K_iy_i(t),\qquad i=1,...,\nu.$$

the poles of the closed-loop system matrix $A_c(K_1,...,K_\nu)=A+\sum_{i=1}^{\nu}B_iK_iC_i$ in a pre-specified region in the complex plane.

In fact, the considered CG methods can be extended to tackle output feedback PAP for decentralized control systems which can be viewed as a special case of the original formulation. Let us define the matrices

$$B = [B_1 \ \dots \ B_\nu], \quad C = \left[C_1^T \ \dots \ C_\nu^T\right]^T, \quad K = \mathrm{diag}(K_1, \dots, K_\nu).$$

Under this setting we have $A_c(K) = A + BKC$ and the minimization problem (3.5) now takes the following form:

$$\min_{K_1,\dots,K_\nu} \ f(K_1, ..., K_\nu) := \frac{1}{2}\sum_{j=1}^{n}(\lambda_j(A_c(\cdot)) - \hat{\lambda}_j)^*(\lambda_j(A_c(\cdot)) - \hat{\lambda}_j). \tag{7.2}$$

Let us further define $\kappa = [\kappa_1^T, \dots, \kappa_\nu^T]^T$, where each $\kappa_i \in \mathbb{R}^{p_i \cdot r_i}$ is obtained by stretching the matrix K_i into a long column vector, where $m = \sum p_i \cdot r_i$. For a given starting point $\kappa_0 \in \mathbb{R}^m$ any of the considered nonlinear CG methods generate a sequence of the form:

$$\kappa_{k+1} = \kappa_k + \alpha_k \delta\kappa_k, \quad k = 0, 1, 2, ...,$$

where $\delta\kappa_k \in \mathbb{R}^m$ is a descent direction for f_k at the point κ_k and α_k is the step-size that must satisfy Wolfe conditions (5.3)–(5.4).

The new search directions d_{k+1} is updated by:

$$\delta\kappa_{k+1} = -g_{k+1} + \beta_k \delta\kappa_k, \qquad \delta\kappa_0 = -g_0, \tag{7.3}$$

where β_k is given by one of the formulas (5.8), (5.10) or (5.15), and $g_k = \nabla f_k$ stretched as a column vector in $\mathbb{R}^m$.

Algorithm 5.1 is extended in the following lines to tackle the decentralized output feedback PAP problem.

Algorithm 7.1 (Nonlinear CG method for the decentralized output feedback PAP)**.**

0. Let $\kappa_0 \in \mathbb{R}^m$, $\epsilon \in (0, 1)$, $0 < \gamma < \hat{\gamma} < 1$ and $\bar{m} \in (0, 1)$ be given. Moreover, let $A, B_1, ..., B_\nu, C_1, ..., C_\nu$ be given constant matrices. Choose $\tilde{\lambda} \in \mathbb{C}^n$ and $\alpha_0 \in (0, 1]$. Compute $g_0 = \nabla f(\kappa_0) \in \mathbb{R}^m$ and set $\delta\kappa_0 = -g_0$. If $\|g_0\| \leq \epsilon$ or $f(\kappa_0) \leq \epsilon$ stop; otherwise set $k := 0$ and go to the next step.

While $\|g_k\| > \epsilon$ or $f(\kappa_k) > \epsilon$, do

1. Compute $\alpha_k > 0$ that satisfies Wolfe conditions (5.3)–(5.4), set $\kappa_{k+1} = \kappa_k + \alpha_k \delta\kappa_k$, and calculate the gradient vector g_{k+1}.
2. If $\|g_{k+1}\| \leq \epsilon$ or $f(\kappa_{k+1}) \leq \epsilon$ stop; otherwise go to the next step.
3. Calculate β_k by one of the formulas (5.8), (5.10) or (5.15), e.g.,

$$\beta_k^{MPRP} = \begin{cases} \frac{||g_{k+1}||^2 - |g_{k+1}^T g_k|}{\max\{0,\ g_{k+1}^T \delta\kappa_k\} + ||g_k||^2}, & \text{if } ||g_{k+1}||^2 \geq |g_{k+1}^T g_k| \geq \bar{m}\,||g_{k+1}|| \\ 0, & \text{otherwise} \end{cases}$$

4. Obtain a new search direction by

$$\delta\kappa_{k+1} = -g_{k+1} + \beta_k\, \delta\kappa_k,$$

set $k \leftarrow k + 1$, and go to Step 1.

End (do)

8 Minimizing the Spectral Condition Number

Motivated by the seminal work of Lam and Yan [23] we consider in this section the pole assignment with robustness measured in terms of the spectral condition number of the closed-loop eigenvector matrix. The spectral condition number of the closed-loop system matrix is one of the most accepted measures of robustness.

In this section we consider the state feedback problem, where $C = I_n$ in the control system (3.2). Without loss any of the generality we assume that B is of full column rank. Given a stable real matrix $\tilde{\Lambda}$, the PAP is to find a feedback gain matrix $K \in \mathbb{R}^{p\times n}$ such that the closed-loop system matrix $A + BK$ and $\tilde{\Lambda}$ have the same eigenvalues. In other words, there is a state transformation matrix $T \in \mathbb{R}^{n\times n}$ such that

$$(A + BK)T = T\tilde{\Lambda}. \tag{8.1}$$

Lam and Yan [23] have replaced (8.1) by the following equations:

$$AT - T\tilde{\Lambda} + BH = 0, \tag{8.2}$$

$$H = KT, \tag{8.3}$$

where $T = T(H)$ is assumed to be nonsingular. The advantage of introducing H to replace KT is that (8.2) is linear in T and H. Furthermore, (8.2) takes the form of a Sylvester equation, where $T(H)$ is an injective function that determines for each H a unique $K = HT^{-1}$ via (8.3).

The state feedback PAP is to find $H \in \mathbb{R}^{p\times n}$ local solution of the following minimization problem:

$$\min_{H} \quad J(H) = \|T(H)\|^2 + \|T^{-1}(H)\|^2, \tag{8.4}$$

where $T(H)$ solves Sylvester's equation (8.2) which is assumed to be nonsingular.

