# An $O(n^3)$ Algorithm for the Frobenius Normal Form*


Arne Storjohann

Institute of Scientific Computing
ETH Zurich, Switzerland
storjoha@inf.ethz.ch
http://www.inf.ethz.ch/personal/storjoha



**Abstract**

We describe an $O(n^3)$ field operations algorithm for computing the Frobenius normal form of an $n \times n$ matrix. As applications we get $O(n^3)$ algorithms for two other classical problems: computing the minimal polynomial of a matrix and testing two matrices for similarity. Assuming standard matrix multiplication, the previously best known deterministic complexity bound for all three problems is $O(n^4)$.


## 1 Introduction

Let $F$ be a commutative field. Every matrix $A \in F^{n\times n}$ is similar to a unique matrix $S$ in Frobenius normal form. The Frobenius form, also called the rational canonical form, has the shape

$$S = \mathrm{diag}(C_{f_1}, C_{f_2}, \ldots, C_{f_l}) = \begin{bmatrix} C_{f_1} & & & \\ & C_{f_2} & & \\ & & \ddots & \\ & & & C_{f_l} \end{bmatrix} \in F^{n\times n}.$$

Each block $C_{f_i}$ is the companion matrix of a monic $f_i \in F[x]$ and $f_i|f_{i+1}$ for $1 \le i \le l-1$. The minimal polynomial of $A$ is $f_l$ and the characteristic polynomial is $(f_1 f_2 \cdots f_l)$. The determinant of $A$ is easily recovered as the constant coefficient of $(f_1 f_2 \cdots f_l)$. Recall that the companion matrix of $g = g_0 + g_1 x + \cdots + g_{r-1}x^{r-1} + x^r \in F[x]$ is given by

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

Before stating the main result of the paper we make some comments. From [2] we know that determinant computation is as difficult as matrix multiplication. Thus, a lower bound for the cost of computing the Frobenius form is $\Omega(\mathrm{MM}(n))$ field operations where $O(\mathrm{MM}(n))$ is the number of field operations required to multiply two $n \times n$ matrices over a field. The current record for matrix multiplication [3] allows $\mathrm{MM}(n) = n^{2.376}$ whereas the standard, eminently practical algorithm has $\mathrm{MM}(n) = n^3$.

The main result of this paper is an algorithm for computing the Frobenius form over an abstract field in $O(n^3)$ field operations. This gives $O(n^3)$ algorithms for two other classical problems: computing the minimal polynomial of a matrix and testing two matrices for similarity. We are currently unable to compute in $O(n^3)$ field operations a similarity transform matrix $U$ which satisfies $UAU^{-1} = S$.

Previous Frobenius form algorithms of Ozello [8], Lüneburg [6] and Steel [9] all require $O(n^4)$ field operations in the worst case. Augot & Camion [1] give various specialized results. For example, knowing the factorization of the characteristic polynomial, the Frobenius form can be computed in $O(n^3 m)$ field operations where $m$ is the number of factors in the characteristic polynomial counted with multiplicities. In the worst case $m = n$. Over a finite field they show that $m = O(\log n)$ in the asymptotic average case. The Frobenius form algorithms in [1, 6, 8, 9] also recover a similarity transform matrix $U$ which satisfies $UAU^{-1} = S$.

Now consider randomized algorithms. Giesbrecht [5] has given a near optimal Las Vegas probabilistic algorithm which requires $O\tilde{~}(\mathrm{MM}(n)) \cdot (\log n)^{O(1)}$ field operations to recover both $S$ and a $U$. When $\#F < n^2$ the $U$ produced may have entries in a small extension field of $F$. Under the assumption of standard matrix multiplication Giesbrecht [4] develops versions of his algorithm which require $O(n^3 \log_q n)$ field operations for small fields with $q = \#F < n^2$ and $O(n^3)$ otherwise; these standard arithmetic algorithms always return a $U$ over $F$.

