International Journal of Control
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tcon20

Finding common quadratic Lyapunov functions for switched linear systems using particle swarm optimisation

R.H. Ordóñez-Hurtado [a] & M.A. Duarte-Mermoud [a]

[a] Department of Electrical Engineering and Advanced Mining Technology Center (AMTC), University of Chile, Avda. Tupper 2007, Casilla 412-3, Santiago, Chile

Available online: 07 Dec 2011

To cite this article: R.H. Ordóñez-Hurtado & M.A. Duarte-Mermoud (2012): Finding common quadratic Lyapunov functions for switched linear systems using particle swarm optimisation, International Journal of Control, 85:1, 12-25

To link to this article: http://dx.doi.org/10.1080/00207179.2011.637133

International Journal of Control
Vol. 85, No. 1, January 2012, 12–25

Taylor & Francis
Taylor & Francis Group

Finding common quadratic Lyapunov functions for switched linear systems using particle swarm optimisation

R.H. Ordóñez-Hurtado* and M.A. Duarte-Mermoud

Department of Electrical Engineering and Advanced Mining Technology Center (AMTC), University of Chile, Avda. Tupper 2007, Casilla 412-3, Santiago, Chile

(*Received 23 May 2011; final version received 28 October 2011*)

It is undoubtedly important to be able to ensure the existence of a common quadratic Lyapunov function (CQLF) for a given switched system because this is proof of its asymptotic stability, but equally important is the ability to calculate it in order to obtain more specific information about the behaviour of the switched system under analysis. This article describes the development of a new methodology for calculating a CQLF based on particle swarm optimisation (PSO) once the existence of a CQLF has been assured. Several comparative analyses are presented to show the strengths and advantages of the proposed methodology.

Keywords: common quadratic Lyapunov function; particle Swarm optimisation; switched linear systems; computation of CQLF

1. Introduction

Switched linear systems (SLS) are the type of linear dynamic systems whose evolution matrix evolves over time depending on a signal called commutation signal $\sigma(t)$, as follows:

$$\dot{x}(t) = A_{\sigma(t)}x(t), \quad A_{\sigma(t)} \in \mathrm{A} := \{A_1, A_2, \ldots, A_N\}. \quad (1)$$

Their mathematical models very accurately represent the dynamics of various types of systems in different application areas (Liberzon 2003; Lin and Antsaklis 2009). However, unlike the stability analysis for linear LTI systems, stability analysis of such systems is not limited to dealing only with stable subsystems, since, on the one hand, a certain commutation rule may result in instability of the whole system despite it being made up of stable systems and, on the other hand, a switched system with unstable subsystems can be stabilised by an appropriate switching rule.

Within the research on the stability of switched systems, Lyapunov stability-based approaches are highlighted and important lines of analysis have been derived, such as the ones based on inverse Lyapunov theorem, switched quadratic Lyapunov functions and common quadratic Lyapunov functions (CQLFs). Among the above, the existence of a CQLF has been widely explored, directing the efforts to both the determination of conditions of existence/non-existence of a CQLF (Lin and Antsaklis 2009) and the design of a method for finding a CQLF given its existence, being the resolution of linear matrix inequalities (LMI) (Boyd, El-Ghaoui, Feron, and Balakrishnan 1994) and gradient-based method (Liberzon and Tempo 2004), the two most popular finding methods.

In general, from the statement of the CQLF approach (referred here as the CQLF Problem), the process of finding a CQLF was addressed earlier using LMIs and appropriate resolution tools (LMI solvers) (e.g. Shorten and Narendra 2003). However, although quite efficient tools exist (e.g. Matlab LMI Toolbox, Matlab Robust Control Toolbox), their algorithms are strongly affected by the order and the number of matrices to analyse (Liberzon and Tempo 2004). Later, other methods for finding a CQLF was developed with the aim of improving the limitations of LMI tools, and the most outstanding is the gradient-based method (Liberzon and Tempo 2004). Although a methodology with a solid analytical support is developed there, and all its advantages over the LMI tools are highlighted, the results of the methodology suggest that improvements in the respect of computing time consumed can be made. Other calculation techniques have been reported in the literature, but most cases are quite restrictive such as those presented in: Cheng, Martin, and Xiang (2000), with high computational complexity because of performing calculations with matrices of dimensionality $N \times n^2$, with N the number of n-th order matrices to be analysed; Nguyen, Mori, Mori, and Kuroe (2003) and Yanami and Anai (2005), with algorithmic and computational complexity that restricts the application of the methodology to systems

*Corresponding author. Email: roordone@ing.uchile.cl

ISSN 0020–7179 print/ISSN 1366–5820 online

http://dx.doi.org/10.1080/00207179.2011.637133
http://www.tandfonline.com

of second-order; Paul, Akar, Safonov, and Mitra (2004), which requires some kind of second-order systems; Chen and Gao (2007), whose area of application is limited to triangular matrices and Ibeas and De la Sen (2009), that requires simultaneously triangularisable matrices.

As a direct consequence of analysing the scenario described above, it is clear that a new methodology for the calculation of a CQLF, more efficient and less restrictive, must be developed. Then in this article, a new general methodology based on the global optimisation technique, particle swarm optimisation (PSO), is proposed. For its development, we use stability theory for switched linear systems as analytical support, and Matlab as main software tool. The rest of this document includes: Section 2 which present the CQLF problem statement and the theoretical framework of both the existing solutions for finding a CQLF and the PSO technique; Section 3 which presents the proposed methodology; Section 4 with the analysis of the methodology and Section 5 for the presentation of relevant conclusions.

2. Preliminaries

This section presents the theoretical basis necessary for understanding the development of the methodology proposed in Section 3.

2.1 *The CQLF problem*

Let the switched linear system (1) be in continuous time (Lin and Antsaklis 2009), where $x(t) \in \Re^n$ is the state vector, A is a set of N Hurwitz matrices in $\Re^{n\times n}$ and σ is an arbitrary signal that define commutation among elements in A. We define a quadratic Lyapunov function candidate as

$$V(x) = x^T P x, \quad P > 0,\ P \in \Re^{n\times n}, \tag{2}$$

which is positive definite, whose time derivative along any non-zero system trajectory of (1) is required to be negative definite, i.e.

$$\dot{V}(x) = x^T\left(PA_i + A_i^T P\right)x < 0, \quad \forall x \neq 0,\ \forall i \in \{1,\ldots,N\}, \tag{3}$$

for which it is necessary that

$$PA_i + A_i^T P < 0, \quad \forall i \in \{1,\ldots,N\}. \tag{4}$$

Then, if a matrix $P > 0$ satisfying (2) and (4) $\forall A_i \in \mathrm{A}$ exists, function $V(x)$ is a CQLF for all systems

$$\Sigma_{A_i} : \dot{x}(t) = A_i x(t), \quad i = 1, 2, \ldots, N, \tag{5}$$

and its existence is a guarantee of uniform asymptotic stability of system (1) (Liberzon 2003; Lin and Antsaklis 2009). Hereafter, and as abuse of notation, a CQLF will be referred to as such (2) as will the respective P matrix associated with it, as well.

Finally, Lemma 2.1, which will be used to develop the proposed methodology, is provided below.

Lemma 2.1 (Horn and Johnson 1985)**:** *Let A be an arbitrary matrix in* $\Re^{n\times n}$. *Then it follows that* (i) $A > 0 \Leftrightarrow -A < 0$ *and* (ii) $A > 0 \Leftrightarrow (A + A^T) > 0$.

2.2 *Methodologies for finding a CQLF*

From the numerical point of view, there are several methodologies for finding a CQLF in the literature, with LMI being the most traditional technique used. For this approach, the LMI system (4) is called *feasible* if it is satisfied for some $P > 0\ \forall i$, and *unfeasible* otherwise. Although there have been efficient methods for solving system's LMI for a couple of decades (Boyd et al. 1994; Shorten and Narendra 2003), the CQLF problem remains an open problem because such methods impose strong conditions for finding solutions, resulting in this methodology becoming ineffective to the extent that it increases the dimensionality of the problem (order and number of subsystems). Needless to say, it is unthinkable to apply methods for solving LMIs in the case of infinite families of subsystems, except in specific cases such as a convex combination of a finite family, and this is where gridding techniques (Tempo, Calafiore, and Dabbene 2004) become important. However, gridding techniques require a number of grid points exponentially growing with the dimensionality of the problem, and a solution is not assured if they were located among the grid points, since it is impossible to access it.

