Journal of Computational Mathematics
Vol.30, No.3, 2012, 279–297.

http://www.global-sci.org/jcm
doi:10.4208/jcm.1109-m3364

A CONJUGATE GRADIENT METHOD FOR DISCRETE–TIME OUTPUT FEEDBACK CONTROL DESIGN*

El-Sayed M.E. Mostafa
Department of Mathematics, Faculty of Science, Alexandria University, Alexandria, Egypt
Email: emostafa@alex-sci.edu.eg

Abstract

In this paper, the discrete–time static output feedback control design problem is considered. A nonlinear conjugate gradient method is analyzed and studied for solving an unconstrained matrix optimization problem that results from this optimal control problem. In addition, through certain parametrization to the optimization problem an initial stabilizing static output feedback gain matrix is not required to start the conjugate gradient method. Finally, the proposed algorithms are tested numerically through several test problems from the benchmark collection.

Mathematics subject classification: 49N35, 49N10, 93D52, 93D22, 65K05.
Key words: Output feedback control, Nonlinear conjugate gradient methods, Nonlinear programming.

1. Introduction

The static output feedback problem (SOF) for discrete or continuous-time control systems is one of the most studied problems, where wide area of applications in engineering and in finance are represented by this problem; see the two surveys [12, 21] and the references therein. Particularly, many special purpose methods are designed by the engineers for solving this problem; see [12, 21].

Various gradient-based methods are available for solving the SOF problem among them is the descent Anderson–Moore method [12] that solves the SOF problem by successfully minimizing particular quadratic approximation of the objective function combined with step-size rule. Mäkilä and Toivonen [12] solves the discrete problem by Newton's method with line search globalization. Rautert and Sachs [20] suggest quasi-Newton method with line search for solving the continuous-time SOF problem. Mostafa [16] introduces trust region method for solving the discrete-time SOF problem.

Levine–Athans method [12] is among the classical techniques for solving this problem. In this method a stationary point of the optimization problem is obtained by solving the system of the necessary optimality conditions under certain assumptions on the constant matrices of the problem. It has been reported that this method is computationally expensive and lacks of convergence properties.

All these methods are based on reformulating the discrete or continuous-time SOF problems into unconstrained matrix optimization problems. The formulation of the SOF problem as a constrained optimization problem allows utilizing numerous available constrained optimization techniques. Leibfritz and Mostafa [9] formulate the SOF problem as a nonlinear

* Received March 22, 2010 / Revised version received May 25, 2011 / Accepted September 1, 2011 / Published online May 7, 2012 /

semi-definite programming problem and suggest for solving this problem an interior-point trust region method. Moreover, they suggest in [10] unconstrained and constrained trust region approaches for solving two formulations of the SOF problem. Kočvara et al. [7] consider the constrained formulation of the SOF problem and introduce an augmented Lagrangian semi-definite programming method. Mostafa [14] and [15] suggests a trust region method for solving the decentralized SOF problem and an augmented Lagrangian SQP method for solving a special class of nonlinear semi-definite programming problem related to the SOF problem, respectively.

In this paper, a nonlinear Conjugate Gradient (CG) method is analyzed and studied for solving the discrete-time SOF problem, which can be written as unconstrained optimization problem of the following form (see, e.g., the two surveys [12, 21]):

$$\min_{F \in \mathcal{S}_F} \quad J(F) = \mathrm{Tr}\,(P(F)Q(F)), \tag{1.1}$$

where the variable $P(F)$ is a matrix that solves the following discrete Lyapunov equation:

$$P(F) = A(F)P(F)A(F)^T + V, \tag{1.2}$$

$Q(F) = Q + C^T F^T RFC$, $A(F) = A + BFC$, and $\mathrm{Tr}\,(\cdot)$ is the trace operator. The variable F is a matrix that must be chosen from the following set of stabilizing output feedback controllers:

$$\mathcal{S}_F = \Big\{ F \in \mathbb{R}^{n_u \times n_y} \ : \ \rho\,(A + BFC) < 1 \ \Big\}, \tag{1.3}$$

where $\rho\,(\cdot)$ is the spectral radius. Moreover, A, B, C, Q, R, and V are given constant matrices of appropriate dimensions, which are defined and explained in Section 2. Problem (1.1)–(1.2) is an unconstrained optimization problem in the matrix variable F, where the eigenvalue condition $F \in \mathcal{S}_F$ will be fulfilled within the considered CG method.

Note, that the set $\mathcal{S}_F$ is open and in general unbounded. Therefore, it is convenient to define the following level set:

$$\mathcal{L}(F_0) = \{F \in \mathcal{S}_F \ : \ J(F) \le J(F_0)\}\,. \tag{1.4}$$

This level set is compact; see [12, Appendix A]. For given $F_0 \in \mathcal{S}_F$ the theorem of Bolzano–Weierstrass ensures the existence of a unique solution to the optimization problem (1.1)–(1.2) in the level set $\mathcal{L}(F_0)$; see [12].

The CG method was proposed by Hestenes and Stiefel [6] early in 1952 for solving linear systems of algebraic equations. Fletcher and Reeves [5] in 1964 developed a CG method for solving unconstrained optimization problems. Moreover, many different CG methods have been proposed in recent years (see, e.g., [1, 3, 4, 11, 18] and the references therein).

There are many large and medium-scale applications in the literature of output feedback control design where higher order optimization methods fail to solve; see, e.g., the benchmark collection [8] for various engineering applications. The attempt in this paper is to apply a modified Dai-Yuan nonlinear CG method which belongs to the class of low storage methods for solving the problem (1.1)–(1.2). Moreover, the convergence theory given in [1] is extended to the considered algorithm.

The existence of an initial stabilizing SOF gain matrix $F_0 \in \mathcal{S}_F$ is one of the main obstacles that typically faces numerical methods that solve this problem class. By parameterizing the optimization problem the resulting CG method does not require initial $F_0 \in \mathcal{S}_F$ to start the iteration sequence. The modified algorithm is denoted by CG2.

This paper is organized as follows. In the next section we state the discrete-time static output feedback control design problem. Section 3 presents the nonlinear CG method for solving the optimization problem (1.1)–(1.2). Section 4 contains the convergence analysis of the CG method. In Section 5 the optimization problem (1.1)–(1.2) is parameterized in a way an initial stabilizing SOF gain matrix $F_0 \in \mathcal{S}_F$ is not required. In Section 6 the proposed algorithms are tested numerically on several test problems from the benchmark collection COMPlib [8]. Then we end by a conclusion.

Notations: Throughout the paper $\|\cdot\|$ denotes the Frobenius norm given by $\|M\| = \sqrt{\langle M, M\rangle}$, where $\langle \cdot, \cdot \rangle$ is the inner product defined by

$$\langle M_1, M_2\rangle = \mathrm{Tr}\,(M_1^T M_2) \quad \text{for} \;\; M_1, M_2 \in \mathbb{R}^{n\times n}$$

and $\mathrm{Tr}\,(\cdot)$ is the trace operator.

2. Application: Discrete Output Feedback Controller Design

The focus in this work is to solve the discrete-time SOF control design problem (see the above mentioned citations), where the goal is to find an optimal discrete stabilizing SOF gain matrix $F \in \mathbb{R}^{n_u \times n_y}$ according to the control law $u_k = F\,y_k$ that minimizes the quadratic index function:

$$J_{x_0}(F) = \mathbb{E}\Big[\sum_{k=0}^{\infty} \big(x_k^T Q x_k + u_k^T R u_k\big)\Big] \tag{2.1}$$

subject to the linear time-invariant control system

$$x_{k+1} = Ax_k + Bu_k, \quad y_k = Cx_k. \tag{2.2}$$

where $\mathbb{E}\,[\cdot]$ is the expected value; $x_k \in \mathbb{R}^{n_x}, u_k \in \mathbb{R}^{n_u}$ and $y_k \in \mathbb{R}^{n_y}$ are the state, the control input and the measured output vectors, respectively; $A \in \mathbb{R}^{n_x\times n_x}, B \in \mathbb{R}^{n_x\times n_u}, C \in \mathbb{R}^{n_y\times n_x}$ are given constant matrices, and $Q \in \mathbb{R}^{n_x\times n_x}, R \in \mathbb{R}^{n_u\times n_u}$ are given symmetric and positive definite weight matrices.

