Computing Simplicial Homology Based on Efficient Smith Normal Form Algorithms

Jean-Guillaume Dumas[1], Frank Heckenbach[2], David Saunders[3], and Volkmar Welker[4]

[1] Laboratoire de Modélisation et Calcul, IMAG-B. P. 53, 38041 Grenoble, France
[2] Universität Erlangen-Nürnberg, Mathematisches Institut, Bismarckstr. 1 1/2, 91054 Erlangen, Germany
[3] University of Delaware, Department of Computer and Information Sciences, Newark, DE 19716, USA
[4] Philipps-Universität Marburg, Fachbereich Mathematik und Informatik, 35032 Marburg, Germany

Abstract. We recall that the calculation of homology with integer coefficients of a simplicial complex reduces to the calculation of the Smith Normal Form of the boundary matrices which in general are sparse. We provide a review of several algorithms for the calculation of Smith Normal Form of sparse matrices and compare their running times for actual boundary matrices. Then we describe alternative approaches to the calculation of simplicial homology. The last section then describes motivating examples and actual experiments with the GAP package that was implemented by the authors. These examples also include as an example of other homology theories some calculations of Lie algebra homology.

1 Introduction

Geometric properties of topological spaces are often conveniently expressed by algebraic invariants of the space. Notably the fundamental group, the higher homotopy groups, the homology groups and the cohomology algebra play a prominent role in this respect. This paper focuses on methods for the computer calculation of the homology of finite simplicial complexes and its applications (see comments on the other invariants in Section 7).

We assume that the topological space, whose homology we want to calculate, is given as an abstract simplicial complex. We recall now the basic notions and refer the reader to Munkres' books [33] and [32] for further details. A finite simplicial complex Δ over the non-empty ground set Ω is a non-empty subset of the power set 2^{Ω} such that $A \subseteq B \in \Delta$ implies $A \in \Delta$. Without loss of generality we will always assume that $\Omega = [n] := \{1, \ldots, n\}$. An element of Δ is called a face. An inclusion-wise maximal element of Δ is called a facet. The dimension $\dim A$ of a face $A \in \Delta$ is given by $\#A - 1$. The dimension $\dim \Delta$ of Δ is the maximal dimension of one of its faces. The definitions imply that every simplicial complex contains the empty set as its unique face of dimension -1. By $f_i = f_i(\Delta)$ we denote the number of

i-dimensional faces of Δ and call $(f_{-1}, \dots, f_d)$, $d = \dim \Delta$, the f-vector of Δ. The ith chain group $C_i(\Delta, R)$ of Δ with coefficients in the ring R is the free R-module of rank f_i with basis e_A indexed by the i-dimensional faces $A \in \Delta$. The ith differential $\partial_i : C_i(\Delta, R) \to C_{i-1}(\Delta, R)$ is the R-module homomorphism defined by $\partial_i e_A = \sum_{j=0}^{i} (-1)^j e_{A \setminus \{a_j\}}$, where $A = \{a_0 < \cdots < a_i\}$. It is easily checked that $\partial_i \circ \partial_{i+1} = 0$ and hence $\mathtt{Im}\partial_{i+1} \subseteq \mathtt{Ker}\partial_i$. This leads to the definition of the ith (reduced) homology group $\widetilde{H}_i(\Delta; R)$ as the R-module $\mathtt{Ker}\partial_i / \mathtt{Im}\partial_{i+1}$. In this paper we will only be concerned with the case when $R = k$ is a field and $R = \mathbb{Z}$ is the ring of integers.

The definition of the reduced homology group clearly implies that its computation reduces to linear algebra over a field or $\mathbb{Z}$.

In case $R = k$ is a field the dimension of $\widetilde{H}_i(\Delta; k)$ as a k-vector space is easily calculated as $\beta_i^k = \dim \mathtt{Ker}\partial_i - \dim \mathtt{Im}\partial_{i+1} = f_i - \mathtt{rank}\partial_i - \mathtt{rank}\partial_{i+1}$. Here and whenever convenient we identify the linear map ∂_i with its matrix $\mathtt{Mat}(\partial_i)$ with respect to the canonical bases e_A and denote by $\mathtt{rank}$ the rank of a matrix. The definitions imply that β_i^k depends on the characteristic of k only and that β_i^k is constant except for fields k whose characteristic lies in a finite set of primes which of course depend on Δ.

If $R = \mathbb{Z}$, then $\widetilde{H}_i(\Delta; R)$ is a a quotient of subgroups of free $\mathbb{Z}$-modules of finite rank. Hence $\widetilde{H}_i(\Delta; R)$ is a finitely generated abelian group – note that we only consider finite simplicial complexes. In order to compute the structure invariants of $\widetilde{H}_i(\Delta; \mathbb{Z})$, we calculate the Smith Normal Form of the matrix $\mathtt{Mat}(\partial_i)$. Recall that the Smith Normal Form of an integer Matrix A is a diagonal matrix of the same size as A which is unimodularly equivalent to A (for more technical details see Definition 4).

Assume that

$$D_{i+1} = \begin{pmatrix} b_1 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & b_2 & \cdots & 0 & 0 & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & 0 & \cdots & 0 \\ 0 & 0 & \cdots & b_t & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix}$$

is the Smith Normal Form of the matrix $\mathtt{Mat}(\partial_{i+1})$, with natural numbers $b_1 | b_2 | \dots | b_t \neq 0$. Let $e_1', \dots, e_s'$ be the basis of $C_i(\Delta)$ which corresponds to the Smith Normal Form D_{i+1}. Then $b_1 e_1', \dots, b_t e_t'$ is a basis of $\mathtt{Im}\partial_{i+1}$. Since $\partial_i \circ \partial_{i+1} = 0$ and ∂_{i+1} is $\mathbb{Z}$-linear it follows that $e_1', \dots, e_t' \in \mathtt{Ker}\partial_i$. Now after possible renumbering, we can choose the basis e_i' such that for $r = f_i - \mathtt{rank}\partial_i$ we have $e_1', \dots, e_r'$ is a basis of $\mathtt{Ker}\partial_i$. Thus

$$\widetilde{H}_i(\Delta; \mathbb{Z}) \cong \mathbb{Z}_{b_1} \oplus \cdots \oplus \mathbb{Z}_{b_t} \oplus \mathbb{Z}^{r-t}.$$

Here we write $\mathbb{Z}_l$ for the abelian group $\mathbb{Z}/l\mathbb{Z}$ of residues modulo l.

Thus the computation of homology groups of simplicial complexes boils down to the following two fundamental problems in exact linear algebra:

(i) Calculate the rank of a matrix over a finite field or $\mathbb{Q}$.
(ii) Calculate the Smith Normal Form of a matrix over $\mathbb{Z}$.

In our paper we will focus on (ii), integer coefficients. But some of the methods relevant for (i) will also be useful in this context. Clearly, for the first problem Gauss' algorithm gives a polynomial time algorithm. But when the size of the matrix grows, more efficient methods are in demand. This is even more so for (ii). Here [23] shows that the problem has an efficient solution. But again the matrices, which arise in our context, request algorithms that take advantage of their special structure in order to have a Smith Normal Form algorithm that terminates within a lifetime. For general information on the complexity of the problem of calculating homology and related problems see [27].

We have implemented our software as a package for the computer algebra system GAP [15]. It can be downloaded from

`http://www.cis.udel.edu/~dumas/Homology/`

The package is split into two parts, a part which consists of an external binary that implements four algorithms calculating the Smith Normal Form and a part that implements algorithms to create and manipulate simplicial complexes in the GAP [15] language.

We report here the key features of the design, implementation, and use of our `Simplicial Homology` package for the GAP [15] system.

We will present the basic definitions and relationships involved in simplicial complexes, their homology, and the Smith Normal Forms. We will discuss some projects in geometric combinatorics which led us to design the package.

Secondly we will describe in some detail the methods we use in the package. This includes information on how the boundary matrices are built, and the properties of the several algorithms for computing Smith Normal Forms. In addition we will mention several possible new methods and possible new connections of the package to other mathematical software.

Finally we will illustrate the usefulness of the approach and of the software in addressing a couple of topics concerning simplicial homology and Lie algebra homology.

In a concluding section we comment on other invariants of simplicial complexes.

2 Generating the Boundary Matrices

For our purposes a simplicial complex is most conveniently described by the set of its facets. Clearly, if we know the facets then we know all faces of the simplicial complex, since a set is a face if and only if there is a facet containing it.

Even though a list of facets gives a succinct presentation of a simplicial complex, a linear algebra approach to simplicial homology requires the then expensive task to generate all faces of the simplicial complex from which the matrices of the boundary maps ∂_i are set up. On the other hand it is an interesting theoretical question whether it is indeed necessary to go through these time consuming computations in order to calculate homology.

The following section will summarize the algorithms we have implemented in our package that generate the set of faces from the set of facets and the boundary matrices from the set of faces.

A simplicial complex Δ, when read from the input data, is stored as an array of its facets. The facets, as well as other simplices in the following steps, are stored as arrays of the points they contain where the points are indexed by integers.

In order to generate a boundary matrix `Mat`(∂_k), we first generate the lists of simplices of dimension k and $k-1$ (`GetFaces` in file `simplicial.pas`). Of course when computing the homology in several consecutive dimensions, we can reuse the list of k-simplices for the generation of ∂_{k+1}, so we have only to generate one new simplex list for each new dimension. Generating the $k-$ or $(k-1)-$ faces basically amounts to generating the $(k+1)-$ and $k-$element subsets of each facet.

