Summary of New Features in Magma V2.20

December 2013

1 Introduction

This document provides a terse summary of the new features installed in Magma for release in version V2.20 (December 2013). A small number of new features were exported in patch releases prior to the main release of V2.20 in December 2013 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 V2.19-x.

Recent releases of Magma were: V2.19 (December 2012), V2.18 (December 2011), V2.17 (December 2010), V2.16 (November 2009), V2.15 (December 2008), V2.14 (October 2007), V2.13 (July 2006).

2 Highlights

Algebraic Geometry

- *Schemes*
  - The first stage of a package for working with isolated singularities that are analytically isomorphic to a hypersurface singularity, has been implemented. Included are functions for classification, computation of normal form and explicit transformation to normal form following Arnold's classification scheme. These normal forms are useful for isomorphism testing and singularity resolution while knowledge of the classification family is important in deformation theory. The implementation also works in positive characteristic. The cases in which the corank does not exceed 2 are implemented in this release and the corank 3 cases will follow in the near future.
  - A function to compute the blow-up of an arbitrary subscheme of a scheme is now available. The computation is global with the result optionally embedded in a product space or ordinary projective space. Blow-ups are important birational maps in the geometry of varieties having dimension 2 or greater.
  - Two functions have been added to compute Segre embeddings of multiple products of projective schemes into ordinary projective space. This is often a useful tool when working with product varieties as it allows access to functionality that is only available for ordinary projective space.
  - It is now possible to compute the reduction mod p of the flat closure of ordinary projective schemes and their divisors. This works over the rationals and number

fields. It ensures that in computing the “naive” reduction of the scheme, the defining ideal is properly saturated with respect to a local parameter before reducing the defining equations.

  – An efficient test has been provided for local-freeness (the Cartier property) of divisors. It is important to know whether or not a divisor is Cartier for the validity of a number of divisor operations.

  – New routines are provided for determining multiplicities of divisors that should provide a faster option in certain circumstances. The main new function returns all multiplicities simultaneously of divisors known to be supported on a particular set of prime divisors without explicitly computing a primary decomposition by reducing to dimension 1 via hyperplane slices.

- *Curves*

  – An intrinsic has been added to return the zeta function of a given singular/affine curve (as opposed to its projective normalisation). This allows the user to obtain the generating function for the number of points over base field extensions for the given curve model, as is needed in some arithmetic problems.

- *Surfaces*

  – Parametrization of degree 9 Del Pezzo surfaces over the rationals has been drastically improved, making it practical for a reasonable range of cases. The improvement is due to applying simplification/reduction techniques at several stages. (This is the most important special case arising in the parametrization of rational surfaces.)

  – New constructors for Kummer surfaces, rational ruled surfaces and random complete intersections have been added. These are user-convenience intrinsics that should be helpful in the initial creation of surfaces of the given types.

Arithmetic Fields

- *Class Groups of Number Fields*
  - A new implementation of the main class group algorithm, undertaken by Steve Donnelly, was released a year ago (version 2.19). This has been developed further, resulting in significant speed-ups and applicability to a larger range of fields. Even bigger speed-ups will be attained in patch releases following shortly after this release, as additional new techniques are switched on.
  - A completely new implementation by J-F. Biasse (closely modelled on his own independent C code) of the sieve algorithm for finding relations is included. This replaces the implementation released two years ago. This code is extremely effective for fields of degree up to 5, and to a lesser extent for degree 6. In this version, the sieve method is used internally by the main class group routine, to whatever extent it is advantageous.
- *Function Fields*
  - Specialised algorithms for computing maximal orders are now used for Kummer extensions and Artin–Schreier–Witt extensions.
- *Galois Groups*
  - A major revision of the code for computing Galois groups of extensions of number fields and function fields has been undertaken by Stephan Elsenhans. The improvements include using smaller transformations, better selection of a prime, and improved use of the subgroup generated by the Frobenius automorphism. In the case of polynomials over Q, this results in generally faster runtimes, particularly for polynomials of degree up to 32, and success in a greater proportion of cases of degree greater than 32.
  - In the case of function fields of characteristic p, additional improvements have been made. These include use of a new invariant in characteristic 2 and fast detection of fields which are extensions of their base field. The overall improvement for these fields is significant.
  - The code for computing a Galois group utilises various heuristics so there is a small possibility that the result returned is not correct. In the case of Galois groups of polynomials over the rationals, Stephan Elsenhans has devised a new method for proving correctness. The user may run this code on the result returned by the Galois group intrinsic to verify correctness of the result.
- *Series Rings over p-adic Rings*

– A package originally written by Xavier Caruso and David Lubicz (Rennes) for computations with power series rings over p-adic rings has been merged into Magma. Features include the ability to work with modules over these rings, as per the original direction of Iwasawa. This package should be of interest to those working in the field of p-adic Hodge theory.

Arithmetic Geometry

- *Elliptic Curves Over Number Fields*

  Over a considerable number of years algorithms for arithmetic over elliptic curves have been developed for the case of curves over $\mathbf{Q}$. Gradually, we are developing much of the same functionality for curves over number fields. The new tools appearing in this release include:

  – A routine for computing the saturation in the Mordell-Weil group of a given set of points.

  – Cremona-Prickett-Siksek bounds on the difference between naive and canonical heights. The code was contributed by J. Cremona.

  – The computation of integral points on curves over totally real number fields.

  – A search routine for elliptic curves having given conductor (and traces), now available over arbitrary number fields.

- *Hyperelliptic Curves*

  – Chabauty's method is a technique for determining the rational points on a curve C of genus at least 2 (when the Mordell-Weil group of Jac(C) has rank less than the genus of C). The technique becomes more powerful when combined with a "Mordell-Weil sieve", which is a technique that combines information obtained using reduction at many primes. The existing Magma routine for Chabauty together with Mordell-Weil sieve (due to Michael Stoll) has been replaced by an improved version supplied by Michael Stoll. The main difference is that now, the user only needs to provide a point of infinite order on Jac(C).

  – The action of the `FrobeniusMatrix` on (a chosen basis of) the differentials of a hyperelliptic curve over the rationals has now been made available (this previously was done as a subcomponent of Kedlaya's algorithm, and was available for elliptic curves vis-a-vis p-adic heights). Requested by M. Stoll.

- *L-Series*

  – The L-series of a Hilbert modular eigenform can now be constructed for forms of all weights (previously only for parallel weights).

Modular Forms

- *Modular Forms*
  - Version 2.20-2 includes an improved version of Alan Lauder's code for computing the characteristic series of U_p on p-adic modular forms. The new version is faster, and handles spaces with nontrivial character.
- *Hilbert Modular Forms*
  - In version 2.20-2, computation of new subspaces is implemented for all spaces that can be constructed. (Previously, for spaces of nontrivial weight there was a restriction on the level.) For all spaces computed using the "definite" algorithm, new subspaces are computed via degeneracy maps.
- *Modular Forms over Imaginary Quadratic Fields*
  - In 2009, Dan Yasaki (UNC) developed a Magma package for modular forms over imaginary quadratic fields. Specifically, the package deals with cuspidal spaces of weight 2 modular forms on $\Gamma_0(N)$, over any imaginary quadratic field. A new faster version of this package, contributed by Dan, is now installed. In other improvements, a new reduction technique is used in combination with the other methods. The overall effect significantly speeds up construction of spaces and calculation of Hecke operators.
  - An algorithm for computing new subspaces of these spaces is now implemented, and the newform decomposition of a new subspace can also be obtained.

