Summary of New Features in Magma V2.17

December 2010

1 Introduction

This document provides a terse summary of the new features installed in Magma for release in version V2.17 (December 2010). A small number of new features were exported in patch releases subsequent to the main release of V2.16 in November 2009 and these are also listed here for completeness. Only significant bugfixes are noted here – for a more complete list of bugfixes the reader should consult the patch release change log for V2.16-x.

Previous releases of Magma were: V2.16 (November 2009), V2.15 (December 2008), 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).

Highlights

Algebraic Geometry

- *Schemes*
  - Machinery has been provided for computing the sheaf of differentials and the tangent bundle sheaf for an ordinary projective scheme.
  - The ability to determine the multiplicity for singular points has been extended from plane curves to general schemes. Computing the tangent line has been extended from plane curves to general curves.
- *Algebraic Surfaces*
  - Code has been developed for the parameterisation of singular Del Pezzo surfaces of degrees 3 and 4 by Michael Harrison and Josef Schicho. The non-singular case was made available in a previous release so that all degree 3 and 4 Del Pezzo surfaces are now handled.
- *Toric Geometry*
  - The toric geometry machinery is more tightly coupled with the existing schemes code. For example, the scheme returned by `ProjectiveSpace` can be passed directly to "toric" intrinsics such as `Fan`.
  - As part of this tighter integration, `Dimension`, `IsProjective`, `IsAffine`, and `EmptySubscheme` now work for a larger collection of subschemes, such as the case when the ambient toric variety is non-$\mathbf{Q}$-factorial.
  - The scrolls type `PrjScrl` has been removed. The constructors `RuledSurface` and `RationalScroll` now return type `TorVar`. This allows a greater range of computations to be performed using these surfaces.
  - Basic quotient singularity analysis is now supported and checking for singularities has been made significantly faster. In the case when the variety is a (fake) weighted projective space it is often sufficient to consider the Cox data, and the fan need not be constructed.
  - A new deterministic algorithm for the resolution of singularities which delivers greatly improved performance has been developed. Similarly, deterministic algorithms for computing a $\mathbf{Q}$-factorial refinement, a $\mathbf{Q}$-factorial terminal refinement, and a $\mathbf{Q}$-factorial canonical refinement of a fan have been added.
  - Databases of the classification of smooth toric Fano varieties up to dimension 8, and of the canonical Fano varieties in dimension 3, are provided.
  - *See also Convex Polytopes and Polyhedra.*

Arithmetic Fields

- *Global Fields*
  - The fast algorithm used to compute the maximal order of a Kummer extension has been extended to produce the maximal order of a radical extension. It is much faster than the Zassenhaus Round 2–4 algorithms.
  - It is now possible to calculate the subfields of any global field that is defined by a single polynomial (algorithm of Klueners and van Hoeij).
- *Galois Theory*
  - Galois groups are now supported for reducible polynomials over number fields (extending the case for polynomials over Z)
  - Galois groups are now supported for relative function fields in characteristic p.

Arithmetic Geometry

- *Conics*
  - The machinery for solving conics over number fields has been extensively rewritten resulting in much improved performance especially for conics over larger degree fields.
  - Other important infrastructure, such as Simon-minimization, has been added.
- *Elliptic Curves Over Finite Fields*
  - The code for computing discrete logarithms for elliptic curves over finite fields has been extensively revised leading to substantial speedups.
- *Elliptic Curves Over* $\mathbf{Q}$
  - Code developed by Tom Fisher for performing 8-descent, 6-descent and 12-descent on an elliptic curve has been included.
  - Improvements have been made to 2-descent, in particular large power products are now successfully handled.
  - The routine for "saturating" Mordell-Weil groups has been revised, resulting in speedups by a large factor especially for curves of high rank. (Improved discrete logs contributed part of the speedup).
  - The computation of p-adic L-series in the case of good ordinary reduction has been added. This can also be applied to quadratic twists.

- *Elliptic Curves Over Number Fields*
  - Periods of elliptic curves over a number field can now be calculated.
  - Analytic information related to the Birch–Swinnerton-Dyer conjecture for elliptic curves over number fields may now be computed: `ConjecturalRegulator`, `ConjecturalSha` and `AnalyticRank`.
- *Models of genus one curves*
  - Systematic code for testing local solubility of models of degree $2, 3$ and 4 is included.
  - A general purpose `Minimisation` routine for genus one normal curves of higher degree is included.
  - A careful `Minimisation` routine for degree two models over number fields is included.
- *Hyperelliptic Curves*
  - The 2-Selmer groups of hyperelliptic Jacobians can now be computed in more cases; in particular, for all genus 2 curves over number fields.
  - It is now possible to construct L-series for general hyperelliptic curves defined over the rationals. This relies on the new machinery for regular models. Magma is the only computer algebra system that supports the computation of L-series for curves in this generality.
  - Given the Jacobian of a genus 2 curve, code has been developed by Nils Bruin to compute all (2,2)-isogenous abelian surfaces.
- *Regular Models*
  - Computation of regular models of (arithmetic surfaces associated to) curves over $\mathbf{Q}$ or number fields has undergone continued development. Plane curves in (weighted) projective space, in particular hyperelliptic curves, are now handled. The main routine has been made more effective, and some new features of regular models are provided.

Modular Arithmetic Geometry

- *Modular Forms and Modular Symbols*
  - Major speedups have been achieved in key routines such as `NewSubspace` and `NewformDecomposition`, especially for spaces of large dimension.
- *Hilbert Modular Forms*
  - The implementation of Dembélé's algorithm (the "definite method") has been massively improved in a variety of ways.
  - The case of parallel weight 2 now has separate code.
  - Both phases of the "precomputation" are now considerably faster.
  - Hecke eigenvalues can be obtained without computing full Hecke operators.
  - Much better linear algebra techniques are used in routines such as `NewformDecomposition`, making it feasible to handle spaces of large dimension.
  - A Hecke-invariant inner product is available in some cases.

Associative Algebras

- *Quaternion Algebras*
  - Computation of right ideal classes and order classes (for algebras over number fields) has been improved using new algorithmic ideas, resulting in massive speedups.
  - Quaternion algebras over function fields $F_q[t]$, with q odd, have been implemented for the first time by Markus Kirschmer. The standard arithmetic operations are supported (maximal orders, ideal classes, and so on).
- *Basic Algebras*
  - V2.17 includes machinery for constructing the basic algebra of a modular group algebra. The method has been successfully applied to groups having order up to several million.
  - As a variant on the above feature, it is now possible to compute the basic algebra associated with any p-block of a modular group algebra.
  - The algorithm to construct a basic algebra from a matrix algebra now detects a diagonal block structure in the input, and exploits that structure to greatly reduce the cost of matrix multiplication.

Combinatorial Theory

- *Graph Theory*
  - The code for graphs with edge labels has been substantially revised to properly handle the labels in all circumstances.
- *Hadamard Matrices*
  - The database of Hadamard matrices has been expanded. New matrices have been added for degrees 36 (1), 48 (60) and 60 (1759). The additional matrices were supplied by Dragomir Djokovic.

Convex Polytopes and Polyhedra

- *Cones, Polytopes, and Polyhedra*
  - A vast number of improvements have been made throughout the package for polytopes and polyhedra after the initial release of the package in V2.16. The functionality has been expanded and major improvements have been made to the efficiency of many key algorithms.
  - Testing for isomorphism and equivalence of pairs of polytopes is now supported.
  - The automorphism group of a polyhedron may now be computed.
  - The f-vector of any polyhedron, along with the set of all i-faces and the face graph may now be computed.
  - The introduction of an improved algorithm has led to a big reduction in the time taken to compute the Hilbert basis of a cone.
  - Possible Minkowski decompositions of a lattice polygon may now be calculated.
  - Databases containing the classification of smooth Fano polytopes up to dimension 8, and canonical Fano polytopes in dimension 3, are included.

