Journal of Complexity 31 (2015) 162–173

Contents lists available at ScienceDirect

Journal of Complexity

journal homepage: www.elsevier.com/locate/jco

# A deterministic algorithm for inverting a polynomial matrix

Wei Zhou, George Labahn *, Arne Storjohann

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

ARTICLE INFO

*Article history:*
Received 19 November 2013
Accepted 9 September 2014
Available online 28 September 2014

*Keywords:*
Polynomial matrix
Matrix inversion
Deterministic algorithm
Nearly optimal algorithm

ABSTRACT

Improved cost estimates are given for the problem of computing the inverse of an $n \times n$ matrix of univariate polynomials over a field. A deterministic algorithm is demonstrated that has worst case complexity $(n^3 s)^{1+o(1)}$ field operations, where $s \geq 1$ is an upper bound for the average column degree of the input matrix. Here, the "$+o(1)$" in the exponent indicates a missing factor $c_1(\log ns)^{c_2}$ for positive real constants $c_1$ and $c_2$. As an application we show how to compute the largest invariant factor of the input matrix in $(n^\omega s)^{1+o(1)}$ field operations, where $\omega$ is the exponent of matrix multiplication.



## 1. Introduction

We consider the problem of computing the inverse of a matrix of polynomials over an abstract field $\mathbb{K}$. Let $\mathbf{F} \in \mathbb{K}[x]^{n\times n}$ be an $n \times n$ matrix over the ring of univariate polynomials $\mathbb{K}[x]$, and let $d \geq 1$ be a bound on the degrees of entries of $\mathbf{F}$. Recall that the determinant $\det \mathbf{F}$ can have degree up to $nd$ and that the adjugate (or classical adjoint) $\det \mathbf{F} \cdot \mathbf{F}^{-1}$ is a polynomial matrix with entries of degree up to $nd$. Thus, $\mathbf{F}^{-1}$ can require on the order of $n^3 d$ field elements to represent: a factor of $n$ more than required to write down $\mathbf{F}$.

In a surprising result, Jeannerod and Villard [4] give an algorithm to compute $\mathbf{F}^{-1}$ for a generic input matrix of dimension a power of 2 that has a cost of $(n^3 d)^{1+o(1)}$ field operations from $\mathbb{K}$. Here, and in the remainder of the paper, the "$+o(1)$" in the exponent of cost estimates indicates a missing

* Corresponding author.
*E-mail addresses:* w2zhou@uwaterloo.ca (W. Zhou), glabahn@uwaterloo.ca (G. Labahn), astorjoh@uwaterloo.ca (A. Storjohann).

http://dx.doi.org/10.1016/j.jco.2014.09.004
0885-064X/

factor $c_1(\log nd)^{c_2}$ for positive real constants $c_1$ and $c_2$. The inversion algorithm of Jeannerod and Villard [4] works for arbitrary input matrices. However, the $(n^3d)^{1+o(1)}$ running time bound is obtained only for inputs that have dimension a power of 2, and which satisfy the genericity requirement that the $n^2(d+1)$ coefficients of $\mathbf{F}$ do not cause a particular polynomial of degree $n^2(d+1)$ to vanish. The genericity requirement ensures that all matrices arising during the construction have uniform row and column degrees. Jeannerod and Villard's recipe is the first essentially optimal inversion algorithm for polynomial matrices, at least for generic matrices with dimension a power of 2, improving on the previously known algorithms which have cost $(n^{\omega+1}d)^{1+o(1)}$, where $\omega$ is the exponent of matrix multiplication. More recently, an alternative inversion algorithm is given by Storjohann [9]: the algorithm is Las Vegas randomized and has expected cost $(n^3d)^{1+o(1)}$ field operations for all input matrices. For a survey of previous work on polynomial matrix inversion we refer to [4,9].

In this paper we give a simple extension of the algorithm of Jeannerod and Villard [4] which works for arbitrary input matrices while maintaining a worst case deterministic $(n^3d)^{1+o(1)}$ bound on the running time in all cases. We illustrate the differences between the algorithm of [4] and our extension using a pair of simple examples.

