Summary of New Features in Magma V2.15

December 2008

1 Introduction

This document provides a terse summary of the new features installed in Magma for release version V2.15 (December 2008).

Previous releases of Magma were: V2.14 (October 2007), V2.13 (July 2006), V2.12 (June 2005), V2.11 (May 2004), V2.10 (April 2003), V2.9 (May 2002), V2.8 (July 2001), V2.7 (June 2000), V2.6 (November 1999), V2.5 (July 1999), V2.4 (December 1998), V2.3 (January 1998), V2.2 (April 1997), V2.1 (October 1996), V2.01 (June 1996) and V1.3 (March 1996).

2 Summary

Lattices and Quadratic Forms

- *Lattices*
  - The enumeration algorithm underlying functions relying on properties of short lattice vectors has been rewritten. The new code is more efficient. Moreover, although it relies on double precision computations as before, the result is now provably guaranteed.
  - A new implementation is provided for minimum, kissing number, theta series and shortest or short vectors. By default, the results returned by the new versions are now guaranteed. Depending on the inputs, the CPU time is either similar or dramatically better than the previous code.
  - Functions are provided to estimate the run-times of the calculation of minimum, kissing number, theta series and shortest or short vectors.
  - (March, 2009) The code for computing the automorphism group of a lattice has been replaced. The new implementation includes some important improvements over the old algorithm. These include the exploitation of stronger invariants and the use of partition refinement in the backtrack search. The new version is able to construct automorphism groups for lattices that were previously well beyond the old program and it also significantly speeds up many examples that were hard for the old algorithm. The same improvements flow through to isometry testing.

Global Arithmetic Fields

- *Number Fields*
  - In the case of Kummer extensions a new special-purpose algorithm that is much faster than the general algorithm is now used to compute maximal orders.
- *Galois Theory*
  - The Fieker-Klüners algorithm for constructing Galois groups has been implemented for irreducible polynomials over $Q(t)$.
  - It is now possible to compute the Galois group of a general polynomial over Z or Q. That is, the polynomial need not be irreducible.
- *Class Field Theory*
  - A new algorithm has been implemented for the computation of maximal orders in abelian extensions.
- *Algebraic Function Fields*
  - In the case of Kummer extensions a new special-purpose algorithm that is much faster than the general algorithm is now used to compute maximal orders.
  - Completion of function fields and their orders at places of degree greater than 1 has been rewritten and extended to fields of characteristic zero.
- *Artin Representations*
  - A package for Artin representations (complex representations of the Galois groups of finite extensions of $\mathbf{Q}$) has been developed by Tim Dokchitser. This is primarily designed for computing the character values efficiently for use in the associated L-series.

Local Arithmetic Fields

- *General Local Fields*
  - A new class of local field is now supported: a local field defined by the extension of any local field by an arbitrary irreducible polynomial. These local fields are referred to as *general* local fields.
  - An isomorphism between a general local field and a local field constructed as an extension by an inertial polynomial followed by an extension by an eisenstein polynomial can be constructed.
  - Subfields and homomorphisms are supported for general local fields.
  - Some support for calculation with automorphisms and Galois groups is provided.

Group Theory

- *Finite Groups*
  - The strategy used when computing all conjugacy classes of subgroups has been extensively reworked to avoid inefficiencies in the case of larger groups. An example of such a group is the wreath product of $P\Gamma L(2, 16)$ with C_2 where the execution time was reduced from 150,000 seconds to 860 seconds.
  - The functions for computing conjugacy classes of subgroups now apply to any matrix group for which Magma can compute the soluble radical and corresponding quotient. This includes finite matrix groups over GF(q), $\mathbf{Z}$, $\mathbf{Q}$, and $\mathbf{Z}/n\mathbf{Z}$.
  - When testing whether a subgroup of a permutation group is maximal, the algorithm will now exploit our ability to construct the maximal subgroups of larger groups. This allows the testing for maximality of subgroups having much larger index and also speeds up the test in the case of subgroups of moderate index.
- *Abelian Groups*
  - The automorphism group of a finitely-generated abelian group may now be constructed using code developed by Derek Holt.
- *Finitely-Presented Groups*
  - The automorphism group of a free group can now be constructed (implemented by Derek Holt).
  - Derek Holt has implemented a version of his group homomorphism algorithm which searches for homomorphisms from a finitely-presented group into a (small) soluble group given by a power-commutator presentation (pc-group).
- *Group Cohomology*
  - Derek Holt has extended his group cohomology package to include the computation of the first cohomology group for an infinite finitely-presented group.

Algebras

- *Exterior Algebras (New)*
  - A type for exterior algebras has been installed. Such an algebra is skew-commutative and its elements have a very efficient representation, resulting in fast Gröbner basis and module theory computations over such algebras.

Commutative Algebra

- *Gröbner Bases*
  - The computation of primary decomposition and the radical has been greatly improved.
  - A new boolean polynomial ring type is supported.
  - The operations for accessing and specifying monomial orders have been expanded.
- *Ideals and Modules*
  - A major expansion of the facilities for modules has been undertaken. A variety of base rings is now available, including localizations of multivariate polynomial rings. A new module homomorphism type is fully supported, gradings are now fully handled in all functions, and the computation of free resolutions has been greatly improved.
  - A function has been added to compute the dimensions of cohomology groups of the coherent sheaf represented by a graded module or its twists.
- *Invariant Theory*
  - Some speedups have been achieved for computing primary and secondary invariants in certain cases.
  - The fast algorithm of Simon King for fundamental invariants has been implemented.

Algebraic Geometry

- *General Schemes*
  - A function to compute the dimension of coherent sheaf cohomology on ordinary projective schemes has been added. This works with sheaves represented by graded modules.
  - A large package for hypersurfaces in 3-dimensional projective space has been provided by Josef Schico and Tobias Beck. Features include formal desingularisation leading to the computation of birational invariants of any non-singular birational model of the surface. These invariants include the arithmetic genus, the geometric genus and any higher plurigenera. Parametrisation routines are provided for line and conic bundles and Del Pezzo surfaces of degree 5 and 7. These enable the user to determine if a surface is rational or birationally ruled. If rational, then a parametrisation over the basefield can be calculated if it exists.

