Comput Econ (2011) 37:375–410
DOI 10.1007/s10614-011-9257-z

A Numerical Toolbox to Solve *N*-Player Affine LQ Open-Loop Differential Games

Tomasz Michalak · Jacob Engwerda · Joseph Plasmans

Accepted: 14 January 2011 / Published online: 12 February 2011

Abstract We present an algorithm and a corresponding MATLAB numerical toolbox to solve any form of infinite-planning horizon affine linear quadratic open-loop differential games. By rewriting a specific application into the standard framework one can use the toolbox to calculate and verify the existence of both the open-loop non-cooperative Nash equilibrium (equilibria) and cooperative Pareto equilibrium (equilibria). In case there is more than one equilibrium for the non-cooperative case, the toolbox determines all solutions that can be implemented as a feedback strategy. Alternatively, the toolbox can apply a number of choice methods in order to discriminate between multiple equilibria. The user can predefine a set of coalition structures for which they would like to calculate the non-cooperative Nash solution(s). It is also possible to specify the relative importance of each player in any coalition structure. Furthermore, the toolbox offers plotting facilities as well as other options to analyse the outcome of the game. For instance, it is possible to disaggregate each player's total loss into its contributing elements. The toolbox is available as a freeware from the authors of this paper.

Keywords Linear quadratic differential games · Riccati equations · Cooperative games · Noncooperative games · Coalition structures · Numerical computation

T. Michalak (✉)
Institute of Informatics, University of Warsaw, ul. Banacha 2, 02-097 Warszawa, Poland
e-mail: tpm@mimuw.edu.pl

J. Engwerda · J. Plasmans
CentER, University of Tilburg, P.O. Box 90153, 5000 LE, Tilburg, The Netherlands
e-mail: J.C.Engwerda@uvt.nl

J. Plasmans
University of Antwerp, Prinsstraat 13, 2000 Antwerpen, Belgium

1 Introduction

Many situations in, for example, economics and management are characterised by multiple decision makers/players who can enforce decisions that have enduring consequences. This often invokes coordination problems and so a natural framework to analyze these kinds of problems involves dynamic game theoretical settings. Dynamic game theory tries to arrive at appropriate models describing a dynamic process affected by different players. Depending on the specific problem such models can sometimes be used by an individual decision maker to optimise her performance. In other cases it may serve as a starting point to introduce new communication lines which may help to improve upon the outcome of the current process. Furthermore it is possible to analyze the robustness of players' strategies w.r.t. worst-case scenarios. To this end one can introduce "nature" as an additional player which tries to work against the other decision makers in a process.

In, for example, policy coordination problems two basic questions usually arise: (i) whether policies are coordinated and, (ii) which information the participating parties have. Both these points are rather unclear and, therefore, strategies for different possible scenarios are calculated and compared with each other. Often, one of these scenarios is the so-called *open-loop* scenario. In this open-loop information scenario it is assumed that all players know just the initial state of the process and the model structure. More specifically, it is assumed that players simultaneously determine their actions for the whole planning horizon of the process before it starts. Next they submit their actions to some central authority who then enforces these plans as binding commitments. In other words, players cannot react on any deviations occurring during the evolution of the process. Obviously, since according to this scenario the participating parties can not react to each other's policies, its economic relevance is mostly rather limited. However, as a benchmark to see how much parties can gain by playing other strategies, its role is fundamental. Due to its analytic tractability the open-loop Nash equilibrium strategy is in particular very popular for problems where the underlying model can be described by a (set of) linear differential equation(s) and the individual objectives can be approximated by functions that quadratically penalise deviations from some (equilibrium) targets. Examples and additional references of differential games in economics and management science can be found e.g. in Dockner et al. (2000), Jørgensen and Zaccour (2003) and Plasmans et al. (2006).

Under the assumption that the parties have a finite-planning horizon, the linear quadratic differential game was first modeled and solved in a mathematically rigorous way by Starr and Ho (1969a,b). A recent exposition (and additional references) on linear quadratic differential games can be found in Engwerda (2005), whereas Başar and Olsder (1999) give a good overview and introduction on dynamic games in general.

In the rest of this paper we will concentrate on the case that the players base their decisions on a performance criterion that has an infinite-planning horizon. We will present an algorithm and a corresponding MATLAB numerical toolbox which solves any form of an infinite-planning horizon affine linear quadratic open-loop differential game. The software, called LQDG Toolbox, is available as a freeware

from the authors of this paper.[1] By rewriting a specific application into the standard framework, one can use the toolbox to calculate and verify the existence of both the open-loop non-cooperative Nash equilibrium (equilibria) and cooperative Pareto equilibrium (equilibria) of any infinite-planning horizon affine linear quadratic open-loop differential game. In case there is more than one equilibrium for the non-cooperative case, the toolbox determines all solutions that can be implemented as a state-feedback strategy that is a common assumption in most of the applications.[2] Alternatively, the toolbox can apply a number of choice methods in order to discriminate between multiple equilibria. For instance, one can choose to report only *Pareto*-undominated solutions or only those that are characterised by the lowest combined loss of all the players. In order to determine the cooperative solution, the user is asked to specify the relative importance of each player in the cooperative game. Moreover, the user can predefine a set of coalition structures for which they would like to calculate the non-cooperative Nash solution(s). Conversely, a coalition structure generator is provided that automatically creates a whole space of coalition structures for a given number of players. Furthermore, the toolbox offers plotting facilities as well as other options to analyze the outcome of the game. For instance, it is possible to disaggregate each player's total loss into its contributing elements, which correspond to the quadratic expressions constituting the player's loss function.

The paper is organised as follows. In Sect. 2 we will outline the model considered and present both necessary and sufficient conditions under which there will exist a unique equilibrium, a number of multiple equilibria or an infinite number of these. The basic algorithm underlying the numerical toolbox that is presented in Sect. 4 is discussed in Sect. 3. In Sect. 4 we will discuss the toolbox that has been constructed to verify in typical applications whether there will be a unique equilibrium, multiple equilibria or infinite number of these and how to obtain the solution in the first two cases. In Sect. 5 we illustrate the use and capabilities of this software with a number of examples. Section 6 concludes.

2 The Open-Loop Game

In this section we consider two players who control a different set of inputs v_i to the single system described by the structural (simultaneous) form model:

$$y(t) = P_1\dot{p}(t) + P_2 p(t) + P_3 y(t) + P_{41}v_1(t) + P_{42}v_2(t) + P_5 c, \tag{1}$$

$$\dot{p}(t) = P_6\dot{p}(t) + P_7 p(t) + P_8 y(t) + P_{91}v_1(t) + P_{92}v_2(t) + P_{10}c,\ p(0) = p_0, \tag{2}$$

where p is the n-dimensional state of the system; v_i is an m_i-dimensional (control) vector that player $i, i = 1, 2$, can manipulate; y is a b-dimensional vector of

[1] A dedicated webpage will be available on-line soon. Mean while contact T. Michalak for obtaining the software.

[2] A state-feedback strategy can be expressed as a linear function of the state variables of the model. Specifically, if we denote a vector of state variables by x, a state-feedback strategy can be written as Fx, where F is a real valued function of appropriate dimension.

endogenous variables; c is a constant (that can be chosen without loss of generality equal to 1) and p_0 is the initial state of the system. We assume that all variables can be observed at any point of time and that $n, m_i \geq 1$ and $b \geq 0$. Model (1,2) is a formulation which one frequently encounters in economic modeling. Since we like to stay close to the original equations formulated by the modeler we choose for this formulation instead of the standard state-space formulation. Notice that, by choosing $P_1 = P_3 = P_6 = P_8 = 0$, our formulation includes the standard state-space formulation.

The performance criterion player i likes to minimise is:

$$\int_0^\infty e^{-\theta t} \left\{ (\Psi_{i1} w(t) - f)^T L_{i1} (\Psi_{i1} w(t) - f) + 2(\Psi_{i2} w(t) - g)^T L_{i2} e \right\} dt, \quad (3)$$

where $w^T(t) := [\dot{p}^T(t)\, p^T(t)\, y^T(t)\, v_1^T(t)\, v_2^T(t)]$; f and g are some constant vectors in $\mathbb{R}^{2n+b+m_1+m_2}$; $e := [1, \ldots, 1]^T \in \mathbb{R}^{2n+b+m_1+m_2}$ and θ is a non-negative discounting factor. In this formulation the introduction of the Ψ_{ij} matrices makes it possible to deal with several interpretations of the model. For instance by choosing in (3) $\Psi_{i1} = \Psi_{i2}$ and $f = g$ the vector $\Psi_{ij} w(t)$ can be interpreted as an output vector of the system player i likes to track towards some specific value. By choosing a specific column of the Ψij, $j = 1, 2$, matrices equal to zero, e.g., one can express the fact that some variable is not an important variable for player i. The matrices L_{i1} are (without loss of generality) assumed to be symmetric. Later on additional assumptions will be made w.r.t. some of the above matrices.

It is assumed that the players act non-cooperatively and that their information structure about the game is of the open-loop type. More specifically, we suppose that the players choose their actions from the following set of actions:

$$\mathcal{U}_s = \left\{ v \in L_2 \mid J_i(x_0, v) \text{ exists in } \mathbb{R} \cup \{-\infty, \infty\}, \lim_{t \to \infty} x(t) = 0 \right\}.$$

The assumption that both players simultaneously use stabilizing controls introduces the cooperative meta-objective to stabilise the system (see e.g. Engwerda (2005), for a discussion).

We are looking for the Nash equilibria of this game. That is, for the combinations of actions of all players which are secure against any attempt by one player to unilaterally alter her strategy, or, stated differently, for such sets of actions that if one player deviates she will only lose. In the literature on dynamic games this problem is known as the open-loop Nash non-zero-sum linear quadratic differential game and has been analyzed by several authors (see e.g. Starr and Ho 1969a; Simaan and Cruz 1973; Başar and Olsder 1999; Abou-Kandil and Bertrand 1986; Feucht 1994; Kremer 2002; Engwerda 2005). To avoid cumbersome notation, we restrict the analyses in this section to the two-player case. The algorithm is, however, implemented for the general N-player case.

To analyze the question under which conditions this game will have a unique equilibrium we first rewrite the model into the standard framework considered in Engwerda (2008).[3]

Assuming that $I - P_6$ is invertible, we have from (2) that

$$\dot{p}(t) = (I - P_6)^{-1} \left(P_7 p(t) + P_8 y(t) + P_{91} v_1(t) + P_{92} v_2(t) + P_{10} c\right). \tag{4}$$

Substitution of this into (1) and, under the assumption that $\bar{P}_1 := I - P_1(I - P_6)^{-1} P_8 - P_3$ is invertible we can perform a number of elementary operations in order to arrive at the following reduced form of the model:

$$\dot{p}(t) = \hat{A} p(t) + \hat{B}_1 v_1(t) + \hat{B}_2 v_2(t) + \hat{E}_1 c, \; p(0) = p_0, \tag{5}$$
$$y(t) = \hat{C} p(t) + \hat{D}_1 v_1(t) + \hat{D}_2 v_2(t) + \hat{E}_2 c, \tag{6}$$

with

$$\hat{C} := \bar{P}_1^{-1}(P_1(I - P_6)^{-1} P_7 + P_2), \tag{7}$$
$$\hat{D}_1 := \bar{P}_1^{-1}(P_1(I - P_6)^{-1} P_{91} + P_{41}), \tag{8}$$
$$\hat{D}_2 := \bar{P}_1^{-1}(P_1(I - P_6)^{-1} P_{92} + P_{42}), \tag{9}$$
$$\hat{E}_2 := \bar{P}_1^{-1}(P_1(I - P_6)^{-1} P_{10} + P_5), \tag{10}$$
$$\hat{A} := (I - P_6)^{-1}(P_8 \hat{C} + P_7), \tag{11}$$
$$\hat{B}_1 := (I - P_6)^{-1}(P_8 \hat{D}_1 + P_{91}), \tag{12}$$
$$\hat{B}_2 := (I - P_6)^{-1}(P_8 \hat{D}_2 + P_{92}), \tag{13}$$
$$\hat{E}_1 := (I - P_6)^{-1}(P_8 \hat{E}_2 + P_{10}). \tag{14}$$

Next, introduce $z^T := \left[p^T \; c \; v_1^T \; v_2^T\right]$. Substituting $\dot{p}$ and y from (5) and (6), respectively, into w yields:

$$w = \begin{bmatrix} \hat{A} & \hat{E}_1 & \hat{B}_1 & \hat{B}_2 \\ I & 0 & 0 & 0 \\ \hat{C} & \hat{E}_2 & \hat{D}_1 & \hat{D}_2 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} z =: \mathcal{N} z. \tag{15}$$

Using this, J_i can be rewritten as:

$$J_i = \int_0^\infty e^{-\theta t} \Big\{ z^T(t)(\Psi_{i1}\mathcal{N} - [0\; f\; 0\; 0])^T L_{i1}(\Psi_{i1}\mathcal{N} - [0\; f\; 0\; 0]) z(t)$$
$$+ 2 z^T(t)(\Psi_{i2}\mathcal{N} - [0\; g\; 0\; 0])^T L_{i2}[0\; e\; 0\; 0] z(t) \Big\} dt$$

[3] See also Engwerda (2005, Chap. 7).

$$= \int_0^\infty e^{-\theta t} z^T(t) \Big\{ (\Psi_{i1}\mathcal{N} - [0\, f\, 0\, 0])^T L_{i1} (\Psi_{i1}\mathcal{N} - [0\, f\, 0\, 0]) + (\Psi_{i2}\mathcal{N} - [0\, g\, 0\, 0])^T L_{i2} [0\, e\, 0\, 0] + [0\, e\, 0\, 0]^T L_{i2}^T (\Psi_{i2}\mathcal{N} - [0\, g\, 0\, 0]) \Big\} z(t) dt$$

$$=: \int_0^\infty e^{-\theta t} z^T(t) M_i z(t) dt. \tag{16}$$

Finally, we introduce the extended state variable $x^T(t) := e^{-\frac{1}{2}\theta t}[p^T(t)\, c]$ and control variable $u_i(t) := e^{-\frac{1}{2}\theta t} v_i(t)$ so that the state of the system is $\bar{n} = n + 1$-dimensional. Now, the game can be rewritten into the standard form. Players minimise:

$$J_i(u_1, u_2) := \int_0^\infty [x^T(t),\ u_1^T(t),\ u_2^T(t)] M_i \begin{bmatrix} x(t) \\ u_1(t) \\ u_2(t) \end{bmatrix} dt, \tag{17}$$

where $x(t)$ is the solution to the linear differential equation:

$$\dot{x}(t) = Ax(t) + B_1 u_1(t) + B_2 u_2(t), \quad x^T(0) = \begin{bmatrix} p_0^T\ c \end{bmatrix}, \tag{18}$$

with:

$$A := \begin{bmatrix} \hat{A} - \frac{1}{2}\theta I & \hat{E}_1 \\ 0 & -\frac{1}{2}\theta \end{bmatrix}; \quad \text{and} \quad B_i := \begin{bmatrix} \hat{B}_i \\ 0 \end{bmatrix}. \tag{19}$$

Now, factorise M_i as follows:

$$M_i =: \begin{bmatrix} Q_i & V_i & W_i \\ V_i^T & R_{1i} & N_i \\ W_i^T & N_i^T & R_{2i} \end{bmatrix},$$

with $Q_i \in \mathbb{R}^{\bar{n}\times\bar{n}}$, $R_{1i} \in \mathbb{R}^{m_i \times m_i}$, $V_i \in \mathbb{R}^{\bar{n}\times m_1}$, $W_i \in \mathrm{IR}^{\bar{n}\times m_2}$ and $N_i \in \mathbb{R}^{m_1 \times m_2}$.

