

Overview of Magma V2.8 Features

1 Introduction

1.1 The Magma Philosophy

Magma is a Computer Algebra system designed to solve problems in algebra, number theory, geometry and combinatorics that may involve sophisticated mathematics and which are computationally hard. Magma provides a mathematically rigorous environment which emphasizes *structural* computation. A key feature is the ability to construct canonical representations of structures, thereby making possible such operations as membership testing, the determination of structural properties and isomorphism testing. The kernel of Magma contains implementations of many of the important concrete classes of structure in five fundamental branches of algebra, namely group theory, ring theory, field theory, module theory and the theory of algebras. In addition, certain families of structures from algebraic geometry and finite incidence geometry are included.

The main features of the Magma system include:

- (a) **Algebraic Design Philosophy:** The design principles underpinning both the user language and system architecture are based on ideas from universal algebra and category theory. The language attempts to approximate as closely as possible the usual mathematical modes of thought and notation. In particular, the principal constructs in the user language are set, (algebraic) structure and morphism.
- (b) **Universality:** In-depth coverage of all the major branches of algebra, number theory, algebraic geometry and finite incidence geometry.
- (c) **Integration:** The facilities for each area are designed in a similar manner using generic constructors wherever possible. The uniform design makes it a simple matter to program calculations that span different classes of mathematical structures or which involve the interaction of structures.
- (d) **Performance:** The intention is that Magma provide the best possible performance both in terms of the algorithms used and their implementation. The design philosophy permits the kernel implementor to choose optimal data structures at the machine level. Most of the major algorithms currently installed in the Magma kernel are state-of-the-art and give performance similar to, or better than, specialized programs.

The purpose of this document is to provide an overview of the structures and operations that are implemented in Magma. A collection of illustrative examples may be found at the Magma home page: <http://www.maths.usyd.edu.au:8000/u/magma/> – also available from this web site are a number of short papers written by experts describing applications of Magma in various branches of mathematics.

1.2 Summary of this Document

Computation in Magma always takes place in one or more explicitly defined structures. A family of structures whose members satisfy a common set of axioms and which share a common representation is termed a *category*. In the first of the following sections we summarise the facilities of the Magma language and environment. Following this we summarise many of the kernel categories grouped together under the broad headings listed below.

- The Magma Language and System
- Groups
- Lie Theory
- Semigroups and Monoids
- Rings and their Fields
- Commutative Rings
- Linear Algebra and Module Theory
- Lattices and Quadratic Forms
- Algebras
- Representation Theory
- Homological Algebra
- Algebraic Geometry
- Finite Incidence Geometry
- Error-correcting Codes
- Cryptography
- Optimization
- Mathematical Databases

All timings given below are for a Sun 400Mhz UltraSPARC 2 unless otherwise indicated.

2 The Magma Language and System

2.1 The Magma User Language

- Imperative language with standard imperative-style statements and procedures
- A functional subset providing closures, higher-order functions, and partial evaluation
- General aggregate data types based on algebraic notions: set, sequence, mapping, magma
- Universal structure constructors providing a general mechanism for the construction of magmas and mappings
- Simple but powerful notation for constructing sets and sequences in a natural mathematical style
- Set and sequence operations which are implemented with a strong emphasis on efficiency
- Coercion between magmas (including automatic coercion)
- A package mechanism to support modular program construction

2.2 The Magma Environment

- Command completion and interactive line editing
- History system with recall and editing of previous lines
- A hierarchical online help facility
- Packages containing user-defined intrinsics with automatic compilation
- Command-line options at startup
- Environment variables for configuring style of output, etc.
- Get/set functions and procedures for configuring style, etc.
- Verbose options for built-in functions
- Logging of output, redirection of I/O
- Mechanism for saving and restoring user workspaces
- Special file type for fully-featured file I/O
- Ability to execute system commands from within Magma
- Input/output pipes for communication with external programs
- UNIX commands and functions such as process ID, alarm setting, etc.
- A socket mechanism for communicating with other processes (in preparation)

3 Semigroups and Monoids

3.1 Finitely Presented Semigroups

- Construction of fp-semigroups and monoids
- Direct product, free product
- Arithmetic (free reduction on words only)
- Definition of ideals and subsemigroups
- Tietze transformations

3.2 Monoids Defined by Rewrite Systems

This is a category of finitely presented monoids where the relations are interpreted as rewrite rules. The most important case is that in which the monoid is defined by a confluent system of rewrite rules. A monoid of this category is typically constructed by applying the Knuth-Bendix procedure to a finitely presented monoid. Magma uses the Knuth-Bendix developed by Derek Holt in his package *kbmag*.

- Construction of an RWS monoid from an fp-monoid using the Knuth-Bendix procedure. Orderings supported include: *RT-recursive*, *recursive*, *ShortLex*, *WT-ShortLex* and *Wreath*
- Test a rewrite system for confluence
- Reduction of a word to normal form
- Operations on words: Product, exponentiation, equality
- Test for a monoid being finite
- Enumeration of elements

4 Groups

Group theory has been one of the Computational Algebra Group's traditional strengths and Magma provides the user with access to nearly all of the significant algorithms for finite groups and finitely presented infinite groups (*fp-groups*). The important categories of group include:

- Permutation groups
- Matrix groups
- Finitely presented groups
- Generic abelian groups
- Finitely-presented abelian groups
- Polycyclic groups
- Soluble groups
- Finite p -groups
- Automorphism groups
- Groups defined by rewrite systems
- Automatic groups
- Groups with elements given as straight-line programs

In addition, Magma provides extensive machinery for the representation theory of groups which is discussed in the Representation Theory section of this document.

4.1 Permutation Groups

A permutation action may be defined on any finite set using a G -set mechanism. A huge range of permutation group algorithms (some 300) are incorporated—many of them being developed specifically for Magma. All algorithms are either deterministic or Las Vegas, that is if an answer is returned it is guaranteed correct but it is possible no answer will be returned.

4.1.1 Construction

- Permutation representations for classical groups, e.g. $\mathrm{PGL}(n, q)$, $\mathrm{PSp}(n, q)$, $\mathrm{PSU}(n, q^2)$, $P\Omega(n, q)$
- Construction of standard groups, e.g., S_n , A_n .
- Construction of wreath products with both types of action
- Random generation of elements (Cellar, Leedham-Green, Murray, Niemeyer, O'Brien algorithm)

4.1.2 Base and Strong Generating Set

- Sims-Schreier algorithm for constructing a base and strong generating set (BSGS)
- Random Schreier algorithm for constructing a BSGS
- Todd-Coxeter Schreier algorithm for constructing a BSGS
- Sims variation of Schreier method for soluble groups
- Fast construction of BSGS when group order is known
- Brownie-Cannon-Sims algorithm for verifying a BSGS

The key concept for representing a permutation group is that of a *base and strong generating set* (BSGS). Given a BSGS for a group, its order may be deduced immediately. Brownie, Cannon and Sims (1991) showed that it is practical, in some cases at least, to construct a BSGS for short-base groups having degree up to ten million. For example, starting with six permutations of degree

8 835 156, generating the Lyons simple group, it takes Magma 5.0 hours to provably determine the order of the group they generate.

The ability to construct a BSGS, coupled with the use of algorithms that make heavy use of the classification theorem for finite simple groups, allow the determination of a great deal of structural information, such as composition factors, for short base groups of degree up to at least a million.

4.1.3 Actions

- Stabilizer of a point, set of points, sequence of points
- Stabilizer of an ordered partition, conjugacy of partitions
- Orbits on points, sets of points, and sequences of points
- Homomorphisms induced by actions on orbits
- Systems of imprimitivity
- Homomorphisms induced by actions systems of imprimitivity
- Induced actions on G -sets
- Action properties: Semiregular, regular, transitive, primitive, Frobenius
- Fast tests for the alternating/symmetric group

4.1.4 Subgroup Constructions

- Construction of subgroups, quotient groups
- Normal closure, core of a subgroup
- Normalizer, centralizer
- Testing subgroups for conjugacy
- Intersection of subgroups
- Sylow p -subgroup (reduction algorithm)
- Permutation representation on the cosets of a subgroup

A key application of our work on the O’Nan-Scott decomposition has been a Sylow algorithm that reduces the problem to computing Sylow subgroups of simple groups. Though this work is not complete (the algorithm involves many complex stages), we have succeeded in computing Sylow subgroups in short-base groups having degree up to 500 000.

4.1.5 Analysis of a Primitive Group

- Elementary abelian regular normal subgroup (EARNs)
- Action of an affine primitive group on its EARNs
- Construction of the socle of a non-affine group via the O’Nan-Scott theorem
- Action of a primitive group on its socle
- Permutation representation of G/N , where G is a primitive group and N is its socle
- O’Nan-Scott decomposition of a primitive group
- Identification of a 2-transitive group

The Magma group has developed efficient methods for obtaining the O’Nan-Scott decomposition of a primitive group. The elementary abelian regular normal subgroup of an affine primitive group is constructed by a polynomial-time algorithm based on ideas published by P. Neumann. For example, Magma finds the EARNs of $AGL(10, 3)$ which has degree 59,049 and order 17046196453240220939126401085378073952125928970649600 in 36 seconds. The construction of the socle and the analysis of a non-affine primitive group is performed by algorithms based on ideas of Cannon, Holt and Kantor. A 2-transitive group is identified as an abstract group using an algorithm published by Cameron and Cannon.

4.1.6 Normal Structure

- Derived subgroup, derived series, soluble residual
- Lower central series, nilpotent residual
- p -core, Fitting subgroup, soluble radical (polynomial-time algorithm)
- Centre, upper central series
- Elementary abelian series, p -central series
- Socle, socle action
- Chief series, chief factors, composition factors
- Agemo, omega subgroups (of a p -group)
- Minimal normal subgroups
- Maximal normal subgroups
- All normal subgroups

4.1.7 Standard Quotients

- Maximal abelian quotient
- Maximal soluble quotient
- Socle quotient (for primitive and trivial Fitting groups)
- Conjugation action on socle factors (trivial Fitting groups)
- Quotient by an abelian normal subgroup
- Radical quotient
- Presentation of quotient by a normal subgroup
- Presentation on given generators (for groups of moderate order)

A variety of methods are used for quotient constructions. Quotients by abelian and solvable groups use a construction due to Easdown and Praeger (1988). The maximal soluble quotient uses an algorithm of Holt, while other finitely presented quotients use the Schreier-Todd-Coxeter-Sims method of Leon (1980).

4.1.8 Conjugacy Classes

- Testing a pair of elements for conjugacy
- Conjugacy classes of elements (lifting algorithm)
- Power map
- Class map, i.e. return the number of the class to which a given element belongs
- Class matrix
- Exponent

The conjugacy classes of elements of a group G are found using a lifting algorithm which first finds the classes of the trivial Fitting quotient of G and then lifts these classes through the layers of an elementary abelian series for G . For example, all classes in the group $2^{12} \cdot (SL_2(4) \wr S_3)$ of degree 4096 and order 5,308,416,000 are computed in 4 seconds.

4.1.9 Subgroup Structure

- Maximal subgroups (new Holt algorithm)
- Frattini subgroup
- Conjugacy classes of complements of a soluble normal subgroup
- Conjugacy classes of subgroups, poset of subgroup classes
- Conjugacy classes of subgroups satisfying a condition: Cyclic, elementary abelian, abelian, nilpotent
- Low index subgroups

Magma V2.8 contains an implementation of a very powerful algorithm, due to Derek Holt, for computing the maximal subgroups of a group G of moderate degree. The algorithm first determines the maximal subgroups of the trivial Fitting quotient of G and then lifts these maximal subgroups down an elementary abelian series. Magma finds the maximal subgroups of the group of Rubik's cube (degree 48, order 43252003274489856000) in 2.4 seconds.

4.1.10 Automorphisms

- Automorphism group (new algorithm)
- Test for two permutation groups being isomorphic

The automorphism group of a permutation group G is found using a lifting algorithm which first finds the automorphisms of the trivial Fitting quotient of G by looking up the automorphism groups of any non-cyclic factors in a database. Then these automorphisms are lifted through the layers of an elementary abelian series for G . For example, the automorphism group of the group of Meffert's puzzle (a non-soluble permutation group of degree 30 and order $2^8 3^{10} 5$) is found in 4 seconds. The group has 16 outer automorphisms.

4.1.11 Cohomology and Representations

- Character table
- Irreducible representations (for groups of moderate order)
- KG -module corresponding to an elementary abelian section
- p -part of Schur multiplier, p -cover
- Dimensions of first and second cohomology groups
- Split and non-split extensions of a group by a module (D. Holt's package)

4.1.12 Databases

- Transitive groups up to degree 22 (Butler-Hulpke)
- Primitive groups up to degree 50 (Sims)
- Irreducible soluble subgroups of $GL(n, p)$ for $p^n < 256$ (Short)
- Almost simple groups of order less than 1.6×10^8 stored with their automorphism groups and maximal subgroups
- A collection of permutation representations of some sporadic simple groups

4.2 Matrix Groups

Matrix groups may be defined over any ring over which the echelonization of matrices is possible. For example, matrix groups may be defined over function fields $K(x)$. The matrix group facilities are mainly restricted to finite groups since there are, as yet, few algorithms of general interest known for infinite groups. Techniques for working with finite matrix groups divide into methods for groups of small degree and methods for groups of large degree.

4.2.1 Construction

- Generators for linear groups: $GL(n, q)$, $SL(n, q)$
- Generators for symplectic groups: $Sp(n, q)$
- Generators for unitary groups: $GU(n, q)$, $U(n, q)$
- Generators for orthogonal groups: $GO(2n+1, q)$, $SO(2n+1, q)$, $\Omega(2n+1, q)$, $GO^+(2n, q)$, $SO^+(2n, q)$, $\Omega^+(2n, q)$, $GO^-(2n, q)$, $SO^-(2n, q)$, $\Omega^-(2n, q)$
- Generators for all exceptional families of groups of Lie type except $E(8)$.
- Direct product, tensor wreath product, tensor power
- Construction of semi-linear groups

4.2.2 Arithmetic with Elements

- Random generation of elements (Cellar, Leedham-Green, Murray, Niemeyer, O'Brien algorithm)
- Order of a matrix (Leedham-Green algorithm for finite fields)
- Test whether a matrix has infinite order

4.2.3 Actions

- Tests for irreducibility, absolute irreducibility, semi-linearity
- Test whether a group over a field of characteristic zero has infinite order
- Orbit, stabilizer of a vector or subspace
- Homomorphism induced by action of a reducible group on G -invariant submodule and its quotient module
- Homomorphism induced by action on an orbit of vectors or subspaces

4.2.4 Subgroup Constructions

- Base and strong generating set: Random Schreier algorithm, Todd-Coxeter Schreier algorithm, Murray-O'Brien base selection strategy
- Construction of subgroups
- Normal closure, core of a subgroup
- Centralizer
- Intersection of subgroups
- Sylow p -subgroup (reduction algorithm)
- Homomorphism induced by action on the cosets of a subgroup

For matrix groups of small degree, we use an analogue of the methods used for permutation groups. We try to find some sequences of objects (subspaces and vectors) in the underlying vector space that defines a stabilizer chain which has the property that the basic orbits are not excessively large. Thus, we have a concept of a base and strong generating set (BSGS) similar to that employed in the case of permutation groups. Once such a BSGS is available, analogues of the permutation group backtrack searches for centralizer, normalizer etc may be described.

4.2.5 Conjugacy Classes

- Testing a pair of elements for conjugacy
- Conjugacy classes of elements
- Power map
- Class map, i.e. return the number of the class to which a given element belongs
- Class matrix
- Exponent

4.2.6 Normal Structure

- Derived subgroup, abelian quotient
- Soluble residual, soluble quotient
- Centre, Fitting subgroup
- Derived series, upper central series, lower central series
- Elementary abelian series, p -central series
- Composition series, composition factors, chief series
- Agemo, omega subgroups (of a p -group)
- Jennings series (of a p -group)

4.2.7 Cohomology and Representations

- Character table
- KG -module corresponding to an elementary abelian section
- Molien series
- Ring of invariants
- Presentation on given generators (for groups of moderate order)

4.2.8 Aschbacher Analysis

- Determine whether a group preserves a form modulo scalars.
- The Niemeyer-Prager classical group recognition algorithm as implemented in Magma by Alice Niemeyer and Anthony Pye.
- Determine whether a subgroup G of $GL(d, q)$ acts imprimitively on the underlying vector space. a block system, respectively.
- Test whether a matrix group G acts as a semilinear group of automorphisms on some vector space.
- Test whether a matrix group G preserves a non-trivial tensor product decomposition.
- Search for decompositions (corresponding to certain Aschbacher families) with respect to the normal closure of a supplied subgroup.
- The Glasby-Howlett algorithm to decide if the absolutely irreducible group $G \leq GL(d, K)$ has an equivalent representation over a subfield of K .
- Given a group G of $d \times d$ matrices over a finite field E having degree e and a subfield F of E having degree f , write G as a group generated by the matrices of G written as $de/f \times de/f$ matrices over F .

