Frobenius form in expected matrix multiplication time over sufficiently large fields

Clément Pernet, Arne Storjohann

David R. Cheriton School of Computer Science, University of Waterloo, Ontario, Canada N2L 3G1

Abstract

A new randomized algorithm is presented for computing the Frobenius form of an $n \times n$ matrix over a sufficiently large field K. Let $2 < \omega \leq 3$ be such that two matrices in $\mathsf{K}^{n\times n}$ can be multiplied together with $O(n^\omega)$ operations in K. If $\#\mathsf{K} \geq 2n^2$, the new algorithm uses an expected number of $O(n^\omega)$ operations in K to compute the Frobenius form of A, matching the lower bound for the cost of this problem. A similarity transformation matrix can be computed with an additional $O(n^\omega \log\log n)$ operations in K. For comparison, the randomized algorithms of Giesbrecht (1993, 1995) and Eberly (2000) use an expected number of $O(n^\omega \log n)$ operations in K to compute both the form and a similarity transformation matrix. Eberly's algorithm has the advantage of being applicable over all fields.

Key words: Frobenius form, characteristic polynomial, minimal polynomial, randomized algorithm, Las Vegas, algebraic complexity

1. Introduction

Like the better known Jordan form, the Frobenius form of a square matrix A over a field K is a unique representative of the set of all matrices similar to A. The Frobenius form, also called the rational canonical form, has the shape

$$U^{-1}AU = F = \begin{bmatrix} C_{f_1} & & & \\ & C_{f_2} & & \\ & & \ddots & \\ & & & C_{f_l} \end{bmatrix} \in \mathsf{K}^{n\times n}.$$

Email addresses: cpernet@uwaterloo.ca (Clément Pernet), astorjoh@uwaterloo.ca (Arne Storjohann).

Each block C_{f_i} is the companion matrix (see §2) of a monic $f_i \in \mathsf{K}[x]$, and $f_{i+1}|f_i$ for $1 \leq i \leq l-1$. The form reveals many invariants of A: the minimal polynomial is f_1, the characteristic polynomial is the product $f_1 f_2 \cdots f_l$, the determinant is the constant coefficient of $f_1 f_2 \cdots f_l$, and the rank is equal to n minus the number of companion blocks corresponding to a polynomial with a zero constant coefficient.

The problem of computing the Frobenius form has been well studied. Storjohann (1998) describes an algorithm that uses $2n^3 + O(n^2)$ operations in K to compute the form F itself. A similarity transformation matrix U such that $U^{-1}AU = F$ can be computed with $6n^3 + O(n^2(\log n)^2))$ operations in K using the algorithm of Storjohann and Villard (2000). For a more thorough survey of previous algorithms we refer to Storjohann (2000).

In this paper our focus is on the problem of reducing the computation of the Frobenius form to that of matrix multiplication. Let ω be a valid exponent for the complexity of matrix multiplication: $O(n^\omega)$ operations from K are sufficient to multiply together two $n \times n$ matrices over K. Henceforth in this paper, all complexity bounds are in terms of number of required field operations from K, and we make the common assumption that $\omega > 2$. The first reduction to matrix multiplication is the randomized Las Vegas algorithm of Giesbrecht (1993, 1995). Over a sufficiently large field, with $\#\mathsf{K} \geq n^2$, Giesbrecht's algorithm computes the Frobenius form together with a similarity transformation matrix in expected time $O(n^\omega \log n)$. More recently, Eberly (2000) describes a Las Vegas algorithm, applicable over a field K of any size, that computes the form and a transformation matrix in expected time $O(n^\omega \log n)$, and Storjohann (2000, 2001) describes a deterministic algorithm for computing a transformation matrix in time $O(n^\omega (\log n)(\log\log n))$. Note that all of these reductions require a polylogarithmic number of matrix multiplications.

To understand the source of the $\log n$ factor in all the cost bounds in the previous paragraph, consider the problem of computing only one of the invariants revealed by the Frobenius form: the characteristic polynomial. Keller-Gehrig (1985) gave three reductions of the problem of computing the characteristic polynomial to that of matrix multiplication. Keller-Gehrig's third algorithm has cost $O(n^\omega)$ but only works for input matrices with restrictive genericity requirements. His first algorithm, a simplified version of the second, also only works for input matrices satisfying certain requirements. Of primary interest here is his second algorithm which works for all input matrices $A \in \mathsf{K}^{n\times n}$ and has a running time of $O(n^\omega \log n)$. The algorithm works by computing a Hessenberg form of A (see §2), from which the characteristic polynomial is easily recovered. The extra $\log n$ factor arises because the algorithm computes the $\Theta(\log n)$ matrix powers $A^2, A^4, A^8, \ldots, A^{\lceil \log_2 n \rceil}$. Similarly, the $\log n$ factor in the cost bounds for previous reductions to matrix multiplication for the problem of computing the Frobenius form arise because Keller–Gehrig's algorithm is used as a subroutine directly (Giesbrecht, 1993, 1995; Storjohann, 2000, 2001) or because a logarithmic number of powers of A might be computed (Eberly, 2000).

In this paper we combine ideas from Giesbrecht (1995), Keller-Gehrig (1985) and Villard (1997) to get a new Las Vegas randomized algorithm for computing the Frobenius form. If K has at least $2n^2$ elements the new algorithm has expected cost $O(n^\omega)$, matching the lower bound for this problem. Unlike Keller-Gehrig's $O(n^\omega \log n)$ algorithm for the characteristic polynomial, we proceed in steps for $k = 1, 2, 3, \ldots, n-1$ and thus the new algorithm converges arithmetically. The algorithm we describe shares more similarities with Keller-Gehrig's $O(n^\omega)$ deterministic algorithm for generic matrices; the main

difference is that we randomize and show how to take into account the block structure that will arise depending on the degrees of the invariant factors of the input matrix.

In Section 2 we introduce some notation and recall some facts about Krylov matrices. Section 3 gives a worked example of the new algorithm and offers an overview of Sections 4—6 which are devoted to presenting the algorithm and proving correctness. Section 7 shows how to compute a similarity transform matrix in time $O(n^\omega \log\log n)$. The new algorithm is not only of theoretical interest but also practical. In Section 8 we describe an implementation, present some timings, and compare with the previously most efficient implementations that we are aware of. Section 9 concludes with some open problems.

2. Notation and preliminaries

We will frequently write matrices using a conformal block decomposition. A block is a submatrix comprised of a contiguous sequence of rows and columns. In special cases, a block may be a single matrix entry or may have row or column dimension zero. The generic block label $*$ denotes that a block is possibly nonzero. Blocks that are necessarily zero are left unlabelled. The label $*$ is also used as a generic index to avoid cluttering the presentation when explicit indices are not required.

Let $g = g_0 + g_1 x + \cdots + g_{r-1}x^{r-1} + x^r \in \mathsf{K}[x]$. The companion matrix of g is denoted by C_g and has the shape

$$C_g = \begin{bmatrix} 0 & \cdots & 0 & -g_0 \\ 1 & \ddots & \vdots & \vdots \\ & \ddots & 0 & -g_{r-2} \\ & & 1 & -g_{r-1} \end{bmatrix} \in \mathsf{K}^{r\times r}.$$

When using C_g as a block label, we allow the special case $g = 1$ in which case C_g has zero rows and columns.

Let $b = b_0 + b_1 x + \cdots + b_d x^d \in \mathsf{K}[x]$. We use the label B_b to denote a block which has the shape

$$B_b = \begin{bmatrix} & b_0 \\ & b_1 \\ & \vdots \\ & b_d \\ & 0 \\ & \vdots \\ & 0 \end{bmatrix}.$$

More precisely, all entries of B_b are zero except for entries in the last column row i equal to b_{i-1}, $1 \le i \le d+1$. The dimensions of a block labeled B_b will be conformal with adjacent blocks. Note that B_b may have zero columns and should have at least $1 + \deg b$ rows if $b \neq 0$.

Every square matrix over K can be written using a block decomposition as

$$\begin{bmatrix} C_{c_1} & B_{b_{12}} & \cdots & B_{b_{1k}} \\ B_{b_{21}} & C_{c_2} & & B_{b_{2k}} \\ \vdots & & \ddots & \vdots \\ B_{b_{k1}} & B_{b_{k2}} & \cdots & C_{c_k} \end{bmatrix}$$

for some (not necessarily unique) choices of the c_* and b_{**}. In particular, we allow that some of the diagonal companion blocks in the decomposition might be 0×0. But if we specify that k should be minimal, the decomposition is unique.

For a square matrix $A \in \mathsf{K}^{n\times n}$ and vector $v \in \mathsf{K}^{n\times 1}$, let $K_A(v,d)$ denote the Krylov matrix

$$K_A(v,d) = \left[\, v \,\middle|\, Av \,\middle|\cdots\middle|\, A^{d-1}v \,\right] \in \mathsf{K}^{n\times d}.$$

For $V \in \mathsf{K}^{n\times j}$ we denote by $\mathrm{Orb}_A(V)$ the subspace of K^n spanned by all the column vectors in $\left[\, V \mid AV \mid A^2V \mid \ldots \,\right]$.

Fact 2.1. *Let $A \in \mathsf{K}^{n\times n}$ be arbitrary and $U \in \mathsf{K}^{n\times n}$ be nonsingular. Then*

(1) $U = \left[\, K_A(v_1,d_1) \,\middle|\, K_A(v_2,d_2) \,\middle|\cdots\middle|\, K_A(v_m,d_m) \,\right]$ for some vectors $v_1, v_2, \ldots, v_m \in \mathsf{K}^{n\times 1}$ and positive integers $d_1, d_2, \ldots, d_m$ if and only if

$$U^{-1}AU = \begin{bmatrix} C_1 & B_* & \cdots & B_* \\ B_* & C_2 & \cdots & B_* \\ \vdots & \vdots & \ddots & \vdots \\ B_* & B_* & \cdots & C_m \end{bmatrix} \tag{2.1}$$

with C_i of dimension d_i, $1 \le i \le m$.

(2) For any j, $1 \le j \le m$, the matrix in (2.1) can be written as

$$\left[\begin{array}{cccc|cccc} C_1 & B_* & \cdots & B_* & B_* & B_* & \cdots & B_* \\ B^* & C_2 & \cdots & B_* & B_* & B_* & \cdots & B_* \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ B_* & B_* & \cdots & C_j & B_* & B_* & \cdots & B_* \\ \hline & & & & C_{j+1} & B_* & \cdots & B_* \\ & & & & B_* & C_{j+2} & \cdots & B_* \\ & & & & \vdots & \vdots & \ddots & \vdots \\ & & & & B_* & B_* & \cdots & C_m \end{array}\right]$$

if and only if the dimension of $\mathrm{Orb}_A([\, v_1 \mid \cdots \mid v_j \,])$ is equal to $d_1 + \cdots + d_j$.

