Computational aspects of the Jordan canonical form

(Running title : Computing the Jordan Form)

Abstract

In this paper we discuss algorithmic aspects of the computation of the Jordan canonical form. Inspired by the Golub & Wilkinson paper [9] on the computation of the Jordan canonical form, an $O(n^3)$ algorithm was developed by Beelen & Van Dooren [3] for computing the Kronecker structure of an arbitrary pencil $\lambda B - A$. Here we show how the ideas of this algorithm lead to a special algorithm for reconstructing the Jordan structure of the standard eigenvalue problem $\lambda I - A$.

Theo Beelen
PICOS Glass, Philips Eindhoven, NL-2600 Eindhoven, The Netherlands

Paul Van Dooren
Philips Research Laboratory Brussels, B-1170 Brussels, Belgium

1 Introduction

The computation of eigenvalues of a general $n \times n$ matrix A and of the generalized eigenvalues of an $n \times n$ regular pencil $\lambda B - A$ can be solved in a rather satisfactory manner using the QR and QZ algorithms, respectively. Both algorithms are indeed backward stable and have a complexity which is cubic in the dimension of the matrices. The problem is quite different when one also wants to know the fine structure of the eigenvalues, i.e. the Jordan structure of A or the Kronecker structure of $\lambda B - A$. It is well-known that the problem of numerically determining the Jordan structure or the Jordan canonical form of a matrix A is ill-posed in the sense that a small perturbation of the matrix A may drastically change the computed Jordan structure or form. The same can of course be said about the generalized eigenvalue problem $\lambda B - A$ and the computation of the Kronecker structure or canonical form. Once the eigenvalues or generalized eigenvalues have been computed, one has in fact to solve two problems before providing a satisfactory answer to the *numerical* computation of the above canonical forms :

(i) *clustering eigenvalues* : find out which computed eigenvalues correspond to *perturbed* multiple eigenvalues

(ii) *computing the structure* : given a *restored* multiple eigenvalue, find out what its Jordan or Kronecker structure is.

Both the algorithmic aspects of these two problems (i.e. issues as complexity and stability [14] [15] [9] [17] [10] [11] [2]) and their theoretical aspects (i.e. issues as sensitivity and robustness [13] [5] [20] [21] [6] [7]) have extensively been treated in the literature during the last 20 years.

In this paper we focus on the second problem and more specifically on the reconstruction of the Jordan structure of a matrix A at a given eigenvalue. This problem is known to be equivalent to a recursive rank search [14], [15], [17], during which the Jordan structure of the corresponding eigenvalue is being reconstructed throughout the recursion. It turns out that in the worst case one has to perform $(n-1)$ rank decompositions of matrices of decreasing dimension $(n-i)$, $i = 1, \ldots, n-1$, hence leading to an overall $O(n^4)$ algorithm.

In their benchmark paper [9], Golub & Wilkinson give an approach to perform this recursion for the standard eigenvalue problem in $O(n^3)$ operations, by adroitely exploiting the decompositions of the previous steps during the recursion. Recently, Beelen & Van Dooren [3] extended these results to arbitrary pencils of matrices, thereby yielding a cubic algorithm for computing the Kronecker structure of an arbitrary pencil. Although this method was strongly inspired from the work of Golub & Wilkinson, it uses a different technique of exploiting the previous decompositions of the recursive algorithm.

Here we show that this new technique can also be efficiently applied to the standard eigenvalue problem and in fact leads to a backward stable algorithm for reconstructing the Jordan structure of a matrix A in $O(n^3)$ operations. A short comparison is also made with the Golub & Wilkinson method and some remaining open problems are briefly mentioned.

2 Jordan structure of a matrix A

Without loss of generality, one may consider the case of reconstructing the Jordan structure of the (real or complex) matrix A at the eigenvalue $\lambda = 0$. From earlier work ([14], [15], [9]) it is known that implicitly one has to compute the null space $\mathcal{N}_i$ of the powers A^i of the matrix A. These null spaces are clearly nested and one computes them until for some index k their dimension $n_i \doteq dim(\mathcal{N}_i)$ does not change anymore :

$$\begin{aligned} \mathcal{N}_1 \subset \mathcal{N}_2 \subset \ldots \subset \mathcal{N}_k = \mathcal{N}_{k+1} \\ n_1 < n_2 < \ldots n_k = n_{k+1} \end{aligned} \tag{1}$$

