MAGMA

COMPUTER • ALGEBRA

Overview of Magma V2.13 Features

Computational Algebra Group

University of Sydney

January 17, 2024

Contents

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://magma.maths.usyd.edu.au/` – 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
- Semigroups and Monoids
- Rings and their Fields
- Commutative Rings
- Linear Algebra and Module Theory

- Lattices and Quadratic Forms
- Algebras
- Representation Theory
- Homological Algebra
- Lie Theory
- Algebraic Geometry
- Finite Incidence Geometry
- Differential Galois Theory
- Error-correcting Codes
- Cryptography
- 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 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
- Black-box groups
- Braid groups
- Congruence subgroups of $PSL(2, \mathbf{R})$

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

3.1 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.

3.1.1 Construction

- – Permutation representations for classical groups, e.g. PGL(n, q), PSp(n, q), 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 (Leedham-Green & Murray (2002)),

3.1.2 Base and Strong Generating Set

- – Sims-Schreier algorithm for constructing a base and strong generating set (BSGS)
- – Random Schreier algorithm for constructing a BSGS (Monte Carlo algorithm)
- – 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.

3.1.3 Elementary Properties

The functions listed in this section require a BSGS to be computed.

- Order
- Exponent
- Whether abelian, nilpotent, soluble etc
- Whether perfect, simple etc.

3.1.4 Conjugacy Classes

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 $2^{12}.(SL_2(4) \wr S_3)$ of degree 4096 and order $5, 308, 416, 000$ are computed in 4 seconds.

- 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

3.1.5 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)

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. that representation

3.1.6 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 (Schönert-Seress algorithm)
- Homomorphisms induced by actions systems of imprimitivity
- Induced actions on G-sets
- Action properties: Semiregular, regular, transitive, primitive, Frobenius
- Permutation representation on the cosets of a subgroup
- Fast tests for the alternating/symmetric group

3.1.7 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 (1986). 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.

3.1.8 Normal Structure

- Derived subgroup, derived series, soluble residual
- Lower central series, nilpotent residual
- p-core, Fitting subgroup, soluble radical (Unger's 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

3.1.9 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 on given generators (for groups of moderate order)
- Presentation on strong generators
- Presentation of quotient by a normal subgroup

A variety of methods are used for quotient constructions. Quotients by abelian groups use the algorithm of Luks & Seress (1997). Finitely presented quotients use a combination of the Schreier-Todd-Coxeter-Sims method of Leon (1980) and the presentation implicit in Sims' verification of a BSGS (see Gebhardt (2000)).

3.1.10 Subgroup Structure

- Maximal subgroups
- 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 V2.13 contains an implementation of a a very powerful algorithm 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.

3.1.11 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.

3.1.12 Cohomology and Representations

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

3.1.13 Databases

- Transitive groups up to degree 30 (Butler, Hulpke)
- Primitive groups up to degree 2499 (Sims, Roney-Dougal & Unger, Roney-Dougal)
- Irreducible soluble subgroups of GL(n, p) for $p^n < 256$ (Short)
- Almost simple groups of order less than 1.6×10^8, plus M_{24}, HS, $L_6(2)$, J_3, McL, $Sz(32)$ stored with their automorphism groups and maximal subgroups
- A collection of permutation representations of some sporadic simple groups
- Representations of ATLAS groups from the Birmingham ATLAS of finite group representations (R. Wilson).

In V2.13 the database of almost simple groups is augmented by the ability to compute automorphism groups and maximal subgroups for alternating groups up degree 999, families of low degree classical groups (Holt & Roney-Dougal): $L_2(q)$, $L_3(q)$, $L_4(q)$ and $L_5(q)$ for all q, $S_4(q)$, $U_3(q)$ and $U_4(q)$ for all q, $L_d(2)$ for $d \leq 14$, and the following groups: $L_6(3)$, $L_7(3)$, $U_6(2)$, $S_8(2)$, $S_{10}(2)$, $O_8^{\pm}(2)$, $O_{10}^{\pm}(2)$, $S_6(3)$, $O_7(3)$, $O_8^{-}(3)$, $G_2(4)$, $G_2(5)$, ${}^3D_4(2)$, ${}^2F_4(2)'$, Co_2, Co_3, He, Fi_{22}.

3.2 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.

3.2.1 Construction

A matrix group is always constructed as a subgroup of the appropriate general linear group, $GL(n, R)$.

- Generators for linear groups: $\mathrm{GL}(n, q)$, $\mathrm{SL}(n, q)$
- Generators for symplectic groups: $\mathrm{Sp}(n, q)$
- Generators for unitary groups: $\mathrm{GU}(n, q)$, $\mathrm{U}(n, q)$
- Generators for orthogonal groups: $\mathrm{GO}(2n{+}1, q)$, $\mathrm{SO}(2n{+}1, q)$, $\Omega(2n{+}1, q)$, $\mathrm{GO}^+(2n, q)$, $\mathrm{SO}^+(2n, q)$, $\Omega^+(2n, q)$, $\mathrm{GO}^-(2n, q)$, $\mathrm{SO}^-(2n, q)$, $\Omega^-(2n, q)$
- Generators for all exceptional families of groups of Lie type except E_8.
- Direct product, tensor wreath product, tensor power, exterior square
- Construction of semi-linear groups
- Group obtained by applying a homomorphism $\phi : R \to S$ to the matrix coefficients.
- Group obtained by restricting the matrix coefficients to a subring of R.

3.2.2 Arithmetic with Elements

- Random generation of elements (Leedham-Green & Murray (2002))
- Invariants of a matrix: Trace, determinant, minimal and characteristic polynomials, Jordan form, rational canonical form
- Order of a matrix (Leedham-Green algorithm for finite fields)
- Test whether a matrix has infinite order

3.2.3 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
- Orbit representatives and orbit lengths for the action of a matrix group (defined over a prime field) on the k-dimensional subspaces of the natural vector space
- Estimate the size of the orbit of a given subspace
- Approximation to the stabilizer of a given subspace
- Enumerate number of k-dimensional subspaces fixed under the action of an element of $GL(d, q)$
- Homomorphism induced by action of a reducible group on a G-invariant submodule and its quotient module
- Homomorphism induced by action on an orbit of vectors or subspaces

3.2.4 Base and Strong Generating Set

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.

- Random Schreier algorithm for constructing a BSGS
- Todd-Coxeter Schreier algorithm for constructing a BSGS
- Murray-O'Brien base selection strategy

3.2.5 Elementary Properties

With the exception of the first, all of the functions listed in this section require the group to be finite and a BSGS to be computed.

- Whether finite or infinite (only for groups defined over **Z** or **Q**)
- Order
- Exponent
- Whether abelian, nilpotent, soluble etc
- Whether perfect, simple etc.

3.2.6 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

3.2.7 Subgroup Constructions

- Construction of a subgroup in terms of generators
- 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
- Computing subgroup normalizers

3.2.8 Normal Structure

- Derived subgroup
- Soluble residual
- Centre, Fitting subgroup
- Derived series, upper central series, lower central series
- Soluble radical, 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)

3.2.9 Standard Quotients

- Maximal abelian quotient, elementary abelian quotient
- Maximal p-quotient, nilpotent quotient
- Maximal soluble quotient
- Presentation on strong generators
- Quotient by soluble radical

3.2.10 Automorphisms

In V2.13, Holt's algorithm for computing automorphism groups and testing isomorphism may be applied to matrix groups with BSGS.

3.2.11 Cohomology and Representations

The Magma machinery for matrix groups together with fast Gröbner basis techniques (see below) provide a very efficient algorithm for computing a Cohen-Macaulay basis for the ring of invariants together with its syzygies.

- Character table
- KG-module corresponding to an elementary abelian section
- Molien series
- Ring of invariants

3.2.12 Aschbacher Analysis

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 $\mathrm{GL}(d, q)$ 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.

- 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 $\mathrm{GL}(d, q)$ 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.
- Test whether a matrix group G is tensor-induced.
- 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 $G \leq \mathrm{GL}(d, K)$ 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.
- Roney-Dougal's algorithm for determining conjugacy of subgroups of $\mathrm{GL}(d, q)$

3.2.13 Databases of Matrix Groups

- Maximal finite subgroups of $GL(n, \mathbf{Q})$ for n up to 31
- The finite absolutely irreducible subgroups of $\mathrm{GL}_n(\mathcal{D})$ where $\mathcal{D}$ is a definite quaternion algebra whose centre has degree d over $\mathbf{Q}$ and $nd \leq 10$
- Irreducible subgroups of $GL(n, p)$ where p is prime and $p^n < 2500$.
- Representations of ATLAS groups from the Birmingham ATLAS of finite group representations (R. Wilson).

3.3 Constructive Recognition

Given generators for a finite group, we may be able to determine an isomorphism to a "well-known" group using black-box group methods. Magma is developing its capabilities in this area. In V2.13 the following are available.

- Alternating and Symmetric groups using the algorithm of Bratus & Pak (2000), implemented by Holt.
- Alternating and Symmetric groups using the algorithm of Beals *et al* (2003), implemented by Roney-Dougal.
- $\mathrm{SL}(d, q)$ and $\mathrm{PSL}(d, q)$ groups using the algorithm of Kantor & Seress (2001), implemented by Brooksbank.

3.4 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).

3.4.1 Free Groups

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

3.4.2 Construction

- Construction as a quotient of a free group
- Standard groups: S_n, A_n, dihedral groups, Coxeter groups, braid groups
- Permutation groups, matrix groups, polycyclic groups as fp-groups
- Direct product, free product
- Maximal central extension

3.4.3 Arithmetic on Elements

- Arithmetic (free reduction only on words)
- Substring operations on words
- Definition of and calculation with homomorphisms

3.4.4 Basic Properties

Determining global properties of an fp-group is known to be intrinsically difficult. If it is suspected that a given fp-group is finite, a function is provided that will attempt to determine the order of the group. While this function basically employs coset enumeration, it does so in a fairly sophisticated manner so that it is able to handle groups that are much too large for a coset enumeration of the trivial subgroup to succeed.

3.4.5 Quotients

- Abelian quotient, elementary abelian quotient
- p-quotient
- Process version of p-quotient allowing the user complete control over its execution
- Nilpotent quotient (W. Nickel's algorithm)
- Soluble quotient (Plesken-Brückner algorithm)
- Process version of the soluble quotient allowing the user complete control over its execution
- Natural homomorphism onto any of the above standard quotients
- Equivalence classes of homomorphisms to an arbitrary permutation group with application to perfect quotients
- Process version of search for equivalence classes of homomorphisms allowing the user complete control over its execution
- Kernel of of the natural homomorphism onto any standard quotient (provided that the quotient is not too large)

The p-quotient program has been developed over a number of years by 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.

The computation of equivalence classes of homomorphisms to a permutation group uses a well known backtrack algorithm. Volker Gebhardt's implementation of this algorithm is capable of determining all classes of homomorphisms from a 2- or 3-generator group to a permutation group of order up to 10^8 in reasonable time.

3.4.6 Constructing a Subgroup

- Construction of a subgroup in terms of generators
- Construction of a subgroup in terms of a coset table
- Coset enumeration (Todd-Coxeter procedure)
- Process version of coset enumeration allowing the user complete control over its execution
- Schreier generators for a subgroup
- Presentation for a subgroup (Reidemeister-Schreier rewriting)

Coset enumeration is performed using George Havas's ACE version of the Todd-Coxeter procedure. It has the capability of enumerating up to one hundred million cosets on a sufficiently large machine.

