Continuation of Invariant Subspaces

L. Dieci[*] and M. J. Friedman
School of Mathematics, Georgia Inst. of Technology, Atlanta, GA, 30332-0160.
Department of Math. Sciences, Univ. Alabama in Huntsville, Huntsville, AL, 35899.

February 19, 2001

Abstract

In this work we consider implementation and testing of an algorithm for continuation of invariant subspaces.

1 Introduction.

In [5], we presented and analyzed a new algorithm for continuation of invariant subspaces of a parameter dependent matrix A, in the context of continuation of connecting orbits. In such context, the algorithm revealed to be both robust and efficient. Thus, we decided to carry further the study and implementation of this technique on its own right. The purpose of the present paper is to report on the results of this study. First we review the basic problem and describe the algorithm, then address specific implementation issues and present results of numerical experiments for several functions A.

The problem.

The basic problem is the one of computing a smooth orthogonal similarity transformation to block triangular form of a matrix valued real function. We have the following setup.

(i) Given a smooth ($C^k,\ k \geq 1$) matrix valued function $A : t \in [0,1] \to \mathbb{R}^{n\times n}$; write this as $A \in C^k([0,1],\ \mathbb{R}^{n\times n})$. Assume that $\Lambda(A)$ (the set of eigenvalues of A) admits the splitting $\Lambda(A) = \Lambda_1 \cup \Lambda_2$ with $\Lambda_1 \cap \Lambda_2 = \emptyset,\ \forall t$.

(ii) Let orthogonal Q_0 be such that

$$Q_0^T A(0) Q_0 =: R_0 = \begin{pmatrix} R_{11}(0) & R_{12}(0) \\ 0 & R_{22}(0) \end{pmatrix}, \quad R_{11}(0) \in \mathbb{R}^{m\times m},\ R_{22}(0) \in \mathbb{R}^{(n-m)\times(n-m)},$$

and $\Lambda(R_{11}(0)) = \Lambda_1(0)$, $\Lambda(R_{22}(0)) = \Lambda_2(0),$.

(iii) For all $t \in [0,1]$, want a C^k orthogonal $Q(t)$,

$$Q^T(t) A(t) Q(t) =: R(t) = \begin{pmatrix} R_{11}(t) & R_{12}(t) \\ 0 & R_{22}(t) \end{pmatrix}, \quad R_{11}(t) \in \mathbb{R}^{m\times m},\ R_{22}(t) \in \mathbb{R}^{(n-m)\times(n-m)},$$

such that $Q(0) = Q_0$, $R(0) = R_0$, and the eigenvalues of the block upper triangular matrix R satisfy $\Lambda(R_{11}) = \Lambda_1$ and $\Lambda(R_{22}) = \Lambda_2,\ \forall t$. (Notice: no assumptions are made on R_{ii} being triangular).

The CIS (*Continuation Invariant Subspace*) algorithm generates a sequence of points $(t_i, Q_i)|_{i=0,N}$, $\{0 = t_0 < t_1 < \ldots < t_{N-1} < t_N = 1\}$, where N is determined adaptively, on the curve $(t, Q(t))$.

The problem of computing a smooth path of orthogonal matrix factorizations has a long history. The prototype of many works on the subject has been a paper of Rheinboldt, [14]. He was concerned with

[*]Supported in part under NSF DMS-9973266.

a smooth QR factorization of A, and exploited the idea of "least variation" with respect to a reference factorization in order to enforce smoothness. The same idea was then used by Bunse-Gerstner et al. in [3] in the context of analytic SVD. In essence, the idea is the following: given a factorization for $A(0)$, and given a stepsize h, compute some factorization of $A(h)$, and then modify this factorization by minimizing the distance of the factors relative to $A(h)$ from those of $A(0)$; solving the minimization problem requires solving an orthogonal Procrustes problem. The advantage of these techniques is that one ends up with an exact factorization for $A(h)$, but the disadvantage is that there is no clear way on how to choose h. To compensate for this fact, already in [3], and more extensively in work by Wright, [16], and Mehrmann & Rath, [13], people begun looking at the underlying differential equations governing the evolution of the SVD factors, and numerically integrated these factors in order to compute the path. The obvious advantage of these latter approaches is that the step-size h (and hence where to compute the factorizations) is chosen adaptively, based upon the variation of the factors themselves. The disadvantage is that one no longer has an exact factorization as the integration proceeds. Some comparisons of the relative merits of the "least variation" vs. "differential equations" approach are in [13].