From these spaces one then retrieves all Jordan blocks of A at the eigenvalue $\lambda = 0$. At first sight one thus needs to construct the powers of A and compute their null spaces via some decomposition (say the QRD or SVD) from which both the range and null spaces can be derived. A first improvement on this, due to Kublanovskaya and Ruhe is to note that one can find these null spaces by directly

working on A and hence avoid the construction of the powers A^i. Indeed, let U be a unitary matrix partitioned as follows :

$$U = [U_1 \mid U_2 \mid \ldots \mid U_k \mid U_{k+1}], \tag{2}$$

with U_i an $n \times d_i$ matrix for $i = 1, \ldots, k$, such that

$$\begin{aligned} \mathcal{N}_1 &= Span([U_1]), \\ \mathcal{N}_i = Span([U_1 \mid \ldots \mid U_i]) &= \mathcal{N}_{i-1} \overset{\perp}{\oplus} Span([U_i]) \ , \ i = 2, \ldots, k, \end{aligned} \tag{3}$$

i.e. U_i completes the orthogonal basis for $\mathcal{N}_{i-1}$ to an orthogonal basis for $\mathcal{N}_i$ and U_{k+1} completes the basis for the last $\mathcal{N}_k$ to a basis for the full space $\mathbf{R}^n$ or $\mathbf{C}^n$. From this, we also have the equalities :

$$\begin{aligned} d_1 &= n_1, \\ d_i &= n_i - n_{i-1} \ , \ i = 2, \ldots, k, \end{aligned} \tag{4}$$

It then follows from (1)-(4) that $U^* \cdot A \cdot U$ has the following "staircase form" :

$$U^* \cdot A \cdot U = A_u \doteq \begin{bmatrix} 0_{d_1} & A_{1,2} & \ldots & A_{1,k} & A_{1,k+1} \\ & 0_{d_2} & & \vdots & \vdots \\ & & \ddots & A_{k-1,k} & \\ & & & 0_{d_k} & A_{k,k+1} \\ & & & & \hat{A} \end{bmatrix}, \tag{5}$$

where $\hat{A}$ is non singular and hence has no eigenvalues at $\lambda = 0$ anymore and where the $A_{i-1,i}$, $i = 2, \ldots, k$ matrices have full column rank d_i. It turns out that a matrix A decomposed as in (5) automatically reveals its null spaces $\mathcal{N}_i$ via (3) if the rank properties of $\hat{A}$ and the $A_{i,i+1}$, $i = 1, \ldots, k-1$ matrices are fulfilled. Indeed, it is easily verified from (5) that these properties are sufficient to ensure that $U^* \cdot A^i \cdot U = A_u^i$ has the form

$$U^* \cdot A^i \cdot U = A_u^i \doteq \left[\begin{array}{c|c} 0 & X \end{array}\right], \tag{6}$$

with n_i leading zero columns and a matrix X of full column rank $n - n_i$, from which (3) indeed follows. Constructing the form (5) can also be performed by directly operating on A instead of on its powers.

The basic idea of this algorithm was suggested by Kublanovskaya [14] and later improved by Ruhe [15] for its numerical robustness. It consists of building

up recursively transformations $V_j,\ j = 1, \ldots, k$ that match with U of (2) up to the block U_j. In each step j one thus reconstructs one of the orthogonal bases U_j in (2) and hence a null space $\mathcal{N}_j$ of (3), or, equivalently, one of the diagonal 0_{d_j} diagonal blocks in (5). Let indeed the situation at the end of step $\ell - 1$ be as follows :

$$V_{\ell-1}^* \cdot A \cdot V_{\ell-1} = A_u \doteq \begin{bmatrix} 0_{d_1} & A_{1,2} & \ldots & A_{1,\ell-1} & A_{1,\ell} \\ & 0_{d_2} & & \vdots & \vdots \\ & & \ddots & A_{\ell-2,\ell-1} & \\ & & & 0_{d_{\ell-1}} & A_{\ell-1,\ell} \\ & & & & A_{\ell,\ell} \end{bmatrix}, \tag{7}$$

where the $A_{i-1,i},\ i = 2, \ldots, \ell - 1$ matrices have full column rank d_i. At step ℓ one then performs a rank decomposition of $A_{\ell,\ell}$ in order to determine a unitary transformation $\hat{V}_\ell$ such that

$$A_{\ell,\ell} \cdot \hat{V}_\ell \doteq \left[\begin{array}{c|c} 0 & \hat{A}_\ell \end{array}\right], \tag{8}$$

where $\hat{A}_\ell$ has full column rank. It then follows that $\hat{V}_\ell^* \cdot A_{\ell,\ell} \cdot \hat{V}_\ell$ can be partitioned as :

$$\hat{V}_\ell^* \cdot A_{\ell,\ell} \cdot \hat{V}_\ell \doteq \left[\begin{array}{c|c} 0_{d_\ell} & A_{\ell,\ell+1} \\ \hline 0 & A_{\ell+1,\ell+1} \end{array}\right]. \tag{9}$$

Updating the transformation $V_{\ell-1}$ as follows :

$$V_\ell \doteq V_{\ell-1} \cdot \left[\begin{array}{c|c} I_{n_{\ell-1}} & 0 \\ \hline 0 & \hat{V}_\ell \end{array}\right], \tag{10}$$

one can then embed (9) in (7) yielding an updated version of (7) for $\ell := \ell+1$. The algorithm terminates with the situation where a diagonal block $A_{\ell,\ell}$ of full rank is encountered, and one has then the final configuration shown in (5) for $k = \ell$. It can be verified that $V_\ell^* \cdot A^i \cdot V_\ell \doteq A_v^i$ has the form (6) for each power $i \leq \ell$ and hence that the first i column blocks of V_ℓ span indeed $\mathcal{N}_i$ and this for $i = 1, \cdots, \ell$. The first ℓ blocks of the matrices U and V_ℓ thus "essentially match" with each other in the sense that they must span the same spaces (notice indeed that the U_i block matrices are only defined up to a unitary column transformation, since any orthogonal basis of $\mathcal{N}_i \cap \mathcal{N}_{i-1}^\perp$ can be chosen for U_i).