3.4.7 Operations on Subgroups of Finite Index

The fp-group package also includes a collection of functions for computing with subgroups of (small) finite index represented by coset tables. Hence the operations in this group assume that the subgroup has finite index and that it is possible to enumerate its cosets.

- Normal closure
- Membership and equality of subgroups
- Core, intersection and normalizer
- All maximal (minimal) overgroups of a subgroup
- Test for conjugacy, maximality, normality
- Schreier system and Schreier coset map

3.4.8 Enumeration of Subgroups

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.*

- Enumeration of low index subgroups (Sims backtrack algorithm)
- Process version of low index subgroups to return subgroups one at a time

3.4.9 Simplifying a Presentation

- Automatic simplification of a presentation
- Interactive simplification of a presentation via a Tietze process
- Tietze process: Eliminate specified generators
- Tietze process: Control of substring searching
- Bijection between original and simplified presentations

3.4.10 Actions

- Actions on coset spaces (Todd-Coxeter procedure)
- Actions on vector spaces (Linton vector enumeration)
- Permutation representation on the cosets of a subgroup

3.4.11 Representation Theory

It is frequently useful to construct the G-modules corresponding to the conjugation action of a finitely presented group G on an elementary abelian section. This can be useful when attempting to construct normal subgroups of G.

- Determine the finite primes dividing the order of the abelian quotient of a subgroup A of a fp-group G
- Given subgroups A and B defining an abelian section of G, construct the G-module corresponding to the conjugation action of G on the maximal p-elementary abelian quotient of A/B
- Given a map f from a normal subgroup A of G onto the G-module M corresponding to the conjugation action of G on the maximal p-elementary abelian quotient of an abelian section A/B of G and a submodule N of M, compute the preimage of N under f using a fast pullback method.

3.4.12 Databases of Finitely Presented Groups

- Fundamental groups of small-volume closed hyperbolic 3-manifolds (Dunfield & Thurston, based on Hodgson & Weeks' census of manifolds).

3.5 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 [9].

3.6 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 **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.

3.7 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.

3.7.1 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
- Wreath products
- Product, inverse, conjugate, commutator for elements; improved collector, much faster, better complexity
- Element normal form and equality testing
- Element order
- Random element generation

3.7.2 Polycyclic Groups: Basic Invariants

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

3.7.3 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)

3.7.4 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

3.8 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.

3.8.1 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 soluble groups contained in the Small Groups Library developed by Besche, Eick and O'Brien. This database contains all groups of order up to 2000, except the groups of order 1024, and a number of infinite series of larger groups.

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.

3.8.2 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

3.8.3 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.

3.8.4 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

3.8.5 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.

3.8.6 Automorphisms and Representations

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

3.9 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.

3.9.1 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)

3.9.2 Normal Structure

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

3.9.3 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)

3.9.4 Character Theory

- Degrees of irreducible characters (Slattery's algorithm)
- Character Table (Conlon's algorithm)

3.10 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
- Definition of homomorphisms whose domain or codomain is an RWS group

3.11 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
- Definition of homomorphisms whose domain or codomain is an automatic group

3.12 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 (Leedham-Green & Murray (2002))

3.13 Braid Groups

This category comprises the family of all braid groups. Note that this special class of finitely presented groups has a solvable word problem. Recently, braid groups have received some interest as possible sources of cryptosystems.

The Magma implementation by Volker Gebhardt supports both Artin's original presentation and the band generator presentation introduced by Birman, Ko and Lee. Elements can be defined as words in the generators or as products of simple elements for either presentation. All possible representations of elements can be used simultaneously; conversions are done automatically if necessary, completely transparent to the user.

3.13.1 Constructing and Accessing Braid Groups

- Definition of a braid group on n strings.
- Controlling default presentation, print format for elements and element representation used for group operations.

3.13.2 Constructing and Accessing Elements

- Identity element and fundamental element.
- Artin generators and band generators.
- Generation of pseudo random elements.
- Representations of an element as words and as products of simple elements for Artin presentation and band generator presentation.
- Infimum, supremum and canonical length of an element with respect to either presentation.

3.13.3 Normal Forms of Elements

- Left and right normal form with respect to either presentation.
- Left and right mixed canonical form with respect to either presentation.

3.13.4 Arithmetic Operations with Elements

- Product, left and right quotient, left and right conjugate of two elements.
- Inverse of an element.
- Cycling and decycling operation for an element with respect to either presentation.

3.13.5 Boolean Predicates

- Functions determining whether an element is the identity, simple, or a representative of its super summit class (with respect to either presentation), respectively.
- Tests for equality and conjugacy of elements.
- Partial orderings of elements with respect to either presentation.

3.13.6 Lattice Operations

- GCD and LCM with respect to either presentation and either partial ordering.

3.13.7 Conjugates

- Infimum, supremum and canonical length of the super summit class of an element with respect to either presentation.
- Computing a representative of the super summit set of an element with respect to either presentation.
- Computing the set of positive conjugates of an element with respect to either presentation.
- Computing the super summit set of an element with respect to either presentation.
- Process versions of the above algorithms for computing positive conjugates and super summit elements.

3.13.8 Homomorphisms

- Natural symmetric representation.
- Integral and modular Burau representations.
- Construction and evaluation of a homomorphism whose domain or codomain is a braid group.

3.14 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 $\mathtt{H} = \{x \in C | \mathrm{Im}(x) > 0\}\}$ 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 $\mathtt{H}$. A *fundamental domain* for the action of Γ is a region of $\mathtt{H}^*$ containing a representative of each orbit of the action. Magma contains a package written by Helena Verrill for working with $\mathtt{H}^*$ and with congruence subgroups and their action on $\mathtt{H}^*$. The subgroups of $\mathrm{PSL}_2(Z)$ 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 $\mathrm{PSL}_2(\mathbf{Z})$
- Construction of cusps, cusp widths, and elliptic points of congruence subgroups
- Farey symbols for congruence subgroups
- Action of elements of $\mathrm{PSL}_2(\mathbf{R})$ 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)

4 Semigroups and Monoids

4.1 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

4.2 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
- Definition of homomorphisms whose domain or codomain is an RWS monoid

5 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 **Q** and its ring of integers **Z**
- Residue class rings of **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

5.1 The Rational Field and its Ring of Integers

5.1.1 Arithmetic

- Multiple precision integer arithmetic
- Integer multiplication via classical, Karatsuba, Toom and Schönhage-Strassen FFT methods
- Integer division via classical, Karatsuba, Toom 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 uses portions of the GMP 4.2 package for the base classical, Karatsuba and Toom algorithms (for which GMP is the state of the art). For larger integers (typically about 30,000 to 50,000 decimals), Magma uses an optimized Schönhage-Strassen FFT method.

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 fast multiplication methods when applicable.

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 2.2GHz Opteron workstation, Magma can multiply 2 arbitrary integers, each having a million *decimal* digits, in 0.07 seconds, or compute their GCD in 0.1 seconds.

5.1.2 Residue Class Rings of 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

5.1.3 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 (ECPP) 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).

Paul Zimmermann's efficient GMP-ECM package is included for integer factorization. The GMP-ECM package also provides fast $p-1$ and $p+1$ integer factorisation algorithms.

5.1.4 The Number Field Sieve

Magma provides an experimental implementation of the fastest general purpose factoring algorithm known: the number field sieve (NFS). The implementation can be used for both general number field and special number field factorizations: the only difference is in the polynomial selection. Presently, Magma does not provide a function to choose a good polynomial for a particular number to be factored. However, Magma does provide some functions that are useful for the implementation of the polynomial selection algorithms developed by Peter Montgomery and Brian Murphy.

5.2 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.

5.2.1 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

5.2.2 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

5.2.3 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.

5.2.4 GCD and Factorization

- Resultant, discriminant (sub-resultant algorithm, Euclidean algorithm)
- Greatest common divisor, extended greatest common divisor
- Extended greatest common divisor
- Hensel lift
- Squarefree factorization
- Distinct degree factorization
- Factorization over 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. For polynomials over an algebraic function field, an evaluation/interpolation algorithm of Allan Steel 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.

5.2.5 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

5.3 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)

5.4 Finite Fields

5.4.1 Construction

- Construction of fields GF(p), p large; GF(p^n), p small and n large
- Optimized representations of GF(p^n) in the case of small p and large n
- Database of sparse irreducible polynomials over 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 GF(2^n), where the degree n may be a ten thousand or more, to fields 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 [5].

5.4.2 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

5.4.3 Roots and Polynomial Factorization

- Square root: Tonelli-Shanks method for 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 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.

5.4.4 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 GF(p)).
- Index calculus method using Emmanuel Thomé's implementation of Coppersmith's method (for fields GF(2^k)).

The index calculus method applied to an arbitrary field 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.

5.4.5 Derived Structures

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

5.5 Galois Rings

Magma provides facilities for computing with Galois rings. The features are currently very basic, but advanced features will be available in the near future, including support for the creation of subrings and appropriate embeddings, allowing lattices of compatible embeddings, just as for finite fields.

Because of the valuation defined on them, Galois rings are Euclidean rings, so they may be used in Magma in any place where general Euclidean rings are valid. This includes many matrix and module functions, and the computation of Gröbner bases. Linear codes over Galois rings will also be supported in the near future.

Features:

- Creation of a default Galois ring (using a default defining polynomial).
- Creation of a Galois ring by a specified defining polynomial.
- Basic structural operations and arithmetic.
- Euclidean operations.

5.6 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 in some situations, e.g., for class group computation, continued fractions for fundamental units.

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.

Elements of number fields and their orders may have several representations. These representations are implemented generally for elements of number fields and function fields. The standard representation is coefficients of the basis elements. When the basis is a power basis, a polynomial representation is used. All elements may have a product representation which can be a compact way of storing otherwise large elements.

5.6.1 Number Fields

- Arithmetic of elements
- Construction of equation orders, maximal orders, arbitrary orders
- Simple and relative extensions, extensions 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
- Computation of Hilbert class fields
- Representation of a number field as a vector space or algebra over a given coefficient field

5.6.2 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
- Reside class ring of an order modulo an arbitrary ideal
- Completion of absolute maximal orders at finite primes

5.6.3 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.
- p-Selmer groups

5.6.4 Diophantine Equations

- Norm equations, relative norm equations (both in the field and the order case, testing for local solubility)
- Thue equations
- Unit equations
- Index form equations
- Integral points on Mordell curves

5.6.5 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 (over $\mathbf{Q}$ or an absolute number field)
- 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 0.17 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.

5.6.6 Class Field Theory

- Computation of defining equations of class fields
- The norm group of an abelian number field can be computed
- Norm symbols, Artin-map is available
- Solvability of norm equations can be tested, extending the Hasse-Norm-Theorem
- Extension of automorphisms of the base field
- Second cohomology of the Ray class group

5.6.7 Quadratic Fields

- All functionality of number fields
- Division of integral elements of $\mathbf{Q}(\sqrt{d})$, for $d = -1, -2, -3, -7, -11, 2, 3, 5, 13$.
- 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)
- Fundamental unit, conductor
- Solution of norm equations (Cornaccia's algorithm for imaginary fields and Cremona's conics for real quadratic fields)
- Facilities for binary quadratic forms (see Lattices and Quadratic Forms)

5.6.8 Cyclotomic Fields

- All functionality of number fields
- Sparse representation for large fields
- Conductor and cyclotomic order
- Cyclotomic subfields
- Creation of roots of unity
- Minimization of elements into smaller fields
- Conjugation and complex conjugation

5.7 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 field extension of finite degree of $k(x_1, \ldots, x_n)$ for elements $x_i \in F$ which are algebraically independent over k.