To summarize: the algorithm we present here improves by a factor of $O(n)$ field operation on the running time of previous deterministic algorithms [1, 6, 8, 9]. Under the assumption of standard matrix multiplication we improve by a factor of $O(\log_q n)$ field operations on the running time of the fastest randomized algorithm [4] in the small field case. On the other hand, the algorithm presented here does not produce a transforming matrix and we don't yet know how to incorporate fast matrix multiplication techniques as in [5]; these are left as open problems.

The approach we take is to first transform $A$ to Zigzag form — a matrix with an "almost" block diagonal structure and fewer than $2n$ nonzero entries. We show how to recover a Zigzag form $Z$ together with a similarity transform matrix $U$ satisfying $Z = UAU^{-1}$ in $O(n^3)$ field operations. Next


*This work has been supported by grants from the Swiss Federal Office for Education and Science in conjunction with partial support by ESPRIT LTR Project no. 20244 — ALCOM–IT.

we compute the Smith normal form of $xI-Z$ over $F[x]$ from which the Frobenius form of $A$ is easily recovered.

In Section 2 we introduce the Zigzag form. In Section 3 we recall the definition of the Smith normal form and observe the good behavior of the Smith form algorithm presented in [10] when applied to a *banded* triangular input matrix. In Section 4 we apply the results presented so far to get an $O(n^3)$ field operations Frobenius form algorithm.

## 2 The Zigzag form

A square matrix $Z$ over $F$ is in *Zigzag form* if

$$Z=\begin{bmatrix} C_{c_1} & B_{b_1} & & & & & & \\ & C^t_{c_2} & & & & & & \\ & B_{b_2} & C_{c_3} & B_{b_3} & & & & \\ & & & C^t_{c_4} & & & & \\ & & & & \ddots & & & \\ & & & & & C^t_{c_{k-2}} & & \\ & & & & & B_{b_{k-2}} & C_{c_{k-1}} & B_{b_{k-1}} \\ & & & & & & & C^t_{c_k} \end{bmatrix} \qquad (1)$$

with $k$ even, the $i$-th diagonal block a companion matrix for $i$ odd, transpose of a companion matrix for $i$ even, and blocks labeled $B_{b_i}$ having entry in the upper left corner equal to $b_i$ and all other entries zero. Each $C_{c_i}$ has $\deg c_i \geq 1$ except for $c_k$ for which we allow $\deg c_k \geq 0$; this conveniently handles the case when the actual number of companion blocks is odd. Note that the dimensions of a block labeled $B_{b_i}$ are always conformal with adjacent blocks. It is easily verified that any matrix in Zigzag form will have fewer than $2n$ nonzero entries.

We will describe an algorithm for reducing a matrix to Zigzag form using only elementary similarity transformations. First recall some definitions. An elementary row operation is one of:
- Interchanging two rows.
- Multiplying a row by an invertible element.
- Adding a multiple of a row to a different row.

The elementary column operations are defined analogously. An elementary similarity transformation of $A$ is given by $A \to EAE^{-1}$ where $E$ is the elementary matrix corresponding to an elementary row operation. For example: switching rows $i$ and $j$ is followed by switching columns $i$ and $j$; multiplying row $i$ by $a$ is followed by multiplying column $i$ by $a^{-1}$; adding $a$ time row $j$ to row $i$ is followed by subtracting $a$ times row $i$ from row $j$.

**Lemma 1** *An $A \in F^{n\times n}$ can be reduced to a similar matrix with the shape shown in (2) using at most $O(n \deg c)$ elementary similarity transformations where $c$ is found during the reduction.*

$$\left[\begin{array}{c|c} C_c & B_b \\ \hline & * \end{array}\right] \qquad (2)$$

*Proof.* There are three stages to the reduction. After stage 1, 2 and 3 the work matrix has the shape shown in (4), (5) and (2) respectively.

