Integer Smith Form via the Valence: Experience with Large Sparse Matrices from Homology[1]

J-G. Dumas
Unité Informatique et Distribution, B.P. 53 X, 38041 Grenoble Cedex, France.
Jean-Guillaume.Dumas@imag.fr
www-apache.imag.fr/~jgdumas

B. D. Saunders
Department of Computer and Information Sciences, University of Delaware, Newark, Delaware 19716.
saunders@udel.edu
www.eecis.udel.edu/~saunders

G. Villard
Laboratoire de Modélisation et Calcul, IMAG, BP 53 F, 38041 Grenoble cedex 9, France.
Gilles.Villard@imag.fr
www-lmc.imag.fr/lmc-cf/Gilles.Villard

ABSTRACT

We present a new algorithm to compute the Integer Smith normal form of large sparse matrices. We reduce the computation of the Smith form to independent, and therefore parallel, computations modulo powers of word-size primes. Consequently, the algorithm does not suffer from coefficient growth. We have implemented several variants of this algorithm (Elimination and/or Black-Box techniques) since practical performance depends strongly on the memory available. Our method has proven useful in algebraic topology for the computation of the homology of some large simplicial complexes.

Keywords

Large sparse matrix. Integer Smith form. Gaussian elimination. Black Box. Wiedemann Algorithm. Valence Algorithm. Simplicial Complexes. Homology groups.

1. INTRODUCTION

In this article we study the computation of the integer Smith form of sparse matrices. The classical Smith form algorithm performs an elimination process with some gcd computations over the integers or modulo large primes [15, 27]. There exists also some theoretical advances for sparse matrices with iterative methods [13] but those are not very practical yet.

Our new probabilistic algorithm reduces the Smith form to computations modulo powers of small primes. Consequently the algorithm does not suffer from coefficient growth. Moreover the modular computations are independent of each other, permitting an easy and very effective parallelization. Depending on some space/time tradeoff considerations, we may choose either iterative or direct methods at certain stages of our algorithm. Some salient features of our approach are that (1) we use the ovals of Cassini to often get a better determinant bound than Hadamard's and (2) we begin with the trailing coefficient (valence) of the minimal polynomial of the symmetrized but unpreconditioned matrix AA^t. The method is particularly effective when this polynomial is of low degree and hence fast to compute by Wiedemann's method. This has proven to be the case for many of the boundary map matrices of the simplicial complexes given to us by Volkmar Welker [4, 1]. We report on experiments involving these matrices which arise in the computation of the homology of the complexes. Indeed, the reduced homology of a simplicial complex is equivalent information to the integer Smith form of its boundary maps [24]. In the cases we studied, the boundary maps can be very large, e.g. around 10^5 rows and columns, and very sparse, e.g. 6 entries per row.

We present an overview of our algorithm in §2; we then discuss some implementation details (§3 and §4). Finally we report on experiments using matrices from homology in §5.

[1]Some of this work was done while the first author was visiting the University of Delaware

2. VALENCE BASED SMITH FORM ALGORITHM

In this section we describe a new algorithm for the computation of the Smith form of an integer matrix. This algorithm has proven effective on some of the boundary matrices discussed in this paper, though the asymptotic complexity is not better than Giesbrecht's algorithm [13, Theorem 2.5]. The method is particularly effective when the degree of the minimal polynomial of AA^t is small. We begin with some definitions:

DEFINITIONS 1. *The* **valence** *of a polynomial is its trailing non-zero coefficient. By extension, the* **characteristic valence** *of a matrix is the valence of its characteristic polynomial.*
The **minimal valence** *or simply the* valence *of a matrix is the valence of its minimal polynomial.*
The **valuation** *is the degree of the corresponding term. The* **characteristic** *and* **minimal valuations** *of a matrix are similarly defined.*
For $1 \leq i \leq min(m, n)$*, the* i-th determinantal divisor *of a*

matrix A, $d_i(A)$, is the greatest common divisor of the $i \times i$ minors of A. Let $d_0(A) = 1$.
The i-th invariant factor *of A is $s_i(A) = d_i(A)/d_{i-1}(A)$, or 0 if $d_i(A) = 0$.*
The Smith form *of A is the diagonal matrix*

$$S = diag(s_1(A), \dots, s_{\min(m,n)}(A))$$

of the same shape as A.
For a positive integer q, we denote by $\mathbb{Z}_q$ the quotient ring $\mathbb{Z}/q\mathbb{Z}$. The set of invertible elements in $\mathbb{Z}_q$ is denoted by $\mathbb{Z}_q{}^*$.

It is well known that for $1 \le i \le \min(m,n)$, we have $d_{i-1}|d_i$, $s_{i-1}|s_i$, and that A is unimodularly equivalent to its Smith form.

DEFINITIONS 2. *For a positive integer q, we define* $\text{rank}_q(A)$, *the* rank of A mod q, *to be the greatest i such that q does not divide the i-th invariant factor of A, and denote this rank by $r_q = \text{rank}_q(A)$.*
The rank of A as an integer matrix will be denoted $r = \text{rank}(A)$.

First we present an overview of the valence method. The individual steps can be done in many ways. Afterwards we will discuss the implementation details.

Algorithm: `VSF [Valence-Smith-Form]`

Input: – a matrix $A \in \mathbb{Z}^{n \times m}$. A may be a "black box" meaning that the only requirement is that left and right matrix-vector products may be computed: $x \longrightarrow Ax$ for $x \in \mathbb{Z}^m$, $y \longrightarrow yA$ for $y^t \in \mathbb{Z}^n$.

Output: – $S = \text{diag}(s_1, \dots, s_{\min(n,m)})$, the integer Smith form of A.

(1) [Valence computation]
If $n < m$, let $B = AA^t$, otherwise let $B = A^tA$.
Let $N = \min(n,m)$
Compute the valence v of B. [See section 3]

(2) [Integer factorization]
Factor the valence v.
Let L be the list of primes which divide v.

(3) [Rank and first estimate of Smith form.]
Choose a prime p not in L (i.e. $p \nmid v$).
Compute $r = \text{rank}_p(A)$.
[we now know the first r invariant factors are nonzero and the last $N - r$ *are* 0's.]
Set $S = \text{diag}(s_1, \dots, s_N)$, where $s_i = 1$ for $i \le r$ and $s_i = 0$ for $i > r$.

(4) [Non-trivial invariant factors]
For each prime $p \in L$
Compute $S_p = \text{diag}(s_{p,1}, \dots, s_{p,N})$, the Smith form of A over the local ring $\mathbb{Z}^{(p)}$. [See section 4]
Set $S = S \otimes S_p$, [That is, set $s_i = s_i s_{p,i}$ for those $s_{p,i}$ which are non-trivial powers of p.]

(5) [Return invariants]
Return $S = \text{diag}(s_1 \dots s_N) \in \mathbb{Z}^{n \times m}$.

In order to prove the correctness of our method we will need the following theorem.

THEOREM 1. *Let A be a matrix in $\mathbb{Z}^{n \times m}$. Let $(s_1, \dots, s_r)$ be its non-zero invariant factors. If a prime $p \in \mathbb{Z}$ divides some non-zero s_i, then p^2 divides the characteristic valence of AA^t and p divides the minimal* valence(AA^t). *The same is true of the valences of A^tA as well.*
PROOF. Let $B = AA^t$. The argument will apply equally well to $B = A^tA$. Let $M(x) = \text{minpoly}(B)$ and let $v = \text{valence}(M) = \text{valence}(B)$. Let $C(x) = \text{charpoly}(B)$, and let $F_1(x), \dots F_k(x)$ be the invariant factors of B. It is well known that these are monic integer polynomials with

$$F_1(x)|F_2(x)|\dots|(F_k(x) = \text{minpoly}(B)) \text{ and } C(x) = \prod F_i(x).$$

Let $v_c = \text{valence}(C) = \text{characteristic valence}(B)$. and let $v_i = \text{valence}(F_i)$. It follows easily from the nature of polynomial arithmetic that $v_1|v_2|\dots|v_k = v$, and $v_c = \prod v_i$, Hence all primes that occur in v_c occur in v. Thus the second part of the conclusion follows from the first and it suffices now to show any prime occurring in the Smith form of A occurs squared in v_c :

NOTATIONS 1. *As in [17], we denote by S_i^n the set of all length i subsequences of $[1..n]$ and by A_{IJ}, $I \in S_i^n$, $J \in S_i^n$, the $i \times i$ determinant of the submatrix of A in the rows I and columns J.*

Using the Cauchy-Binet formula [12, Proposition I.§2.14], note that v_c, as a coefficient of the characteristic polynomial, satisfies for some i:

$$\pm v_c = \sum_{I \in S_i^n} B_{II} = \sum_{I \in S_i^n} \sum_{K \in S_i^n} A_{IK} A_{KI}^t = \sum_{I \in S_i^n} \sum_{K \in S_i^n} A_{IK}^2.$$

As a sum of squares, such a coefficient is nonzero if and only if some A_{IK} is nonzero. Hence v_c is the coefficient for the case $i = \text{rank}(A)$. Therefore if p occurs in the Smith form, then $p|gcd(A_{IK})$, the r-th determinantal divisor of A. It follows that $p^2|v_c$. □

It is straightforward to show that this theorem holds when $\mathbb{Z}$ is replaced by any principal ideal domain of characteristic zero (in general replacing transpose by conjugate transpose).