5.7.1 Rational Function Fields

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

- Creation of a rational function field of a given rank over a given ring
- Retrieval of the ring of integers, coefficient ring and rank
- Ring predicates
- Arithmetic
- Numerator and Denominator
- Degree and weighted degree
- Evaluation
- Derivative
- Partial fraction expansion
- Partial fraction decomposition (squarefree or full factorization)

5.7.2 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.* An algebraic function field can be extended to create fields of the form $k(x, a_1, \ldots, a_r)$ where each extension occurs by adjoining a root of an irreducible and separable polynomial. Extensions may be formed using several polynomials simultaneously giving a non simple representation.

- Creation of simple, relative and non simple extensions and mixed towers thereof
- Creation of extensions of the constant field using bivariate polynomials
- Retrieval of information defining the field
- Exact constant field and genus
- Change of representation from finite degree extensions to infinite and vice versa
- Change of coefficient field to one lower in the extension tower
- Computation of basis
- Computation of subfields and automorphisms
- Homomorphisms from function fields into any ring by specifying the image of the primitive element and an optional map on the coefficient field
- Computation of Galois Groups of degree less that 24
- L-polynomial and ζ-function
- Construction of a function field with an extended constant field
- Construction of Artin-Schreier-Witt extensions from finite dimensional Witt-vectors

5.7.3 Orders of Algebraic Function Fields

- Finite and infinite equation orders
- Finite and infinite maximal orders using the Round 2 algorithm
- Creation of orders whose basis is a transformation of an existing order
- Integral closure
- Retrieval of information defining the order
- Basis of the order with the option to have the elements returned in a specified ring
- Discriminant and also with respect to the bottom coefficient ring of the tower
- Simplification of an order to a transformation of its equation order
- Unit Group and unit rank, independent and fundamental units and regulator
- Ring predicates
- Basis size reduction for finite, simple, non relative orders

5.7.4 Elements of Algebraic Function Fields and their Orders

Elements of function fields and their orders have 3 different representations. These representations are implemented generally for function field elements and number field elements. Standard elements are represented using coefficients of the basis elements. Elements of orders (and fields) with a power basis are represented using a polynomial representation. Elements of all orders or fields may have a product representation, being thought of as a formal product of a list of elements each to the power of some exponent. This can be a great advantage when the element is prohibitively large when represented using coefficients.

- Arithmetic and modular arithmetic
- Predicates
- Creation of random elements and conversion to and from sequences
- Norm and trace with respect to any given coefficient ring
- Representation matrix, minimal and characteristic polynomials
- Numerators and Denominators with respect to a given order
- Module generated by a sequence of elements
- Strong approximation theorem

5.7.5 Ideals of Orders of Algebraic Function Fields

- Creation of ideals from generators or a basis
- Arithmetic
- Roots of ideals
- Predicates for integrality, prime, principal zero and one ideals
- Predicate for prime ideals determining the type of ramification
- Intersection, GCD and LCM
- Factorization
- p-radicals and p-maximal orders
- Taking valuations of elements and ideals at prime ideals
- Denominator
- Retrieving basis and generators
- Residue class field and the map to and from the order into it
- Ramification and inertia degree

5.7.6 Places of Algebraic Function Fields

- Creation of places as zeros and poles of elements of a field
- Creation from prime ideals
- Creation of random places of global fields
- Creation of places of a given degree of global fields
- Decomposition of places
- Arithmetic
- Residue class field, lifting elements out of and evaluating functions into
- Valuation of elements and expanding elements at a place
- Completion of fields and orders at places of degree 1 (non-global fields) or places of any degree (global fields).
- Ramification and inertia degree
- Retrieval of generators and a uniformizing element
- Weierstrass places
- Counting the number of places of a given degree over the exact constant field of global fields
- The Serre and Ihara bounds on the number of places of degree 1 over the exact constant field of global fields

5.7.7 Divisors of Algebraic Function Fields

- Creation from places, elements and ideals
- Canonical and different divisor
- Arithmetic including GCD and LCM
- Support and Degree
- Numerator and Denominator
- Testing for properties of effective, positive, principal, special and canonical
- Riemann–Roch space $\mathcal{L}(D)$ of a divisor D, given by a k-basis of algebraic functions
- Reduction of a divisor
- Index of Speciality
- Gap numbers, ramification divisors, Wronskian orders and Weierstrass places
- Parametrization of a field at a divisor
- Number of smooth divisors of global fields

5.7.8 Differentials of Algebraic Function Fields

- Creation of a differential space
- Creation of differentials from field elements
- Arithmetic
- Valuation of a differential at a place
- Divisor of a differential
- Differential spaces and bases for given divisors
- Space and basis of holomorphic differentials of a field
- Differentiations of elements of a function field
- Residue of a differential at a place of degree one
- Cartier operator and representation matrix of the Cartier operator (global case)
- Module generated by a sequence of differentials

5.7.9 Divisor Class Groups of Global Algebraic Function Fields

- Bounds on the generation of the class group
- Computation of the class number and approximations to it
- Construction of the divisor class group, 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
$$0 \to U(S) \to F^{\times} \to Div(S) \to Cl(S) \to 0$$
- Image and preimage computation possible for the maps of the exact sequence
- Similar functionality for the ideal class group of the finite maximal order
- p-rank of the divisor class group (separate method) and Hasse–Witt invariant
- Tate–Lichtenbaum pairing
- Global units

5.7.10 Class Field Theory for Algebraic Function Fields

- Ray divisor class groups
- Defining equation for class fields
- Conductor and norm group
- Genus, discriminant, number of places of given degree
- Decomposition type of places of the base field
- Exact constant field

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

5.8 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

$$O_{p(x)} = \{\frac{f(x)}{g(x)} : f(x), g(x) \in K[x], p(x) \not| \, g(x)\}.$$

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

5.9 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.

These are currently both being replaced by an MPFR-based model, though internal computations still often use PARI.

5.10 Newton Polygons

- Construction of a newton polygon: Compact, infinite or including the origin
- Construction of newton polygons from different types of data: $f \in k[x, y]$, $f \in k \langle\langle x \rangle\rangle [y]$, $f \in k[y]$ and some prime object, a finite set of points, a finite set of faces (weighted dual vectors)
- Finding faces, vertices and slopes
- If polygon is derived from a polynomial f, finding restrictions of f to faces
- Locating a given point relative to a newton polygon
- Giving the valuations of the roots of a polynomial (with respect to a prime if not implicit)

Newton polygons can be used with polynomials over series rings in order to find roots of the polynomial.

- Walker's [42] algorithm for computing Puiseux expansions
- Duval's [15] algorithm for computing Puiseux expansions

5.11 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. Magma supports both fixed and free precision models, allowing the user to trade an increase in speed for automated precision management.

5.11.1 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. Any arbitrary tower of extensions can be constructed, as long as each step is either ramified or unramified or both.

5.11.2 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

5.11.3 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

5.12 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
- Factorization of polynomials over series rings
- Unramified and ramified extensions of series rings

5.13 Lazy Power Series Rings

These power series rings contain only series of infinite precision. All coefficients of such series are computable but only finitely many will be known.

- Creation of rings and elements
- Arithmetic of elements
- Retrieval of coefficients
- Printing some specified terms of a series
- Simple predicates on series
- Derivative, integral and evaluation of series

5.14 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 [14]), 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 system developed for Magma by Allan Steel avoids these problems, and is described in [37]. Consequently, ACF's behave in the same way as 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 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.

6 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.

6.1 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.

6.1.1 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 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.

6.1.2 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

6.1.3 Polynomial Rings: GCD and Factorization

- Greatest common divisor (sparse EEZ-GCD, fast GCD-HEU, evaluation-interpolation algorithms)
- Newton polygon
- Squarefree factorization
- Bivariate factorization over GF(q), $\mathbf{Z}$ and $\mathbf{Q}$ (polynomial-time trace-based algorithm of Belabas et al., interpolation algorithms)
- Multivariate factorization over GF(q), $\mathbf{Z}$ and $\mathbf{Q}$ (reducing to bivariate factorization to solve combination problem)
- Factorization over arbitrary algebraic function fields (including inseparable field extensions)
- Resultant (modular and sub-resultant algorithms), discriminant

Since V2.11, factorization of bivariate polynomials over all supported rings is accomplished by a new algorithm which extends van Hoeij's knapsack ideas for $Z[x]$ to solve the hard combination problem for $\mathrm{GF}(q)[x, y]$. The new algorithm runs in polynomial time and performs extremely well in practice. General multivariate factorization is reduced to this new bivariate algorithm, so a combination problem never arises for any number of variables. Shoup's tree Hensel lifting algorithm has also been adapted for power series, making the lifting stages of all kinds of bivariate/multivariate factorization much faster than previously.

Factorization over general algebraic function fields of small characteristic is accomplished by a new algorithm of Allan Steel [38]. This can handle extensions which are inseparable, and may have an arbitrary number of both algebraic and transcendental generators.

Resultants are computed using asymptotically-fast modular and evaluation/interpolation algorithms. Options are provided for Monte Carlo-style stopping on stability, which greatly speeds up the computation (since the bounds are usually very much worse than the required lifting level).

6.1.4 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 finite fields and rationals, using optimized Faugère F_4 algorithm
- Gröbner bases of ideals over general Euclidean rings, using an extension by 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

Since V2.11, an optimized implementation of Faugère's F_4 algorithm (which uses sparse linear algebra) to compute Gröbner bases is available for ideals over finite fields and the rationals. See [40] for detailed timings.

On a Athlon 2800+ XP machine, Magma computes the `lex` order Gröbner basis for the Katsura-5 problem in 0.3 seconds and the `lex` order Gröbner basis for the cyclic 6-th roots problem in 0.07 seconds. Taking a more difficult example, Magma computes the `grevlex` order Gröbner basis for the Cyclic-7 roots ideal in 2.2 seconds and transforms this to the `lex` order Gröbner basis in a further 23 seconds (using the p-adic FGLM algorithm). These examples are from the POSSO polynomial systems library (`ftp posso.dm.unipi.it`).

6.1.5 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
- Noether normalisation of ideals
- Normalisation of the affine quotient algebra of an ideal

6.1.6 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
- Equidimensional decomposition 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

Primary decomposition of ideals over general algebraic function fields of small characteristic is handled by a new algorithm of Allan Steel [38].

6.1.7 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)

6.2 Affine Algebras

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

6.2.1 Affine Algebras: Creation and Operations

- Creation of an affine algebra
- Arithmetic with elements

6.2.2 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 $A = K[x_1, \ldots, x_n]/J$ 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.

6.3 Modules over Affine Algebras

Modules over a multivariate polynomial ring $R[x_1, \ldots, x_n]$ (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.

6.3.1 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

6.3.2 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

6.3.3 Modules over Affine Algebras: Homology

- Syzygy modules
- Construction of the image and kernel of a homomorphism
- $Hom(M, N)$, where M and N are modules
- Free resolutions, minimal free resolutions
- Betti numbers, homological dimension
- Homology of a complex

7 Linear Algebra and Module Theory

- Matrix operations
- Vector spaces
- Free modules
- Modules over Dedekind domains

7.1 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.