Arithmetic Geometry

- *Conics*
  - Solution and parametrization of conics over number fields is now available, using a new implementation of solution of diagonal conics (based on a variant of Lagrange's method) that is faster and also more effective than the previous implementation. Care is taken to avoid unnecessarily factorizing large integers as a result of diagonalizing.
- *Elliptic Curves over Finite Fields*
  - Point-counting over prime finite fields smaller than 2^{29} has been sped up by a factor of about two on many computers.
  - p-adic deformation methods for fast point-counting have been added for small characteristics greater than about 40.
- *Elliptic Curves over the Rationals*
  - A new fast implementation of 2-descent is included, which avoids much of the general machinery. The new function has been tested on the Cremona database (via the Parity conjecture) and other curves.
  - An efficient new implementation of the Cassels-Tate pairing between 2-coverings is provided. The algorithm, due to Donnelly, is much cleaner than the version previously implemented.
  - The Cassels-Tate pairing between a 4-covering and a 2-covering is implemented, using a new algorithm due to Donnelly.
- *Elliptic Curves over Number Fields*
  - The Cassels-Tate pairing between 2-coverings is implemented.
- *Elliptic Curves over Rational Function Fields*
  - The `TwoDescent` routine is now much more efficient.
  - The Cassels-Tate pairing between 2-coverings is implemented.
- *Models of Genus One Curves*
  - New routines for minimisation and reduction of genus one curves of degrees $2, 3, 4$ and 5 are included. This code was written by Tom Fisher, using new algorithms for some of cases due to Cremona, Fisher and Stoll.
- *Hyperelliptic Curves over Finite Fields*

- A number of functions written by R. Lercier and C. Ritzenthaler and based on a new set of absolute invariants have been added for genus 2 curves. These allow more general computation of twists and geometric automorphism groups, which works in all characteristics. Most of the functions also work over the rationals.

- *Hyperelliptic Curves over the Rationals*

  - For genus 2 curves, a new implementation of Chabauty's method combined with the "Mordell-Weil" sieve is provided. This determines precisely the set of rational points on the curve. It works by combining information from local calculations at many different primes with knowledge of the Mordell-Weil group. This was developed by Michael Stoll.

- *L-series*

  - The L-series attached to an Artin representation can now be created and efficiently computed. More generally, the twist of an arbitrary L-series by an Artin representation can be constructed. This functionality was developed by Tim Dokchitser.

Modular Arithmetic Geometry

- *Hilbert Modular Forms*

  - A substantial package for computing Hilbert modular forms is now available. This is the first released version, and will continue to be developed in subsequent releases. The current functionality concentrates on computing Hecke operators on spaces of cusp forms and new forms. The package handles arbitrary totally real fields, and computes spaces of forms on $\Gamma_0(N)$ for any ideal N of the ring of integers, for any weight whose components are all at least 2.

    The algorithms make use of the Jacquet-Langlands correspondence, and work by computing the Hecke action on a space of automorphic forms on some order in a suitable quaternion algebra. Two separate, complementary algorithms are implemented: the main one, due to Dembele, uses definite quaternion algebras and is essentially an efficient reformulation of the Brandt module approach. The other algorithm, currently only implemented for weight 2, uses Fuchsian groups arising from units in indefinite quaternion algebras, and is an application of Voight's algorithm for computing fundamental domains of these groups.

    Many of the key steps in the first algorithm are already highly optimized. A very substantial range of fields can be handled, for instance the program has been used to compute elliptic newforms over a degree 8 field, and Hecke operators can often be computed even when the dimension of the space is large.

Coding Theory

- *Linear Codes over Fields*
  - An $[n, k]$ linear code C is said to be a best known linear $[n, k]$ code (BKLC) if C has the highest minimum weight among all known $[n, k]$ linear codes. A number of new codes have been added to the BKLC database constructed and maintained by Markus Grassl. Specifically, 17 codes over $GF(3)$, 15 codes over $GF(4)$, and 5 codes over $GF(7)$ have been added.
- *Quantum Codes*
  - An $[[n, k]]$ quantum stabiliser code Q is said to be a best known $[[n, k]]$ quantum code (BKQC) if Q has the highest minimum weight among all known $[[n, k]]$ quantum codes. The BKQC database constructed and maintained by Markus Grassl has been expanded as follows: The maximal length of codes has been increased from 35 to 50 while the minimum distance of 10 codes has been improved.

3 Documentation

New chapters in the Handbook for V2.15 are:

- General Local Fields
- Local Polynomial Rings
- Artin Representations
- Hilbert Modular Forms
- Algebraic Power Series
- Algebraic Surfaces

4 Language and System Features

Changes:

- The `error` statement now prints `Runtime error:` before the given arguments.

New Features:

- The mechanism for attaching and managing package signature files has been sped up. Consequently, the time for starting up Magma (which attaches all the standard system package files) is rather faster than for previous versions.
- A larger number of internal memory blocks is now allowed in the system on 64-bit machines. This fixes some problems with some objects like very high degree polynomials with large integers as coefficients.

5 Aggregates

5.1 Sequences

New Features:

- One may now use $x\hat{}\hat{}m$ in sequence constructions for m consecutive copies of x.

5.2 Multisets

New Features:

- Arbitrarily large integers are now allowed for multiplicities of elements in multisets.
- Multisets are now printed in sorted order when possible (including nested multisets) which makes it much easier to examine multisets visually.

6 Basic Rings

6.1 Integer Ring

New Features:

- New functions `ShiftLeft`, `ShiftRight`, `ModByPowerOf2` for fast bit operations on integers.

6.2 Multivariate Polynomial Rings

New Features:

- New function `Polynomial(C, M)` to create a multivariate polynomial from parallel sequences C and M of coefficients and monomials, respectively (thus matching `Coefficients` and `Monomials`).
- New function `CoefficientsAndMonomials(f)` to return the coefficients and monomials of f as parallel sequences.
- New function `Degree` as synonym for `WeightedDegree` (which respects the grading).
- Powering of a polynomial now uses the Frobenius map in characteristic p when applicable.

6.3 Finite Fields

New Features:

- Computing roots of a polynomial over a finite field has been greatly sped up in the case that the degree of the polynomial is large and the field is small.

7 Linear Algebra and Module Theory

7.1 Matrices

New Features:

- Some fundamental matrix algorithms have been greatly sped up for matrices over GF(2)[x] and GF(2)(t).
- A fast modular algorithm for matrix multiplication over $K[x]$ and $K[x]/\langle f(x)\rangle$ for K a small finite field has been implemented. Several algorithms based on multiplication have been similarly sped up.
- One may now compute the rank of a matrix over a ring for which a field of fractions is defined. This includes most domains which are not fields or Euclidean.

7.2 Modules over Dedekind domains

Changes:

- Module printing has been economised and maps are returned from creation functions only when requested.

New Features:

- Given two pseudo matrices their intersection can be computed using `meet`.

8 Lattices and Quadratic Forms

8.1 Minima and Vector Enumeration

Changes:

- The enumeration algorithm underlying many tasks related to the study of short lattice vectors has been rewritten. The new code is more efficient. Moreover, although it relies on double precision computations like the previous implementation, the result is provably guaranteed.
- The functions `Minimum`, `KissingNumber`, `ShortVectors`, `ShortVectorsMatrix`, `ShortestVectors`, `ShortestVectorsMatrix`, and `ThetaSeries` have been rewritten. By default, the results returned by the new versions are now guaranteed to be correct. Depending on the inputs, the execution time is similar to or dramatically better than previously.