COROLLARY 1. *Algorithm* Valence-Smith-Form *correctly computes the Smith Form.*
PROOF. The theorem shows that we consider the relevant primes. It is evident that the integer Smith form may be composed from the relevant local Smith forms, since the integer Smith form *is* the local Smith form at p up to multiples which are units mod p. □

The remaining sections are devoted to details, variants, and experiments concerning the valence algorithm (section §3 is devoted to details on part 1 of the Valence algorithm and section §4 will focus on part 4).

3. COMPUTING THE VALENCE

The first two steps of the valence algorithm have the purpose of determining a small, at any rate finite, set of primes which includes all primes occurring in the Smith form. In this section we propose two different ways to compute such a set. One method is by computing the valence, the other is a partial integer elimination.

3.1 Chinese remaindering

We compute the integer minimal valence, v, of a matrix B (the valence of its minimal polynomial over the integers) by Chinese remaindering valences of its minimal polynomials mod p for various primes p. The algorithm has three steps. First compute the degree of the minimal polynomial by doing a few trials modulo some primes. Then compute a sharp bound on the size of the valence using this degree. End by Chinese remaindering the valences modulo several primes.

3.1.1 Degree of the minimal polynomial

To compute the degree of the integer minimal polynomial, we choose some primes at random. The degree of the integer minimal polynomial will be the maximal degree of the minimal polynomials mod p with high probability. Primes may have a lower degree minimal polynomial. We call them *"bad" primes*. Let d be the degree of the integer minimal polynomial. There exists a vector u such that the Krylov subspace associated to B and u, $\text{span}(u, Bu, \dots, B^i u)$ is of rank d. Therefore there exists a square $d \times d$ non-zero minor, M_d, of the matrix $[u, Bu, \dots, B^i u]$. "Bad" primes must divide this minor. Given an upper bound on M_d we can then give an upper bound U on the number of bad primes. Let β be an upper bound to the norm of the rows of B. Next, using Hadamard's inequality [31, Theorem 16.6], we can state that $|M_d| \leq U$, where $U = 2^n \beta^{n^2}$. Suppose we choose primes at random from a set P of primes greater than a lower bound l. There can be no more than $\log_l(U)$ primes greater than l dividing M_d. It suffices to pick from an adequately large set P to reduce the probability of choosing bad primes. The distribution of primes assures that adequatly large P can be constructed containing primes that are not excessively large. For instance, we know these bounds on the n-th prime, p_n [21, Theorem A] :

$$n(log(n) + log(log(n)) - 1.002872) \leq p_n \qquad n \geq 2$$

$$p_n \leq n\big(log(n) + log(log(n)) - 1 + 1.8\frac{log(log(n))}{log(n)}\big) \qquad n \geq 13$$

Now, for random primes, each polynomial modulo a prime has a probability at least $1 - \frac{\log_l(U)}{|F|}$ of being of degree d. Suppose that there are k polynomials of maximal degree. The probability that this degree is d is then at least $1 - (\frac{\log_l(U)}{|F|})^k$.

In practice the actual number of distinct primes greater than 2^{15} dividing valences of homology matrices is very small (no more than 50, say) and we often picked primes between 2^{15} and 2^{16} where there are 3030 primes. This giving us only 1.7% of bad primes. With only 10 polynomials this reduces to a probability of failure less than 2×10^{-16}.

The next question is how many primes must be used for the Chinese remaindering. Using Hadamard's inequality again would induce a use of $O(n)$ minimal polynomial computations. We found several methods to reduce this number. In the two next sections we develop two methods for this purpose. First is an early termination of the Chinese remaindering, which is directly useful for sequential computation. Then, for parallel computation, it is interesting to have a sharper estimate. We can use the ovals of Cassini to bound the spectral radius and thence the valence.

3.1.2 Early termination

In our computations for which timings and results are reported here, we compute v_{p_i}, the minimal valence mod p_i, for several primes chosen at random in the vicinity of 2^{16}. We accept the result of Chinese remaindering when the product of the primes p_i exceeds the smaller of the Hadamard bound or a bound computed by considering ovals of Cassini as we will see in section 3.1.3. But when both these bounds are large, we use a probabilistic termination condition. Let v_k be v reduced mod $M = \prod_{i=1}^{k} p_i$, for randomly chosen primes p_i. Suppose at this point that we believe $v_k = v$, that is to say we believe to have sufficiently many primes even if M is lower than our bound (this can happen for instance if this v_k remains the same for successive k). Then it is possible to compute a quick check of this belief in the following manner. Choose another random prime p^* in a sufficiently large set and compute v^*, the minimal valence mod p^*. Compute also $v_k \mod p^* = v_k^*$, then if $v_k = v$, the two values modulo p^* must also be equal. Now there are two cases. On the one hand if those two values are distinct we know that our computation is not finished. On the other hand if the values modulo p^* are equal then v_k is v with high probability ν. We make this probability explicit in the following lemma.

LEMMA 1. *Let $v \in Z$ and an upper bound U be given such that $v < U$. Let P be a set of primes and let $\{p_1 \dots p_k, p^*\}$ be a random subset of P. Let l be a lower bound such that $p^* > l$ and let $M = \prod_{i=1}^{k} p_i$. Let $v_k = v \mod m$, $v^* = v \mod p^*$ and $v_k^* = v_k \mod p^*$ as above. Suppose now that $v_k^* = v^*$. Then $v = v_k$ with probability at least $\nu = 1 - \frac{\log_l(\frac{U}{M})}{|P|}$.*

PROOF. The only way to give an incorrect answer is to have $v \neq v_k$ and at the same time $v_k^* = v^*$. This means that $v_k = v \mod m$ and $v_k = v \mod p^*$ and therefore, as m and p^* are co-prime, p^* must divide $\frac{v - v_k}{M}$. To finish we see that there are at most $\log_l(\frac{v-v_k}{M})$ distinct prime numbers greater than l dividing this quotient and that $v - v_k < U$. $\square$

For instance, the worst example given in section 5.1 is a matrix for which the valence is bounded by $U = 117^{827}$. We choose some primes greater than $l = 2^{15}$. We can suppose that M, the product of primes, is greater than 2, therefore $\log_l(\frac{U}{M}) \leq 379$. On the other hand, we know that there are exactly 3030 primes between 2^{15} and 2^{16}. Therefore by choosing a prime between 2^{15} and 2^{16} we still have more than 87% chance of being right and by using this trick four times this grows to 99.97%. Usually, for the homology matrices the bound is closer to 10^{200}. There, only one application of the trick gives more than 98.5% confidence!

3.1.3 Ovals of Cassini

For a parallel computation of the valence, in particular, a sharp bound is very useful. The Hadamard bound seems too pessimistic an estimation for sparse matrices. Therefore, we use a bound determined by consideration of Gershgörin disks and ovals of Cassini. This bound is of the form β^d where β is a bound on the eigenvalues and d is the degree of the minimal polynomial. Thus n does not appear as an exponent as in the Hadamard bound. To compute the degree we compute several polynomials and take the maximal degree as described in the beginning of this section. To compute a bound on the eigenvalues we use Gershgörin disks and Cassini ovals.