Associative Algebras

- *General Associative Algebras*
  - Maximal orders in algebras over number fields can now be computed.
- *Basic Algebras*
  - The package developed by Jon Carlson for computing the automorphism group and testing isomorphism of basic algebras has been improved.

Basic Rings and Fields

- *Finite Fields*

- For discrete logarithm computations in finite fields of small characteristic, a new implementation of the 'double rational' version of the Function Field Sieve (FFS) was developed. In this version, the two function fields used to generate the relations on the logarithms are both rational function fields. This algorithm is more efficient than the existing Coppersmith's algorithm for fields of very high degree and is automatically selected in V2.20 for larger fields.
- The database of factor bases used for fast evaluation of discrete logarithms in fields of small characteristic is continually being extended and now includes data for larger fields. In particular, the fields GF(2^d) are now covered for all prime degrees $d \leq 509$ and most degrees up to 500.

Combinatorial Theory

- *Graph Theory*
  - The version of the Brendan McKay's graph automorphism/isomorphism program *nauty* installed in Magma has been upgraded to version 2.5r6.
  - The graph automorphism/isomorphism program *Traces* developed by Adolfo Piperno has also been installed. For certain types of graph, such as large Hadamard graphs, it can be a great deal faster than *nauty*.
- *Hadamard Matrices*
  - The intrinsics that test Hadamard matrices for equivalence and compute their automorphism group may now use the Traces algorithm (see above) by specifying an appropriate parameter. For larger Hadamard matrices it can be faster by more than a factor of 1000 compared with the default version that uses *nauty*.

Commutative Algebra

- *Gröbner Bases*
  - Magma V2.20 contains a new **dense variant** of the Faugere F_4 algorithm for computing Groebner bases developed by Allan Steel. This variant is currently only practically applicable to input systems over a finite field where the input polynomials are considered "**dense**"; such systems include various cryptographic inputs such as HFE, MQQ and Minrank systems. The algorithm takes significantly less time and memory for several types of input and if an NVIDIA GPU is available, then this is also exploited, yielding greater speedups for larger examples. An optional heuristic can lead to even more dramatic speedups.

Group Theory

- *Finitely-Presented Groups*
  - A finitely presented group G is said to be *large* if it has a finite index subgroup possessing a homomorphism onto a nonabelian free group. A package for testing whether a group G is large, developed by Jack Button, has been installed in V2.20. The package implements a one-sided test: if it returns true then G is large, but if it returns false, then no conclusion can be drawn.
- *Infinite Matrix Groups (Over Fields)*
  - A new version of the *Infinite* package developed by Detinko, Flannery, and O'Brien is available. As well as upgrades to existing functions, this provides access to new intrinsics for working with soluble-by-finite matrix groups defined over number fields. The new intrinsics include tests to decide if such a group has finite Hirsch and Pruefer rank; to determine if a subgroup has finite index; and to construct an isomorphism from a completely reducible abelian matrix group to a polycyclic group.
- *Quasisimple Groups*
  - A new implementation of the algorithm of Jambor et al (2013) for the constructive black box recognition of alternating and symmetric groups is included. A variant can be used to perform constructive recognition of perfect central extensions of the alternating and symmetric group either in matrix and permutation representation. The implementation was undertaken by Jonathan Conder (Auckland) and Sebastian Jambor (Auckland).
- *Polycyclic Groups*
  - The basic arithmetic and subgroup routines for polycyclic groups (`GrpGPC`) have been revised. In particular, a major problem with leaking memory blocks has been fixed. This will allow much more extensive computations with groups of this type than were previously possible. This group type allows the user to create either finite or infinite groups, and large integer exponents are allowed in the presentations. The alternative group type (`GrpPC`), while it supports many more group operations, is restricted to finite groups where the exponents may not exceed $2^{30} - 1$.

Linear Algebra and Module Theory

- *Over Finite Fields*
  - This release of Magma (2.20) is the first with the capability of utilising GPUs in certain circumstances. In particular, code has been developed by Allan Steel for matrix multiplication over GF(2^k) and GF(p) that will run on systems with an NVIDIA Tesla GPU with CUDA support. For large matrices over GF(p), the speedup is by a factor of 15 to 20 compared to a state-of-the-art Intel Ivy Bridge processor.
  - Since matrix multiplication over some characteristic zero rings is mapped to the GF(p) case, this type of linear algebra will also benefit from the use of a GPU.
  - Matrix multiplication and echelonization over GF(2^k) has been sped up for $1 < k \leq 5$.
  - The basic matrix multiplication algorithm over the small finite fields GF(p) for $p = 3, 5, 7$ has been sped up (typically by a factor of 1.5 to 2). This leads to substantial speedups for all linear algebra computations over such finite fields.
  - The iterative algorithm of Lanczos for solving large sparse linear systems modulo a large integer has been been made faster.
  - A multi-threaded parallel version of the Lanczos algorithm has been developed to allow applications to utilise multiple cores on any system. The typical speedup achieved is a factor of about $0.7 \times k$ on a machine with k cores.

Representation Theory

- *Group Algebras*
  - The code for constructing the basic algebra of a matrix algebra over a finite field has been improved in a number of ways with the result that it is now much faster in the case of matrix algebras having large degree. For example, it has been used to construct the basic algebras of the group algebra of the Mathieu group M_{23} in all finite characteristics.

System

- *NVIDIA GPU support*
  - A CUDA-based executable will be available from V2.20-3 onwards. This will allow Magma to exploit an NVIDIA Tesla GPU when present.

3 Documentation

New Handbook Chapters:

– Series rings over p-adic rings

4 Language and System Features

New Features:

– An AVX-based executable has been available for suitable Intel and AMD processors. This version includes speedups for several types of Groebner basis and linear algebra computations (up to 1.5 times faster than the SSE-based Intel 64 version in some cases).

– A CUDA-based executable will be available from V2.20-3 onwards. This will allow Magma to exploit an NVIDIA Tesla GPU when present. Currently GPUs are exploited in some types of matrix multiplication and Gröbner basis computations; see the relevant sections. Further information will be given on the AMD/Intel 64-bit downloads page as to how this executable should be used on systems with GPUs and CUDA installed.

– The debugger has been updated to include a new `identifiers` command which lists the set identifiers in the current context.

– User-defined types may now define a Compare intrinsic that will be used by system sorting operations.

5 Aggregates and Mappings

Changes and Removals:

– The construction of a set now insists that coercion into its universe is possible so that empty sets cannot be constructed when the universe does not have a corresponding element type (V2.19-5).

6 Algebraic Geometry

6.1 Schemes

New Features:

- An intrinsic `BlowUp` has been added that computes the global blow-up of an arbitrary subscheme of a scheme in affine, projective or product projective space. The result can be chosen to lie in the product of the original ambient and a new projective space or it can be embedded into ordinary projective space (V2.19-7).
- Intrinsics `SegreProduct` and `SegreEmbedding` have been added to allow the embedding of multiple products into ordinary projective space via an iterated Segre embedding. The first Segre-embeds the product of a sequence of ordinary projective schemes. The second Segre-embeds a scheme lying in one of Magma's product projective spaces (V2.19-7).
- Reduction mod p of the flat closure of an ordinary projective scheme defined over the rationals or a number field is now possible with the `Reduction` intrinsic.
- `RandomCompleteIntersection` is a new intrinsic to give random complete intersections of specified dimension and degree in ordinary projective space (V2.19-5).

