TOWARDS REAL-TIME, NON-LINEAR QUADRATIC OPTIMAL REGULATION

Robert F Harrison and Stephen P Banks

Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK.

Abstract: A novel non-linear design method based on linear quadratic optimal control theory is presented that applies to a wide class of non-linear systems. The method is easy to apply and results in a near optimal solution that has the potential to be implemented in real-time in the sense that the solution is causal, in contrast to the conventional, quadratic regulator solution, which is anti-causal (backwards in time) and must be computed off-line. The key feature of the design method is the introduction of state-dependence in the weight matrices of the usual linear quadratic cost function, leading to a non-linear control policy, even for linear dynamics. To demonstrate the method, a simple vehicle suspension model with a cubic damping force is used, in conjunction with non-linear state penalties that better reflect the engineering objectives of active vehicle vibration suppression. A number of simulations is conducted and compared with a passively mounted vehicle. *Copyright © 1998 IFAC*

Keywords: Non-linear control systems; quadratic optimal regulators; active vehicle suspension; Riccati equations; stability.

1. INTRODUCTION

Linear quadratic (LQ) optimal control theory is a highly developed approach for the synthesis of linear optimal control laws that has been widely applied. In particular, the infinite-time-horizon solution has appeal for the regulation of processes that are well modelled by linear time-invariant dynamics because the solution comprises a set of static gains that are calculated once, off-line, and are implemented causally thereafter. The finite-time-horizon solution, while being more general, and admitting of time-varying dynamics or weighting parameters, is essentially an off-line procedure because the associated (differential) Riccati equation must be solved backwards in time. Incorporation of such a scheme in a closed-loop, real-time process is not therefore possible. The receding horizon approach attempts to overcome this difficulty by repeatedly solving the open-loop, finite-time-horizon problem for short periods into the future, and using this as the feedback gain over the time step. It does this at the expense of optimality, which may or may not be important from the point of view of the practising engineer.

A further disadvantage of the LQ philosophy is that, being a linear feedback, the control signal is affected in the same way by small and large signals. In many applications it may be preferable to ignore small error signals (due, for instance, to measurement noise) as far as possible, while responding optimally to large errors. It may also be desirable to switch attention between control objectives depending on their current values. The receding horizon strategy approximates the former behaviour to a certain extent, but is unable to address the latter.

For non-linear dynamics the situation is exacerbated, with few explicit solutions for their optimal quadratic control as yet known except those based on series expansions. These are unrealisable, unless via truncation, leading to a loss of optimality and possibly stability. Normally optimal quadratic control for non-linear systems is conducted numerically and tends, inherently, to be non-causal.

In this paper we make use of a new result that generalises the LQ theory to non-linear systems to provide a non-linear design method that overcomes some of the difficulties mentioned above. This non-linear quadratic (NLQ) method applies to systems having a broad class of non-linear dynamics with state-dependent weighting matrices (the design degrees of freedom). In brief, it turns out that the infinite-time-horizon LQ regulator problem when solved afresh at every point on the state trajectory leads to a near-optimal control policy (Banks and Mhana, 1992). For admissible system dynamics, the weighting parameters can be made to be functions of the state variables. Thus, in addition to handling non-linear dynamics, the design stage allows for the introduction of state-dependence in the weighting matrices, leading to a more flexible control strategy.

Our method is causal, but has considerable computational overhead. However, by using a solution to the Riccati equation based upon the matrix sign function (Gardiner and Laub, 1986), it is possible to derive a parallel algorithm (Gardiner and Laub, 1991) that may be suitable for real-time implementation.

In order to demonstrate the approach we consider the active control of a vehicle suspension system in order to reject disturbances induced by surface asperity: a problem that has received much attention in the past although usually for linearised dynamics. Clearly, the assumption of linearity may often be valid but some designs are inherently non-linear such as the oleopneumatic shock struts of aircraft landing gear.