The i-th Gershgörin disk is centered at $a_{i,i}$ and has for a radius the sum of the absolute values of the other entries on the i-th row. Gershgörin's theorem is that all of the eigenvalues are contained in the union of the Gershgörin disks [28]. One can then go further and consider the ovals of Cassini [5, 6], which may produce sharper bounds. For our purposes here it suffices to note that each Cassini oval is a subset of two Gershgörin circles, and that all of the eigenvalues are contained in the union of the ovals. We can then use the following proposition bounding the coefficients of the minimal polynomial :

PROPOSITION 1. *Let $B \in I\!R^{n\times n}$ with its spectral radius bounded by β. Let $M(X) = \text{minpoly}(B) = \sum_{k=0}^{d} m_i X^i$. Then $|\text{valence}(B)| \leq \beta^d$ and $\forall i \in [0..d]$, $|m_i| \leq \max(\sqrt{d\beta}, \beta)^d$.*

PROOF. It suffices to note that $|m_i| \leq \binom{d}{i}\beta^i$ [22, Theorem IV.§4.1]. $\square$

When d is small relative to n this is an improvement on the Hadamard bound since the latter would be of order n rather than d.

This is the case for the Homology matrices in our experiments. Indeed, for those, AA^t has very small minimal polynomial degree and has some other useful properties (e.g. the matrix AA^t is diagonally dominant).

There remains to compute a bound on the spectral radius. We remark that it is expensive to compute any of the bounds mentioned above while staying strictly in the black box model. It seems to require two matrix vector products (with A) to extract each row or column of B. But, if one has access to the elements of A, a bound for the spectral radius of B can easily be obtained with very few arbitrary precision operations :

Algorithm: `OCB [Ovals-of-Cassini-Bound]`

Input: – a matrix $A \in \mathbb{R}^{n\times m}$.

Output: – $\beta_{OC} \in \mathbb{R}$, such that for every eigenvalue λ of AA^t, $|\lambda| \leq \beta_{OC}$.

(1) [Centers]
$\forall i \in [1..n]$, set $q_i = \sum_{j\in[1..m]} a_{ij}^2$.

(2) [Radii]
Form $|A|$, the matrix whose entries are the absolute values of those of A.
Compute $v = |A||A|^t[1, 1, \dots , 1]$.
$\forall i \in [1..n]$, set $r_i = v_i - q_i$.

(3) [Gershgörin bound]
Set $q = \max_{i\in[1..n]} q_i$.
Set i_1 such that $r_{i_1} = \max_{i\in[1..n]} r_i$.
Set i_2 such that $r_{i_2} = \max_{i\in[1..n]\setminus\{i_1\}} r_i$.

(3) [Return Cassini bound]
Return $\beta_{OC} = q + \sqrt{r_{i_1} r_{i_2}}$.

THEOREM 2. *Let $A \in I\!R^{n\times m}$ with Ω non-zero elements. Algorithm* Ovals-of-Cassini-Bound *correctly computes a bound on the eigenvalues of AA^t, using no more than $6\Omega + n$ field operations and $3n$ comparisons.*

PROOF. For the correctness of the bound we use the fact that the eigenvalues lie in the union of the ovals of Cassini. Now suppose that $q_1 \geq q_2$, r_1, r_2 are the two centers and two radii of such an oval. Then any point λ of this oval satisfies the following : $|\lambda - q_1||\lambda - q_2| < r_1 r_2$ [5, Theorem 1]. We want to know the maximal absolute value of such a λ. Set $\delta = q_1 - q_2 > 0$. First if $\lambda > q_1$, set $X = \lambda - q_1$. Then X satisfies $X^2 + \delta X - r_1 r_2 < 0$ and $X > 0$. Hence $|X| < \sqrt{r_1 r_2}$. Next if $q_1 > \lambda > q_2$, then $|\lambda| < q_1$. Last if $q_2 > \lambda$. set $X = \lambda - q_2$. Then X satisfies $X^2 - \delta X - r_1 r_2 < 0$ and $X < 0$. Hence $|X| < \sqrt{r_1 r_2}$. Therefore β_{OC} as in the algorithm matches the requirements. The complexity analysis is straightforward. $\square$

3.2 A Point about the Symmetrization

The choice between A^tA and AA^t is easily made in view of the following.

THEOREM 3. minpoly(A^tA) *and* minpoly(AA^t) *are equal or differ by a factor of x.*

PROOF. Let $m_{A^tA}(x)$ and $m_{AA^t}(x)$ be respectively the minimal polynomials of A^tA and AA^t. The Cayley-Hamilton theorem [12, Theorem IV.§4.2] states that $m_{A^tA}(A^tA) = 0$. Then $Am_{A^tA}(A^tA)A^t = 0$ and $(Xm_{A^tA})(AA^t) = 0$. Since m_{AA^t} is the minimum polynomial of AA^t it follows that $m_{AA^t}|Xm_{A^tA}$. We can similarly prove that $m_{A^tA}|Xm_{AA^t}$. Then either $m_{AA^t} = Xm_{A^tA}$ or $m_{AA^t} = m_{A^tA}$ or $Xm_{AA^t} = m_{A^tA}$. $\square$

Thus the difference of degree has a negligible effect on the run time of the algorithm. It is advantageous to choose the smaller of AA^t and A^tA in the algorithm, to reduce the cost of the inner products involved. Moreover any bound on the coefficients of minpoly(A^tA) can then be applied to those of minpoly(AA^t) and vice versa.

3.3 An Alternative to the Valence

Finally, we mention that, in some experiments, we also have used integer elimination to compute a multiple of the last non-zero invariant factor. Here we have used essentially Storjohan's Bareiss-based probabilistic Smith form algorithm [27], abbreviated. We compute an integer Bareiss-LU decomposition of the matrix storing only some parts of U. The last non-zero row of this matrix, say the $i - th$ row, contains only $i \times i$ minors [2, 19]. Therefore the last non-zero invariant factor divides the gcd of the last non-zero row of U. We were able to succeed with this heuristic on some sparse matrices, where complete execution of Storjohan's algorithm would have required too much memory. Moreover, as this algorithm is deterministic, the gcd computed is a true multiple of the invariant factor.

4. LOCAL SMITH FORM AT P

Next consider the question of computing the local Smith form in $Z^{(p)}$. This is equivalent to a computation of the rank mod p^k for sufficiently many k. Recall that we define the rank mod p^k as the number of nonzero invariant factors mod p^k. We do not mean the McCoy rank, the size of the largest nonzero minor mod p^k. In a number of cases, we have had success with an elimination approach, despite the fill-in problem. We first present this elimination method then the iterative method with lower space requirements.

4.1 Elimination Method

Due to intermediate expression swell, it is not effective to compute directly in $\mathbb{Z}^{(p)}$, the local ring at p, so we perform a computation mod p^e, which determines the ranks mod p^k for $k < e$ and hence the powers of p in the Smith form up to p^{e-1}. Suppose by this means we find that s_r is not zero mod p^e, where r is the previously determined integer rank of A. Then we have determined the Smith form of A locally at p. If, however, the rank mod p^e is less than r, we can repeat the LRE computation with larger exponent e until s_r is nonzero mod p^e.

Algorithm: LRE [Local-Ranks-by-Elimination-mod-p^e]

Input: – a matrix $A \in \mathbb{Z}^{(m+1)\times(n+1)}$, with elements (a_{ij}) for $i, j \in [0..m] \times [0..m]$.
– a prime p, and a positive integer e.

Output: – r_{p^i}, for $1 \le i \le e$.