An example of an analytical method for finding a CQLF is the classic (Narendra–Balakrishnan) NB Algorithm, based on the results of these authors presented in Narendra and Balakrishnan (1994). This method has the advantage of guaranteeing a feasible solution as long as certain conditions on the tested systems are met. Such conditions are quite restrictive because they require that the matrices commute under multiplication, and an extension of the NB Algorithm requires that the matrices meet certain condition on the Lie bracket (Zhu, Cheng, and Qin 2007). The concept of this algorithm is based on the fact that N matrices, fulfilling the condition, they share a CQLF P_N that can be calculated with the formula

$$P_i A_i + A_i^T P_i = -P_{i-1}, \quad i = 1, \ldots, N, \tag{6}$$

all this regardless of the order in which matrices are taken or which initial matrix $P_0 > 0$ is chosen.

Another analytical method for finding a CQLF is the one proposed in Cheng, Hu, and Martin (2006), based on the calculation of the smallest ball that contains a finite number of points $p_1, \ldots, p_n \in \Re^n$. Although the method was not developed directly to calculate a CQLF, one of the examples of application shows that possibility. Thus, the authors propose assimilating each matrix as a point in a multidimensional space, and then the smallest ball that includes all points (matrices) is calculated. If the centre of this enclosing smallest ball represents an unstable matrix, then it follows that the set of matrices does not share a CQLF. On the other hand, if the centre leads to a stable matrix then a Lyapunov function P for that centre (matrix) is calculated, and in this way P becomes a CQLF candidate. The last step in the method is to check the Lyapunov equations one by one to confirm that P is a CQLF. The process of finding such a CQLF candidate P using this method is called the Enclosing Balls (EB) algorithm in this work.

Despite the above methods, among the most important method for calculating a CQLF is the Liberzon and Tempo (2004) approach, which raises the appropriate minimisation of a pair of functionals using the gradient method. This is based on the traditional formula (Liberzon and Tempo 2004)

$$f(R + \Delta R) \approx f(R) + \langle \partial_R f, \Delta R \rangle, \tag{7}$$

where ΔR denotes a small perturbation with respect to R, $\partial_R f$ denotes the gradient of f with respect to R, and $\approx$ denotes equality up to terms of the first-order in ΔR. From this, a function $v(P, A)$ is generated in the form

$$v(P, A) := f\big(PA + A^T P + Q\big), \tag{8}$$

where f is a suitable functional to solve the CQLF problem, with $Q > 0$ arbitrary and given. The generic formulas for finding a CQLF are summarised in the following algorithm:

$$P_{k+1} = \begin{cases} \big[P_k - \mu_k \partial_P v\big(P_k, A_{h(k)}\big)\big]^+ & v\big(P_k, A_{h(k)}\big) > 0 \\ P_k & \text{in other case} \end{cases} \tag{9}$$

$$\mu_k := \frac{\alpha v\big(P_k, A_{h(k)}\big) + r\big\|\partial_P v\big(P_k, A_{h(k)}\big)\big\|}{\big\|\partial_P v\big(P_k, A_{h(k)}\big)\big\|^2}, \tag{10}$$

$$h(k) := (k \bmod N) + 1 = k - N\left\lfloor \frac{k}{N} \right\rfloor + 1, \tag{11}$$

where $[\cdot]^+$ denotes the projection into space of positive semi-definite matrices, $\lfloor \cdot \rfloor$ denotes approximation to the lower nearest integer, μ_k is the step forward with $0 \le \alpha \le 2$, $r > 0$ and $\|\cdot\|$ being the Frobenius norm. In Liberzon and Tempo (2004) two functionals are proposed, the respective partial derivatives $\partial_P v$ are calculated, and their continuity and differentiability are demonstrated in an elegant way. Results obtained are compared with LMI techniques, adding the respective analysis of deterministic and probabilistic convergence based on the proper tuning of parameters r and α.

2.3 *PSO technique*

PSO (Eberhart and Kennedy 1995a; Del Valle, Venayagamoorthy, Mohagheghi, Hernandez, and Harley 2008) is an heuristic global optimisation technique that is part of the category called swarm intelligence (SI), which in turn is a subcategory of evolutionary computation (EC). This technique allows solving optimisation problems using cumuli (swarms) of particles, which computationally simulate the behaviour of social groups in nature (flocks, banks, crowds, etc.) in their search for a common benefit. Thus the potential solutions are constituted by populations of individuals (particles) that evolve iteratively using strategies and operations related to the movement in a λ-dimensional space, where λ is the number of unknowns in the function to optimise. As was proposed since its statement in 1995 (Eberhart and Kennedy 1995a,b), PSO retains the best individual solution $\mathbf{p}_i$ and global solution $\mathbf{g}$ throughout the iterative process, and in its basic version combines these values in its two equations of evolution

$$\begin{aligned} v_{i,d}(k+1) = v_{i,d}(k) &+ r_1(k)c_1\big(p_{i,d}(k) - x_{i,d}(k)\big) \\ &+ r_2(k)c_2\big(g_d(k) - x_{i,d}(k)\big) \end{aligned} \tag{12}$$

$$x_{i,d}(k+1) = x_{i,d}(k) + v_{i,d}(k+1) \tag{13}$$

$$\mathbf{p}_i(k) = \begin{cases} \mathbf{p}_i(k-1), & \text{if } f\big(\mathbf{p}_i(k-1)\big) \le f(\mathbf{x}_i(k)) \\ \mathbf{x}_i(k), & \text{if } f\big(\mathbf{p}_i(k-1)\big) > f(\mathbf{x}_i(k)) \end{cases} \tag{14}$$

$$\mathbf{g}(k) = \operatorname{argmin}\big\{f\big(\mathbf{p}_1(k)\big), \ldots, f\big(\mathbf{p}_s(k)\big)\big\}, \tag{15}$$

where $d \in \{1, 2, \ldots, \lambda\}$ is the dimension of the ith particle with $i \in \{1, 2, \ldots, s\}$, $v(k)$ and $x(k)$ represent the velocity and position of the particle in the iteration $k \in \{1, 2, \ldots, iter_{\max}\}$, c_1 and c_2 are the social and cognitive acceleration coefficients, $r_1(k)$ and $r_2(k)$ are a pair of random numbers uniformly distributed in the interval [0, 1] representing the stochastic element of any swarm, and f is the fitness function to be optimised.

However, it is not practical to use the basic version of PSO because stability and convergence of the algorithm are strongly limited. Therefore, shortly after its creation the two variants of PSO most widely known and used because of their simplicity of implementation and computational complexity are formulated: (1) PSO with inertia weight (PSOiw) (Shi and Eberhart 1998) and (2) PSO with constriction factor (PSOcf) (Clerc 1999).

PSOiw incorporates the parameter ω (inertia weight) in the velocity Equation (12), so that a new equation for speed is obtained as

$$v_{i,d}(k+1) = \omega v_{i,d}(k) + c_1 r_1(k)\big(p_{i,d}(k) - x_{i,d}(k)\big) + c_2 r_2(k)\big(g_d(k) - x_{i,d}(k)\big), \quad (16)$$

with $\omega \in [0,1]$ that serves to limit the particle velocities, and consequently achieve convergence to an equilibrium point. For its part, PSOcf incorporates the term χ called the factor of constriction, which operates similarly to the inertia weight but multiplying the entire right-hand side of (12), rather than only $v_{i,d}(k)$. Nevertheless, in Eberhart and Shi (2000) it is shown that both versions of the algorithm are equivalent. Although PSOcf and PSOiw are the first and best known variants of original PSO, many other variations and/or hybrids of PSO (e.g. Reyes-Sierra and Coello 2006; Del Valle et al. 2008; Kameyama 2009; Oca, Sttzle, Birattari, and Dorigo 2009) have emerged as the result of the incorporation of various approaches to technique, each of which eventually may come to have an outstanding performance in some search spaces but do poorly in others. Because of this, in Bratton and Kennedy (2007) the possibility of establishing a technical standard by defining a baseline is raised for issues including topology, equations of speed upgrade, size and initialisation of the swarm and edge conditions, providing a means of comparison in future developments and improvements.

Although PSO has been shown to be effective and efficient in many practical problems of diverse nature (Poli 2008), PSO benefits are not only the result of a detailed tuning of its parameters, but also of the choice of an appropriate fitness function to qualify the benefits of potential solutions. As an heuristic technique, PSO has the advantage of being able to drive most versatile fitness functions in the case of a deterministic technique, being able to deploy non-differentiable, nonlinear and/or discontinuous functions without any problems.