While the LQ approach is attractive in that it is possible to penalise different variables so as to trade-off between, say, ride comfort and handling, or comfort and suspension travel, the way these variables are treated is essentially fixed – no provision is made to allow the suspension to distinguish between a smooth surface and a rough one. Evidently, while comfort might be a prime objective under normal circumstances, in extreme conditions the suspension should be stiffened to avoid hitting its limits, hence incurring damage. This is true even if the dynamics are linear up to this point. Although, in principle, time-varying weighting parameters are allowed in the (finite-time) LQ approach, lack of prior knowledge of the surface profile, and the acausal calculation for the solution makes the introduction of these difficult. The required amplitude dependence can never, therefore, be achieved through the LQ approach.

We illustrate our approach on a simple non-linear model incorporating a non-linear damping element, and compare our results with the passive system and the optimal NLQ solution using fixed matrices.

The remainder of the paper is organised as follows. In §2, to motivate the work, the LQ regulator problem is first set out, the generalised results are stated and the robust solution of the infinite-time-horizon LQ problem is outlined. In §3 the suspension model is presented and the choice of design parameters is discussed. The results of a series of experiments are described in §4, followed by conclusions in §5.

2. THE DESIGN METHOD

2.1 Linear quadratic regulator

The LQ optimal regulation problem is expressed as follows: minimise the cost function

$$J = \int_0^\infty \left(\mathbf{x}^t Q\mathbf{x} + \mathbf{u}^t R\mathbf{u}\right)dt \qquad (1)$$

subject to the linear time invariant dynamics:

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u} \qquad (2)$$

where $\mathbf{x}$ is an n-vector of system states, $\mathbf{u}$ is an m-vector of control variables, A and B are matrices of appropriate dimension and the superscript, t, indicates transposition. The matrices Q and R are positive semi-definite and definite, respectively, and are used to penalise particular states and controls according to the engineering objective.

It is well known, e.g. (Friedland, 1987), that the control policy which solves the above problem is a linear combination of the system states, given by:

$$\mathbf{u} = K\mathbf{x} \qquad (3)$$

$$K = -R^{-1}B^t P \qquad (4)$$

$$0 = PA + A^t P - PBR^{-1}B^t P + Q \qquad (5)$$

where K is given by (4) and P is the positive definite solution of the algebraic matrix Riccati equation (5). A unique, positive definite solution to the above exists if the pair (A,B) is stabilizable and (A,Γ) is detectable, with $Q = \Gamma^t\Gamma$.

2.2 Non-linear quadratic regulator

The extension of the above to non-linear systems looks identical, except that, instead of performing a single optimisation and applying the resulting gain-matrix for all time, the optimisation has to be carried out at every time step. Consider a non-linear dynamical system that can be expressed in the form:

$$\dot{\mathbf{x}} = A(\mathbf{x})\mathbf{x} + B(\mathbf{x})\mathbf{u} \qquad (6)$$

where the Jacobians of $A(\mathbf{x})$ and $B(\mathbf{x})$ are subject to some bounded growth conditions (Lipschitz), then at each point, $\overline{\mathbf{x}}$, on the state trajectory, a linear system is defined with fixed $A = A(\overline{\mathbf{x}})$ and $B = B(\overline{\mathbf{x}})$. In (Banks and Mhana, 1992) it is shown that solving the infinite-time LQ optimal control problem, point-wise on the state trajectory, results in the near-optimal, stabilising quadratic control policy for systems described by equation (6). Thus, by choosing the $\mathbf{u}$ that minimises the usual quadratic cost function at every time step, we have a near-optimal control policy for a very wide class of non-linear systems. Evidently, $A(\overline{\mathbf{x}}), B(\overline{\mathbf{x}})$ and Q are subject, point-wise, to the same conditions as for the linear case. It is clear that the proposed solution is identical to the one obtained from equations (3, 4 and 5) when the dynamics are linear.