The above algorithm has the advantage of being backward stable [15] since it only applies and updates unitary transformations on the original matrix A. Although this algorithm is faster than the conceptual algorithm (1)-(5) using the

null spaces $\mathcal{N}_i$ of A^i, it is remarked in [9] that when the rank increases d_i are small (e.g. $d_i = 1$) then this algorithm is still rather expensive. The subsequent rank decompositions of the matrices $A_{\ell,\ell}$ (each of dimension $n - \ell$) ultimately leads to an $O(n^4)$ algorithm if indeed all eigenvalues of A are at $\lambda = 0$. Notice that this corresponds to the case where A has a single Jordan block at $\lambda = 0$ since then the dimension of the null spaces $\mathcal{N}_\ell$ grow with 1 at each step. As indicated in [13] this is the most frequently occurring case when considering only matrices A with a repeated root at $\lambda = 0$. Golub & Wilkinson then proceed to develop an economical algorithm (which in fact turns out to be cubic in the dimension n) for reconstructing such long Jordan chains. In [8], [1], [2] and [3] these ideas are gradually extended to arbitrary pencils of matrices $\lambda B - A$. Although Golub & Wilkinson were a direct source of inspiration for this extension to the generalized eigenvalue problem, there is a basic difference in the treatment of the problem. This is explained in the next section where we specialize the techniques developed for the generalized eigenvalue problem to the standard eigenvalue problem $\lambda I - A$ and develop a new algorithm for reconstructing Jordan structure of a matrix A.

3 A novel algorithm for the Jordan structure

In [3] it is shown how to exploit subsequent rank decompositions in the "staircase" algorithm for pencils $\lambda B - A$ by making use of two basic techniques. First one uses a preliminary unitary transformation to bring one of the two matrices (say B) in condensed form. Secondly one uses QR-decompositions for all rank decompositions such that the work performed in step ℓ can also be exploited in the next step $\ell + 1$. We now present a modified algorithm for computing the Jordan form of a matrix A which is based on the same principles as the Beelen & Van Dooren algorithm for the Kronecker structure of a pencil $\lambda B - A$. Yet, it differs slightly from it because here the special properties available from the standard eigenvalue problem can nicely be exploited.

We start with performing a QR-decomposition of the matrix A :

$$Q_1 \cdot A \doteq R_1 = \begin{bmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ & & r_{nn} \end{bmatrix}. \tag{11}$$

This preliminary computation requires $O(n^3)$ operations. The eigenvalue problem $\lambda I - A$ is clearly equivalent to the problem

$$\lambda Q_1 - R_1, \tag{12}$$

which has the special property of having a *unitary* coefficient Q_1 of λ and an upper triangular constant coefficient R_1. These two properties are preserved throughout the algorithm that we describe below. If A has a (possibly multiple) eigenvalue at

$\lambda = 0$ then R_1 must be singular and hence should have some diagonal element(s) equal to 0. Notice though that when A, or equivalently R_1, is *close* to being singular (i.e. has at least one *small* singular value) this is not necessarily true anymore for the diagonal elements of R_1. This is e.g. illustrated by the well-known example due to Kahan (see also [9]) :

$$R_1 = \begin{bmatrix} 1 & -1 & \dots & -1 \\ & 1 & \ddots & \vdots \\ & & \ddots & -1 \\ & & & 1 \end{bmatrix}, \tag{13}$$

which has a singular value of the order of magnitude of 2^{-n}. Therefore great care has to be put in finding the nullity of the triangular matrix R_1. This can e.g. be done by using the rank-revealing QR-decomposition made popular by T. Chan [4]. This technique allows one to update the R_1 factor of the QR-decomposition using left and right unitary transformations in order to display the nullity (say d_1) of R as follows :

$$U_1 \cdot R_1 \cdot V_1 \doteq \hat{R} = \left[\begin{array}{c|ccc} 0_{d_1} & * & \cdots & * \\ \hline & \hat{r}_{d_1+1,d_1+1} & & \vdots \\ & & \ddots & * \\ & & & \hat{r}_{n,n} \end{array}\right]. \tag{14}$$

This factorization has the additional advantage of preserving the triangular structure of R_1 while displaying its nullity. One can exploit this in a recursive algorithm as follows. Applying these transformations also to Q_1 yields a pencil which is unitarily equivalent to $\lambda I - A$:

$$U_1 Q (\lambda I - A) V_1 = \lambda \hat{Q} - \hat{R}, \tag{15}$$

and which can be partitioned as follows :

$$\lambda \hat{Q} - \hat{R} = \lambda \left[\begin{array}{c|c} \hat{Q}_1 & \hat{Q}_2 \end{array}\right] - \left[\begin{array}{c|c} 0 & \hat{R}_2 \end{array}\right]. \tag{16}$$