Finally, it should be noted that PSO in its different variants and/or modifications is a very good alternative solution, compared with techniques such as genetic algorithms (GA) (Rahmat-Samii 2003; Habib and Al-kazemi 2005) and differential evolution (DE) (Dong 2009; Semnani, Kamyab, and Rekanos 2009), to global optimisation problems with multiple maximum/minimum, discontinuities and deterministic solutions in no-polynomial time. This can also be reflected in the increase (close to exponential) of successful applications based on PSO (Poli 2008).

3. New PSO-based methodology for finding CQLFs

The design process for the new CQLF calculation methodology proposed in this article is introduced in this section, which involves the next three steps:

(1) Design suitable fitness function(s) to optimise with PSO.
(2) Define how particles represent potential feasible solutions of the problem to be solved.
(3) Establish appropriate implementation for good performance of PSO. This includes: setting PSO parameters, debugging in programming to minimise processing times and identifying best initial conditions, among others.

The first two points of the above-mentioned method are explicated in the following subsections, with the purpose of describing the theoretical steps involved in developing the new methodology for finding a CQLF for a set of stable matrices $\mathrm{A} = \{A_1, \ldots, A_N\} \in \Re^{n \times n}$ based in PSO.

3.1 *Fitness functional*

Based on the second functional presented in Liberzon and Tempo (2004), which is here called f_g and defined as

$$f_g(R) := \max\{\mathrm{eig}(RA_i + A_i^T R)\} \quad (17)$$

with

$$\partial_P v(P, A_i) = A_i x x^T + x x^T A_i^T, \quad (18)$$

a generic functional $f(R)$ is proposed. However, this proposed functional is not used in the same way as f_g (minimise with respect to one matrix and iterate the process for the other matrices), but permits the minimisation with respect to all matrices in each iteration. Additionally, two specific functionals $f_i(R)$ are defined beginning with a pair of given representations for a potential feasible solution R.

Without loss of generality, let R be a symmetric matrix in $\Re^{n \times n}$ candidate to solve simultaneously the N Lyapunov equations generated by N systems (see (4)),

i.e. candidate to be a CQLF. Then the generic functional is defined as

$$f(R) := \max_i\{\text{eig}(RA_i + A_i^T R)\}, \quad \forall i = 1, \ldots, N, \tag{19}$$

with eig(·) the function that gives the eigenvalues of a matrix. It is clear that although R is a symmetric matrix, it does not imply that it is a positive definite matrix, which is an essential prerequisite for R to produce a CQLF of the form $V(x) = x^T R x$. For this reason, the search space must be restricted to positive definite matrices in order to improve the convergence of the optimisation process, by not having to evaluate unfeasible solutions.

One way to restrict the search space is based on the representation $R = V^{-1} D V$, where V is the matrix that diagonalises R, and D a diagonal matrix similar to R with entries $d_1, \ldots, d_n$, i.e. $D = \text{diag}(d_1, \ldots, d_n)$. Then (19) is used but unfeasible solutions are repaired through the projection

$$\tilde{R} = V^{-1} \tilde{D} V, \tag{20}$$

with

$$\tilde{D} = \text{diag}(|d_1|, \ldots, |d_n|), \tag{21}$$

and $d_1, \ldots, d_n$ obtained from D. Considering a diagonalisable R is reasonable since R is symmetric and this kind of matrix has that property (Horn and Johnson 1985). Note that (20) is a modified version of the matrix projection on the cone of non-negative definite matrices presented in Liberzon and Tempo (2004), having that for a symmetric matrix R, $V^{-1} = V^T$ is valid, and using $\tilde{D}$ instead of the diagonal matrix $D^+ = \text{diag}(\max(d_1, 0), \ldots, \max(d_n, 0))$. With this projection, we can define from generic fitness (19) a first specific fitness

$$f_1 := f(\tilde{R}_1) \tag{22}$$

with

$$\tilde{R}_1 = V^{-1} \tilde{D} V \tag{23}$$

and D, V obtained from a symmetric matrix $R_1 = V^{-1} D V$ to be evolved.

A second way to restrict the search space of a generic functional (19) is obtained from Proposition 3.1 stated below, which is an alternate and original equivalent of the CQLF problem.

Proposition 3.1: *Let* (1) *be a switched system with* $x \in \Re^n$, *and let A be a set of Hurwitz matrices in* $\Re^{n \times n}$. *Then, the existence of a CQLF for A is equivalent to the existence of an upper* (*lower*) *triangular matrix* $B \in \Re^{n \times n}$, *such that*

$$B^{-1} A_i^T B < 0, \quad \forall i \in \{1, 2, \ldots, N\}. \tag{24}$$

Note that B is a common similarity transformation for all A_i, *and the CQLF is given by* $P = BB^T$.

Proof:

Part 1: First, the implication $B \rightarrow$ CQLF will be shown. Suppose that (24) is fulfilled. By extension (Lemma 2.1), it is also true that

$$B^{-1} A_i^T B + \left(B^{-1} A_i^T B\right)^T < 0, \quad \forall i \in \{1, \ldots, N\},$$

and hence

$$A_i^T B B^T + B B^T A_i < 0, \quad \forall i \in \{1, \ldots, N\}. \tag{25}$$

By defining $P = BB^T > 0$, inequality (4) is obtained, and thus satisfies the conditions so that P is a CQLF.

Part 2: Now the implication CQLF $\rightarrow B$ will be shown. Assume that there exists a CQLF P for A, and without loss of generality, assume $P = P^T$. Since $P > 0$, by Cholesky factorisation (Horn and Johnson 1985) $P = BB^T$ is obtained for some upper (lower) triangular matrix B, and thereby satisfies (25). Finally, after a reverse procedure to that submitted in Part 1 (right multiplying by the term B^{-T} and using Lemma 2.1), (24) is reached. □

Thus, based on Proposition 3.1 and set from generic fitness (19), we have the second specific fitness

$$f_2 := f(\tilde{R}_2) \tag{26}$$

with

$$\tilde{R}_2 = R_2 R_2^T, \tag{27}$$

where R_2 is an upper (lower) triangular matrix to be evolved in $\Re^{n \times n}$.

After the fitness functions to be used are defined, we proceeded to design how the particles would represent the different matrices to evolve, i.e. R_1 and R_2.

3.2 *Particles representation*

Since two fitness functionals will be used, the design of two ways in which the s particles represent matrices $R_{1,2}^{(i)}$ $i = 1, \ldots, s$, (and consequently $\tilde{R}_{1,2}^{(i)}$ through (21) and (27)) to evolve is needed. Initially, it is convenient to redefine the position of the particles as

$$\mathbf{x}_i = \left[C_j^{(i)}\right], \quad j = 1, 2, \ldots, \lambda, \quad i = 1, 2, \ldots, s. \tag{28}$$

Up to this point it seemed intuitive to relate each component j of the particles to each element of $R_1^{(i)}$ or

$R_2^{(i)}$, but this is not very helpful for the optimisation process due to the nature of the CQLF problem: if a matrix P is an arbitrary CQLF, then αP is also CQLF for all $\alpha > 0$ (see Equation (4)). Then again it is necessary to restrict the search space limiting the norm of vectors $[C_j^{(i)}]$. This requires defining a vector $M_i = [m_k^{(i)}]$ for $k = 1, 2, \ldots, n_p$, with

$$m_k^{(i)} = \begin{cases} \sin\left(C_1^{(i)}\right), & k = 1. \\ \sin\left(C_k^{(i)}\right) \prod_{h=1}^{k-1} \cos\left(C_h^{(i)}\right), & k = 2, 3, \ldots, n_p - 1, \\ \prod_{h=1}^{2N-1} \cos\left(C_h^{(i)}\right), & k = n_p, \end{cases} \tag{29}$$

so that n_p is the freedom degree of $R_1^{(i)}$ or $R_2^{(i)}$ as appropriate, imposing $\lambda = n_p - 1$. It is now necessary to define a pair of functions $g_{1,2}$: $\Re^{n_p} \longrightarrow \Re^{n \times n}$ so that $R_{1,2}^{(i)} = g_{1,2}(M_i)$ are the matrices that will be evolved with PSO, in order to match $f_{1,2}$: $\Re^{n \times n} \to \Re$. However, both $R_1^{(i)}$ and $R_2^{(i)}$ have $n_p = \frac{n(n+1)}{2}$, since $R_1^{(i)}$ are symmetric matrices in $\Re^{n \times n}$ and $R_2^{(i)}$ are upper (lower) triangular matrices in $\Re^{n \times n}$. Then, matrices $R_1^{(i)}$ used with f_1 are defined by

$$R_1^{(i)} = \begin{bmatrix} m_1^{(i)} & m_2^{(i)} & m_3^{(i)} & \cdots & m_n^{(i)} \\ m_2^{(i)} & m_{n+1}^{(i)} & m_{n+2}^{(i)} & \cdots & m_{2n-1}^{(i)} \\ m_3^{(i)} & m_{n+2}^{(i)} & m_{2n}^{(i)} & \cdots & m_{3n-3}^{(i)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m_n^{(i)} & m_{2n-1}^{(i)} & m_{3n-3}^{(i)} & \cdots & m_{n_p}^{(i)} \end{bmatrix}, \tag{30}$$

and without loss of generality $R_2^{(i)}$ matrices used with f_2 are defined by

$$R_2^{(i)} = \begin{bmatrix} m_1^{(i)} & m_2^{(i)} & m_3^{(i)} & \cdots & m_n^{(i)} \\ 0 & m_{n+1}^{(i)} & m(i)_{n+2} & \cdots & m_{2n-1}^{(i)} \\ 0 & 0 & m_{2n}^{(i)} & \cdots & m_{3n-3}^{(i)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & m_{n_p}^{(i)} \end{bmatrix}. \tag{31}$$