We call the matrix in (2.1) a *shifted form* with degree sequence $(d_1, d_2, \ldots, d_m)$, corresponding to the dimensions of the diagonal blocks. A shifted form that is block upper

triangular with all diagonal blocks of the form C_* is called a shift-Hessenberg form.

Polynomial matrices $A, B \in \mathsf{K}[x]^{n\times n}$ are *equivalent* over $\mathsf{K}[x]$ if there exists unimodular matrices $U, V \in \mathsf{K}[x]^{n\times n}$ such that $UAV = B$. Recall that the unimodular matrices over $\mathsf{K}[x]$ are precisely those that are invertible over $\mathsf{K}[x]$, thus those with determinant a nonzero constant polynomial. The following classical result links the notion of similarity over K (denoted by $\sim$) with equivalence over $\mathsf{K}[x]$ (denoted by $\equiv$).

Theorem 2.2. (Fundamental Theorem of Similarity over a Field)

$$\begin{bmatrix} C_{c_1} & B_{b_{12}} & \cdots & B_{b_{1n}} \\ B_{b_{21}} & C_{c_2} & & B_{b_{2n}} \\ \vdots & & \ddots & \vdots \\ B_{b_{n1}} & B_{b_{n2}} & \cdots & C_{c_n} \end{bmatrix} \sim \begin{bmatrix} C_{\bar{c}_1} & B_{\bar{b}_{12}} & \cdots & B_{\bar{b}_{1n}} \\ B_{\bar{b}_{21}} & C_{\bar{c}_2} & & B_{\bar{b}_{2n}} \\ \vdots & & \ddots & \vdots \\ B_{\bar{b}_{n1}} & B_{\bar{b}_{n2}} & \cdots & C_{\bar{c}_n} \end{bmatrix}$$

if and only if

$$\begin{bmatrix} c_1 & b_{12} & \cdots & b_{1n} \\ b_{21} & c_2 & & b_{2n} \\ \vdots & & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & c_n \end{bmatrix} \equiv \begin{bmatrix} \bar{c}_1 & \bar{b}_{12} & \cdots & \bar{b}_{1n} \\ \bar{b}_{21} & \bar{c}_2 & & \bar{b}_{2n} \\ \vdots & & \ddots & \vdots \\ \bar{b}_{n1} & \bar{b}_{n2} & \cdots & \bar{c}_n \end{bmatrix}.$$

Mathematical background information can be found in (Hoffman and Kunze, 1971, Chapter 7) and (Gantmacher, 1990, Chapter 7).

3. Overview

The key to our algorithm is what we call a *k-uniform shifted form*: each diagonal companion block has dimension k, except for possibly the last which may have dimension less than k. For brevity we will refer to such a matrix as a k-shifted form. The algorithm proceeds in steps for $k = 1, 2, 3, \ldots, n-1$. Step k involves the transformation of a k-shifted form to a $(k+1)$-shifted form. We begin directly with a complete worked example of the algorithm.

Since every square matrix over a field is in 1-shifted form, we may consider our input matrix to be in 1-shifted form. For our example we will start with the following 1-shifted

form of dimension 14 over $\mathbb{Z}/(97)$.

$$A_1 = \begin{bmatrix} 15 & 5 & 24 & 26 & 15 & 43 & 75 & 49 & 45 & 53 & 72 & 50 & 10 & 75 \\ 41 & 47 & 83 & 21 & 42 & 50 & 45 & 94 & 18 & 52 & 42 & 26 & 13 & 67 \\ 32 & 31 & 68 & 30 & 37 & 8 & 12 & 27 & 43 & 22 & 61 & 12 & 25 & 65 \\ 84 & 26 & 85 & 41 & 37 & 29 & 7 & 77 & 92 & 26 & 43 & 73 & 37 & 13 \\ 91 & 8 & 45 & 2 & 29 & 89 & 45 & 73 & 95 & 36 & 51 & 7 & 75 & 34 \\ 55 & 36 & 52 & 77 & 64 & 46 & 82 & 4 & 84 & 29 & 25 & 35 & 18 & 46 \\ 64 & 23 & 61 & 12 & 83 & 26 & 37 & 45 & 67 & 1 & 4 & 34 & 58 & 44 \\ 53 & 48 & 69 & 53 & 58 & 91 & 83 & 71 & 26 & 60 & 18 & 60 & 62 & 18 \\ 19 & 25 & 14 & 60 & 33 & 88 & 39 & 94 & 14 & 81 & 85 & 83 & 30 & 70 \\ 87 & 20 & 11 & 26 & 86 & 42 & 30 & 94 & 35 & 88 & 66 & 22 & 33 & 18 \\ 4 & 91 & 5 & 13 & 32 & 66 & 8 & 51 & 11 & 37 & 6 & 27 & 32 & 18 \\ 60 & 81 & 62 & 26 & 16 & 11 & 17 & 11 & 26 & 28 & 18 & 10 & 16 & 0 \\ 27 & 74 & 41 & 30 & 95 & 18 & 91 & 10 & 10 & 36 & 15 & 74 & 65 & 83 \\ 41 & 93 & 13 & 81 & 39 & 7 & 39 & 19 & 86 & 86 & 59 & 9 & 64 & 59 \end{bmatrix} \in \mathbb{Z}/(97)^{14\times 14}.$$

Let e_i denote column i of the identity matrix of appropriate dimension. The following striped Krylov matrix corresponding to A_1 will be the identity matrix:

$$\left[K_{A_1}(e_1, 1) \middle| K_{A_1}(e_2, 1) \middle| \cdots \middle| K_{A_1}(e_{14}, 1) \right].$$

Our idea, made precise in Section 4, is to compute what we call the Krylov extension of A_1 by increasing the rank of each Krylov slice by at most one in a lexicographically maximal fashion. In this example, the Krylov extension of A_1 is $(2, 2, 2, 2, 2, 2, 2)$, corresponding to the nonsingular matrix

$$K_2 = \left[K_{A_1}(e_1, 2) \middle| K_{A_1}(e_2, 2) \middle| \cdots \middle| K_{A_1}(e_7, 2) \right]$$

$$= \left[\begin{array}{cc|c|cc|c|cc|c|cc|c|cc} 1 & 15 & 5 & 24 & 26 & 15 & 43 & 75 \\ 41 & 1 & 47 & 83 & 21 & 42 & 50 & 45 \\ 32 & 31 & 1 & 68 & 30 & 37 & 8 & 12 \\ 84 & 26 & 85 & 1 & 41 & 37 & 29 & 7 \\ 91 & 8 & 45 & 2 & 1 & 29 & 89 & 45 \\ 55 & 36 & 52 & 77 & 64 & 1 & 46 & 82 \\ 64 & 23 & 61 & 12 & 83 & 26 & 1 & 37 \\ 53 & 48 & 69 & 53 & 58 & 91 & 83 \\ 19 & 25 & 14 & 60 & 33 & 88 & 39 \\ 87 & 20 & 11 & 26 & 86 & 42 & 30 \\ 4 & 91 & 5 & 13 & 32 & 66 & 8 \\ 60 & 81 & 62 & 26 & 16 & 11 & 17 \\ 27 & 74 & 41 & 30 & 95 & 18 & 91 \\ 41 & 93 & 13 & 81 & 39 & 7 & 39 \end{array}\right].$$

Note that the slices corresponding to the basis vectors $e_8, e_9, \ldots, e_{14}$ have been annihi-

lated. Applying the similarity transformation K_2 to A_1 yields the 2-shifted form

$$A_2 = K_2^{-1} A_1 K_2 = \left[\begin{array}{ccccccc} 57 & 44 & 13 & 85 & 79 & 18 & 63 \\ 1\ 11 & 74 & 33 & 46 & 76 & 76 & 83 \\ 81 & 21 & 46 & 57 & 44 & 50 & 27 \\ 79\ 1\ 6 & 57 & 1 & 20 & 81 & 14 \\ 10 & 33 & 53 & 77 & 44 & 2 & 23 \\ 43 & 40\ 1\ 30 & 20 & 96 & 32 & 17 \\ 37 & 42 & 44 & 12 & 41 & 54 & 27 \\ 88 & 37 & 50\ 1\ 89 & 30 & 73 & 96 \\ 30 & 57 & 75 & 47 & 40 & 8 & 24 \\ 70 & 61 & 18 & 57\ 1\ 34 & 94 & 61 \\ 85 & 8 & 22 & 72 & 92 & 82 & 70 \\ 21 & 13 & 9 & 80 & 19\ 1\ 93 & 48 \\ 90 & 24 & 72 & 58 & 81 & 55 & 56 \\ 90 & 52 & 78 & 41 & 35 & 2\ 1\ 42 \end{array}\right].$$

Continuing on, the Krylov extension of A_2 is $(3, 3, 3, 3, 2)$, corresponding to

$$K_3 = \left[\, K_{A_2}(e_1, 3) \middle| K_{A_2}(e_3, 3) \middle| K_{A_2}(e_5, 3) \middle| K_{A_2}(e_7, 3) \middle| K_{A_2}(e_9, 2) \,\right]$$

$$= \left[\begin{array}{ccccccccc} 1 & 57 & & 44 & & 13 & & 85 & \\ & 1\ 11 & & 74 & & 33 & & 46 & \\ & & 81 & 1 & 21 & & 46 & & 57 & \\ & & 79 & & 1\ 6 & & 57 & & 1 & \\ & & 10 & & & 33 & 1 & 53 & & 77 & \\ & & 43 & & & 4 & & 1\ 30 & & 20 & \\ & & 37 & & & 42 & & & 44 & 1 & 12 \\ & & 88 & & & 37 & & & 50 & & 1\ 89 \\ & & 30 & & & 57 & & & 75 & & 47 & 1 \\ & & 70 & & & 61 & & & 18 & & 57 & & 1 \\ & & 85 & & & 8 & & & 22 & & 72 \\ & & 21 & & & 13 & & & 9 & & 80 \\ & & 90 & & & 24 & & & 72 & & 58 \\ & & 90 & & & 52 & & & 78 & & 41 \end{array}\right],$$

giving the 3-shifted form

$$A_3 = K_3^{-1} A_2 K_3 = \left[\begin{array}{ccccccc} 57 & 93 & 63 & 32 & 29 \\ 1\ 15 & 13 & 78 & 92 & 33 \\ 1\ 26 & 88 & 53 & 70 & 35 \\ 0 & 22 & 4 & 23 & 78 \\ 21\ 1 & 64 & 16 & 18 & 43 \\ 76 & 1\ 12 & 77 & 56 & 73 \\ 62 & 50 & 92 & 57 & 30 \\ 13 & 6\ 1 & 27 & 31 & 65 \\ 22 & 41 & 1\ 76 & 9 & 2 \\ 64 & 59 & 47 & 67 & 55 \\ 39 & 19 & 83\ 1 & 46 & 36 \\ 91 & 19 & 64 & 1\ 82 & 6 \\ 55 & 73 & 49 & 66 & 86 \\ 42 & 24 & 48 & 31\ 1\ 12 \end{array}\right].$$

From a complexity point of view it is important that A_k has only about $1/k$ times as many nonzero elements as A_1. Since $A_3 = (K_2 K_3)^{-1} A_1 (K_2 K_3)$, we could have computed A_3 directly from A_1 by constructing the single transformation

$$K_2 K_3 = \left[\, K_{A_1}(e_1, 3) \middle| K_{A_1}(e_2, 3) \middle| K_{A_1}(e_3, 3) \middle| K_{A_1}(e_4, 3) \middle| K_{A_1}(e_5, 2) \,\right],$$

but the overall cost is reduced by applying the transformation matrices sequentially. Note that the last diagonal block of A_3 only has dimension 2 because 3 does not divide 14, the dimension of the matrix.

