Fast algorithms for computing with integer matrices: normal forms and applications

by

Stavros Birmpilis

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2021

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Claude-Pierre Jeannerod Researcher, INRIA, LIP laboratory, ENS de Lyon
Supervisors:	George Labahn Professor, Cheriton School of Computer Science, University of Waterloo
	Arne Storjohann Associate Professor, Cheriton School of Computer Science, University of Waterloo
Internal Members:	Mark Giesbrecht Professor, Cheriton School of Computer Science, University of Waterloo
	Lap Chi Lau Professor, Cheriton School of Computer Science, University of Waterloo
Internal-External Member:	William Cook Professor, Department Combinatorics and Optimization, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The main results of this thesis are based on the following papers that I have co-authored.

1. (Birmpilis, Labahn, and Storjohann, 2019): Deterministic reduction of integer nonsingular linear system solving to matrix multiplication. In Proceedings of the Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'19.

2. (Birmpilis, Labahn, and Storjohann, 2020): A Las Vegas algorithm for computing the Smith form of a nonsingular integer matrix. In Proceedings of the Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'20.

3. (Birmpilis, Labahn, and Storjohann, 2021): A fast algorithm for computing the Smith normal form with the multiplier matrices for a nonsingular integer matrix. Submitted to the Journal of Symbolic Computation.

Abstract

The focus of this thesis is on fundamental computational problems in exact integer linear algebra. Specifically, for a nonsingular integer input matrix $A \in \mathbb{Z}^{n \times n}$, we consider problems such as linear system solving and computing integer matrix normal forms.

Our goal is to design algorithms that have complexity about the same as the cost of multiplying together two integer matrices of the same dimension and size of entries as the input matrix A. If $2 \leq \omega \leq 3$ is a valid exponent for matrix multiplication, that is, if two $n \times n$ matrices can be multiplied in $O(n^\omega)$ basic operations from the domain of entries, then our target complexity is

$$(n^\omega \log ||A||)^{1+o(1)}$$

bit operations. Here $||A|| = \max_{ij} |A_{ij}|$ denotes the largest entry in absolute value, and the exponent $1 + o(1)$ indicates some missing $\log n$ and $\operatorname{loglog} ||A||$ factors.

The first contribution is solving the problem of computing the Smith normal form $S \in \mathbb{Z}^{n \times n}$ of a nonsingular matrix $A \in \mathbb{Z}^{n \times n}$ along with computing unimodular matrices $U, V \in \mathbb{Z}^{n \times n}$ such that

$$AV = US$$

in time $(n^\omega \log ||A||)^{1+o(1)}$. The algorithm we give is a Las Vegas probabilistic algorithm, which means that we are able to verify the correctness of its output.

The second contribution of the thesis is with respect to linear system solving. We present a deterministic reduction to matrix multiplication for the problem of linear system solving: given as input a nonsingular $A \in \mathbb{Z}^{n \times n}$ and $b \in \mathbb{Z}^{n \times 1}$, compute $A^{-1}b$. The system solution is computed in $(n^\omega \log ||A||)^{1+o(1)}$ bit operations.

Acknowledgements

I would like to thank all the people who made this thesis possible.

First and foremost, I want to warmly thank my supervisors George Labahn and Arne Storjohann who provided me with endless support and interesting discussions. They were always reachable and actively offering me feedback at every step. I really could not ask for more.

Moreover, all the members of the Symbolic Computation Group, each in their own unique way, contributed to making my time in the University of Waterloo particularly pleasant and creative.

In addition, it would be challenging to put into words how grateful I am towards my family. Their unconditional support and care during my graduate studies was necessary for the completion of this thesis.

Finally, I would like to give special thanks to all my friends who were by my side during this adventure (even though some of them were geographically far).

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The focus of this thesis is on fundamental computational problems in exact linear algebra. Specifically, for a nonsingular integer input matrix $A \in \mathbb{Z}^{n \times n}$, we consider problems such as linear system solving and computing integer matrix normal forms. Along the way, we provide a collection of useful computational tools which work as ingredients of our main algorithms, but could also be applied in other contexts. Exact linear algebra on integer matrices appears in a wide variety of applications including for example cryptography, number theory and group theory.

The goal is to design fast algorithms for our integer matrix problems. In particular, we wish to design algorithms that have complexity about the same as the cost of multiplying together two integer matrices of the same dimension and size of entries as the input matrix A. If $2 \leq \omega \leq 3$ is a valid exponent for matrix multiplication, that is, if two $n \times n$ matrices can be multiplied in $O(n^\omega)$ basic operations from the domain of entries, then our target complexity is

$$(n^\omega \log ||A||)^{1+o(1)}$$

bit operations. Here $||A|| = \max_{ij} |A_{ij}|$ denotes the largest entry in absolute value, and the exponent $1 + o(1)$ indicates some missing $\log n$ and $\operatorname{loglog} ||A||$ factors.

Reducing problems in computational linear algebra to matrix multiplication is important both theoretically and in practice. Strassen (1969) showed that $\omega = \log_2 7 < 2.81$ is feasible, which initiated a long series of research results that gradually decreased the upper bound on ω towards its information lower bound 2. The currently best known cost bound for the matrix multiplication exponent, by Alman and Williams (2021), ensures that any $\omega \geq 2.37286$ is feasible. Most of those theoretically fast algorithms are not practical. However, one often considers hardware and software optimized implementations which are

either cubic with $\omega = 3$ or follow Strassen's algorithm. The practical effectiveness of reducing exact linear algebra computations to matrix multplication has been convincingly demonstrated by the success of software packages such as FFLAS-FFPACK: Finite Field Linear Algebra Subroutines / Package (Dumas et al., 2008), a source code library for basic linear algebra operations over a finite field inspired by the BLAS interface (Basic Linear Algebra Subprograms), and IML: Integer Matrix Library (Chen and Storjohann, 2005), a library of C source code which implements algorithms for diophantine linear system solving.

While the ideal scenario would be to solve our problems deterministically, in some cases we can only construct randomized algorithms. For randomized algorithms, in addition to stating the running time, we also indicate the type. In order to simplify things, we will say that a Monte Carlo type algorithm is allowed to return an incorrect result with probability at most 1/2, while a Las Vegas type algorithm is allowed to report failure with probability at most 1/2, and if failure is not reported the output is certified to be correct.

Our two main problems are integer linear system solving and finding the Smith normal form of an integer matrix along with unimodular transformation matrices. We begin with the latter.

1.1 Smith normal form

Main computational problems in exact linear algebra that take as input a single nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ include computing the determinant, the Smith normal form, the characteristic polynomial, and the Frobenius normal form of A. The latter three problems are generalizations of the first.

In our case, we have focused on the computation of the Smith normal form of a nonsingular integer matrix. The problem underlying the Smith normal form is that of matrix equivalence. Any integer matrix $A \in \mathbb{Z}^{n\times n}$ is unimodularly equivalent to another matrix $B \in \mathbb{Z}^{n\times n}$ if and only if there exist unimodular (with determinant ± 1 over the $\mathbb{Z}$) matrices $U, V \in \mathbb{Z}^{n\times n}$ such that $AV = UB$. This equivalence relationship partitions $\mathbb{Z}^{n\times n}$ into disjoint equivalence classes, and we seek to determine to which equivalence class an element of $\mathbb{Z}^{n\times n}$ belongs to by computing its Smith form. In 1861, Smith showed that any nonsingular integer matrix $A \in \mathbb{Z}^{n\times n}$ is unimodularly equivalent to a unique diagonal

matrix

$$S = \begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_n \end{bmatrix},$$

where each s_i is positive and they form a divisibility sequence $s_1 \mid s_2 \mid \cdots \mid s_n$. The diagonal entries are called the invariant factors of A. Two integer matrices are equivalent if and only if they have the same Smith form. Also, since there exist unimodular matrices $U, V \in \mathbb{Z}^{n\times n}$ such that

$$AV = US,$$

it means that $|\det A| = s_1 s_2 \cdots s_n$. Notice that $s_n \mid \det A$, and so, by Hadamard's bound for $|\det A|$, the largest invariant factor can be as large as $n^{n/2}||A||^n$. The equivalence relation can also be written in the form $A = USV$ or $UAV = S$ but we choose to have it as $AV = US$.

The applications of Smith normal forms are numerous. The original purpose behind the invention of the Smith form was to find the solution of systems of linear diophantine equations. Suppose that we have the system $xA = b$ with S the Smith form of A satisfying that $AV = US$. Then, it has an integral solution if and only if the vector bVS^{-1} is integral. Another application is determining the canonical structure of abelian groups (Newman, 1997). For example, let x be a vector representing the generators of an abelian group and let A be the relation matrix such that we can express the relations among the generators by $xA = 0$. Then, the vector xU represents the new generators, and the new relations are single power relations defined by the invariant factors of A. Such a classification in turn can be used, for example, to efficiently compute Gröbner bases of ideals invariant under the action of an abelian group (Faugère and Svartz, 2013). Other applications include integer programming (Hu, 1969), system theory (Kailath, 1980), and the study of symplectic spaces (Chandler et al., 2010).

Outside of the domain of integers, another application of the Smith form occurs with respect to similarity. Two $n \times n$ matrices A, B over a field F are similar if and only if there exists a nonsingular $n \times n$ matrix T over F such that $B = TAT^{-1}$. We can answer that question since A and B are similar over F if and only if $xI - A$ and $xI - B$ are equivalent over $\mathsf{F}[x]$ (Newman, 1972). For $A \in \mathbb{Z}^{n\times n}$, the Smith form of $xI - A$ over $\mathbb{Q}[x]$ gives the factorization of $\det(xI - A)$, the characteristic polynomial of A, into monic integer polynomials. Note that the constant coefficient of $\det(xI - A)$ equals $\det A$ in absolute value.

Previous work

The simplest algorithm for computing the Smith normal form uses the Extended Euclidean Algorithm in order to eliminate the off-diagonal entries in the input matrix. However, such algorithms are known to suffer from intermediate expression swell, namely, the intermediate integers during the process can become very large. For example, Hafner and McCurley (1991) give an example of a 20×20 input matrix with only single-digit entries which produces numbers of more than 5000 digits during the standard process of first going to a triangular matrix. To put that into perspective, Hadamard's inequality bounds the number of digits of the determinant of such a matrix by 34.

During the past forty years, there has been a considerable progress on algorithms for computing the Smith form of an integer matrix. The following table summarizes the most important results. The time complexity is given without the extra $\log n$ and $\mathrm{loglog}\,||A||$ factors. The last column gives the type of the algorithms which is either deterministic (Det), Monte Carlo (MC) or Las Vegas (LV).

Citation	Time complexity	Type
Kannan and Bachem (1979)	$poly(n, \log\|\|A\|\|)$	Det
Iliopoulos (1989a)	$n^5(\log\|\|A\|\|)^2$	Det
Hafner and McCurley (1991)	$n^5(\log\|\|A\|\|)^2$	Det
Storjohann (1996)	$n^{\omega+1}\log\|\|A\|\|$	Det
Eberly, Giesbrecht, and Villard (2000)	$n^{2+\omega/2}\log\|\|A\|\|$	MC
Kaltofen and Villard (2004)	$n^{2.695591}\log\|\|A\|\|$	MC
Chapter 5	$n^{\omega}\log\|\|A\|\|$	LV

Table 1.1: List of work on the Smith form problem over the integers.

The first algorithm, indicated in Table 1.1, proven to run in polynomial time, was given in 1979 by Kannan and Bachem (1979). They managed to successively control the size of the entries of the integer matrix after each elimination step. Other early solutions to the problem, by Iliopoulos (1989a) and Hafner and McCurley (1991), achieved a running time bounded in $(n^5(\log||A||)^2)^{1+o(1)}$ by using the idea that the Smith form of A can be computed modulo the determinant of A. The next substantial improvement in complexity was by Storjohann (1996, 2000b) who computed the Smith form in $(n^{\omega+1}\log||A||)^{1+o(1)}$ by reducing the exponent of n from 5 down to 4, and by incorporating matrix multiplication.

The previously fastest algorithm for Smith form — and the currently fastest algorithm for characteristic polynomial and Frobenius form — is given by Kaltofen and Villard

(2004). They give a Las Vegas algorithm for computing the characteristic polynomial in time $(n^{3.2} \log ||A||)^{1+o(1)}$ assuming $\omega = 3$, and in time $(n^{2.695591} \log ||A||)^{1+o(1)}$ assuming the currently best known upper bound for $\omega < 2.37286$ by Alman and Williams (2021) and the best known bound for rectangular matrix multiplication by Le Gall and Urrutia (2018). Using their characteristic polynomial algorithm together with ideas of Giesbrecht (2001) and Storjohann (2000a) they obtain a Monte Carlo algorithm for the Smith form and Frobenius form, respectively, with the same running time.

A simpler problem than Smith form is to compute only the largest invariant factor s_n of A, that is, the smallest positive integer that clears denominators of $A^{-1} \in \mathbb{Q}^{n \times n}$. Combining the randomized approach of Eberly et al. (2000, Theorem 2.1) with fast linear system solving gives an algorithm that achieves the target complexity, but in a Monte Carlo fashion: the algorithm might produce a proper divisor of s_n. Furthermore, Eberly, Giesbrecht, and Villard (2000) showed that also the ith invariant factor can be computed by means of random perturbations. By employing binary search, all distinct invariant factors can be found in time $(n^{2+\omega/2} \log ||A||)^{1+o(1)}$.

Our result

In Chapter 5, we develop a probabilistic algorithm that computes the Smith normal form of a nonsingular matrix A within the target complexity, that is, using $(n^{\omega} \log ||A||)^{1+o(1)}$ bit operations. The algorithm we give is the first that matches, up to logarithmic factors, the cost of matrix multiplication. In addition, it is also the first that certifies, in a Las Vegas fashion, the correctness of the Smith form S. The result has appeared in (Birmpilis, Labahn, and Storjohann, 2020), and it was awarded an ACM/SIGSAM ISSAC Distinguished Student Author Award.

Moreover, using the same algorithm, we obtain a very useful object that we call the Smith massager and that we extensively discuss in Chapter 4. The Smith massager is a matrix $M \in \mathbb{Z}^{n \times n}$ which, along with the Smith form, gives a fraction-free representation of the fractional part of the inverse of the input matrix A. In Chapter 6, the object will prove to be of further use by being the main ingredient for computing Smith multiplier matrices.

1.2 Smith multipliers

Consider the linear system solving problem of the type

$$xA = b, \tag{1.1}$$

that is, given a matrix $A \in \mathbb{Z}^{n\times n}$ and a row vector $b \in \mathbb{Z}^{1\times n}$, find a row vector $x \in \mathbb{Q}^{1\times n}$ such that $xA = b$. The problem becomes trivial if matrix A is diagonal.

For any integer matrix $A \in \mathbb{Z}^{n\times n}$, there are unimodular matrices $U, V \in \mathbb{Z}^{n\times n}$ which describe the set of invertible integer row and column operations which transform A into its Smith form S or vice versa. These row and column operations are typically defined as satisfying matrix equations in the form $USV = A$ or $A = USV$. However, in our case, it will be convenient to specify the Smith form multipliers U, V as unimodular matrices satisfying

$$AV = US. \tag{1.2}$$

The form in (1.2) is useful, for example, to transform the linear system in (1.1) to

$$\bar{x}S = \bar{b}$$

with $\bar{x} = xU$ and $\bar{b} = bV$. Since S is in Smith form, the new system allows for easier determination of possible properties of the solutions. For example, the denominator of x, the vector solving $xA = b$, will be the same as the denominator of $\bar{x} = S^{-1}\bar{b}$.

The above example gives an application where both the Smith form and its unimodular multipliers are needed. These multipliers are also needed in a number of other settings. For example, when one not only wants the classification of a finite abelian group into the direct sum of its cyclic components, but also the isomorphism which takes the group to the direct sum of cyclic factors. Similarly, if two integer matrices are row and column equivalent then the Smith form with multipliers provides the matrices which specify the row and column equivalence. Hubert and Labahn (2016) require both the Smith form and its multipliers when one looks for possible rational symmetry by a finite abelian group action for a set of polynomials equations along with determining the rational invariants and rewriting rules of such an action. Other applications which make use of the Smith multipliers include determining lattice rules for quadrature formulas over the unit cube (Lyness and Keast, 1995), its use in chip-firing for finite connected graphs in combinatorics (Stanley, 2016), and many more.

Previous work

Although algorithms which solve the Smith form with multipliers problem for a nonsingular integer matrix have a long history, efficient methods are still not common. The original paper by Smith (1861) included an algorithm modeled on Gaussian elimination where greatest common divisors and the associated solutions of linear diophantine equations replace division. Bradley (1970, 1971) later extended these via algorithms to compute the greatest common divisors of more than two integers. However, these early algorithms all encountered rapid growth of intermediate computations since they needed to retain the row and column operations. One issue is that, although the Smith form is unique, the same cannot be said for the unimodular multipliers. As such, methods like homomorphic imaging followed by Chinese Remaindering do not appear to be applicable.

Citation	Time complexity	Type
Kannan and Bachem (1979)	$poly(n, \log \|\|A\|\|)$	Det
Iliopoulos (1989b)	$n^5(\log \|\|A\|\|)^2$	Det
Storjohann (2000b)	$n^{\omega+1} \log \|\|A\|\|$	Det
Chapter 6	$n^{\omega} \log \|\|A\|\|$	LV

Table 1.2: List of work on the Smith form with multipliers problem over the integers.

The first algorithm, indicated in Table 1.2 to compute the Smith form with multipliers in polynomial time originated with Kannan and Bachem (1979). Later, Iliopoulos (1989b) was able to use matrix multiplication and prove running time for the problem in $(n^5(\log ||A||)^2)^{1+o(1)}$. Storjohann (1997, 2000b) was the first to consider the problem of space-efficient unimodular multipliers for Smith computation and compute them in $(n^{\omega+1} \log ||A||)^{1+o(1)}$. Finally, Jäger (2005) followed this up with a procedure that reduces one of the unimodular multipliers using LLL reduction at the expense of the other.

Our result

In Chapter 6, we give an efficient algorithm which allows us to compute S, U and V in a Las Vegas fashion. The algorithm uses $(n^{\omega} \log ||A||)^{1+o(1)}$ bit operations and it is much faster that the previous state of the art which had an extra factor of n. As we already have a fast way to compute the Smith form S, our goal is to design an efficient extension that also returns the unimodular matrices U and V. Previously, determining the Smith form

alone had been considered easier than determining the Smith form and its multipliers. We show that finding the multipliers can also be done in the same time complexity as required by our previous algorithm to find only the Smith form. This result has been submitted for publication to the Journal of Symbolic Computation.

The main tool that we will employ to efficiently compute Smith multipliers will be the Smith massager matrix that we compute using the Smith form algorithm in Chapter 5. We show that by perturbing a Smith massager M by a random matrix scaled by the Smith form S, we are able to construct a Smith multiplier using fast Hermite normal form computation. Efficiency is guaranteed by the fact that the Hermite form has many ones on its diagonal.

1.3 Linear system solving

Another classical linear algebra problem is that of computing a rational solution vector $A^{-1}b \in \mathbb{Q}^{n\times 1}$, where $A \in \mathbb{Z}^{n\times n}$ is a nonsingular integer matrix and $b \in \mathbb{Z}^{n\times 1}$ an integer vector. Linear system solving is a fundamental problem which ubiquitously arises in science or engineering. For example, in the field of cryptography, there is very often the need to solve large systems of equations involving very large integers. In addition, there exist multiple situations where it is desirable to obtain exact solutions. For example, in problems of number theory and in cases of ill-conditioned inputs where floating point calculations will be unsuccessful.

One of the challenges of attaining the target complexity for this problem comes from the size of the inverse A^{-1} and the solution $A^{-1}b$. By Cramer's rule, the bitlength of the numerators and denominators of entries in $A^{-1}b$ is about n times the bitlength of entries in A.

Previous work

The first important result on the problem was the X-adic linear lifting algorithm of Dixon (1982), which achieves an expected running time of $(n^3 \log ||A||)^{1+o(1)}$ bit operations. The algorithm computes the solution $A^{-1}b$ modulo some power X^m large enough such that the exact solution can be recovered using rational reconstruction. The number X is called the *lifting modulus*, and it must be relatively prime to $\det A$. The technique is called lifting since it proceeds in steps, where each step takes the solution modulo X^i and returns a "lifted" solution modulo X^{i+1}. The arithmetic operations at each step handle integers with about

the same bitlength as that of the entries in the input matrix A. The essence of linear lifting is that instead of solving one linear system at full precision, the algorithm computes initially $A^{-1} \bmod X$ and uses it to solve m linear systems of the form $A^{-1}b_i \bmod X$, where b_0 is the original column vector and $b_1, b_2, \ldots$ are the residues produced by the previous lower precision systems. The final solution would be given by its X-adic expansion modulo p^m, that is,

$$A^{-1}b = c_0 + c_1X + \cdots + c_{m-1}X^{m-1}.$$

Linear X-adic lifting is deterministic once a lifting modulus X with $X \perp \det A$ is known. Unfortunately, the only known method to find a lifting modulus within the alotted time is to use randomization: let X be the power of a prime that is randomly chosen in a range where fewer than half of the primes selected can possibly divide $\det A$. A second issue with linear lifting is that each iteration increases the precision only by one, and that the final expansion requires m coefficients. On the other hand, there is also quadratic lifting (or Newton iteration) which is a technique very similar to linear lifting, but it manages to double the precision at each iteration. The downside of quadratic lifting is that it requires $A^{-1} \bmod X$, $A^{-1} \bmod X^2$, $A^{-1} \bmod X^4, \ldots$ to be explicitly computed at each step. The space needed to write down these intermediate objects exceeds our target cost, which is $(n^\omega \log \|A\|)^{1+o(1)}$ for any $2 \leq \omega \leq 3$.

The next significant complexity improvement was improving from 3 to ω in the exponent of n. The high-order lifting technique presented by Storjohann (2003) over the polynomials computes only a critical high-order component of the expansion, and it allows the incorporation of matrix multiplication to reduce the expected running time to $(n^\omega \log ||A||)^{1+o(1)}$. Storjohann (2005) extended his technique over the integers by introducing randomization with the invention of the shifted number system. In particular, over the integers there is the phenomenon of carry propagation, which does not allow the application of an arithmetic operation only to the high-order component of an integer. The shifted number system technique provided a reduction of rational system solving to matrix multiplication in a Las Vegas fashion.

Subsequently, the shifted number system was replaced by the double-plus-one lifting technique of Pauderis and Storjohann (2012). The article gives a deterministic high-order lifting algorithm for linear system solving in $(n^\omega \log ||A||)^{1+o(1)}$ assuming that there exists a lifting modulus X relatively prime to $\det A$ like in Dixon's algorithm. The algorithm is the most efficient for the problem if we know such an X for the input matrix A. Nonetheless, if a suitable lifting modulus X is not known, then the method is still of type Las Vegas since the lifting modulus X is chosen randomly as described above.

Our result

In Chapter 7, we present a deterministic reduction to matrix multiplication for the problem of rational system solving. As mentioned above, the previously known algorithms (Storjohann, 2005; Pauderis and Storjohann, 2012) with running time within our cost, required randomization to find an integer modulus $X \perp \det A$. Our strategy is based on computing the minimal integer t such that all denominators of the entries in $2^t A^{-1}$ are relatively prime to 2. In that case, for an integer vector b having entries with bitlength $O(n)$ times as large as the bitlength of entries in A, our algorithm also produces the 2-adic expansion of $2^t A^{-1} b$ up to a precision high enough such that $A^{-1}b$ over $\mathbb{Q}$ can be recovered using rational number reconstruction. The result has appeared in (Birmpilis, Labahn, and Storjohann, 2019).

1.4 Other contributions

The contributions introduced in Sections 1.1-1.3 and developed in Chapters 4-7 require the use of fast algorithms for linear algebra that have been developed over the past five decades. In Chapter 2, we gather together results related to system solving and we give an updated treatment using the latest technology. The chapter, using the double-plus-one lifting construction of Pauderis and Storjohann (2012), gives a thorough description and analysis of linear system solving, that is, for a nonsingular $A \in \mathbb{Z}^{n\times n}$ and a $B \in \mathbb{Z}^{n\times m}$, it computes the system solution $A^{-1}B$ up to some magnitude X^d where X is a lifting modulus satisfying $X \perp \det A$. Furthermore, using the updated linear system solving algorithm, we present a version of integrality certification, that is, given an integer $s \in \mathbb{Z}_{>0}$ along with A, B, we check whether s is a multiple of the denominator of $A^{-1}B$ and, if yes, actually compute the fractional part of $A^{-1}B$. Algorithms for both of these problems that are within our target complexity have been given by Storjohann (2005), but they require the use of the randomized shifted number system. The algorithms presented in Chapter 2 are slightly improved in terms of complexity, and the only randomization now required is the choice of a small prime that does not divide A.

In Chapter 3, we present a partial linearization technique which will allow us to extend the algorithms from Chapter 2 so that their cost estimates depend on the average bitlength of the entries of the input matrix and not on the entry with largest bitlength. The construction has been developed for polynomial matrices by Gupta et al. (2012, Section 6), and we adapt it to the case of integer matrices. The technique allows us to transform an input matrix A into a new matrix D which can be used in place of A for all of the algo-

rithms presented in Chapter 2 that involve A^{-1}, the Smith form algorithm of Chapter 5, and many more.

1.5 Cost model

In this section, we give definitions and assumptions regarding the cost model that we will follow in this thesis.

We begin by defining the function $\mathsf{M} : \mathbb{Z}_{>0} \to \mathbb{R}_{>0}$ such that integers bounded in magnitude by 2^d can be multiplied using at most $\mathsf{M}(d)$ bit operations. The recent algorithm of Harvey and Van der Hoeven (2021) allows for

$$\mathsf{M}(d) \in O(d \log d).$$

Moreover, we will assume that

$$\mathsf{M}(a) + \mathsf{M}(b) \leq \mathsf{M}(a+b) \quad \text{and} \quad \mathsf{M}(ab) \leq \mathsf{M}(a)\mathsf{M}(b) \tag{1.3}$$

for $a, b \in \mathbb{Z}_{\geq 2}$, as per von zur Gathen and Gerhard (2013, Section 8.3), where you can find a further discussion about integer multiplication.

In addition, following the approach of previous authors (Storjohann, 2005; Gupta et al., 2012; Neiger and Pernet, 2021), it will be useful to define an additional function $\mathsf{B} : \mathbb{Z}_{>0} \to \mathbb{R}_{>0}$ for bounding the cost of integer gcd-related computations. Then, the extended Euclidean algorithm with two integers bounded in magnitude by 2^d, and the rational number reconstruction problem from von zur Gathen and Gerhard (2013, Section 5.10) with modulus bounded by 2^d, can be solved with $O(B(d))$ bit operations (Schönhage, 1971). This means that $\mathsf{B}(d) \in O(\mathsf{M}(d) \log d)$ if pseudo-linear integer multiplication is used, or $\mathsf{B}(d) \in O(\mathsf{M}(d))$ if $\mathsf{M}(d) \in \Omega(d^{1+\epsilon})$ for some $\epsilon > 0$. The assumptions on M given in (1.3) also apply to B.

The motivation behind our cost model is that almost all of our algorithms are reduced to a number of matrix multiplications with integer matrices that have size similar to the size of the input matrix in the overall problem. Therefore, cost estimates are given in terms of a multiplication time $\mathsf{M}(d)$ and the matrix multiplication exponent $\omega \leq 3$ such that two $n \times n$ integer matrices with entries bounded by 2^d can be multiplied in

$$O(n^\omega \, \mathsf{M}(d))$$

bit operations. Furthermore, we will assume that $\omega > 2$.

Finally, since $\omega > 2$ and to simplify cost estimates, we make the assumption that $\mathsf{M}(d) \in O(d^{\omega-1})$. This assumption simply stipulates that if fast matrix multiplication techniques are used, then fast integer multiplication techniques should also be used. The same assumption applies also to B.

Before we move on, let us mention that, only in Chapter 7, we plan to use a slightly more general cost model which will still be along the lines of this one, that is, counting the number of matrix multiplications. The main difference will be that, in Chapter 7, we do not assume that the complexity of matrix multiplication is given in terms of the matrix multiplication exponent ω, but allow the cost to be described by any function that depends on the dimension n and the magnitude 2^d. The reason that we don't assume that more general cost model everywhere is that it can be more restrictive in simplifying cost estimates and using the function B.

Chapter 2

Computational Tools

In this chapter, we present some fundamental computational tools that we will need throughout the rest of the thesis in order to establish the complexity bounds of our main algorithms.

We denote by $\mathbb{Z}/(X) = \{0, 1, \ldots, X-1\}$ the ring of integers modulo X for any positive integer X. In addition, for any rational number α that has denominator relatively prime to a positive integer X, we let $\mathrm{Rem}(\alpha, X)$ denote the unique integer obtained by reducing α modulo X, that is, $\mathrm{Rem}(\alpha, X) \equiv \alpha \bmod X$ and $\mathrm{Rem}(\alpha, X) \in \mathbb{Z}/(X)$.

The main computational tool that we need is fast nonsingular system solving. Given a nonsingular $A \in \mathbb{Z}^{n \times n}$ and matrix $B \in \mathbb{Z}^{n \times m}$, together with a lifting modulus $X \in \mathbb{Z}_{>0}$ that satisfies $X \perp \det A$ and $\log X \in O(\log n + \log \|A\|)$, the linear system solution expansion problem is to compute

$$\mathrm{Rem}(A^{-1}B, X^d)$$

for a given precision d. The second problem that we study is integrality certification. Given an $s \in \mathbb{Z}_{>0}$ in addition to B, determine whether $sA^{-1}B$ is integral, and, if so, return the matrix

$$\mathrm{Rem}(sA^{-1}B, s).$$

Provided the "dimension $\times$ precision $\leq$ invariant" compromises $m \times d \in O(n)$ and $m \times (\log \|B\| + \log s) \in O(n \log X)$ hold, our target complexity for solving these problems is

$$O(n^\omega \, \mathsf{M}(\log n + \log \|A\|) \log n) \tag{2.1}$$

bit operations.

The earlier algorithms of Storjohann (2005) for these problems do match our target complexity, except that: (a) the running time bound has an extra $O(n^\omega \mathsf{B}(\log n + \log \|A\|))$ term; (b) the algorithms require randomization. The reason for (a) is that when X is composite with an unknown factorization, a more general algorithm for the computation of $\mathrm{Rem}(A^{-1}, X)$ is required. We show how to avoid the $O(n^\omega \mathsf{B}(\log n + \log \|A\|))$ term by choosing X to be the power of a small random prime. Furthermore, our main contribution is that assuming that we have an X such that $X \perp \det A$, we design deterministic algorithms for linear system solving and integrality certification within our target complexity.

Section 2.1 shows how to choose X as the power of a small random prime. Section 2.2 recalls the double-plus-one lifting algorithm of Pauderis and Storjohann (2012) which forms the basis of the linear solving and integrality certification algorithms. Sections 2.3 and 2.4 present the linear solving and integrality certification algorithms, respectively, to work with an X as chosen in Section 2.1.

Finally, Section 2.5 reduces some kinds of unbalanced matrix multiplication, that is, with matrices whose entries' size is skewed, to general matrix multiplication.

2.1 Lifting initialization

If C is a bound for $|\det A|$, then a prime p chosen uniformly and randomly in the range

$$6 \log C < p < 12 \log C$$

will satisfy $p \perp \det A$ with probability at least $1/2$ by von zur Gathen and Gerhard (2013, Theorem 18.10). We can check whether $p \perp \det A$ by trying to compute an LUP decomposition of $A \bmod p$ over $\mathbb{Z}/(p)$. If $p \perp \det A$, then we can choose our lifting modulus X to be a power of p.