Group Theory

- *Matrix Groups (Over Finite Fields)*
  - The Composition Tree (CT) package developed by Henrik Bäärnhielm, Derek Holt, Charles Leedham-Green and Eamonn O'Brien, working with numerous collaborators, is being released for the first time. This package is designed for computing structural information for large degree matrix groups potentially having very large orders. The traditional approach involves constructing a base and strong generating set where the orbits of the base points have to have length at most a million and so is limited to small groups. The CT package uses a constructive version of Aschbacher's Theorem to construct a composition series for the group represented in the form of a tree.

  - Code built on top of the CT package allows the user to determine basic properties of the group such as the derived subgroup, chief series, and Sylow subgroup.

- *Matrix Groups (Over Rings of Characteristic 0)*
  - Code has been developed to compute the normaliser or centraliser of a finite subgroup of $GL(n, \mathbf{Z})$.
  - Machinery is provided which enumerates the isomorphism classes of $\mathbf{Z}G$-lattices, in the case in which the endomorphism ring of the group G is a number field.
  - A new package, "Infinite", has been developed by Alla Detinko, Dane Flannery and Eamonn O'Brien for groups defined over number fields, or (rational) function fields in zero or positive characteristic. This package contains algorithms to decide finiteness; if the group is found to be finite, an isomorphic matrix group defined over a finite field is returned. It also contains algorithms to decide nilpotency, solubility, the Tits Alternative and other "virtual" properties for such groups.
  - It is now possible to test irreducibility and primitivity of large finite nilpotent matrix groups defined over a number field or a rational function field. The method does not use either the BSGS or CT representations and is due to Tobias Rossmann.

- *Permutation Groups*
  - An implementation of the "Jellyfish" algorithm of Law, Niemeyer, Praeger, and Seress was written by Bill Unger. This will form part of a composition tree approach to permutation groups in Magma. The Magma implementation extends the original algorithm by providing a tolerably efficient inverse image facility.

- *Classical Groups*
  - Quadratic spaces have been implemented.
  - Clifford algebras have been implemented as structure constant algebras, using the above quadratic space machinery. An immediate application will be to the construction of the spinor group associated with certain orthogonal groups.
  - Witt's theorem has been implemented for orthogonal and symplectic geometries over finite fields, including fields of characteristic two. That is, an isometry defined on a subspace can be extended to an isometry of the entire space.
  - A standard generating set can be constructed for any classical group given in its natural representation.
  - A presentation for a classical group can be constructed on its standard generators.

Lattices and Quadratic Forms

- *Lattices*
  - A new version of the backtrack search algorithm for computing the automorphism group of an integral lattice has been developed. This can handle lattices having much larger sets of vectors of minimal norm than the Souvignier version which has been widely used up until the present.
  - Functionality is provided to compute the Hermitian and quaternionic automorphism groups of lattices when applicable (though the functions themselves work on the Gram matrix of such lattices).
- *Quadratic Forms*
  - Code has also been developed for computing automorphism groups (testing isometry) of definite quadratic/bilinear forms over $F_q[t]$, with q odd.

L-functions

- Recent work by Tim and Vladmir Dokchitser enables Magma to work with Artin representations for much larger groups. It is now possible to apply the machinery to groups of order 50000.
- The above improvement makes computations with Dedekind ζ-functions much more efficient than was previously the case.
- Improvements have been made to the functions for working with L-functions of twists of elliptic curves by Artin representations. This makes the calculation of analytic rank and related objects much more efficient.
- The p-adic L-function of an elliptic curve at a given prime can now be computed.

Linear Algebra

- *Matrices over Algebras*
  - Basic operations are now supported for matrices defined over a quaternion algebra.
- *Sparse Matrices*
  - The operations on sparse matrices have been greatly expanded to include many more of the standard operations provided for dense matrices.

Lie Theory

- *Reflection Groups*
  - The package for complex reflection groups has been rewritten to ensure that the matrix entries belong to the ring of definition.
  - A complex root datum has been implemented for every finite complex reflection group. This is a generalisation of the "Demazure" root datum.
- *Coxeter Groups*
  - A highly efficient algorithm for computing the growth function of a Coxeter group was developed and implemented by Bob Howlett and Bill Unger.
- *Lie Algebras*
  - Code has been developed to compute a Chevalley basis for a Lie algebra in characteristics 2 and 3. Taken together with previous work, the construction of a Chevalley basis is now available in every characteristic.
  - An experimental algorithm for calculating split toral subalgebras in characteristics 2 and 3 has been released.
  - A construction for Melikyan Lie algebras has been implemented.

Numerical Analysis

- Eigenvalues of a matrix may now be computed to a fixed precision using a stable numerical algorithm.
- Numerical approximations of eigenvectors of a given approximate eigenvalue can be computed in cases that are not ill-conditioned
- The numerical derivative of a function can be computed in cases where this is numerically workable.

Representation Theory

- A package has been developed which constructs the principal indecomposable modules for a group algebra over a given finite field.
- The Meataxe-based algorithms now detect any diagonal block structure in the input and use the corresponding decomposition to speed up matrix multiplication.

System

- *Runtime System*
  - The memory manager for Magma has been upgraded significantly by the introduction of a new Mark-Sweep garbage collection scheme. This leads in general to less memory usage for many typical computations, especially those involving long loops over databases. In particular:
    - Leaks of objects arising from circular references from attributes defined in package code have been removed.
    - Leaks of objects arising from internal circular references have been removed.
    - Slowdowns involving algebraic number fields (caused by complex circular reference analysis) have been removed.
  - A large number of leaks in internal algorithms have also been fixed (independent of the above garbage collection scheme).
- *Language*
  - The extended types machinery now supports arbitrary levels of nesting for sets, indexed sets, and multisets. (Formerly only sequences would nest.)

2 Language and System Features

New Features:

- The memory manager for Magma has been upgraded significantly by the introduction of a new Mark-Sweep garbage collection scheme. This leads in general to less memory usage for many typical computations, especially those involving long loops over databases. In particular:
  - Leaks of objects arising from circular references from attributes defined in package code have been removed.
  - Leaks of objects arising from internal circular references have been removed.
  - Slowdowns involving algebraic number fields (caused by complex circular reference analysis) have been removed.
- A large number of leaks in internal algorithms have also been fixed (independent of the above garbage collection scheme).
- The extended typing feature now enables arbitrary levels of nesting for sets, indexed sets, and multisets. (Formerly only sequences would nest.)
- There is now a level for assertions and besides the existing `assert E` (level 1) for some boolean expression E, there are new statements `assert2 E` (level 2) and `assert3 E` (level 3). For these statements, the expression E is checked to be true iff the current assertion level is at least the level corresponding to the statement. It is thus recommended that when developing package code, `assert` is used for important tests (always to be tested in any mode), while `assert2` is used for more expensive tests, only to be checked in a debug mode (`assert3` can be used for extremely stringent tests which are very expensive). Finally, `SetAssertions` now takes an integer and `GetAssertions` returns an integer.
- Strings may now have lengths greater than 2^{32} on 64-bit platforms (and the `Read` and `ReadTest` functions allow files greater than 4GB).
- The intrinsic `ISA` now allows a set of categories as the second argument, simplifying argument checking code in intrinsics.
- `AddAttribute` now raises an error when called on a type which may not have attributes added.

Bug Fixes:

- If intrinsic `ListSignatures` is called with type `Any`, the the value of the `Isa` parameter is now properly respected.
- A rare crash when compiling packages that used `eval` has been fixed.
- An occasional crash with `require` statements in package files has been fixed.
- An occasional crash which occurred when deleting an associative array over some universes has been fixed.

3 Aggregates and Mappings

New Features:

- One may now obtain a sublist of a list L via $L[S]$ where S is a sequence of indices.
- The operators `diff`, `diff:=`, `sdiff`, and `sdiff:=` have been extended to work for multisets.

Bug Fixes:

- A bug which arose when printing multisets whose elements were sequences with no sorting order defined has been fixed.
- A crash when calling `IsDisjoint` on multisets has been fixed.
- When mapping a sequence which is not coercible into the domain of the mapping, a sequence containing the images of the elements of the input sequence will only be returned if all those images lie in the same universe.