The main problem here is that many simplices, especially in the low and middle dimensions, are subsets of many facets so they are generated many times when processing all the facets. Storing all the copies in memory first and removing duplicates at the end would therefore require much more memory than is needed to store the distinct simplices. Indeed there are cases where an otherwise feasible computation would fail at this point when keeping the duplicates.

For sorting the simplices we use the lexicographic order on the ordered set of point indices which is a natural choice to allow for fast binary searching in the following steps (`FindFace`). A merge sort algorithm is implemented to sort the simplices and remove duplicates early. More precisely, we use an array of lists of simplices with the restriction that the ith list must never contain more than 2^{i-1} entries. Newly generated simplices are always inserted into the first list (`NewFace`). Whenever a list grows too big, it is merged with the following list, and when elements exist in both lists, one copy is removed (`Merge`). This process is repeated with the subsequent lists until the size of each list is again below its limit. When all facets have been processed, all the lists are merged together (`CollectFaces`).

With the lists of simplices generated, it is straightforward to generate the boundary matrices, by iterating over the k-simplices, corresponding to the rows of the matrix. For technical reasons we consider the transposed boundary matrix. Clearly, transposition does not change the Smith Normal Form. This way we obtain the matrix row by row rather than column by column. For the elimination algorithm we can take advantage of this procedure. Namely instead of generating the whole matrix at once, we do it one row at a time (`GetOneRow`) and process each row before the next one is generated. This makes it possible to work with matrices whose total size would exceed memory limits.

Each row is generated by taking the $(k-1)$-subsimplices, obtained by deleting one point from the k-simplex, looking them up via binary search in the list of all $(k-1)$-simplices to get the column index, and alternating the matrix entries between 1 and -1. By processing the subsimplices in the natural lexicographic order, we automatically get the row correctly ordered in the index/value format that we use in the further steps for storing matrix rows and other sparse vectors.

3 The Elimination Algorithm

One of the algorithms for computing the Smith Normal Form is a Gaussian elimination with some specific modifications described in this section.

This algorithm is implemented in two variants, one using machine word size integers, the other one using integers of unlimited size, provided by the GMP library. The first version checks for overflows and aborts when it detects one. The second version initially uses machine integers as well. When it detects an overflow, it switches to GMP integers, but only for the affected values. This is based on the experience that even in situations where machine integers are not sufficient, often only few values really overflow, and for the majority of small integers, computing with machine integers is faster than with GMP integers. Apart from this, the two implementations of the algorithm are identical.

Before the elimination is started, we make an attempt to avoid it entirely. If we have computed $\texttt{Mat}(\partial_{k-1})$ before, we know an upper bound on the rank of $\texttt{Mat}(\partial_k)$, namely $\dim C_{k-1} - \texttt{rank}Mat(\partial_{k-1})$. Therefore, a quick pass is made over the matrix, trying to find as many linearly independent rows as *easily* possible (`QuickCheck` in file `arith.pas`). It is worth noting that due to the lexicographic ordering of both the k and $(k-1)$-simplices, the column index of the first non-zero entry, in the following denoted by r_i where i is the row index, is monotonic over the rows. Therefore, it is easy to count the number of rows with distinct r_i which are clearly linearly independent. If this number is as big as the known bound, we are done since the matrix contains only ± 1 entries at this point, so we know that the Smith Normal Form can only have 1 entries. This short-cut typically works for some smaller matrices, not for the largest and most difficult matrices, and it can only work if the homology group $\widetilde{H}_{k-1}$ is in fact 0, but since it does not take much time, it seems worth trying it, anyway.

The actual elimination (`SmithNormalForm`) processes the matrix row by row (`DoOneRow`), first trying to find as many rows as possible with distinct r_i

and $A_{i,r_i} = \pm 1$ (where $A_{i,j}$ denotes the matrix entries). Each row is modified using row additions until one of the following situations occurs:

- The row is the 0-vector and can be removed. Clearly, in this case the size of the matrix to deal with is reduced before it is fully generated, as noted in the description of the generation of the boundary matrix.
- r_i is distinct from r_j for all previously considered rows j, and $A_{i,r_i} = \pm 1$. Then the row is kept in this form.
- r_i is distinct from r_j for all previously considered rows j, but $|A_{i,r_i}| > 1$. Such rows are deferred in this pass, to avoid having to deal with non-1 pivots in the subsequent row operations, in the hope that the row can be further modified after more rows have been considered, and that the total number of such problematic rows is small compared to the size of the matrix. This is confirmed by observations.

After all rows have been processed, the rows that were deferred are processed again in the same way, but without deferring them again. This makes it necessary to use more complicated row operations at this point, such as a gcd step when dealing with two rows i and j with $r_i = r_j$ and $|A_{i,r_i}|, |A_{j,r_j}| > 1$ and A_{i,r_i} is not divisible by A_{j,r_j} or vice versa.

At the end of this second pass, we have the matrix in echelon form, with many pivots being ± 1. In the (frequent) case that in fact all pivots are ± 1, we only have to count the remaining rows and be done.

Otherwise, we can now remove the rows with ± 1 pivots, counting them, to vastly reduce the size of the matrix. The empirical observation which makes this algorithm work is that in cases with several hundreds of thousands of rows, the number of remaining rows was rarely found to be more than a few hundred. Since the ± 1 pivots are generally not the leftmost ones, this requires some more row additions.

The remaining matrix is now transformed to Smith Normal Form using a standard algorithm as described (`GetTorsion`), e.g., in [32, §11], involving both row and column operations. Since the latter are expensive with the sparse vector representation used, they are avoided as far as possible, but in general they are necessary in this step.

A result of Storjohann [37] allows us to do this last step modulo the determinant of a nonsingular maximal rank minor. This is very helpful to avoid coefficient explosion which was often observed in this step without this modulo reduction. Since the remaining matrix is of full row rank, we can choose as the minor the leftmost $n \times n$ submatrix (if the matrix is of size $m \times n$, $m \geq n$), i.e., the smallest matrix containing all the pivots. Since the matrix is in echelon form at this point, the determinant is just the product of all the pivots.

4 Valence Algorithm

In this section we describe a second algorithm for the computation of the Smith Normal Form of an integer matrix. This algorithm has proven very effective on some of the boundary matrices discussed in this paper, though the worst case asymptotic complexity is not better than that of the best currently known [17]. One practically important aspect of our method is that we avoid preconditioners which introduce a log factor in the running time. When the matrix size is near a million, this factor is near 20, and is hard to bear when it applies to a computation which already requires hours or days. In this paper we offer a brief overview of our method. For more detail about the valence algorithm, the reader can refer to [10]. The method is particularly effective when the degree of the minimal polynomial of AA^t is small.

There are two kinds of problems arising when computing the Smith Normal Form via elimination over the integers: one is the fill-in of the sparse matrices, and the other one is coefficient growth of the intermediate values. Both of those can lead to memory thrashing, thus killing the computation (see attempts to avoid these phenomena in Section 3). We want to bypass those problems. For the fill-in, the ideas are to use reordering to reduce fill-in and to use iterative methods to maintain a constant level of memory usage. For the coefficient growth, we work modulo primes, or small powers of primes and reconstruct as integers only the results (not the intermediate values).

We begin with some definitions. The *valence* of a polynomial is its trailing nonzero coefficient. The *characteristic valence* of a matrix is the valence of its characteristic polynomial. The *minimal valence* or simply the *valence* of a matrix A is the valence of its minimal polynomial, $\texttt{minpoly}_A(X)$. The *valuation* of a matrix is the degree of the trailing term of its minimal polynomial. 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, denoted by $d_i(A)$, is the greatest common divisor of the $i \times i$ minors of A. Define $d_0(A) = 1$. For $1 \leq i \leq \min(m,n)$, the i-th *invariant factor* of A is $s_i(A) = d_i(A)/d_{i-1}(A)$, or 0 if $d_i(A) = 0$. Let $s_0 = 1$. And as before, the *Smith Normal Form* of a A is the diagonal matrix

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

of the same shape as A. It is well known that for $1 \leq i \leq \min(m,n)$, we have $d_{i-1}|d_i$, $s_{i-1}|s_i$, and that A is unimodularly equivalent to its Smith Normal Form.

For a positive integer q, we define 'rank of $A \bmod q$', to be the greatest i such that q does not divide the i-th invariant factor of A, and we denote this rank by $r_q = \texttt{rank}_q(A)$.

The next algorithm statement is an overview of the valence method. The individual steps can be done in many ways. Afterwards we will discuss the implementation details.

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

Theorem 4.1 ([10]). *Let A be a matrix in $\mathbb{Z}^{m\times n}$. Let $(s_1, \ldots, s_r)$ be its nonzero invariant factors. If a prime $p \in \mathbb{Z}$ divides some nonzero s_i, then p^2 divides the characteristic valence of AA^t, and p divides the minimal valence of AA^t. The same is true of the valences of A^tA.*

Corollary 4.2. *Algorithm* Valence-Smith-Form *correctly computes the Smith Normal Form.*

Proof. The theorem shows that we consider the relevant primes. It is evident that the integer Smith Normal Form may be composed from the relevant local Smith Normal Forms, since the integer Smith Normal Form *is* the local Smith Normal Form at p up to multiples which are units `mod` p.