Our method tries to unify the best features of both approaches above. In the context of smooth block Schur factorization, we will use the underlying differential equations in order to obtain initial approximations of the exact factors; a Newton iteration will then be used to refine these approximations. Convergence behavior of the Newton iterates will be used to adaptively choose the stepsizes. This way, computation of the Q_i's will be the determining factor in adaptively choosing the points t_i.

The problem of computing invariant subspaces of parameter dependent matrix value functions arises quite frequently in applications. Our own interest stems from applications in dynamical systems. In one such case, continuation of invariant subspaces is a key step in techniques to compute connecting orbits, and thus it ultimately impacts our understanding of complicated dynamics (see [5]); in another case, an adaptation of the CIS algorithm to an improved detection of bifurcations is considered (see [9]). However, the basic computational task is much more pervasive, and one may legitimately argue that computation and continuation of invariant subspaces lie at the very essence of eigenvalue computation for a parameter dependent matrix valued function.

The following theorem justifies the existence of a C^k orthogonal function $Q(t)$, $t \in [0,1]$, achieving the block triangularization of A of which in (iii) above. This result is well known, and can be traced back to the work of Beyn, [2]. But, to understand how our algorithm works, we find it convenient to give the result in the flavor of [6]. Hereafter, a "dot" refers to differentiation with respect to t.

Theorem 1 *Let $A \in C^k([0,1], \mathbb{R}^{n\times n})$, $k \geq 1$, and orthogonal Q_0 be given satisfying* (i)-(ii) *above. Then, there exist R and Q, both in $C^k([0,1],\ \mathbb{R}^{n\times n})$, Q orthogonal, satisfying* (iii)*. The matrix Q can be written as $Q(t) = Q(\tau)U(t,\tau)$, for any t: $0 \leq t \leq 1$, and fixed τ: $\tau \leq t$, with $Q(0) = Q_0$ and $U(\tau,\tau) = I$. Further, $R,\ U \in C^k([0,1],\ \mathbb{R}^{n\times n})$ satisfy the differential system*

$$\begin{aligned} \dot{R} &= U^T(t,\tau)\ [Q^T(\tau)\dot{A}(t)Q(\tau)]\ U(t,\tau) + R(t)H(t) - H(t)R(t)\,, & (1)\\ \dot{U}(t,\tau) &= U(t,\tau)H(U,t)\,, & (2)\end{aligned}$$

where for all t (with obvious block notation)

$$H = \begin{pmatrix} H_{11} & H_{12} \\ -H_{12}^T & H_{22}\end{pmatrix}\,, \quad H_{12}:\ \ R_{22}H_{12}^T - H_{12}^T R_{11} = \{U^T(\cdot,\tau)[Q^T(\tau)\dot{A}(\cdot)Q(\tau)]U(\cdot,\tau)\}_{21}\,. \qquad (3)$$

The matrices H_{11} and H_{22} must be skew-symmetric and C^{k-1}, but are otherwise arbitrary.

Proof. We want $Q^T(t)A(t)Q(t) = R(t)$, $Q^T(t)Q(t) = I$, and R block triangular, for all t. To achieve this, we derive differential equations for Q and R by differentiating $Q^T(t)A(t)Q(t) = R(t)$ and $Q^T(t)Q(t) = I$. By formally letting $H = Q^T\dot{Q}$, we have

$$\begin{aligned} \dot{R}A &= Q^T\dot{A}Q + RH - HR \\ \dot{Q} &= QH\,. \end{aligned}$$

Now, partition H as above, and notice that since R must be block upper triangular, we then must have

$$0 = (Q^T\dot{A}Q)_{21} - R_{22}H_{12}^T + H_{12}^T R_{11}\,,$$

which is uniquely solvable for H_{12} since R_{11} and R_{22} do not have common eigenvalues. Finally, we can fix H_{11} and H_{22} skew-symmetric and sufficiently smooth (but otherwise arbitrarily). So doing, we obtain H which is Lipschitz in Q and continuous in t. Thus, the above coupled system of differential equations is uniquely solvable for Q and R as desired. Finally, the rewriting $Q(t) = Q(\tau)U(t,\tau)$ is immediate. ■

Clearly, the matrices Q and R are not unique (the H_{ii}'s are skew symmetric, but otherwise arbitrary). This freedoom can be used in a number of different ways. We will use it to impose a particular structure for Q in the same way as Stewart in [15] and Demmel in [4] did to refine computed invariant subspaces; this way, the CIS algorithm can be viewed as a continuous analog of these standard linear algebra techniques. In the next section, we outline the complete process and discuss implementation aspects.