Lemma 2.1. *There exists a Las Vegas algorithm that takes as input a nonsingular $A \in \mathbb{Z}^{n \times n}$, and returns as output an integer X that satisfies*

- *X is the power of a prime p with $\log p \in \Theta(\log n + \mathrm{loglog} \|A\|)$,*
- *$X \perp \det A$,*
- *$X \geq \max(10000, 3.61 n^2 \|A\|)$, and*
- *$\log X \in O(\log n + \log ||A||)$.*

The cost of the algorithm is $O(n^\omega \,\mathsf{M}(\log n + \log \|A\|))$ *bit operations.*

Proof. By Hadamard's bound we have $C := n^{n/2}\|A\|^n \geq |\det A|$. By von zur Gathen and Gerhard (2013, Theorem 18.10), choosing a random prime p as described above can be done within the allotted time. Working over $\mathbb{Z}/(p)$, use $O(n^\omega \,\mathsf{M}(\log p) + n\,\mathsf{B}(\log p))$ bit operations to compute an LUP decomposition (Aho et al., 1974, Section 6.4) of $\mathrm{Rem}(A, p)$. The $n\,\mathsf{B}(\log p)$ term in this cost estimate is for inverting the n nonzero pivots arising during the elimination. Computing $\mathrm{Rem}(A, p)$ and then its LUP decomposition is within our target cost since $\log p \in O(\log n + \mathrm{loglog}\,\|A\|)$ and $\mathsf{B}(\log p) \in O(\mathsf{M}(\log p)(\mathrm{loglog}\, p))$. If, during the course of the LUP decomposition, it is determined that A is singular modulo p, then return FAIL. Otherwise, let X be the smallest power of p which satisfies the third requirement of the lemma. Then, X also satisfies the fourth requirement. □

We note that the lower bound on the lifting modulus X that is specified by the third item in Lemma 2.1 is required by the lifting algorithm of Pauderis and Storjohann (2012, Section 3) which we recall in Lemma 2.3 and base our fast system solving routines.

Corollary 2.2. *If X is a lifting modulus as in Lemma 2.1, then* $\mathrm{Rem}(A^{-1}, X)$ *can be computed in time* $O(n^\omega \,\mathsf{M}(\log n + \log \|A\|))$.

Proof. Let p and LUP be as in the proof of Lemma 2.1. Compute $\mathrm{Rem}(A^{-1}, p) = \mathrm{Rem}(P^T U^{-1} L^{-1}, p)$, and use $O(\mathrm{loglog}\, X)$ steps of algebraic Newton iteration (von zur Gathen and Gerhard, 2013, Algorithm 9.3) to lift $\mathrm{Rem}(A^{-1}, p)$ to $\mathrm{Rem}(A^{-1}, X)$. The running time is dominated by the last step of the lifting, which is within the claimed cost. □

2.2 Double-plus-one lifting

Any rational number can be written as an integer and a proper fraction. For example,

$$\frac{9622976468279041913}{21341} = 450914974381661 + \frac{14512}{21341},$$

where 450914974381661 is the quotient and 14512 is the remainder of the numerator with respect to the denominator. A similar construction replaces the quotient with a truncated p-adic expansion of the fraction, where p should be relatively prime to the denominator. For example,

$$\frac{9622976468279041913}{21341} = 9035820194880943821 - \frac{10453}{21341} \times 2^{64}. \tag{2.2}$$

In some applications, we only require the remainder. Multiplying (2.2) by the denominator 21341 shows that

$$14512 = -10453 \times 2^{64} \bmod 21341,$$

where -10453 is what we call a high-order residue.

The same idea can be extended to integer matrices. Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular and let X be a lifting modulus as in Lemma 2.1. Given a $k \in \mathbb{Z}_{>0}$, double-plus-one lifting (Pauderis and Storjohann, 2012, Section 3) computes a straight line formula that is congruent modulo X^k to the X-adic expansion

$$A^{-1} \equiv * + *X + *X^2 + \cdots + *X^{k-1} \bmod X^k. \tag{2.3}$$

The straight line formula consists of only $O(\log k)$ matrices instead of k as in (2.3). More precisely, given a $k \in \mathbb{Z}_{>0}$ that is one less than a power of 2, double-plus-one lifting computes a residue $R \in \mathbb{Z}^{n\times n}$ such that

$$A^{-1} = D + A^{-1}RX^k, \tag{2.4}$$

where $D \in \mathbb{Z}^{n\times n}$ satisfies $||D|| \leq 0.6X^k$ and $\|R\| \leq n\|A\|$. Note that $D \equiv A^{-1} \bmod X^k$. Instead of computing D explicitly, double-plus-one lifting computes a formula

$$D = (\cdots((A_0(I + R_0X) + M_0X^2)(I + R_1X^3) + M_1X^6)(I + R_2X^7) + M_2X^{14})\cdots), \tag{2.5}$$

where A_0, R_*, M_* are $n \times n$ integer matrices with magnitude strictly less than X. For example, for $k = 7$ we have

$$D = (A_0(I + R_0X) + M_0X^2)(I + R_1X^3) + M_1X^6.$$

The following result is (Pauderis and Storjohann, 2012, Corollary 6) except that we use Corollary 2.2 to compute $\mathrm{Rem}(A^{-1}, X)$ in the allotted time.

Lemma 2.3. (Pauderis and Storjohann, 2012, Corollary 6) *Assume we have a lifting modulus X as in Lemma 2.1. Let $k \in \mathbb{Z}_{>0}$ be one less than a power of two. If $\log k \in O(\log n)$, then a residue R as in (2.4) and a straight line formula for D as shown in (2.5) can be computed in time $O(n^\omega\, \mathsf{M}(\log n + \log \|A\|) \log n)$.*

For the integrality certification problem only the residue R is required. For solving $A^{-1}b$, the straight line formula is applied in $O(\log n)$ steps to the X-adic expansion of b.

2.3 System solving

In this section, we present an algorithm that computes a linear system solution expansion that matches our target complexity. The double-plus-one lifting construction from Section 2.2 immediately implies an algorithm of that computes a linear system solution expansion, but, as far as we are aware, a careful description and cost analysis has not been presented before in the literature. We offer a treatment here.

Let X be a lifting modulus as in Lemma 2.1. Consider equations (2.4) and (2.5). If $k \geq d$, then given a $B \in \mathbb{Z}^{n\times m}$, we can compute $\operatorname{Rem}(A^{-1}B, X^d)$ by premultiplying B by the straight line formula for $D \equiv A^{-1} \bmod X^k$ on the right hand side of (2.5), keeping intermediate expressions reduced modulo X^d. Applying the formula requires doing the following operation $O(\log k)$ times: premultiplying an $n \times m$ matrix with entries reduced modulo X^d by an $n \times n$ matrix $*$ with $\| * \| < X$.

When X is not a power of 2, we need to use radix conversion to go between the binary and X-adic representation of integers. To avoid unnecessary radix conversions, we can compute the X-adic expansion of B once at the beginning, and then keep intermediate results in X-adic form. The following result is a corollary of Storjohann (2005, Theorem 33).

Lemma 2.4. *Let $X \in \mathbb{Z}_{>0}$ satisfy $\log X \in O(\log n + \log \|A\|)$. Let $C \in \mathbb{Z}^{n\times n}$ with $\|C\| < X$ and $B \in \mathbb{Z}^{n\times m}$ with $B = \operatorname{Rem}(B, X^d)$. If $m \times d \in O(n)$, then $\operatorname{Rem}(CB, X^d)$ can be computed in time $O(n^\omega\, \mathsf{M}(\log n + \log \|A\|))$, assuming the input parameter B and output $\operatorname{Rem}(CB, X^d)$ are given as X-adic expansions.*

Theorem 2.5. *Assume we have a lifting modulus X as in Lemma 2.1. If entries in $B \in \mathbb{Z}^{n\times m}$ are reduced modulo X^d and $m \times d \in O(n)$, then $\operatorname{Rem}(A^{-1}B, X^d)$ can be computed in time*

$$O(n^\omega\, \mathsf{M}(\log n + \log \|A\|) \log n).$$

Proof. Using radix conversion (von zur Gathen and Gerhard, 2013, Theorem 9.17), compute the X-adic expansion of B in time $O(nm\, \mathsf{M}(d \log X) \log d)$. Simplifying this cost estimate using

$$\mathsf{M}(d \log X) \in O(d^{\omega-1}\, \mathsf{M}(\log X)) \quad \text{and} \quad d \in O(n/m)$$

shows that this is within the allotted time. Compute a straight line formula congruent to $A^{-1} \bmod X^d$ using Lemma 2.3. Applying the straight line formula to $B \bmod X^d$ to compute the X-adic expansion of $\operatorname{Rem}(A^{-1}B, X^d)$ now requires $O(\log d) \subseteq O(\log n)$ applications of Lemma 2.4, plus some matrix additions which do not dominate the cost. This

can be illustrated by the following scheme. Since the straight line formula has $O(\log d)$ coefficients, then for some $i \in O(\log d)$:

$$\left[\begin{array}{l} E, F := 0_{n\times m}, B \\ \textbf{while } i \geq 0 \textbf{ do} \\ \quad E := \mathrm{Rem}(E + (X^{2^{i+1}-1})^2 M_i F, X^d) \\ \quad F := \mathrm{Rem}(F + X^{2^{i+1}-1} R_i F, X^d) \\ \quad i := i - 1 \\ \textbf{od} \\ E := \mathrm{Rem}(E + A_0 F, X^d) \\ \textbf{return } E \end{array}\right. \tag{2.6}$$

Note that the multiplications with powers of X are free since we are working with X-adic expansions throughout. Finally, compute $\mathrm{Rem}(A^{-1}B, X^d)$ from its X-adic expansion using another radix conversion. □

Example 2.6. *We give an example of a step-by-step application of the scheme in (2.6). For $d = 7$, the straight line formula will be*

$$A^{-1} \equiv (A_0(I + R_0 X) + M_0 X^2)(I + R_1 X^3) + M_1 X^6 \bmod X^7.$$

Then, after the first iteration, we have that

$$E_{(i=1)} = X^6 M_1 B \quad \textit{and} \quad F_{(i=1)} = B + X^3 R_1 B,$$

and after the second, it will be that

$$E_{(i=0)} = E_{(i=1)} + X^2 M_0 F_{(i=1)} \quad \textit{and} \quad F_{(i=0)} = F_{(i=1)} + X R_0 F_{(i=1)},$$

with everything being reduced modulo X^7. Finally, we return

$$E = E_{(i=0)} + A_0 F_{(i=0)}.$$

2.4 Integrality certification

In this section, we present an algorithm for integrality certification.

A rational system solution $A^{-1}B$ can have entries with large numerators compared to denominators. Given an $s \in \mathbb{Z}_{>0}$, integrality certification can be used to determine whether

$sA^{-1}B$ is integral in a cost that depends on $\log s + \log \|B\|$ but not on $\log \|sA^{-1}B\|$. If $sA^{-1}B$ is integral, then this version of integrality certification also returns $\mathrm{Rem}(sA^{-1}B, s)$. This is the fractional part of $A^{-1}B$, and it will be used by the Smith massager algorithm of Chapter 5.

Our approach is adapted from the algorithm in (Storjohann, 2005, Section 11) which is randomized. Here we show how to solve the integrality certification problem deterministically assuming that we have a lifting modulus $X \perp \det A$.

We begin by using double-plus-one lifting to compute a high-order residue $R \in \mathbb{Z}^{n\times n}$ such that

$$A^{-1} = D + A^{-1}R \times X^h \tag{2.7}$$

for some $h \in \mathbb{Z}_{>0}$ such that

$$X^h > 2sn^{n/2}||A||^{n-1}||B||. \tag{2.8}$$

The matrix D, which will satisfy $||D|| \leq 0.6X^h$, is not needed and not computed explicitly. If the "dimension $\times$ precision" compromise

$$m \times (\log s + \log ||B||) \in O(n(\log n + \log ||A||)) \tag{2.9}$$

holds, then, by Lemma 2.3, such an R can be computed in time

$$O(n^\omega \,\mathsf{M}(\log n + \log ||A||) \log n). \tag{2.10}$$

Now multiply equation (2.7) on the right by sB to see that

$$sA^{-1}B = sDB + A^{-1}(sRB) \times X^h. \tag{2.11}$$

The next step is to use the linear system solving routine of Theorem 2.5 to compute

$$\mathrm{Rem}(A^{-1}(sRB), X^\ell)$$

for some $\ell \in \mathbb{Z}_{>0}$ such that

$$X^\ell > 2n||A||(0.6sn||B||). \tag{2.12}$$

Assuming (2.9), this can also be done in time (2.10).

Adjusting slightly the proof of Storjohann (2005, Theorem 46) to account for the fact that D in (2.7) satisfies $||D|| \leq 0.6X^h$, it can be shown, for the choices of h and ℓ in (2.8) and (2.12), respectively, that if C is set to be the matrix equal to $\mathrm{Rem}(A^{-1}(sRB), X^\ell)$ but with entries reduced in the symmetric range modulo X^ℓ, then $C = sA^{-1}RB$ (and hence $sA^{-1}RB$ is integral) if and only if $||C|| < 0.6sn||B||$. Considering (2.11), it then follows that $\mathrm{Rem}(C \times X^h, s)$ is equal to $\mathrm{Rem}(sA^{-1}B, s)$.

Theorem 2.7. *Assume we have a lifting modulus X as in Lemma 2.1. Let $s \in \mathbb{Z}_{>0}$ and $B \in \mathbb{Z}^{n\times m}$ be given. There exists an algorithm that determines whether $sA^{-1}B$ is integral, and, if so, returns $\mathrm{Rem}(sA^{-1}B, s)$. If $m \times (\log s + \log \|B\|) \in O(n \log X)$ and $m \in O(n)$, then the running time is*

$$O(n^\omega \,\mathsf{M}(\log n + \log \|A\|) \log n).$$

2.5 Unbalanced multiplications reduced to balanced

In this section, we are presenting some statements regarding unbalanced matrix multiplication. In many of the algorithms presented in this thesis, we have to deal with matrices that have entries of skewed size but bounded overall size. The following lemmas reduce specific types of multiplications of such skewed matrices to multiplications of uniform ones.

Let us define $\mathrm{length}(a)$ for an integer a to be the number of bits in its binary representation, that is,

$$\mathrm{length}(a) = \begin{cases} 1 & \text{if } a = 0 \\ 1 + \lfloor \log_2 |a| \rfloor & \text{otherwise} \end{cases}. \tag{2.13}$$

By extension, for a matrix we define $\mathrm{length}(A) = \mathrm{length}(||A||)$, so $\mathrm{length}(A)$ is the length of the largest entry of A in absolute value.

We begin by showing that we can multiply a skewed matrix M together with a vector w whose length is about as much as the sum of lengths of the columns of M in roughly the same time as multiplying two matrices of dimension and average length as M.

Lemma 2.8. *Let a matrix $M \in \mathbb{Z}^{n\times m}$ and a vector $w \in \mathbb{Z}^{m\times 1}$. Furthermore, assume that*

$$\sum_{j=1}^{m} \mathrm{length}(M_{1..n,j}) \le d \quad \textit{and} \quad \mathrm{length}(w) \le d$$

for some $d \in \mathbb{Z}_{\ge 0}$. We can compute the product Mw in time $O(nm^{\omega-1}\mathsf{M}(d/m + \log m))$.

Proof. Choose $X := 2^{\lceil d/m \rceil}$, and let

$$M = M_0 + M_1 X + \cdots + M_{m-1} X^{m-1}$$

and

$$w = w_0 + w_1 X + \cdots + w_{m-1} X^{m-1}$$

be the X-adic expansions of M and w. (The coefficients are computed in the symmetric range modulo X.) Our approach is to compute the product

$$\overbrace{\begin{bmatrix} M_0 & M_1 & \cdots & M_{m-1} \end{bmatrix}}^{\bar{M}} \overbrace{\begin{bmatrix} w_0 & w_1 & \cdots & w_{m-1} & & & \\ & w_0 & \cdots & w_{m-2} & w_{m-1} & & \\ & & \ddots & \vdots & \vdots & \ddots & \\ & & & w_0 & w_1 & \cdots & w_{m-1} \end{bmatrix}}^{\bar{W}},$$

from which Mw can be recovered fast. (Notice that the operations to compute the X-adic expansion from a matrix or the matrix from an X-adic expansion take linear time on the number of entries when X is a power of 2.)

Now, the column dimension of $\bar{M}$ and row dimension of $\bar{W}$ is m^2 which is too large to fit within our target complexity. However, because of the assumption that $\sum_{j=1}^{m} \text{length}(M_{1..n,j}) \leq d$ and the fact that $\log(X) = \lceil d/m \rceil$, matrix $\bar{M}$ must contain many zero columns. More specifically, the number of non-zero columns in $\bar{M}$ cannot exceed

$$\sum_{j=1}^{m} \left\lceil \frac{\text{length}(M_{1..n,j})}{\lceil d/m \rceil} \right\rceil \leq \sum_{j=1}^{m} \left(m \frac{\text{length}(M_{1..n,j})}{d} + 1 \right) \leq 2m.$$

Therefore, let $\tilde{M} \in \mathbb{Z}^{n \times 2m}$ be the matrix obtained from $\bar{M}$ by omitting $m^2 - 2m$ zero columns, and let $\tilde{W} \in \mathbb{Z}^{2m \times 2m-1}$ be the matrix obtained from $\bar{W}$ by omitting $m^2 - 2m$ rows corresponding to the columns that were omitted in $\bar{M}$. This transformation reduces the multiplication of $\bar{M}\bar{W}$ to the multiplication of $\tilde{M}\tilde{W}$ which can be done in time $O(nm^{\omega-1}\mathsf{M}(d/m + \log m))$ since $\log \|\tilde{M}\tilde{W}\| \in O(d/m + \log m)$. □

The next lemma establishes that we can multiply two matrices U, M, where matrix U has skewed row lengths and matrix M has skewed column lengths, in roughly the same time as multiplying two matrices of the same dimension and average length as U, M.

Lemma 2.9. *For matrices $U, M \in \mathbb{Z}^{n \times n}$, assume that*

$$\sum_{i=1}^{n} \text{length}(U_{i,1..n}) \leq nd \quad \textit{and} \quad \sum_{j=1}^{n} \text{length}(M_{1..n,j}) \leq nd$$

for some $d \in \mathbb{Z}_{\geq 0}$. We can compute the product UM in time $O(n^\omega \, \mathsf{M}(d + \log n))$.

Proof. Choose $X := 2^d$, and let

$$U = U_0 + U_1 X + \cdots + U_{n-1} X^{n-1}$$

and

$$M = M_0 + M_1 X + \cdots + M_{n-1} X^{n-1}$$

be the X-adic expansions of U and M. (The coefficients are computed in the symmetric range modulo X.) Our approach is to compute the product

$$\overbrace{\begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_{n-1} \end{bmatrix}}^{\bar{U}} \overbrace{\begin{bmatrix} M_0 & M_1 & \cdots & M_{n-1} \end{bmatrix}}^{\bar{M}},$$

from which UM can be recovered fast. (Notice that the operations to compute the X-adic expansion from a matrix or the matrix from an X-adic expansion take linear time on the number of entries when X is a power of 2.)

Now, the row dimension of $\bar{U}$ and column dimension of $\bar{M}$ are n^2 which is too large to fit within our target complexity. However, because of the assumption that

$$\sum_{i=1}^{n} \text{length}(U_{i,1..n}) \leq nd \quad \text{and} \quad \sum_{j=1}^{n} \text{length}(M_{1..n,j}) \leq nd$$

and the fact that $\log(X) = d$, matrix $\bar{U}$ and matrix $\bar{M}$ must contain many zero rows and columns respectively. More specifically, the number of non-zero columns in $\bar{M}$ cannot exceed

$$\sum_{j=1}^{n} \left\lceil \frac{\text{length}(M_{1..n,j})}{d} \right\rceil \leq \sum_{j=1}^{n} \left(\frac{\text{length}(M_{1..n,j})}{d} + 1 \right) \leq 2n.$$

Similarly, the same bound holds for the number of non-zero rows in $\bar{U}$.

Therefore, let $\tilde{U} \in \mathbb{Z}^{2n \times n}$ be the matrix obtained from $\bar{U}$ and $\tilde{M} \in \mathbb{Z}^{n \times 2n}$ the matrix obtained from $\bar{M}$ by omitting $n^2 - 2n$ zero rows and columns respectively. This transformation reduces the multiplication of $\bar{U}\bar{M}$ to the multiplication of $\tilde{U}\tilde{M}$, which can be done in time $O(n^\omega \, \mathsf{M}(d + \log n))$ since $\log \|\tilde{U}\tilde{M}\| \in O(d + \log n)$. □

Finally, a corollary follows that replaces the skewed matrix U with a uniform one. The corollary can be also seen as a generalization of Lemma 2.4.

Corollary 2.10. *For matrices $A, M \in \mathbb{Z}^{n\times n}$, assume that*

$$\operatorname{length}(A) \leq d \quad \textit{and} \quad \sum_{j=1}^{n} \operatorname{length}(M_{1..n,j}) \leq nd$$

for some $d \in \mathbb{Z}_{\geq 0}$. We can compute the product AM in time $O(n^{\omega}\,\mathsf{M}(d + \log n))$.

Chapter 3

Partial linearization

In this chapter, we present a partial linearization technique which will allow us to extend many of our algorithms, especially the ones from Chapter 2, so that their cost estimates depend on the average size of entries of the input matrix and not the largest one.

Typically, the cost of algorithms that take as input an integer matrix $A \in \mathbb{Z}^{n\times m}$ is expressed in terms of the dimensions n and m, and $\log \|A\|$ which is proportional to the bitlength of the largest entry of A in absolute value, namely, $\text{length}(A)$ as defined in (2.13).

But consider decomposing A into columns as

$$A = \left[\ v_1 \ \middle|\ \cdots \ \middle|\ v_m \ \right] \in \mathbb{Z}^{n\times m}.$$

For some inputs, the lengths of the columns v_i can be skewed, that is, the *average* column length

$$d = \left\lceil \sum_{i=1}^{m} \text{length}(v_i)/m \right\rceil$$

can be asymptomatically smaller than $\text{length}(A) = \max_i \text{length}(v_i)$. Even $\text{length}(A) \approx md$ is possible in the case of one column of large length. For such inputs, being able to replace the term $\text{length}(A)$ with the average length d can give significantly improved cost estimates.

Example 3.1. *Let $A \in (\mathbb{Z}/(2))^{n\times m}$ be a matrix containing just bits. In this case,* $\text{length}(A)$ *is* 1 *and the average column* length d *is also* 1. *Furthermore, let $A' \in \mathbb{Z}^{n\times m}$ be matrix A but with the last column multiplied by $2^{m+1} - 1$. Now,* $\text{length}(A') = m + 1$ *but the average* length *is only $d' = 2$.*

In this chapter, we adapt a partial linearization technique for polynomial matrices from Gupta et al. (2012, Section 6) to the case of integer matrices. The technique was introduced by Storjohann (2006, Section 2) and generalized to the version we give here by Gupta et al. (2012, Section 6). The main motivation is to extend the algorithms from Chapter 2, so that their cost estimates depend on the average length d and not length(A).

The technique can transform the input matrix A into a new matrix D which can be used in place of A for all of the algorithms presented in Chapter 2 that involve A^{-1}, and many more. Matrix D will satisfy that length$(D) \leq d + 1$, at the cost of at most m more rows and columns than $A \in \mathbb{Z}^{n \times m}$.

More importantly, the constructed matrix D will "imitate" A in a way such that the output of the routines with D as input includes the original output in a direct way. Specifically, matrix D will satisfy the following two fundamental properties with respect to A:

(i) D can be obtained from $\text{diag}(A, I)$ using unimodular row and column operations.

(ii) The principal $n \times n$ submatrix of the adjoint of D equals the adjoint of A (for square matrices).

Property (i) establishes that the rank, the determinant (for square matrices) and the Smith form of matrix A can be trivially deduced from the same objects for matrix D. In Section 4.4, we also show that computing the Smith massager of a nonsingular A can be directly reduced to computing the Smith massager of D.

Furthermore, property (ii) provides us with a direct extension of system solving. For any matrix $B \in \mathbb{Z}^{n \times *}$, we have that the first n rows of

$$D^{-1} \begin{bmatrix} B \\ 0 \end{bmatrix}$$

are equal to $A^{-1}B$ for a nonsingular $A \in \mathbb{Z}^{n \times n}$. Finally, because of the fact that $\det D = \det A$ and property (ii) one can easily check that the principal $n \times n$ submatrix of the lower row Hermite form of D equals the lower row Hermite form of A.

Before continuing, we give an example of a matrix A with skewed column lengths and its partial linearization D.

Example 3.2. *Let*

$$A = \begin{bmatrix} 2 & 4 & 44199 & 3061969404 \\ 4 & 8 & 19644 & 765492351 \\ 7 & 8 & 44199 & 5358446457 \\ 7 & 5 & 9822 & 765492351 \end{bmatrix} \in \mathbb{Z}^{4 \times 4}$$

with $\text{length}(A) = 33$ *and* $d = 14$. *Then, let*

$$D = \begin{bmatrix} 2 & 4 & 11431 & 12796 & 2 & 6663 & 11 \\ 4 & 8 & 3260 & 15487 & 1 & 13953 & 2 \\ 7 & 8 & 11431 & 10105 & 2 & 15757 & 19 \\ 7 & 5 & 9822 & 15487 & 0 & 13953 & 2 \\ 0 & 0 & -16384 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -16384 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -16384 & 1 \end{bmatrix} \in \mathbb{Z}^{7\times 7}.$$

Notice that $\|D\| \leq 2^d = 16384$.

3.1 The partial linearization construction

In this section, we give all the details regarding the construction of the partially linearized matrix and its relation with the original matrix.

Let $e \in \mathbb{Z}_{\geq 0}$ and $d \in \mathbb{Z}_{\geq 1}$ be given and assume for the moment that a column vector $v \in \mathbb{Z}_{\geq 0}^{n\times 1}$ contains only nonnegative entries. Then, we define $C_{e,d}(v)$ to be the unique $n \times e$ matrix over $\mathbb{Z}_{\geq 0}$ that satisfies

$$\text{Quo}(v, 2^d) = C_{e,d}(v) \begin{bmatrix} 1 \\ 2^d \\ \vdots \\ 2^{(e-1)d} \end{bmatrix}, \tag{3.1}$$

with all but possibly the last column (if $e > 0$) of magnitude strictly less than 2^d. If $e = 0$ then $C_{e,d}(v)$ is the $n \times 0$ matrix, while for $e \geq 1$,

$$v = \text{Rem}(v, 2^d) + \text{Col}(C_{e,d}(v), 1)2^d + \cdots + \text{Col}(C_{e,d}(v), e)2^{ed} \tag{3.2}$$

is the 2^d-adic series expansion of v, except that the coefficient $\text{Col}(C_{e,d}(v), e)$ of 2^{ed} may have magnitude greater than or equal to 2^d.

Example 3.3. *For* $v = \left[\begin{array}{c} 15 \end{array}\right]$, $\text{Rem}(v, 2) = 1$ *and* $C_{2,1}(v) = \left[\begin{array}{c|c} 1 & 3 \end{array}\right]$.

We can extend the definition of $C_{e,d}$ to an arbitrary vector $v \in \mathbb{Z}^{n\times 1}$ in the following way. Let $v^{(+)}$ denote the vector v but with all negative entries zeroed out, and $v^{(-)} = v - v^{(+)}$

denote the vector v but with all but the positive entries zeroed out. Then, $v^{(+)}$ and $-v^{(-)}$ are over $\mathbb{Z}_{\geq 0}$, and $v = v^{(+)} - (-v^{(-)})$. Finally we let

$$C^*_{e,d}(v) := C_{e,d}(v^{(+)}) - C_{e,d}(-v^{(-)}),$$

which still satisfies equations (3.1) and (3.2) if we replace Rem and Quo by

$$\mathrm{Rem}^*(v, 2^d) := \mathrm{Rem}(v^{(+)}, 2^d) - \mathrm{Rem}(-v^{(-)}, 2^d),$$

$$\mathrm{Quo}^*(v, 2^d) := \mathrm{Quo}(v^{(+)}, 2^d) - \mathrm{Quo}(-v^{(-)}, 2^d).$$

We define structured matrices E_d and F_d by

$$E_d := -2^d \,\mathrm{Col}(I, 1) = \begin{bmatrix} -2^d \\ \\ \\ \end{bmatrix} \quad \text{and} \quad F_d := \begin{bmatrix} 1 & & & & \\ -2^d & 1 & & & \\ & -2^d & \ddots & & \\ & & \ddots & 1 & \\ & & & -2^d & 1 \end{bmatrix},$$

with the dimensions of E_d and F_d to be determined by the context. We remark that F_d^{-1} will be the unit lower triangular Toeplitz matrix with 2^{id} on the ith subdiagonal. The next lemma follows from the definitions of E_d and F_d and equations (3.1) and (3.2).

Lemma 3.4. *Given $v \in \mathbb{Z}^{n\times 1}$, $e \in \mathbb{Z}_{\geq 0}$ and $d \in \mathbb{Z}_{\geq 1}$, let*

$$c = \begin{cases} v & \text{if } e = 0 \\ \mathrm{Rem}^*(v, 2^d) & \text{if } e > 0 \end{cases},$$

and

$$Q_{e,d}(v) = \left[\; \mathrm{Quo}^*(v, 2^d) \;\middle|\; \cdots \;\middle|\; \mathrm{Quo}^*(v, 2^{ed}) \;\right].$$

Then,

$$\left[\begin{array}{c|c} c & C^*_{e,d}(v) \\ \hline E_d & F_d \end{array}\right] = \left[\begin{array}{c|c} I_n & Q_{e,d}(v) \\ \hline & I_e \end{array}\right] \left[\begin{array}{c|c} v & \\ \hline & I_e \end{array}\right] \left[\begin{array}{c|c} 1 & \\ \hline E_d & F_d \end{array}\right]. \tag{3.3}$$

By replacing the single column vector v with a matrix $A = \left[\; v_1 \;\middle|\; \cdots \;\middle|\; v_m \;\right]$ of m column vectors v_i we obtain:

Corollary 3.5. *Given* $A = \left[\begin{array}{c|c|c} v_1 & \cdots & v_m \end{array}\right] \in \mathbb{Z}^{n\times m}$, $\bar{e} = (e_1, \ldots, e_m) \in \mathbb{Z}^m_{\geq 0}$ *and* $d \in \mathbb{Z}_{\geq 1}$. *Let*

$$c_i = \begin{cases} v_i & \text{if } e_i = 0 \\ \operatorname{Rem}^*(v_i, 2^d) & \text{if } e_i > 0 \end{cases},$$

for $1 \leq i \leq m$, *and define the matrix*

$$D = D_{\bar{e},d}(A) := \left[\begin{array}{c|c|c||c|c|c} c_1 & \cdots & c_m & C^*_{e_1,d}(v_1) & \cdots & C^*_{e_m,d}(v_m) \\ \hline E_d & & & F_d & & \\ \hline & \ddots & & & \ddots & \\ \hline & & E_d & & & F_d \end{array}\right] \in \mathbb{Z}^{\bar{n}\times\bar{m}},$$

with $\bar{n} = n + e_{[m]}$ *and* $\bar{m} = m + e_{[m]}$, *where* $e_{[m]} = e_1 + \cdots + e_m$. *Then, matrix* D *satisfies*

$$D = \begin{bmatrix} I_n & Q \\ & I_{e_{[m]}} \end{bmatrix} \begin{bmatrix} A & \\ & I_{e_{[m]}} \end{bmatrix} \begin{bmatrix} I_m & \\ E & F \end{bmatrix}, \tag{3.4}$$

where $Q = \left[\begin{array}{c|c|c} Q_{e_1,d}(v_1) & \cdots & Q_{e_m,d}(v_m) \end{array}\right] \in \mathbb{Z}^{n\times e_{[m]}}$, $E = \operatorname{diag}(E_d, \ldots, E_d) \in \mathbb{Z}^{e_{[m]}\times m}$ *and* $F = \operatorname{diag}(F_d, \ldots, F_d) \in \mathbb{Z}^{e_{[m]}\times e_{[m]}}$.

From equation (3.4), it is apparent that D enjoys the following properties:

Corollary 3.6. *Given* $A \in \mathbb{Z}^{n\times m}$, $\bar{e} = (e_1, \ldots, e_m) \in \mathbb{Z}^m_{\geq 0}$ *and* $d \in \mathbb{Z}_{\geq 1}$. *Let* $D = D_{\bar{e},d}(A)$ *as in Corollary 3.5. Then*

(i) $\operatorname{rank}(D) = \operatorname{rank}(A) + e_{[m]}$.

(ii) D *has the same Smith form as* A *up to additional trivial invariant factors.*

Furthermore, if $n = m$, *then:*

(iii) $\det D = \det A$.

(iv) The principal $n \times n$ *submatrix of the adjoint of* D *equals the adjoint of* A.

Notice that Corollary 3.5 does not make any assumptions on the parameters $\bar{e}$ and d. The properties of matrix $D = D_{\bar{e},d}(A)$ corresponding to the original matrix A are true for any $\bar{e}$ and d. However, the partial linearization technique is particularly useful if we pick $\bar{e}$ and d in a way such that $\bar{m} \in O(m)$ and $\log \|D\|$ corresponds to the average length of the columns of A. The following is the main result of this section.