**Stage 1:** Using the method in [8] we reduce column $j$ of the work matrix to correct form for $j=1,2,\ldots,\deg c$ in succession. The algorithm is inductive and it is sufficient to consider a single column. After the first $j-1$ columns have been reduced the work matrix can be written as

$$\left[\begin{array}{cccc|ccc} 0 & \cdots & 0 & A[1,j] & * & \cdots & * \\ 1 & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & \ddots & 0 & A[j-1,j] & * & \cdots & * \\ & & 1 & A[j,j] & * & \cdots & * \\ \hline & & & A[j+1,j] & * & \cdots & * \\ & & & \vdots & \vdots & \ddots & \vdots \\ & & & A[n,j] & * & \cdots & * \end{array}\right]. \qquad (3)$$

Note that the input matrix can be written as in (3) with $j=1$. If the lower left block of the work matrix (3) is zero then the principal block is $C_c$ with $\deg c = j$ and we are finished this stage. Otherwise, choose $i$ with $j+1 \leq i \leq n$ and $A[i,j] \neq 0$. Perform the following (at most) $n+1$ row operations to reduce column $j$ to correct form: switch rows $i$ and $j$ so that $A[j+1,j] \neq 0$; multiply row $j+1$ by $A[j+1,j]^{-1}$ so that $A[j+1,j]=1$; add appropriate multiples of row $j+1$ to the other $n-1$ rows to zero out entries above and below entry $A[j+1,j]$ in column $j$. Directly following each of these row operations we must also perform the corresponding inverse column operation on the work matrix. It is easily verified that none of these column operations will affect the entries in the first $j$ columns of the work matrix. Since we perform this elimination process for columns $j=1,2,\ldots,\deg c$ this stage requires at most $(n+1)\deg c$ elementary similarity transformations.

**Stage 2:** At this point the work matrix can be written as

$$\left[\begin{array}{cccc|ccc} 0 & \cdots & 0 & A[1,j] & * & \cdots & * \\ 1 & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ & \ddots & 0 & A[j-1,j] & * & \cdots & * \\ & & 1 & A[j,j] & * & \cdots & * \\ \hline & & & & * & \cdots & * \\ & & & & \vdots & \ddots & \vdots \\ & & & & * & \cdots & * \end{array}\right] \qquad (4)$$

where $j = \deg c$. For $i=j,j-1,j-2,\ldots,2$ in succession, zero out entries to the right of entry $A[i,j]$ in row $i$ by adding appropriate multiples of column $i-1$ to the last $n-j$ columns. This requires at most $n-j$ column operations for each row for a total of $(n-j)(j-1)$ column operations. When working on row $i$, the corresponding inverse row operations that we must perform involve adding multiples of the last $n-j$ rows of the work matrix to row $i-1$. Because of the structure of the work matrix, the only effect these row operations can have is to change the last $n-j$ entries in the unprocessed row $i-1$. Thus, it is important that we perform the elimination in the order specified, that is, for row $i=j,j-1,j-2,\ldots,2$ in succession.

**Stage 3:** At this point the work matrix can be written as

$$\left[\begin{array}{cccc|ccc} 0 & \cdots & 0 & A[1,j] & * & \cdots & * \\ 1 & \ddots & \vdots & \vdots & & & \\ & \ddots & 0 & A[j-1,j] & & & \\ & & 1 & A[j,j] & & & \\ \hline & & & & * & \cdots & * \\ & & & & \vdots & \ddots & \vdots \\ & & & & * & \cdots & * \end{array}\right] \qquad (5)$$

where $j = \deg c$ and all entries below the first row of the upper right block are zero. If the entire upper right block is zero we are finished. Otherwise, choose $k$ with $j+1 \leq k \leq n$ and $A[1,k] \neq 0$. Perform the following (at most) $n-j+1$ column operations to complete the reduction: switch columns $j+1$ and $k$; multiply column $j+1$ by $A[1,j+1]^{-1}$; add appropriate multiples of column $j+1$ to the last $n-j-1$ columns of the matrix to zero out entries to the right of entry $A[1,j+1]$. The inverse row operations corresponding to these column operations only affect the last $n-j$ rows of the work matrix. ■

We now outline our approach for reducing an $A \in F^{n\times n}$ to Zigzag form. The key idea can be understood by considering the first few steps. First apply the algorithm of Lemma 1 to transform the work matrix to a similar matrix with shape shown in (2). Then transpose the work matrix so that it has the block lower triangular shape

$$\left[\begin{array}{c|c} C_{c_1}^t & \\ \hline B_{b_1} & * \end{array}\right]. \tag{6}$$