In the rest of the paper we will assume that the matrices R_{ii} are positive definite and matrix G (see Appendix A for some additional notation including matrix G) is invertible. In the solution of this game the next algebraic Riccati equations play an important role:

$$\begin{aligned} A^T K_1 + K_1 A - (K_1 B_1 + V_1) R_{11}^{-1} \left(B_1^T K_1 + V_1^T \right) + Q_1 = 0, \\ A^T K_2 + K_2 A - (K_2 B_2 + W_2) R_{22}^{-1} \left(B_2^T K_2 + W_2^T \right) + Q_2 = 0, \end{aligned} \tag{20}$$

as well as the set of (coupled) algebraic Riccati equations:

$$0 = \widetilde{A}_2^T \tilde{P} + \tilde{P} \widetilde{A} - \tilde{P} B G^{-1} \widetilde{B}^T \tilde{P} + \widetilde{Q}, \tag{21}$$

or, equivalently:

$$0 = A_2^T \tilde{P} + \tilde{P} A - \left(\tilde{P} B + \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \right) G^{-1} (\tilde{B}^T \tilde{P} + Z) + Q.$$

In particular, the existence of a so-called strongly stabilizing solution of (21) plays an important role. A solution $\tilde{P}^T =: (\tilde{P}_1^T, \tilde{P}_2^T)$, with $\tilde{P}_i \in \mathbb{R}^{\bar{n}}$, of (21) is called a strongly stabilizing solution if both $\sigma\left(\tilde{A} - BG^{-1}\tilde{B}^T\tilde{P}\right)$ and $\sigma\left(\tilde{A}_2^T - \tilde{P}BG^{-1}\tilde{B}^T\right) \subset \mathbb{C}^-$.[4]

The next relationship between certain invariant subspaces of matrix:

$$\mathcal{M} := \begin{bmatrix} \tilde{A} & -BG^{-1}\tilde{B}^T \\ -\tilde{Q} & -\tilde{A}_2^T \end{bmatrix} \tag{22}$$

and the solutions of the Riccati equations (21) are well-known (see e.g. Engwerda 2007). This property will also be used to calculate the strongly stabilizing solutions of (21).

Lemma 1 *Let $V \subset \mathbb{R}^{3\bar{n}}$ be an $\bar{n}$-dimensional invariant subspace of M, and let $X_i \in \mathbb{R}^{\bar{n}\times\bar{n}}$, $i = 0, 1, 2$, be three real matrices such that:*

$$V = \text{im}\left[X_0^T,\ X_1^T,\ X_2^T\right]^T. \tag{23}$$

If X_0 is invertible, then $\tilde{P}_i := X_i X_0^{-1}$, $i = 1, 2$, solves (21) and $\sigma(A - BG^{-1}(Z + \tilde{B}^T\tilde{P})) = \sigma(\mathcal{M}|_V)$. Furthermore, $(\tilde{P}_1, \tilde{P}_2)$ is independent of the specific choice of a basis of V.

When X_0 is invertible, the $\bar{n}$-dimensional invariant subspace V in (23) is called a graph subspace. The next lemma provides a characterisation of the strongly stabilizing solution of (21):

Lemma 2 1. *The set of algebraic Riccati equations (21) has a strongly stabilizing solution $\tilde{P}$ if and only if matrix $\mathcal{M}$ has an $\bar{n}$-dimensional stable graph subspace and $\mathcal{M}$ has $2\bar{n}$ eigenvalues (counting algebraic multiplicities) with a real part larger than or equal to zero.*
2. *If the set of algebraic Riccati equations (21) has a strongly stabilizing solution, then it is unique.*

A proof of both this lemma and the next theorem follow directly from Engwerda (2008).

Theorem 3 *Consider the differential game (17–18) with symmetric $M_i, R_{ii} > 0$, invertible matrix G and (A, B_i) stabilizable.*
This game has a unique open-loop Nash equilibrium for every initial state if and only if

[4] $\sigma(H)$ denotes the spectrum of matrix H; $\mathbb{C}^- = \{\lambda \in \mathbb{C} \,|\, \text{Re}(\lambda) < 0\}$.

1. *The set of coupled algebraic Riccati equations* (21) *has a strongly stabilizing solution, and*
2. *the two algebraic Riccati equations* (20) *have a (strongly) stabilizing solution.*

Moreover, in case this game has a unique equilibrium, the equilibrium actions are given by:

$$\begin{bmatrix} u_1^*(t) \\ u_2^*(t) \end{bmatrix} = -G^{-1}\left(Z + \tilde{B}^T\tilde{P}\right)\tilde{\Phi}(t,0)x_0, \tag{24}$$

where $\tilde{\Phi}(t,0)$ *is the solution of the transition equation:*

$$\dot{\tilde{\Phi}}(t,0) = \left(A - BG^{-1}\left(Z + \tilde{B^T}\tilde{P}\right)\right)\tilde{\Phi}(t,0);\ \tilde{\Phi}(0,0) = I.$$

The costs by using the actions (24) for the players are:

$$x_0^T\tilde{L}_i x_0, \quad i = 1,2, \tag{25}$$

where, with $A_{cl} := A - BG^{-1}\left(Z + \tilde{B}^T\tilde{P}\right)$, $\tilde{L}_i$ *is the unique solution of the Lyapunov equation:*

$$\left[I,\ -G^{-1}\left(Z + \tilde{B}^T\tilde{P}\right)\right]M_i\left[I,\ -G^{-1}\left(Z + \tilde{B}^T\tilde{P}\right)\right]^T + A_{cl}^T\tilde{L}_i + \tilde{L}_i A_{cl} = 0.$$

In case the game has more than one equilibrium, generically, there exists an infinite number of equilibria (see Engwerda 2005; Kremer 2002). However, usually for a finite number of them only, the corresponding actions can be implemented as a state-feedback strategy (i.e., such strategy that can be written as a linear function of the state variables of the model like, for instance, in Theorem 3; see also footnote 1 in this paper). In economics it is often argued that actions by policymakers have a (state-)feedback structure (Taylor rule etc.). For that reason, in case there is more than one equilibrium, the algorithm will determine only those that can be implemented as a (state-)feedback rule. From Engwerda (2005) we recall the following result.

Theorem 4 *The differential game* (17–18)*, with* M_i *symmetric,* $R_{ii} > 0$*, matrix* G *invertible and* (A, B_i) *stabilizable, has for every initial state an open-loop Nash set of equilibrium actions* (u_1^*, u_2^*) *which permit a feedback synthesis if and only if*

1. *The set of coupled algebraic Riccati equations* (21) *has a stabilizing solution, and*
2. *the two algebraic Riccati equations* (20) *have a stabilizing solution.*

Moreover, if $\tilde{P}$ *is stabilizing solution of* (21)*, the actions* (24) *yield an open-loop Nash equilibrium that can be synthesised as a state feedback. The corresponding costs are given by* (25)*.*

Finally, if one would like to consider the game (17–18) without a constant that, in particular, allows for a zero discount rate, the state variable should not be extended,

i.e., $x^T(t) := e^{-\frac{1}{2}\theta t}\left[p^T(t)\right]$ is to be used instead of $x^T(t) := e^{-\frac{1}{2}\theta t}\left[p^T(t)\, c\right]$. The system in (18) becomes:

$$\dot{x}(t) = Ax(t) + B_1u_1(t) + B_2u_2(t), \quad x^T(0) = p_0^T, \tag{26}$$

where:

$$A := \hat{A} - \frac{1}{2}\theta I; \tag{27}$$

$$B_i := \hat{B}_i. \tag{28}$$

The rest of the above analysis remains valid but with $\bar{n} = n$.

3 The Computational Framework

To verify the existence, the number and numerical values of equilibria in the game (17–18), in the numerical toolbox we use the following algorithm[5]:

Algorithm 5

Step A1: Verify whether $R_{ii} > 0$, G is invertible and (A, B_i) are stabilizable. If this is not the case, go to Step A6.

Step A2: Calculate the eigenstructure of:

$$H_1 := \begin{bmatrix} A - B_1R_{11}^{-1}V_1 & -S_1 \\ -Q_1 & -\left(A - B_1R_{11}^{-1}V_1\right)^T \end{bmatrix} \text{ and}$$

$$H_2 := \begin{bmatrix} A - B_2R_{22}^{-1}W_2 & -S_2 \\ -Q_2 & -\left(A - B_2R_{22}^{-1}W_2\right)^T \end{bmatrix}.$$

If H_i, $i = 1, 2$, has an $\bar{n}$-dimensional stable graph subspace, then proceed. Otherwise there is not an equilibrium and go to Step A6.[6]

StepA3: Calculate matrix $\mathcal{M} := \begin{bmatrix} \tilde{A} & -BG^{-1}\tilde{B}^T \\ -\tilde{Q} & -\tilde{A}_2^T \end{bmatrix}$.
If $\mathcal{M}$ has $s \geq \bar{n}$ stable eigenvalues and $u \leq 2\bar{n}$ unstable eigenvalues (counting algebraic multiplicities) then proceed to Step A4. Otherwise, i.e., if $s < \bar{n}$, the game has no equilibrium and go to Step A6.

Step A4: The number of equilibria equals the number of different $\bar{n}$-dimensional stable invariant graph subspaces of matrix $\mathcal{M}$. In particular this implies that if

[5] Notice that we just present those equilibria that can be synthesized as a feedback strategy.

[6] With an equilibrium we mean everywhere in the algorithm an equilibrium that can be synthesized as a state feedback.

matrix $\mathcal{M}$ has eigenvalues that have a geometric multiplicity that is larger than one, it may happen that there are an infinite number of equilibria (see e.g. Engwerda 2005, Example 7.11). The algorithm does not elaborate this case because (i) it is numerically more involved to decide whether an eigenvalue has a geometric multiplicity that is larger than one; (ii) the occurrence of such cases requires a more detailed inspection of how many different $\bar{n}$-dimensional subspaces can be generated; and (iii) generically, the algebraic multiplicity of the eigenvalues is one. If $s \geq \bar{n}$ and there is at least one eigenvalue that has an algebraic multiplicity larger than one, then the algorithm terminates and the user is informed about the possibility that in the simulation an infinite number of equilibria might occur. It is then up to the user either to choose some parameters that differ a little bit from the previous choice (yielding probably a simulation that produces no difficulties), or to take a serious look at the eigenstructure of matrix $\mathcal{M}$ and draw own conclusions. Therefore the algorithm proceeds as follows:

A4.1: If at least one eigenvalue has an algebraic multiplicity larger than one then the game may have an infinite number of equilibria if $s > \bar{n}$. If $s = \bar{n}$ there probably exists a unique equilibrium, but this has to be verified using the generalised eigenspace(s). For both cases: go to Step A6. Otherwise proceed with Step A4.2.

A4.2: All stable eigenvalues have an algebraic multiplicity of one.

A4.2.1: Case $s = \bar{n}$. Let λ^s denote the set of all stable eigenvalues and $\tilde{V}^s$ denote the matrix which columns consist of the eigenvectors corresponding with these stable eigenvalues.

A4.2.1.1: If λ^s contains no complex eigenvalues then the image of $\tilde{V}^s$ represents the $\bar{n}$-dimensional stable invariant subspace of $\mathcal{M}$. Go to Step A5.

A4.2.1.2: If λ^s contains complex eigenvalues then replace every pair of conjugate complex eigenvectors $x + iy$ and $x - iy$ in $\tilde{V}^s$ by the real part of this eigenvector, x, and the imaginary part, y, respectively (see Example 6 below for further details). The image of this modified real matrix represents then the $\bar{n}$ -dimensional stable invariant subspace of $\mathcal{M}$. Go to Step A5.

A4.2.2: If $s > \bar{n}$ and:

A4.2.2.1: If there are no complex eigenvalues then:

- Calculate the set $C^{\bar{n}}_{\lambda^s}$ of all $\frac{s!}{(s-\bar{n})!\bar{n}!}\bar{n}$-element combinations from s stable eigenvalues;
- For every combination $\lambda^s_i \in C^{\bar{n}}_{\lambda^s}$: (i) let $\tilde{V}^s_i$ be the matrix which columns consist of eigenvectors corresponding with eigenvalues from the set λ^s_i; (ii) go to Step A5.

A4.2.2.2: If there are complex eigenvalues then:

- Calculate the set $C^{\bar{n}}_{\lambda^s}$ of all $\frac{s!}{(s-\bar{n})!\bar{n}!}\bar{n}$-element combinations from s stable eigenvalues that

have the property where if λ_i^s contains a complex eigenvalue it contains the complex conjugate of this eigenvalue too.

- For every such combination $\lambda_i^s \in C_{\lambda^s}^{\bar{n}}$: (i) let $\tilde{V}_i^s$ be the matrix constructed in a similar way as in item A4.2.1.2; (ii) go to Step A5.

Step A5: For every matrix of eigenvectors $\tilde{V}_i^s$ calculate the corresponding $\mathcal{M}$-invariant graph subspace $\mathcal{P}$. To that end proceed as follows:

A5.1: Let X_1, X_2 and X_3 be such that

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \tilde{V}_i^s.$$

Verify whether matrix X_1 is invertible. If not then $\mathrm{Im}\,\tilde{V}_i^s$ is not a graph subspace and there is no equilibrium corresponding with this set of eigenvalues. Choose another matrix $\tilde{V}_i^s$ from the set of potential candidates and return to the beginning of the current step (Step A5.1). If there are no other candidates left, go to Step 6.