To understand the behaviour of the inversion algorithm [4] for generic inputs it will suffice to consider a $4\times 4$ input matrix of degree 3. In our examples we only show the degree profile of the matrices, that is, the degrees of the polynomials inside the matrix and not the polynomials themselves. Blocks of the matrix that are necessarily zero are left blank. The algorithm begins by computing a matrix $\mathbf{A}_1$ such that

$$\begin{matrix}\\ \\ \\ \\ \end{matrix}\!\!\begin{matrix}\text{degs }\mathbf{F}\\ \end{matrix}\!\!\begin{bmatrix} 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3\\ \hline 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3 \end{bmatrix}\!\!\begin{matrix}\\ \\ \\ \\ \end{matrix}\!\!\begin{matrix}\text{degs }\mathbf{A}_1\\ \end{matrix}\!\!\begin{bmatrix} 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3 \end{bmatrix} = \begin{bmatrix} 6 & 6 & & \\ 6 & 6 & & \\ \hline & & 6 & 6\\ & & 6 & 6 \end{bmatrix}.$$

The first 2 columns of $\mathbf{A}_1$ comprise a kernel basis for the last 2 rows of $\mathbf{F}$ while the last 2 columns of $\mathbf{A}_1$ comprise a kernel basis for the first 2 rows of $\mathbf{F}$. The algorithm now proceeds recursively on the two $2\times 2$ diagonal blocks of $\mathbf{F}\cdot\mathbf{A}_1$, continuing until the matrix is diagonalized. For this example two levels of recursion suffices to obtain a diagonalization $\mathbf{B}$ of the input matrix.

$$\mathbf{F}\!\!\begin{matrix}\text{degs }\mathbf{A}_1\\ \end{matrix}\!\!\begin{bmatrix} 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3\\ 3 & 3 & 3 & 3 \end{bmatrix}\!\!\begin{matrix}\\ \\ \\ \\ \end{matrix}\!\!\begin{matrix}\text{degs }\mathbf{A}_2\\ \end{matrix}\!\!\begin{bmatrix} 6 & 6 & & \\ 6 & 6 & & \\ \hline & & 6 & 6\\ & & 6 & 6 \end{bmatrix} = \begin{matrix}\\ \\ \\ \\ \end{matrix}\!\!\begin{matrix}\text{degs }\mathbf{B}\\ \end{matrix}\!\!\begin{bmatrix} 12 & & & \\ & 12 & & \\ & & 12 & \\ & & & 12 \end{bmatrix}. \tag{1}$$

By multiplying (1) on the left by $\mathbf{F}^{-1}$ and on the right by $\mathbf{B}^{-1}$ a structured decomposition is obtained for $\mathbf{F}^{-1}$. The genericity condition required for the cost analysis in [4] ensures that the property "dimension $\times$ degree $=$ $nd$" holds for all the recursive subproblems. In general, for a generic input matrix $\mathbf{F}$ of degree $d$, and dimension $n$ a power of 2, the decomposition has the form

$$\mathbf{F}^{-1} = \mathbf{A}_1\mathbf{A}_2\cdots\mathbf{A}_{\log n}\cdot\mathbf{B}^{-1}, \tag{2}$$

with $\mathbf{B} = (\det\mathbf{F})\cdot\mathbf{I}_n$ and $\mathbf{A}_{i+1}$ block diagonal with blocks of dimension $n/2^i$ and degree $2^id$, $0\le i\le \log n-1$. Thus, if $T(n,d)$ denotes the running time of the method to compute the structured decomposition on the right hand side of (2), then

$$T(n,d)\le 2\,T(n/2,2d)+(n^\omega d)^{1+o(1)}, \tag{3}$$

and it follows that $T(n,d)\in(n^\omega d)^{1+o(1)}$. Note that each of the output matrices $\mathbf{A}_1,\ldots,\mathbf{A}_{\log n},\mathbf{B}$ in the inverse decomposition requires at most $n^2(d+1)$ field elements to represent, so the total size of the output is $O(n^2d\log n)$ field elements. In [4] it is also shown that multiplying together the decomposition to obtain $\mathbf{F}^{-1}$ explicitly can be done in time $(n^3d)^{1+o(1)}$.

For a non-generic input matrix the degrees of columns in the kernel basis in the $\mathbf{A}_i$ matrices need not be uniform. Even a so-called minimal kernel basis, for which the sum of the column degrees is