Changes and Removals:

- The test for establishing that a projective space is not a scroll has been improved (V2.19-5).

Bug Fixes:

- A memory bug in the construction of maps between schemes has been fixed (V2.19-5).
- The intrinsic `PointSearch` now coerces the points it finds into the given scheme (rather than have them be in the ambient).

6.2 Analytical-hypersurface Singularities

New Features:

- The first stage of a package to classify isolated hypersurface singularities following V. I. Arnold's classification scheme has been undertaken. Apart from identifying the Arnold type of a singularity, its normal form along with an explicit analytic isomorphism to the normal form up to any desired precision is also computed.
- The classification contains singularities of corank ≤ 3 (with one corank 4 case). The code in this release covers the corank ≤ 2 cases (and the corank 4 one) apart from a few of the corank 2 families. It is planned to implement the last few corank 2 cases and the corank 3 cases for the next release.
- The main intrinsic is `NormalFormOfHypersurfaceSingularity`. This takes an analytic hypersurface singularity at the origin which may be defined exactly by a multivariate polynomial or given by the truncation of a defining power series along with a data object to allow further expansion of the power series if necessary. It returns whether the singularity lies in one of Arnold's families and if so, the name of the family, the normal form and, optionally, the transformation to normal form up to specified precision.

– The package works in characteristic p as well as characteristic 0 with a careful analysis of precisely which are the bad characteristics for each singularity family and appropriate runtime warnings for the user.

– The `Corank2Case` is made available to call directly for rank 2 singularities with non-trivial quadratic part.

– The new intrinsic `MilnorNumberAnalyticHypersurface` computes the Milnor number of an analytic-hypersurface singularity from the data object returned by `IsHypersurfaceSingularity`.

– The new intrinsic `TjurinaNumberAnalyticHypersurface` computes the Tjurina number of an analytic-hypersurface singularity from the data object returned by `IsHypersurfaceSingularity`.

6.3 Scheme Divisors

New Features:

– An intrinsic `IsCartier` has been added to efficiently test whether a divisor is Cartier (locally defined by a single equation).

– `MultiplicityFast` is a variant on the `Multiplicity` intrinsic that can compute the multiplicity of a Cartier prime divisor in another divisor D more quickly (but without being able to simplify the ideal factorisation of D).

– A faster method to compute all of the multiplicities of a divisor D with respect to a set of prime factors that it is known to be supported on, is provided by the new intrinsic `Multiplicities`. This uses an iterative slicing by good hyperplanes to reduce to the curve case where Magma's curve functionality for divisors and places can be used.

– A `Reduction` intrinsic has been provided for divisors – mirroring the one for schemes. This intrinsic computes the reduction mod p of the divisor as a divisor on the reduction mod p of the flat closure of its (ordinary projective) variety X.

– The new intrinsic `EffectiveHypersurfaceTwist` produces an effective divisor in the rational equivalence class $D + rH$ where $r \geq 0$ and H is the hyperplane divisor of the ordinary projective variety X on which D lies. This intrinsic is often useful as it allows a reduction to the effective case when testing properties that are not affected by such twisting.

6.4 Algebraic Curves

New Features:

– A new intrinsic `ZetaFunctionOfCurveModel` has been added to compute the zeta function of an irreducible curve over a finite field in the model given by its defining equations, which may be affine and/or singular. (The standard `ZetaFunction` is for the projective normalisation of the curve.) The algorithm is to compute the standard zeta function and adjust it.

Bug Fixes:

– A bug in the preimage of a map between a function field of a curve and its representation as an algebraic function field has been fixed (V2.19-8).

– A bug in the conversion of places of a function field places to places of a curve in `WeierstrassPlaces` of a divisor of a curve has been fixed (V2.19-8).

– Some bugs (runtime errors) in `RegularModel` have been fixed.

6.5 Algebraic Surfaces

New Features:

- New surface constructors, `KummerSurfaceScheme`, `RationalRuledSurface` and `RandomCompleteIntersection` have been provided (V2.19-5).

Changes:

- The Lie algebra/simple algebra splitting routines for `ParametrizeDegree9DelPezzo` has been greatly improved. More care is taken in the choice of basis and Cartan subalgebra in the early phase of dealing with the Lie algebra to avoid the massive coefficient blow-ups that plagued the older version. For the simple algebra trivialisation phase, more efficient lattice methods of Fisher and Ivanyos-Ronyai-Schicho are now used (V2.19-7). The intrinsic now also returns a parametrization map with inverse equations included.

7 Arithmetic Geometry

7.1 Rational Curves and Conics

New Features:

- The routine `HasRationalPoint` for solving conics over number fields has been carefully tuned in several respects. The switch-over point to the norm equation method is based on a better estimate of the cost of the heuristic class group and now also the unit group computations.

7.2 Elliptic Curves

7.2.1 Elliptic Curves over the Rational Field

New Features:

- The Cremona-Prickett-Siksek bounds for the difference between naive and canonical heights are available using code contributed by J. Cremona. The bounds are accessed via `CPSHeightBounds`, or combined with other available bounds via `HeightDifferenceBounds`.

Changes and Removals:

- In `ThreeSelmerGroup`, the parameter `MethodForFinalStep` is obsolete, and its value now has no effect. A much better version of this step is now implemented.

Bug Fixes:

- Previously, `LocalInformation(E, p)` at a prime p computed a global minimal model, which meant that it sometimes got stuck trying to factorize a large discriminant. This problem is now avoided.
- A minor problem with the options `RemoveTorsion` or `RemoveGenerators` in `FourDescent` has been patched. The previous code could enter an infinite loop in checking local images, but now reverts to a global method instead.
- In `NineDescent`, some runtime errors which occurred occasionally have been fixed.
- The `Rank` of some elliptic curves over the rationals was incorrectly computed in some cases, when there was more than one large quartic. Example reported by J. Balakrishnan.

7.2.2 Elliptic Curves over Number Fields

New Features:

- The Cremona-Prickett-Siksek bounds for the difference between naive and canonical heights are available using code contributed by J. Cremona. The bounds are accessed via `CPSHeightBounds`, or combined with other available bounds via `HeightDifferenceBounds`.
- The `Saturation` routine is implemented for curves over number fields.
- The `EllipticCurveSearch` routine is implemented for curves over arbitrary number fields.

- The `IntegralPoints` can be computed for a curve over a totally real number field. Currently, the parameter `FBasis` must be given: the routine finds all integral points in the group generated by the given points.
- The `Periods`, `EllipticExponential`, and `EllipticLogarithm` intrinsics can now be called for an elliptic curve over a number field with an extra argument corresponding to a real embedding.

Changes and Removals:

- The normalisation of the `ConjecturalRegulator` has been changed to be consistent with `Regulator` (and other functions involving heights).

Bug Fixes:

- In the `RationalPoints` search routine, the search region was incorrectly defined. The corrected version is much more likely to find points of small height sooner.
- In some instances, `MordellWeilShaInformation` printed a warning message and hung for an extended time. This occurred while computing the Cassels-Tate pairing for 2-coverings that had not been minimized at some large prime. This behaviour should not occur now.
- A bug has been fixed in the routine for elliptic curve Chabauty. This only affects curves of rank greater than 1. Reported by N. Freitas.

7.3 Genus One Models

Changes:

- A small fix to the `AddCovers` routine was provided by Tom Fisher, which provides functionality to allow nonminimised models to be added.