2 The CIS algorithm.

- Given Q_0, $Q_0^T Q_0 = I$, such that

$$Q_0^T A(0) Q_0 =: R(0) = \begin{bmatrix} R_{11}(0) & R_{12}(0) \\ 0 & R_{22}(0) \end{bmatrix}, \quad \Lambda_1(0) \cap \Lambda_2(0) = \emptyset, \tag{4}$$

  where $\Lambda_1(0) = \Lambda(R_{11}(0))$, $\Lambda_2(0) = \Lambda(R_{22}(0))$, and $\Lambda(A(0)) = \Lambda(R_{11}(0)) \cup \Lambda(R_{22}(0))$ is the set of eigenvalues of $A(0)$.

- Given h, $0 < h < 1$ (see below how we choose h), we want to find Q, $Q^T Q = I$, which puts $A(h)$ in the block triangular form:

$$Q^T A(h) Q =: R(h) = \begin{bmatrix} R_{11}(h) & R_{12}(h) \\ 0 & R_{22}(h) \end{bmatrix}, \quad \Lambda_1(h) \cap \Lambda_2(h) = \emptyset, \tag{5}$$

  where $\Lambda_1(h) =: \Lambda(R_{11}(h))$, $\Lambda_2(h) =: \Lambda(R_{22}(h))$, and the blocks $R_{11}(h)$ and $R_{22}(h)$ are the values at h of $R_{11}(t)$ and $R_{22}(t)$, respectively.

To take advantage of knowing Q_0 in order to determine Q, upon making use of Theorem 1, we look for Q in the form

$$Q = Q_0 U(h,0) \ . \tag{6}$$

The issue is how to obtain $U(h,0)$, which –for short– we will just call U. Partition $U =: [U_1 \ U_2]$ according to the dimensions of the blocks in (4). Now, assume that h is sufficiently small, so that if U_1 is partitioned as $U_1 =: \begin{bmatrix} U_{11} \\ U_{21} \end{bmatrix}$, then U_{11} is invertible. Thus

$$U_1 = \begin{bmatrix} I \\ Y \end{bmatrix} U_{11} \,, \qquad \text{where} \qquad Y := U_{21} U_{11}^{-1} \,. \tag{7}$$

Since $U_1^T U_1 = I$, we must have

$$U_{11}^T \ (I + Y^T Y) \ U_{11} \ = \ I \ , \quad \text{or} \quad U_{11} U_{11}^T = (I + Y^T Y)^{-1} \,. \tag{8}$$

In other words, U_{11} must be a "square root" of the matrix $(I + Y^T Y)^{-1}$. We will want to find Y so that U_1 spans the invariant subspace relative to $\Lambda_1(h)$. To find Y, we further restrict consideration to a special solution U_{11} of (8). Recall that we expect, for h small, to have U_{11} close to the identity. The following lemma is known in a number of equivalent formulations.

Lemma 2 *Let $B \in \mathbb{R}^{m\times m}$ be a positive definite matrix. Then, the minimization problem*

$$\min \|I - BQ\|_F \,, \ \ \textit{subject to} \ \ Q \in \mathbb{R}^{m\times m} \ : \ Q^T Q = I \,,$$

has the unique solution $Q = I$. Here, $\|\cdot\|_F$ is the Frobenius norm ($\|B\|_F^2 = \sum_{i,j} B_{ij}^2$).