Using the control law $u_k = F\,y_k = FCx_k$ the associated closed-loop system is written as:

$$x_k = (A + BFC)x_{k-1} = A(F)\,x_{k-1} = A(F)^k x_0, \tag{2.3}$$

Obviously this restricts F to lie within the set $\mathcal{S}_F$ so that all state variables decay to the zero state as $k \to \infty$, namely $\lim\limits_{k\to\infty} x_k = 0$.

By substituting (2.3) into (2.1) we obtain

$$\begin{aligned} J_{x_0}(F) &= \mathbb{E}\Big[x_0^T \sum_{k=0}^{\infty} \big[(A(F)^k)^T (Q + C^T F^T RFC) A(F)^k\big] x_0\Big] \\ &= \mathbb{E}\big[x_0^T K(F) x_0\big] \\ &= Tr\,(K(F)V), \end{aligned}$$

where $V = \mathbb{E}\,[x_0 x_0^T]$ is the covariance matrix and

$$K(F) = \sum_{k=0}^{\infty} (A(F)^k)^T (Q + C^T F^T RFC) A(F)^k$$

solves the discrete Lypaunov equation:

$$K(F) = A(F)^T K(F)A(F) + (Q + C^T F^T RFC). \tag{2.4}$$

The objective function depends on $x_0 \in \mathbb{R}^{n_x}$. To remove this dependency it is assumed that x_0 is a random variable satisfying $\mathbb{E}[x_0] = 0$. In addition, the constant covariance matrix V is often chosen as the identity or any symmetric and positive definite matrix.

From (1.2) and (2.4) and using the trace properties it holds that:

$$Tr\,(K(F)V) = Tr\,(P(F)Q(F)),$$

which is the objective function (1.1).

3. A CG Method for Computing Discrete SOF Controllers

First, let us obtain the gradient of the objective function required in the derivation of the proposed CG method. The next lemma provides a discrete Lyapunov equation required for obtaining $\nabla J(F)$.

Lemma 3.1. *Let $F \in \mathcal{S}_F$. Then $P(F)$ defined by* (1.2) *is differentiable and its directional derivative $\Delta P(\Delta F)$ is given by the discrete Lyapunov equation:*

$$\Delta P(\Delta F) = A(F)\Delta P(\Delta F)A(F)^T + A(F)PC^T\Delta F^T B^T + B\Delta FCPA(F)^T. \tag{3.1}$$

Proof. Let us define

$$\Phi(P, F) = -P + A(F)PA(F)^T + V.$$

The directional derivatives of Φ are given by

$$\Phi_P(P, F)\Delta P = -\Delta P + A(F)\Delta PA(F)^T$$

$$\Phi_F(P, F)\Delta F = A(F)PC^T\Delta F^T B^T + B\Delta FCPA(F)^T.$$

Since F is chosen from the open set $\mathcal{S}_F$ and $\Phi_P(P, F)$ is surjective, then the implicit function theorem implies that $P(F)$ is differentiable. Obviously, the total derivative of Φ satisfies:

$$\Phi_P(P, F)\Delta P + \Phi_F(P, F)\Delta F = 0,$$

which gives (3.1). □

The next lemma gives the first-order directional derivative of the objective function $J(F)$. For any two matrices M_1 and M_2 of appropriate dimensions the following properties of the trace operator are used:

$$Tr\,(M_1M_2) = Tr\,(M_2M_1), \qquad Tr\,(M_1^TM_2) = Tr\,(M_2^TM_1).$$

Theorem 3.1. *Let $F \in \mathcal{S}_F$. The first-order directional derivative of $J(F)$ in the direction of ΔF is given by*

$$J_F(F)\Delta F = 2\,Tr\,\big((B^T K(F)A(F) + RFC)P(F)C^T\Delta F^T\big), \tag{3.2}$$

where $P(F)$ and $K(F)$ solve the discrete Lyapunov equations (1.2) *and* (2.4)*, respectively.*

Proof. By differentiating (1.1) with respect to F in the direction of ΔF gives:

$$\begin{aligned} J_F(F)\Delta F &= Tr(Q(F)\Delta P(\Delta F)) + Tr((C^T\Delta F^T RFC + C^T F^T R\Delta FC)P(F)) \\ &= Tr(Q(F)\Delta P(\Delta F)) + 2Tr(C^T\Delta F^T RFCP(F)). \end{aligned}$$

On the other side, the discrete Lyapunov equations (2.4) and (3.1) imply that

$$\begin{aligned} Tr(Q(F)\Delta P(\Delta F)) &= Tr\Big(K(F)\big(A(F)PC^T\Delta F^T B^T + B\Delta FCPA(F)^T\big)\Big) \\ &= 2\,Tr\,(B^T K(F)A(F)P(F)C^T\Delta F^T). \end{aligned}$$

Hence,

$$\begin{aligned} J_F(F)\Delta F &= 2\,Tr\,(B^T K(F)A(F)P(F)C^T\Delta F^T) + 2\,Tr\,(C^T\Delta F^T RFCP(F)) \\ &= 2\,Tr\,\big((B^T K(F)A(F) + RFC)P(F)C^T\Delta F^T\big). \end{aligned}$$

$\square$

In order to have $\nabla J(F)$ explicitly let us write the directional derivative $J_F(F)\Delta F$ in a gradient form as:

$$J_F(F)\Delta F = Tr\,(\nabla J(F)\Delta F^T), \quad \text{where} \quad \Delta F \in \mathbb{R}^{n_u\times n_y}.$$

Hence, (3.2) gives:

$$\nabla J(F) = 2\,(B^T K(F)A(F) + RFC)P(F)C^T. \tag{3.3}$$

The above theorem yields the first-order necessary optimality conditions for the optimization problem (1.1)–(1.2).

Lemma 3.2. *If $F \in \mathcal{S}_F$ is a solution to the optimization problem* (1.1)–(1.2)*, then*

$$(B^T K(F)A(F) + RFC)P(F)C^T = 0, \tag{3.4}$$

where $P(F)$ and $K(F)$ solve the discrete Lyapunov equations (1.2) *and* (2.4)*, respectively.*

Given $F_0 \in \mathcal{S}_F$, the nonlinear CG method for solving the minimization problem (1.1)–(1.2) generates a sequence of iterates of the form:

$$F_{k+1} = F_k + \alpha_k\Delta F_k \in \mathcal{S}_F, \qquad k = 0, 1, 2, \ldots, \tag{3.5}$$

where ΔF_k is a descent direction for J at F_k and α_k is the step size determined by the Wolfe line search rule. The new search direction ΔF_{k+1} is obtained by using a modified Dai-Yuan nonlinear CG method [1]:

$$\Delta F_{k+1} = -\theta_{k+1}\nabla J(F_{k+1}) + \beta_k\,S_k, \qquad \Delta F_0 = -\nabla J(F_0), \tag{3.6}$$

where

$$\theta_{k+1} = \frac{\|\nabla J(F_{k+1})\|^2}{Tr(Y_k^T\nabla J(F_{k+1}))}, \tag{3.7}$$

$$\beta_k = \frac{1}{Tr(Y_k^T S_k)}\,Tr\left[\left(\nabla J(F_{k+1}) - \delta_k\frac{\|\nabla J_{k+1}\|^2}{Tr(Y_k^T S_k)}S_k\right)^T\nabla J(F_{k+1})\right], \tag{3.8}$$

$$\delta_k = \frac{Tr(Y_k^T\nabla J(F_{k+1}))}{\|\nabla J(F_{k+1})\|^2} = \frac{1}{\theta_{k+1}}, \tag{3.9}$$

where $Y_k = \nabla J(F_{k+1}) - \nabla J(F_k)$ and $S_k = F_{k+1} - F_k$.

Using the weak Wolfe line search rule the CG method (3.6)–(3.9) generates a descent direction. The parameter β_k is chosen such that the sufficient descent condition is satisfied every iteration. Moreover, θ_{k+1} and δ_k are evaluated in such away that the following conjugancy condition holds:

$$Tr\,(Y_k^T \Delta F_{k+1}) = 0.$$

The following theorem shows that the search direction ΔF_{k+1} evaluated by (3.6)–(3.9) is a descent direction to $J(F)$.