It has been shown in [23] that the solution of the problem (8.4) is equivalent to solve the following minimization problem for an increasing sequence of positive integers $\{q\}$:

$$\min_{H} \psi^q(H) = \left(\frac{1}{2} J([T(H)^T\ T(H)]^q)\right)^{\frac{1}{2q}}, \tag{8.5}$$

where $T(H)$ solves Sylvester's equation (8.2). This formulation is characterized by the gradient of the objective function $\psi^q(H)$. In order to obtain the gradient of $\psi^q(H)$ let us define the following mapping:

$$S(H) = (T(H)^T T(H))^{2q} + (T(H)^T T(H))^{-2q} = S(H)^T. \tag{8.6}$$

Then the gradient of the objective function $\psi^q(H)$ is given by (see [23]):

$$\nabla\psi^q(H) = \frac{1}{(\psi^q(H))^{2q-1}} B^T D, \tag{8.7}$$

where D solves Sylvester's equation:

$$A^T D - D\tilde{\Lambda}^T + T(H)^{-T} S(H) = 0. \tag{8.8}$$

We have extended the three CG methods NCG, VLS+CG and MPRP-CG to tackle Problem (8.5) in which Wolfe conditions (5.3)–(5.4) that globalize the CG methods are

rewritten as:

$$\psi^q(H_k + \alpha_k d_k) \leq \psi^q(H_k) + \gamma\, \alpha_k\, Tr(\nabla\psi^q(H_k)^T d_k) \tag{8.9}$$

$$Tr(\nabla\psi^q(H_k + \alpha_k d_k)^T d_k) \geq \hat{\gamma}\, Tr(\nabla\psi^q(H_k)^T d_k), \tag{8.10}$$

where $Tr(\cdot)$ is the trace operator and $d_k \in \mathbb{R}^{p\times n}$ is the search direction evaluated by the CG method.

Algorithm 8.1 (Nonlinear CG methods for Problem (8.5))**.**

0. Let $H_0 \in \mathbb{R}^{p\times n}$, $\epsilon \in (0,1)$ and $0 < \gamma < \hat{\gamma} < 1$ be given. Moreover, let A, B be given constant matrices. Choose $\tilde{\Lambda}$ diagonal matrix of desired eigenvalues and $\alpha_0 \in (0,1]$. Solve (8.2) for $T_0 = T(H_0)$. Calculate $S(T_0)$ by (8.6) and obtain D_0 solution of Sylvester's equation (8.8). Compute

$$\psi^q(H_0) = \left(\frac{1}{2} J([T_0^T\ T_0]^q)\right)^{\frac{1}{2q}},$$

and then use (8.7) to compute $\nabla\psi^q(H_0)$. Set $d_0 = -\nabla\psi^q(H_0)$. If $||\nabla\psi^q(H_0)|| \leq \epsilon$ stop; otherwise set $k := 0$, $q = 1$ and go to the next step.

While $||\nabla\psi^q(H_k)|| > \epsilon$, do

1. Compute $\alpha_k > 0$ that satisfies Wolfe conditions (8.9)–(8.10). Set $H_{k+1} = H_k + \alpha_k d_k$ and then calculate $\nabla\psi^q(H_{k+1})$.
2. If $\|\nabla\psi^q(H_{k+1})\| \leq \epsilon$ stop; otherwise go to the next step.
3. Calculate β_k by one of the formulas (5.8)or

(5.10) or (5.15), where $g_k = \nabla\psi^q(H_{k+1})$. Then obtain a new search direction by:

$$d_{k+1} = -\nabla\psi^q(H_{k+1}) + \beta_k d_k.$$

4. Set $q \leftarrow \lceil \frac{q}{2} \rceil + 1$ and $k \leftarrow k+1$ and go to Step 1.

End (do)

The symbol $\lceil q \rceil$ denotes the smallest integer greater than or equal q.

9 Numerical Results

In this section, some preliminary test of the proposed three CG methods NCG, VLS+CG and MPRP-CG for solving the two considered PAP. The methods are implemented using Matlab and all results are using a 3.07 Ghz Pentium 4 CPU with 1 GB RAM. Numerical results for the three CG methods for finding the local solution of the least–squares problem (3.5) are given first followed by the results on the formulation (8.5).

In the following, we consider eight test problems in details that quite demonstrate the performance of the considered methods. Among the considered test problems are two examples that test the ability of the considered methods for achieving the desired poles for

the decentralized PAP problem. The methods are compared with respect to number of iterations as well as the CPU time.

In feedback control it is desirable for the poles of the closed-loop system matrix $A_c(K)$ to be in the negative side of its complex plane. Therefore, the vector $\hat{\lambda}$ of desired eigenvalues is chosen as follows: $\hat{\lambda} = \lambda(A) - \vartheta(A) - s$, where

$$\vartheta(A) = \max \operatorname{Re} \Lambda(A),$$

is the spectral abscissa of the system matrix A and s is the chosen shift, which is assigned the two values $s = 0.1$ and 0.3. Let $\tilde{\Lambda}$ be a diagonal matrix with desired eigenvalues $\hat{\lambda}$ on its main diagonal.

Most of the considered test problems were chosen from the benchmark collection COMPlib [24], while other test problems were chosen from different sources of the literature. For all test problems the methods start with K_0 being an array of ones. In the three methods NCG, VLS+CG and MPRP-CG the globalization strategy uses the sufficient decrease condition (5.3), where $\gamma = 10^{-4}$. The CG methods stop if the objective function f_k is strictly less than the prescribed accuracy 1×10^{-4}.

Example 9.1. This test problem is from the benchmark collection COMPlib [24, REA1] of a chemical reactor model. The constant data matrices are:

$$A = \begin{bmatrix} 1.3800 & -0.2077 & 6.715 & -5.676 \\ -0.5814 & -4.2900 & 0 & 0.675 \\ 1.0670 & 4.2730 & -6.654 & 5.893 \\ 0.0480 & 4.2730 & 1.343 & -2.104 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ 5.679 & 0 \\ 1.136 & -3.146 \\ 1.136 & 0 \end{bmatrix}, C^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ -1 & 0 & -1 \end{bmatrix}.$$

The system matrix A has the eigenvalues $(1.9910, 0.0635, -5.0566, -8.6659)$.