Proof. This is a version of the orthogonal Procrustes problem (see [11, p. 582]). Let orthogonal U be such that $U^T B U = \Lambda$, where Λ is the diagonal matrix of (positive) eigenvalues of B, and let $Z = UQU^T$. We want to minimize the trace of $(I - \Lambda Z)^T (I - \Lambda Z)$, or –which is the same– to maximize the trace of ΛZ. But the latter is simply $\sum_{i=1}^{n} \lambda_i z_{ii}$, and since Z is orthogonal, the maximum is attained for $z_{ii} = 1$, that is $Q = I$. ■

The next lemma links our approach and the "least variation idea".

Lemma 3 *In the Frobenius norm, the closest solution to the identity of (8) is the unique positive definite square root of* $(I + Y^T Y)^{-1}$*:*

$$U_{11} \;=\; (I + Y^T Y)^{-1/2}\,. \tag{9}$$

Proof. The result follows from Lemma 2 upon realizing that any solution of (8) is of the form $\tilde{U}_{11} = U_{11} C$ with $C^T C = I$ and U_{11} given in (9). ■

In a similar way to the above, and making use of the orthogonality requirement $U^T U = I$, one eventually gets the following form for U:

$$U = \left[\begin{pmatrix} I \\ Y \end{pmatrix} (I + Y^T Y)^{-1/2}, \begin{pmatrix} -Y^T \\ I \end{pmatrix} (I + Y Y^T)^{-1/2} \right]. \tag{10}$$

The matrix $Y \in \mathbb{R}^{(n-m)\times m}$ must be found by requiring that $U^T (Q_0^T A(h) Q_0) U$ has the form specified in (5). Thus, if we let

$$Q_0^T A(h) Q_0 = \begin{bmatrix} \widehat{R}_{11} & \widehat{R}_{12} \\ E_{21} & \widehat{R}_{22} \end{bmatrix}, \tag{11}$$

then Y must solve the algebraic Riccati equation

$$F(Y) = 0\,, \quad F(Y) := \widehat{R}_{22} Y - Y \widehat{R}_{11} + E_{21} - Y \widehat{R}_{12} Y \;\; . \tag{12}$$

Remark. As we said, the representation for U as in (10) was first used in [15], and later in [4]. But, as far as we know, the justification for the form (10) as an exact solution of the differential equation (2), via Lemma 2 and 3, is original. In fact, our contribution to continuation of invariant subspaces is to have tied together the representation (10) with Theorem 1. By so doing, we will be able to let continuation of the factors U influence choice of the stepsize h. Moreover, we will be able to provide second order initial guesses for solving (12), at essentially no added expense with respect to existing approaches. Use of this Euler (or tangent) predictor will bring the computational task of continuation of invariant subspaces on equal turf with standard continuation techniques (e.g., see [12] or [7]).

Once the structure (10) for U has been fixed, one has to solve the Riccati equation (12). There are a host of possibilities for solving (12), but in the present context we expect Y to be reasonably small, for small h, and hence we can focus on the following two techniques.

For Y_0 given, and for $k = 1, 2, ..., K_{\max}$, consider

- Simple iteration
$$\widehat{R}_{22} \Delta_k - \Delta_k \widehat{R}_{11} = -F(Y_{k-1}),\;\; Y_k = \Delta_k + Y_{k-1},\;\; \text{or} \tag{13}$$
- Newton iteration
$$(\widehat{R}_{22} - Y_{k-1} \widehat{R}_{12}) \Delta_k - \Delta_k (\widehat{R}_{11} + \widehat{R}_{12} Y_{k-1}) = -F(Y_{k-1}),\;\; Y_k = \Delta_k + Y_{k-1}. \tag{14}$$

Both iterations (13) and (14) have been extensively used before; e.g., see [15] and [4]. In these works, the rationale that Y ought to be small for small h, reflected in the choice $Y_0 = 0$ as initial guess for the above iterations. In the continuation literature, this is the so-called *trivial predictor*. It is usually not a satisfactory choice, and in continuation works people have routinely adopted the so-called *tangent* or *Euler* predictor to provide the initial guess for the iterative process (the so-called *corrector*); see [1]. In [5], we showed how to derive such tangent predictor at minimal extra cost. To do this, it is mandatory to exploit the connection between the differential equations of Theorem 1 and the Riccati equation (12). We summarize the relevant facts in a lemma.