(1) [Initializations]
set $k = e$
set $r = 0$

(2) [e successive Gauss steps]
for (exponent $k = 1$ to e) do
while $(\exists (s,t) \in [r..m] \times [r..n], p \not| a_{st})$
[a_{st} is the pivot]
Swap rows r, s, and columns r, t
for all $(i,j) \in [r+1..m] \times [r+1..n]$ do
[elimination, with division, mod p^{e-k+1}]
set $a_{i,j} = a_{i,j} - a_{r,j}a_{i,r}/a_{r,r} (\mod p^{e-k+1})$
set $r = r + 1$.
set $r_{p^k} = r$.
[and invariant factors $s_i = p^{k-1}$ for $r_{p^{k-1}} < i \le r_{p^k}$.]
for all $(i,j) \in [r..m] \times [r..n]$ do
set $a_{ij} = a_{ij}/p$

(3) [Return local ranks]
return r_{p^i}, $\forall i \in [1..e]$

THEOREM 4. *For a positive integer e, algorithm* Local Ranks by Elimination mod p^e *is correct and runs using $O(r\ m\ n)$ arithmetic operations mod p^e where r is the rank mod p^e.*

PROOF. It is equivalent to consider the case when the row and column permutations are done in advance so that the pivots are already in the (r, r) position in the while loop. For each k, then, we have an elimination phase determining r_{p^k} followed by a division phase. The elimination may be viewed as multiplication by a unit lower triangular matrix, call it's inverse L_k. The division is multiplication by D_k^{-1}, where $D_k = diag(1, \dots, 1, p, \dots, p)$, with r_{p^k} 1's. Then in effect, A has been factored as $P \prod_{k=1}^{e} L_k D_k A' Q$, where P and Q are permutations, A' is the upper triangular final form of A after the elimination, and the D's and L's are as above. We note that $D_e A' = D_e U$, where U is the unimodular matrix obtained by replacing the last $n - r_{p^e}$ rows of A' by those of the identity matrix. It follows that A is equivalent to $B = \prod_{k=1}^{e} L_k D_k$. From this it is easily seen that the the Smith form of A in $\mathbb{Z}_{p^e}$ is $S = diag(s_i) = \prod_{k=1}^{e} D_k$, because for both B and S and for each j, the leading principal $j \times j$ minor contains the the least power of p, namely $\prod_{i=0}^{j} s_i$. To see this for B, use Cauchy-Binet expansion of the minors together with the lower triangularity of the L_k. □

4.2 Iterative Methods

4.2.1 *Wiedemann's algorithm and diagonal Scaling*

For some matrices the elimination approach just described fails due to excessive memory demand (thrashing). It is desirable to have a memory efficient method for such cases. This section propose three iterative methods. The first one is a quick check that the integer rank is the rank mod p. The second one is to use Wiedemann's algorithm [32] with diagonal scaling [11, Theorem 6.2] to compute the rank mod p. This algorithm has much lower memory requirements than elimination, has better asymptotic complexity and is effective in practice for actually large sparse matrices over large fields [8]. Still it doesn't give the complete local Smith form at p. In 4.2.3 we propose a p-adic way to compute the last invariant factor of this local Smith form at p, completing the algorithm.

4.2.2 *Null Space Method*

Consider a prime p which occurs in the Smith Form of A. We know that p^2 divides the characteristic valence of AA^t. It seems more likely in general that p^2 divides the minimal valence than that it divides two or more successive invariant factors of the characteristic polynomial. Of course one can construct examples to the contrary. For instance consider $A = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$; it has Smith form $\begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$. However $AA^t = A^tA = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ and has minimum polynomial $x - 2$ and characteristic polynomial $(x-2)^2$. At any rate, in the boundary matrix examples from homology that we examined we have never encountered a prime occuring singly in the valence which actually occurs in the Smith form of A.

Thus we take as the next goal after the valence computation in algorithm VSF, to determine for a prime p occuring singly in the valence, if the rank mod p is the integer rank. Moreover we would like to be able to decide this as quickly as possible. This job may be done by computing the rank mod p via Wiedemann's algorithm as is discussed below or via elimination. However the run time of those methods is a function of the rank. We therefore propose here a method with run time a function of the degree of the minimal polynomial of $B = AA^t$. However this method requires arbitrary precision integer arithmetic while the rank mod p approach does not. Despite this, when the rank is large relative to this degree, this method is likely to be less costly.

The idea is to use a factor R of $M = \text{minpoly}(B)$ such that $M = RN$. We would like to know if this factor is repeated in the Frobenius normal form of A, i.e. if the dimension of the kernel of $R(B)$ is the degree of R or a multiple of this degree. We will show in the following lemma that in the case where R and N are coprime, the dimension of the span of $N(B)$ is equal to the dimension of the kernel of $R(B)$. This leads to a probabilistic algorithm: For $d = \deg(R)$, we try $d+1$ random vectors u_i and see if the $v_i = N(B)u_i$ are dependent. Then if the vectors are dependent we know that with high probability the dimension of the kernel of $R(B)$ is d and that R is not repeated. It follows that any prime occuring singly in valence(R) and valence(M) will occur also singly in the characteristic valence of AA^T. And then such a prime cannot appear in the Smith form of A.

We now give the complete algorithm, then the dimension lemma and end this section with the probabilistic analysis.

Algorithm: `NSD [Null-Space-Dimension]`

Input: – a matrix $A \in \mathbb{Z}^{n\times m}$, A may be a BlackBox.
– the minimal polynomial M of AA^t, and a factor R of M.
– $\epsilon \in \mathbb{R}$ such that $0 < \epsilon < 1$.

Output: – a list $\overline{L}$ of all the primes in valence(R) which do not occur in the Smith form of A. The list is correct with probability at least $1-\epsilon$.

(1) [Initializations]
Set $\overline{L} = \emptyset$
Set $d = \deg(R)$.
Set $N = \frac{M}{R}$
Set $g = \lceil \epsilon^{-\frac{1}{d}} \rceil$.
Set p_0 a random prime such that $p_0 > g$ and $p_0 \nmid$ valence(M).
Form the BlackBox $N(B)$.

(2) [Probabilistic Null-Space dimension]
Select $d+1$ random nonzero vectors $u_i \in (\mathbb{Z}_{p_0})^n$.
$\forall i \in [0..d]$, set $v_i = N(B)u_i$.
if $rank([\, v_0\ v_1\ \cdots\ v_d\,]) < d+1$ modulo p_0.
for each prime p dividing valence(R)
if p^2 does not divide valence(M), add p to $\overline{L}$.

(3) [Return not occuring primes]
return $\overline{L}$.

Of course this algorithm can be applied to any factor of M to determine primes which can be removed from the candidate list L of primes which divides valence(M). To prove its correctness we need the following lemma.

Lemma 2. *Let $B \in \mathbb{Z}^{n\times n}$. Let $N, R \in \mathbb{Z}[X]$ coprime such that $N(B)R(B) = 0 \in \mathbb{Z}^{n\times n}$. Then* $\text{span}(R(B)) = \ker(N(B))$ *and* $\text{span}(N(B)) = \ker(R(B))$.

Proof. First, since $R(B)N(B) = N(B)R(B) = 0$ the span of one matrix polynomial is included in the kernel of the other one. Now, using [12, Theorem VII.§2.1], we know that $\ker(R(B))$ and $\ker(N(B))$ are supplementary. We conclude by the dimension theorem, proving the lemma. ☐

Theorem 5. *Algorithm* Null-Space-Dimension *is correct.*

Proof. Let $B = AA^t$. B is symmetric, so its minimal polynomial P is squarefree. We now suppose that this minimum polynomial is not irreducible. Let N and R be two cofactors of P. As P is squarefree, N and R are coprime. By the lemma, the span of $N(B)$ is the nullspace of $R(B)$. Thus the nullspace of $R(B)$ has dimension kd, where k is the multiplicity of R in the characteristic polynomial of B. Hence algorithm NSD may give incorrect result only if it's $d+1$ random vectors are preimages of $d+1$ dependent vectors in a space of dimension kd over a field with more than g scalars. Note that the v_i are uniformly distributed in span$(N(B))$ if the u_i are uniformly distributed in $(\mathbb{Z}_{p_0})^n$.

We now quantify the probability of such a dependence. Let $P(j,n)$ be the probability of a dependency among j random vectors in a space of dimension n ($j \le n$) over the field $\mathbb{Z}_{p_0}$ with h elements. Then $P(j,n) = P(j-1,n) + P($ first $j-1$ independent but j-th dependent$)$ which gives $P(j,n) \le P(j-1,n) + \frac{h^{j-1}}{h^n} \le \frac{h^j - 1}{(h-1)h^n} < (h-1)^{j-1-n}$. Hence, as $h > g$, if $k \ge 2$ then $P(d+1, kd) \le g^{d-kd} \le \frac{1}{g^d} \le \epsilon$. ☐

4.2.3 The Last Invariant Factor of the Local Smith Form at p

We have not entirely worked out an extension of Wiedemann's approach suitable for computation of the rank mod a power p^e. The method of Reeds and Sloan [25] can be adapted to compute the annihilator of our matrix $\mathbb{Z}_{p^e}$. It may be possible to adapt this to the purpose of computing the rank of the matrix in $\mathbb{Z}_{p^e}$. We do not currently know how to do this in a memory efficient way.

In practice we have encountered matrices whose invariant factors are square free. To verify this it suffices to show that the exponent of p is 1 in the last invariant factor (last nonzero Smith form entry). The following method will do this and a little more.