Finally, since $\tilde{R}_{1,2}^{(i)}$, $R_{1,2}^{(i)}$, $\mathbf{x}_i$ and M_i represent the same element, it follows that $f_{1,2}(R_{1,2}^{(i)}) \doteq f_{1,2}(\mathbf{x}_i)$. In this way, and once the analytical part of the design was defined, the stage of implementation and performance analysis was then faced.

4. Performance analysis of the proposed methodology

In this section the implementation of the PSO-based methodology for finding CQLFs is shown, and results of tests to evaluate the performance of the proposed methodology are presented. Among other issues, the implementation involved the development of two main programs in Matlab to define f_1 and f_2 as designed in Section 3. The PSO ToolBox (a set of Matlab files implementing the PSO algorithm for systems optimisation) developed by Jagatpreet Singh PSO (Singh 2003) was used as a software tool. Programs developed required the implementation of an auxiliary file in which the corresponding fitness functions are defined, and an amendment to the PSO Toolbox main file (PSO.m) to define the initial population appropriately.

The following subsections present PSOiw settings to use in the validation tests, and a set of experimental results used in comparisons with other methodologies is shown.

4.1 *Choice of PSO parameters*

For practical purposes related to the analysis of the applicability of PSO to the problem of finding a CQLF, it was decided to use one of the simplest versions of PSO: the PSOiw version. Moreover, the linearly decreasing variation with respect to the progress of the iterations was chosen among the different variation schemes of $\omega(k)$ (Hu, Xu, Wang, and Xu 2009), because it is a scheme typically used (Del Valle et al. 2008) and defined by

$$\omega(k) = \omega_{\max} - [\omega_{\max} - \omega_{\min}] \frac{k}{iter_{\max}} \tag{32}$$

with $\omega_{\max} = 0.9$, $\omega_{\min} = 0.4$ and a maximum number of iterations equal to $iter_{\max}$. It should be noted that the advantage of using the linearly decreasing inertia weight scheme is offering a balance between exploration and exploitation, for which $c_1 = c_2 = 2$ (Del Valle et al. 2008) are also commonly chosen as acceleration constants.

In relation to the number of particles to be used, it is known that the proper choice is directly related to the complexity of the optimisation problem that seeks to solve, complexity that is usually determined by the size of the particles or the definition of fitness functional. Choosing the size s of the population to evolve in an automatic way (Particle Swarm Central 2007) is presented, which depends on the size of the particles and is calculated as

$$s = 10 + 2\sqrt{\lambda}, \tag{33}$$

which was used by default for the tests developed in this work, with λ defined by

$$\lambda = \frac{n(n+1)}{2} - 1, \tag{34}$$

n being the order of the matrices A_i. As a termination criterion, it was decided to operate two simultaneously: (1) the crossing of a threshold defined by $f(\mathbf{x}_i) < 0$ and (2) reaching a maximum number of iterations defined by default for $iter_{\max} = 200$.

Test data used are not from databases, but are from sets of matrices that *a priori* are known to share a CQLF, such as commuting matrices (Narendra and Balakrishnan 1994), and triangular matrices (Shorten and Narendra 1998), among others. Finally, the initialization of $\mathbf{x}_i$ may be chosen randomly (uniform distribution) or predefined (solving s individual Lyapunov equations).

4.2 *Comparative analysis: computing time for the first feasible solution*

In the comparative analysis of the time of calculation, NB and EB methods are not taken into account because they are deterministic algorithms that use a fixed time (number of iterations) to find the solution. Besides this, it makes no sense to use the methodology NB in the general case since it does not always meet the conditions for its use. Moreover, although the EB methodology can be applied to the general case without any problem, this is a methodology that analytically becomes unable to find a CQLF although it exists, which is reflected in a quite low success rate. Finally, as in Liberzon and Tempo (2004) in which a comparison of gradient with the LMI approach and demonstration of the advantages of gradient over LMI is made, this article will compare the proposed methodology (based on PSO) with respect to the methodology of Liberzon and Tempo (2004) based on gradient technique.

As a first set of comparative testing, each methodology is validated in groups of upper triangular matrices (that are known to share a CQLF (Shorten and Narendra 1998)) of orders 2, 3, 4 and 5. Figure 1 shows the results of these tests.

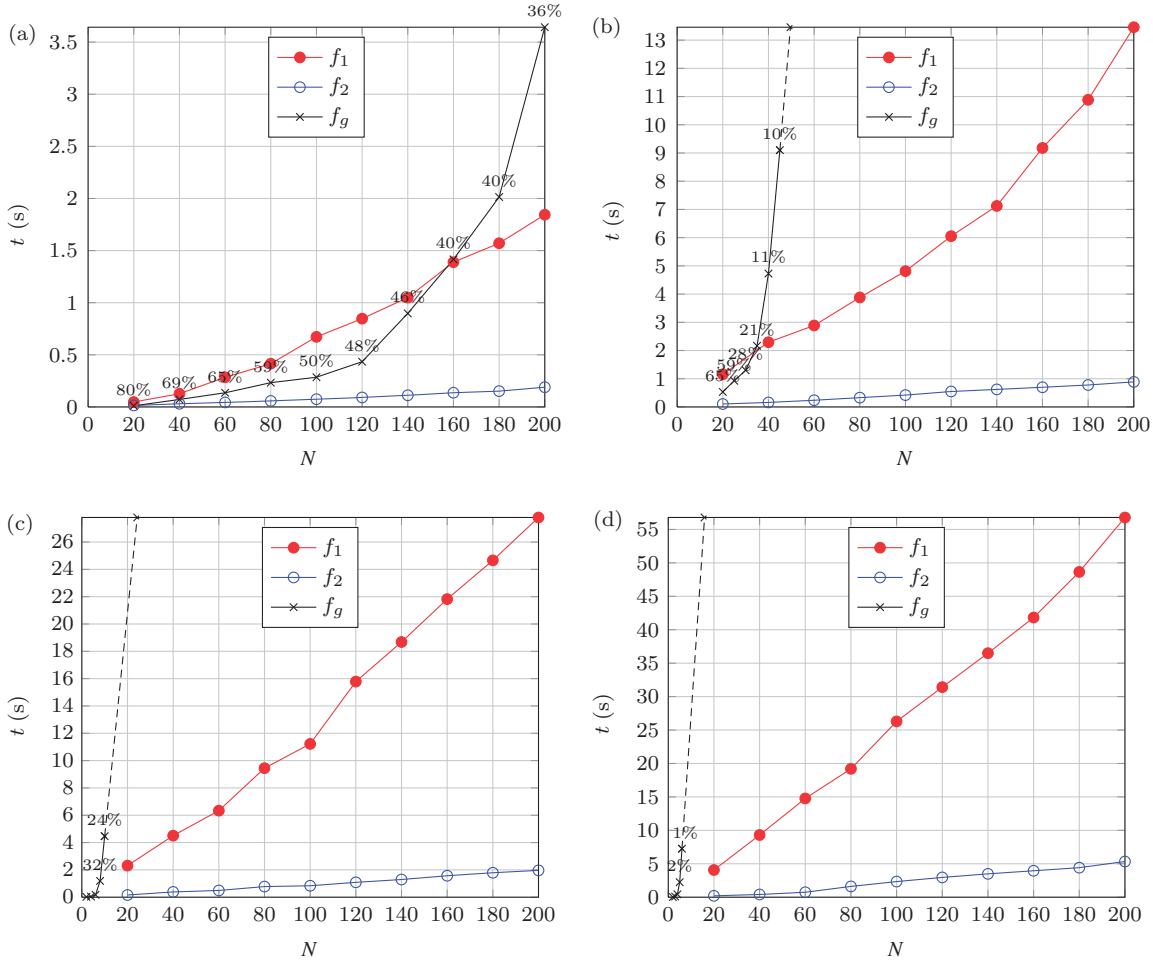

Figure 1. Average computing time up to the first feasible solution for sets of N upper triangular matrices. (a) Matrices of 2×2, (b) matrices of 3×3, (c) matrices of 4×4 and (d) matrices of 5×5. (Each mark: average of 20 measurements; dashed lines: projections because of complexity in collecting data; percentages: success rates; no-percentages: 100% success rates; initial conditions for PSO: Random).