The basic facilities provided by Magma for computing with matrix groups over finite fields depend upon being able to construct a chain of stabilizers. However, there are many examples of groups of moderately small degree where we cannot find a suitable chain. An on-going international research project seeks to develop algorithms to explore the structure of such groups. The main theoretical underpinning of the project comes from the classification by Aschbacher (1984) of the (maximal) subgroups of $GL(d, q)$ into nine families. Much of the research effort to date has been devoted to designing algorithms to decide whether G belongs to one of the eight families whose members have a normal subgroup preserving a “natural linear structure”; here, we plan to exploit this information to explore G further, ultimately producing a composition series for G .

4.3 Finitely Presented Groups

Given a finitely presented group (fp-group) about which nothing is known, the immediate problems are to determine whether it is trivial, finite, infinite, free etc. and to determine its finite homomorphic images, finite index subgroups and so on. The central strategy for analyzing an fp-group is to attempt to construct non-trivial homomorphisms, which may be onto an abelian group, p -group, nilpotent group, soluble group, permutation group (the Todd-Coxeter algorithm) or matrix group (vector enumeration).

4.3.1 Free Groups

- Construction
- Reduction of a word to normal form
- Product, exponentiation, inverse, equality

4.3.2 Construction and Arithmetic

- Construction as a quotient of a free group
- Standard groups as fp-groups
- Permutation groups, matrix groups, polycyclic groups as fp-groups
- Direct product, free product
- Maximal central extension
- Arithmetic (free reduction only on words)
- Substring operations on words
- Tietze transformations
- Definition of and calculation with homomorphisms

4.3.3 Subgroup Methods

- Construction of a subgroup
- Normal closure
- Coset enumeration (Todd-Coxeter procedure)
- Process version of coset enumeration allowing the user complete control over its execution
- Transversal and transversal map for a subgroup of finite index (SFI)
- Membership and equality for subgroups of finite index
- Core, intersection and normalizer for SFI
- Properties of SFI: Conjugacy, maximality, normality
- Enumeration of low index subgroups (Sims backtrack algorithm)
- Process version of low index subgroups to return subgroups one at a time
- Schreier generators for a subgroup
- Presentation for a subgroup (Reidemeister-Schreier rewriting)
- Simplification of a presentation (Havas-Lian algorithm)
- Process version of presentation simplification allowing the user complete control over its execution

Coset enumeration is performed using the Todd-Coxeter procedure used by Magma is an unpublished version due to George Havas and has the capability of enumerating up to ten million cosets on a sufficiently large machine. Subgroups of small index may be enumerated using the so-called *low index subgroups* algorithm. The low index algorithm used in Magma is the backtrack method described by Sims in his book *Computation in Finitely Presented Groups*, CUP, 1993. The Magma fp-group package also includes a collection of functions for computing with subgroups of (small) finite index represented by coset tables.

4.3.4 Quotient Group Methods

- Abelian quotient algorithm
- p -quotient algorithm
- Process version of p -quotient allowing the user complete control over its execution
- Nilpotent quotient algorithm (W. Nickel's algorithm)
- Soluble quotient algorithm (Plesken-Brückner algorithm)
- Process version of the soluble quotient allowing the user complete control over its execution
- Actions on coset spaces (Todd-Coxeter procedure)
- Actions on vector spaces (Linton vector enumeration)
- Permutation representation on the cosets of a subgroup

The p -quotient program has been developed over a number of years by a group at ANU that includes George Havas, Mike Newman and Eamonn O'Brien. It has been used to construct p -quotients of composition length several thousand for small primes p . Soluble quotients are computed using Herbert Brückner's implementation of the Plesken algorithm and is capable of constructing soluble quotients having order in excess of a million. Unlike previous algorithms, no information is required other than the fp-group.

4.4 Generic Abelian Groups

A generic abelian group is a set whose elements form an abelian group with respect to a given law of composition. The user specifies the set A together with functions for composing two elements of A , constructing the inverse of an element of A , and recognizing the identity element of A .

- Definition as a set with given operations
- Arithmetic
- Random elements
- Order of an element: Baby-step giant-step algorithm; Pollard-rho algorithm
- Discrete logarithm: Baby-step giant-step algorithm; Pohlig-Hellman algorithm; Pollard-rho algorithm
- Order of the group
- Generating set and presentation
- Torsion invariants
- Construction of subgroups from generators
- Sylow p -subgroup
- Homomorphisms and isomorphisms

The three major calculations supported are: find the order of an element, compute the discrete logarithm of an element relative to a given base and determine the structure of the group. The algorithms used are improvements of those described in J. Buchmann, M.J. Jacobson and E. Teske [4].

4.5 Finitely-Presented Abelian Groups

Abelian groups are of interest not only for their intrinsic interest but also because many of the important groups arising in number theory and topology are abelian.

- Construction as a quotient of a free abelian group
- Direct product, free product
- Arithmetic
- Construction of subgroups and quotient groups
- Elementary divisors, primary invariants
- Factor basis, divisor basis, primary basis
- Torsion subgroup, torsion-free subgroup, p -primary component
- Homomorphisms: Image, kernel, cokernel
- Composition series, maximal subgroups, subgroup lattice (of a finite group)
- Character table of a finite group
- The group of homomorphisms $Hom(A, B)$, where A and B are finite abelian groups
- Abelian quotient of any group (with its natural homomorphism)
- Conversion between \mathbf{Z} -modules and abelian groups
- Functors from rings and fields onto abelian groups

The Hermite and Smith normal form algorithms are used to construct a normal form for subgroups and quotient groups of abelian groups.

4.6 Polycyclic Groups

The category described here comprises the family of all groups defined by a polycyclic presentation. Note that such a group may be infinite. Even so, algorithms for element arithmetic analogous to those used for finite soluble groups are available and a growing number of structural computations are possible within such a group.

4.6.1 Polycyclic Groups: Construction and Arithmetic

- Construction as a quotient of a free group
- Standard groups as polycyclic groups
- Permutation groups, matrix groups, abelian groups and finite soluble groups as polycyclic groups
- Nilpotent quotient of a finitely presented group (W. Nickel's algorithm)
- Direct products
- Product, inverse, conjugate, commutator for elements; improved collector, much faster, better complexity
- Element normal form and equality testing
- Element order
- Random element generation

4.6.2 Polycyclic Groups: Basic Invariants

- Test finiteness, group order
- Test for abelian, elementary abelian, cyclic, nilpotent
- Nilpotency class
- Hirsch number

4.6.3 Polycyclic Groups: Subgroup Constructions

- Subgroup and quotient group construction
- Normal closure of a subgroup
- Conjugation of subgroups
- Commutator subgroups
- Test subgroup membership and inclusion
- Test for normal and central subgroups
- Centraliser of elements (in a nilpotent group)
- Normaliser and centraliser of subgroups (nilpotent group)
- Test for conjugacy of elements and subgroups (nilpotent group)
- Intersection of subgroups (nilpotent group)

4.6.4 Polycyclic Groups: Normal Structure

- Normal series with free- or elementary-abelian factors
- Centre, upper central series
- Lower central series, derived subgroup and series
- Fitting subgroup, Fitting series
- G-module construction from action on free- or elementary abelian sections
- Abelian quotient structure

4.7 Finite Soluble Groups

A large number of efficient algorithms have been developed for computing information about finite soluble groups defined by a polycyclic presentation. The category described here comprises the family of all finite soluble groups defined by polycyclic presentations. Note that while p -groups are not considered as a separate formal category, in many cases more efficient algorithms are employed. Further, some important operations particular to p -groups are described in a separate section.

4.7.1 Construction

- Construction of polycyclic presentation for the maximal finite p -quotient of an fp-group (O'Brien's program)
- Construction of polycyclic presentation for the maximal finite soluble quotient of an fp-group (Plesken-Brückner algorithm)
- Construction of a polycyclic presentation for a soluble group given as a permutation group or a matrix group
- Split and non-split extensions, wreath products
- Representation of a soluble group in terms of a *SAG-presentation*
- The Besche-Eick database of all soluble groups of order at most 1000 excepting 512 and 768.

In 1991, Leedham-Green with Brownie and Cannon developed an echelonization algorithm capable of constructing a form of polycyclic presentation for a finite soluble group which exhibits much of the structure. This polycyclic presentation (known as a *SAG-presentation*) is given in terms of generators defining a chain of subgroups which refines a nilpotent series for the group. Each nilpotent section exhibits a lower p -central series for each prime p involved. Finally, the quotient of the group by a term of the nilpotent series splits over the layer below.

4.7.2 Conjugacy Classes

- Testing a pair of elements for conjugacy
- Conjugacy classes of elements
- Power map
- Class map, i.e. return the number of the class to which a given element belongs
- Class matrix
- Exponent

4.7.3 Subgroup Constructions

- Construction of subgroups, quotient groups
- Normal closure, core of a subgroup
- Normalizer, centralizer
- Testing subgroups for conjugacy
- Intersection of subgroups
- Permutation representation on the cosets of a subgroup
- System of double coset representatives for a pair of subgroups (Slattery algorithm)
- Sylow p -subgroup
- Hall π -subgroups, Sylow basis, complement basis
- System normalizer, relative system normalizer

Simple variations of the SAG-algorithm may be used to compute normal closures, the lower central series and the derived series. Sylow p -subgroups, Hall π -subgroups, a Sylow basis, and a complement basis may be read directly from the presentation. The availability of such a presentation together with sophisticated module theory machinery allowed us to design fast algorithms for finding the centre and maximal subgroups.

4.7.4 Normal Structure

- Centre, hypercentre, derived subgroup
- Derived series, upper central series, lower central series
- Chief series, composition series
- Elementary abelian series, p -central series
- Fitting subgroup
- Frattini subgroup
- Normal subgroups

4.7.5 Subgroup Structure

- Maximal subgroups
- Conjugacy classes of complements of a normal subgroup
- Conjugacy classes of subgroups, poset of subgroup classes
- Conjugacy classes of subgroups satisfying a condition: Cyclic, elementary abelian, abelian, nilpotent

The availability of an SAG-presentation combined with sophisticated module theory machinery allowed us to design fast a algorithm for finding the maximal subgroups of a soluble group.

4.7.6 Automorphisms and Representations

- Automorphism group of a soluble group (M Smith’s algorithm)
- Character table (Dixon-Schneider algorithm)
- Modular irreducible representations (Glasby-Howlett algorithm)
- Ordinary irreducible representations (Brückner algorithm)
- KG -module corresponding to an elementary abelian section

4.8 Finite p -Groups

Following the development, in the early 1970’s, of the p -quotient algorithm for constructing polycyclic presentations of a finitely presented p -group, Leedham-Green and others developed an extensive family of elegant and efficient algorithms for finite p -groups. The facilities described here apply to the family of all finite p -groups defined by so-called *power-conjugate presentations*. Note that as p -groups form a subcategory of the category of finite soluble groups discussed above, most of the soluble group operations apply to p -groups. However, in some cases more efficient algorithms are employed for p -groups than for soluble groups.

4.8.1 Construction

- As for finite soluble groups
- Construction of polycyclic presentation for the maximal finite p -quotient of an fp-group (O’Brien algorithm)
- p -group generation (Eamonn O’Brien algorithm)

4.8.2 Normal Structure

- As for finite soluble groups
- Agemo, omega subgroups
- Jennings series of a p -group

4.8.3 Isomorphisms and Automorphism Groups

- Construct a *standard presentation* for a p -group
- Test two p -groups for isomorphism
- Automorphism group of a p -group (E O’Brien’s algorithm)
- Degrees of irreducible characters (Slattery algorithm)

4.9 Groups Defined by Rewrite Systems

This is a category of finitely presented groups where the relations are interpreted as rewrite rules. If the group is defined by a confluent system of rewrite rules then we have a normal form for its elements and hence a solution to the word problem. A group belonging to this category is typically constructed by applying the Knuth-Bendix procedure. As in the case of monoids, Magma uses the Knuth-Bendix developed by Derek Holt as part of his package *kbmag*.

- Construction of an RWS group from an fp-group using the Knuth-Bendix procedure. Orderings supported include: *RT-recursive*, *recursive*, *ShortLex*, *WT-ShortLex* and *Wreath*
- Test a rewrite system for confluence
- Reduction of a word to normal form
- Operations on words: Product, exponentiation, inverse, equality
- Enumeration of elements
- Test for a group being finite

4.10 Automatic Groups

This category corresponds to short-lex automatic groups. A group is represented by four automata: first and second word-difference machines, a word-acceptor, and a multiplier. These automata are constructed using the Knuth-Bendix procedure. This category is implemented by Derek Holt’s package *kbmag*.

- Construction of an automatic group from an fp-group using the Knuth-Bendix procedure.
- Reduction of a word to normal form
- Product, exponentiation, inverse, equality of elements
- Enumeration of words without repetition
- Test for a group being finite
- Growth function for a group

4.11 Groups with Elements given as Straight-Line Programs

This is a class of finitely generated free groups whose elements are represented as “straight-line” programs and which are referred to as SLP-groups for brevity. Typically a SLP-group is used when it is necessary to evaluate long words in a permutation or matrix group G . If G is defined on d generators then a d -generator SLP-group F is defined together with the homomorphism of F onto G which sends the i -th generator of F to the i -th generator of G . Words corresponding to elements of G are built as elements of F where they are represented as expression trees thereby allowing very fast evaluation of long words in G .

- Construction
- Arithmetic with straight-line programs
- Homomorphism from a blackbox group onto an arbitrary group
- Random generation of elements (Cellar, Leedham-Green, Murray, Niemeyer, O’Brien algorithm)

4.12 Subgroups of $PSL(2, R)$

The group $GL_2^+(\mathbf{R})$ of 2 by 2 matrices defined over \mathbf{R} with positive determinant acts on the upper half complex plane $\mathbb{H} = \{x \in C | \text{Im}(x) > 0\}$ by fractional linear transformation:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} : z \mapsto \frac{az + b}{cz + d}.$$

Any subgroup Γ of $GL_2^+(\mathbf{R})$ also acts on \mathbb{H} . A *fundamental domain* for the action of Γ is a region of \mathbb{H}^* containing a representative of each orbit of the action. Magma contains a package written by Helena Verrill for working with \mathbb{H}^* and with congruence subgroups and their action on \mathbb{H}^* . The subgroups of $PSL_2(\mathbf{Z})$ currently allowed are those of the form $\Gamma_0(N)$, $\Gamma_1(N)$, $\Gamma(N)$, $\Gamma^1(N)$, $\Gamma^0(N)$, and intersections of these groups. The package allows the computation of generators for congruence subgroups, and various other information, such as coset representatives.

- Computation of generators of congruence subgroups
- Coset representatives for a subgroup of finite index in $PSL_2(\mathbf{Z})$
- Construction of cusps, cusp widths, and elliptic points of congruence subgroups
- Farey symbols for congruence subgroups
- Action of elements of $PSL_2(\mathbf{R})$ on the upper half complex plane
- Determination of vertices of a fundamental domain for the action of a congruence subgroup
- Equivalence of points under the action of a congruence subgroup
- Graphics: postscript output of pictures of fundamental domains, points and geodesics, and polygons with geodesic edges (all on the upper half complex plane)

5 Lie Theory

The current elements of the machinery for Lie theory comprise:

- Root datum for Lie groups
- Coxeter groups
- Groups of Lie type
- Complex reflection groups
- Finite-dimensional Lie algebras (See section on Algebras)

5.1 Root Datum for Lie Theory

A datatype for root datum has been implemented by Don Taylor and Scott Murray.

5.1.1 Cartan matrices and Cartan types

- Creation of Cartan matrices.
- Identification of Cartan matrices.
- Printing Dynkin diagrams.
- Predicates: `IsIrreducible`, `IsCrystallographic`, `IsSimplyLaced`.
- Computing the vectors in a root system.

5.1.2 Creating Root datum

Any semisimple root datum may be constructed (adjoint, simply connected and everything in between). We plan to add facilities for more general reductive root data in a future release.

5.1.3 Operators on Root Datum

- Basic access functions: (co)root space, simple (co)roots Cartan matrix, number of positive roots, rank, dimension, Cartan type
- Computing isogeny group
- Dynkin diagram
- Roots are stored in the standard ordering, and are accessed by specifying their position in this order. Also computed in this manner are: coroots, reflection matrices of roots and coroots, and the action on the roots.
- Duals, root subdatum, direct sums.
- Properties: Irreducible, crystallographic, simply laced, adjoint, simply connected.

5.1.4 Related Invariants

- Root norms and root heights.
- Killing forms and dual Killing forms.
- The string through one root in the direction of another.
- The constants defined in Carter for constructing Lie algebras and groups of Lie type: p , q , A (the Cartan integer), N , M , C , η .

5.2 Coxeter Groups

Finite Coxeter groups are implemented as a subclass of permutation groups so that they inherit all the operations for permutation groups as well as having many specialized functions. This module was implemented by Don Taylor and Scott Murray. Frank Lübeck and the Chevie team provided helpful assistance.