4.1 Minimal valence computation

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

Lemma 4.3 ([10]).
For a matrix A the minimal polynomials of A^tA and AA^t are equal or differ by a factor of x.

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 $\texttt{minpoly}_{A^tA}$ can then be applied to those of $\texttt{minpoly}_{AA^t}$ and vice versa.

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.

The first question is how many primes must be used for the Chinese remaindering. Using Hadamard's inequality [16, Theorem 16.6] would induce a use of $O(n)$ primes. The Hadamard bound may be used, but is too pessimistic an estimate for many 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.

The i-th Gershgörin disk is centered at $b_{i,i}$ 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 [4,38,18]. One can then go further and consider the ovals of Cassini [5,6,40], 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 to bound the coefficients of the minimal polynomial.

Proposition 4.4 ([10]). *Let $B \in \mathbb{C}^{n\times n}$ with its spectral radius bounded by β. Let $\texttt{minpoly}_B(X) = \sum_{k=0}^{d} m_i X^i$. Then the valence of B is no more than β^d in absolute value and $|m_i| \le \max\{\sqrt{d\beta};\beta\}^d$, for $i = 0,\dots,d$.*

For matrices of constant size entries, both β and d are $\mathtt{O}(n)$. However, when d and/or β is small relative to n (especially d) this may be a striking improvement over the Hadamard bound since the length of the latter would be of order $n\log(n)$ rather than $d\log(\beta)$.

Our experiments suggest that this is often the case for homology matrices. Indeed, for those, $B = AA^t$ has very small minimal polynomial degree and has some other useful properties. For instance it is diagonally dominant. But we do not think that these facts are true for all homology matrices. On the other hand, we neither have a proof of this assertion nor its converse.

There remains to compute the 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: The diagonal values can be computed first, by a simple dot product: $q_i = \sum_{j\in 1\dots n} a_{ij}^2$, and serve as centers for the disks and ovals. Then the radii are compute via the absolute value of $A : v = |A||A|^t[1,1,\dots,1]^t$ and set $r_i = v_i - |q_i|$ for $i = 1,\dots,m$.

Minimal Polynomial over the Integers and Bad Primes. The next question is to actually compute the minimal polynomial of a matrix modulo primes. To perform this, we use Wiedemann's probabilistic algorithm [42]. It is probabilistic in the sense that it returns a polynomial which is deterministically a factor of the minimum polynomial and is likely to be the minimal polynomial. When we know the degree of the minimal polynomial, just by a degree check, we can then be sure that the computed polynomial is correct. Thus in order to complete the valence computation we must be sure of the degree of this polynomial over the integers. To compute this degree, 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. Some primes may give a lower degree minimal polynomial. We call them *bad primes.* We next bound the probability of choosing a bad prime at random, by bounding the size of a minor of the matrix (M_δ) that such a prime must divide.

Let δ be the degree of the integer minimal polynomial and U an upper bound on the number of bad primes. We proved that U can have the following value [10]:

$$|M_\delta| \leq \lceil \frac{\delta}{2} \rceil \sqrt{n} \beta^{\frac{\delta^2}{2}} = U.$$

Suppose we choose primes at random from a set P of primes each greater than a lower bound l. There can be no more than $\log_l(U)$ primes greater than l dividing M_δ. It suffices to pick from an adequately large set P to reduce the probability of choosing bad primes. The distribution of primes assures that adequately large P can be constructed containing primes that are not excessively large.

We now give the complete algorithm for the computation of the Valence. The algorithm involves computation of minimal polynomials over Z_p. For the fast probabilistic computation of these we use Wiedemann's method (and probability estimates) [42] with early termination as in [26]. We then construct the integer minimal polynomial using Chinese remaindering.

In the following we will denote by $\mu_l(x)$ a lower bound on the number of distinct primes between l and x; this bound is easily computed using direct bounds on $\pi(x)$, the number of primes lower than x, [11, Theorem 1.10].

The binary cost of the multiplication of two integers of length n, will be denoted by $I_m(n)$: classical multiplication uses $I_m(n) = 2n^2$ bit operations, Karatsuba's method uses $I_m(n) = \mathtt{O}(n^{1.59})$ and Schönhage & Strassen's method uses $I_m(n) = \mathtt{O}(n \log(n) \log\log(n))$ [16]. For convenience, we will also use "soft-Oh" notation: for any cost functions f and g, we write $f = \mathtt{O}^{\sim}(g)$ if and only if $f = \mathtt{O}(g \log^c(g))$ for some constant $c > 0$. For a matrix $A \in \mathbb{C}^{m \times n}$ let e be the number of nonzero entries. Let $E = \max\{m, n, e\}$.

Theorem 4.5 ([10]). *Algorithm* Integer-Minimal-Polynomial *is correct. Let* $s = d \max\{\log_2(\beta(A)); \log_2(d)\}$, *which bounds the lengths of the minimal polynomial coefficients. The algorithm uses expected time*

$$\mathtt{O}(sdE \log(\epsilon^{-1}))$$

for constant size entries. It uses $\mathtt{O}(n \log(s) + ds)$ *memory if every coefficient of the minimal polynomial is computed, and* $\mathtt{O}(n \log(s) + s)$ *if only the valence is computed. The latter is also* $\mathtt{O}^{\sim}(n)$.

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, at most, only 1.7% of bad primes. With only 10 polynomials this reduces to a probability of failure less than 2×10^{-16}.

4.2 Ranks Computation

Once the valence is computed, we know all the primes involved in the Smith Normal Form. We then have to factor the valence and perform rank computations modulo those primes. Two problems can arise. First we might need to compute the rank modulo a *power* of a prime: we deal with this via an elimination method. Second, the valence can be hard to factor: in this case we consider the big composite factor to be prime. If problems arise in the elimination or in the iterative resolution, in fact a factorization of this number has been achieved. Thus we do not have to pay the high cost of explicit factorization of large numbers with large prime factors. [10]).

Next consider the question of computing the local Smith Normal Form in $\mathbb{Z}_{(p)}$. This is equivalent to a computation of the rank $\bmod\ p^k$ for sufficiently many k. Recall that we define the rank $\bmod\ p^k$ as the number of nonzero invariant factors $\bmod\ p^k$. 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.

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

Theorem 4.6 ([10]). *For a positive integer e, algorithm* Local Ranks by Elimination $\bmod\ p^e$ *is correct and runs using* $O(rmn)$ *arithmetic operations* $\bmod\ p^e$, *where r is the rank* $\bmod\ p^e$, *and* $O(emn)$ *memory.*

Wiedemann's algorithm and Diagonal Scaling. For some matrices the elimination approach just described fails due to excessive memory demand (disk thrashing). It is desirable to have a memory efficient method for such cases. Two iterative methods are proposed for use here. The first one is "off the shelf". It is to use Wiedemann's algorithm with the diagonal scaling of [13] to compute the rank $\bmod\ p$. This scaling ensures with high probability that the minimal polynomial is a shift of the characteristic polynomial, in fact that it is of the form $m(x) = xf(x)$ where $f(0) \neq 0$ and the characteristic polynomial is of the form $x^k f(x)$ for some k. It follows that the rank is the degree of f. For a given ϵ, to do this with probability of correctness greater than $1 - \epsilon$ requires computation in a field of size $\mathtt{O}(n^2/\epsilon)$ [13]. If p is insufficiently large, an extension field may be used. To avoid the large field requirement, we may use the technique as an heuristic, computing the Wiedemann polynomial over a smaller field. The resulting polynomial, $w(x)$, is guaranteed to be a factor of the true minimal polynomial, so that it suffices to verify that $w(A) = 0$. This may be probabilistically done by choosing a vector v at random and computing $w(A)v$. The probability that $w(A)v$ is zero while $w(A)$ is nonzero is no more than $1/p$, hence repetition of this process $\log_2(\epsilon^{-1})$ times ensures that the rank has been computed correctly with probability no less than $1-\epsilon$.

This algorithm has much lower memory requirements than elimination, requiring $\mathtt{O}(E)$ field elements, for $A \in \mathbb{C}^{m\times n}$ with e the number of nonzero entries and $E = \max\{m, n, e\}$. The algorithm also has better asymptotic time complexity, $\mathtt{O}(dE\log_2(\epsilon^{-1}))$ field operations, and it is effective in practice for large sparse matrices over large fields. However it does not give the complete local Smith Normal Form at p. In [10] we propose a p-adic way to compute the last invariant factor of this local Smith Normal Form at p. From this, one may infer the complete structure of the local Smith Normal Form at p in some other cases.

4.3 Experiments with Homology Matrices

In this section, we will talk mainly 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 complex (see 6.1) with i vertices.
▷ **ch**i-k.**b**j denotes the boundary matrix j from the i by k chessboard complex (see 6.1).
▷ **n**i**ck**.**b**j denotes the boundary matrix j from the not i-connected graph (see 6.1) with k vertices.

The boundary matrices are sparse matrices with a fixed number k of nonzero 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 $A_i A_i^t$ and $A_i^t A_i$ 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 nonzero elements as A_i. Thus, we did not perform the matrix multiplications to compute $A_i A_i^t v$. We performed two matrix-vector products, $A_i(A_i^t v)$, instead.

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.

Our experiments were realized on a cluster of 20 Sun Microsystem Ultra Enterprise 450 each with four 250 MHz 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 Zech logarithms with precomputed tables, as in [36]. The algorithms were implemented in C++ with the LINBOX[1] library for computer algebra and the ATHAPASCAN[2] environment for parallelism.

