Technische Universität Chemnitz-Zwickau

DFG-Forschergruppe "SPC" · Fakultät für Mathematik

Peter Benner Ralph Byers

Newton's Method with Exact Line Search for Solving the Algebraic Riccati Equation

Preprint-Reihe der Chemnitzer DFG-Forschergruppe
"Scientific Parallel Computing"

SPC 95_24 Oktober 1995

Newton's Method with Exact Line Search for Solving the Algebraic Riccati Equation

Peter Benner *   Ralph Byers †

Abstract

This paper studies Newton's method for solving the algebraic Riccati equation combined with an exact line search. Based on these considerations we present a Newton–like method for solving algebraic Riccati equations. This method can improve the sometimes erratic convergence behavior of Newton's method.

1 Introduction

We study the generalized continuous–time algebraic Riccati equation (CARE)

$$\begin{aligned} 0 = \mathcal{R}(X) = C^TQC \;&+\; A^TXE \;+\; E^TXA \\ &-\; (B^TXE + S^TC)^T R^{-1} (B^TXE + S^TC) \end{aligned} \qquad (1)$$

Here $A, E, X \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $R = R^T \in \mathbb{R}^{m\times m}$, $Q = Q^T \in \mathbb{R}^{p\times p}$, $C \in \mathbb{R}^{p\times n}$ and $S \in \mathbb{R}^{p\times m}$. This equation arises frequently in control problems. We will assume that E and R are nonsingular and $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0$, where $M \geq 0$ denotes positive semidefinite matrices M.

Often, the desired solution X is *stabilizing* in the sense that the eigenvalues of the pencil $(E, A - BR^{-1}(B^TXE + S^TC))$ have negative real parts. (By assumption E is nonsingular, so all eigenvalues of the pencil are finite.) In the sequel, this will be denoted by $\lambda\left(E, A - BR^{-1}(B^TXE + S^TC)\right) \subset \mathbb{C}^-$. Assuming (E, A, B) strongly stabilizable and (E, A, C) strongly detectable, such a positive semidefinite, stabilizing solution exists and is unique [24]. Throughout this paper, we will call this stabilizing solution X^*.

*Technische Universität Chemnitz–Zwickau, Fakultät für Mathematik, 09107 Chemnitz, FRG. E-mail: benner@mathematik.tu-chemnitz.de. Some of this work was complete while this author was visiting the University of Kansas.

†405 Snow Hall, Department of Mathematics, University of Kansas, Lawrence, KS 66045, USA. E-mail: byers@math.ukans.edu. This author was partially supported by National Science Foundation grants INT-8922444, and CCR-9404425, and the Kansas Institute for Theoretical and Computational Science.

The algebraic Riccati equation (1) is a nonlinear system of equations. One of the oldest, best studied, numerical methods for solving (1) is Newton's method [15, 19, 24, 27].

Algorithm 1 (Newton's method for solving CARE)

1. Choose some initial starting guess $X_0 = X_0^T$.

2. FOR $j = 0, 1, 2, \ldots$

   2.1 $K_j \leftarrow R^{-1}(B^T X_j E + S^T C)$.

   2.2 Solve for N_j in the Lyapunov equation
   $(A - BK_j)^T N_j E + E^T N_j (A - BK_j) = -R(X_j)$.

   2.3 $X_{j+1} \leftarrow X_j + N_j$.

   END FOR

Algorithm 1 modifies none of the coefficient matrices and corrects iterate X_j by adding a (hopefully) small step N_j. We prefer Algorithm 1 to some other mathematically equivalent versions of Newton's method, because of its robustness in the presence of rounding errors. The expensive part of Algorithm 1 is Step 2.2. The cost mainly depends upon the chosen method for solving the Lyapunov equation. This may be done using the *Bartels–Stewart algorithm* [3] or an extension to the case $E \neq I$ [11, 12].

It is well known that if (E, A, B) is strongly stabilizable, (E, A, C) is strongly detectable, and X_0 is stabilizing, then the iteration converges to the desired stabilizing solution X^* [15, 19, 24, 27]. Ultimately, convergence is quadratic. At each step $\lambda(E, A - BK_j) \subset \mathbb{C}^-$. *After* the first step, convergence is monotone. Besides these convergence properties, Algorithm 1 provides all the ingredients for a condition estimate of CARE and N_j may be considered as estimate for the error $X^* - X_j$. (See [6] for details.)

Although it ultimately converges rapidly, initially, the iteration may converge slowly. Automatic stabilizing procedures like proposed in [1, 28, 29] may give choices of X_0 that lie far from the solution X^*. Sometimes the first Newton step N_0 is disastrously large and many iterations are needed to find the region of rapid convergence. If the Lyapunov equation is ill-conditioned it may be difficult to compute an accurate Newton step. (This sometimes signals that the algebraic Riccati equation is ill-conditioned [6].) If the Newton step can not be calculated accurately, then the usual convergence theory breaks down. Sometimes rounding errors or a poor choice of X_0 cause Newton's method to converge to a non-stabilizing solution. For these reasons, Newton's method is not often used by itself to solve algebraic Riccati equations. However, when it is used as a defect correction method or for iterative refinement of an approximate solution obtained by a more robust method, it is often able to squeeze out the maximum possible accuracy. (See, for example, [2, 7, 17, 18]).

To illustrate the difficulties of Newton's method, consider the following example. (See also Example 6 in Section 4.)

Example 1 This example is contrived to demonstrate both the above difficulties. The coefficient matrices are

$$A = S = 0, \quad E = C = B = R = I_2, \quad Q = \mathrm{diag}(1, \sqrt{\delta}),$$

where $0 < \delta < 1$. The stabilizing solution is $X^* = \mathrm{diag}(1, \delta^{1/4})$. Choosing $X_0 = \mathrm{diag}(1, \delta)$ we obtain $\|X^* - X_0\|_F \approx \sqrt{\delta}\|X^*\|_F$, but $\|N_0\|_F \approx 0.5\delta^{-\frac{1}{2}}$. Figure 1 shows the behavior for $\delta = 10^{-8}$.

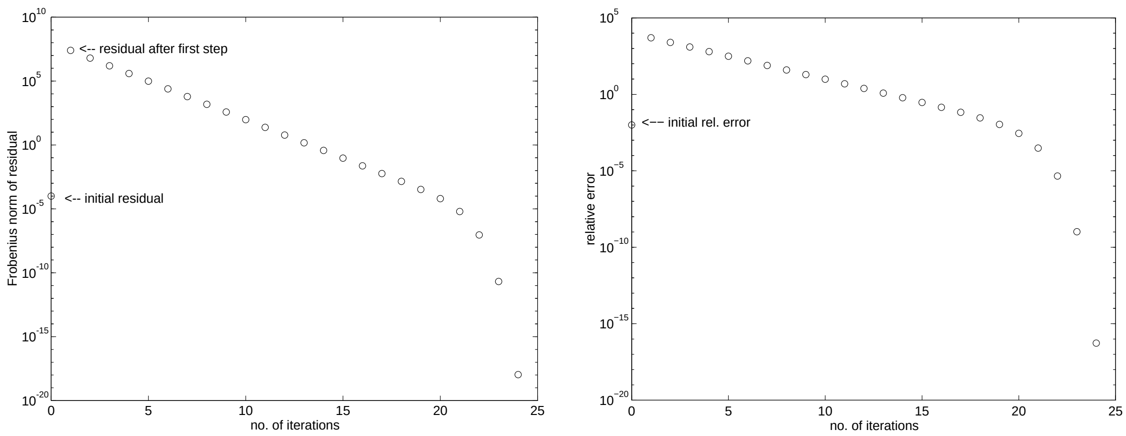

Figure 1: Example 1, residuals and relative errors for $\delta = 10^{-8}$

From the point of view of optimization theory, the Newton step gives a search direction along which $\|R(X_j)\|_F$ may be (at least approximately) minimized. Looking at the abovementioned problems of Newton's method, one can think of a disastrous first step as a too long step in the search direction, whereas the initial slow but monotonic convergence suggests that one could take longer steps in that direction. Our goal is to restore some robustness to Newton's method and to accelerate convergence through step size control by exact line search.

The idea discussed throughout this paper is to choose $t_j > 0$ to minimize the Frobenius norm of the next residual $R(X_{j+1}) = R(X_j + t_j N_j)$, i.e., to use an exact line search along the Newton direction. Exact line searches along conjugate gradient directions were used in [13] to solve (1). Line searches were also used in the Fletcher-Powell/Davidon's method proposed in [22]. Section 2 shows that the extra cost of doing an exact line search along the Newton direction is little more than the cost of the unmodified Newton Algorithm 1. In Section 3 we prove that exact line search along the Newton direction converges quadratically to the stabilizing solution, if the starting guess X_0 is stabilizing. Numerical examples in Section 4 demonstrate that step size control usually saves some iterations compared to Newton's method. Some final remarks and conclusions are given in Section 5.