* Denote $\tilde{P}_1 := X_2 X_1^{-1}$ and $\tilde{P}_2 := X_3 X_1^{-1}$.
* Then:

$$\begin{bmatrix} u_1^*(t) \\ u_2^*(t) \end{bmatrix} := -G^{-1}(Z + \widetilde{B}^T \tilde{P}) x^*(t),$$

where with $\tilde{P}^T := \begin{bmatrix} \tilde{P}_1^T & \tilde{P}_2^T \end{bmatrix}$ is the open-loop Nash equilibrium strategy. Here $x^*(t)$ is the solution of the differential equation $\dot{x}(t) = (A - BG^{-1}(Z + \widetilde{B}^T \tilde{P}))x(t)$, $x(0) = x_0$.

* The spectrum of the corresponding closed-loop matrix A_{cl} equals λ_i^s.
* Players' losses can be computed from $J_i = p_0^T \tilde{L}_i p_0$ where $\tilde{L}_i$ solves the following Lyapunov equation:

$$A_{CL}^T \tilde{L}_i + \tilde{L}_i A_{CL} + \frac{1}{2} \begin{bmatrix} I, F^T \end{bmatrix} M_i \begin{bmatrix} I \\ F \end{bmatrix} = 0. \tag{29}$$

Step A6: End of algorithm.

Step A2 in the above algorithm verifies whether the algebraic Riccati equations (20) have a stabilizing solution. Of course one can use here MATLAB to verify this. Concerning the numerical stability of Algorithm 5 we notice that various suggestions have been made in the literature to calculate solutions of Riccati equations in a numerically reliable way (see e.g. Laub (1979, 1991), Paige and van Loan (1981), van Dooren (1981), Mehrmann (1991) and Abou-Kandil and Bertrand (1986) for a more general survey on various types of Riccati equations). These methods can also be

used to improve the numerical stability of Algorithm 5. In particular, if one considers the implementation of large scale models one should consult this literature.

As already indicated one can also try to solve the (set of coupled) algebraic Riccati equations (20) iteratively. In particular, for large scale systems one might hope that such algorithms will be more efficient. For that reason various iteration schemes have been suggested in literature (see e.g. Abou-Kandil and Bertrand 1986 or Engwerda 2007). However, since Eq. 20 may admit several solutions, convergence of any game is quite difficult to obtain under general conditions (see Azevedo-Perdicoúlis and Jank (2005), for a result on positive solutions). An important problem with these algorithms is related to the *a priori* verification of the system's strong stabilizability. If one does not want to do it there is an open question how to proceed in case the algorithm terminates at a non-stabilizing solution.

4 LQDG Toolbox

Next we proceed with an outline of the numerical toolbox. The software verifies the existence of and, provided that a finite number of equilibria exists, calculates the outcome of the N-player extension of the game (1–3). The scheme presenting all the components of the toolbox software is displayed in Fig. 1. The main file, called LQDGsolver.exe, solves the LQDG which is to be defined in the input. The input file can be created by the user using an intuitive input interface provided (file TBXinputGUI.exe). Alternatively, more proficient users might choose to create the input file directly (in MATLAB or text formats). LQDGsolver produces the following output for every coalition structure considered:

- number of equilibria, intermediate matrices constructed during the solution process (H_1, H_2, H_3 and $\mathcal{M}$) and eigenvalues;
- output actions;
- closed-loop matrices A_{cl};
- solution of Lyapunov equations $\tilde{L}_i$ for every player; and
- loss J_i for every player.

The above output is saved in MATLAB and text formats in a directory that corresponds to the chosen name of the project. The following output files are created:

- PROJECT_NAME_model.txt: text file containing the structural and reduced form of the dynamic system;
- inputPROJECT_NAME.m: binary MATLAB file containing (processed) input to the project;
- PROJECT_NAME_validation.txt: text file containing all the information about the various stages of model validation;
- PROJECT_NAME_output.txt: text file containing all the output produced; and
- outputPROJECT_NAME.m: binary MATLAB file containing all the project's output.

The plotting tool is provided that uses the above output to draw the dynamics of every variable in the model for a chosen coalition structure and equilibrium. Less advanced users can use a simple output interface (file TBXoutputGUI.exe) that allows

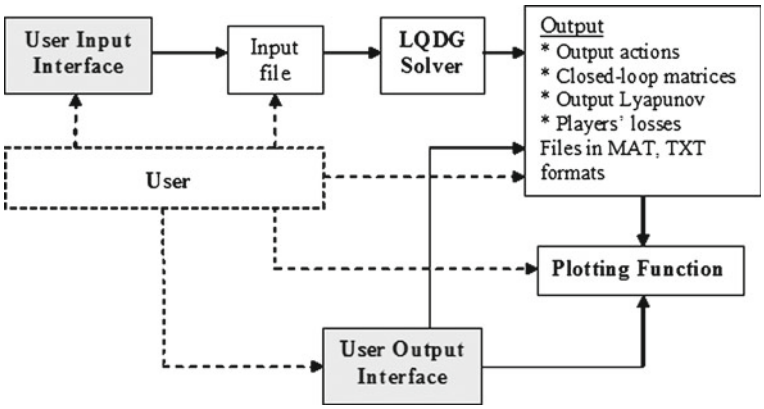

Fig. 1 Scheme of the LQDG Toolbox software

both to edit the toolbox output and to plot the graphs required. Conversely, more advanced users can directly analyse output of all numerical simulations and create graphs.

The input for the toolbox is the following model:

$$y(t) = P_1\dot{p}(t) + P_2 p(t) + P_3 y(t) + \sum_{i=1}^{N} P_{4i} v_i(t) + P_5, \tag{30}$$

$$\dot{p}(t) = P_6\dot{p}(t) + P_7 p(t) + P_8 y(t) + \sum_{i=1}^{N} P_{9i} v_i(t) + P_{10}, \quad p(0) = p_0, \tag{31}$$

where p is the n-dimensional state of the system, v_i is the m_i-dimensional (control) vector player i (where $i = 1, \ldots, |N|$) can manipulate (with $\sum m_i =: m$), y is the b-dimensional vector of endogenous variables and p_0 is the initial state of the system. Defining $z(t) := [p(t)\ \dot{p}(t)\ y(t)\ v(t)\ c]^T$, the general form of the performance criterion that player i likes to minimise is:

$$J_i = \frac{1}{2}\int_0^{\infty} \left\{ z^T(t)\Phi_i z(t) \right\} e^{-\theta(t-t_0)} dt, \tag{32}$$

where:

$$\Phi_i := \begin{bmatrix} \phi_{(1,1),i} & \phi_{(1,2),i} & \cdots \phi_{(1,2n+b+m+1),i} \\ \phi_{(2,1),i} & \phi_{(2,2),i} & \cdots \phi_{(2,2n+b+m+1),i} \\ \cdots & \cdots & \cdots\cdots \\ \phi_{(2n+b+m+1,1,i)} & \phi_{(2n+b+m+1,2),i} & \cdots \phi_{(2n+b+m+1,2n+b+m+1),i} \end{bmatrix}.$$

Matrix Φ_i can be factorised as follows:

$$\Phi_i := \begin{bmatrix} \Phi_{p^2,i} & \Phi_{p\dot{p},i} & \Phi_{py,i} & \Phi_{pv,i} & \Phi_{pc,i} \\ \Phi_{\dot{p}p,i} & \Phi_{\dot{p}^2,i} & \Phi_{\dot{p}y,i} & \Phi_{\dot{p}v,i} & \Phi_{\dot{p}c,i} \\ \Phi_{yp,i} & \Phi_{y\dot{p},i} & \Phi_{y^2,i} & \Phi_{yv,i} & \Phi_{yc,i} \\ \Phi_{vp,i} & \Phi_{v\dot{p},i} & \Phi_{vy,i} & \Phi_{v^2,i} & \Phi_{vc,i} \\ \Phi_{cp,i} & \Phi_{c\dot{p},i} & \Phi_{cy,i} & \Phi_{cv,i} & \Phi_{c^2,i} \end{bmatrix}.$$

with all submatrices $\Phi_{.,i}$ defined in Appendix B. Furthermore, in Appendix C it is shown how, using the appropriate notation, it is possible to define any linear quadratic loss function containing the variables from vector $z(t)$ in the input file (if the user chooses to create this file by himself not using the provided interface). Of course, the user is supposed to define only those coefficients ϕ (i.e., elements of matrix $\tilde{\Phi}_i$) that are different from zero. The cost criterion (32) can be rewritten as follows:

$$\begin{aligned} J_i &:= \frac{1}{2}\int_0^{\infty} \left\{ \begin{bmatrix} p(t) & c & v(t) \end{bmatrix} \Gamma^T \Phi_i \Gamma \begin{bmatrix} p(t) \\ c \\ v(t) \end{bmatrix} \right\} e^{-\theta(t-t_0)} dt \\ &:= \frac{1}{2}\int_0^{\infty} \left\{ \begin{bmatrix} p(t) & c & v(t) \end{bmatrix} M_i \begin{bmatrix} p(t) \\ c \\ v(t) \end{bmatrix} \right\} e^{-\theta(t-t_0)} dt, \end{aligned}$$

where

$$\Gamma := \begin{bmatrix} \begin{bmatrix} I_{n\times n} & 0_{n\times 1} & 0_{n\times m} \end{bmatrix}_{n\times(n+m+1)} \\ \begin{bmatrix} \hat{A}_1 & \hat{E}_1 & \hat{B} \end{bmatrix} \\ \begin{bmatrix} \hat{C} & \hat{E}_2 & \hat{D} \end{bmatrix} \\ \begin{bmatrix} 0_{m\times n} & 0_{m\times 1} & I_{m\times m} \end{bmatrix}_{m\times(n+m+1)} \\ \begin{bmatrix} 0_{1\times n} & 1 & 0_{1\times m} \end{bmatrix}_{1\times(n+m+1)} \end{bmatrix} \tag{33}$$

and $M_i := \Gamma^T \Phi_i \Gamma$.

The toolbox also offers the possibility to analyze equilibria for different coalition structures which are formally defined as divisions of all the players in the game into exhaustive and disjoint coalitions. That is, if, for instance, five players participate in the game and players 1, 3, 4 and players 2, 5 decide to cooperate, then the toolbox offers the possibility to calculate the open-loop Nash equilibrium for the resulting two-player game. Obviously the outcome depends on the relative weight of every players' performance criterion within the coalition. Therefore, the user is asked to specify both the coalition structure and these relative weights.

The number of coalitions structures that can be created from even a small number of players is a non-trivial issue. More in detail, let Π denote the set of all possible coalition structures. The number of all possible coalition structures is a function of the number of players N and equals the so-called Bell number $B_{|N|}$. The Bell number is the number of ways a set of $|N|$ elements can be partitioned into non-empty subsets.

The following Dobinsky's formula is one way to compute Bell numbers (Comtet 1974): $B_{|N|} = \frac{1}{e}\sum_{j=0}^{\infty}\frac{j^{|N|}}{j!}$.[7]

To simplify the presentation of coalition structures we will use the following shorthand notation: $[C_1|C_2|\ldots|C_m]$ where $C_{1\leq k\leq m}$ is represented by the sequence of players that belong to this coalition. For example: [123|4|56] stands for a coalition structure where players 1, 2, 3 and players 5, 6 cooperate, respectively and player 4 remains single.

Example 6 Let $N = \{1, 2, 3, 4\}$. Listing all the possible partitions of the set N into coalitions, we obtain: [1234], [123|4], [124|3], [134|2], [1|234], [12|3|4], [13|2|4], [14|2|3], [23|1|4], [24|1|3], [1|2|34], [12|34], [13|24], [14|23], [1|2|3|4]. Indeed, using simple recursive software to compute Dobinsky's formula we obtain: $B_4 = 15$. The number of possible partitions increases exponentially when n increases linearly. For $|N| = 1, 2, \ldots, 15$, we have the following numbers of coalitions according to Dobinski's formula: $B_1 = 1$, $B_2 = 2$, $B_3 = 5$, $B_4 = 15$, $B_5 = 52$; $B_6 = 203$, $B_7 = 877$, $B_8 = 4140$; $B_9 = 21147$, $B_{10} = 115975$, $B_{11} = 678570$, $B_{12} = 4213597$, $B_{13} = 27644437$; $B_{14} = 190899322$, and $B_{15} = 1382958545$.

It is clear that an examination of all possible coalition structures is not always interesting for the user. Usually, the researcher should choose and restrict her attention to a subset of coalition structures. In the sequel we will use the notation Π^F for the full set of feasible coalition structures and Π^R for a reduced set of relevant feasible coalition structures.

We will now briefly discuss the various steps the user is confronted with while using the toolbox.

Step T1: LQDG Toolbox Initialisation To define the LQDG problem the user is supposed to provide a number of compulsory components of the model. As it has been mentioned before, it is the most convenient to use the interface provided in order to create the project. The main window of the user interface is shown in Fig. 2. More proficient users also may create the input file directly in MATLAB or text formats. Compulsory components of the LQDG problem include:

- the name of the new project;
- number of players, number of state and output variables, whether the model include constant or not;
- the number of control instruments per player;
- the nonzero P_i and P_{ij} matrices from the structural form model (30–31);
- the parameters from the performance criterion (32);
- the initial condition p_0; and
- the coalition structures to be considered.[8]
- If the model includes constants, i.e., at least one element of matrices P_5 and P_{10} is non-zero, then it is mandatory to specify a strictly positive discount rate.

[7] See Chapter 5 of Plasmans et al. (2006) for more details on this issue.

[8] For less than 6 players LQDG Toolbox has predefined sets of all coalition structures so that the user does not have to define them himself.

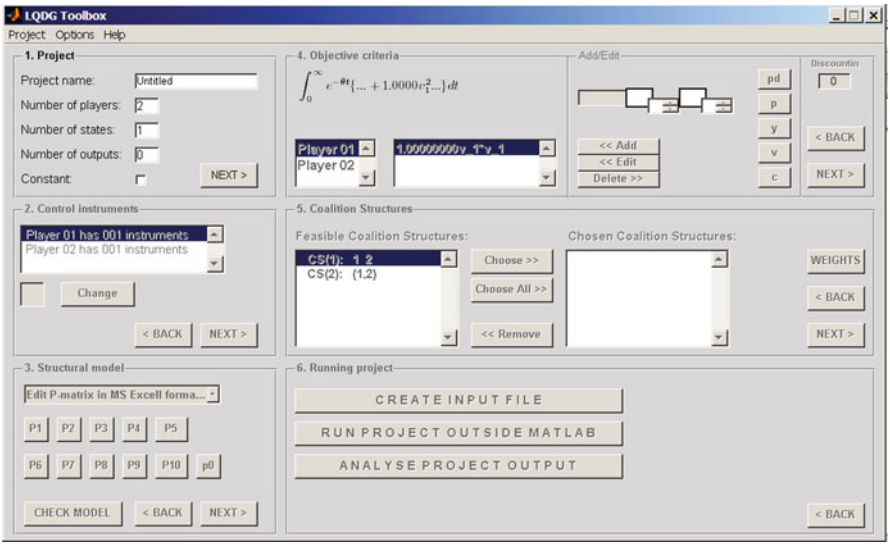

Fig. 2 Main window of the User Input Interface

LQDG Toolbox offers an opportunity to control and verify every important intermediate step of the solution process as well as the results obtained. The list of all the available options can be edited in the user interface.