Because the control synthesis takes place point-wise, the designer is now free to select Q and R in ways which are more directly applicable to the control *engineering* objectives. These can be made to be functions of the instantaneous state variables, i.e.

$$J = \int_0^\infty \left(\mathbf{x}'Q(\overline{\mathbf{x}})\mathbf{x} + \mathbf{u}'R(\overline{\mathbf{x}})\mathbf{u}\right)dt \qquad (7)$$

subject to the requirements for the solution of the Riccati equation and the invertibility of R. Ensuring that $A(\overline{\mathbf{x}}), B(\overline{\mathbf{x}}), R(\overline{\mathbf{x}})$ and $Q(\overline{\mathbf{x}})$ satisfy these requirements *a priori*, is difficult in general, however, for polynomial functions which are not identically zero, the required properties will be lost only on sets of zero measure and will not, therefore, persist.

2.3 Solving the matrix Riccati equation

In order to calculate the optimal solution, it is necessary to solve the matrix Riccati equation at each point in time. In practice this will be done in a computer and it will be necessary to solve the equation at each discrete time-step. The usual approach to the solution of this problem is via an eigen-decomposition of the Hamiltonian matrix for the system (Laub, 1979). For sizeable dynamics such an approach is computationally intensive and may not be able to deliver solutions at the required sample rate (i.e. in "real-time"). It can also be sensitive, depending as it does on the numerical solution of an eigen-problem. It should be noted that, even though the problem is to be solved in *discrete-time*, we do not solve the discrete-time Riccati equation: the dynamics are essentially continuous-time.

The matrix-sign-function (MSF) is an appealing alternative to eigen-decomposition owing to its simplicity (Roberts, 1980), requiring only the operations of matrix inversion and addition, and multiplication of a matrix by a scalar (Gardiner and Laub, 1986). This simplicity also suits the MSF algorithm to parallel computation (Gardiner and Laub, 1896, 1991) as follows:

$$Z_0 = H$$

$$Z_{i+1} = \frac{1}{2}\left(\frac{1}{c_i}Z_i + c_i Z_i^{-1}\right) \qquad (8)$$

where $c_i = \left|\det(Z_i)\right|^{1/2n}$ is a scaling used to speed convergence, and n is the dynamical order. H is the Hamiltonian matrix of order $2n$, given by:

$$H = \begin{bmatrix} A' & Q \\ BR^{-1}B' & -A \end{bmatrix}$$

Assuming H has no eigen-values on the imaginary axis Z_i converges to $\mathrm{Sign}(H) = S$, say, (Gardiner and Laub, 1986). For the solution of the Riccati equation we require the quantity (Roberts, 1980)

$$S^+ = \mathrm{Sign}^+(S) = \frac{1}{2}(I_{2n} + S)$$

Now, by decomposing S^+ thus: $S^+ = \begin{bmatrix} V & W \end{bmatrix}$ we write, P, the solution of the Riccati equation (5) thus:

$$P = V'W(W'W)^{-1} \qquad (9)$$

The convergence of the algorithm has been investigated in (Roberts, 1980) and its global convergence is established in (Balzer, 1980). In (Gardiner and Laub, 1991) an algorithm for diagonal pivoting factorisation – a form of Gaussian elimination with partial pivoting – is used to develop a parallel algorithm involving no sequential computations. Complexity is of order $(2n)^3$, which dominates the communication burden. The speed-up achieved over the conventional method (Laub, 1979) is demonstrated on a hypercube architecture. More recently (Bunse-Gerstner Faßbender, 1997), a Jacobi-like method has been proposed with a simple function of the matrix size to predict the number of iterations needed for convergence. This of course does not guarantee that a solution can be found within any arbitrary time period.

We do not implement the parallel solution here. The discussion is included simply to underline that real-time operation may be possible.

3. SUSPENSION MODEL

The two-degree-of-freedom, quarter-car model of figure 1 has been widely studied in the literature. It represents an active element operating in parallel with passive elements – a linear spring, k_1, and a non-linear damper with characteristic $f_d = c_1\xi + c_2\xi^3$, where f_d represents the damping force and ξ, the relative velocity between the sprung and unsprung masses. The model parameters are based on the one published in (Lin and Kanellakopoulos, 1997).