Similar to the previous steps $A_1 \to A_2$ and $A_2 \to A_3$, the striped Krylov matrix

$$\left[\, K_{A_3}(e_1, 3) \middle| K_{A_3}(e_4, 3) \middle| K_{A_3}(e_7, 3) \middle| K_{A_3}(e_{10}, 3) \middle| K_{A_3}(e_{13}, 2) \,\right]$$

will be the identity matrix since the dimensions of the slices corresponding to the basis vectors $(e_1, e_4, e_7, e_{10}, e_{13})$ match the degree sequence $(3, 3, 3, 3, 2)$ of the diagonal blocks in A_3. The Krylov extension of A_3 is $(4, 4, 3, 2, 1)$. If the the Krylov extension is not monotonically nonincreasing or does not correspond to a square matrix the algorithm will abort. For this step, like the previous two steps, the Krylov extension satisfies these conditions and we may continue. The Krylov extension of A_3 corresponds to

$$K_4 = \left[\, K_{A_3}(e_1, 4) \middle| K_{A_3}(e_4, 4) \middle| K_{A_3}(e_7, 3) \middle| K_{A_3}(e_{10}, 2) \middle| K_{A_3}(e_{13}, 1) \,\right]$$

$$= \left[\begin{array}{cccc|cccc|c|c|c|c}
1 & & 57 & & & & 93 & & & & & \\
 & 1 & & 15 & & & 13 & & & & & \\
 & & 1 & 26 & & & 88 & & & & & \\
 & & & 0 & 1 & & 22 & & & & & \\
 & & & 21 & & 1 & 64 & & & & & \\
 & & & 76 & & & 1 & 12 & & & & \\
 & & & 62 & & & & 50 & 1 & & & \\
 & & & 13 & & & & 6 & & 1 & & \\
 & & & 22 & & & & 41 & & & 1 & \\
 & & & 64 & & & & 59 & & & & 1 \\
 & & & 39 & & & & 19 & & & & \\
 & & & 91 & & & & 19 & & & & \\
 & & & 55 & & & & 73 & & & & \\
 & & & 42 & & & & 24 & & & &
\end{array}\right].$$

As demonstrated by K_2, K_3 and K_4, it is always the case that K_{k+1} will be comprised entirely of identity vectors and columns of A_k. This follows from the fact that A_k is in k-shifted form and we are extending each Krylov slice to dimension at most $k + 1$. Applying the similarity transform K_4 to A_3 we obtain the 4-shifted form

$$A_4 = K_4^{-1} A_3 K_4 = \left[\begin{array}{c|c} \bar{A}_4 & B_4 \\ \hline C_4 & D_4 \end{array}\right] = \left[\begin{array}{cccc|cc}
 & 19 & & 21 & 72 & 56\ 69 \\
1 & & 89 & & 65 & 3 & 69\ 58 \\
 & 1 & & 55 & & 1 & 69 & 82\ 85 \\
 & & 1 & 32 & & 10 & 50 & 84\ 79 \\
 & & & 5 & & 77 & 42 & 33\ 74 \\
 & & & 14 & 1 & 9 & 73 & 78\ 73 \\
 & & & 54 & & 1 & 41 & 19 & 36\ 68 \\
 & & & 24 & & & 1 & 68 & 60 & 47\ 76 \\
 & & & 29 & & & & 45 & 6 & 8\ 32 \\
 & & & 39 & & & & 7 & 1 & 84 & 81\ 69 \\
 & & & 29 & & & & 45 & & 1 & 8 & 8\ 93 \\
\hline
 & & & & & & & & & & & 96\ 63 \\
 & & & & & & & & & & & 1\ 2\ 34 \\
 & & & & & & & & & & & 1
\end{array}\right].$$

Although the entire matrix A_4 is not in 4-shifted form, we may partition the Krylov extension $(4, 4, 3, 2, 1)$ into two parts: $(4, 4, 3)$ corresponding the the principal block $\bar{A}_4$ which is necessarily in 4-shifted form; and $[2, 1]$ corresponding to the trailing 3×3 block

D_4 that is completed. Let F_4 be the matrix obtained from D_4 by zeroing out all off-diagonal blocks B_*. For this example the Krylov extension is what we call normal: the southwest block C_4 is filled with zeroes; the trailing block D_4 is in shift-Hessenberg form;

$$\left[\begin{array}{c|c} \bar{A}_4 & B_4 \\ \hline & D_4 \end{array}\right] \qquad \left[\begin{array}{c|c} \bar{A}_4 & \\ \hline & F_4 \end{array}\right]$$

$$\left[\begin{array}{cccccc|cc} & 19 & & 21 & & 72 & 56 & 69 \\ 1 & & 89 & & 65 & & 69 & 58 \\ & 1 & & 55 & & 69 & 82 & 85 \\ & & 1 & 32 & & 50 & 84 & 79 \\ & & & 5 & & 42 & 33 & 74 \\ & & & 14 & 1 & 73 & 78 & 73 \\ & & & 54 & & 19 & 36 & 68 \\ & & & 24 & & 60 & 47 & 76 \\ & & & 29 & & 6 & 8 & 32 \\ & & & 39 & & 84 & 81 & 69 \\ & & & 29 & & 8 & 8 & 93 \\ \hline & & & & & & 96 & 63 \\ & & & & & & 1 & 2 & 34 \\ & & & & & & & & 1 \end{array}\right] \sim \left[\begin{array}{cccccc|cc} & 19 & & 21 & & 72 & & \\ 1 & & 89 & & 65 & & & \\ & 1 & & 55 & & 69 & & \\ & & 1 & 32 & & 50 & & \\ & & & 5 & & 42 & & \\ & & & 14 & 1 & 73 & & \\ & & & 54 & & 19 & & \\ & & & 24 & & 60 & & \\ & & & 29 & & 6 & & \\ & & & 39 & & 84 & & \\ & & & 29 & & 8 & & \\ \hline & & & & & & & 96 \\ & & & & & & 1 & 2 & \\ & & & & & & & & 1 \end{array}\right]; \tag{3.1}$$

and

$$\text{Frobenius}(A_4) = \text{Diag}(\text{Frobenius}(\bar{A}_4), F_4). \tag{3.2}$$

The algorithm assays that (3.1) holds by applying Theorem 2.2 to transform to the ring of polynomials and then checking some divisibility conditions. Indeed, the matrix on the left of (3.1) corresponds to the shift-Popov matrix

$$\left[\begin{array}{ccc|cc} * & * & * & 56+69x+82x^2+84x^3 & 69+58x+85x^2+79x^3 \\ * & * & * & 33+78x+36x^2+47x^3 & 74+73x+68x^2+76x^3 \\ * & * & * & 8+81x+8x^2 & 32+69x+93x^2 \\ \hline & & & -96-2x+x^2 & 63+34x \\ & & & & -1+x \end{array}\right] \in \mathbb{Z}/(97)^{5\times 5}.$$

If last diagonal entry $-1+x$ divides all polynomials above, then polynomial multiples of the last row may be added to the previous rows to zero out all entries above $-1+x$. Next we check that $-96-2x+x^2$ divides all diagonal entries above. The general case should be clear from this example. These divisibility checks are similar to step 3 of the randomized polynomial Smith normal form algorithm of Kaltofen et al. (1990).

Finally, consider the transformation to 5-shifted form if we proceed directly with A_4, shown on the left of (3.1), instead of the sparser matrix on the right of (3.1). The Krylov extension of $\bar{A}_4$ is $(5, 4, 2)$, which is normal and thus corresponds to the nonsingular 11×11 matrix $\bar{K}_5$. By augmenting $\bar{K}_5$ with I_3 we obtain the similarity transformation

K_5 of dimension 14 which may be applied to A_4.

$$K_5 = \left[\begin{array}{c||c} \bar{K}_5 & \\ \hline\hline & I_3 \end{array}\right] = \left[\begin{array}{cccccccc|cc||ccc} 1 & & 5 & & & 0 & & & & & & \\ & 1 & 64 & & & 0 & & & & & & \\ & & 1 & 50 & & 0 & & & & & & \\ & & & 1 & 76 & 0 & & & & & & \\ & & & & 74 & 1 & 0 & & & & & \\ & & & & 48 & & 1 & 0 & & & & \\ & & & & 70 & & & 1 & 0 & & & \\ & & & & 2 & & & & 1 & & & \\ & & & & 53 & & & & 0 & 1 & & \\ & & & & 88 & & & & 0 & & 1 & \\ & & & & 53 & & & & 0 & & & \\ \hline\hline & & & & & & & & & & 1 & \\ & & & & & & & & & & & 1 \\ & & & & & & & & & & & & 1 \end{array}\right].$$

Applying the transform to A_4 yields

$$A_5 = K_5^{-1} \left[\begin{array}{c|c} \bar{A}_4 & B_4 \\ \hline & D_4 \end{array}\right] K_5 = \left[\begin{array}{cccccccc|cc} & & 1 & & & 40 & 42 & & 86 & 45 \\ 1 & & & 92 & & 34 & 72 & & 59 & 92 \\ & 1 & & 10 & & 46 & 32 & & 37 & 84 \\ & & 1 & 87 & & 34 & 37 & & 84 & 39 \\ & & & 1 & 5 & 40 & 11 & & 25 & 31 \\ & & & & & 96 & 59 & & 65 & 70 \\ & & & & 1 & 4 & 54 & & 80 & 86 \\ & & & & & 1 & 91 & 6 & 33 & 28 \\ & & & & & & 1 & 4 & 75 & 16 & 10 \\ & & & & & & & & 96 & 0 & 81 \\ & & & & & & & 1 & 2 & 0 & 16 \\ \hline & & & & & & & & & 96 & 46 \\ & & & & & & & & 1 & 2 & 51 \\ & & & & & & & & & & 1 \end{array}\right] \tag{3.3}$$

There are two important points to note. On the one hand, the construction of K_5 is based entirely on the 4-shifted form $\bar{A}_4$, and did not depend on the northeast block B_4. On the other hand, the computation of A_5 did require the premultiplication of B_4 by $\bar{K}_5^{-1}$. In general, the premultiplication of offdiagonal blocks B_k by $\bar{K}_{k+1}^{-1}$ could lead to a bad complexity of the algorithm. It is for this reason that we apply the recursive formula (3.2) to compute the Frobenius form instead of proceeding as in (3.3).