7.1.1 Representation of Matrices

- Optimal packed representation for matrices over 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 **Q**, reducing computations to those over **Z**.
- Internal modular representations of matrices over **Z** or **Q**, for several algorithms.
- Input of matrices using the “Cambridge” compact format

7.1.2 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
- Determinant

There are many algorithms to compute determinants, based on the different coefficient rings, including modular and heuristic methods. Since V2.11, an efficient elimination and minor expansion algorithm is used to compute determinants over multivariate polynomial rings with many variables.

7.1.3 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 **Z** and **Q**)
- Solution of systems of linear equations

Given a 301 by 300 matrix over **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[8] 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.

7.1.4 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 **Z** and **Q**)
- LLL-reduction of matrices over **Z**.

Over finite fields, a fast algorithm due to Allan Steel is used to construct the various matrix canonical forms: generalized Jordan, rational, and primary rational [36]. Since V2.11, a new asymptotically-fast algorithm of Allan Steel is used to construct the canonical forms over fields of characteristic zero.

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 **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.

7.2 Sparse Matrices

A special type for sparse matrices is provided so that the user can build up such matrices and then apply some non-trivial algorithms to them. An extended example in the Handbook implements the basic linear sieve for discrete logarithms in the Magma language, thus demonstrating how one can use the sparse matrix facilities when implementing index-calculus methods.

Features:

- Creation of sparse matrices in compact form.
- Creation of trivial sparse matrices followed by dynamic expansion.
- Basic properties (density, etc.).
- Conversion between sparse and normal (dense-representation) matrices.
- Multiplication of dense vectors by sparse matrices.
- Non-trivial invariants of sparse matrices: nullspace, rank and elementary divisors (equivalent to Smith form).
- Computation of non-zero solution vector for sparse systems arising in index-calculus algorithms (Structured Gaussian elimination and Lanczos algorithms [25]).
- Computation of general nullspace over fields and Euclidean rings using Markowitz-pivoting techniques.

7.3 Vector Spaces

7.3.1 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

7.3.2 Construction

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

7.3.3 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

7.3.4 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

7.3.5 Homomorphisms

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

7.3.6 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

7.4 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.

7.4.1 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

7.4.2 Homomorpisms

- Construction of modules $Hom(M, N)$ for any free modules M and N
- Explicit construction of $\mathrm{Hom}(U, V)$ for proper subspaces U and V
- Explicit construction of $\mathrm{Hom}(H_1, H_2)$ 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)

7.5 Modules over Dedekind domains

Modules over dedekind domains are supported only for maximal orders of Algebraic number fields and Algebraic function fields.

- Creation of modules
- Arithmetic with module elements
- Submodules and quotients of modules
- Determinant, 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
- Construction of modules $Hom(M, N)$ and standard calculations with morphisms

8 Lattices and Quadratic Forms

8.1 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. The rigorous LLL of Nguyen and Stehlé will be implemented in a future release.

8.1.1 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.

8.1.2 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 **Z** or **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 $4649359/4213820620800 \approx 0.000001103359$.

8.1.3 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). Indefinite Gram matrices can also be reduced in some cases.

8.1.4 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.

8.1.5 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

8.1.6 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.

8.2 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 $\mathrm{SL}(2, \mathbf{Z})$ and $\mathrm{PSL}(2, \mathbf{Z})$
- Evaluation of modular functions on quadratic forms
- Discrete logarithm of a form (for imaginary quadratic fields)

9 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)
- Quantized enveloping algebras (aka quantum groups)

9.1 Finitely Presented Associative Algebras

Finitely-presented (FP) associative algebras (or noncommutative polynomial rings) are defined by taking R-linear combinations of elements of a semigroup, where R is some ring. Since V2.11, these are handled by an extension of the commutative algebra machinery to noncommutative data structures and algorithms, where applicable. These include a noncommutative analogue for Gröbner bases.

– Construction of free algebras over arbitrary fields

– Arithmetic

– Mappings into other associative algebras

– Definition of left, right, two-sided ideals

– Noncommutative Gröbner bases of ideals, with specialized algorithms for different coefficient fields (fraction-free methods for the rational field and rational function fields)

– Gröbner bases of ideals over finite fields and rationals, using noncommutative extension of the Faugère F_4 algorithm

– Construction of degree-d (truncated) Gröbner bases

– Normal form of a polynomial with respect to an ideal

– Construction of FP-algebras as quotient rings

– Enumeration of the basis of finite-dimensional FP algebras

– Matrix and structure-constant representations of finite-dimensional FP algebras

– Construction of a matrix representation (Linton's vector enumerator)

There are two major tools for computing with these algebras. The main approach is to apply a noncommutative 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. An extension of Faugère's F_4 algorithm, due to Allan Steel, works by sparse linear algebra and is often much quicker.

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.

9.2 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

9.3 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.

9.3.1 Orders of Associative Algebras

- Construction of orders of algebras over the rationals or a number field
- Construction of a maximal order of a central simple algebra defined over the rational numbers or a number field
- Basis of an order
- Construction of elements of an order of an algebra

- Arithmetic of elements of an order of an algebra
- Norm, trace, conjugate, minimal polynomial and representation matrix of elements
- Construction of left, right and two-sided ideals of orders
- Addition and multiplication of ideals
- Left and right order of an ideal, colon ideal
- Basis and basis matrix of an ideal

9.4 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 **Z**, $k[x]$ and orders of number fields is provided for quaternions over the rational field **Q**, $k(x)$ or a number field. Special functions for enumeration all ideals in definite quaternion algebras over **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 **Z**
- Left and right orders of an ideal in a definite order over **Z**

9.5 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

9.6 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.

9.7 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.

9.7.1 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 Lie algebra given by generators and relations
- Construction of a Lie algebra from a p-group, by using its Jennings series.
- Construction of a specified simple Lie algebra
- Direct sum
- Arithmetic
- Trace and minimal polynomial

9.7.2 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

9.7.3 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

9.7.4 Lie Algebras: Structure

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

9.7.5 Lie Algebras: Representations

- Construction of a faithful module over a Lie algebra of characteristic zero
- Construction of highest-weight modules over split semisimple Lie algebras
- Construction of tensor products, symmetric powers, antisymmetric powers of Lie algebra modules

9.7.6 Lie Algebras: universal enveloping algebras

- Construction of a universal enveloping algebra of a Lie algebra
- A special construction of the universal enveloping algebra of a split semisimple Lie algebra, via a Kostant basis

9.8 Quantized Enveloping Algebras

A quantized enveloping algebra (corresponding to a given root datum) is represented with respect to an integral basis, as defined by Lusztig.

- Constructing of quantized enveloping algebras with respect to a given root datum
- Arithmetic: sum and product
- Representations: construction of highest-weight modules, and tensor products of them
- Construction of the canonical basis of a highest-weight module
- Construction of elements of the canonical basis of the negative part of a quantized enveloping algebra
- Action of the Kashiwara operators
- Littelmann's path model: action of the path operators, construction of the crystal graph

10 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

10.1 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 $\mathrm{Hom}(U, V)$. For the important case of modules over finite fields, different representations of vector arithmetic, depending upon the field, have been implemented.

10.1.1 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.

10.1.2 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.

10.1.3 A-Modules: Submodules and Quotient Modules

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

10.1.4 *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 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.

10.1.5 *A*-Modules: Homomorphisms

- Construction of $\mathrm{Hom}(U, V)$, 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 $\mathrm{Hom}(U, V)$ which is applicable to modules having dimension several hundred.

10.2 Representations of Symmetric Groups

Special functionality for representations of a symmetric group concentrates on characters as indexed by partitions of weight the degree of the group.

- Integral, seminormal and orthogonal representations of a permutation.
- Values of a character of a symmetric group indexed by a partition on a permutation.
- Characters of symmetric groups corresponding to partitions.
- Values of a character of an alternating group indexed by a partition on a permutation.
- Characters of symmetric groups corresponding to partitions.

10.3 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, Unger's algorithm)

10.4 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 [22]. 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.

10.4.1 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.

10.4.2 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

10.4.3 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

10.4.4 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

11 Homological Algebra

11.1 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

11.2 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

12 Lie Theory

The current elements of the machinery for Lie theory comprise:

- Coxeter matrices, Coxeter graphs, Cartan matrices, Dynkin digraphs, and Cartan names.
- Root systems
- Root data
- Coxeter groups
- Coxeter groups as permutation groups
- Reflection groups
- Groups of Lie type
- Finite-dimensional Lie algebras (See section on Algebras)

12.1 Coxeter systems

We support all the standard descriptions for Coxeter systems and reflection groups: Coxeter matrices, Coxeter graphs, Cartan matrices, Dynkin digraphs, and Cartan names.

- Creation of these descriptions.
- Conversion between them.
- Isomorphism and Cartan equivalence of Coxeter systems.
- Testing for finite, affine, hyperbolic, and compact hyperbolic Coxeter systems.
- Constructing all finite, affine and hyperbolic systems.
- Computing the size and number of roots of (finite) Coxeter systems.
- Printing Dynkin diagrams for finite systems.
- Predicates: IsIrreducible, IsCrystallographic, IsSimplyLaced.

12.2 Root Systems

A datatype for finite root systems has been implemented by Scott H. Murray and Don Taylor.

12.2.1 Creating Root systems

Any finite root system can be constructed by giving its simple roots and coroots, including non-semisimple systems (where the dimension of the vector space is larger than the rank). Special function exist to construct semisimple systems from a Coxeter matrix, Coxeter graph, Cartan matrix, Dynkin digraph, or Cartan name. Standard root systems can be constructed—these are systems whose pairing is the Coxeter form, which is how root systems are frequently given in the literature.

12.2.2 Operations and properties for Root Data

- equality, isomorphism and Cartan equivalence.
- Operations: Cartan name, Coxeter diagram, Dynkin diagram, Coxeter matrix, Coxeter graph, Cartan matrix, Dynkin digraph, base field, rank, dimension, Coxeter group order.
- Properties: IsIrreducible, IsSemisimple, IsSimplyLaced.

12.2.3 Roots and coroots

The (co)roots are stored in an indexed set, with positive roots first. The can be described and manipulated via their index, or as vectors with respect to either the standard basis or the basis of simple (co)roots.

- Root space and coroot space.
- construction of the complete set of roots or coroots.
- conversion between indices and vectors.
- Highest long or short root.
- Reflection actions of the (co)roots: given as matrices, permutations, or words in the simple reflections.
- Basic arithmetic with (co)root indices: Sum, Negative, IsPositive, IsNegative, heights, norms.
- Coxeter form and dual Coxeter form.

12.2.4 New root systems from old

Direct sums and duals of root systems can be computed.

12.3 Root data

A datatype for finite root data has been implemented by Scott H. Murray and Don Taylor. This is designed primarily for use with Lie algebras and groups of Lie type.

12.3.1 Creating Root data

Any split (untwisted) root datum can be constructed by giving its simple roots and coroots. Special function exist to construct semisimple systems from a Cartan matrix, Dynkin digraph, or Cartan name. By default the adjoint datum is returned, by the isogeny type can be specified by optional parameters. Standard classical root data can be constructed—these are systems whose pairing is the Coxeter form, which is how root data are frequently given in the literature. For technical reasons it is impossible to define standard root data for the exceptional types.