The motions of the body and wheel (sprung, m_1, and unsprung, m_2, masses, respectively) are denoted by y_1 and y_2 respectively, while the deviation of the surface from some datum is denoted by d. The tyre is represented by a linear spring, k_2, with no damping, for simplicity. We assume that the control force, f, can be applied directly as a result of the control signal, with negligible actuator dynamics. Again this is chosen for simplicity, so as not to obscure the main point of the paper.

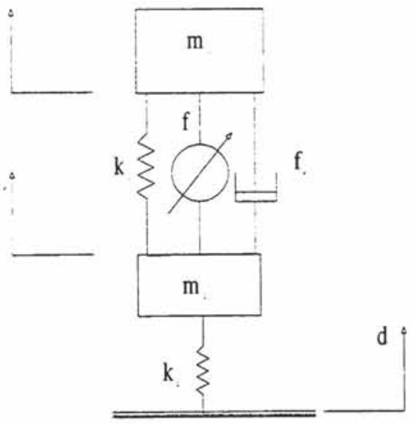

Fig. 1. Schematic of the two-degree-of-freedom, quarter-car model.

The equations of motion for the quarter car are given by:

$$\begin{aligned}\ddot{y}_1 &= -\frac{k_1}{m_1}(y_1 - y_2) - \frac{c_1}{m_1}(\dot{y}_1 - \dot{y}_2) \\ &\quad -\frac{c_2}{m_1}(\dot{y}_1 - \dot{y}_2)^3 + \frac{1}{m_1}f \\ \ddot{y}_2 &= \frac{k_1}{m_2}(y_1 - y_2) + \frac{c_1}{m_2}(\dot{y}_1 - \dot{y}_2) \\ &\quad +\frac{c_2}{m_1}(\dot{y}_1 - \dot{y}_2)^3 - \frac{k_2}{m_2}(y_2 - d) - \frac{1}{m_2}f\end{aligned} \tag{10}$$

We choose state variables thus: $x_1 = y_1, x_2 = \dot{y}_1, x_3 = y_2, x_4 = \dot{y}_2$, and identify the control signal, u, with the force, f. Evidently equations (10) can be put into the required form (6), and the Jacobians of both A and B are clearly Lipschitz.

3.1 Design objectives

For the purposes of this paper let us suppose that our primary objective is to minimise passenger discomfort. We do this by attempting to reduce the accelerations to which the passenger is subject – vertical only, in this simple case. Thus a candidate for the cost function is $\ddot{y}_1 = C_a(\overline{\mathbf{x}})\mathbf{x} + D_a(\overline{\mathbf{x}})\mathbf{u}$, where $C_a(\overline{\mathbf{x}})$ is the second row of $A(\overline{\mathbf{x}})$ and $D_a(\overline{\mathbf{x}})$ is the second element of $B(\overline{\mathbf{x}})$. However, ride comfort can only take precedence when safety and integrity are not compromised. Thus it is necessary to penalise some measure which embodies these ideas, usually the "rattlespace deflection", $y_1 - y_2 = C_r\mathbf{x} = [1 \quad 0 \quad -1 \quad 0]\mathbf{x}$. In the conventional LQ approach we construct a cost function thus:

$$\begin{aligned}J &= \int_0^\infty \left(q_a\ddot{y}_1^2 + q_r(y_1 - y_2)^2 + ru^2\right)dt \\ &= \int_0^\infty \begin{pmatrix}\mathbf{x}'\left(q_a C_a^t C_a + q_r C_r^t C_r\right)\mathbf{x} \\ +2\mathbf{x}' q_a C_a^t D_a u + \left(q_a D_a^2 + r\right)u^2\end{pmatrix} dt\end{aligned} \tag{11}$$

Letting $N = q_a C_a^t D_a$ and $R = q_a D_a^2 + r$ we accommodate the cross-term in the usual way $Q \leftarrow Q - NR^{-1}N'$, $A \leftarrow A - BR^{-1}N'$ with the original $Q = q_a C_a C_a^t + q_r C_r C_r^t$ (Friedland, 1987). q_a, q_r are used to control the trade-off between ride and handling. The new Q and A are now used in the standard equations (6) and (7).