In Section 4 we define precisely what we mean by the Krylov extension of a k-shifted form and give an algorithm for its computation that has cost $O(k(n/k)^\omega)$. As discussed in the example above, for the algorithm to be able to continue the Krylov extension must be *normal*, that is, must satisfy certain conditions (see Definition 4.1). In Section 6 we show how to precondition the input matrix so that all the Krylov extensions computed during the course of the algorithm will be normal with high probability.

In Section 5 we present an algorithm that takes as input a square matrix $A \in \mathsf{K}^{n\times n}$ over a field, and either returns the Frobenius form of A or reports "fail." The algorithm transform the principal block of the work matrix from k-shifted to $(k+1)$-shifted form for $k = 1, 2, \ldots, n-1$ in succession. The running time of the algorithm is bounded by $O(\sum_{k=1}^{n-1} k(n/k)^\omega)$, which can be shown to be $O(n^\omega)$ under the assumption that $\omega > 2$. In Section 7 we establish that $K = K_2 K_3 \cdots K_n$ can be computed with $O(n^\omega \log\log n)$ field operations, thus showing that a similarity transform matrix to achieve the Frobenius form can be computed in the same time.

4. Normal Krylov extension

Note that the number of non–trivial diagonal blocks in a k-shifted form $A \in \mathsf{K}^{n\times n}$ is given by $m := \lceil n/k \rceil$, and that the dimension of the trailing block is $n-(m-1)k$. If we let $v_i = e_{(i-1)k+1}$ for $1 \le i \le m$, then the block Krylov matrix

$$\Big[K_A(v_1,k)\Big|K_A(v_2,k)\Big|\cdots\Big|K_A(v_{m-1},k)\Big|K_A(v_m,n-(m-1)k)\Big] \tag{4.1}$$

will be equal to I_n.

Definition 4.1. The *Krylov extension* of a k-shifted form $A \in \mathsf{K}^{n\times n}$ with $m := \lceil n/k \rceil$ diagonal blocks is the lexicographically maximal sequence $(d_1, d_2, \ldots, d_m)$ of nonnegative integers that satisfies the following restrictions:

- $d_i \le k+1$ for all $1 \le i \le m$;
- $K = \Big[\, K_A(v_1,d_1)\Big|K_A(v_2,d_2)\Big|\cdots\Big|K_A(v_m,d_m)\,\Big]$ has full column rank;

where $v_i = e_{(i-1)k+1}$ for $1 \le i \le m$.

The Krylov extension is said to be *normal* if:

(1) $d_1 + d_2 + \cdots + d_m = n$ and $(d_1, d_2, \ldots, d_m)$ is monotonically nonincreasing;

(2) The shifted form $K^{-1}AK$ has the shape

$$K^{-1}AK = \left[\begin{array}{c|c} \bar{A} & B \\ \hline & D \end{array}\right],$$

where D is a shift-Hessenberg form (possibly of dimension zero) and $\bar{A}$ is $(k+1)$-shifted form of dimension $\bar{n} = d_1 + \cdots + d_{\bar{m}}$, where $\bar{m}$ is the minimal index such that $d_{\bar{m}} < k+1$. Moreover, for each diagonal block of D, the polynomial corresponding to that block divides all polynomials corresponding to offdiagonal blocks B_* in $K^{-1}AK$ above that diagonal block, so that

$$\left[\begin{array}{c|c} \bar{A} & B \\ \hline & D \end{array}\right] \sim \left[\begin{array}{c|c} \bar{A} & \\ \hline & F \end{array}\right],$$

where F is the matrix obtained from D by zeroing out all offdiagonal blocks B_*.

(3) $\mathrm{Frobenius}(\mathrm{Diag}(\bar{A}, F)) = \mathrm{Diag}(\mathrm{Frobenius}(\bar{A}), F)$

Condition 1, which will be checked during computation of the Krylov extension, ensures that the matrix K corresponding to the Krylov extension is nonsingular and can be used as as similarity transformation matrix. Condition 2 ensure that our recipe for recursively computing the Frobenius form can be applied. Condition 3 is slightly stronger than condition 2, and ensures that the final answer of the algorithm will indeed be in Frobenius form, and not just a block diagonal matrix that is similar to the Frobenius form. Condition 3 will be verified once for all Krylov extensions at the end of the algorithm.

We now describe an algorithm that computes the Krylov extension. Actually, the algorithm is only guaranteed to work if the Krylov extension is normal. If condition 1 of Definition 4.1 does not hold the algorithm will detect this and report failure. The idea of the algorithm is straightforward. Consider the $n \times (n+m-1)$ matrix E obtained from the matrix in (4.1) by extending the dimension of each Krylov slice from k to $k+1$, except for the last. Then E has all the columns of I_n plus an additional $m-1$ columns from

A. Recall that the *column rank profile* of E is the lexicographically smallest subsequence $(i_1, i_2, \ldots, i_n)$ of $(1, 2, \ldots, n+m-1)$ such that columns $i_1, i_2, \ldots, i_n$ have full rank.

The following result follows from Fact 2.1 by considering the shape of $K^{-1}AK$ in case the Krylov extension is normal.

Lemma 4.2. *If the Krylov extension $(d_1, d_2, \ldots, d_m)$ of a k-shifted form $A \in \mathsf{K}^{n\times n}$ is normal, then the submatrix of E comprised of the rank profile columns is equal to the matrix K of Definition 4.1.*

We next describe how to compute the column rank profile of the matrix E taking advantage of its structure.

Computing the column rank profile

For the purposes of giving a particularly simple example, suppose we are computing the Krylov extension of a 3-shifted form with degree sequence $(3, 3, 3)$, giving rise to the matrix

$$E = \begin{bmatrix} 1 & & 10 & & 20 & \\ & 1 & 11 & & 21 & \\ & & 1\ 12 & & 22 & \\ & & 13\ 1 & & 23 & \\ & & 14 & 1 & 24 & \\ & & 15 & & 1\ 25 & \\ & & 0 & & 26\ 1 & \\ & & 0 & & 27 & 1 \\ & & 0 & & 28 & & 1 \end{bmatrix} \in \mathbb{Z}/(97)^{9\times 11}.$$

The column rank profile of E can be computed using gaussian elimination, processing each column in turn, starting from the first column to the last. Processing of a column involves either determining that the column has already been zeroed out, and hence is not included in the rank profile, or performing gaussian elimination to zero entries to the right of the last non-zero entry in the column (the pivot). Processing of the first three columns consists in zeroing the coefficients to the right of the ones. After processing the fourth column the work matrix has the following shape:

$$E = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & 13\ 1 & & 25\ 66 & & \\ & & 14 & 1\ 12\ 33 & & & \\ & & 15 & & 0 & & \\ & & 0 & & & 26\ 1 & \\ & & 0 & & & 27 & 1 \\ & & 0 & & & 28 & & 1 \end{bmatrix}$$

The key observation now is that after processing columns 5 and 6, column 7 will be zeroed out and is therefore not in the rank profile. After the elimination is completed the matrix has the form

$$E = \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & 13\ 1 & & & & \\ & & 14 & 1 & & & \\ & & 15 & & 0 & & \\ & & 0 & & & 26\ 1 & \\ & & 0 & & & 27 & 1 \\ & & 0 & & & 28 & & 0 \end{bmatrix}.$$

The rank profile is thus $(1, 2, 3, 4, 5, 6, 8, 9, 10)$.

Recall that $m := \lceil n/k \rceil$ is the number of Krylov slices, and that the matrix E is obtained from I_n by extending the dimension of each Krylov slice from k to $k+1$, except for the last. Thus, there are only $m-1$ columns of E which are not known columns of I_n. To take advantage of this structure, we perform the elimination on the $n \times (m-1)$ submatrix G of E formed by the $m-1$ columns with index $k+1, 2(k+1), \ldots, (m-2)(k+1), n$. In the previous example

$$G = \begin{bmatrix} 10 & 20 \\ 11 & 21 \\ 12 & 22 \\ 13 & 23 \\ 14 & 24 \\ 15 & 25 \\ 0 & 26 \\ 0 & 27 \\ 0 & 28 \end{bmatrix}.$$

It is sufficient to keep track of the structured columns by the vector ℓ of their indices: if H is the submatrix of E formed by these n columns, then $\ell[i] = j \Leftrightarrow H_{i,j} = 1$. At the beginning of the elimination $H = I_n$, so $\ell[i] = i$ for $1 \leq i \leq n$.

Now consider the processing of the ith column of G if we include pivoting.

- The coefficients $G_{\ell[j],i}$ $\forall j \leq k \times i$ are set to zero to simulate the elimination by the corresponding structured rows to the left.
- The vector ℓ has to be updated with the permutation that may be used to find the last non zero entry (the pivot) in the current column.

The elimination on G can be performed in time $O(nm^{\omega-1})$ using LQUP decomposition Ibarra et al. (1982). The only modification is to incorporate the operations listed above into the last recursion level of the algorithm (for $m = 1$). In Algorithm 1 (`Extension`) we denote the subroutine just described by `StructuredRankProfile`. Since $m = \Theta(n/k)$ we have $nm^{\omega-1} = O(k(n/k)^{\omega})$, giving the following result.

Algorithm 1 `Extension`(A, n, k)

Require: A k-shifted form $A \in \mathsf{K}^{n \times n}$.
Ensure: The Krylov extension $(d_1, \ldots, d_m)$ of A, or fail.
`/* Fail will be returned if condition 1 of Definition 4.1 is not satisfied. Fail will not be returned if the Krylov extension is normal. */`
Form the $n \times (n+m)$ matrix E from (4.1) by extending the dimension of each Krylov slice by one.
$[j_1, \ldots, j_r] :=$ `StructuredRankProfile`(E, k).
if there exists a monotonically nonincreasing sequence $(d_1, \ldots, d_m)$ such that $[j_1, \ldots, j_r]$ is equal to $[1, \ldots, d_1,\ (k+1)+1, \ldots, (k+1)+d_2,\ \ \ldots,\ \ (m-1)(k+1)+1, \ldots, (m-1)(k+1)+1+d_m]$ **then**
  return $(d_1, \ldots, d_m)$
else
  return `fail`.
end if

Theorem 4.3. *Algorithm* `Extension` *is correct. The cost of the algorithm is* $O(k(n/k)^{\omega})$.