- Cartan matrix corresponding to a given Dynkin diagram
- Construction of a Coxeter group from a root datum or Cartan matrix
- Dynkin diagram of a Cartan matrix or Coxeter group
- Root datum for a Coxeter group
- Element as a reduced word in the standard generators
- Element of maximal length
- Unique long (short) root of greatest height
- Long word
- Short root of maximal height
- Reflections in Coxeter group
- Reflection subgroup
- Reduced representatives for cosets of the reflection subgroup
- Actions on roots and co-roots
- Coxeter group as a real reflection group
- Coxeter and parabolic subgroups; Transversals
- Braid group, pure braid group and Coxeter group presentation

5.3 Finite Groups of Lie Type

We have implemented code for computing in non-groups of Lie type, with elements represented by their Bruhat decomposition. These groups can be defined over any Magma ring. The twisted groups will be implemented in a future release, but for now it is possible to compute with them by treating them as subgroups of the non-twisted groups.

The standard and regular (adjoint) representations for each group may be computed.

- Generators for all classical families of groups of Lie type over a finite field.
- Generators for all exceptional families of groups of Lie type over a finite field.
- Killing form of the Cartan algebra associated with a given Weyl group
- Root elements
- Fundamental roots and their negatives of a simple Lie algebra of given type and rank
- Lie algebra of a Chevalley group as a structure constant algebra
- Adjoint action
- Graph automorphism of a Coxeter group
- Degree of a representation with specified weight
- The BN-pair for a Chevalley group

5.4 Complex Reflection Groups

- Construction of all finite irreducible unitary reflection groups over the complex field.

6 Rings and their Fields

This section is concerned with fields (mainly local and global arithmetic fields), their rings of integers and valuation rings.

- The rational field \mathbf{Q} and its ring of integers \mathbf{Z}
- Residue class rings of \mathbf{Z}
- Univariate polynomial rings
- Finite fields
- Number fields and their orders
- Rational function fields
- Algebraic function fields
- Valuation rings
- Real and complex fields
- Local fields
- Power series rings and Laurent series rings

In the case of arithmetic fields, the major facilities include:

- Construction of a basis for the maximal order(s)
- Decomposition of ideals into prime ideals
- Recognition of principal ideals
- The class group
- Fundamental units

6.1 The Rational Field and its Ring of Integers

6.1.1 Arithmetic

- Multiple precision integer arithmetic
- Integer multiplication via classical, Karatsuba and Schönhage-Strassen FFT methods
- Integer division via classical, Karatsuba and Schönhage-Strassen FFT methods
- Greatest common divisor via Weber Accelerated GCD and Schönhage algorithms
- Extended Greatest common divisor via Lehmer and Schönhage algorithms
- Alternative representation of integers in factored form
- Arithmetic functions: Jacobi symbol, Euler ϕ function, etc.

Magma includes both the Karatsuba algorithm and the Schönhage-Strassen FFT-based algorithm for the multiplication of integers. The crossover point (where the FFT method beats the Karatsuba method) is currently 2^{15} bits (approx. 10000 decimal digits) on Sun SPARC workstations and 2^{17} bits (approx. 40000 decimal digits) on Digital Alpha workstations. Assembler macros are used for critical operations and 64-bit operations are used on DEC-Alpha machines. On Sun SPARC workstations, integer multiplication is 25-40% faster than GNU `gmp` 2.0.2 for integers having up to 10,000 decimal digits. As the size of integers increases beyond this point, the differential between Magma and `gmp` becomes much greater. In Magma, computing the product of two 10 000-bit integers takes 0.0023 seconds on a 400MHz Sun SPARC workstation and 0.0011 seconds on a 333MHz Digital Alpha workstation (a 64-bit machine), respectively. Computing the product of two integers, each having one million *decimal* digits, takes 3.0 seconds and 2.2 seconds, respectively, on these machines.

Magma also contains an asymptotically-fast integer (and polynomial) division algorithm which reduces division to multiplication with a constant scale factor that is in the practical range. Thus division of integers and polynomials are based on the Karatsuba and Schönhage-Strassen (FFT) methods when applicable. The crossover point for integer division (when the new method outperforms the classical method) is currently at the point of dividing a 2^{12} bit (approx. 1200 decimal digit) integer by a 2^{11} (approx. 600 decimal digit) integer on Sun SPARC workstations.

Finally, Magma contains implementations of the fast classical Lehmer extended GCD ('XGCD') algorithm (which is about 5 times faster than the Euclidean XGCD algorithm) and the Schönhage recursive ('half-GCD') algorithm, yielding asymptotically-fast GCD and XGCD algorithms. On a Sun SPARC workstation, the crossover point for the Schönhage GCD algorithm (where it beats the classical Accelerated GCD algorithm) is 32768 bits (about 10000 decimal digits), while the crossover point for the Schönhage XGCD algorithm (when it beats the Lehmer XGCD algorithm) is 6000 bits (about 2000 decimal digits). On a 400Mhz Sun SPARC workstation, Magma can compute the GCD of two arbitrary integers, each having a million *decimal* digits, in 9.0 seconds, and the extended GCD multipliers for the same numbers in 29.9 seconds.

6.1.2 Primality and Factorization

- Probabilistic primality testing (Miller-Rabin)
- Rigorous primality testing (Morain's Elliptic Curve Primality Prover)
- Primality certificates; Verification of certificates
- Generation of primes
- Elementary factorization techniques: Trial division, SQUFOF, Pollard ρ , Pollard $p - 1$
- Elliptic curve method for integer factorization (A. Lenstra)
- Multiple prime multiple polynomial quadratic sieve algorithm for integer factorization (A. Lenstra)
- Database of factorizations of integers of the form $p^n \pm 1$

The Elliptic Curve Primality Prover (ECP) designed and implemented by François Morain at INRIA is installed in Magma. This provides fast rigorous primality proofs for integers having several hundred digits. The primality of a 100 digit integer is established in 24 seconds (on a Sun 200Mhz SPARC workstation 2).

6.2 Residue Class Rings of \mathbf{Z}

A quotient ring $\mathbf{Z}/\langle m \rangle$ of \mathbf{Z} is trivially represented by the integers taken modulo m . In the Magma implementation, m may be taken to be a long integer.

- Arithmetic; square root, all square roots
- Testing elements for: nilpotency, primitivity, regularity, zero-divisor
- Order of a unit
- Gcd and lcm
- Location of a primitive element
- Unit group
- Functor from additive group to an object in the category of abelian groups
- One or all square roots of an element

6.3 Univariate Polynomial Rings

A polynomial ring may be formed over any ring, including a polynomial ring. Since computational methods for univariate polynomial rings are often much simpler and more efficient than those for multivariate rings (especially over a field), we discuss the two cases separately. In this section, the symbol K , appearing as a coefficient ring, will denote a field.

6.3.1 Creation and Ring Operations

- Creation of a polynomial ring
- Definition of a ring map
- Kernel of a ring map
- Determining whether a ring map is surjective
- Determining whether a ring map is an isomorphism

6.3.2 Creation of Special Polynomials

- Orthogonal polynomials: Bernoulli polynomial
- Orthogonal polynomials: Chebyshev polynomials of the first and second kinds
- Orthogonal polynomials: Chebyshev polynomials of types T and U
- Orthogonal polynomials: Gegenbauer polynomial, Hermite polynomial
- Orthogonal polynomials: Generalised Laguerre polynomial
- Orthogonal polynomials: Legendre polynomial
- Binomial polynomial
- Conway polynomial of degree n over $GF(p)$
- Primitive polynomial of degree n over $GF(q)$
- Cyclotomic polynomial of order n
- Permutation polynomials: Dickson polynomials of the first and second kinds

6.3.3 Arithmetic with Polynomials

- Polynomial product via classical, Karatsuba and Schönhage-Strassen FFT algorithms
- Polynomial quotient via classical and Karatsuba algorithms
- Pseudo quotient and remainder
- Modular exponentiation and inverse
- Modular composition over $GF(q)$ (Brent-Kung)
- Norms
- Differentiation and integration
- Evaluation and interpolation
- Properties: prime, primitive, separable, permutation
- Properties: a unit, a zero-divisor, nilpotent

Magma employs asymptotically fast algorithms for performing arithmetic with univariate polynomials. These include two FFT-based methods for multiplication: the Schönhage-Strassen FFT method for situations where the coefficients are large compared with the degree, and the small-prime modular FFT with Chinese remaindering method for where the coefficients are small compared with the degree. These methods are applied to multiplication of polynomials over \mathbf{Z} , \mathbf{Q} , $\mathbf{Z}/m\mathbf{Z}$ and $GF(q)$. For some coefficient rings, at least one of the FFT methods outperforms the Karatsuba method for polynomials having degree as small as 32 or 64; for each of the coefficient rings listed, the FFT method beats the Karatsuba method for degree 128 or greater. An asymptotically-fast division algorithm (which reduces division to multiplication) is also used for polynomials over all coefficient rings.

6.3.4 GCD and Factorization

- Resultant, discriminant (sub-resultant algorithm, Euclidean algorithm)
- Greatest common divisor, extended greatest common divisor
- Extended greatest common divisor
- Hensel lift
- Square free factorization
- Distinct degree factorization
- Factorization over $\text{GF}(q)$: Small field Berlekamp, large field Berlekamp, Shoup algorithms
- Factorization over \mathbf{Z} and \mathbf{Q} : van Hoeij algorithm
- Factorization over \mathbf{Q}_p and its extensions
- Factorization over $\mathbf{Q}(\alpha)$: Trager algorithm

Greatest common divisors for polynomials over \mathbf{Z} are computed using either a modular algorithm or the GCD-HEU method, while for polynomials over a number field, the modular method is used. Factorization of polynomials over \mathbf{Z} uses the exciting new algorithm of Mark van Hoeij, which efficiently finds the correct combinations of modular factors by solving a Knapsack problem via the LLL lattice-basis reduction algorithm.

6.3.5 Arithmetic with Ideals

- Construction of ideals and subrings (over K)
- Construction of quotient rings
- Arithmetic with ideals (over K)
- Properties of an ideal: Maximal, prime, primary

6.4 Residue Class Rings of Univariate Polynomial Rings

- Arithmetic with elements
- Construction of ideals and subrings (over K)
- Construction of quotient rings
- Arithmetic with ideals (over K)

6.5 Finite Fields

6.5.1 Construction

- Construction of fields $\text{GF}(p)$, p large; $\text{GF}(p^n)$, p small and n large
- Optimized representations of $\text{GF}(p^n)$ in the case of small p and large n
- Database of sparse irreducible polynomials over $\text{GF}(2)$ for all degrees up to 11000.
- Special optimized packed arithmetic for fields of characteristic 2.
- Construction of towers of extensions
- Construction of subfields
- Compatible embedding of subfields
- Enumeration of irreducible polynomials

The finite field module uses different representations of finite field elements depending upon the size of the field. Thus, in the case of small to medium sized fields, the Zech logarithm representation is used. For fields of characteristic 2, a packed representation is used (since V2.4), which is very much faster than the representation used in previous versions of Magma. For a large degree extension K of a (small) prime field, K is represented as an extension of an intermediate field F whenever possible. The intermediate field F is chosen to be small enough so that the fast Zech logarithm representation may be used. Thus, Magma supports finite fields ranging from $\text{GF}(2^n)$, where the degree n may be a ten thousand or more, to fields $\text{GF}(p)$, where the characteristic p may be a thousand-bit integer. The finite field code is highly optimized for very small finite fields and especially for linear algebra over such fields.

A noteworthy feature of the facility is that no matter how a field is created, its embedding into an overfield may be determined. One may create and work within a lattice of subfields with create ease. The system is described in detail in a paper [2].

6.5.2 Arithmetic

- Trace and norm; Relative trace and norm
- Order of an element
- Characteristic and minimum polynomials
- Testing elements for: Normality, primitivity
- Construction of primitive and normal elements

6.5.3 Roots and Polynomial Factorization

- Square root: Tonelli-Shanks method for $\text{GF}(p)$
- n -th root
- One root of a polynomial; All roots root of a polynomial
- Polynomial factorization: Small field Berlekamp, large field Berlekamp
- Polynomial factorization: Shoup algorithm
- Construction of a splitting field
- Factorisation over splitting field

Factorization of polynomials over $\text{GF}(q)$ for large q uses Shoup's algorithm. On a 64-bit 200MHz SGI Origin 2000, the $n = 2048$ polynomial from the von zur Gathen challenge benchmarks (of degree 2048 with sparse coefficients modulo a 2050-bit prime) is factored by Magma V2.7 in about 18.8 hours and the $n = 3000$ polynomial from the same benchmarks (of degree 3000 with sparse coefficients modulo a 3002-bit prime) is factored by Magma V2.7 in about 75.8 hours. Also, the $n = 2048$ polynomial from the Shoup challenge benchmarks (of degree 2048 with dense coefficients modulo a 2048-bit prime) is factored by Magma V2.7 in about 36.0 hours on the same machine.

6.5.4 Discrete Logarithms

- Shanks baby-step giant-step algorithm
- Pollard rho algorithm
- Polhig-Hellman algorithm
- Index calculus method using either the Gaussian integer sieve or linear sieve (for prime fields $\text{GF}(p)$).

The index calculus method applied to an arbitrary field $\text{GF}(p)$, where p is a 100-bit prime such that $(p - 1)/2$ is prime (the worst case), takes 10 seconds to perform the sieving and about 0.8 seconds to compute an individual logarithm. For a 20-decimal-digit prime p such that $(p - 1)/2$ is prime, Magma takes only 1 second for the sieving and about 0.3 seconds to compute an individual logarithm.

6.5.5 Derived Structures

- Unit group
- Additive group
- Field as an algebra over a subfield

6.6 Number Fields and their Orders

Magma currently has three main categories corresponding to number fields: general number fields, quadratic fields, and cyclotomic number fields. It should be noted, that quadratic fields and cyclotomic fields are in fact number fields that allow special algorithms (e.g. the use of quadratic forms for class group computation, continued fractions for fundamental units, trivial detection of splitting behaviour of primes).

While logically only the general number field case is necessary, in practice faster algorithms are available for some problems in the case of quadratic and cyclotomic fields.

6.6.1 Quadratic Fields

- Discriminant, signature, conductor
- Maximal order, integral basis
- Division of integral elements of $\mathbf{Q}(\sqrt{d})$, for $d = -1, -2, -3, -7, -11, 2, 3, 5, 13, 17$.
- Factorization of integral elements of $\mathbf{Q}(\sqrt{d})$, for $d = -1, -2, -3, -7, -11$.
- Class number (Shanks' algorithm)
- Ideal class group (Buchmann's method)
- Unit group, fundamental unit, regulator
- Decomposition of primes
- Factorization of an ideal
- Solution of norm equations (Cornaccia's algorithm for imaginary fields and Cremona's conics for real quadratic fields)
- Facilities for binary quadratic forms (see Modular Forms)

6.6.2 Cyclotomic Fields

- Discriminant, signature, conductor
- Cyclotomic subfields
- Field as an algebra over a subfield
- Schur function, trace and norm
- Maximal order, integral basis
- Factorization of an ideal
- Unit group
- Galois group, automorphisms

6.7 General Number Fields

Facilities for general number fields have been developed in a joint project with the KANT group in Berlin. The core consists of machinery for performing arithmetic with arbitrary orders and their ideals. The major capabilities include facilities to determine the maximal order (round 2 and 4 algorithms), class group, unit group, Galois group (up to degree 23) and the computation of defining equations for ray class fields.

6.7.1 Number Fields

- Arithmetic
- Construction of equation orders, maximal orders, arbitrary orders
- Simple and relative extensions, extension defined by several polynomials
- Subfields

- transfer between relative and absolute representations
- Discriminant, reduced discriminant, signature
- Factorization of polynomials over number fields
- Completion of absolute fields at finite primes
- Number fields with arbitrary bases

6.7.2 Orders and Fractional Ideals

- Multiple relative extensions
- Maximal order, integral basis (Round 2 and Round 4 algorithms)
- Suborders, extension orders
- Construction of integral and fractional ideals
- Ideal arithmetic: product, quotient, gcd, lcm
- Determination of whether an ideal is: integral, prime, principal
- Decomposition of primes
- Valuations of order elements and ideals at prime ideals
- Ramification index
- Factorization of an ideal
- Residue field of an order modulo a prime ideal
- Residue class ring of an order modulo an arbitrary ideal
- Completion of absolute maximal orders at finite primes

6.7.3 Invariants

- Class group: Conditional (GRH) and unconditional algorithms
- Unit group: Conditional (GRH) and unconditional algorithms
- Regulator
- Exceptional units, S -units
- Ray class groups, unit group of ray class rings of absolute maximal orders.

6.7.4 Diophantine Equations

- Norm equations, relative norm equations
- Thue equations
- Unit equations
- Index form equations
- Integral points on Mordell curves

6.7.5 Automorphisms

- Determination of subfields
- Automorphism groups of normal and abelian fields
- Isomorphism of number fields
- Galois group of number fields having degree less than 24
- Galois correspondence
- Ramification theory

Consider the extension K of \mathbf{Q} by a root of $x^9 - 57x^6 + 165x^3 - 6859$. The maximal order of K is found in 0.25 seconds, the class group $(\mathbf{Z}/3\mathbf{Z})$ is found unconditionally in 41 seconds (conditionally, under GRH, in 26 seconds, using even smaller bounds it is possible to compute the class group in 4.11 seconds), the unit group $(\mathbf{Z}/2\mathbf{Z} \oplus \mathbf{Z} \oplus \mathbf{Z} \oplus \mathbf{Z} \oplus \mathbf{Z})$ in 3.50 seconds and the Galois group of order 18 in 0.25 seconds. The four subfields of degree 3 are found in 0.10 seconds.