In the first case of the desired eigenvalues $\hat{\lambda}$ the methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points, where the total number of iterations are 68, 150 and 53, and the corresponding CPU times are 1.66, 4.63 and 1.06 sec., respectively. The achieved final output feedback gain matrices are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} 0.6461 & 0.0385 & 0.3534 \\ 2.5001 & 1.0344 & 0.2251 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} 0.4250 & 0.4498 & 0.2307 \\ 3.7810 & 2.2761 & -0.3072 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} 0.5579 & -0.0698 & 0.2239 \\ 2.3165 & 1.0754 & 0.2158 \end{bmatrix}.$$

For the second choice of $\hat{\lambda}$ the three methods converge to three different stationary points after 89, 44 and 50 iterations with CPU times 1.55, 0.91 and 1.47 sec., respectively.

Example 9.2. This test problem is the aircraft model in cruise flight conditions [24, AC3] and the given data matrices are the following:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & -0.154 & -0.0042 & 1.54 & 0 \\ 0 & 0.249 & -1 & -5.2 & 0 \\ 0.0386 & -0.996 & -0.0003 & -0.117 & 0 \\ 0 & 0.5 & 0 & 0 & -0.5 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ -0.744 & -0.032 \\ 0.337 & -1.12 \\ 0.02 & 0 \\ 0 & 0 \end{bmatrix},$$

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

The system matrix A has the eigenvalues $(-0.5, -0.0882 \pm 1.2695\mathrm{i}, -1.0855, -0.0092)$.

The three methods are executed for the above mentioned choices of the desired poles $\hat{\lambda}$. In the first case the methods NCG, VLS+CG and MPRP-CG successfully converge to different stationary points after 49, 30 and 49 iterations and the corresponding CPU times are 0.56, 0.58 and 0.42 sec., respectively. The achieved final output feedback gain matrices are, respectively:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} 0.2142 & 0.3095 & 0.8971 & -0.0219 \\ -2.2316 & 0.4350 & -0.3988 & 0.2360 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} 0.0327 & 1.0830 & 1.6250 & 0.4066 \\ -2.4399 & 0.8059 & -2.1840 & -0.7261 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CGCG}} = \begin{bmatrix} 0.2399 & 0.2702 & 0.8491 & -0.0855 \\ -1.5322 & 0.3904 & -0.2745 & 0.3993 \end{bmatrix}.$$

In the second choice of $\hat{\lambda}$ the three methods converge to different stationary points after 22, 14 and 22 iterations with CPU times 0.28, 0.2 and 0.25 sec., respectively. The following final output feedback gain matrices are obtained:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} 1.1080 & 0.5960 & 0.6715 & 0.2006 \\ -1.4487 & 0.7922 & -0.6779 & 0.4254 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} 1.1267 & 0.6777 & 0.9852 & 0.1800 \\ -1.2914 & 0.8189 & -0.5278 & 0.5902 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} 1.5231 & 0.6225 & 0.7912 & 0.1456 \\ -1.2219 & 0.5060 & -0.3288 & 0.5238 \end{bmatrix}.$$

The method VLS+CG has the best performance with respect to number of iterations, while the methods NCG and MPRP-CG have the same performance for the two cases.

Example 9.3. This test problem is the helicopter model from the benchmark collection [24, HE4]. The data matrices A, B, C are with sizes $n = 8, p = 4, r = 6$; therefore we skip listing them. The system matrix A has the eigenvalues

$$(-11.4968, -2.3036, 0.2342 \pm 0.5513\mathrm{i}, -0.1593 \pm 0.5990\mathrm{i}, -0.7104, -0.2923).$$

In the first case the methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points with total number of iterations 375, 160 and 107 while the CPU times are 20.08, 6.89 and 2.91 sec., respectively. The achieved final output feedback gain matrices are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -0.2013 & 0.5121 & 1.6305 & -0.0763 & -0.0454 & 1.2444 \\ -0.0415 & -0.5246 & 1.8226 & 0.5507 & 1.9665 & -0.4091 \\ 0.2899 & 1.3814 & 1.2405 & 0.1876 & 0.5078 & 1.8371 \\ 0.3483 & 1.2725 & 0.8788 & 0.4450 & 0.9255 & 1.1491 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} 0.0164 & 1.1334 & 1.9249 & -2.3744 & 0.0063 & 0.8307 \\ -0.1374 & -1.4122 & 2.0033 & 0.8213 & 2.3653 & -0.2127 \\ 0.0638 & 2.1951 & 1.5525 & 0.2561 & 0.6073 & 1.9756 \\ -0.2038 & 1.3392 & 0.2809 & 1.5014 & 0.8677 & 1.4136 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.2521 & 0.4579 & 1.3451 & -0.3887 & -0.0993 & 1.3143 \\ -0.0734 & -0.2506 & 1.6620 & 1.2145 & 1.4120 & 0.1027 \\ 0.2784 & 1.0921 & 1.7064 & 0.2689 & 0.4127 & 1.6028 \\ 1.0728 & 1.2140 & 0.8327 & 0.8170 & 0.9485 & 1.0981 \end{bmatrix}.$$

In the second choice of $\hat{\lambda}$ the methods NCG, VLS+CG and MPRP-CG require 159, 190 and 73 iterations to converge to three different stationary points with CPU times 4.14, 6.55 and 2.7 sec., respectively. As can be seen the method MPRP-CG has the best performance with respect to number of iterations and CPU time for the two cases.

Example 9.4. This example is a modified version of a test problem borrowed from [43] which has the following data matrices:

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

It represents a system with a symmetric state space realization, i.e., $A = A^T, C = B^T$. Moreover, the eigenvalues of A are $(1, -2, -3, -4)$.

For the first choice of $\hat{\lambda}$ the methods NCG, VLS+CG and MPRP-CG successfully converge to different stationary points after 14, 21 and 18 iterations, where the corresponding CPU times are 0.19, 0.47 and 0.33 sec., respectively. The achieved final output feedback gain matrices are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -2.2562 & 1.1457 & 1.5732 \\ 1.1457 & -1.5743 & 0.3055 \\ 1.5732 & 0.3055 & -1.3899 \end{bmatrix}, K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -1.4410 & 0.7021 & 0.8503 \\ 0.7021 & -1.1394 & -0.0908 \\ 0.8503 & -0.0908 & -1.0829 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -1.6510 & 0.8364 & 1.1695 \\ 0.8364 & -1.0734 & -0.1315 \\ 1.1695 & -0.1315 & -1.3558 \end{bmatrix}.$$

For the second choice of $\hat{\lambda}$ the methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points after 23, 32 and 22 iterations with CPU times 0.33, 0.61 and 0.3 sec., respectively. The corresponding final output feedback gain matrices are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -2.5859 & 1.3030 & 1.5822 \\ 1.3030 & -1.7263 & 0.3960 \\ 1.5822 & 0.3960 & -1.5756 \end{bmatrix}, K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -1.9321 & 1.0445 & 1.1332 \\ 1.0445 & -1.5454 & -0.2000 \\ 1.1332 & -0.2000 & -1.0925 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -1.6042 & 0.3555 & 0.8681 \\ 0.3555 & -1.5577 & 0.5924 \\ 0.8681 & 0.5924 & -1.2472 \end{bmatrix}.$$

Table 1 shows the overall performance of the three methods NCG, VLS+CG and MPRP-CG. In this table we list the average number of iterations, the average CPU time in seconds, and the success rate. The method VLS+CG is the best among the three methods with respect to number of iterations in the considered two cases, while the method NCG was more robust. Furthermore, the performance of the method MPRP-CG lies between the two methods NCG and VLS+CG.