In Table 1 we report some comparisons between Wiedemann's algorithm and elimination with reordering for computing the rank. 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.

Table 1. Rank `mod65521` – Elimination vs. Wiedemann

Matrix	Elimination	Wiedemann
mk9.b3	0.26	2.11
mk13.b5	MT	23 hours
ch7-7.b6	4.67	119.53
ch7-6.b4	49.32	412.42
ch7-7.b5	2179.62	4141.32
ch8-8.b4	19 hours	33 hours
ch8-8.b5	MT	55 hours
n2c6.b6	6.44	64.66
n2c6.b7	3.64	49.46
n4c5.b6	2.73	47.17
n4c6.b12	231.34	4131.06
n4c6.b13	8.92	288.57

In Table 2 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

[1] Symbolic linear algebra library, `http://www.linalg.org`

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

p to compute the invariant factors locally at p. Simplicial Homology [9] is a proposed GAP [15] share package. It computes homology groups of simplicial complexes via the Smith Normal Form of their boundary maps. It features a version of our Valence algorithm as well as the classical elimination method for homology groups. The entry "Hom-Elim-GMP" in this table refers to this elimination-based method using Gnu Multi Precision integers. Fermat [31] is a computer algebra system for Macs and Windows. Its Smith Normal Form routine is an implementation of [2].

Table 2. Fermat vs. Hom-Elim-GMP vs. SFV

Matrix	Fermat	Hom-Elim-GMP	Valence (Eliminations)	
ch6-6.b4	49.4	2.151	27.417	(6)
mk9.b3	2.03	0.211	0.946	(4)
mk10.b3	8.4	0.936	13.971	(7)
mk11.b4	98937.27	2789.707	384.51	(7)
mk12.b3	189.9	26.111	304.22	(7)
mk12.b4	MT	MT	13173.49	(7)

"Hom-Elim-GMP" and "Valence" ran on a 400 MHz sparc SUNW, Ultra-4 processor with 512 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 512 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 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 (the number of those eliminations is shown between parenthesis in the column Valence (Eliminations) of the table) - of course in parallel this difference will weaken. 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 1 we can see some of these effects and we present some of those partial results in Table 3. Ffor some matrices we were able to compute ranks modulo some primes, and therefore the occurrence of these primes in the Smith Normal Form, but not the actual powers of these primes. These, however, are the only currently known results about these matrices.

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. It remains open how

Algorithm A: VSF [Valence-Smith-Form]

```
Data   : a matrix $A \in \mathbb{Z}^{m \times n}$. 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 $x \in \mathbb{Z}^n$, $y \longrightarrow yA$ for $y^t \in \mathbb{Z}^m$.
Result : $S = \mathtt{diag}(s_1, \dots, s_{\min(m,n)})$, the integer Smith Normal Form of
         $A$.
begin
    (1) [ Valence computation ]
    if $m < n$ then
        let $B = AA^t$
    else
        let $B = A^tA$.
    Let $N = \min(m, n)$
    Compute the (minimal) valence $v$ of $B$.                [See section 4.1 ]

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

    (3) [Rank and first estimate of Smith Normal Form.]
    Choose a prime $p$ not in $L$ (i.e. $p$ does not divide $v$).
    Compute $r = \mathtt{rank}_p(A)$.                [ This is the integer rank.
      The first $r$ invariant factors are nonzero and the last $N - r$ are 0's. ]
    Set $S = \mathtt{diag}(s_1, \dots, s_N)$, where $s_i = 1$ for $i \leq r$ and $s_i = 0$ for $i > r$.

    (4) [Nontrivial invariant factors]
    foreach prime $p \in L$ do
        Compute $S_p = \mathtt{diag}(s_{p,1}, \dots, s_{p,N})$, the Smith Normal Form of $A$
        over the localization $\mathbb{Z}_{(p)}$ of $\mathbb{Z}$ at the prime ideal $(p)$.
                                                             [See section 4.2 ]
        Set $S = SS_p$.      [ That is, set $s_i = s_i s_{p,i}$ for those $s_{p,i}$ which are
                                                   nontrivial powers of $p$. ]

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

Table 3. Valence Smith Normal Form with Black Box techniques

Matrix	Time		Results
mk13.b5	98 hours	Partial	133991 ones & 220 powers of 3
ch7-8.b5	180 hours	Partial	92916 ones, 35 powers of 3 & 8 powers of 2 and 3
ch8-8.b4	264 hours	Complete	100289 ones

Algorithm B: IMP [Integer-Minimal-Polynomial]

```
Data   : a matrix $A$ in $\mathbb{Z}^{n\times n}$
         an error tolerance $\epsilon$, such that $0 < \epsilon < 1$
         an upper bound $m$ on primes for which computations are fast,
         $m > 2^{15}$.
Result : the integer minimal polynomial of A, correct with probability at
         least $1 - \epsilon$.
begin
    (1) [Initialization, first set of primes ]
    set $l = 2^{15}$;
    set $d = 0$; set $F = \emptyset$; set $P = \emptyset$;
    $\beta$ = Ovals of Cassini Bound of $A$;
    set $M = m$;                              [ computations will be fast ]

    (2) [ Compute polynomials mod $p_i$ ]
    repeat
        Choose a prime $p_i$, with $l < p_i < M$.     [ at least $\mu_l(M)$ of those ]
        Compute polynomial $w_{A,p_i}$ by Wiedemann's method.
        [ $w_{A,p_i} = \mathtt{minpoly}_{A,p_i}$ with probability at least $1 - \frac{1}{p_i}$]
        correct with probability at least $1 - \frac{1}{p_i}$]
        if $\deg(w_{A,p_i}) > d$ then
            set $d = \deg(w_{A,p_i})$; set $F = \{p_i\}$; set $P = \{w_{A,p_i}\}$;
            set $U = \sqrt{n}\lceil\frac{d}{2}\rceil\beta^{\frac{(d+1)^2}{2}}$;
            set $bad = \log_l(U)$;             [ at most that many bad primes ]
            set $b_i = 2 \times bad(1 + \frac{2}{l-2}) + 3512$ ;      [ 3512 primes $< 2^{15}$ ]
            set $M_i$ = upper bound for $p_{b_i}$ ;  [ at least $b_i$ primes are $< M_i$ ]
            if $(M_i > M)$ then
                set $M = M_i$;                 [ computations will be slower,
                          degree will be correct with probability at least $\frac{1}{2}$ ]
        else
            if $\deg(\mathtt{minpoly}_{A,p_i}) = d$ then
                $F = F \cup \{p_i\}$;
                $P = P \cup \{\mathtt{minpoly}_{A,p_i}\}$;
    until $\prod_F p_i \geq \max\{\sqrt{d\beta};\beta\}^d$ and $\epsilon \geq \prod_F(\frac{1}{p_i} + \frac{bad}{\mu_l(M)})$
    (4) [ Chinese remainders ]
    return $\mathtt{minpoly}_A = \sum_{j=1}^{d} \alpha_j X^j$,
    where each $\alpha_j \in \mathbb{Z}$ is built from $P$ and $F$.
end
```

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

```
Data    : a matrix $A \in \mathbb{Z}^{(m+1)\times(n+1)}$, with elements $(a_{ij})$ for $i, j \in 0 \dots m \times 0 \dots n$.
          a prime $p$.
          a positive integer $e$.
Result  : the ranks $r_{p^i}$, for $1 \leq i \leq e$.