Our goal now is to improve the structure of the trailing block labeled $*$ to have the shape shown in (2) whilst leaving the other blocks unchanged. We claim we can accomplish this by applying the algorithm of Lemma 1 to the trailing block of (6). In particular, the proof of Lemma 1 indicates which elementary similarity transformations should be applied to the trailing $n - \deg c_1$ rows and columns of the work matrix to effect the desired transformation. It follows from the next observation that the required row operations will not change the block labeled $B_{b_1}$ since this block has no nonzero entries below the first row.

**Observation 1** *The only possibly required row operations involving row one in the algorithm given in the proof of Lemma 1 are those which add a multiple of a different row to row one.*

In general, the important point is that there be no nonzero entries below the first row in any block to the left of the trailing block.

**Theorem 1** *There exists an algorithm which takes as input an $A \in F^{n\times n}$, and returns as output a $U \in F^{n\times n}$ such that $Z = UAU^{-1}$ is in Zigzag form. The cost of the algorithm is $O(n^3)$ field operations.*

*Proof.* Initialize $U$ to be the identity matrix and $Z$ to be a copy of $A$. Perform the following steps:
[Zig:] Using the algorithm of Lemma 1 transform $Z$ to have the shape shown in (2). Apply all row operations also to $U$.
[Zag:] Transpose $Z$ and $U$. Apply the algorithm of Lemma 1 to the trailing $(n - \deg c_1) \times (n - \deg c_1)$ submatrix of $Z$. Apply all column operations also to $U$. Transpose $Z$ and $U$. At this point

$$UAU^{-1} = Z = \left[\begin{array}{c|c|c} C_{c_1} & B_{b_1} & \\ \hline & C_{c_2}^t & \\ \hline & B_{b_2} & * \end{array}\right]. \tag{7}$$

Recursively apply the Zig and Zag steps on the lower right block $*$ of $Z$ as shown in (7). Terminate when $Z$ is in Zigzag form. The cost follows from Lemma 1 and by noting that $\deg c_1 + \deg c_2 + \cdots + \deg c_k = n$. ■

## 3 Smith normal form of a banded triangular matrix

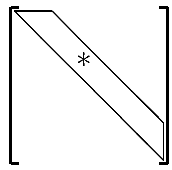
First recall some definitions about matrices over $F[x]$. Nonsingular $A, B \in F[x]^{n\times n}$ are *equivalent* if each is transformable to the other by applying a sequence of elementary (and invertible over $F[x]$) row and column operations. A nonsingular $S \in F[x]^{n\times n}$ is in *Smith normal form* if $S = \mathrm{diag}(s_1, s_2, \ldots, s_n)$ with each $s_i$ monic and $s_i | s_{i+1}$ for $i \leq i \leq n-1$. Every nonsingular $A \in F[x]^{n\times n}$ is equivalent to exactly one matrix in Smith form.

Let $A \in F[x]^{n\times n}$ be nonsingular upper triangular with degrees of off-diagonal entries bounded by $d = \deg \det A$. Using the approach in [10] we can transform $A$ to Smith form in $O(n^2d^2)$ field operations. The algorithm works in stages for $r = 1, 2, \ldots, n-1$. At the end of stage $r$ the principal $r$-th submatrix has been diagonalized and the work matrix has the shape

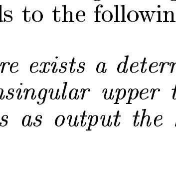
$$\left[\begin{array}{c|c} D & B \\ \hline & T \end{array}\right] \tag{8}$$

where $D$ is $r \times r$ diagonal, $T$ is the unmodified $(n-r)$-th trailing submatrix of the input matrix, and $B$ is a dense $r \times (n-r)$ matrix. The cost of completing stage $r+1$ is bounded by $O((n-r+1)d^2)$ field operations [10, Theorem 3.2]. Summing this cost for $r = 1, 2, \ldots, n$ leads to the claimed bound $O(n^2d^2)$ field operations for the entire computation.