In Table 2 we compare the performance of the three methods NCG, VLS+CG and MPRP-CG on test problems from different sources. In Table 3 we do the same comparison on test problems chosen from the benchmark collection [24]. Similar to the above four examples we test each test problem for two instances of the vector $\hat{\lambda}$ of desired poles, namely

Table 1: The performance of the three methods for $s = 0.1,\ 0.3$

	NCG		VLS+CG		MPRP-CG	
	$s = 0.1$	$s = 0.3$	$s = 0.1$	$s = 0.3$	$s = 0.1$	$s = 0.3$
Av. No. of iterations	148.63	252.28	76.58	121.67	110.71	160
Av. CPU time (sec.)	5.98	9.62	3.35	4.06	3.22	5.14
Success rate (%)	97.56	95.12	92.68	87.8	92.68	90.24

$\hat{\lambda} = \lambda(A) - \vartheta(A) - 0.1$ and $\hat{\lambda} = \lambda(A) - \vartheta(A) - 0.3$. For each test problem we include the problem size (n, p, r), the spectral abscissa $\vartheta(A)$ of the system matrix A, the total number of iterations and the CPU time.

Table 2: Performance of the methods NCG, VLS+CG and MPRP-CG on test problems from different sources for two different choices of the desired poles $\hat{\lambda}$.

	Problem size				No. of iterations and CPU-time (sec.)					
					NCG		VLS+CG		MPRP-CG	
Problem	n	p	r	$\vartheta(A)$	# it.	CPU	# it.	CPU	# it.	CPU
[22]	2	2	2	−1.0e−002	128	1.67	96	0.83	128	1.25
					392	6.42	24	1.7	290	3.66
[1]	4	2	2	7.1e−001	71	1.5	91	3.81	79	1.11
					104	1.92	110	4.45	108	2.13
[19]	4	2	4	1.9e+000	31	0.75	42	1.06	41	1
					26	0.75	37	1.2	26	0.58
[43]	4	3	2	2	71	0.96	50	0.84	104	1.41
					124	25.63	135	2.3	166	3.3
[43]	4	2	2	2	644	9.08	166	3.89	162	3.27
					84	4.61	218	4.52	77	2.27
[43]	4	3	3	1	14	0.19	21	0.47	18	0.33
					23	0.33	32	0.61	22	0.3
[31]	3	1	3	1.1e−001	71	0.72	50	1.33	55	0.7
					109	2.64	276	7.34	77	1.13
[6]	6	2	3	−4.0e−001	345	12.36	118	4.23	122	7.14
					54	1.39	17	0.7	169	3.72
[6]	3	2	2	9.85e+000	25	0.72	32	0.88	18	0.73
					31	0.67	23	0.81	38	1.27
[3]	5	3	2	2.3e−002	1225	46.53	443	33.27	1077	33.78
					1893	61.92	459	15.17	1252	39.3
[34]	3	2	2	3	34	1.77	47	1.77	39	1.49
					73	2.36	38	1.45	85	2.63
[39]	2	2	2	−0.1e−002	127	9.61	49	1.47	91	3.94
					1962	69.19	141	4	23	0.47
[27]	2	2	2	−568.23e−002	22	0.55	21	0.67	16	0.2
					25	0.66	30	0.69	14	0.31
[36]	2	2	2	1	20	0.31	15	0.3	15	0.25
					20	0.28	15	0.45	16	0.33
[9]	2	2	2	14.14e−002	10	0.06	9	0.28	9	0.16
					7	0.13	9	0.27	7	0.11
[36]	2	2	2	1	7	0.11	8	0.16	12	0.09
					9	0.19	5	0.11	13	0.14

It is important to point out that for the considered set of test problems the three methods often converge to different stationary points despite the fact that they start from the same point K_0. We may return this behavior to the inherent characteristics of the problem. On the other side, despite the assumption that $\Lambda(A_c(K))$ must have distinct eigenvalues for all K the methods NCG, VLS+CG and MPRP-CG successfully achieve stationary points while $\hat{\lambda}$ contain repeated values.

9.1 Performance of the CG methods for decentralized PAP

The performance of the considered CG methods for tackling the decentralized output feedback PAP is described by the following examples.

Example 9.5. This test problem is from [50]. It has two control stations and the constant data matrices are the following:

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

$$C_1 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \end{bmatrix}, C_2 = \begin{bmatrix} 0 & 0 & 1 & -1 & 1 \end{bmatrix}.$$

The system matrix A has the eigenvalues $(-0.2592, -0.7904 \pm 0.1470\mathrm{i}, -1.4535, -1.8564)$. In the first case of the desired eigenvalues $\hat{\lambda}$, starting from

$$K_0 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

the methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points, where the total number of iterations are 138, 159 and 219, and the corresponding CPU times are 3.84, 25.16 and 6.23 sec., respectively. The achieved final output feedback gain matrices are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} 0.0043 & 0.1988 & 0 \\ 0.3422 & 0.4097 & 0 \\ 0 & 0 & 0.4482 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} 0.1192 & 0.0111 & 0 \\ 2.1671 & 0.2765 & 0 \\ 0 & 0 & 0.2531 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.0058 & 0.1929 & 0 \\ 0.3198 & 0.3981 & 0 \\ 0 & 0 & 0.4471 \end{bmatrix}.$$

In the second choice of $\hat{\lambda}$ starting from the zero matrix for K_0 the three methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points, where the total number of iterations are 3, 5 and 4 and the corresponding CPU times are 0.47, 0.52 and 0.44 sec., respectively. The following final output feedback gain matrices are obtained:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -0.0832 & 0.0245 & 0 \\ -0.2356 & -0.0709 & 0 \\ 0 & 0 & -0.0169 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -0.1332 & 0.0065 & 0 \\ -0.3953 & -0.1399 & 0 \\ 0 & 0 & -0.0176 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.0987 & 0.0184 & 0 \\ -0.2885 & -0.0927 & 0 \\ 0 & 0 & -0.0171 \end{bmatrix}.$$

Table 4 indicates the final eigenvalues of the matrix $A_c = A + B_1K_1C_1 + B_2K_2C_2$ after applying the three methods on this test problem comparing with the desired eigenvalues $\tilde{\lambda}$ for $s = 0.3$.