5. Frobenius form via arithmetic progression

Let $A \in \mathsf{K}^{n\times n}$ be a k-shifted form with a normal Krylov extension $(d_1, d_2, \ldots, d_m)$. Let K be the striped Krylov matrix associated to the extension. A key step of the algorithm is the change of basis $K^{-1}AK$. To perform this efficiently, the structure of the matrices A, K and $K^{-1}AK$ have to be taken into account.

Note that all the columns of $K^{-1}AK$ will be known columns of I_n except for the at most m columns $\{d_1, d_1 + d_2, \ldots, d_1 + d_2 + \cdots + d_m\}$. Let Y be the submatrix of K corresponding to these columns. To recover $K^{-1}AK$ we need to compute $K^{-1}AY$.

Let $*_p$ denote a permutation matrix. Up to a row and column permutations, which may be deduced from the degree sequence of diagonal blocks in A, we have

$$A = *_p \left[\begin{array}{c|c} I_{n-m} & * \\ \hline & * \end{array}\right] *_p .$$

Similarly, since K will have fewer than $\lfloor n/(k+1)\rfloor < m$ columns which are not identity vectors, and up to row and column permutations, which may be deduced from $(d_1, d_2, \ldots, d_m)$, we have

$$K = *_p \left[\begin{array}{c|c} I_{n-m} & * \\ \hline & * \end{array}\right] *_p .$$

Note that K^{-1} can be expressed similarly to K. This shows

$$K^{-1}AY = *_p \left[\begin{array}{c|c} I_{n-m} & * \\ \hline & * \end{array}\right]^{-1} *_p \left[\begin{array}{c|c} I_{n-m} & * \\ \hline & * \end{array}\right] *_p Y.$$

This gives the following result.

Lemma 5.1. *Let $K \in \mathsf{K}^{n\times n}$ be the striped Krylov matrix corresponding to the uniform Krylov extension $(d_1, d_2, \ldots, d_m)$ of a k-shifted form $A \in \mathsf{K}^{n\times n}$. There exists an algorithm* `Transform` *that takes as input $(A, k, (d_1, d_2, \ldots, d_m))$ and returns $K^{-1}AK$. The cost of the algorithm is $O(k(n/k)^\omega)$ field operations from K.*

Once $K^{-1}AK$ has been computed, the algorithm must perform some polynomial divisions to check that condition 2 of Definition 4.1 is satisfied. Since the sum of the degrees of the polynomials corresponding to the diagonal blocks is n, and the sum of the degrees of the polynomials corresponding to offdiagonal blocks above any given diagonal block is at most n, the total cost of all these division checks will be bounded by $O(n^2)$ field operations assuming standard polynomial multiplication.

Assembling these components together gives Algorithm 2 (`FrobeniusRec`) that recursively computes a block diagonal form similar to the input matrix or returns `fail`. Each recursive step corresponds to the transformation from a k-shifted form to a $k+1$-shifted form.

Theorem 5.2. *Algorithm 2 (`FrobeniusRec`) is correct. The cost of the algorithm is $O(n^\omega)$.*

Algorithm 2 `FrobeniusRec`(A, n, k)

Require: A k-shifted form $A \in \mathsf{K}^{n \times n}$.
Ensure: return a diagonal Hessenberg form similar to A, or `fail`.

```
if n = k or n = 0 then
  Return A
else
  (d_1, d_2, ..., d_m) := Extension(A, k)
  /* If the call to Extension fails then abort and return fail */
  m̄ := minimal index with d_m̄ < k + 1
  n̄ := d_1 + d_2 + ··· + d_m̄
  [ Ā | B ]
  [ C | D ] := Transform(A, k, (d_1, d_2, ..., d_m))
  if C is not the zero matrix or D is not shift-Hessenberg then
    abort and return fail
  end if
  Let F be the matrix obtained from D by zeroing out offdiagonal blocks B_*
  if [ Ā | B ]   [ Ā |  ]
     [   | D ] ≁ [   | F ] then
    abort and return fail
  end if
  Return Diag(FrobeniusRec(Ā, n̄, k + 1), F)
end if
```

Proof. The complexity is deduced from the following arithmetic progression:

$$\sum_{k=1}^{n} k(n/k)^{\omega} = n^{\omega} \sum_{k=1}^{n} (1/k)^{\omega-1} < n^{\omega} \sum_{k=1}^{\infty} (1/k)^{\omega-1} = \zeta(\omega - 1) n^{\omega} = O(n^{\omega})$$

since $\omega - 1 > 1$. □

To ensure that the algorithm will only fail with a bounded probability, the input matrix A has to be preconditioned by a random similarity transformation. This precondition also ensures that condition 3 of Definition 4.1 holds with high probability. This gives Algorithm 3 (`Frobenius`). The probability analysis of this algorithm will be detailed in Section 6; the cost of the algorithm is still $O(n^{\omega})$ field operations.

6. Preconditioning

Let $A \in \mathsf{K}^{n \times n}$ be an arbitrary matrix. In this section we prove that Algorithm 2 (`FrobeniusRec`) will not fail when given as input the tuple $(B, n, 1, x)$, where $B = V^{-1}AV$ and V is filled with algebraically independent indeterminates. Upon specialization of the indeterminates with random field elements, as is done by Algorithm 3 (`Frobenius`), a bound of $1/2$ on the probability of failure will follow due to the Schwartz / Zippel Lemma (Schwartz, 1980; Zippel, 1979).

The proof of the following theorem is similar to and inspired by (Villard, 1997, Proof of Proposition 6.1). Note that for convenience we assume that the Frobenius form of A

Algorithm 3 `Frobenius`(A, n)

Require: A matrix $A \in \mathsf{K}^{n\times n}$.
Ensure: return the Frobenius form of A, or fail.
`/* Fail will be returned with probability at most` $1/2$`. We require` $\#\mathsf{K} \geq 2n^2$`. */`
$\Lambda :=$ a subset of K with $\#\Lambda \geq 2n^2$
Choose $V \in \mathsf{K}^{n\times n}$ with entries uniformly and randomly from Λ.
$B := V^{-1}AV$ `/* If` V `is singular then abort and return fail */`
$F :=$ `FrobeniusRec`$(B, n, 1)$
if F is not in Frobenius form **then**
  abort and return `fail`
end if
Return F

has n blocks, some of which may trivial (i.e., 0×0). In the statement of the theorem this means that some of the f_* and d_* may be zero.

Theorem 6.1. *Let $A \in \mathsf{K}^{n\times n}$ have Frobenius form with blocks of dimension $f_1 \geq \cdots \geq f_n$, and let $v_1, \ldots, v_n$ be the columns of a matrix V filled with algebraically independent indeterminates. Suppose $(d_1, \ldots, d_n)$ is monotonically nonincreasing sequence of nonnegative integers. Then*

$$K = \left[K_A(v_1, d_1) \middle| \cdots \middle| K_A(v_n, d_n) \right]$$

has full column rank if and only if $\sum_{j=1}^{i} d_j \leq \sum_{j=1}^{i} f_j$ for all $1 \leq i \leq n$.

Proof. The "only if" direction follows because for any block X of i vectors, even a generic block $X = [v_1 \mid \cdots \mid v_i]$, the dimension of $\mathrm{Orb}_A(X)$ is at most $\sum_{j=1}^{i} f_i$.

To prove the other direction we specialize the indeterminates in the vectors v_i. In particular, it will be sufficient to construct a full column rank matrix

$$K = \left[K_1 \middle| \cdots \middle| K_n \right]$$

over K such that each K_i is in Krylov form and has dimension d_i, $1 \leq i \leq n$. Consider a change of basis matrix $U \in \mathsf{K}^{n\times n}$ such that $U^{-1}AU$ is in Frobenius form. Then

$$U = \left[K_A(u_1, f_1) \middle| \cdots \middle| K_A(u_n, f_n) \right]$$

is nonsingular. Let

$$\bar{K} = \left[\bar{K}_1 \middle| \cdots \middle| \bar{K}_n \right]$$

be the submatrix of U such that each $\bar{K}_i$ has the form

$$\bar{K}_i = \left[K_A(u_i, \min(f_i, d_i)) \middle| E_i \right],$$

where E_i has dimension $d_i - \min(f_i, d_i)$, and the columns of $E_1, E_2, \ldots, E_n$ are filled with unused columns of U, using the columns in order from left to right. Then $\bar{K}$ has full column rank and each $\bar{K}_i$ has the correct dimension. Our goal now is to demonstrate the existence of an invertible matrix T such that $K = \bar{K}T$ has the desired form. We will construct $T = I + \sum_{i=1}^{n}(T_i - I)$ where each T_i is unit upper triangular. For all i with

$d_i \le f_i$ no transformation of $\bar{K}_i$ is required: set $T_i = I$. If $f_i < d_i$ then

$$\bar{K}_i = \Big[\, K_A(u_i, f_i) \Big| K_A(A^{s_1} u_{j_1}, t_1) \Big| \cdots \Big| K_A(A^{s_k} u_{j_k}, t_k) \,\Big]$$

where, by construction of the E_i, we have $j_1 < j_2 < \cdots < j_k$, $t_l = f_{j_l} - s_l$ for $1 \le l \le k-1$, and $t_k \le f_k$. Using the property $\sum_{j=1}^{i} d_j \le \sum_{j=1}^{i} f_j$ we have $j_k < i$. Since $(d_1, \ldots, d_n)$ is monotonically nondecreasing and $K_A(v_l, d_l)$ is a submatrix of $\bar{K}_i$ for $1 \le l \le k$, it follows that

$$s_l \ge d_i \quad \text{for} \quad 1 \le l \le k. \tag{6.1}$$

We can write $\bar{K}_i$ as the sum of the following $k+1$ matrices:

$$\bar{K}_i = \Big[\, K_A(u_i, f_i) \Big| 0, \ldots, 0 \,\Big] \tag{6.2}$$

$$+ \sum_{l=1}^{k-1} \Big[\, 0, \ldots, 0 \Big| K_A(A^{s_l} u_{j_l}, f_{j_l} - s_l) \Big| 0, \ldots, 0 \,\Big] \tag{6.3}$$

$$+ \Big[\, 0, \ldots, 0 \Big| K_A(A^{s_k} u_{j_k}, t_k) \,\Big] \tag{6.4}$$

To bring the matrix in (6.2) to Krylov form we may add suitable linear combinations of the first f_i columns to the last $d_i - f_i$ columns to obtain

$$\Big[\, K_A(u_i, f_i) \Big| K_A(A^{f_i} u_i, d_i - f_i) \,\Big].$$

This is possible since the i^{th} invariant subspace has dimension f_i. Denote by $T_i^{(1)}$ the unit upper triangular matrix which effects this transformation on $\bar{K}$.