Two kinds of conclusions can be extracted from Figure 1: the first comes from a comparison of f_1 and f_2, and the second comparison is between the PSO and gradient methodologies. With respect to the first analysis, a marked improvement in the computing time and success rates of f_2 on f_1 is shown in Figure 1, which is supported by the type of search space that is used. While the compensation in f_1 (see (20)) projects potential solutions on a search space restricted to positive definite matrices, f_2 always manages potential solutions within that space, without having to resort to any kind of repair. Another important aspect to be highlighted is the linear trend in the consumption of time taken to compute an initial feasible solution versus the number of matrices analysed, with a steeper slope for f_1 than for f_2.

As for the comparison between the methodologies, the fact that f_2 is generally much more efficient than gradient is highlighted, but not so with f_1 that is more efficient than gradient only after a certain number of matrices. These aspects are justifiable because gradient exhibits a roughly exponential trend in the consumption of time used up to the first feasible solution versus the number of matrices, and the previously mentioned linear trend with PSO reaches a point where the time consumed by a gradient is greater than one consumed by f_1.

An important element to be noticed is that for each realisation of PSO the same configuration (same set of parameters) was used, which was not so for the case of the gradient, because every time the number of matrices to analyse was increased, a new scan to determine the best experimental values of the parameters r and α to be used was needed. The consequence is that a larger number of matrices needs a smaller value of r and α, which directly slows down the optimisation process and gives the characteristic exponential curve to the relationship computation time versus the number of matrices. However, it is clear that each PSO setting must have an upper bound of effectiveness, but expandable by increasing $iter_{\max}$ and/or s.

Although PSO shows a marked advantage over gradient using triangular matrices, it is interesting to observe the behaviour of the proposed methodology on other kinds of matrices, preferring to use f_2 because of the advantages described with respect to f_1. Thus tests on other sets of matrices also known *a priori* to share a CQLF were run: commuting matrices (Narendra and Balakrishnan 1994), simultaneously triangularisable matrices (Ibeas and De la Sen 2009), negative definite matrices (Shorten, Wirth, Mason, Wulff, and King 2007), diagonal matrices (are commuting and negative definite) and a set of generic matrices obtained from an arbitrary symmetric positive definite matrix P and a random search looking for stable matrices for which P solves their individual Lyapunov Equation (4). The results of applying both methods in these four kinds of matrices are shown in Figure 2.

By analysing Figure 2, it is interesting to note the fact that PSO does not always have superior performance than gradient. Indeed, we find that PSO has better performance than gradient in the case of diagonal and simultaneously triangularisable matrices, which is not so in the case of generic matrices and matrices that commute.

Another interesting aspect to analyse is the success rate (percentage of success in seeking feasible solutions) of methodologies. While gradient ensures obtaining a CQLF in a given number of steps for a given optimal values for r and α, such optimal values are unknown and consequently each gradient settings has a certain success rate. It was found in tests that increasing the number of matrices analysed with gradient requires a detailed reconfiguration of its parameters, for maintaining at least the same rate of effectiveness. By using PSO the latter usually is not required, and the same initial configuration allows to maintain high success rates even in large sets of matrices. Unlike gradient, PSO convergence is not analytically assured, yet experiments showed that it is much easier to get a proper tuning of its two parameters, the population size s and the maximum number of iterations $iter_{\max}$, through which feasible solutions are obtained. It is important to note that although PSO has many more tuning parameters, the choice of variables such as c_1, c_2 and $w(k)$ is supported by previous researches for general use (Del Valle et al. 2008), but $iter_{\max}$ and s are problem-dependent parameters.

However, beyond the partial conclusion on the effective applicability of PSO to the calculation of a CQLF, it is interesting to analyse a factor which is much more important: the robustness of the solutions obtained by PSO (specifically f_2) over other methodologies. Therefore, the next section is devoted to this issue.

4.3 *Comparative analysis: robustness of the best solution*

The purpose of this analysis is to obtain both quantitative and qualitative measures of robustness for the best solutions (CQLFs) obtained by different methodologies. For this type of analysis it is necessary to note that graphical comparison is possible only for sets of second-order matrices, because of the freedom degrees of the solutions. Therefore, the comparison for sets of matrices of order greater than 2 is only possible

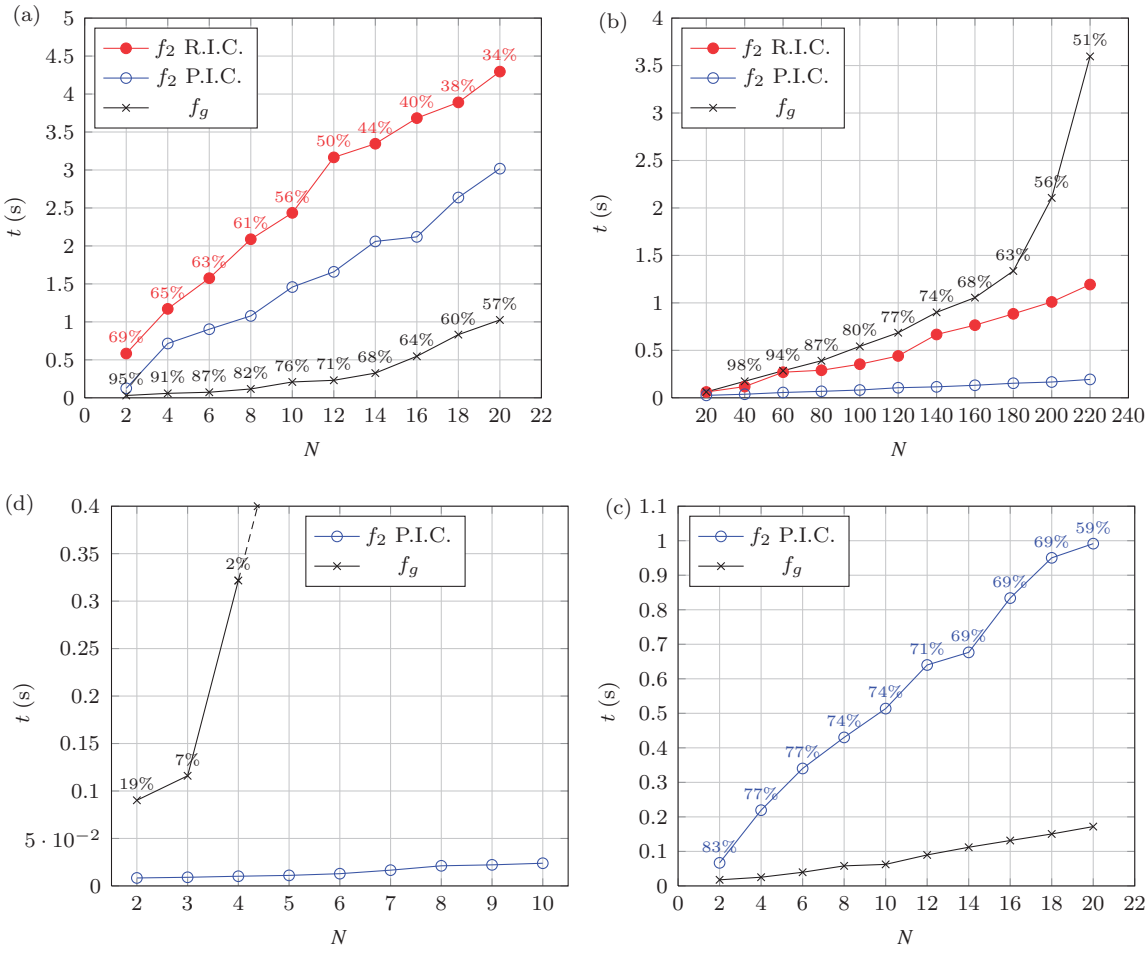

Figure 2. Average computing time up to the first feasible solution for other kind sets of N matrices in $\Re^{5\times5}$. (a) Commuting matrice, (b) diagonal matrices, (c) commonly triangularizable matrices and (d) generic matrices. (Each mark: average of 20 measurements; dashed lines: projections because of complexity in collecting data; percentages: success rates; no-percentages: 100% success rates; R.I.C.: Random Initial Conditions; P.I.C.: Predefined Initial Conditions).

by defining a performance index. This index is defined here as the maximum percentage of change that any element of a matrix may endure, so while a previously calculated CQLF remains despite such variations.