Theorem 3.7. *Given matrix* $A = \left[\begin{array}{c|c|c} v_1 & \cdots & v_m \end{array}\right] \in \mathbb{Z}^{n\times m}$, *let*

$$d = \left\lceil \sum_{i=1}^{m} \text{length}(v_i)/m \right\rceil,$$

$\bar{e} = (e_1, \ldots, e_m) \in \mathbb{Z}^m_{\geq 0}$ *where each* $e_i \in \mathbb{Z}_{\geq 0}$ *is chosen minimal such that* $\text{length}(v_i) \leq (e_i + 1)d$, *and* $D = D_{\bar{e},d}(A)$. *Then:*

- $||D|| \leq 2^d$,
- $\bar{n} < n + m$ *and* $\bar{m} < 2m$.

Proof. The choice of e_i ensures that, for each v_i, the expansion in (3.2) is the 2^d-adic expansion of v. This shows that the length of all entries in the first n rows of D are bounded by d. Since the entries in the last $\bar{n} - n$ rows of D are bounded in magnitude by 2^d, the claimed bound for $||D||$ follows.

To prove our upper bounds for $\bar{n}$ and $\bar{m}$ we show that $\sum_{i=1}^{m} e_i < m$. Note that e_i is precisely defined as

$$e_i = \left\lceil \frac{\text{length}(v_i)}{d} - 1 \right\rceil < \frac{\text{length}(v_i)}{d},$$

and so

$$\sum_{i=1}^{m} e_i < \sum_{i=1}^{m} \frac{\text{length}(v_i)}{d} \leq m.$$

□

Example 3.8. *Let*

$$A = \begin{bmatrix} 2 & 4 & 44199 & 3061969404 \\ 4 & 8 & 19644 & 765492351 \\ 7 & 8 & 44199 & 5358446457 \\ 7 & 5 & 9822 & 765492351 \end{bmatrix}.$$

Then, with the average (column) length $d = 14$ *and* $\bar{e} = (0, 0, 1, 2)$ *we get*

$$D = \left[\begin{array}{cccc||ccc} 2 & 4 & 11431 & 12796 & 2 & 6663 & 11 \\ 4 & 8 & 3260 & 15487 & 1 & 13953 & 2 \\ 7 & 8 & 11431 & 10105 & 2 & 15757 & 19 \\ 7 & 5 & 9822 & 15487 & 0 & 13953 & 2 \\ \hline\hline 0 & 0 & -16384 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -16384 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -16384 & 1 \end{array}\right].$$

One can easily verify that the adjoint of A lies in the principal 4×4 sub-matrix of the adjoint of D, and that the Smith form of A lies in the trailing 4×4 sub-matrix of the Smith form of D.

The approach of Corollary 3.5 can also be used to partially linearize the rows of a matrix A. If we transpose a matrix A with skewed row lengths, then it has skewed column lengths. Then, by transposing the linearization of A^T, it satisfies all the properties given in Corollary 3.6. We can see that from the row linearization equivalent of equation (3.4), which is

$$D_{\bar{e},d}(A^T)^T = \begin{bmatrix} I & E^T \\ & B^T \end{bmatrix} \begin{bmatrix} A & \\ & I \end{bmatrix} \begin{bmatrix} I & \\ Q^T & I \end{bmatrix}. \tag{3.5}$$

Corollary 3.9. *Let $A \in \mathbb{Z}^{m \times n}$, and consider the matrix $D = D_{\bar{e},d}(A^T)^T$. The magnitude of the entries in D will then be bounded by 2^d where d is the average* length *over the rows of A, and D will enjoy all the properties following from Corollary 3.5 and Theorem 3.7.*

3.2 The permutation bound

In this section, we will see how to extend the partial linearization technique in order to handle square matrices that have both large columns and rows.

Our approach so far is particularly effective for matrices $A \in \mathbb{Z}^{n \times n}$ where the average of the sum of the lengths of the columns (or rows) is small compared to length(A). However, the technique is not useful for input matrices that have, simultaneously, some columns and rows of large length. For this reason, as in the case of polynomial matrices (Gupta et al., 2012, Section 6), we develop an approach to handle such inputs based on the following *a priori* upper bound for $|\det A|$.

By definition, $\det A = \sum_{\sigma \in S_n} \operatorname{sign}(\sigma) \prod_{i=1}^{n} A_{i,\sigma_i}$, where S_n is the set of all permutations of $(1, 2, \ldots, n)$. Therefore,

$$\det A \leq n! \max_{\sigma \in S_n} \prod_{i=1}^{n} |A_{i,\sigma_i}|,$$

and so, we define

$$\text{PermutBnd}(A) := \max_{\sigma \in S_n} \sum_{i=1}^{n} \text{length}(A_{i,\sigma_i}).$$

As in the polynomial case, up to a row and column permutation, we may assume that $d_i = \text{length}(A_{i,i})$ bounds the length of the submatrix $A_{i\ldots n,i\ldots n}$, for $1 \leq i \leq n$. Such a row

and column permutation can be found by sorting the set of triples $\{(i, j, |A_{i,j}|)\}_{1 \le i,j \le n}$ into nonincreasing order according to their third component. Then, by definition, $d_1 + \cdots + d_n \le \text{PermutBnd}(A)$.

Let $d = \lceil \sum_{i=1}^{n} d_i/n \rceil$ and $\bar{e} = (e_1, \ldots, e_n)$ with $e_i \in \mathbb{Z}_{\ge 0}$ minimal such that $d_i \le (e_i+1)d$. Then, due to the choice of d_i, row i of matrix $D_{\bar{e},d}(A)$ will have length bounded by $d_i + 1$ for $1 \le i \le n$, and all the extra rows will have length bounded by $d + 1$. Furthermore, let $\bar{e}'$ contain $\bar{e}$ augmented with $\sum_{i=1}^{n} e_i$ zeros. We have the following corollary for matrix $D := D_{\bar{e}',d}(D_{\bar{e},d}(A)^T)^T$.

Corollary 3.10. *Let $A \in \mathbb{Z}^{n \times n}$ be given. Using the choices for d, $\bar{e}$ and $\bar{e}'$ as specified above, the matrix $D := D_{\bar{e}',d}(D_{\bar{e},d}(A)^T)^T \in \mathbb{Z}^{\bar{n}' \times \bar{n}'}$ satisfies:*

- $\|D\| \le 2^d$ *with* $d \le \lceil \text{PermutBnd}(A)/n \rceil$,
- $\bar{n}' < 3n$,

along with all the properties from Corollary 3.6.

Remark 3.11 (Application to system solving)**.** *The fact that the principal $n \times n$ submatrix of the adjoint of the partially linearized matrix D is equal to the adjoint of the original matrix $A \in \mathbb{Z}^{n \times n}$ provides us with a direct extension to system solving. For any matrix $B \in \mathbb{Z}^{n \times m}$, we have that the first n rows of*

$$D^{-1} \begin{bmatrix} B \\ 0 \end{bmatrix}$$

are equal to $A^{-1}B$. Therefore, Theorem 2.5 can have cost which depends on the average bitlength d of A and not the bitlength of the largest entry. The average bitlength d can assume any of the three definitions given by Theorem 3.7, Corollary 3.9 and Corollary 3.10.

Remark 3.12 (Application to integrality certification)**.** *Suppose D is a partial linearization of $A \in \mathbb{Z}^{n \times n}$. For any $s \in \mathbb{Z}_{>0}$ and $B \in \mathbb{Z}^{n \times m}$, it follows from equations (3.4) and (3.5) that*

$$sD^{-1} \begin{bmatrix} B \\ 0 \end{bmatrix}$$

will be integral if and only if $sA^{-1}B$ is integral. Therefore, Theorem 2.7 can have cost which depends on the average bitlength d of A and not the bitlength of the largest entry. The average bitlength d can assume any of the three definitions given by Theorem 3.7, Corollary 3.9 and Corollary 3.10.

Remark 3.13 (Application to inverting unimodular matrices)**.** *Suppose D is a partial linearization of a unimodular matrix $A \in \mathbb{Z}^{n\times n}$. A straight line formula for A^{-1} is given by*

$$\left[\begin{array}{c|c} I_n & 0 \end{array}\right] T \left[\begin{array}{c} I_n \\ \hline 0 \end{array}\right]$$

where T is a straight line formula for the inverse of D. Such a straight line formula for A^{-1} can thus be computed deterministically in $O(n^\omega\, \mathsf{M}(\log n + d) \log n)$ bit operations by Lemma 2.3 (Pauderis and Storjohann, 2012, Section 3), where d is the average bitlength of A according to any of the three definitions given by Theorem 3.7, Corollary 3.9 and Corollary 3.10.

Remark 3.14 (Application to computing the Hermite form)**.** *If $A \in \mathbb{Z}^{n\times n}$ is nonsingular, then the lower triangular row Hermite form of A shows up as the principal $n \times n$ submatrix of the Hermite form of the partially linearized matrix D.*

Example 3.15. *The lower triangular row Hermite form of the matrix D from Example 3.8 is*

$$\left[\begin{array}{cccc||ccc} 777 & 0 & 0 & 0 & 0 & 0 & 0 \\ 401 & 1 & 0 & 0 & 0 & 0 & 0 \\ 174 & 0 & 4911 & 0 & 0 & 0 & 0 \\ 762 & 0 & 0 & 765492351 & 0 & 0 & 0 \\ \hline\hline 696 & 0 & 3260 & 0 & 1 & 0 & 0 \\ 762 & 0 & 0 & 765475967 & 0 & 1 & 0 \\ 762 & 0 & 0 & 497056895 & 0 & 0 & 1 \end{array}\right]$$

with the 4×4 principal sub-matrix being the corresponding lower triangular row Hermite form of A.

Chapter 4

Smith massager

In this chapter, we introduce the Smith massager of a nonsingular integer matrix. We give the definition of the object, and we present important mathematical properties that follow from the definition.

As we already mentioned, any nonsingular integer matrix $A \in \mathbb{Z}^{n\times n}$ is unimodularly row and column equivalent to a unique diagonal matrix

$$S = \begin{bmatrix} s_1 & & \\ & \ddots & \\ & & s_n \end{bmatrix} \in \mathbb{Z}^{n\times n}.$$

Each s_i must be positive and they all form a divisibility sequence $s_1 \mid s_2 \mid \cdots \mid s_n$. Equivalent means that there exist unimodular matrices $U, V \in \mathbb{Z}^{n\times n}$ such that

$$AV = US. \tag{4.1}$$

A Smith massager matrix $M \in \mathbb{Z}^{n\times n}$ can be interpreted as a relaxed version of the unimodular matrix V. More specifically, matrix V in (4.1) must satisfy the following two properties.

(V1) The product AVS^{-1} is integral.

(V2) There exists another matrix $V' \in \mathbb{Z}^{n\times n}$ such that $V'V = I_n$.

A Smith massager M will satisfy the same properties, except that we relax condition (V2) with the equality only holding with the columns taken modulo the corresponding invariant factors.

(M1) The product AMS^{-1} is integral.

(M2) There exists another matrix $M' \in \mathbb{Z}^{n\times n}$ such that $M'M = I_n \bmod S$.

We use the notation $\bmod S$ to show that the ith column of a matrix or an equivalence is taken modulo the diagonal entry S_{ii}. Apart from the fact that a Smith massager M is closely related to a Smith multiplier V, it is easy to see that the ith column of the massager can always be reduced modulo s_i without violating properties (M1) and (M2). This is an important fact because it gives an explicit bound on the size of M.

Before we move on to a detailed discussion regarding the Smith massager, we will give a high level description on how we plan to compute such a massager in Chapter 5. Our idea for efficiently recovering the Smith form of a matrix A is inspired by the fact that if we let v be a random column vector, then the denominator of $A^{-1}v$ is likely to be the largest invariant factor s_n (see Wan (2005); Eberly et al. (2000); Pan (1988); Abbott et al. (1999)). Since

$$A^{-1}v = VS^{-1}U^{-1}v,$$

the denominator of $A^{-1}v$ will be the largest invariant factor s_n if the last entry of the vector $U^{-1}v$ is relatively prime to s_n (with U^{-1} unimodular). If that is the case, apart from the fact that we recover s_n, then, we also have that the vector $A^{-1}v$ is "equivalent" to the last column of VS^{-1}. The same idea can be extended to recover more invariant factors and columns "equivalent" to VS^{-1} by choosing more random column vectors.

After we compute the complete Smith form S of A, then we let the Smith massager $M \in \mathbb{Z}^{n\times n}$ be the matrix comprised of all those random projection vectors to the space of A corresponding to each one of the invariant factors. The massager will be such that MS^{-1} is "equivalent" to A^{-1}, or to be more specific, the matrix M will satisfy that

$$s_n A^{-1} \equiv_R M(s_n S^{-1}) \bmod s_n.$$

This means that matrices M and S can give a representation of the fractional part of A^{-1}, while asymptotically using almost the same space as the input matrix A. We use $\equiv_R$ to denote equivalence up to unimodular column operations over the integers $\mathbb{Z}$, or over a ring $\mathbb{Z}/(s)$ if the relation is given modulo s for some positive integer s.

We proceed by providing the main definition along with some column operations that keep the massager properties intact. Also, in Section 4.2, we discuss an important property of the Smith massager used for giving compact representations of the inverse of A, while, in Section 4.3, we present additional properties of massagers which will help us to compute Smith multipliers in Chapter 6. Finally, in Section 4.4, we show that the partial linearization technique of Chapter 3 also preserves the Smith massager of a matrix.

4.1 Definition

In this section, we properly define the Smith massager of a nonsingular integer matrix.

Definition 4.1. *Let $A \in \mathbb{Z}^{n\times n}$ be a nonsingular integer matrix with Smith form S. A matrix $M \in \mathbb{Z}^{n\times n}$ is a* Smith massager *for A if*

(i) it satisfies that

$$AM \equiv 0 \bmod S, \tag{4.2}$$

(ii) there exists a matrix $W \in \mathbb{Z}^{n\times n}$ such that

$$WM \equiv I_n \bmod S. \tag{4.3}$$

Property (i) of a Smith massager M implies that the matrix AMS^{-1} is integral, while property (ii) implies that M is unimodular up to the columns of S. As we already mentioned, any Smith massager reduced column modulo S is still a Smith massager. Therefore, if $M = (M \bmod S)$, then M is called a *reduced Smith massager*.

Example 4.2. *The Smith form of*

$$A = \begin{bmatrix} -6 & 3 & -13 & -15 \\ -4 & 19 & 12 & -1 \\ -4 & 10 & -6 & 17 \\ -26 & -13 & 1 & -2 \end{bmatrix}$$

is $S = \mathrm{diag}(1, 1, 9, 29088)$. *For*

$$M = \begin{bmatrix} 0 & 0 & 7 & 805 \\ 0 & 0 & 5 & 23668 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 4 & 10224 \end{bmatrix}.$$

we have $AM \equiv 0 \bmod S$. *Setting*

$$W = \begin{bmatrix} 4 & -19 & -12 & 1 \\ -306 & 3 & 133 & 0 \\ 5156 & 805 & 6332 & 0 \\ 12017 & -403 & 11356 & 0 \end{bmatrix}$$

gives

$$WM = I_4 + \begin{bmatrix} -1 & 0 & -99 & -436320 \\ 0 & -1 & -1728 & -174528 \\ 0 & 0 & 59112 & 23241312 \\ 0 & 0 & 116172 & 203616 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 9 & \\ & & & 29088 \end{bmatrix},$$

implying that $WM \equiv I_4 \bmod S$. *It follows that* M *is a Smith massager for* A.

It will be useful to notice that a Smith massager M for some matrix A remains a valid Smith massager under some specific columns operations.

Lemma 4.3. *Assume* $M \in \mathbb{Z}^{n\times n}$ *is a Smith massager for* A. *Then the matrix obtained from* M *by*

(i) adding any integer column vector multiplied by s_i *to column* i,

(ii) adding any multiple of a latter to a former column, or

(iii) multiplying (or dividing exactly) the i^{th} *column by an integer relatively prime to* s_i

is also a Smith massager for A.

Proof. For each one of these operations, we need to show that the modified matrix M still satisfies properties (i) and (ii) of Definition 4.1.

Let $\bar{M}$ be the matrix obtained from M by performing operation (i). Then $\bar{M} \equiv M \bmod S$ and thus $A\bar{M} \equiv 0 \bmod S$ and $W\bar{M} \equiv I_n \bmod S$ still hold.

For operation (ii), let $1 \leq i_1 < i_2 \leq n$ and $c \in \mathbb{Z}$. Let $\bar{M}$ be the matrix obtained from M by adding c times column i_2 to column i_1. Because $s_{i_1} \mid s_{i_2}$, $A\bar{M} \equiv 0 \bmod S$ still holds. Let $\bar{W}$ be the matrix obtained from W by adding $-c$ times row i_1 to row i_2. Then $\bar{W}\bar{M} \equiv I_n \bmod S$.

For operations (iii), let $c \in \mathbb{Z}$ be relatively prime to s_i. Let $\bar{M}$ be the matrix obtained from M by multiplying column i by c. Then $A\bar{M} \equiv 0 \bmod S$ still holds. Let $\bar{W}$ be the matrix obtained from M by multiplying row i by $\mathrm{Rem}(1/c, s_n) \in \mathbb{Z}$. Then $\bar{W}\bar{M} \equiv I_n \bmod S$. The case for $1/c$ is similar. □

4.2 Compact representation of A^{-1}

An important property of Smith massagers is that the column space of the inverse of a matrix $A \in \mathbb{Z}^{n\times n}$ can be represented by the Smith massager M with each column divided by the corresponding invariant factor of A, that is, MS^{-1}.

Theorem 4.4. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form S and Smith massager M.*

(i) For any row vector $v \in \mathbb{Z}^{1\times n}$, vA^{-1} is integral if and only if vMS^{-1} is integral.

(ii) It is true that

$$s_n A^{-1} \equiv_R M(s_n S^{-1}) \bmod s_n. \tag{4.4}$$

Proof. Let

$$B = \begin{bmatrix} A & \\ & I_n \end{bmatrix} \begin{bmatrix} I_n & \\ -W & I_n \end{bmatrix} \begin{bmatrix} I_n & M \\ & I_n \end{bmatrix} \begin{bmatrix} & I_n \\ S^{-1} & \end{bmatrix} = \begin{bmatrix} AMS^{-1} & A \\ (I_n - WM)S^{-1} & -W \end{bmatrix}.$$

By Definition 4.1 and since $|\det A| = \det S \neq 0$, matrix B is integral and unimodular. If we pre-multiply B by $\operatorname{diag}(A^{-1}, I_n)$ and then restrict to the first n rows, we obtain

$$\begin{bmatrix} A^{-1} & \end{bmatrix} B = \begin{bmatrix} MS^{-1} & I_n \end{bmatrix}. \tag{4.5}$$

Since both B and B^{-1} are integral, we conclude that for any $v \in \mathbb{Z}^{1\times n}$, vA^{-1} is integral if and only if vMS^{-1} is integral. Furthermore, if we multiply (4.5) by the largest invariant factor s_n, then the equation is over the integers, and, by taking it modulo s_n, we conclude

$$s_n A^{-1} \equiv_R M(s_n S^{-1}) \bmod s_n.$$

□

Corollary 4.5. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form S and Smith massager M. For any row vector $v \in \mathbb{Z}^{1\times n}$, the denominator of vA^{-1} equals the denominator of vMS^{-1}.*

A key observation from Theorem 4.4 is that MS^{-1} can be represented with only $O(n^2(\log n + \log \|A\|))$ bits. This compares to $O(n^3(\log n + \log \|A\|))$ bits required by A^{-1}. The property is also useful for computing left equivalent canonical forms of A.

Example 4.6. *Matrix*

$$A = \begin{bmatrix} -6 & 3 & -13 & -15 \\ -4 & 19 & 12 & -1 \\ -4 & 10 & -6 & 17 \\ -26 & -13 & 1 & -2 \end{bmatrix},$$

from Example 4.2, has Smith form $S = \operatorname{diag}(1, 1, 9, 29088)$ *and Smith massager*

$$M = \begin{bmatrix} 0 & 0 & 7 & 805 \\ 0 & 0 & 5 & 23668 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 4 & 10224 \end{bmatrix}.$$

Then,

$$A^{-1} = \frac{1}{29088} \begin{bmatrix} -271 & -402 & -373 & -937 \\ 580 & 920 & 524 & -356 \\ -1074 & 804 & -870 & 258 \\ -784 & -352 & 1008 & 80 \end{bmatrix},$$

and from Corollary 4.5, for any row vector $v \in \mathbb{Z}^{1\times n}$*, the denominator of* vA^{-1} *equals the denominator of*

$$v \begin{bmatrix} 7 & 805 \\ 5 & 23668 \\ 3 & 6 \\ 4 & 10224 \end{bmatrix} \begin{bmatrix} 1/9 & \\ & 1/29088 \end{bmatrix},$$

where the first two columns can be omitted because the corresponding invariant factors are 1*. Equivalently, from equation (4.4), we have that*

$$\begin{bmatrix} -271 & -402 & -373 & -937 \\ 580 & 920 & 524 & -356 \\ -1074 & 804 & -870 & 258 \\ -784 & -352 & 1008 & 80 \end{bmatrix} \equiv_R \begin{bmatrix} 7 & 805 \\ 5 & 23668 \\ 3 & 6 \\ 4 & 10224 \end{bmatrix} \begin{bmatrix} 3232 & \\ & 1 \end{bmatrix} \bmod 29088.$$

The equivalence between the inverse of A and the Smith massager proves to be very useful for computing left equivalent matrix canonical forms for a matrix $A \in \mathbb{Z}^{n\times n}$. Two matrices are said to be *left equivalent* if one can be obtained from the other by premultiplying with a unimodular matrix. The following theorem is a corollary of Theorem 4.4.

Theorem 4.7. *Let* $A \in \mathbb{Z}^{n\times n}$ *be nonsingular with Smith form* S *and a Smith massager* M*. If a matrix* $H \in \mathbb{Z}^{n\times n}$ *satisfies that* $|\det H| = \det S$ *and*

$$HM \equiv 0 \,\mathbf{c}\mathrm{mod}\, S,$$

then H is left equivalent to A.

In other words, the Smith form S and a Smith massager M can be used to describe an left equivalent canonical form of a matrix A in a compact and fraction-free way. We will use Theorem 4.7 in Section 6.2.

4.3 Approach to create a unimodular Smith massager

In this section, we present some statements regarding Smith massagers which are perturbed by a matrix post-multiplied by the Smith form and regarding the Hermite form of Smith massagers. We will need these statements later in Chapter 6 where we compute Smith multiplier matrices by creating a unimodular Smith massager.

In order to motivate the lemmas that follow, we will also describe, in parallel, our approach to computing a unimodular Smith massager for a matrix A, given an algorithm that computes a reduced Smith massager. The lemmas that we present here, along with Lemma 4.3, constitute the main ingredients of the correctness of our process in Chapter 6. Chapter 6 will mainly be devoted to establishing the efficiency of our algorithm.

Our approach involves first obtaining a nonsingular Smith massager for the input matrix A and then perturbing it by a matrix post-multiplied by the Smith form. We obtain a nonsingular Smith massager by computing a reduced Smith massager M for the matrix $2A$. From Proposition 4.8, one sees that a Smith massager for $2A$ has full rank over $\mathbb{Z}/(2)$, and thus, is nonsingular over $\mathbb{Z}$. We then perturb M by a matrix R post-multiplied by the Smith form $2S$, with Proposition 4.8(i) ensuring that $M + 2RS$ is still a Smith massager for A. In Section 6.1, we will prove that if R is chosen to be a random matrix, then the Hermite form of $M + 2RS$ will have only one non-trivial column with high probability, something which will be important for efficiency purposes.

Proposition 4.8. *Let $c \in \mathbb{Z}_{>0}$ and $A \in \mathbb{Z}^{n\times n}$. If $M \in \mathbb{Z}^{n\times n}$ is a Smith massager for cA, then for any matrix $R \in \mathbb{Z}^{n\times n}$:*

(i) $M + R(cS)$ is a Smith massager for A,

(ii) the last i columns of $M + R(cS)$ have full rank over $\mathbb{Z}/(p)$ for any p that divides (cs_{n-i+1}).

The proof of Proposition 4.8 follows directly from the next two lemmas.

Lemma 4.9. *Let $c \in \mathbb{Z}_{>0}$ and $A \in \mathbb{Z}^{n\times n}$. If $M \in \mathbb{Z}^{n\times n}$ is a Smith massager for cA, then M is also a Smith massager for A.*

Proof. First note that if $S \in \mathbb{Z}^{n\times n}$ is the Smith form of A, then cS is the Smith form of cA. Since M is a Smith massager for cA, Definition 4.1 states that

$$cAM \equiv 0 \,\mathbf{c}\mathrm{mod}\, cS, \tag{4.6}$$

and that there exists a $W \in \mathbb{Z}^{n\times n}$ such that

$$WM \equiv I_n \,\mathbf{c}\mathrm{mod}\, cS. \tag{4.7}$$

It follows from (4.6) that $AM \equiv 0 \,\mathbf{c}\mathrm{mod}\, S$ and from (4.7) that $WM \equiv I_n \,\mathbf{c}\mathrm{mod}\, S$, and thus by Definition 4.1, M is a Smith massager for A. □

Lemma 4.10. *For any prime p that divides s_{n-i+1}, the last i columns of a Smith massager M have full rank over $\mathbb{Z}/(p)$.*

Proof. The claim follows from Definition 4.1 of the Smith massager since

$$\begin{aligned} WM &\equiv I_n \,\mathbf{c}\mathrm{mod} \begin{bmatrix} s_1 & & \\ & \ddots & \\ & & s_n \end{bmatrix} \\ &\equiv I_n \,\mathbf{c}\mathrm{mod} \begin{bmatrix} s_1 & & & & & \\ & \ddots & & & & \\ & & s_{n-i} & & & \\ & & & p & & \\ & & & & \ddots & \\ & & & & & p \end{bmatrix}. \end{aligned}$$

If the last i columns of $WM \bmod p$ have full rank, then the last i columns of $M \bmod p$ also have full rank. □

The second part of our process involves computing a lower triangular row Hermite form of the perturbed and nonsingular Smith massager $M + 2RS$, and then, obtaining a unimodular Smith massager by post-multiplying with the inverse of that Hermite form.

We define the *lower triangular row Hermite form* of a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ to be the unique matrix

$$H = \begin{bmatrix} h_1 & & & \\ * & h_2 & & \\ \vdots & \vdots & \ddots & \\ * & * & \cdots & h_n \end{bmatrix} \in \mathbb{Z}^{n\times n}$$

that is left equivalent to A, it has positive diagonal entries, and the off-diagonal entries in each column are reduced by the diagonal entries.

Lemma 4.12 provides the final ingredient for our correctness argument by proving that a nonsingular Smith massager for A, post-multiplied by the inverse of its lower triangular row Hermite form, is still a Smith massager for A, and trivially unimodular. Lemma 4.11 is an intermediate result.

Lemma 4.11. *Let $M \in \mathbb{Z}^{n\times n}$ be a Smith massager that is nonsingular and S the corresponding Smith form. If h_i is the i^{th} diagonal entry of the lower row Hermite form H of M, then $gcd(h_i, s_i) = 1$.*

Proof. The lemma follows from the fact that a matrix and its row Hermite form share the same column rank profile. Therefore, since by Lemma 4.10, the last i columns of M have full rank over $\mathbb{Z}/(p)$ for any $p \mid s_{n-i+1}$, then the last i columns of H have full rank over $\mathbb{Z}/(p)$, and thus, $p \nmid h_{n-i+1}$. □

Lemma 4.12. *Let $M \in \mathbb{Z}^{n\times n}$ be a Smith massager that is nonsingular for a matrix A, and let $H \in \mathbb{Z}^{n\times n}$ be the lower triangular row Hermite form of M. Then, MH^{-1} is a unimodular Smith massager for A.*

Proof. The inverse of any lower triangular matrix can be decomposed as follows.

$$H^{-1} = \prod_{i=0}^{n-1} \begin{bmatrix} I & & & \\ & 1 & & \\ & -h_{n-i+1,n-i} & 1 & \\ & \vdots & & \ddots \\ & -h_{n,n-i} & & 1 \end{bmatrix} \begin{bmatrix} I & & & \\ & 1/h_{n-i} & & \\ & & 1 & \\ & & & \ddots \\ & & & & 1 \end{bmatrix} \tag{4.8}$$

Therefore, multiplying M with H^{-1} can be represented as a series of n products, where each multiplication first applies an operation of the type as described in Lemma 4.3(ii), and second applies one of the type as in Lemma 4.3(iii) as certified by Lemma 4.11. Therefore, MH^{-1} is a unimodular Smith massager for A. □

4.4 Smith massager and partial linearization

As with the algorithms from Chapter 2, if we have an algorithm which, for a matrix $A \in \mathbb{Z}^{n\times n}$, computes the Smith form and a Smith massager with cost that depends on

$\log \|A\|$, we can employ the partial linearization technique to replace the $\log \|A\|$ term with the average bitlength d of the columns (or rows) in A. This fact will be exploited when computing Smith multiplier matrices in Chapter 6.

Theorem 4.13. *Let $A \in \mathbb{Z}^{n\times n}$ and $D \in \mathbb{Z}^{\bar{n}\times\bar{n}}$ be the partially linearized version of A from Theorem 3.7. If*

$$\left(\begin{bmatrix} I_{\bar{n}-n} & \\ & S \end{bmatrix}, \begin{bmatrix} 0 & M_1 \\ 0 & M_2 \end{bmatrix} \right)$$

is a Smith massager for D, where $S \in \mathbb{Z}^{n\times n}$, $M_1 \in \mathbb{Z}^{n\times n}$ and $M_2 \in \mathbb{Z}^{(\bar{n}-n)\times n}$, then (S, M_1) is a Smith massager for A.

Proof. We will show that $S, M_1 \in \mathbb{Z}^{n\times n}$ satisfy Definition 4.1 for A.

From Theorem 3.7, we have that

$$\begin{aligned} D \begin{bmatrix} 0 & M_1 \\ 0 & M_2 \end{bmatrix} &= \begin{bmatrix} I_n & Q \\ & I_{\bar{n}-n} \end{bmatrix} \begin{bmatrix} A & \\ & I_{\bar{n}-n} \end{bmatrix} \begin{bmatrix} I_n & \\ E & F \end{bmatrix} \begin{bmatrix} 0 & M_1 \\ 0 & M_2 \end{bmatrix} \\ &= \begin{bmatrix} I_n & Q \\ & I_{\bar{n}-n} \end{bmatrix} \begin{bmatrix} 0 & AM_1 \\ 0 & EM_1 + FM_2 \end{bmatrix}. \end{aligned}$$

Therefore, from the first property of the Smith massager for D,

$$D \begin{bmatrix} 0 & M_1 \\ 0 & M_2 \end{bmatrix} \equiv 0 \bmod \begin{bmatrix} I_{\bar{n}-n} & \\ & S \end{bmatrix},$$

it follows that

$$\begin{bmatrix} 0 & AM_1 \\ 0 & EM_1 + FM_2 \end{bmatrix} \equiv 0 \bmod \begin{bmatrix} I_{\bar{n}-n} & \\ & S \end{bmatrix},$$

and that

$$AM_1 \equiv 0 \bmod S.$$

Moreover, since B is unit lower triangular, we see that

$$M_2 \equiv -F^{-1}EM_1 \bmod S.$$

Finally, from the second property of the Smith massager for D, there exist a matrix $W_D \in \mathbb{Z}^{\bar{n}\times\bar{n}}$ such that

$$W_D \begin{bmatrix} 0 & M_1 \\ 0 & M_2 \end{bmatrix} \equiv \begin{bmatrix} I_{\bar{n}-n} & \\ & I_n \end{bmatrix} \bmod \begin{bmatrix} I_{\bar{n}-n} & \\ & S \end{bmatrix}.$$

The last equation can be transformed to

$$\left(W_D \begin{bmatrix} I_n & \\ -F^{-1}E & I_{\bar{n}-n} \end{bmatrix}\right) \begin{bmatrix} 0 & M_1 \\ 0 & 0 \end{bmatrix} \equiv \begin{bmatrix} I_{\bar{n}-n} & \\ & I_n \end{bmatrix} \mathbf{c}\mathrm{mod} \begin{bmatrix} I_{\bar{n}-n} & \\ & S \end{bmatrix},$$

from which it directly follows that there exists a matrix $W \in \mathbb{Z}^{n\times n}$ such that

$$WM_1 \equiv I_n \,\mathbf{c}\mathrm{mod}\, S.$$

□

Furthermore, by equation (3.5) and by following the same steps as in the proof Theorem 4.13, we obtain the following Corollary.