12.3.2 Operations and properties for Root Data

- equality, isomorphism, Cartan equivalence, and isogeny.
- Operations: Cartan name, Coxeter diagram, Dynkin diagram, Coxeter matrix, Coxeter graph, Cartan matrix, Dynkin digraph, base field, rank, dimension, Coxeter group order, group of Lie type order.
- Fundamental and (co)isogeny groups.
- Properties: IsIrreducible, IsSemisimple, IsCrystallographic, IsSimplyLaced, IsAdjoint, IsSimplyConnected.

12.3.3 Roots, coroots and weights

The (co)roots are stored in an indexed set, with positive roots first. The can be described and manipulated via their index, or as vectors with respect to the standard basis or the basis of simple (co)roots or the basis of fundamental weights.

- root space and coroot space.
- construction of the complete set of roots or coroots.
- conversion between indices and vectors.
- Highest long or short root.

- Reflection actions of the (co)roots: given as matrices, permutations, or words in the simple reflections.
- Basic arithmetic with (co)root indices: Sum, Negative, IsPositive, IsNegative, heights, norms.
- Left and right strings through one root in the direction of another.
- Coxeter form and dual Coxeter form.
- (co)weight lattice and fundamental (co)weights

12.3.4 New root systems from old

Direct sums, duals and subdata of root data can be computed.

12.3.5 Constants

The standard constants used to define Lie algebras and groups of Lie type can be computed: p, q, N, ϵ, M, C, and eta.

12.4 Coxeter Groups

Coxeter groups are implemented as a subclass of finitely presented groups so that they inherit all the operations for finitely presented groups as well as having many specialized functions. The main difference is that every word is automatically converted into normal form using an algorithm designed and implemented by Bob Howlett. This module was implemented by Bob Howlett, Scott Murray, and Don Taylor.

A Coxeter group can be constructed from a Cartan matrix, Dynkin digraph, Cartan name, root system, or root datum. Note that a Coxeter group does not have a unique reflection representation, and so does not have a unique Cartan matrix.

- isomorphism as Coxeter groups.
- Operations: Cartan name, Coxeter diagram, Coxeter matrix, Coxeter graph, rank.
- Properties: IsFinite, IsAffine, IsHyperbolic, IsCompactHyperbolic, IsIrreducible, IsSimplyLaced.
- Arithmetic of words: identity, multiplications, inversion, powers.
- Degrees of the basic invariant polynomials.
- Coxeter element and Coxeter number.
- Braid group and pure braid group
- Conversion to and from permutation and reflection representations.

12.5 Coxeter Groups as Permutation 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.

A permutation Coxeter group can be constructed from a Cartan matrix, Dynkin digraph, Cartan name, root system, or root datum.

In addition to the standard functions for groups, almost all of the functions for root systems and root data also work on permutation Coxeter groups.

A reflection subgroup can be represented two ways: as a permutation group on the roots of the larger groups, or as a permutation group on its own roots. We use an efficient algorithm due to Don Taylor to compute transversals of reflection subgroups.

We can also compute the "standard" permutation action of a Coxeter group, which is usually the smallest degree permutation action. For example, the standard action of the group of type A_n, gives the symmetric group on $n + 1$ points.

12.6 Complex Reflection Groups

- construction and identification of a reflection group over an arbitrary ring, given the simple roots, coroots and orders
- construction of real reflection groups from a Cartan matrix, Dynkin digraph, Cartan name, root system, or root datum
- construction of all finite complex reflection groups
- Most of the functions available for Coxeter groups are also available for real reflection groups.

12.7 Groups of Lie Type

We have implemented code for computing in split (untwisted) groups of Lie type with the Steinberg presentation. These groups can be defined over any Magma field. Elements can be normalised using the Bruhat decomposition: a flag for each group determines whether this is done automatically or manually by the user.

12.7.1 Creating Groups of Lie type

A group of Lie type can be created from a field and a Cartan name, Weyl group, root datum, Cartan matrix or Dynkin digraph.

12.7.2 Operations and Properties

- Most of the operations and properties for root data also apply to groups of Lie type.
- equality, algebraic isomorphism, isogeny.
- algebraic group generators; abstract group generators for certain fields
- element arithmetic and normalisation
- Bruhat decomposition and multiplicative Jordan decomposition.

12.7.3 Automorphisms

The inner, diagram, diagonal and field automorphisms can be constructed. These include all algebraic group automorphisms, and in many cases all abstract group automorphisms.

12.7.4 Representation Theory

- Standard, regular and highest weight representations can be constructed.
- The inverse of any representation over the base field can be computed using a generalised row reduction function.

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 curves
- Divisor groups and places of curves

- Resolution graphs and splice diagrams
- Graded rings and geometric databases
- Plane conic curves and general rational curves
- Elliptic curves
- Hyperelliptic curves
- Module of supersingular points
- Modular forms
- Modular symbols
- Brandt modules
- Modular curves
- Modular Abelian Varieties

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. A 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
- Saturation of the defining ideal of a scheme in the projective cases
- 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
- Determining whether a scheme over a number field is locally solvable

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, have additional map functionality
- Availability of maps with multiple definitions

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: Isomorphic Projections

- Computation of the tangent and secant varieties of schemes
- Isomorphic projection of projective schemes to smaller dimensional ambient spaces
- Birational embedding of plane curves as non-singular projective space curves in 3-dimensional space

13.1.6 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 algebraic curves. These are schemes of dimension 1, a particular case being plane curves that are defined by the vanishing of a single polynomial in 2-dimensional affine or projective space. The main features of interest are for integral (reduced and irreducible) curves. For these Magma computes explicit representations of the curve's field of rational functions allowing computations with places, divisors and differentials, calculation of the geometric genus, computation of Riemann-Roch spaces and more. This relies on the underlying function field machinery, but the results are available purely in the context of curve structures.

13.2.1 Curves: Construction and General Properties

- Creation of affine and projective curves together with the ambient spaces
- 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
- Point searches on plane curves using a p-adic method of Elkies
- Global invariants of curves, such as genus and dimension
- Function field and divisor computations (see below)

13.2.2 Curves: Mappings

- Creation of the basic automorphisms of the affine plane (translation, flip, automorphism) and general maps between curves.
- Evaluation of the image of a point under a rational map as a power series expansion
- Ramification divisor of a non-constant dominant map
- Pullbacks and pushforwards along maps between curves.

13.2.3 Curves: Local Analysis

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

13.2.4 Curves: Function Field

The following functions apply to integral curves in general.

- Construction of the function field of a curve
- Function field arithmetic
- Exact constant field
- Enumeration of places of degree m (over a finite field)
- Residue class field
- Class number
- Divisor class group of a curve defined over a finite field
- Group of global units of the function field of a curve

13.2.5 Curves: Divisors and the Riemann-Roch Theorem

The following functions apply to integral curves in general.

- Creation of places and 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
- Ramification divisor
- Riemann–Roch space of a divisor
- Computation of Weierstrass places on a curve

13.2.6 Curves: Differentials

The following functions apply to integral curves in general.

- 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

Rational curves and conics in Magma are nonsingular plane curves of degree 1 and 2, respectively. The central functionality for conics concerns the existence of points over the rationals. If a point is known to exist, then a conic can be parametrized by a projective line or a rational curve. A rational curve in Magma is a linearly embedded image of the projective line, to which the full machinery of algebraic plane curves may be applied. Tools provided include the local-global theory, the existence of points on conics, efficient algorithms for finding rational points, parametrizations and isomorphisms of genus zero curves.

- Diagonalized Legendre model for a conic defined over the rationals
- Simon's algorithm for finding a point on a rational conic; this involves LLL-reduction of an indefinite form, and includes the Cremona-Rusin reduction as a special case
- Variant of Simon's algorithm for parametrisation of a rational conic.
- Reduction of a rational point on a conic so that its coordinates satisfy Holzer's bounds; the algorithm used is due to Mordell
- Isomorphisms with standard models
- 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
- Isomorphism of rational curves and conics
- Automorphism groups of rational curves and conics

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 an elliptic curve with given j-invariant
- Creation of an elliptic curve from a general genus 1 curve
- Models: Weierstrass form, integral model, minimal model, simplified model
- Arithmetic with rational points, including division
- Extension and lifting of curves induced by maps of base rings
- Function fields, divisors, places
- Invariants: b-invariants, c-invariants, j-invariant, discriminant
- Division polynomials
- Subgroups and subschemes of elliptic curves as separate types

13.4.2 Elliptic Curves: Morphisms

- Extension and lifting of maps induced by change of base ring of curves
- Isomorphisms, isogenies and rational maps between curves, translation maps on a curve
- Isogeny operations: 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
- Endomorphisms
- Isomorphism operations: inverses, composition
- Testing for isomorphic curves over fields with root-finding
- Testing for isogenous curves over finite fields via point-counting, and over the rationals via kernels (a map is returned also here)

13.4.3 Elliptic Curves: Operations over 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, minimal twist
- Heights: local height, naive height, canonical height, height pairing
- Heights: Silverman bound, Siksek bound
- Elliptic logarithm, p-adic elliptic logarithm
- Elliptic exponential ($\wp$-function) using Newton iteration
- Mordell-Weil rank, bounds on Mordell-Weil rank
- Mordell-Weil group and torsion subgroup
- Analytic rank, special values of the L-series, root numbers
- Isogeny class (and isogenies), modular degree
- Heegner points for rank 1 curves, and general modular parametrisation

- Algebraic two-descent to complement Cremona's invariant method
- Four-descent, with point-searching via the Elkies ANTS-IV p-adic method
- S-integral points: determination of the finite set of affine points with denominator generated over the finite set of primes S.
- S-integral points: quartic, Ljunggren and Desboves points
- John Cremona's database of all elliptic curves over **Q** having conductor up to 40 000

Standard models are provided together with heights and invariants. For curves over **Q** invariants such as conductor, regulator, and local information (Tate's algorithm) are available. The Mordell-Weil rank and group are computed using code following algorithms developed by John Cremona and implemented in his `MWRANK` program. Magma takes 10.06 seconds to determine the full group of the rank 4 curve

$$y^2 = x^3 - 73705x - 7526231$$

and 4.24 seconds to determine that the curve

$$y^2 = x^3 + 2\,429\,469\,980\,725\,060x^2 + 275\,130\,703\,388\,172\,136\,833\,647\,756\,388x$$

has rank 14.

13.4.4 Elliptic Curves: Operations over Algebraic Number Fields

- Enumeration of rational points
- Reduction at a prime, local and global minimal models
- Torsion bound, p-torsion, torsion subgroup
- Bounds on the Mordell-Weil rank
- Mordell-Weil group for suitably "small" curves
- 2-Selmer groups
- 2-isogeny Selmer groups
- Representation of elements of the Selmer group as elements of an etale algebra
- Chabauty methods
- Local solubility

13.4.5 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 algorithms for such curves are the SEA and canonical lift algorithms 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 for small p
- 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 and p-adic canonical lift methods 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. In characteristic two, Magma takes around 0.04 seconds for a curve over $GF(2^162)$, where Gaussian Normal Bases are available and 0.27 seconds over $GF(2^160)$. Over $GF(2^268)$ (again with GNB) it takes around 0.17 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 contributions 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
- Function fields, divisors, places
- Invariants: degree, genus, discriminant
- Igusa and related invariants
- Construction of a curve given its Igusa invariants
- Extension and lifting of curves induced by maps of base rings
- Reduction types and conductors of a genus 2 curve at odd primes
- Determination of points at infinity
- Special optimized algorithm for determining local solubility
- Determination of whether a genaral algebraic curve C is hyperelliptic and, if so, computation of a Weierstrass model for C
- Determination of whether a general algebraic curve C is geometrically hyperelliptic - ie, is hyperelliptic over the algebraic closure of the ground field.