Here $\hat{Q}_1$ has d_1 columns and $\hat{Q}_2$ and $\hat{R}_2$ have $n - d_1$ columns. In [3] a technique is shown to update then this form by unitary left and right transformations U_2 and $\left[\begin{array}{c|c} I_{d_1} & 0 \\ \hline 0 & V_2 \end{array}\right]$ such that :

$$U_2 \hat{Q}_1 = \left[\begin{array}{c} \tilde{Q}_{11} \\ \hline 0 \end{array}\right], \tag{17}$$

with $\tilde{Q}_{11}$ upper triangular, and

$$U_2 \hat{R}_2 V_2 = \left[\begin{array}{c} \tilde{R}_{12} \\ \hline \tilde{R}_{22} \end{array}\right], \tag{18}$$

with $\tilde{R}_{22}$ upper triangular. Since the matrix

$$U_2 \hat{Q} \left[\begin{array}{c|c} I_{d_1} & 0 \\ \hline 0 & V_2 \end{array}\right] = \left[\begin{array}{c|c} \tilde{Q}_{11} & \tilde{Q}_{12} \\ \hline 0 & \tilde{Q}_{22} \end{array}\right] \tag{19}$$

is still unitary, $\tilde{Q}_{12}$ must be zero and the matrices $\tilde{Q}_{11}$ and $\tilde{Q}_{22}$ must be unitary. Moreover, $\tilde{Q}_{11}$ – being upper triangular – must also be diagonal, and it is easy to see that the degrees of freedom in the construction of U_2 is such that $\tilde{Q}_{11}$ can be chosen equal to I_{d_1}. We thus have :

$$U_2(\lambda \hat{Q} - \hat{R}) \left[\begin{array}{c|c} I_{d_1} & 0 \\ \hline 0 & V_2 \end{array}\right] = \lambda \tilde{Q} - \tilde{R} = \lambda \left[\begin{array}{c|c} I_{d_1} & 0 \\ \hline 0 & \tilde{Q}_{22} \end{array}\right] - \left[\begin{array}{c|c} 0_{d_1} & \tilde{R}_{12} \\ \hline 0 & \tilde{R}_{22} \end{array}\right]. \tag{20}$$

Notice that meanwhile we have performed one step of the basic Kublanovskaya-Ruhe algorithm. Indeed, multiplying out $\lambda \tilde{Q} - \tilde{R}$ with $\tilde{Q}^*$ yields :

$$\lambda \left[\begin{array}{c|c} I_{d_1} & 0 \\ \hline 0 & I_{n-d_1} \end{array}\right] - \left[\begin{array}{c|c} 0_{d_1} & \tilde{R}_{12} \\ \hline 0 & \tilde{Q}_{22}^* \tilde{R}_{22} \end{array}\right], \tag{21}$$

which is the first step of that algorithm. The reduced problem

$$\lambda I_{n-d_1} - A_{22} \doteq \lambda I_{n-d_1} - \tilde{Q}_{22}^* \tilde{R}_{22}, \tag{22}$$

which is normally used in the Kublanovskaya-Ruhe algorithm, is already given here in factorized form. We can thus repeat the above procedure (14)-(20) on the reduced pencil

$$\lambda Q_2 - R_2 \doteq \lambda \tilde{Q}_{22} - \tilde{R}_{22} = \tilde{Q}_{22}(\lambda I_{n-d_1} - A_{22}). \tag{23}$$

The algorithm then continues recursively until at step k it yields the form (5), where $\hat{A}$ is given by the factorization $\hat{A} = \tilde{Q}_{k+1,k+1}^* \tilde{R}_{k+1,k+1}$. The novelty of this algorithm is thus that instead of constructing recursively the matrices $A_{\ell,\ell}$ in (7), it directly constructs their QR-factors $\tilde{Q}_{\ell,\ell}$ and $\tilde{R}_{\ell,\ell}$, and this in an economical manner. The complexity of the algorithm is discussed in the next section.

4 Advantages and disadvantages of the method

As was already pointed out by Golub & Wilkinson [9], one has to pay attention to several aspects when trying to develop an algorithm for computing the Jordan

canonical form. In short, these can be described as : *numerical stability*, *low complexity*, *robustness* and *elegance*. We now give below a few words of explanation about each of these issues.

Numerical stability of an algorithm can in fact be defined in various ways. One of the most popular ones in numerical linear algebra, is to say that all computations (together with their rounding errors) can be interpreted as the *exact* result of slightly perturbed input data. This concept of *backward stability* is largely due to Wilkinson and extensively used in [19] for a whole variety of matrix problems. For the problem of the Jordan canonical form it is easy to prove that such a (strong) form of stability holds indeed for both the Kublanovskaya-Ruhe algorithm and the one described in the previous section. Both algorithms clearly consist of a sequence of unitary transformations applied to the rows and columns of the original pencil $\lambda I - A$. The errors performed in each elementary step (say, a Householder transformation or a Givens rotation) can be bounded using the classical analysis of Wilkinson [19]. Moreover, these errors can easily be transformed back to the original pencil. When doing this, their bound is not enlarged since all intermediate transformations are unitary. Therefore the *total* error can be bounded by just the sum of the bounds for the elementary steps. As shown in [19] this technique typically yields satisfactory bounds for the *total* backward error.