Step T2: Model Validation by LQDG Toolbox Based on the input that is provided in Step A1, the toolbox verifies various regularity conditions:

1. The invertibility of the matrices $I - P_3$ and $I - P_7(I - P_3)^{-1}P_1 - P_5$;
2. Positive definiteness of the matrices R_{ii};
3. Invertibility of G; and stabilizability of (A, B_i); and
4. The final validation step that is performed checks whether the algebraic Riccati equations (20) have a stabilizing solution.

If conditions 1 or 2 fail the toolbox terminates. If conditions 3 and 4 fail then it means that no equilibrium can be found for the particular coalition structure and the toolbox considers the next coalition structure in the queue.

Step T3: Calculation of Equilibria In this phase, open-loop Nash equilibria as outlined in Theorem 3 are calculated for every specified coalition structure (if such equilibria exists).

5 Examples

In this section we illustrate the various steps of the algorithm in three simple examples. Firstly, we analyze a dynamic duopoly game with sticky prices as considered by Fershtman and Kamien (1987); secondly, we present the solution of the problem used in Example 7.10 by Engwerda (2005), where a multiple finite number of equilibria

emerges; thirdly, we present the solution of the problem used in Example 7.12 by Engwerda (2005) characterised by complex eigenvalues.

5.1 A Game on Dynamic Duopolistic Competition from Fershtman and Kamien (1987)

The problem we address here is to find the open-loop Nash equilibria of the game defined by the revenue functions:

$$J_i(v_1, v_2) = \int_0^\infty e^{-\theta t} \left\{ p(t)v_i(t) - c_v v_i(t) - \frac{1}{2}v_i^2(t) \right\} dt, \tag{34}$$

subject to the dynamic constraint:

$$\dot{p}(t) = s\{a - (v_1(t) + v_2(t)) - p(t)\}, \quad p(0) = p_0. \tag{35}$$

Recall that, in this model, $\theta > 0$ denotes the discount rate of future profits and $s \in (0, \infty)$ is the adjustment speed parameter of the market price, $p(t)$, towards the price dictated by the demand function. That is, for larger values of s the market price adjusts more quickly along the demand function. The cost functions of the companies are assumed to be $C(v_i) := c_v v_i + \frac{1}{2}v_i^2$, where $c_v \in (0, a)$ is a fixed parameter. Furthermore, the inverse demand function is assumed to be given by $\tilde{p} = a - (v_1 + v_2)$.

To determine the open-loop equilibrium actions for this game (34–35) we proceed along the steps outlined in Sect. 3. To that end we first notice that the maximisation of (34) can be rewritten as the minimisation of $-J_i$.

Step T1: LQDG Toolbox Initialisation The above maximisation problems can be rewritten in terms of (32) as:

$$-\min_{v_1} J_1(v_1, v_2) = \frac{1}{2}\int_0^T e^{-\theta t} \left\{ z^T(t) \underbrace{\begin{bmatrix} 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 2c_v \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\Phi_1} z(t) \right\} dt \text{ and}$$

$$-\min_{v_2} J_2(v_1, v_2) = \frac{1}{2}\int_0^T e^{-\theta t} \left\{ z^T(t) \underbrace{\begin{bmatrix} 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2c_v \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\Phi_2} z(t) \right\} dt,$$

subject to the dynamics (35), where $z^T(t) := [p(t)\ \dot{p}(t)\ v_1(t)\ v_2(t)c]$.

The model fits into our standard model (30–31) if we specify the various parameters in the input file of the toolbox as follows:

$$P_1 = P_2 = P_3 = P_{4i} = P_5 = P_6 = P_8 = 0,\ P_7 = -s,\ P_{9i} = -s \quad \text{and} \quad P_{10} = c.$$

Notice that in this example $n = m_1 = m_2 = 1$ (and therefore $m = 2$) and $b = 0$. In the performance specification all parameters are zero for both players except the next ones. For player 1 we have $\phi_{(1,3),1} = -2$, $\phi_{(3,3),1} = 1$, and $\phi_{(3,5),1} = 2c_v$, whereas for player 2 $\phi_{(1,4),2} = -2$, $\phi_{(4,4),2} = 1$ and $\phi_{(4,5),2} = 2c_v$.

There are two players in the game. Hence $B_2 = 2$ and $\Pi^F = \{[1|2],\ [12]\}$. As already mentioned before, the user can either choose to compute the Nash equilibria for all coalition structures (if such equilibria exist) or only for some elements of Π^F by creating own set of feasible coalition structures Π^R. We choose in this example to calculate analytically the equilibrium for non-cooperative CS [1|2].

Step T2: Model Validation by LQDG Toolbox With this input the toolbox next calculates the standard form (17–18) by considering the new variables:

$$x_1(t) := e^{-\frac{1}{2}\theta t} p(t),\ x_2(t) := e^{-\frac{1}{2}\theta t} \quad \text{and} \quad u_i(t) := e^{-\frac{1}{2}\theta t} v_i(t),\ i = 1, 2. \tag{36}$$

With these variables the problem (34–35) can be rewritten as finding the open-loop Nash equilibrium of

$$-\min_{u_i(.)} \frac{1}{2} \int_0^\infty [x_1(t)\ x_2(t)\ u_1(t)\ u_2(t)] M_i \begin{bmatrix} x_1(t) \\ x_2(t) \\ u_1(t) \\ u_2(t) \end{bmatrix} dt, \tag{37}$$

subject to the dynamics

$$\dot{x}(t) = \begin{bmatrix} -s - \frac{1}{2}\theta & as \\ 0 & -\frac{1}{2}\theta \end{bmatrix} x(t) + \begin{bmatrix} -s \\ 0 \end{bmatrix} u_1(t) + \begin{bmatrix} -s \\ 0 \end{bmatrix} u_2(t). \tag{38}$$

where $M_i = \Gamma^T \Phi_i \Gamma$ as in (33) with Φ_i defined by the user in Step T1.[9]

Following the notation in Sect. 2 the toolbox generates the following matrices:

$$A = \begin{bmatrix} -s - \frac{1}{2}\theta & as \\ 0 & -\frac{1}{2}\theta \end{bmatrix},\ B_i = \begin{bmatrix} -s \\ 0 \end{bmatrix},\ Q_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix},\ R_{ii} = 1,\ R_{ij} = 0\ (i \neq j),$$
$$V_1 = W_2 = \begin{bmatrix} -1 \\ c_v \end{bmatrix},\ W_1 = V_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \text{and } N_i = 0.$$

[9] Note that $y(t)$ is not taken into account in vector $z(t)$ in the cost criteria in Step T1 as $y(t) := 0$. Toolbox automatically extends matrices Φ_1 and Φ_2 by two appropriate zero vectors.

So, clearly $R_{ii} > 0$ and (A, B_i) is stabilizable. Furthermore:

$$G := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B := \begin{bmatrix} -s & -s \\ 0 & 0 \end{bmatrix}, \quad Z_1 := \begin{bmatrix} -1 & 0 \\ c_v & 0 \end{bmatrix}, \quad Z_2 := \begin{bmatrix} 0 & -1 \\ 0 & c_v \end{bmatrix},$$
$$Z := \begin{bmatrix} -1 & c_v \\ -1 & c_v \end{bmatrix}, \quad \tilde{B}_1^T := \begin{bmatrix} -s & 0 \\ 0 & 0 \end{bmatrix}, \quad \tilde{B}_2^T := \begin{bmatrix} 0 & 0 \\ -s & 0 \end{bmatrix}, \quad \tilde{B}^T := \begin{bmatrix} -s & 0 & 0 & 0 \\ 0 & 0 & -s & 0 \end{bmatrix}.$$

and from this:

$$\mathcal{M} = \begin{bmatrix} -3s - \frac{1}{2}\theta & s(a+2c_v) & -s^2 & 0 & -s^2 & 0 \\ 0 & -\frac{1}{2}\theta & 0 & 0 & 0 & 0 \\ 1 & -c_v & 2s + \frac{1}{2}\theta & 0 & 0 & 0 \\ -c_v & c_v^2 & -s(a+c_v) & \frac{1}{2}\theta & 0 & 0 \\ 1 & -c_v & 0 & 0 & 2s + \frac{1}{2}\theta & 0 \\ -c_v & c_v^2 & 0 & 0 & -s(a+c_v) & \frac{1}{2}\theta \end{bmatrix}$$
$$=: \begin{bmatrix} \tilde{A} & -\tilde{S} \\ -\tilde{Q} & -\tilde{A}_2^T \end{bmatrix}.$$

Following Sect. 2 the game has, for all p_0, a unique open-loop Nash equilibrium if the following sets of algebraic Riccati equations have a (strongly) stabilizing solution P_i and K_i, (where $i = 1, 2$, respectively):

1. The set of coupled Riccati differential equations:

$$0 = -\tilde{A}_2^T P - P\tilde{A} + PBG^{-1}\tilde{B}^T P - \tilde{Q}. \tag{39}$$

2. The two Riccati equations:

$$0 = -A^T K_1 - K_1 A + (K_1 B_1 + V_1) R_{11} \left(B_1^T K_1 + V_1^T \right) - Q_1, \tag{40}$$
$$0 = -A^T K_2 - K_2 A + (K_2 B_2 + W_2) R_{22} \left(B_2^T K_2 + W_2^T \right) - Q_2. \tag{41}$$

To verify whether, e.g., the algebraic Riccati equation (40) has a stabilizing solution it is sufficient to check whether the following equation satisfies this property:

$$\left(A - B_1 R_{11}^{-1} V_1^T \right)^T K + K \left(A - B_1 R_{11}^{-1} V_1^T \right) - K S_1 K + Q_1 - V_1 R_{11}^{-1} V_1^T = 0. \tag{42}$$

It is easily verified that the Hamiltonian matrix associated with this Riccati equation:

$$\begin{bmatrix} A - B_1 R_{11}^{-1} V_1^T & -S_1 \\ -Q_1 + V_1 R_{11}^{-1} V_1^T & -\left(A - B_1 R_{11}^{-1} V_1^T \right)^T \end{bmatrix} = \begin{bmatrix} -2s - \frac{1}{2}\theta & s(a+c_v) & -s^2 & 0 \\ 0 & -\frac{1}{2}\theta & 0 & 0 \\ 1 & -c_v & 2s + \frac{1}{2}\theta & 0 \\ -c_v & c_v^2 & -s(a+c_v) & \frac{1}{2}\theta \end{bmatrix}$$

has a two-dimensional stable graph subspace. Thus, Eq. 40 has a stabilizing solution. In a similar way one can verify whether (41) has a strongly stabilizing solution. The procedure to verify whether (39) has a strongly stabilizing solution is presented in the next step.

Step T3: Calculation of Equilibria Again, following Algorithm 5, the toolbox first computes the eigenvalues of matrix $\mathcal{M}$ to determine whether (42) has a strongly stabilizing solution. These eigenvalues are $\{-\frac{1}{2}\theta, -\frac{1}{2}s - \frac{1}{2}\lambda_1, \frac{1}{2}\theta, \frac{1}{2}\theta, \frac{1}{2}\theta + 2s, -\frac{1}{2}s + \frac{1}{2}\lambda_1\}$, where:

$$\lambda_1 = \sqrt{17s^2 + 10s\theta + \theta^2}.$$

Clearly, matrix $\mathcal{M}$ has 2 stable and 4 unstable eigenvalues. Therefore, the only open-loop Nash equilibrium candidate is obtained by considering the eigenspaces of $\mathcal{M}$ corresponding with the eigenvalues $-\frac{1}{2}\theta$ and $-\frac{1}{2}s - \frac{1}{2}\lambda_1$. The eigenspaces corresponding with these eigenvalues are:

$$\begin{aligned}
&\mathcal{T}_1 = \text{Span}\{T_1\} \text{ where}\\
&T_1 = [-(2c_v\theta + 2c_vs + \theta a + 2sa),\ -(3\theta + 4s),\ a - c_v,\\
&\qquad (a - c_v)(as - c_vs - c_v\theta)/\theta,\ a - c_v,\ (a - c_v)(as - c_vs - c_v\theta)/\theta]^T, \text{ and}\\
&\mathcal{T}_2 = \text{Span}\{T_2\} \text{ where}\\
&T_2 = [-5s - \theta - \lambda_1,\ 0,\ 2,\ v,\ 2,\ v]^T,
\end{aligned}$$

respectively, with $v := -\frac{(s+\theta-\lambda_1)(2sa-3sc_v-c_v\theta-c_v\lambda_1)}{4s(2s+\theta)}$. In particular we see that for the values $\{-\frac{1}{2}\theta, -\frac{1}{2}s - \frac{1}{2}\lambda_1\}$ the corresponding eigenspace is a graph subspace (i.e., $\tilde{V}_i^s = \text{Span}\{T_1, T_2\}$).