4 Algebraic Geometry

4.1 Schemes

New Features:

- Multiplicities for singular points on arbitrary schemes over a field are now available. The computation uses local Groebner bases.
- Tangent Cones are now computed for general schemes (the old version was really only guaranteed correct for hypersurfaces). This again makes use of the new local Groebner basis functionality.

Changes and Removals:

- Affine patches of subschemes of length 1 schemes are not available (as has always been with subschemes of length 0 schemes).
- The `PointSearch` function had a few bug fixes and some improvements.

4.2 Sheaves

New Features:

- Functions `SheafOfDifferentials` and `TangentSheaf` have been added to compute the sheaf of 1-differentials and the tangent sheaf of ordinary projective varieties. Along with other applications (computation of the dimension of the connected component of automorphism groups or the dimension of deformation spaces), combined with the `IsLocallyFree` intrinsic, this gives an alternative method to test for non-singularity that can be superior to the default Jacobian method for low dimensional varieties in high dimensional projective spaces.

4.3 Algebraic Surfaces

New Features:

- Functionality has been added to handle the parametrization of degree 3 and 4 Del Pezzo surfaces in a more efficient manner. This also fixes some incorrect processing in the old version that resulted in some singular parametrizable surfaces being labelled as non-parametrizable. The new methods work more directly and generally avoid blowing down exceptional curves to give higher degree surfaces. This is incorporated in the general Del Pezzo parametrization routine, but can also be accessed through intrinsics `ParametrizeSingularDegree3(4)DelPezzo`.

4.4 Algebraic Curves

New Features:

- `MatrixRepresentation` is a new function that gives the representation of a curve automorphism group on the space of global differentials when the genus is at least 2.

– Maps on curves: a scheme map from a curve with function field into a general scheme will now automatically try to compute the image of points in the base scheme of the map using the function field place machinery. Basically, if the map on the projective normalisation of C is defined at all points over the given point p (always true if the codomain is projective) and these points have the same image, which is also defined over the base field of the point set of p, then that will be returned as the image of p. This will give the correct result at all points to which the map can be extended, but avoids the expensive global `Extend` function.

– Tangent lines and cones, multiplicities of singular points and most of the other local singularity functions have been extended from plane curves to general curves.

– The intrinsic `IsHypersurfaceDivisor` returns whether an effective divisor on an ordinary projective curve is the scheme-theoretic intersection of the curve with a hypersurface. If this is the case, such a hypersurface is also returned.

4.5 Toric Varieties

New Features:

– The existing subscheme intrinsics `Dimension`, `IsProjective`, `IsAffine`, and `EmptySubscheme`, along with all intrinsics which build on them, give correct answers for a wider class of ambient spaces via toric computations. Although slower, this allows the user to work with, for example, subschemes in non-$\mathbf{Q}$-factorial toric varieties.

– A well-known limitation with the Magma schemes is the strict definition of affine patch which makes working with graded spaces such as $\mathbf{P}(2,3,5,7)$ problematic. In the toric case, several new intrinsics are provided to help. The intrinsic `ToricAffinePatch` will return the affine patch realised as a toric variety, along with the embedding. Similarly, intrinsics `ToricFunctionField` and `ToricLiftRationalFunction` do not require the traditional affine patch in order to perform computations.

– The `BigTorus` of a toric variety X can be recovered as a toric variety, along with its embedding into X and the rational map from X to the torus. The map of the big tori of the domain and codomain of a toric map can be recovered via `UnderlyingToriMap`.

– The intrinsic `RestrictionToSubtorus(X)` returns the restriction of a subscheme Z to the largest subtorus of the ambient containing Z.

– A toric map can be tested for being the identity map via `IsIdentity`.

– The toric ideal generated by a cone C can be created using the `Ideal` intrinsic.

– The non-$\mathbf{Q}$-factorial locus of a toric variety can be obtained from `NonQFactorialLocus`.

– The computation of the (geometric) genus of a curve realised as a subscheme in a toric ambient space can be computed using the toric (rather than scheme) methods via `GeometricGenusUsingToricGeometry`.

Changes and Removals:

– Intrinsics `IsTerminal`, `IsCanonical`, and `IsGorenstein` are now substantially faster in the (fake) weighted projective space case, through use of the inequalities of `arXiv:0805.1008` to minimise the amount of singularity analysis.

– An improvement to the convex hull algorithms means that computing resolutions of singular toric varieties is much faster. The algorithm is deterministic by default, with an optional parameter to make use of randomisation.

- The intrinsics `IsWeightedProjectiveSpace` and `IsFakeWeightedProjectiveSpace` no longer compute the fan; the result is read from the Cox data.
- The intrinsic `DirectProduct` now avoids creating the fan. If the fan is required later, it will be constructed using the fans of the product varieties. Because of the closer integration with the toric calculations it is possible to compute the direct product of a much wider class of ambient schemes (where the scheme is in fact a toric variety).
- The parameters `relevant` and `homogeneous` have been added to map constructors.

5 Arithmetic Geometry

5.1 Elliptic Curves

5.1.1 General Elliptic Curves

Bug Fixes:

- The application of elliptic curve maps to ring elements (specifically 0) will treat the input argument as an element of the coordinate ring of the ambient of the domain if possible.
- Calling `LocalInformation` on an elliptic curve defined over a function field no longer alters the print name of the function field.
- Fixed a leak when computing `IsomorphismData`.

5.1.2 Elliptic Curves over the Rational Field

New Features:

- `SixDescent` and `TwelveDescent` are included (code by Tom Fisher). The point is to combine 3-descent with 2- or 4-descent, obtaining genus one normal curves of degree 6 or 12 on which it is easier to find rational points. (On the level of Selmer groups and rank, no additional information is obtained.) The functions take a 3-covering and a 2- or 4- covering, and return the possible combination coverings. The returned coverings are minimised and reduced.
- A new `EightDescent` function is included (code by Tom Fisher). This partly incorporates and partly replaces the previous implementation. The 8-coverings are now returned as genus one normal curves in P^8, and they arise naturally in this form in the new version of the algorithm. The returned coverings are minimised and reduced.
- The p-adic L-series at a good ordinary prime, or one of multiplicative reduction, can now be computed. This can also be done efficiently in quadratic twist families.
- The ability to translate by 2-torsion with Heegner descent on covers has been added.
- The intrinsic `Saturation` has a new option `TorsionFree`, which signals that generators of the torsion subgroup are not top be included in the output.
- The intrinsic `(S)IntegralPoints` has a new optional argument `SafetyFactor`, providing a convenient safety checking mechanism.

Changes and Removals:

- The normalisation of the `LocalHeight` of a point has been changed in some cases.
- Twisted L-functions have been improved.
- Curves extracted from the Cremona database now have their group information (rank and generators of the free part of the Mordell-Weil group) assigned.
- The intrinsic `CremonaReference` has been sped up, and curves now remember their reference information.
- The all-purpose routine `MordellWeilShaInformation` has been made more effective.

Bug fixes:

- Some memory leaks in the rank computation were fixed.
- Some minor errors in `HeegnerPoint` were fixed.
- Fixed a possible crash in the computation of the `TamagawaNumber` in the `I*n` case when the elliptic curve had large coefficients.
- The manner in which 2-covers are added has been corrected.
- A bug when quotienting out by elements in 4-descent was patched.
- The condition of "nonzero element" in `TwoCover` has been changed to require nonzero norm instead.
- A minor printing problem with variables in `TwoDescent` was fixed.
- A problem with local completions in `TwoDescent` was fixed.
- A problem with domain in maps for `TwoDescent` was fixed.
- A problem with re-using a polynomial in local solubility with 2-descent has been fixed.
- A minor bug with Hints in 4-descent has been fixed.
- `PointsQI` now tests local points for global solubility in all cases.