Now consider the case when $T$ is not only upper triangular but also $k$-banded, that is, with at most the first $k-1$ off-diagonal entries in each row nonzero. The input matrix now has the shape



Up to and including stage $r$ no row operations involve rows $r+1, r+2, \ldots, n$. Also, the only column operations involving columns $r+1, r+2, \ldots, n$ are those which add a multiple of the first $r$ columns to these latter columns to reduce entries modulo the diagonal entry in the same row. It follows that at the start of stage $r+1$ the work matrix can be written as in (8) with $B$ having at most the first $k-1$ columns nonzero. For example, for $r$ approximately $n/2$ the work matrix at the end of stage $r$ and beginning of stage $r+1$ has the shape



where the dashed lines demark the principal $(r+1) \times (r+k+1)$ submatrix; all row and column operations applied during stage $r+1$ will change entries only in this principal submatrix. The cost of moving to stage $r+1$ is now bounded by $O(kd^2)$. This leads to the following.

**Proposition 1** *There exists a deterministic algorithm that takes as input a nonsingular upper triangular matrix $A \in F[x]^{n\times n}$, and returns as output the Smith normal form of*

*A. If $A$ is upper $k$-banded and degrees of off-diagonal entries in $A$ are bounded by $d = \deg \det A$, then the cost of the algorithm is $O(nkd^2)$ field operations assuming standard polynomial multiplication.*

## 4 Frobenius normal form

A fundamental theorem from linear algebra relates the Smith and Frobenius form as follows: if the Frobenius form of $Z \in F^{n\times n}$ is $\mathrm{diag}(C_{f_1}, C_{f_2}, \ldots, C_{f_l})$ then the Smith form of $xI - Z \in F[x]^{n\times n}$ is $\mathrm{diag}(1, 1, \ldots, 1, f_1, f_2, \ldots, f_l)$ and vice versa (see [7]).

The next two purely technical lemmas show how to transform the problem of computing the Frobenius form of a Zigzag form $Z \in F^{n\times n}$ to that of computing the Smith form of an upper 4-banded matrix $T \in F[x]^{k\times k}$ with $k \le n$.

**Lemma 2** *Let $c \in F[x]$ be monic of degree $r$. There exist unimodular matrices matrices $U_c, V_c \in F[x]^{r\times r}$ with $U_c$ unit upper triangular and*

$$U_c(xI - C_c)V_c = S_c = \left[\begin{array}{c|c} & c \\ \hline I_{r-1} & \end{array}\right] \in F[x]^{r\times r}. \qquad (9)$$

*Proof.* Choose $U_c$ to be the upper triangular Toeplitz matrix with $U_c[i,j] = x^{j-i}$ for $1 \le i \le j \le r$. The existence of a $V$ satisfying the lemma follows easily by noting that

$$U_c(xI - C_c) = \left[\begin{array}{c|c} & c \\ \hline -I_{r-1} & * \end{array}\right].$$

■

In the picture below we use $\bar{i}$ to indicate $k - i$.

**Lemma 3** *Let $Z \in F^{n\times n}$ be in Zigzag form as in (1). Then $xI - Z \in F[x]^{n\times n}$ is equivalent to $\mathrm{diag}(I_{n-k}, T)$ where*

$$T = \left[\begin{array}{ccccccccccc}
c_1 & & b_1 & & & & & & & & \\ \hline
 & c_3 & b_2 & & b_3 & & & & & & \\
 & & c_2 & & & & & & & & \\
 & & & c_5 & b_4 & & b_5 & & & & \\
 & & & & c_4 & & & & & & \\
 & & & & & \ddots & \ddots & & \ddots & & \\
 & & & & & & c_{\bar{3}} & b_{\bar{4}} & & b_{\bar{3}} & \\
 & & & & & & & c_{\bar{4}} & & & \\ \hline
 & & & & & & & & c_{\bar{1}} & b_{\bar{1}} & b_k \\
 & & & & & & & & & c_{\bar{2}} & \\
 & & & & & & & & & & c_k
\end{array}\right] \in F[x]^{k\times k}$$