Since (A, B_i), where $i = 1, 2$, are stabilizable, $R_{ii} > 0$, G is invertible, $\mathcal{M}$ has a 2-dimensional stable graph subspace and 4 unstable eigenvalues, and the algebraic Riccati equations (39–40) have a stabilizing solution, then all conditions are satisfied for the game to have a unique open-loop Nash equilibrium. The corresponding actions are:

$$\begin{bmatrix} u_1^*(t) \\ u_2^*(t) \end{bmatrix} = -\left(\begin{bmatrix} -1 & c_v \\ -1 & c_v \end{bmatrix} + \begin{bmatrix} -s & 0 \\ -s & 0 \end{bmatrix}\tilde{P}\right) x(t), \tag{43}$$

where

$$\begin{aligned}
\tilde{P}_1 = \tilde{P}_2 &= \begin{bmatrix} a - c_v & 2 \\ \frac{(a-c_v)(as-c_vs-c_v\theta)}{\theta} & v \end{bmatrix}\\
&\quad \begin{bmatrix} -(2c_v\theta + 2c_vs + \theta a + 2sa) & -5s - \theta - \lambda_1 \\ -(3\theta + 4s) & 0 \end{bmatrix}^{-1}\\
&=: \begin{bmatrix} f_o & g_o \\ h_o & l_o \end{bmatrix},\ i = 1, 2.
\end{aligned}$$

Using the equilibrium actions:

$$u_i^*(t) = (1 + sf_0)x_1(t) + (sg_0 - c_v)x_2(t),$$

the resulting closed-loop system is:

$$\dot{x}(t) = \begin{bmatrix} -\frac{1}{2}s - \frac{1}{2}\lambda_1 & \frac{1}{2}(s - \theta + \lambda_1)\frac{as+(a+2c_v)(\theta+s)}{3\theta+4s} \\ 0 & -\frac{1}{2}\theta \end{bmatrix} x(t).$$

Next, we reformulate this result in terms of our original model parameters. From the above differential equation in x, one obtains the next differential equation for the equilibrium price path $p(t)$:

$$\dot{p}(t) = \frac{1}{2}(\theta - s - \lambda_1)\left[p(t) - \frac{as + (a + 2c_v)(\theta + s)}{3\theta + 4s}\right].$$

The equilibrium actions are:

$$v_i^*(t) = (1 + sf_0)p(t) + sg_0 - c_v, \quad i = 1, 2.$$

For the parameters' values: $a = 4, s = 0.1, c_v = 1.5, \theta = 0.05$ and initial condition $p_0 = [3]$, the toolbox produces the following numerical values (N^e non-cooperative and C^e cooperative equilibrium between players):

1. The equilibrium actions:

$$CL^*(N^e) = \begin{bmatrix} 0.8042 & -1.4385 \\ 0.8042 & -1.4385 \end{bmatrix} \quad \text{and} \quad CL^*(C^e) = \begin{bmatrix} 0.6559 & -1.2898 \\ 0.6559 & -1.2898 \end{bmatrix};$$

2. The closed-loop system:

$$A_{cl}^*(N^e) = \begin{bmatrix} -0.2858 & 0.6877 \\ 0 & -0.0250 \end{bmatrix} \quad \text{and} \quad A_{cl}^*(C^e) = \begin{bmatrix} -0.2562 & 0.6580 \\ 0 & -0.0250 \end{bmatrix};$$

3. Lyapunov equations' solutions (see Eq. 29 in Step A5 of the algorithm)[10]:

$$\tilde{L}_1^*(N^e) = \begin{bmatrix} -0.8411 & 0.5327 \\ 0.5327 & -7.8099 \end{bmatrix}, \quad \tilde{L}_1^*(C^e) = \begin{bmatrix} -0.8603 & 0.5255 \\ 0.5255 & -8.2266 \end{bmatrix};$$
$$\tilde{L}_2^*(N^e) = \begin{bmatrix} -0.8411 & 0.5327 \\ 0.5327 & -7.8099 \end{bmatrix}, \quad \tilde{L}_2^*(C^e) = \begin{bmatrix} -0.8603 & 0.5255 \\ 0.5255 & -8.2266 \end{bmatrix};$$

[10] These matrices can be used to calculate players' losses for any given initial condition p_0.

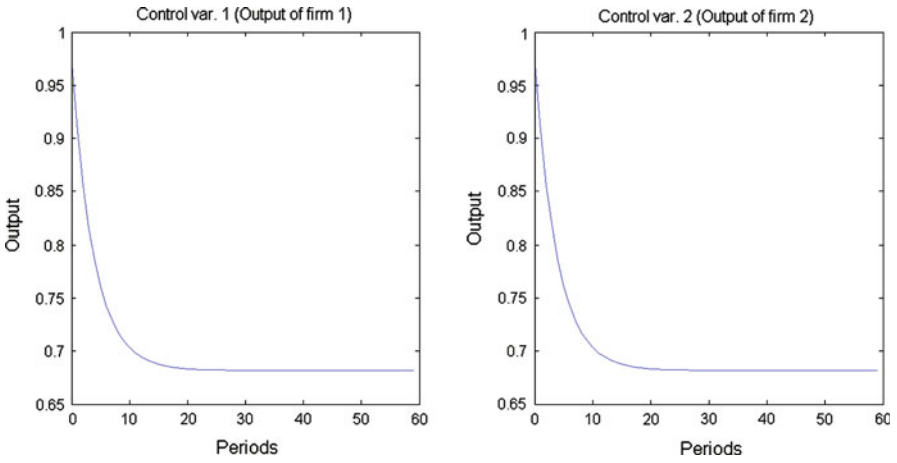

Fig. 3 Control variables in the duopoly game (non-cooperative equilibrium)

4. The optimal losses of players[11]: $J_1^*(N^e) = p_0^T \tilde{L}_1(N^e) p_0 = J_2^*(N^e) = p_0^T \tilde{L}_2(N^e) p_0 = -12.1836$; and $J_1^*(C^e) = p_0^T \tilde{L}_1(C^e) p_0 = J_2^*(C^e) = p_0^T \tilde{L}_2^*(C^e) p_0 = -12.8162$.

The toolbox offers a possibility to plot graphs of control, state and output variables' dynamics. To plot graphs for a particular equilibrium the toolbox command *plot_graph* should be used.[12] Figure 3 shows the dynamics of the control variables for both players (i.e., output of both firms) for the assumed parameter values, whereas Fig. 4 shows the dynamics of the state variables (i.e., market price as well as a constant, which is added to the state variables of the system as outlined in Sect. 2).

Furthermore, as it has been already mentioned, the toolbox offers an option to decompose the loss function of every player into its linear components showing what is each element's contribution to the total outcome. It is especially useful in both model calibration and analysis. When *disaggregate losses* option is chosen the toolbox creates a dummy player j for every non-zero element $\phi_{(a,b),i}$ of matrix Φ_i in the loss function of player i. Player j's matrix Φ_j has $\phi_{(a,b),j}$ as the only non-zero entry. Furthermore, every dummy player is assigned a control variable added to the first state equation of the system with a coefficient 0 so that it does not have any influence on the system. The weight of dummy players' control variables in their loss functions can be defined by the user using the (advanced) option *dummy_control_instrument_multiplier* (δ) whose default value is 1.

In the above example, there are 3 non-zero entries in each Φ_i matrix, namely, $\phi_{(1,3),1}, \phi_{(3,3),1}$, and $\phi_{(3,5),1}$ in Φ_1 and $\phi_{(1,4),2}, \phi_{(4,4),2}$ and $\phi_{(4,5),2}$ in Φ_2. Consequently, the toolbox creates 6 dummy players $j = 3, 4, 5, 6, 7$ and 8 with:

[11] Note that it is not possible to calculate the analytic solution of the corresponding Lyapunov equations, so it is not possible to calculate the analytic formulas for players' losses.

[12] See the toolbox manual for more details.

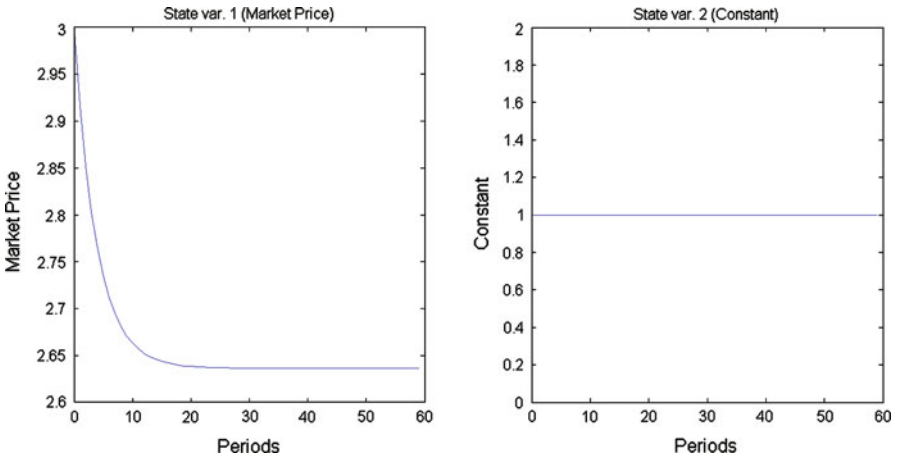

Fig. 4 State variable and constant in the duopoly game (non-cooperative equilibrium)

$$\Phi_3 = \begin{bmatrix} 0 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and analogous Φ_4, Φ_5, Φ_6, Φ_7 and Φ_8.

Furthermore, the toolbox adds 6 new control variables to the model and solves the following game[13]:

$$-\min_{v_i} J_i(v_1, v_2, \ldots, v_8) = -\min_{v_i} \frac{1}{2} \int_0^\infty e^{-\theta t} \left\{ -p(t)v_i(t) + c_v v_i(t) + \frac{1}{2} v_i^2(t) \right\} dt,$$

for $i = 1, 2$ and

$$-\min_{v_3} J_3(v_1, v_2, \ldots, v_8) = -\min_{v_3} \frac{1}{2} \int_0^\infty e^{-\theta t} \{ -p(t)v_1(t) + v_3^2(t) \} dt,$$

[13] Note that dummy players which corresponds to elements that involve a constant are always positioned last.

$$-\min_{v_4} J_3(v_1, v_2, \ldots, v_8) = -\min_{v_4} \frac{1}{2} \int_0^\infty e^{-\theta t} \left\{ \frac{1}{2} v_1^2(t) + v_4^2(t) \right\} dt,$$

$$-\min_{v_5} J_5(v_1, v_2, \ldots, v_8) = -\min_{v_5} \frac{1}{2} \int_0^\infty e^{-\theta t} \{-p(t)v_2(t) + v_5^2(t)\} dt,$$

$$-\min_{v_6} J_6(v_1, v_2, \ldots, v_8) = -\min_{v_6} \frac{1}{2} \int_0^\infty e^{-\theta t} \left\{ \frac{1}{2} v_2^2(t) + v_6^2(t) \right\} dt,$$

$$-\min_{v_7} J_7(v_1, v_2, \ldots, v_8) = -\min_{v_7} \frac{1}{2} \int_0^\infty e^{-\theta t} \{c_v v_1(t) + v_7^2(t)\} dt,$$

$$-\min_{v_8} J_8(v_1, v_2, \ldots, v_8) = -\min_{v_8} \frac{1}{2} \int_0^\infty e^{-\theta t} \{c_v v_2(t) + v_8^2(t)\} dt,$$

subject to dynamics:

$$\dot{p}(t) = s\{a - (v_1(t) + v_2(t)) - p(t)\} + 0v_3 + \cdots + 0v_8, \ p(0) = p_0.$$

Since dummy players have no influence on the system then their optimal strategy is to never use the control instrument as it cannot improve their loss. Consequently, all the obtained results will be the same up to additional dimensions in losses corresponding to dummy players. These losses can be always interpreted in terms of the losses of real players. In our example, since $v_3^2(t)$ and $v_5^2(t)$ are always zero, J_3 and J_5 can be interpreted as the loss from $\int_0^\infty e^{-\theta t}\{-p(t)v_1(t)\}dt$ and $\int_0^\infty e^{-\theta t}\{-p(t)v_2(t)\}dt$ obtained by players 1 and 2, respectively. Thus, J_3^* and J_5^* are revenues from sales, whereas $J_4^* + J_7^*$ and $J_6^* + J_8^*$ are costs functions of both oligopolistic firms. For the parametrisation assumed above the toolbox produces the following disaggregation of optimal losses in both regimes:

	N^e	C^e	Interpretation
J_3^*	−39.4144	−34.2076	Revenue (player 1)
J_4^*	5.3650	3.5454	Cost (player 1)
J_5^*	−39.4144	−34.2076	Revenue (player 2)
J_6^*	5.3650	3.5454	Cost (player 2)
J_7^*	21.8658	17.8460	Cost (player 1)
J_8^*	21.8658	17.8460	Cost (player 2)

which has the following interpretation: while non-cooperating the first/second company has a revenue with an equilibrium of $-J_{3/5}^*(N^e) = 39.4144$ while it incurs a cost of $J_{4/6}^*(N^e) + J_{7/8}^*(N^e) = 5.3650 + 21.8658 = 27.2308$. Total profit for each company is $-J_{3/5}^*(N^e) - J_{4/6}^*(N^e) - J_{7/8}^*(N^e) = 12.1836 = J_{1/2}^*(N^e)$. In contrast, when companies cooperate, their production is much smaller $J_{4/6}^*(C^e) << J_{4/6}^*(N^e)$

but due to a degree of monopolistic power they are able to obtain higher profits: $J^*_{1/2}(C^e) = 12.8162$.

5.2 Example 7.10 of Engwerda (2005) With Multiple Equilibria

A two-player Example 7.10 from Engwerda (2005) can be rewritten using the notation outlined in Sect. 2 as follows:

$$A = \begin{bmatrix} -0.1 & 0 \\ 0 & -2 \end{bmatrix},\ B_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ Q_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0.1 \end{bmatrix},\ Q_2 = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix},$$
$$R_1 = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix},\ R_2 = 1,\ R_{ij} = 0\ (i \neq j),\ V_1 = V_2 = W_1 = W_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \text{ and } N_i = 0.$$

Step T1: LQDG Toolbox Initialisation We can rewrite the above example to fit our standard model (30–31) by specifying:

$$P_1 = P_2 = P_3 = P_{4i} = P_5 = P_6 = P_8 = 0,\ P_7 = A,\ P_{9i} = B_i \text{ and } P_{10} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Notice that in this example $n = m_1 = 2, m_2 = 1$ (and therefore $m = 3$) and $b = 0$. In the performance specification all parameters are zero for both players, except for $\phi_{(1,1),1} = 1$; $\phi_{(2,2),1} = 0.1$; $\phi_{(5,5),1} = 2$; $\phi_{(5,6),1} = -1$; $\phi_{(6,5),1} = -1$; and $\phi_{(6,6),1} = 1$, for the first player, and: $\phi_{(1,1),2} = 1$; $\phi_{(2,1),2} = 1$; $\phi_{(1,2),2} = 1$; $\phi_{(2,2),2} = 2$; and $\phi_{(7,7),2} = 1$ for the second player. In this example, we specify the discount rate to be 0.

Step T2: Model Validation by LQDG Toolbox With this input the toolbox next calculates the standard form (17–18). Clearly, (i) $R_{ii} > 0$; (ii) (A, B_i) is stabilizable; and (iii) G is invertible (see Appendix A for a definition of G).