Bug Fixes:

- In `Minimise` for 2-coverings over number fields, two problems which occurred occasionally have been fixed: an infinite loop, and not minimising at a large prime.

7.4 Hyperelliptic Curves and Jacobians

New Features:

- A new version of the `Chabauty` routine (Chabauty plus Mordell-Weil sieve for genus 2 curves) has been included. The argument `ptJ` now is only required to have infinite order – previously it was required to generate the the Mordell-Weil group modulo torsion. An optional parameter `Saturate` has been added: if this is set to false, `ptJ` is assumed to be a generator (making the computation faster); otherwise, a proven generator is obtained during the computation. Contributed by M. Stoll.
- New routines `IsDivisibleBy` and `Saturation` for points on Jacobians are included. Contributed by M. Stoll.
- The computation of the `FrobeniusMatrix` for the action on differentials is now made directly available to the user. This uses Kedlaya's algorithm (the Magma implementation is due to M. Harrison).

Bug Fixes:

- The `HeightPairing` for points on a Jacobian has been fixed in some cases. Reported by J.-S. Mueller.
- In the `RationalPoints` search routine for curves over number fields, the search region was incorrectly defined. The corrected version is much more likely to find points of small height sooner.
- In `TwoCoverDescent`, a runtime error concerning precision for local factorization has been fixed.
- A minor bug has been fixed in `Degree2Subcovers` (the degree 2 subcover elliptic curves of genus 2 curves). Reported (and fixed) by M. Stoll.

7.5 Hypergeometric Motives

New Features:

- The computations at wild primes have been improved in some cases, due to code from D. Roberts.
- Some rudimentary ability to compute data over fields other than the rationals is now available, but only with the hypergeometric traces and not yet with the L-function.

7.6 L-Series

New Features:

- The `LSeries` for a Hilbert modular newform of non-parallel weight is now implemented.
- The `HodgeStructure` has been added for various L-functions.
- A `TensorProduct` now inherits the minimal precision of its constituent parts. (V2.19-8).

Bug Fixes:

- A bug with computing the `HodgeStructure` of a tensor product was fixed. Reported by D. Roberts (V2.19-7).

7.7 Modular Forms

New Features:

- Version 2.20-2 includes a new version of `OverconvergentHeckeSeries`, for computing the characteristic series of U_p on spaces of p-adic overconvergent modular forms. The improved version is substantially faster, due to better techniques and strategy in several steps of the algorithm. The routine is now also implemented for spaces with nontrivial character. Contributed by Alan Lauder.

7.8 Hilbert Modular Forms

New Features:

- In version 2.20-2, `NewSubspace` is implemented for all spaces that can be constructed. Previously, for spaces of nontrivial weight (i.e. any weight except parallel weight 2), this was implemented only in cases where the computation involved primes that divide the level exactly once. For all spaces computed using the "definite" algorithm, the method for computing new spaces is to use degeneracy maps (the preferred method).
- For spaces computed using the "definite" algorithm, the Hecke action on the full space (the space used internally for all the computations) is accessible using `HeckeMatrixRaw`.

Changes and Removals:

- In previous versions, all spaces of Hilbert modular forms were cached on their base field. This saves time when computing different spaces over the same field, but means that one eventually runs out of memory. In the new version, caching is controlled by the user. By default there is no caching; caching is switched on/off by calling `SetStoreModularForms`, and the cache can be cleared by calling `ClearStoredModularForms`.

Bug Fixes:

- For spaces using the "indefinite" algorithm, for moderately large level the memory usage was catastrophically bad. It has been improved to the extent that memory should not be a major issue for computations that finish in a reasonable time, at least.

7.9 Modular Forms over Imaginary Quadratic Fields

New Features:

- A new version of the package has been contributed by Dan Yasaki. One of the main improvements is that cusp forms are computed directly so that the full cohomology groups are not computed. This makes computation of spaces and Hecke operators substantially faster.
- A new reduction algorithm for elements of hyperbolic 3 space is used, which cuts off most of the looping in the sharbly reduction step.
- An algorithm for new/old subspaces is implemented. Intrinsics `NewSubspace` and (lower level) `NewAndOldSubspaces` are available.
- The `NewformDecomposition` of a new space can be computed.
- Caching of spaces of forms can be controlled using `SetStoreModularForms`. This is the same as for Hilbert modular forms (see above).

Changes and Removals:

- The (undocumented) intrinsic `FullDimension` has been removed (because this does not feature in the new algorithm).

8 Arithmetic Fields (Global)

8.1 Algebraic Number Fields

New Features:

- The main ClassGroup algorithm has been significantly improved.
- An iterator for quotients of orders by ideals has been provided (V2.19-6).
- Random has been implemented for quotients of orders by ideals (V2.19-6).
- The euclidean operations for elements of quotients of orders by ideals have been fixed and expanded, including the provision of div, mod, Quotrem, Lcm, Gcd, XGcd and EuclideanNorm (V2.19-6).
- The intrinsic Annihilator has been provided for elements of quotients of orders by ideals (V2.19-6).
- RelativeField is now available for FldOrd as well as FldNum.

Changes and Removals:

- In ClassGroup, the options "Sieve"/"NoSieve" for the Al parameter are deprecated. These choices are now made automatically.
- The check on the Al parameter to Subfields has been improved (V2.19-4).
- Subfields are now returned in the same order each time they are requested for a field (V2.19-4).
- A more efficient minimal polynomial computation is used in the computation of defining polynomials of Subfields of number fields in absolute representation.
- Error checking has been improved so that PrimitiveElement can only be computed for ideals of maximal orders (V2.19-9).
- The AbsoluteField of a linear extension is now the AbsoluteField of its coefficient field.
- Constructing an order from a ModDed has been made more flexible. For example, if the first argument is an equation order whose coefficient ring is not necessarily maximal and the coefficient ring of the module is the maximal order of the coefficient ring of the order, the order constructed will be a transformation of the order defined by the defining polynomial of the input order over the (maximal) coefficient ring of the module. This equation order over a maximal order does not need to be constructed independently. The parameter NFBasis has been added to this Order intrinsic.

Bug Fixes:

- A problem has been fixed in AbsoluteAlgebra where it would fail to terminate due to unnecessarily attempting to factorize a large discriminant. The problem sometimes showed up in descent routines, for instance TwoCoverDescent, as reported by N. Freitas.
- A bug in the computation of Subfields of a number field defined by a non-monic polynomial has been fixed (V2.19-3).
- A bug has been fixed in RelativeField when the first argument is a degree 1 extension of **Q** (V2.19-3).
- A bug has been fixed in the KluenersvanHoeij Subfields algorithm (V2.19-4).
- A repeated instance of an automorphism has been removed when Automorphisms is applied to a field of fractions of an order (V2.19-5).

- A bug involving denominators in `PowerProduct` has been fixed (V2.19-6).
- The management of memory when constructing a quotient of an order by an ideal which has already been constructed has been fixed (V2.19-7).
- A memory leak in coercion between orders has been fixed (V2.19-8).
- A fix has been made to `CharacteristicPolynomial` so that properties of orders are not altered during the computation (V2.19-8).
- A memory leak has been fixed in `IsSquare` for elements of number fields (V2.19-10).
- A crash has been fixed in addition of orders when one of the summands is an equation order (V2.19-10).
- A crash in some `SolveByRadicals` computations has been fixed (V2.19-10).
- If a number field which is represented as an extension of another number field has an equation order and that equation order is known to be maximal it will now be returned as the maximal order of the number field. Previously, a newly constructed order was returned but this caused compatibility issues in some places where certain orders were expected to be identical.
- Some orders returned from `MaximalOrder` when the input was an order in a Kummer extension but not an equation order did not store their known maximality. This has been fixed.