Now consider the matrix in (6.4). The Krylov space needs to be extended on the left to fill in the zero columns as follows:

$$\Big[\, K_A(A^s u_{j_k}, s_k - s) \Big| K_A(A^{s_k} u_{j_k}, t_k) \,\Big].$$

From (6.1) we may conclude that $s \ge 0$. Since $K_A(A^s u_{j_k}, s_k - s)$ is a submatrix of $\Big[\, \bar{K}_1 \Big| \cdots \Big| \bar{K}_{i-1} \,\Big]$, we need only copy former to latter columns. Denote by $T_i^{(2)}$ the unit upper triangular matrix which effects the copying on these columns. Similarly, there exists a unit upper triangular matrix $T_i^{(3)}$ which extends the Krylov sequence of the matrix in (6.3) to the left and right. Let $T_i = T_i^{(1)} + T_i^{(2)} + T_i^{(3)}$. $\square$

In the following corollary the matrix A and V are as in Theorem 6.1, that is, $A \in \mathsf{K}^{n \times n}$ has Frobenius form with blocks of dimension $f_1 \ge f_2 \ge \cdots \ge f_n$ and V is an $n \times n$ matrix filled with indeterminates.

Corollary 6.2. *Let $B := V^{-1} A V$ and k satisfy $2 \le k \le n$. The lexicographically maximal sequence $(d_1, d_2, \ldots, d_n)$ of nonnegative integers such that:*
- *$d_i \le k$ for all $1 \le i \le m$, and*
- *$K = \Big[\, K_B(e_1, d_1) \Big| K_B(e_2, d_2) \Big| \cdots \Big| K_B(e_n, d_n) \,\Big]$ has full column rank,*

will satisfy $d_1 + d_2 + \cdots + d_n = n$ and can be written as

$$(d_1, d_2, \ldots, d_n) = (k, k, \ldots, k, d_{\bar{m}}, f_{\bar{m}+1}, f_{\bar{m}+2}, \ldots, f_n)$$

with $k > d_{\bar{m}} \geq f_{\bar{m}+1}$. *Moreover,*

$$K^{-1}BK = \left[\begin{array}{c|c} \bar{A} & * \\ \hline & D \end{array}\right]$$

is in shifted form, where $\bar{A}$ *is in* $(k+1)$*-shifted form of dimension* $\bar{n} = d_1 + \cdots + d_{\bar{m}}$, *and* D *is in shift-Hessenberg form.*

Proof. By Theorem 6.1, and because $(d_1, d_2, \ldots, d_m)$ is chosen lexicographically maximal, the index $\bar{m}$ will be the minimal integer for which $\bar{m}k > f_1 + f_2 + \cdots + f_{\bar{m}}$, and in particular $(\bar{m}-1)k + d_{\bar{m}} = f_1 + f_2 + \cdots + f_{\bar{m}}$. But Theorem 6.1 also gives that $(\bar{m}-1)k \leq f_1 + f_2 + \cdots + f_{\bar{m}-1}$. Substituting this inequality into the previous equality shows that $d_{\bar{m}} \geq f_{\bar{m}} \geq f_{\bar{m}+1}$. Similarly, because $(d_1, d_2, \ldots, d_n)$ is chosen lexicographically maximal, the inequality of Theorem 6.1 will hold with equality for $\bar{m}+1 \leq i \leq n$, and it follows that $d_i = f_i$ for $\bar{m}+1 \leq i \leq n$. The claim about the shape of $K^{-1}BK$ now follows from Fact 2.1. □

Each entry of $VK = \left[\, K_A(v_1, d_1) \middle| \cdots \middle| K_A(v_n, d_n) \,\right]$ is a linear combination of indeterminates of V. It follows that the determinant of VK is a nonzero polynomial in the indeterminates of V with total degree at most n.

Let $K^{(1)} = I_n$ and $K^{(i)}$ be the matrix K of Corollary 6.2 for $k = i$, $2 \leq i \leq n$. Given as input $(B, n, 1)$, Algorithm 2 (`FrobeniusRec`) will perform a change of basis at each step, computing the matrix $K_{k+1} = (K^{(k)})^{-1}K^{(k+1)}$ corresponding to the Krylov extension of the principal block $\bar{A}_k$ of A_k, $i = 1, 2, \ldots, n-1$. Let Δ be the product of the determinant of V and each matrix $K^{(i)}$. Then Δ is a nonzero polynomial of total degree bounded by n^2. The next result now follows from the Schwartz / Zippel Lemma.

Theorem 6.3. *Algorithm 2 (`Frobenius`) will return fail with probability at most* $1/2$.

We remark that Corollary 6.2 captures the notion of what we might call a *generic Krylov extension*, very similar to that of *generic degree profile* for matrix minimal polynomials studied by Villard (1997). We end this section with some examples of generic Krylov extensions. The matrix used for the worked example in Section 3 had Frobenius form with blocks of dimension $[5, 4, 2, 2, 1]$, and the generic Krylov extension was as follows:

$k = 1$	$(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)$
$k = 2$	$(2,2,2,2,2,2,2)$
$k = 3$	$(3,3,3,3,2)$
$k = 4$	$(4,4,3) \oplus [2,1]$
$k = 5$	$[5,4,2,2,1]$

The generic Krylov extension for an input matrix whose Frobenius form has blocks of

dimension $[8,5,5,2,1]$ will be as follows:

$k=1$	$(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)$
$k=2$	$(2,2,2,2,2,2,2,2,2,2,1)$
$k=3$	$(3,3,3,3,3,3,3)$
$k=4$	$(4,4,4,4,4,1)$
$k=5$	$(5,5,5,5,1)$
$k=6$	$(6,6,6)\oplus[2,1]$
$k=7$	$(7,6)\oplus[5,2,1]$
$k=8$	$[8,5,5,2,1]$

Note that once some blocks have been completed, the number of blocks in the work matrix will no longer decrease.

7. Computing a similarity transformation matrix

In this section we establish the following complexity result.

Theorem 7.1. *There exists a Las Vegas algorithm that takes as input a matrix $A \in \mathsf{K}^{n\times n}$ over a field K with $\#\mathsf{K} \geq 2n^2$, and returns as output a $U \in \mathsf{K}^{n\times n}$ such that $U^{-1}AU$ is in Frobenius form. The algorithm uses an expected number of $O(n^\omega \log\log n)$ field operations.*

To establish Theorem 7.1 it will be sufficient to compute in time $O(n^\omega \log\log n)$ a matrix $W \in \mathsf{K}^{n\times n}$ such that $W^{-1}AW$ is in quasi-Frobenius form: a shift-Hessenberg form that has the same diagonal blocks as the Frobenius form of A. In particular, a matrix that transforms a quasi-Frobenius form to Frobenius form can be computed with $O(n^2)$ field operations. Suppose

$$H = \begin{bmatrix} C_{c_1} & B_{b_{12}} & \cdots & B_{b_{1k}} \\ & C_{c_2} & \cdots & B_{b_{2k}} \\ & & \ddots & \vdots \\ & & & C_{c_k} \end{bmatrix}$$

is in quasi-Frobenius form with degree sequence $(d_1, d_2, \ldots, d_k)$. Let $D_i = d_1 + d_2 + \cdots + d_{i-1}$ for $1 \leq i \leq k$. Because H is in quasi-Frobenius form, c_j divides b_{ij} and the quantity $h_{ij} := -b_{ij}/c_j$ is a polynomial for $1 \leq i < j \leq k$. Let

$$v_j = e_{D_j} + \sum_{i=1}^{j-1} B_{x^{D_i} h_{ij}} \in \mathsf{K}^{n\times 1}$$

for $1 \leq j \leq k$. Giesbrecht (1993, Section 2.3) shows that that matrix

$$T = \Big[\, K_H(v_1, d_1) \,\Big|\, K_H(v_2, d_2) \,\Big|\cdots\Big|\, K_H(v_k, d_k) \,\Big] \in \mathsf{K}^{n\times n}$$

is such that $T^{-1}HT$ will be in Frobenius form. For example, the transformation matrix T corresponding to the quasi-Frobenius form shown in (3.3) is

$$T = \left[\begin{array}{cccccc|cccc|cc|c}
1 & & & & & & 57 & & & & 55 & & 11 & & 45 \\
 & 1 & & & & & & 57 & & & 38 & 55 & 6 & 11 & 4 \\
 & & 1 & & & & & & 57 & & 86 & 38 & 72 & 6 & 27 \\
 & & & 1 & & & & & & 57 & & 86 & & 72 & 66 \\
 & & & & 1 & & & & & & & & & & \\
\hline
 & & & & & & 1 & & & & 38 & & 32 & & 7 \\
 & & & & & & & 1 & & & 22 & 38 & 81 & 32 & 59 \\
 & & & & & & & & 1 & & & 22 & & 81 & 87 \\
 & & & & & & & & & 1 & & & & & \\
\hline
 & & & & & & & & & & 1 & & & & 81 \\
 & & & & & & & & & & & 1 & & & \\
\hline
 & & & & & & & & & & & & 1 & & 46 \\
 & & & & & & & & & & & & & 1 & \\
\hline
 & & & & & & & & & & & & & & 1
\end{array}\right].$$

As this example shows, because H is quasi-Frobenius and $\deg h_{ij} \leq d_i - d_j - 1$ for $1 \leq i < j \leq k$, all vectors in the Krylov slice $K_H(v_i, d_i)$ are shifts of the first vector v_i, $1 \leq i \leq k$. Thus, to write down T we only need to compute the h_{ij}, which we noted in Section 5 can be accomplished with $O(n^2)$ field operations.

A call to Algorithm 3 (`Frobenius`) that does not return fail will compute the matrices V, $\bar{A}_k$ and K_k and the Krylov extension of $\bar{A}_k$ for $k = 2, 3, \ldots, n$, as shown in the following simplified, iterative version of the algorithm.