Let A be a matrix of rank r in $\mathbb{Z}^{m\times n}$ whose local Smith normal form at p is $S_p = \text{diag}\,(p^{k_1}, p^{k_2}, \ldots, p^{k_r}, 0, \ldots, 0)$. The problem is to compute the multiplicity $\kappa = k_r$ of p in the last nonzero invariant factor. Since determining whether κ is zero reduces to comparing the rank modulo p and the rank over $\mathbb{Q}$, we assume that $\kappa \ge 1$.

Our purpose is to derive a black-box algorithm with cost linear in κ rather than in $\Sigma_i k_i$, say.

• **Invertible matrix case.** We assume for the moment that A is $n \times n$ invertible. For a vector x of reduced integer fractions, we define the order $\text{ord}_p(x)$ of x as the largest exponent of p in the denominators of the entries of x. For a random b the solution x to $Ax = b$ satisfies in general $\text{ord}_p(x) = \kappa$. To reduce the cost of computing κ, let us ensure the same property for the order of the first entry of a well chosen system solution. Let v be a nonzero $n \times 1$ vector with first nonzero entry v_I, $1 \le I \le n$, and let u be the first canonical vector. Define the $n \times n$ matrix $E(v)$ by:

$$\begin{cases} E_{i+1,i} = 1 \text{ for } 1 \le i < I \ \text{ and } \ E_{i,i} = 1 \text{ for } I < i \le n, \\ E_{i,j} = 0, \text{ otherwise.} \end{cases}$$

Lemma 3. *Let b and v be two random integer vectors with entries chosen uniformly in $[0, p-1]$. With probability $(1 - 1/p)^2$, $E(v) + uv^t$ is invertible ($v \neq 0$) and κ is the order of the first entry of the solution y to $A(E(v) + uv^t)^{-1}y = b$. When $v \neq 0$, the order cannot be strictly greater than κ.*

Proof. If $E(v) - uv^t$ is invertible and x denotes the solution to $Ax = b$ then $y = (E(v) + uv^t)x$ and the first entry of y is $y_1 = \Sigma_i v_i x_i$. Let U and V be unimodular transformations such that $S = UAV$ is in Smith form, U and V define bijections on $[0, p-1]^n$ and on $(\mathbb{Z}/p\mathbb{Z})^n$ thus:

$$\begin{aligned}\mathcal{P}_b &= \mathcal{P}rob_b(\mathrm{ord}_p(x) = \kappa; Ax = b)\\ &= \mathcal{P}rob_b(\mathrm{ord}_p(x) = \kappa; S(V^{-1}x) = Ub)\\ &= \mathcal{P}rob_b(\mathrm{ord}_p(z) = \kappa; Sz = b)\\ &\geq \mathcal{P}rob_b(1 \leq b_n \leq p-1) = 1 - 1/p.\end{aligned}$$

If $\mathrm{ord}_p(x) = \kappa$ then $p^\kappa x \not\equiv 0 \mod p$. With probability $1 - 1/p$ we have also $p^\kappa \sum_i v_i x_i \not\equiv 0 \mod p$ and the order of y_1 is κ. In this latter case v is nonzero and by construction $E(v) + uv^t$ is invertible. This proves the first assertion of the lemma. For the last assertion it suffices to notice that since the v_i's are integers, the order of $\Sigma_i v_i x_i$ is lower than the order of x. □

Let us now associate to A a new matrix $\tilde{A}$ whose Smith form is related to the one of A and such that one entry of the solution to $\tilde{A}x = b$ can be computed fast:

LEMMA 4. *Let $q \neq p$ be a prime, for $\tau \geq 1$ integer define $\tilde{A} = p^\tau + qA$. Locally at p, the nonzero invariant factors of A and $\tilde{A}$ having exponents strictly lower than τ are the same.*

PROOF. The matrices A and $\tilde{A}$ as matrices in $(\mathbb{Z}/(p^\tau))^{n\times n}$ are equivalent. □

LEMMA 5. *Let b be an integer vector with $||b||_\infty \leq ||A||_\infty = \beta$. Any entry of the solution x to $\tilde{A}x = b$ may be computed in $N = n\left(\log_2 n + 2\log_2(p^\tau + q\beta)\right)/(\log_2 q) + 2$ multiplications of $\tilde{A}$ times a vector with entries bounded by q and $O\left(n^2(\log_2 n + 2\log_2(p^\tau + q\beta))^2\right)$ additional binary operations. In addition to the matrix storage, the algorithm requires an $\mathcal{O}(n\left(\log_2 n + 2\log_2(p^\tau + q\beta)\right))$ bits of storage.*

PROOF. We apply the algorithm of Dixon [7] based on a q-adic expansion of the solution. The matrix $\tilde{A}$ is invertible in $\mathbb{Q}^{n\times n}$ and in $(\mathbb{Z}/p\mathbb{Z})^{n\times n}$: $\tilde{A}^{-1} \equiv p^{-\tau} Id \mod q$. The number N of iterations is given in the lemma, each iteration has binary cost $\mathcal{O}(N \log_2^2 q)$ when the q-adic expansion of only one entry of the solution is computed. The rational value of that entry is constructed from the expansion within the same cost. The extra amount of storage needed is $\mathcal{O}(N \log_2 q)$. □

For any integer τ, from lemma 3 and lemma 4 we may give a randomized algorithm which compares τ to κ with an arbitrary error tolerance $\epsilon > 0$: By repeated random choices of q, b, v, it is possible to produce an algorithm returning, with probability as low as required, κ if $\kappa \leq \tau$ (maybe wrong) and reporting that κ is strictly greater than τ otherwise (always correct).

Algorithm: `LIF [Largest Invariant Factor] order`

Input: – $A \in \mathbb{Z}^{n\times n}$ invertible,
– a prime p, an integer $\tau \geq 1$,
– an error tolerance $\epsilon > 0$.

Output: – with probability at least $1-\epsilon$, returns κ if $\kappa \leq \tau$ (maybe wrong) and reports that κ is strictly greater than τ otherwise (always correct).

(1) [Conditioning]
Set max_order = 0.
Choose a prime $q \neq p$.
Build $\tilde{A} = p^{\tau+1} + qA$.

(2) [Order]
For t from 1 to $\lceil \log(\epsilon)/\log(1-(1-1/p)^2) \rceil$ do
  Choose random b and v in $[0, p-1]^n$.
  If $v \neq 0$ construct $E(v) + uv^t$
    Compute $y_1 \in \mathbb{Q}$ the first entry of y such that $\tilde{A}(E(v) + uv^t)^{-1} y = b$.
    If $\mathrm{ord}_p(y_1) > \tau$ then Return("$\kappa >$" τ).
    max_order := max{max_order, $\mathrm{ord}_p(y_1)$}.
Return("$\kappa =$" max_order).

PROPOSITION 2. *The algorithm LIF works as specified, if $\kappa \leq \tau$ then it returns κ with probability at least $1-\epsilon$ or a lower value with probability less than ϵ. If $\kappa > \tau$ then it discovers the latter inequality with probability at least $1-\epsilon$, this result is always correct, or returns a wrong value lower than τ with probability less than ϵ. The cost of the algorithm is bounded by the cost in lemma 5 times the number of passing through the* for loop *(a function of the error tolerance and of p).*

PROOF. If $\kappa \leq \tau$ then by lemma 4 all the invariant factors of $\tilde{A} = p^{\tau+1} + qA$ are also of order less than κ. By lemma 3, the probability that $v = 0$ or that y_1 gives an order strictly less than κ is $1-(1-1/p)^2$. The computed order cannot be strictly greater than κ. After $\lceil \log(\epsilon)/\log(1-(1-1/p)^2) \rceil$ trials, the probability to have a wrong result is less than ϵ. If $\kappa > \tau$, the largest invariant factor of $\tilde{A}$ is at least of order $\tau + 1$, in the same way, this will be found with probability at least $1-\epsilon$. This latter result is always correct since by lemma 3 the computed order cannot be greater than the actual one. The cost bounds are immediately derived (multiplying by the inverse of $E(v) + uv^t$ requires linear time only). □

One may see for instance that with 24 random choices for b and v, it is possible with probability more than $1 - 10^{-6}$ independently of the dimension of the matrix, to certify that $\kappa \geq 2$ for $p = 3$. By constructing orders of number fields, as done in [14, §5] for Diophantine equations, the number of trials could be reduced up to some increase in time for the applications of lemma 5.