5.1.3 Elliptic Curves over Number Fields

New Features:

- The ability to compute the periods corresponding to the embeddings of the number field into the complexes has been added.
- Functionality for computing objects related to the Birch–Swinnerton-Dyer conjecture has been added.

Changes and Removals:

- `PseudoMordellWeilGroup` should not be used for curves over **Q** (this will be disabled). Instead `MordellWeilShaInformation` should be used.

Bugs:

- Bugs in `Chabauty` for elliptic curves have been corrected.

5.1.4 Elliptic Curves over Finite Fields

New Features:

- Addition of `SupersingularPolynomial` which returns the separable, monic polynomial over F_p whose roots are precisely the j-invariants of supersingular elliptic curves in characteristic p. This is computed by a simple formula coming from a truncated power-series expansion reduced mod p of certain hypergeometric functions.

Changes and Removals:

- The code for computing discrete logarithms for elliptic curves over finite fields has been extensively revised leading to substantial speedups.
- A bug that could occasionally cause the discrete logarithm to incorrectly report failure has been fixed.
- A bug in the simplified model computation in characteristic 2 was fixed.
- A bug in the `TatePairing` when one of the arguments was 0 was fixed.
- A crash in `EtaqPairing` was fixed.

Bug Fixes:

- Fixed a crash in `EtaqPairing`.

5.2 Genus One Models

New Features:

- Local solubility testing: `IsLocallySolvable` is implemented using efficient special-purpose code for models of degree $2, 3$ and 4. If the option `Random` is set to true, the local points found are randomised, as needed in descent-related routines.
- For genus one curves of degree higher than 4, `MinimizationMatrix` finds a change of coordinates to a minimal or near-minimal model. (Not guaranteed to work in all situations.) This was developed for use in the 6-, 8- and 12-descent code.
- Models of degrees $2, 3$ and 4 can be ‘added” (in the sense of addition in the Selmer group) using `AddModels`.
- Intrinsics `SL4Covariants`, `RemoveCrossTerms`, `AddCrossTerms`, `TransformationtoMap`, `CoveringMap` are included.

Changes:

- Transformations between genus one models now have their own type, `TransG1`. All relevent intrinsics now accept/return this type.
- `CompleteTheSquare` has changed slightly, and returns a transformation.

5.3 Hyperelliptic Curves

New Features:

- L-series can now be (automatically) computed for most hyperelliptic curves.

Bug fixes:

- A bug in constructing a map from a hyperelliptic curve to its ambient has been fixed.

5.4 L-Series

New Features:

- The L-series of a Hilbert modular form can now be constructed.
- Automatic construction of the L-series of a hyperelliptic curve defined over $\mathbf{Q}$ is now possible. This uses the regular model machinery, and works in considerable generality.

Bug fixes:

- Some fixes and speedups for symmetric powers particularly in the case of modular forms have been made.
- Faster code from Anton Mellit has been added for symmetric powers and tensor products.
- A typo with `EulerFactor` for a modular form has been corrected.
- A bug with `eq` that could cause an infinite loop was fixed.

6 Arithmetic Geometry (Modular Forms)

6.1 Modular Forms

Changes and Removals:

- Functions such as `DeleteAllAssociatedData` and `DisownChildren` for spaces of modular forms and modular symbols are deprecated. They are no longer needed, since effective garbage collection is now provided by the system which, in particular, handles circular references which previously prevented such objects from being deleted.

6.2 Modular Symbols

Changes:

- The intrinsic `NewSubspace` has been improved through the use of better linear algebra techniques.
- A lot of unnecessary work has been avoided in `NewformDecomposition`.
- The routine `ModularSymbols(E)` for an elliptic curve E has also been improved through the use of better linear algebra techniques.

Bug fixes:

- A bug in `FullPrimaryComponents` has been fixed, where an infinite loop occurred due to a failure to update the random seed.

6.3 Hilbert Modular Forms

New Features:

- Access intrinsics `CentralCharacter` and `IsParallelWeight` have been added.
- A Hecke-invariant `InnerProduct` is implemented for some spaces.
- A `HeckeEigenvalueBound` is implemented for cuspidal newforms.
- The `LSeries` of a cuspidal newform is implemented.
- New or improved algorithms have been implemented in the following areas. Improvements in the precomputation for the “definite” method are indicated above. The non-precomputation part of it has been implemented more efficiently; in particular, parallel weight 2 is handled by separate code and is reasonably efficient. The inner product is used in several places to save work; Hecke eigenvalues can be computed without computing all columns of the corresponding Hecke operator on the full space. Better techniques for dealing with abstract Hecke modules over number fields have been implemented; work on this is ongoing.
- An option is provided to suppress taking the quotient by the Eisenstein series (which arise naturally in the “definite” algorithm for weight 2). It is invoked by setting `RemoveEisenstein:=false` when creating the space. This makes computations faster for large spaces.

– `DeleteHeckePrecomputation` is provided, to allow the user to conserve memory when necessary by deleting bulky "precomputed" data after it has been used. (This data is used to compute Hecke operators at given primes, for *all* spaces over the same number field. The precomputation phase to produce this data is the most expensive step in computing each space.)

Changes and Removals:

– More signatures for creating `HilbertCuspForms` are provided, for convenience.

– The behaviour of `SetRationalBasis` has changed: in particular, the circumstances in which this happens automatically have changed. (This feature may become completely automatic in a future version.)

– The method used to compute a `NewSubspace` can now be controlled by the user, via the `Al` option. The options are `Degeneracy` (degeneracy maps are used), or `Naive` (the old space is recognised as a Hecke module and split off).

Bug Fixes:

– Some runtime errors that occurred during the precomputation phase of the "definite" algorithm for fields of large discriminant have been fixed.

7 Arithmetic Fields (Global)

7.1 Algebraic Number Fields

New Features:

- The `GaloisGroup` of a number field defined over a number field which is itself a relative extension can now be directly computed.
- The `GaloisGroup` of a reducible polynomial over a number field can be computed. It can now also be computed over any extension of a number field.

Changes and Removals:

- A number field of degree 1 can be obtained using `RationalsAsNumberField()`. However, `NumberField(Rationals())` now returns `Rationals()`. (This change is necessary due to compatibility issues.)
- The computation of `Subfields` of a number field represented as an extension of another number field has been replaced by a new algorithm being developed by Klüners and van Höij. `Subfields` can now be computed of any number field defined by a single polynomial.
- The (optional) use of the older algorithm to compute Galois groups has been restricted (mostly) to the `GaloisGroup` signatures which take a field as input rather than a polynomial.
- The algorithm used to compute maximal orders of Kummer extensions has been extended to also compute maximal orders of radical extensions.
- The computation of Dedekind ζ-functions has been improved through the use of Artin representations.

Bug Fixes:

- The computation of residue fields in orders having a multiply relative representation has been fixed
- A failure in the evaluation of a polynomial in the straight-line representation for elements not in the coefficient ring has been fixed.
- A fix has been made to addition of orders.
- Some fixes have been made to the computation of Galois groups.

7.2 Algebraic Function Fields

New Features:

- `Subfields` can be computed for all global function fields which are defined by a single polynomial.
- `GaloisGroup`s of function fields which are extensions of global function fields by a single polynomial can now be computed.
- The intrinsics `Completion` and `Expand` are now available for function fields represented as a relative extension and elements thereof.

Changes and Removals:

- The GaloisGroup and Subfields computations for global function fields have been improved.
- The (optional) use of the older algorithm to compute Galois groups has been restricted to the GaloisGroup signatures which take a polynomial as input rather than a field.
- The algorithm used to compute maximal orders of Kummer extensions has been extended to also compute maximal orders of radical extensions.
- It is now possible to compute the maximal order of an Artin-Schreier extension which is an extension of an extension of a rational function field.
- Orders of function fields are now recognized as being domains and some calculations such as matrix determinants over these orders will speed up as a result.
-

Bug Fixes:

- A fix has been made to GaloisGroup for characteristic p fields.
- The bug in the calculation of the genus of a function field in non-simple representation has been fixed.
- A bug in the computation of residue fields of orders having a multiply relative representation has been fixed.
- A failure in the evaluation of a polynomial in the straight-line representation in the case of elements not in the coefficient ring has been fixed.