Step T3: Calculation of Equilibria Again, we will follow Algorithm 5 for the non-cooperative case where matrix M is:

$$\mathcal{M} = -\begin{bmatrix} 0.1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 1 & 2 & 0 & 0 \\ 1 & 0 & -0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & -2 & 0 & 0 \\ 1 & 1 & 0 & 0 & -0.1 & 0 \\ 1 & 2 & 0 & 0 & 0 & -2 \end{bmatrix}.$$

In Step A4 of the algorithm, the toolbox determines the spectrum of $\mathcal{M}$. Numerical calculations show that $\mathcal{M} = TJT^{-1}$, where J is a diagonal matrix with entries $\lambda = \{2; -2.2073; -1.0584; 2.0637; -0.1648; 1.4668\}$ and

$$T = \begin{bmatrix} 0 & 0.2724 & -0.6261 & -0.0303 & -0.1714 & 0.3326 \\ 0 & 0.7391 & 0.5368 & -0.0167 & 0.3358 & 0.0633 \\ 0 & 0.1181 & -0.5405 & 0.0154 & -0.6473 & -0.2433 \\ 0 & 0.0176 & 0.0176 & 0.0262 & 0.0155 & 0.0119 \\ 0 & 0.4384 & -0.0771 & 0.0239 & 0.6207 & -0.2897 \\ 1 & 0.4161 & 0.1463 & 0.9987 & 0.2311 & 0.8614 \end{bmatrix}.$$

Let us factorise matrix T into $[T_1\ T_2\ T_3\ T_4\ T_5\ T_6]$. Matrix $\mathcal{M}$ has six different eigenvalues, where three of them are negative, i.e., $\lambda^s = \{-2.2073; -1.0584; -0.1648\}$. They correspond to eigenvectors T_2, T_3 and T_5. Since no eigenvalue has an algebraic multiplicity higher than one and there are no complex eigenvalues, the toolbox determines in Step A4.2.2.1 of the algorithm that there are at most $\binom{3}{2} = 3$ different equilibrium strategies that permit a feedback synthesis, i.e., $C^{\bar{n}}_{\lambda^s} = \{(-2.2073, -1.0584), (-2.2073, -0.1648), (-1.0584, -0.1648)\}$ or $\tilde{V}^s \in \{(T_2, T_3), (T_2, T_5), (T_3, T_5)\}$. For all three combinations of two eigenvectors $\tilde{V}^s$ the toolbox executes Step A5.

Consider $\mathcal{P}_1 := \mathrm{Im}[T_2\ T_3]$. The first 2×2 block of this matrix equals $\begin{bmatrix} 0.2724 & -0.6261 \\ 0.7391 & 0.5368 \end{bmatrix}$. This matrix is invertible. $\mathcal{P}_1$ is an element of $\mathcal{P}^{pos}$ and $\sigma(\mathcal{M}|_{\mathcal{P}_1}) = \{-2.2073, -1.0584\}$. In a similar way it can be verified that $\mathcal{P}_2 := \mathrm{Im}[T_2\ T_5]$ and $\mathcal{P}_3 := \mathrm{Im}[T_3\ T_5]$ are also appropriate graph subspaces. On the whole, all three $\mathcal{M}$-invariant subspaces satisfy all conditions.

As an example, we will calculate the equilibrium strategy that permits a feedback synthesis resulting from $\mathcal{P}_3$. To that end we factorise $\mathcal{P}_3$ as follows:

$$\mathcal{P}_3 = \begin{bmatrix} -0.6261 & -0.1714 \\ 0.5368 & 0.3358 \\ -0.5405 & -0.6473 \\ 0.0176 & 0.0155 \\ -0.0771 & 0.6207 \\ 0.1463 & 0.2311 \end{bmatrix} =: \begin{bmatrix} X \\ Y \\ Z \end{bmatrix},$$

where X, Y and Z are 2×2 matrices. We obtain then:

$$\tilde{P}_1 := YX^{-1} = \begin{bmatrix} -0.5405 & -0.6473 \\ 0.0176 & 0.0155 \end{bmatrix} \begin{bmatrix} -0.6261 & -0.1714 \\ 0.5368 & 0.3358 \end{bmatrix}^{-1} \text{ and}$$

$$\tilde{P}_2 := ZX^{-1} = \begin{bmatrix} -0.0771 & 0.6207 \\ 0.1463 & 0.2311 \end{bmatrix} \begin{bmatrix} -0.6261 & -0.1714 \\ 0.5368 & 0.3358 \end{bmatrix}^{-1}.$$

The corresponding open-loop Nash strategy is $u_i^*(t) := -R_i^{-1} B_i^T \tilde{P}_i e^{A_{cl}t} x_0$. Here the closed-loop matrix $A_{cl} := A - S_1\tilde{P}_1 - S_2\tilde{P}_2$ is $\begin{bmatrix} -1.7538 & -0.8112 \\ 1.3622 & 0.5305 \end{bmatrix}$. It is easily verified that the spectrum of this matrix A_{cl} indeed equals $\{-1.0584, -0.1648\}$. The cost for both players in this equilibrium is obtained by solving the corresponding

Lyapunov equation (29). This gives a cost:

$$J_1(N_3^e) = x_0^T \begin{bmatrix} 10.2694 & 12.6875 \\ 12.6875 & 16.1956 \end{bmatrix} x_0 \quad \text{and} \quad J_2(N_3^e) = x_0^T \begin{bmatrix} 15.7425 & 18.3929 \\ 18.3929 & 21.7370 \end{bmatrix} x_0.$$

The other two non-cooperative equilibria are:

$$J_1(N_1^e) = x_0^T \begin{bmatrix} 0.3207 & -0.0476 \\ -0.0476 & 0.0221 \end{bmatrix} x_0 \quad \text{and} \quad J_2(N_1^e) = x_0^T \begin{bmatrix} 0.1708 & 0.1065 \\ 0.1065 & 0.2636 \end{bmatrix} x_0$$

as well as

$$J_1(N_2^e) = x_0^T \begin{bmatrix} 7.9338 & -2.7856 \\ -2.7856 & 1.0062 \end{bmatrix} x_0 \quad \text{and} \quad J_2(N_2^e) = x_0^T \begin{bmatrix} 8.1006 & -3.2333 \\ -3.2333 & 1.6482 \end{bmatrix} x_0.$$

Thus, there are altogether 3 non-cooperative equilibria N_1^e, N_2^e and N_3^e. Assuming $x_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ we obtain the following optimal losses of players in this regime:

	N_1^e	N_2^e	N_3^e
J_1	0.2476	3.3688	51.8401
J_2	0.6473	3.2821	74.2653

The key question is whether it is possible to discriminate between multiple equilibria of a particular coalition structure. It is especially important in more complex games, where a practically intractable number of equilibria can be obtained. One of the widely accepted solution concepts to discriminate between various equilibria of a game is *Pareto*-domination. An equilibrium is said to be *Pareto*-dominated if it is possible to find another equilibrium in which all the players in the game will not be worse off and at least one of them will be better off. If option *discriminate between equilibria* using *Pareto domination* is chosen, the toolbox is going to report a list of *Pareto*-undominated equilibria. In the above case, it is clear that in the non-cooperative regime equilibrium N_1^e dominates the other two as $J_1(N_1^e) < J_1(N_2^e) < J_1(N_3^e)$ and at the same time $J_2(N_1^e) < J_2(N_2^e) < J_2(N_3^e)$. Consequently, apart from all the equilibria, the toolbox is going to report also a set of *Pareto*-undominated equilibria which consists of N_1^e.

The particular advantage of the Pareto concept is based on the fact that it is in the interest of all the players to play only *Pareto* -undominated equilibria. However, the downside is that there can be a whole spectrum of them. Because of this, we propose another two ways to discriminate between multiple equilibria. The first method is based on the concept of so called *social optimum*, i.e., such an equilibrium is chosen that minimises the sum of all the players' losses, or, in our two-player game: $J_1(N_1^e) + J_2(N_1^e)$. In our example $J_1(N_1^e) + J_2(N_1^e) < J_1(N_2^e) + J_2(N_2^e) < J_1(N_3^e) + J_2(N_3^e)$ so the first equilibrium is chosen when this option is on. The other method we propose to discriminate between multiple equilibria refers to the adjustment speed of the equilibrium closed-loop system towards its long-term equilibrium, measured by the smallest

absolute value of the eigenvalues of the A_{cl} matrix that are located in the left half of the complex plane. In our example:

$$A_{cl}(N_1^e) = \begin{bmatrix} -1.7538 & -0.8112 \\ 1.3622 & 0.5305 \end{bmatrix}, \quad \lambda^s(N_1^e) = \{-1.0584,\ -2.2073\};$$
$$A_{cl}(N_2^e) = \begin{bmatrix} -1.0212 & -0.4372 \\ -2.3234 & -1.3510 \end{bmatrix}, \quad \lambda^s(N_2^e) = \{-0.1648,\ -2.2073\};\ \text{and}$$
$$A_{cl}(N_3^e) = \begin{bmatrix} -1.7538 & -0.8112 \\ 1.3622 & 0.5305 \end{bmatrix}, \quad \lambda^s(N_3^e) = \{-1.0584,\ -0.1648\}.$$

Consequently, the toolbox will choose the equilibrium such that their graph subspace consists of eigenvector corresponding to $-1.0584, -2.2073$, i.e., N_1^e.

5.3 Example 7.12 of Engwerda (2005) With Complex Eigenvalues

A two-player Example 7.12 from Engwerda (2005) can be rewritten in terms of notation outlined in Sect. 2 as follows:

$$A = \begin{bmatrix} -\frac{1}{2} & 0 \\ 0 & -\frac{1}{4} \end{bmatrix},\ B_1 = B_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ Q_1 = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1 & 3 \end{bmatrix},\ Q_2 = \frac{1}{2}\begin{bmatrix} 2 & -\frac{7}{9} \\ -\frac{7}{9} & 1 \end{bmatrix},$$
$$R_1^{-1} = \frac{1}{2}\begin{bmatrix} 1 & -\frac{7}{90} \\ -\frac{7}{90} & 1 \end{bmatrix},\ R_2^{-1} = \frac{1}{2}\begin{bmatrix} 1 & -\frac{1}{10} \\ -\frac{1}{10} & \frac{3}{4} \end{bmatrix},$$
$$V_1 = V_2 = W_1 = W_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \text{and } N_i = 0.$$

Step T1: LQDG Toolbox Initialisation We can rewrite the above example to fit our standard model (30–31) by specifying:

$$P_1 = P_2 = P_3 = P_{4i} = P_5 = P_6 = P_8 = 0,\ P_7 = A,\ P_{9i} = B_i \text{ and } P_{10} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Notice that in this example $n = m_1 = m_2 = 2$ (and therefore $m = 4$) and $b = 0$. In the performance specification, all parameters are zero for both players, except for $\phi_{(1,1),1} = \frac{1}{2}, \phi_{(1,2),1} = \frac{1}{2}, \phi_{(2,1),1} = \frac{1}{2}, \phi_{(2,2),1} = \frac{3}{2}, \phi_{(5,5),1} = r_{11}, \phi_{(5,6),1} = r_{12}, \phi_{(6,5),1} = r_{21}, \phi_{(6,6),1} = r_{22}, \phi_{(1,1),2} = 1, \phi_{(1,2),2} = -\frac{7}{18}, \phi_{(2,1),2} = -\frac{7}{18}, \phi_{(2,2),2} = \frac{1}{2}, \phi_{(7,7),2} = s_{11}, \phi_{(7,8),2} = s_{12}, \phi_{(8,7),2} = s_{21}$, and $\phi_{(8,8),2} = s_{22}$, where:

$$\begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} = \left(\frac{1}{2}\begin{bmatrix} 1 & -\frac{7}{90} \\ -\frac{7}{90} & 1 \end{bmatrix}\right)^{-1} \text{ and } \begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix} = \left(\frac{1}{2}\begin{bmatrix} 1 & -\frac{1}{10} \\ -\frac{1}{10} & \frac{3}{4} \end{bmatrix}\right)^{-1},$$

for the first and second players, respectively. Also in this example, we specify the discount rate to be 0.

Step T2: Model Validation by LQDG Toolbox With this input, the toolbox next calculates the standard form (17–18) and checks that both Q_i and R_i are positive definite, (A, B_i) are stabilizable and G is invertible.

Step T3: Calculation of Equilibria Again, we will follow Algorithm 5 for the non-cooperative case. We have that:

$$\mathcal{M} = -\begin{bmatrix} 0.5000 & 0.0000 & 0.5000 & -0.0389 & 0.5000 & -0.0500 \\ 0.0000 & 0.2500 & -0.0389 & 0.5000 & -0.0500 & 0.3750 \\ 0.5000 & 0.5000 & -0.5000 & 0.0000 & 0.0000 & 0.0000 \\ 0.5000 & 1.5000 & 0.0000 & -0.2500 & 0.0000 & 0.0000 \\ 1.0000 & -0.3889 & 0.0000 & 0.0000 & -0.5000 & 0.0000 \\ -0.3889 & 0.5000 & 0.0000 & 0.0000 & 0.0000 & -0.2500 \end{bmatrix}.$$

In Step A4 of the algorithm the toolbox determines the spectrum of M. Numerical calculations show that $M = TJT^{-1}$, where J is a diagonal matrix with entries $\lambda = \{-1.0004 + 0.0227i; -1.0004 - 0.0227i; 0.2525; 0.4983; 1; 1\}$ and

$$T = \begin{bmatrix} 0.1358 + 0.4406i & 0.1358 - 0.4406i & -0.0045 & -0.0002 & 0.3848 & 0.0003 \\ 0.4499 - 0.1379i & 0.4499 + 0.1379i & 0.0005 & 0.0026 & 0.0134 & -0.3767 \\ 0.1966 + 0.0979i & 0.1966 - 0.0979i & -0.0080 & 0.7063 & -0.3982 & 0.3764 \\ 0.5940 & 0.5940 & 0.5979 & -0.0156 & -0.2833 & 0.7531 \\ -0.0211 + 0.3297i & -0.0211 - 0.3297i & -0.0189 & -0.7077 & -0.7593 & -0.2935 \\ 0.1341 - 0.1946i & 0.1341 + 0.1946i & -0.8013 & -0.0056 & 0.1906 & 0.2513 \end{bmatrix}.$$

Matrix $\mathcal{M}$ has two (complex) eigenvalues with a negative real part. Let x be the real part of the eigenvector corresponding with the eigenvalue $-1.0004 + 0.0227i$ and y the imaginary part of this eigenvector

$$x^T := [0.1358,\ 0.4499,\ 0.1966,\ 0.5940,\ -0.0211,\ 0.1341] \text{ and}$$
$$y^T := [0.4406,\ -0.1379,\ 0.0979,\ 0,\ 0.3279,\ -0.1964].$$

The invariant subspace corresponding with eigenvalues $\{-1.0004 + 0.0227i; -1.0004 - 0.0227i\}$ is $S = \text{Im}[x\ y]$. According to Sect. 2, the unique equilibrium actions are $u_i^*(t) = -R_i^{-1} B_i^T \tilde{P}_i x(t)$, where:

$$\tilde{P}_1 = \begin{bmatrix} 0.1966 & 0.0979 \\ 0.5940 & 0 \end{bmatrix} \begin{bmatrix} 0.1358 & 0.4406 \\ 0.4499 & -0.1379 \end{bmatrix}^{-1} = \begin{bmatrix} 0.3280 & 0.3380 \\ 0.3776 & 1.2063 \end{bmatrix} \text{ and}$$
$$\tilde{P}_2 = \begin{bmatrix} -0.0211 & 0.3297 \\ 0.1341 & -0.1964 \end{bmatrix} \begin{bmatrix} 0.1358 & 0.4406 \\ 0.4499 & -0.1379 \end{bmatrix}^{-1} = \begin{bmatrix} 0.6703 & -0.2493 \\ -0.3183 & 0.3942 \end{bmatrix}.$$