2 Step Size Control by Exact Line Search

Line searches are a well understood technique commonly used in numerical methods for optimization [9]. The approach is to replace Step 2.3 in Algorithm 1 by

2.3' $X_{j+1} = X_j + t_j N_j$

where t_j is a real scalar controlling the "length" of the step tN_j. In our approach, t_j is chosen to minimize some measure of the error. The line search is said to be *exact* if t_j is an exact (as opposed to approximate) minimizer.

At first we introduce some formulas which will often be used in the sequel.

$$\begin{aligned} R(X_j + tN_j) = R(X_j) \quad &+ \quad t\left((A - BK_j)^T N_j E + E^T N_j (A - BK_j)\right) \\ &- \quad t^2 E^T N_j B R^{-1} B^T N_j E. \end{aligned} \tag{2}$$

Defining

$$V_j = E^T N_j B R^{-1} B^T N_j E,$$

and using Step 2.2 of Algorithm 1, i.e.,

$$(A - BK_j)^T N_j E + E^T N_j (A - BK_j) = -R(X_j), \tag{3}$$

we can rewrite (2) as

$$R(X_j + tN_j) = (1 - t) R(X_j) - t^2 V_j. \tag{4}$$

The *Frobenius* norm of a matrix M is defined by $\|M\|_F^2 = \mathrm{trace}(M^T M)$. For any symmetric matrix M, we have $\|M\|_F^2 = \mathrm{trace}(M^2)$, and for any two matrices M and N, $\mathrm{trace}(MN) = \mathrm{trace}(NM)$.

Using these properties and (4), it is easy to derive that finding t_j to minimize $\|R(X_{j+1})\|_F$ is equivalent to minimizing

$$\begin{aligned} f_j(t) &= \mathrm{trace}\left(R(X_j + tN_j)^2\right) \\ &= \alpha_j (1-t)^2 - 2\beta_j (1-t) t^2 + \gamma_j t^4. \end{aligned} \tag{5}$$

where

$$\begin{aligned} \alpha_j &= \mathrm{trace}(R(X_j)^2), \\ \beta_j &= \mathrm{trace}\left(R(X_j) V_j\right), \\ \gamma_j &= \mathrm{trace}(V_j^2). \end{aligned}$$

This is a polynomial of degree at most four. If $\gamma_j = \mathrm{trace}(V_j^2) > 0$, it has either one local minimum and no local maxima or two local minima and one local maximum. If $\gamma_j = 0$, then $V_j = 0$ and $f_j(1) = 0$. Choosing $t = t_j = 1$ minimizes f_j.

Differentiating f_j and using (4), we have

$$\begin{aligned} f_j'(t) &= -2 \cdot \mathrm{trace}\left((R(X_j) + 2tV_j) R(X_j + tN_j)\right) \\ &= -2 \cdot \mathrm{trace}\left((R(X_j) + 2tV_j)((1-t)R(X_j) - t^2 V_j)\right). \end{aligned} \tag{6}$$

Remark 1 There exists a local minimum of f_j at some value of $t_j \in [0, 2]$, because $f_j'(0) = -2 \cdot \text{trace}(R(X_j)^2) \leq 0$, and $f_j'(2) = 2 \cdot \text{trace}((R(X_j) + 4V_j)^2) \geq 0$.
If $R(X_j) \neq 0$, i.e., if X_j is not a solution of (1), then $f_j'(0) < 0$ and the Newton step is a descent direction of $\|R(X_j + tN_j)\|_F$ from $t = 0$. It follows that for the minimizing $t_j \in [0, 2]$, $\|R(X_j + t_jN_j)\|_F \leq \|R(X_j)\|_F$ and $\|R(X_j + t_jN_j)\|_F = \|R(X_j)\|_F$ if and only if $R(X_j) = 0$. That is, choosing $t_j \in [0, 2]$ to be the local minimizer of f_j implies that the residual decreases as long as X_j is not a solution of the CARE (1).

Remark 1 suggests that we modify Algorithm 1 as follows.

Algorithm 2 (Exact Line Search)

1. Choose some initial starting guess $X_0 = X_0^T$.

2. FOR $j = 0, 1, 2, \ldots$

   2.1 $K_j \leftarrow R^{-1}(B^TX_jE + S^TC)$.

   2.2 Solve for N_j in the Lyapunov equation
   $(A - BK_j)^TN_jE + E^TN_j(A - BK_j) = -R(X_j)$.

   2.3 $V_j \leftarrow E^TN_jBR^{-1}B^TN_jE$.

   2.4 Find a local minimizer $t_j \in [0, 2]$ of $f_j(t)$ using (5).

   2.5 $X_{j+1} \leftarrow X_j + t_jN_j$

   END FOR

Remark 2 In case there are two local minima of f_j in $[0, 2]$, we can choose the one giving the smaller residual. This is easily achieved using (5).

Remark 3 Using exact arithmetic, Algorithm 2 finds the solution of scalar Riccati equations in the first step.

Remark 4 There is a cheaper way to compute the residuals in each step. We can compute $R_0 = R(X_0)$ and then using (4) we may set $R_{j+1} = (1 - t_j)R_j - t_j^2V_j$ for $j \geq 1$. This recursion suffers from subtractive cancellation as the R_j's tend to zero. The accuracy of the Newton step depends critically upon the accuracy with which the residual is calculated [24]. Hence, we prefer to use the formulation of Step 2.2.

Compared to Algorithm 1, the exact line search method requires some additional computations. We need to compute the symmetric matrix

$$V_j = E^TN_jBR^{-1}B^TN_jE$$

in each iteration step. One way to compute V_j efficiently is as follows. Before starting the iteration, we compute a Cholesky factorization of R, $R = L^TL$, and then store the product $\hat{B} = BL^{-1}$ in an n-by-m array. Since this is part of the residual computation, it is already part of Newton's method and hence does not contribute any extra computations. With these settings, we can compute V_j by

$$V_j = (E^T N_j \hat{B})(E^T N_j \hat{B})^T$$

which requires $5n^2m + nm$ flops. (Following [14], we define each floating point arithmetic operation together with the associated integer indexing as a *flop.*) In case $E = I$, this reduces to $3n^2m + nm$ flops. In many applications, $m \ll n$, and hence the computation of this matrix is cheap compared to the Newton step.

Storing V_j requires an extra work space of size n^2. This amount of work space is usually needed for the solution of the Lyapunov equation in Step 2.2 and can thus be used to store V_j. In case extra work space is not available, we could use the fact that it is not necessary to store V_j explicitly since it is only needed to compute $\beta_j = \text{trace}(R(X_j)V_j)$ and $\gamma_j = \text{trace}(V_j^2)$. Thus we can compute V_j columnwise and update β_j, γ_j with each computed column. Therefore, we only need work space of size nm to store $E^T N_j \hat{B}$ and a vector of length n to store the currently computed column of V_j. This strategy adds another n^2m flops to the computational cost since it does not take advantage of the symmetry of V_j.

Computing the coefficients α_j, β_j, γ_j of f_j and finding the minimizing t_j contributes $3n$ inner products and some scalar operations which is negligible compared to the $O(n^3)$ matrix multiplications and Lyapunov equation solutions. Using work estimates from [11, 14] for solving the Lyapunov equation, we can conclude that for $m = n$, each step of Algorithm 2 does less than 10% more work if $E = I$ and less than 5% more work if $E \neq I$ than a single step of Algorithm 1. This comparison becomes even more favorable the smaller m is in relation to n.

Note also that the cost of V_j is approximately the same as the cost of calculating the residual $R(X_j)$, so exact line search is competitive with other strategies that require one or more extra residual calculations.

Remark 5 Newton's method as discussed in [19, 24, 27] differs from Algorithm 1 by updating $A_0 \leftarrow A$, $A_j \leftarrow A_{j-1} - BK_j$ for $j = 1, 2, \ldots$, and computing X_{j+1} directly from the Lyapunov equation. This version is slightly faster than Algorithm 1, but is less robust in the presence of rounding errors.

Were we to formulate Algorithm 2 analogously, the exact line search would add no significant extra cost. (See [4].)

3 Convergence

Algorithm 2 recasts the nonlinear equation (1) as a non-linear least squares problem. The convergence theory for this approach is well known and largely satisfactory. (For example, see [9, § 6.5].) However, convergence — even convergence to a solution — is not sufficient. Often the symmetric, positive semidefinite, stabilizing solution is required. In this section, we show that under certain assumptions, Algorithm 2 has guaranteed convergence from a stabilizing starting guess to the stabilizing solution.