8 Arithmetic Fields (Local)

8.1 p-adic Rings and their Extensions

Bug Fixes:

- A bug in IsDivisibleBy for local ring elements has been fixed.
- The Log intrinsic has been extended to arbitrary units rather than 1-units only.

8.2 Series Rings

Changes and Removals:

- A fix in the precision handling of field elements has led to some field elements having a reduced (but correct) precision. One effect of this is that polynomials over a series extension field may no longer have enough precision to be used to construct an extension (where full precision in each coefficient is required). This problem can be avoided by putting the coefficients of the polynomial into a sequence and calling Polynomial on this sequence. This avoids the addition of any O terms to individual terms of the polynomial with the consequent loss of precision. (x is really $x + O(t^{field\_prec})$, where x is a polynomial variable and t is the generator of the series field or extension).

Bug Fixes:

– Factorization of inseparable polynomials over series rings has been fixed.
– A bug in giving a series the full precision of the ring has been fixed.
– Fixed a bug which could causes `Evaluate` to return an answer with too much precision when evaluating a series at another series.

8.3 Lazy Series Rings

Bug Fixes:

– A bug in the computation of `Coefficients` when the range is empty has been fixed.

8.4 General Local Fields

New Features:

– The intrinsic `RelativePrecision` has been added for elements of general local fields.
– The intrinsic `Degree` taking a general local field and a coefficient ring of that field has been added.
– The intrinsic `RelativeField` has been added which computes a local field as an extension of a subfield given a field and a map from the subfield into the local field.

Changes and Removals:

– The mapping to the ramified representation of a general local field has been improved.
– The intrinsic `FixedField` of a general local field has been reimplemented using invariants. Precision issues in `FixedField` are now handled better.
– Loss of precision in some operations in General Local Fields has been reduced, specifically in application of automorphisms, although as a result they are no longer created as `hom<>` objects.

Bug Fixes:

– Fixes to the `RamifiedRepresentation` map have been made for fields of degree 1.
– The mapping of a general local field to its ramified representation has been fixed to avoid unnecessary precision loss.

8.5 Algebraic Power Series

New Features:

– Arithmetic operations may now be specified using the infix symbols `+, -` and `*`. Equality testing can be specified using the operator `eq`.

Changes and Removals:

– The parent of a `RngPowAlgElt` is now a power structure and the type `RngPowAlg` is no longer used. The type `RngPowAlgElt` has been changed to `SerPowAlg`.

9 Basic Rings and Fields

9.1 Integer Ring

Changes and Removals:

- The computation of `Binomial` is now permitted when both arguments are large but the result is still relatively small.
- The behaviour of `ShiftRight` on negative integers has been adjusted to make it consistent across all shift ranges.

9.2 Finite Fields

New Features:

- The algorithm for factoring a polynomial over a finite field when all the coefficients lie in a proper subfield has been improved.
- The finite field relationship machinery has been significantly modified so that fewer auxiliary fields are retained in the background and the updating of relationship information has been sped up.
- One may now use `sub<K | S>` to construct a subfield of a finite field K where S is a sequence or set of elements coercible into K.

9.3 Real and Complex Fields

New Features:

- The computation of the `NumericalDerivative` of a function is now possible for well-behaved functions.
- The `NumericalEigenvectors` corresponding to an approximation of a complex eigenvalue can now be computed in cases that are not ill-conditioned.
- `HypergeometricSeries2F1` now allows rational and integral values of the complex argument.

Changes and Removals:

- Complex arithmetic in Magma has been updated to use MPC 0.8.
- The inverse trigonometric functions and power functions have been updated to use MPC 0.8.

9.4 Polynomial Rings

New Features:

- For various fundamental algorithms (such as GCD and factorization), special efficient techniques are now used when it is detected that the input involves symmetric polynomials.
- Multivariate polynomial GCD has been sped up in the case when the degree in one variable is much larger than that in the other variables.
- A p-adic method is now used for powering multivariate polynomials over finite fields (and other inefficiencies have been fixed).

10 Coding Theory

Bug Fixes:

- Calling `Aut` on a code not defined on a finite field now produces an error message instead of crashing.
- Some memory leaks when re-creating an already-existing code were fixed.
- The intrinsics `PunctureCode` and `ShortenCode` no longer allow a zero-dimensional code to be created.

10.1 Linear Codes over Finite Fields

Changes and Removals:

- The speed of the MacWilliams transform has been improved in some cases.
- The computation of the weight enumerator or the number of words of specific weight now uses the dual code information more effectively in several cases.

Bug Fixes:

- Some memory leaks in the creation of QR codes were fixed.
- `ConstacyclicCode` now raises an error if a shift factor of zero is specified.
- Fixed a crash in the verbose output of `CanteautChabaudsAttack`.

10.2 Linear Codes over Finite Rings

Changes and Removals:

- The intrinsic `KernelZ2CodeZ4` has been improved.
- The `Zimmermann` option for the minimum weight of a $\mathbf{Z}_4$ code has been removed.

Bug Fixes:

- The intrinsics `LeeWeightEnumerator` and `EuclideanWeightEnumerator` now raise an error instead of crashing if the code is not defined over $\mathbf{Z}_4$.

10.3 Additive Codes

New Features:

- The two intrinsics `IsProjective` and `IsAdditiveProjective` have been added to handle the two possibilities for additive codes.

Bug Fixes:

- Fixed a bug in the computation of meet when an additive code is involved.
- Fixed a bug in the computation of `CodeComplement` when using an additive and a linear code.

10.4 Quantum Error-Correcting Codes

Bug Fixes:

- The dimension of quantum codes is now returned as a rational.
- Fixed a bug in the computation of the minimum weight when using the Zimmermann algorithm.
- Fixed the verbose printing of which words were impurities in the minimum weight computation.

11 Combinatorial Theory

11.1 Symmetric Functions

Bug Fixes:

- Several crashes or erroneous successes when coercing into a symmetric function algebra have been fixed.

11.2 Hadamard Matrices

New Features:

- The database of Hadamard matrices has been expanded. New matrices have been added for degrees 36 (1), 48 (60) and 60 (1759). The additional matrices were supplied by Dragomir Djokovic.

11.3 Graphs

New Features:

- Intrinsics `NumberOfVertices` and `NumberOfEdges` added.

Bug Fixes:

- `MinimumCut` now returns an error instead of crashing when the source and sink vertices are the same.

12 Commutative Algebra

12.1 Polynomial Rings

New Features:

- New function `ElementToSequencePad` (abbreviation `EltseqPad`) to return the coefficients of a univariate polynomial quotient ring, with zero padding.

Bug Fixes:

- Fixed a bug that could occasionally cause the constant term of a polynomial defined over the complex numbers to be printed incorrectly.
- Fixed a bug in the root computation of polynomials over local fields where the polynomial was not coercible into the corresponding local ring.

13 Convex Polytopes and Polyhedra

13.1 Toric Lattices

New Features:

- A map between toric lattices can be tested for being the identity map via `IsIdentity`.
- The value $v \cdot v$ for a lattice point v can be computed more efficiently using the new `Norm` intrinsic.
- Many operations on sequences of lattice points are now also possible on sets.
- A lattice point can now be multiplied on the right by a square matrix with rational or integer entries.

13.2 Fans

Changes and Removals:

- The intrinsic `NormalFan` has been added as an alias for the `DualFan` of a polytope.
- The intrinsic `DualFaceInDualFan` now accepts a face defined by a polyhedron, rather than simply a sequence of (ray) indices.
- The intrinsic `NonSimplicialCones` allows fast extraction of the non-$\mathbf{Q}$-factorial cones from a fan.
- Improved speed on fan creation from Cox data.

Bug Fixes:

- Fixed a bug in the creator of the Cox data from a degenerate fan.