• **General case.** When A is $m \times n$ singular of rank r, unlike in lemma 4, the use of the modified matrix $\tilde{A}$ will always introduce new nonzero invariant factors that prevent from computing κ using algorithm LIF. An alternative way to proceed is to apply the algorithm to an invertible $r \times r$ matrix A_S constructed from A, whose local Smith form is $S_I = \mathrm{diag}\,(p^{k_1}, p^{k_2}, \ldots, p^{k_r})$. Such a matrix A_S may be obtained by conditioning A. We generalize the construction of [23, Lemma 11]. Let us denote by U and V two unimodular multipliers such that UAV is in Smith form: $UAV = \left[\begin{smallmatrix} S_I & 0 \\ 0 & 0\end{smallmatrix}\right]$ and consider two pre-conditioners $P \in \mathbb{Z}^{r\times m}$ and $Q \in \mathbb{Z}^{n\times r}$. If $T \in \mathbb{Z}^{m\times r}$ is the matrix constructed with the first r columns of U^{-1} and if $W \in \mathbb{Z}^{r\times n}$ is constructed with the first r rows of V^{-1} then: $A = TS_IW$ thus $PAQ = (PT)S_I(WQ)$. We see that if PT and WQ are invertible modulo p, we can take $A_S = PAQ$. Since U and V are unimodular, T and W have rank r modulo p and the condition depends on the choice of P and Q. We have proven:

LEMMA 6. *There exists two matrices $W \in \mathbb{Z}^{r\times n}$, $T \in \mathbb{Z}^{m\times r}$ only depending on A giving : If $P \in \mathbb{Z}^{r\times m}$ and $Q \in \mathbb{Z}^{n\times r}$ are such that $p \not| \det(PT)$ and $p \not| \det(WQ)$ then $A_S = PAQ$ is invertible and its local Smith form at p is the invertible submatrix $S_I = diag\ (p^{k_1}, p^{k_2}, \ldots, p^{k_r})$ of the local Smith form of A.*

For pre-conditioners satisfying the lemma condition one may consider Toeplitz matrices as used in [18, Theorem 2] or in [14, §5]. Define $\eta = \lceil \log_2 2r(r+1) \rceil$ and let $\Gamma \in \mathbb{Z}[x]$ be a monic polynomial irreducible modulo p, of degree η and with coefficients bounded in absolute value by p. Let $\theta \equiv x \mod \Gamma$ and $\mathcal{V}_p$ be the set of the polynomials of degree less than η with coefficients 0 or 1 over $\mathbb{Z}[\theta]$.

PROPOSITION 3. *[14] Let $A \in \mathbb{Z}^{m\times n}$ be of rank r, p a prime. Consider as pre-conditioners $\mathcal{P}$ and $\mathcal{Q}$, an upper and a lower triangular Toeplitz matrix with unit diagonal and whose other entries are randomly chosen in $\mathcal{V}_p$. The local Smith form of $A_S = \mathcal{P}A\mathcal{Q}$ at p is S_I with probability greater than 1/2.*

The algorithm LIF may then be applied to A_S or the random choice of the conditioners may be included in the *for loop* of a modified algorithm. In that case [14], if $y_1 = \sum_{0\le l<\eta} y_i^{(l)}\theta^l \in \mathbb{Q}(\theta)$ is the first entry of the solution to $\tilde{A}_S(E(v)+uv^t)y = b$, one look at the order of $y_1^{(0)}$. The sizes of the integers (numerators and denominators) involved in the computations are a factor $\mathcal{O}(\eta)$ *i.e.* $\mathcal{O}(\log r)$ greater than for the regular case. The cost bounds of lemma 5 will be modified by the same factor.

Instead of Toeplitz matrices one may use sparse matrices as in [32, §III] or [30] to avoid order fields.

PROPOSITION 4. *[32, Theorem 1'] Let $A \in \mathbb{Z}^{m\times n}$ be of rank r, p a prime. A random procedure exists for generating sparse pre-conditioners $\mathcal{P}$ and $\mathcal{Q}$ with coefficients in $\{0,1\}$ and of total Hamming weight $\mathcal{O}(r\log r)$. With probability at least $1-\epsilon$ they satisfies that the local Smith form of $A_S = \mathcal{P}A\mathcal{Q}$ at p is S_I.*

In any case, one can include the preconditionning in the preceding process; thus only increasing the computation of the matrix-vector products by a factor $\mathcal{O}(log(r)$.

5. EXPERIMENTS WITH HOMOLOGY MATRICES

In this section we describe the structure of the boundary maps of a simplicial complex. More details on the connection between homology groups of simplicial complexes and linear algebra can be found in [24]. We will mainly talk about three homology matrix classes. The matrices will be denoted by three naming patterns:

- **mk**i.**b**j denotes the boundary matrix j from the matching k-complex with i vertices.
- **ch**i-k.**b**j denotes the boundary matrix j from the i by k chessboard complex.
- **n**i**c**k.**b**j denotes the boundary matrix j from the not i-connected graph with k vertices.

For details on those simplicial complexes see [1, 3].

The boundary matrices are sparse matrices with a fixed number k of non-zero elements per row and l per column. If A_i is the boundary map between the dimensions i and $i-1$ of a simplicial complex then $k = i+1$. All entries are -1, 0 or 1. Moreover, the Laplacians $A_iA_i^t$ and $A_i^tA_i$ also have -1, 0 and 1 as entries except for the diagonal which is respectively all k and all l. However, as expected, those Laplacians have more than twice as many non-zero elements as A_i. Thus, we did not perform the matrix multiplications to compute $A_iA_i^tv$. We performed two matrix-vector products, $A_i(A_i^tv)$, instead.

We will also check in section 5.1 that the Laplacians have indeed a very low degree minimal polynomial (say up to 25 for the matching and chessboard matrices, close to 200 for the not-connected). This fact was our chief motivation to develop the Valence method. Figure 1 illustrates the patterns occurring in these matrices. It shows ch4-4.b2, the boundary map between the second and first dimensions of the 4×4 chessboard complex, together with its Laplacians. It is of size 96×72 with 288 non-zero elements. It has 3 elements per row and 4 per column. On the other hand AA^t is of size 96×96 with 960 non-zero elements and A^tA is of size 72×72 with 648 non-zero elements.

Our experiments were realized on a cluster of 20 Sun Microsystem Ultra Enterprise 450 each with four 250MHz Ultra-II processor and 1024 Mb or 512 Mb of memory. We computed ranks of matrices over finite fields $GF(q)$ where q has half word size. The chosen arithmetic used discrete logarithms with precomputed tables, as in [26]. The algorithms were implemented in C++ with the LINBOX[1] library for computer algebra and the ATHAPASCAN[2] environment for parallelism.

5.1 Experiments

We will call ω the number of non-zero elements per row, s the shape and r the integer rank of the matrix under consideration. We will produce these elements only in table 1. The name of the matrix will be repeated in the following tables. For several cases, fill-in causes a failure of elimination. This is due to memory thrashing (MT). All the timings presented are in seconds except as otherwise specified and reflect the cpu time for sequential computations and the real time for parallel computations.

In table 1 we present computations of the integer minimum polynomial of some homology matrices. We indicate the Cassini bound, CB, for those and present the number of Chinese remainders needed, Rem, the degree of the minimum polynomial, d_m and the minimal valence, v_m. As some v_m are quite large, we write explicitly only the smaller prime factors and denote by C_i a product of i larger prime factors. In one case, C_{59+} denotes a product of at least 59 primes. This number has 57 known prime factors and a another composite factor of 376 digits that we were unable to factor. We give timings for sequential and parallel computation of the minimal valence. A first approach for parallel computation is to use sequential routines for Cassini bounds,

[1] Symbolic linear algebra library, http://www.cis.udel.edu/~caviness/linbox

[2] Parallel execution support of the APACHE Project, http://www-apache.imag.fr/software