6.8 General Algebraic Function Fields

A general algebraic function field F/k of n variables over a field k is a field extension F of k such that F is a finite field extension of $k(x_1, \dots, x_n)$ for elements $x_i \in F$ which are algebraically independent over k .

6.8.1 Rational Function Fields

Given any field k and indeterminates x_1, \dots, x_n , the user may form the field of rational functions $k(x_1, \dots, x_n)$ as the localization of the polynomial ring $k[x_1, \dots, x_n]$ at the prime ideal $\langle x \rangle$.

- Arithmetic
- Evaluation
- Derivative
- Partial fraction expansion
- Partial fraction decomposition (squarefree or full factorization)

6.8.2 Algebraic Function Fields

Within Magma, algebraic function fields of one variable can be created by adjoining a root of an irreducible, separable polynomial in $k(x)[y]$ to the rational function field $k(x)$. If k is a finite field, the function field is said to be *global*.

- Element arithmetic
- Norm, trace of an element with respect to the field extension $F/k(x)$
- Minimal and characteristic polynomials of an element wrt. $F/k(x)$
- Representation matrices of algebraic functions wrt. $F/k(x)$
- Construction of (“finite” and “infinite”) equation orders
- Construction of (“finite” and “infinite”) maximal orders (Round 2 algorithm)

6.8.3 Function Fields: Invariants

- The genus
- Exact constant field
- Wronskian orders, gap numbers, ramification divisor and Weierstrass places
- Space of holomorphic differentials
- Intermediate fields of $F/k(x)$

Additionally for global function fields:

- Places of degree one (in constant field extensions)
- L -polynomial and ζ -function
- Structure of the divisor class group
- p -rank of the divisor class group (separate method) and Hasse-Witt invariant

6.8.4 Function Fields: Orders and Fractional Ideals

- Discriminant
- Integral closures resp. maximal orders
- Construction of integral and fractional ideals
- Ideal arithmetic: product, quotient, gcd, lcm
- Determination of whether an ideal is: integral, prime, principal (principal in the global case)
- Decomposition of primes

- Valuations of order elements and ideals at prime ideals
- Ideal factorization
- Basis reduction (in the global, tamely ramified case)
- Unit rank, independent, fundamental units, regulator (in the global case).

6.8.5 Function Fields: Places

- Zeros and poles of algebraic functions
- Valuation (zero, pole orders) of algebraic functions at places
- Local uniformizer at a place
- Residue class field at a place
- Evaluation of algebraic functions at places
- Expansion of algebraic functions at a place of degree one
- Decomposition of places of $k(x)$ within $F/k(x)$
- Ramification index, inertia degree
- Bounds on the number of places of degree 1 (global case)
- All places of a prescribed degree (global case)
- Random place of a prescribed degree (global case)

6.8.6 Function Fields: Divisors

- Degree, arithmetic and various other, related operations
- Divisor of an algebraic function
- Canonical divisor
- Different divisor
- Ramification divisor
- Properties: Effective, positive, principal, special, canonical
- Riemann-Roch space $\mathcal{L}(D)$ of a divisor D , given by a k -basis of algebraic functions
- Index of speciality
- Divisor reduction
- Wronskian orders, gap numbers, ramification divisor and Weierstrass places with respect to a divisor
- Number of smooth divisors (global case)

6.8.7 Function Fields: Differentials

- Arithmetic and various, related operations
- Valuation of a differential at a place
- Divisor of a differential
- Differential spaces for given divisors (e.g. holomorphic differentials)
- Higher differentiations
- Residue of a differential at a place of degree one
- Cartier operator and representation matrix of the Cartier operator (global case)

6.8.8 Global Function Fields: Divisor class group

- Bounds on the generation of the class group
- Approximation of the class number
- Structure of the divisor class group, representation of divisor classes as abelian group elements
- S -class group, S -units and S -regulator for a finite set of places S
- Exact sequence

$$0 \rightarrow U(S) \rightarrow F^\times \rightarrow Div(S) \rightarrow Cl(S) \rightarrow 0$$

- Image and preimage computation possible for the maps of the exact sequence
- Similar functionality for the finite maximal order
- p -rank of the divisor class group (separate method)

The development of this module is a joint project with the KANT group.

6.9 Discrete Valuation Rings

Valuation rings are available for the rational field and for rational function fields. For rational function fields, given an arbitrary monic irreducible polynomial $p(x) \in K[x]$, the valuation ring is

$$O_{p(x)} = \left\{ \frac{f(x)}{g(x)} : f(x), g(x) \in K[x], p(x) \nmid g(x) \right\}.$$

Valuations corresponding both to an irreducible element and to ∞ are allowed.

- Valuation ring corresponding to the discrete non-Archimedean valuation v_p of \mathbf{Q}
- Valuation ring corresponding to the discrete non-Archimedean valuation v_p of a rational function field
- Valuation ring corresponding to the valuation v_∞ of a rational function field
- Arithmetic
- Euclidean norm, valuation
- Greatest common divisor

6.10 The Real and Complex Fields

The real and complex fields are different from most structures in that exact computation in them is almost never possible.

- Arithmetic
- Square root, arithmetic-geometric mean
- Continued fraction expansion of a real number
- Constants: π , Euler's constant, Catalan's constant
- Logarithm, dilogarithm, exponential
- Trigonometric functions, hyperbolic functions and their inverses
- Bernoulli numbers
- Γ function, incomplete Γ function, complementary incomplete Γ function, logarithm of Γ function
- J -Bessel function, K -Bessel function
- U -confluent hypergeometric function
- Logarithmic integral, exponential integral
- Error function, complementary error function
- Dedekind η function
- Jacobi sine theta-function and its k -th derivative
- Log derivative (ψ) function, i.e. $\frac{\Gamma'(x)}{\Gamma(x)}$
- Riemann- ζ function
- Polylogarithm, Zagier's modifications of the polylogarithm
- Weber's f -function, Weber's f_2 -function, j -invariant
- Integer polynomial having a given real or complex number as an approximate root (Hastad, Lagarias and Schnorr LLL-method)
- Roots of an exact polynomial to a specified precision (Schönhage splitting circle method)
- Summation of a series (Euler-Wijngaarden method for alternating series)
- Numerical integration of a function (Romberg-type methods)

Two different models of the real and complex field are available in Magma. The default version is based on semantics developed by Henri Cohen for PARI. In this model, the precision of a real or complex number is determined by the accuracy of the operands and the operation. Whenever a real or complex number is known exactly, it is kept in exact form and only converted to real/complex form when it has to be used as an argument in some operation. Thus, during a calculation, real or complex numbers will appear with varying precisions, where the precision for a particular number is chosen in such a way that all digits should be meaningful. This is achieved through use of a form of interval arithmetic. We call this model of the real or complex field, the *free* model. Magma implements its free model using a modified version of the PARI code. The PARI code achieves its speed by using assembler for a small number of critical operations and by careful organization. This has been carried over into the Magma version so that the speed of the Magma version is virtually the same as the native PARI code in all but a few instances. The hundred or more real and complex functions implemented in PARI are available in Magma.

A second model of the real field is provided whereby all numbers are stored to a fixed precision. This version is based on Richard Brent's MP package and is known as the *truncated* model.

6.11 Newton Polygons

- Construction of a newton polygon: Compact, infinite or including the origin
- Construction of newton polygons from different types of data: $f \in k[x, y]$, $f \in k \langle\langle x \rangle\rangle [y]$, a finite set of points; a finite set of faces (weighted dual vectors)
- Finding faces and vertices
- If polygon is derived from a polynomial f , finding restrictions of f to faces
- Locating a given point relative to a newton polygon
- Determining whether a given line supports a face
- Determining whether two polygons are the same

6.12 Local Rings and Fields

A p -adic ring arises as the completion of the ring of integers at a prime while a local field arises as the completion at a prime ideal of a number field.

6.12.1 Local Rings: Construction

- Construction of a p -adic ring or field
- Unramified extension of a local ring or field
- Totally ramified extension of a local ring or field
- Ring of integers of a local field
- Field of fractions of a local ring
- Change precision of a ring, field or element

A local ring is a finite degree extension of a p -adic ring and may be either ramified or unramified or both. It is constructed in the form of a two-step extension, the first step being an unramified extension I of degree f over \mathbf{Z}_p , generated by a root of unity of order $p^f - 1$, and the second being a totally ramified extension of I defined by an Eisenstein polynomial.

6.12.2 Local Rings: Arithmetic

- Arithmetic operations
- Valuation of an element
- Norm and trace of an element
- Logarithm, exponential of an element
- Square root, n -th root of an element
- Minimal polynomial of an element over the p -adic subring or field
- Image of an element under a power of the Frobenius automorphism
- Linear algebra over local rings and fields

6.12.3 Local Rings: Polynomial Factorization

- Polynomial algebra over local rings and fields
- Greatest common divisor of two polynomials
- Hensel lifting of the factors of a polynomial
- Hensel lifting of the roots of a polynomial
- Test a polynomial for irreducibility
- Roots of a polynomial over a local ring or field
- Factorization of polynomials over local ring or field

6.13 Power, Laurent and Puiseux Series Rings

Magma contains an extensive package for formal power series. The fact that we may only work with a finite number of terms, n say, of a power series, i.e., a truncated power series, is made precise by noting that we are working in the quotient ring $R[[x]]/\langle x^{n+1} \rangle$, for some n , rather than in the full ring $R[[x]]$. Provided this is kept in mind, calculations with elements of a power series ring (though not field) are always precise.

Given a field K , a field of Laurent series $K((x))$ is regarded as the localization of the power series ring $K[[x]]$ at the ideal $\langle 0 \rangle$. More simply, it is the field of fractions of $K[[x]]$. Since elements of such a field are infinite series, calculation is necessarily approximate.

A power series ring $R[[x]]$ is regarded as the completion of the polynomial ring $R[x]$ at the ideal $\langle 0 \rangle$.

Puiseux series with arbitrary fractional exponents are also supported (since V2.4).

- Arithmetic
- Inversion of units
- Derivative, integral
- Square root, valuation
- Exponentiation, composition, convolution, reversion
- Power series expansions of transcendental functions
- $R[[x]]/\langle x^{n+1} \rangle$ as an algebra over R

6.14 Algebraically Closed Fields

Algebraically closed fields (ACF's) have the property that they always contain all the roots of any polynomial defined over them.

- Automatic extension of the field by the roots of any polynomial over the field, and operations on conjugates of roots
- Basic arithmetic
- All standard algorithms for rings over generic fields work over such fields
- Minimal polynomial
- Simplification of the field
- Construction of the corresponding absolute field together with the isomorphism
- Pruning of useless variables and relations

It is not possible to construct explicitly the closure of a field, but the system works by automatically constructing larger and larger algebraic extensions of an original base field as needed during a computation, thus giving the illusion of computing in the algebraic closure of the base field.

A similar system was suggested by D. Duval and others (the D5 system [6]), but this has difficulty with the parallelism which occurs when one must compute with several conjugates of a root of a reducible polynomial, leading to situations where a certain expression evaluated at a root is invertible but evaluated at a conjugate of that root is not invertible. The scheme developed for Magma by Allan Steel avoids these problems. Consequently, ACF's behave in the same way as any other field implemented in Magma; all standard algorithms implemented for generic fields and which use factorization work without change (for example, the Jordan form of a matrix). The new system avoids factorization over algebraic number fields when possible, and automatically splits the defining polynomials of a field when factors are found. The field may also be simplified and expressed as an absolute field. Especially significant is also the fact that all the Gröbner basis algorithms work well over ACF's. One can now compute the variety of any zero-dimensional multivariate polynomial ideal over the algebraic closure of its base field. Puiseux expansions of polynomials are now also computed using an algebraically closed field.

7 Commutative Algebra

The Magma facility for commutative rings allows the user to define any ring, starting from the ring of integers, by repeatedly applying the four basic constructions: *transcendental extension*, *quotient by an ideal*, *localization*, and *completion*. Rings derived from a polynomial ring will be considered in this section, while fields, their orders and valuation rings will be presented in the following section. The following rings and modules are considered here:

- Multivariate polynomial rings
- Ideal theory of multivariate polynomial rings
- Affine algebras
- Modules over affine algebras

The basic computational problems for commutative rings include:

- A canonical form for elements
- Efficient arithmetic
- A canonical representation (i.e., standard basis) for ideals
- Arithmetic with ideals
- Formation of quotient rings
- Ideal decomposition, i.e., primary decomposition
- The study of modules over rings

The fundamental tools on which most machinery for computational (commutative) ring theory is based include factorization of elements in a UFD, the efficient construction of standard bases for ideals and the factorization of ideals.

7.1 Multivariate Polynomial Rings

Multivariate polynomial rings in any number of variables may be formed over any coefficient ring, including a polynomial ring. Multivariate polynomials are represented in distributive form, using ordered arrays of coefficient-monomial pairs. Different orderings are allowed on the monomials; these become significant in the construction of Gröbner bases of ideals. Computations with ideals are available (since V2.8) for ideals defined over general Euclidean rings as well as for ideals defined over fields.

7.1.1 Polynomial Rings: Creation and Ring Operations

- Creation of a polynomial ring
- Monomial orders: lexicographical, graded lexicographical, graded reverse lexicographical, block elimination, general weight vectors
- Dynamic data structures for monomials providing optimal packing and rigorous detection of overflow
- Base extend
- Definition of a ring map
- Kernel of a ring map
- Properties of ring maps: Surjective, bijective

Since Magma V2.7, the original linked-list representation of polynomials has been replaced with a more compact random-access array structure, resulting in less memory usage and faster access. A new fraction-free representation for polynomials at the lowest level gives very significant speedups for arithmetic over some fields (particularly the rational field and rational function fields). A new representation employing variable byte sizes for monomials is also introduced in V2.7, requiring less memory and providing greater speed. The maximum total degree of any monomial has been increased to $2^{30} - 1 = 1073741823$. Monomial overflow is rigorously detected.

7.1.2 Polynomial Rings: Arithmetic with Polynomials

- Arithmetic with elements
- Recursive coefficient, monomial, term, and degree access
- Differentiation, integration
- Evaluation and interpolation
- Properties: a unit, a zero-divisor, nilpotent

7.1.3 Polynomial Rings: GCD and Factorization

- Resultant (modular and sub-resultant algorithms), discriminant
- Greatest common divisor (sparse EEZ-GCD, fast GCD-HEU, evaluation-interpolation algorithms)
- Newton polygon
- Square free factorization
- Factorization over $\text{GF}(q)$, \mathbf{Z} and \mathbf{Q} (EEZ algorithm, interpolation algorithm)
- Factorization over $K(x)$, where K is one of $\text{GF}(q)$, \mathbf{Q}

7.1.4 Polynomial Rings: Gröbner Basis

- Construction of ideals and subrings
- Gröbner bases of ideals over fields, with specialized algorithms for different coefficient fields (fraction-free methods for the rational field and rational function fields)
- Gröbner bases of ideals over general Euclidean rings, using an extension of Allan Steel of Faugère's F_4 algorithm
- Gröbner Walk algorithm for converting the Gröbner basis of an ideal over a field from one monomial order to another order
- FGLM algorithm for converting the Gröbner basis of a zero-dimensional ideal over a field from one monomial order to a different order (a fast p -adic method is used in the case of the rational field)
- Construction of degree- d (truncated) Gröbner bases
- Normal form of a polynomial with respect to an ideal
- Reduction of ideal bases
- S -polynomial of two polynomials

Magma computes the **lex** order Gröbner basis for the Katsura-5 problem in 4.5 seconds and the **lex** order Gröbner basis for the cyclic 6-th roots problem in 3.9 seconds. Taking a much more difficult example, Magma computes the **grevlex** order Gröbner basis for the Cyclic-7 roots ideal in 6.4 minutes and transforms this to the **lex** order Gröbner basis in a further 2.6 minutes (using the p -adic FGLM algorithm). These examples are from the POSSO polynomial systems library (<ftp://posso.dm.unipi.it>).