Theorem 3.2. *If* $Tr\,(Y_k^T S_k) \neq 0$ *and* ΔF_{k+1} *is evaluated using* (3.6)–(3.9) *such that* $F_{k+1} \in \mathcal{S}_F$*, then*

$$Tr\,(\nabla J(F_{k+1})^T \Delta F_{k+1}) \leq -(\theta_{k+1} - \frac{1}{4\delta_k})\,\|\nabla J(F_{k+1})\|^2. \tag{3.10}$$

Proof. (see also [1, Theorem 1]) Since $\Delta F_0 = -\nabla J(F_0)$, then

$$Tr\,(\nabla J(F_0)^T \Delta F_0) = -\|\nabla J(F_0)\|^2,$$

which satisfies (3.10). Next, pre-multiplying (3.6) by $\nabla J(F_{k+1})^T$ and applying the trace operator on both sides gives:

$$\begin{aligned}
&Tr\,(\nabla J(F_{k+1})^T \Delta F_{k+1})\\
&= -\,\theta_{k+1}\, Tr\,(\nabla J(F_{k+1})^T \nabla J(F_{k+1})) + \beta_k\, Tr\,(\nabla J(F_{k+1})^T S_k)\\
&= -\,\theta_{k+1}\,\|\nabla J(F_{k+1})\|^2 + \beta_k\, Tr\,(\nabla J(F_{k+1})^T S_k)\\
&= -\,\theta_{k+1}\,\|\nabla J(F_{k+1})\|^2 + \frac{\|\nabla J(F_{k+1})\|^2}{Tr\,(Y_k^T S_k)}\, Tr\,(\nabla J(F_{k+1})^T S_k)\\
&\qquad -\,\delta_k \|\nabla J(F_{k+1})\|^2 \frac{\big(Tr\,(\nabla J(F_{k+1})^T S_k)\big)^2}{\big(Tr\,(Y_k^T S_k)\big)^2}.
\end{aligned} \tag{3.11}$$

Using the fact that for any two matrices M_1 and M_2 of appropriate dimensions

$$Tr\,(M_1\, M_2) \leq \frac{1}{2}(\|M_1\|^2 + \|M_2\|^2)$$

then, it holds for the second term on the right-hand-side of (3.11) that:

$$\begin{aligned}
&\frac{\|\nabla J_{k+1}\|^2\, Tr(\nabla J_{k+1}^T S_k)}{Tr(Y_k^T S_k)}\\
=&\frac{Tr\Big[\Big(Tr(Y_k^T S_k)\,\nabla J_{k+1}/\sqrt{2\delta_k}\Big)^T \Big(\sqrt{2\delta_k}\ Tr(\nabla J_{k+1}^T S_k)\ \nabla J_{k+1}\Big)\Big]}{\big(Tr(Y_k^T S_k)\big)^2}\\
\leq&\frac{1}{2\big(Tr\,(Y_k^T S_k)\big)^2}\left[\frac{1}{2\delta_k}\big(Tr\,(Y_k^T S_k)\big)^2 \|\nabla J_{k+1}\|^2 + 2\delta_k \big(Tr(\nabla J_{k+1}^T S_k)\big)^2 \|\nabla J_{k+1}\|^2\right]\\
=&\frac{1}{4\delta_k}\,\|\nabla J_{k+1}\|^2 + \delta_k \frac{\big(Tr(\nabla J_{k+1}^T S_k)\big)^2}{\big(Tr(Y_k^T S_k)\big)^2} \|\nabla J_{k+1}\|^2.
\end{aligned} \tag{3.12}$$

From (3.12) in (3.11) one obtains:

$$\begin{aligned}
&Tr\,(\nabla J(F_{k+1})^T \Delta F_{k+1})\\
&\leq -\,\theta_k \|\nabla J(F_{k+1})\|^2 + \frac{1}{4\delta_k}\|\nabla J(F_{k+1})\|^2 = -\big(\theta_k - \frac{1}{4\delta_k}\big)\|\nabla J(F_{k+1})\|^2.
\end{aligned}$$

□

The calculated search direction ΔF_{k+1} by the CG method (3.6)–(3.9) must satisfy the following weak Wolfe conditions:

$$J(F_k + \alpha_k \Delta F_k) - J(F_k) \leq \tilde{\sigma}_1 \alpha_k \, Tr(\nabla J(F_k)^T \Delta F_k) \tag{3.13}$$
$$Tr(\nabla J(F_k + \alpha_k \Delta F_k)^T \Delta F_k) \geq \tilde{\sigma}_2 \, Tr(\nabla J(F_k)^T \Delta F_k), \tag{3.14}$$

where $0 < \tilde{\sigma}_1 < \tilde{\sigma}_2 < 1$ are parameters.

Algorithm 3.1. (CG1: Computing discrete stabilizing SOF controllers)

Let $F_0 \in \mathcal{S}_F$ be given and let $\epsilon^{\text{tol}} \in (0,1)$ be the given tolerance. Choose $0 < \tilde{\sigma}_1 < \tilde{\sigma}_2 < 1$ and calculate $P(F_0)$ and $K(F_0)$ solutions of the discrete Lyapunov ehquations (1.2) and (2.4), respectively. Calculate $\|\nabla J(F_0)\|$; Set $\Delta F_0 := -\nabla J(F_0)$ and $k \leftarrow 0$.
While $\|\nabla J(F_k)\| \geq \epsilon^{\text{tol}}$, do

1. Calculate the first element α_k of a decreasing sequence, e.g., $\{1/2^j\}_{j\geq 0}$ that satisfies the weak Wolfe conditions (3.13)–(3.14) and the stability condition (3.5), i.e., $F_k + \alpha_k \Delta F_k \in \mathcal{S}_F$.
2. Set $F_{k+1} := F_k + \alpha_k \Delta F_k$.
3. Given F_{k+1} solve the discrete Lyapunov equations (1.2) and (2.4) for $P(F_{k+1})$ and $K(F_{k+1})$, respectively. Then calculate $\nabla J(F_{k+1})$; Set $Y_k := \nabla J(F_{k+1}) - \nabla J(F_k)$ and $S_k := F_{k+1} - F_k$; Calculate θ_{k+1}, β_k, and δ_k using (3.7)–(3.9).
4. Calculate:
$$\Delta F = -\theta_{k+1} \nabla J(F_{k+1}) + \beta_k S_k,$$
and choose the new direction as:
$$\Delta F_{k+1} = \begin{cases} \Delta F, & Tr(\nabla J(F_{k+1})^T \Delta F) \leq -10^{-3} \|\Delta F\| \, \|\nabla J(F_{k+1})\| \\ -\nabla J(F_{k+1}), & \text{otherwise.} \end{cases} \tag{3.15}$$
5. Set $k \leftarrow k+1$ and go to 1.

End(while)

Observe that the stability constraint is embedded within the weak Wolfe rule in item 1 of Algorithm 3.1. Moreover, as can be seen in item 4, if the angle between ΔF and $-\nabla J(F_{k+1})$ is not acute enough, the new direction is taken as $-\nabla J(F_{k+1})$. This property prevents the method to generate a sequence of tiny steps.

It is important to point out that the Wolfe step size conditions (3.13)–(3.14) are required to show convergence to a stationary point $F_* \in \mathcal{L}(F_0)$. However, in practice starting the step-size rule with the whole step, i.e., $\alpha_0 = 1$, and using a backtracking approach it suffices to use the sufficient decrease condition (3.13).

4. Convergence Analysis

In the following we assume that $\nabla J(F_k) \neq 0$ for all k, for otherwise a stationary point is found.

Assumption 4.1. *The following assumptions hold:*

1. *The objective function J is bounded from below.*
2. *The level set* (1.4) *is bounded.*
3. *At any iteration k there always exists an $\alpha_k > 0$ such that $F_k + \alpha_k \Delta F_k \in \mathcal{S}_F$.*

The compactness of the level set $\mathcal{L}(F_0)$; see [12, Appendix A], implies that it is bounded.

Lemma 4.1. *Let $F \in \mathcal{S}_F$ and let $P(F)$ and $K(F)$ solve the discrete Lyapunov equations* (1.2) *and* (2.4)*, respectively. Then $\nabla J(F)$ given by* (3.3) *is Lipschitz continuous.*

Proof. The proof is straight forward. Similar result can also be found in [13] for the case of continuous-time SOF problem. □

The proof of the next theorem is based on [1, Theorem 2].