Lemma 4 *In the block form of H in (3), let H_{11} and H_{22} be any two sufficiently smooth skew-symmetric matrices, while H_{12} is determined as in (3). Let $U = [U_1\, U_2]$ be the solution at h of (2), i.e., $U = U(h, 0)$, and let Y be the solution of (12). Then, we have*

$$Y \;=\; U_{21}U_{11}^{-1}\,, \quad \textit{where} \quad U_1 = \begin{pmatrix} U_{11} \\ U_{21} \end{pmatrix}. \tag{15}$$

Moreover, the solution Y of (12) is unaffected by the choice made for H_{11} and H_{22}.

Proof. The only part we need to justify is that it does not matter how we select H_{11} and H_{22}. But this is a consequence of the fact that the possible solutions U_1 are of the type $U_1 = \tilde{U}_1 C$, where C is orthogonal and $\tilde{U}_1$ is what one obtains with a fixed choice of H_{11} and H_{22}; for example, setting them both to 0. ■

Now we let $V_1 = \begin{pmatrix} V_{11} \\ V_{21} \end{pmatrix}$ be the approximation to U_1 obtained by taking a step of Forward Euler method on (2) with H_{11} and H_{22} both 0 (of appropriate dimensions). Then, we obtain an initial guess for the solution of (12) by letting $Y_0 = V_{21}V_{11}^{-1}$. That is:

$$\begin{pmatrix} V_{11} \\ V_{21} \end{pmatrix} = \begin{pmatrix} I \\ 0 \end{pmatrix} + h \begin{pmatrix} 0 \\ -H_{12}^T(0) \end{pmatrix}, \qquad \text{and} \qquad Y_0 = -hH_{12}^T(0)\,.$$

Since $H_{12}^T(0)$ solves (see (3))

$$R_{22}(0)H_{12}^T(0) - H_{12}^T(0)R_{11}(0) \;=\; \left(Q_0^T \dot{A}(0) Q_0\right)_{12},$$

we replace $\dot{A}(0)$ by the difference quotient

$$\dot{A}(0) \approx \frac{A(h) - A(0)}{h} \Rightarrow \left(Q_0^T \dot{A}(0) Q_0\right)_{12} \;\approx\; \frac{1}{h} E_{21}\,,$$

and eventually solve

$$Y_0 \;:\; R_{22}(0)Y_0 - Y_0 R_{11}(0) \;=\; -E_{21}\,. \tag{16}$$

This value of Y_0 is a second order approximation to Y, and can be used as initial guess for solving (12).

We are ready to summarize the CIS algorithm.

Step 0. **Initialization**. Compute Q_0 in (4), and also select stepsize $h < 1$.

- The very first time, we will have $t_0 = 0$, and $A(0)$ may well be in real Schur form. Also, in our experiment we took $h = 10^{-3}$, since the adaptive criterion quickly adjusts h.

Step 1. **Solving the Riccati equation (12)**.

1. **Compute the coefficients** $\widehat{R}_{22}$, $\widehat{R}_{11}$, E_{21}, and $\widehat{R}_{12}$ in (12) using equation (11).
2. **Prediction**. Let Y_0 be the initial guess. Comparisons will be made by choosing the following options for Y_0 (see (16))

$$Y_0 \;=\; \begin{cases} 0\,, & \text{trivial guess}\,; \\ R_{22}(0)Y_0 - Y_0 R_{11}(0) = -E_{21}\,, & \text{Euler guess}\,. \end{cases} \tag{17}$$

   - Using the Euler guess requires solving a Sylvester equation to get Y_0.
3. **Corrector**. For $k = 1, 2, ..., K_{\max}$ to find Y use
   (a) Simple iteration (13), or
   (b) Newton iteration (14).
   - With either methods we need to solve Sylvester equations at each step.
   - In our experiments, we used $K_{\max} = 7$.

4. **Stopping criterion**. If

$$\frac{\|Y_{k+1}-Y_k\|}{1+\|Y_{k+1}\|} \leq \epsilon, \tag{18}$$

is satisfied, set $Y := Y_{k+1}$. In the examples below, unless otherwise noted, we used $\epsilon = 10^{-8} \approx \sqrt{\texttt{EPS}}$, where `EPS` is the machine precision.

Step 2. **Update Q**. Compute $U(h)$ in (10), and hence Q in (6).