13.3 Cones

New Features:

- The intrinsic `MatrixOfInequalities` returns the inequalities of a cone arranged in a matrix, with each row representing an inequality.
- The intrinsic `RandomPositiveCone` can be used to create a random cone.
- The intrinsic `NormalCone(P,F)` has been added to return the (outer) normal cone to a face F of a polyhedron P.
- The intrinsic `BoxElements` generates the points in the fundamental domain of a simplicial cone.
- The toric ideal associated with a cone can be constructed via `Ideal`.

Changes and Removals:

- An improved version of Hemmeke's algorithm for calculating minimal generators of a semigroup has been implemented. This can be called directly via the `MinimalPositiveGenerators` intrinsic, and is used indirectly when computing the Hilbert basis of a cone.

- An improved version of the intrinsic `ZGenerator` makes intelligent use of Hemmecke's algorithm when computing the Hilbert basis.
- `MinimalPositiveGenerators` can now handle huge gradings.
- Point finding in a cone is now significantly faster and requires a great deal less memory.
- A cone can be multiplied on the right by a square matrix over **Z** or **Q**.

Bug Fixes:

- Removed memory leaks associated to the Barvinok algorithm.
- Given an identity map *id* and cone C, `Image(Id, C)` now returns C rather than constructing a copy of C.
- The ambient lattice of the data returned by `QuotientGenerators` has been fixed to be mathematically consistent with the underlying maps.

13.4 Polytopes and Polyhedra

New Features:

- A pair of polytopes can be tested for isomorphism or equivalence.
- The `hVector` for a simplicial polytope can now be computed.
- The intrinsic `IsFace(P,F)` is provided for polyhedra.
- Perfect centring for a polytope P may be determined using the intrinsic `IsPerfectlyCentered`.
- The Minkowski decompositions of a lattice polygon can be computed.
- The minimum lattice width of a polytope can be computed.
- Intrinsics `Degree` and `Codegree` are provided for lattice polytopes.
- Added `IsFlag` for a simplicial polytope.
- The intrinsic `VertexFacetHeightMatrix` returns the lattice height of each vertex of a polytope with respect to each facet. The `VertexFacetIncidenceMatrix` can be used to tell which vertex is contained in which facet.
- Several databases of polytope classifications have been included. For further details see the online Graded Ring Database (`http://grdb.lboro.ac.uk/`).
- A polytope can be saved in `PALP` format for use with Kreuzer and Skarke's polytope analysis software.
- The `ViewWithJavaview` intrinsic can be used to visualise a polytope (of dimension at most three) using the Javaview software, if installed.

Changes and Removals:

- The intrinsic `EhrhartCoefficient` now accepts negative dilation factors.
- The intrinsic `IsIntegral` has been modified to avoid calculating the vertices unless absolutely necessary, improving the speed of a number of operations.

- The intrinsics `InteriorPoints`, `NumberOfInteriorPoints`, and `NumberOfBoundaryPoints` now make better use of reflexivity when appropriate. Special-case low-dimensional algorithms improve point finding and counting in many cases.
- The intrinsic `fVector` now uses the results of `arXiv:1002.2815v2` in the low-dimensional smooth cases.
- Ehrhart calculations for smooth Fano polytopes now take advantage of the results in `arXiv:1004.3817v1`.
- Ehrhart calculations for low-dimensional reflexive polytopes now take advantage of the results in `arXiv:1002.2815v2`.
- The time taken for computing the volume of a simplicial polytope has been improved. More generally, the algorithm may choose to make use of the Ehrhart data.
- The volume of a 0-dimensional polytope is now defined to be 1 rather than 0.
- A polyhedron can be multiplied on the right by a square matrix over **Z** or **Q**.

Bug Fixes:

- The algorithm for computing the volume of a one-dimensional polytope (i.e. line segment) didn't handle the case when the end points were not integral. This has been corrected.

14 Groups

14.1 Finite Groups

- Code has been developed by Derek Holt and Bill Unger for constructing a semidirect product of two finite groups.
- A utility function for `DicyclicGroup` has been added.

Changes and Removals:

- The two functions `G/N` and `quo<G|N>` have been made equivalent, so the first now returns the quotient epimorphism, as the second has always done. In the case of quotients of permutation and matrix groups, where the return type is always a permutation group, that group need not be a regular group as previously has been the case.
- The memory behaviour of the straight-line program representation of group elements has been revised, thereby greatly improving the speed when there are many elements in use.
- The printing format for a subgroup lattice has been inverted. The full group is now printed on the first line, and the trivial subgroup is printed on the last line. The numbering of the subgroup classes has been left unchanged.

Bug Fixes:

- A number of errors in the filtering stage of the `Subgroups` intrinsic have been fixed, particularly in the cases of the `IsNilpotent`, `OrderEqual` and `OrderDividing` filters.

14.2 Permutation Groups

New Features:

- An implementation of the Law-Niemeyer-Praeger-Seress "jellyfish" algorithm is available. The initial construction is by the `JellyfishConstruction` function. If this function succeeds, the `JellyfishImage` function maps from the large degree group to the small, while `JellyfishPreimage` provides an inverse.

14.3 Matrix Groups – General

New Features:

- For matrix groups represented in terms of a base and strong generating set, the `Representation` function now returns this data structure. In a similar vein, the `WordInStrongGenerators` function has been extended to matrix groups.
- For matrix groups represented in terms of a base and strong generating set, Magma makes more effort to retain a useful base when constructing the image and kernel of homomorphisms and when performing `ChangeRing` operations.
- Matrix groups over quaternion algebras can now be constructed, and various computations can be performed. These include: conjugacy testing and character tables, invariant forms, isomorphism testing, and G-modules.

14.4 Matrix Groups Over Finite Fields

The Composition Tree (CT) package developed by Henrik Bäärnhielm, Derek Holt, Charles Leedham-Green and Eamonn O'Brien, working with numerous collaborators, is being released for the first time. This implements the Composition Tree representation for a matrix group defined over F_q. It is an alternative representation when it is not practical to compute a BSGS for a matrix group defined over F_q.

New Features:

- The intrinsic `CompositionTree` and its associated functions allow the user to construct and manipulate a composition tree data structure for a matrix group defined over F_q.
- A number of functions have been implemnted by Derek Holt that take as their starting point the CT representation and construct the "trivial Fitting" model (TF model) for a matrix group defined over F_q.
- Holt has further developed a first set of functions for working with this TF model. For example, the intrinsic `LMGCompositionSeries` constructs a composition series for the group. Similar intrinsics have been provided which compute such things as a chief series, soluble radical, the derived subgroup, the centre and the Sylow p-subgroups.
- A number of functions for computing orders of standard matrix groups over finite fields have been modified to work with factored integers. This can improve performance.

14.5 Matrix Groups Over Characteristic 0 Fields

A new package, "Infinite", has been developed by Alla Detinko, Dane Flannery and Eamonn O'Brien for groups defined over number fields, or (rational) function fields in zero or positive characteristic.

New Features:

- The intrinsic `CongruenceImage` constructs various types of congruence images – defined over a finite field – for a finitely-generated linear group defined over a number field, or a (rational) function field in zero or positive characteristic.
- The intrinsic `IsFinite` includes new algorithms to decide finiteness of a finitely-generated linear group G defined over a number field or a (rational) function field in zero or positive characteristic. If the group is finite, then `IsIsomorphic` constructs an isomorphic copy of G as a matrix group of the same degree defined over a finite field.
- The intrinsic `HasFiniteOrder` can now be used to test whether an invertible matrix defined over a (rational) function field in zero or positive characteristic has finite order.
- The intrinsics `IsNilpotent`, `IsSoluble`, `IsCompletelyReducible` and `IsUnipotent` can now be applied to groups defined over a number field, or a (rational) function field in zero or positive characteristic.
- Functions which decide "virtual" properties are available for the first time for a linear group defined over number fields and in some cases a (rational) function field in zero or positive characteristic. The functions are `IsPolycyclicByFinite`, `IsSolubleByFinite`, `IsNilpotentByFinite`, `IsAbelianByFinite`, and `IsCentralByFinite`.