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

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

    (3) [ Return local ranks ]
    return $r_{p^i}$, for $i \in \{1 \dots e\}$
end
```

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

5 Other Methods

Algebraic Shifting. It would be desirable to be able to read off the homology of a simplicial complex 'directly' from the list of its maximal faces. For shellable complexes for example a shellability of its maximal faces will allow to determine the homology. But calculating a shelling order probably is hard, even if one knows a priori that the complex is shellable (see [27]). But for a particular class of shellable complexes a simple procedure for determining the homology is known. A simplicial complex Δ over ground set $\Omega = [n]$ is called shifted if for $\sigma \in \Delta$ and number $1 \leq i < j \leq n$ such that $i \notin \sigma$ and $j \in \sigma$ we have $\sigma \setminus \{j\} \cup \{i\} \in \Delta$.

Lemma 5.1. *If Δ is a shifted complex then Δ is shellable and its ith homology group $\widetilde{H}_i(\Delta;\mathbb{Z})$ is free and its rank is given by the maximal faces of Δ of cardinality $i+1$ that do not contain the element* 1.

Clearly, this is a very simple algorithm that allows a linear time determination of the homology groups from the maximal faces. But we must admit that shifted complexes are a very small subclass even of the class of shellable complexes. But there is an algorithmic procedure that determines a shifted complex $\Delta(\Delta)$ from a simplicial complex Δ. This procedure is called algebraic shifting and was discovered by Gil Kalai (see [25] for the original references and details).

Let Δ be a simplicial complex over $[n]$. Let E be the exterior algebra freely generated by $e_1,\dots,e_n$. We write $e_A = e_{j_1} \wedge \cdots \wedge e_{j_s}$ for ordered subsets $\{j_1 < \cdots < j_s\}$ of $[n]$. By J_Δ be denote the ideal generated by the monomials e_A where $A \notin \Delta$. The ideal J_Δ is called the Stanley-Reisner ideal of Δ in E. Clearly, by virtue of exterior algebra, all monomials in E are of the form e_A for some $A \subseteq [n]$. Thus in order to specify a linear order on the monomials in E it suffices to order the subsets of $[n]$ linearly. Here we use lexicographic order '$\prec_l$', where $A \prec_l B$ if the minimum of $(A-B)\cup(B-A)$ lies in A. Now let us perform a generic linear transformation of the generators $e_1,\dots,e_n$. Thus we obtain new generators $f_i = \sum_{j=1}^n a_{ij}e_j$. Now we consider the set system:

$$\Delta^{\text{shifted}} = \left\{A \subseteq [n] \,\middle|\, f_A \notin \mathsf{span}_k\{f_B | B \prec_l A\} + J_\Delta\right\}.$$

Proposition 5.2 (Kalai; see [25]). *The set system Δ^{shifted} is a shifted simplicial complex.*

Now we state the theorem which allows us to use algebraic shifting for homology calculations.

Theorem 5.3 (Kalai; see [25]). *Let Δ be a simplicial complex and Δ^{shifted} be the result of algebraically shifting Δ over the field k then:*

$$\widetilde{H}_i(\Delta;k) = \widetilde{H}_i(\Delta^{\text{shifted}};k).$$

We recall a question by Kalai [25].
Is there a polynomial time algorithm for computing the algebraic shifted complex of a simplicial complex Δ which is given by its maximal faces ?

Minimal Free Resolutions of Stanley-Reisner Ideals. Let Δ be a simplicial complex over ground set $[n]$. Let $S = k[x_1,\dots,x_n]$ be the polynomial ring in n variables over k. The Stanley-Reisner ideal of Δ in S is, analogous to the situation in the exterior algebra, the ideal I_Δ generated by monomials $x_A = \prod_{i\in A} x_i$, where $A \notin \Delta$. A multigraded module M over S is an S-module which allows as a vectorspace a direct sum decomposition $M = \bigoplus_{\alpha\in\mathbb{N}^n} M_\alpha$, such that $\underline{\mathbf{x}}^\beta \mathbf{M}_\alpha \subseteq \mathbf{M}_{\alpha+\beta}$, where $\underline{x}^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ for $\alpha = (\alpha_1, \ldots, \alpha_n)$. Clearly, S itself and I_Δ are multigraded. We consider multigraded free resolutions of I_Δ as a module over the polynomial ring.

$$\mathcal{F} : \cdots \to \bigoplus_{\alpha \in \mathbb{N}^n} S(-\alpha)^{\beta_i^\alpha} \to \cdots \to \bigoplus_{\alpha \in \mathbb{N}^n} S(-\alpha)^{\beta_0^\alpha} \to I_\Delta \to 0,$$

such that each ∂_i is a homogeneous map of multigraded modules and $\mathtt{Ker}\partial_i = \mathtt{Im}\partial_{i+1}$. By tensoring $\mathcal{F}$ with the multigraded S-module $k = S/(x_1, \ldots, x_n)$ we obtain a complex

$$\mathcal{F} \otimes k : \cdots \to \bigoplus_{\alpha \in \mathbb{N}^n} k(-\alpha)^{\beta_i^\alpha} \to \cdots \to \bigoplus_{\alpha \in \mathbb{N}^n} k(-\alpha)^{\beta_0^\alpha} \to 0$$

of k-vectorspaces, with homogeneous differential $\partial_i \otimes k$, i.e., $\partial_{i-1} \otimes k \circ \partial_i \otimes k = 0$. Its homology groups $H_i(\mathcal{F} \otimes k)$ are known as $\mathtt{Tor}_i^S(I_\Delta, k)$. Since all differentials are multigraded this induces a decomposition

$$\bigoplus_{\alpha \in \mathbb{N}^n} \mathtt{Tor}_i^S(I_\Delta, k)_\alpha.$$

A special case of a result by Hochster says:

Theorem 5.4 (Hochster [20]). *Let Δ be a simplicial complex over $[n]$. Then*

$$\mathtt{Tor}_i^S(I_\Delta, k)_{(1,\ldots,1)} \cong \widetilde{H}_{n-2-i}(\Delta, k).$$

Thus the differential $\partial_i \otimes k$ yields a differential that is in general different from the simplicial differential of Δ but computes the same homology groups with field coefficients. Now the goal is to construct a most economic $\partial_i \otimes k$ (i.e. find a free resolution of I_Δ which has a very small $(1, \ldots, 1)$ strand).

Clearly the most economic resolution in this respects is the minimal free resolution of I_Δ for which the differential assumes it simplest form $\partial_i \otimes k = 0$ for all i. But even though computer algebra packages like Macaulay 2 [19] and Cocoa [7] can compute the minimal free resolution, it involves computing Gröbner bases and hence the feasibility of the approach is most likely very low – we have not performed comparative experiments though.

There are other free resolutions which are much simpler to construct given monomial generators of I_Δ. As an example we would like to mention Taylor's resolution [8].

We consider it an interesting and challenging problem to analyze free resolution of I_Δ with respect to their effectiveness calculating the homology of Δ in their $(1, \ldots, 1)$ strand.

Deducing homology of a simplicial complex from its dual. The algebraic methods described above use ideals that have in their description the minimal non-faces of a simplicial complex. This fundamentally differs from the description of a simplicial complex by facets. Here we discuss how this effects homology computations.

If Δ is a simplicial complex over ground set $[n]$ then we denote by $\Delta^* = \{A \mid [n] - A \not\in \Delta\}$ the dual simplicial complex. So A is a minimal non-face of Δ if and only if $[n] - A$ is a maximal face of Δ^*. Now the homology with field coefficients of Δ and Δ^* are related by Alexander duality [32, Cor. 72.4] and the isomorphism of homology and cohomology with field coefficients [32, Cor. 53.6]:

$$\widetilde{H}_i(\Delta; k) \cong \widetilde{H}_{n-i-3}(\Delta^*; k).$$

In particular this shows that for calculating homology it is irrelevant whether the simplicial complex is given by its maximal faces or its minimal non-faces even though in general it seems to be a difficult problem to pass from one representation to the other (we are grateful to Marc Pfetsch for pointing us to [14] or [28]). In particular, the complexity of the algebraic methods – algebraic shifting and finite free resolutions – described above does not depend on whether the simplicial complex is given by its maximal faces or its minimal non-faces.

Deducing integer homology from field coefficient homology. The method of algebraic shifting and the method of free resolutions will only calculate simplicial homology with coefficients in a field k. Clearly there are only finitely many primes, we call them the critical primes, for which a simplicial complex can have p-torsion. By the Universal Coefficient Theorem [32, Thm. 53.5] the sizes of the torsion parts for the various primes for which torsion occurs can be deduced from the homology computations over primes fields for all critical primes and in characteristic 0. Also the set of critical primes can be deduced as it is done within the Valence Algorithm (see Section 4). But then the problem arises how to distinguish p, p^2, ... torsion. We find it a challenging problem to find efficiently computable invariants that help to distinguish the p^i from p^j torsion for $i \neq j$.

6 Sample Applications

6.1 Complexes of Graphs

Complexes of graphs have been the motivating source for the development of our GAP [15] package. Let $G = (V, E)$ be a graph on vertex set V with edge set E. We assume $V = [n]$ and also assume that G contains no loops and no multiple edges. Thus E can be considered as a subset of $\binom{[n]}{2} = \{A \subseteq [n] \mid \#A = 2\}$. Conversely any subset of $\binom{[n]}{2}$ can be considered as the edge set of a graph. Hence from now on we do not distinguish between graphs and subsets of $\binom{[n]}{2}$. We call a simplicial complex Δ on ground set $\binom{[n]}{2}$ a complex of graphs. This concept is very general but usually we are interested in graph complexes that have a high symmetry with respect to the set of vertices.

Complexes of not i-connected graphs. A graph G is called 1-connected if for all $i, j \in V$ there is a sequence of edges that connects i and j – this is what we usually call a connected graph. For $i \geq 2$ a graph G is called i-connected if it is j-connected for all $1 \leq j \leq i-1$ and whenever $(i-1)$ vertices together with their adjacent edges are deleted the graph remains 1-connected. Clearly, removal of edges preserves the property of being not i-connected. Thus the set $\Delta_{n,i}$ of all not i-connected graphs on $[n]$ is a simplicial complex. The case $i = 2$ has been the primary motivation and starting point for the development of our GAP [15] package. Indeed calculations of early versions and predecessors of the package led to a conjecture that turned into the following theorem.

Theorem 6.1 ([1] and [39]).

$$\widetilde{H}_j(\Delta_{n,i}; \mathbb{Z}) = \begin{cases} 0 & j \neq 2n-5 \\ \mathbb{Z}^{(n-2)!} & j = 2n-4 \end{cases}$$

Indeed matrices coming from this example have been used as timing examples in previous parts of this manuscript.

Next we demonstrate in a sample session the use of some of the functions of the homology package.

First, start GAP and load the package.