8.1.1 Cyclotomic Fields

Bug Fixes:

- A crash in `RootOfUnity` in a cyclotomic field when the degree of the field is less than half the order of the field has been fixed (V2.19-3).

8.2 Characters and Artin Representations

New Features:

- Additional functionality has been provided for characters over number fields with the intrinsic `TotallyUnitTrivialSubgroup` for Dirichlet characters (required by L. Dembele), `NormInduction` for Hecke characters, and also `GrossenTwist` for Grossencharacters (V2.19-4).

Bug Fixes:

- The `GrossenTwist` intrinsic erroneously had its return type given as the non-existent "Grossenchar" rather than `GrossenChar`. Reported by D. R. Kohel (V2.19-5).
- The `LSeries` of a real Dirichlet character could return the values -1, 0, and +1 in a cyclotomic field (rather than the integers) when defined via a large `DirichletGroup`. Reported by T. Ward (V2.19-5).

8.3 Algebraic Function Fields

New Features:

- The computation of `LPolynomial` and `ZetaFunction` is now available for larger combinations of genus and constant field size (V2.19-10).
- A new `StrongApproximation` intrinsic has been added.
- The intrinsic `ArtinSchreierReduction` computes Artin–Schreier quotients of elements of fields having prime characteristic.
- A `Decomposition` of a prime can now be computed in any order or field the prime lies in a coefficient ring of. Previously a prime could only be decomposed in a direct extension of the ring it was contained in.
- The intrinsic `IntegralSplit` can now be applied to elements of field of fractions of orders.

Changes and Removals:

- The intrinsic `FunctionField([RngMPolElt])` has been removed.
- When a non-simple extension is constructed it is now checked that the resulting extension will be separable and if not an error occurs (V2.19-7).
- The computation of a `CoveringStructure` has been refined for sets of ideals of orders of function fields. In particular, there is no covering structure for a set of ideals of a finite order and a set of ideals of an infinite order (V2.19-9).
- Elements are constructed with respect to an equation order basis as this is considerably more efficient for arithmetic (V2.19-10).
- The `CoveringStructure` of orders now ensures that if two orders are contained in a field then so is their covering structure.
- Constructing an order from a `ModDed` has been made more flexible. For example, if the first argument is an equation order whose coefficient ring is not necessarily maximal and the coefficient ring of the module is the maximal order of the coefficient ring of the order, the order constructed will be a transformation of the order defined by the defining polynomial of the input order over the (maximal) coefficient ring of the module. This equation order over a maximal order does not need to be constructed independently. The parameter `NFBasis` has been added to this `Order` intrinsic.
- The denominator of an element of a field of fractions of an order of a function field is now used when computing the `Eltseq`.

Bug Fixes:

- A bug has been fixed in `Poles` for non-squarefree elements with nontrivial denominators (V2.19-5).
- The `StrongApproximation` intrinsic has been improved to better handle input of zero (V2.19-9).
- Factorization of polynomials over some representations of function fields has been fixed (V2.19-7).
- A crash has been fixed in addition of orders when one of the summands is an equation order (V2.19-10).

8.4 Abelian Extensions of Function Fields

Changes:

- The intrinsics `MaximalOrderFinite` and `MaximalOrderInfinite` when applied to abelian extensions of function fields are now more efficient using specialised algorithms to compute maximal orders of the Kummer (V2.19-4) and Artin–Schreier–Witt extensions which are their components.

Bug Fixes:

- Bugs in the computation of maximal orders of Kummer extensions in some instances have been fixed.
- Bugs in the computation of maximal orders of abelian extensions in some instances, where the maximal order is obtained by combining those of component subfields, have been fixed.

8.5 Galois Groups

New Features:

- Various improvements have been made to the computation of Galois groups especially over characteristic p function fields. These include using smaller transformations, a new invariant in characteristic 2, improved efficiency and cost comparisons (V2.19-5), selection of a prime (V2.19-8) and improvements to the verbose printing and improved use of the subgroup generated by the Frobenius automorphism (V2.19-9).
- A shortcut has been implemented which can sometimes determine whether a field has no subfields, by making use of information about cycle types occurring in the Galois group computation.
- For a function field extension which can be defined as an extension of the constant field, the `GaloisGroup` is now calculated directly using this. A fast test is used to check for this case, by considering cycle types, rather than always computing the `ExactConstantField` (which can be expensive).
- The `GaloisProof` intrinsic for polynomials with rational coefficients now uses relative resolvents and set stabilizers in permutation groups. This produces a significant speedup. Further, reducible polynomials and some degenerate cases (not treated by the old code) can now be handled.

Changes and Removals:

- The roots returned as the second argument from `GaloisGroup` when the input is a reducible polynomial are now scaled to be algebraic integers, consistent with the case of an irreducible polynomial.

Bug Fixes:

- The `Bound` computation for `ProdSum` invariants of type `RngSLPolElt` has been fixed (V2.19-5).
- Some fixes to `GaloisGroup` computations supplied by J. Klüners have been included (V2.19-7).
- A fix has been made to the computation of the `GaloisGroup` of a non-monic reducible polynomial (V2.19-10).

9 Arithmetic Fields (Local)

9.1 p-adic Rings and their Extensions

Changes and Removals:

- Direct coercion has been enabled in certain situations; for instance, from a p-adic ring or field into an extension of a p-adic ring or field of a differing precision (V2.19-5).

Bug Fixes:

- `Factorization` of polynomials over local rings and fields with the `Extensions` parameter set to `true` has been fixed (V2.19-9).

9.2 Series Rings

Bug Fixes:

- The handling of zero roots of polynomials over series rings has been improved, sometimes zero roots were listed twice (V2.19-10).
- A crash has been fixed in `HasRoot` for polynomials over series rings when a root could not be computed (V2.19-10).

9.3 Series Rings over p-adic Rings

New Features:

- A new package for computations over $\mathbf{Z}_p[[u]]$ and related rings has been contributed by Xavier Caruso and David Lubicz (Rennes), and adapted for use in Magma. This includes methods for working in the localizations of these rings (extending by either $1/p$ or a suitable inversion of u), and to compute with modules over these localizations.

  These are termed "slope" rings, for given a rational $\nu = a/b$ (the slope) one considers the ring S^ν whose elements have nonnegative `GaussValuation` with respect to ν. This Gauss valuation is defined for $\sum_i a_i u^i \in \mathbf{Q}_p[[u]]$ as the minimum of $v_p(a_i) + i\nu$, where v_p is the p-adic valuation. The case of $\nu = 0$ has $S^0 = \mathbf{Z}_p[[u]]$. The p-localization S^ν_p is $S^\nu[1/p]$, while the u-localization S^ν_u is $S^\nu[p^a/u^b]$. Both these latter rings are Euclidean.

  This package in particular implements `Quotrem` and `ExtendedGcd` in the localizations, which then allow matrix and module theory to be used, via `EchelonForm`, `HermiteForm`, `SmithForm`, `Kernel`, `Image`, etc.