Step 3. **Step size control.** Update (restrict/enlarge) stepsize h. We used (see [12, p. 138])

$$h := 2^{(4-k)/3}\ h,$$

where k is the number of iterations performed. Update $t_0 := t_0 + h$, $Q_0 := Q$, and go to Step 1.

- Upon updating h, we ensure that we remain within the specified interval. Also, we require that $h \geq h_{\min}$; in the examples, unless otherwise noted, we used $h \geq h_{\min} = 10^{-8}$. If the predicted value of h falls below $h_{\min}$, then we stop.

Remarks.

- Some of the criteria above, such as those on stepsize control, are based on experience gained during the years on problems of continuation type; e.g., see the choices adopted in the code `AUTO` ([7]).

- We included comparisons with the linearly convergent iteration (13) for completeness. But, in continuation techniques, the standard choice is Newton's method.

- We need to repeatedly solve Sylvester equations. For example, this is needed to get Y_0 in (16), as well as at each iterate of (13) or (14). Clearly, when using the iteration (13) –and unlike the case of Newton's method– one needs only one Schur factorization of the blocks in the Sylvester equation at each step. We solved the Sylvester equation in the textbook way (see [11]):

$$AX + XB = C$$

is transformed to a logically triangular system via Schur reduction of A and B and the logically triangular system is then solved. We opted for this strategy, rather than the less expensive one of [10], because for the application we have in mind we eventually need the eigenvalues of the triangularized system (e.g., for monitoring bifurcations). Notice that if one uses Newton's method and the initial guess (16), then to get Y_0 from the latter does not require Schur factorization of the matrices arising in the Sylvester equation, since these are the same we have already triangularized at convergence of the previous step.

- The finite precision behavior of most components of our algorithm is quite standard: Schur reduction, solution of Sylvester equation, etc., have all been extensively analyzed (e.g., see [11]). The one component which is not routinely analyzed is the computation of the matrices $(I + Y^TY)^{-1/2}$ and $(I + YY^T)^{-1/2}$ in (10). The subtlety here is that, even when Y is fully accurate, a naive implementation which explicitely forms Y^TY may introduce numerical instability and unneeded loss of precision, particularly when Y has entries of widely different order of magnitude. To avoid this from happening, we have adopted the following algorithm: (i) form the SVD of Y: $Y = UDV^T$, from which we get $(I + Y^TY)^{-1/2} = V(I + D^TD)^{-1/2}V^T$ and $(I + YY^T)^{-1/2} = U(I + DD^T)^{-1/2}U^T$; (ii) to reduce cancellation when finding $(I + D^TD)^{-1/2}$, we first compute in the obvious way the quantity $x = \sqrt{1 + d^2}$ and then refine these values with a Newton step on the system

$$yz = 1\ ,\ \ z - y = 2d\ ,\ \ \text{where}\ \ y = x - d\ ,\ z = x + d\,.$$

We compared this algorithm to the "obvious" one: (a) form $(I + Y^TY)$, find its Schur decomposition, and compute the square root, and (b) similarly for $(I + YY^T)^{1/2}$. On the limited comparison pool provided by the examples of the next section, our prefereed algorithm (i)-(ii) was superior in efficiency and reliability to the "obvious" one (a)-(b).

3 Numerical Experiments.

All computations below are done with a `Fortran77` program (see [8]) for the continuation of invariant subspaces of a parameter dependent matrix A.

- In our experiments, we want to look at the following features:

  (1) convergence behavior on "easy problems": these are the ones with well separated blocks and slowly varying eigenvalues and subspaces;

  (2) number of rejections as a function of different initial guesses on "harder problem": weakly separated blocks and/or rapidly varying eigenvalues or subspaces;

  (3) how we approach bifurcation points: blocks coalesce.

- We build our example by giving a block triangular matrix $R(t)$, and then transforming it with an orthogonal matrix $V(t)$ to obtain $A(t) = V^T(t)R(t)V(t)$.

  - *Construction of the orthogonal matrix $V(t)$.* We specify the orthogonal V as $V(t) = e^{S(t)}$ where S is a skew-symmetric function. We have taken $S(t)$ to have upper triangular entries given by

  $$S_{ij}(t) = (-1)^{i+j}\frac{t-1}{j+1}t^{j-i}, \quad 1 \le i < j \le n.$$

  This requires computation of the matrix exponential, which we have done using diagonal Padé approximation along with scaling and squaring, as outlined in [11, Section 11.3].