In order to simplify notation, we will use the following definitions. By assumption E is nonsingular, so we may rewrite (1) as

$$R(X) = \tilde{R}(\tilde{X}) = \tilde{F} + \tilde{A}^T\tilde{X} + \tilde{X}\tilde{A} - \tilde{X}\tilde{G}\tilde{X} \tag{7}$$

where

$$\begin{aligned} \tilde{A} &= E^{-1}(A - BR^{-1}S^TC), \\ \tilde{X} &= E^TXE, \\ \tilde{F} &= C^T(Q - SR^{-1}S^T)C, \\ \tilde{B} &= E^{-1}B, \\ \tilde{G} &= \tilde{B}R^{-1}\tilde{B}^T. \end{aligned}$$

Let X_j be the sequence of approximate solutions produced by Algorithm 2 applied to (1) with starting guess X_0 and let $\tilde{X}_j$ be the sequence of approximate solutions produced by Algorithm 2 applied to (7) with starting guess $\tilde{X}_0 = E^TX_0E$. It is easy to verify that $\tilde{X}_j = E^TX_jE$. Note that because E is nonsingular, the boundedness, convergence (or lack thereof), and rate of convergence of the two sequences X_j and $\tilde{X}_j$ are identical and $\lambda(E, A - BX_j) \subset \mathbb{C}^-$ if and only if $\tilde{A} - \tilde{G}\tilde{X}_j$ is stable. The residual satisfies $\tilde{R}(\tilde{X}_j) = R(X_j)$, and X^* satisfies $R(X^*) = 0$ if and only if $\tilde{X}^* = E^TX^*E$ satisfies $\tilde{R}(\tilde{X}^*) = 0$. Note further that the sequence of step sizes t_j produced by Algorithm 2 is equal in both cases. The coefficient matrices $\tilde{F}$ and $\tilde{G} = \tilde{B}R^{-1}\tilde{B}^T$ are symmetric, positive semidefinite, because $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix}$ and R are.

In Remark 1, it was observed that a local minimizer of $\|R(X_j + tN_j)\|_F$ can be found in the interval $[0, 2]$. Now we show that this interval is the natural search interval in order to obtain stabilizing iterates X_j.

The inertia of a matrix M is the triple $\mathrm{In}(M) = (\pi(M), \nu(M), \delta(M))$ where $\pi(M)$, $\nu(M)$, and $\delta(M)$ are the number of eigenvalues with positive, negative, and zero real part respectively. In the sequel, we will write $\mathrm{In}(M) \leq \mathrm{In}(N)$ for any two $n \times n$ matrices M and N such that $\pi(M) \leq \pi(N)$ and $\nu(M) \leq \nu(N)$.

We will need the following version of Lyapunov's inertia theorem which can be found, for example, in [20, page 447].

Theorem 6 *If $AH + HA^T = W \geq 0$, where H is symmetric and $\delta(A) = 0$, then $\mathrm{In}(H) \leq \mathrm{In}(A)$.*

We will use a trivial corollary of Theorem 6.

Corollary 7 *If $AH + HA^T = -W \leq 0$, where H is symmetric and $\delta(A) = 0$, then $\mathrm{In}(-H) \leq \mathrm{In}(A)$.*

Further, we need a result relating the Lyapunov stability theory and detectability.

Lemma 8 *If $H = H^T$ satisfies*

$$AH + HA^T = -W \leq -C^TC \tag{8}$$

where (A, C) defines a detectable pair, then

$$\nu(A) = n \Longleftrightarrow \nu(H) = 0.$$

Proof: Assume first $\nu(A) = n$, that is, A is stable. Then $\nu(H) = 0$ follows immediately from Corollary 7.

Now, for the opposite direction, $\nu(H) = 0$ implies that H is positive semidefinite. To prove that $\nu(A) = n$, we assume the contrapositive, i.e., A has at least one eigenvalue λ with $\mathrm{Re}(\lambda) \geq 0$. We denote the corresponding right eigenvector by w. Since (A, C) is detectable, $Cw \neq 0$. Thus, from (8) we obtain

$$0 > -w^H C^T C w \geq w^H (A^T H + HA) w = 2\mathrm{Re}(\lambda) w^H H w$$

which contradicts the positive semidefiniteness of H. □

The following lemma shows that the iterates $\tilde{X}_j + t_j \tilde{N}_j$ are stabilizing if the starting guess $\tilde{X}_0$ is stabilizing,

Lemma 9 *Suppose that $(\tilde{A}, \tilde{C})$ is detectable where $\tilde{F} = \tilde{C}^T \tilde{C}$ is a full-rank factorization of $\tilde{F}$. If $\tilde{A} - \tilde{G}\tilde{X}_j$ is stable and $t \in [0, 2]$, then $\tilde{A} - \tilde{G}(\tilde{X}_j + t\tilde{N}_j)$ is also stable.*

Proof: The Newton Step $\tilde{N}_j$ is determined by

$$\begin{aligned} (\tilde{A} - \tilde{G}\tilde{X}_j)^T (\tilde{X}_j + \tilde{N}_j) + (\tilde{X}_j + \tilde{N}_j)(\tilde{A} - \tilde{G}\tilde{X}_j) &= -\tilde{F} - \tilde{X}_j \tilde{G} \tilde{X}_j \\ &\leq -\tilde{F}. \end{aligned} \tag{9}$$

Since $\tilde{A} - \tilde{G}\tilde{X}_j$ is stable, Lemma 8 implies that $\tilde{X}_j + \tilde{N}_j$ is positive semidefinite. On the other hand, for $t \in [0, 2]$, (9) is equivalent to

$$\begin{aligned} (\tilde{A} - \tilde{G}(\tilde{X}_j + t\tilde{N}_j))^T (\tilde{X}_j + \tilde{N}_j) + (\tilde{X}_j + \tilde{N}_j)(\tilde{A} - \tilde{G}(\tilde{X}_j + t\tilde{N}_j)) \\ = -\tilde{F} - (\tilde{X}_j + t\tilde{N}_j)\tilde{G}(\tilde{X}_j + t\tilde{N}_j) + t(t-2)\tilde{N}_j \tilde{G} \tilde{N}_j \\ \leq -\tilde{F}. \end{aligned}$$

Now Lemma 8 and the positive semidefiniteness of $\tilde{X}_j + \tilde{N}_j$ imply that $\tilde{A} - \tilde{G}(\tilde{X}_j + t\tilde{N}_j)$ is stable. □

The *Lyapunov operator* corresponding to the Lyapunov equations in Step 2.2 of Algorithm 2 is defined by

$$\tilde{\Omega}_j(Z) = (\tilde{A} - \tilde{G}\tilde{X}_j)^T Z + Z(\tilde{A} - \tilde{G}\tilde{X}_j)$$

for $Z \in \mathbb{R}^{n \times n}$ and $j = 1, 2, \ldots$. A corollary of Lemma 9 is that with a stabilizing starting guess, Algorithm 2 can not fail due to a singular Lyapunov operator.

Corollary 10 *If $(\tilde{A}, \tilde{C})$ is detectable, $\tilde{X}_0$ is stabilizing, and Algorithm 2 is applied to (7), then the Lyapunov operator $\tilde{\Omega}_j$ in Step 2.2 is nonsingular for all j and the sequence of approximate solutions $\tilde{X}_j$ is well defined.*

Remark 11 Under the stronger hypothesis of observability of the matrix pair $(\tilde{A}, \tilde{C})$ and by using the Lyapunov inertia theorem for controllable matrix pairs given, e.g., in [20, Section 13.1, Theorem 4] in an "observability form", the proof of Lemma 9 can be modified to obtain the following result.

If the Lyapunov operator in Step 2.2 of Algorithm 2 is nonsingular, then for all $t \in [0, 2]$*, we have* $\mathrm{In}\left(\tilde{A} - \tilde{G}\tilde{X}_j\right) = \mathrm{In}\left(\tilde{A} - \tilde{G}(\tilde{X}_j + tN_j)\right)$ *and*

$$\delta\left(\tilde{A} - \tilde{G}\tilde{X}_j\right) = \delta\left(\tilde{A} - \tilde{G}(\tilde{X}_j + tN_j)\right) = 0.$$

In other words, the inertia of $\tilde{A} - \tilde{G}\tilde{X}_j$ is invariant throughout Algorithm 2 and Corollary 10 applies to every starting guess for which $\delta(\tilde{A} - \tilde{G}\tilde{X}_0) = 0$. This observation shows that Algorithm 2 can also be used for computing a solution of (1) different from the stabilizing one. For example, the *antistabilizing* solution $\tilde{X}^+$ of (7) is characterized by the property that the eigenvalues of $\tilde{A} - \tilde{G}\tilde{X}^+$ are in the open *right* half plane. According to the above remark, this solution can be computed by Algorithm 2 using an antistabilizing starting guess.

We will need the following technical characterization of controllability.