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

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

13.5.4 Hyperelliptic Curves: Operations over Number Fields

- Enumeration of all rational points
- Selmer groups

13.5.5 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
- Creation of points
- Normal form and addition of points
- Fast addition algorithm of R. Harley for genus 2 curves
- Kummer surface of the Jacobian of a genus 2 curve
- Analytic model of the Jacobian: translation between the algebraic and analytic Jacobians

13.5.6 Hyperelliptic Curves: Operations on the Jacobian over 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 **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 **Q** with Mordell–Weil rank at most 1

13.5.7 Hyperelliptic Curves: Operations on the Jacobian over F_q

The p-adic techniques used for point counting in small characteristic p are Kedlaya's algorithm (p odd) and Mestre's canonical lift method as adapted by Lercier and Lubicz ($p = 2$) [implemented by M. Harrison]. The other 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
- Counting points: p-adic methods of Kedlaya and Mestre/Lercier/Lubicz
- Counting points: Shanks and Pollard methods
- Counting points: An index calculus method for when the genus is large compared to the base field
- Counting points: Schoof algorithm for finding the order modulo small primes in the case of a genus 2 curve
- Counting points: Class group methods using the function field machinery.
- Structure of the abelian group of rational points
- Weil pairing of points

13.5.8 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 [28] and in general using quaternion ideal theory [16, 32], which has deep connections with the theory of Shimura curves [23].

13.6.1 Module of Supersingular Points

Machinery for computing with the Hecke module of divisors on the supersingular points on $X_0(N)$ in characteristic p is now included. This module is the free abelian group on the supersingular elliptic curves in characteristic p enhanced with level N structure. We compute this module using the Method of Graphs of Mestre and Oesterlé and the Brandt modules algorithm.

- Computation of Hecke operators and Atkin-Lehner involution.
- Decomposition into invariant subspaces.
- The monodromy pairing.

13.6.2 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
- Computation of Atkin-Lehner operators

13.6.3 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 the Stein's thesis [41] 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

13.6.4 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.

13.6.5 Dirichlet Characters

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

13.6.6 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 rom 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.

13.6.8 Modular Abelian Varieties

This package offers some nontrivial functionality for modular abelian varieties, which we view as explicitly given quotients or subvarieties of modular Jacobians. No explicit algebraic defining equations are used in these algorithms, so computations with abelian varieties of large dimension are feasible.

- Construction of quite general modular abelian varieties, in the sense that arbitrary finite direct sums and quotients may be formed.
- Explicit computation of the group $\mathrm{Hom}(A, B)$ or the ring $\mathrm{End}(A)$, as a subgroup of homology, for modular abelian varieties A, B over $\mathbf{Q}$.
- Computation of kernels, cokernels, and images of homomorphisms of abelian varieties.
- Intersections of subvarieties.
- Computation of discriminants of subgroups of endomorphism rings, such as Hecke algebras.
- A divisor and a multiple of the order of the K-rational torsion subgroup of A.
- The determination of whether or not two modular abelian varieties are isomorphic (in some cases).
- Characteristic polynomial of Frobenius.
- Tamagawa numbers and component group orders (in some cases).
- Computation of all inner and CM twists (not provably correct).
- Computation with torsion points as elements of rational homology.

13.7 Graded Rings and Geometric Databases

The computation of generic families of K3 surfaces embedded in weighted projective spaces of small dimension is a project that has been running since the mid-1970s. Until two years ago, a main result was a collection of about 400 families of K3 surfaces (which had been recorded as a database in Magma). Magma now contains a database of 24,099 K3 surfaces (including correcting half a dozen mistakes in codimension 4) that, in a well-defined sense, contains a sketch of all possible examples.

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. It is possible that the final classification of Fano 3-folds (expected to contain a few thousand families) will almost be contained in the K3 database.

The methods applied to K3 surfaces work in many other contexts. Currently Magma contains the calculations for: subcanonical curves, K3 surfaces, Fano 3-folds and Calabi–Yau 3-folds; other cases are in progress. The basic method is to assemble a lot of data that should occur on these varieties, feed it into the Riemann–Roch formula to get a Hilbert series, and then attempting to describe a plausible variety embedded in weighted projective space that has that Hilbert series. The ideas are described in the paper [1], while improved handling of high codimension possibilities using projection/unprojection theorems is described in [7]. Other cases are covered by the papers [10], [33].

Notice that these routines do not describe explicit graded rings: they do not explain how to write the equations, but only say which weighted variables should be used in the equations. (The Hilbert series does include more information, but still much less than would dictate explicit equations, in general.) The process of recovering equations through projections is called ‘unprojection’, and that is still some way off being implemented as a method in commutative algebra.

These families exhibit large Gorenstein rings with as yet unknown structure. The search for structure theorems for Gorenstein rings occupied much of the 1970s after the Buchsbaum–Eisenbud structure theorem in codimension 3. However no substantial new cases were discovered. These examples, not available in the 1970s, are a major motivation for a renewed assault on this problem.

- Functions that create subcanonical curves, K3 surfaces, Fano 3-folds and Calabi–Yau 3-folds from data associated to the Riemann–Roch formula in each case
- A K3 database containing 24,099 candidates for polarised K3 surfaces. There are also smaller databases of Fano 3-folds as a first step in the final classification
- Functions for interrogating databases, including functions that can add new entries or rewrite existing entries

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 Differential Galois Theory

The Galois theory of linear differential equations is the analogue of the classical Galois theory of polynomial equations for linear differential equations. The natural analogue of the field in the classical case is the differential field. This is a field equipped with a derivation. We have constructed a basic facility for differential fields and rings. These types can be built from the algebraic function field or affine algebra types. Our medium term goal is to construct a fast solver for linear differential equations.

14.1 Differential Rings and Fields

- Construction of the rational differential field and the more general differential ring
- Coercions, arithmetic and functionality for elements as for the underlying ring.
- Changing the derivation of a differential ring.
- Extending the constant ring of a differential ring
- Wronskian matrix and Wronskian determinant
- The differential constant field of a rational differential field
- Ring and field extensions of differential rings and fields
- Construction of a differential ideal
- Quotient rings, rings and field of fractions of differential rings and fields

14.2 Differential Operator Rings

- Creation of a differential operator ring
- Coercion, arithmetic and simple predicates for elements
- Accessing coefficients of elements
- Changing the derivation of a differential operator ring.
- Changing the operator ring by extending the constant ring
- Making a differential operator monic.
- Adjoint of an operator
- Applying an operator to an element of its basering
- Euclidean algorithms, left and right (extended) GCD, (extended) left LCM
- Companion matrix of an operator.
- Determination of whether a place is regular, regular singular or irregular singular at an operator
- All singular points of an operator
- The indicial polynomial of an operator at a place
- All rational solutions of an operator within a rational differential field
- Newton polygon and Newton polynomial
- Differential field extension of the base ring of an operator by adjoining a formal solution and formal derivatives
- The symmetric power of an operator

15 Finite Incidence Structures

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

- Counting functions
- Partitions and tableaux
- Graphs—directed and undirected
- Networks
- Incidence structures
- Designs
- Finite planes
- Incidence geometry

15.1 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)

15.2 Partitions and Young Tableaux

Extensive facilities were installed for working with Young tableaux and symmetric functions. The facility is heavily based on the Symmetrica package developed by A. Kerber and associates in Bayreuth. A highlight of the tableau machinery is the Robinson-Schensted-Knuth (RSK) correspondence. The collection of all symmetric functions defined over some ring is viewed as forming an algebra where the chief interest lies in the change of basis matrix relative to different bases. The Young tableaux code was written by G. White while the machinery for symmetric functions was adapted from Symmetrica by A. Kohnert (Bayreuth) during a year-long visit to Sydney.

- Number of partitions of n
- Enumeration of restricted and unrestricted partitions
- Conjugate partitions
- Longest increasing sequences within words of integers
- Lexicographical ordering of words
- Creation of skew or non-skew tableaux over the integers or arbitrary label sets
- Enumeration of tableaux
- Random tableaux
- Properties: shape, skew-shape, weight, hooklength's, content, row/column words
- Operations: diagonal sum, product, conjugate, jeu de taquin, row insertion, inverse jeu de taquin, inverse row insertion
- RSK correspondence, inverse RSK correspondence
- Enumeration of tableaux having specified size and alphabet

15.3 Symmetric Function Algebras

Symmetric functions are defined as polynomials in an infinite number of variables, invariant under the action of the symmetric group. The set of all symmetric functions denoted by Λ is an algebra. A symmetric function algebra may be created over an arbitrary commutative ring (with unity) R. Five different bases for an algebra of symmetric functions are supported.

- Creation of algebras with any one of 5 possible bases: consisting of Schur functions, Elementary, Monomial, Homogeneous (complete) or Power Sum functions.
- Creation of elements as linear combinations of basis elements (indexed by partitions) or from coercing a polynomial or a scalar.
- An algebra can be tested for which basis it represents its elements with respect to.
- Alternative print styles for elements of an algebra
- Addition, subtraction, multiplication and composition (plethysm) of symmetric functions. Testing for homogeneity and equality.
- Decomposition of symmetric functions into basis elements and coefficients thereof.
- Symmetric functions can be coerced into polynomial rings.
- Frobenius homomorphism, inner product, tableax with a given generating function and corresponding characters
- Matrices converting from any of the 5 bases to any other of the 5 bases

15.4 Graphs

Graphs may be directed or undirected. In addition, their vertices and/or their edges may be labelled. A graph may be represented by means of its adjacency matrix. or by means of its adjacency list.

- 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
- Triconnectivity
- General vertex and edge connectivity in graphs and digraphs
- MaximumMatching in bipartite graphs
- Spanning trees, cut-vertices
- Cliques and maximal cliques, independent sets
- Chromatic number, chromatic index, chromatic polynomial
- Planarity testing, obstruction isolation, faces, embedding
- 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
- Interface for B. McKay's graph generation algorithm [29]

Brendan McKay's automorphism program `nauty` [30] 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.

The planarity tester and obstruction isolator are linear time algorithms; they are due to Boyer and Myrvold [4].

The triconnectivity algorithm is the classical linear time algorithm from Hopcroft and Tarjan [20] with corrections of our own and from Gutwenger and Mutzel [18].

The general vertex and edge connectivity machinery rests on two flow-based algorithms for networks: the Dinic algorithm and the push-relabel method (see below).

15.5 Networks

A network is a digraph whose edges are associated with a capacity. It may have multiple (i.e. parallel) edges and loops. A network is represented by means of its adjacency list.

- Operations: union, join, contraction
- Maximum flow and minimum cut

There are two implementations of the maximum flow algorithm. One is the classical Dinic algorithm with heuristics due to McKay, the other is a push-relabel method with heuristics mainly due to Cherkassky and Golberg *et al.* [11, 12].

15.6 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.

15.7 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 1 202 seconds. As with graphs and designs the G-set mechanism gives the action of the collineation group on any appropriate set.