7.1.5 Polynomial Rings: Arithmetic with Ideals

- Sum, product, intersection, colon ideal,
- Saturation of an ideal, leading monomial ideal
- Elimination ideals
- Determination of whether a polynomial is in an ideal or its radical
- Properties of an ideal: Zero, principal, proper, zero-dimensional
- Extension and contraction of ideals
- Variable extension of ideals

7.1.6 Polynomial Rings: Invariants for Ideals

- Dimension and maximally independent sets
- Hilbert series and Hilbert polynomial
- Primary decomposition of an ideal
- Triangular decomposition of a zero-dimensional ideal (algorithm of D. Lazard).
- Probabilistic prime decomposition of the radical of an ideal
- Radical of an ideal
- Computation of the variety of a zero-dimensional ideal
- Relation ideals (determination of algebraic relations between polynomials)
- Syzygy modules
- Construction of a minimal generating set of a polynomial subalgebra
- Computations with polynomial generators of a submodule over a subalgebra in a polynomial ring

7.1.7 Polynomial Rings: Gradings

- Construction of graded polynomial rings with specific weights on the variables
- All monomials of specific total or weighted degree
- Homogeneous components of polynomials
- Homogenization and dehomogenization of an ideal
- Hilbert-driven Buchberger algorithm for fast computation of the Gröbner basis of a homogeneous ideal when the Hilbert series is known
- Construction of degree- d (truncated) Gröbner bases (respecting the grading of the polynomial ring)

7.2 Affine Algebras

Let K be a field, $R = K[x_1, \dots, x_n]$ a polynomial ring over K and I an ideal of R . The quotient ring $A = R/I$ is called an *affine algebra*.

7.2.1 Affine Algebras: Creation and Operations

- Creation of an affine algebra
- Arithmetic with elements

7.2.2 Affine Algebras: Arithmetic with Ideals

- Construction of ideals and subrings
- Gröbner bases of ideals
- Ideal arithmetic: addition, multiplication, powers, quotients, colon ideals, intersections
- Membership test for ideals
- Test equality and inclusion of ideals

Affine algebras arise commonly in commutative algebra and algebraic geometry. They can also be viewed as generalizations of number fields and algebraic function fields.

If the ideal J of relations defining an affine algebra $A = K[x_1, \dots, x_n]/J$ is *maximal*, then A is a field and may be used with any algorithms in Magma which work over fields. Factorization of polynomials over such affine algebras is also supported.

If an affine algebra has finite dimension considered as a vector space over the coefficient field, extra special operations are available on its elements.

7.3 Modules over Affine Algebras

Modules over a multivariate polynomial ring $R[x_1, \dots, x_n]$ (R a Euclidean ring or field) and quotient rings of such (affine algebras) form a special category in Magma. Multivariate polynomial rings are not principal ideal rings in general, so the standard matrix echelonization algorithms are not applicable. Magma allows computations in modules over such rings by adding a column field to each monomial of a polynomial and then by using the ideal machinery based on Gröbner bases. This method is much more efficient than introducing new variables to represent the columns since the number of columns does not affect the total number of variables.

7.3.1 Modules over Affine Algebras: Creation and Operations

- Construction of modules with TOP (“term over position”) or POT (“position over term”) module orders
- Construction of graded modules with weights on the columns (determining homogeneity)
- Arithmetic with elements
- Construction of Gröbner bases of modules
- Row and column operations on elements

7.3.2 Modules over Affine Algebras: Submodules

- Construction of submodules and quotient modules
- Membership testing
- Tensor product
- Hilbert series of (homogeneous) modules
- Multiplicity
- Submodule sum, intersection, colon operation
- Minimal bases for homogeneous modules
- Multiplicity of a submodule

8 Linear Algebra and Module Theory

- Matrix operations
- Vector spaces
- Free modules
- Modules over Orders

8.1 Matrices

In this section we list the basic facilities available for computing with matrices over various rings. These algorithms underpin the vector space and module theory machinery.

8.1.1 Representation of Matrices

- Optimal packed representation for matrices over $\text{GF}(2)$, using bit operations.
- Efficient packed representation for matrices over all small finite fields, using lookup tables for vector operations.
- Fraction-free algorithms for matrices over \mathbf{Q} , reducing computations to those over \mathbf{Z} .
- Internal modular representations of matrices over \mathbf{Z} or \mathbf{Q} , for several algorithms.
- Input of matrices using the “Cambridge” compact format

8.1.2 Arithmetic

- Multiplication: Strassen algorithm over finite fields
- Multiplication: Modular algorithm over large prime finite fields
- Tensor products of matrices
- Fast inverse of a rational matrix using modular methods
- Fast algorithm for powering matrices over finite fields using the primary rational form
- Order/projective order of matrices over finite fields (Leedham-Green algorithm)
- Row and column operations

8.1.3 Echelon Form and Nullspace

- Echelonization over fields and euclidean domains
- Nullspace (many different techniques including sparse, p -adic and modular algorithms)
- Determinant (modular algorithms over \mathbf{Z} and \mathbf{Q})
- Solution of systems of linear equations

Given a 301 by 300 matrix over \mathbf{Z} with random entries between 0 and 10, Magma can compute its nullspace in 9.1 seconds on a 400Mhz Sun SPARC workstation (using the fast p -adic algorithm). (The nullity is nearly always 1, and the integer entries of the non-zero null vector usually have about 450 digits each.) This can be compared with an algorithm of J. Buchmann and D. Squirrel for computing nullspaces in the Lidia system[3] which takes 3 hours and 54 minutes for the same problem on an SPARC (speed unspecified). Even assuming conservatively that they have used a 200Mhz processor (so we halve their time), Magma is thus about 750 (sic) times faster.

8.1.4 Canonical Forms

- Generalized Jordan canonical form of matrices over fields
- Rational and primary rational canonical forms of matrices over fields
- Hermite and Smith forms for matrices over Euclidean domains
- Characteristic polynomial, minimal polynomial,
- Eigenvalues and eigenspaces (modular algorithms used over \mathbf{Z} and \mathbf{Q})

- LLL-reduction of matrices over \mathbf{Z} .

Over fields, a fast algorithm due to Allan Steel is used to construct the various matrix canonical forms: generalized Jordan, rational, and primary rational [15]. Over an Euclidean domain, algorithms of Havas and others are used to compute characteristic polynomials and the Hermite and Smith normal forms. Given a 100×100 matrix over \mathbf{Z} with random one-digit entries Magma finds its Smith form in 1.3 seconds (the largest elementary divisor is 50 digits) and its characteristic polynomial in 46 seconds.

8.1.5 Sparse Matrix Techniques

- Structured-Gaussian elimination [12]
- Lanczos algorithm for finding dependencies among rows of large sparse matrices over finite fields.

8.2 Vector Spaces

8.2.1 Construction

- Construction of vector spaces of n -tuples over a field
- Construction of vector spaces comprising $m \times n$ matrices over a field
- Extension and restriction of the field of scalars
- Direct sum

8.2.2 Construction

- Vector arithmetic
- Normalization, rotation
- Tensor product of vectors
- Trace of a vector in a subfield
- Weight, support

8.2.3 Subspaces and Quotient Spaces

- Construction of a subspace
- Membership of a subspace
- Transversal of a subspace (over a finite field)
- Complement of a subspace
- Sum and intersection of subspaces
- Reduction of vectors over a subspace
- Quotient spaces

8.2.4 Bases

- Construction of a vector space with specified basis
- Coordinates of a vector with respect to a basis
- Test for linear independence of a set of vectors
- Extend a linearly independent set to a basis

8.2.5 Homomorphisms

- Construction of $\text{Hom}(U, V)$, U and V vector spaces
- Image, kernel cokernel
- Echelon form

8.2.6 Quadratic Forms

Every vector space is equipped with the standard inner product. Commencing with V2.7, the user may specify an arbitrary quadratic form.

- Creation of a vector space with a designated quadratic form (a *quadratic space*)
- Inner product and norm of vectors

8.3 Free Modules

In this section we are mainly concerned with free modules. We consider R -modules M of n -tuples where R has scalar action on M . The case in which the action of R on M is via a matrix algebra is considered in the section on Representation Theory.

8.3.1 Basic Operations

- Construction of free modules of n -tuples
- Construction of modules comprising $m \times n$ matrices
- Arithmetic
- Extension and restriction of the ring of scalars
- Direct sum
- Construction of submodules, quotient modules
- Sum and intersection of submodules
- Basis operations

8.3.2 Homomorphisms

- Construction of modules $\text{Hom}(M, N)$ for any free modules M and N
- Explicit construction of $\text{Hom}(U, V)$ for proper subspaces U and V
- Explicit construction of $\text{Hom}(H_1, H_2)$ for homomorphism modules H_1 and H_2 with left or right matrix action
- Construction of the reduced module of a homomorphism module whose elements are with respect to the bases of the domain and codomain (not just the generic bases of these)
- Image, kernel, cokernel
- Echelon form (over a field)
- Hermite and Smith normal forms (over an ED)

8.4 Modules over Orders

Modules over orders are currently supported only for maximal orders so that the modules are over a Dedekind domain.

8.4.1 Basic Operations

- Creation of modules over orders
- Arithmetic with module elements
- Submodules and quotients of modules
- Determininant, dimension, pseudo-generators
- Equality of modules, membership
- Intersection of submodules
- Product of a module by an ideal
- Pseudo-basis and elementary divisors
- Dual of a module
- Steinitz class and Steinitz form

8.4.2 Homomorphisms

- Construction of modules $Hom(M, N)$
- Image, kernel, cokernel
- Hermite and Smith normal forms

9 Lattices and Quadratic Forms

9.1 Lattices

A lattice in Magma is a \mathbf{Z} -module contained in \mathbf{Q}^n or \mathbf{R}^n , together with a positive definite inner product. The information specifying a lattice is a basis, given by a sequence of elements in \mathbf{Z}^n , \mathbf{Q}^n or \mathbf{R}^n , and a bilinear product (\cdot, \cdot) , given by $(v, w) = vMw^{tr}$ for a positive definite matrix M . Central to the lattice machinery in Magma is a highly optimized LLL algorithm. The LLL algorithm takes a basis of a lattice and returns a new basis of the lattice which is *LLL-reduced* which usually means that the vectors of the new basis have small norms. The Magma LLL algorithm is based on the FP-LLL algorithm of Schnorr and Euchner and the de Weger integral algorithm but includes various optimizations, with particular attention to different kinds of input matrices.

9.1.1 Lattices: Construction and Operations

- Creation of a lattice by a given generating matrix or basis matrix together with an optional inner product matrix
- Creation of a lattice by a given Gram matrix
- Construction of lattices from codes
- Construction of lattices from algebraic number fields
- Construction of special lattices, including the root lattices A_n, D_n, E_n ; the laminated lattices Λ_n (including the Barnes-Wall lattice Λ_{16} and the Leech Lattice Λ_{24}); the Kappa lattices K_n , etc.
- Creation of and arithmetic with lattice elements
- Inner product, norm, and length of lattice elements with respect to the inner product of the lattice
- Conversion between a lattice element and its coordinates with respect to the basis of a lattice (in both directions)
- Action on lattice elements by matrices
- Creation of sublattices and superlattices, scaling of lattices
- Creation of quotient lattices (abelian group with isomorphism)
- Dual of a lattice, dual quotient of a lattice
- Arithmetic on lattices: sum, intersection, direct sum, tensor product, exterior square, symmetric square
- Conversion between lattices and \mathbf{Z} -modules and \mathbf{Q} -modules.

Several interesting lattices are directly accessible inside Magma using standard constructions, e.g., root lattices and preimages of linear codes. For each lattice, a LLL-reduced basis for the lattice is computed and stored internally when necessary and subsequently used for many operations. This gives maximum efficiency for the operations, yet all the operations are presented using the external (“user”) basis of the lattice.

9.1.2 Lattices: Properties

- Rank, determinant, basis, basis matrix, inner product matrix, Gram matrix, centre density, testing for integrality and evenness, index in a superlattice
- Minimum of a lattice (which can also be asserted)
- Kissing number of a lattice
- Theta series of a lattice
- Enumeration of all short vectors of a lattice having norm in a given range
- Enumeration of all shortest vectors of a lattice
- Enumeration of all vectors of a lattice having squared distance from a vector (possibly) outside the lattice in a given range

- Enumeration of all vectors of a lattice closest to a vector (possibly) outside the lattice
- Process to enumerate short or close vectors of a lattice thereby allowing manual looping over short vectors having norm in a given range or close vectors having squared distance in a given range
- Pure lattice of a lattice over \mathbf{Z} or \mathbf{Q}
- Construction of a fundamental Voronoi cell of a small-dimensional lattice
- Holes, deep holes and covering radius of a lattice
- Successive minima of a lattice

Magma includes a highly optimized algorithm for enumerating all vectors of a lattice of a given norm. This algorithm is used for computing the minimum, the shortest vectors, short vectors in a given range, and vectors close to or closest to a given vector (possibly) outside the lattice. As an example, the 98280 (normalized) shortest vectors of the Leech lattice Λ_{24} are constructed in 6.8 seconds. The genus of the 12-dimensional Coxeter-Todd lattice K_{12} is enumerated in 16 seconds and has 16 classes of lattices and mass $4649359/4213820620800 \approx 0.000001103359$.

9.1.3 Lattices: Reduction

- LLL reduction of lattices, basis matrices and Gram matrices (with numerous parameters)
- Seysen reduction of lattices, basis matrices and Gram matrices (for reducing a lattice and its dual simultaneously)
- Pairwise reduction of lattices, basis matrices and Gram matrices
- Orthogonalization and orthonormalization (Cholesky decomposition) of a lattice
- Testing matrices for positive or negative (semi-)definiteness

The LLL algorithm can operate on either a basis matrix or a Gram matrix (and will use the Gram method even if given a basis matrix and it is deemed appropriate) and can be controlled by many parameters (δ constant, exact de Weger integral method or Schnorr-Euchner floating point method, step and time limits, selection of methods, etc.). The LLL algorithm can reduce matrices with very large entries as well as matrices having large sizes (e.g., number of rows well over 500).

9.1.4 Lattices: Automorphisms

- Automorphism group of a lattice
- Subgroup of the automorphism group of a lattice fixing specified bilinear forms
- Determination of whether two lattices are isometric
- Determination of whether two lattices are isometric in such a way that specified bilinear forms are fixed

The computation of the automorphism group of a lattice and the testing of lattices for isometry is performed using the AUTO and ISOM programs of Bernd Souvignier. The automorphism group Co_0 of the 24-dimensional Leech lattice Λ_{24} is found in 175 seconds.

9.1.5 Lattices: Neighbors and Genera

- Computation of the p -neighbour of a lattice with respect to a given vector and prime p
- The sequence of p -neighbours a lattice at a prime p
- The transitive closure of the p -neighbour relation of a lattice at a prime p
- Spinor genus of an integral lattice, returned as a sequence of isometry representatives
- Genus of an integral lattice, returned as a sequence of isometry representatives
- Adjacency matrix of a sequence of lattices under the p -neighbour relation, for a sequence closed under the p -neighbour relation
- p -Adic Jordan form for lattices

9.1.6 Lattices: G -Lattices

- Creation of G -lattices with associated operations
- Invariant lattice of a rational matrix group and the associated action (thus yielding an integral representation of the group)
- Bravais group of a finite rational matrix group
- Invariant sublattices of finite index
- Space of invariant bilinear forms
- Positive definite invariant form of a rational matrix group
- Endomorphism ring
- Centre of the endomorphism ring
- Dimension of the space of invariant bilinear forms, the endomorphism algebra or its centre using a modular algorithm

The lattice machinery has been developed by Bernd Souvignier and Allan Steel.

9.2 Binary Quadratic Forms

Binary quadratic forms serve as a model for ideals in quadratic fields. Forms of negative discriminant are identified with lattices in the complex plane, which is the natural domain for modular forms and functions. The class of binary quadratic forms is placed here for its connections to this theory. Magma contains full functionality for forms of positive discriminant which extends beyond the applications in this context.

- Prime form, random form
- Reduction of forms
- Composition and powering
- Enumeration of reduced forms and reduced orbits
- Treatment of fundamental and nonfundamental discriminants
- Class number and class group
- Action of $SL(2, \mathbf{Z})$ and $PSL(2, \mathbf{Z})$
- Evaluation of modular functions on quadratic forms
- Discrete logarithm of a form (for imaginary quadratic fields)

10 Algebras

The three chief ways of defining algebras in Magma are in terms of a finite presentation, in terms of structure constants, or as a matrix (linear) algebra.

- Finitely presented associative algebras
- General finite dimensional algebras (defined by structure constants)
- Finite dimensional associative algebras (defined by structure constants)
- Quaternion algebras
- Group algebras
- Matrix algebras
- Finite dimensional Lie algebras (defined by structure constants)

10.1 Finitely Presented Associative Algebras

Finitely-presented associative algebras (non-commutative polynomial rings) are defined by taking R -linear combinations of elements of a semigroup or group, where R is some ring.

- Construction over a monoid, semigroup or group
- Arithmetic (free reduction only)
- Definition of left, right, two-sided ideals
- Construction of a matrix representation (Linton's vector enumerator)

There are two major tools for computing with these algebras. Linton's vector enumerator uses the Todd-Coxeter algorithm in an attempt to construct a matrix representation. If the user has some idea as to how to select ideals that might give rise to matrix representations of reasonable degree, this approach is very successful. The other approach is to apply a version of Buchberger's algorithm to construct a Gröbner basis for an ideal. This technique has been developed chiefly by Teo Mora in Genova and Ed Green in Virginia.

10.2 General Finite-Dimensional Algebras

These algebras are presented in terms of a basis for a free module M together with a set of structure constants defining the multiplication of these basis elements. It is assumed that we have an echelonization algorithm for M so that standard bases may be constructed for submodules.