New features:

- A new `Proof` option has been added for the functions `Minimum`, `CentreDensity`, `CenterDensity`, `KissingNumber`, `ShortVectors`, `ShortVectorsMatrix`, `ShortestVectors`, `ShortestVectorsMatrix` and `ThetaSeries`. It allows the user to disable the correctness guarantee: in most cases the code is unlikely to run significantly faster, but there could be inputs for which the provable variant fails because it deems impossible to provide correctness guarantees.
- A new `SetVerbose`("Enum", b) flag allows the user to obtain information during the execution of the enumeration algorithm.
- The function `EnumerationCost` provides a priori estimates of the run-times of the executions of the functions listed in the first item. More precisely, `EnumerationCost`(L, u) provides a heuristic evaluation of the size of the tree to be visited during the enumeration of vectors within the prescribed norm u. If u is not given, an upper bound to the lattice minimum is computed.

Bug Fixes:

- Two inconsistencies in the lattice database have been corrected. For lattice "Q32"' the stored automorphism group has been corrected and it now has order equal to the group order stored in the database. For lattice "LAMBDA(G1)" the stored group order is corrected to be the order of the stored group. In both cases the automorphism group of the lattice has been recomputed and checked to be equal to the stored group.

8.2 Attributes of Lattices

New Features:

- A lattice attribute L`MinimumBound allows the user to give an upper bound on the minimum of the lattice L.

9 Global Arithmetic Fields

9.1 Algebraic Number Fields

Changes:

- `PowerProduct` for sequences of number field elements has been added with functionality the same as that of `ProductRepresentation`. Both have been improved for elements represented in a maximal order of an absolute extension.

New Features:

- A new algorithm for p-maximal orders of Kummer extensions based on Stichtenoth. This has in most cases sped up both p-maximal and maximal order computations in Kummer extensions.
- The intersection of two orders of a number field sharing the same equation order can be computed using the intrinsic `meet`.
- `InfinitePlaces` are now accessible from an order as well as a field.
- The `CoefficientIdeals` of an order or an ideal can now be retrieved.
- The `Codifferent` of an ideal can be calculated.

Bug Fixes:

- A bug has been fixed in residue field computations which fixed a problem which was seen in prime splitting.
- A restrictive check on the input orders to `subset` has been removed so that `subset` can be used on orders which are not identical.

9.2 Algebraic Function Fields

Removals and Changes:

- `Completion` of a global function field or an order thereof at a place of degree greater than 1 has been reimplemented.
- `ResidueClassField` of a place now returns the map from the order of the place into the field same as `ResidueClassField(Ideal)`.
- Determining second generators of prime ideals creating during prime splitting has been improved.
- More storing of prime splitting has been implemented to avoid splittings being calculated twice in a row.
- `CommonZeros` has been reimplemented.
- It is now possible to coerce between a field and its subfields.

New Features:

- Function fields and their orders with constant field having characteristic zero can now be completed at places of degree greater than 1. This functionality is also accessible via the intrinsic `Expand`.

– Intersections of orders of function fields sharing the same equation order can now be gained using the operation `meet`.

– `PowerProduct` for sequences of function field elements has been added with functionality the same as that of `ProductRepresentation`.

– The `CoefficientIdeals` of an order or an ideal can now be retrieved.

– Inclusion of an order in another order can be tested using `subset`.

– The `Codifferent` of an ideal can be calculated.

Bug Fixes:

– A bug has been fixed in residue field computations which fixed a problem which was seen in prime splitting.

– Some bugs in elements in product representation have been fixed, in taking norms and mapping to the residue field.

9.3 Abelian Extensions

New Features:

– A new algorithm for computing maximal orders in abelian extensions has been completed. It is available through the parameter `Al := "Kummer"` but is used by default. The algorithm computes maximal orders of kummer extensions (using the new implementation for number fields) known to the abelian extension then combines these with the components of the extension to gain a maximal order faster than by using the Round2 algorithm. The `"Round2"` and `"Discriminant"` algorithms are still available as options to the `"Al"` parameter.

9.4 Artin Representations

Features:

– This new package handles complex representations on the Galois groups of finite extensions of $\mathbf{Q}$. A new type `ArtRep` has been introduced for these objects.

– For a finite extension $K/\mathbf{Q}$, `ArtinRepresentations(K)` returns the irreducible Artin representations of K.

– Key invariants, such as the `Conductor` and other ramification data, can be computed.

– The character of the representation may be obtained as an element of the character group of the finite Galois group. This means that the Magma functionality for finite group characters may be used.

– The direct sum and the tensor product of two Artin representation can be constructed.

10 Local Arithmetic Fields

10.1 Local Fields

Bug Fixes:

- Added `IsWildlyRamified` which had been wrongly named `IsTamelyRamified`.
- Calculation of multiplicities of roots of polynomials over local fields has been fixed.

10.2 Local Fields defined by Arbitrary Polynomials (New)

The local fields which have been previously available in Magma have had the restriction that they must be constructed by taking extensions using only inertial or eisenstein polynomials. These new local fields can be defined using extensions by any irreducible polynomial.

New Features:

- The isomorphism between one of these local fields and a local field constructed by extensions by inertial and eisenstein polynomials only is accessed through `RamifiedRepresentation`.
- A `QuotientRepresentation` of the field as a quotient of a polynomial ring by the defining polynomial is available.
- The `BaseRing` and `DefiningPolynomial` of a field can be retrieved as can its `Degree`, `InertiaDegree` and `RamificationDegree`. Local fields also have an accessible `Precision` and retrievable `Prime`.
- Subfields of these local fields can be constructed using `<sub | >`.
- The `Discriminant` of a local field can be calculated and its `ResidueClassField` constructed.
- Various ramification properties of a field can be tested.
- These local fields have elements on which one can use basic arithmetic and predicates.
- An `IntegralBasis` can be calculated and elements tested for being integral.
- An `InertialElement` and a `UniformizingElement` are available.
- Homomorphisms from these local fields can be defined and applied to the elements.
- `Valuation`, `RepresentationMatrix` and `Eltseq` of a local field element can all be calculated.
- The `AutomorphismGroup` of a field can be calculated as can `Decomposition`, `Ramification` and `InertiaGroup`s. A `FixedField` of any one of these groups is also available.
- `Factorization` of polynomials over these local fields is available.

10.3 Algebraic Power Series

New Features:

- This is a new type (`RngPowAlgElt`) for working by lazy evaluation with multivariate power series over a field which are roots of an polynomial equation. The package was designed and implemented by Töbias Beck.