Corollary 4.14. *Let $A \in \mathbb{Z}^{n\times n}$ and $D \in \mathbb{Z}^{\bar{n}\times\bar{n}}$ be the partially linearized version of A from Corollary 3.9 or Corollary 3.10. If*

$$\left(\begin{bmatrix} I_{\bar{n}-n} & \\ & S \end{bmatrix}, \begin{bmatrix} 0 & M_1 \\ 0 & M_2 \end{bmatrix}\right)$$

is a Smith massager for D, where $S \in \mathbb{Z}^{n\times n}$, $M_1 \in \mathbb{Z}^{n\times n}$ and $M_2 \in \mathbb{Z}^{(\bar{n}-n)\times n}$, then (S, M_1) is a Smith massager for A.

Chapter 5

Smith normal form and Smith massager algorithm

In this chapter, we present a Las Vegas randomized algorithm to compute the Smith normal form of a nonsingular integer matrix. As we already mentioned, any nonsingular integer matrix $A \in \mathbb{Z}^{n\times n}$ is unimodularly row and column equivalent to a unique diagonal matrix

$$S = \begin{bmatrix} s_1 & & \\ & \ddots & \\ & & s_n \end{bmatrix} \in \mathbb{Z}^{n\times n}.$$

Each s_i must be positive and they all form a divisibility sequence $s_1 \mid s_2 \mid \cdots \mid s_n$. Along with the Smith form S, the algorithm will also return a matrix $M \in \mathbb{Z}^{n\times n}$ which is a Smith massager as defined in Definition 4.1.

In order to recover all of the invariant factors of A efficiently, we utilize a "dimension $\times$ precision $\leq$ invariant" compromise such as the one required by the algorithms in Chapter 2. By Hadamard's bound, $|\det A| = s_1 s_2 \cdots s_n \leq \Delta^n$ where $\Delta = n^{1/2}||A||$. Thus, the number of invariant factors with bitlength between $(1/2^i)\times n \log \Delta$ and $(1/2^{i-1})\times n \log \Delta$ is bounded by 2^i. So, we compute one invariant factor with bitlength at most $n \log \Delta$, then two with bitlength at most $n \log \Delta/2$, then four with bitlength at most $n \log \Delta/4$, and so on. We will need only $O(\log n)$ iterations of this type.

In order to facilitate certification of the output, we take a structured approach and compute a *full Smith massager* for A. This is a tuple of $n \times n$ integer matrices (U, M, T, S) such that

$$B = \begin{bmatrix} A & \\ & I_n \end{bmatrix} \begin{bmatrix} I_n & \\ U & I_n \end{bmatrix} \begin{bmatrix} I_n & M \\ & T \end{bmatrix} \begin{bmatrix} I_n & \\ & S^{-1} \end{bmatrix} \in \mathbb{Z}^{2n\times 2n} \tag{5.1}$$

is integral, with T unit upper triangular and S nonsingular and in Smith form. The algorithm succeeds if we compute a *maximal Smith massager*, meaning that S is the Smith form of A. Since (5.1) implies that $(\det B)(\det S) = \det A$, we can conclude from the uniqueness of the Smith form that the massager is maximal if and only if B is unimodular.

In addition, from equation (5.1), the matrix

$$\begin{bmatrix} A & AM \\ U & UM+T \end{bmatrix} \begin{bmatrix} I_n & \\ & S^{-1} \end{bmatrix} \tag{5.2}$$

is integral. This entails that

$$AM \equiv 0 \,\mathbf{c}\mathrm{mod}\, S$$

and

$$UM + T \equiv 0 \,\mathbf{c}\mathrm{mod}\, S$$

which, since T is unit upper triangular, is equivalent to

$$(-T^{-1}U)M \equiv I_n \,\mathbf{c}\mathrm{mod}\, S.$$

Therefore, if the full Smith massager is maximal, then the matrix M is a Smith massager for A.

Example 5.1. *The matrix*

$$A = \begin{bmatrix} -13 & 10 & -20 & 27 \\ 27 & 30 & 15 & 30 \\ 0 & 15 & 15 & 6 \\ -21 & 0 & -15 & 9 \end{bmatrix}$$

is equivalent to the Smith form

$$S = \begin{bmatrix} 1 & & & \\ & 3 & & \\ & & 15 & \\ & & & 105 \end{bmatrix}.$$

Moreover, a Smith massager for A is given by

$$M = \begin{bmatrix} 0 & 2 & 0 & 55 \\ 0 & 0 & 7 & 32 \\ 0 & 2 & 2 & 41 \\ 0 & 2 & 10 & 10 \end{bmatrix}.$$

Our algorithm for computing the Smith form and Smith massager consists of three phases. Phase 1 obtains a random lifting modulus X for all linear system solving procedures, and it uses the Monte Carlo approach of Eberly et al. (2000, Theorem 2.1) to compute the largest invariant factor s_n of A. Phase 2 iteratively computes a full Smith massager for A, together with the massaged matrix B in (5.1). Phase 3 uses a known algorithm, by Pauderis and Storjohann (2012) that we recall in Lemma 2.3, to assay whether B is unimodular and certify the output.

Phase 2 is the main part of our algorithm. It uses $O(\log n)$ iterations to build a full Smith massager that extracts more and more invariant factors from A. The algorithm begins by initializing (U, M, T, S) to be the trivial full Smith massager, with $U = M = 0_{n\times n}$ and $T = S = I_n$. At the start of iteration $i = 0, 1, 2, \ldots$ we assume that the current full Smith massager is such that B in (5.1) has the same Smith form as A but with the largest $2^i - 1 = 2^0 + 2^1 + \cdots + 2^{i-1}$ invariant factors replaced by 1. The goal at iteration i is then to compute and extract the next largest 2^i invariant factors. For example, at iterations $i = 0, 1$ and 2, the largest $1, 2$ and 4 invariant factors of the current B are equal to (s_n), (s_{n-1}, s_{n-2}) and $(s_{n-3}, s_{n-4}, s_{n-5}, s_{n-6})$, respectively.

Note that the matrix U of the full Smith massager is not expected to be unimodular and should not be confused with a unimodular Smith multiplier U. Apart from M being a Smith massager for A when the algorithm succeeds, the rest of the full Smith massager structure is mainly used to efficiently extract invariant factors.

Example 5.2. *For the matrix A of Example 5.1, we begin by initializing*

$$B_0 := \operatorname{diag}(A, I_n) = \begin{bmatrix} -13 & 10 & -20 & 27 & & & & \\ 27 & 30 & 15 & 30 & & & & \\ 0 & 15 & 15 & 6 & & & & \\ -21 & 0 & -15 & 9 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{bmatrix}.$$

The Smith form $S = \operatorname{diag}(*, *, *, *)$ of A (or the Smith form of B_0 with n additional trivial invariant factors) is still unknown and $\det B_0 = \det A$.

In the first iteration, we compute the largest invariant factor s_4 and we apply unimodular

column operations on B_0 to make its last column a multiple of s_4.

$$B_0' = \begin{bmatrix} -13 & 10 & -20 & 27 & & & & 0 \\ 27 & 30 & 15 & 30 & & & & 43 \cdot 105 \\ 0 & 15 & 15 & 6 & & & & 9 \cdot 105 \\ -21 & 0 & -15 & 9 & & & & -15 \cdot 105 \\ & & & & 1 & & & 0 \\ & & & & & 1 & & 0 \\ & & & & & & 1 & 0 \\ 0 & 41 & 17 & 67 & & & & 40 \cdot 105 \end{bmatrix}$$

Then, by removing that factor we have

$$B_1 := \begin{bmatrix} -13 & 10 & -20 & 27 & & & & 0 \\ 27 & 30 & 15 & 30 & & & & 43 \\ 0 & 15 & 15 & 6 & & & & 9 \\ -21 & 0 & -15 & 9 & & & & -15 \\ & & & & 1 & & & 0 \\ & & & & & 1 & & 0 \\ & & & & & & 1 & 0 \\ 0 & 41 & 17 & 67 & & & & 40 \end{bmatrix}$$

with

$$S = \operatorname{diag}(*, *, *, 105) \text{ and } \det B_1 = \det A/105.$$

In the second iteration, we recover the two largest invariant factors of B_1 which are s_2, s_3 and we repeat the same transformation to obtain

$$B_1' = \begin{bmatrix} -13 & 10 & -20 & 27 & & -2 \cdot 3 & 12 \cdot 15 & 0 \\ 27 & 30 & 15 & 30 & & 24 \cdot 3 & 46 \cdot 15 & 43 \\ 0 & 15 & 15 & 6 & & 7 \cdot 3 & 16 \cdot 15 & 9 \\ -21 & 0 & -15 & 9 & & -9 \cdot 3 & -5 \cdot 15 & -15 \\ & & & & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 0 & & 1 \cdot 3 & 1 \cdot 15 & 0 \\ 0 & 8 & 0 & 0 & & 0 & 7 \cdot 15 & 0 \\ 0 & 41 & 17 & 67 & & 28 \cdot 3 & 59 \cdot 15 & 40 \end{bmatrix}$$

and

$$B_2 := \left[\begin{array}{cccc|c|ccc} -13 & 10 & -20 & 27 & & -2 & 12 & 0 \\ 27 & 30 & 15 & 30 & & 24 & 46 & 43 \\ 0 & 15 & 15 & 6 & & 7 & 16 & 9 \\ -21 & 0 & -15 & 9 & & -9 & -5 & -15 \\ \hline & & & & 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 2 & 0 & & 1 & 1 & 0 \\ 0 & 8 & 0 & 0 & & 0 & 7 & 0 \\ 0 & 41 & 17 & 67 & & 28 & 59 & 40 \end{array}\right]$$

with

$$S = \mathrm{diag}(*, 3, 15, 105) \text{ and } \det B_2 = \det A/(105 \cdot 15 \cdot 3).$$

Finally,

$$B := \left[\begin{array}{cccc|c|ccc} -13 & 10 & -20 & 27 & 0 & -2 & 12 & 0 \\ 27 & 30 & 15 & 30 & 0 & 24 & 46 & 43 \\ 0 & 15 & 15 & 6 & 0 & 7 & 16 & 9 \\ -21 & 0 & -15 & 9 & 0 & -9 & -5 & -15 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 1 & 2 & 0 & 0 & 1 & 1 & 0 \\ 0 & 8 & 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 41 & 17 & 67 & 0 & 28 & 59 & 40 \end{array}\right]$$

with

$$S = \mathrm{diag}(1, 3, 15, 105) \text{ and } \det B = -1.$$

The algorithm requires $O(n^3(\log n + \log \|A\|)^2(\log n)^2)$ bit operations using standard integer and matrix arithmetic. By employing fast integer and matrix multiplication, we establish that the Smith form and the Smith massager can be computed in about the same number of bit operations as required to multiply two matrices of the same dimension and size of entries as the input matrix. Specifically, the time compexity of our algorithm is in

$$O(n^\omega \, \mathsf{B}(\log n + \log \|A\|)(\log n)^2),$$

where ω is the matrix multiplication exponent and $\mathsf{B}(d)$ bounds the cost of gcd-related operations on integers with bitlength at most d.

Section 5.1 shows how to recover the largest 2^i invariant factors of B with high probability by computing a projection $B^{-1}J$ for a randomly chosen J with column dimension $O(2^i)$. We exploit the fact that if s is a multiple of the largest invariant factor of B, then the smallest 2^i invariant factors of sB^{-1} correspond to the largest 2^i invariant factors of B.

In Sections 5.2 and 5.3, we show how to compute an index Smith massager that will extract the largest 2^i invariant factors from B, while, in Section 5.4, we show how to combine the partial index Smith massagers obtained at each iteration.

5.1 Largest invariant factors

In this section we show how the largest r invariant factors of a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ can be recovered with high probability by randomly sampling $r + O(\log r)$ vectors from the columns space of A^{-1}. The method assumes we know an $s \in \mathbb{Z}_{>0}$ that is a multiple of the largest invariant factor s_n of A.

Let $U, V \in \mathbb{Z}^{n\times n}$ be unimodular such that $AV = US$ where $S = \operatorname{diag}(s_1, \ldots, s_n)$ is the Smith form of A. Then, the *reverse Smith form* of $sA^{-1} \in \mathbb{Z}^{n\times n}$ is equal to

$$sS^{-1} = \begin{bmatrix} s/s_1 & & \\ & \ddots & \\ & & s/s_n \end{bmatrix}.$$

By reverse Smith form we simply mean that the order of both the rows and the columns is reversed. The smallest invariant factor is, thus, located in the last row and column.

Example 5.3. *The reverse Smith form of the* 4×3 *matrix*

$$\begin{bmatrix} 1 & & \\ & 2 & \\ & & \\ & & \end{bmatrix}$$

is equal to

$$\begin{bmatrix} & & \\ & 2 & \\ & & 1 \end{bmatrix}.$$

Since the largest invariant factor s/s_1 of the Smith form of sA^{-1} is a divisor of s, the (reverse) Smith form of sA^{-1} can be computed modulo s over $\mathbb{Z}/(s)$. For convenience, should a diagonal entry in the Smith form over $\mathbb{Z}/(s)$ vanish modulo s, we replace it with s. For example, the reverse Smith form of $\operatorname{diag}(1, 2, 8, 16, 16)$ over $\mathbb{Z}/(16)$ is equal to $\operatorname{diag}(16, 16, 8, 2, 1)$.

To recover only the largest r invariant factors of A, the idea is to choose a matrix $J \in \mathbb{Z}/(s)^{n\times r}$ uniformly at random and hope that the submatrix comprised of the last r rows of the reverse Smith form of $sA^{-1}J \in \mathbb{Z}/(s)^{n\times r}$ is equal to

$$S_1 = \begin{bmatrix} s/s_{n-r+1} & & \\ & \ddots & \\ & & s/s_n \end{bmatrix},$$

where $s_{n-r+1}, \ldots, s_n$ are the r largest invariant factors of A. To ensure a high probability of success, we adjust the recipe slightly by augmenting J with a small number of additional columns k. The main result of this section is:

Theorem 5.4. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form $S = \operatorname{diag}(s_1, \ldots, s_n)$. Let $s \in \mathbb{Z}_{>0}$ be a multiple of s_n. If $J \in \mathbb{Z}/(s)^{n\times(r+k)}$ is chosen uniformly at random for $r \geq 1$ and $k \geq 2$, then the trailing $r \times r$ submatrix of the reverse Smith form of $sA^{-1}J$ over $\mathbb{Z}/(s)$ is equal to $S_1 = \operatorname{diag}(s/s_{n-r+1}, \ldots, s/s_n)$ with probability at least*

$$1 - \frac{1}{2^{k-1}}.$$

Before we prove Theorem 5.4, we establish a property of $J \in \mathbb{Z}/(s)^{n\times(r+k)}$ that is sufficient to ensure success. In the following lemma, recall that $U, V \in \mathbb{Z}^{n\times n}$ are unimodular matrices such that $AV = US$, and thus $sA^{-1} = VsS^{-1}U^{-1}$.

Lemma 5.5. *If the $r \times (r+k)$ submatrix comprised of the last r rows of $U^{-1}J \in \mathbb{Z}^{n\times(r+k)}$ is right equivalent to $\left[\begin{array}{c|c} 0_{r\times k} & I_r \end{array}\right]$ over $\mathbb{Z}/(s)$, then the trailing $r \times r$ submatrix of the reverse Smith form of $sA^{-1}J$ over $\mathbb{Z}/(s)$ is equal to $S_1 = \operatorname{diag}(s/s_{n-r+1}, \ldots, s/s_n)$.*

Proof. Decompose

$$sS^{-1} = \begin{bmatrix} S_2 & \\ & S_1 \end{bmatrix},$$

where S_1 is as in the statement of the theorem, and $S_2 = \operatorname{diag}(s/s_1, \ldots, s/s_{n-r})$.

We work entirely over $\mathbb{Z}/(s)$. By assumption, we have that

$$U^{-1}J \equiv_R \left[\begin{array}{c|c} U_1 & U_2 \\ \hline & I_r \end{array}\right] \tag{5.3}$$

for $U_1 \in \mathbb{Z}/(s)^{(n-r)\times k}$ and $U_2 \in \mathbb{Z}/(s)^{(n-r)\times r}$. Since all entries in S_2 are divisible by the largest invariant factor s/s_{n-r+1} of S_1, it will be sufficient to show that $sA^{-1}J$ is row and column equivalent to

$$\left[\begin{array}{c|c} S_2U_1 & \\ \hline & S_1 \end{array}\right]$$

over $\mathbb{Z}/(s)$. We have that

$$\begin{aligned} sA^{-1}J &= V(sS^{-1})U^{-1}J && (5.4)\\ &\equiv_L (sS^{-1})U^{-1}J && (5.5)\\ &\equiv_R (sS^{-1})\left[\begin{array}{c|c} U_1 & U_2 \\ \hline & I_r \end{array}\right] && (5.6)\\ &= \left[\begin{array}{c|c} S_2U_1 & S_2U_2 \\ \hline & S_1 \end{array}\right] \\ &\equiv_L \left[\begin{array}{c|c} S_2U_1 & \\ \hline & S_1 \end{array}\right], && (5.7) \end{aligned}$$

where (5.4) follows from $AV = US$, (5.5) because V is unimodular, and (5.6) from (5.3). To obtain (5.7), we can use a unimodular left transformation to zero out the block S_2U_2 since its entries are all multiples of the diagonal entries in S_1. □

We need two additional technical lemmas before proving the main theorem.

Lemma 5.6. *If $k \geq 1$, $t \geq 0$ and $0 < x \leq 1/2$, then*

$$\prod_{i=k}^{k+t} \left(1 - x^i\right) \geq 1 - 2x^k + x^{k+t}.$$

Proof. We will use induction on t. For $t = 0$ the inequality is trivially true. We assume that $\prod_{i=k}^{k+t} (1 - x^i) \geq 1 - 2x^k + x^{k+t}$ for a fixed t, and we need to show the same for $t + 1$.

$$\begin{aligned} \prod_{i=k}^{k+t+1} \left(1 - x^i\right) &= (1 - x^{k+t+1}) \prod_{i=k}^{k+t} \left(1 - x^i\right) \\ &\geq (1 - x^{k+t+1})(1 - 2x^k + x^{k+t}) \\ &= 1 - 2x^k + x^{k+t} - x^{k+t+1} + 2x^{2k+t+1} - x^{2(k+t)+1} \\ &= 1 - 2x^k + x^{k+t+1}\left(1 + \frac{1}{x} - 2 + 2x^k - x^{k+t}\right) \\ &\geq 1 - 2x^k + x^{k+t+1}. \end{aligned}$$

In the last step we used that $x \leq 1/2$. □

Lemma 5.7. *If $k \geq 2$, then*

$$\zeta(k+1) - 1 < 2^{-k},$$

where ζ denotes the Riemann zeta function.

Proof. The lemma inequality is equivalent to:

$$\zeta(k+1) - 1 < 2^{-k} \Leftrightarrow \sum_{n=2}^{\infty} \frac{1}{n^{k+1}} < 2^{-k} \Leftrightarrow \sum_{n=2}^{\infty} \left(\frac{2}{n}\right)^{k+1} < 2.$$

Since the left-hand side of the last inequality is a decreasing function on k, it suffices to show the claim for $k = 2$, i.e., $\zeta(3) - 1 < \frac{1}{4}$. □

Proof (of Theorem 5.4). We start by defining the following event.

FR_p: For a prime p that divides s, the last r rows of the random matrix $J \in (\mathbb{Z}/(s))^{n\times(r+k)}$ have full row rank over $\mathbb{Z}/(p)$.

If the last i rows of J over $\mathbb{Z}/(p)$ are linearly independent, then they span a vector space containing p^i rows. The probability that an additional row avoids that space is $1 - p^i/p^{r+k}$, and, thus,

$$\Pr[\mathsf{FR}_p] = \prod_{j=k+1}^{r+k} \left(1 - \frac{1}{p^j}\right).$$

The above statement has already been shown and extensively used in the literature (Blömer et al., 1997; Cooper, 2000). Furthermore, by applying Lemma 5.6, we obtain

$$\Pr[\neg\mathsf{FR}_p] \leq 2\frac{1}{p^{k+1}}. \tag{5.8}$$

Next, we define the event described by Lemma 5.5.

FR_U: For a matrix $U \in \mathbb{Z}^{n\times n}$, the last r rows of the random matrix $UJ \in (\mathbb{Z}/(s))^{n\times(r+k)}$ are right equivalent to $\left[\begin{array}{c|c} 0_{r\times k} & I_r \end{array}\right]$ over $\mathbb{Z}/(s)$.

A matrix J is right equivalent to $\left[\begin{array}{c|c} 0_{r\times k} & I_r \end{array}\right]$ over $\mathbb{Z}/(s)$ if and only if it has full row rank over $\mathbb{Z}/(p)$ for all primes p that divide s. Therefore,

$$\begin{aligned}
\Pr[\neg \mathsf{FR}_{I_n}] &\leq \sum_{\substack{p \mid s \\ p \text{ prime}}} \Pr[\neg \mathsf{FR}_p] && (5.9)\\
&\leq 2 \sum_{p=2}^{\infty} \frac{1}{p^{k+1}} && (5.10)\\
&= 2(\zeta(k+1) - 1) \\
&< 2^{1-k}. && (5.11)
\end{aligned}$$

We applied the union bound in (5.9), equation (5.8) in (5.10), and Lemma 5.7 in (5.11).

Finally, multiplying matrices from $\mathbb{Z}/(s)^{n\times(r+k)}$ with a unimodular matrix is an isomorphism back to $(\mathbb{Z}/(s))^{n\times(r+k)}$, which implies that $\Pr[\mathsf{FR}_{I_n}] = \Pr[\mathsf{FR}_{U^{-1}}]$. So, according to Lemma 5.5, the probability described in Theorem 5.4 must be at least

$$\Pr[\mathsf{FR}_{U^{-1}}] = \Pr[\mathsf{FR}_{I_n}] > 1 - \frac{1}{2^{k-1}}.$$

□

5.2 Projection basis

Throughout this section let $A \in \mathbb{Z}^{n\times n}$ be nonsingular. In Section 5.1, we showed that the projection $A^{-1}J$, for a suitable random integer matrix J, can reveal the r largest invariant factors of A with high probability. In this section, we define suitable notation which will help us show how these invariant factors can be extracted from A to produce a matrix B that has the same Smith form as A but with the r largest invariant factors replaced by trivial ones.

For any $J \in \mathbb{Z}^{n\times r}$ with $r \in \mathbb{Z}_{>0}$, the set

$$\mathrm{Proj}(A, J) = \{v \in \mathbb{Z}^{1\times n} \mid vA^{-1}J \in \mathbb{Z}^{1\times r}\}$$

forms an integer lattice (a sub-lattice of $\mathbb{Z}^n$). A basis of $\mathrm{Proj}(A, J)$ can be given by a matrix $H \in \mathbb{Z}^{n\times n}$ such that the set of all integer linear combinations of rows of H is equal to $\mathrm{Proj}(A, J)$. Bases of $\mathrm{Proj}(A, J)$ are always nonsingular and are unique up to left equivalence.

Example 5.8. *A basis of* $\mathrm{Proj}(A, 0_{n\times r})$ *is* I_n*, while a basis of* $\mathrm{Proj}(A, I_n)$ *is given by* A *itself.*

The next two lemmas follow directly from the definition of $\mathrm{Proj}(A, J)$. Lemma 5.9 is useful in a way that it allows the projection set $\mathrm{Proj}(A, J)$ to be represented only by the fractional part of $A^{-1}J$, while Lemma 5.10 will help us reduce the computation of a basis of $\mathrm{Proj}(A, J)$ to computing a basis of another projection set that has A conditioned by unimodular column operations. The conditioning is supposed to reveal the basis in an obvious way.

Lemma 5.9. *If* $s \in \mathbb{Z}_{>0}$ *is such that* $sA^{-1}J$ *is integral, and* $P = \mathrm{Rem}(sA^{-1}J, s)$*, then*

$$\mathrm{Proj}(A, J) = \mathrm{Proj}(sI, P) = \{v \in \mathbb{Z}^{1\times n} \mid vP = 0 \bmod s\}.$$

Lemma 5.10. *Let* $W \in \mathbb{Z}^{n\times n}$ *be unimodular. Then* H *is a basis of* $\mathrm{Proj}(AW^{-1}, J)$ *if and only if* HW *is a basis of* $\mathrm{Proj}(A, J)$*.*

Lemma 5.11 shows that if H is a basis of $\mathrm{Proj}(A, J)$, then, for any J, it can be factored out from matrix A, that is, AH^{-1} is integral. This is useful because we plan to recover the Smith form of a matrix A in parts, and we want to extract the part of the Smith profile already computed in order to continue with computing the next part.

Lemma 5.11. *If* H *is a basis of* $\mathrm{Proj}(A, J)$*, then* AH^{-1} *is integral.*

Proof. Since the rows of A belong to $\mathrm{Proj}(A, J)$, there exists a $B \in \mathbb{Z}^{n\times n}$ such that $A = BH$, hence $AH^{-1} = B$ is integral. □

5.3 Maximal index Smith massager

In this section, we combine several results from previous sections to present a randomized algorithm for the Problem `IndexMassager` shown in Figure 5.1. We begin with the following definition.

Definition 5.12 (Index-(m, r) Smith massager)**.** *Let* $B \in \mathbb{Z}^{2n\times 2n}$ *be nonsingular with the shape*

$$B = \begin{bmatrix} A & & * \\ & I_{n-m} & \\ * & & * \end{bmatrix}.$$

For $m, r \in \mathbb{Z}_{\geq 0}$ such that $m + r \leq n$, an index-(m, r) Smith massager for B *is a tuple $(U, M, T, S) \in (\mathbb{Z}^{r\times n}, \mathbb{Z}^{n\times r}, \mathbb{Z}^{r\times r}, \mathbb{Z}^{r\times r})$ such that the matrix*

$$C = B \begin{bmatrix} I_n & & & \\ & I & & \\ U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & M & \\ & I & & \\ & & T & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S^{-1} & \\ & & & I_m \end{bmatrix} \tag{5.12}$$

is integral, with S nonsingular and in Smith form, and T unit upper triangular. We say that (U, M, T, S) is maximal *for B if S is comprised of the r largest invariant factors of the Smith form of B.*

Notice that when $m = 0$ the matrix B is equal to $\mathrm{diag}(A, I_n)$. When, in addition, $r = n$, an index-(m, r) Smith massager for $\mathrm{diag}(A, I_n)$ corresponds to a full Smith massager for A as defined in the introduction.

`IndexMassager`$(B, n, m, r, s, \epsilon)$

Input: B, n, m and r are as in Definition 5.12. In addition, $s \in \mathbb{Z}_{>0}$ and ϵ is such that $0 < \epsilon < 1$.

Output: An index-(m, r) Smith massager (U, M, T, S) for B with $T = I_r$, entries in U and M reduced modulo s, and S_{rr} a divisor of s.

Note: If s is a positive integer multiple of the largest invariant factor of B, and the last n rows and columns of B^{-1} are integral, then a maximal index-(m, r) Smith massager for B is returned with probability at least $1 - \epsilon$.

Figure 5.1: Problem `IndexMassager`

In the design of the algorithm we are assuming that s is a multiple of the largest invariant factor of B and that the last n rows and columns of B^{-1} are integral. If, during the course of the algorithm, we detect that either of these conditions is not satisfied then we simply return the trivial index-(m, r) Smith massager $(0_{r\times n}, 0_{n\times r}, I_r, I_r)$ in order to satisfy the output requirements of the problem.

As shown in Section 5.2, we can "massage" away a block of the largest invariant factors of B by computing a basis of $\mathrm{Proj}(B, J)$ for a well chosen

$$J := \left[\frac{J_1}{J_2}\right] \in \mathbb{Z}^{(n+n)\times r}.$$

Note that under the assumption that the last n columns of B^{-1} are integral, the basis $\mathrm{Proj}(B, J)$ will remain invariant of the choice of entries in the block $J_2 \in \mathbb{Z}^{n\times r}$. For this reason, we set J_2 to be the zero matrix. Entries in J_1 are chosen independently and uniformly at random from $\mathbb{Z}/(s)$.

Next, we use the algorithm supporting Theorem 2.7 to check whether $sB^{-1}J$ is integral, and, if so, compute the projection

$$P := \mathrm{Rem}(sB^{-1}J, s) = \left[\frac{P_1}{P_2}\right] \in \mathbb{Z}/(s)^{(n+n)\times r}.$$

Under the assumption that the last n rows of B^{-1} are integral, we expect P_2 to be the $n \times r$ zero matrix. If $sB^{-1}J$ is determined not to be integral, or P_2 is not the zero matrix, then we abort and return the trivial index-(m, r) massager for B.

At this point, by Lemma 5.9, we have reduced the problem of computing a basis of $\mathrm{Proj}(B, J)$ to that of computing a basis of $\mathrm{Proj}(sI, P)$. A basis of $\mathrm{Proj}(sI, P)$ can be computed as follows. First, using the Smith form algorithm of Storjohann (2000b, Section 7), compute matrices $U \in \mathbb{Z}^{r\times n}$ and $V \in \mathbb{Z}^{r\times r}$, such that $\det V \perp s$ and $D := \mathrm{Rem}(-UP_1V, s)$ is congruent to the reverse Smith form of $P_1 \in \mathbb{Z}^{n\times r}$ over $\mathbb{Z}/(s)$. Then, we have that

$$-UP_1V = D \bmod s$$

and

$$P_1V = MD \bmod s,$$

for some integer matrix $M \in \mathbb{Z}^{n\times r}$. We can put those two together and obtain

$$\begin{bmatrix} I_n & & & \\ & I & & \\ -U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} P_1 \\ \\ \\ \end{bmatrix} V = \begin{bmatrix} MD \\ \\ D \\ \\ \end{bmatrix} \bmod s.$$

Next, we apply a unimodular left transformation to zero out the block MD, and we take right equivalence to omit V.

$$\begin{bmatrix} I_n & & -M & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ -U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} P_1 \\ \\ \\ \end{bmatrix} \equiv_R \begin{bmatrix} \\ D \\ \\ \end{bmatrix} \bmod s \tag{5.13}$$

Finally, define $S := sD^{-1}$, which will be in regular Smith form, and notice that S is a basis of $\mathrm{Proj}(S, I_r) = \mathrm{Proj}(sI, D)$, which corresponds to the non-zero part of the matrix in the right-hand side of (5.13). Therefore,

$$\begin{bmatrix} I_n & & & \\ & I & & \\ & & S & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & -M & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ -U & & I_r & \\ & & & I_m \end{bmatrix} \tag{5.14}$$

must be a basis of $\mathrm{Proj}(sI, P)$ according to Lemma 5.10, since the two matrices containing M and U are unimodular. Postmultiplying B by the inverse of this basis results in an integer matrix according to Lemma 5.11. That is,

$$C := B \begin{bmatrix} I_n & & & \\ & I & & \\ U & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & M & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S^{-1} & \\ & & & I_m \end{bmatrix}.$$

Therefore, matrices (U, M, I_r, S) form an index-(m, r) Smith massager in accordance with Definition 5.12.

Theorem 5.13. *Assume that we have a lifting modulus X as in Lemma 2.1 for the row partial linearization of B. If $r \times \log s \in O(n(d + \log n))$, where d is the average* length *of the rows of B, and $\epsilon = \frac{1}{8r}$, then Problem* `IndexMassager` *can be solved in time*

$$O(n^\omega \mathsf{B}(d + \log n) \log n).$$

Proof. The correctness of the algorithm follows directly from the preceding discussion. It is apparent that the proposed massager fits the description of Definition 5.12.

We achieve the probabilistic result, for $\epsilon = \frac{1}{8r}$, by exploiting Theorem 5.4. Instead of working with the projection $sB^{-1}J \in \mathbb{Z}^{2n \times r}$, we augment J with $k := \log_2 r + 4$ columns. After the Smith form computation, we keep only the last r rows of U, the last r columns of M, and the r largest invariant factors of S. This massager will be maximal with probability at least $1 - \frac{1}{2^{k-1}} = 1 - \frac{1}{8r}$.

Finally, regarding the running time, the algorithm consists of only two computational parts. The first is to verify that $sB^{-1}J$ is integral, and, if so, compute $P = \mathrm{Rem}(sB^{-1}J, s)$. By Theorem 2.7 and Remark 3.12, this can be done in time $O(n^\omega \mathsf{M}(d + \log n) \log n)$. The second is the reverse Smith form computation, which, by Storjohann (2000b, Corollary 7.17), can be done in time $O(n^\omega \mathsf{B}(d + \log n) \log n)$, after simplifying the cost estimate using our assumptions on B. □

5.3.1 Reduced index Smith massager

In this subsection, we introduce the notion of the reduced Smith massager, which keeps the overall size of the matrices well bounded.