The with these actions corresponding closed-loop system matrix is then:

$$A_{cl} := \begin{bmatrix} -1.0004 & 0.0222 \\ -0.0231 & -1.003 \end{bmatrix}.$$

The eigenvalues of this matrix are $\{-1.0004 + 0.0227i; -1.0004 - 0.0227i\}$. The corresponding equilibrium costs are

$$J_1(N^e) = x_0^T \tilde{L}_1(N^e)x_0 = \frac{1}{2}x_0^T \begin{bmatrix} 0.2990 & 0.3715 \\ 0.3715 & 1.1344 \end{bmatrix} x_0 \quad \text{and}$$
$$J_2(N^e) = x_0^T \tilde{L}_2(N^e)x_0 = \frac{1}{2}x_0^T \begin{bmatrix} 0.6479 & -0.2644 \\ -0.2644 & 0.2936 \end{bmatrix} x_0.$$

6 Concluding Remarks

In this paper we considered a dynamic linear affine structural form model that is affected by different players who all like to minimise their own performance criterion that is a quadratic affine function of the variables occurring in the model. The costs are assumed to be discounted over time and the considered planning horizon by the players is assumed to be infinite. Under the assumption that in the minimisation of their performance the players do not cooperate, we presented both necessary and sufficient conditions under which this problem has a unique open-loop Nash equilibrium, a multiple but finite number of equilibria and or an infinite number of equilibria. A computational framework was provided for how one can numerically solve the problem. The algorithm has been implemented in a form of a numerical toolbox available on the internet. Users, starting from the structural model, can calculate for their specific application the equilibrium strategies and involved cost (if they exist). The toolbox also provides the possibility to calculate for different coalition structures whether the corresponding game will have an open-loop Nash solution. For that purpose the user has to define which coalition structures they like to analyze and what the relative importance is of each player within a certain coalition. We demonstrated both theoretically and numerically in a worked example on dynamic duopolistic competition the use of the toolbox.

LQDG Toolbox is implemented in MATLAB. In particular, it uses some standard functions of MATLAB to calculate the eigenstructure of a $(|N|\bar{n}) \times (|N|\bar{n})$ matrix, where $|N|$ is the number of involved players and $\bar{n}$ is the state dimension of the model. Since no additional efforts are taken to calculate this eigenstructure in a numerically efficient way, the practical use of the current toolbox is limited to some extent. This is because for either a large number of players and/or a large state dimension, the accuracy and efficiency is restricted by that of the implemented MATLAB functions. So for large N and/or $\bar{n}$ the user should look for an own code to implement the algorithm. Another way one might choose to calculate the equilibrium strategies is by using iterative algorithms. In the literature a number of iterative schemes have been suggested (see e.g. Engwerda 2007). A disadvantage of these schemes is that on the one hand they do not provide an answer to the question whether the game will have a unique equilibrium. On the other hand these schemes may converge without providing the appropriate equilibrium strategy. If this happens one is stuck with the question how to proceed.

For the corresponding problem with a finite planning horizon, at least from a theoretical point, it is clear under which conditions there exists a unique equilibrium

(see e.g. Engwerda 2005). From a computational point it is also clear how one can calculate this equilibrium. Either one can solve the involved set of nonlinear differential two-point boundary-value equations directly using standard MATLAB functions. Another possibility is to transform the involved set of Riccati differential equations (in the spirit of Reid (1972)) to a set of linear differential equations and then solve this set first (see e.g. Tabak 1975; Engwerda 2007). Since the calculations require the numerical solution of a set of (nonlinear) differential equations the dimension of the games for which one can still calculate the equilibrium actions (using standard MATLAB functions) without problems is usually smaller than in the infinite horizon case.

In the literature, also different equilibrium concepts have been studied for games considered above. Probably the most well-known is the feedback Nash concept. Unfortunately, for this case there are no general conditions known, except for the scalar case, i.e., $\bar{n} = 1$ (see e.g. Engwerda 2005) under which the game has a unique feedback Nash equilibrium.

Finally, we would like to mention that for discrete time systems much work has been done by Neck et al. in the development of the numerical software OPTGAME for the calculation of Nash equilibria in (non-)linear systems in case the performances of players are quadratic (see e.g. Neck et al. 2001).

Acknowledgments Tomasz Michalak acknowledges support from (a) the EPSRC under the project ALADDIN (Autonomous Learning Agents for Decentralised Data and Information Systems) project and is jointly funded by a BAE Systems and EPSRC strategic partnership; and (b) the FWO (Fonds voor Wetenschappelijk Onderzoek Vlaanderen, Belgium).

Appendix A: Notation

The following shorthand notation is used:

$$S_i := B_i R_{ii}^{-1} B_i^T, \quad G := \begin{bmatrix} [0 & I & 0] & M_1 \\ [0 & 0 & I] & M_2 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ I & 0 \\ 0 & I \end{bmatrix} = \begin{bmatrix} R_{11} & N_1 \\ N_2^T & R_{22} \end{bmatrix},$$

where we assume throughout that this matrix G is invertible. Furthermore:

$$A_2 := \begin{bmatrix} A & 0 \\ 0 & A \end{bmatrix}, \; B := [B_1, \; B_2], \; \widetilde{B}^T := \begin{bmatrix} B_1^T & 0 \\ 0 & B_2^T \end{bmatrix},$$
$$\widetilde{B}_1^T := \begin{bmatrix} B_1^T \\ 0 \end{bmatrix}, \; \widetilde{B}_2^T := \begin{bmatrix} 0 \\ B_2^T \end{bmatrix},$$
$$Q := \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}, \; Z := \begin{bmatrix} [0 & I & 0] M_1 \\ [0 & 0 & I] M_2 \end{bmatrix} \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} V_1^T \\ W_2^T \end{bmatrix},$$

$$Z_i := [I \quad 0 \quad 0] M_i \begin{bmatrix} 0 & 0 \\ I & 0 \\ 0 & I \end{bmatrix} = [V_i, \ W_i], \ i = 1, 2,$$

$$\widetilde{A} := A - BG^{-1}Z, \ \widetilde{A}_2^T := A_2^T - \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} G^{-1}\widetilde{B}^T,$$

$$\widetilde{Q}_i := Q_i - Z_i G^{-1} Z, \ \widetilde{Q} := \begin{bmatrix} \widetilde{Q}_1 \\ \widetilde{Q}_2 \end{bmatrix}, \ \widetilde{S}_i := BG^{-1}\widetilde{B}_i^T,$$

$$\widetilde{S} := [\widetilde{S}_1, \ \widetilde{S}_2], \text{ and } \mathcal{M} := \begin{bmatrix} \widetilde{A} & -\widetilde{S} \\ -\widetilde{Q} & -\widetilde{A}_2^T \end{bmatrix}.$$

Notice that:

$$\mathcal{M} = H_1 + H_2 G^{-1} H_3,$$

where:

$$H_1 := \begin{bmatrix} A & 0 & 0 \\ -Q_1 & -A^T & 0 \\ -Q_2 & 0 & -A^T \end{bmatrix}, \quad H_2 := \begin{bmatrix} -B \\ Z_1 \\ Z_2 \end{bmatrix} \quad \text{and} \quad H_3 := \begin{bmatrix} Z, \ \tilde{B}_1^T, \ \tilde{B}_2^T \end{bmatrix}.$$

Matrices H_1, H_2, H_3 and G to construct the $\mathcal{M}$ are stored in the LQDG Toolbox output file.

Appendix B: Cost Function Submatrices

$$\Phi_{p^2,i} := \begin{bmatrix} \phi_{(1,1),i}\, p_1^2 & \phi_{(1,2),i}\, p_1 p_2 & \cdots & \phi_{(1,n),i}\, p_1 p_n \\ \phi_{(2,1),i}\, p_2 p_1 & \phi_{(2,2),i}\, p_2^2 & \cdots & \phi_{(2,n),i}\, p_2 p_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n,1),i}\, p_n p_1 & \phi_{(n,2),i}\, p_n p_2 & \cdots & \phi_{(n,n),i}\, p_n^2 \end{bmatrix},$$

$$\Phi_{\dot{p}p,i} := \begin{bmatrix} \phi_{(n+1,1),i}\, \dot{p}_1 p_1 & \phi_{(n+1,2),i}\, \dot{p}_1 p_2 & \cdots & \phi_{(n+1,n),i}\, \dot{p}_1 p_n \\ \phi_{(n+2,1),i}\, \dot{p}_2 p_1 & \phi_{(n+2,2),i}\, \dot{p}_2 p_2 & \cdots & \phi_{(n+2,n),i}\, \dot{p}_2 p_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n+n,1),i}\, \dot{p}_n p_1 & \phi_{(n+n,2),i}\, \dot{p}_n p_2 & \cdots & \phi_{(n+n,n),i}\, \dot{p}_n p_n \end{bmatrix},$$

$$\Phi_{yp,i} := \begin{bmatrix} \phi_{(2n+1,1),i}\, y_1 p_1 & \phi_{(2n+1,2),i}\, y_1 p_2 & \cdots & \phi_{(2n+1,n),i}\, y_1 p_n \\ \phi_{(2n+2,1),i}\, y_2 p_1 & \phi_{(2n+2,2),i}\, y_2 p_2 & \cdots & \phi_{(2n+2,n),i}\, y_2 p_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b,1),i}\, y_b p_1 & \phi_{(2n+b,2),i}\, y_b p_2 & \cdots & \phi_{(2n+b,n),i}\, y_b p_n \end{bmatrix},$$

$$\Phi_{vp,i} := \begin{bmatrix} \phi_{(2n+b+1,1),i}\, v_1 p_1 & \phi_{(2n+b+1,2),i}\, v_1 p_2 & \cdots & \phi_{(2n+b+1,n),i}\, v_1 p_n \\ \phi_{(2n+b+2,1),i}\, v_2 p_1 & \phi_{(2n+b+2,2),i}\, v_2 p_2 & \cdots & \phi_{(2n+b+2,n),i}\, v_2 p_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b+m,1),i}\, v_m p_1 & \phi_{(2n+b+m,2),i}\, v_m p_2 & \cdots & \phi_{(2n+b+m,n),i}\, v_m p_n \end{bmatrix},$$

$$\Phi_{cp,i} := \begin{bmatrix} \phi_{(2n+b+m+1,1),i}\, c p_1 & \phi_{(2n+b+1,2,i)} c p_2 & \cdots & \phi_{(2n+b+1,n,i)} c p_n \end{bmatrix},$$

$$\Phi_{p\dot{p},i} := \begin{bmatrix} \phi_{(1,n+1),i}\, p_1\dot{p}_1 & \phi_{(1,n+2),i}\, p_1\dot{p}_2 & \cdots & \phi_{(1,n+n),i}\, p_1\dot{p}_n \\ \phi_{(2,n+1),i}\, p_2\dot{p}_1 & \phi_{(2,n+2),i}\, p_2\dot{p}_2 & \cdots & \phi_{(2,n+n),i}\, p_2\dot{p}_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n,n+1),i}\, p_n\dot{p}_1 & \phi_{(n,n+2),i}\, p_n\dot{p}_2 & \cdots & \phi_{(n,n+n),i}\, p_n\dot{p}_n \end{bmatrix},$$

$$\Phi_{\dot{p}^2,i} := \begin{bmatrix} \phi_{(n+1,n+1),i}\, \dot{p}_1^2 & \phi_{(n+1,n+2),i}\, \dot{p}_1\dot{p}_2 & \cdots & \phi_{(n+1,n+n),i}\, \dot{p}_1\dot{p}_n \\ \phi_{(n+2,n+1),i}\, \dot{p}_2\dot{p}_1 & \phi_{(n+2,n+2),i}\, \dot{p}_2^2 & \cdots & \phi_{(n+2,n+n),i}\, \dot{p}_1\dot{p}_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n+n,n+1),i}\, \dot{p}_n\dot{p}_1 & \phi_{(n+n,n+2),i}\, \dot{p}_n\dot{p}_2 & \cdots & \phi_{(n+n,n+n),i}\, \dot{p}_n^2 \end{bmatrix},$$

$$\Phi_{y\dot{p},i} := \begin{bmatrix} \phi_{(2n+1,1,i)}\, y_1\dot{p}_1 & \phi_{(2n+1,2,i)}\, y_1\dot{p}_2 & \cdots & \phi_{(2n+1,n),i}\, y_1\dot{p}_n \\ \phi_{(2n+2,1,i)}\, y_2\dot{p}_1 & \phi_{(2n+2,2,i)}\, y_2\dot{p}_2 & \cdots & \phi_{(2n+2,n),i}\, y_2\dot{p}_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b,1,i)}\, y_b\dot{p}_1 & \phi_{(2n+b,2,i)}\, y_b\dot{p}_2 & \cdots & \phi_{(2n+b,n),i}\, y_b\dot{p}_n \end{bmatrix},$$

$$\Phi_{v\dot{p},i} := \begin{bmatrix} \phi_{(2n+b+1,n+1,i)}\, v_1\dot{p}_1 & \phi_{(2n+b+1,n+2,i)}\, v_1\dot{p}_2 & \cdots & \phi_{(2n+b+1,n+n),i}\, v_1\dot{p}_n \\ \phi_{(2n+b+2,n+1,i)}\, v_2\dot{p}_1 & \phi_{(2n+b+2,n+2,i)}\, v_2\dot{p}_2 & \cdots & \phi_{(2n+b+2,n+n),i}\, v_2\dot{p}_n \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b+m,n+1,i)}\, v_m\dot{p}_1 & \phi_{(2n+b+m,n+2,i)}\, v_m\dot{p}_2 & \cdots & \phi_{(2n+b+m,n+n),i}\, v_m\dot{p}_n \end{bmatrix},$$

$$\Phi_{c\dot{p},i} := \begin{bmatrix} \phi_{(2n+b+m+1,n+1),i}\, c\dot{p}_1 & \phi_{(2n+b+1,n+2,i)}\, c\dot{p}_2 & \cdots & \phi_{(2n+b+1,n+n,i)}\, c\dot{p}_n \end{bmatrix},$$

$$\Phi_{py,i} := \begin{bmatrix} \phi_{(1,2n+1),i}\, p_1y_1 & \phi_{(1,2n+2),i}\, p_1y_2 & \cdots & \phi_{(1,2n+b),i}\, p_1y_b \\ \phi_{(2,2n+1),i}\, p_2y_1 & \phi_{(2,2n+2),i}\, p_2y_2 & \cdots & \phi_{(2,2n+b),i}\, p_2y_b \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n,2n+1),i}\, p_ny_1 & \phi_{(n,2n+2),i}\, p_ny_2 & \cdots & \phi_{(n,2n+b),i}\, p_ny_b \end{bmatrix},$$