Things are quite different with e.g. the matrix powering method, explained conceptually in section 2 and made explicit in the first method described by Golub & Wilkinson (in [9], section 9). Since here at each step one multiplies with the matrix A it is virtually impossible to map back the errors performed at a certain step ℓ to the original data matrix A without significantly enlarging their norm (especially when A is badly conditioned with respect to inversion). For Golub & Wilkinson's "economical algorithm" (in [9], section 11) bounds on the backward error were obtained in [12]. They depend on a quantity which can be arbitrarily large, in spite of the use of stable decompositions at each step. The backward error of this algorithm is indeed linked with the condition number $\kappa(Z)$ of the similarity transformation Z that puts A in Jordan normal form (for the considered eigenvalue), and this can be arbitrarily large for some matrices A. Although backward stability is an important guarantee for an algorithm, we want to remind here that the absence of this property does not condemn yet an algorithm as being "unreliable".

Low complexity is of course always desirable and for most matrix problems (involving a dense matrix) one would like to have a complexity of $O(n^3)$ operations, where n is the order of the matrix. As discussed earlier, this does not hold for the Kublanovskaya-Ruhe algorithm in general. A specific counter example is the single large Jordan block at $\lambda = 0$ which leads to an operation count of $O(n^4)$ since $O(n)$ matrix factorizations have to be performed. In the new algorithm presented in the previous section we in fact derive directly a recursion for the *factors*

of the relevant matrices. We now shortly indicate why this leads to a complexity of $O(n^3)$. The initial QR-factorization (11) is $O(n^3)$ and is done only once. In each subsequent step one then performs the updating transformations (14) for obtaining a rank revealing triangular factor $\hat{R}$. Assuming the nullity is d_1 at step 1, this involves only $O(d_1 n^2)$ operations (see [4]). Then one needs to update the pencil $\lambda\hat{Q} - \hat{R}$ to yield $\lambda\tilde{Q} - \tilde{R}$ as given in (20). It is shown in [3] that the updating transformation U_2 can be performed with $d_1(n - \frac{d_1+1}{2})$ Givens transformations and V_2 with $d_1(n - d_1)$ Givens transformations. Indeed, the Givens transformations on the rows of $\hat{Q}$ are needed to eliminate the $d_1(n - \frac{d_1+1}{2})$ elements below the diagonal of $\hat{Q}_1$ and the Givens transformations on the columns are needed to zero out again each element (created by one of the row transformations) below the diagonal of $\tilde{R}_{22}$. The total operation count for this updating is then $O(n^2 d_1)$ according to [3]. For the second step one then requires $O(n^2 d_2)$ flops, where d_2 is the nullity of $\tilde{R}_{22}$, and so on. By induction one then finally has $O(n^2 \Sigma d_i)$ as total operation count. Since $\Sigma d_i = n_k < n$, we thus clearly have an $O(n^3)$ algorithm for performing the so-called staircase reduction. In [9], Golub & Wilkinson recognize the need of such a $O(n^3)$ algorithm and their "economical method" ([9], section 11) also has this complexity (see [12]). There also the dimension of the problem is deflated at each step, even in a more pronounced manner. As pointed out above, though, their method suffers from the lack of a proof of backward stability. When talking about complexity, one should also consider the issue of the work space requested by the algorithm. The present algorithm clearly needs to store and update the additional matrix Q in comparison with methods directly working on A. The algorithm needs thus one additional $n \times n$ array to carry along.

With *robustness* we mean the way an algorithm reacts in the presence of a possibly ill-posed problem. For the clustering problem e.g. a robust algorithm is expected to retrieve the same multiplicity of an eigenvalue, for all matrices within a certain (ϵ-small) neighbourhood of the given matrix A. In order to analyze robustness one has to understand well the sensitivity of the problem. For the clustering problem e.g., the various sensitivity results [20] [21] [6] [7] have led to tests for deciding when eigenvalues should be considered as multiple and when eigenvectors should be merged into one space in order to improve its sensitivity. For the problem at hand, a robust algorithm should retrieve the same Jordan structure of a given eigenvalue, for all matrices within a certain (ϵ-small) neighbourhood of the given matrix A. Good sensitivity results for the null spaces $\mathcal{N}_i$ directly in terms of perturbations on A are missing in this context. Of course, the sensitivity of the first space $\mathcal{N}_1$ is linked with the singular values of A (see [16]). This is also the reason why often SVD techniques are preferred over QR techniques when ranges and null spaces are involved. But as we show with a very simple example in the next section, this only holds for the first space $\mathcal{N}_1$. The sensitivity of the subsequent spaces $\mathcal{N}_i$ is a more complex function of perturbations on A and the use of SVD techniques does not guarantee anymore

to yield robust answers.