```
> gap

        #########           ######         ###########           ###
     #############          ######         ############         ####
    ##############         ########        #############       #####
   ###############         ########        #####   ######      #####
  ######         #         #########       #####    #####     ######
 ######                   ##########       #####    #####    #######
 #####                    ##### ####       #####   ######   ########
 ####                    #####  #####      #############   ###  ####
 #####     #######       ####    ####      ###########    ####  ####
 #####     #######      #####    #####     ######        ####   ####
 #####     #######      #####    #####     #####         #############
  #####      #####     ################    #####         #############
  ######     #####     ################    #####         #############
  ################    ##################   #####                ####
   ###############    #####        #####   #####                ####
     #############    #####        #####   #####                ####
      #########      #####          #####  #####                ####

    Information at:  http://www-gap.dcs.st-and.ac.uk/~gap
    ? for help. Copyright and authors list by ?copyright, ?authors

   Loading the library. Please be patient, this may take a while.
GAP4, Version: 4.1 fix 2 of 27-Aug-1999, sparc-sun-solaris2.6-gcc
Components:  small, small2, small3, id2, id3, trans, prim, tbl,
             tom  installed.
gap>  RequirePackage("homology");
true
```

Let us calculate the homology of the complex of not 2-connected graphs on 7 vertices. By Theorem 6.1 it is know that the homology is concentrated in dimension $2 \cdot 7 - 5 = 9$ and is free of dimension $(7-2)! = 120$. So let us reconfirm the homology in dimension 9 only.

```
gap> SimplicialHomology(SCNot2ConnectedGraphs(7),9);
#I:D Simplicial complex of dimension 15 with 217 facets
#I:D   homology: Computing homology groups
#I:D      faces: Finding faces of dimension 8
#I:D      faces: Found 247100 faces of dimension 8
#I:D      faces: Finding faces of dimension 9
#I:D      faces: Found 219135 faces of dimension 9
#I:D   homology: Finding rank of boundary map d9
#I:D      elimination: Triangulating matrix by modified Gauss
                                                elimination
#I:D      elimination: current torsion-free rank: 0, current rank:
                                                   0, max. rank: 247100
#I:D      elimination: current torsion-free rank: 715, current rank:
                                                   715, max. rank: 247100
```

The last line tells us that rows of the matrix of the 9th differential processed so far have a span of rank 715. Using general implications from the fact that the sequence of differential is a complex (i.e., $\partial_{i-1} \circ \partial_i = 0$) the maximal possible rank this matrix can have is 247100. Let us look at the output a few minutes later.

```
#I:D      elimination: current torsion-free rank: 128311,
                              current rank: 128311, max. rank: 247100
#I:D      elimination: current torsion-free rank: 128330,
                              current rank: 128330, max. rank: 247100
#I:D      elimination: Matrix triangulated
#I:R   homology: Rank d9 = 128330
#I:D      faces: Finding faces of dimension 10
#I:D      faces: Found 135765 faces of dimension 10
#I:D   homology: Finding rank of boundary map d10
#I:D      elimination: Triangulating matrix by modified Gauss
                                                     elimination
#I:D      elimination: current torsion-free rank: 0,
                               current rank: 0, max. rank: 90805
```

The 9th differential has been brought into Smith Normal form. But for calculation the homology group in homological dimension 9 the program has already started to analyze the matrix of the 10th differential. Another few minutes waiting bring us to:

```
#I:D      elimination: current torsion-free rank: 90344,
                              current rank: 90344, max. rank: 90805
#I:D      elimination: current torsion-free rank: 90685,
                              current rank: 90685, max. rank: 90805
#I:D      elimination: Matrix triangulated
#I:R    homology: Rank d10 = 90685
#I:R    homology: H_9 = 120.
#I:D    homology: Homology groups computed
[ [ 120 ] ]
```

After Smith Normal forms of the 9th and 10th differential are calculated the program determines the homology group we have asked for and indeed it is as expected. Note that in the process of calculating this group we have brought within roughly 15 minutes (on a Linux PC, 700 MHZ) two matrices into Smith Normal Form whose row and column sizes are in the 5 or 6 digit numbers.

For $i = 1$ the homology of $\Delta_{n.i}$ had been determined as a one of the first examples of homology computations in combinatorics (see [1] and [39]). For $i = 3$ a conjecture from [1] also based on computer calculations was later verified by Jakob Jonsson. For $i > 3$ the problem is wide open. But for $i = n - 2$ a simple Alexander duality argument reduces the problem to calculating homology of matching complexes.

Matching Complexes. If G is a graph with vertex set E then a partial matching on G is a subset H of E such that no two edges in G share a vertex. Clearly the set Δ_M^G of all partial matchings on G is a simplicial complex. If G is the complete graph on n vertices then we simply call $\Delta_M^G = \Delta_M^n$ the matching complex and when G is the complete bipartite complex on $2n$ vertices with n vertices in each partition then we call $\Delta_M^G = \Delta_M^{n,n}$ the chessboard complex – the name is derived from the fact that the complex can also defined as the set of non-taking rook positions on an n by n chessboard.

Both Δ_M^n and $\Delta_M^{n,n}$ are ubiquitous in mathematics (see [41]) for an excellent survey). Also both complexes have raised interest from the computational point of view. In contrast to the complexes $\Delta_{n,i}$ for $i \leq 3$ these complexes exhibit torsion. Strikingly, the torsion seems to be 3-torsion only.

Indeed Shareshian and Wachs (see [41]) have confirmed this, but their proof still is based on some computer calculations with our package. We regard this a prototype application of our package: Run examples, Get a Conjecture, Prove the Conjecture. But in this case the computer calculations are not yet out of the the picture. Going along with this philosophy Wachs [41] poses the problem to eliminate the computer calculations from their proof.

6.2 Lie Algebra Homology

In this section we show that the computational tools we have implemented are not restricted to calculating homology of simplicial complexes, but can be applied to computation of the homology of more general algebraic complexes. As an example we here consider the homology of finite dimensional Lie algebras. Let L be a Lie algebra of finite dimension over the field k. Consider the following map:

$$\partial_i : \bigwedge^{i} L \to \bigwedge^{i-1} L$$

defined by

$$\partial_i(a_1 \wedge \cdots \wedge a_i) = \sum_{1 \le l < j \le i} (-1)^{l+j} [a_j, a_j] \wedge a_1 \wedge \cdots \wedge \widehat{a_l} \wedge \cdots \wedge \wedge a_j \wedge \cdots \wedge a_i.$$

Then $\partial_i \circ \partial_{i+1} = 0$ and $\mathtt{Im}\partial_{i+1} \subseteq \mathtt{Ker}\partial_i$. Therefore, we can define a homology group

$$\widetilde{H}_i(L) = \mathtt{Ker}\partial_i / \Im\partial_{i+1}.$$

The definition makes sense for arbitrary Lie algebras L over a field k. If L is finite-dimensional as a vector-space over k then each $\bigwedge^i L$ is a finite-dimensional vectorspace over k and hence the computation of $\widetilde{H}_i(L)$ reduces to the computation of ranks of matrices.

In this section we report about experiments run on Lie algebras associated with finite partially ordered sets. Let us consider $P = \{p_1, \ldots, p_n\}$, a finite partially ordered set with order relation $\prec$, and k a field. We denote by $N_k(P)$ the set of matrices $N = (n_{ij}) \in k^{n \times n}$ such that $n_{ij} \neq 0$ implies $p_i \prec p_j$ (strict inequality). If we assume that $p_i \prec p_j$ implies $i \le j$ then $N_k(P)$ is the set of lower triangular nilpotent matrices with 0's in places where "there is no order relation in P." Is is easy to see that $N_k(P)$ with $[A, B] = AB - BA$ is a Lie algebra. The dimension of $\dim N_k(P)$ is given by the number of strict order relations in P. For example if $P = C(n)$ is a chain $C(n) = \{p_1 \prec p_1 \prec \cdots \prec p_n\}$ then $N_k(C(n))$ is the set of all lower triangular nilpotent matrices and $\dim N_k(C(n)) = \binom{n}{2}$. If $\mathtt{char}\ k = 0$ then a well known result by Kostant [30] gives complete information on the homology groups. Here we denote by $\mathtt{Inv}(\sigma)$ the set of inversions of a permutation σ; that is the set of pairs (i, j) where $1 \le i < j \le n$ and $\sigma(j) < \sigma(i)$.

Theorem 6.2 (Kostant [30]). *Let k be field of characteristic 0. Then:*

$$\dim_k H_i(N_k(C(n)) = \#\{\sigma \in S_n \mid \#\mathtt{Inv}(\sigma) = i\}.$$

It is also well known that there is the same number of permutations in S_n with i inversions as with $\binom{n}{2} - i$ inversions. This is an incarnation of Poincaré duality for Lie algebra homology (see [29, Ch. VII 9]).

Theorem 6.3. *Let L be a finite dimensional Lie algebra of dimension $d = \dim_k L$. Then $\dim_k H_i(L) = \dim_k H_{d-i}(L)$, $0 \le i \le d$.*

Thus in oder to calculate the homology of a finite dimensional Lie algebra it suffices to calculate the homology groups half the dimension up. There are two challenging problems that arise for Lie algebras $N_k(P)$:

▷ Express the dimension $\dim_k H_k(N_k(P))$ in terms of P when k is a field of characteristic 0.
▷ Express the dependency of $\dim_k H_k(N_k(P))$ on the characteristic of the field k.

For the latter question integer coefficients become of interest. If we consider the same differential with coefficients in the integers, calculating homology again amounts to computing Smith Normal Form of the matrices of the differentials. For $P = C(n)$ we observe the following picture:

The ranks of the free parts obey Poincare-duality and as usual with integer coefficients the torsion groups obey Poincare-duality in homological dimension shifted by 1.

We do not know an interpretation for the torsion, but we think that this is an interesting question to construct a combinatorial model determining the torsion coefficients.

There is a result by Dwyer [12] which says that p^l torsion occurs in the homology of $N_{\mathbb{Z}}(C(n))$ only if $p \leq n-2$. Moreover, for $p \leq n-2$ there is p-torsion in $N_{\mathbb{Z}}(C(n))$.

Thus for both questions raised above and their analogs over the integers a variant of our software can help to calculate examples and infer conjectures.

As an example we present values for the case when $P = B_n$ is the Boolean algebra of all subsets of $[n]$.

7 Other Invariants of Simplicial Complexes

Algebraic topology has developed many other invariants of topological spaces that can also be looked at in the case of simplicial complexes. We just want to comment on a few of them which we consider the most prominent.

Cohomology. The calculation of the cohomology $\widetilde{H}^i(\Delta; R)$ of a simplicial complex Δ reduces to the same problems as for homology. Note that the boundary matrices for cohomology are, when choosing the canonical bases, the transpose of the boundary matrices for homology.

Explicit cycles and co-cycles. More challenging is the problem of actually calculating explicit generators for the homology and cohomology groups. Thus the goal is to find a set of elements of $C_i(\Delta; R)$ that are representatives of a minimal generating set of $\widetilde{H}_i(\Delta; R)$ (resp. $\widetilde{H}^i(\Delta; R)$). Representatives of elements of $\widetilde{H}_i(\Delta; R)$ (resp. $\widetilde{H}^i(\Delta; R)$) are called cycles (resp. co-cycles). If we

Table 4. Lie Algebra homology for $P = C(n)$

$i\backslash n$	2	3	4	5	6	7
0	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$
1	$\mathbb{Z}$	$\mathbb{Z}^2$	$\mathbb{Z}^3$	$\mathbb{Z}^4$	$\mathbb{Z}^5$	$\mathbb{Z}^6$
2		$\mathbb{Z}^2$	$\mathbb{Z}^5 \oplus \mathbb{Z}_2$	$\mathbb{Z}^9 \oplus \mathbb{Z}_2^2$	$\mathbb{Z}^{14} \oplus \mathbb{Z}_2^3$	$\mathbb{Z}^{20} \oplus \mathbb{Z}_2^4$
3		$\mathbb{Z}$	$\mathbb{Z}^6 \oplus \mathbb{Z}_2$	$\mathbb{Z}^{15} \oplus \mathbb{Z}_2^8 \oplus \mathbb{Z}_3^2$	$\mathbb{Z}^{29} \oplus \mathbb{Z}_2^{20} \oplus \mathbb{Z}_3^4$	$\mathbb{Z}^{49} \oplus \mathbb{Z}_2^{35} \oplus \mathbb{Z}_3^6$
4			$\mathbb{Z}^5$	$\mathbb{Z}^{20} \oplus \mathbb{Z}_2^{10} \oplus \mathbb{Z}_3^3$	$\mathbb{Z}^{49} \oplus \mathbb{Z}_2^{47} \oplus$ $\mathbb{Z}_4^3 \oplus \mathbb{Z}_3^{13}$	$\mathbb{Z}^{98} \oplus \mathbb{Z}_2^{124} \oplus$ $\mathbb{Z}_4^6 \oplus \mathbb{Z}_3^{21}$
5			$\mathbb{Z}^3$	$\mathbb{Z}^{22} \oplus \mathbb{Z}_2^{10} \oplus \mathbb{Z}_3^3$	$\mathbb{Z}^{71} \oplus \mathbb{Z}_2^{79} \oplus$ $\mathbb{Z}_4^9 \oplus \mathbb{Z}_3^{26}$	$\mathbb{Z}^{169} \oplus \mathbb{Z}_2^{303} \oplus \mathbb{Z}_4^{28} \oplus$ $\mathbb{Z}_3^{78} \oplus \mathbb{Z}_5^4$
6			$\mathbb{Z}$	$\mathbb{Z}^{20} \oplus \mathbb{Z}_2^8 \oplus \mathbb{Z}_3^2$	$\mathbb{Z}^{90} \oplus \mathbb{Z}_2^{118} \oplus$ $\mathbb{Z}_4^{12} \oplus \mathbb{Z}_3^{35}$	$\mathbb{Z}^{259} \oplus \mathbb{Z}_2^{635} \oplus \mathbb{Z}_4^{65} \oplus$ $\mathbb{Z}_3^{168} \oplus \mathbb{Z}_5^{17}$
7				$\mathbb{Z}^{15} \oplus \mathbb{Z}_2^2$	$\mathbb{Z}^{101} \oplus \mathbb{Z}_2^{138} \oplus$ $\mathbb{Z}_4^{12} \oplus \mathbb{Z}_3^{36}$	$\mathbb{Z}^{359} \oplus \mathbb{Z}_2^{1122} \oplus \mathbb{Z}_4^{112} \oplus$ $\mathbb{Z}_3^{275} \oplus \mathbb{Z}_5^{39}$
8				$\mathbb{Z}^9$	$\mathbb{Z}^{101} \oplus \mathbb{Z}_2^{118} \oplus$ $\mathbb{Z}_4^{12}\mathbb{Z}_3^{35}$	??
9				$\mathbb{Z}^4$	$\mathbb{Z}^{90} \oplus \mathbb{Z}_2^{79} \oplus$ $\mathbb{Z}_3^{26} \oplus \mathbb{Z}_4^9$	??
10				$\mathbb{Z}$	$\mathbb{Z}^{71} \oplus \mathbb{Z}_2^{47} \oplus$ $\mathbb{Z}_4^3 \oplus \mathbb{Z}_3^{13}$	??
11					$\mathbb{Z}^{49} \oplus \mathbb{Z}_2^{20} \oplus \mathbb{Z}_3^9$	??
12					$\mathbb{Z}^{29} \oplus \mathbb{Z}_2^3$	??
13					$\mathbb{Z}^{14}$	??
14					$\mathbb{Z}^5$	$\mathbb{Z}^{359} \oplus \mathbb{Z}_2^{635} \oplus \mathbb{Z}_4^{65} \oplus$ $\mathbb{Z}_3^{168} \oplus \mathbb{Z}_5^{17}$
15					$\mathbb{Z}$	$\mathbb{Z}^{259} \oplus \mathbb{Z}_2^{303} \oplus \mathbb{Z}_4^{28} \oplus$ $\mathbb{Z}_3^{78} \oplus \mathbb{Z}_5^4$
16						$\mathbb{Z}^{169} \oplus \mathbb{Z}_2^{124} \oplus$ $\mathbb{Z}_4^6 \oplus \mathbb{Z}_3^{27}$