Theorem 4.1. *Let $\mathcal{S}_F \neq \emptyset$ and let $F_0 \in \mathcal{S}_F$. Assume further that the level set* (1.4) *is bounded and $\nabla J(F)$ is Lipschitz continuous. Assume that for all $k \geq 0$ there exist constants $\omega, \Omega > 0$ such that $0 < \omega \leq \theta_k \leq \Omega$. Then for the CG method* (3.6)–(3.9) *satisfying the weak Wolfe conditions* (3.13)–(3.14)*, either*

$$\nabla J(F_k) = 0 \qquad \text{for some} \quad k \geq 0,$$

or

$$\liminf_{k\to\infty} \quad \|\nabla J(F_k)\| = 0. \tag{4.1}$$

Proof. Suppose that $\nabla J(F_k) \neq 0$ for all $k \geq 0$ and

$$\liminf_{k\to\infty} \quad \|\nabla J(F_k)\| > 0.$$

Define

$$\gamma := \inf\{\|\nabla J(F_k)\| \ : \ k \geq 0\} > 0.$$

From the Wolfe condition (3.14) we have:

$$\begin{aligned} Tr(Y_k^T S_k) &= Tr\big((\nabla J(F_{k+1}) - \nabla J(F_k))^T S_k\big) \\ &\geq (\tilde{\sigma}_2 - 1)\, Tr(\nabla J(F_k)^T S_k) \\ &= -(1-\tilde{\sigma}_2) Tr(\nabla J(F_k)^T S_k). \end{aligned} \tag{4.2}$$

On the other hand, Theorem 3.2 implies that:

$$\begin{aligned} Tr(\nabla J(F_k)^T \Delta F_k) &\leq -\,(\theta_k - \frac{1}{4\delta_{k-1}})\, \|\nabla J(F_k)\|^2 \\ &= -\,\frac{3}{4}\theta_k\, \|\nabla J(F_k)\|^2 \leq -\frac{3}{4}\omega\, \|\nabla J(F_k)\|^2, \end{aligned} \tag{4.3}$$

and then

$$-Tr(\nabla J(F_k)^T \Delta F_k) \geq \frac{3}{4}\omega\gamma^2. \tag{4.4}$$

From (4.2) and (4.4) we have

$$Tr(Y_k^T S_k) \geq \frac{3}{4}(1-\tilde{\sigma}_2)\omega\alpha_k\gamma^2. \tag{4.5}$$

Moreover, from the definition of Y_k:

$$Tr(\nabla J(F_{k+1})^T S_k) = Tr(Y_k^T S_k) + Tr(\nabla J(F_k)^T S_k) < Tr(Y_k^T S_k),$$

which by Wolfe condition (3.14) gives:

$$\begin{aligned} Tr(\nabla J(F_{k+1})^T S_k) \geq& \tilde{\sigma}_2\, Tr(\nabla J(F_k)^T S_k) \\ =& -\tilde{\sigma}_2\, Tr(Y_k^T S_k) + \tilde{\sigma}_2 Tr(\nabla J(F_{k+1})^T S_k). \end{aligned}$$

Hence,

$$(1-\tilde{\sigma}_2)\, Tr(\nabla J(F_{k+1})^T S_k) \geq -\tilde{\sigma}_2\, Tr(Y_k^T S_k).$$

Consequently, one has

$$\left| \frac{Tr(\nabla J(F_{k+1})^T S_k)}{Tr(Y_k^T S_k)} \right| \leq \max\Big(1, \frac{\tilde{\sigma}_2}{1-\tilde{\sigma}_2}\Big). \tag{4.6}$$

The Lipschitz continuity of ∇J implies that

$$\|Y_k\| = \|\nabla J(F_{k+1}) - \nabla J(F_k)\| \leq L\, \|S_k\|,$$

where $L > 0$ is the Lipschitz constant. Then

$$|\delta_k| = \frac{|Tr(Y_k^T \nabla J(F_{k+1}))|}{\|\nabla J(F_{k+1})\|^2} \leq \frac{\|Y_k\|}{\|\nabla J(F_{k+1})\|} \leq \frac{L\,\|S_k\|}{\|\nabla J(F_{k+1})\|}. \tag{4.7}$$

Using (4.5)–(4.7), the boundedness of the level set (1.4), then (3.8) implies that:

$$\begin{aligned} |\beta_k| &\leq \frac{1}{|Tr(Y_k^T S_k)|}\left[\|\nabla J(F_{k+1})\|^2 + |\delta_k|\, \|\nabla J(F_{k+1}\|^2 \frac{|Tr(\nabla J(F_{k+1})^T S_k|}{|Tr(Y_k^T S_k)|} \right] \\ &\leq \frac{4}{3(1-\tilde{\sigma}_2)\omega\alpha_k\gamma^2}\left[\kappa_1^2 + L\kappa_1 \|S_k\| \max\big(1, \frac{\tilde{\sigma}_2}{1-\tilde{\sigma}_2}\big) \right] \\ &\leq \kappa_2 + \kappa_3\, \|S_k\| \leq \kappa_2 + \kappa_3\, \kappa_4, \end{aligned}$$

where $\kappa_i,\ i = 1,2,3,4$ are constant of values:

$$\kappa_1 = \max_{F\in\mathcal{L}}\ \|\nabla J(F)\|, \qquad \kappa_2 = \frac{4\kappa_1^2}{3(1-\tilde{\sigma}_2)\omega\alpha_k\gamma^2},$$

$$\kappa_3 = \frac{4L\kappa_1}{3(1-\tilde{\sigma}_2)\omega\alpha_k\gamma^2}\ \max\big(1, \frac{\tilde{\sigma}_2}{1-\tilde{\sigma}_2}\big), \quad \kappa_4 = \max\ \{\|F_1 - F_2\|\ :\ F_1, F_2 \in \mathcal{L}\},$$

and κ_4 is the diameter of the level set $\mathcal{L}$. Hence, the direction (3.6) is bounded as follows:

$$\|\Delta F_{k+1}\| \leq |\theta_{k+1}|\, \|\nabla J(F_{k+1})\| + |\beta_k|\, \|S_k\| \leq \Omega\, \kappa_1 + (\kappa_2 + \kappa_3\kappa_4)\kappa_4. \tag{4.8}$$

Moreover, from Wolfe condition (3.14) the Lipschitz continuity of ∇J implies that:

$$\begin{aligned}-(1-\tilde{\sigma}_2)\, Tr(\nabla J(F_k)^T \Delta F_k) \leq & Tr\big((\nabla J(F_{k+1}) - \nabla J(F_k))^T \Delta F_k\big) \\ \leq & \|\nabla J(F_{k+1}) - \nabla J(F_k)\| \, \|\Delta F_k\| \leq \alpha_k L \|\Delta F_k\|^2. \end{aligned} \tag{4.9}$$

Since the level set $\mathcal{L}$ is bounded and the objective J is bounded from below, then it follows from Wolfe condition (3.13) and (4.9) that; see [4]:

$$\sum_{k=0}^{\infty} \frac{\big(Tr(\nabla J(F_k)^T \Delta F_k)\big)^2}{\|\Delta F_k\|^2} < +\infty. \tag{4.10}$$

The sufficient descent property (4.3) and (4.10) give

$$\gamma^4 \sum_{k=0}^{\infty} \frac{1}{\|\Delta F_k\|^2} \leq \sum_{k=0}^{\infty} \frac{\|\nabla J(F_k)\|^4}{\|\Delta F_k\|^2} \leq \sum_{k=0}^{\infty} \frac{16}{9\omega^2} \frac{\big(Tr(\nabla J(F_k)^T \Delta F_k)\big)^2}{\|\Delta F_k\|^2} < +\infty,$$

which contradicts with (4.8). Hence, it must hold that

$$\liminf_{k\to\infty} \|\nabla J(F_k)\| = 0.$$

□

5. Stabilized CG Method

In this section the optimization problem (1.1)–(1.2) is modified so that one can start the CG method with $F_0 = 0$. This is achieved by replacing the system matrix A by $A_\mu = (1-\mu)A$, where $\mu \in [0,1)$ is a parameter chosen such that $\rho(A_\mu) < 1$. Consequently, the unconstrained optimization problem (1.1)–(1.2) is modified as follows:

$$\min_{F \in \mathcal{S}_F^\mu} \qquad J^\mu(F) = Tr\,(P^\mu(F) Q(F)), \tag{5.1}$$

where $P^\mu(F)$ solves the discrete Lyapunov equation

$$P^\mu(F) = A_\mu(F) P^\mu(F) A_\mu(F)^T + V, \tag{5.2}$$

$A_\mu(F) = A_\mu + BFC$, and

$$\mathcal{S}_F^\mu = \{F \in \mathbb{R}^{n_u \times n_y} \ : \ \rho(A_\mu(F)) < 1\}, \tag{5.3}$$

is the perturbed set of discrete stabilizing output feedback gains. In this formulation (5.1)–(5.2) is treated as an unconstrained optimization problem in the matrix variable F. A modified nonlinear CG method is considered for solving this problem for a decreasing sequence $\{\mu_k\}_{k\geq 0} \downarrow 0$ of the stabilizing parameter μ, where the eigenvalue condition $F \in \mathcal{S}_F^\mu$ will be fulfilled within the CG method. Obviously, $\mathcal{S}_F^\mu \to \mathcal{S}_F$ as $\mu \to 0$.