Finally, we come to the point of *elegance*, with which we mean various things : that the method should be simple to encode (it should not require too much overhead or book-keeping), it should be flexible (when e.g. moving to another eigenvalue, one should not have to restart the algorithm on the whole matrix, but on a deflated one), it should be easy to understand, and so on. Elegance is of course usually missing in most sophisticated algorithms, but when having to choose between different performant algorithms it can play an important role. In our example, one could say that elegance is in favour of the Kublanovskaya-Ruhe algorithm. It is relatively easy to understand, easy to implement (with e.g. QR-decompositions), flexible when switching to another eigenvalue, and usually not too expensive when the Jordan chains are reasonably small. When large Jordan chains are present the "economical method" of Golub & Wilkinson, and the above new algorithm have the advantage of being much faster. For both these methods, though, switching to another eigenvalue requires some work. Moreover, as we show in the next section, large Jordan chains are not easy to handle with *any* algorithm.

5 An embarassing example

In this section we want to draw the attention to the fact that the stability of an algorithm does not imply that all will go well when trying to reconstruct the Jordan canonical form of a matrix A. In order to show this we use an example borrowed from [18]. Let ϵ be the relative precision of our computer and define then the following matrix :

$$A = \begin{bmatrix} 0 & \sqrt{\epsilon} & \epsilon \\ \sqrt{\epsilon} & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}. \tag{24}$$

When computing the powers of this matrix :

$$A^2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \epsilon & \epsilon\sqrt{\epsilon} \\ 0 & -\sqrt{\epsilon} & -\epsilon \end{bmatrix}, \; A^3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \; \ldots \tag{25}$$

one sees readily that A must have a single Jordan block of size 3 at $\lambda = 0$. The nullities $n_i, i = 1, 2, 3$ of the powers A^i are indeed equal to 1, 2 and 3 respectively. The unitary transformation reducing A to the staircase form (5) is :

$$U^* \cdot A \cdot U = \begin{bmatrix} 0 & s & -c \\ 1 & 0 & 0 \\ 0 & c & s \end{bmatrix} \begin{bmatrix} 0 & \sqrt{\epsilon} & \epsilon \\ \sqrt{\epsilon} & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ s & 0 & c \\ -c & 0 & s \end{bmatrix} = \begin{bmatrix} 0 & a & 0 \\ 0 & 0 & a\sqrt{\epsilon} \\ 0 & 0 & 0 \end{bmatrix}, \tag{26}$$

where $a = \sqrt{1+\epsilon},\ c = 1/a,\ s = \sqrt{\epsilon}/a$. Consider now also the slightly perturbed matrix A_ϵ obtained by deleting the (1,3) element of A, which is ϵ-small. The powers of A_ϵ now become :

$$A_\epsilon = \begin{bmatrix} 0 & \sqrt{\epsilon} & 0 \\ \sqrt{\epsilon} & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix},\ A_\epsilon^2 = \begin{bmatrix} \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & -\sqrt{\epsilon} & 0 \end{bmatrix},\ A_\epsilon^3 = \begin{bmatrix} 0 & \epsilon\sqrt{\epsilon} & 0 \\ \epsilon\sqrt{\epsilon} & 0 & 0 \\ -\epsilon & 0 & 0 \end{bmatrix}. \quad (27)$$

Clearly, A_ϵ is not nilpotent anymore, and its form (5) now looks like :

$$U_\epsilon^* \cdot A_\epsilon \cdot U_\epsilon = \begin{bmatrix} 0 & 0 & -1 \\ b & b & 0 \\ b & -b & 0 \end{bmatrix} \begin{bmatrix} 0 & \sqrt{\epsilon} & 0 \\ \sqrt{\epsilon} & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & b & b \\ 0 & b & -b \\ -1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & b & b \\ 0 & \sqrt{\epsilon} & 0 \\ 0 & 0 & -\sqrt{\epsilon} \end{bmatrix}, \quad (28)$$

where $b = \sqrt{2}/2$.

Below we give the singular values σ_i of the consecutive powers of both matrices (only the order of magnitude of these singular values is given) :

	A	A^2	A^3
σ_1	1	$\sqrt{\epsilon}$	0
σ_2	$\sqrt{\epsilon}$	0	0
σ_3	0	0	0

	A_ϵ	A_ϵ^2	A_ϵ^3
σ_1	1	$\sqrt{\epsilon}$	ϵ
σ_2	$\sqrt{\epsilon}$	ϵ	$\epsilon\sqrt{\epsilon}$
σ_3	0	0	0

(29)

Looking at these tables it is reasonable to claim that the consecutive powers of A_ϵ have nullities 1, 2 and 3, respectively. What is it then that makes the decompositions (26) and (28) look so different ? Although the matrices A^i and A_ϵ^i and also their respective singular values are ϵ-close to each other, the same does not hold for the corresponding singular vectors. Indeed let us compare the corresponding null spaces $\mathcal{N}_1$ and $\mathcal{N}_{1_\epsilon}$. Basis vectors for these two spaces are :

$$\mathcal{N}_1 = Span(\begin{bmatrix} 0 \\ \sqrt{\epsilon} \\ -1 \end{bmatrix}), \qquad \mathcal{N}_{1_\epsilon} = Span(\begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}). \quad (30)$$