As a base experiment, finding a CQLF for the same set of three matrices by using different methodologies is proposed, and then evaluating the tolerance of the best solution found for a possible parametric variation in some parameters of the matrices under analysis (matrices given as defined in the previous section, with the analytical or experimental support that a CQLF is shared). The methodologies chosen to compare are: gradient (f_g) LMI, EB Algorithm, PSO (f_2), and as a special case the NB Algorithm although it is not applicable in general.

4.3.1 *Case Order = 2: graphical analysis*

The fact that the P matrix associated with a second-order CQLF can be assumed to be symmetric without loss of generality, permits that areas of feasible solutions for a second-order system can be represented as an ellipsoid cap on a sphere of radio equal to one (also without loss of generality). Thus, to the extent that two caps associated with two systems (matrices) have an intersection, it follows that these two systems share a CQLF area, and this fact can be extended directly to a set of N matrices.

A set of graphical results as outlined above, obtained from the best solution calculated with each methodology is shown in Figure 3, in which the following matrices were used:

(1) Commuting matrices

$$A \approx \begin{bmatrix} -5.3788 & 4.5974 \\ -0.7009 & -1.4067 \end{bmatrix} \quad B \approx \begin{bmatrix} -6.9024 & -2.4484 \\ 0.37327 & -9.0178 \end{bmatrix}$$

$$C \approx \begin{bmatrix} -0.7287 & -7.6940 \\ 1.1730 & -7.3760 \end{bmatrix},$$

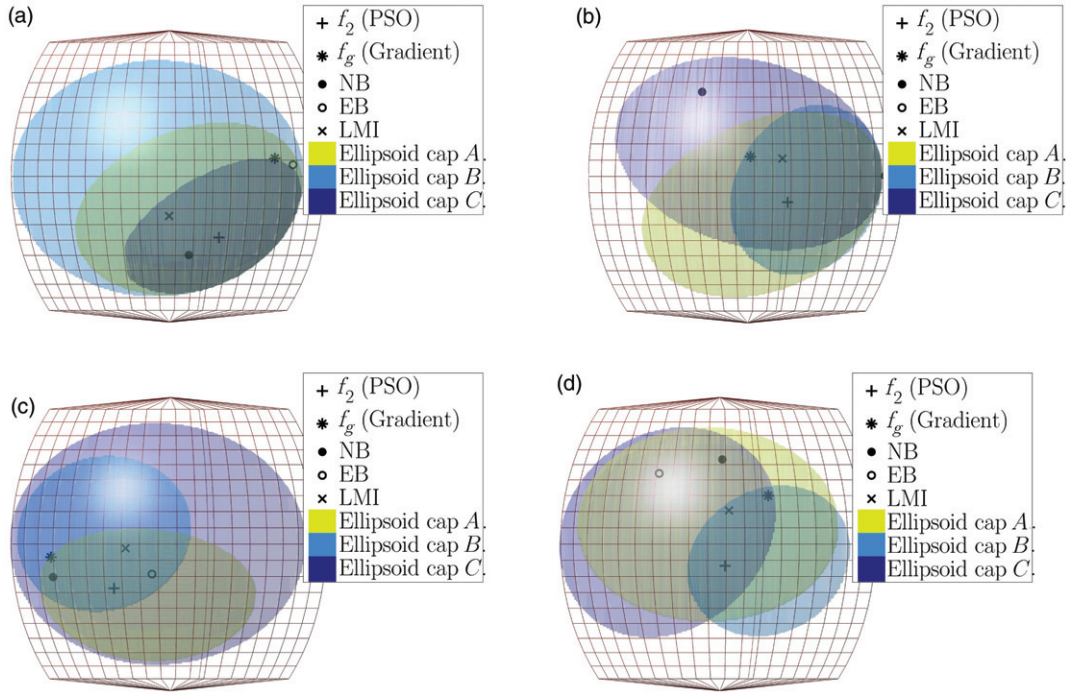

Figure 3. Graphic comparison for CQLF calculation in a set of three matrices of second-order that share a CQLF. (a) Commuting matrices, (b) triangular matrices, (c) negative definite matrices and (d) generic matrices.

(2) Triangular matrices

$$A = \begin{bmatrix} -3.74 & -10.06 \\ 0 & -13.71 \end{bmatrix} \quad B = \begin{bmatrix} -15.53 & 25.43 \\ 0 & -9.90 \end{bmatrix}$$
$$C = \begin{bmatrix} -5.37 & 10.98 \\ 0 & -21.21 \end{bmatrix},$$

(3) Definite negative matrices

$$A = \begin{bmatrix} -9.62 & 5.65 \\ -6.88 & -0.73 \end{bmatrix} \quad B = \begin{bmatrix} -4.48 & 3.22 \\ -11.17 & -6.39 \end{bmatrix}$$
$$C = \begin{bmatrix} -10.83 & 1.61 \\ 1.05 & -11.76 \end{bmatrix},$$

(4) Generic matrices

$$A = \begin{bmatrix} -9.83 & -5.56 \\ 4.85 & -4.21 \end{bmatrix} \quad B = \begin{bmatrix} -2.47 & -7.70 \\ 1.47 & -4.45 \end{bmatrix}$$
$$C = \begin{bmatrix} -8.69 & 1.16 \\ 8.84 & -5.27 \end{bmatrix}.$$

In Figure 3, each ellipsoid cap represents the space of feasible solutions to an individual Lyapunov equation for each system. Therefore, through the intersection of areas it can easily be verified that matrices under analysis share a CQLF, and likewise the area of feasible solutions to the CQLF problem can be define. Although Figure 3 shows the result of specific realisations for a particular set of matrices, an approach to the robustness of the solutions obtained with different methodologies compared can be made. To begin with, it is clearly seen that as a solution is further away from the edges of the CQLF area, it will be more robust to parametric variations on the matrices analysed, variations that make the caps change their shape, orientation and location. By analysing the location of the best PSO solution obtained, it is seen that it is located towards what might be called the centre of mass of the CQLF area, unlike what happens with the other solutions in which the solution is usually located near the edges of that area. Therefore, the solutions obtained by PSO are more robust compared to other methods for parametric variations in the matrices tested (which can be checked graphically) because a small change in some matrix can reduce the CQLF area causing that a feasible solution near the edges ceases to be.

4.3.2 *Case order > 2: performance index analysis*

For this case a performance index is defined, to quantify the level of parametric variation that may occur in a parameter of the matrices for which a CQLF was calculated, so that such CQLF remains as such despite such variations. The two maximum possible variation, defined as Var_+ and Var_-, are the maximum/minimum absolute percentage of change that

can be added/subtracted to or from the parameter with respect to its original magnitude, i.e.

$$\mathrm{Var}_{\pm}(a_{i,j}, P) = \left\{ \underset{N_{\%}}{\mathrm{Max}} \left(a_{i,j} \pm N_{\%} abs(a_{i,j}) \right) | P \text{ remains } CQLF \right\}. \tag{35}$$

As an illustrative example, a group of three stable matrices in $\Re^{4\times4}$ was chosen, consisting of the following randomly calculated matrices with generic structure:

$$A = \begin{bmatrix} -1 & -3 & 2 & 0 \\ -1.8 & 0 & -0.9 & 2 \\ -1 & 6 & -5 & 1 \\ 4.8 & -3.2 & 2.9 & -5.1 \end{bmatrix}$$

$$B = \begin{bmatrix} -1.65 & -4 & 2.85 & -0.3 \\ -3 & 0.15 & -1.4 & 3.1 \\ -1.35 & 9.6 & -7.35 & 2.1 \\ 7.5 & -4.65 & 4.3 & -7.65 \end{bmatrix}$$

$$C = \begin{bmatrix} -2.2 & -6.5 & 3.8 & 0.5 \\ -4 & 0.2 & -1.9 & 4.2 \\ -1.8 & 13 & -9.8 & 3 \\ 10 & -6.2 & -5.8 & -10.2 \end{bmatrix}.$$