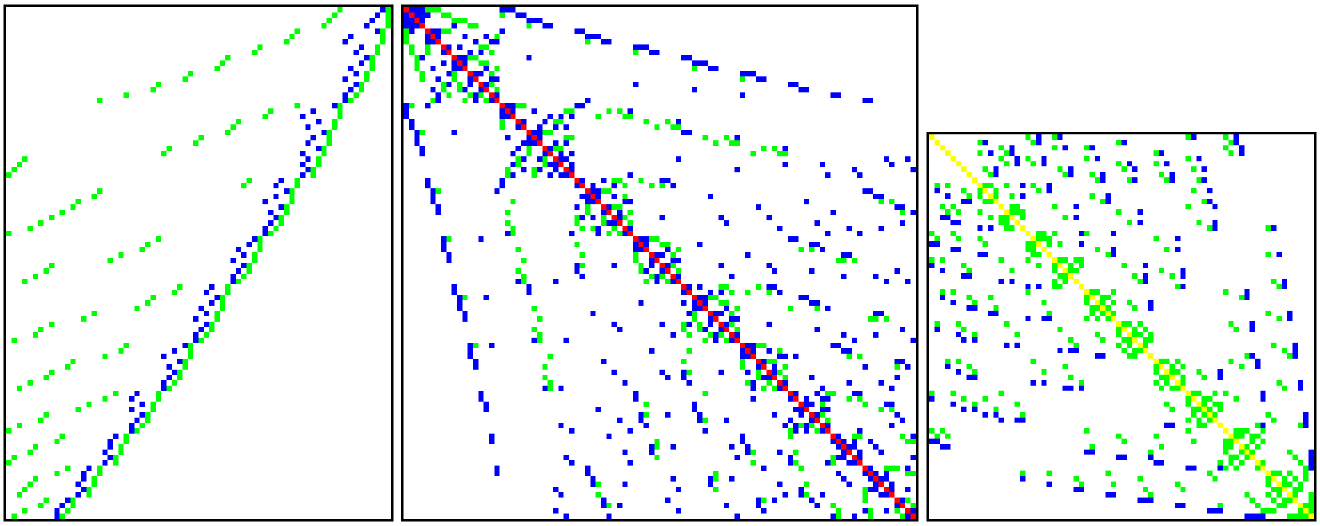

Figure 1: Chessboard complex 4-4, boundary matrix 2 with AA^t and A^tA.

Matrix	ω, s, r	CB	Rem	d_m	v_m	Sequential	Parallel
mk9.b3	4, 945x1260, 875	12	2	X^6	$2^6.3^4$	0.31	0.39
mk10.b3	4,4725x3150,2564,	24	3	X^7	$2.3^4.5^3.7.13$	1.16	0.81
mk11.b4	5, 9450x17325, 9324	14	3	X^9	$-2^5.3^3.5^2.7.11$	4.00	1.80
mk12.b3	4, 51975x13860, 12440	60	3	X^7	$2^{10}.3^5.5.7.11.13$	16.68	5.55
mk12.b4	5, 62370x51975, 39535	30	4	X^{11}	$-2^{13}.3^7.5.7.11$	34.49	11.39
mk13.b4	5, 270270x135135, 111463	50	4	X^{11}	$-23.2^{11}.3^7.5^2.7.11.13$	149.06	42.13
mk13.b5	6, 135135x270270, 134211	18	4	X^{12}	$2^{10}.3^5.5^2.11.13$	96.11	29.41
ch4-4.b2	3, 96x72, 57	12	1	X^4	$2^7.3$	0.08	0.24
ch5-5.b3	4, 600x600, 424	16	3	X^8	$2^5.3^3.5^2.7$	0.35	0.39
ch6-6.b4	5, 4320x5400, 3390	20	3	X^{10}	$2^{10}.3^5.5.11$	1.83	1.43
ch7-7.b4	5, 52920x29400, 22884	45	6	X^{15}	$-19.17.2^{11}.3^7.5^3.7^3.11.13$	45.17	10.28
ch7-7.b5	6, 35280x52920, 29448	24	4	X^{12}	$-29.41.367.34504396919539$	23.88	7.52
ch7-8.b5	6,141120x141120, 92959	36	7	X^{21}	$19.17.2^18.3^9.5^4.7^3.11.13$	123.18	24.85
ch7-9.b3	4, 105840x17640, 16190	96	7	X^{16}	$29.19.2^{19}.3^{10}.5^5.7^4.11.13.31$	82.57	15.04
ch7-9.b4	5, 317520x105840, 89650	75	9	X^{22}	$-1433.C_4$	471.22	64.67
ch7-9.b5	6, 423360x317520, 227870	48	9	X^{24}	$23.19.17.2^{24}.3^{10}.5^4.7^4.11^2.13^2$	775.33	106.15
ch8-8.b4	5, 376320x117600, 100289	80	7	X^{16}	$29.19.17.2^{22}.3^8.5^4.7^2.11.13$	386.13	64.78
ch8-8.b5	6, 564480x376320, 276031	54	7	X^{19}	$-19.17.2^{21}.3^9.5^4.7^2.11^2.13$	732.44	127.57
ch8-8.b6	7, 322560x564480, 279237	28	5	X^{14}	$2^{15}.3^4.5^3.7.11.13$	337.72	77.52
ch8-8.b7	8, 40320x322560, 40320	8	1	X^1	-2^3	10.63	10.47
ch9-9.b3	4, 381024x42336, 39824	144	5	X^{10}	$-73.6287.C_2$	190.74	45.10
ch9-9.b4	5, 1905120x381024, ??	125	8	X^{17}	$-2^{13}.3^{12}.5^6.7^3.13^2.C_5$	2279.32	358.97
n2c6.b6	7, 5715x4945, 2943	63	25	X^{66}	$17.2^{26}.3^{21}.5^2.7.13^2.C_{10}$	55.43	3.44
n2c6.b7	8, 3990x5715, 2772	64	58	X^{152}	$17^2.2^{83}.3^{55}.5^{50}.7^3.13^2.C_{11}$	206.18	7.51
n3c6.b9	10, 2511x4935, 1896	60	15	X^{40}	$17^2.2^9.3^{13}.5^8.13^4.7.C_6$	15.80	1.83
n4c5.b5	6, 4340x2852, 1866	60	39	X^{103}	$-2^{16}.3^{10}.5^{16}.13^2.191^2.89^2.C_{21}$	82.76	4.04
n4c5.b6	7, 4735x4340, 2474	63	82	X^{216}	$19.17.2^{45}.3^{10}.5^{26}.7.67^2.59^2.C_{34}$	401.05	11.49
n4c5.b7	8, 3635x4735, 2261	64	100	X^{263}	$-17.2^{47}.3^{13}.5^{28}.11.13^2.59^2.C_{36}$	511.69	14.26
n4c5.b8	9, 1895x3635, 1374	63	62	X^{164}	$17^2.2^{35}.3^{25}.5^{23}.11.13^2.379^2.C_{21}$	125.93	5.30
n4c6.b5	6, 51813x20058, 15228	96	44	X^{104}	$-23.2.52009.C_3$	1161.39	35.35
n4c6.b12	13, 25605x69235, 20165	117	358	X^{827}	$-2^{243}.3^{43}.5^{36}.7^7.11^4.13^2.31^2.C_{59+}$	66344.40	1181.61
n4c6.b13	14, 6300x25605, 5440	98	37	X^{87}	$-2^{31}.3^8.5^8.7^2.11^2.13.31^2.C_{18}$	249.28	9.90

Table 1: Computing the Integer Minimal Polynomial of AA^t via Chinese remaindering

minimum polynomial over Z_p, and Chinese remaindering of integers. The algorithm has 3 steps. First compute the Cassini bound and some minimum polynomials in parallel. Using the bound and the degree of the minimum polynomials, the maximum number of remainders needed is known. Therefore, the second step is the computation in parallel of some more minimum polynomials as required [9]. The last step is a Chinese remaindering of the coefficients. A future implementation will also use parallel matrix-vector products as well as block methods [16, 29] to improve speed.

Matrix	Elimination	Wiedemann
mk9.b3	0.26	2.11
ch7-7.b6	4.67	119.53
ch7-6.b4	49.32	416.97
ch7-7.b5	2179.62	4283.4
ch8-8.b5	MT	55 hours
n2c6.b6	6.44	72.96
n2c6.b7	3.64	57.10
n4c5.b6	2.73	51.75
n4c6.b12	231.34	4131.06
n4c6.b13	8.92	288.57

Table 2: rank modulo 65521, Elimination vs. Wiedemann

In table 2 we report some comparisons between Wiedemann's algorithm and elimination with reordering for computing the rank. More sequential comparisons and some parallel ones are reported in [8]. We just want to emphasize the fact that for these matrices from homology, as long as enough memory is available, elimination is more efficient. However, for larger matrices, Wiedemann's algorithm is competitive and is sometimes the only solution.