Lemma 12 *Suppose that* $A \in \mathbb{R}^{n\times n}$*,* $B \in \mathbb{R}^{n\times m}$*,* $R \in \mathbb{R}^{m\times m}$*, and* R *is symmetric positive definite. The pair* (A, B) *is controllable if and only if the only matrix* $Y = Y^T$ *satisfying* $YBR^{-1}B^TY = 0$ *and* $A^TY + YA \geq 0$ *is* $Y = 0$*.*

Since the proof is rather technical, it is given in Appendix A.

As seen in Remark 1, the sequence of residuals $\tilde{R}(\tilde{X}_j)$ produced by Algorithm 2 is monotonically decreasing and, in particular, bounded. The next lemma shows that if $(\tilde{A}, \tilde{B})$ is controllable, then the iterates $\tilde{X}_j$ are also bounded.

Lemma 13 *Suppose that* $\tilde{X}_j$*,* $j = 1, 2, 3, \ldots$ *is a sequence of symmetric, n-by-n matrices such that* $\tilde{R}(\tilde{X}_j)$ *is bounded. If* $(\tilde{A}, \tilde{B})$ *is a controllable pair, then the sequence* $\tilde{X}_j$ *is bounded.*

Proof: We will prove the contrapositive: if $\tilde{X}_j$ is unbounded, then $(\tilde{A}, \tilde{B})$ is not controllable. Without loss of generality we may assume that $\lim_{j\to\infty} \|\tilde{X}_j\|_F = \infty$. (If not, we may consider a subsequence for which this assertion holds.) Define $\xi_j = \|\tilde{X}_j\|_F$ and $\tilde{Y}_j = \tilde{X}_j/\xi_j$. The $\tilde{Y}_j$'s are bounded, so there is a convergent subsequence which we may assume without loss of generality is the whole sequence. Let $\tilde{Y} = \lim_{j\to\infty} \tilde{Y}_j$. Note that $\tilde{Y} \neq 0$. From definition (7), we have

$$\frac{1}{\xi_j}\left(\tilde{F} - \tilde{R}(\tilde{X}_j)\right) + \tilde{A}^T\tilde{Y}_j + \tilde{Y}_j\tilde{A} = \xi_j\tilde{Y}_j\tilde{B}R^{-1}\tilde{B}^T\tilde{Y}_j \tag{10}$$

Because $\tilde{R}(\tilde{X}_j)$ is bounded, the first term on the left-hand-side of (10) tends to zero as $j \to \infty$. The second term approaches the finite limit $\tilde{A}^T\tilde{Y} + \tilde{Y}\tilde{A}$. From the right-hand-side, it is clear that this is a limit of positive semidefinite matrices, and hence is positive semidefinite. Dividing (10) by ξ_j and letting $j \to \infty$ gives $\tilde{Y}\tilde{B}R^{-1}\tilde{B}^T\tilde{Y} = 0$. It follows from Lemma 12 that $(\tilde{A}, \tilde{B})$ is uncontrollable. □

We are now ready to prove that Algorithm 2 reduces the residual $\tilde{R}(\tilde{X}_j)$ (and hence $R(X_j)$) asymptotically to zero if the computed step sizes are bounded away from zero.

Theorem 14 *If $(\tilde{A}, \tilde{B})$ is a controllable pair, and the sequence of step sizes t_j computed by Algorithm 2 is uniformly bounded from below by $t_L > 0$, then the residual norms $\|\tilde{R}(\tilde{X}_j)\|_F$ decrease monotonically to zero and cluster points of the sequence $\tilde{X}_j$ are solutions of the algebraic Riccati equation (1).*

Proof: Lemma 13 shows that the sequence of approximate roots $\tilde{X}_j$ is bounded. Consequently, the steps $t_j \tilde{N}_j$ are also bounded. Here $\tilde{N}_j = E^T N_j E$ and t_j is the step size computed by minimizing $\tilde{f}_j(t) = \|\tilde{R}(\tilde{X}_j + t\tilde{N}_j)\|_F^2$. The $t_j \in [0, 2]$ also form a bounded sequence and since we assumed $0 < t_L \leq t_j$ for all j, the $\tilde{N}_j$'s are bounded, too. Select a subsequence $\tilde{X}_{j_k}$ of the $\tilde{X}_j$'s such that $\hat{X} = \lim_{k\to\infty} \tilde{X}_{j_k}$, $\hat{t} = \lim_{k\to\infty} t_{j_k}$, and $\hat{N} = \lim_{k\to\infty} \tilde{N}_{j_k}$ exist. Note that the residual norms $\|\tilde{R}(\tilde{X}_j)\|_F$ are monotonically decreasing, so they approach a limit and hence

$$\|\tilde{R}(\hat{X} + \hat{t}\hat{N})\|_F = \|\tilde{R}(\hat{X})\|_F. \tag{11}$$

Thus, the coefficients α_{j_k}, β_{j_k}, and γ_{j_k} in (5) approach limits and the minimum value of the polynomial $\hat{f}(t) = \|\tilde{R}(\hat{X} + t\hat{N})\|_F^2$ is the limit of the minimum values of the $\tilde{f}_{j_k}$'s, i.e., $\lim_{k\to\infty} f_{j_k}(t_{j_k}) = \hat{f}(\hat{t}) \leq \hat{f}(0)$. However, using (11), we obtain

$$\hat{f}(0) = \|\tilde{R}(\hat{X})\|_F = \|\tilde{R}(\hat{X} + \hat{t}\hat{N})\|_F = \hat{f}(\hat{t}).$$

It follows that $\hat{f}'(0) = 0$. But as observed in Remark 1, $\hat{f}'(0) = -2\|\tilde{R}(\hat{X})\|^2$. Therefore, $\tilde{R}(\hat{X}) = 0$. $\square$

Collecting the results derived so far, we have the following convergence result for Newton's method with exact line search.

Theorem 15 *Suppose $(\tilde{A}, \tilde{B})$ defines a controllable matrix pair and $(\tilde{A}, \tilde{C})$ is detectable where $\tilde{F} = \tilde{C}^T\tilde{C}$ is a full-rank factorization of $\tilde{F}$. If Algorithm 2 is applied to the algebraic Riccati equation (7) with a stabilizing starting guess $\tilde{X}_0$ and the computed step sizes t_j are bounded away from zero, then $\tilde{X}^* = \lim_{j\to\infty} \tilde{X}_j$ exists and is the stabilizing solution of (7).*

Proof: Lemma 9 and Corollary 10 imply that $\tilde{X}_j$ is well defined and stabilizing for all j. Lemma 13 implies that the sequence $\tilde{X}_j$ is bounded and we can therefore apply Theorem 14 from which we can conclude that $\lim_{j\to\infty} \tilde{R}(\tilde{X}_j) = 0$ and cluster points of the sequence X_j are stabilizing solutions of (1). However, under the given assumptions, the stabilizing solution of (7) is unique. A bounded sequence with only one cluster point is convergent. $\square$

Remark 16 The above convergence result relies on the fact that $t_j \geq t_L$ for all j and a given constant $t_L > 0$. We can modify Algorithm 2 such that the step size is set to one if t_j drops below a prescribed (small) constant. By (9) it is clear that the so-defined new iterate $X_{j+1} = X_j + N_j$ is positive semidefinite. We can now apply the Newton iteration (Algorithm 1) with the "starting guess" X_{j+1} and use the standard convergence theory for Newton's method [19, 24, 27] to show that iterates produced by this hybrid algorithm converge to the stabilizing solution of (1).

In our numerical experiments, very small step sizes occured only at the very beginning of the iteration if the starting guess already yielded a residual norm within the order of the limiting accuray. In such a case, neither Newton's method nor Exact Line Search can be expected to improve the accuracy of the approximate solution of (1) any further.