It is precisely this first null space (or rather, its orthogonal complement) that determines what the A_{22} matrix will be (see (7) for $\ell = 2$) and for both matrices we have :

$$A_{22} = \begin{bmatrix} 0 & a\sqrt{\epsilon} \\ 0 & 0 \end{bmatrix}, \qquad A_{22_\epsilon} = \begin{bmatrix} \sqrt{\epsilon} & 0 \\ 0 & -\sqrt{\epsilon} \end{bmatrix}, \quad (31)$$

or a unitary similarity of these matrices (since the orthogonal complement of the first null space is only defined up to a 2×2 unitary transformation). Clearly one cannot say anymore now that the reduced matrix A_{22_ϵ} is nilpotent since its singular values are both $\sqrt{\epsilon}$. Hence the deflated problem does not reflect the properties of the full matrix anymore !

While in step 1 the rank determination of A_ϵ was reliable, its corresponding null space was sensitive because the second singular value is quite small ($\sqrt{\epsilon}$) but not neglectable. This *always* occurs when small but non neglectable singular values are encountered. It corresponds to staircase forms with "weak" stairs, such as in (26). One could say that the algorithm building up the staircase form (5) runs into trouble as soon as a "weak" stair is encountered. The rest of the staircase construction then becomes very sensitive and the algorithm is likely to "collapse". With that we mean here that the rest of the Jordan blocks will get perturbed and will fall apart in a cluster of nonzero eigenvalues, as is seen in (28). Such problems are to be expected in *any* algorithm that uses the idea of deflating the problem each time a null space is found. Unfortunately, all the economical methods use precisely this idea !

In order to see the effect of a particular algorithm on A we should first perform a *random* unitary transformation, say $V^* \cdot A \cdot V$, on A in order to avoid trivial arithmetic due to the sparsity of the matrix. In this case both the Kublanovskaya-Ruhe algorithm and the new algorithm presented above, yielded a form (5) which looks more like (28) than like (26). Also the economical method of Golub & Wilkinson ran into troubles here. The reason for the preference of (28) over (26) under the presence of rounding errors is that the "uncertainty cone" of the first singular vector is quite large and the probability that one will compute one that looks like the "true" $\mathcal{N}_1$ is neglectable.

Yet, we do not want to advocate here that one should go back to the definitions (1)-(3) and compute null spaces of A^i again. For the above example this indeed works fine, but it is even easier to construct difficult example for that approach as well. Any matrix A (say of norm 1) with eigenvalues both at 0 and at $\sqrt{\epsilon}$ will of course give erroneous results as soon as one computes A^2, since in that matrix they can not be told apart anymore. Moreover as we said above, these methods are unstable.

6 Concluding remarks

In previous sections we focussed on the computation of the Jordan structure of a real or complex matrix A at one eigenvalue λ_0. Thereby we assumed that this eigenvalue was *known*. (Without loss of generality we then also assumed that $\lambda_0 = 0$). But this is not generally the case and one thus has to compute the eigenvalues of A. For a general matrix A, it is recommended to use the so-called

Schur decomposition of A :

$$U^* \cdot A \cdot U \doteq A_s = \begin{bmatrix} a_{11} & * & \cdots & * \\ & a_{22} & & \vdots \\ & & \ddots & * \\ & & & a_{nn} \end{bmatrix} \tag{32}$$

which displays the (computed) eigenvalues $\lambda_i = a_{ii},\ i = 1, \ldots, n$ via the diagonal elements of A_s. When A has Jordan blocks at a specific eigenvalue λ_0, they also turn out to be very sensitive to rounding errors. Therefore, one should not expect to retrieve them unperturbed on the diagonal of A_s. In this case one is bound to use techniques that recover "clusters" from the diagonal elements of A_s. This is perhaps the most delicate step in the construction of the Jordan canonical form (see also [9]).

But since the Schur form (32) is often a starting point for constructing the Jordan canonical form, one could wish to use the fact that $(\lambda_0 I - A_s)$ is already triangular and e.g. perform a rank revealing QR-decomposition of that matrix instead of the full matrix $(\lambda_0 I - A)$. Ideally one would then like to *update* the Schur form in order to display the first zero block (say 0_{d_1}) of the form (5), while *preserving* the rest of the updated matrix $\tilde{A}_s$ in Schur form. This turns out to be possible in an economical manner for only a very limited class of problems. In general, though, this update requires $\mathrm{O}(n^3)$ operations. This has to do with the fact that the so-called Schur vectors of A_s and $\tilde{A}_s$ can be very different and the updating transformation has to transform one set into the other. The compromise given in section 5 is in fact to try to update the QR factors of A instead of trying to update its Schur form, and this appears to be always possible in $\mathrm{O}(d_1 n^2)$ operations.

We want to conclude by recalling some of the remarks made in this paper when dealing with such a tough problem as the computation of the Jordan canonical form :

- When developing an algorithm for reconstructing the Jordan structure of a matrix, one should first aim at a numerically stable algorithm. Since ranks are an important issue here, small singular values should be detected adequately. If possible one should also aim for a *reasonably* fast algorithm.