Using f_g (gradient-based methodology) and f_2 with predefined initial conditions (PSO-based methodology), it was possible to achieve two types of Lyapunov functions, which depend on whether the P matrix is associated with the first feasible solution (P^{first}) or the best feasible solution (P^{best}). The respective Lyapunov functions obtained for the above set matrices are:

$$P_G^{\text{first}} = \begin{bmatrix} 0.2919 & -0.1773 & 0.1365 & -0.1944 \\ -0.1773 & 0.6055 & -0.3181 & 0.2619 \\ 0.1365 & -0.3181 & 0.3613 & -0.2020 \\ -0.1944 & 0.2619 & -0.2020 & 0.3452 \end{bmatrix}$$

$$P_G^{\text{best}} = \begin{bmatrix} 0.3090 & -0.1976 & 0.1550 & -0.1936 \\ -0.1976 & 0.5909 & -0.3173 & 0.2601 \\ 0.1550 & -0.3173 & 0.3501 & -0.2153 \\ -0.1936 & 0.2601 & -0.2153 & 0.3430 \end{bmatrix}$$

$$P_{PSO}^{\text{first}} = \begin{bmatrix} 0.8673 & 0.3741 & 0.0819 & -0.1429 \\ 0.3741 & 0.2415 & -0.0277 & 0.1046 \\ 0.0819 & -0.0277 & 0.0706 & 0.0336 \\ -0.1429 & 0.1046 & 0.0336 & 0.0681 \end{bmatrix}$$

$$P_{PSO}^{\text{best}} = \begin{bmatrix} 0.6452 & -0.1355 & 0.1545 & -0.1854 \\ -0.1355 & 0.5835 & -0.0752 & 0.1015 \\ 0.1545 & -0.0752 & 0.1787 & 0.0915 \\ -0.1854 & 0.1015 & 0.0915 & 0.3322 \end{bmatrix}.$$

Table 1. Comparison of the robustness for first solution.

	P_G^{first} (%)		P_{PSO}^{first} (%)	
	Var_+	Var_-	Var_+	Var_-
$a_{1,1}$	**6**	**1733**	2.5	31
$a_{2,1}$	**55.6**	4.28	4.83	**27.22**
$a_{1,2}$	**99.7**	**3.33**	1.9	1.2
$b_{1,1}$	0.42	**1363**	**3.03**	50.3
$b_{2,1}$	**53.33**	13	0.2	**23.33**
$b_{1,2}$	**89.58**	0.19	7.08	**0.625**
$c_{1,1}$	**2.73**	**1318**	1.68	55.09
$c_{2,1}$	**55**	1.23	8.25	**25**
$c_{1,2}$	**87.69**	**1.07**	8.2	0.41

Table 2. Comparison of the robustness for best solution.

	P_G^{best} (%)		P_{PSO}^{best} (%)	
	Var_+	Var_-	Var_+	Var_-
$a_{1,1}$	16.7	**1935**	**70**	1022
$a_{2,1}$	73.44	9.77	**94.44**	**38.89**
$a_{1,2}$	**86.67**	10.07	73.6	**21.67**
$b_{1,1}$	7.27	**1575**	**51.51**	799.39
$b_{2,1}$	69.33	3.27	**70**	**25**
$b_{1,2}$	**83.33**	2.9	63.46	**12.5**
$c_{1,1}$	9.11	**795.45**	**50**	792.18
$c_{2,1}$	69.2	3.98	**70**	**22.5**
$c_{1,2}$	**80**	3.37	60.92	**11.54**

The analysis of the robustness for P_G^{first}, P_{PSO}^{first}, P_G^{best} and P_{PSO}^{best} against parametric variations of certain parameters of A, B and C was developed using the performance index defined by (35), and the results are shown in Tables 1 and 2, where the boldface values indicate for which method the maximum upper limit (Var_+) and the minimum lower limit (Var_-) for parameter variation are obtained.

When analysing Tables 1 and 2 two important aspects are noticed. The first aspect is if a CQLF calculated from the first feasible solution is used (Table 1), the gradient-based method offers better results in terms of robustness than the PSO-based method. However, usually one of the two indexes of performance is very low when using gradient and thus eventually PSO becomes better. The second aspect is that when using a CQLF calculated from the best feasible solution (Table 2), it is seen that the PSO-based method provides more robust solutions compared to gradient. It should also be noted that when PSO does not improve the index with respect to gradient, it is because the latter offered rates well above those obtained with PSO.

4.4 *Convergence analysis*

While the functional (19) is convex for each individual matrix A_i, this is no longer true when all matrices A_i are analysed simultaneously. The feasible solution space is the intersection of convex spaces associated to each individual solution (and therefore convex), but the search space is the union of the individual search spaces, making the functional (19) not convex when considering multiple minima. Moreover, the complexity of the search space increases as the number of matrices to be analysed increases.

Since PSO has a non-zero probability of getting stuck at a point that could not even be a local minimum, the convergence to the feasible solution space cannot always be assured (Jiang, Luo, and Yang 2007). However, that probability is very small, and with a good exploration and exploitation characteristics obtained through a suitable choice of the PSO settings, high success rates with moderate execution times can be achieved.

Nevertheless, there is a way to assure convergence to the feasible solution space with probability 1, but this is achieved in detriment of the computation time demanded by the methodology. The idea is to employ a modified version of PSO proposed by Van den Bergh (2002), called guaranteed convergence PSO (GCPSO) because it is used to get guaranteed local convergence, and once the local minimum is attained to start a new search, storing the best solution ever reached and randomly reinitialising the swarm. Using this modified version of PSO, there is an increase in computing time due to the incorporation of more updating equations (with more tuning parameters specifics of GCPSO) and the linkage of the new searches, but the convergence is always assured.

5. Conclusions

In this article a new methodology for finding a CQLF based on a functional proposed by Liberzon and Tempo, but using the PSO technique instead of gradient as a tool for optimising, is presented. One of the main conclusions reached is that PSO is a successful technique for finding a CQLF with comparative advantages over other techniques including LMI, gradient, NB and EB algorithms. Advantages obtained are related to the computing time of a CQLF with for certain types of matrices, and the robustness of the CQLF with respect to parametric variations. Although it was shown that the computing time can be improved in large proportions in certain cases by using PSO, the robustness of the best solution obtained with each methodology under certain configurations regardless of the time necessary to obtain it was also analysed. In light of this analysis, it was possible to conclude that a PSO-based methodology has a tendency to find CQLFs that are a more robust solution (with respect to parameter variations of the matrices under analysis) than gradient-based methodology, both qualitatively and quantitatively, making the proposed methodology one that offers more reliability in ensuring stability of a switched system through the CQLF approach.

As for limitations, two points of interest that affect the effectiveness and efficiency of the proposed methodology were detected: (1) the nature of the matrices under analysis and (2) the dimensionality of the problem. For the first limitation, by analysing triangular and diagonal matrices, and matrices with properties such as simultaneous triangularisation/diagonalisation, the PSO-based methodology showed significant advantages over gradient-based methodology, but this was not so in the case of commuting matrices, generic matrices or defined negative matrices. However, it is important to note that precisely in cases where PSO is not efficient, the gradient is, and therefore it must be said that the two methodologies are complementary rather than exclusionary. Regarding the dimensionality of the problem, the increase in the order of the systems is the variable that most affects the PSO-based methodology, making the calculation time increase greatly because, among other things, the size of the population depends directly on that variable. For its part, the increase in the number of systems to be analysed using PSO showed no effects as occurred in the case of gradient. An illustrative example is the case of triangular matrices, where the relationship of number-of-systems versus calculation-time is linear with a small slope when using PSO, but about exponential when using gradient. However, regardless of computing time, PSO achieves feasible solutions that appear to be more robust to parametric variations of the matrices analysed.

While two options for optimising the fitness functional with PSO were shown, other functional and/or representations of matrices could be chosen for improving convergence times, which is also affected by the type of initialisation of the population. It was found that the PSO search process is benefited by including in the initial conditions, for example, a particle with position representing the identity matrix (which is CQLF for every set of negative definite matrices), or a lot of solutions to individual Lyapunov equations. For gradient-based methodology, some improvements could also be added, such as making α and r time-variant parameters to enhance convergence. However, these additions are outside the scope of this article.

Finally, through the above results it can be established that the proposed methodology is a new alternative solution to the CQLF Problem, with boldly marked advantages and strengths over the methodologies most commonly used today.