$A_1 := V^{-1}AV$;
for $k = 1$ **to** $n - 1$ **do**
  Write $A_k = \left[\begin{array}{c|c} \bar{A}_k & B_k \\ \hline & D_k \end{array}\right]$ with $\bar{A}_k$ in k-shifted form and D_k in shift-Hessenberg form;
  Compute the Krylov extension of the principal block $\bar{A}_k$ of A_k;
  $K_{k+1} := \mathrm{Diag}(\bar{K}_k, I) \in \mathsf{K}^{n \times n}$, the matrix corresponding to the Krylov extension;
  $A_{k+1} := K_{k+1}^{-1} A_k K_{k+1}$
od

Note that many of the K_i matrices may be equal to I_n. In any case, the matrix $K = K_2 K_3 \cdots K_n$ will be such that $(VK)^{-1}A(VK)$ is in quasi-Frobenius form. Our goal is thus to compute K. For $2 \leq k \leq s \leq n$, let $K_{k,s}$ denote the subproduct $K_k K_{k+1} \cdots K_s$. First we show how to compute $K_{k,s}$ for $s = O(k^{\omega-1})$ in time $O(n^\omega)$.

Lemma 7.2. *For $2 \leq k < n$, let $s = \min(\lceil k^{\omega-1} \rceil, n)$. Then $K_{k,s}$ can be computed with $O(n^\omega)$ field operations from K.*

Proof. Let $m := \lceil n/k \rceil$, and let $v_i = e_{(i-1)k+1}$ for $1 \leq i \leq m$. Let $(d_1, d_2, \ldots, d_m)$ be the first m elements of the lexicographically maximal sequence of Corollary 6.2. Note that $(d_1, d_2, \ldots, d_m)$ is equal to the Krylov extension of $\bar{A}_{s-1}$. Then

$$K_{k,s} = \left[K_{A_k}(v_1, d_1) \middle| K_{A_k}(v_2, d_2) \middle| \cdots \middle| K_{A_k}(v_m, d_m) \right].$$

Let $V = \left[v_1 \middle| v_2 \middle| \cdots \middle| v_m \right]$. Considering the shape of A_k and choice of v_i, we have $K_{A_k}(v_i, d_i) = K_{\mathrm{Diag}(\bar{A}_k, I)}(v_i, d_i)$, $1 \leq i \leq m$. Since each $d_i \leq k$, we may compute all

required columns of $K_{k,s}$ by computing $\mathrm{Diag}(\bar{A}_k^i, I)V$ for $i = 1, 2, \ldots, s$. Since premultiplying V by $\mathrm{Diag}(\bar{A}_k, I)$ can be done in time $O(n^\omega (1/k)^{\omega-1})$ field operations, and $s = O(k^{\omega-1})$, the overall cost is as stated. □

Now compute $K = K_{2,n}$ as follows.

$K := I_n$;
$k := 2$;
while $k \leq n$ **do**
  $s := \min(\lceil k^{\omega-1} \rceil, n)$;
  Compute $K_{k,s}$ using the algorithm of Lemma 7.2;
  $K := KK_{k,s}$;
  $k := k + s + 1$
od

The value of k increases at least as rapidly as the sequence $2^{\omega-1}$, $2^{(\omega-1)^2}$, $2^{(\omega-1)^3}$, ... and thus will exceed n after $O(\log \log n)$ iterations. Since each loop iteration has runtime $O(n^\omega)$, this shows that K may be computed in time $O(n^\omega \log \log n)$. Compute $W := VK$, the quasi-Frobenius form $H := W^{-1}AW$, and then the matrix T that transforms H to Frobenius form. Finally, set $U := WT$. Then U is such that $U^{-1}AU$ is in Frobenius form. This completes the proof of Theorem 7.1.

8. Implementation

We have implemented a slightly simplified version of the Frobenius form algorithm that computes the characteristic polynomial in a Las Vegas fashion. Algorithm 4 (`CharPolyRec`) avoids some of the divisibility checks because these are only required to certify the Frobenius form. The running time of Algorithm 4 (`CharPolyRec`) will not be improved com-

Algorithm 4 `CharPolyRec`(A, n, k, x)

Require: A k-shifted form $A \in \mathsf{K}^{n \times n}$, an indeterminate x.
Ensure: return $\det xI - A$, or `fail`.
  if $n = 0$ or $n = k$ **then**
    Return $\det(xI - A)$
  else
    $(d_1, d_2, \ldots, d_m) :=$ `Extension`(A, k)
    `/* If the call to Extension fails then abort and return fail */`
    $\bar{m} :=$ minimal index with $d_{\bar{m}} < k + 1$
    $\bar{n} := d_1 + d_2 + \cdots + d_{\bar{m}}$
    $\left[\begin{array}{c|c} \bar{A} & B \\ \hline C & D \end{array}\right]$:= `Transform`$(A, k, (d_1, d_2, \ldots, d_m))$ `/* If` C `is not the zero matrix then abort and return fail */`
    Return `CharPolyRec`$(\bar{A}, \bar{n}, k + 1, x) \times \det(xI - D)$
  end if

pared to Algorithm 2 (`FrobeniusRec`) since the divisibility checks only cost $O(n^2)$, but the change allows us to reduce the cost of the preconditioning phase by introducing

a block Krylov preconditioner. We use the dimension of the blocks as a parameter to optimize the efficiency. Since the optimal value for this parameter is highly architecture-dependant it has to be set experimentally. We present experiments comparing the computation speed in practice our implementation with, to the best our our knowledge, the two fastest existing softwares for computing the characteristic polynomial.

The implementation described here is part of the `FFLAS-FFPACK` library [1] . This C++ library provides the efficient basic routines such as matrix multiplications and LQUP decomposition that make use of the level 3 BLAS numerical routines Dumas et al. (2002, 2004), and Strassen-Winograd fast matrix multiplication algorithm.

8.1. Efficient preconditioning

Although it does not affect the asymptotic complexity, the preconditioning phase $V^{-1}AV$ of Algorithm 3 (`CharPoly`) is expensive in practice. First note that this preconditioning consists in replacing the identity vectors in the computation of the first Krylov extension of Algorithm 4 (`CharPolyRec`) by n random vectors. Our approach is to reduce the number of random vectors and combine the algorithm with a standard block Krylov method. More precisely we compute a block Krylov matrix $M = \left[U|AU|\dots|A^{c-1}U\right]$ where U is formed by $\lceil n/c \rceil$ random vectors, for a given parameter c.

- If this matrix is non singular, then the matrix $M^{-1}AM$ will be in c-shifted form (up to row and column permutations) and Algorithm 4 (`CharPolyRec`) can be called with shift parameter $k = c$ instead of $k = 1$.
- If $r = \text{rank}(M) < n$ then the first linearly independent columns of M can be completed into a non singular matrix $\overline{M}$ by adding $n - r$ columns at the end. This matrix transforms A into the block upper triangular matrix

$$\overline{M}^{-1} A \overline{M} = \begin{bmatrix} H_c & * \\ & R \end{bmatrix}$$

  where the $r \times r$ matrix H_c is in c-shifted form (up to row and column permutations). Its characteristic polynomial obtained as the product of the characteristic polynomials of the matrices H_c and R, computed recursively, with shift parameter c and 1 respectively.

Lemma 8.1 indicates how the block Krylov matrix M can be completed into a non-singular matrix $\overline{M}$.

Lemma 8.1 (Generalization of (Dumas et al., 2005, Theorem 2.1)). *Let M be a $n \times n$ matrix of rank r and (L, Q, U, P) be the LQUP decomposition of M^T. The $n \times n$ matrix $\overline{M}$ defined by*

$$\overline{M} = \left[MQ \begin{bmatrix} I_r \\ 0 \end{bmatrix} \middle| P^T \begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix} \right]$$

is non singular and its first r columns are the rank profile columns of M.

---

[1] This library is available online at `http://www-ljk.imag.fr/membres/Jean-Guillaume.Dumas/FFLAS` or within the LinBox library `http://www.linalg.org`

Proof. First note that $Q^T LQ$ is lower triangular, non singular and has the following shape

$$Q^T LQ = \begin{bmatrix} L_1 & \\ L_2 & I_{n-r} \end{bmatrix}$$

Therefore $Q^T M = Q^T LQUP$ has the generic rank profile: all its leading $i \times i$ principal minors are non zero. In particular its first r rows are the first linearly independent rows of M^T.

Let us write

$$U = \begin{bmatrix} U_1 & U_2 \\ 0 & 0 \end{bmatrix},$$

where U_1 is $r \times r$.

Considering the product

$$\begin{bmatrix} L_1 & \\ 0 & I_{n-r} \end{bmatrix} \begin{bmatrix} U_1 & U_2 \\ 0 & I_{n-r} \end{bmatrix} P = \left[\begin{array}{c} \left[I_r \ 0 \right] Q^T M \\ \hline \left[0 \ I_{n-r} \right] P \end{array} \right] = \overline{M}^T \tag{8.1}$$

one sees that $\overline{M}$ is non singular. □

Algorithm 5 (`CharPoly`) gives the algorithm with this modified preconditioning step. Again, note that only c columns of the matrix H_c have to be computed, which makes the computation of B much cheaper.

Algorithm 5 `CharPoly`(A, n, x)

Require: A matrix $A \in \mathsf{K}^{n \times n}$, an indeterminate x, a preconditioning parameter c.
Ensure: $\det(xI - A)$, or fail.

`/* Fail will be returned with probability at most 1/2 if` $\#\mathsf{K} > 2n^2$ `*/`

$\Lambda :=$ a subset of K with $\#\Lambda \geq 2n^2$

$m := \lceil n/c \rceil$

Choose $V \in \mathsf{K}^{n \times m}$ with entries uniformly and randomly from Λ.

Compute the $n \times (c\lceil n/c \rceil)$ matrix $M = \left[V | AV | \ldots | A^{c-1} V \right]$

Compute (L, Q, U, P), the LQUP decomposition of M^T. Let $r = \operatorname{rank}(M^T)$

$$\overline{M} := \left[MQ \begin{bmatrix} I_r \\ 0 \end{bmatrix} \middle| P^T \begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix} \right]$$

$$B := \overline{M}^{-1} A \overline{M} = \begin{bmatrix} H_c & * \\ & R \end{bmatrix}$$

Return `CharPolyRec`$(H_c, n, c, x) \times$ `CharPolyRec`$(R, n, 0, x)$

As c gets larger, the width of the slices of the block Krylov matrix K become smaller. In the extreme case $c = n$, the algorithm computes the usual Krylov matrix of only one vector. In this case, the algorithm is equivalent to the algorithm `LU-Krylov` presented in (Dumas et al., 2005, algorithm 2.2). Assuming $\omega = 3$ the leading term in the complexity of algorithm `LU-Krylov` is competitive ($2.667n^3$) but the algorithm does not fully exploit

matrix multiplication (as it also performs n matrix-vector products). At the opposite, the case $c = 1$ corresponds to Algorithm 3 (`CharPoly`). It fully reduces the problem to matrix multiplication, but involves more field operations: the only preconditioning phase $V^{-1}AV$ already costs $4.667n^3$ assuming $\omega = 3$. Therefore the preconditioning parameter c is used to balance the computation between these two algorithms.