  One would want to be able to deal with modules over S_0 directly, but this is typically not feasible (even up to quasi-isomorphism) without passing to localizations, whereupon the resulting precision analysis is ameliorated via the introduction of these slope rings. In general, precision issues become more acute as ν increases, which can be a problem both in theory and in practice.

10 Basic Rings and Fields

10.1 Integer Ring

Bug Fixes:

- Several functions for Dirichlet characters were not implemented to handle the case of characters with values in large finite fields, causing them to never terminate. These include `DirichletCharacterFromValuesOnUnitGenerators`, and several features such as coercion and equality testing.
- A problem with not detecting powers when stored factors were used has been resolved. For instance: pq^2, where p was stored and q was large, such as $p = 10^{15} + 37$ and $q = 10^{50} + 151$ (V2.19-8).

10.2 Finite Fields

New Features:

- For discrete logarithm computations in finite fields of small characteristic, a new implementation of the 'double rational' version of the Function Field Sieve (FFS) was developed. In this version, the two function fields used to generate the relations on the logarithms are both rational function fields. This algorithm is more efficient than the existing Coppersmith's algorithm for fields of very high degree and is automatically selected in V2.20 for larger fields.

  The database of logarithms is continually being extended and now includes data for larger fields. In particular, the fields GF(2^d) are now covered for all prime degree $d \leq 509$ and most degrees up to 500.
- Arithmetic in the fields GF(2^k) for $k > 1$ has been sped up on Intel processors with AVX support (using Intel's AVX instruction for polynomial multiplication of packed polynomials over GF(2)).

11 Combinatorial Theory

11.1 Graphs

New Features:

- The version of the Brendan McKay's graph automorphism/isomorphism program *nauty* installed in Magma has been upgraded to version 2.5r6.
- The graph automorphism/isomorphism program *Traces* developed by Adolfo Piperno has been installed. For certain types of graph, such as large Hadamard graphs, it can be much faster than *nauty.*

11.2 Hadamard Matrices

New Features:

- The intrinsics that test Hadamard matrices for equivalence and compute their automorphism group may now use the Traces algorithm (see above) by specifying an appropriate parameter. For larger Hadamard matrices it can be faster by more than a factor of 1000 compared with the default version that uses *nauty.*

12 Commutative Algebra

12.1 Polynomial Rings

Bug Fixes:

- Some inefficiencies when performing polynomial operations over large residue class rings has been fixed. Reported by J. Klueners (V2.19-5).

12.2 Multivariate Polynomial Rings

Bug Fixes:

- Coercion into a quotient polynomial ring has been fixed so that elements from recursive base rings are handled properly (V2.19-1).
- A crash when computing sygyzies over polynomial rings over residue class fields has been fixed. Reported by F. Lunnon (V2.19-5).
- A bad inefficiency in exact division of multivariate polynomials was fixed. Reported by M. Grassl (V2.19-7).

12.3 Affine Algebras

New Features:

- Rational function fields over fields of fractions of affine algebras are now fully supported.

Bug Fixes:

- Problems with meet for affine algebras has been fixed Reported by C. Quitte (V2.19-1).

12.4 Ideal Theory and Gröbner Bases

Magma V2.20 contains a new **dense variant** of the Faugere F_4 algorithm for computing Groebner bases developed by Allan Steel. This variant is currently only practically applicable to input systems over a finite field where the input polynomials are considered "**dense**"; that is, if the input polynomials are written as a matrix with columns labelled by the monomials, then the input matrix must be dense.

Dense input systems which are applicable include various cryptographic inputs such as HFE, MQQ and Minrank systems. (The algorithm does work correctly on sparser input systems but is typically not as efficient as the standard F_4 algorithm for such inputs.)

Some key features of the dense variant of the F_4 algorithm are the following:

- This variant essentially uses the same matrices as in the standard F_4 algorithm (i.e., uses the same S-polynomials and reductors as usual) but exploits dense portions of the matrices where possible in the linear algebra phase, leading to significant speedups for larger examples.

- There is often also significant savings in memory usage over the standard F_4 algorithm.
- If an NVIDIA GPU is available, then this is also exploited, yielding greater speedups for larger examples.
- Finally, there is an optional heuristic which can be selected in the algorithm when solving systems of equations over GF(2), called the *'Reduction Heuristic'* which can give an even greater reduction in time and memory usage in large examples. See the documentation or the webpage below for details.
- As an example, the 80-variable first HFE challenge of Patarin can now be solved in 186.6 seconds on 3.2GHz Xeon system or in 86.0 seconds on such a system which also has a NVIDIA Telsa C2075 GPU.

Magma V2.20 will automatically select the dense F4 algorithm if it considers that the input system is sufficiently dense. But one can force the dense variant to be used or not by setting the parameter `Dense` to `true` or `false` respectively. If the dense variant is turned off, Magma will use the standard F4 algorithm which is unchanged since V2.19.

For more timing examples of the new algorithm and further information, please see the webpage:

`tinyurl.com/DenseF4`

Bug Fixes:

- A bug in `Radical` which caused a proper subset of the radical to be returned has been fixed. Reported by E. Rains (V2.19-3).
- A bug when computing Gröbner bases over the integers has been fixed. Reported by E. Rains (V2.19-4).
- A bug in `Radical` in small characteristic (arising when computing reduced subschemes) has been fixed. Reported by Damiano Testa (2.19-6).
- A problem when coercing between function fields (affecting `PrimaryDecomposition`) has been fixed. Reported by D. Kohel (V2.19-7).
- A crash in `SyzygyModule` has been fixed. Reported by M. Reid (V2.19-7).
- Some bugs in PrimaryDecomposition and Radical involving ideals with graded orders were been fixed (V2.19-8).

13 Geometry

13.1 Convex Polytopes and Polyhedra

The items in this subsection have been contributed by Al Kasprzyk.

New Features:

– The new intrinsic `VertexEdgeIncidenceMatrix` can be used to recover the vertex-edge incidence matrix of a polyhedron.

– The new intrinsic `NormalEdgeCones` can be used to recover the (outer) normal cones of the edges of a polyhedron. When working with polytopes, this is significantly faster than the equivalent sequence of calls to the intrinsic `NormalCone`.

– The new intrinsic `EdgeFacetIncidenceMatrix` can be used to calculate the edge-facet incidence matrix of a polyhedron.

– The computation time required by `IsFano` has been significantly reduced. A new intrinsic `IsWeakFano` has been added.

– The new fan constructor FanWithWeights has been added. This allows the creation of a fan based on the given weights and optional ample divisor.

Changes:

– The computation of the automorphism group of a polytope has been improved.

– The choice of generators selected by intrinsic `QuotientGradings` has been improved. Suggested by Antonio Laface and Damiano Testa.

– Attempts to construct a fan from contradictory cone data now gives the explanation of what went wrong as part of the error message, rather than as a separate piece of output.

– A particular sequence of transformations on the `Zero` element of a toric lattice could result in `IsPrimitive` giving an incorrect answer; this has now been corrected.

Bug Fixes:

– Under very special conditions, isomorphism testing of rational polytopes could fail with an assertion error. This has now been fixed. Reported by Antonio Laface and Damiano Testa.

– A bug has been fixed in `IsEquivalent` when analysing rational polytopes of non-zero codimension.

13.2 Finite Geometry

Changes:

– The construction of a subplane of a finite plane now uses the square root bound (Bruck 1955) on the order of a proper subplane to speed up the case where the subplane generated is the whole plane.