– A basic series is defined as the root of a univariate polynomial with coefficients in a multivariate polynomial ring (the approximation domain) with a specified initial expansion. The series must be expandable in e-th roots of the variables, for some $e \geq 1$, and the finite expansions of the series to a given degree are as elements of the approximation domain with variables then representing their e-th roots. e is part of the series data.

– It is possible to specify a set of algebraic power series which are substituted into the coefficient variables of the given defining polynomial to give the actual defining polynomial. In this way, elements in composite algebraic extensions can be defined recursively.

– There is a second variant, which allows a composite algebraic series to be defined by substituting a sequence of such series into the variables of another one.

– All series are "lazy". At any time, they store an expansion to a certain degree and the main function `Expand` computes the expansion to a specified higher degree as required. For series with substitutions, the expansions are carried out recursively.

– There are also functions for testing for (exact) equality of algebraic power series and testing finiteness (is the series actually a multivariate polynomial?).

– There is a function `SimplifyRep` to express any series as a simple series with an irreducible defining polynomial with polynomial coefficients and no substitutions.

– There are functions for basic arithmetic (addition, multiplication etc.) on series.

– Amongst various constructors, `RationalPuiseux` returns all roots of a *quasi-ordinary* univariate polynomial over a multivariate polynomial ring. For such a defining polynomial, all roots are expandable in power series in e-th roots of the variables for some e with a possible algebraic extension of the basefield. Basefield extensions are worked out by the function.

10.4 Newton Polygons

Bug Fixes:

– Given a face of a newton polygon defined by $a * x + b * y + c = 0$ a fix has been made to avoid problems when a, b or c are large.

11 Groups

11.1 Finitely Presented Groups

New Features:

- The automorphism group of a free group may be constructed using code developed by Derek Holt.
- Homomorphisms of a finitely-presented group onto a (small) finite polycyclic group may be found using the intrinsic `Homomorphism`. The code was provided by Derek Holt.

Bug Fixes:

- A bug when rewriting an element of a group as an element of a subgroup has been fixed.
- A bug to do with the Todd-Coxeter algorithm and involution generators of the group has been fixed.
- A bug to do with using the Todd-Coxeter algorithm with the trivial subgroup has been fixed.
- A bug where `Simplify` may have substituted protected generators has been fixed.

11.2 Finite Groups

Changes:

- The values returned by `AbelianInvariants` and `PrimaryAbelianInvariants` have been standardised across the four types of groups `GrpAb, GrpMat, GrpPerm, GrpPC`. The functions `AbelianBasis` and `PrimaryAbelianBasis` have been altered to match the output of the corresponding `Invariants` functions.

New Features:

- The algorithm used by the `Subgroups` family of functions to expand maximal subgroups of a group to all subgroups has been revised. There are two main changes. When a subgroup needing expansion is encountered where either the subgroup has a large elementary abelian chief factor or the expansion algorithm would previously have been called recursively, only maximal subgroups of the subgroup are computed, and are treated the same way as maximal subgroups of the original group needing expansion. The other change is in the handling of p-subgroups, where, as far as is convenient, one subgroup is chosen for each prime p, and the expansion of this subgroup gives all p-subgroups of the group. Any p-subgroups from other expansions are ignored.

  These changes eliminate much conjugacy testing of subgroups, and also allow the construction of all subgroups of groups with larger elementary abelian chief factors than was previously possible.
- The arbitrary limit on the size of input group to the `FPGroup` function has been removed. The order limit is now the limit on the degree of a permutation group, $2^{30} - 1$.

Bug Fixes:

- Some instances of taking an extremely long time to compute a presentation of a subgroup when using the `Subgroups` algorithm have been fixed.
- The maximal subgroups of $PSp(4, 4) : 4$ have been corrected.
- A bug when computing elementary abelian subgroups has been fixed.

11.3 Permutation Groups

Changes:

– The `CompositionSeries` function for permutation groups now gives a full composition series for the group, not merely a series for the abelian quotient.

New Features:

– The default algorithm for computing the regular representation of a group has been changed to the "Wang" algorithm. This is faster than the "Canonical" algorithm, and is the first step in a call to the `FPGroup` function, for instance.

– The use of STCS applied to a regular group has been improved to avoid any computation of elements in the (trivial) point stabilizer. This is aimed at speeding the `FPGroup` function in particular.

– When defining homomorphisms with domain a permutation group, they may be defined using images of any generating set.

– The textttIsMaximal(G,H) function call now computes the maximal subgroups of `G` and checks `H` against the list found.

Bug Fixes:

– Return `FewGenerators` as a sequence as documented rather than a set.

– A bug in the code for computing normal structure of a permutation group has been fixed. This bug caused crashes when it turned up, rather than incorrect answers, and needed quite exotic combinations of circumstances to occur, but was responsible for several reported crashes.

– The default maximum number of random elements tried when applying `RandomSchreier` to a low degree group has been increased to make getting the full group more likely.

– The default subgroup centralizer algorithm now checks to see if using the `CentralizerOfNormalSubgroup` method is possible, and uses this if it is.

– A bug when constructing a homomorphism with permutation group domain has been fixed.

11.4 Matrix Groups Over General Rings

Changes:

– The meaning of the `CommutatorSubgroup` command for matrix groups has been changed to agree with other group types. The call `CommutatorSubgroup(G,H,K)` now returns $[H, K]$ as a subgroup of G, rather than the normal closure $[H, K]^G$, as it used to.

New Features:

– The `Subgroups` family of functions now applies to any matrix group where Magma can compute the soluble radical and corresponding quotient of the group. This includes finite matrix groups over finite fields, the integers, the rationals and integers modulo n.

– The algorithm used to compute the order of a matrix over the integers has been improved and is much quicker for matrices of large, but finite, order. This affects matrix groups over the integers, the rationals, and number fields.

Bug Fixes:

- A bug in testing matrix groups over number fields for finiteness has been fixed. The previous algorithm could say that the group was finite when it was not. This has been corrected, and the algorithm improved for speed.
- Very slow membership tests in sets of matrix group elements have been addressed and improved.
- When forming a conjugate of a group, the group is no longer tested for finiteness.

11.5 Matrix Groups over Finite Fields

New Features:

- In matrix groups over finite fields which contain the special linear group, conjugacy classes are now computed by enumerating class invariants. This is much faster than the previous method, and works for much larger degrees and field sizes.

Bug Fixes:

- A bug in testing membership of orthogonal groups in characteristic 2 has been fixed. The previous code could say that a matrix was in the group when it preserved the bilinear form, but not the quadratic form.
- A bug in memory management associated with the use of the `IsAbsolutelyIrreducible` function has been fixed. The bug involved the second return value appearing to be assigned even when the result was true.
- Generators for `OmegaPlus(4,q)` have been corrected.