Algorithm 2 inherits its quadratic convergence from Newton's method. To show this, we show that in the region of quadratic convergence of Newton's method the choice of $t = 1$ does a good job of minimizing $\|R(X + tN)\|_F$ [25]. Suppose that $\tilde{X}_j$ is within the region of quadratic convergence of Newton's method. In this case [24],

$$\tilde{N}_j = \tilde{X}^* - \tilde{X}_j + O\left(\|\tilde{X}^* - \tilde{X}_j\|_F^2\right) \tag{12}$$

and

$$\|\tilde{R}(\tilde{X}_j + \tilde{N}_j)\|_F = O\left(\|\tilde{X}^* - \tilde{X}_j\|_F^2\right). \tag{13}$$

The residual produced by Algorithm 2 satisfies

$$\begin{aligned}
&\tilde{R}(\tilde{X}_{j+1}) \\
&\quad = \tilde{R}(\tilde{X}^* + (\tilde{X}_j + \tilde{N}_j - \tilde{X}^*) + (t_j - 1)\tilde{N}_j) \\
&\quad = (\tilde{A} - \tilde{G}\tilde{X}^*)^T(\tilde{X}_j + \tilde{N}_j - \tilde{X}^*) + (\tilde{X}_j + \tilde{N}_j - \tilde{X}^*)(\tilde{A} - \tilde{G}\tilde{X}^*) \\
&\qquad + (t_j - 1)\left((\tilde{A} - \tilde{G}\tilde{X}^*)^T\tilde{N}_j + \tilde{N}_j(\tilde{A} - \tilde{G}\tilde{X}^*)\right) \\
&\qquad - \left((\tilde{X}_j + \tilde{N}_j - \tilde{X}^*) + (t_j - 1)\tilde{N}_j\right)\tilde{G}\left((\tilde{X}_j + \tilde{N}_j - \tilde{X}^*) + (t_j - 1)\tilde{N}_j\right).
\end{aligned}$$

Taking norms, using (12), and recognizing that $|t_j - 1| \le 1$ gives

$$\begin{aligned}
&\|\tilde{R}(\tilde{X}_j + t_j\tilde{N}_j)\|_F \\
&\qquad \le 2|t_j - 1|\|\tilde{X}^* - \tilde{X}_j\|_F\|A - GX^*\|_F + O\left(\|\tilde{X}_j - \tilde{X}^*\|_F^2\right).
\end{aligned} \tag{14}$$

Recall that $t_j \in [0, 2]$ is chosen to minimize $\|\tilde{R}(\tilde{X}_j + t\tilde{N}_j)\|_F$, so (13) implies

$$\|\tilde{R}(\tilde{X}_j + t_j\tilde{N}_j)\|_F \le \|\tilde{R}(\tilde{X}_j + \tilde{N}_j)\|_F = O\left(\|\tilde{X}^* - \tilde{X}_j\|_F^2\right). \tag{15}$$

It follows from (14) and (15) that $|t_j - 1| = O(\|\tilde{X}^* - \tilde{X}_j\|_F)$. Hence,

$$\begin{aligned}
\|\tilde{X}^* - \tilde{X}_{j+1}\|_F &= \|\tilde{X}^* - \left(\tilde{X}_j + \tilde{N}_j + (t_j - 1)\tilde{N}_j\right)\|_F \\
&\le \|\tilde{X}^* - \left(\tilde{X}_j + \tilde{N}_j\right)\|_F + |1 - t_j|\|\tilde{N}_j\|_F \\
&= O\left(\|\tilde{X}^* - \tilde{X}_j\|_F^2\right).
\end{aligned}$$

The following theorem summarizes the convergence theory. At the risk of some ambiguity, the theorem does not specify which precise variation of controllability and detectability of descriptor systems is required. Common controllability and detectability definitions for descriptor systems coincide in this case, because E is nonsingular [30, 31].

Theorem 17 *If (E, A, B) is controllable, $(E, A, C^T(Q - SR^{-1}S^T)C)$ is detectable, and $X_0 = X_0^T$ is stabilizing in the sense that $\lambda(E, A - BK_0) \subset \mathbb{C}^-$, then the sequence of approximate solutions X_j produced by the modified Algorithm described in Remark 16 converges quadratically to the stabilizing solution X^*, at each step, $\lambda(E, A - BK_j) \subset \mathbb{C}^-$, and the residual norms $\|R(X_j)\|_F$ converge monotonically and quadratically to zero.*

This theorem is more general than the one stated in [24] since it does not require X_0 to be positive semidefinite. In contrast to Newton's method, the iterates X_j are not necessarily positive semidefinite and they do not necessarily converge monotonically (in terms of definiteness). On the other hand, the theorem needs the strong hypothesis of controllability. The numerical examples in Section 4 suggest that this can be weakened to stabilizability but at this writing, we are not aware of a proof for this conjecture.

4 Numerical examples

Newton's Method (Algorithms 1), and Exact Line Search (Algorithm 2) were implemented as MATLAB [23] functions. We compared the algorithms on the examples of the benchmark collection for continuous-time algebraic Riccati equations [5], several randomly generated examples, and some examples contrived to make Algorithm 1 perform poorly.

Although theoretical results ensure the existence of a minimizing $t_j \in [0, 2]$ as well as $\|R(X_j + t_jN_j)\|_F \leq \|R(X_j)\|_F$ in each step, rounding errors may cause loss of these properties. This is usually caused by catastrophic loss of significance in the computation of the residual $R(X_j)$. It is a sign that the limiting accuracy has been reached.

For our experiments we did not use the hybrid algorithm proposed in Remark 16 in order to demonstrate the behaviour of Exact Line Search and to monitor the possible convergence of the t_j's to zero.

All computations were done under MATLAB Version 4.2a [23] on Hewlett Packard Apollo series 700 computers under IEEE double precision and machine precision $\varepsilon = 2.2204 \cdot 10^{-16}$ at the Technical University Chemnitz–Zwickau, Germany.

Example 1, continued
Using the same starting guess X_0 as before, one step of Exact Line Search reduced $\|R(X_j)\|_F$ as much as twenty-four steps of Newton's method. This is consequence of Remark 3, since the example actually consists of two uncoupled scalar Riccati equations. The starting guess fortuitously satisfies one of the two uncoupled scalar Riccati equations. With the poor starting guess $X_0 = 100I$ Exact Line Search took 9 iterations to converge while Newton's method took 17.

Example 2 We randomly generated about 50 examples with $5 \leq n \leq 100$ and $1 \leq m, p \leq n$, using normal distribution. Q and R were either set to I_p and I_m respectively or generated by $Q = Q_0^TQ_0$, $R = R_0^TR_0$. A stabilizing starting guess X_0

was generated by the method described in [1, 15, 28]. On average, the exact line search method needed 20% fewer iterations than Newton's method. The number of iterations saved tended to be greater for larger values of n, m, and p. For $n = m = p = 100$, Exact Line Search needed up to 40% fewer iterations.

We also explicitly generated examples with stabilizable, but uncontrollable data. Although the convergence theory derived in Section 3 was based on assuming controllability, Exact Line Search converged to the stabilizing solution for all these examples. This suggests that the convergence theory also holds if controllability is weakened to stabilizability.

Convergence of the step sizes t_j to zero was never observed for the randomly generated examples.

The next examples are taken from [5]. Here, we report only the most intriguing results obtained by testing the exact line search method for all those examples. As for the randomly generated examples (see Example 2), no convergence problems occured for uncontrollable data and convergence of the step sizes to zero was never observed.

Example 3 This is Example 15 in [5] and Example 4 in [21]. The system matrices describe a mathematical model of position and velocity control for a string of N high-speed vehicles. We have $n = 2N - 1$, $m = N$, and $p = N - 1$.

$$A = \begin{bmatrix} A_{11} & A_{12} & 0 & \dots & & 0 \\ 0 & A_{22} & A_{23} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & 0 & A_{N-2,N-2} & A_{N-2,N-1} & 0 \\ & & & 0 & A_{N-1,N-1} & \begin{matrix} 0 \\ -1 \end{matrix} \\ 0 & \dots & & & 0 \quad 0 & -1 \end{bmatrix}$$

where

$$\begin{aligned} A_{k,k} &= \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}, \quad 1 \leq k \leq N-1, \\ A_{k+1,k} &= \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix}, \quad 1 \leq k \leq N-2, \end{aligned}$$

and

$$\begin{aligned} E &= R = C = I_n, \qquad S = 0, \\ B &= \mathrm{diag}(1, 0, 1, 0, \dots, 1, 0, 1), \\ Q &= \mathrm{diag}(0, 10, 0, 10, \dots, 0, 10, 0), \end{aligned}$$

Stabilizing starting guesses X_0 were generated by the method described in [1, 15, 28].

Table 1 shows the number of iterations and the Frobenius norm of the last absolute and relative residual. ($\hat{X}$ denotes the computed approximation to X^*.)

From these figures we see that the number of saved iterations tends to increase with growing problem size n as it was also observed for the randomly generated examples, see Example 2.