- Creation of algebras in terms of structure constants
- Direct sum
- Arithmetic including Lie bracket operation
- Identities: associative, commutative, Lie, etc
- Properties of elements: idempotent, unit, zero-divisor, nilpotent
- Trace and minimal polynomial
- Creation of subalgebras, ideals and quotient algebras
- Ideal arithmetic: Sum, product, powers, intersection
- Ideal structure: Jacobson radical, maximal (minimal) left, right, two-sided ideals
- Decomposition: Simplicity, semi-simplicity, composition series

10.3 Finite-Dimensional Associative Algebras

These algebras are presented in terms of a basis for a free module M together with a set of structure constants defining the multiplication of these basis elements. It is assumed that we have an echelonization algorithm for M so that standard bases may be constructed for submodules. We shall refer to these algebras as *ASC-algebras*.

- Creation of algebras in terms of structure constants
- Direct sum
- Arithmetic including Lie bracket operation
- Properties of elements: idempotent, unit, zero-divisor, nilpotent
- Trace and minimal polynomial
- Creation of subalgebras, ideals and quotient algebras
- Ideal arithmetic: Sum, product, powers, intersection
- Centralizer, idealizer
- Characteristic ideals: Centre, commutator ideal, Jacobson radical
- Ideal structure: Maximal (minimal) left, right, two-sided ideals
- Decomposition: Simplicity, semi-simplicity, composition series
- Construction of the (left, right) regular matrix representation
- Lie algebra defined by the Lie product

Functions relating to the ideal structure (Jacobson radical, composition series, maximal and minimal ideals etc) are implemented by applying the module theory machinery to the regular representation of the algebra.

10.4 Quaternion Algebras

A quaternion algebra is a central, simple algebra of dimension four over a field. A special type for quaternion algebras is released in Magma V2.7. Support for orders over \mathbf{Z} and $k[x]$ and their ideals is provided for quaternions over the rational field \mathbf{Q} or $k(x)$. Special functions for enumeration all ideals in definite quaternion algebras over \mathbf{Q} , with connections to modular forms.

- Arithmetic of elements
- Norm, trace, and conjugation
- Minimal polynomial of elements
- Discriminant and ramified primes
- Creation of prime ideals
- Testing for principal ideals
- Enumeration of left and right ideals of an definite order over \mathbf{Z}
- Left and right orders of an ideal in a definite order over \mathbf{Z}

10.5 Group Algebras

A group algebra may be created for a finite group of moderate order over a Euclidean Domain.

- Creation of group algebras: a vector and term representation are provided allowing the construction of algebras for groups of arbitrary size.
- Arithmetic including Lie bracket operation
- Properties of elements: idempotent, unit, zero-divisor, nilpotent
- Trace and minimal polynomial
- Creation of subalgebras, ideals and quotient algebras
- Ideal arithmetic: Sum, product, powers, intersection
- Centralizer, idealizer
- Augmentation ideal, augmentation map
- Characteristic ideals: Centre, commutator ideal, Jacobson radical
- Ideal structure: Maximal (minimal) left, right, two-sided ideals
- Decomposition: Simplicity, semi-simplicity, composition series
- Construction of the (left, right) regular matrix representation

10.6 Matrix Algebras

While a matrix algebra may be defined over any ring R , most non-trivial computations require R to be an Euclidean Domain.

- Arithmetic
- Extension and restriction of coefficient ring
- Direct sum, tensor product
- Determinant (including modular algorithm), trace, characteristic polynomial, minimum polynomial
- Order of a unit (Leedham-Green algorithm)
- Canonical forms over a field: echelon, Jordan, rational, primary rational
- Canonical forms over an ED: echelon, Hermite, Smith
- Characteristic polynomial, minimal polynomial
- Properties of an element: unit, zero-divisor, nilpotent
- Standard basis for subalgebras, left, right and two-sided ideals
- Quotient algebras
- Sum, intersection, product, power of ideal
- Radical of an ideal
- Centre, commutator algebra, Jacobson radical
- Centralizer of a subalgebra in the complete matrix algebra
- Maximal (minimal) left, right, two-sided ideals
- Construction of the (left, right) regular matrix representation

The order of a unit over a finite field is found using the very efficient algorithm of Leedham-Green.

10.7 Finite-Dimensional Lie Algebras

A finite-dimensional Lie algebra L over a field K is presented in terms of a basis for a K -vector space V together with a set of structure constants defining the multiplication of these basis elements.

The major structural machinery for Lie algebras has been implemented for Magma by Willem de Graaf and is based on his ELIAS package originally written in GAP.

10.7.1 Lie Algebras: Construction and Arithmetic

- Creation of Lie algebras in terms of structure constants
- Construction of a Lie algebra from an associative algebra via the Lie bracket product
- Construction of a specified simple Lie algebra
- Direct sum
- Arithmetic
- Properties of an element: idempotent, unit, zero-divisor, nilpotent
- Trace and minimal polynomial

10.7.2 Lie Algebras: Properties and Invariants

- Test for abelian, nilpotent, solvable, restricted
- Test for simple, semisimple
- Killing form
- Adjoint representation of an element; Associated adjoint algebra
- Root system of a semisimple Lie algebra with a split Cartan subalgebra

10.7.3 Lie Algebras: Arithmetic of Subalgebras and Ideals

- Creation of subalgebras, ideals and quotient algebras
- Ideal arithmetic: Sum, product, powers, intersection
- Centre
- Centralizer, normalizer
- Jacobson radical, nil radical, solvable radical

10.7.4 Lie Algebras: Structure

- Composition series
- Derived series, lower central series, upper central series
- Cartan subalgebra, Levi subalgebra
- Maximal (minimal) left, right, two-sided ideals
- Decomposition of a semisimple Lie algebra into a direct sum of simple algebras
- Type of a simple or semisimple algebra

11 Representation Theory

This section describes facilities in Magma that relate to the representation theory of groups and associative algebras. The main topics considered include:

- Modules over an algebra
- $K[G]$ -modules
- Representations of groups
- Character theory
- Invariant theory

11.1 Modules over an Algebra

We consider a module whose elements are n -tuples over a field K with an action given by a matrix representation of an associative algebra A . We will refer to these modules as A -modules. These include $K[G]$ -modules.

The four fundamental algorithms for computational module theory are echelonization, the spinning algorithm, the meataxe algorithm and an algorithm for $\text{Hom}(U, V)$. For the important case of modules over finite fields, different representations of vector arithmetic, depending upon the field, have been implemented.

11.1.1 A -Modules: Creation

- Creation from the matrix representation of an associative algebra.
- Creation from group actions of different kinds
- Permutation module of a group corresponding to its action on the cosets of a subgroup
- $K[G]$ -modules corresponding to actions of a permutation or matrix group on a polynomial ring.

11.1.2 A -Modules: Constructions

- Extension and restriction of the field of scalars
- Direct sum
- Tensor product, symmetric square, exterior square ($K[G]$ -modules only)
- Dual ($K[G]$ -modules only)
- Induction and restriction ($K[G]$ -modules only)
- All irreducible $K[G]$ -modules of a finite soluble group where K is a finite field or field of characteristic zero
- All irreducible $K[G]$ -modules of a finite group where K is restricted to be a finite field.

11.1.3 A -Modules: Submodules and Quotient Modules

- Submodules via the spinning algorithm
- Membership of a submodule
- Basis operations
- Sum and intersection of submodules
- Quotient modules

11.1.4 A-Modules: Structure

- Splitting a reducible module (Holt-Rees Meataxe)
- Testing a module for irreducibility, absolute irreducibility
- Centralizing algebra of an irreducible module
- Composition series, composition factors, constituents
- Maximal and minimal submodules
- Jacobson radical, socle
- Socle series
- Existence of a complement of a submodule
- One complement, all complements of a direct summand
- Testing modules for indecomposability; indecomposable components
- Submodule lattice for modules over a finite field

The Magma algorithm for splitting modules (the Meataxe algorithm) is a deterministic version of the Holt-Rees algorithm and is capable of splitting modules over $\text{GF}(2)$ having dimension up to at least 20 000. The Magma meataxe is currently restricted to finite fields though it is expected that this restriction will be removed in the near future.

11.1.5 A-Modules: Homomorphisms

- Construction of $\text{Hom}(U, V)$, U and V R -modules
- Endomorphism ring of a module
- Automorphism group of a module
- Testing modules for isomorphism

Magma includes a new algorithm for the construction of $\text{Hom}(U, V)$ which is applicable to modules having dimension several hundred.

11.2 Character Theory

The character theory machinery is currently restricted to characters defined over the complex field.

- Definition of class functions
- Construction of permutation characters
- Arithmetic on class functions: sum, difference, tensor product
- Frobenius-Schur indicator
- Norm, order, kernel, centre of a character
- Properties: generalized character, character, irreducible, faithful, linear
- Induction and restriction of a character
- Decomposition of a tensor power: orthogonal components, symmetric components
- Action of a group on the characters of a normal subgroup
- Decomposition of characters
- Class matrix, structure constants for centre of group algebra
- Table of ordinary irreducible characters (Dixon-Schneider algorithm)

11.3 Invariants of Finite Groups

A module for constructing both characteristic zero and modular invariants of finite groups has been developed by Gregor Kemper and Allan Steel [9]. This includes a new algorithm for computing primary invariants that guarantees that the degrees of the invariants constructed are optimal (with respect to their product and their sum). Magma allows computation in invariant rings over ground fields of arbitrary characteristic. Of particular interest is the *modular* case, i.e., the case where the characteristic of the ground field divides the order of the group.

11.3.1 Construction of Primary and Secondary Invariants

- Permutation and matrix group actions on polynomials
- Independent homogeneous invariants of a specific degree
- Molien series
- Primary invariants having optimal degrees (with respect to their product and then sum)
- Secondary invariants of optimal degrees (using a new algorithm for the modular case)
- Efficient construction of fundamental invariants

For the 4-dimensional representation of A_5 over \mathbf{F}_2 , optimal primary invariants (of degrees 3, 5, 8 and 12) are found in 1.5 seconds. For the cyclic matrix group of order 8 generated by the 5-dimensional Jordan form over \mathbf{F}_2 , optimal primary invariants (of degrees 1, 2, 2, 4 and 8) are found in 0.6 seconds and secondary invariants with respect to these (of degrees 0, 3, 3, 3, 4, 4, 5, 5, 5, 6, 6, 6, 7, 7, 7, 7, 8, 8, 9, 9, 10 and 11) are found 14.4 seconds. This last computation took many hours with algorithms prior to those implemented in Magma V2.2.

11.3.2 The Ring of Invariants

- Invariant ring as a graded module over the algebra generated by the primary invariants and explicit construction of the isomorphism
- Invariant ring as a polynomial algebra
- Determination of algebraic relations between secondary invariants
- Module syzygies between secondary invariants
- Algebraic relations between invariants

11.3.3 Properties

- Hilbert series
- Free resolution
- Depth and homological dimension
- Attribute control of invariant ring information
- Determine whether an invariant ring is a polynomial ring or a Cohen-Macaulay ring

11.3.4 Invariants of the Symmetric Group

- Construction of elementary symmetric polynomials
- Determination of whether a polynomial is symmetric
- Presentation of a symmetric polynomial in terms of the elementary symmetric polynomials

12 Homological Algebra

12.1 Basic Algebras

A basic algebra is a finite dimensional algebra A over a field, all of whose simple modules have dimension one. In the literature such an algebra is known as a “split” basic algebra. The type in Magma is optimized for the purposes of doing homological calculations.

- Creation from a sequence of projective modules and a path tree for each module
- Creation of the basic algebra corresponding to the group algebra of a p -group over $GF(p)$.
- Arithmetic
- Extension and restriction of the coefficient ring
- Tensor product
- Opposite algebra
- Construction of modules over basic algebras
- Submodules, quotient modules, radicals and socles
- Projective covers and injective hulls
- Algebra as a right regular module over itself

12.2 Chain Complexes

Complexes of modules are a fundamental object in homological algebra. Conceptually, a complex is an infinite sequence of modules, indexed by integers, with maps between successive modules such that the composition of any two maps is zero.

- Creation of a complex from a list of A -modules
- Subcomplexes and quotient complexes
- Operations on complexes: Splice, shift, direct sum
- Exact extensions, zero extensions
- Dual of a complex
- Homology groups of a complex
- Boundary maps
- Construction of chain maps between complexes
- Composition of chain maps
- Image, kernel and cokernel of a chain map
- Predicates for chain maps: Surjection, injection, isomorphism
- Injective resolution (for modules over a basic algebra)
- Projective resolution (for modules over a basic algebra)
- Extending cohomology elements as chain maps
- Maps induced on homology by chain maps, long exact homology sequence

13 Algebraic Geometry

The algebraic geometry module includes machinery for studying general algebraic varieties and families of special curves (e.g. elliptic curves). The major categories include:

- Schemes and maps of schemes
- Rational scrolls
- Zero-dimensional schemes
- Algebraic curves
- Function fields and differentials of plane curves
- Divisor groups of plane curves
- Resolution graphs and splice diagrams
- Newton polygons
- Plane conic curves and general rational curves
- Elliptic curves
- Hyperelliptic curves
- Modular symbols
- Brandt modules
- Modular forms
- Modular curves
- Database of K3 surfaces

13.1 Schemes

This module comprises general tools for working with schemes defined by polynomial equations in affine or projective space. Such tools include Gröbner basis computation, dimension and image of maps, and linear algebra calculations formalised as linear systems on projective space. Maps between spaces may also be constructed and studied.

13.1.1 Schemes: Ambient Spaces

A scheme is contained in some ambient space, either an affine space or one of a small number of standard projective spaces. It is characteristic of these spaces is that they have some kind of polynomial ring as their coordinate ring.

- Affine and projective spaces including weighted projective space
- Ruled surfaces
- Rational scrolls
- Direct products of ambient spaces
- Points of schemes with coefficients in k -algebras

13.1.2 Schemes: Creation and Properties

- Creation of schemes by equations or implicit methods
- Changing the base ring
- Point set operations
- Projective closure and affine patches
- Global properties: Dimension, reducibility, singularity
- Prime components, primary components
- Local geometric analysis: singularities, tangent spaces
- Zero-dimensional schemes

13.1.3 Schemes: Mappings

- Construction of maps between spaces
- Coordinate manipulation functions, including birational transformations of the projective plane
- Calculation of pullbacks of schemes by maps
- Tools to enable the calculation of images of schemes by maps
- A large number of specialised types of schemes, especially curves, with added functionality

13.1.4 Schemes: Automorphisms

Automorphisms of schemes defined over a field may be constructed. The main cases where there is significant functionality is for automorphisms of affine and projective spaces.

- Construction of general automorphisms of affine space in terms of functions or matrices
- Construction of special automorphisms of affine space: translation, permutation automorphisms, Nagata automorphism, projectivities
- Construction of general automorphisms of projective space in terms of polynomials or matrices
- Construction of special automorphisms of projective space: translation, quadratic transformation
- Automorphism group of a projective space defined over a finite field

13.1.5 Schemes: Linear Systems

The *complete linear system on \mathbf{P} of degree d* is the collection of all homogeneous polynomials of degree d on \mathbf{P} , or equivalently, the degree d hypersurfaces thereby defined. A general linear system corresponds to some vector subspace of the coefficient space of a complete linear system.

- Creation of a linear system explicitly
- Creation of a linear system satisfying geometric conditions
- Properties: Sections, degree, dimension, base scheme
- Properties: Base component, base points, generic multiplicity at a point
- Complement with respect to a subsystem or scheme
- Intersection
- Maps of a linear system

13.2 General Algebraic Curves

Magma includes a general package for working with plane curves. These are interpreted as being the scheme defined by the vanishing of a polynomial defined on either an affine or projective space. Its main features are flexible tools for translating between affine and projective curves, the calculation of geometric genus of any plane curve and the explicit manipulation of divisors on curves. Many more specialised will be included in future releases. In particular, the functions and usage for different kinds of curves is converging to a standard. Most of the new curve functionality is based on our new function field machinery, but in the context of curves the results are available directly without function field knowledge.

13.2.1 Curves: Plane Curves

- Creation of affine and projective curves together with the ambient affine and projective planes
- Basic manoeuvres between affine and projective curves: projective closure, affine patches and so on
- Basic scheme-type functions: e.g., irreducibility
- Specialised data types for distinct classes of curves such as elliptic curves
- Linear systems of curves in the projective plane with assigned basepoints
- Implicitization of parametric curves
- Parametrization of rational curves
- Global invariants of curves, such as genus and dimension
- Function field and divisor computations for plane curves (see below)

13.2.2 Curves: Local Analysis

- Calculation of tangent spaces and cones
- Identification of all singularities of a curve, together with the basic analysis
- Blowups, including weighted blowups, of points on curves
- Local intersections

13.2.3 Curves: Divisors and the Riemann-Roch Theorem

The following functions apply to irreducible curves in general.