*Proof.* Define $U = \mathrm{diag}(U_{c_1}, V_{c_2}^t, \ldots, U_{c_{\bar{1}}}, V_{c_k}^t))$ and $V = \mathrm{diag}(V_{c_1}, U_{c_2}^t, \ldots, V_{c_{\bar{1}}}, U_{c_k}^t)$ where the blocks are defined as in (9). Then $U, V \in F[x]^{n\times n}$ are unimodular and

$$U(xI - Z)V = \begin{bmatrix} S_{c_1} & B_{b_1} & & & \\ & S_{c_2}^t & & & \\ & & \ddots & & \\ & & & S_{c_{k-1}} & B_{b_{k-1}} \\ & & & & S_{c_k}^t \end{bmatrix}$$

The important point is that premultiplication by an upper triangular $U_{c_*}$ and postmultiplication by a lower triangular $U_{c_*}^t$ leaves blocks labeled $B_{b_*}$ unchanged. The above matrix can be transformed to the equivalent upper 4-banded matrix $\mathrm{diag}(I_{n-k}, T)$ by permuting the rows and columns. ■

**Theorem 2** *There exists a deterministic algorithm that takes as input an $A \in F^{n\times n}$, and returns as output the Frobenius normal form $S$ of $A$. The cost of the algorithm is $O(n^3)$ field operations using standard matrix and polynomial multiplication.*

*Proof.* Recover a Zigzag form $Z$ of $A$ in $O(n^3)$ field operations using the algorithm of Theorem 1. Then $xI - Z$ is equivalent to $\mathrm{diag}(I_{n-k}, T)$ where $T \in F[x]^{k\times k}$ is the upper 4-banded matrix defined in Lemma 3. By Proposition 1 we can recover the Smith form of $T$ (and hence the Frobenius form of $A$) in $O(n^3)$ field operations. ■

## 5 Open problems

We have given an $O(n^3)$ field operations algorithm for computing the Frobenius form $S$ of an $n \times n$ matrix $A$ over an abstract field. This result leads naturally to two open problems. First, can we recover a similarity transform matrix $U$ such that $UAU^{-1} = S$ in the same time? Second, can we incorporate fast matrix multiplication techniques? This second question is also given in [4, Open Question 1]. Finally, what about computing the Frobenius form of a sparse input matrix? See [4, Open Question 2] for details; this is an exciting problem but so far we know nothing.

**Acknowledgement** I'm grateful to Mark Giesbrecht and Gilles Villard for many useful discussions. Thanks also to the second referee.

## References

[1] Augot, D., and Camion, P. Frobenius form and cyclic vectors. *C.-R.-Acad.-Sci.-Paris-Ser.-I-Math. 318*, 2 (1994), 183–188.

[2] Baur, W., and Strassen, V. The complexity of partial derivatives. *Theoretical Computer Science 22*, 3 (1983), 317—330.

[3] Coppersmith, D., and Winograd, S. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation 9* (1990), 251–280.

[4] Giesbrecht, M. *Nearly Optimal Algorithms for Canonical Matrix Forms.* PhD thesis, University of Toronto, 1993.

[5] Giesbrecht, M. Nearly optimal algorithms for canonical matrix forms. *SIAM Journal of Computing 24* (1995), 948–969.

[6] Lüneburg, H. *On Rational Normal Form of Endomorphisms: a Primer to Constructive Algebra.* Wissenschaftsverlag, Mannheim, 1987.

[7] Newman, M. *Integral Matrices.* Academic Press, 1972.

[8] Ozello, P. *Calcul Exact Des Formes De Jordan et de Frobenius d'une Matrice.* PhD thesis, Université Scientifique Technologique et Medicale de Grenoble, 1987.

[9] Steel, A. A new algorithm for the computation of canonical forms of matrices over fields. *Journal of Symbolic Computation 24* (1997), 409—432.

[10] Storjohann, A. Computing Hermite and Smith normal forms of triangular integer matrices. *Linear Algebra and its Applications* (1998). To appear.