In the non-linear design procedure we make q_r state-dependent, thus: $q_r(\overline{\mathbf{x}}) = 500\psi(y_1 - y_2, .02, .001)$ with

$$\psi(\xi,\theta,\delta) = \begin{cases}\left((\xi-\theta)/\delta\right)^4, & \xi > \theta \\ 0, & |\xi| \le \theta \\ \left((\xi+\theta)/\delta\right)^4, & \xi < -\theta\end{cases} \tag{12}$$

where $\theta \ge 0$ defines a dead-zone and $\delta > 0$, the distance within which ψ first reaches unity. The rationale for this functional form is as follows. The primary objective is to reduce body acceleration hence the constant $q_a = 10000$. The secondary objective, which can over-ride the first for safety reasons, is to reduce overly large excursions in the suspension strut. Thus, for a rattlespace of ±0.055m, a dead-zone of ±0.02m is allowed before control action is taken; the non-linearity increasing to unity within the next 0.001m of travel and dominating the cost function very rapidly as the limits are approached. We have been guided here by the function chosen in (Lin and Kanellakopoulos, 1997), however, any other suitable function is a candidate. In addition, acceleration or other variables could equally well be weighted in this way. For comparison, we also consider a conventionally weighted rattlespace deflection with $q_r = 1000$. r = 0.0001 throughout.

We use the passive system (equation (10) with $f(t) = 0$ for all t) as a reference and compare its behaviour with that of the actively controlled models for the profile (Lin and Kanellakopoulos, 1997)

$$d(t) = \begin{cases}a(1 - \cos 8\pi t), & 0 \le t \le 0.25 \\ 0, & \text{otherwise}\end{cases} \tag{13}$$

where a is one-half the height of the hump.

4. RESULTS

Each of figures 2 onwards shows the passive, conventionally weighted (CW) and state-dependently weighted (SDW) behaviour of the vehicle model. These are indicated by a dashed, a dotted and a solid curve, respectively. Figures 2–5 relate to a modest disturbance of maximum height 2 cm which does not threaten an approach towards the rattlespace limits. Here, because $|\dot{y}_1 - \dot{y}_2| < 0.02$ for all time, the rattlespace deflection is never penalised, and because acceleration dominates strongly in the cost function for both CW and SDW, it is not possible to distinguish between these behaviours. Figures 6–9 show the behaviour for a much more severe disturbance, with a maximum height of 11cm that forces the rattlespace weighting into play.

Figures 2, 3 and 4 indicate the improvements attained by both CW and SDW, while figure 5 hints that these come at the expense of marginally increased unsprung mass displacement.

Figure 6 shows how, for a severe disturbance the transient acceleration is degraded while tending towards an improved steady-state. While the acceleration is penalised conventionally, there is an effect when the rattlespace SDW operates (~0.06s).

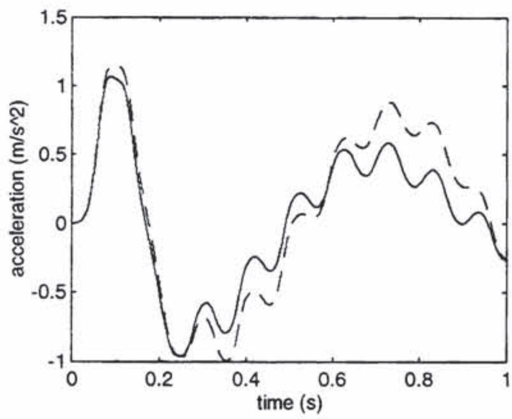

Fig. 2. Acceleration vs. time, a=0.01.