Example 9.6. This test problem has three control stations and the constant data matrices are [42]:

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 2 \\ 1 & 2 & 3 \end{bmatrix}, B_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, B_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, B_3 = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$
$$C_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, C_2 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}, C_3 = \begin{bmatrix} 1 & 2 & 2 \end{bmatrix}.$$

The system matrix A has the eigenvalues $(-0.4495, 1.0000, 4.4495)$. For the first case of the desired eigenvalues $\hat{\lambda}$, starting from the zero matrix for K_0. The three methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points after 40, 76 and 48 iterations and with CPU times 0.89, 1.34 and 0.89 sec., respectively. The achieved stationary points are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -0.4291 & 0 & 0 \\ 0 & 9.3166 & 0 \\ 0 & 0 & -3.4151 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -0.4296 & 0 & 0 \\ 0 & 9.3119 & 0 \\ 0 & 0 & -3.4142 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.4259 & 0 & 0 \\ 0 & 9.2909 & 0 \\ 0 & 0 & -3.4095 \end{bmatrix}.$$

For the second choice of $\hat{\lambda}$ starting from the zero matrix for K_0 the three methods NCG, VLS+CG and MPRP-CG successfully converge to three different stationary points after 61, 54 and 31 iterations, where the corresponding CPU times are 0.99, 1.64 and 0.72 sec., respectively. The final stationary points are:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -0.4897 & 0 & 0 \\ 0 & 9.9403 & 0 \\ 0 & 0 & -3.5601 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -0.4926 & 0 & 0 \\ 0 & 9.9653 & 0 \\ 0 & 0 & -3.5657 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.4888 & 0 & 0 \\ 0 & 9.9353 & 0 \\ 0 & 0 & -3.5589 \end{bmatrix}.$$

9.2 Performance of the CG methods on Problem (8.5)

The following two examples represent the application of the three CG methods on Problem (8.5) of the state feedback PAP.

Example 9.7. This test problem is taken from [22]. The given data matrices are the following:

$$A = \begin{bmatrix} 0.9512 & 0 & 0 & 0 \\ 0 & 0.9048 & -1.1895 & 3.569 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 4.877 & 4.877 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, C = I_4.$$

The system matrix A has the eigenvalues $(0.9512, 0.9048, 0, 0)$.

The three methods are executed for the above mentioned choices of the desired poles $\tilde{\Lambda}$. In the first case the methods NCG, VLS+CG and MPRP-CG successfully converge to different stationary points after 7, 6 and 7 iterations and the corresponding CPU times are 0.25, 0.22 and 0.24 sec., respectively. The achieved final output feedback gain matrices are, respectively:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -0.3411 & 0.4426 & -0.4469 & 1.6809 \\ -0.1130 & -0.4281 & 0.5497 & -1.5430 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -0.3402 & 0.4150 & -0.4326 & 1.6281 \\ -0.1152 & -0.4358 & 0.5599 & -1.5515 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CGCG}} = \begin{bmatrix} -0.3463 & 0.3942 & -0.4044 & 1.6060 \\ -0.1189 & -0.4302 & 0.5660 & -1.5318 \end{bmatrix}.$$

In the second choice of $\tilde{\Lambda}$ the three methods converge to different stationary points after 5, 6 and 3 iterations with CPU times 0.17, 0.16 and 0.09 sec., respectively. The following final output feedback gain matrices are obtained:

$$K_*^{\mathrm{NCG}} = \begin{bmatrix} -0.4451 & 0.5654 & -0.3299 & 2.1692 \\ -0.1489 & -0.5674 & 0.7257 & -1.7777 \end{bmatrix},$$

$$K_*^{\mathrm{VLS+CG}} = \begin{bmatrix} -0.4467 & 0.5327 & -0.3076 & 2.1193 \\ -0.1509 & -0.5738 & 0.7346 & -1.7831 \end{bmatrix},$$

$$K_*^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.4261 & 0.6495 & -0.4289 & 2.2558 \\ -0.1512 & -0.5606 & 0.7283 & -1.7601 \end{bmatrix}.$$

Example 9.8. This test problem is borrowed from [51] which has the following data matrices:

$$A = \begin{bmatrix} -50 & 0 & 0 \\ 0.25 & -0.125 & 0.357 \\ 1000.05 & -0.025 & -2.245 \end{bmatrix}, B = \begin{bmatrix} 50 \\ 0 \\ -100 \end{bmatrix}, C = I_3,$$

The system matrix A has the eigenvalues $(-2.2408, -0.1292, -50)$.

In the first case of the desired poles $\tilde{\Lambda}$, the methods NCG, VLS+CG and MPRP-CG converge to the same stationary points after 22, 17 and 21 iterations, where the corresponding CPU times are 1.16, 0.91 and 1.09 sec., respectively. The achieved final output feedback gain matrices in the first case are:

$$K_{1*}^{\mathrm{NCG}} = K_{1*}^{\mathrm{VLS+CG}} = K_{1*}^{\mathrm{MPRP-CG}} = \begin{bmatrix} 0.0019 & 0.0002 & 0.0001 \end{bmatrix}.$$

In the second choice of $\tilde{\Lambda}$ the methods NCG, VLS+CG and MPRP-CG also successfully converge to the same stationary points after 25, 39 and 23 iterations and CPU times 1.2, 2 and 0.84 sec., respectively. The corresponding final output feedback gain matrices are the following:

$$K_{2*}^{\mathrm{NCG}} = K_{2*}^{\mathrm{VLS+CG}} = K_{2*}^{\mathrm{MPRP-CG}} = \begin{bmatrix} -0.0110 & -0.0012 & -0.0004 \end{bmatrix}.$$

Table 5 shows the overall performance of the three methods NCG, VLS+CG and MPRP-CG for finding a local solution of the minimization problem (8.5). In this table we list the average number of iterations and the average CPU time in seconds. We conclude that the method VLS+CG is the best among the three methods with respect to the CPU-time in the two considered cases.