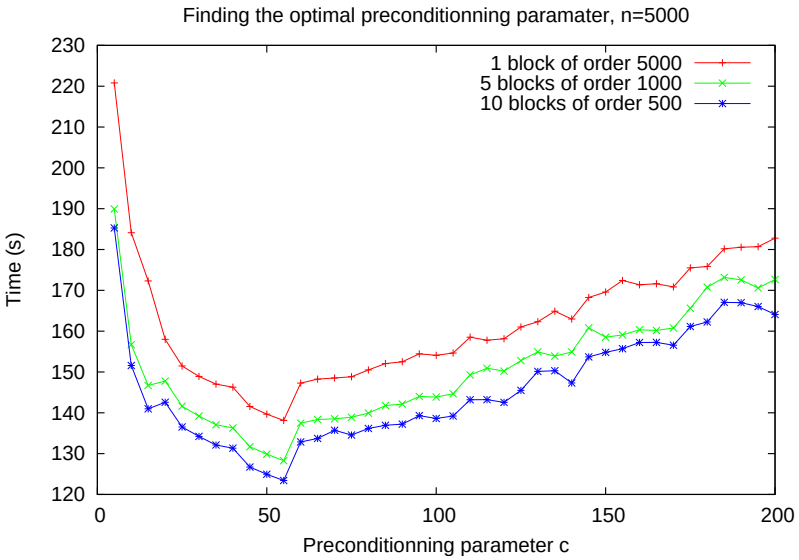

Fig. 1. Finding the optimal preconditioning parameter c for matrices of order 5000, Itanium2-64 1.3Ghz, 192Gb

Figure 1 displays the computation time of the algorithm for different values of c. Three matrices of order 5000 are used: they differ in the number of blocks in their Frobenius form. For $c < 55$, the timings are decreasing when c increases, which shows the advantage of using the block Krylov preconditioning. Then the timings increase again for larger c. In these cases, the dominant operation is the computation of the block Krylov matrix M using many matrix multiplication with rectangular dimensions. Now the matrix multiplication routine is less efficient with these dimensions: computing c $n \times n$ by $n \times n/c$ products is slower than computing one $n \times n$ by $n \times n$ product for two main reasons: worse data locality, and fewer gain of the sub-cubic algorithm. The optimal value $c = 55$ gives here the best timings. This value is not only depending on the matrix dimension, but also on the architecture and the BLAS that are used, since it is linked with the ratio between the efficiency of the matrix vector product and the matrix matrix multiplication.

Note that the algorithm gets faster as the dimension of the largest block decreases, since it reduces the total number of iterations in the algorithm.

8.2. Timing comparisons

We now compare the running time of our implementation of Algorithm 5 `CharPoly` with that of other state of the art implementations of characteristic polynomial algorithms. The routine `LU-Krylov`, available in the `FFLAS-FFPACK` and `LinBox`, libraries was shown to be the most efficient implementation in most cases Dumas et al. (2005).

For all the experiments we used the finite field $\mathbb{Z}/(547\,909)$. On one hand, the prime is large enough to ensure a high probability of success; none of the computations returned `fail`. On the other hand, the prime is small enough so that the `FFLAS-FFPACK` routines can make efficient use of the level 3 BLAS subroutines, using delayed modular reductions with the 53 bits of the `double` floating point mantissa. We used the C BLAS provided by the ATLAS-3.7 library, and the version 4.1 of the `gcc` compiler.

n	`magma-2.13`	`LU-Krylov`	`New algorithm`
100	**0.010s**	0.012s	0.016s
300	**0.20s**	0.26s	0.27s
800	4.84s	4.00s	**2.79s**
3000	243.0s	181.3s	**77.0s**
5000	1116s	854.5s	**283.3s**
10 000	9805s	6904s	**2513s**
15 000	30 597s	21 302s	**7478s**

Table 1. Computation time with random dense matrices over $\mathbb{Z}_{547909}$, Athlon-865, 1.8Ghz, 64Gb

Table 1 presents the timings for the computation of the characteristic polynomial of matrices having only one block on their Frobenius form. The preconditioning parameter c has been set to 85 for these experiments. The new algorithm improves the computation time of both `magma` and `LU-Krylov` for dimension larger than 300. This improvement factor increases with the dimension and reaches 4.09 with `magma` and 2.84 with `LU-Krylov` for $n = 15\,000$.

Figure 2 presents these timings in a log scale graph. The slopes of the two lines, which corresponds to the exponent of their complexity, are both close to 3. However, the slope of the `new algorithm` is slightly lower, indicating the effective use of sub-cubic matrix multiplication for this computation.

9. Conclusions

We have presented a randomized algorithm for computing the Frobenius form of an $n \times n$ matrix over a field that improves the worst case time complexity for this problem by a factor of $\log n$, and matches the complexity of matrix multiplication. The algorithm is randomized of the Las Vegas type and fails with a probability less than $1/2$ provided that the field has more than $2n^2$ elements. If the field is too small we can work over an extension but a better solution (currently) would be to apply an alternative algorithm such as the Frobenius form algorithm of Eberly (2000). For comparison, Eberly's Las Vegas Frobenius form algorithm has expected cost $O(n^\omega \log n)$, no restrictions on the field size, and it computes a similarity transform matrix as well as the form itself. We have established that a similarity transform matrix can be computed with an additional $O(n^\omega \log n \log n)$ field operations, but again we require the field to have at least $2n^2$ distinct elements.

The main open problem we identify is to eliminate the condition on the field size while maintaining the cost bound $O(n^\omega)$: ideally the algorithm could be derandomized

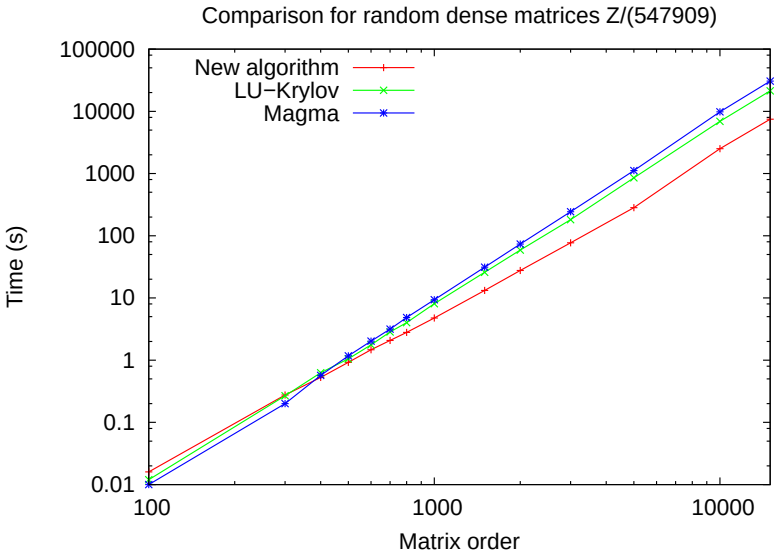

Fig. 2. Timing comparison between the new algorithm and LU-Krylov, logarithmic scales, Athlon-865 1.8Ghz, 64Gb

entirely. The currently fastest deterministic algorithm has cost $O(n^\omega(\log n)(\log\log n))$ (Storjohann, 2000, 2001).

References

Dumas, J.-G., Gautier, T., Pernet, C., 2002. Finite field linear algebra subroutines. In: Mora, T. (Ed.), Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '02. ACM Press, New York, pp. 63–74.

Dumas, J.-G., Giorgi, P., Pernet, C., 2004. Finite field linear algebra package. In: Gutierrez, J. (Ed.), Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '04. ACM Press, New York, pp. 119–126.

Dumas, J.-G., Pernet, C., Wan, Z., 2005. Efficient computation of the characteristic polynomial. In: Kauers, M. (Ed.), Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '05. ACM Press, New York, pp. 140–147.

Eberly, W., 2000. Asymptotically efficient algorithms for the Frobenius form. Tech. rep., Department of Computer Science, University of Calgary.

Gantmacher, F. R., 1990. The Theory of Matrices. Vol. 1. Chelsea Publishing Company, New York, NY.

Giesbrecht, M., 1993. Nearly optimal algorithms for canonical matrix forms. Ph.D. thesis, University of Toronto.

Giesbrecht, M., 1995. Nearly optimal algorithms for canonical matrix forms. SIAM Journal of Computing 24, 948–969.

Giesbrecht, M., Storjohann, A., 9 2002. Computing rational forms of integer matrices. Journal of Symbolic Computation 34 (3), 157–172.

Hoffman, K., Kunze, R., 1971. Linear Algebra. Prentice-Hall, Englewood Clikffs, N.J.

Ibarra, O., Moran, S., Hui, R., 1982. A generalization of the fast LUP matrix decomposition algorithm and applications. Journal of Algorithms 3, 45–56.

Kaltofen, E., Krishnamoorthy, M. S., Saunders, B. D., 1990. Parallel algorithms for matrix normal forms. Linear Algebra and its Applications 136, 189–208.

Keller-Gehrig, W., 1985. Fast algorithms for the characteristic polynomial. Theoretical Computer Science 36, 309—317.

Schwartz, J. T., 1980. Fast probabilistic algorithms for verification of polynomial identities. J. ACM 27, 701–717.

Storjohann, A., 1998. An $O(n^3)$ algorithm for the Frobenius normal form. In: Gloor, O. (Ed.), Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC '98. ACM Press, New York, pp. 101–104.

Storjohann, A., 2000. Algorithms for matrix canonical forms. Ph.D. thesis, Swiss Federal Institute of Technology, ETH–Zurich.

Storjohann, A., 2001. Deterministic computation of the Frobenius form (Extended Abstract). In: Proc. 42nd Annual Symp. Foundations of Comp. Sci. IEEE Computer Society Press, Los Alamitos, California, pp. 368–377.

Storjohann, A., Villard, G., 2000. Algorithms for similarity transforms. *extended abstract*. In: Mulders, T. (Ed.), Proc. Seventh Rhine Workshop on Computer Algebra: RWCA'00. Bregenz, Austria, pp. 109—118.

Villard, G., April 1997. A study of Coppersmith's block Wiedemann algorithm using matrix polynomials. Tech. Rep. RR 975-I-M, IMAG Grenoble France.

Zippel, R., 1979. Probabilistic algorithms for sparse polynomials. In: Proc. EUROSAM 79. Marseille, pp. 216–226.