- One should not expect too much from the computational results even when using a numerically stable algorithm. The problem remains a very sensitive one and robustness is not easy to achieve. Particularly large Jordan blocks with "weak" stairs are difficult to handle.

- The sensitivity of the spaces $\mathcal{N}_i$ corresponding to an eigenvalue and the sensitivity of the whole invariant subspace $\mathcal{S}$ corresponding to that eigenvalue, are clearly linked since $\mathcal{S}$ is nothing but the last $\mathcal{N}_k$. While the sensitivity of the invariant subspaces is well studied, this is not the case for the chain of spaces $\mathcal{N}_i$.

- The recommendation that clustering and Jordan structure computation should not be treated as two separate numerical problems, is retrieved in the connection of the robustness issues of both problems.

The toughest questions are (as is often the case) the ones that aren't really treated in this paper. Nevertheless, we hope that the present ideas will help understanding a bit better the computational aspects of the Jordan structure of a matrix A. Of course, most of the numerical aspects encountered here also carry over to the computation of the Kronecker structure of a pencil of matrices $\lambda B - A$.

Acknowledgement :

It is obvious from the numerous references in the text that the work of Golub & Wilkinson had a large influence on this research. Bo Kågström drew our attention to reference [12] and made valuable remarks on an earlier version of this paper. Last but not least, we want to mention that several personal discussions with Jim Wilkinson helped us gain insight into this problem. We deeply regret his untimely death.

References

[1] T. Beelen, P. Van Dooren, M. Verhaegen, A class of fast staircase algorithms for generalized state-space systems, in *Proceedings American Control Conference*, Seattle, Washington, 425-426, 1986.

[2] T. Beelen, *New algorithms for computing the Kronecker structure of a pencil with applications to systems and control theory*, Doctoral Thesis, Techn. University Eindhoven, 1987.

[3] T. Beelen, P. Van Dooren, An improved algorithm for the computation of Kronecker's canonical form of a singular pencil, *Linear Algebra & Applications* to appear.

[4] T. Chan, Rank revealing QR-factorizations, *Linear Algebra & Applications* **88/89** (1987) 67-82.

[5] J. Demmel, *A numerical analyst's Jordan canonical form*, Ph.D. Diss., Computer Science Div., University of California, Berkeley, 1983.

[6] J. Demmel, Computing stable eigendecompositions of matrices, *Linear Algebra & Applications* **79** (1986) 163-193.

[7] J. Demmel, B. Kågström, Computing stable eigendecompositions of matrin pencils, *Linear Algebra & Applications* **88/89** (1987) 139-186.

[8] A. Emami-Naeini, P. Van Dooren, On the computation of transmission zeros of a state space system, *Automatica* **18** (1982) 415-430.

[9] G. H. Golub, J. H. Wilkinson, Ill-conditioned eigensystems and the computation of the Jordan canonical form, *SIAM Review* **18** (1976) 578-619.

[10] B. Kågström, A. Ruhe, An algorithm for numerical computation of the Jordan normal form of a complex matrix, *ACM Trans. on Math. Soft.* **6** (1980) 398-419.

[11] B. Kågström, A. Ruhe, ALGORITHM 560, JNF, An algorithm for numerical computation of the Jordan normal form of a complex matrix, *ACM Trans. on Math. Soft.* **6** (1980) 437-443.

[12] B. Kågström, *How to compute the Jordan normal form - the choice between similarity transformations and methods using the chain relations* Intern. Rept. UMINF 91.81, Inst. of Information Processing, Univ. of Umeå, 1981.

[13] W. M. Kahan, *Conserving confluence curbs ill-condition*, Computer Sci. Techn. Rep. 6, Univ. of California at Berkeley, 1972.

[14] V. N. Kublanovskaya, On a method for solving the complete eigenvalue problem for a degenerate matrix, *Z. Vycisl. Mat. i Fiz.* **6** (1966) 611-620 = *USSR Computational Math. and Math. Phys.* **6** (1968) 1-14.

[15] A. Ruhe, An algorithm for numerical determination of the structure of a general matrix, *BIT* **10** (1970) 196-216.

[16] G. Stewart, Error and perturbation bounds for subspaces associated with certain eigenvalue problems, *SIAM Review* **15** (1973) 752-764.

[17] P. Van Dooren, The computation of Kronecker's canonical form of a singular pencil, *Linear Algebra & Applications* **27** (1979) 103-140.

[18] P. Van Dooren, *The generalized eigenstructure problem. Applications in linear system theory*, Doctoral Thesis, Catholic University Leuven, 1979.

[19] J. Wilkinson, *The algebraic eigenvalue problem*, Oxford University Press, Oxford, 1965.

[20] J. Wilkinson, Sensitivity of eigenvalues, *Utilitas Mathematica* **25** (1984) 5-76.

[21] J. Wilkinson, Sensitivity of eigenvalues II, *Utilitas Mathematica* **30** (1986) 243-286.