14 Groups

14.1 Automatic Groups

Changes:

- The lengths of words accepted by the reduction machine has been increased beyond the previous 2^{15}, to 2^{30}.

14.2 Finitely Presented Groups

New Features:

- A package developed by Jack Button which determines whether an fp-group has the property of "largeness".

Changes:

- The default size of a coset table for the Todd-Coxeter procedure has been increased by a factor of 10 to 4×10^7. This affects order, index, and coset table calculations for finitely presented groups. This limit, and many others, may be reset by the user using the `SetGlobalTCParameters` intrinsic procedure. The relevant parameter for this change is `Workspace`.

Bug Fixes:

- Bugs in `LowIndexNormalSubgroups` have been fixed. Bugs reported by Primoz Potocnik. (V2.19-4)
- A crash when the intrinsic `Homomorphisms` was applied to a finitely-presented Coxeter group has been fixed. This also fixes a crash when applying `SimpleQuotients` to such a group. (V2.19-6)
- A crash computing the abelian quotient of an fp-group has been fixed. Bug reported by Derek Holt. (V2.19-7)
- A crash when computing the Gröbner basis of some finitely presented algebras has been fixed. Reported by D. Simpson (V2.19-9).
- A crash when constructing subgroups of a finitely presented group has been fixed. This crash was found by an unknown user of the online calculator. (V2.19-10)

14.3 Polycyclic Groups

Major Improvements:

- A large amount of memory wastage in the basic arithmetic and subgroup routines for `GrpGPC` has been fixed. This has allowed much more extensive computations with groups of this type without running out of memory.

Bug Fixes:

- A bug with hashing of `GrpGPC` elements has been fixed, fixing incorrect handling of sets of `GrpGPC` elements.

– A bug where the centre of a `GrpGPC` was not fully computed has been fixed.

– A bug where the `PolycyclicGroup` constructor failed to detect an uncollected word appearing as the RHS of a relation has been fixed. Such a relation will now give a runtime error. Reported by E. O'Brien. (V2.19-7)

14.4 Infinite Matrix Groups (over Fields)

New Features:

– A new version of the *Infinite* package developed by Detinko, Flannery, and O'Brien is available. As well as upgrades to existing functions, this provides access to new functions for working with soluble-by-finite matrix groups defined over number fields. The new functions include tests to decide if such a group has finite Hirsch and Pruefer rank; to determine if a subgroup has finite index; and to construct an isomorphism from a completely reducible abelian matrix group to a polycyclic group.

14.5 Matrix Groups Over Finite Fields

New Features:

– A new function `ClassicalRewrite` has been added to the CompositionTree package which allows the user to write an arbitrary element of a classical group as an SLP in its standard generators. A special version, `ClassicalRewriteNatural`, performs the same task for a classical group in its natural representation. The new code was prepared by Csaba Schneider.

Changes:

– The generators of `ChevalleyGroup` E7 have been changed to a Steinberg pair where the first generator is diagonal. The second generator has not been changed. The corrected generator was supplied by Don Taylor, at the request of Eamonn O'Brien. (V2.19-6)

Bug Fixes:

– A problem when computing normalizers of matrix subgroups has been fixed. This has improved computing subgroups of a matrix group in some cases. (V2.19-3)

– An error and crash when computing conjugacy classes of a general linear group has been fixed. This bug was reported, as a crash in `PermutationCharacter`, by Alex Bartel. (V2.19-3)

– A crash due to the incorrect construction of the order of the matrix group `GOPlus(2,4)` has been fixed. Bug reported by U. Thiel. (V2.19-7)

14.6 Permutation Groups

New Features:

– A new intrinsic `TableOfMarks` has been installed. This may be applied to sufficiently small permutation groups and pc-groups to get Burnside's table of marks for the group. The algorithm first computes the lattice of subgroup classes of the group. (V2.19-9)

Changes:

- The algorithms for socle of a permutation group have been extensively revised and corrected. A number of problems with primitive groups of degree between 10^6 and 10^7 have been corrected. The algorithm for dealing with imprimitive groups with one minimal block system has been rewritten. This includes preparatory work for extending the degree limit for this suite of algorithms beyond 10^7, as this limit has become computationally feasible in recent times.
- The algorithm used for intersection with a normal subgroup has been changed to not use a group of double the degree of the original problem, as the double degree algorithm was causing timing and memory problems.

Bug Fixes:

- A crash computing MaximalSubgroups has been fixed. Bug reported by Andreas-Stephan Elsenhans. (V2.19-3)
- A crash computing normal subgroups of a permutation group has been fixed. Bug reported by Peter Mueller. (V2.19-3)
- An error in the stored order of a normal closure within permutation groups has been fixed. Bug reported by Peter Mueller. (V2.19-3)
- A crash when computing the core of a subgroup of a permutation group has been fixed. Bug reported by Peter Mueller. (V2.19-3)
- A bug causing a crash in `LowIndexSubgroups` when applied to a permutation group has been fixed. (V2.19-5)
- Changes to the socle algorithm (noted above) have fixed crashes with groups of degree between 10^6 and 10^7. (V2.19-7)

14.7 Quasisimple Groups

New Features:

- A new implementation of the black box recognition algorithm of Jambor et al (2013) for alternating and symmetric groups has been included. This replaces the algorithm of Beals et al (2003). A variant of the algorithm will perform constructive recognition of perfect central extensions of the alternating and symmetric group either in a matrix and permutation representation. The implementation is by S. Jambor and J. Conder.
- The recognition algorithm is accessed by means of the existing intrinsic `RecogniseAlternatingOrSymmetric`, which now returns different values to the previous version. If the algorithm succeeds given a group G, then G is isomorphic to H which is either alternating or symmetric. In this case it returns true, an isomorphism from G to H, an isomorphism from H to G, the map from G to its word group, and the map from the word group to G. The sixth value returned is true if H is the symmetric group, otherwise false. If the algorithm fails, then the first and only return value is false, the remaining five possible return values are unassigned.

14.8 Finite Soluble Groups

New Features:

- The algorithm for computing the centre of a pc-group (which is not a p-group) has been improved, avoiding the use of very large field extensions. The underlying method, due to Leedham-Green, remains as it was. This also fixes a crash reported by Stephen Humphries. (V2.19-7)
- A new parameter `MaxAuts` has been added to the intrinsics `GeneratepGroups` and `Descendants` to compute maximally automorphic descendants. The code was supplied by Eamonn O'Brien.
- A new intrinsic `TableOfMarks` has been installed. This may be applied to sufficiently small permutation groups and pc-groups to get Burnside's table of marks for the group. The algorithm first computes the lattice of subgroup classes of the group. (V2.19-9)

System Improvements:

- Homomorphism construction and evaluation with domain a `GrpPC` has been improved.

Improvements:

- Computing automorphism groups (using `AutomorphismGroupSolubleGroup`) and performing isomorphism testing (using `IsIsomorphicSolubleGroups`) are now faster and can tackle more complex examples due to improvements to p-group isomorphism and automorphism computations.

Changes:

- Magma level printing of `GrpPC` and their elements has been changed to the compact presentation format for groups, and to sequence of integers for group elements.