– The intrinsics `IsIrreducibleFiniteNilpotent` and `IsPrimitiveFiniteNilpotent` decide irreduciblity and primitivity of finite nilpotent matrix groups defined over a number field or a rational function field.

– Given a finite subgroup of $GL(n, \mathbf{Q})$, its $GL(n, \mathbf{Z})$ conjugacy classes can be determined using the intrinsic `ZClasses`.

– Given two finite subgroups of $GL(n, \mathbf{Z})$, the intrinsic `IsGLZConjugate` will determine whether they are conjugate. Similarly, `IsGLQConjugate` will determine whether two finite subgroups of $GL(n, \mathbf{Q})$ are conjugate.

– A database of representatives of the $GL(n, \mathbf{Z})$-conjugacy classes of irreducible maximal finite subgroups of $GL(n, \mathbf{Z})$ for $n \leq 11$ and $n \in \{13, 17, 19, 23\}$ is included.

Changes and Removals:

– The intrinsic `EndomorphismRing` now returns a subalgebra of $\mathbf{Z}^{n\times n}$ when applied to subgroups of $\mathrm{GL}_n(\mathbf{Z})$.

14.6 Finite Chevalley Groups

Changes and Removals:

– The intrinsic `ChevalleyGroup` for twisted groups has been changed to ensure consistency with ATLAS notation. In particular, the group ${}^3D_4(q)$ is returned by either `ChevalleyGroup("3D",4,q)` or by `ChevalleyGroup("3D",4,GF(q^3))` and the group ${}^2E_6(q)$ is returned by either `ChevalleyGroup("2E",6,q)` or by `ChevalleyGroup("2E",6,GF(q^2))`. This is consistent with the current Magma implementation for Chevalley groups of type ${}^2A_n$ (unitary groups).

– The order and factored order of any finite Chevalley group is now efficiently computed without setting up the corresponding group.

14.7 Classical Groups

New Features:

– Quadratic spaces have been implemented.

– Clifford algebras have been implemented as structure constant algebras, using the above quadratic space machinery. An immediate application will be to the construction of the spinor group associated with certain orthogonal groups.

– Witt's theorem has been implemented for orthogonal and symplectic geometries over finite fields, including fields of characteristic two. That is, an isometry defined on a subspace can be extended to an isometry of the entire space.

– The intrinsic `ClassicalStandardGenerators` returns the Leedham-Green and O'Brien standard generators for a classical group of specified type.

– The intrinsic `ClassicalConstructiveRecognition` constructs the standard generators for a classical group $G = \langle X \rangle$ in its natural representation as straight line programs (SLPs) in X.

- The intrinsic `ClassicalStandardPresentation` sets up a “short” presentation for a classical group on its standard generators.
- The intrinsic `InvolutionClassicalGroupEven` constructs an involution in $G = \langle X \rangle$, a classical group in natural representation and even characteristic, and returns both the involution and its SLP in X.

14.8 Finite Soluble Groups

New Features:

- Two new intrinsics, `MinimalNormalSubgroups` and `Socle`, have been provided for finite soluble groups given by pc-presentations.

Bug Fixes:

- An error which would sometimes lead to the erroneous calculation of the lower p-central series of a group has been fixed.
- An error that could occasionally result in the incorrect calculation of the p-class of a p-group, arising either from the above problem or from the use of `pCoveringGroup` with a group having nuclear rank 0, has been fixed.

14.9 Finitely Presented Groups

New Features:

- The intrinsic `DicyclicGroup` constructs a dicyclic group of order $4q$ given either an integer q or a cyclic group of order $2q$ and an appropriate automorphism.

Changes and Removals:

- Changes to the way in which chains of subgroups are managed may improve execution times in some calculations.
- The various `SimpleQuotients` intrinsics have been made consistent regarding the parameters they accept, and the version which took a tuple argument has been removed.

14.10 Finitely Presented Abelian Groups

Bug Fixes:

- A bug in abelian groups which led to crashes when the order (or prime divisors of the order) exceeded 2^{30} has been fixed.

14.11 Automatic Groups and RWS Groups

New Features:

- The `AutomaticGroup` function has two new parameters. The `Huge` parameter turns on/off the `-h` option of the KBMAG program.

  The `IgnoreHaltingFactor` parameter allows the user to disable stopping construction by halting factor.

Bug Fixes:

- A crash that occurred occasionally when converting a RWS group to sequences has been fixed.

15 Lattices

15.1 Lattices

New Features:

- Hermitian automorphism groups (including quaternionic structures) can now be computed in some cases, though not on lattices *per se*, but rather on Gram matrices.
- The `LatticeDatabase` includes about 10 more extremal even unimodular lattices. These include a recently discovered 72-dimensional lattice from Nebe called `(SL2(25)xPSL2(7)):2`, two 80-dimensional lattices from Stehlé and Watkins called `SL2(79)` and `(SL2(41).circ.S3~).2`, three 64-dimensional lattices including one from coding theory by Ozeki called `T64` (the others are "generic" lattices `G64` and `H64`), and a cyclotomic lattice of dimension 56 called `B_{56,1}^4` from a 1995 paper of Batut, Quebbemann, and Scharlau.
- The facility to obtain a lattice from a linear code over $\mathbf{Z}/m\mathbf{Z}$ has been added.

Bug Fixes:

- A memory leak in the shortest vector computation was fixed.
- The behaviour of th `EnumerationCost` function has been modified in the context of search-tree pruning.
- A bug in the computation of the inverse of the quotient map of a lattice by a sublattice has been fixed.
- A bug giving incorrect results in `ClosestVectors` for a lattice with non-standard basis and non-trivial inner product has been fixed.

16 Lie Theory

16.1 Reflection Groups

New Features:

- The intrinsic `ComplexRootDatum` computes a root datum for a finite complex reflection group.
- The implementation of pseudo-reflections has been revised to ensure consistency with the Bourbaki definition. (Pseudo-reflections include projections and transvections.)
- New functions `SymplecticTransvection` and `UnitaryTransvection` have been implemented.

Changes and Removals:

- `ComplexReflectionGroup` and `ImprimitiveReflectionGroup` have been deprecated and will be removed in a future release. Their functionality has been superseded by a new implementation of `ShephardTodd`. The old implementation of `ShephardTodd` has been renamed `ShephardToddOld`.
- The intrinsic `IsPseudoreflection` has been renamed to `IsPseudoReflection` to reflect the changed functionality.

16.2 Lie Algebras

New Features:

- The intrinsic `ChevalleyBasis` is now able to compute Chevalley bases for Lie algebras of algebraic groups over fields of characteristic zero as well as over finite fields of non-zero characteristic. Furthermore, there is a new variant that takes a root datum with respect to which the Chevalley basis should be computed as one of its arguments. Finally, the Chevalley basis for Lie algebras constructed by `LieAlgebra(R, k)` for a root datum R and a field k now correctly returns a Chevalley basis with respect to the root datum R.
- The new intrinsic `IsChevalleyBasis` tests whether a particular basis is a Chevalley basis.
- The intrinsic `SemisimpleType` now works in many cases for Lie algebras over finite fields of characteristic 2 or 3.
- The intrinsic `LieAlgebraOfDerivations` is provided to construct the Lie algebra of derivations for a structure constant Lie algebra.
- The new intrinsic `ReductiveType` is provided to recognize a reductive Lie algebra. It is designed to work in all characteristics, and for both split and twisted Lie algebras. Moreover, upon successful recognition a proof in the form of a Chevalley basis or a twisted basis is returned.
- Several intrinsics dealing with (split) toral subalgebras have been added, notably `SplitToralSubalgebra`, `SplitMaximalToralSubalgebra`, and `IsSplitToralSubalgebra`.
- The new intrinsic `MelikianLieAlgebra` is provided to construct Melikian Lie algebras.
- The new intrinsic `QuotientWithPullback` is provided to construct the quotient L/I of a Lie algebra L by an ideal I, togther with the pullback map from I to L.
- Several improvements have been made in the interface for computations with finitely presented Lie algebras. For example, it is now possible to construct homomorphisms from finitely presented Lie algebras and to construct quotients of such Lie algebras by relations.