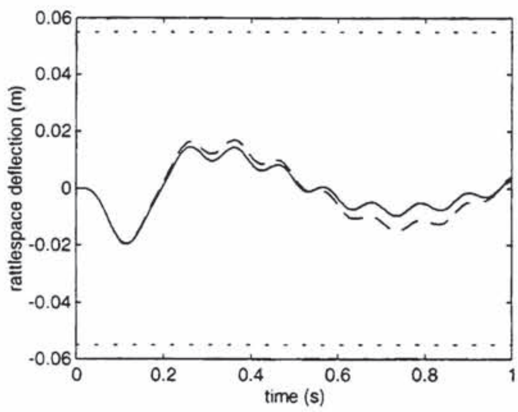

Fig. 3. Rattlespace deflection vs. time, a=0.01.

This sacrifices comfort for safety leading to a response which is worse than the passive system for the first ~0.5s. There is little difference between the passive and CW systems. Figure 7 highlights a more important difference in rattlespace behaviour. Again CW and SDW deliver improvements over the passive behaviour but now the value of SDW is evident.

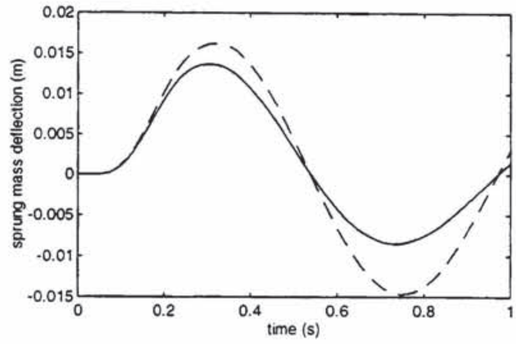

Fig. 4. Sprung mass deflection vs. time, a=0.01.

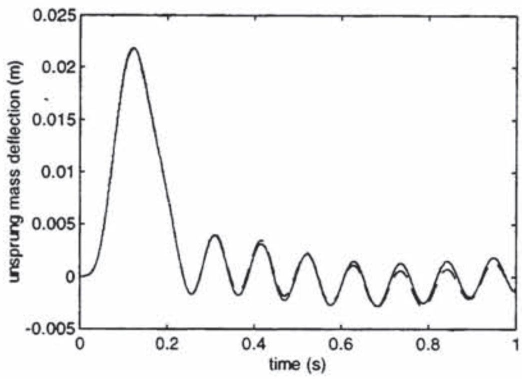

Fig. 5. Unsprung mass deflection vs. time, a=0.01.

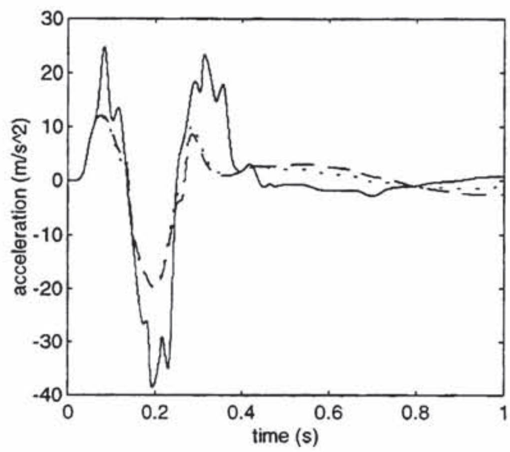

Fig. 6. Acceleration vs. time, a=0.055.

The CW system exceeds the limits of travel in the suspension strut – "bottoming" occurs – while SDW rapidly counteracts the approach (albeit for a deterioration in ride). Nonetheless, only by de-tuning the CW system can bottoming be avoided, and this could not be guaranteed.

Figures 8 and 9 indicate that improvements in ride and handling may come at the expense of increased excursions of the sprung and unsprung masses themselves. However, in a fuller design it would be straightforward to penalise these too.

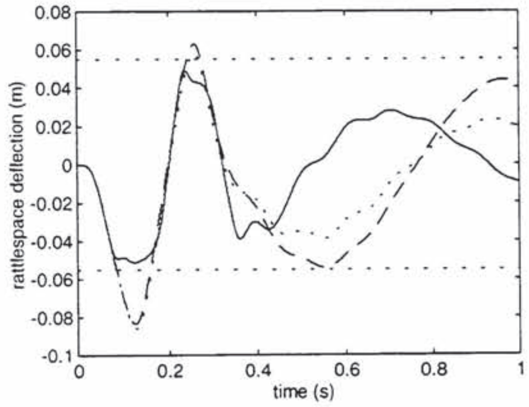

Fig. 7. Rattlespace deflection vs. time, a=0.055.