15.8 Incidence Geometry

Magma 2.13 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 Bekenhout, 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*;
- $t : X \to I : x \mapsto t(x)$ 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.*

15.8.1 Incidence Geometries

- Creation of incidence geometries
- Conversion of an incidence geometry to a coset geometry
- Set of types, rank
- Diagram, incidence graph, elements
- Residue, truncation, shadow, shadowspace
- Properties: flag-transitive geometry, residually connected, firm, thin, thick
- Test if a geometry is a graph and conversion of such a geometry to a graph
- Automorphism group
- Correlation group

15.8.2 Coset Geometries

- Creation of coset geometries
- Conversion of an incidence geometry to a coset geometry
- Set of types, rank
- Diagram
- Residue, truncation
- Properties: flag-transitive geometry, residually connected, firm, thin, thick
- Test if a geometry is a graph and conversion of such a geometry to a graph
- Borel subgroup, group of the geometry
- Maximal and minimal parabolic subgroups
- Kernel of a geometry, i-kernels, quotient

16 Error-correcting Codes

Currently, the coding theory module is designed for linear codes over finite fields, linear codes over Galois rings (including special functionality for codes over $\mathbf{Z}_4$), additive codes over finite fields, and quantum stabilizer codes.

16.1 Linear Codes over Finite Fields

16.1.1 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

16.1.2 Linear Codes: Operations on Codewords

- Vector space operations: sum, difference scalar multiplication
- Syndrome
- Distance and weight
- Coordinates, support
- Trace

16.1.3 Linear Codes: Elementary Operations

- Sum, intersection
- Dual code
- Hull of a code
- External direct sum, Plotkin sum
- Modifying the codewords: augment, extend, expurgate, lengthen, puncture, shorten, etc
- Subcode generated by given codewords
- Subcode of a specified dimension
- Subcode generated by words of a given weight
- Coset leaders (in the case of a small code)

16.1.4 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-orthogonal
- Properties: perfect, nearly perfect
- Properties: maximum-distance separable, equidistant
- Properties: optimal for the Griesmer bound
- Properties: projective
- Determine whether two codes are equivalent

16.1.5 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, the weight distribution, and word collection algorithms. 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.

16.1.6 Linear Codes over Finite Fields: Construction of Families

- Hamming code, simplex code
- Reed-Muller codes
- Quadratic residue code, Golay codes, doubly circulant QR-code, twisted QR-code, power residue code
- BCH code
- Goppa code
- Chien-Choy code
- Alternant code, Fire code, Gabidulin code
- Srivastava, generalized Srivastava codes
- Reed-Solomon code, generalized Reed-Solomon code, Justesen code
- Maximum distance separable code

16.1.7 Linear Codes over Finite Fields: Algebraic-Geometric codes

Functions are provided for the construction of algebraic–geometric codes. The user chooses a plane curve X, and specifies a set of places of degree 1 on X and a divisor on X.

16.1.8 Linear Codes over Finite Fields: Decoding Algebraic-Geometric codes

Algebraic-geometric codes can be decoded efficiently up to the Goppa designated distance.

16.1.9 Linear Codes over Finite Fields: Changing the Alphabet

- Extension of the base field
- Restricting the alphabet to a subfield
- Subfield subcode
- Trace code
- Rewriting the alphabet, taken as the elements of $GF(q^m)$, as m-dimensional vectors over $GF(q)$

16.1.10 Linear Codes over Finite Fields: Combining Codes

- Concatenation of two codes
- Concatenated code
- Construction X
- Construction X3
- Construction XX
- Zinoviev code
- Construction Y1

16.1.11 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
- Tables of best known bounds, based on the codes database

16.1.12 Linear Codes over Finite Fields: Best Known Codes

Magma V2.13 incorporates a database containing constructions of the best known linear codes over F_2 of length up to 256, over F_3 of length up to 100, and over F_4 of length up to 100.

The binary codes database is complete in the sense that it contains a construction for every set of parameters, with the codes of length up to 36 known to be optimal. The database for codes over F_3 is also complete, with codes of length up to 21 known to be optimal. The databse for codes over F_4 is over 99% complete, with only 40 of the $4,150$ codes missing (the first such missing code coming at length 96. The codes of length up to 18 are known to be optimal.

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).

16.1.13 Linear Codes over Finite Fields: Decoding Algorithms

- Syndrome decoding
- Alternant decoding

16.1.14 Linear Codes over Finite Fields: Automorphism Group

- Automorphism groups of linear codes over GF(p) (prime p), GF(4)
- Testing pairs of codes for isomorphism over GF(p) (prime p), 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.

16.1.15 Linear Codes over Finite Fields: LDPC Codes

- Construction of LDPC codes from sparse matrices
- Deterministic LDPC constructions
- Random constructions from regular and irregular LDPC ensembles
- Iterative LDPC decoding
- Simulation of decoding performance on specified channels
- Density evolution on binary symmetric and Gaussian channels for given channel parameters, as well as threshold determination.
- Small database of good irregular LDPC ensembles.

16.1.16 Linear Codes over Finite Fields: Attacks on the McEliece Cryptosystem

All of the best known decoding attacks on the McEliece cryptosystem are available, as well as improved attacks.

- McEliece's attack
- Lee and Brickell's attack
- Leon's attack
- Stern's atack
- Canteaut and Chabaud's attack
- Generalized combinations of attacks

16.2 Linear Codes over Finite Rings

16.2.1 Linear Codes over Finite Rings: Creation

- Creation from a subspace of a vector space
- Creation in terms of a generator matrix
- Construction from a permutation
- 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

16.2.2 Linear Codes over Finite Rings: Operations on Codewords

- Module operations: sum, difference scalar multiplication
- Syndrome
- Hamming distance, Hamming weight
- Lee distance, Lee weight (codes over $\mathbf{Z}_4$)
- Coordinates, support
- Trace

16.2.3 Linear Codes over Finite Rings: Elementary Operations

- Sum, intersection
- Dual code
- External direct sum, Plotkin sum
- Modifying the codewords: augment, extend, expurgate, lengthen, puncture, shorten, etc
- Coset leaders (in the case of a small code)

16.2.4 Linear Codes over Finite Rings: Weight Distribution

- Minimum Hamming weight
- Hamming weight distribution
- Hamming weight enumerator
- Complete weight enumerator

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

- Minimum Lee Weight (Zimmermann algorithm)
- Lee weight distribution
- Symmetric weight enumerator
- Lee weight enumerator

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

- Kerdock code
- Preparata codes

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

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

16.3 Additive Codes

16.3.1 Additive Codes: Creation

- Additive codes defined as some K-additive subspace of F^n for some subfield K of F.
- Creation in terms of a generator matrix
- Construction of a cyclic code given the generator polynomial
- Universe code, repetition code, zero code
- Random additive code

16.3.2 Additive Codes: Operations on Codewords

- Vector space operations: sum, difference scalar multiplication
- Distance and weight
- Coordinates, support
- Trace

16.3.3 Additive Codes: Elementary Operations

- Sum, intersection
- Dual code
- Symplectic Dual code
- Plotkin sum
- Modifying the codewords: augment, extend, lengthen, puncture, shorten, etc
- Subcode generated by given codewords
- Subcode of a specified dimension
- Subcode generated by words of a given weight

16.3.4 Additive Codes over Finite Fields: Basic Properties

- Properties: cyclic
- Properties: self-dual, self-orthogonal
- Properties: symplecic self-dual, symplectic self-orthogonal

16.3.5 Additive 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

The fast algorithm for minimum weight and word collection is adapted from linear codes and provides performance similar to a calculation with an equivalently sized linear code.

16.3.6 Additive Codes over Finite Fields: Construction of Families

- Cyclic codes
- Quasicyclic codes

16.3.7 Additive Codes over Finite Fields: Automorphism Group

- Automorphism and permutation groups of additive codes over GF(4)
- Group actions on a code

16.4 Quantum Error-Correcting Codes

16.4.1 Quantum Codes: Creation

- Quantum stabilizer codes defined by symplectic self dual additive codes.
- Creation in terms of a generator matrix
- Construction of a cyclic code given the generator polynomial
- Universe code, repetition code, zero code
- Random quantum code

16.4.2 Quantum Codes: Basic Properties

- Properties: Cyclic, quasicyclic
- Properties: Self-dual
- Properties: Stablizer code

16.4.3 Quantum Codes: Error Group

- The group of errors on a quantum space
- Stabiliser group associated with a quantum code

16.4.4 Quantum Codes: Weight Distribution

- Minimum weight (Zimmermann algorithm)
- Weight distribution, weight enumerator
- Complete weight enumerator

The algorithm for minimum weight uses the Zimmermann algorithm for the underlying self dual code.

16.4.5 Quantum Codes: Constructions

- Cyclic codes
- Quasicyclic codes
- CSS code
- Direct sums
- Extend, shorten

16.4.6 Quantum Codes: Automorphism Group

- Automorphism and permutation groups of binary quantum codes (codes based on quaternary stabilizer codes)

17 Cryptography

17.1 Pseudo-Random Sequences

Magma provides tools for the creation and analysis of pseudo-random bit sequences. The universe of these sequences is generally 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

18 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:

18.1 Group Theory

- **Small Groups**: The Small Groups Library developed by Besche, Eick and O'Brien. This database contains all groups of order up to 2000, except the groups of order 1024, and a number of infinite series of larger groups.

- **The ATLAS Database:** Representations of nearly simple groups, as in the Birmingham ATLAS of Finite Group Representations. The data was supplied by Rob Wilson.

- **Almost Simple Groups:** A database of almost simple groups stored with their automorphism groups and maximal subgroups. In particular, the database includes all simple groups having a permutation representation of degree less than 1000. The following groups are included:

  - Alternating groups: A_n for $n \leq 999$;
  - Classical groups: $L_2(q)$, $L_3(q)$, $L_4(q)$ and $L_5(q)$ for all prime powers q; $L_6(3)$, $L_7(3)$; $L_d(2)$ for $d \leq 14$;
    $Sp_4(q)$ for all odd prime powers q and even $q \leq 16$; $Sp_6(3)$, $Sp_8(2)$, $Sp_{10}(2)$;
    $U_3(q)$ for all primes q and prime powers $q \leq 25$; $U_4(q)$ for prime powers $q \leq 7$, $U_6(2)$;
    $O_7(3)$, $O_8^{\pm}(2)$, $O_8^{m}(3)$, $O_{10}^{\pm}(2)$
  - Exceptional groups: $G_2(3)$, $G_2(4)$, $G_2(5)$
  - Twisted groups: Sz(8), $Sz(32)$, ${}^3D_4(2)$, ${}^2F_4(2)'$
  - Sporadic groups: M_{11}, M_{12}, M_{22}, M_{23}, M_{24}, J_1, J_2, J_3, HS, McL, He, Co_2, Co_3, Fi_{22}

- **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.

- **Perfect Groups** : The database of perfect groups of order up to a million constructed by Holt and Plesken.

- **Transitive Groups**: The transitive permutation groups of degree up to 30. The transitive groups of degree up to 15 were determined by Butler and McKay while the classification was extended to degree 30 by Hulpke.

- **Primitive Groups**: The table of primitive groups of degree up to 2499 (Sims, Roney-Dougal & Unger, Roney-Dougal).

- **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.

- **Irreducible Matrix Groups**: The table of irreducible subgroups of $GL(n,p)$ where p is prime and $p^n \leq 2499$ (Sims, Roney-Dougal & Unger, Roney-Dougal).