Remember that denote by $M \bmod S$ the matrix obtained from M by reducing entries in column j modulo S_{jj}, $1 \le j \le r$. Similarly, we denote by $U \bmod S$ the matrix obtained from U by reducing entries in row i modulo S_{ii}, $1 \le i \le r$.

Definition 5.14 (Reduced index-(m,r) Smith massager)**.** *Let (U,M,T,S) be an index-(m,r) Smith massager for $B \in \mathbb{Z}^{2n\times 2n}$ as in Definition 5.12. We say that (U,M,T,S) is* reduced *if $U = U \bmod S$, $M = M \bmod S$ and $T = ((T - I_r) \bmod S) + I_r$.*

Note that if T is unit (upper) triangular, then $((T - I_r) \bmod S) + I_r$ is as well since only the off-diagonal entries are reduced $\bmod S$.

Lemma 5.15. *Suppose (U,M,T,S) is an index-(m,r) Smith massager for $B \in \mathbb{Z}^{2n\times 2n}$ as in Definition 5.12. Let $U' = U \bmod S$, $M' = M \bmod S$ and $T' = ((-U'M' - I_r) \bmod S) + I_r$. Then, (U,M',T,S) and (U,M,T',S) are index-(m,r) massagers for B, and (U',M',T',S) is a reduced index-(m,r) Smith massager for B.*

Proof. Without loss of generality, and in order to simplify the presentation, we consider the case of an index-$(0,r)$ Smith massager. By multiplying together the first three matrices in (5.12) we obtain

$$B = \begin{bmatrix} A & & AM \\ & I_{n-r} & \\ U & & T + UM \end{bmatrix} \begin{bmatrix} I_n & & \\ & I_{n-r} & \\ & & S^{-1} \end{bmatrix}.$$

Note that the property that AMS^{-1} is integral is equivalent to $AM \bmod S$ being the zero matrix. But then $A(M \bmod S) \bmod S$ is also the zero matrix. This shows that (U,M',T,S) is still an index massager. A similar argument shows that (U,M,T',S) is an index massager. By the definition of T',

$$(T' + (U \bmod S)(M \bmod S)) \bmod S$$

is also the zero matrix. Since T is unit upper triangular and also $(T + UM) \bmod S$ is the zero matrix, we have that $-UM \bmod S$ is unit upper triangular, except that the i'th diagonal entry will be zero for $S_{ii} = 1$. Using the fact that $U = U' + SQ$ for some $Q \in \mathbb{Z}^{n\times n}$ and the property that $S_{11} \mid S_{22} \mid \cdots \mid S_{rr}$, it follows that T' is also unit upper triangular. □

5.4 Maximal Smith massager

In this section, we develop a randomized algorithm for computing the Smith form along with a Smith massager for a nonsingular $A \in \mathbb{Z}^{n\times n}$. Section 5.4.1 gives a subroutine for combining an index-$(0,m)$ and index-(m,r) Smith massager to obtain an index-$(0,m+r)$ Smith massager. The algorithm is given in Section 5.4.2 with a proof of correctness and running time given in Sections 5.4.3 and 5.4.4, respectively.

5.4.1 Combining index massagers

In this section, we show how an index-$(0,n)$ Smith massager for $\mathrm{diag}(A, I_n)$ can be computed in a block iterative fashion.

Suppose we have an index-$(0,m)$ Smith massager (U, M, T, S) for $\mathrm{diag}(A, I_n)$. Then, for some $r \in \mathbb{Z}_{\geq 0}$ such that $m + r \leq n$, we deduce from (5.12) that

$$B = \begin{bmatrix} A & & & \\ & I & & \\ & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & I_r & \\ U & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & M \\ & I & & \\ & & I_r & \\ & & & T \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & I_r & \\ & & & S^{-1} \end{bmatrix} \tag{5.15}$$

is integral. Let (U', M', T', S') be an index-(m,r) Smith massager for B. Then,

$$C = B \begin{bmatrix} I_n & & & \\ & I & & \\ U' & & I_r & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & M' & \\ & I & & \\ & & T' & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S'^{-1} & \\ & & & I_m \end{bmatrix} \tag{5.16}$$

is integral. A direct computation shows that the product of the first trio of matrices post-multiplying $\mathrm{diag}(A, I_n)$ in (5.15) with the second trio of matrices post-multiplying B in (5.16) is equal to

$$\begin{bmatrix} I_n & & & \\ & I & & \\ U' & & I_r & \\ U & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & M' & M \\ & I & & \\ & & T' & -U'M \\ & & & T \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ & & S'^{-1} & \\ & & & S^{-1} \end{bmatrix}. \tag{5.17}$$

Thus, the result of post-multiplying $\mathrm{diag}(A, I_n)$ by the combined trio in (5.17) is integral. The next result follows as a result of the above discussion and as a corollary of Lemma 5.15.

Theorem 5.16. *Let (U, M, T, S) be a reduced index-$(0, m)$ Smith massager for* $\mathrm{diag}(A, I_n)$*, and let (U', M', T', S') be a reduced index-(m, r) Smith massager for the matrix B in (5.15). If S'_{rr} is a divisor of S_{11}, then a reduced index-$(0, m + r)$ Smith massager for* $\mathrm{diag}(A, I_n)$ *is given by (U'', M'', T'', S'') where*

$$U'' = \left[\begin{array}{c} U' \\ \hline U \end{array}\right], \quad M'' = \left[\begin{array}{c|c} M' & M \end{array}\right], \quad S'' = \left[\begin{array}{c|c} S' & \\ \hline & S \end{array}\right],$$

and

$$T'' = \left[\begin{array}{c|c} T' & -U'M \bmod S \\ \hline & T \end{array}\right].$$

5.4.2 Algorithm

Algorithm `SmithMassager`(A) is shown in Figure 5.2.

Phase 2 of the algorithm initializes the working matrix $B := \mathrm{diag}(A, I_n)$ and the index massager $(U, M, T, S) \in (\mathbb{Z}^{0\times n}, \mathbb{Z}^{n\times 0}, \mathbb{Z}^{0\times 0}, \mathbb{Z}^{0\times 0})$ to be the trivial index-$(0, 0)$ Smith massager. Then, the loop uses $\lceil \log_2(n+1) \rceil$ applications of Theorem 5.16 to update (U, M, T, S) to be an index-$(0, n)$ Smith massager for $\mathrm{diag}(A, I_n)$, which is a full Smith massager for A. The technique of Lemma 5.15 is used to keep the intermediate index massagers reduced. At the beginning of iteration i of the for-loop, (U, M, T, S) is a reduced index-$(0, m)$ Smith massager where $m = 2^{i-1} - 1$. Iteration i then updates (U, M, T, S) to be a reduced index-$(0, m + r)$ Smith massager where $r = 2^{i-1}$. In the last iteration, it can be $n - m < 2^{i-1}$, and so, r is defined as the min of these two quantities.

At the end of phase 2, the algorithm has computed a full Smith massager (U, M, T, S) for A. It remains only to assay whether (U, M, T, S) is maximal. This is done by checking that the massaged matrix B is unimodular.

5.4.3 Correctness

We begin with two lemmas regarding properties of the massaged matrix B in phase 2(c) of the algorithm.

Lemma 5.17 states that if B is massaged by a maximal index-$(0, m)$ Smith massager, then it has the Smith form of $\mathrm{diag}(A, I_n)$ but with largest m invariant factors replaced by trivial ones. The lemma follows from the fact that matrix $M \in \mathbb{Z}^{n\times m}$ consists of the last m columns of a Smith massager for A, and so, $\mathrm{diag}(A, I_n)$ is properly conditioned so that the largest m invariant factors are extracted.

`SmithMassager`(A)
Input: Nonsingular $A \in \mathbb{Z}^{n\times n}$.
Output: The Smith form $S \in \mathbb{Z}^{n\times n}$ of A and a reduced Smith massager $M \in \mathbb{Z}^{n\times n}$.
Note: FAIL will be returned with probability less than $1/2$.

1. [Compute a lifting modulus X and the largest invariant factor s_n]

   (a) $X :=$ a lifting modulus for A and all B as in Lemma 2.1
   # a modulus may not be found with probability $\leq 1/8$.

   (b) $s :=$ the largest invariant factor s_n of A
   # s may be a proper divisor of s_n with probability $\leq 1/8$.

2. [Compute an index-$(0,n)$ Smith massager for $\mathrm{diag}(A, I_n)$]
   $(U, M, T, S) \in (\mathbb{Z}^{0\times n}, \mathbb{Z}^{n\times 0}, \mathbb{Z}^{0\times 0}, \mathbb{Z}^{0\times 0})$
   $B := \mathrm{diag}(A, I_n)$
   for $i = 1$ **to** $\lceil \log_2(n+1) \rceil$ **do**
   $m := 2^{i-1} - 1$
   $r := \min(2^{i-1}, n - m)$
   if $i > 1$ **then** $s := S_{11}$

   (a) [Compute an index-(m,r) massager of B and reduce]
   $(U', M', I, S') :=$ `IndexMassager`$(B, m, r, s, 2^{-(i+2)})$
   $U', M' := U' \,\mathbf{r}\mathrm{mod}\, S', M' \,\mathbf{c}\mathrm{mod}\, S'$
   $T' := -U'M' \,\mathbf{c}\mathrm{mod}\, S'$ (with 0 diagonal entries replaced by 1)

   (b) [Augment massager and reduce]
   $$U, M, T, S := \left[\frac{U'}{U}\right], \left[\begin{array}{c|c} M' & M \end{array}\right], \left[\begin{array}{c|c} T' & -U'M \,\mathbf{c}\mathrm{mod}\, S \\ \hline & T \end{array}\right], \left[\begin{array}{c|c} S' & \\ \hline & S \end{array}\right]$$

   (c) [Apply massager]
   $$B := \begin{bmatrix} A & AMS^{-1} \\ & I \\ U & (T + UM)S^{-1} \end{bmatrix}$$

3. [Certify that (U, M, T, S) is maximal]
   if $|\det B| = 1$ **then return** (S, M)
   else return FAIL

Figure 5.2: Algorithm `SmithMassager`

Lemma 5.17. *If* (U, M, T, S) *is a maximal index-*$(0, m)$ *Smith massager for* $\mathrm{diag}(A, I_n)$ *with Smith form* $\mathrm{diag}(I_n, S', S)$*, then the Smith form of the massaged matrix* $B \in \mathbb{Z}^{2n \times 2n}$ *as in (5.15) is* $\mathrm{diag}(I_n, I_m, S')$*.*

Lemma 5.18 establishes that if B is massaged by a maximal index-$(0, m)$ Smith massager, then the last n rows and columns of B^{-1} are integral which is an assumption for computing a new index massager in the following iteration.

Lemma 5.18. *If* (U, M, T, S) *is a maximal index-*$(0, m)$ *Smith massager for* $\mathrm{diag}(A, I_n)$ *and* $B \in \mathbb{Z}^{2n \times 2n}$ *the massaged matrix as in (5.15), then the last* n *rows and columns of* B^{-1} *are integral.*

Proof. Notice that the augmenting operation of Theorem 5.16 can also be reversed to separate a Smith massager. We will use induction on m. The base case, for $m = 0$, holds vacuously. Next, assume that the statement of the lemma holds for a maximal index-$(0, m)$ Smith massager (U, M, T, S). This means that: (1) the largest invariant factor of the massaged matrix B is s_{n-m} according to Lemma 5.17, and: (2) the last n rows of B^{-1} are integral according to the induction hypothesis. Now, let (U', M', T', S') be a maximal index-$(m, 1)$ Smith massager for B. The product of the trio of matrices defined by (U, M, T, S) and the product defined by (U', M', T', S') correspond to a trio of matrices defined by a maximal index-$(0, m+1)$ Smith massager. The inverse of the massaged matrix C will be

$$\begin{bmatrix} I_n & & & \\ & I & & \\ & & s_{n-m} & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & -M' & \\ & I & & \\ & & 1 & \\ & & & I_m \end{bmatrix} \begin{bmatrix} I_n & & & \\ & I & & \\ -U' & & 1 & \\ & & & I_m \end{bmatrix} B^{-1}.$$

We see that the largest invariant factor of the product of the last three matrices is still s_{n-m}. In addition, the row of the product that is multiplied with s_{n-m}, is the only one from the last n rows of B^{-1} to which elements from the non-integral part of B^{-1} are added. Of course, multiplying with the matrix's largest invariant factor ensures that the last n rows of C^{-1} remain integral.

Finally, the last n columns are necessarily integral since they are the product of integral parts. □

Theorem 5.19. *Algorithm* `SmithMassager` *shown in Figure 5.2 is correct. The algorithm returns* `FAIL` *with probability less than* $1/2$*.*

Proof. The correctness of the algorithm follows from Sections 5.3 and 5.4.1 and the certification of the output by the unimodularity check in phase 3.

Regarding the probability of success, we define the following events.

- E_0: In phase 1, s is not the largest invariant factor s_n of A.
- E_i: At iteration $i = 1, \ldots, \lceil \log_2(n+1) \rceil$ of phase 2, massager (U, M, T, S) is not maximal.

In other words, in order to prove the theorem, it is enough to show that $\Pr[E_{\lceil \log_2(n+1) \rceil}] < 3/8$, where we excluded the initial $1/8$ error probability of finding a lifting modulus X such that $(\det B) \mid (\det A) \perp X$.

From the specification of the routine `IndexMassager` in Figure 5.1 and Lemma 5.18, we obtain that

$$\Pr[E_0] \leq \frac{1}{8} \quad \text{and} \quad \Pr[E_i | \neg E_{i-1}] \leq 2^{-(i+2)}.$$

So, if the choice of the lifting modulus X is correct, Algorithm `SmithMassager` returns `FAIL` with probability less than

$$\Pr[E_{\lceil \log_2(n+1) \rceil}] \leq \sum_{i=1}^{\lceil \log_2(n+1) \rceil} 2^{-(i+2)} + \Pr[E_0] < \sum_{i=1}^{\infty} \frac{1}{2^{i+2}} + \frac{1}{8} = \frac{3}{8}.$$

□

5.4.4 Complexity

We begin by bounding the cost of phases 2(b) and 2(c). Lemma 5.20 shows how to compute matrix B in phase 2(c), and Lemma 5.21 how to realize the construction of Theorem 5.16 in phase 2(b).

Lemma 5.20. *There exists a procedure that takes a reduced index-$(0, m)$ Smith massager (U, M, T, S) for $\operatorname{diag}(A, I_n)$, and returns a matrix B as in (5.15). The running time of the procedure is $O(n^\omega \mathsf{M}(\log n + \log ||A||))$.*

Proof. It is enough to prove the claim for $m = n$. We have that

$$B = \begin{bmatrix} A & AMS^{-1} \\ U & (T + UM)S^{-1} \end{bmatrix}.$$

The cost-dominating operation is the product $UM \in \mathbb{Z}^{n\times n}$.

Recall that $(\det S) \mid (\det A) \leq n^{n/2}\|A\|^n$, and that the entries in row i of matrix U and in column i of matrix M are reduced modulo S_{ii}. Thus, the computation of the product UM fits under the conditions of Lemma 2.9 for $d = \log(n^{1/2}\|A\|)$, which proves our claim. □

Lemma 5.21. *The reduced index-$(0, m + r)$ massager of Theorem 5.16 can be computed in time $O(n^\omega \,\mathsf{M}(\log n + \log ||A||))$.*

Proof. The only nontrivial computation of Theorem 5.16 is the product $U'M \in \mathbb{Z}^{r\times m}$, and the required complexity can be achieved by the same arguments as in Lemma 5.20. □

Theorem 5.22. *The running time of the Algorithm* `SmithMassager` *shown in Figure 5.2 is*

$$O(n^\omega \,\mathsf{B}(\log n + \log ||A||)\,(\log n)^2).$$

Proof. Phase 1(a) is done in time

$$O(n^\omega \,\mathsf{M}(\log n + \log ||A||))$$

by Lemma 2.1 and phase 1(b) in time

$$O(n^\omega \,\mathsf{B}(\log n + \log ||A||)\log n)$$

using the Monte Carlo approach of Eberly et al. (2000, Theorem 2.1) combined with the fast linear system solving of Theorem 2.5 and rational number reconstruction (von zur Gathen and Gerhard, 2013, Section 5.10). Phase 2 consists of $O(\log n)$ iterations of the `IndexMassager` algorithm. Since matrix U is always reduced row modulo S, the average of the lengths of the rows of U, and consequently of B, is $O(\log n + \log ||A||)$. Hence, phase 2 requires time

$$O(n^\omega \,\mathsf{B}(\log n + \log ||A||)(\log n)^2).$$

Finally, according to Pauderis and Storjohann (2012, Section 4), the unimodularity check in phase 3 can be performed in time

$$O(n^\omega \,\mathsf{M}(\log n + \log ||A||)\log n).$$

□

Remark 5.23 (Smith massager for a skewed matrix)**.** *Theorem 4.13 and Corollary 4.14 from Section 4.4 show that the Smith massager of any partial linearization of A will include the Smith massager of A. Therefore, Theorem 5.22 can have cost which depends on the average bitlength d of A and not the bitlength of the largest entry. The average bitlength d can assume any of the three definitions given by Theorem 3.7, Corollary 3.9 and Corollary 3.10.*

Chapter 6

Smith multipliers algorithm

In this chapter, given a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$, we present a Las Vegas randomized algorithm for efficiently computing its Smith form $S \in \mathbb{Z}^{n\times n}$ along with unimodular matrices $U, V \in \mathbb{Z}^{n\times n}$ such that

$$AV = US.$$

The algorithm's complexity will match the complexity of the algorithm in Chapter 5 which was intended to just compute the Smith form, namely, we can compute S, U, V with

$$O(n^{\omega}\,\mathsf{B}(\log n + \log \|A\|)(\log n)^2)$$

bit operations.

Our plan is to solve our Smith form with multipliers problem by converting a reduced Smith massager M into a unimodular Smith massager V. To do this we suggest to use the following steps.

1. Pick a random matrix $R \in \mathbb{Z}^{n\times n}$ where each entry is chosen independently and uniformly from $\mathbb{Z}/(\lambda)$ for some $\lambda \in O(n\|A\|)$.
2. Perturb the massager by R scaled with S, and let $B := M + RS$.
3. Compute the lower triangular row Hermite form $H \in \mathbb{Z}^{n\times n}$ of B.
4. The matrix $V := BH^{-1}$ will be a unimodular Smith massager.

All the arguments that we need to prove that our proposed process is correct are in Section 4.3. This chapter is devoted to establishing the efficiency of the method. We begin by illustrating our algorithm using the following example.

Example 6.1. *Let*

$$A := \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 1 & 2 & 4 & 1 \\ 1 & 3 & 2 & 6 & 4 & 5 & 1 \\ 1 & 4 & 2 & 1 & 4 & 2 & 1 \\ 1 & 5 & 4 & 6 & 2 & 3 & 1 \\ 1 & 6 & 1 & 6 & 1 & 6 & 1 \end{bmatrix}.$$

Algorithm `SmithMassager` *from Section 5.4 then returns the Smith form and Smith massager*

$$S := \begin{bmatrix} 1 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 2 & & \\ & & & & & 8 & \\ & & & & & & 80 \end{bmatrix}, \quad M := \begin{bmatrix} & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 35 \\ 1 & 5 & 65 \\ 1 & 4 & 38 \\ 0 & 1 & 5 \\ 1 & 2 & 55 \\ 1 & 2 & 42 \end{bmatrix}.$$

We shall always take the matrix M column modulo S.

Pick a random matrix

$$R := \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix},$$

where each entry is chosen independently and uniformly from a set of $\lambda \in O(n\|A\|)$ consecutive integers. (For the example, we let $\lambda := 2$.)

By perturbing M by the random choice of R post-multiplied with S, we obtain

$$B := M + RS = \begin{bmatrix} 0 & 0 & 1 & 1 & 2 & 8 & 80 \\ 1 & 1 & 0 & 0 & 3 & 0 & 105 \\ 0 & 0 & 0 & 0 & 1 & 5 & 45 \\ 1 & 1 & 1 & 1 & 1 & 0 & 58 \\ 0 & 1 & 0 & 1 & 0 & 9 & 25 \\ 0 & 1 & 1 & 1 & 2 & 8 & 45 \\ 1 & 1 & 0 & 0 & 3 & 10 & 42 \end{bmatrix}.$$

Computing the lower triangular row Hermite form of the random matrix B, gives

$$H := \begin{bmatrix} 1286 & & & & & & \\ 1106 & 1 & & & & & \\ 441 & & 1 & & & & \\ 839 & & & 1 & & & \\ 669 & & & & 1 & & \\ 1125 & & & & & 1 & \\ 546 & & & & & & 1 \end{bmatrix}.$$

Our aim is for H to have only the first diagonal entry non-trivial. If B is not left equivalent to such a matrix H, then the algorithm fails.

To obtain a unimodular Smith massager, we then simply extract H from B by post-multiplying with H^{-1}.

$$V := BH^{-1} = \begin{bmatrix} -43 & 0 & 1 & 1 & 2 & 8 & 80 \\ -47 & 1 & 0 & 0 & 3 & 0 & 105 \\ -24 & 0 & 0 & 0 & 1 & 5 & 45 \\ -27 & 1 & 1 & 1 & 1 & 0 & 58 \\ -20 & 1 & 0 & 1 & 0 & 9 & 25 \\ -29 & 1 & 1 & 1 & 2 & 8 & 45 \\ -29 & 1 & 0 & 0 & 3 & 10 & 42 \end{bmatrix}.$$

By construction, the matrix V is integral and unimodular.

The fact that H has only one non-trivial column allows us to easily establish a nice bound on the size of matrix V. Notice that the columns of V have the same bitlength as the columns of B except for only the first column. In addition, the bitlength of the columns of B equals the bitlength of the columns of the Smith massager M plus the bitlength of λ.

Finally, all the operations applied to M are included in Lemma 4.3, and so V is still a valid Smith massager for A. This makes the matrix

$$U := AVS^{-1} = \begin{bmatrix} -43 & 0 & 1 & 1 & 1 & 1 & 1 \\ -219 & 5 & 3 & 4 & 6 & 5 & 5 \\ -445 & 10 & 6 & 8 & 12 & 11 & 10 \\ -648 & 19 & 12 & 16 & 16 & 13 & 15 \\ -473 & 12 & 4 & 8 & 12 & 10 & 11 \\ -692 & 17 & 10 & 12 & 18 & 10 & 17 \\ -734 & 20 & 13 & 14 & 21 & 10 & 18 \end{bmatrix}$$

also integral and unimodular. By construction, the two unimodular matrices $V, U \in \mathbb{Z}^{n\times n}$ satisfy $AV = US$.

Section 6.1 establishes that the Hermite form of a randomly perturbed massager is trivial with high probability. Section 6.2 shows that we can efficiently certify if the Hermite form of a matrix is trivial and, if so, return it. Section 6.3 gives the exact algorithm to compute the Smith multiplier matrices. Finally, in Section 6.4 we show how we can obtain an outer product adjoint formula given the Smith multiplier matrices. An outer product adjoint formula is useful, for example, in computing fast the proper fractional part of a linear system solution.

6.1 Random perturbations of Smith massagers

Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form S. In this section, we show how to perturb a Smith massager M for A into a unimodular Smith massager V. The first step will be to obtain a Smith massager $B := M + RS$ that is left equivalent (over $\mathbb{Z}$) to a lower triangular row Hermite form with the shape

$$\begin{bmatrix} |\det B| & & & & \\ * & 1 & & & \\ * & & 1 & & \\ \vdots & & & \ddots & \\ * & & & & 1 \end{bmatrix} \in \mathbb{Z}^{n\times n}. \tag{6.1}$$

The property that the last $n-1$ diagonal entries of B are equal to 1 coincides with the property that the last $n-1$ columns of $B \bmod p$ are linearly independent over $\mathbb{Z}/(p)$ for all primes p.

Our approach is inspired by and follows that of Eberly et al. (2000, Section 6), where the following general result is established: for $\lambda \geq 2$, a matrix $R \in \mathbb{Z}^{n\times n}$ with entries chosen uniformly and randomly from $[0, \lambda - 1]$ will have an expected number of $O(\log_\lambda n)$ nontrivial invariant factors. Since our purpose here is not to establish a result for arbitrary λ, we can improve the probability of success by first starting with a Smith massager for $2A$, and then by choosing λ to be a multiple of $105 = 3 \times 5 \times 7$.

Theorem 6.2. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form S. Let M be a reduced Smith massager for $2A$. For any $R \in \mathbb{Z}^{n\times n}$,*

(i) the matrix $B = M + 2RS$ is a Smith massager for A, and

(ii) if entries in R are chosen chosen uniformly and randomly from $[0, \lambda - 1]$, where

$$\lambda = 105 \max(n, \lceil (\det 2S)^{1/n} \rceil),$$

then the probability that there exists a prime p such that the last $n-1$ columns of $B \bmod p$ are linearly dependent over $\mathbb{Z}/(p)$ is less than $1/2$.

Part (i) of Theorem 6.2 follows directly from Proposition 4.8, so it remains only to prove part (ii). This will be done using a sequence of lemmas. For the rest of this section, we let A, S, M, R, λ and $B = M + 2RS$ be as defined in Theorem 6.2.

We start by defining a set of probabilistic events that will facilitate the proofs in this section. For a prime p and $1 \leq m \leq n-1$, let Dep_m^p denote the event that the last m columns of B are linearly dependent modulo p. To complete the proof of Theorem 6.2 we show that $\Pr[\vee_p \mathsf{Dep}_{n-1}^p] < 0.5$, where $\vee_p$ means ranging over all primes. Following Eberly et al. (2000, Section 6), we will separately consider the small primes $p < \lambda$, and the large primes $p \geq \lambda$. We begin with some lemmas that hold for all primes p.

Lemma 6.3. *For any prime p we have*

$$\Pr[\mathsf{Dep}_1^p] \leq \left(\frac{1}{\lambda} \left\lceil \frac{\lambda}{p} \right\rceil \right)^n, \tag{6.2}$$

and for any $2 \leq m \leq n-1$,

$$\Pr[\mathsf{Dep}_m^p \mid \neg \mathsf{Dep}_{m-1}^p] \leq \left(\frac{1}{\lambda} \left\lceil \frac{\lambda}{p} \right\rceil \right)^{n-m+1}. \tag{6.3}$$

Proof. We have Dep_1^p precisely when the last column of B is zero modulo p. By Lemma 4.10, for any prime p that divides $2s_n$ we have $\Pr[\mathsf{Dep}_1^p] = 0$. For a prime p that does not divide $2s_n$, Dep_1^p is equivalent to the vector

$$\underbrace{(2s_n)^{-1} M_{1..n,n}}_{\text{fixed}} + R_{1..n,n} \bmod p \tag{6.4}$$

being zero modulo p. Each random entry $R_{i,n}$ is equal to $-(2s_n)^{-1} M_{i,n}$ modulo p with probability at most

$$\frac{1}{\lambda} \left\lceil \frac{\lambda}{p} \right\rceil.$$

The bound (6.2) now follows by noting that vector in (6.4) has n entries.

Now consider the case $2 \leq m \leq n-1$. By Lemma 4.10, we have that $\Pr[\mathsf{Dep}_m^p] = 0$ for any prime p that divides $2s_{n-m+1}$. Assume henceforth that p does not divide $2s_{n-m+1}$. Given $\neg\mathsf{Dep}_{m-1}^p$, there is an $(m-1) \times (m-1)$ submatrix D in the last $m-1$ columns of B that is nonsingular modulo p. Assume, without loss of generality, up to a row permutation of B, that D is the trailing $(m-1) \times (m-1)$ submatrix of B. Decompose the last m columns of B as follows:

$$\left[\begin{array}{c|c} v & C \\ \hline w & D \end{array}\right] \in \mathbb{Z}^{n \times m}.$$

Then C and D are fixed at this point and vectors v and w still depend on the random choice of column $n-m+1$ of R. Fix the choice of w also. Note that

$$\left[\begin{array}{c|c} I_{n-m+1} & -CD^{-1} \\ \hline & D^{-1} \end{array}\right] \left[\begin{array}{c|c} v & C \\ \hline w & D \end{array}\right] = \left[\begin{array}{c|c} a & \\ \hline * & I_{m-1} \end{array}\right] \bmod p.$$

Then Dep_m^p is equivalent to the vector

$$(2s_{n-m+1})^{-1}a = \underbrace{(2s_{n-m+1})^{-1}M_{1..n-m+1,n-m+1} - CD^{-1}w}_{\text{fixed}} + R_{1..n-m+1,n-m+1} \bmod p$$

being zero modulo p. By a similar argument as before, the probability of this happening is bounded by (6.3). □

The next lemma follows simply from the union bound on the set of events for $1 \leq i \leq n-1$ that happen when the ith column from the end is the first that is linearly dependent.

Lemma 6.4. *For any prime p we have*

$$\Pr[\mathsf{Dep}_{n-1}^p] \leq \Pr[\mathsf{Dep}_1^p] + \sum_{i=2}^{n-1} \Pr[\mathsf{Dep}_i^p \mid \neg\mathsf{Dep}_{i-1}^p].$$

6.1.1 Small primes

We first deal with the specific small primes $\{3, 5, 7\}$. Notice that from Proposition 4.8, we know that $\Pr[\mathsf{Dep}_{n-1}^2] = 0$.

Lemma 6.5. $\Pr[\vee_{p \in \{3,5,7\}} \mathsf{Dep}_{n-1}^p] < 0.23$.

Proof. We exploit the fact that λ is a multiple of $105 = 3 \times 5 \times 7$. Let $p \in \{3, 5, 7\}$. Since $p \mid \lambda$, the bound of Lemma 6.3 simplifies to

$$\Pr[\mathsf{Dep}^p_m \mid \neg\mathsf{Dep}^p_{m-1}] \leq \left(\frac{1}{p}\right)^{n-m+1},$$

and Lemma 6.4 gives

$$\Pr[\mathsf{Dep}^p_{n-1}] \leq \sum_{i=1}^{n-1} \left(\frac{1}{p}\right)^{i+1} < \frac{1}{p} \sum_{i=1}^{\infty} \left(\frac{1}{p}\right)^{i} = \frac{1}{p(p-1)}. \tag{6.5}$$

Since the events Dep^3_{n-1}, Dep^5_{n-1} and Dep^7_{n-1} are independent,

$$\Pr[\vee_{p\in\{3,5,7\}}\mathsf{Dep}^p_{n-1}] = 1 - \prod_{p\in\{3,5,7\}} (1 - \Pr[\mathsf{Dep}^p_{n-1}]). \tag{6.6}$$

The result now follows by bounding from above the probabilities on the right hand size of (6.6) using (6.5). □

Next we handle the small primes in the range $7 < p < \lambda$.

Lemma 6.6. $\Pr[\vee_{7<p<\lambda}\mathsf{Dep}^p_{n-1}] < 0.23$

Proof. Let $7 < p < \lambda$. Since $p < \lambda$,

$$\frac{1}{\lambda}\left\lceil\frac{\lambda}{p}\right\rceil < \frac{1}{\lambda}\left(\frac{\lambda}{p} + 1\right) = \frac{1}{p} + \frac{1}{\lambda} < \frac{2}{p} = \frac{1}{p/2},$$

and the bound of Lemma 6.3 simplifies to

$$\Pr[\mathsf{Dep}^p_m \mid \neg\mathsf{Dep}^p_{m-1}] \leq \left(\frac{1}{p/2}\right)^{n-m+1}. \tag{6.7}$$

Lemma 6.4 together with (6.7) gives

$$\Pr[\mathsf{Dep}^p_{n-1}] \leq \frac{1}{(p/2)(p/2-1)} < \frac{1}{((p-1)/2)^2}. \tag{6.8}$$

Using the union bound and then (6.8) gives

$$\begin{aligned}
\Pr[\vee_{7<p<\lambda}\mathsf{Dep}^p_{n-1}] &\leq \sum_{7<p<\lambda} \Pr[\mathsf{Dep}^p_{n-1}] \\
&< \sum_{7<p<\lambda} \frac{1}{((p-1)/2)^2} \\
&< \sum_{x\geq 11,\ \text{odd}} \frac{1}{((x-1)/2)^2} \\
&= \sum_{x\geq 5} \frac{1}{x^2} \\
&= \zeta(2) - \sum_{x=1}^{4} \frac{1}{x^2} \\
&= \frac{\pi^2}{6} - \frac{205}{144} \\
&< 0.23.
\end{aligned}$$

□

6.1.2 Large primes

Consider now the large primes $p \geq \lambda$. Although it follows from Lemmas 6.3 and 6.4 that $\Pr[\mathsf{Dep}^p_{n-1}] \leq 1/(\lambda(\lambda-1))$ for any particular prime $p \geq \lambda$, this doesn't help us to bound $\Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_{n-1}]$ using the union bound since there exist an infinite number of such primes. Instead, we follow the approach of Eberly et al. (2000, Section 6) and show that we only need to consider those primes which divide some necessarily nonzero minors of B.