Changes and Removals:

- The intrinsic `TwistedRootDatum` now accepts a string as first argument (whereas previously it would only accept a root datum), and it now also accepts a permutation for the optional argument `Twist`. Furthermore, when `TwistedRootDatum` is used to construct an untwisted root datum it emits a warning.
- In addition to the existing method for creating twisted Lie algebras by a split root datum R and a permutation p (using `LieAlgebra(R, k, p)`) it is now possible to construct twisted Lie algebras by a twisted root datum S (using `Lie Algebra(S, k)`) or, if the split root datum R is irreducible and not of type D_4, simply by `TwistedLieAlgebra(R, k)`.
- Construction of a Cartan-type Lie algebra L by means of the intrinsics `WittLieAlgebra`, `(Conformal)SpecialLieAlgebra`, `(Conformal)HamiltonianLieAlgebra`, and `ContactLieAlgebra` now also returns a map between a polynomial ring and L, to aid in the identification of the basis elements of L.
- The intrinsics `AdjointRepresentation` and `StandardRepresentation` now take an optional argument `ComputePreImage`, which is `true` by default.
- The number of intrinsics for constructing Lie algebras has been reduced: `SemisimpleLieAlgebra`, `SemisimpleMatrixLieAlgebra`, `ReductiveLieAlgebra`, `ReductiveMatrixLieAlgebra`, `SimpleLieAlgebra`, and `SimpleMatrixLieAlgebra` are now deprecated in favour of `LieAlgebra` and `MatrixLieAlgebra`. For the time being, the deprecated intrinsics remain available with the suffix `-Old`.

Bug Fixes:

- Some assertion failures that could arise when using universal enveloping algebras have been fixed.
- Fixed the `quo` constructor for finitely presented Lie algebras.
- The intrinsic `CompositionSeries` for structure constant algebras has been sped up.
- Several small bugs in the intrinsic `TwistedRootDatum` were fixed.
- The intrinsic `IsSplittingCartanSubalgebra` has been sped up.

16.3 Universal Enveloping Algebras

Bug Fixes:

- Fixed some incorrect internal assertions when using universal enveloping algebras.

16.4 Representations

Bug Fixes:

- Three omissions in `RestrictionMatrix` were fixed. Bug reported and solution provided by R. Zeier.
- A problem in the arithmetic with representation decompositions was fixed.

17 Linear Algebra and Module Theory

17.1 Matrices

New Features:

- New function `IsMonomial` to test whether a matrix is monomial.
- Matrices over associative algebras have various added functionality, such as `EchelonForm`.
- Matrix rings/spaces now allow coercion of sequences of sequences of elements.
- The intrinsic `Cofactors` has been added for cases where the positional signs of the minors are wanted.
- The function `IsDiagonal` now also returns a sequence giving the entries on the diagonal when the matrix is diagonal.
- New function/procedure `RemoveZeroRows`.

Changes and Removals:

- The intrinsic `Minors` now returns the minors in adjugate order and no longer accepts a `Signed` argument.

Bug Fixes:

- The universe is now set correctly when `RowSequence` returns an empty sequence.

17.2 Sparse Matrices

The operations on sparse matrices have been expanded to include many more which correspond to the operations provided for dense matrices.

New Features:

- Sparse matrices with many zero rows now have a more compact representation, which results in less memory usage and some speedups.
- Arithmetic operations such as negation, scalar multiplication, inverse and powering are now supported.
- New function `Eltseq` for a sparse matrix to give its entries as tuples.
- The predicates `IsZero`, `IsOne`, `IsMinusOne`, `IsScalar`, `IsDiagonal`, `IsSymmetric`, `IsUpperTriangular`, `IsLowerTriangular` are now supported.
- New functions/procedures `Submatrix` (`ExtractBlock`), `SubmatrixRange` (`ExtractBlockRange`), `InsertBlock`, `RowSubmatrix`, `RowSubmatrixRange`, `ColumnSubmatrix`, `ColumnSubmatrixRange` have been provided to extract or replace submatrices.
- New functions/procedures `SwapRows`, `SwapColumns`, `ReverseRows`, `ReverseColumns` have been provided for row and column operations.
- New functions/procedures `AddRow`, `AddColumn`, `MultiplyRow`, `MultiplyColumn`.
- New functions/procedures `RemoveRow`, `RemoveColumn`, `RemoveRowColumn`, `RemoveZeroRows`.

18 Linear Associative Algebras

18.1 Quaternion Algebras

New Features:

- Most features for quaternion algebras over number fields are now supported for rational function fields over finite fields of odd characteristic.
- Matrices and matrix groups over quaternion algebras (and general associative algebras in some cases) are now supported to some extent.
- The algorithm for computing right ideal classes now incorporates a range of new techniques which cut down the search. These include a refined mass formula, smarter search strategies, use of automorphisms, use of theta series, and some implementation speedups.

Changes and Removals:

- The intrinsic `IsSplittingField(AlgQuat, FldQuad)` has been removed. Instead `IsSplittingField(Fld, AlgQuat)` should be used.
- The comparison `cmpeq` is now available for quaternion orders as well as associative orders.

Bug fixes:

- A number of bug fixes have been made in code that determines the equality of orders for quaternion algebras over **Z** and polynomial rings.
- A bug in orders of quaternion algebras over rings other than **Z** has been fixed.
- Determining equality of ideals of orders of quaternion algebras over rings other than orders of number fields has been fixed.
- For ideals of orders of quaternion algebras, a bug in the intrinsic `Conjugate` has been fixed.

18.2 Matrix Algebras

New Features:

- It is now possible to compute the split basic algebra corresponding to a matrix algebra defined over F_q. This uses the Carlson-Matthews algorithm for computing a presentation for a matrix algebra.

Changes and Removals:

- The implementation of the Carlson-Matthews algorithm for computing a presentation for a matrix algebra defined over a finite field has been improved in a number of ways. This resulted in a substantial reduction in the runtime for harder examples.

19 Representation Theory

19.1 $K[G]$-Modules

New Features:

- The Meataxe tools in Magma have been revised so as to take advantage of situations where the action matrices have a block diagonal structure. This can result in significant speedups.
- A package has been developed to compute the projective indecomposable KG-modules that project onto specified irreducible KG-modules, for a finite field K. Thus, the intrinsic `ProjectiveIndecomposableModules(G, K)` finds the projective indecomposables corresponding to the inequivalent irreducible KG-modules for G.
- The intrinsic `ProjectiveIndecomposableDimensions(G, K)` returns the dimension of each projective indecomposable KG-module.
- The intrinsic `CartanMatrix(G, K)` returns the Cartan matrix corresponding to the irreducible KG-modules. Similarly, `AbsoluteCartanMatrix(G, K)` returns the Cartan matrix corresponding to the absolutely irreducible KG-modules.
- The intrinsic `DecompositionMatrix(G, K)` specifies the absolutely irreducible constituents for the mod-p restriction of each ordinary irreducible character.
- The intrinsic `BasicAlgebraFG` constructs the basic algebra of a modular group algebra. The method has been successfully applied to groups having order up to several million.
- As a variant on the above feature, the intrinsic `BasicAlgebraOfBlockAlgebra` computes the basic algebra associated with any p-block of a modular group algebra.

19.2 Modules over Algebras

New Features:

- The function `QuaternionicGModule` allows the user to write a G-module over a quaternion algebra when a splitting is known.

19.3 Cohomology

New Features:

- Higher cohomology groups can be computed using the new intrinsic `CohomologicalDimensions`. This function determines the dimensions of $H^n(G, M)$, where M is a KG-module, using projective covers and dimension shifting.
- The intrinsic `Ext` computes $Ext^1(M, N)$ for G-modules M and N.
- The availability of $E = Ext^1(M, N)$ enables the user to construct module extensions described in terms of an element of E.