One can follow the same steps of the previous sections on this parameterized problem. The first-order directional derivative of the parameterized objective function $J^\mu(F)$ is given in the following lemma.

Lemma 5.1. *Let $F \in \mathcal{S}_F^\mu$. The first-order directional derivative of $J^\mu(F)$ in the direction of ΔF is given by*

$$J_F^\mu(F)\Delta F = 2\, Tr\left((B^T K^\mu(F) A_\mu(F) + RFC)P^\mu(F)C^T \Delta F^T\right), \tag{5.4}$$

where $P^\mu(F)$ and $K^\mu(F)$, respectively, solve the discrete Lyapunov equations (5.2) *and*

$$K^\mu(F) = A_\mu(F)^T K^\mu(F) A_\mu(F) + Q(F). \tag{5.5}$$

Proof. Similar to Theorem 3.1. □

As explained in Section 3 one can obtain the gradient operator $\nabla J^\mu(F)$ explicitly as:

$$\nabla J^\mu(F) = 2\,(B^T K^\mu(F) A_\mu(F) + RFC)P^\mu(F)C^T, \tag{5.6}$$

where $P^\mu(F)$ and $K^\mu(F)$ solve the discrete Lyapunov equations (5.2) and (5.5), respectively.

The nonlinear CG method for solving (5.1)–(5.2) generates a sequence of iterates of the form:

$$F_{k+1} = F_k + \alpha_k \Delta F_k \in \mathcal{S}_F^\mu, \qquad k = 0, 1, 2, \ldots, \tag{5.7}$$

where ΔF_k is a descent direction for $J^\mu(F_k)$ and α_k is the step size determined by the weak Wolfe-Powell rule:

$$J^\mu(F_k + \alpha_k \Delta F_k) - J^\mu(F_k) \le \tilde{\sigma}_1 \alpha_k\, Tr(\nabla J^\mu(F_k)^T \Delta F_k) \tag{5.8}$$

$$Tr(\nabla J^\mu(F_k + \alpha_k \Delta F_k)^T \Delta F_k) \ge \tilde{\sigma}_2\, Tr(\nabla J^\mu(F_k)^T \Delta F_k), \tag{5.9}$$

where $0 < \tilde{\sigma}_1 < \tilde{\sigma}_2 < 1$ are parameters.

The new search direction ΔF_{k+1} is also obtained by using the following modified Dai-Yuan nonlinear CG method:

$$\Delta F_{k+1} = -\theta_{k+1} \nabla J^\mu(F_{k+1}) + \beta_k S_k, \qquad \Delta F_0 = -\nabla J^\mu(F_0), \tag{5.10}$$

where

$$\theta_{k+1} = \frac{\|\nabla J^\mu(F_{k+1})\|^2}{Tr(Y_k^T \nabla J^\mu(F_{k+1}))}, \tag{5.11}$$

$$\beta_k = \frac{1}{Tr(Y_k^T S_k)}\, Tr\left[\left(\nabla J^\mu(F_{k+1}) - \delta_k \frac{\|\nabla J_{k+1}^\mu\|^2}{Tr(Y_k^T S_k)} S_k\right)^T \nabla J^\mu(F_{k+1})\right], \tag{5.12}$$

$$\delta_k = \frac{Tr(Y_k^T \nabla J^\mu(F_{k+1}))}{\|\nabla J^\mu(F_{k+1})\|^2} = \frac{1}{\theta_{k+1}}, \tag{5.13}$$

where $Y_k = \nabla J^\mu(F_{k+1}) - \nabla J^\mu(F_k)$ and $S_k = F_{k+1} - F_k$.

The next theorem shows that the search direction ΔF_{k+1} obtained by the CG method (5.10)–(3.9) is descent.

Theorem 5.1. *If $Tr\,(Y_k^T S_k) \neq 0$ and ΔF_{k+1} is evaluated using* (5.10)–(5.13)*, where $F_{k+1} \in \mathcal{S}_F^\mu$, then*

$$Tr\,(\nabla J^\mu(F_{k+1})^T \Delta F_{k+1}) \le -(\theta_{k+1} - \frac{1}{4\delta_k})\, \|\nabla J^\mu(F_{k+1})\|^2.$$

Proof. Similar to the proof of Theorem 3.2. □

The stabilized nonlinear CG method that solves (5.1)–(5.2) for decreasing values of the stabilizing parameter $\mu_k \geq 0$ is stated in the following lines.

Algorithm 5.1. (CG2: Modified CG1)

Set $F_0 = 0$ and choose $\mu_0 \in [0, 1)$ such that $\rho(A_{\mu_0}) < 1$. Let $P_0^{\mu_0}$ and $K_0^{\mu_0}$ be the solutions of the discrete Lyapunov equations (5.2) and (5.5), respectively. Let $\epsilon^{\text{tol}} \in (0, 1)$ be the tolerance. Choose $0 < \tilde{\sigma}_1 < \tilde{\sigma}_2 < 1$, and a constant $a \in (0, 1)$. Calculate $\nabla J_0^{\mu_0}$; set $\Delta F_0 = -\nabla J_0^{\mu_0}$, and $k \leftarrow 0$.
While $\|\nabla J_k^{\mu_k}\| \geq \epsilon^{\text{tol}}$, do

1. Calculate the step size α_k that satisfies the weak Wolfe-Powell conditions (5.8)–(5.9) and the stability condition (5.7).
2. Set $F_{k+1} = F_k + \alpha_k \Delta F_k$ and calculate $P^{\mu_k}(F_{k+1})$ and $K^{\mu_k}(F_{k+1})$ solutions of the discrete Lyapunov equations (5.2) and (5.5), respectively.
3. Calculate $\nabla J^{\mu_k}(F_{k+1})$. If $\|\nabla J_{k+1}^{\mu_k}\| < \epsilon^{\text{tol}}$, stop; otherwise calculate θ_{k+1}, β_k, and δ_k according to (5.11)–(5.13).
4. Calculate a new step ΔF using (5.10) and choose the new direction as:

$$\Delta F_{k+1} = \begin{cases} \Delta F, & Tr(\nabla J_{k+1}^{\mu_k} \Delta F^T) \leq -10^{-3} \|\Delta F\| \, \|\nabla J_{k+1}^{\mu_k}\| \\ -\nabla J_{k+1}^{\mu_k}, & \text{otherwise.} \end{cases} \tag{5.14}$$

5. Set $\mu_{k+1} \in (0, a\,\mu_k)$; $k \leftarrow k + 1$, and go to 1.

End(do)

Observe that if the uncontrolled system is discrete-time Schur stable, i.e. $\rho(A) < 1$, then Algorithm CG2 reduces to Algorithm CG1, where the stabilizing parameter $\mu_k = 0$ for all k.

6. Numerical Results

In this section an implementation of the two algorithms CG1 and CG2 is described. Two MATLAB codes were written corresponding to this implementation. Five test problems from the benchmark collection COMPlib [8] are considered in detail. The method CG1 is compared with Levine–Athans method [12] and the classical Polak-Ribiére conjugate gradient (CG-PR) method [18, p. 120–131]. The test problems in [8] are for continuous-time models. The function c2d from the Control System Toolbox of MATLAB have been used for converting them into the discrete-time counterparts. The constant data matrices for the low size models are given at the beginning of each example. These test problems can quite demonstrate the performance of the proposed CG methods.