Lemma 6.7. *Any minor of B is bounded in magnitude by $\lambda^{2.5n}$.*

Proof. It will suffice to bound $|\det B|$ using Hadamard's inequality, which states that $|\det B|$ is bounded by the product of the Euclidean norms of the columns of B. Recall that $B = M + 2RS$ where $M = M \bmod 2S$ and entries in R are chosen from $[0, \lambda - 1]$,

with $\lambda \geq \max((\det 2S)^{1/n}, n)$. Then

$$\begin{aligned} |\det B| &\leq \prod_{j=1}^{n} \|B_{1\ldots n,j}\|_2 \\ &= \prod_{j=1}^{n} \|M_{1\ldots n,j} + 2s_j R_{1\ldots n,j}\|_2 \\ &\leq \prod_{j=1}^{n} n^{1/2}(2s_j - 1 + 2s_j(\lambda - 1)) \\ &< (\det 2S) n^{n/2} \lambda^n \\ &\leq \lambda^{2.5n}. \end{aligned}$$

□

Next we develop the following analogue of Lemma 6.3.

Lemma 6.8. *We have*

$$\Pr[\vee_{p\geq\lambda} \mathsf{Dep}_1^p] \leq 2.53n \left(\frac{1}{\lambda}\right)^{n-1}$$

and for any $2 \leq m \leq n-1$,

$$\Pr[\vee_{p\geq\lambda} \mathsf{Dep}_m^p \mid \neg \vee_{p\geq\lambda} \mathsf{Dep}_{m-1}^p] \leq 2.53n \left(\frac{1}{\lambda}\right)^{n-m}.$$

Proof. By Proposition 4.8, $B = M + R(2S)$ is nonsingular modulo 2, independent of the choice of R. Thus, up to an initial row permutation of M, we may assume that the trailing $j \times j$ submatrix of B mod 2 is nonsingular over $\mathbb{Z}/(2)$ for every $1 \leq j \leq n$.

First consider the case for $m = 1$. Decompose the last column of B as

$$\begin{bmatrix} v \\ w \end{bmatrix} \in \mathbb{Z}^{n\times 1},$$

where $v \in \mathbb{Z}^{(n-1)\times 1}$ and $w \in \mathbb{Z}$. Fix the choice of w, that is, fix the last entry in the last column of R. By assumption, $w \neq 0 \bmod 2$ and thus $w \neq 0$ over $\mathbb{Z}$. For every prime $p \nmid w$ we have $\Pr[\mathsf{Dep}_1^p] = 0$, and since there are $n-1$ entries in v that are still free to be chosen, the union bound gives

$$\begin{aligned} \Pr[\vee_{p\geq\lambda} \mathsf{Dep}_1^p] &= \Pr[\vee_{p\geq\lambda, p\mid w} \mathsf{Dep}_1^p] \\ &\leq (\log_\lambda |w|) \left(\frac{1}{\lambda}\right)^{n-1}. \end{aligned}$$

Lemma 6.7 gives $\log_\lambda |w| \le 2.5n < 2.53n$, establishing the first part of the lemma.

Now consider $2 \le m \le n-1$. Decompose the last m columns of B as follows:

$$\left[\begin{array}{c|c} v & C \\ \hline w & D \end{array}\right] \in \mathbb{Z}^{n\times m},$$

where $D \in \mathbb{Z}^{(m-1)\times(m-1)}$. Then C and D are fixed at this point and vectors v and w still depend on the random choice of column $n-m+1$ of R. Let $d = \det D$, which we know to be nonzero. There are at most $\log_\lambda |d|$ primes $p \ge \lambda$ that divide d. Using Lemma 6.3 with the union bound gives

$$\sum_{p\ge\lambda, p|d} \Pr[\mathsf{Dep}_m^p \mid \neg\mathsf{Dep}_{m-1}^p] \le (\log_\lambda |d|) \left(\frac{1}{\lambda}\right)^{n-m+1}. \tag{6.9}$$

Next we consider the primes $p \nmid d$. Note that

$$\left[\begin{array}{c|c} dI_{n-m+1} & -dCD^{-1} \\ \hline & dD^{-1} \end{array}\right] \left[\begin{array}{c|c} v & C \\ \hline w & D \end{array}\right] = \left[\begin{array}{c|c} \begin{matrix} a_1 \\ \vdots \\ a_{n-m} \\ a_{n-m+1} \end{matrix} & \\ \hline * & dI_{m-1} \end{array}\right] \in \mathbb{Z}^{n\times m},$$

where, by Cramer's rule, a_{n-m+1} is the determinant of the trailing $m \times m$ submatrix of B. Since $p \nmid d$, event Dep_m^p holds if and only if the vector

$$\begin{bmatrix} a_1 \\ \vdots \\ a_{n-m} \\ a_{n-m+1} \end{bmatrix} = d \begin{bmatrix} v_1 \\ \vdots \\ v_{n-m} \\ v_{n-m+1} \end{bmatrix} - dCD^{-1}w. \tag{6.10}$$

is zero modulo p. Fix the choice of w and v_{n-m+1}. Then $a_{n-m+1} \ne 0$ is also fixed, and for every prime $p \nmid a_{n-m+1}$ we have $\Pr[\mathsf{Dep}_m^p \mid \neg\mathsf{Dep}_{m-1}^p] = 0$. Since there can be at most $\log_\lambda |a_{n-m+1}|$ primes $p \ge \lambda$ that divide a_{n-m+1}, and since $v_1, \ldots, v_{n-m}$ are still free to be chosen, we have

$$\sum_{p\ge\lambda, p\nmid d} \Pr[\mathsf{Dep}_m^p \mid \neg\mathsf{Dep}_{m-1}^p] \le (\log_\lambda |a_{n-m+1}|) \left(\frac{1}{\lambda}\right)^{n-m}. \tag{6.11}$$

Combining the bounds (6.9) and (6.11) and using the estimate of Lemma 6.7 for $|d|$ and $|a_{n-m+1}|$ we obtain

$$\begin{aligned}\Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_m \mid \neg \vee_{p\geq\lambda} \mathsf{Dep}^p_{m-1}] &\leq 2.5n\left(\left(\frac{1}{\lambda}\right)^{n-m+1}+\left(\frac{1}{\lambda}\right)^{n-m}\right)\\ &= 2.5n\left(\frac{1}{\lambda}\right)^{n-m}\left(\frac{1}{\lambda}+1\right)\\ &< 2.53n\left(\frac{1}{\lambda}\right)^{n-m}. \qquad (6.12)\end{aligned}$$

Here, (6.12) follows using $\lambda \geq 105$. □

Lemma 6.9. $\Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_{n-1}] < 0.03$.

Proof. Analogous to Lemma 6.4, we have

$$\Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_{n-1}] \leq \Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_1] + \sum_{i=2}^{n-1}\Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_i \mid \neg \vee_{p\geq\lambda} \mathsf{Dep}^p_{i-1}].$$

Using the estimates of Lemma 6.8 now gives

$$\begin{aligned}\Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_{n-1}] &\leq 2.53n\left(\frac{1}{\lambda}\right)^{n-1} + 2.53n\sum_{i=2}^{n-1}\left(\frac{1}{\lambda}\right)^{n-i}\\ &< 2.53n\left(\frac{1}{\lambda-1}\right).\end{aligned}$$

Simplifying the last bound using the assumption $\lambda \geq 105n$ gives the result. □

Proof of Theorem 6.2. The probability defined by Theorem 6.2 is bounded by the sum of probabilities in Lemmas 6.5, 6.6 and 6.9, that is,

$$\begin{aligned}\Pr[\mathsf{Dep}_{n-1}] &\leq \Pr[\vee_{p\in\{3,5,7\}}\mathsf{Dep}^p_{n-1}] + \Pr[\vee_{7<p<\lambda}\mathsf{Dep}^p_{n-1}] + \Pr[\vee_{p\geq\lambda}\mathsf{Dep}^p_{n-1}]\\ &< 0.23 + 0.23 + 0.03\\ &< 0.5.\end{aligned}$$

□

6.2 Almost trivial Hermite form certification

In this section, we will see how we can verify whether the last $n-1$ columns of the matrix $B \in \mathbb{Z}^{n\times n}$ from Theorem 6.2 are linearly independent for any prime $p \in \mathbb{Z}$. As we already mentioned, this means that B is left equivalent to a lower triangular row Hermite form with the shape

$$H = \begin{bmatrix} |\det B| & & & \\ * & 1 & & \\ \vdots & & \ddots & \\ * & & & 1 \end{bmatrix} \in \mathbb{Z}^{n\times n}. \tag{6.13}$$

Our main tool will once more be the Smith form and a Smith massager for B.

The following theorem is a corollary of Theorem 4.7.

Theorem 6.10. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular with Smith form S and a Smith massager M. If $H \in \mathbb{Z}^{n\times n}$ is a matrix in Hermite form which satisfies that $\det H = \det S$ and*

$$HM \equiv 0 \bmod S,$$

then H is the row Hermite form of A.

Proof. The statement follows from Theorem 4.7 and the uniqueness of the Hermite form of A. □

We plan to use the description of Theorem 6.10 here in order to check whether the lower triangular row Hermite form H of the matrix B has $n-1$ trailing trivial columns, and if yes, then also compute the first non-trivial one. For this section, matrices S and M refer to the Smith form and Smith massager of matrix B.

First of all, we need to ensure that the Smith form $S = \operatorname{diag}(s_1, \ldots, s_n)$ of B also has only one non-trivial invariant factor. If otherwise, then H does not have the desired structure. Let $h_1, h_2, \ldots, h_n$ be the diagonal entries of H. The product $h_2 \cdots h_n$ equals the gcd of all the $(n-1)\times(n-1)$ minors in the last $n-1$ columns of B. On the other hand, the product $s_1 \cdots s_{n-1}$ equals the gcd of all the $(n-1)\times(n-1)$ minors of B, which means that $(s_1 \cdots s_{n-1}) \mid (h_2 \cdots h_n)$. So, if $s_1 \cdots s_{n-1} \neq 1$, then $h_2 \cdots h_n \neq 1$.

Now, assuming that $S = \operatorname{diag}(1, \ldots, 1, s_n)$, we are looking to see whether there exists a vector $\bar{h} \in \mathbb{Z}^{(n-1)\times 1}$ such that

$$\begin{bmatrix} s_n & \\ \bar{h} & I_{n-1} \end{bmatrix} M_{1..n,n} \equiv 0 \bmod s_n,$$

which is equivalent to

$$M_{1,n}\bar{h} + M_{2..n,n} \equiv 0 \bmod s_n. \tag{6.14}$$

Since the Hermite form H must be unique, equation (6.14) must have exactly one solution, which is true if and only if $\gcd(M_{1,n}, s_n) = 1$.

The algorithm follows in Figure 6.1.

`TrivialLowerHermiteForm`(B)
Input: A nonsingular matrix $B \in \mathbb{Z}^{n\times n}$.
Output: The lower triangular Hermite form $H \in \mathbb{Z}^{n\times n}$ of B if only the first column is non-trivial, otherwise NOT TRIVIAL.
Note: FAIL might be returned with probability less than 1/8.

1. [Compute a Smith massager for B.]
   (**if** `SmithMassager` fails **then return** FAIL)
   $S, M :=$ `SmithMassager`(B)

2. [Certify that B is left equivalent to a matrix H as in (6.13).]
   if $S_{n-1,n-1} \neq 1$ **then return** NOT TRIVIAL
   if $\gcd(S_{n,n}, M_{1,n}) \neq 1$ **then return** NOT TRIVIAL

3. [Compute matrix H and return.]
   $H := \begin{bmatrix} h_1 & \\ \bar{h} & I_{n-1} \end{bmatrix}$
   where $h_1 := S_{n,n}$ and $\bar{h} := \mathrm{Rem}(-M_{1,n}^{-1} M_{2..n,n}, S_{n,n})$.
   return H

Figure 6.1: Subroutine `TrivialLowerHermiteForm`

Theorem 6.11. *Subroutine* `TrivialLowerHermiteForm` *is correct and runs in*

$$O(n^\omega \mathsf{B}(d + \log n)(\log n)^2),$$

where d is the average bitlength of the columns of $B \in \mathbb{Z}^{n\times n}$.

Proof. The correctness follows from the preceding discussion.

The probability of the algorithm failing follows from Theorem 5.19 and running the first step (at most) three times.

Regarding the time complexity, the computation of the Smith form $S \in \mathbb{Z}^{n\times n}$ of B along with a Smith massager $M \in \mathbb{Z}^{n\times n}$ dominates the rest of the operations. Let D_B be the partially linearized version of matrix B as specified by Theorem 3.7. Then, by Theorem 4.13, we can obtain S, M from the Smith form and a Smith massager for D_B without any extra computation. Therefore, the complexity of step 1 is bounded by the complexity of computing a Smith massager for D_B, which is $O(n^\omega \mathsf{B}(d+\log n)\,(\log n)^2)$ by Theorem 5.22. □

6.3 A Las Vegas algorithm for the Smith form with multipliers

In this section, we combine the results established in Chapters 4, 5, and 6 to develop the following algorithm. We show that there exists a Las Vegas randomized algorithm that computes the Smith form $S \in \mathbb{Z}^{n\times n}$ of a nonsingular $A \in \mathbb{Z}^{n\times n}$ along with two unimodular matrices $V, U \in \mathbb{Z}^{n\times n}$ such that

$$AV = US,$$

using

$$O(n^\omega \mathsf{B}(\log n + \log \|A\|)\,(\log n)^2)$$

bit operations. The algorithm will return the correct output with probability at least $1/4$ or FAIL otherwise.

The algorithm follows in Figure 6.2.

Theorem 6.12. *Algorithm* `SmithFormMultipliers` *is correct and runs in*

$$O(n^\omega \mathsf{B}(\log n + \log ||A||)\,(\log n)^2).$$

Proof. Step 1 of the algorithm computes the Smith form and a Smith massager for matrix $2A$. From the Smith form of matrix $2A$ we can trivially obtain the Smith form S of A. Furthermore, a Smith massager M for $2A$ is also a Smith massager for A by Lemma 4.9. Step 1 runs in $O(n^\omega \mathsf{B}(\log n + \log \|A\|)\,(\log n)^2)$ by Theorem 5.22, and it will return FAIL with probability at most $1/8$ with three applications of Theorem 5.19.

In step 2, we are perturbing the Smith massager M by a random matrix $R \in \mathbb{Z}^{n\times n}$ multiplied with the Smith form $2S$. By Proposition 4.8, matrix $B = M + R(2S)$ is also a Smith massager for A, and it is nonsingular. Moreover, by Theorem 6.2, the last $n-1$ columns of B are linearly independent over $\mathbb{Z}/(p)$ for every prime p with probability

`SmithFormMultipliers`(A)
Input: A nonsingular matrix $A \in \mathbb{Z}^{n\times n}$.
Output: The Smith form $S \in \mathbb{Z}^{n\times n}$ of A and two unimodular matrices $U, V \in \mathbb{Z}^{n\times n}$ such that $AV = US$.
Note: FAIL will be returned with probability less than 3/4.

1. [Compute the Smith form and a Smith massager for $2A$.]
   (If `SmithMassager` fails, return FAIL)
   $(2S, M) :=$ `SmithMassager`$(2A)$

2. [Perturb the Smith massager M by a random matrix.]
   Pick a uniformly **random** matrix $R \in \mathbb{Z}/(\lambda)^{n\times n}$ for
   $\lambda := 105\max(n, \lceil(\det 2S)^{1/n}\rceil)$ as in Theorem 6.2.
   $B := M + R(2S)$

3. [Certify that B is left equivalent to a matrix H as in (6.13) and return it.]
   (If `TrivialLowerHermiteForm` fails, return FAIL)
   $H :=$ `TrivialLowerHermiteForm`(B)
   if H is NOT TRIVIAL **then return** FAIL

4. [Compute a unimodular Smith massager.]
   $V := BH^{-1}$

5. [Compute matrix U and return.]
   $U := AVS^{-1}$
   return (S, V, U)

Figure 6.2: Algorithm `SmithFormMultipliers`

greater than 1/2. As we already mentioned in Section 6.1, this is equivalent to B being left equivalent to a matrix

$$H = \begin{bmatrix} h_1 & \\ \bar{h} & I_{n-1} \end{bmatrix}, \tag{6.15}$$

where $h_1 = |\det B|$. The runtime of step 2 is dominated by the claimed complexity.

Then, Subroutine `TrivialLowerHermiteForm` called in step 3 certifies that B has the desired structure and returns matrix H. The complexity of the subroutine depends on the

average length of the columns of B, for which

$$\frac{1}{n}\sum_{j=1}^{n}\text{length}(B_{1..n,j}) \leq \frac{1}{n}\left(\log\left(\prod_{j=1}^{n}\|B_{1..n,j}\|\right)+n\right) \leq 2.5\log\lambda+1$$

as per Lemma 6.7. Since $\lambda \in O(n\|A\|)$, the complexity of step 3 is also $O(n^{\omega}\,\mathsf{B}(\log n + \log\|A\|)\,(\log n)^2)$.

Subroutine `TrivialLowerHermiteForm` itself might return FAIL with probability at most $1/8$. In addition, if it does not fail, the output of the subroutine will be NOT TRIVIAL with probability at most $1/2$ by Theorem 6.2. This makes the probability of success of Algorithm `SmithFormMultipliers` to be at least $1-(1/8+1/2+1/8) = 1/4$ as claimed.

Now, since we know that $B \equiv_L H$, the matrix $V := BH^{-1}$ in step 4 must be integral and unimodular. The evaluation of the product

$$BH^{-1} = B\begin{bmatrix} 1 & \\ -\bar{h} & I_{n-1}\end{bmatrix}\begin{bmatrix} 1/h_1 & \\ & I_{n-1}\end{bmatrix}$$

is covered exactly under Lemma 2.8 and can be computed, for $d = n(2.5\log\lambda+1)$, in time $O(n^{\omega}\mathsf{M}(\log n+\log\|A\|))$. Furthermore, by Lemma 4.12, V is a unimodular Smith massager for A.

Finally, by the properties of the Smith massager, matrix $U := AVS^{-1}$ is integral, and unimodular since V is unimodular. By Lemma 2.10, matrix U can be computed in $O(n^{\omega}\mathsf{M}(\log n+\log\|A\|))$ bit operations. □

6.3.1 Sizes of V and U

It will be important to have good bounds on the magnitude of entries in matrices V and U, in order to facilitate the complexity analysis of operations that may use V and U in general.

Lemma 6.13. *Algorithm* `SmithFormMultipliers` *returns unimodular Smith multiplier matrices $V, U \in \mathbb{Z}^{n\times n}$ which satisfy that:*

- $\|V_{1..n,j}\| \leq cn\|A\| \cdot \begin{cases} |\det A|+n & \text{if } j=1 \\ s_j & \text{otherwise}\end{cases}$,

- $\|U_{1..n,j}\| \le cn^2\|A\|^2 \cdot \begin{cases} |\det A| + n & \text{if } j = 1 \\ 1 & \text{otherwise} \end{cases}$.

for $c = 420$.

Proof. First of all, for $\lambda := 105\max(n, \lceil(\det 2S)^{1/n}\rceil)$, we have, by Hadamard's bound, that $\lambda \le 210n\|A\|$.

By construction, we know that $\|B_{1..n,j}\| \le 2\lambda s_j$ for every $j = 1, \ldots, n$. Then, multiplying B with H^{-1} alters only the first column of B. The magnitude of the first column of $V = BH^{-1}$ satisfies that

$$\|V_{1..n,1}\| \le \left(2\lambda h_1 \sum_{j=1}^{n} s_j\right)/h_1 \le 2\lambda(|\det A| + n).$$

Furthermore, since $U = AVS^{-1}$, the magnitude of every column of U is bounded by

$$\|U_{1..n,j}\| \le n\|A\|\|V_{1..n,j}\|/s_j.$$

By replacing λ with $210n\|A\|$, the claimed bounds follow. □

Corollary 6.14. *The average bitlength of the columns of both V and U is bounded by $O(\log n + \log \|A\|)$.*

6.4 Computing an outer product adjoint formula

In this section, we mention an application of the Smith form with the multiplier matrices. Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular and assume that we have precomputed the Smith form S of A, together with unimodular matrices U and V such that $AV = US$.

Let $s := S_{n,n}$ be the largest invariant factor of A. As a tool to compute A^{-1}, Storjohann (2015) developed an algorithm to compute an *outer product adjoint formula* for A: a triple of matrices $(\bar{V}, S, \bar{U})$ such that

$$\mathrm{Rem}(sA^{-1}, s) = \mathrm{Rem}(\bar{V}(sS^{-1})\bar{U}, s).$$

Moreover, $\bar{V} = (\bar{V}\ \mathbf{c}\mathrm{mod}\ S)$ and $\bar{U} = (\bar{U}\ \mathbf{r}\mathrm{mod}\ S)$. While a tight upper bound for the number of bits required to represent $\mathrm{Rem}(sA^{-1}, s)$ explicitly is $O(n^3(\log n + \log \|A\|))$, we note that the triple $(\bar{V}, S, \bar{U})$ requires only $O(n^2(\log n + \log \|A\|))$ bits.

Example 6.15. *Matrix*

$$A = \begin{bmatrix} -6 & 3 & -13 & -15 \\ -4 & 19 & 12 & -1 \\ -4 & 10 & -6 & 17 \\ -26 & -13 & 1 & -2 \end{bmatrix}$$

has Smith form $S = \operatorname{diag}(s_1, s_2, s_3, s_4) = \operatorname{diag}(1, 1, 9, 29088)$ *and*

$$s_4 A^{-1} = \begin{bmatrix} -271 & -402 & -373 & -937 \\ 580 & 920 & 524 & -356 \\ -1074 & 804 & -870 & 258 \\ -784 & -352 & 1008 & 80 \end{bmatrix}.$$

An outer product adjoint formula for A *is given by* $(\bar{V}, S, \bar{U})$ *where*

$$\bar{V} = \begin{bmatrix} 0 & 0 & 7 & 805 \\ 0 & 0 & 5 & 23668 \\ 0 & 0 & 3 & 6 \\ 0 & 0 & 4 & 10224 \end{bmatrix} \quad \textit{and} \quad \bar{U} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 2 & 2 & 0 & 2 \\ 20829 & 1750 & 28943 & 16203 \end{bmatrix}.$$

For this particular A*, which is well conditioned, multiplying out* $\bar{V}(s_4 S^{-1})\bar{U}$ *and reducing entries in the symmetric range modulo* s_4 *gives* $s_4 A^{-1}$*. Because* $s_1 = s_2 = 1$ *the first two columns of* V *and first two rows of* U *can be omitted, giving*

$$\begin{bmatrix} 7 & 805 \\ 5 & 23668 \\ 3 & 6 \\ 4 & 10224 \end{bmatrix} \begin{bmatrix} 3232 & \\ & 1 \end{bmatrix} \begin{bmatrix} 2 & 2 & 0 & 2 \\ 20829 & 1750 & 28943 & 16203 \end{bmatrix} \equiv s_4 A^{-1} \bmod s_4.$$

There is a close relationship between an outer product adjoint formula and the unimodular Smith multipliers U and V.

Lemma 6.16. *Let* $U, V \in \mathbb{Z}^{n \times n}$ *be unimodular matrices such that* $AV = US$*. Then, the triple* $(V \,\mathbf{c}\bmod S, S, U^{-1} \,\mathbf{r}\bmod S)$ *gives an outer product adjoint formula for* A*.*

Proof. We have that $sA^{-1} = V(sS^{-1})U^{-1}$. Furthermore, $V(sS^{-1}) = (V \,\mathbf{c}\bmod S)(sS^{-1}) \bmod s$ and $(sS^{-1})U^{-1} = (sS^{-1})(U^{-1} \,\mathbf{r}\bmod S) \bmod s$, and so

$$\operatorname{Rem}(sA^{-1}, s) = \operatorname{Rem}((V \,\mathbf{c}\bmod S)(sS^{-1})(U^{-1} \,\mathbf{r}\bmod S), s).$$

□

Storjohann (2015) gives a randomized Las Vegas algorithm to compute an outer product adjoint formula in

$$O(n^2(\log n)\mathsf{B}(n(\log n + \log \|A\|)) \tag{6.16}$$

plus

$$O(n^3 \max(\log n, \log \|A^{-1}\|)\mathsf{B}(\log n + \log \|A\|)) \tag{6.17}$$

bit operations. Note that (6.16) implies that fast (pseudo-linear) integer arithmetic needs to be used to achieve a cost that is softly cubic in n, while (6.17) reveals a sensitivity to the condition number of A. Indeed, we may have $\log \|A^{-1}\| \in \Omega(n)$, in which case the upper bound in (6.17) becomes quartic in n. It was left as an open question if an outer product adjoint formula can be computed in time $(n^\omega \log \|A\|)^{1+o(1)}$ bit operations. Here, we can resolve this question by using the approach of Lemma 6.16.

Theorem 6.17. *Assume we have the output (S, V, U) of Algorithm* `SmithFormMultipliers`(A). *Then, an outer product adjoint formula for A can be computed in time $O(n^\omega\, \mathsf{M}(\log n + \log \|A\|) \log n)$.*

Proof. Let $\bar{V} := V \bmod S$. This can be done in time $O(n \sum_{i=1}^{n} \mathsf{M}(\text{length}(V_{1\ldots n,i}))$. By Corollary 6.14, $\sum_{i=1}^{n} \text{length}(V_{1\ldots n,i}) \in O(n(\log n + \log \|A\|))$, which shows that the matrix $\bar{V}$ can be computed in time $O(n\mathsf{M}(n(\log n + \log \|A\|))$.

It remains to compute $\bar{U} := U^{-1} \bmod S$. Let $D \in \mathbb{Z}^{m\times m}$ be the partial column linearization of U as in Theorem 3.7. It will be that $m \in O(n)$, and again by Corollary 6.14, $\log \|D\| \in O(\log n + \log \|A\|)$. Therefore, by Lemma 2.3, we can compute a straight line formula for D^{-1} in time $O(n^\omega\, \mathsf{M}(\log n + \log \|A\|) \log n)$. The formula consists of $O(\log n)$ integer matrices of dimension m and bitlength in $O(\log n + \log \|A\|)$.

Finally, we can compute $U^{-1} \bmod S$ by evaluating $D^{-1} \bmod \text{diag}(S, I_{m-n})$ using the straight line formula. The evaluation of the formula requires $O(\log n)$ matrix multiplications where the first operand is an $m \times m$ integer matrix reduced $\bmod \text{diag}(S, I)$ and the second operand is an $m\times m$ integer matrix with bitlength in $O(\log n+\log \|A\|)$. This type of matrix multiplication falls exactly under Lemma 2.10 by simply transposing the operation. Therefore, we can compute $U^{-1} \bmod S$ in time $O(n^\omega\, \mathsf{M}(\log n + \log \|A\|) \log n)$. □

An application of the outer product adjoint formula is to compute the proper fractional part of a linear system solution. Let $b \in \mathbb{Z}^{n\times 1}$ satisfy $\log ||b|| \in (n \log ||A||)^{1+o(1)}$. Then

$$A^{-1}b = \overbrace{A^{-1}b - \text{Rem}(sA^{-1}b, s)/s}^{\in \mathbb{Z}^n} + \text{Rem}(sA^{-1}b, s)/s,$$

where $\mathrm{Rem}(sA^{-1}b, s)/s$ is a vector of proper fractions. By Lemma 2.5, $A^{-1}b \in \mathbb{Q}^{n\times 1}$ can be computed in a Las Vegas fashion in $(n^\omega \log ||A||)^{1+o(1)}$ bit operations, or $O(n^3 \log ||A||)$ bit operations if $\omega = 3$. If an outer product adjoint formula for A is known, then the proper fractional part of $A^{-1}b$ can be computed in only $(n^2 \log ||A||)^{1+o(1)}$ bit operations. The following result is a corollary of (Storjohann, 2015, Lemma 4.11).

Lemma 6.18. *Assume we have an outer product adjoint formula $(\bar{V}, S, \bar{U})$ for a nonsingular $A \in \mathbb{Z}^{n\times n}$, and let $s = S_{n,n}$. Given a vector $b \in \mathbb{Z}^{n\times 1}$ with $\log \|b\| \in O(\log s)$, we can compute $\mathrm{Rem}(sA^{-1}b, s)$ in time $O(n\,\mathsf{M}(\log s))$.*

Example 6.19. *Let $A \in \mathbb{Z}^{n\times n}$ be the matrix of Example 6.15 and*

$$b = \begin{bmatrix} 25 \\ 94 \\ 12 \\ -2 \end{bmatrix}.$$

Then

$$\bar{V}(29088S^{-1})\bar{U}b \equiv \begin{bmatrix} 11011 \\ 20716 \\ 8682 \\ 17424 \end{bmatrix} \bmod 29088.$$

Indeed, we have

$$A^{-1}b = \begin{bmatrix} -2 \\ 3 \\ 1 \\ -2 \end{bmatrix} + \begin{bmatrix} 11011 \\ 20716 \\ 8682 \\ 17424 \end{bmatrix} \frac{1}{29088}.$$

Applying Lemma 6.18 with $b = I_n$ gives the following corollary of Theorems 6.12 and 6.17.

Corollary 6.20. *Given a nonsingular integer input matrix $A \in \mathbb{Z}^{n\times n}$, the largest invariant factor s of A, together with $\mathrm{Rem}(sA^{-1}, s)$, can be computed in a Las Vegas fashion in*

$$O(n^\omega\, \mathsf{B}(\log n + \log ||A||)(\log n)^2 + n^2\, \mathsf{M}(\log s))$$

bit operations. This is bounded by $(n^3 \log ||A||)^{1+o(1)}$ bit operations.

Chapter 7

Deterministic linear system solving

In this chapter, we present a deterministic reduction to matrix multiplication for the problem of linear system solving: given as input a nonsingular $A \in \mathbb{Z}^{n \times n}$ and $b \in \mathbb{Z}^{n \times 1}$, compute $A^{-1}b$.

In Chapter 2, we saw that we can solve linear systems within our target complexity using high-order lifting. The algorithm, however, was not deterministic, as we had to pick a random lifting modulus X such that $X \perp \det A$ in a Las Vegas fashion. In this chapter, we will show how we can derandomize this procedure at the expense of an extra $\log n$ factor in the time complexity.

In addition, we will describe a more general cost model than the one used so far in the thesis. We will express the asymptotic time complexity of the algorithm in terms of the function $\mathsf{MM}(n, d)$ which bounds the cost to multiply together, modulo 2^d, two $n \times n$ integer matrices. So far we have used the slightly less general case that $\mathsf{MM}(n, d) \in O(n^\omega \mathsf{M}(d))$. But in this case, the algorithm will be given only in terms of matrix multiplications with operand matrices that have dimension n and length of entries $d \in O(\log n + \log \|A\|)$ where $A \in \mathbb{Z}^{n \times n}$ is the input matrix. Apart from the fact that the problem is reduced to just a number of matrix multiplications, using the more general cost model allows the function MM in the cost of the algorithm to be replaced by any reasonable matrix multiplication cost function.

We design an algorithm that computes the minimal integer t such that all denominators of the entries in $2^t A^{-1}$ are relatively prime to 2. In this case, for an integer vector b having entries with bitlength $O(n)$ times as large as the bitlength of entries in A, our algorithm also produces the 2-adic expansion of $2^t A^{-1} b$ up to a precision high enough so that $A^{-1}b$ over $\mathbb{Q}$ can be recovered using rational number reconstruction. Both t and the 2-adic

expansion can be computed in

$$O(\mathsf{MM}(n, \log n + \log ||A||) \cdot (\log n + \operatorname{loglog} ||A||)(\log n))$$

bit operations.

Our cost analysis will make the following regularity assumptions on $\mathsf{MM}(n, d)$: super-quadricity in n ($\mathcal{H}^{2\leq n}_{\mathsf{MM}}$), super-linearity in d ($\mathcal{H}^{1\leq d}_{\mathsf{MM}}$), and at most quadratic in d ($\mathcal{H}^{d\leq 2}_{\mathsf{MM}}$). Under these assumptions, our cost analysis is valid for any MM that satisfies $\mathsf{MM}(n, d) \in \Omega(n^2 d)$ and $\mathsf{MM}(n, d) \in O(n^3 d^2)$.