- The computation and arithmetic of the function field of a curve
- Creation of divisors and their arithmetic
- Compute the divisor of a function on a curve and other constructors
- Determine whether a divisor is principal and if so find a function with the given divisor
- Compute the Riemann–Roch space of a divisor
- Compute the Divisor Class Group of a curve defined over a finite field

13.2.4 Curves: Differentials

- Arithmetic and various, related operations
- Valuation of a differential at a place
- Divisor of a differential
- Differential spaces for given divisors (e.g. holomorphic differentials)
- Higher differentiations
- Residue of a differential at a place of degree one
- Cartier operator and representation matrix of the Cartier operator (curves over finite fields)

13.3 Rational Curves and Conics

There are two parametrisation algorithms for rational curves. The first assumes you have a rational point in hand. The second makes the anticanonical embedding whose image is a conic curve which may or may not have points.

When defined over the rational numbers, we now have a fast point finding and parametrisation algorithm for conics which have a rational point. For example, it takes 12 seconds to find a point (with reduced coordinates) on each of 100 curves of the form $ax^2 + by^2 + cz^2 = 0$ where a, b, c are prime numbers of size around 10^{100} . Finding a point on a single curve of the same form with coefficients of size around 1000 digits takes 24 seconds.

- A data type for rational curves
- A parametrisation algorithm for rational curves given a rational point
- A canonical parametrisation algorithm reducing rational curves to conics irrespective of the existence of a point
- A data type for plane conics
- A new algorithm for finding a point with small coefficients over the rationals if one exists
- Type translation from curves of genus 0 to rational curves

13.4 Elliptic Curves

Magma contains an extensive module for computing with elliptic curves. The reader should note that elliptic curves inherit from the general plane curve datatype so that all plane curve operations are applicable in addition to those listed below.

13.4.1 Elliptic Curves: Construction and Properties

- Creation of an elliptic curve over a field
- Creation of a curve with given j -invariant
- Models: Weierstrass form, integral model, minimal model, simplified model
- Arithmetic with rational points
- Extension and lifting of curves induced by maps of base rings
- Function field of the curve
- Invariants: b -invariants, c -invariants, j -invariant, discriminant
- Division polynomials, m -torsion subgroups
- Subgroups and subschemes of elliptic curves as separate types

13.4.2 Elliptic Curves: Morphisms

- Isomorphisms, isogenies and rational maps between curves, translation maps on a curve
- Operations with isogenies: degree, composition, construction with given kernel, Frobenius endomorphism
- Kernel and image of an isogeny as subgroups, image and preimage of a subgroup under an isogeny
- Operations with isomorphisms: inverses, composition
- Extension and lifting of maps induced by change of base ring of curves
- Testing for isomorphic or isogenous curves

13.4.3 Elliptic Curves: Operations over \mathbf{Q}

- Standard models and conversions
- Construction from plane curves
- Invariants: conductor, regulator, periods, and period lattice
- Tamagawa numbers
- Tate’s algorithm for computing local information, Kodaira symbols
- Twist of an elliptic curve by an integer
- Heights: local height, naive height, canonical height, height pairing
- Elliptic logarithm and p -adic elliptic logarithm
- Mordell-Weil rank, bounds on Mordell-Weil rank, analytic rank
- Mordell-Weil group and torsion subgroup
- S -integral points: determination of the finite set of affine points with denominator generated over the finite set of primes S .
- John Cremona’s database of all elliptic curves over \mathbf{Q} having conductor up to 10 000

Standard models are provided together with heights and invariants. For curves over \mathbf{Q} invariants such as conductor, regulator, and local information (Tate’s algorithm) are available. The Mordell-Weil rank and group is computed using code based on John Cremona’s `MWRANK` program. Magma takes 140 seconds to determine that the curve

$$y^2 + xy = x^3 - 215x + 1192$$

has group $\mathbf{Z}_2 \oplus \mathbf{Z} \oplus \mathbf{Z}$, and 100 seconds to determine that the curve

$$y^2 = x^3 + 36\,861\,504\,658\,225x^2 + 1\,807\,580\,157\,674\,409\,809\,510\,400x$$

has rank 13.

13.4.4 Elliptic Curves: Operations over F_q

In the case of elliptic curves defined over a finite field, specialised functions are provided for the construction and analysis of maps. The major algorithm for such curves is the SEA algorithm for point counting.

- Characterization of ordinary and supersingular elliptic curves
- Representative supersingular curve
- Enumeration of all points (small fields), random point
- Quadratic twist, all quadratic twist, all twists
- Order of a point via baby-step giant-step
- Schoof-Elkies-Atkin (SEA) algorithm for finding the order of the group of rational points
- Schoof-Elkies-Atkin (SEA) algorithm with early abort facility for searching for cryptographically secure curves
- Lercier’s extension of the SEA algorithm for finite fields of characteristic two
- Structure of the abelian group of rational points
- Zeta function of a curve
- Discrete logarithm (parallel collision search version of the Pollard rho algorithm)
- Weil pairing

For a random curve taken over a 168-bit prime field $GF(p)$, Magma takes an average of 47 seconds to determine the order of the group. In the case of a random curve taken over a 400-bit prime field the average runtime is 2200 seconds.

13.5 Hyperelliptic Curves

Magma contains a package for computing with hyperelliptic curves and their Jacobians. The development of this package was undertaken by Michael Stoll (Düsseldorf), with contribution from P. Gaudry, E. Howe and the Magma group.

The reader should note that hyperelliptic curves inherit from the general plane curve datatype so that all plane curve operations are applicable in addition to those listed below.

13.5.1 Hyperelliptic Curves: Construction and Properties

- Standard quadratic models, simplified models
- Integral model, minimal Weierstrass model, p -normal model, reduced model
- Invariants: degree, genus, discriminant
- Function field of the curve
- Extension and lifting of curves induced by maps of base rings
- Reduction types and conductors of a genus 2 curve at odd primes
- Igusa and related invariants
- Construction of a curve given its Igusa invariants
- Determination of points at infinity

13.5.2 Hyperelliptic Curves: Morphisms

- Creation of isomorphisms, automorphisms
- Arithmetic with isomorphisms: composition, inversion, evaluation
- Test for curves being GL_2 equivalent, isomorphic
- Automorphism group

13.5.3 Hyperelliptic Curves: Operations over F_q

In the case of elliptic curves defined over a finite field, specialised functions are provided for the construction and analysis of maps. The major algorithm for such curves is the SEA algorithm for point counting.

- Enumeration of all rational points (small fields)
- Random point
- Quadratic twist, all quadratic twists
- Zeta function of a curve

13.5.4 Hyperelliptic Curves: Construction of the Jacobian

- Creation of the Jacobian of a given curve as the divisor class group
- The Jacobian as a curve with defining equation
- Construction from plane curves
- Creation of points
- Normal form and addition of points
- Kummer surface of the Jacobian of a genus 2 curve

13.5.5 Hyperelliptic Curves: Operations on the Jacobian over \mathbf{Q}

Most of the functions listed here apply only in the case of Jacobians of genus 2 curves.

- Order of a point
- Height constant, canonical heights, naive height
- Height pairing matrix and regulator
- Enumeration of all \mathbf{Q} -rational points up to a given height bound (Elkies-Stahlke-Stoll method)
- 2-Selmer rank (via 2-descent)
- 2-torsion subgroup, full torsion subgroup
- Rank of the subgroup of the Mordell-Weil group generated by a given set of points on the Jacobian
- Chabauty's method for determining rational points on a curve over \mathbf{Q} with Mordell-Weil rank at most 1

13.5.6 Hyperelliptic Curves: Operations on the Jacobian over F_q

The techniques for point counting are due to P. Gaudry and R. Harley and have been implemented in Magma by P. Gaudry.

- Enumeration of all rational points (small fields), random point
- Order of a point via the Shanks or the Pollard-rho algorithms
- Baby-step giant-step algorithm for finding the order of the group of rational points
- Schoof algorithm for finding the order of the group of rational points in the case of a genus 2 curve
- Structure of the abelian group of rational points
- Weil pairing of points

13.5.7 Hyperelliptic Curves: Kummer Surface

- Construction from the Jacobian of a genus 2 curve
- The defining equation of the surface
- Creation of points
- Pseudo addition, and duplication of points
- Naive height and canonical height
- Search for rational points having three coordinates specified
- Preimage on the Jacobian of a given point on the Kummer surface

13.6 Modular Forms

Magma V2.8 includes packages developed by William Stein and David Kohel as part of a major new initiative in modular forms and modular curves computation. The central object in each package is a finite-rank module equipped with the action of a ring of Hecke operators. The space of modular forms is perhaps the most familiar example of such a Hecke module. Other examples include the space of modular symbols and the group of divisors generated by the supersingular points on the reduction of certain modular curves. This latter module can be computed in some cases using the method of Mestre and Oesterlé [13] and in general using quaternion ideal theory [7, 14], which has deep connections with the theory of Shimura curves [10].

13.6.1 Modular Forms

The modular forms package provides the following features:

- Computation of a basis of modular forms on $\Gamma_1(N)$ of any integer weight greater than one
- Computation of all newforms of given level, all of their reductions modulo a prime, and their embeddings in the complex and p -adic fields
- Arithmetic with modular forms
- Characteristic polynomials of Hecke operators
- Decomposition into Eisenstein, cuspidal, and new subspaces
- Dimension formulas

13.6.2 Modular Symbols

A program for computing with modular symbols was developed by William Stein as part of his Ph.D. thesis, and redesigned and made part of Magma during a visit to the Magma group in 1999. Algorithmic details, research applications, and references can be found in Stein's thesis [16] and subsequent work (see <http://modular.fas.harvard.edu/papers>).

- Construction of spaces of modular symbols of any character, level, and weight
- Computation of Hecke operators
- Decomposition into invariant subspaces
- Computation of certain invariants of the modular abelian varieties
- The intersection pairing on the integral homology of modular curves
- Special values of L -functions and computation of complex period lattices

13.6.3 Brandt Modules

Brandt modules provide a representation in terms of quaternion ideals of certain cohomology subgroups associated to Shimura curves $X_0^D(N)$ which generalize the classical modular curves $X_0(N)$. The Brandt module datatype is that of a Hecke module – a free module of finite rank with the action of a ring of Hecke operators – which is equipped with a canonical basis (identified with left quaternion ideal classes) and an inner product which is adjoint with respect to the Hecke operators.

Features:

- Construction of a Brandt module on the left ideal class of a definite order in a quaternion algebra over \mathbf{Q} .
- Arithmetic operations with module elements.
- Inner product of elements with respect to the canonical pairing on their parent.
- Elementary invariants of a Brandt module: Level, discriminant, conductor etc.
- Decomposition of a Brandt module under the action of Atkin–Lehner and Hecke operators.

- Eisenstein subspace, cuspidal subspace.
- Operations on subspaces: Orthogonal complement, intersection.
- Properties of subspaces: Eisenstein, cuspidal, decomposable.
- Construction of Hecke and Atkin Lehner operators.
- q -expansions associated with a pair of elements of a Brandt module.
- Determination of the dimension of a Brandt module of given level obtained using standard formulae.

13.6.4 Dirichlet Characters

The Dirichlet characters package provides support for computing with the group of homomorphism $(\mathbf{Z}/N\mathbf{Z})^* \rightarrow R^*$, where R is a ring.

- Dirichlet group over a field K as the set of rational characters from $(\mathbf{Z}/N\mathbf{Z})^*$ to \overline{K}^* .
- Computation of group generators, enumeration of elements, and representatives of Galois-conjugacy classes of characters.
- Construction and evaluation of Dirichlet characters.
- Invariants of characters, such as their conductor, modulus, and order.

Dirichlet characters have been developed in support of the new Hecke module types. Future applications to exponential sums will also be developed.

13.6.5 Hecke Modules of Eichler, Mestre, and Birch

- Construction of Hecke modules from quaternion ideals.
- Construction of Hecke modules from supersingular elliptic curves.
- Construction of Hecke modules from genera of ternary lattices.

Various alternative constructions are possible for Hecke modules associated to spaces of modular forms. Eichler [7] solved the basis problem for modular forms of square-free level using theta series of quaternion ideals. Hijikata, Pizer, and Shemanske [8] extended this to general level, and Pizer [14] proposed this as the foundation for effective computation of modular forms (see [14]). The theory is based on the matrix operators of Brandt, so we refer to the underlying Hecke module as the Brandt module. The construction is categorically equivalent to the construction using supersingular elliptic curves, but asymptotically more effective. Moreover, it generalizes to encompass the theory of Shimura curves.

A new quaternion algebra package, allowing ideal enumeration, provides the basis for the construction of Brandt modules. The equivalent construction of Mestre and Osterlé [13] via supersingular elliptic curves is also provided. The method of Birch [1] provides an alternative very efficient algorithm for computing a specific subspace of this module.

The modules of Brandt and Mestre provide the only known effective means for computing component groups of the Néron model of Jacobians of Shimura curves and modular curves (see [10] and [11]).

13.6.6 Classical Modular Forms and Functions

The standard collection of modular forms can be created as modular forms.

- Dedekind η -function
- Eisenstein series
- Theta series
- Evaluation of modular forms on binary quadratic forms
- Evaluation of Weber's f -function, Weber's f_2 -function, j -invariant, on binary quadratic forms
- Hilbert class polynomial and Weber class polynomial

13.6.7 Modular Curves

A package for computing with modular curves has been developed by David Kohel. Modular curves in Magma are a special type of plane curve. A modular curve X is defined in terms of standard affine modular equations which are stored in precomputed databases. Those modular curves presently available are defined by modular equations—a bivariate polynomial relation between the j -invariant and one of several standard functions on $X_0(N)$. These give singular, affine models for $X_0(N)$ designed for computing isogenies of elliptic curves.

Features:

- Creation of a modular curve of specified level from a database. Possible model types are *Atkin*, *Canonical* and *Classical*.
- Database of modular equations: *Atkin*, *Canonical* and *Classical*.
- Parametrization of the isogenies of an elliptic curve by points on some $X_0(N)$.
- j -invariant of a modular curve over a field.
- Base curve of a modular curve.
- Automorphisms: Atkin–Lehner involution.
- Hilbert and Weber class polynomials.

13.7 K3 Surface Database

The computation of generic families of K3 surfaces embedded in weighted projective spaces of dimension at most 6 is a project that has been running for 20 years. The result is a collection of about 400 families of K3 surfaces. From a standing start, we can make the complete computation (including correcting half a dozen mistakes in codimension 4) in a few minutes. The database of results is included in Magma (loading in a few seconds) together with functions which carry out cross-referencing analyses that in the past have taken weeks to compute by hand. In the next phase we will apply these techniques in the unknown higher codimensions.

One main point is that these results are a major step on the way to understanding Fano 3-folds and their birational automorphisms. These are the 3-dimensional analogues of rational curves or Del Pezzo surfaces and are a very exciting research frontier.

Another point is that these families exhibit large Gorenstein rings with as yet unknown structure. The search for structure theorems for Gorenstein rings occupied much of the 70s after the Buchsbaum–Eisenbud structure theorem in codimension 3. However no substantial new cases were discovered. These examples, not available in the 70s, are a major motivation for a renewed assault on this problem.

- Raw data for the K3 database in codimension at most 4
- Functions for interrogating the database
- Functions for modifying the database in light of new geometrical constructions

13.8 Resolution Graphs and Splice Diagrams

- Creation of resolution graphs and their vertices, either implicitly using resolution routines or Newton Polygons, or explicitly by listing the required data
- Calculation of numerical data associated to blowups, e.g., the canonical class of certain rational surfaces
- The Cartan matrix of a graph and associated calculations such as the contribution to the genus of a plane curve of a singularity having a given graph as its resolution
- Surgery on resolution graphs such as cutting an edge of a graph
- Creation of splice diagrams implicitly and explicitly
- Edge determinants and linking numbers of splice diagrams
- Test for regularity of splice diagrams
- Translation between resolution graphs and splice diagrams

These decorated graphs encode data generated by the resolution machinery. At present they are only attached to resolutions of plane curve singularities, but in due course they may be extended to resolutions of linear systems, surface singularities, special fibres in curve fibrations or other geometrical contexts. The package includes a resolution function for curves adapted to present its output in resolution graph format. This does not supersede other resolution machinery such as Puiseux expansions or genus calculations, but is intended as a complementary tool for those users with this kind of geometric background.

14 Finite Incidence Structures

The categories in this variety are chosen to represent the fundamental incidence structures.

- Graphs—directed and undirected
- Incidence structures
- Designs
- Finite planes
- Incidence geometry

14.1 Enumerative Combinatorics

- Factorial
- Binomial, multinomial coefficients
- Stirling numbers of the first and second kind
- Fibonacci numbers, Bernoulli numbers, Harmonic numbers, and Eulerian numbers
- Number of partitions of n
- Enumeration of restricted and unrestricted partitions
- Sets of subsets, multisets, and subsequences of sets
- Permutations of sets (as sequences)

14.2 Graphs

Graphs may be directed or undirected. In addition, their vertices and/or their edges may be labelled.