In Tables 6 and 7 we compare the performance of the three methods NCG, VLS+CG and MPRP-CG on test problems from different sources as well as from the benchmark collection [24], respectively. We test each test problem for two shifts in the negative side of the complex plane namely $s = 0.1, 0.3$. For each test problem we include the problem size (n, p), the spectral abscissa $\vartheta(A)$ of the system matrix A, the total number of iterations and the CPU time.

Conclusions and future research

Conjugate gradient methods have played a special role for solving large scale optimization problems due to the simplicity of their iteration, convergence properties and their low memory requirements. In this article the output feedback PAP is considered which is a special algebraic inverse eigenvalue problem. Two formulations of the problem are considered; the first one is a nonlinear least–squares problem while the second is based on minimizing the spectral condition number of the closed–loop eigenvector matrix. The CG methods are extended to tackle the output feedback PAP for decentralized control systems. Global convergence is established for the three CG methods under a nonmonotonic backtracking strategy. Moreover, some preliminary numerical tests arising from output feedback pole assignment are presented and demonstrated the performance of the proposed methods.

Our future work is concentrated on the following directions not only solving output feedback PAP but also improving the current implementation:

- smooth the proposed CG methods for output feedback PAP and its variants;
- use spectral conjugate gradient methods based on a modified secant equation [29] for output feedback PAP and study the convergence properties of these methods using different inexact line searches [14], [40], [46];
- design a conjugate gradient method that is suitable to solve ill-conditioned minimization problems (the Hessian of objective functions at a stationary point is ill-conditioned). Can we integrate trust region technique to the conjugate gradient methods for output feedback PAP and its variants?;
- apply three-term conjugate gradient algorithms [2] for output feedback PAP and its variants.

Acknowledgments

We are grateful to the anonymous reviewers for constructive feedback and insightful suggestions which greatly improved this article.

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] N. Andrei, On three-term conjugate gradient algorithms for unconstrained optimization, *Applied Math. Comp.* 219 (2013) 6316–6327.

[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] J. Baranowski, State estimation in linear multi-output systems-design-Design example and discussion of optimality, *Automatica* 10 (2006) 119–131.

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

[6] L. Carotenuto, C. Franze and P. Muraca, New computational procedure to the pole placement problem by static output feedback, *Inst. Elect. Eng. Proc. Control Theory Appl.* 148 (2001) 466–471.

[7] F.H. Clarke, *Optimization and Nonsmooth Analysis*, John Wiley, New York, 1983.

[8] A. Eremenko and A. Gabrielov, Pole placement by static output feedback for generic linear systems, *SIAM J. Control Optim.* 41 (2002) 303–312.

[9] Y. A. Fiagbedzi, Memoryless reconstruction of an extended state of a control delayed system, *System & Control letters* 59 (2010) 596–600.

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

[11] Z. Gong, Decentralized robust control of uncertain interconnected systems with prescribed degree of exponential convergence, *IEEE Transactions on Automatic Control* 40 (1995) 704–707.

[12] Q.P. Ha and H. Trinh , Observer-based control of multi-agent systems under decentralized information structure, *Int. J. Sys. Sci.* 35 (2004) 719–728.

[13] W.W. Hager and H. Zhang, A survey of nonlinear conjugate gradient methods, *Pacific J. Optim.* 1 (2006) 35–58.

[14] J. Han, J. Sun and W. Sun, Global convergence of non-monotone descent methods for unconstrained optimization problems, *J. Comp. Applied Math.* 146 (2002) 89–98.

[15] U. Helmke and J.B. Moore, *Optimization and Dynamical Systems*, Springer-Verlage, London, 1994.

[16] R.A. Horn and C.A. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[17] M. Ikeda, Decentralized control of large scale systems, in *Three Decades of Mathematical System Theory, Lecture Notes in Control and Inform. Sci.*, vol. 135, Springer, Berlin, 1989, pp. 219–242.

[18] T. Kato, *Perturbation Theory for Linear Operators*, Springer, Berlin, 1995.

[19] J. Kautsky, N.K. Nichols, P. Van Dooren, Robust pole assignment in linear feedback, *Int. J. Control* 41 (1985) 1129–1155.

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

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

[22] H. Kwakernaak and R. Sivan, *Linear Optimal Control Systems*, Wiley Interscience, 1972.

[23] J. Lam and W.Y. Yan, A pole assignment with optimally conditioned eigenstructure, *Conference on Decision and Control* 2 (1996) 2008–2013.

[24] F. Leibfritz, COMPlib: COnstraint Matrix-optimization Problem library—a collection of test examples for nonlinear semi-definite programs, control system design and related problems, Technical Report 2004.

[25] T. Liu, Z.P. Jiang and D.J. Hill, Decentralized output-feedback control of large-scale nonlinear systems with sensor noise, *Automatica J. IFAC* 48 (2012) 2560–2568.

[26] W.Q. Liu and V. Sreeram, New algorithm for computing LQ suboptimal output feedback gains of decentralized control systems, *J. Optim. Theory Appl.* 93 (1997) 597–607.

[27] P.L. Liu and T.J. Su, Robust stability of interval time-delay systems with delay-dependence, *System & Control letters* 33 (1998) 231–239.

[28] J.K. Liu, L.M. Zou and X.Q. Song, Global convergence of a nonlinear conjugate gradient method, *Math. Problems in Engineering*, doi:10.1155/2011/463087.

[29] I.E. Livieris and P. Pintelas, A new class of spectral conjugate gradient methods based on a modified secant equation for unconstrained optimization, *J. Comp. Applied Math.* 239 (2013) 396–405.

[30] R.E. Mahony and U. Helmke, System assignment and pole placement for symmetric realisations, *J. Math. Systems, Estimation Control* 5 (1995) 1–32.

[31] W. Michiels, K. Engelborghs, P. Vansevenant and D. Roose, Continuous pole placement for delay equations, *Automatica* 38 (2002) 747–761.

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

[33] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operations Research, Springer, New York, Berlin, 1999.