	Newton's method			Exact Line Search		
n	it.	$\Vert R(\tilde{X})\Vert_F$	$\frac{\Vert R(\tilde{X})\Vert_F}{\Vert\tilde{X}\Vert_F}$	it.	$\Vert R(\hat{X})\Vert_F$	$\frac{\Vert R(\hat{X})\Vert_F}{\Vert\hat{X}\Vert_F}$
9	5	$1.2\cdot 10^{-13}$	$6.1\cdot 10^{-15}$	5	$5.6\cdot 10^{-15}$	$2.9\cdot 10^{-16}$
29	7	$1.4\cdot 10^{-14}$	$3.3\cdot 10^{-16}$	5	$5.0\cdot 10^{-14}$	$1.2\cdot 10^{-15}$
49	7	$6.9\cdot 10^{-14}$	$1.1\cdot 10^{-15}$	6	$2.2\cdot 10^{-14}$	$3.6\cdot 10^{-16}$
99	8	$8.2\cdot 10^{-14}$	$8.1\cdot 10^{-16}$	6	$3.8\cdot 10^{-14}$	$3.8\cdot 10^{-16}$
149	9	$6.5\cdot 10^{-14}$	$4.7\cdot 10^{-16}$	6	$6.9\cdot 10^{-14}$	$5.0\cdot 10^{-16}$
199	9	$1.1\cdot 10^{-13}$	$6.5\cdot 10^{-16}$	6	$8.0\cdot 10^{-14}$	$4.6\cdot 10^{-16}$

Table 1: Example 3

Example 4 This is Example 14 from [5] and Example 2 from [2]. Here, A depends upon a parameter δ. If $\delta \to 0$, the system approaches one which is unstabilizable and a conjugate complex pair of the closed loop eigenvalues approaches the imaginary axis. The system matrices are given by

$$A = \begin{bmatrix} -\delta & 1 & 0 & 0 \\ -1 & -\delta & 0 & 0 \\ 0 & 0 & \delta & 1 \\ 0 & 0 & -1 & \delta \end{bmatrix}, \quad B = C^T = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$
$$E = I_4, \quad Q = R = 1, \quad S = 0.$$

Stabilizing starting guesses X_0 were generated by the method described in [1, 15, 28]. Figures 2–5 show the behavior of the algorithms for several values of δ. The number of iterations saved by using the exact line search method instead of Newton's method increases with δ approaching zero.

Example 5 The data of this example describes a magnetic tape control problem [8], [5, Example 13].

$$A = \begin{bmatrix} 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0.345 & 0 \\ 0 & -0.524/\delta & -0.465/\delta & 0.262/\delta \\ 0 & 0 & 0 & -1/\delta \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1/\delta \end{bmatrix}, \quad R = 1,$$
$$E = I_4, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad Q = I_2, \quad S = 0.$$

Since A has one zero and 3 stable eigenvalues,

$$X_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

is a stabilizing starting guess. As $\delta \to 0$, (A, B) gets close to an unstabilizable matrix pair. Figure 6 shows the behavior for $\delta = 10^{-3},\ 10^{-5}$.

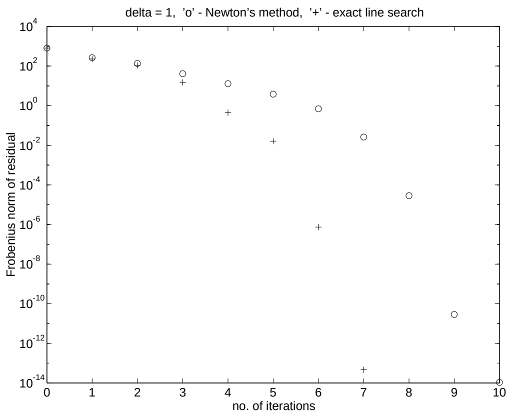

Figure 2: Example 4, $\delta = 1$

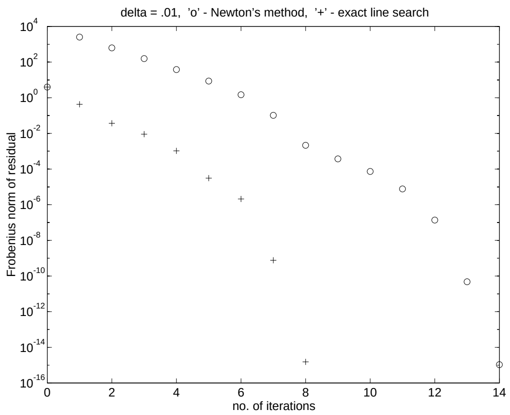

Figure 3: Example 4, $\delta = 10^{-2}$

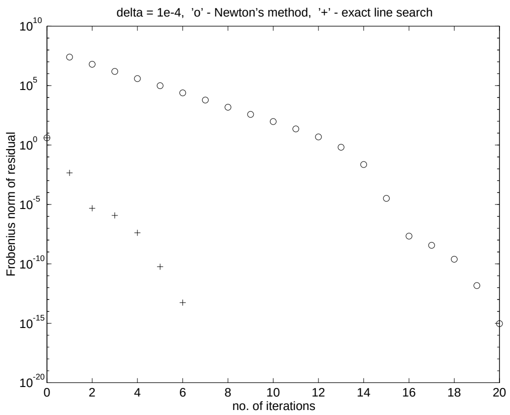

Figure 4: Example 4, $\delta = 10^{-4}$

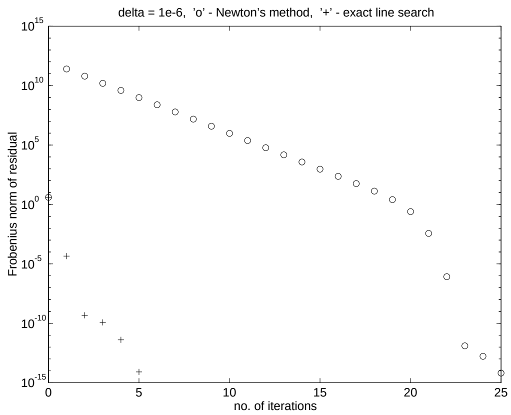

Figure 5: Example 4, $\delta = 10^{-6}$

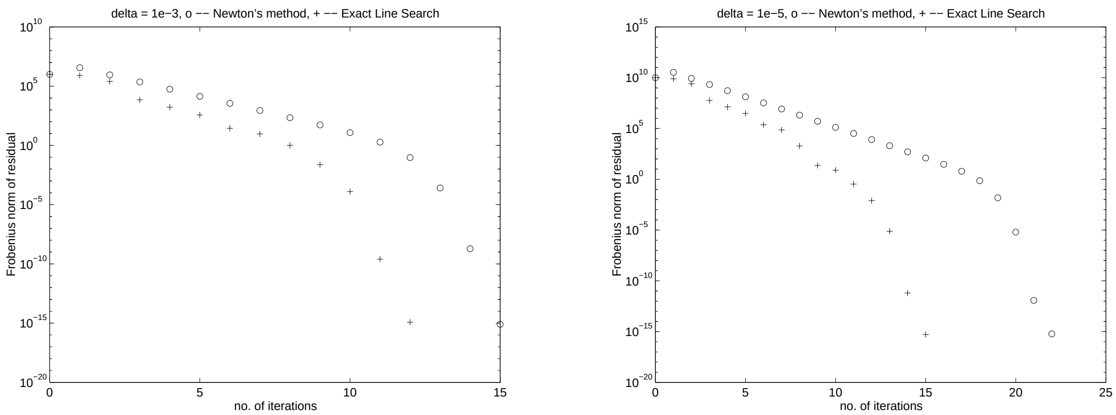

Figure 6: Residual norms in Example 5

Example 6 One of the situations in which defect correction or iterative refinement [7, 17, 18] has the most to offer is when the Riccati equation is highly ill-conditioned. Rounding errors make it unlikely that any Riccati solver will produce much accuracy, but with its excellent, structure preserving rounding error properties, Newton's method is likely to squeeze out as much accuracy as possible. This example is contrived to be highly ill-conditioned. Here we use $n = m = p$,

$$E = R = I \qquad A = S = 0 \qquad B = 10^3 I \qquad C = I - \frac{2}{n} ee^T$$

where $e \in \mathbb{R}^n$ is the vector of ones, and

$$Q = \mathrm{diag}(\frac{1}{9^1}, \frac{1}{9^2}, \frac{1}{9^2}, \frac{1}{9^3}, \frac{1}{9^3}, \ldots).$$

The exact stabilizing solution is given by

$$X^* = 10^{-6}\, C^T Q C.$$

We obtained the starting guess as $X_0 = (X + X^T)/2$ where X is the "solution" of (1) computed by the MATLAB function `care` provided by A. Laub which implements the Schur vector method [21] extended to the generalized algebraic Riccati equation (1) as discussed in [2]. (Although the stabilizing solution X^* is symmetric, rounding errors in `care` may cause it to return a nonsymmetric "solution.") Observe in Figures 7 and 8 that Newton's method increases the initial residual norm by several orders of magnitude. Predictably, the graph of relative errors closely matched the graph of residuals.

This is an extremely ill-conditioned example. Using the condition number K^+ proposed in [6, 16] we obtain $K^+ = 1.8 \cdot 10^9$ for $n = 40$ and $K^+ = 4.3 \cdot 10^{11}$ for $n = 50$. Rounding errors made while forming $C^T Q C$ are sufficient to change the smaller eigenvalues and corresponding invariant subspaces of the solution X^* and the closed loop system $A - BR^{-1}B^T X^*$ by over 100%. The closed loop poles are so close to the imaginary axis that the symmetrized `care` solution for $n = 50$ did not appear to be stabilizing as it should have been; one of the smaller eigenvalues of $A - BR^{-1}B^T X_0$ computed by MATLAB was of the wrong sign. (Exact Line Search preserves inertia, so for $n = 50$ it did not converge to a stabilizing solution either.)