- **Irreducible Soluble Groups**: The irreducible soluble subgroups of GL(n, p) for $n > 1$ and $p^n < 256$, as classified by Short.

- **Finite Groups of Rational Matrices:** The maximal finite subgroups of $GL(n, \mathbf{Q})$, for n up to 31.

- **Quaternionic Matrix Groups:** The finite absolutely irreducible subgroups of $\mathrm{GL}_n(\mathcal{D})$ where $\mathcal{D}$ is a definite quaternion algebra whose centre has degree d over $\mathbf{Q}$ and $nd \leq 10$

- **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.

18.2 Number Theory

- **Cunningham Factorizations**: A database containing 237,578 factors f of numbers $a^n \pm 1$, where $a < 10000, n < 10000$, and $f > 10^9$. The factorizations of integers of the form $a^n \pm 1$, $a \leq 12$, were produced by Sam Wagstaff and collaborators (for n up to 1200), with contributions from Arjen Bot, Will Edgington, Alexander Kruppa and Paul Leyland (for larger n); for $13 < a < 99$ they are mainly from the Brent-Montgomery-te Riele extension of the Cunningham tables, with contributions by ECMNET and various individuals; for $100 < a < 1000$ they are mainly from the tables produced by Hisanori Mishima and Mitsuo Morimoto with additions from Rob Hooft, Pete Moore and others. For prime bases $a < 1000$ Richard Brent computed many of the factors for an unpublished extension of the Brent-Montgomery-te Riele tables.

- **Irreducible polynomials**: A database of sparse irreducible polynomials over GF(2) for all degrees up to 23,030, constructed by Allan Steel. The polynomials have the form $f(x) = x^n + g(x)$, where $g(x)$ is of minimal degree such that $f(x)$ is irreducible.

- **Conway polynomials**: A database of Conway polynomials for F_2 to F_127. These polynomials are primitive and provide standard definitions for finite field extensions (used in modular representation theory, for example). This database is based on lists built by Richard Parker and Frank Lübeck.

- **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. These polynomials have been determined by J. Klüners and G. Malle.

18.3 Algebraic Geometry

- **Cremona database of Elliptic Curves**: A database constructed by John Cremona that contains all isogeny classes of elliptic curves having conductor up to 30,000 is available.

- **Stein-Watkins Database of Elliptic Curves:** The Stein-Watkins database of 136,924,520 elliptic curves of conductor up to 10^8 is now available in Magma.

- **K3 Surfaces:** This comprises a collection of 24,099 K3 surfaces. For $g = -1, 0, 1, 2$, all K3 surfaces of genus g are included, there being 4281, 6479, 6627 and 6628 surfaces, respectively. For higher genus, the data associated to the 6628 K3 surfaces of genus 2 propagates in a predictable way, so only those K3 surfaces with codimension at most 7 and genus in the range 3 to 9 are included.

- **3-folds:** Basic machinery is provided that allows the user to generate lists of Fano 3-folds and Calabi–Yau 3-folds.

18.4 Topology

- **Fundamental Groups of 3-manifolds:** A database containing the 11,126 small-volume closed hyperbolic manifolds of the Hodgson-Weeks census. Each manifold record contains a presentation of the fundamental group and a homomorphism to S_n whose kernel has positive betti number.

18.5 Incidence Structures

- **Simple Graphs:** Magma contains an interface to the graph enumeration program of Brendan McKay which allows the user to rapidly construct all simple graphs on a given number of vertices. The graph generation program allows the user to specify one or more conditions thereby allowing the construction of all graphs on a given number of vertices satisfying the condition.

- **Strongly Regular Graphs:** A database containing a list of strongly regular graphs constructed by Brendan McKay, Ted Spence and others. This database contains strongly regular graphs on 25, 26, 27, 28, 29, 35, 36, 37 and 40 vertices. The graphs are indexed by the order of the graph, its degree, the number of common neighbours to each pair of adjacent vertices, and the number of common neighbours to each pair of non-adjacent vertices.

- **Hadamard Matrices;** This database includes all inequivalent Hadamard matrices of degree at most 28, and examples of matrices of all degrees up to 256. The matrices up to degree 28 have been provided by Neil Sloane while most of those of larger order have been provided by C. Koukouvinos, I. Kotsireas and G. Stelios.

- **Skew Hadamard Matrices;** This database includes known skew Hadamard matrices of degree up to 256. The matrices up to degree 28 have been provided by Neil Sloane while most of those of larger order have been provided by C. Koukouvinos, I. Kotsireas and G. Stelios.

18.6 Linear Codes

- **Best Known Binary Linear Codes:** A database containing constructions of the best known linear codes over F_2 of length up to 256 has been implemented by M. Grassl and the Magma group from tables of A. E. Brouwer. 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 $33,152$ binary codes.

- **Best Known F_3 Linear Codes:** A database containing constructions of the best known linear codes over F_3 of length up to 100. This database has been constructed by M. Grassl and the Magma group.

- **Best Known F_4 Linear Codes:** A database containing constructions of the best known linear codes over F_4 of length up to 100. This database has been constructed by M. Grassl and the Magma group.

19 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**, Eight Volumes, 3100 pages.
- W. Bosma, J. Cannon et al.: **Solving Problems with Magma**, 220 pages

References

[1] Selma Altınok, Gavin Brown, and Miles Reid. Fano 3-folds, $K3$ surfaces and graded rings. In *Topology and geometry: commemorating SISTAG*, volume 314 of *Contemp. Math.*, pages 25–53. Amer. Math. Soc., Providence, RI, 2002.

[2] R. Beals and C. R. Leedham-Green and A. C. Niemeyer and C. E. Prager and A. Seress, A black-box algorithm for recognising finite symmetric and alternating groups. *Trans. Amer. Math. Soc.* to appear.

[3] B. J. Birch. Hecke actions on classes of ternary quadratic forms. *Computational number theory (Debrecen, 1989)*, 191–212, de Gruyter, Berlin, 1991.

[4] J. Boyer and W. Myrvold. Simplified O(n) Planarity Algorithms. Submitted, 2001.

[5] W. Bosma, J. J. Cannon and A. Steel. Lattices of Compatibly Embedded Finite Fields, *J. Symb. Comp.* **24** (1997), pp. 351–369.

[6] S. Bratus and I. Pak. Fast constructive recognition of a black box group isomorphic to S_n or A_n using Goldbach's conjecture. *J. Symb. Comput.* **29** (2000), pp. 33–57.

[7] Gavin Brown. Datagraphs in algebraic geometry. In *Symbolic and Numerical Scientific Computing – Proc. of SNSC'01*, volume 2630 of *LNCS*, Heidelberg, 2003. Springer-Verlag. F. Winkler and U. Langer (eds.).

[8] 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>.

[9] J. Buchmann, M.J. Jacobson Jr., and E. Teske. On Some Computational Problems in Finite Abelian Groups. *Mathematics of Computation*, 66:1663–1687, 1997.

[10] Anita Buckley and Balázs SzendrHoi. The Riemann–Roch formula for 3-folds with canonical singularities and applications to Calabi–Yau 3-folds. 19pp preprint as math.AG/0309033, 2002

[11] B.V. Cherkassky and A.V. Golberg. On Implementing the Push-Relabel Method for the Maximum Flow Problem. *Algorithmica*, 19:390–410, 1997.

[12] B.V. Cherkassky, A.V. Golberg, Paul Martin, J.C. Setubal, and J. Stolfi. Augment or Push? A Computational Study of Bipartite Matching and Unit Capacity Flow Algorithms. Technical report, NEC Research Institute, 1998.

[13] J. Cremona. *Algorithms for modular elliptic curves.* Second edition. Cambridge University Press, Cambridge, 1997.

[14] 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.

[15] Dominique Duval. Rational Puiseux Expansions. *Compositio Mathematica*, 70:119–154, 1989.

[16] 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.

[17] V. Gebhardt. Constructing a short defining set of relations for a finite group. *J. Algebra* **233** (2000), pp. 526–542.

[18] C. Gutwenger and P. Mutzel. A Linear Time Implementation of SPQR-Trees. In J. Marks, editor, *Graph Drawing 2000*, volume 1984 of *Lecture Notes in Computer Science*, pages 70–90. Springer-Verlag, 2001.

[19] 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.

[20] J.E. Hopcroft and R.E. Tarjan. Dividing a Graph into Triconnected Components. *SIAM J. Comput.*, 2(3):135–158, 1973.

[21] W. Kantor and A. Seress. Black box classical groups. *Mem. Amer. Math. Soc.* **149** (2001), no. 708.

[22] 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.

[23] 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, 2001.

[24] D. R. Kohel and W. A. Stein. Component groups of quotients of $J_0(N)$. *Proceedings of ANTS IV*, 2000.

[25] 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.

[26] C. R. Leedham-Green and S. H. Murray. Variants of product replacement. *Contemp. Math.* **298** (2002), pp. 97–104.

[27] E. M. Luks and A. Seress. Computing the Fitting subgroup and solvable radical of small-base permutation groups in nearly linear time. *Groups and computation II (New Brunswick, NJ, 1995)*, 169–181, Amer. Math. Soc., 1997.

[28] 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.

[29] B. D. McKay. Isomorph-free exhaustive generation. *J. Algorithms*, **26** (1998), 306–324.

[30] B. D. McKay. Practical Graph Isomorphism. *Congressus Numerantium*, **30** (1981), 45–87.

[31] P. M. Neumann. Some algorithms for computing with finite permutation groups. *Groups - St. Andrews 1985*, London Math. Soc. Lecture Notes 121, 1986.

[32] A. Pizer. An algorithm for computing modular forms on $\Gamma_0(N)$, *J. Algebra* **64** (1980), no. 2, 340–390.

[33] Miles Reid and Kaori Suzuki. Cascades of projections from log del Pezzo surfaces. In *Number theory and algebraic geometry – to Peter Swinnerton-Dyer on his 75th birthday*, pages 227–249. Cambridge University Press, Cambridge, 2003.

[34] C. M. Roney-Dougal and W. R. Unger. The affine primitive permutation groups of degree less than 1000. *J. Symb. Comput.* **35** (2003), pp. 421–439.

[35] C. M. Roney-Dougal. The primitive groups of degree less than 2500. *Submitted.*

[36] A. Steel. A New Algorithm for the Computation of Canonical Forms of Matrices over Fields, *J. Symb. Comp.*, **24** (1997), 409–432.

[37] A. Steel. A New Scheme for Computing with Algebraically Closed Fields, *Proceedings of ANTS V*, ed. C. Fieker and D.R. Kohel, Springer LCNS **2369** (2002), 493–507.

[38] A. Steel. Conquering Inseparability: Primary Decomposition and Multivariate Factorization over Algebraic Function Fields of Positive Characteristic. JSC **40** (2005), 1053–1075.

[39] A. Steel. Magma V2.12 is up to 2.3 times faster than GMP 4.1.4 for large integer multiplication, `http://magma.maths.usyd.edu.au/users/allan/gb/`.

[40] A. Steel. Gröbner Basis Timings Page, `http://magma.maths.usyd.edu.au/users/allan/gb/`.

[41] W. A. Stein. *Explicit approaches to the theory of modular abelian varieties.* Ph.D. thesis, University of California, 2000.

[42] Robert J. Walker. *Algebraic Curves*, pages 98–99. Springer-Verlag, 1978.