[34] R.V. Patel, On output feedback pole sssignability, *Int. J. Control* 20 (1974) 955–959.

[35] L. Qi, Convergence analysis of some algorithms for solving nonsmooth equations, *Math. Oper. Research* 18 (1993) 227–244.

[36] M. Razzaghi, A pole assignment technique for multivariable systems with input delay, *Kybernetika* 22 (1986) 368–371.

[37] J. Rosenthal and F. Sotille, Some remakes on real and complex output feedback, *System Control Letter* 33 (1998) 73–80.

[38] M. Saif and Y. Guan, Decentralized state estimation in large-scale interconnected dynamical systems, *Automatica* 28 (1992) 215–219.

[39] C. Seatzu, Decentralized controllers design for open-channel hydraulic systems via eigenstructure assignment, *Applied Mathe. Modeling* 24 (2000) 915–930.

[40] Z.J. Shi and J. Guo, A new family of conjugate gradient methods, *J. Computational and Applied Mathematics* 224 (2009) 444–457.

[41] D.D. Siljak, *Decentralized Control of Complex Systems*, Mathematics in Science and Engineering, 184, Academic Press, Inc., Boston, MA, 1991.

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

[43] B. Sridhar and D.P. Lindorf, Pole placement with constraint gain output feedback, *Int. J. Control* 18 (1973) 993–1003.

[44] J.G. Sun, Eigenvalues and eigenvectors of a matrix dependent on several parameters, *J. Comp. Math.* 3 (1985) 351–364.

[45] D. Sun and J. Han, Newton and quasi-Newton for a class of nonsmooth equations and related problems, *SIAM J. Optim.* 7 (1997) 463–480.

[46] W. Sun, J. Han and J. Sun, Global convergence of nonmonotone descent methods for unconstrained optimization problems, *J. Comp. Applied Math.* 146 (2002) 89–98.

[47] D. Sun and J. Sun, Strong semismoothness of eigenvalues of symmetric matrices and its application to inverse eigenvalue problems, *SIAM J. Numer. Anal.* 40 (2003) 2352–2367.

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

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

[50] M. Tarokh, A unified approach to exact,approximate, optimized and decentralized output feedback pole assignment, *Int. J. Control, Autom. Sys.* 6 (2008) 939–947.

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

[52] S.F. Xu, *An introduction to Inverse Algebraic Eigenvalue Problems*, Peking University Press, Beijing, China, 1998.

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

[54] G. Yu, Y. Zhao and Z. Wei, A descent nonlinear conjugate gradient method for large-scale unconstrained optimization, *Applied Math. Comp.* 187 (2007) 636–643.

Manuscript received 20 February 2015
revised 9 July 2015
accepted for publication 12 July 2015

El-Sayed M.E. Mostafa
Department of Mathematics and Computer Science
Faculty of Science, Alexandria University
Moharam Bek 21511, Alexandria, Egypt
E-mail address: emostafa@alex-sci.edu.eg

Mohamed A. Tawhid
Department of Mathematics and Statistics, Faculty of Science
Thompson Rivers University, Kamloops, BC, Canada V2C 0C8

Department of Mathematics and Computer Science
Faculty of Science, Alexandria University
Moharam Bek 21511, Alexandria, Egypt
E-mail address: Mtawhid@tru.ca

Eman R. Elwan
Department of Mathematics, Faculty of Education
Alexandria University, Alexandria, Egypt
E-mail address: zmmsoe28@gmail.com

Table 3: Performance of the methods NCG, VLS+CG and MPRP-CG on test problems from the benchmark collection [24] for two different choices of the desired poles $\hat{\lambda}$.

Problem	Problem size			$\vartheta(A)$	No. of iterations and CPU-time (sec.)					
					NCG		VLS+CG		MPRP-CG	
	n	p	r		# it.	CPU	# it.	CPU	# it.	CPU
AC1	5	3	3	0	489	12.42	53	2	216	7.59
					641	13.59	114	2.58	891	24.27
AC2	5	3	3	−9.2e−003	489	12.33	53	2.02	216	7.63
					641	13.47	114	2.55	891	24.33
AC3	5	2	4	−7.9e−003	49	0.56	30	0.58	49	0.42
					22	0.28	14	0.2	22	0.25
AC12	4	3	4	5.8e−001	185	7.25	131	9.74	226	7.05
					756	22.84	301	8.25	388	11.8
AC15	4	2	3	−1.1e−002	29	0.28	26	0.41	24	0.34
					40	0.77	18	0.31	30	0.45
AC16	4	2	4	−1.1e−002	17	0.28	15	0.41	26	0.39
					36	0.39	25	0.53	36	0.66
HE2	4	2	2	−2.9e−002	94	2.53	84	3.34	173	3.16
					259	6.69	429	13.72	196	6.44
HE3	8	4	6	8.71−002	52	0.92	67	1.83	42	0.74
					101	1.72	244	11.55	239	9.92
HE4	8	4	6	2.3e−001	375	20.08	160	6.89	107	2.91
					159	4.14	190	6.55	73	2.7
REA1	4	2	3	2.0	68	1.66	150	4.63	53	1.06
					89	1.55	44	0.91	50	1.47
REA2	4	2	2	2.0	26	0.38	82	1.41	21	0.49
					37	1	78	1.67	56	0.73
DIS1	8	4	4	−8.8e−002	124	20.5	77	6.94	256	13.13
					799	38.66	588	37.52	152	27.16
DIS2	3	2	3	1.7e+000	12	0.11	14	0.19	19	0.19
					20	0.33	16	0.16	16	0.24
DIS4	6	4	6	1.4e+000	37	1.07	37	1.53	30	0.86
					36	1.19	41	1.41	44	1.55
NN2	2	1	1	0	9	0.39	9	0.39	9	0.38
					–	–	–	–	–	–
NN4	4	2	3	−4.1e−002	30	0.42	48	0.72	23	0.41
					17	0.33	17	0.42	14	0.27
NN8	3	2	2	−2.9e−002	11	0.17	8	0.22	22	0.3
					23	0.23	14	0.3	14	0.31
NN16	8	4	4	0	104	38.44	–	–	–	–
					–	–	–	–	–	–
HF2D10	5	2	3	1.3e−001	53	0.97	61	1.52	166	1.95
					122	2.05	68	1.67	61	1.22
HF2D11	5	2	3	2.5e−001	62	0.7	53	0.75	66	0.72
					119	0.97	89	1.31	108	1.17
HF2D12	5	2	4	−1.7e−001	180	2.52	40	1.14	160	1.75
					161	2.97	31	1	86	2.69
HF2D13	5	2	4	−2.5e−001	165	7.61	168	8.95	151	7.72
					151	7.13	156	7.74	108	7.81
HF2D14	5	2	4	2.2e−001	110	7.58	–	–	–	–
					70	3.75	–	–	–	–
HF2D15	5	2	4	1.6e+000	–	–	286	16.53	–	–
					466	33.86	–	–	–	–
HF2D17	5	2	4	5.4e−001	330	16.28	–	–	166	6.39
					138	5.8	–	–	62	3.27

Table 4: Comparison between the achieved final eigenvalues of Example 9.5.