$$\Phi_{\dot{p}y,i} := \begin{bmatrix} \phi_{(n+1,2n+1),i}\, \dot{p}_1y_1 & \phi_{(n+1,2n+2),i}\, \dot{p}_1y_2 & \cdots & \phi_{(n+1,2n+b),i}\, \dot{p}_1y_b \\ \phi_{(n+2,2n+1),i}\, \dot{p}_2y_1 & \phi_{(n+2,2n+2),i}\, \dot{p}_2y_2 & \cdots & \phi_{(n+2,2n+b),i}\, \dot{p}_2y_b \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n+n,2n+1),i}\, \dot{p}_ny_1 & \phi_{(n+n,2n+2),i}\, \dot{p}_ny_2 & \cdots & \phi_{(n+n,2n+b),i}\, \dot{p}_ny_b \end{bmatrix},$$

$$\Phi_{y^2,i} := \begin{bmatrix} \phi_{(2n+1,2n+1),i}\, y_1^2 & \phi_{(2n+1,2n+2),i}\, y_1y_2 & \cdots & \phi_{(2n+1,2n+b),i}\, y_1y_b \\ \phi_{(2n+2,2n+1),i}\, y_2y_1 & \phi_{(2n+2,n+2),i}\, y_2^2 & \cdots & \phi_{(2n+2,2n+b),i}\, y_2y_b \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b,2n+1),i}\, y_by_1 & \phi_{(2n+b,2n+2),i}\, y_by_2 & \cdots & \phi_{(2n+b,2n+1),i}\, y_b^2 \end{bmatrix},$$

$$\Phi_{vy,i} := \begin{bmatrix} \phi_{(2n+b+1,2n+1,i)}\, v_1y_1 & \phi_{(2n+b+1,2n+2,i)}\, v_1y_2 & \cdots & \phi_{(2n+b+1,2n+b),i}\, v_1y_b \\ \phi_{(2n+b+2,2n+1,i)}\, v_2y_1 & \phi_{(2n+b+2,2n+2,i)}\, v_2y_2 & \cdots & \phi_{(2n+b+2,2n+b),i}\, v_2y_b \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b+m,2n+1,i)}\, v_my_1 & \phi_{(2n+b+m,2n+2,i)}\, v_my_2 & \cdots & \phi_{(2n+b+m,2n+b),i}\, v_my_b \end{bmatrix},$$

$$\Phi_{cy,i} := \begin{bmatrix} \phi_{(2n+b+m+1,2n+1,i)}\, cy_1 & \phi_{(2n+b+m+1,2n+2,i)}\, cy_2 & \cdots & \phi_{(2n+b+m+1,2n+b),i}\, cy_b \end{bmatrix},$$

$$\Phi_{pv,i} := \begin{bmatrix} \phi_{(1,2n+b+1),i}\, p_1v_1 & \phi_{(1,2n+b+2),i}\, p_1v_2 & \cdots & \phi_{(1,2n+b+m),i}\, p_1v_m \\ \phi_{(2,2n+b+1),i}\, p_2v_1 & \phi_{(2,2n+b+2),i}\, p_2v_2 & \cdots & \phi_{(2,2n+b+m),i}\, p_2v_m \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n,2n+b+1),i}\, p_nv_1 & \phi_{(n+n,2n+b+2),i}\, p_nv_2 & \cdots & \phi_{(n,2n+b+m),i}\, p_nv_m \end{bmatrix},$$

$$\Phi_{\dot{p}v,i} := \begin{bmatrix} \phi_{(n+1,2n+b+1),i}\, \dot{p}_1v_1 & \phi_{(n+1,2n+b+2),i}\, \dot{p}_1v_2 & \cdots & \phi_{(n+1,2n+b+m),i}\, \dot{p}_1v_m \\ \phi_{(n+2,2n+b+1),i}\, \dot{p}_2v_1 & \phi_{(n+2,2n+b+2),i}\, \dot{p}_2v_2 & \cdots & \phi_{(n+2,2n+b+m),i}\, \dot{p}_2v_m \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(n+n,2n+b+1),i}\, \dot{p}_nv_1 & \phi_{(n+n,2n+b+2),i}\, \dot{p}_nv_2 & \cdots & \phi_{(n+n,2n+b+m),i}\, \dot{p}_nv_m \end{bmatrix},$$

$$\Phi_{yv,i} := \begin{bmatrix} \phi_{(2n+1,2n+b+1),i}\, y_1 v_1 & \phi_{(2n+1,2n+b+2),i}\, y_1 v_2 & \cdots & \phi_{(2n+1,2n+b+m),i}\, y_1 v_m \\ \phi_{(2n+2,2n+b+1),i}\, y_2 v_1 & \phi_{(2n+2,2n+b+2),i}\, y_2 v_2 & \cdots & \phi_{(2n+2,2n+b+m),i}\, y_2 v_m \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b,2n+b+1),i}\, y_b v_1 & \phi_{(2n+b,2n+b+2),i}\, y_b v_2 & \cdots & \phi_{(2n+b,2n+b+m),i}\, y_b v_m \end{bmatrix},$$

$$\Phi_{v^2,i} := \begin{bmatrix} \phi_{(2n+b+1,2n+b+1),i}\, v_1^2 & \phi_{(2n+b+1,2n+b+2),i}\, v_1 v_2 & \cdots & \phi_{(2n+b+1,2n+b+m),i}\, v_1 v_m \\ \phi_{(2n+b+2,2n+b+1),i}\, v_2 v_1 & \phi_{(2n+b+2,2n+b+2),i}\, v_2 v_2 & \cdots & \phi_{(2n+b+2,2n+b+m),i}\, v_2 v_m \\ \cdots & \cdots & \cdots & \cdots \\ \phi_{(2n+b+m,2n+b+1),i}\, v_m v_1 & \phi_{(2n+b+m,2n+b+2),i}\, v_m v_2 & \cdots & \phi_{(2n+b+m,2n+b+m),i}\, v_m^2 \end{bmatrix},$$

$$\Phi_{cv,i} := \begin{bmatrix} \phi_{(2n+b+m+1,2n+b+1,i)} c v_1 & \phi_{(2n+b+m+1,2n+b+2,i)} c v_2 & \cdots & \phi_{(2n+b+m+1,2n+b+m),i} c v_m \end{bmatrix},$$

$$\Phi_{pc,i} := \begin{bmatrix} \phi_{(1,2n+b+m+1),i}\, p_1 c \\ \phi_{2,2n+b+m+1),i}\, p_2 c \\ \cdots \\ \phi_{n,2n+b+m+1),i}\, p_n c \end{bmatrix}, \Phi_{\dot{p}c,i} := \begin{bmatrix} \phi_{(n+1,2n+b+m+1),i}\, \dot{p}_1 c \\ \phi_{n+2,2n+b+m+1),i}\, \dot{p}_2 c \\ \cdots \\ \phi_{n+n,2n+b+m+1),i}\, \dot{p}_n c \end{bmatrix},$$

$$\Phi_{yc,i} := \begin{bmatrix} \phi_{(2n+1,2n+b+m+1),i}\, y_1 c \\ \phi_{2n+2,2n+b+m+1),i}\, y_2 c \\ \cdots \\ \phi_{2n+m,2n+b+m+1),i}\, y_b c \end{bmatrix}, \Phi_{vc,i} := \begin{bmatrix} \phi_{(2n+b+1,2n+b+m+1),i}\, v_1 c \\ \phi_{2n+b+2,2n+b+m+1),i}\, v_2 c \\ \cdots \\ \phi_{2n+b+m,2n+b+m+1),i}\, v_n c \end{bmatrix}, \text{ and}$$

$$\Phi_{c^2,i} := \begin{bmatrix} \phi_{(2n+b+m+1,2n+b+m+1),i}\, c^2 \end{bmatrix}.$$

Appendix C: Defining Linear Quadratic Loss Function in the Input File

If the user chooses to create by themselves the input file, the linear-quadratic loss of every player in the game should be defined as follows. According to the definition of vector $z(t)$, i.e., $z(t) := [p(t)\ \dot{p}(t)\ y(t)\ v(t)\ c]^T$, coefficient ϕ (in the loss of player i) that regards:

(i) variable $p_{1\le k\le n}$ has an index $(k, .), i$, i.e. , $\phi_{(k,.),i}$;
(ii) variable $\dot{p}_{1\le k\le n}$ has an index $(n + k, .), i$, i.e., $\phi_{(n+k,.),i}$;
(iii) variable $y_{1\le k\le b}$ has an index $(2n + k, .), i$, i.e., $\phi_{(2n+k,.),i}$;
(iv) variable $v_{1\le k\le m}$ has an index $(2n + b + k, .), i$, i.e., $\phi_{(2n+b+k,.),i}$; and
(v) variable c has an index $(2n + b + m + 1, .), i$, i.e., $\phi_{(2n+b+m+1,.),i}$.

For example, to define in the toolbox the following loss function of player i:

$$J_i = \frac{1}{2}\int_0^\infty \left\{5p_1^2 - \frac{1}{2}p_2 p_6 + \alpha_1 \dot{p}_3(p_3 + \dot{p}_2) + v_2\,(p_3 + 2v_i) + \sqrt{3}v_i^2 + 4\right\} e^{-\theta t} dt,$$

where α_1 is some parameter, we need to simplify it first into linear quadratic form:

$$J_i = \frac{1}{2}\int_0^\infty \left\{5p_1^2 - \frac{1}{2}p_2 p_6 + \alpha_1 p_3 \dot{p}_3 + \alpha_1 \dot{p}_2 \dot{p}_3 + p_3 v_2 + 2v_i v_2 + \sqrt{3}v_i^2 + 4\right\} e^{-\theta t} dt,$$

and set:

$$\phi_{(1,1),i} = 5,\ \phi_{(2,6),i} = -\frac{1}{2}, \phi_{(3,n+3),i} = \alpha_1,\ \phi_{(n+2,n+3),i} = \alpha_1, \phi_{(3,2n+b+2),i} = 1,$$
$$\phi_{(2n+b+i,2n+b+2),i} = 2, \phi_{(2n+b+i,2n+b+i),i} = \sqrt{3},\ \text{and}\ \phi_{(2n+b+m+1,2n+b+m+1),i} = 4.$$

Of course, instead of $\phi_{(2,6),i} = -\frac{1}{2}$ one might as well set $\phi_{(6,2),i} = -\frac{1}{2}$ or $\phi_{(2,6),i} = -\frac{1}{4}$ and $\phi_{(6,2),i} = -\frac{1}{4}$ or any other linear combination that sums up to $-\frac{1}{2}$.

References

Abou-Kandil, H., & Bertrand, P. (1986). Analytic solution for a class of linear quadratic open-loop Nash games. *International Journal of Control, 43*, 997–1002.

Azevedo-Perdicoúlis, T. P., & Jank, G. (2005). Iterative solution of algebraic matrix Riccati equations in open loop Nash games. *International Journal of Robust and Nonlinear Control, 15*, 55–62.

Başar, T., & Olsder, G. J. (1999). *Dynamic noncooperative game theory*. Philadelphia: SIAM.

Comtet, L. (1974). *Advanced combinatorics*. Dordrecht: Kluwer.

Dockner, E., Jørgensen, S., van Long, N., & Sorger, G. (2000). *Differential games in economics and management science*. Cambridge: Cambridge University Press.

Engwerda, J. C. (2005). *LQ dynamic optimization and differential games*. Chichester: Wiley.

Engwerda, J. C. (2007). Algorithms for computing Nash equilibria in LQ games. *Computational Management Science*, *4*, 113–140.

Engwerda, J. C. (2008). Uniqueness conditions for the affine open-loop linear quadratic differential game. *Automatica, 44*, 504–511. Preliminary version in: E. F. Camacho, et al. (Eds.), *Proceedings of the CDC-ECC'05 conference*, Sevilla, December 2005, Spain (pp. 3507–3512).

Fershtman, C., & Kamien, I. (1987). Dynamic duopolistic competition with sticky prices. *Econometrica, 55*, 1151–1164.

Feucht, M. (1994). *Linear-quadratische Differentialspiele und gekoppelte Riccatische Matrixdifferentialgleichungen*. Ph.D. Thesis, Universität Ulm, Germany.

Jørgensen, S., & Zaccour, G. (2003). *Differential games in marketing*. Deventer: Kluwer.

Kremer, D. (2002). *Non-symmetric Riccati theory and noncooperative games*. Ph.D. Thesis, RWTH-Aachen, Germany.

Laub, A. J. (1979). A Schur method for solving algebraic Riccati equations. *IEEE Transactions on Automatic Control, 24*, 913–921.

Laub, A. J. (1991). Invariant subspace methods for the numerical solution of Riccati equations. In S. Bittanti, A. Laub, & J. Willems (Eds.), *The Riccati equation* (pp. 163–199). Berlin: Springer-Verlag.

Mehrmann, V. L. (1991). The autonomous linear quadratic control problem: Theory and numerical solution. In M. Thoma & A. Wyner (Eds.), *Lecture notes in control and information sciences 163*. Berlin: Springer-Verlag.

Neck, R., Hager, M., & Behrens, D. (2001). Solving dynamic macroeconomic policy games using the algorithm OPTGAME 1.0. *Optimal Control Applications and Methods, 22*, 301–332.

Paige, C., & van Loan, C. (1981). A Schur decomposition for Hamiltonian matrices. *Linear Algebra and Its Applications, 41*, 11–32.

Plasmans, J., Engwerda, J., van Aarle, B., di Bartolomeo, G., & Michalak, T. (2006). *Dynamic modeling of monetary and fiscal cooperation among nations*. Berlin: Springer-Verlag.

Reid, W. T. (1972). *Riccati differential equations*. London: Academic Press.

Simaan, M., & Cruz, J. B., Jr. (1973). On the solution of the open-loop Nash Riccati equations in linear quadratic differential games. *International Journal of Control, 18*, 57–63.

Starr, A. W., & Ho, Y. C. (1969a). Nonzero-sum differential games. *Journal of Optimization Theory and Applications, 3*, 184–206.

Starr, A. W., & Ho, Y. C. (1969b). Further properties of nonzero-sum differential games. *Journal of Optimization Theory and Applications, 3*, 207–219.

Tabak, D. (1975). Numerical solution of differential game problems. *International Journal of Systems Science, 6*, 591–599.
van Dooren, P. (1981). A generalized eigenvalue approach for solving Riccati equations. *SIAM Journal of Scientific Statistical Computation, 2*, 121–135.