  - *Construction of the block triangular matrix $R(t)$.* This construction is based on one in [10]. Take $R(t) = \begin{pmatrix} D & C \\ 0 & -B \end{pmatrix}$, where $D \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{p\times p}$, and $C \in \mathbb{R}^{m\times p}$. We let

  $$D = f(t)[\mathrm{diag}(1, 2, ..., m) + L_m], \quad B = g(t)[\mathrm{diag}(p, p-1, ..., 1) + L_p^T + \alpha^t I_p],$$

  and $C = DX + XB$, where $X \in \mathbb{R}^{m\times p}$ is made up by all 1's. The invariant subspace decomposition we are after is the one with the dimensions m and p and eigenvalues blocked as in R. In the above, the scalar functions $f(t)$ and $g(t)$ are arbitrary, and the matrix L_k is the $k \times k$ lower triangular matrix made up by all 1's.

Example 1 *Moderately fast rotating subspaces, rapidly increasing entries and eigenvalues from a modest size in the range* $[-2, 4]$ *to the range* $[-124, 4]$. *Here* $f(t) = g(t) = 1$, $n = 8$, $m = 4$, $\alpha = 5$, $1 < t < 3$.

	number of continuation steps	number of iterations	time
(13), Euler guess for Y_0	$1,401$	$5,603$	86.7
(14), trivial guess for Y_0	$8,217$	$46,942$	670
(14), Euler guess for Y_0	736	$2,926$	51.0

Here simple iteration with the trivial initial guess $Y_0 = 0$ is extremely slow.

Example 2 *Slowly rotating subspaces, modest size eigenvalues in the range* $[-2, 4]$*, slowly varying. Here* $f(t) = g(t) = 1$, $n = 8$, $m = 4$, $\alpha = 5$*; t decreases from* 1 *until the subspaces coalesce at* $t \approx 0.68261$*, reaching the double eigenvalue* 1.

	number of continuation steps	number of iterations	time
(13), trivial guess for Y	22	80	1.75
(13), Euler guess for Y_0	21	78	1.91
(14), trivial guess for Y_0	15	43	1.06
(14), Euler guess for Y_0	15	35	1.38

Example 3 *Fast rotating subspaces, modest size eigenvalues, in the range* $[-2, 4]$*, slowly changing. Here* $f(t) = g(t) = 1$*,* $n = 8$*,* $m = 4$*,* $\alpha = 5$*;* t^{j-i} *is replaced by* $(t+3)^{j-i}$ *in the definition of* $S_{ij}(t)$ *to make the subspaces rotate fast, as* t *decreases from* 1 *until the subspaces coalesce at* $t \approx 0.68261$*, reaching the double eigenvalue* 1*.*

	number of continuation steps	number of iterations	time
(13), Euler guess for Y_0	$26,033$	$139,134$	$1,840$
(14), trivial guess for Y_0	$2,178$	$9,795$	113
(14), Euler guess for Y_0	657	$3,239$	40.6

Note that in the first case the continuation breaks when the coalescing eigenvalues are 1 and 0.9999317, while in the third case it breaks when they are 1 and 0.9999999, while sep $= 1.4 \times 10^{-9}$. For the meaning of sep and its role in the convergence behavior of (13) and (14), we refer to [5, (14)-(25)].

Example 4 *Rapidly increasing matrix entries and eigenvalues, with the approximate range* $[-295, 4]$*. Here* $\epsilon = h_{\min} = 10^{-7}$*,* $f(t) = g(t) = 1$*,* $n = 64$*,* $m = 4$*,* $\alpha = 61$*,* $1.3 < t < 1.303$*.*

	number of continuation steps	number of iterations	time
(14), trivial guess for Y_0	642	$3,205$	$7,146$
(14), Euler guess for Y_0	4	14	63.3

Here, the simple iteration is extremely slow.