11.6 Almost Simple Groups

New Features:

- Functions have been installed for creating the conformal orthogonal, symplectic and unitary groups. The functions are `ConformalOrthogonalGroup`, `ConformalOrthogonalGroupMinus`, `ConformalOrthogonalGroupPlus`, `ConformalSymplecticGroup`, and `ConformalUnitaryGroup`.
- A more efficient version of the spinor norm algorithm has been implemented, which works for elements of the full orthogonal group (rather than just the special orthogonal group).

Bug Fixes:

- The code for determining whether a matrix group defined over a finite field fixes a quadratic form now works in the case in which the field is defined as an extension of a subfield.

11.7 Finite Soluble Groups

Changes:

– As part of improvements to `CompactPresentation` and the reconstruction function `PCGroup`, there has been a small change to the compact presentation produced. Note that old-style compact presentations are still valid input to the reconstruction function, will continue to be valid indefinitely, and still give the same presentation.

New Features:

– A new algorithm has been installed to compute the `Agemo` function on a p-group. This function does not compute the conjugacy classes of the group as the old algorithm did, so is applicable to larger groups. This has also improved computation of `JenningsSeries`.

– The algorithm for reconstructing a group stored as a compact presentation has been improved to reduce the time of this operation for large groups. As part of this improvement there has been a small change to the compact presentations.

Bug Fixes:

– Various bugs to do with the trivial group, particularly with centralizers and p-quotients, have been fixed. The bug with centralizers caused certain character theory calculations in soluble groups to go awry.

– Inclusion and projection maps for direct products of a sequence of groups have been fixed.

– A number of operations with group elements were unnecessarily slow and have been improved.

– A bug when computing the normal closure of a group in itself has been fixed.

11.8 Databases of Groups

Bug Fixes:

– Problems that occurred when the Rational and Quaternionic matrix groups databases were open simultaneously have been fixed.

11.9 Straight Line Program Groups

New Features:

– When defining a homomorphism with domain an SLP-group, it is now possible to use the generator image pairs notation `x -> y`.

Bug Fixes:

– Some bugs with `Evaluate` when the word group and the target group are equal have been fixed.

– Giving `Evaluate` an empty list as first argument no longer crashes.

12 Algebras

12.1 Associative Algebras

Bug Fixes:

- `RepresentationMatrix` of an element has been rewritten to allow for proper selection of right or left multiplication.

12.2 Exterior Algebras (New)

A new type for exterior algebras is now supplied (created with the function `ExteriorAlgebra`). Such an algebra is skew-commutative and is a quotient of the free algebra $K\langle x_1, \ldots, x_n\rangle$ by the relations $x_i^2 = 0$ and $x_i x_j = -x_j x_i$ for $1 \leq i, j \leq n$, $i \neq j$.

Because of these relations, elements of the algebra can be written in terms of commutative monomials in the variables (via a collection algorithm), and the associated algorithms are much more efficient than for the general noncommutative case. Gröbner bases of ideals can be computed very efficiently (the Faugere F_4 algorithm has been specially adapted for this). Furthermore, the extensive module theory also works over exterior algebras.

12.3 Basic Algebras

New Features:

- Functions for computing the Ext algebra of a basic algebra have been added. This includes functions to compute the basic algebra of the Ext algebra.
- There are now several function that create basic algebras of special interest. Included in these are the basic algebras of group algebras, Schur algebras and Hecke algebras over finite fields, as well as split basic algebras of matrix algebras and endomorphism algebras.

12.4 Characters of Finite Groups

New Features:

- The computation of conjugacy class fusion in elementary subgroups of a group has been improved. This is a fundamental operation in Unger's algorithms for the character table and for computing Schur indices.

Bug Fixes:

- The intrinsic function `CharacterTable` now applies to finite groups of type `GrpAb`.
- A serious memory problem when computing the characters of an abelian group has also been fixed.

13 Lie Theory

13.1 Coxeter Groups as FP Groups

Changes:

– `BruhatDescendents` has been renamed to `BruhatDescendants`.

14 Commutative Algebra

14.1 Ideal Theory and Gröbner Bases

Changes:

- The function `PolynomialRing(R, W)`, where W is a sequence of weights, now uses the *weighted grevlex* (`grevlexw`) order with `W` as the main order for the ring.
- The function `Homogenization` now uses the grading to determine the homogenization.

New Features:

- Monomial orders can now be described by tuples. The functions `PolynomialRing(R, n, T)`, `PolynomialRing(R, W, T)` and `ChangeOrder(I, T)` may now take a tuple T for the order. Conversely, the new function `MonomialOrder` returns a tuple describing the monomial order of a ring or ideal. A new function `MonomialOrderWeightVectors` also returns the weight vectors corresponding to the monomial order.
- A new boolean polynomial ring type (created by `BooleanPolynomialRing`) has been introduced to provide efficient Gröbner basis computations with ideals in polynomial rings over GF(2). The effect of computing with an ideal in this ring is the automatic inclusion of the *field polynomials* $x_i^2 + x_i$. This saves time and memory because a compact bit vector representation is available for the monomials and conversion to and from this is not needed. The `lex`, `glex` and `grevlex` monomial orders are supported. Large boolean polynomials are easily created using the `BooleanPolynomial` function.
- Computing the minimal basis of an ideal has been greatly improved in both speed and memory usage by a new algorithm based directly on the F_4 algorithm (instead of the former algorithm which was based on linear algebra).
- A new Hilbert-driven Buchberger algorithm has been implemented (based on the F_4 algorithm) and is dramatically faster for several kinds of input. This speeds up the function `HilbertGroebnerBasis` and the computation of primary invariants for invariant rings.
- The algorithms for the computation of the radical or primary decomposition of a positive-dimensional ideal have been greatly sped up. A new fast test for primality of a positive-dimensional ideal has also been implemented. Several leaks have also been fixed. Thus the functions `Radical`, `RadicalDecomposition`, `PrimaryDecomposition`, `IsPrime` and `IsPrimary` have all been greatly improved.
- New function `Polynomial(C, M)` to construct a polynomial from parallel sequences of coefficients (C) and monomials (M).
- New function `Degree` as synonym for `WeightedDegree` (which respects the grading).
- New function `Ideal(f)` to create the principal ideal generated by the polynomial f.
- New functions `EasyBasis` to access the easy ideal basis and `SmallBasis` to give the shortest length basis known of an ideal.
- New function `QuotientDimension` to give the (vector space) dimension of the quotient of a ring by an ideal.
- New functions `MilnorNumber` and `TjurinaNumber` to compute Milnor and Tjurina numbers for multiplicity information at the origin of multivariate polynomials.

– New function `Saturation` to return the saturation of an ideal I by a polynomial f and a minimal degree polynomial g so that dividing I by g is equivalent to saturation by f.