Our approach to derandomize integer linear solving is based on ideas from Gupta et al. (2012) for polynomial matrices. Corresponding to any A there exists a *2-decomposition*: a tuple (P, H) of matrices where P is a permutation, H is a row Hermite form with powers of 2 on the diagonal, and the matrix $U := APH^{-1}$ is nonsingular with odd determinant.

Example 7.1. *Assuming the permutation is already applied,*

$$\underset{AP}{\begin{bmatrix} 27 & 99 & 92 \\ 32 & 116 & -124 \\ 195 & -121 & -8 \end{bmatrix}} = \underset{U}{\begin{bmatrix} 27 & 18 & -19 \\ 32 & 21 & -37 \\ 195 & -79 & -127 \end{bmatrix}} \underset{H}{\begin{bmatrix} 1 & 1 & 12 \\ & 4 & 4 \\ & & 16 \end{bmatrix}}. \tag{7.1}$$

The utility of this construction is that it replaces the linear system solution

$$A^{-1}b$$

with two linear system solutions

$$y = U^{-1}b \quad \text{and} \quad H^{-1}y,$$

such that we can deterministically find a lifting modulus for U since $2 \perp \det U$ and we can exploit the structure of H to solve them. However, unlike the case when working with polynomial entries, in the integer setting, we do not have good *a priori* bounds on the magnitude of entries in $U := APH^{-1}$.

As such, we will introduce the notion of a *2-massager*. This is a tuple of matrices (P, S, M) such that P is a permutation, S is in Smith form with powers of 2 on the diagonal, and M is unit upper triangular with offdiagonal entries in each column of magnitude strictly less than the corresponding diagonal entry of S. In addition, the matrix $APMS^{-1}$ is nonsingular with odd determinant and satisfies $||APMS^{-1}|| \leq n||A||$.

Example 7.2. *For the matrix in (7.1) we have*

$$\underset{AP}{\begin{bmatrix} 27 & 99 & 92 \\ 32 & 116 & -124 \\ 195 & -121 & -8 \end{bmatrix}} \underset{M}{\begin{bmatrix} 1 & 3 & 5 \\ & 1 & 15 \\ & & 1 \end{bmatrix}} \underset{S^{-1}}{\begin{bmatrix} 1 & & \\ & 1/4 & \\ & & 1/16 \end{bmatrix}} = \underset{APMS^{-1}}{\begin{bmatrix} 27 & 45 & 107 \\ 32 & 53 & 111 \\ 195 & 116 & -53 \end{bmatrix}}.$$

We will now properly define the more general cost model that we will follow in this chapter along with extensions of some routines that we will need under the new model. In Section 7.2, we will show how the derandomization approach of Gupta et al. (2012) for polynomial matrices can be adapted to the integer case in order to compute a 2-decomposition of A and defines the 2-massager along with a duality property between 2-massagers and 2-decompositions. Then, Section 7.3 shows how to compute a suitable left equivalent triangular form of A up to a column permutation and over $\mathbb{Z}/(2^d)$ for some $d \in \mathbb{Z}_{>0}$. The main operations that we need to compute a 2-massager, along with a complexity analysis, are given in Section 7.4. Finally, Section 7.5 gives the deterministic algorithm for linear system solving.

7.1 A more general cost model

We assume that integers are stored using their binary representation. Thus, for any X a power of two, we can deduce the X-adic representation of a positive integer without any computation. The algorithms we propose in this chapter are designed to not require any radix conversions.

We will give cost estimates using a function

$$\mathsf{MM}(n, d) : (\mathbb{R}_{\geq 0}, \mathbb{R}_{\geq 0}) \to \mathbb{R}_{\geq 0}$$

which for any nonnegative integers $n' \leq n$ and $d' \leq d$ bounds the number of bit operations required to multiply modulo $2^{d'}$ two square matrices over $\mathbb{Z}/(2^{d'})$ of dimension n'. A lower bound is $\mathsf{MM}(n, d) \in \Omega(n^2 d)$ and, using an obvious block decomposition, we have $\mathsf{MM}(cn, d) \in O(\mathsf{MM}(n, d))$ for any positive constant c.

Furthermore, we assume that MM satisfies the following regularity assumptions:

$$\begin{aligned}
\mathcal{H}_{\mathsf{MM}}^{1\leq d} &: k\,\mathsf{MM}(n, d/k) \leq \mathsf{MM}(n, d) && \text{for all } 1 \leq k \leq d \\
\mathcal{H}_{\mathsf{MM}}^{2\leq n} &: k^2\,\mathsf{MM}(n/k, d) \leq \mathsf{MM}(n, d) && \text{for all } 1 \leq k \leq n \\
\mathcal{H}_{\mathsf{MM}}^{d\leq 2} &: \mathsf{MM}(n, kd) \leq k^2\,\mathsf{MM}(n, d) && \text{for all } 1 \leq k
\end{aligned}$$

The first two of these assumptions allow us to simplify the cost estimates of algorithms that recurse on the precision d and dimension n, respectively. Also, many of the cost estimates derived in susbsequent sections are of the form $\mathsf{MM}(n,d)$ for a d that satisfies $d \in O(\log n + \log \|A\|))$ where A is the input matrix to the overall problem. In that case, the third assumption gives that $\mathsf{MM}(n,d) \in O(\mathsf{MM}(n, \log(n||A||)))$.

7.1.1 Computational tools in terms of MM

We will need to make use of several computational tools in this chapter as well. For this reason, we summarize them here so that their cost estimate is given in terms of the function MM. We begin with some well known algorithms which reduce computations to matrix multiplication, and later, we extend some of our results given in Chapter 2.

The first two results are needed only for matrices over $\mathbb{Z}/(2)$. The cost of the triangular matrix inversion algorithm (Bunch and Hopcroft, 1974) follows the recurrence $I(n) \leq 2I(n/2) + O(\mathsf{MM}(n/2,1))$. Assuming $\mathcal{H}^{2\leq n}_{\mathsf{MM}}$, gives the following.

Lemma 7.3. *If $A \in \mathbb{Z}/(2)^{n\times n}$ is unit upper triangular, then its inverse can be computed in time $O(\mathsf{MM}(n,1))$.*

For the next result we can use the LQUP-decomposition of Ibarra et al. (1982). For an $m \times n$ matrix with $m \leq n$, the algorithm recurses on m and has complexity following $T(m) \leq 2T(m/2) + I(m) + O((n/m)\mathsf{MM}(m,1))$. Again assuming $\mathcal{H}^{2\leq n}_{\mathsf{MM}}$, gives the following for $m = n$.

Lemma 7.4. *Given $A \in \mathbb{Z}/(2)^{n\times n}$, the rank r of A together with a nonsingular $U \in \mathbb{Z}/(2)^{n\times n}$ and an $n \times n$ permutation matrix P such that UAP has its first r columns those of I_n and its last $n-r$ rows zero can be computed in time $O(\mathsf{MM}(n,1)\log n)$.*

A matrix $A \in \mathbb{Z}/(2^d)^{n\times n}$ is unimodular if its determinant is odd. In this case the inverse of A, denoted by A^{-1}, is the unique matrix from $\mathbb{Z}/(2^d)^{n\times n}$ such that $\mathrm{Rem}(A^{-1}A, 2^d) = I_n$. Algebraic Newton iteration (von zur Gathen and Gerhard, 2013, Algorithm 9.3) gives the following, assuming $\mathcal{H}^{1\leq d}_{\mathsf{MM}}$.

Lemma 7.5. *Let $A \in \mathbb{Z}/(2^d)^{n\times n}$ be unimodular. Assuming that $\mathrm{Rem}(A^{-1}, 2)$ is known, then $\mathrm{Rem}(A^{-1}, 2^d)$ can be computed in time $O(\mathsf{MM}(n,d))$.*

Moreover, we will extend the linear system solving routine supporting Theorem 2.5, and the unbalanced to balanced multiplication reductions of Section 2.5 to equivalent statements using the cost model of this section.

The following theorem is the equivalent of Theorem 2.5. In this chapter, we will avoid the use of randomization by being able to always choose a power of 2 as lifting modulus, or in other words, the input matrix for the solving will have odd determinant.

Theorem 7.6. *Assume that we have a lifting modulus X as in Lemma 2.1. If entries in $B \in \mathbb{Z}^{n\times m}$ are reduced modulo X^d and $m \times d \in O(n)$, then* $\mathrm{Rem}(A^{-1}B, X^d)$ *can be computed in time* $O(\mathsf{MM}(n, \log n + \log \|A\|) \log n)$.

The next lemma can be shown using the same technique as in Lemma 2.8. The difference here is that we compute the product reduced modulo a given power of 2 and that we have a matrix W in place of a vector.

Lemma 7.7. *Let a matrix $M \in \mathbb{Z}^{n\times m}$, a matrix $W \in \mathbb{Z}^{m\times r}$ and an integer $k \in \mathbb{Z}_{\geq 0}$. Furthermore, assume that $kr \in O(d)$ and*

$$\sum_{j=1}^{m} \mathrm{length}(M_{1..n,j}) \leq d \quad \textit{and} \quad \mathrm{length}(W) \leq d/r$$

for some $d \in \mathbb{Z}_{\geq 0}$. We can compute $\mathrm{Rem}(MW, 2^k)$ *in time* $O((n/m)\mathsf{MM}(m, d/m + \log m))$.

Finally, Lemma 7.8 and Corollary 7.9 that follow are extensions of Lemma 2.9 and Corollary 2.10 respectively. Note that we will not need Lemma 7.8 in this chapter, but we give it for completeness.

Lemma 7.8. *For matrices $U, M \in \mathbb{Z}^{n\times n}$, assume that*

$$\sum_{i=1}^{n} \mathrm{length}(U_{i,1..n}) \leq nd \quad \textit{and} \quad \sum_{j=1}^{n} \mathrm{length}(M_{1..n,j}) \leq nd$$

for some $d \in \mathbb{Z}_{\geq 0}$. We can compute the product UM in time $O(\mathsf{MM}(n, d + \log n))$.

Corollary 7.9. *For matrices $A, M \in \mathbb{Z}^{n\times n}$, assume that*

$$\mathrm{length}(A) \leq d \quad \textit{and} \quad \sum_{j=1}^{n} \mathrm{length}(M_{1..n,j}) \leq nd$$

for some $d \in \mathbb{Z}_{\geq 0}$. We can compute the product AM in time $O(\mathsf{MM}(n, d + \log n))$.

7.2 Triangular 2-Smith form inverse decomposition

In this section, we will describe how we can efficiently compute the inverse of a matrix in triangular Smith form. We already discussed in the introduction that our motivation is to derandomize linear system solving by decomposing the input matrix A into one matrix with odd determinant and another which will be upper triangular with even determinant.

Subsection 7.2.1 illustrates how the derandomization approach of Gupta et al. (2012) for polynomial matrices can be adapted to the integer case in order to compute a triangular 2-Smith form H of A, that is, an upper triangular matrix where the ith diagonal entry is the largest power of 2 that divides the ith invariant factor of A and, also divides the ith row of H. At the end of the subsection, we show with an example why the same technique won't work in the integer setting. Then, in Subsection 7.2.2, we define a new object, that we call the 2-massager, which allows us to efficiently represent the inverse of H.

7.2.1 2-decompositions

As we have already mentioned, two matrices, in this case over $\mathbb{Z}/(2^d)$, are said to be left equivalent if one can be obtained from the other by premultiplying with a unimodular matrix. The unimodular matrix represents a set of row operations converting one matrix into the other. Corresponding to every $A \in \mathbb{Z}/(2^d)^{n\times n}$ there is a permutation matrix P such that $\mathrm{Rem}(AP, 2^d)$ is left equivalent to a matrix

$$\begin{bmatrix} 2^{e_1} & * & \cdots & * & * & \cdots & * \\ & 2^{e_2} & \cdots & * & * & \cdots & * \\ & & \ddots & \vdots & \vdots & \cdots & * \\ & & & 2^{e_r} & * & \cdots & * \\ & & & & & & \\ & & & & & & \\ & & & & & & \end{bmatrix} \in \mathbb{Z}/(2^d)^{n\times n} \tag{7.2}$$

that is in *triangular Smith form*: $e_1 \le e_2 \le \cdots \le e_r$ and all entries in row i are divisible by 2^{e_i}, $1 \le i \le r$. The e_i are unique.

The above discussion is for a matrix A over $\mathbb{Z}/(2^d)$. Now let $A \in \mathbb{Z}^{n\times n}$ be a nonsingular integer matrix. Then, the 2-*Smith form* of A is the matrix $\mathrm{diag}(2^{e_1}, 2^{e_2}, \ldots, 2^{e_n})$ with 2^{e_i} the largest power of two which divides the i'th invariant factor of the Smith form of A over $\mathbb{Z}$, $1 \le i \le n$. Since 2^{e_n} divides $\det A$, and $|\det A| \le n^{n/2}||A||^n$, the 2-Smith form of A over $\mathbb{Z}$ can be recovered by computing a triangular Smith form of $\mathrm{Rem}(A, X^{n+1})$ over $\mathbb{Z}/(X^{n+1})$,

where X is the smallest power of two such that $X \geq n^{1/2}||A||$. For the remainder of this section we will refer to the exponent of the modulus X^p as the "precision" p.

Gupta et al. (2012) show how one can compute a permutation matrix P such that $\mathrm{Rem}(AP, X^{n+1})$ is left equivalent (over $\mathbb{Z}/(X^{n+1})$) to a triangular Smith form H. (The algorithms in (Gupta et al., 2012) were developed for polynomial matrices but the approach carries over directly to integer matrices.) As mentioned previously, the precision $n+1$ is large enough to ensure that H will be as in (7.2) with $r = n$. The matrix $U := APH^{-1}$ is integral with $2 \perp \det U$. We refer to the pair of matrices (P, H) as a 2-*decomposition of* A and note that $A = UHP^{-1}$.

Working with precision $n+1$ to compute (P, H) in one fell swoop is too expensive. Instead, as we did in the Smith form algorithm in Chapter 5, Gupta et al. (2012) use the observation that many of the initial invariant factors can be recovered using a considerably lower precision. For any $0 \leq m \leq n$, we can partition the invariant factors in the 2-Smith form of A as follows:

$$\mathrm{diag}(\overbrace{2^{e_1}, 2^{e_2}, \ldots, 2^{e_m}}^{\text{first } m}, \overbrace{2^{e_{m+1}}, 2^{e_{m+2}}, \ldots, 2^{e_*}}^{\text{next } \lceil (n-m)/2 \rceil}, \overbrace{2^{e_*}, 2^{e_*}, \ldots, 2^{e_n}}^{\text{last } \lfloor (n-m)/2 \rfloor})$$

Lemma 7.10. *Let $A \in \mathbb{Z}^{n \times n}$ be nonsingular and $0 \leq m < n$. If $X \in \mathbb{Z}$ satisfies $X \geq n^{1/2}||A||$ then the 2-Smith form of A has at most $\lfloor (n-m)/2 \rfloor$ invariant factors $\geq X^{2n/(n-m)}$.*

An application of Lemma 7.10 with $m = 0$ states that precision 2 is sufficient to compute a permutation P such that $\mathrm{Rem}(AP, X^2)$ is left equivalent to a triangular Smith form as in (7.2) with $r \geq \lceil n/2 \rceil$, that is, the first half of the invariant factors require only constant precision. Next, the algorithm works at precision $\lceil 2n/(n-r) \rceil$ to recover at least $\lceil (n-r)/2 \rceil \geq n/4$ of the remaining $n-r$ invariant factors, and so on. At each step, the algorithm exploits our usual dimension $\times$ precision $\leq$ invariant compromise: the number of remaining invariant factors times the precision is $\Theta(n)$.

At the beginning of an iteration, the algorithm has already computed a permutation P_1 such that for the previous working precision p, the matrix $\mathrm{Rem}(AP_1, X^p)$ is left equivalent (over $\mathbb{Z}/(X^p)$) to a triangular Smith form as in (7.2) with the first m rows nonzero. (At the beginning of the first iteration $r = 0$.) Notice that, in contrast with the Smith form algorithm, in this case, the invariant factors are computed iteratively starting from the beginning and not the end. Let

$$H_1 = \left[\begin{array}{c|c} E_1 & \\ \hline & I_{n-m} \end{array}\right]$$

be the matrix in (7.2) but with last $n-m$ columns replaced by those of I_n. We refer to the pair of matrices (P_1, H_1) as an *index*-$(0, m)$ 2-*decomposition* of A: the matrix $B := AP_1H_1^{-1}$ will be integral with first m columns having full rank modulo 2. If $m = n$, we are done, so, assume that $m < n$.

Next, the algorithm updates the precision to $p := \lceil 2n/(n-m) \rceil$ and computes a permutation $P_2 = \mathrm{diag}(I_m, *)$ such that $\mathrm{Rem}(BP_2, X^p)$ is left equivalent to a triangular Smith form having the shape

$$\left[\begin{array}{c|c|c} I_m & V_2 & * \\ \hline & E_2 & * \\ \hline & & \end{array}\right].$$

By Lemma 7.10, the column dimension r of E_2 satisfies $r \geq \lceil (n-m)/2 \rceil$. (We remark that because the first m columns of B have full rank modulo 2, there exists a triangular Smith form of B over $\mathbb{Z}/(X^p)$ with first m columns those of I_n.) Set

$$H_2 = \left[\begin{array}{c|c|c} I_m & V_2 & \\ \hline & E_2 & \\ \hline & & I_{n-m-r} \end{array}\right].$$

We call the pair (P_2, H_2) an *index*-(m, r) 2-*decomposition* of B. Because of the structure of the matrices, we have $(P_1H_1^{-1})(P_2H_2^{-1}) = (P_1P_2)(H_2H_1)^{-1}$. Then P_1P_2 is a permutation with $\mathrm{Rem}(AP_1P_2, X^p)$ being left equivalent to a triangular Smith form that has first $m+r$ columns equal to those of

$$H_2H_1 = \left[\begin{array}{c|c|c} E_1 & V_2 & \\ \hline & E_2 & \\ \hline & & I_{n-m-r} \end{array}\right].$$

Since each iteration computes at least half of the remaining invariant factors of the 2-Smith form, the number of iterations is bounded by $O(\log n)$. The following theorem captures the approach of Gupta et al. (2012) described above to compute a 2-decomposition.

Theorem 7.11. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular, $0 \leq m \leq n$, and $0 \leq r \leq n-m$. If (P_1, H_1) is an index-$(0, m)$ 2-decomposition for A, and (P_2, H_2) is an index-(m, r) 2-decomposition for $B := AP_1H_1^{-1}$, then (P_1P_2, H_2H_1) is an index-$(0, m+r)$ 2-decomposition for A.*

To avoid expression swell, the H_i computed at each step can be transformed into *Hermite canonical form*: offdiagonal entries are reduced modulo the diagonal entry in the same column. The H_i are then unique up to the choice of the permutations P_i. In the polynomial

setting, the inverse H^{-1} of a matrix H in Hermite form will be a proper matrix fraction, that is, degrees of entries in $(\det H)H^{-1}$ will be bounded by $\deg \det H$, thus ensuring that $U := AH^{-1}$ will have degree bounded by the degree of A. However, over the integers, H^{-1} may not be proper, frustrating attempts to obtain a good bound for the bitlength of entries in U. We end this section with an example of a class of ill-conditioned Hermite forms.

Example 7.12. *For $n = 2, 3, 4, \ldots$ consider the family of Hermite forms $H \in \mathbb{Z}^{n\times n}$ that are Toeplitz, with diagonal entry 2 and offdiagonal entries alternating between 1 and 0. For example, for $n = 6$,*

$$H = \begin{bmatrix} 2 & 1 & 0 & 1 & 0 & 1 \\ & 2 & 1 & 0 & 1 & 0 \\ & & 2 & 1 & 0 & 1 \\ & & & 2 & 1 & 0 \\ & & & & 2 & 1 \\ & & & & & 2 \end{bmatrix} \in \mathbb{Z}^{6\times 6}.$$

Offdiagonal entries in the first row of H^{-1} satisfy

$$H^{-1}_{1,j} = \begin{cases} -1/4 & \text{if } j = 2 \\ 1/8 & \text{if } j = 3 \\ (-1/2)H^{-1}_{1,j-1} + H^{-1}_{1,j-2} & \text{if } j \geq 4 \end{cases}$$

The closed form for this recurrence shows that $\log |H^{-1}_{1,j}| \in \Theta(j)$. For $n \geq 500$ the largest entry in $(\det H)H^{-1}$ has bitlength $\approx 1.35n$ compared to $\det H$ which has bitlength n.

7.2.2 2-massagers

Rather than computing a 2-decomposition for our input matrix, we propose the notion of a 2-massager.

Definition 7.13. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular. A* 2-massager *for A is a triple of matrices (P, S, M) from $\mathbb{Z}^{n\times n}$ such that:*

- *P is a permutation,*
- *$S = \mathrm{diag}(s_1, \ldots, s_n)$ is the 2-Smith form of A,*
- *M is unit upper triangular, and*
- *$APMS^{-1}$ is integral with $2 \perp \det APMS^{-1}$.*

We say that (P, S, M) *is a* reduced 2*-massager if the off-diagonal entries in* M *are reduced column modulo* S*.*

We begin by showing that there is an one to one correspondence between 2-massagers and 2-decompositions. Note that it follows from the uniqueness of the 2-Smith form S of A that the triangular Smith form H from any 2-decomposition of A will have the same diagonal entries as S.

Theorem 7.14. *Duality between* 2*-decompositions and* 2*-massagers:*

- *If* (P, H) *is a* 2*-decomposition of* A*, then* $(P, S, (S^{-1}H)^{-1})$ *is a* 2*-massager for* A*.*
- *If* (P, S, M) *is a* 2*-massager for* A*, then* (P, SM^{-1}) *is a* 2*-decomposition of* A*.*

Proof. Since H is in triangular Smith form, $S^{-1}H$ will be unit upper triangular and integral. It suffices to note that $APH^{-1} = AP(SS^{-1}H)^{-1} = AP(S^{-1}H)^{-1}S^{-1}$. □

Theorem 7.14 illustrates that a 2-massager is just a way to further decompose a 2-decomposition. And since we are interested in extracting factors of 2, that is, we want a compact description of the inverse of H, the massager M is able to achieve just that due to the fact that we can reduce its columns modulo S.

Suppose (P, S, M) is a 2-massager for A. Since $APMS^{-1}$ is integral, column i of APM must be congruent to zero modulo s_i, $1 \leq i \leq n$. This gives the following.

Lemma 7.15. *If* (P, S, M) *is a* 2*-massager for* A*, then a reduced* 2*-massager for* A *can be obtained by reducing offdiagonal entries in column* i *of* M *by* s_i*,* $1 \leq i \leq n$*.*

Our algorithm to compute a 2-massager for A will proceed in a similar manner to the algorithm for 2-decomposition sketched in the previous section. In order to simplify the discussion, it will be useful to introduce the following definition.

Definition 7.16. *Let* $B \in \mathbb{Z}^{n\times n}$ *be nonsingular with first* m *columns full rank modulo* 2*. An* index-(m, r) 2-massager *for* B *is a triple of matrices* (P, S, M) *from* $\mathbb{Z}^{n\times n}$ *such that*

- $P = \mathrm{diag}(I_m, *)$ *is a permutation,*
- $S = \mathrm{diag}(I_m, s_{m+1}, \ldots, s_{m+r}, I_{n-m-r})$ *with the principal* $(m + r) \times (m + r)$ *submatrix equal to that of the* 2*-Smith form of* B*,*

- *M is unit upper triangular with first m and last $n-m-r$ columns those of I_n.*
- *$BPMS^{-1}$ is integral with first $m+r$ columns having full rank modulo 2.*

(P, S, M) is a reduced *index-$(0, m+r)$ 2-massager if the off-diagonal entries in M are reduced column modulo S.*

Notice that an index-$(0, n)$ 2-massager for A is just a 2-massager for A as per Definition 7.13.

Suppose we have already computed an index-$(0, m)$ 2-massager (P_1, S_1, M_1) for A. (At the start of the first iteration $m = 0$, and we have the trivial index-$(0, 0)$ 2-massager (I_n, I_n, I_n).) Then, $B := AP_1M_1S_1^{-1}$ will be integral with first m columns of full rank modulo 2. If $m = n$, we are done, so, assume $m < n$. Our goal now is to compute an index-(m, r) 2-massager for B. By Lemma 7.10, we can guarantee to achieve $r \geq \lceil (n-m)/2 \rceil$ by working modulo X^p where X is the smallest power of two $\geq n^{1/2}||A||$ and $p = \lceil 2n/(n-m) \rceil$. Compute an index-$(m, r)$ 2-decomposition (P_2, H_2) for B. For

$$H_2 := \left[\begin{array}{c|c|c} I_m & V_2 & \\ \hline & E_2 & \\ \hline & & I_{n-m-r} \end{array}\right],$$

let S_2 be the diagonal matrix with same diagonal entries as H_2. Define D_2 by writing $S_2 = \mathrm{diag}(I_m, D_2, I_{n-m-r})$, that is, D_2 is the diagonal matrix with same diagonals as E_2. As a corollary of Theorem 7.14, for

$$M_2 := (S_2^{-1}H_2)^{-1} = \left[\begin{array}{c|c|c} I_r & -V_2(D_2^{-1}E_2)^{-1} & \\ \hline & (D_2^{-1}E_2)^{-1} & \\ \hline & & I_{n-m-r} \end{array}\right], \tag{7.3}$$

(P_2, S_2, M_2) will be an index-(m, r) 2-massager for B.

Then,

$$A(P_1M_1S_1^{-1})(P_2M_2S_2^{-1}) = BP_2M_2S_2^{-1}$$

will be integral with first $m+r$ columns having full rank modulo 2. By exploiting the duality of Theorem 7.14, we can combine (P_1, S_1, M_1) and (P_2, S_2, M_2) to obtain an index-$(0, m+r)$ 2-massager. Define D_1 by writing $S_1 = \mathrm{diag}(D_1, I_{n-m})$, that is, D_1 is comprised of the first m entries of the 2-Smith form of A. By duality, $(P_1, S_1M_1^{-1})$ is an index-$(0, m)$ 2-decomposition of A. Let $P = P_1P_2$ and $S = S_1S_2$. By Theorem 7.11, $(P, H_2S_1M_1^{-1})$ is then an index-$(0, m+r)$ 2-decomposition of A. Using duality in the opposite direction

shows that an index-$(0, m+r)$ 2-massager for A is given by $(P, S, (S^{-1}H_2S_1M_1^{-1})^{-1})$. Note that

$$(S^{-1}H_2S_1M_1^{-1})^{-1} = (S_1^{-1}S_2^{-1}H_2S_1M_1^{-1})^{-1} = M_1S_1^{-1}M_2S_1,$$

and so, obtain the following result.

Theorem 7.17. *Let $A \in \mathbb{Z}^{n\times n}$ be nonsingular, $0 \le m \le n$, and $0 \le r \le n-m$. If (P_1, S_1, M_1) is an index-$(0, m)$ 2-massager for A, and (P_2, S_2, M_2) is an index-(m, r) 2-massager for $B := AP_1M_1S_1^{-1}$, then $(P_1P_2, S_1S_2, M_1S_1^{-1}M_2S_1)$ is an index-$(0, m+r)$ 2-massager for A.*

If we write

$$M_1 = \left[\begin{array}{c|c} F_1 & \\ \hline & I_{n-m} \end{array}\right] \tag{7.4}$$

and

$$M_2 = \left[\begin{array}{c|c|c} I_m & W_2 & \\ \hline & F_2 & \\ \hline & & I_{n-m-r} \end{array}\right], \tag{7.5}$$

then, by Theorem 7.17, an index-$(0, m+r)$ 2-massager for A is given by (P, S, M) where

$$M = M_1S_1^{-1}M_2S_1 = \left[\begin{array}{c|c|c} F_1 & F_1D_1^{-1}W_2 & \\ \hline & F_2 & \\ \hline & & I_{n-m-r} \end{array}\right]. \tag{7.6}$$

The remainder of the chapter will be focused on bounding the complexity of the procedure we just described in this section.

7.3 Triangular Smith form algorithm

In this section, we give an algorithm for computing a triangular Smith form of a matrix A over $\mathbb{Z}/(2^d)$.

Theorem 7.18. *Problem* `TriangularSmithForm` *in Figure 7.1 can be solved in time*

$$O(\mathsf{MM}(n, d)(\log n + \log d)).$$

`TriangularSmithForm`(A, n, d)

Input: $A \in \mathbb{Z}/(2^d)^{n\times n}$ for $d \in \mathbb{Z}_{>0}$.

Output: U, P such that $U \in \mathbb{Z}/(2^d)^{n\times n}$ is unimodular, P is an $n\times n$ permutation matrix, and $\mathrm{Rem}(UAP, d)$ is in triangular Smith form over $\mathbb{Z}/(2^d)$.

Figure 7.1: Problem `TriangularSmithForm`

Proof. We describe a divide and conquer algorithm that recurses on the precision parameter d and has running time bounded by the recurrence

$$T(d) \leq \begin{cases} T(\lceil d/2\rceil) + T(\lfloor d/2\rfloor) + O(\mathsf{MM}(n,d)) & \text{if } d > 1 \\ O(\mathsf{MM}(n,1)\log n) & \text{if } d = 1. \end{cases}$$

Assuming $\mathcal{H}^{1\leq d}_{\mathsf{MM}}$, we have the solution

$$T(d) \in O(d\,\mathsf{MM}(n,1)\log n + \mathsf{MM}(n,d)\log d),$$

which simplifies to the target complexity since $d\,\mathsf{MM}(n,1) \leq \mathsf{MM}(n,d)$ using $\mathcal{H}^{1\leq d}_{\mathsf{MM}}$.

For the base case $d = 1$ use Lemma 7.4. Assume now that $d > 1$. Then set $d_1 = \lceil d/2\rceil$ and $d_2 = \lfloor d/2\rfloor$ and recursively compute

$$U_1, P_1 := \texttt{TriangularSmithForm}(\mathrm{Rem}(A, 2^{d_1}), n, d_1).$$

Let r be the number of nonzero rows of $\mathrm{Rem}(U_1AP_1, 2^{d_1})$ and let S be the $r \times r$ diagonal matrix with $S_{ii} = \mathrm{Rem}(U_1AP_1, 2^{d_1})_{ii}$, $1 \leq i \leq r$. Then,

$$\left[\begin{array}{c|c} S^{-1} & \\ \hline & I_{n-r} \end{array}\right] \mathrm{Rem}(U_1AP_1, 2^d)$$

is an integral matrix. We can thus split $\mathrm{Rem}(U_1AP_1, 2^d)$ into two parts using Rem and Quo. Let

$$\left[\begin{array}{c|c} T & T' \\ \hline & \end{array}\right] = \left[\begin{array}{c|c} S & \\ \hline & I_{n-r} \end{array}\right] \mathrm{Rem}\left(\left[\begin{array}{c|c} S^{-1} & \\ \hline & I_{n-r} \end{array}\right] \mathrm{Rem}(U_1AP_1, 2^d), 2^{d_1}\right)$$

and

$$\left[\begin{array}{c|c} B & B' \end{array}\right] = \left[\begin{array}{c|c} S & \\ \hline & I_{n-r} \end{array}\right] \mathrm{Quo}\left(\left[\begin{array}{c|c} S^{-1} & \\ \hline & I_{n-r} \end{array}\right] \mathrm{Rem}(U_1AP_1, 2^d), 2^{d_1}\right).$$

Then,

$$\mathrm{Rem}(U_1 A P_1, 2^d) = \left[\begin{array}{c|c} T & T' \\ \hline & \end{array}\right] + \left[\begin{array}{c|c} B & B' \end{array}\right] 2^{d_1}.$$

where T is $r \times r$ and B is $n \times r$. Note that by construction both

$$\left[\begin{array}{c|c} S^{-1} & \\ \hline & I_{n-r} \end{array}\right] \left[\begin{array}{c|c} T & T' \\ \hline & \end{array}\right]$$

and

$$\left[\begin{array}{c|c} S^{-1} & \\ \hline & I_{n-r} \end{array}\right] \left[\begin{array}{c|c} B & B' \end{array}\right]$$

will be integral.