$\tilde{\lambda}$	-0.3000	-0.8312	-0.8312	-1.4943	-1.8972
$\lambda^{NCG}(A_c)$	-0.2978	-0.8326	-0.8326	-1.4982	-1.8888
$\lambda^{VLS+CG}(A_c)$	-0.2961	-0.8296	-0.8296	-1.4983	-1.8931
$\lambda^{MPRP-CG}(A_c)$	-0.2994	-0.8299	-0.8299	-1.4983	-1.8901

Table 5: State feedback PAP: Performance of the three CG methods for the two cases $s = 0.1,\ 0.3$

	NCG		VLS+CG		MPRP-CG	
	$s = 0.1$	$s = 0.3$	$s = 0.1$	$s = 0.3$	$s = 0.1$	$s = 0.3$
Av. No. of iterations	13.66	17.17	11.24	12.83	9.66	13.59
Av. CPU time (sec.)	5.17	3.29	2.39	2.54	3.41	2.64

Table 6: State feedback PAP: Performance of the methods NCG, VLS+CG and MPRP-CG on test problems from different sources for two different choices of $\tilde{\Lambda}$.

				No. of iterations and CPU-time (sec.)					
	Problem size			NCG		VLS+CG		MPRP-CG	
Problem	n	p	$\vartheta(A)$	# it.	CPU	# it.	CPU	# it.	CPU
[43]	4	3	2	11	0.11	12	0.11	6	0.08
				12	0.11	15	0.09	12	0.09
[43]	4	2	2	8	0.06	6	0.09	4	0.09
				3	0.08	5	0.09	4	0.05
[43]	4	3	1	4	0.19	26	0.52	5	0.14
				4	0.13	6	0.16	4	0.11
[49]	5	2	54.52e−001	6	0.13	4	0.14	4	0.09
				6	0.34	7	0.11	9	0.38
[22]	2	2	−1.0e−002	6	0.11	11	0.23	7	0.16
				3	0.06	2	0.08	4	0.09
[22]	4	2	95.12e−002	7	0.25	6	0.22	7	0.24
				5	0.17	6	0.16	3	0.09
[19]	4	2	1.9e+000	17	0.48	6	0.19	10	0.3
				6	0.16	8	0.2	5	0.17
[34]	3	2	3	20	0.23	16	0.23	23	0.2
				10	0.11	21	0.19	8	0.13
[39]	2	2	−0.1e−002	7	0.16	7	0.13	6	0.14
				3	0.09	7	0.08	4	0.08
[4]	3	2	− 58.58e−002	4	0.11	5	0.11	4	0.09
				5	0.05	4	0.11	4	0.08
[51]	3	1	− 1.29e−001	22	1.16	17	0.91	21	1.09
				25	1.2	39	2	23	0.84

Table 7: State feedback PAP: Performance of the methods NCG, VLS+CG and MPRP-CG on test problems from the benchmark collection [24] for two different choices of $\tilde{\Lambda}$.

Problem	Problem size n	p	$\vartheta(A)$	No. of iterations and CPU-time (sec.) NCG # it.	NCG CPU	VLS+CG # it.	VLS+CG CPU	MPRP-CG # it.	MPRP-CG CPU
AC4	4	1	25.79e-001	14	0.5	11	0.39	14	0.47
				16	0.47	13	0.38	10	0.28
AC11	4	2	54.52e−001	7	0.17	7	0.19	6	0.25
				5	0.31	9	0.24	4	0.28
REA1	4	2	2.0	43	1.08	29	0.73	13	0.3
				2	0.16	3	0.16	3	0.14
REA2	4	2	2.0	7	0.27	4	0.28	5	0.24
				8	0.22	5	0.14	5	0.2
HF1	130	1	−1.9e−002	30	138.2	12	58.27	13	89.64
				17	79.92	12	60.75	11	64.11
IH	21	11	0	29	2.23	13	1.66	13	1.39
				18	1.11	10	1.5	10	0.59
NN1	3	1	36.06e−001	13	0.2	6	0.14	11	0.28
				8	0.09	6	0.11	8	0.08
NN8	3	2	−2.9e−002	7	0.17	5	0.17	1	0.11
				7	0.17	6	0.14	7	0.16
NN17	3	2	1.17e+000	6	0.09	8	0.16	9	0.11
				3	0.09	5	0.13	2	0.11
HF2D10	5	2	1.3e−001	21	0.7	23	0.84	12	0.39
				35	0.99	25	0.81	27	0.92
HF2D11	5	2	2.5e−001	18	0.5	11	0.36	15	0.44
				11	0.33	15	0.47	13	0.41
HF2D12	5	2	−1.7e−001	18	0.69	17	0.78	18	0.83
				24	0.86	12	0.47	24	1
HF2D13	5	2	−2.5e−001	12	0.41	12	0.44	12	0.39
				17	0.58	22	0.91	16	0.53
HF2D14	5	2	2.2e−001	10	0.36	11	0.45	10	0.38
				113	3.58	34	1.14	104	3.53
HF2D15	5	2	1.6e+000	25	0.53	21	0.59	12	0.3
				12	0.27	8	0.28	12	0.34
HF2D16	5	2	11.35e−002	5	0.25	4	0.22	5	0.22
				26	0.77	20	0.53	16	0.74
HF2D17	5	2	5.4e−001	10	0.38	10	0.38	7	0.22
				87	2.52	20	0.61	39	1.13
HF2D18	5	2	28.08e−002	9	0.34	6	0.25	7	0.19
				7	0.48	27	1.72	3	0.27