- Directed and undirected graphs
- Optional vertex and edge labels
- Operations: union, join, product, contraction, switching etc
- Standard graphs: complete, complete bipartite, k -cube
- Properties: connected, regular, eulerian
- Algebraic invariants: Characteristic polynomial, spectrum
- Diameter, girth, circumference
- Connectedness, paths and circuits
- Spanning trees, cut-vertices
- Cliques and maximal cliques, independent sets
- Chromatic number, chromatic index, chromatic polynomial
- Graphs from groups: Cayley graph, orbital graph
- Automorphism group (B. McKay's algorithm), canonical labelling
- Testing of pairs of graphs for isomorphism
- Group actions on a graph: orbits and stabilizers of vertex and edge sequences
- Symmetry properties: vertex transitive, edge transitive, k -arc transitive, distance transitive, distance regular
- Intersection numbers of a distance regular graph

Brendan McKay's automorphism program (**nauty**) is used for computing automorphism groups and for testing pairs of graphs for isomorphism. In accordance with the Magma philosophy, a graph may be studied under the action of an automorphism group. Using the G -set mechanism an automorphism group can be made to act on any desired set of objects derived from the graph.

14.3 Incidence Structures and Designs

General incidence structures provide a universe in which families of incidence structures satisfying stronger conditions (linear spaces, t -designs, etc) reside.

- Creation of a general incidence structure, near-linear space, linear space, design
- Difference sets: standard difference sets, development
- Hadamard designs, Witt designs
- Unary operations: complement, contraction, dual, residual
- Binary operations: sum, union
- Invariants for an incidence structure: point degrees, block degrees, covalence
- Invariants for a design: replication number, order, covalence, intersection numbers, Pascal triangle
- Properties: balanced, complete, uniform, self-dual, simple, Steiner
- Near-linear space operations: connection number, point and line regularity, restriction
- Resolutions, parallelisms, parallel classes
- Graphs and codes from designs: block graph, incidence graph, point graph, linear code
- Automorphism group (J. Leon’s algorithm), isomorphism testing
- Group actions on a design: orbits and stabilizers of points and blocks
- Symmetry properties: point transitive, block transitive

Tools are provided for constructing designs from difference sets, Hadamard matrices, codes and other designs. The standard families of difference sets are incorporated. A major feature is the ability to compute automorphism groups and to test pairs of incidence structures for isomorphism.

14.4 Finite Planes

Although finite planes correspond to particular families of designs, separate categories are provided for both projective and affine planes in order to exploit the rich structure possessed by these objects.

- Creation of classical and non-classical finite projective and affine planes
- Subplanes, dual of a projective plane
- Numerical invariants: order, p -rank
- Properties: Desarguesian, self-dual
- Parallel classes of an affine plane
- k -arcs: testing, complete, tangents, secants, passants
- Conics: through given points, knot, exterior, interior
- Unitals: testing, tangents, feet
- Affine to projective planes and vice versa
- Related structures: design, incidence matrix, incidence graph, linear code
- Collineation group, isomorphism testing (optimized algorithm for projective planes)
- Central collineations: testing, groups
- Group actions on a plane: orbits and stabilizers of points and lines
- Symmetry properties: point transitive, line transitive

Apart from elementary invariants, a reasonably fast method is available for testing whether a plane is desarguesian. Among special configurations of interest, a search procedure for k -arcs is provided. A specialized algorithm developed by Jeff Leon is used to compute the collineation group of a projective plane while the affine case is handled by the incidence structure method. The collineation group (order $2^3 3^8$) of a “random” projective plane of order 81 supplied by Gordon Royle was found in 1202 seconds. As with graphs and designs the G -set mechanism gives the action of the collineation group on any appropriate set.

14.5 Incidence Geometry

Magma 2.7 contains facilities for creating and computing with incidence geometries and coset geometries. These have been developed by Dimitri Leemans (Brussels).

The Magma Incidence Structure type comprises a set of points and a set of blocks together with an incidence relation. Following Buekenhout, we define a more general object as follows: An *incidence geometry* is a 4-tuple $\Gamma = (X, *, t, I)$ where

- X is a set of *elements*;
- I is a non-empty set whose elements are called *types*;
- $t : X \rightarrow I : x \mapsto t(x)$ is a *type function* which maps an element to its type;
- $*$ is an *incidence relation* that is a reflexive and symmetric relation such that $\forall x, y \in X$, $x * y$ and $t(x) * t(y) \Rightarrow x = y$.

We also introduce group-geometry pairs or *coset geometries*. Roughly speaking, these are geometries constructed from a group and some of its subgroups in the following way. Let I be a finite set and let G be a group together with a finite family of subgroups $(G_i)_{i \in I}$. We define the *incidence geometry* $\Gamma = \Gamma(G, (G_i)_{i \in I})$ as follows. The set X of *elements* or *varieties* of Γ consists of all cosets gG_i , $g \in G$, $i \in I$. We define an *incidence relation* $*$ on X by:

$$g_1G_i * g_2G_j \text{ iff } g_1G_i \cap g_2G_j \text{ is non-empty in } G.$$

$\Gamma(G, (G_i)_{i \in I})$ may also be called a *group-geometry pair*.

14.5.1 Incidence Geometries

- Creation of incidence geometries
- Set of types, rank
- Diagram, incidence graph, elements
- Residue, truncation
- Properties: flag-transitive geometry, residually connected, firm, thin, thick
- Automorphism group
- Correlation group

14.5.2 Coset Geometries

- Creation of coset geometries
- Conversion of an incidence geometry to a coset geometry
- Set of types, rank
- Residue, truncation
- Properties: flag-transitive geometry, residually connected, firm, thin, thick,
- Borel subgroup
- Maximal parabolic subgroups

15 Error-correcting Codes

15.1 Linear Codes

Currently, the coding theory module is designed for linear codes over finite fields and the residue class ring \mathbf{Z}_4 , though this will be extended to linear codes over general finite rings in the near future.

15.1.1 Linear Codes: Creation

- Creation from a subspace of a vector space
- Creation in terms of a generator matrix
- Creation from a design
- Creation from a finite plane
- Construction of a cyclic code given the generator polynomial
- Construction of a cyclic code given the roots of the generator polynomial
- Universe code, repetition code, zero code
- Random linear code

15.1.2 Linear Codes: Elementary Operations

- Combining codes: sum, intersection
- Combining codes: (external) direct sum, Plotkin sum
- Combining codes: concatenation
- Modifying a code: augment, extend, expurgate, lengthen, puncture, shorten, etc
- Changing the alphabet: extend field, restrict field, subfield subcode, trace code

15.1.3 Linear Codes over Finite Fields: Bounds

- BCH bound on minimum distance
- Upper and lower bounds on the cardinality of a largest code having given length and minimum distance
- Upper asymptotic bounds on the information rate

15.1.4 Linear Codes over Finite Fields: Best Known Codes

Magma V2.8 incorporates a database containing constructions of the best known linear codes over F_2 of length up to 256. The codes of length up to 36 are optimal. The database is complete in the sense that it contains a construction for every set of parameters. Thus the user has access to 33,152 best-known binary codes. Similar databases for other small fields will be added in the near future.

The Magma BKLC database makes use of the tables of bounds compiled by A.E. Brouwer. It should be noted that the Magma BKLC database is unrelated to the similar (but rather incomplete) BKLC database forming part of GUAVA, a share package in GAP3. A significant number of entries in the Magma BKLC database provide better codes than the corresponding ones listed in the Brouwer tables.

The construction of the Magma BKLC database has been undertaken by John Cannon (Sydney), Markus Grassl (Karlsruhe) and Greg White (Sydney).

15.1.5 Linear Codes over Finite Fields: Basic Properties

- Standard form
- All information sets of a code
- Idempotent of a cyclic code
- Properties: cyclic
- Properties: even, doubly even, equidistant
- Properties: self-dual, weakly self-dual
- Properties: perfect, nearly perfect
- Properties: maximum-distance separable, equidistant
- Properties: optimal for the Griesmer bound
- Properties: projective
- Determine whether two codes are equivalent

15.1.6 Linear Codes over \mathbf{Z}_4 : Basic Properties

- Standard form
- Gray map
- Lee weights
- Properties: cyclic
- Properties: even, self-dual, weakly self-dual

15.1.7 Linear Codes over Finite Fields: Construction of Families

- Hamming code
- Reed-Muller codes
- Ordinary and extended Golay codes
- Quadratic residue code
- Alternant codes
- BCH code
- Goppa code
- Reed-Solomon, generalized Reed-Solomon codes
- Srivastava, generalized Srivastava codes
- Justesen code
- Fire code
- Simplex code
- Chien-Choy code

15.1.8 Linear Codes over \mathbf{Z}_4 : Construction of Families

- Kerdock code
- Preparata codes

15.1.9 Linear Codes over Finite Fields: Weight Distribution

- Minimum weight (Zimmermann algorithm)
- Weight distribution, weight enumerator
- MacWilliams transform
- Complete weight enumerator, MacWilliams transform
- Number of words of designated weight
- Number of words of constant weight
- List all words of designated weight
- List all words of constant weight
- Coset weight distribution, covering radius, diameter

Carefully crafted algorithms are provided for computing the minimum weight and weight distribution. For example, computation of the minimum weight of a $[96, 60, 8]$ code takes 77 seconds; computation of the weight distribution of the $[64, 22, 16]$ Reed-Muller code ($r = 2, m = 6$) takes 1.4 seconds.

15.1.10 Linear Codes over \mathbf{Z}_4 : Weight Distribution

- Minimum weight
- Weight distribution
- Complete weight enumerator
- Symmetric weight enumerator
- Lee weight enumerator
- Hamming weight enumerator

15.1.11 Linear Codes over Finite Fields: Decoding Algorithms

- Syndrome decoding
- Alternant decoding

15.1.12 Linear Codes over Finite Fields: Automorphism Group

- Automorphism groups of linear codes over $\text{GF}(p)$ (prime p), $\text{GF}(4)$
- Testing pairs of codes for isomorphism over $\text{GF}(p)$ (prime p), $\text{GF}(4)$
- Group actions on a code

Automorphism groups may be computed over any field \mathbf{F}_p , p a prime, and \mathbf{F}_4 , again using Leon's PERM package.

16 Cryptography

16.1 Pseudo-Random Sequences

Magma provides tools for the creation and analysis of pseudo-random bit sequences. The universe of these sequences is generally $\text{GF}(2)$. However, some functions, such as Berlekamp-Massey, apply to sequences defined over arbitrary finite fields.

- Generation of n elements of a Linear Feedback Shift Register sequence (LFSR sequence)
- Next state of a LFSR sequence
- Characteristic polynomial of a LFSR sequence (BerlekampMassey algorithm)
- Shrinking generator
- Random sequence via the RSA random bit generator
- Random sequence via the Blum Blum Shub generator
- Auto-correlation of a binary sequence
- Cross correlation of two binary sequences
- Decimation of a sequence

17 Mathematical databases

Magma includes a growing number of mathematical databases. Typically, such a database contains a complete classification of all structures of some given type up to a specified bound. A number of these databases are an integral part of algorithms installed in Magma. The current databases include:

- **Cunningham Factorizations:** A database of factorizations of 199,044 integers of the form $a^n \pm 1$. This database includes the factorizations of integers of the form $a^n \pm 1$, $a \leq 12$, produced by Sam Wagstaff and collaborators. In addition, it includes factorizations of integers of the form $p^n \pm 1$, for primes $13 \leq p \leq 1000$ as tabulated by Brent and collaborators.
- **Elliptic Curves:** The database of all elliptic curves of conductor up to 5300, constructed by Cremona.
- **Galois Polynomials:** For each transitive group G with degree between 2 and 15, the database contains a univariate polynomial over the integers which has G as its Galois group.
- **Small Groups:** A table of groups of order less than 1000, excluding the groups of order 512 and 768, prepared by Besche, Eick, Newman and O'Brien.
- **Perfect groups :** The database of perfect groups of order up to a million constructed by Holt and Plesken.
- **Simple Groups:** A database containing a presentation, the conjugacy classes and maximal subgroups for each simple group of order less than a million. The database was prepared by Jamali, Robertson and Campbell.
- **Transitive Groups:** The transitive permutation groups of degree up to 22. The transitive groups of degree up to 15 were determined by Butler and McKay while the classification was extended to degree 22 recently by Hulpke.
- **Primitive Groups:** The table of primitive groups of degree up to 50 prepared by Sims.
- **Irreducible Soluble Groups:** The irreducible soluble subgroups of $\text{GL}(n, p)$ for $n > 1$ and $p^n < 256$, as classified by Short.
- **Permutation Representations:** A collection of finite groups given in terms of permutation representations. A particular group is included if:
 - It is an ‘interesting’ group. In practice this means a sporadic simple group or a close relative of such; or
 - It is representative of some class of groups which is useful for testing conjectures and algorithms.
- **Matrix Representations:** A collection of modular representations of simple groups (mainly sporadic groups) and coverings of simple groups. This collection is a subset of the Parker database of modular representations.
- **Maximal Finite Subgroups of $\text{GL}(n, \mathbf{Z})$:** The maximal finite subgroups of $\text{GL}(n, \mathbf{Z})$ for $n = 2, 3, 4, 5$, as constructed by R. Bülow.
- **Irreducible polynomials:** A database of sparse irreducible polynomials over $\text{GF}(2)$ for all degrees up to 11000.

18 Documentation

Magma has an extensive online help system. It includes specially-written material on all aspects of Magma, suitable for the first-time user, and complete access to the Magma Handbook which provides a full description of all the facilities of the system. The help nodes are structured as a tree, allowing a full-scale browsing facility in addition to simple help requests.

In addition to the online help, there are five main components to the printed documentation:

- J. Cannon and C. Playoust: **First Steps in Magma**, 16 pages
- J. Cannon and C. Playoust: **An Introduction to Algebraic Programming with Magma: The Language**, 350 pages
- W. Bosma, J. Cannon and C. Playoust: **An Introduction to Algebraic Programming with Magma: The Categories**, 500 pages
- W. Bosma and J. Cannon: **The Magma Handbook**, Five Volumes, 2200 pages.
- W. Bosma, J. Cannon et al.: **Solving Problems with Magma**, 190 pages

References

- [1] B. J. Birch. Hecke actions on classes of ternary quadratic forms. *Computational number theory (Debrecen, 1989)*, 191–212, de Gruyter, Berlin, 1991.
- [2] W. Bosma, J.J. Cannon and A. Steel. Lattices of Compatibly Embedded Finite Fields, *J. Symb. Comp.* **24** (1997), pp. 351–369.
- [3] J. Buchmann and D. Squirrel. Kernels of Integer Matrices via Modular Arithmetic, *Technical Report No. TI-4/99, University of Darmstadt*, <ftp://ftp.informatik.tu-darmstadt.de/pub/TI/TR/TI-99-04.ps.gz>.
- [4] J. Buchmann, M.J. Jacobson Jr., and E. Teske. On Some Computational Problems in Finite Abelian Groups. *Mathematics of Computation*, 66:1663–1687, 1997.
- [5] J. Cremona. *Algorithms for modular elliptic curves*. Second edition. Cambridge University Press, Cambridge, 1997.
- [6] J. Della Dora, C. Dicrescenzo and D. Duval. About a new method for computing in algebraic number fields, *Proc. EUROCAL '85, Vol. 2. Lecture Notes in Computer Science* (1985), **204**, 289–290.
- [7] M. Eichler. The basis problem for modular forms and the traces of the Hecke operators. *Modular functions of one variable, I*, 75–151, Lecture Notes in Math., **320**, Springer, Berlin, 1973.
- [8] H. Hijikata, A. Pizer, T. Shemanske. The basis problem for modular forms on $\Gamma_0(N)$. *Proc. Japan Acad. Ser. A Math. Sci.* **56** (1980), no. 6, 280–284.
- [9] G. Kemper and A. Steel. Some Algorithms in Invariant Theory of Finite Groups, *Proceedings of the Euroconference on Computational Methods for Representations of Groups and Algebras*, P. Dräxler, G.O. Michler, and C.M. Ringel, eds., Birkhäuser, Basel, 1998.
- [10] D. R. Kohel. Hecke module structure of quaternions. *Class Field Theory – it's Centenary and Prospect*, K. Miyake, ed., The Advanced Studies in Pure Mathematics Series, Math. Soc. Japan, to appear.
- [11] D. R. Kohel and W. A. Stein. Component groups of quotients of $J_0(N)$. *Proceedings of ANTS IV*, to appear.
- [12] B. A. LaMacchia and A. M. Odlyzko. Solving Large Sparse Linear Systems over Finite Fields, *Advances in Cryptology: Proceedings of Crypto '90*, A. Menezes, S. Vanstone, eds., *Lecture Notes in Computer Science* **537**, Springer-Verlag, NY (1991), 109–133.

- [13] J.-F. Mestre. Sur la méthode des graphes, Exemples et applications, *Proceedings of the international conference on class numbers and fundamental units of algebraic number fields*, Nagoya University, 1986, 217–242.
- [14] A. Pizer. An algorithm for computing modular forms on $\Gamma_0(N)$, *J. Algebra* **64** (1980), no. 2, 340–390.
- [15] A. Steel. A New Algorithm for the Computation of Canonical Forms of Matrices over Fields, *J. Symb. Comp.*, **24** (1997), 409–432.
- [16] W. A. Stein. *Explicit approaches to the theory of modular abelian varieties*. Ph.D. thesis, University of California, 2000.