The relative errors in Figures 7 and 8 are consistent with the condition number and the precision of the arithmetic. Notice in Figure 7 that for $n = 40$, refining the `care` solution reduced the relative error by more than three orders of magnitude. In both examples, the first Newton step is a disaster. Although in the $n = 50$ case, refining the `care` solution did not significantly reduce its relative error, Exact Line Search keeps the residual small but Newton's method does not.

5 Conclusions

We have introduced and studied an exact line search method based on Newton's method for solving (generalized) continuous–time algebraic Riccati equations. This

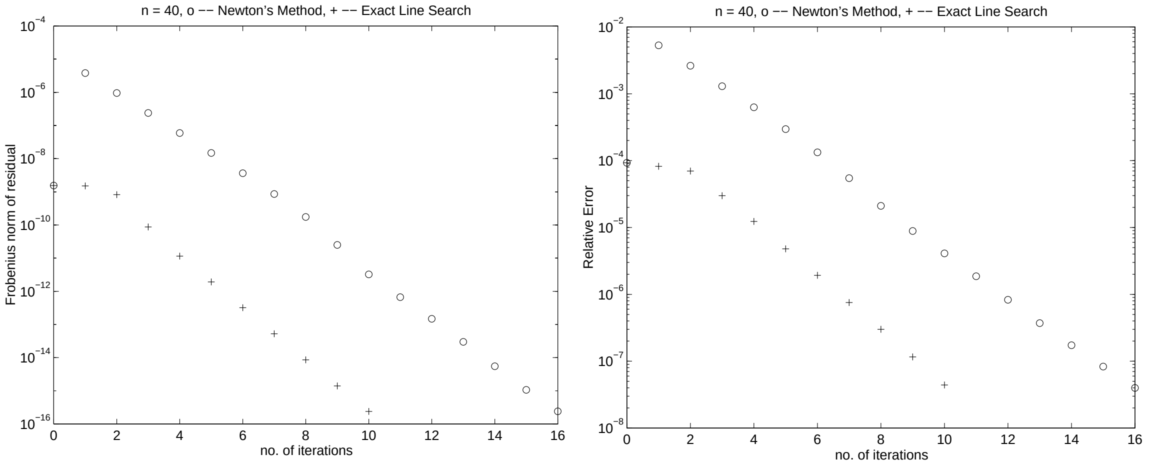

Figure 7: Residual norms and relative errors in Example 6, $n = 40$

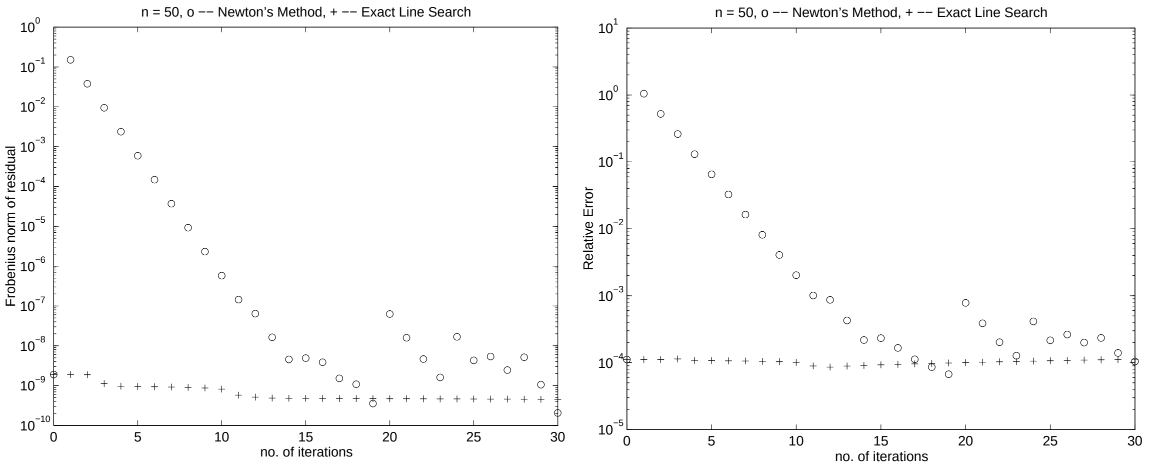

Figure 8: Residual norms and relative errors in Example 6, $n = 50$

method has convergence properties similar to Newton's method. Numerical experiments show that in some cases, it significantly reduces the number of iteration steps. In addition, it is more robust in the presence of rounding errors. Used as a defect correction method or for iterative refinement it maximizes the efficiency of Newton's method to obtain the highest possible accuracy.

The same technique may be used to improve the behavior of the secant method proposed in [10], the simplified Newton method [26] or for Newton's method applied to the solution of (generalized) discrete–time algebraic Riccati equations.

Numerical experiments suggest that the convergence theory for the exact line search method also holds if the assumed controllability is weakened to stabilizability and the assumption that the step sizes are bounded from below is dropped. However, at this writing we can not prove these conjectures. This will be the topic of further investigations.

6 Acknowledgements

We wish to thank Stephen G. Nash for his helpful comments on the proof for quadratic convergence of the exact line search method and Alan Laub for providing the MATLAB function `care.m` for solving continuous-time algebraic Riccati equations.

A Proof of Lemma 12

We will prove the contrapositive of the statement in Lemma 12: the pair (A, B) is uncontrollable if and only if there exists $Y = Y^T \neq 0$ such that $YBR^{-1}B^TY = 0$ and $A^TY + YA \geq 0$.

If (A, B) is uncontrollable, then there exists a left eigenvector w of A that lies in the left null space of B. Let λ_r be the real part of the corresponding eigenvalue of A. If $Y = \mathrm{sign}(\lambda_r)\, ww^T$, then $YBR^{-1}B^TY = ww^TBR^{-1}B^Tww^T = 0$ and $A^TY + YA = 2|\lambda_r|Y = |\lambda_r|ww^T$ is positive semidefinite.

For the converse, assume that there exists a symmetric matrix $Y \neq 0$ such that $A^TY + YA \geq 0$ and $YBR^{-1}B^TY = 0$. We will show that (A, B) is uncontrollable by constructing a left eigenvector of A belonging to the left null space of B.

By choosing an appropriate orthonormal basis, we may arrange that A, Y and $BR^{-1}B^T$ take the form

$$BR^{-1}B^T = \begin{array}{c} \\ h \\ k \\ n-h-k \end{array} \begin{array}{c} \begin{array}{ccc} h & k & n-h-k \end{array} \\ \left[\begin{array}{ccc} G_{11} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right] \end{array}$$

and by analogous partitioning

$$Y = \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & Y_{22} & 0 \\ 0 & 0 & 0 \end{array}\right], \qquad A = \left[\begin{array}{ccc} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{array}\right]$$

where G_{11} and Y_{22} are nonsingular. The assumption that $Y \neq 0$ implies that $k > 0$. However, it is possible that either $h = 0$ or $n - h - k = 0$ in which case the corresponding rows and columns do not appear. In this basis, $A^TY + YA$ takes the form

$$A^TY + YA = \begin{array}{c} \\ h \\ k \\ n-h-k \end{array} \begin{array}{c} \begin{array}{ccc} h & k & n-h-k \end{array} \\ \left[\begin{array}{ccc} 0 & A_{21}^TY_{22} & 0 \\ Y_{22}A_{21} & A_{22}^TY_{22} + Y_{22}A_{22} & Y_{22}A_{23} \\ 0 & A_{23}^TY_{22} & 0 \end{array}\right] \end{array}.$$

By hypothesis, this matrix is positive semidefinite, so $Y_{22}A_{21} = 0$ and $Y_{22}A_{23} = 0$. It follows from the nonsingularity of Y_{22} that $A_{21} = 0$ and $A_{23} = 0$.

Let $w_2 \in \mathbb{R}^k$ be a left eigenvector of A_{22}. Define $w \in \mathbb{R}^n$ as $w = [w_1, w_2, w_3]$ where $w_1 = 0 \in \mathbb{R}^h$ and $w_3 = 0 \in \mathbb{R}^{n-h-k}$. The vector w is a left eigenvector of A belonging to the left null space of B. $\square$

References

[1] E. S. Armstrong, *An extension of Bass' algorithm for stabilizing linear continuous constant systems*, IEEE Trans. Automat. Control, AC–20 (1975), pp. 153–154.

[2] W. Arnold, III and A. Laub, *Generalized eigenproblem algorithms and software for algebraic Riccati equations*, Proc. IEEE, 72 (1984), pp. 1746–1754.

[3] R. Bartels and G. Stewart, *Solution of the matrix equation $AX+XB=C$: Algorithm 432*, Comm. ACM, 15 (1972), pp. 820–826.

[4] P. Benner and R. Byers, *Step size control for Newton's method applied to algebraic Riccati equations*, in Proc. Fifth SIAM Conf. Appl. Lin. Alg., Snowbird, UT, J. Lewis, ed., Philadelphia, PA, 1994, SIAM, pp. 177–181.