– New function `HilbertSeries(I, s)` to give the hilbert series to precision s. Related new functions `HilbertNumerator` and `HilbertDenominator`.

– Several low-level leaks have been fixed, particularly involving monomial orders.

14.2 Modules over Multivariate Rings

Changes and Removals:

– The earlier function `Module(R, k)` (to create a free embedded module) has been renamed to `EModule(R, k)`. `Module` has been kept in for now for backwards compatibility but will be removed in a future release.

– The earlier function `MinimalFreeResolution` will be removed in the future, since `FreeResolution` now minimizes by default and one may compute a non-minimal resolution via the new parameter `Minimize`.

– The column ordering for the `TOP` monomial order has been reversed to make it consistent with the `POT` monomial order.

New Features:

– Modules and most relevant operations are now supported over 3 commutative multivariate rings:

  – Multivariate polynomial rings.

  – Localizations of Multivariate polynomial rings.

  – Affine Algebras.

  Furthermore, modules may now be defined over exterior algebras and most standard operations (including free resolution computations) are supported for them.

– Monomial orders can now be described by tuples. The functions `Module(R, n, T)` and `Module(R, W, T)` may now take a tuple T for the order. Conversely, the new function `MonomialOrder` returns a tuple describing the monomial order of a module.

– Some new monomial orders have been added (besides the original `TOP` and `POT` orders).

– The Schreyer monomial order is now used more extensively for many internal computations, especially for computing syzygy modules and for free resolutions. This has led to general speedups for many computations with modules.

– A *module homomorphism* type (`ModMPolHom`) is now fully supported to represent homomorphisms between modules (and is used for boundary maps of free resolutions, etc.). This allows greater much power and flexibility than the old use of direct matrices.

– Computing the minimal basis of a module has been greatly improved in both speed and memory usage by a new algorithm based directly on the F_4 algorithm (instead of the former algorithm which was based on linear algebra).

– One may now have negative weights in the grading for a module. This is supported properly by all functions which use the grading (including the computation of Hilbert series).

- The computation of free resolutions has been greatly improved in general and associated functions have been added:
  - The terms of a free resolution are now graded properly if the input is graded. The associated boundary maps are now graded homomorphisms of degree 0.
  - The hybrid La Scala/F_4 algorithm now works over the rational field as well as over any finite field.
  - A new fast method has been implemented for minimizing a resolution coming out of the La Scala/F_4 algorithm.
  - One may now select the alternative resolution algorithm with iterated syzygies and minimization (which can be faster, particularly in characteristic 0).
  - For non-homogeneous modules, an option for computing a resolution of a homogenization (which is then specialized) is now available.
  - One can now efficiently access Betti number information of a module (via new functions `BettiNumbers`, `BettiNumber`, `BettiTable` and `MaximumBettiDegree`). If the module is graded, then this is also faster than computing a full resolution in general (since the minimization is not needed).
  - New function `Regularity` to compute the Castelnuovo-Mumford regularity of a module.
- The function `HilbertSeries` now handles graded modules fully and now returns a consistent value for a module, whether it is represented as an embedded or reduced module (both for sub- and quotient modules).
- New function `HilbertSeries(M, s)` to give the Hilbert series as a power series to precision s. Related new functions `HilbertNumerator` and `HilbertDenominator`.
- New functions `f*M`, `M*f`, `I*M`, `M*I` for products of modules by ring elements and ideals.
- New functions `Rank` and `IsFree` for testing fundamental properties of modules.
- New functions `ColonIdeal`, `ColonModule` and `Annihilator`.
- New functions `Grading` and `Twist` for simple operations with gradings of modules.
- New functions `FittingIdeal` and `FittingIdeals` for computing Fitting ideals of modules.
- New function `DirectSum` for computing direct sums.
- The function `Hom` to compute $\mathrm{Hom}(M, N)$ has been improved and now computes a grading on the result and the associated maps when the inputs are graded.
- New functions `Hom(C, N)`, where C is a complex, and `Ext(i, M, N)`.
- New function `TensorProduct` to compute the tensor product of two modules.
- New functions `TensorProduct(C, N)`, where C is a complex, and `Tor(i, M, N)`.
- New function `CohomologyDimension` to compute the dimension of cohomology groups $H^r(\tilde{M}(n))$ where M is a graded module over $k[x_0, \ldots, x_m]$ and $\tilde{M}$ is the associated coherent sheaf on $\mathbf{P}^m_k$. n signifies a Serre twist. The algorithm used is that of Decker, Eisenbud, Floystad and Schreyer based on the BGG correspondence and makes use of the new functionality for modules over exterior algebras.

14.3 Invariant Theory

New Features:

- Computing invariants of given degree has been dramatically sped up when the group is a matrix group of diagonal matrices.
- The computation of primary invariants has been sped up in general by using the new F_4-based Hilbert-driven Buchberger algorithm. Extra improvements has also been added in the case that there are very many invariants of small degree.
- The computation of secondary invariants in the modular case has been sped up because of general improvements in the computation of syzygy modules.
- The algorithm of Simon King for computing fundamental invariants has been implemented and is dramatically faster than the previous one in general. Fundamental invariants can now be computed (each modulo a prime coprime with the group order) for all transitive groups of degree 7 in about 1 second total, and for all transitive groups of degree 8 in about 2 minutes total.

15 Algebraic Geometry

15.1 Schemes

Removals and Changes:

- Equality for scheme maps will now expand composite maps (rather than declaring that a composite and non-composite scheme map are automatically unequal). Also scheme maps without inverses and isomorphisms are properly checked for equality rather than being declared automatically unequal as before.
- The command `RationalPoints` is no longer available as an alias for `PointSearch`, as this caused confusion. (The latter is a p-adic search method for rational points on general schemes over $\mathbf{Q}$.)

New Features:

- Function `CohomologyDimension` has been added to compute the dimensions of cohomology groups of (twisted) coherent sheaves on ordinary projective schemes represented by graded modules. The function is further described in the modules over affine algebras section 14.2.

Bug Fixes:

- A problem with maps between schemes defined over p-adic fields has been fixed.

15.2 Algebraic Curves

Changes:

- The function `Place` has been partly rewritten to speed up the computation of the place for a non-singular point on the curve defined over the basefield or a finite extension.
- `Genus` now uses `ArithmeticGenus` for curves whose projective closure is known to be non-singular. For affine or ordinary projective curves in small dimensional ambients (2 or 3) or in $\mathbf{P}^4$ and defined by ≤ 4 equations, a non-singularity check is carried out if the singularity/non-singularity is unknown. This is a fairly cheap check that greatly speeds up the genus computation for non-singular curves over function fields.

Bug Fixes:

- Curve divisors can no longer be multiplied by large integers.