Example 5 *Rapidly increasing matrix entries and the condition number increasing from* 4 *to* 10^8*. Here* $h_{\min} = 10^{-5}$*,* $f(t) = 10^{-t}$*,* $g(t) = 10^{t}$*,* $n = 8$*,* $m = 4$*,* $\alpha = 5$*,* $1 < t < 3$*.*

	number of continuation steps	number of iterations	time
(14), trivial guess for Y_0	$8,278$	$47,294$	593
(14), Euler guess for Y_0	635	$2,887$	39.2

Here, the simple iteration is extremely slow.

4 Conclusions.

1. Newton iteration with either the trivial or Euler initial guesses for Y_0 always takes smaller number of continuation steps, total number of iterations and time than simple iteration with the trivial/Euler initial guesses for Y_0.

2. Newton iteration with Euler initial guess for Y_0 always takes smaller number of continuation steps and total number of iterations than with the trivial initial guess for Y_0.

3. When the matrix entries (eigenvalues) do not grow fast and the subspaces are not rotated fast Newton iteration with trivial initial guess for Y_0 is comparable or even takes slightly less time than Newton iteration with Euler initial guess.

4. Newton iteration with Euler initial guess for Y_0 takes considerably (about 3 times when the matrix is small and over 100 times when the matrix is of moderate size) fewer continuation steps, total number of iterations, and shorter time, when the matrix entries (eigenvalues) grow fast and/or the subspaces are rotated fast; and about 10 times less in the case of an ill conditioned matrix.

5. Approaching a bifurcation point by itself does not seem to be a critical factor (see Example 2).

References

[1] E. Allgower and K. Georg, Continuation and path following, *Acta Numerica*, (1993), 1–64.

[2] W. J. Beyn, The numerical computation of connecting orbits in dynamical systems, *IMA J. Numer. Analysis*, **10** (1990), 379–405.

[3] A. Bunse-Gerstner, R. Byers, V. Mehrmann, and N. K. Nichols, Numerical computation of an analytic singular value decomposition by a matrix valued function, *Numer. Math.*, **60** (1991), 1–40.

[4] J. Demmel, Three methods for refining estimates of invariant subspaces, *Computing*, **38** (1987), 43–57.

[5] J. Demmel, L. Dieci, and M. Friedman, Computing connecting orbits via an improved algorithm for continuing invariant subspaces, *SIAM J. Scientific Computing* **22**, No. 1 (2001), 81-94.

[6] L. Dieci and T. Eirola, On Smooth Decomposition of Matrices, *SIAM J. Matrix Analysis & Applications*, **20** (1999), 800-819.

[7] E.J. Doedel, A.R. Champneys, T.F. Fairgrieve, Yu.A. Kuznetsov, B. Sandstede, and X.J. Wang, *AUTO97: Continuation and bifurcation software for ordinary differential equations (with HomCont)* (1997).

[8] M.J. Friedman, *SUBCON, a collection of subroutines for continuing invariant subspaces of a parameter dependent matrix, an experimental version*. University of Alabama in Huntsville, 1998.

[9] M.J. Friedman, An improved detection of bifurcations in large nonlinear systems via the Continuation of Invariant Subspaces algorithm, to appear in *Int. J. Bifur. & Chaos* (2000).

[10] G. H. Golub, S. Nash and C. Van Loan, A Hessenberg-Schur method for the problem $AX + XB = C$, *IEEE Trans. Auto. Control*, **24** (1979), 909-913.

[11] G.H. Golub and C.F. Van Loan, *Matrix Computations*, The John Hopkins University Press (1989).

[12] H. Keller, *Numerical Methods in Bifurcation Problems*, Springer Verlag, Tata Institute of Fundamental Research, Bombay (1987).

[13] V. Mehrmann and W. Rath, Numerical methods for the computation of analytic singular value decompositions, *Electronic Trans. Numerical Analysis*, **1** (1993), 72–88.

[14] W. Rheinboldt, On the computation of multi-dimensional solution manifolds of parametrized equations, *Numer. Math.*, **53** (1988), 165–181.

[15] G. W. Stewart. Error and perturbation bounds for subspaces associated with certain eigenvalue problems, *SIAM Review* **15**, No. 4 (1973), 727–764.

[16] K. Wright, Differential equations for the analytical singular value decomposition of a matrix, *Numer. Math.*, **63** (1992), 283.