[5] P. Benner, A. Laub, and V. Mehrmann, *A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case*, Tech. Report SPC 95_22, Fak. f. Mathematik, TU Chemnitz–Zwickau, 09107 Chemnitz, FRG, 1995.

[6] R. Byers, *Numerical condition of the algebraic Riccati equation*, Contemp. Math., 47 (1985), pp. 35–49.

[7] ——, *Solving the algebraic Riccati equation with the matrix sign function*, Linear Algebra Appl., 85 (1987), pp. 267–279.

[8] J. Chow and P. Kokotovic, *A decomposition of near–optimum regulators for systems with slow and fast modes*, IEEE Trans. Automat. Control, AC-21 (1976), pp. 701–705.

[9] J. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall, Englewood Cliffs, New Jersey, 1983.

[10] L. Dieci, Y. Lee, and R. Russel, *Iterative methods for solving algebraic Riccati equations*, report, Dept. Math. & Stat., Simon Fraser University, Burnaby, Canada, 1988.

[11] J. Gardiner, A. Laub, J. Amato, and C. Moler, *Solution of the Sylvester matrix equation $AXB+CXD=E$*, ACM Trans. Math. Software, 18 (1992), pp. 223–231.

[12] J. Gardiner, M. Wette, A. Laub, J. Amato, and C. Moler, *Algorithm 705: A Fortran-77 software package for solving the Sylvester matrix equation $AXB^T+CXD^T=E$*, ACM Trans. Math. Software, 18 (1992), pp. 232–238.

[13] A. Ghavimi, C. Kenney, and A. Laub, *Local convergence analysis of conjugate gradient methods for solving algebraic Riccati equations*, IEEE Trans. Automat. Control, 37 (1992), pp. 1062–1067.

[14] G. Golub and C. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, second ed., 1989.

[15] S. Hammarling, *Newton's method for solving the algebraic Riccati equation*, NPL Report DITC 12/82, National Physical Laboratory, Teddington, Middlesex TW11 OLW, U.K., 1982.

[16] C. Kenney and G. Hewer, *The sensitivity of the algebraic and differential Riccati equations*, SIAM J. Cont. Optim., 28 (1990), pp. 50–69.

[17] C. Kenney, A. Laub, and M. Wette, *A stability-enhancing scaling procedure for Schur-Riccati solvers*, Sys. Control Lett., 12 (1989), pp. 241–250.

[18] ——, *Error bounds for Newton refinement of solutions to algebraic Riccati equations*, Math. Control, Signals, Sys., 3 (1990), pp. 211–224.

[19] D. Kleinman, *On an iterative technique for Riccati equation computations*, IEEE Trans. Automat. Control, AC-13 (1968), pp. 114–115.

[20] P. Lancaster and M. Tismenetsky, *The Theory of Matrices*, Academic Press, Orlando, 2nd ed., 1985.

[21] A. Laub, *A Schur method for solving algebraic Riccati equations*, IEEE Trans. Automat. Control, AC-24 (1979), pp. 913–921. (see also *Proc. 1978 CDC (Jan. 1979)*, pp. 60-65).

[22] F. Man, *The Davidon method of solution of the algebraic matrix Riccati equation*, Internat. J. Control, 10 (1969), pp. 713–719.

[23] *MATLAB*. The MathWorks, Inc., Cochituate Place, 24 Prime Park Way, Natick, Mass, 01760, 1984–94.

[24] V. Mehrmann, *The Autonomous Linear Quadratic Control Problem, Theory and Numerical Solution*, no. 163 in Lecture Notes in Control and Information Sciences, Springer-Verlag, Heidelberg, July 1991.

[25] S. Nash, *Private communication*, 1994.

[26] P. Pandey, *Quasi–Newton methods for solving algebraic Riccati equations*, in Proc. American Control Conf., Chicago, IL, June 1992, pp. 654–658.

[27] N. Sandell, *On Newton's method for Riccati equation solution*, IEEE Trans. Automat. Control, AC-19 (1974), pp. 254–255.

[28] V. Sima, *An efficient Schur method to solve the stabilization problem*, IEEE Trans. Automat. Control, AC-26 (1981), pp. 724–725.

[29] A. Varga, *On stabilization methods of descriptor systems*, Sys. Control Lett., 24 (1995), pp. 133–138.

[30] G. Verghese, B. Lévy, and T. Kailath, *A general state space for singular systems*, IEEE Trans. Automat. Control, AC-26 (1981), pp. 811–831.

[31] E. Yip and R. Sincovec, *Solvability, controllability and observability of continous descriptor systems*, IEEE Trans. Automat. Control, AC-26 (1981), pp. 702–707.

Other titles in the SPC series:

95_1 T. Apel, G. Lube. Anisotropic mesh refinement in stabilized Galerkin methods Januar 1995.

95_2 M. Meisel, A. Meyer. Implementierung eines parallelen vorkonditionierten Schur-Komplement CG-Verfahrens in das Programmpaket FEAP. Januar 1995.

95_3 S. V. Nepomnyaschikh. Optimal multilevel extension operators. January 1995

95_4 M. Meyer. Grafik-Ausgabe vom Parallelrechner für 3D-Gebiete. Januar 1995

95_5 T. Apel, G. Haase, A. Meyer, M. Pester. Parallel solution of finite element equation systems: efficient inter-processor communication. Februar 1995

95_6 U. Groh. Ein technologisches Konzept zur Erzeugung adaptiver hierarchischer Netze für FEM-Schemata. Mai 1995

95_7 M. Bollhöfer, C. He, V. Mehrmann. Modified block Jacobi preconditioners for the conjugate gradient method. Part I: The positive definit case. January 1995

95_8 P. Kunkel, V. Mehrmann, W. Rath, J. Weickert. GELDA: A Software Package for the Solution of General Linear Differential Algebraic Equation. February 1995

95_9 H. Matthes. A DD preconditioner for the clamped plate problem. February 1995

95_10 G. Kunert. Ein Residuenfehlerschätzer für anisotrope Tetraedernetze und Dreiecksnetze in der Finite-Elemente-Methode. März 1995

95_11 M. Bollhöfer. Algebraic Domain Decomposition. March 1995

95_12 B. Nkemzi. Partielle Fourierdekomposition für das lineare Elastizitätsproblem in rotationssymmetrischen Gebieten. März 1995

95_13 A. Meyer, D. Michael. Some remarks on the simulation of elasto-plastic problems on parallel computers. March 1995

95_14 B. Heinrich, S. Nicaise, B. Weber. Elliptic interface problems in axisymmetric domains. Part I: Singular functions of non-tensorial type. April 1995

95_15 B. Heinrich, B. Lang, B. Weber. Parallel computation of Fourier-finite-element approximations and some experiments. May 1995

95_16 W. Rath. Canonical forms for linear descriptor systems with variable coefficients. May 1995

95_17 C. He, A. J. Laub, V. Mehrmann. Placing plenty of poles is pretty preposterous. May 1995

95_18 J. J. Hench, C. He, V. Kučera, V. Mehrmann. Dampening controllers via a Riccati equation approach. May 1995

95_19 M. Meisel, A. Meyer. Kommunikationstechnologien beim parallelen vorkonditionierten Schur-Komplement CG-Verfahren. Juni 1995

95_20 G. Haase, T. Hommel, A. Meyer and M. Pester. Bibliotheken zur Entwicklung paralleler Algorithmen. Juni 1995.

95_21 A. Vogel. Solvers for Lamé equations with Poisson ratio near 0.5. June 1995.

95_22 P. Benner, A. J. Laub, V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations I: Continuous-time case. October 1995.

95_23 P. Benner, A. J. Laub, V. Mehrmann. A collection of benchmark examples for the numerical solution of algebraic Riccati equations II: Discrete-time case. to appear: December 1995.

95_24 P. Benner, R. Byers. Newton's method with exact line search for solving the algebraic Riccati equation. October 1995.

95_25 P. Kunkel, V. Mehrmann. Local and Global Invariants of Linear Differential-Algebraic Equations and their Relation. July 1995.

95_26 C. Israel. NETGEN69 - Ein hierarchischer paralleler Netzgenerator. August 1995.

95_27 M. Jung. Parallelization of multigrid methods based on domain decomposition ideas. September 1995.

95_28 P. Benner, H. Faßbender. A restarted symplectic Lanczos method for the Hamiltonian eigenvalue problem. October 1995.

95_29 G. Windisch. Exact discretizations of two-point boundary value problems. October 1995.

Some papers can be accessed via anonymous ftp from server `ftp.tu-chemnitz.de`, directory `pub/Local/mathematik/SPC`. (Note the capital L in Local!)
The complete list of current and former preprints is available via
`http://www.tu-chemnitz.de/~pester/sfb/spc95pr.html`.