Let $V = \mathrm{Rem}((S^{-1}T)^{-1}, 2^{d_1})$. Since the diagonal entries of S are powers of 2 of degree at most $d_1 - 1$, the matrix $2^{d_1} S^{-1}$ will be integral and, moreover, $\mathrm{Rem}(2^{d_1} S^{-1}, 2) = 0_{r \times r}$. The matrix

$$U_1' = \left[\begin{array}{c|c} I_r & \\ \hline & I_{n-r} \end{array}\right] - \left[\begin{array}{c|c} 2^{d_1} B V S^{-1} & \end{array}\right]$$

thus satisfies that $\mathrm{Rem}(U_1', 2) = I_n$, and hence, it is unimodular.

Next we show that

$$\mathrm{Rem}(U_1' U_1 A P_1, 2^d) = \left[\begin{array}{c|c} T & * \\ \hline & C 2^{d_1} \end{array}\right]$$

for an $(n-r) \times (n-r)$ matrix C. Considering the structure of U_1', it suffices to note, on the one hand, that

$$\begin{aligned} &\left[\begin{array}{c|c} 2^{d_1} B V S^{-1} & \end{array}\right] \left[\begin{array}{c|c} T & T' \\ \hline & \end{array}\right] \\ &\quad \equiv \left[\begin{array}{c|c} 2^{d_1} B (V S^{-1} T) & 2^{d_1} B V (S^{-1} T') \end{array}\right] \pmod{2^d} \\ &\quad \equiv \left[\begin{array}{c|c} 2^{d_1} B (I_r + *2^{d_1}) & 2^{d_1} B V * \end{array}\right] \pmod{2^d} \\ &\quad \equiv \left[\begin{array}{c|c} B 2^{d_1} & *2^{d_1} \end{array}\right] \pmod{2^d}, \end{aligned}$$

where the $*$ are integral matrices. Recall that $2d_1 \geq d$ and thus the term 2^{2d_1} vanishes modulo 2^d. On the other hand, we have that

$$\begin{aligned} &\left[\begin{array}{c|c} 2^{d_1} B V S^{-1} & \end{array}\right] \left[\begin{array}{c|c} B & B' \end{array}\right] 2^{d_1} \\ &\quad \equiv \left[\begin{array}{c|c} B V 2^{d_1} & \end{array}\right] \left[\begin{array}{c|c} S^{-1} & \\ \hline & I \end{array}\right] \left[\begin{array}{c|c} B & B' \end{array}\right] 2^{d_1} \pmod{2^d} \\ &\quad \equiv \left[\begin{array}{c|c} B V 2^{d_1} & \end{array}\right] \left[\begin{array}{c|c} * & * \end{array}\right] 2^{d_1} \pmod{2^d} \\ &\quad \equiv 0_{n \times n} \pmod{2^d}. \end{aligned}$$

We can then compute

$$U_2, P_2 := \texttt{TriangularSmithForm}(C, n-r, d_2)$$

and return

$$U, P = \operatorname{Rem}\left(\left[\begin{array}{c|c} I_r & \\ \hline & U_2 \end{array}\right] U_1' U_1, 2^d\right), P_1 \left[\begin{array}{c|c} I_r & \\ \hline & P_2 \end{array}\right].$$

It remains to bound the cost of the nonrecursive work. Other than some multiplications of matrices bounded in dimension by n and precision d, the only other computation is that of the inverse V. Lemmas 7.3 and 7.5 show that V can be computed in time $O(\mathsf{MM}(n, d))$. □

7.4 The 2-massager algorithm

In this section, we gather together everything that we presented so far in this chapter to design an algorithm that computes a reduced 2-massager (P, S, M) for a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ in time

$$O(\mathsf{MM}(n, \log n + \log \|A\|)(\log n + \log\log \|A\|) \log n).$$

The complexity translates into $O(\log n)$ number of iterations to compute all the columns of the massager, and to $O(\log n + \log\log \|A\|)$ matrix multiplications for each iteration with the $\log \|A\|$ factor appearing because of algorithm `TriangularSmithForm` recursing on the precision.

We begin by showing how to apply a reduced (index) 2-massager as per Definitions 7.13 and 7.16 to an input matrix A in order to produce the massaged matrix $U := APMS^{-1}$. If the massager is a 2-massager, then $2 \perp \det U$. If, on the other hand, we apply an index-(m, r) 2-massager to a matrix B which has the first m columns of full rank modulo 2, then the matrix $BPMS^{-1}$ has the first $m + r$ columns of full rank modulo 2.

Theorem 7.19. *Given a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ and a reduced index-(m, r) 2-massager (P, S, M) for A, we can apply the massager, that is, compute $APMS^{-1}$ in time*

$$O(\mathsf{MM}(n, \log n + \log \|A\|)).$$

Proof. By Corollary 7.9, we can compute the product $(AP)M$ in time $O(\mathsf{MM}(n, \log n + \log \|A\|))$ by setting $d = \operatorname{length}(n\|A\|)$. □

Subsection 7.4.1 proves the correctness and complexity bound for the algorithm solving the `Index2Massager` problem seen in Figure 7.2 which, given a nonsingular matrix $B \in \mathbb{Z}^{n\times n}$ that has been massaged by a reduced index-$(0,m)$ 2-massager, computes an index-(m,r) 2-massager for B. Then, Subsection 7.4.2 bounds the complexity of combining a reduced index-$(0,m)$ 2-massager and an index-(m,r) 2-massager into a reduced index-$(0,m+r)$ 2-massager. Finally, Subsection 7.4.3 gives an iterative algorithm which combines $O(\log n)$ index 2-massagers into a 2-massager for the overall input matrix $A \in \mathbb{Z}^{n\times n}$.

7.4.1 Computing an index 2-massager

In this subsection, we show how to use the system solving algorithm supporting Theorem 7.6 and algorithm `TriangularSmithForm` to compute an index-(m,r) 2-massager for an input matrix $B \in \mathbb{Z}^{n\times n}$. The algorithms assume that B is the matrix obtained after applying a reduced index-$(0,m)$ 2-massager to the original input matrix A of the overall problem.

`Index2Massager`(B, n, m, p, X)

Input: Nonsingular $B \in \mathbb{Z}^{n\times n}$, $m \in \mathbb{Z}$ with $0 \leq m < n$, $p \in \mathbb{Z}_{\geq 1}$, and X the smallest power of 2 such that $X \geq n^{1/2}||A||$ for a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$.

Output: P, S, M, r such that (P, S, M) is an index-(m,r) 2-massager for B, with r maximal such that invariant factor $m+r$ of the 2-Smith form of A is $< X^p$. The matrix M will satisfy $M = \mathrm{Rem}(M, X^p)$.

Condition: $B = AP_1M_1{S_1}^{-1}$ where (P_1, M_1, S_1) is a reduced index-$(0,m)$ 2-massager for A.

Figure 7.2: Problem `Index2Massager`

Theorem 7.20. *If $p = \lceil 2n/(n-m) \rceil$, then Problem* `Index2Massager` *in Figure 7.2 can be solved in time*

$$O(\mathsf{MM}(n, \log n + \log \|A\|)(\log n + \log\log \|A\|)).$$

Proof. We describe a 7 step algorithm.

Step 1: Let Q be the permutation `P` from the `LQUP`-decomposition of $\mathrm{Rem}(B,2)^T$. Then, since the first m columns of B have full rank modulo 2, QB can be written in a block decomposition as

$$QB = \left[\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array}\right],$$

where $B_{11} \in \mathbb{Z}^{m\times m}$ is nonsingular modulo 2.

Cost 1: $O(\mathsf{MM}(n,1)\log n)$.

Step 2: Compute

$$\left[\begin{array}{c} C_1 \\ \hline C_2 \end{array}\right] = \mathrm{Rem}\left(\left[\begin{array}{c|c} B_{11} & \\ \hline B_{21} & I_{n-m} \end{array}\right]^{-1}\left[\begin{array}{c} B_{12} \\ \hline B_{22} \end{array}\right], X^p\right)$$

using the algorithm supporting Theorem 7.6. We now have the partial triangularization

$$\mathrm{Rem}\left(\left[\begin{array}{c|c} B_{11} & \\ \hline B_{21} & I_{n-m} \end{array}\right]^{-1}\left[\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array}\right], X^p\right) = \left[\begin{array}{c|c} I_m & C_1 \\ \hline & C_2 \end{array}\right] \tag{7.7}$$

of QB over $\mathbb{Z}/(X^p)$.

Cost 2: $O(\mathsf{MM}(n, \log n + \log\|A\|)\log n)$, by Theorem 7.6

Step 3: Compute

$$U', P' := \texttt{TriangularSmithForm}(C_2, n-m, p\log_2 X)$$

using the algorithm supporting Theorem 7.18. Set $P := \mathrm{diag}(I_m, P')$.

Cost 3: By Theorem 7.18,

$$O(\overbrace{\mathsf{MM}(n-m, p\log X)}^{T_1}\overbrace{(\log(n-m)+\log(p\log X))}^{T_2}).$$

First note that

$$\begin{aligned} T_1 &= \mathsf{MM}\left(\frac{n}{n/(n-m)}, \lceil 2n/(n-m)\rceil \log X\right) \\ &\in O((n/(n-m))^2\,\mathsf{MM}\left(\frac{n}{n/(n-m)}, \log X\right) \\ &\in O(\mathsf{MM}(n, \log X)) \end{aligned}$$

using $\mathcal{H}^{d\le 2}_{\mathsf{MM}}$ and $\mathcal{H}^{2\le n}_{\mathsf{MM}}$ in succession. Next, the logarithmic factor T_2 is simplified using $\log(n-m) \le \log n$ and $\log(p\log X) \in O(\log n + \log\log\|A\|)$.

Step 4: We can now complete the triangularization of (7.7):

$$\mathrm{Rem}\left(\left[\begin{array}{c|c} I_m & \\ \hline & U' \end{array}\right]\left[\begin{array}{c|c} I_m & C_1 \\ \hline & C_2 \end{array}\right]P, X^p\right) = \left[\begin{array}{c|cc} I_m & V & * \\ \hline & E & * \\ \hline & & \end{array}\right].$$

Here is the point where via the $\mathrm{Rem}(\cdot, X^p)$ operation we discover the value of r. Let E be an $r \times r$ triangular Smith form and let V contain the first r columns of C_1P'. In addition, let D be the $r \times r$ diagonal matrix with the same diagonal entries as E, and set $S := \mathrm{diag}(I_m, D, I_{n-m-r})$.

Cost 4: $O(\mathsf{MM}(n-m, p\log X))$.

It remains to compute M. By Lemma 7.15 we may compute M modulo X^p. As shown in (7.3), we can take

$$M = \mathrm{Rem}\left(\left[\begin{array}{c|c|c} I_m & -V(D^{-1}E)^{-1} & \\ \hline & (D^{-1}E)^{-1} & \\ \hline & & I_{n-m-r} \end{array}\right], X^p\right).$$

We will compute this M in the final three steps.

Step 5: First compute $T := \mathrm{Rem}((D^{-1}E)^{-1}, X^p)$.

Cost 5: $O(\mathsf{MM}(n-m, p\log X))$, by Lemmas 7.3 and 7.5.

Step 6: Instead of computing the product $\mathrm{Rem}(-VT, X^p)$ we will proceed as follows. Let B'_{12} be the first r columns of $B_{12}P'$. Compute the product $\mathrm{Rem}(B'_{12}T, X^p)$.

Cost 6: $O(\mathsf{MM}(n, \log n + \log\|A\|))$, by Corollary 7.9.

Step 7: Compute $\mathrm{Rem}(VT, X^p) = \mathrm{Rem}(B_{11}^{-1}(B'_{12}T), X^p)$ using the algorithm supporting Theorem 7.6.

Cost 7: Same as the cost of step 2.

□

7.4.2 Combining index 2-massagers

In this section, we show how to combine an index-$(0, m)$ and an index-(m, r) 2-massager to obtain an index-$(0, m + r)$ 2-massager.

Theorem 7.21. *Given a reduced index-$(0, m)$ 2-massager (P_1, S_1, M_1) for a nonsingular $A \in \mathbb{Z}^{n\times n}$, an index-$(m, r)$ 2-massager (P_2, S_2, M_2) for $\bar{A} := AP_1M_1S_1^{-1}$ with $M_2 = \mathrm{Rem}(M_2, X^p)$ for $p = \lceil 2n/(n-m)\rceil$, and X the smallest power of 2 such that $X \geq n^{1/2}||A||$, we can compute a reduced index $(0, m + r)$ 2-massager (P, S, M) for A in time*

$$O(\mathsf{MM}(n, \log n + \log \|A\|)).$$

Proof. Write M_1 and M_2 using a block decomposition as shown in (7.4) and (7.5). By Theorem 7.17, the only computation required to produce (P, S, M) is to compute $F_1D_1^{-1}W_2$ as shown in (7.6), where $F_1 \in \mathbb{Z}^{m\times m}$ and $V := D_1^{-1}W_2 \in \mathbb{Z}^{m\times r}$. By Lemma 7.15, it will suffice to compute $\mathrm{Rem}(F_1V, X^p)$. For simplicity, and without loss of generality, we will assume that $m = n$ so that F_1 has dimension $n \times n$ and V has dimension $n \times r$.

The multiplication can be realized in $O(\mathsf{MM}(n, \log n + \log \|A\|))$ by Lemma 7.7. □

7.4.3 Computing a reduced 2-massager

In this section, we show how to use the algorithms presented in the previous three subsections to compute a reduced 2-massager.

Theorem 7.22. *Algorithm* `2Massager` *in Figure 7.3 is correct. The running time is*

$$O(\mathsf{MM}(n, \log n + \log \|A\|)(\log n + \log\log \|A\|) \log n).$$

Proof. Correctness of the algorithm follows from the input and output specifications of the subroutines supporting Theorems 7.19, 7.20 and 7.21. By Lemma 7.10, the number of loop iterations is bounded by $\log_2 n$ with the cost of each loop iteration being dominated by the call to `Index2Massager`. The running time estimate is obtained by multiplying the cost estimate of Theorem 7.20 by $\log n$. □

`2Massager`(A, n)

Input: Nonsingular $A \in \mathbb{Z}^{n \times n}$.

Output: (P, S, M), a reduced 2-massager for A.

$X :=$ the smallest power of 2 such that $X \geq n^{1/2}||A||$
$P, S, M, m := I_n, I_n, I_n, 0$
while $m < n$ **do**
  $B := APMS^{-1}$
  $p := \lceil 2n/(n-m) \rceil$
  $P', S', M', r :=$ `Index2Massager`(B, n, m, p, X)
  $P, S, M :=$ `CombineMassager`$(P, S, M, n, m, P', S', M', r, X)$
  $m := m + r$
od
return (P, S, M)

Figure 7.3: Algorithm `2Massager`

7.5 Linear system solving

Suppose $a/b \in \mathbb{Q}$ is a signed fraction with $a \in \mathbb{Z}$, $b \in \mathbb{Z}_{>0}$, $a \perp b$ and $b \perp 2$. Then, rational number reconstruction from von zur Gathen and Gerhard (2013, Section 5.10) can be used to reconstruct a/b from its image $\mathrm{Rem}(a/b, 2^d)$ for large enough d. More precisely, given upper bounds N and D such that $|a| \leq N$ and $b \leq D$, then

$$\texttt{RatRecon}(\mathrm{Rem}(a/b, 2^d), 2^d, N, D)$$

will reconstruct a/b for any d that satisfies $2^d \geq 2ND$. If the first argument to `RatRecon` is a vector then the intent is to apply rational reconstruction elementwise to the entries.

`Solve`(A, b, n)

Input: Nonsingular $A \in \mathbb{Z}^{n\times n}$ and $b \in \mathbb{Z}^{n\times 1}$.

Output: $x, t \in \mathbb{Z}^{n\times 1}, \mathbb{Z}_{\geq 0}$ with t minimal such that all denominators of the entries in $2^t A^{-1}$ are relatively prime to 2, and $x = \mathrm{Rem}(2^t A^{-1} b, 2^d)$ where d is as defined in step 3.

Note: $2^t A^{-1} b =$ `RatRecon`$(x, 2^d, N, D)$.

1. $(P, S, M) :=$ `2Massager`(A, n)
   $t := \log_2 S_{nn}$
2. $U := APMS^{-1}$
3. $N := \lfloor n^{n/2} ||A||^{n-1} ||b|| \rfloor$, $D := \lfloor n^{n/2} ||A||^n / 2^t \rfloor$ and $d := \lceil \log(2ND) \rceil$
   $y := \mathrm{Rem}(U^{-1} b, 2^d)$
4. $x := \mathrm{Rem}(PM(2^t S^{-1})y, 2^d)$
   return x, t

Figure 7.4: Algorithm `Solve`

Our algorithm for system solving is based on the following observation. If (P, M, S) is a 2-massager for A and $U := APMS^{-1}$, then $2^t A^{-1} = PM(2^t S^{-1})U^{-1}$, where $t = \log_2 S_{nn}$.

Theorem 7.23. *Algorithm* `Solve` *in Figure 7.4 is correct. If* $\log ||b|| \in O(n(\log n + \log \|A\|))$ *the running time is*

$$O(\mathsf{MM}(n, \log n + \log \|A\|)(\log n + \log\log \|A\|) \log n).$$

Proof. The correctness of the algorithm follows from the input and output specifications of `2Massager`, `ApplyMassager` and the algorithm supporting Theorem 7.6. Using Hadamard's bound and Cramer's rule, the denominators and numerators of entries of $2^t A^{-1} b$ are bounded by D and N as computed in step 3 of the algorithm. This shows that the note added to the algorithm header also holds.

Now consider the running time. By Theorems 7.22, 7.19 and 7.6, the cost of steps 1, 2 and 3, respectively, are within the target cost.

Finally, consider step 4. Let $z = 2^t S^{-1} y$. The computation of Mz falls under the conditions of Lemma 7.7 and can be done in time $O(\mathsf{MM}(n, \log n + \log \|A\|))$. □

Chapter 8

Conclusion

In this thesis, we have studied algorithms that solve several central problems in exact integer linear algebra by reducing them to matrix multiplication operations with integer matrices that have almost the same size as the input matrix to the overall problem.

Solving the problem of computing the Smith normal form $S \in \mathbb{Z}^{n\times n}$ of a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ along with computing unimodular matrices $U, V \in \mathbb{Z}^{n\times n}$ such that $AV = US$ in time

$$O(n^\omega \mathsf{B}(\log n + \log \|A\|)(\log n)^2)$$

is one of our main contributions. The algorithm we give is a Las Vegas probabilistic algorithm which means that we are able to verify the correctness of its output. In other words, one can keep running the algorithm until the correct output is returned, and the expected running time will still be the claimed one.

Moreover, in the context of the Smith normal form, we define an object that we call the Smith massager. The Smith massager is a matrix $M \in \mathbb{Z}^{n\times n}$ from which many interesting properties follow. The central one is that the Smith form S and Smith massager M, whose ith column is reduced modulo the ith invariant factor s_i, give a nice fraction-free representation of the fractional part of the inverse of A.

The other main contribution of the thesis has been with respect to linear system solving. We present a deterministic reduction to matrix multiplication for the problem of linear system solving: given as input a nonsingular $A \in \mathbb{Z}^{n\times n}$ and $b \in \mathbb{Z}^{n\times 1}$, compute $A^{-1}b$. As an intermediate algorithm we also give a system solving routine which, for a nonsingular $A \in \mathbb{Z}^{n\times n}$ and a $B \in \mathbb{Z}^{n\times m}$, computes the system solution $A^{-1}B$ up to some magnitude X^d where X is a lifting modulus satisfying $X \perp A$.

In addition, we have presented a partial linearization technique which allows us to extend many of our algorithms, so that, their cost estimates depend on the average size of entries of the input matrix and not the largest one. More specifically, the technique transforms an input matrix A into a new matrix D which can be used in place of A for the system solving algorithms, the Smith form and Smith massager algorithm, and many more.

8.1 Hermite normal form

As we already mentioned in Section 4.2, we see a promising approach to compute the Hermite normal form of a nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ by utilizing the property of the Smith massager stated in Theorem 6.10.

Recall that corresponding to every nonsingular matrix $A \in \mathbb{Z}^{n\times n}$ there exists a unique left equivalent matrix (which in the normal case is upper triangular)

$$H = \begin{bmatrix} h_1 & h_{12} & \cdots & h_{1n} \\ & h_2 & \cdots & h_{2n} \\ & & \ddots & \vdots \\ & & & h_n \end{bmatrix} \tag{8.1}$$

with all entries nonnegative, and off-diagonal entries h_{*i} strictly smaller than the diagonal entry h_i in the same column. H is the unique Hermite row basis of A. The product $h_1 h_2 \cdots h_i$ of the first i diagonal entries is equal to the greatest common divisor of all $i \times i$ minors of the first i columns of A, for $1 \le i \le n$. Moreover, $h_1 h_2 \cdots h_n = \det H = |\det A|$.

So, assume that we compute the Smith form S and a Smith massager M for the input matrix A, and that we are looking for some left equivalent canonical form H of A. Theorem 6.10 states that we can find H by solving the equation

$$HM \equiv 0 \,\mathbf{c}\mathrm{mod}\, S$$

for an H that is in the desired form and has minimal determinant (or $\det H = \det S$). In other words, it is only a system solving problem under the condition that the ith column of HM must be a 0 modulo the ith invariant factor s_i. For example, row i of H is the solution to

$$\begin{bmatrix} h_i & h_{i,i+1} & \cdots & h_{i,n} \end{bmatrix} M = \begin{bmatrix} 0 \end{bmatrix} \,\mathbf{c}\mathrm{mod}\, (s_1, \ldots, s_n)$$

while satisfying that $h_{i,k} < h_k$, for $i+1 \le k \le n$, and with minimality of determinant being ensured by the choice of the diagonal entries.

8.2 Computing the Smith normal form of a polynomial matrix

The focus of this thesis has solely been on algorithms that involve integer matrices. We highlight the fact that algebraic algorithms designed over the integers can be naturally extended to work over the polynomials. This extension is particularly useful for the Smith normal form, since a deterministic and fast algorithm for the polynomial version of the problem is basically the last algorithm missing in polynomial matrix algebra.

Let K be a field, and let the input be a nonsingular matrix $A \in \mathsf{K}[x]^{n\times n}$. In addition to the matrix dimension n, it has become customary to give cost estimates in terms of the *generic determinant bound* by Gupta et al. (2012, Section 6):

$$D(A) = \max_{\pi\in S_n} \sum_{1\le i\le n} \deg(A_{i,\pi_i})$$

where S_n is the set of permutation of $\{1, 2, \ldots, n\}$. (If $p = 0$, let $\deg(p) = 0$.) Then $\deg(\det A) \le D(A)$. The generic determinant bound is the equivalent of the permutation bound that we give in Section 3.2. Let $d = \lceil D(A)/n \rceil$, and assume that $d \ge 1$.

A highlight among recent results, following a long line of work, is an algorithm from Jeannerod et al. (2016) that computes the Popov form of A for an arbitrary shift in time $(n^\omega d)^{1+o(1)}$. In addition, Zhou et al. (2014) present an algorithm that computes a representation of the inverse of A in the same time. We aim to compute *deterministically* the Smith form of $A \in \mathsf{K}[x]^{n\times n}$ in time $(n^\omega d)^{1+o(1)}$. Zhou et al. (2014) also give a way to compute the largest invariant factor s_n of A within our target cost deterministically. Can the remaining invariant factors be computed quickly?

We could utilize the usual degree $\times$ dimension compromise, and compute the invariant factors in $O(\log n)$ iterations as in the algorithm from Chapter 5. At each iteration, we can use the compact representation of the inverse to infer the r largest invariant factors. The main challenge is to invent a similar massager construction which will let us extract those r invariant factors from the working matrix deterministically within our target complexity.

The problem seems difficult because of the same reasons that it is difficult over the integers. Even though, in the polynomial case, we can deterministically compute the largest invariant factor s_n, finding a suitable massager, without introducing any randomness, in matrix multiplication time might be out of reach.

References

J. Abbott, M. Bronstein, and T. Mulders. Fast deterministic computation of determinants of dense matrices. In S. Dooley, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'99*, pages 197–204. ACM Press, New York, 1999.

A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.

J. Alman and V. V. Williams. A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 522–539, 2021. doi: 10.1137/1.9781611976465.32.

S. Birmpilis, G. Labahn, and A. Storjohann. Deterministic reduction of integer nonsingular linear system solving to matrix multiplication. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'19*, page 58–65, New York, NY, USA, 2019. ACM. ISBN 9781450360845. doi: 10.1145/3326229.3326263.

S. Birmpilis, G. Labahn, and A. Storjohann. A Las Vegas algorithm for computing the Smith form of a nonsingular integer matrix. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'20*, page 38–45, New York, NY, USA, 2020. ACM. ISBN 9781450371001. doi: 10.1145/3373207.3404022.

S. Birmpilis, G. Labahn, and A. Storjohann. A fast algorithm for computing the Smith normal form with the multiplier matrices for a nonsingular integer matrix. 2021. Submitted to the Journal of Symbolic Computation.

J. Blömer, R. Karp, and E. Welzl. The rank of sparse random matrices over finite fields. *Random Structures and Algorithms*, 10(4):407–419, July 1997. ISSN 1042-9832. doi: 10.1002/(SICI)1098-2418(199707)10:4⟨407::AID-RSA1⟩3.0.CO;2-Y.

G. H. Bradley. Algorithm and bound for the greatest common divisor of n integers. *Communications of the ACM*, 13(7):433–436, July 1970.

G. H. Bradley. Algorithms for Hermite and Smith normal form matrices and linear diophantine equations. *Mathematics of Computation*, 25(116):897–907, October 1971.

J. Bunch and J. Hopcroft. Triangular factorization and inversion by fast matrix multiplication. *Mathematics of Computation*, 28:231–236, 1974.

D. B. Chandler, P. Sin, and Q. Xiang. Incidence modules for symplectic spaces in characteristic two. *Journal of Algebra*, 323(12):3157 – 3181, 2010. ISSN 0021-8693. doi: https://doi.org/10.1016/j.jalgebra.2010.02.038.

Z. Chen and A. Storjohann. A BLAS based C library for exact linear algebra on integer matrices. In M. Kauers, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'05*, pages 92–99. ACM Press, New York, 2005.

C. Cooper. On the distribution of rank of a random matrix over a finite field. *Random Structures and Algorithms*, 17(3-4):197–212, oct 2000. ISSN 1042-9832. doi: 10.1002/1098-2418(200010/12)17:3/4⟨197::AID-RSA2⟩3.0.CO;2-K.

J. D. Dixon. Exact solution of linear equations using p-adic expansions. *Numer. Math.*, 40:137–141, 1982.

J.-G. Dumas, P. Giorgi, and C. Pernet. Dense linear algebra over word-size prime fields: the fflas and ffpack packages. *ACM Trans. on Mathematical Software (TOMS)*, 35(3):1–42, 2008. ISSN 0098-3500. doi: 10.1145/1391989.1391992.

W. Eberly, M. Giesbrecht, and G. Villard. Computing the determinant and Smith form of an integer matrix. In *Proc. 31st Ann. IEEE Symp. Foundations of Computer Science*, pages 675–685, 2000.

J.-C. Faugère and J. Svartz. Gröbner bases of ideals invariant under a commutative group: The non-modular case. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'13*, pages 347–354. ACM Press, New York, 2013.

J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3rd edition, 2013.

M. Giesbrecht. Fast computation of the Smith form of a sparse integer matrix. *Computational Complexity*, 10(1):41–69, 11 2001.

S. Gupta, S. Sarkar, A. Storjohann, and J. Valeriote. Triangular x-basis decompositions and derandomization of linear algebra algorithms over $\mathsf{K}[x]$. *Journal of Symbolic Computation*, 47(4), 2012. doi: 10.1016/j.jsc.2011.09.006. Festschrift for the 60th Birthday of Joachim von zur Gathen.

J. L. Hafner and K. S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM Journal of Computing*, 20(6):1068–1083, December 1991.

D. Harvey and J. Van der Hoeven. Integer multiplication in time O(n log n). *Annals of Mathematics*, 193(2):563 – 617, 2021. ISSN 0021-8693. doi: https://doi:10.4007/annals.2021.193.2.4.

T. C. Hu. *Integer Programming and Network Flows*. Addison-Wesley, Reading, MA, 1969.

E. Hubert and G. Labahn. Computation of invariants of finite abelian groups. *Mathematics of Computation*, 85:3029–3050, 2016.

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

C. S. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of finite abelian groups and the Hermite and Smith normal forms of an integer matrix. *SIAM Journal of Computing*, 18(4):658–669, 1989a.

C. S. Iliopoulos. Worst-case complexity bounds on algorithms for computing the canonical structure of infinite abelian groups and solving systems of linear diophantine equations. *SIAM Journal of Computing*, 18(4):670–678, 1989b.

G. Jäger. Reduction of Smith normal form transformation matrices. *Computing*, 74(4): 377–388–22, 2005.

C.-P. Jeannerod, V. Neiger, É. Schost, and G. Villard. Fast computation of minimal interpolation bases in Popov form for arbitrary shifts. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'16*. ACM Press, New York, 2016.

T. Kailath. *Linear Systems*. Prentice Hall, Englewood Cliffs, N.J., 1980.

E. Kaltofen and G. Villard. On the complexity of computing determinants. *Computational Complexity*, 13(3–4):91–130, 2004.

R. Kannan and A. Bachem. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM Journal of Computing*, 8(4):499–507, November 1979.

F. Le Gall and F. Urrutia. Improved rectangular matrix multiplication using powers of the Coppersmith-Winograd tensor. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1029–1046, 2018. doi: 10.1137/1.9781611975031.67.

J. N. Lyness and P. Keast. Application of the Smith Normal Form to the structure of lattice rules. *SIAM J. Matrix Anal. Appl.*, 16(1):218–231, 1995.

V. Neiger and C. Pernet. Deterministic computation of the characteristic polynomial in the time of matrix multiplication. *Journal of Complexity*, page 101572, 2021. ISSN 0885-064X. doi: https://doi.org/10.1016/j.jco.2021.101572.

M. Newman. *Integral Matrices*. Academic Press, 1972.

M. Newman. The Smith normal form. *Linear Algebra and its Applications*, 254:367—381, 1997.

V. Y. Pan. Computing the determinant and the charactersitic polynomial of a matrix via solving linear systems of equations. *Inf. Proc. Letters*, 28:71–75, 1988.

C. Pauderis and A. Storjohann. Deterministic unimodularity certification. In *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'12*, page 281–288. ACM Press, New York, 2012. ISBN 9781450312691. doi: 10.1145/2442829.2442870.

A. Schönhage. Schnelle Berechnung von Kettenbruchentwicklungen. *Acta Informatica*, 1: 139–144, 1971.

H. J. S. Smith. On systems of linear indeterminate equations and congruences. *Phil. Trans. Roy. Soc. London*, 151:293–326, 1861.

R. Stanley. Smith normal form in combinatorics. *Journal of Combinatorial Theory, Series A*, pages 476–495, 2016.

A. Storjohann. Near optimal algorithms for computing Smith normal forms of integer matrices. In Y. N. Lakshman, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'96*, pages 267–274. ACM Press, New York, 1996.

A. Storjohann. A solution to the extended gcd problem with applications. In W. W. Küchlin, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'97*, pages 109–116. ACM Press, New York, 1997.

A. Storjohann. Computing the Frobenius form of a sparse integer matrix., 2000a. Unpublished note.

A. Storjohann. *Algorithms for Matrix Canonical Forms*. PhD thesis, Swiss Federal Institute of Technology, ETH–Zurich, 2000b.

A. Storjohann. High–order lifting. Extended Abstract. In T. Mora, editor, *Proc. Int'l. Symp. on Symbolic and Algebraic Computation: ISSAC'02*, pages 246–254. ACM Press, New York, 2002.

A. Storjohann. High–order lifting and integrality certification. *Journal of Symbolic Computation*, 36(3–4):613–648, 2003. Extended abstract in Storjohann (2002).

A. Storjohann. The shifted number system for fast linear algebra on integer matrices. *Journal of Complexity*, 21(4):609–650, 2005. Festschrift for the 70th Birthday of Arnold Schönhage.

A. Storjohann. Notes on computing minimal approximant bases. In W. Decker, M. Dewar, E. Kaltofen, and S. Watt, editors, *Challenges in Symbolic Computation Software*, number 06271 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006.

A. Storjohann. On the complexity of inverting integer and polynomial matrices. *Computational Complexity*, 24:777–821, 2015. doi: http://dx.doi.org/10.1007/s00037-015-0106-7.

V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13:354–356, 1969.

Z. Wan. *Computing the Smith Forms of Integer Matrices and Solving Related Problems*. PhD thesis, University of Deleware, 2005.

W. Zhou, G. Labahn, and A. Storjohann. A deterministic algorithm for inverting a polynomial matrix. *Journal of Complexity*, 2014.