The Levine-Athans method [12, Section IV] is based on solving the nonlinear system (1.2), (2.4) and (3.4) of the optimality conditions iteratively. In this method under the assumption that R, Q and V are positive definite together with B and C having maximum column and row ranks, respectively, then (3.4) implies that

$$F_{+} = -(B^T K B + R)^{-1} B^T K A P C^T (C P C^T)^{-1}, \tag{6.1}$$

where P is positive definite solution of the Lyapunov equation (1.2). Assuming that the inverses $(B^T K B + R)^{-1}$ and $(CPC^T)^{-1}$ exist, then F_{+} is defined.

In this algorithm (6.1) plays two roles. First, it updates F every iteration for given P and K. Moreover, it is used for eliminating the variable F in (2.4) and yields a nonlinear matrix equation in K for given P. This algorithms is iterative and requires in every iteration solving one Lyapunov equation for updating P and one nonlinear matrix equation for updating K.

As explained in the introduction this method is computationally expensive, because it requires solving one nonlinear equation every iteration. Moreover, no convergence results are available for this method. In our implementation we solved the nonlinear equation by the MATLAB solver fsolve.

Note that, it is not convinent to compare the method CG1, which is first order vs. Levine-Athans method with respect to number of iterations. Therefore we consider the CPU time for this purpose, while the number of iterations is considered for comparing the methods CG1 vs. CG-PR.

In each of the following examples we list some of the obtained data in tables. In these tables the first to the fourth columns are, respectively, the iteration counter k, the objective function $J(F_k)$, the convergence criterion $\|\nabla J(F_k)\|$, and the spectral radius $\rho(A(F_k))$ as indicator of fulfilling the stability condition. For every iteration of the CG method two discrete Lyapunov equations are solved using the MATLAB function $\mathsf{dlyap}(\cdot,\cdot)$. All computations were carried out on a Laptop with 1.8 Ghz Core Duo CPU and 214.09 MB RAM. The following values have been assigned to the parameters of Algorithms 3.1 and 5.1:

$$\tilde{\sigma}_1 = 10^{-4}, \quad \tilde{\sigma}_2 = 0.1, \quad a = 0.5, \quad \text{and} \quad \epsilon^{\text{tol}} = 10^{-4}.$$

Instead of using the Wolfe conditions (3.13)–(3.14) we have also tried the simple sufficient decrease condition (3.13) with $\tilde{\sigma}_1 = 10^{-4}$, where the initial step size is chosen as $\alpha_0 = 1$ in the backtracking rule. The two CG methods have given satisfactory results using this alternative.

Example 6.1. The first test problem represents an aircraft model [8, AC1]. The discrete-time counterpart has the following data matrices:

$$A = \begin{bmatrix} 1.0000 & 0.0014 & 0.1132 & 0.0005 & -0.0967 \\ 0 & 0.9945 & -0.0171 & -0.0005 & 0.0068 \\ 0 & 0.0003 & 1.0000 & 0.0957 & -0.0048 \\ 0 & 0.0060 & -0.0000 & 0.9131 & -0.0936 \\ 0 & -0.0277 & 0.0002 & 0.0973 & 0.9287 \end{bmatrix},$$

$$B = \begin{bmatrix} -0.0076 & 0.0000 & 0.0003 \\ -0.0115 & 0.0997 & 0.0000 \\ 0.0212 & 0.0000 & -0.0081 \\ 0.4152 & 0.0003 & -0.1589 \\ 0.1742 & -0.0014 & -0.0154 \end{bmatrix}, \quad C^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$Q = I_5, \quad R = 1.5\, I_3, \quad V = 0.8\, I_5.$$

Table 6.1: Performance of the method CG1 on Example 6.1

iter	$J(F_k)$	$\|\nabla J(F_k)\|$	$\rho(A(F_k))$
0	6.3101e+002	2.70576e+002	9.69e-001
1	6.2793e+002	2.43796e+002	9.70e-001
2	5.5072e+002	1.18018e+002	9.72e-001
3	5.3280e+002	1.12749e+002	9.68e-001
4	5.2724e+002	7.96503e+001	9.68e-001
$\vdots$	$\vdots$	$\vdots$	$\vdots$
214	1.9764e+002	1.87418e-004	9.72e-001
215	1.9764e+002	1.60415e-004	9.72e-001
216	1.9764e+002	9.88938e-005	9.72e-001

The uncontrolled system is not discrete-time Schur stable, where $\rho(A) = 1.0$. Starting with the following initial $F_0 \in \mathcal{S}_F$ the methods CG-PR and CG1 require 2679 and 216 iterations to reach the stationary point F_*, while Levine-Athans method fails to converge due to violating the stability condition $F \in \mathcal{S}_F$. The starting and final feedback gain matrices are:

$$F_0 = \begin{bmatrix} 1.1143 & -0.9611 & 2.0923 \\ 1.6336 & -5.9955 & 5.6370 \\ 3.9464 & -5.9739 & 9.6638 \end{bmatrix}, \quad F_* = \begin{bmatrix} 0.1771 & 0.0062 & 0.2524 \\ -0.2361 & -1.0548 & -0.9299 \\ 0.6372 & 0.5192 & 1.9222 \end{bmatrix}.$$

Table 6.1 shows the convergence behavior of the method CG1 to reach the stationary point $F_* \in \mathcal{S}_F$ of the optimization problem (1.1)–(1.2).

Example 6.2. The second test problem represents the L-1011 aircraft application [8, AC6]. The discrete model has the following constant data matrices:

$$A = \begin{bmatrix} 1.0000 & 0.0020 & 0.0952 & -0.0234 & 0 & 0.0009 & -0.0027 \\ 0.0003 & 0.9776 & -0.0004 & 0.1374 & 0 & -0.0317 & -0.0011 \\ -0.0009 & 0.0467 & 0.9048 & -0.4423 & 0 & 0.0125 & -0.0384 \\ 0.0035 & -0.0888 & 0.0002 & 0.8026 & 0 & 0.0027 & 0.0001 \\ 0.0000 & 0.0408 & -0.0000 & 0.0031 & 0.6703 & -0.0009 & -0.0000 \\ 0 & 0 & 0 & 0 & 0 & 0.1353 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0821 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.0007 & -0.0027 \\ -0.0419 & -0.0020 \\ 0.0176 & -0.0682 \\ 0.0026 & 0.0001 \\ -0.0007 & -0.0000 \\ 0.8647 & 0 \\ 0 & 0.9179 \end{bmatrix}, \quad C^T = \begin{bmatrix} 0 & 0 & 1.0000 & 0 \\ -0.1540 & 0.2490 & 0 & 0 \\ -0.0042 & -1.0000 & 0 & 0 \\ 1.5400 & -5.2000 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \\ -0.7440 & 0.3370 & 0 & 0 \\ -0.0320 & -1.1200 & 0 & 0 \end{bmatrix},$$

$Q = I_7, R = 1.5\,I_2, V = 0.8\,I_7$ The uncontrolled system is discrete-time Schur stable, since $\rho(A) = 0.9992 < 1$. Starting from the following $F_0 \in \mathcal{S}_F$ the methods CG-PR and CG1 require 1411 and 129 iterations to converge to the stationary point $F_* \in \mathcal{S}_F$, respectively, while Levine-Athans method fails to reach the stationary point. The starting and final feedback gain matrices are the following:

$$F_0 = \begin{bmatrix} -0.3279 & -0.0576 & -0.0170 & 0.2643 \\ -1.3502 & -0.2078 & 0.4329 & -0.0634 \end{bmatrix},$$

$$F_* = \begin{bmatrix} -0.4104 & -0.0640 & 0.0436 & 0.2233 \\ -0.9470 & -0.1539 & 0.2635 & 0.0354 \end{bmatrix}.$$

The final closed-loop system matrix $A(F_*)$ is Schur stable, since $\rho(A(F_*)) = 0.9162$.

Example 6.3. This test problem is of a plate experiment for the active vibration damping of large flexible space structures [8, DLR2]. The discrete-time model has the dimensions $n_x = 40, n_u = 2$, and $n_y = 2$. The data matrices A, B, C, Q, R, and V are not included because of their large sizes.

The uncontrolled system is discrete-time Schur stable; $\rho(A) = 0.9895$. Starting from an initial $F_0 \in \mathcal{S}_F$ Levine-Athans method requires 4 outer iterations and 21 inner iterations of the nonlinear solver to reach the stationary point $F_* \in \mathcal{S}_F$, while the two methods CG-PR and CG1 require 21 and 23 iterations, respectively, to converge to the same stationary point $F_* \in \mathcal{S}_F$. The initial and final feedback gains are the following:

$$F_0 = \begin{bmatrix} -2.3953 & 12.5208 \\ 2.3953 & 12.5208 \end{bmatrix}, \qquad F_* = \begin{bmatrix} -0.5809 & 0.9729 \\ 0.5809 & 0.9729 \end{bmatrix},$$

and the spectral radius of the final closed-loop system matrix is $\rho(A(F_*)) = 0.9994$.