Acknowledgements

The results reported in this article have been supported by CONICYT-CHILE through grant FONDECYT No. 1090208 and Programa de Financiamiento Basal 'Centro de Tecnología para la Minería' FB0809.

References

Boyd, S., El-Ghaoui, L., Feron, E., and Balakrishnan, V. (1994), *Linear Matrix Inequalities in System and Control Theory*, Volume 15 of SIAM Studies in Applied Mathematics, Philadelphia: SIAM.

Bratton, D., and Kennedy, J. (2007), 'Defining a Standard for Particle Swarm Optimisation', in *Proceedings of the IEEE Swarm Intelligence Symposium 2007 (SIS 2007)*, Honolulu, HI, April 1–5, pp. 120–127.

Chen, Z., and Gao, Y. (2007), 'The Computation of a Common Quadratic Lyapunov Function for a Linear Control System', *Journal of Information and Computing Science*, 2, 299–304.

Cheng, D., Hu, X., and Martin, C. (2006), 'On the Smallest Enclosing Balls', *Communications in Information and Systems*, 6, 137–160.

Cheng, D., Martin, C., and Xiang, J. (2000), 'An Algorithm for Common Quadratic Lyapunov Function', in *Proceedings of the 3rd World Congress on Intelligent Control and Automation* (Vol. 4), pp. 2965–2969.

Clerc, M. (1999), 'The Swarm and the Queen, Towards a Deterministic and Adaptive Particle Swarm Optimisation', in *Proceedings of 1999 Congress on Evolutionary Computation*, Washington DC, pp. 1951–1957.

Del Valle, Y., Venayagamoorthy, G.K., Mohagheghi, S., Hernandez, J.C., and Harley, R.G. (2008), 'Particle Swarm Optimisation: Basic Concepts, Variants and Applications in Power Systems', *IEEE Transactions on Evolutionary Computation*, 12, 171–195.

Dong, R. (2009), 'Differential Evolution Versus Particle Swarm Optimisation for PID Controller Design', in *Fifth International Conference on Natural Computation 2009 (ICNC '09)*, Tianjin, China, Auguest 14–16, pp. 236–240.

Eberhart, R.C., and Kennedy, J.F. (1995a), 'A New Optimiser using Particle Swarm Theory', in *Proceedings of International Symposium on Micro Machine and Human Science (MHS)*, Nagoya, Japan, pp. 39–43.

Eberhart, R.C., and Kennedy, J.F. (1995b), 'Particle Swarm Optimisation', *Proceedings of IEEE International Conference on Neural Networks (ICNN)* (Vol. 4), Piscataway, NJ, pp. 1942–1948.

Eberhart, R.C., and Shi, Y. (2000), 'Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimisation', in *Proceedings of the 2000 Congress on Evolutionary Computation-CEC00* (Vol. 1), La Jolla, CA, USA, pp. 84–88.

Habib, S.J., and Al-kazemi, B.S. (2005), 'Comparative Study Between the Internal Behaviour of GA and PSO Through Problem-specific Distance Functions', in *The 2005 IEEE Congress Evolutionary Computation* (Vol. 4), September 2–5, pp. 2190–2195.

Horn, R.A., and Johnson, C.R. (1985), *Matrix Analysis*, Cambridge (Cambridgeshire), NY: Cambridge University Press.

Hu, J.Z., Xu, J., Wang, J.Q., and Xu, T. (2009), 'Research on Particle Swarm Optimisation with Dynamic Inertia Weight', in *International Conference on Management and Service Science 2009 (MASS '09)*, September 20–22, Wuhan, China, pp. 1–4.

Ibeas, A., and De la Sen, M. (2009), 'Exponential Stability of Simultaneously Triangularizable Switched Systems with Explicit Calculation of a Common Lyapunov Function', *Applied Mathematics Letters*, 22, 1549–1555.

Jiang, M., Luo, Y.P., and Yang, S.Y. (2007), 'Stochastic Convergence Analysis and Parameter Selection of the Standard Particle Swarm Optimisation Algorithm', *Information Processing Letters*, 102, 8–16.

Kameyama, K. (2009), 'Particle Swarm Optimisation – A Survey', *IEICE Transactions on Information and Systems*, E92-D, 1354–1361.

Liberzon, D. (2003), *Switching in Systems and Control*, Systems & Control: Foundations & Applications Series, Boston, MA: Birkhuser.

Liberzon, D., and Tempo, R. (2004), 'Common Lyapunov Functions and Gradient Algorithms', *IEEE Transactions on Automatic Control*, 49, 990–994.

Lin, H., and Antsaklis, P.J. (2009), 'Stability and Stabilisability of Switched Linear Systems: A Survey of Recent Results', *IEEE Transactions on Automatic Control*, 54, 308–322.

Narendra, K.S., and Balakrishnan, J. (1994), 'A Common Lyapunov Function for Stable LTI Systems with Commuting A-matrices', *IEEE Transactions on Automatic Control*, 39, 2469–2471.

Nguyen, T.V., Mori, Y., Mori, T., and Kuroe, Y. (2003), 'QE Approach to Common Lyapunov Function Problem', *Journal of Japan Society for Symbolic and Algebraic Computation*, 10, 52–62.

Oca, M.A.M.D., Sttzle, T., Birattari, M., and Dorigo, M. (2009), 'Frankenstein's PSO: A Composite Particle Swarm Optimisation Algorithm', in *Proceedings of IEEE Transactions Evolutionary Computation*, pp. 1120–1132.

Particle Swarm Central (2007), Standard PSO 2007 (SPSO-2007), in Particle Swarm Central, Programs Section. [Online]. Available: http://www.particleswarm.info

Paul, A., Akar, M., Safonov, M.G., and Mitra, U. (2004), 'Necessary and Sufficient Conditions for Stability of a Class of Second-Order Switched Systems', in *Proceedings of the 2004 American Control Conference*, 30 June–2 July, pp. 4561–4562.

Poli, R. (2008), 'Analysis of the Publications on the Applications of Particle Swarm Optimisation', *Journal of Artificial Evolution and Applications*, 2008, 10pp.

Rahmat-Samii, Y. (2003), 'Genetic Algorithm (GA) and Particle Swarm Optimisation (PSO) in Engineering Electromagnetics', in *Proceedings of the 17th International Conference on Applied Electromagnetics and Communications (ICECom)*, Dubrovnik, Croatia, October 1–3, pp. 1–5.

Reyes-Sierra, M., and Coello, C.A.C. (2006), 'Multi-objective Particle Swarm Optimisers: A Survey of the State-of-the-Art', *International Journal of Computational Intelligence Research*, 2, 287–308.

Semnani, A., Kamyab, M., and Rekanos, I.T. (2009), 'Reconstruction of One-dimensional Dielectric Scatterers using Differential Evolution and Particle Swarm Optimisation', *IEEE Geoscience and Remote Sensing Letters*, 6, 671–675.

Shi, Y.H., and Eberhart, R.C. (1998), 'A Modified Particle Swarm Optimiser', in *IEEE International Conference on Evolutionary Computation*, Anchorage, AK, May 4–9, pp. 69–73.

Shorten, R.N., and Narendra, K.S. (1998), 'On the Stability and Existence of Common Lyapunov Functions for Stable Linear Switching Systems', in *Proceedings of the 37th Conference on Decision and Control* (Vol. 4), Tampa, FL, December 16–18, pp. 3723–3724.

Shorten, R.N., and Narendra, K.S. (2003), 'On Common Quadratic Lyapunov Functions for Pairs of Stable LTI Systems Whose System Matrices Are in Companion Form', *IEEE Transactions on Automatic Control*, 48, 618–621.

Shorten, R.N., Wirth, F., Mason, O., Wulff, K., and King, C. (2007), 'Stability Criteria for Switched and Hybrid Systems', *SIAM Review*, 49, 545–592.

Singh, J. (2003), 'The PSO ToolBox', http://sourceforge.net/projects/psotoolbox [accessed 2010-04-27].

Tempo, R., Calafiore, G., and Dabbene, F. (2004), *Randomized Algorithms for Analysis and Control of Uncertain Systems*, London, UK: Springer-Verlag.

Van den Bergh, F. (2002), 'An Analysis of Particle Swarm Optimisers', PhD Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa.

Yanami, H., and Anai, H. (2005), 'Development of SyNRAC', in *International Conference on Computational Science*, pp. 602–610.

Zhu, Y.H., Cheng, D.Z., and Qin, H.S. (2007), 'Constructing Common Quadratic Lyapunov Functions for a Class of Stable Matrices', *Acta Automatica Sinica*, 33, 202–204.