In table 3 we compare timings of our algorithm to some implementations of other methods. We compare here only the results obtained using the version of the Valence Smith Form algorithm in which we use Wiedemann's algorithm to compute the Valence and then elimination modulo small powers of primes p to compute the invariant factors locally at p. Simplicial Homology [10] is a proposed GAP share package. It computes homology groups of simplicial complexes via the Smith form of their boundary maps. It features a version of our Valence algorithm as well as an elimination method for homology groups by Frank Heckenbach . The latter is a variant of the classical elimination method over arbitrary precision integers for Smith form [15], taking advantage of the particular structures of the boundary maps. The entry "Hom-Elim-GMP" in this table refers to this elimination-based method using Gnu Multi Precision integers. Fermat [20] is computer algebra system for Macs and Windows. Its Smith form routine is an implementation of Kannan & Bachem's [15] as is Heckenbach's but with a different pivot strategy.

"Hom-Elim-GMP" and "Valence" ran on the 250MHz Ultra-II processor with 1024 Mb, but Fermat is only available on Mac and Windows. We therefore report on experiments with Fermat on a 400 MHz Intel i860 processor with only 320 Mb. First we see that "Fermat" cannot compete with "Hom-Elim-GMP" in any case. The main explanation is that the pivot strategy used by "Hom-Elim-GMP" is very well suited to the homology matrices. We can see also that as long as

Matrix	Fermat	Hom-Elim-GMP	Valence
ch6-6.b4	1138.04	3.48	41.51
mk9.b3	39.49	0.34	1.27
mk10.b3	33.24	1.60	20.23
mk11.b4	98937.27	4269.14	210.24
mk12.b3	3900	46.10	491.37
mk12.b4	MT	MT	31688.20

Table 3: Fermat vs. Hom-Elim-GMP vs. SFV

no coefficient growth is involved "Hom-Elim-GMP" is often better than "Valence". Indeed, where "Hom-Elim-GMP" performs only one integer elimination, "Valence" performs an elimination for every prime involved - of course in parallel this difference will disappear. But as soon as coefficient growth becomes important "Valence" is winning. Moreover, "Valence" using only memory efficient iterative methods can give some partial results where memory exhaustion due to fill-in prevents any eliminations from running to completion. In table 2 we can see some of these effects.

6. CONCLUSION

The preceding comparison of two elimination implementations and our Valence method provides a convenient basis for summary remarks.

(1) Elimination can be effective on these sparse but patterned simplicial complex boundary matrices. However this is true only if the pivoting strategy is well suited to this situation.

(2) For large enough sparse matrices, fill-in makes elimination more time consuming than the Valence method, and for the largest examples, elimination fails altogether due to excessive memory demand. With the Valence approach, we were able to compute the rank modulo primes for matrices with 500,000 or more rows and columns, while elimination was failing for matrices of sizes larger than about 50,000.

(3) It remains open how to efficiently determine the ranks modulo powers (> 1) of primes while using memory-efficient iterative methods.

7. REFERENCES

[1] E. Babson, A. Björner, S. Linusson, J. Shareshian, and V. Welker. Complexes of not i-connected graphs. *Topology*, 38(2):271–299, 1999.

[2] E. H. Bareiss. Sylvester's identity and multistep integer-preserving Gaussian elimination. *Mathematics of Computation*, 22(103):565–578, July 1968.

[3] A. Bjöerner, L. Lovász, S. Vrećica, and R. T. Živaljević. Chessboard complexes and matching complexes. *Journal of the London Mathematical Society, II. Ser. 49, No.1, 25-39*, 1994.

[4] A. Björner and V. Welker. Complexes of directed graphs. *SIAM Journal on Discrete Mathematics*, 12(4):413–424, Nov. 1999.

[5] A. Brauer. Limits for the characteristic roots of a matrix. *Duke Mathematical Journal*, 14:21–26, 1947.

[6] R. A. Brualdi and S. Mellendorf. Regions in the complex plane containing the eigenvalues of a matrix. *American Mathematical Monthly*, 101(10):975–985, Dec. 1994.

[7] Dixon, John D. Exact solution of linear equations using p-adic expansions. *Numerische Mathematik*, 40:137–141, 1982.

[8] J.-G. Dumas. David and Goliath : computing the rank of sparse matrices. Technical report, Sept. 1999. Internal report, Linbox group.

[9] J.-G. Dumas. Calcul du polynôme minimal entier en athapascan-1 et linbox. In *RenPar '2000. Actes des douzièmes rencontres francophones du paralllismes*, June 19–22 2000.

[10] J.-G. Dumas, F. Heckenbach, B. D. Saunders, and V. Welker. *Simplicial Homology, a share package for GAP*, Mar. 2000. Manual (http://www.cis.udel.edu/~heckenba/Homology).

[11] W. Eberly and E. Kaltofen. On randomized Lanczos algorithms. In W. W. Küchlin, editor, *ISSAC '97. Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, Maui, Hawaii*, pages 176–183. ACM Press, July 21–23, 1997.

[12] F. R. Gantmacher. *The Theory of Matrices.* Chelsea, New York, 1959.

[13] M. W. Giesbrecht. Probabilistic computation of the Smith Normal Form of a sparse integer matrix. *Lecture Notes in Computer Science*, 1122:173–186, 1996.

[14] M. W. Giesbrecht. Efficient parallel solution of sparse systems of linear diophantine equations. In *Parallel Symbolic Computation (PASCO'97)*, pages 1–10, Maui, Hawaii, USA, July 1997.

[15] 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 on Computing*, 18(4):658–669, 1989.

[16] E. Kaltofen. Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. *Mathematics of Computation*, 64(210):777–806, Apr. 1995.

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

[18] E. Kaltofen and B. D. Saunders. On Wiedemann's method of solving sparse linear systems. In H. F. Mattson, T. Mora, and T. R. N. Rao, editors, *Proceedings of Applied Algebra, Algebraic Algorithms and Error–Correcting Codes (AAECC '91)*, volume 539 of *LNCS*, pages 29–38, Berlin, Germany, Oct. 1991. Springer.

[19] H. R. Lee and B. D. Saunders. Fraction free Gaussian elimination for sparse matrices. *Journal of Symbolic Computation*, 19(5):393–402, Apr. 1995.

[20] R. H. Lewis. *Fermat, a computer algebra system for polynomial and matrix computations*, 1997. http://www.bway.net/~lewis.

[21] J.-P. Massias and G. Robin. Bornes effectives pour certaines fonctions concernant les nombres premiers. *Journal de Théorie des Nombres de Bordeaux*, 8:213–238, 1996.

[22] M. Mignotte. *Mathématiques pour le calcul formel*, chapter Majoration de la taille des facteurs d'un polynôme, page 168. PUF, 1989.

[23] T. Mulders and A. Storjohann. Diophantine linear system solving. In *International Symposium on Symbolic and Algebraic Computation (ISSAC 99)*, pages 181–188, Vancouver, BC, Canada, July 1999.

[24] J. R. Munkres. *Elements of algebraic topology*, chapter The computability of homology groups, pages 53–61. Advanced Book Program. The Benjamin/Cummings Publishing Company, Inc., 1994.

[25] J. A. Reeds and N. J. A. Sloane. Shift-register synthesis (modulo m). *SIAM Journal on Computing*, 14(3):505–513, Aug. 1985.

[26] E. Sibert, H. F. Mattson, and P. Jackson. Finite Field Arithmetic Using the Connection Machine. In R. Zippel, editor, *Computer algebra and parallelism, Proceedings of the second International Workshop on Parallel Algebraic Computation*, volume 584 of *LNCS*, pages 51–61. Springer Verlag, May 1990.

[27] A. Storjohann. Near optimal algorithms for computing Smith normal forms of integer matrices. In Lakshman Y. N., editor, *ISSAC '96: Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation, July 24–26, 1996, Zurich, Switzerland*, pages 267–274. ACM Press, 1996.

[28] O. Taussky. Bounds for characteristic roots of matrices. *Duke Mathematical Journal*, 15:1043–1044, 1948.

[29] G. Villard. Further analysis of Coppersmith's block Wiedemann algorithm for the solution of sparse linear systems. In W. W. Küchlin, editor, *ISSAC '97. Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, Maui, Hawaii*, pages 32–39, July 21–23, 1997.

[30] G. Villard. Applications of sparse pre-conditioners over finite fields. Technical report, Feb. 1999. Internal report, Linbox group.

[31] J. Von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, New York, NY, USA, 1999.

[32] D. H. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Transactions on Information Theory*, 32(1):54–62, Jan. 1986.