The following two examples are not discrete-time Schur stable, which are chosen to test the performance of the method CG2.

Example 6.4. This test problem represents a 2D heat flow over a rectangular domain for thermal properties of copper [8, HF2D10]. The data matrices of the discrete-time model are the following:

$$A = \begin{bmatrix} 1.0115 & 0.0067 & 0.0104 & 0.0792 & 0.0225 \\ 0.0053 & 0.9255 & -0.0639 & 0.0015 & -0.1065 \\ 0.0083 & -0.0587 & 0.8498 & -0.0053 & -0.1639 \\ 0.0106 & 0.0191 & 0.0062 & 0.3279 & 0.0126 \\ -0.0029 & -0.0683 & -0.1523 & 0.0152 & 0.1259 \end{bmatrix}, \; B = \begin{bmatrix} -0.4617 & -2.2370 \\ 1.5457 & 1.5627 \\ 0.6741 & 2.4761 \\ -1.3139 & 5.8252 \\ 0.8135 & 1.7861 \end{bmatrix},$$

$$C = \begin{bmatrix} 0.0265 & 0.0202 & 0.0075 & 0.0140 & 0.0020 \\ 0.0093 & -0.0306 & -0.0085 & 0.0379 & -0.0599 \\ 0.0121 & -0.0036 & -0.0031 & -0.0345 & -0.0245 \end{bmatrix},$$

$Q = 100\, I_5, R = 1.5\, I_2, V = 0.8\, I_5$. The uncontrolled system is not discrete-time Schur stable ($\rho(A) = 1.0133 > 1$). The method CG2 started with $F_0 = 0$ and $\mu_0 = 0.0133$, where $\rho(A_{\mu_0}) = 0.9998$, and after 91 iterations the following stationary point is reached:

$$F_* = \begin{bmatrix} 0.2369 & 3.8199 & 3.6033 \\ 2.3252 & -0.1429 & 2.0819 \end{bmatrix}, \qquad \mu_* = 5.3732 \times 10^{-30}.$$

The final closed-loop system matrix $A_{\mu_*}(F_*)$ is Schur stable, where $\rho(A_{\mu_*}(F_*)) = 0.9221 < 1$.

Table 6.2 shows the convergence behavior of the method CG2 on Example 6.4, where the second column is included to show the decrease occurring in the stabilizing parameter μ_k throughout iterations.

Table 6.2: Performance of the method CG2 on Example 6.4

$iter$	μ_k	$J^{\mu_k}(F_k)$	$\|\nabla J^{\mu_k}(F_k)\|$	$\rho(A_{\mu_k}(F_k))$
0	1.33e-002	6.9740e+003	1.7550e+003	9.4511e-001
1	6.65e-003	6.9740e+003	1.7550e+003	9.5023e-001
2	3.32e-003	6.8112e+003	2.3669e+003	9.5113e-001
3	1.66e-003	6.4322e+003	9.9492e+002	9.4681e-001
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
89	2.15e-029	2.2087e+003	3.1142e-004	9.2212e-001
90	1.07e-029	2.2087e+003	1.3676e-004	9.2212e-001
91	5.37e-030	2.2087e+003	5.6430e-005	9.2212e-001

Example 6.5. This example is the discrete-time model of a decentralized interconnected system (see [8, DIS4]). The constant data matrices are the following:

$$A = \begin{bmatrix} 0.9942 & 0.1079 & 0.0598 & 0.1053 & 0.0660 & 0.0041 \\ -0.1855 & 0.7234 & 0.0833 & -0.0028 & 0.0011 & 0.0881 \\ -0.0222 & 0.1865 & 1.0660 & 0.1063 & 0.1093 & 0.0585 \\ 0.0803 & 0.2789 & 0.0168 & 1.0552 & 0.0022 & -0.0368 \\ -0.0106 & 0.1003 & 0.1129 & 0.0053 & 1.1107 & 0.0076 \\ -0.2594 & -0.3493 & -0.0232 & 0.0363 & 0.0430 & 0.9803 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.1051 & 0.0085 & 0.0207 & 0.0068 \\ 0.0763 & 0.0133 & -0.0004 & 0.0139 \\ 0.0088 & 0.3091 & 0.0208 & 0.0190 \\ 0.0186 & 0.0017 & 0.4107 & -0.0060 \\ 0.0047 & 0.0163 & 0.0007 & 0.2115 \\ -0.0319 & -0.0024 & 0.0082 & 0.3025 \end{bmatrix}, \quad C = I_6,$$

$Q = 100\, I_6, R = 1.5\, I_4, V = 0.8\, I_6$. The uncontrolled system matrix A is not discrete-time Schur stable, where $\rho(A) = 1.1551$. Starting with $\mu_0 = 0.1344$ and $F_0 = 0$ such that $\rho(A_{\mu_0}) = 0.9998 < 1$, the method CG2 requires 55 iterations to reach the stationary point:

$$F_* = \begin{bmatrix} -4.9165 & -2.6761 & -0.4861 & -0.2655 & -3.1104 & 1.8121 \\ 0.8371 & -1.3776 & -3.1233 & -0.0953 & -1.3041 & 0.5810 \\ -0.5940 & 0.0016 & 0.0255 & -2.4101 & 0.6404 & -0.3524 \\ 0.5850 & -0.0017 & -0.1536 & -0.0213 & -2.3383 & -1.5829 \end{bmatrix},$$

with $\mu_* = 3.7303 \times 10^{-18}$ and $\rho(A_{\mu_*}(F_*)) = 0.8780$.

In Table 6.3 we report some preliminary tests for Algorithm 3.1 on 47 test problems derived from the benchmark [8]. The method CG1 is compared vs. Levine-Athans method and PR-CG method with respect to number of iterations and CPU time (sec.). For each test problem we list the problem name, the problem dimensions (n_x, n_u, n_y), the spectral radii $\rho(A)$ and $\rho(A(F_*))$ of the open- and closed-loop systems, respectively, the total number of iterations, and CPU time. The fifth column of Table 6.3 shows that the open-loop system is not discrete-time Schur stable for 24 test problems. The symbols SE and OME in the 7th column indicate that the method breaks down because of violating the stability condition and out of memory, respectively. The iterations were terminated when $\|\nabla J(F_k)\| \leq 1 \times 10^{-4}$ or after 3000 iterations. The overall results show that Algorithm 3.1 outperforms Polak-Ribiére CG method with respect to number of iterations as well as number of wins. Moreover, the proposed CG method outperforms Levine-Athans method with respect to CPU time as well as number of wins.

7. Conclusion

In this paper a modified Dai-Yuan nonlinear conjugate gradient method is studied and analyzed for solving the discrete-time static output feedback control problem. The special

Table 6.3: Performance of the method CG1 vs. Levine-Athans method and Polak-Ribiére CG method on the discrete forms of engineering applications from the benchmark [8].