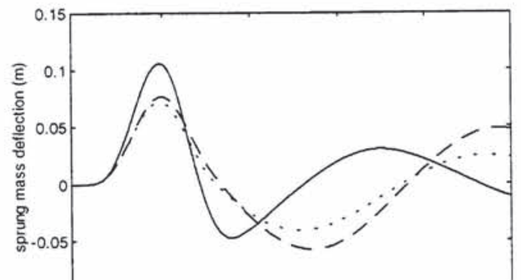

Fig. 8. Sprung mass deflection vs. time, a=0.055.

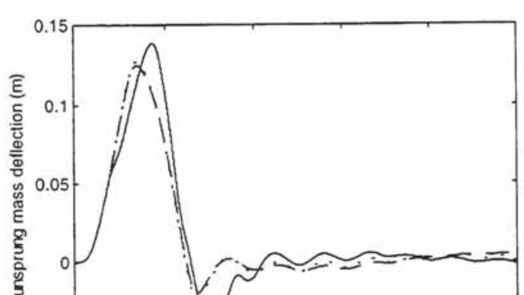

Fig. 9. Unsprung mass deflection vs. time, a=0.055.

Finally, we observe that, using the Matlab computing environment, with non-optimised code, the MSF method uses on the order of 5,000 flops per iteration and requires a maximum of nine iterations to converge at each time-step, with a minimum of six and a mean of 8.55. The total number of flops per time step is comparable with those required for the non-iterative eigen-decomposition-based solution (26,000 approximately).

5. CONCLUSIONS

A new method for the design and synthesis of near-optimal, non-linear control laws is proposed, based on a generalisation of LQ optimal control theory. The method is simple to apply and affords greater design flexibility (SDW) than the conventional approach (CW). The resulting controller is non-linear, even for linear dynamics, and can be implemented in real-time. To illustrate the method a simple two-degree-of-freedom quarter-car model with cubic damping has been studied using a non-linear penalty function. Preliminary results show that the method has applicability and could easily be tuned to provide desirable closed-loop behaviour.

The main issue to be addressed in the future is the real-time implementation of such a system. Clearly, without guarantees on the maximum number of iterations needed *per time step* for parallel solutions, the method is severely restricted. However, at the expense of a small loss of "optimality" the essential problem of real-time, non-linear optimal control, *viz.* the anti-causal nature of the solution, has been overcome and a way has been opened towards real-time optimal quadratic regulation.

REFERENCES

Balzer, L.A. (1980). Accelerated convergence of the matrix sign function method of solving Lyapunov, Riccati and other matrix equations. *Int. Jnl. of Control*, **32**, 1057–1078.

Banks, S.P. and K.J. Mhana (1992). Optimal control and stabilisation for non-linear systems. *IMA Jnl. of Math. Control and Information*, **9**, 179–196.

Bunse-Gerstner, A. and H. Faßbender (1997). A Jacobi-like method for solving algebraic Riccati equations on parallel computers. *IEEE Trans. on Automatic Control*, **42**, 1071–1084.

Friedland, B. (1987). Control system design: an introduction to state-space methods, McGraw-Hill Book Co., New York.

Gardiner, J. D. and A.J. Laub (1986). A generalisation of the matrix-sign-function for algebraic Riccati equations. *Int. Jnl. of Control*, **44**, 823–832.

Gardiner, J.D. and A.J. Laub (1991). Parallel algorithms for algebraic Riccati equations, *Int. Jnl. of Control*, **54**, 1317–1333.

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

Lin, J-S. and I. Kanellakopoulos (1997). Non-linear design of active suspensions. *IEEE Control Systems*, **17**, 45–59.

Roberts, J.D. (1980). Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Int. Jnl of Control*, **32**, 677–687.