15.3 Algebraic Surfaces

New Features:

- A new package by Töbias Beck and Joseph Schicho for working with algebraic hypersurfaces in 3-dimensional projective space has been added. This has three main components: formal desingularisation, computation of m-adjoint linear systems and full classification and parametrisation of rational hypersurfaces over $\mathbf{Q}$.
- Formal Desingularisation of X (main function `ResolveProjectiveSurface`) computes the power series completions of the generic rings of the components of some (not necessarily minimal) birational desingularisation of X that lie over components of the singular subscheme of X. This is based on a Jung-Hirzebruch desingularisation of X and the algorithm was designed and implemented by T. Beck in his PhD. It uses algebraic power series.
- m-adjoint sheaves on X are the pushforwards of tensor powers of the canonical sheaf from a birational desingularisation of X. The m-adjoint linear systems are the global sections of Serre twists of these sheaves. They are independent of the desinglarisation and give important birational invariants like the desingularised arithmetic genus and geometric plurigenera. They also give important maps into projective space as for the multi-canonical maps on curves.
- The main function `HomAdjoints` computes the m-adjoint linear systems as a sublinear system of all homogeneous polynomials of an appropriate degree on X. The linear conditions on the polynomials are computed from the formal desingularisation of X.
- Using the m-adjoint systems, there are now functions `ArithmeticGenusOfDesingularization`, `GeometricGenusOfDesingularization` and, more generally, `Plurigenus` for the genera and plurigenera of any desingularisation of X. There is also a function `IsRational` to determine whether X is rational using Castelnuevo's criterion on a desinglarisation. This says that X is rational iff the arithmetic genus and second plurigenus of a desingularisation are both zero.
- For a rational hypersurface X, a well-chosen m-adjoint map will take X to a special rational surface nicely embedded in projective space (usually a ruled or Del Pezzo surface). There is a function `Classify` that finds such an adjoint map, following an algorithm of Josef Schicho, and the special surface image of X in one of the families of Schicho's classification.
- Finally, there is a suite of functions to parametrise these special surfaces. These are all based on algorithms of Schicho and were implemented by him and Beck. The existing Magma functions for non-singular degree 6,8 and 9 Del Pezzo surfaces are used. The rest are new. They cover quadrics in $\mathbf{P}^3$, ruled surfaces, Del Pezzo surfaces of degree 7 and singular Del Pezzos of degrees 6 and 8 and functions to reduce low degree Del Pezzos to degree 5 or higher. The only case currently missing is for degree 5 Del Pezzos and this will be filled with code from Josef Schicho in the near future.
- There is a function `ParametrizeProjectiveHypersurface` that puts the above ingredients together to give a single parametrisation function. More generally, `ParametrizeProjectiveSurface` finds a birational projection of a projective surface to a hypersurface and then calls the first function.

16 Arithmetic Geometry

16.1 Rational Curves and Conics

Removals and Changes:

- The routine `LegendresMethod` has been removed. This is superseded by routines for solving conics that are accessed via the standard commands for conics, such as `HasRationalPoint` and `Parametrization`.

New Features:

- `HasRationalPoint`, `RationalPoint` and `Parametrization` are now implemented for general conics over (absolute) number fields. These make use of the following two items.
- The routine for solving diagonal conics over number fields has been completely rewritten. The new implementation is fast, and also far more powerful than the previous one (that is, conics can now be solved over fields with much larger discriminant than before). The algorithm is a variant of Lagrange's method, and involves a series of reduction steps in which the conic is replaced by a simpler conic. Reduction is achieved by finding a short vector in a suitable lattice sitting inside two copies of the base field. After several reductions, one is often able to find a solution with an easy search. In other cases, one is unable to reduce further and must call `NormEquation` to solve the reduced conic. (This is still vastly superior to calling `NormEquation` on the original conic.) The basic reduction loop is enhanced with several tricks; in particular the class group of the base field may be used, when it is not too large, to reduce much further than would otherwise be feasible. This code is accessed by `HasRationalPoint`.
- Diagonalization of conics over number fields is now done by special purpose code, which ensures that the coefficients of the diagonal are not hard to factor if one is able to factor the discriminant of the original conic. This code is accessed by `DiagonalForm`.

16.2 Elliptic Curves

16.2.1 Elliptic Curves over the Rational Field

Removals and Changes:

- In `IntegralPoints` and `SIntegralPoints`, the optional arguments `Fast` and `Safe` have been removed.

Bug Fixes:

- Some bugs have been fixed in `SIntegralPoints`.
- The ability to `RemoveTorsion` and remove generators for `FourDescent` has been fixed for curves with $j = 0, 1728$.

New Features:

– A new fast implementation of 2-descent is used in `TwoSelmerGroup` and `TwoDescent` (for curves over $\mathbf{Q}$). The new code avoids much of the cumbersome machinery previously used. This is now suitable for large scale experiments (as it is fast enough, and moreover performance does not suffer much during long runs). For example it is possible to run through the Cremona database in a day or so. Furthermore, it has been checked (up to parity, using the root number) for all such curves. The calling structure has been largely preserved. An additional function `TwoCover` has been added, which simplifies some of the work previously performable via `CasselsMap`; it simply takes a nonzero element of a cubic algebra over $\mathbf{Q}$ and returns a corresponding hyper-elliptic curve.

– A new fast implementation of `CasselsTatePairing` is provided for curves over $\mathbf{Q}$. A single call typically takes less than 0.05 seconds (with the dependence on the conductor of the curve being very mild).

– The Cassels-Tate pairing between an element of order 4 and an element of order 2 in the Tate-Shafarevich group of an elliptic curve E over $\mathbf{Q}$ is also implemented. The input elements are given as curves C and D, where C is a 4-covering over E given as a quadric intersecion, and D is a 2-covering of E (as returned by `FourDescent` and `TwoDescent` respectively).

16.2.2 Elliptic Curves over Number Fields

New Features:

– `CasselsTatePairing` is implemented for curves over number fields. This computes the Cassels-Tate pairing on the 2-torsion subgroup of the Tate-Shafarevich group. The input consists of two hyperelliptic curves, as returned by `TwoDescent`.

16.2.3 Elliptic Curves over Finite Fields

New Features:

– We have added new quasi-quadratic point-counting code from Hendrik Hubrechts for small characteristic p elliptic curves. The code uses p-adic deformation methods and is a highly-tuned adaptation of his earlier code for general hyperelliptic curves. The algorithm is used in place of canonical lift or small characteristic SEA for p roughly between 40 and 1000 and for extensions of the prime field of degree larger than a small number depending on p.

– Point-counting over prime subfields smaller than 2^{29} has been made faster by a factor of almost 2 on many processors, taking advantage of 64-bit arithmetic. This can be quite useful when calculating with L-functions of elliptic curves.