17						$\mathbb{Z}^{98} \oplus \mathbb{Z}_2^{35} \oplus \mathbb{Z}_3^6$
18						$\mathbb{Z}^{49} \oplus \mathbb{Z}_2^4$
19						$\mathbb{Z}^{20}$
20						$\mathbb{Z}^6$
21						$\mathbb{Z}$

Table 5. Lie-Algebra homology for $P = B_n$

$i \backslash n$	1	2	3
0	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$
1		$\mathbb{Z}^4$	$\mathbb{Z}^{12}$
2		$\mathbb{Z}^5$	$\mathbb{Z}^{60}$
3		$\mathbb{Z}^5$	$\mathbb{Z}^{179} \oplus \mathbb{Z}_2$
4		$\mathbb{Z}^4$	$\mathbb{Z}^{486} \oplus \mathbb{Z}_2^{13}$
5		$\mathbb{Z}$	$\mathbb{Z}^{1026} \oplus \mathbb{Z}_2^{81} \oplus \mathbb{Z}_3^4$
6			$\mathbb{Z}^{1701} \oplus \mathbb{Z}_2^{250} \oplus \mathbb{Z}_4^6 \oplus \mathbb{Z}_3^{30} \oplus \mathbb{Z}_5^6$
7			$\mathbb{Z}^{2353} \oplus \mathbb{Z}_2^{643} \oplus \mathbb{Z}_4^{55} \oplus \mathbb{Z}_3^{118}$
8			$\mathbb{Z}^{2701} \oplus \mathbb{Z}_2^{1094} \oplus \mathbb{Z}_4^{38} \oplus \mathbb{Z}_3^{141} \oplus \mathbb{Z}_5^{12}$
9			$\mathbb{Z}^{2773} \oplus \mathbb{Z}_2^{1350} \oplus \mathbb{Z}_4^{48} \oplus \mathbb{Z}_3^{268} \oplus \mathbb{Z}_5^2$
10			$\mathbb{Z}^{2773} \oplus \mathbb{Z}_2^{1094} \oplus \mathbb{Z}_4^{38} \oplus \mathbb{Z}_3^{141} \oplus \mathbb{Z}_5^{12}$
11			$\mathbb{Z}^{2701} \oplus \mathbb{Z}_2^{643} \oplus \mathbb{Z}_4^{55} \oplus \mathbb{Z}_3^{118}$
12			$\mathbb{Z}^{2353} \oplus \mathbb{Z}_2^{250} \oplus \mathbb{Z}_4^6 \oplus \mathbb{Z}_3^{30} \oplus \mathbb{Z}_5^6$
13			$\mathbb{Z}^{1701} \oplus \mathbb{Z}_2^{81} \oplus \mathbb{Z}_3^4$
14			$\mathbb{Z}^{1026} \oplus \mathbb{Z}_2^{13}$
15			$\mathbb{Z}^{486} \oplus \mathbb{Z}_2$
16			$\mathbb{Z}^{179}$
16			$\mathbb{Z}^{60}$
17			$\mathbb{Z}^{12}$
18			$\mathbb{Z}$

analyze our algorithms then we see that elimination in principle can achieve this goal but with an enormous extra amount of memory consumption – we have to store transformations that are applied in order to get Smith Normal Form (see [24] for worst case considerations of the integer linear algebra problems involved in a solution of the problem). Note that the Valence algorithm will not even be able to give us explicit generators.

Cohomology algebra. One can impose on the direct sum of Abelian groups $H^*(\Delta; \mathbb{Z}) = \bigoplus_{i \geq 0} H^i(\Delta; \mathbb{Z})$ naturally the structure of an algebra [32, §48] (note that here we use non-reduced cohomology, which differs from reduced cohomology only by a copy of $\mathbb{Z}$ in dimension 0). Once explicit cocycles are known a presentation of this algebra can be given using linear algebra only. Applying Gröbner base techniques even allows standardized presentations and calculations of algebra invariants. Even though there is not problem in theory the explicit calculations seem feasible only for very small examples. On the other hand the importance of the cohomology algebra makes it a challenging goal to find efficient algorithms for its computation.

Fundamental group. Let us assume in the following that the simplicial complex Δ is connected. The fundamental group of Δ is a finitely presented group $\pi_1(\Delta)$. Conversely it is easily seen that every finitely presented group G can occur as the fundamental group of a simplicial complex Δ (see [33, Exercise 2, p. 445]). For a simplicial complex a presentation of the fundamental group can be efficiently constructed. But results by Boone [3] and Novikov [34] show that the word problem for such groups is not decidable. Thus the fundamental group is an invariant which we can easily give a presentation for but whose properties are in general not computable. Nevertheless, there are results that show that algorithms give results in nice cases [35]. In this paper Rees and Soicher also use the fundamental group to give an algorithm for calculating the first homology group with coefficients in a field of characteristic $p > 0$. They use the well know fact that the Abelianization of $\pi_1(\Delta)$ is isomorphic to $\widetilde{H}_1(\Delta; \mathbb{Z})$.

References

1. Eric Babson, Anders Björner, Svante Linusson, John Shareshian and Volkmar Welker Complexes of not i-connected graphs. *Topology* **38** (1999) 271-299.
2. Achim Bachem and Ravindran Kannan. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. *SIAM J. Comput.* **8** 499–507 (1979).
3. William W. Boone. Certain simple unsolvable problems in group theory. Indig. Math. **16** 231-237 (1955).
4. Alfred Brauer. Limits for the characteristic roots of a matrix. I. *Duke Math. J.* **13** 387–395 (1946).
5. Alfred Brauer. Limits for the characteristic roots of a matrix. II. *Duke Math. J.* **14** 21–26 (1947).
6. Richard A. Brualdi and Stephen Mellendorf. Regions in the complex plane containing the eigenvalues of a matrix. *American Mathematical Monthly* **101** 975–985 (1994).
7. Antonio Capani, Gianfranco Niesi and Lorenzo Robbiano, *CoCoA, a system for doing Computations in Commutative Algebra.* Available via anonymous ftp from: `cocoa.dima.unige.it`
8. David Eisenbud. *Commutative Algebra with a View Toward Algebraic Geometry*, Springer, 1993.
9. Jean-Guillaume Dumas, Frank Heckenbach, B. David Saunders and Volkmar Welker. *Simplicial Homology, a (proposed) share package for GAP*, March 2000. Manual (`http://www.cis.udel.edu/~dumas/Homology`).
10. Jean-Guillaume Dumas, B. David Saunders and Gilles Villard. On efficient sparse integer matrix Smith normal form computations. *J. Symb. Comp.* **32** 71–99 (2001).
11. Pierre Dusart. *Autour de la fonction qui compte le nombre de nombres premiers.* PhD thesis, Université de Limoges, 1998.
12. William G. Dwyer. Homology of integral upper-triangular matrices. Proc. Amer. Math. Soc. **94** 523–528 (1985).
13. Wayne Eberly and Erich Kaltofen. On randomized Lanczos algorithms. In Wolfgang W. Küchlin, editor, *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, Maui, Hawaii*, pages 176–183. ACM Press, New York, July 1997.

14. Michael L. Fredman and Leonid Khachiyan. On the complexity of dualization of monotone disjunctive normal forms, *J. Algorithms* **21** (1996) 618-628.
15. The GAP Group, GAP — Groups, Algorithms, and Programming, Version 4.2; 2000. (`http://www.gap-system.org`)
16. Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra.* Cambridge University Press, New York, NY, USA, 1999.
17. Mark W. Giesbrecht. *Probabilistic Computation of the Smith Normal Form of a Sparse Integer Matrix.* Lecture Notes in Comp. Sci. 1122, 173–186. Springer, 1996.
18. Gene H. Golub and Charles F. Van Loan. *Matrix computations.* Johns Hopkins Studies in the Mathematical Sciences. The Johns Hopkins University Press, Baltimore, MD, USA, third edition, 1996.
19. Daniel R. Grayson and Michael E. Stillman, *Macaulay 2, a software system for research in algebraic geometry,* Available at `http://www.math.uiuc.edu/Macaulay2/`
20. Melvin Hochster. Cohen-Macaulay rings, combinatorics, and simplicial complexes. *Ring Theory II, Proc. 2nd Okla. Conf. 1975.* Lecture Notes in Pure and Applied Mathematics. Vol. 26. New York. Marcel Dekker. 171-223 (1977).
21. Phil Hanlon. A survey of combinatorial problems in Lie algebra homology. Billera, Louis J. (ed.) et al., *Formal power series and algebraic combinatorics.* Providence, RI: American Mathematical Society. DIMACS, Ser. Discrete Math. Theor. Comput. Sci. 24, 89-113, 1996.
22. Iztok Hozo, *Inclusion of poset homology into Lie algebra homology.* J. Pure Appl. Algebra **111** 169-180 (1996).
23. Costas S. Iliopoulos. Worst case bounds an algorithms for computing the canonical structure of finite Abelian groups and the Hermite and Smith normal forms of an integer matrix. *SIAM J. Comput.* **18** (1989) 658-669.
24. Costas S. Iliopoulos. Worst case bounds an algorithms for computing the canonical structure of infinite Abelian groups and solving systems of linear diophantine equations. *SIAM J. Comput.* **18** (1989) 670-678.
25. Gil Kalai. Algebraic Shfiting. *Computational Commutative Algebra and Combinatorics* Advanced Studies in Pure Math., Vol. 33. Tokyo. Math. Soc. of Japan. 121-165 (2002).
26. Erich Kaltofen, Wen-Shin Lee and Austin A. Lobo. Early termination in Ben-Or/Tiwari sparse interpolation and a hybrid of Zippel's algorithm. In Carlo Traverso, editor, *Proceedings of the 2000 International Symposium on Symbolic and Algebraic Computation, Saint Andrews, Scotland,* pages 192–201. ACM Press, New York, 2000.
27. Volker Kaibel and Marc E. Pfetsch. Some algorithmic problems in polytope theory, this volume, pages 23–47.
28. Dimitris Kavvadias and Elias Stavroploulos. Evaluation of an algorithm for the transversal hypergraph problem. in: *Proc. 3rd Workshop on Algorithm Engineering (WAE'99),* Lecture Notes in Comp. Sci. 1668, Springer, 1999.
29. Anthony W. Knapp. *Lie Groups, Lie Algebras, and Cohomology,* Mathematical Notes **34**, Princeton University Press, 1988.
30. Bertram Kostant. Lie algebra cohomology and the generalized Borel-Weil theorem, *Ann. Math. (2)* **74** (1961) 329-387.
31. Robert H. Lewis. *Fermat, a computer algebra system for polynomial and matrix computations,* 1997. `http://www.bway.net/~lewis`.

32. James R. Munkres, *Elements of Algebraic Topology*, Addison-Wesley, Menlo Park, 1984.
33. James R. Munkres, *Topology*, 2nd Edition, Prentice Hall, 2000.
34. Sergey P. Novikov. On the algorithmic unsolvability of the word problem in group theory, *Trudy Mat. Inst.* **44** (1955) 143.
35. Sarah Rees and Leonard H. Soicher An algorithmic approach to fundamental groups and covers of combinatorial cell complexes *J. of Symb. Comp.* **29** (2000) 59-77.
36. Ernest Sibert, Harold F. Mattson and Paul Jackson. Finite Field Arithmetic Using the Connection Machine. In Richard Zippel, editor, *Proceedings of the second International Workshop on Parallel Algebraic Computation, Ithaca, USA*, volume 584 of *Lecture Notes in Computer Science*, 51–61. Springer, 1990.
37. Arne Storjohann, Near Optimal Algorithms for Computing Smith Normal Forms of Integer Matrices, Lakshman, Y. N. (ed.), *Proceedings of the 1996 international symposium on symbolic and algebraic computation, ISSAC '96*, New York, NY: ACM Press. 267-274 (1996).
38. Olga Taussky. Bounds for characteristic roots of matrices. *Duke Math. J.* **15** 1043–1044 (1948).
39. Vladimir Tuchin. Homologies of complexes of doubly connected graphs. Russian Math. Surveys **52** 426–427 (1997).
40. Richard S. Varga. *Matrix iterative analysis.* Number 27 in Springer series in Computational Mathematics. Springer, second edition, 2000.
41. Michelle Wachs. Topology of matching, chessboard and general bounded degree graph complexes. Preprint 2001.
42. Douglas H. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theory* **32** 54–62 (1986).