						Number of iterations and CPU-time					
	Problem dimension			Stability indicators		Levine-Athans		CG-PR		CG1	
Problem	n_x	n_u	n_y	$\rho(A)$	$\rho(A(F_*))$	No. it.	CPU-time	No. it.	CPU-time	No. it.	CPU-time
AC1	5	3	3	1.0	9.72e-001	SE	–	2679	74.12	216	5.62
AC3	5	2	4	0.9991	9.32e-001	307(1142)	38.83	263	5.41	60	1.25
AC4	4	1	2	1.2942	9.95e-001	2(4)	0.94	188	5.37	34	1.01
AC6	7	2	4	0.9992	9.13e-001	SE	–	1411	41.50	129	3.99
AC8	9	1	5	1.0012	9.61e-001	4(9)	1.73	3000	–	1703	64.71
AC15	4	2	3	0.9990	9.68e-001	32(137)	3.96	301	6.29	81	1.89
AC16	4	2	4	0.9990	9.73e-001	2(12)	1.05	506	11.42	72	1.64
AC17	4	1	2	0.9723	9.47e-001	17(52)	2.14	16	0.23	18	0.28
HE1	4	2	1	1.0280	9.91e-001	13(42)	2.06	50	0.73	17	0.23
HE2	4	2	2	0.9971	9.70e-001	SE	–	217	8.07	45	1.48
HE3	8	4	6	1.0088	9.61e-001	SE	–	2664	85.72	214	6.47
REA1	4	2	3	1.2203	9.07e-001	28(73)	2.93	28	0.39	23	0.25
REA2	4	2	2	1.2227	9.18e-001	9(28)	1.50	60	0.94	23	0.27
REA3	12	1	3	1.0	9.98e-001	SE	–	3000	–	155	9.81
MFP	4	3	2	0.9979	9.96e-001	SE	–	729	24.51	79	2.56
DIS1	8	4	4	0.9912	9.59e-001	8(46)	3.93	139	2.82	50	1.05
DIS2	3	2	2	1.1824	7.83e-001	11(47)	1.56	64	1.09	29	0.41
DIS3	6	4	4	0.9620	9.05e-001	6(28)	1.98	66	1.01	35	0.58
DIS4	6	4	6	1.1551	8.71e-001	3(8)	1.30	54	0.81	31	0.45
PSM	7	2	3	0.9495	9.06e-001	5(13)	1.62	27	0.44	16	0.22
NN2	2	2	2	1.0	9.50e-001	5(8)	0.87	5	0.08	8	0.19
NN4	4	2	3	0.9959	9.32e-001	SE	–	350	7.36	72	1.15
NN8	3	2	2	0.9971	9.24e-001	10(27)	1.33	47	0.72	32	0.41
NN11	16	3	5	0.9048	9.17e-001	SE	–	48	1.31	23	0.66
NN15	3	2	2	1.0	9.82e-001	10(52)	1.76	202	4.88	48	1.22
NN16	8	4	4	1.0	9.81e-001	SE	–	113	4.37	45	1.53
NN17	3	2	1	1.1241	9.74e-001	SE	–	238	5.09	32	0.70
HF2D10	5	2	3	1.0133	9.22e-001	6(10)	1.19	44	0.19	52	0.28
HF2D13	5	2	4	0.9756	8.01e-001	3(4)	1.00	882	12.76	76	0.97
HF2D14	5	2	4	1.0227	9.62e-001	5(11)	1.12	3000	–	497	13.17
HF2D15	5	2	4	1.1689	8.73e-001	4(8)	1.14	2448	47.72	137	2.49
HF2D16	5	2	4	1.0114	9.57e-001	4(8)	1.11	105	3.04	43	1.23
HF2D17	5	2	4	1.0557	8.07e-001	5(9)	1.12	83	1.75	34	0.53
HF2D18	5	2	2	1.0285	9.96e-001	4(11)	1.19	297	5.80	96	1.87
Nile	7	2	3	0.9282	9.28e-001	5(5)	1.47	11	0.23	14	0.28
UWV	8	2	2	0.9989	3.07e-001	3(4)	1.23	18	0.89	16	0.83
DLR1	10	2	2	0.9995	9.99e-001	8(52)	6.07	67	3.57	36	1.93
DLR2	40	2	2	0.9995	9.99e-001	4(21)	199.40	21	5.35	23	6.22
AGS	12	2	2	0.9786	9.79e-001	SE	–	1841	125.60	99	7.54
EB1	10	1	1	0.9990	9.83e-001	5(34)	3.37	9	0.48	24	1.61
CSE1	20	2	10	1.0000	9.95e-001	4(13)	11.45	67	4.48	35	1.95
CSE2	60	2	30	1.0000	9.99e-001	4(19)	1179.0	486	321.10	96	55.07
CM1	20	1	2	1.0000	9.99e-001	SE	–	581	38.45	42	2.93
IH	21	11	10	1.0	9.79e-001	SE	–	20	1.37	21	1.17
LAH	48	1	1	0.9742	9.74e-001	2(7)	154.1	4	1.90	17	7.83
HF1	130	1	2	0.9981	9.96e-001	OME	–	19	52.93	14	29.42
ISS1	270	3	3	0.9997	9.99e-001	SE	–	101	278.2	56	182.0

structure of the problem is taken into consideration in two folds: in fulfilling an eigenvalue condition concerning the stability of the corresponding control system as well as dealing with matrix variables. In particular the stability condition is fulfilled within the line search rule. By parameterizing the optimization problem the resulting CG method, denoted by CG2, does not require initial stabilizing SOF gain $F_0 \in \mathcal{S}_F$ to start the iteration sequence.

The proposed methods performed quite well on wide range of test problems of engineering applications. One of the main advantages of these methods is that stationary points are determined in few seconds for quite large problems.

References

[1] N. Andrei, A Dai-Yuan conjugate gradient algorithm with sufficient descent and conjugancy conditions for unconstrained optimization, *Appl. Math. Lett.*, **21** (2008), 165–171.

[2] Y.-Y. Cao and J. Lam, A computational method for simultaneous LQ optimal control design via piecewise constant output feedback, *IEEE T. Syst. MAN Cy. B.*, **31** (2001), 836–842.

[3] Y.H. Dai and Y. Yuan, An efficient hybrid conjugate gradient method for unconstrained optimization, *Ann. Oper. Res.*, **103** (2001), 33-47.

[4] Y.H. Dai and Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, *SIAM J. Optimiz.*, **10** (1999), 177–182.

[5] R. Fletcher and C. Reeves, Function minimization by conjugate gradients, *Computer Journal*, **7** (1964), 149-154.

[6] M.R. Hestenes and E.L. Stiefel, Methods of conjugate gradients for solving linear systems, *Journal Res. Natl. Bur. Stand.*, **49** (1952), 409–436.

[7] M. Kočvara, F. Leibfritz, M. Stingl and D. Henrion, *A nonlinear SDP algorithm for static output feedback problems in COMPlib*, in Proceedings of the 16th IFAC World Congress on Automatic Control, Prague, 2005, Elsevier, ISBN: 0-08-045108-X.

[8] F. Leibfritz, COMP*lib*: COnstraint Matrix-optimization Problem *lib*rary–a collection of test examples for nonlinear semi-definite programs, control system design and related problems. Technical Report 2004. `http://www.complib.de/`

[9] F. Leibfritz and E.M.E. Mostafa, An interior point constrained trust region method for a special class of nonlinear semidefinite programming problems, *SIAM J. Optimiz.*, **12** (2002), 1048–1074.

[10] F. Leibfritz and E.M.E. Mostafa, Trust region methods for solving the optimal output feedback design problem, *Int. J. Control*, **76** (2003), 501–519.

[11] L. Lukšan, C. Matonoha and J. Vlček, Computational experience with modified conjugate gradient methods for unconstrained optimization, *Inst. of Comp. Sci., Academy of Sci., Czech Rep., Tech. Rep. No. 1038*, 2008.

[12] P.M. Mäkilä and H.T. Toivonen, Computational methods for parametric LQ problems–a survey, *IEEE T. Automat. Contr.*, **32** (1987), 658–671.

[13] E.M.E. Mostafa, Efficient trust region methods in numerical optimization, Ph.D. Thesis, Dept. of Mathematics, Faculty of Science, Alexandria University, Egypt, 2000.

[14] E.M.E. Mostafa, A trust region method for solving the decentralized static output feedback design problem, *Journal of Applied Mathematics & Computing*, **18** (2005), 1–23.

[15] E.M.E. Mostafa, An augmented Lagrangian SQP method for solving some special class of nonlinear semi-definite programming problems, *Computational and Applied Mathematics*, **24** (2005), 461–486.

[16] E.M.E. Mostafa, Computational design of optimal discrete-time output feedback controllers, *Journal of the Operations Research Society of Japan*, **51** (2008), 15–28.

[17] E. Najson and J.L. Speyer, Stabilization via output static feedback in discrete-time linear systems, Proceedings of the 42nd IEEE Conference on Decision and Control, Hawaii, (2003), 6466–6473.

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

[19] E. Polak and G. Ribiére, Note sur la convergence de directions conjugue, *Rev. Francaise Informat Recherche Operationelle*, 3e Anne, **16** (1969), 35–43.

[20] T. Rautert and E. W. Sachs, Computational design of optimal output feedback controllers. *SIAM J. Optimiz.*, **7** (1997), 837–852.

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

[22] H.T. Toivonen, A globally convergent algorithm for the optimal constant output feedback problem, *Int. J. Control*, **41** (1985), 1589–1599.