16.2.4 Elliptic Curves over Function Fields

New Features:

– `TwoSelmerGroup` (and `TwoDescent`) are now much faster. One improvement is to skip working out the local conditions at places of "not so bad reduction". Also as noted above, large elements in product representation are now handled better (such elements arise in the S-unit calculations).

– `CasselsTatePairing` is implemented for curves over rational function fields $F(t)$, where F is a finite field of odd characteristic. This computes the Cassels-Tate pairing on the 2-torsion subgroup of the Tate-Shafarevich group. The input consists of two hyperelliptic curves, as returned by `TwoDescent`.

16.3 Genus One Models

New Features:

- New routines for minimisation and reduction of genus one curves of degrees $2, 3, 4$ and 5 are included. Some of these cases were not available before, and all are new implementations except for reduction in degrees 2 and 3, which use existing Magma routines (`QuarticReduce` for degree 2).

16.4 Hyperelliptic Curves

16.4.1 Hyperelliptic Curves over Finite Fields

New Features:

- We have added a small package of functions for genus 2 curves written by Reynald Lercier and Christophe Ritzenthaler and released under a GPL license. They are based on a new set of absolute invariants that the authors refer to as Cardona-Quer-Nart-Pujola invariants. The advantage of this package is that it works in all characteristics.
- There are functions `G2Invariants`, `G2ToIgusaInvariants`, and `IgusaToG2Invariants` to compute the invariants of a genus 2 curve and convert between these and the Igusa J-invariants. `HyperellipticCurveFromG2Invariants` finds a curve with given invariants.
- `Twists` gives all (not just the quadratic) twists of a genus 2 curve over a finite field (of any characteristic). The curve or its invariants can be given as the argument.
- The old `GeometricAutomorphismGroup` has been replaced with a Lercier/Ritzenthaler version that works in all characteristics.
- `GeometricAutomorphismGroupClassification` in their package computes all possible geometric automorphism groups for genus 2 curves over a given finite field (of any characteristic) and also the number of isomorphism (over the algebraic closure) classes of curves over the field with each group of automorphisms.

16.4.2 Hyperelliptic Curves over the Rationals

New Features:

- For a genus 2 curve C, a new implementation of Chabauty's method combined with the "Mordell-Weil" sieve is provided, which returns the full set of rational points on C. This is called by `Chabauty(ptJ)` where `ptJ` is a point on the Jacobian of the curve. This point is assumed to generate the Mordell-Weil group (modulo torsion) of the Jacobian, which must have rank 1 in order to apply Chabauty's method. The algorithm also requires one rational point on C (so does not apply for curves with no points); if such a point is known, it may be passed in as the optional argument `ptC`.

16.5 L-Functions

New Features:

- The L-series attached to an Artin representations can now be created and efficiently computed. (Previously, only the special case of the ζ-function of a number field was implemented.)
- A new option is available in `LSeries` of a number field K. If the parameter `Method` is set to "Artin", this creates ζ_K represented internally as a product of L-series attached to the irreducible representations of the Galois group of K. This allows for more efficient computation in many cases (although clearly, this depends on the size of the Galois group). When K is abelian, this is the default method.
- The twisted L-series of an elliptic curve E twisted by an Artin representation A can be constructed using `LSeries(E,A)`.
- More generally, twists of arbitrary L-series can be constructed using `TensorProduct`.

17 Modular Arithmetic Geometry

17.1 Hilbert Modular Forms (New)

Features:

- This package computes spaces of cuspidal Hilbert modular forms over an arbitrary totally real number field F, of any weight whose components are at least 2, on congruence subgroups $\Gamma_0(N)$ where N is any ideal in the ring of integers of F. These spaces are created using `HilbertCuspForms`.
- Computations are done, via the Jacquet-Langlands correspondence, with the aid a suitable order in some quaternion algebra; this is chosen internally and does not feature in the interface. However, when creating the space, the user may specify the `QuaternionOrder` to be used.
- A well-optimized implementation of Dembélé's algorithm is used for the computations. This relies in particular on fast routines for certain computations involving ideal classes in quaternion orders. A major advantage of this algorithm is that these computations (which, in principle, are the most expensive part of the process) need to be done only once for each base field, and the results can then be used to compute spaces of (almost) all levels and weights over that field.
- A separate algorithm due to Greenberg and Voight is also implemented for spaces of parallel weight 2. This is used if the user specifies a `QuaternionOrder` that is unramified at one infinite place.
- The dimension of a space may be computed (which is a nontrivial computation, as dimension formulae are not available in general).
- Hecke operators can be computed, with respect to some fixed basis of the space. (Currently there are no conventions regarding the choice of basis.)
- The subspace of new forms, or more generally the subspace new at a given divisor of the level N, can be created using `NewSubspace`. Currently this is restricted to cases where the "new level" is squarefree.
- The `NewformDecomposition` of a new space is available, and systems of eigenvalues can be obtained for the newform corresponding to each Hecke-irreducible component in the decomposition.

17.2 Arithmetic Fuchsian Groups and Shimura Curves

Bug fixes:

- Some improvements have been made to the effectiveness of the `FundamentalDomain` routine for Fuchsian groups.

18 Geometry

18.1 Incidence Geometry

New Features:

- Intrinsics have been added by D. Leemans for testing whether a coset geometry is primitive (`IsPrimitive`), residually primitive (`IsRresiduallyPrimitive`), weakly primitive (`IsWeaklyPrimitive`) and (`IsRresiduallyWeaklyPrimitive`).
- The function `IsLocallyTwoTransitive` tests whether a coset geometry is locally two-transitive, i.e., whether all of its minimal parabolic subgroups have a two-transitive action on the cosets of the Borel subgroup.

19 Incidence Structures

19.1 Designs

New Features:

- The function `Design(I, t, lambda)` has been added to allow the construction of a design from an incidence structure without checking the design parameters.
- When constructing an incidence structure, the incidence matrix may now be given either as a sparse matrix or dense matrix.

20 Coding Theory

20.1 Linear Codes over Finite Fields

New Features:

- An $[n, k]$ linear code C is said to be a best known linear $[n, k]$ code (BKLC) if C has the highest minimum weight among all known $[n, k]$ linear codes. A number of new codes have been added to the BKLC database constructed and maintained by Markus Grassl. Specifically, 17 codes over $GF(3)$, 15 codes over $GF(4)$, and 5 codes over $GF(7)$ have been added.

20.2 Quantum Stabilizer Codes

New Features:

- An $[[n, k]]$ quantum stabiliser code Q is said to be a best known $[[n, k]]$ quantum code (BKQC) if Q has the highest minimum weight among all known $[[n, k]]$ quantum codes. The BKQC database constructed and maintained by Markus Grassl has been expanded as follows: The maximal length of codes has been increased from 35 to 50 while the minimum distance of 10 codes has been improved.