Bug Fixes:

- A crash using `Subgroups` to find the normal subgroups of a pc-group has been fixed. Bug reported by Nils Bruin. (V2.19-2)
- A crash when computing `IrreducibleModules` of pc-group has been fixed. Bug reported by Primoz Potocnik. (V2.19-3)
- A crash when computing `DistinctExtensions` of a pc-group has been fixed. Bug reported by Primoz Potocnik. (V2.19-3)
- A crash in `RepresentativeCocycles` for a pc-group has been fixed. Bug reported by Stephen Humphries. (V2.19-4)
- A crash in the routines for computing cohomology of pc-groups has been fixed. Bug reported by Stephen Humphries, fixed by Eamonn O'Brien and Ronan Egan. (V2.19-5)
- A bug whereby an incomplete subgroup lattice of a pc-group was stored in the group as if it were the complete subgroup lattice has been fixed. This led to subsequent calls to Subgroups intrinsics returning incomplete answers. Bug reported by Gabriel Verret. (V2.19-5)

14.8.1 Finite p-Groups

New Features:

- Homomorphisms with domain a p-group obtained from `pQuotient` can be defined by the action on the defining generators rather than the pc-generators.

Improvements:

- The character degrees function for a p-group has been overhauled to increase speed. The algorithm used remains Slattery's 1986 algorithm. The algorithm has proved useful as a means of finding the number of conjugacy classes of a p-group.
- The standard presentation routine for a 2-group has been improved, with a 5-fold speed-up for the hardest examples for the order 256, for instance.
- Testing of isomorphism of p-groups has improved in speed by a factor of 2 for previously hard examples (for 2-groups this can be as high as a 10-fold improvement).
- Computing the automorphism group of a p-group has been improved, particularly for groups of large p-class.
- Computing descendants of a p-group, particulary of large p-class, has been improved.
- A new parameter `MaxAuts` has been added to the intrinsics `GeneratepGroups` and `Descendants` to compute maximally automorphic descendants. The code was supplied by Eamonn O'Brien. (V2.19-3)

Changes:

- Magma level printing of `GrpPC` and their elements has been changed to the compact presentation format for groups, and to sequence of integers for group elements.

Bug Fixes:

- A bug with storing the incorrect subgroup lattice of a `GrpPC` has been fixed.

14.9 Finitely Presented Abelian Groups

Bug Fixes:

- A crash in `HasComplement` for type `GrpAb` has been fixed, by installing a new algorithm from Derek Holt. Bug reported by Tim Dokchitser. (V2.19-2)

14.10 Databases of Groups

Bug Fixes:

- The database of irreducible quasisimple matrix groups having degrees up to 100 has had a number of small errors corrected. In some cases these errors prevented a group from being read from the database. In addition, the group 3U35 in degree 48 is included for the first time. The problems were reported by Claus Fieker. The errors were corrected by Derek Holt; the missing representation of 3U35 was constructed by Allan Steel. (V2.19-5)

15 Lattices and Quadratic Forms

15.1 Quadratic Forms

Bug Fixes:

- A bug in `QuadraticClassGroupTwoPart` for integers with large even 2-valuation and an odd number prime factors 3 mod 4 was fixed. Reported by J. Stankewicz.

15.2 Lattices

Bug Fixes:

- A bug with vector lengths with `SuccessiveMinima` was fixed. The problem was that when the minimal vectors do not generate the lattice (but only a sublattice), the internal code would incorrectly update the norms. Reported by the Dortmund group of R. Scharlau, M. Juergens, M. Zimmermann.
- A bug with the returned vectors in `ClosestVectors` was fixed. On a technical basis, the error would occur for lattices whose Gram matrix would have the maximum entry on the diagonal *increase* when LLL was applied, as the LLL was then delayed, and its effect on the cosets was not correctly handled. For most lattices, this delayed LLL would simply return an identity matrix (on the cosets), and so caused no problems. Reported by the Dortmund group of R. Scharlau, M. Juergens, M. Zimmermann.
- A bug with `WittInvariants` and allied intrinsics was fixed, where the bad primes were computed from the determinant, and when the matrix was over the rationals, this need not be correct. Reported by Stefan Hoeppner (V2.19-3).
- A crash when computing the automorphism group of a lattice has been fixed. The bug was due to the `ShortestVectorsMatrix` of the lattice being possibly incorrect when it is very large (64 bit problem). This problem has been fixed. Bug reported by G. Nebe (V2.19-10).

16 Linear Algebra and Module Theory

16.1 Matrices

New Features:

- Magma V2.20 (to be released in V2.20-3 onwards) contains new code for matrix multiplication on systems with an NVIDIA Tesla GPU with CUDA support. Currently there are three classes of finite fields which are supported:
  - GF(p) (machine int size): The speedup for large matrices is about 15 to 20 compared with a state-of-the-art 3.2GHz Intel Ivy Bridge processor. Several characteristic-zero modular algorithms of Magma map to this representation and thus gain speedups also.
  - GF(2): The speedup for large matrices is about 3 to 5 compared with a 3.2GHz Intel Ivy Bridge processor.
  - GF(2^k): this basically depends on the code for GF(2) and yields similar speedups.

Since most linear algebra operations map to matrix multiplication in Magma, the speedups benefit general linear algorithms.

- Matrix multiplication and echelonization over GF(2^k) has been sped up for $1 < k \leq 5$.
- The basic matrix multiplication algorithm over the small finite fields GF(p) for $p = 3, 5, 7$ has been sped up (typically by a factor of 1.5 to 2). This leads to fundamental speedups for all linear algebra computations over such finite fields.
- The algorithm for transposing matrices over small finite fields has been sped up (because of the special packing, it is difficult to make this operation very fast).
- The algorithm for computing the Hermite Normal Form of matrices with some sparsity has been sped up.

Bug Fixes:

- Some crashes involving huge matrices over medium prime finite fields were fixed (V2.19-8).

16.2 Sparse Matrices

New Features:

- The Lanczos iterative algorithm for solving large sparse linear systems modulo a large integer M has been sped up. Furthermore, there is new multi-threaded parallel version of the algorithm which can be run on any system with multiple cores. If the algorithm is to be run on k cores, then the user should include the statement:

  ```
  SetNthreads(k);
  ```

  before calling the function `ModularSolution` (with the `Lanczos` parameter set to `true`). A typical speedup achieved is about $(0.7 \times k)$ for k cores (which is reasonable since the algorithm is not purely parallel and needs several synchronisation points in every step.

  As an example, computing the logarithms of the factor base for the finite field $K = \mathrm{GF}(2^{509})$ via the Function Field Sieve involves solving a sparse linear system with 425882 equations and 395878 unknowns modulo the largest prime divisor M of $2^{509} - 1$, which has 343 bits. On a 16-core 2.6GHz Intel Xeon node, the multi-threaded Lanczos algorithm takes 929241 core seconds (equivalent to 10.7 days) to solve the system, but only 82657 secs (22.9 hr) wall-clock time. This is a wall-clock time speedup by a factor of 11.2.
- The dense back end of the sparse Smith Normal Form algorithm has been improved.

16.3 Modules over Dedekind Domains

New Features:

- `IsZero` can now be applied to elements of modules of Dedekind domains (V2.19-9).
- The `DirectSum` of two modules or a sequence of modules can now be computed.

Bug Fixes:

- Compatibility of modules has been fixed so they can be correctly compared using `cmpeq` (V2.19-2).
- Construction of a module from a sequence of `ModRng` elements has been fixed (V2.19-10).

17 Linear Associative Algebras

17.1 Associative Algebras

New Features:

- For an associative algebra over a number field, the `MaximalOrder` can now be computed.
- For an associative algebra over a number field, `RestrictionOfScalars` computes an isomorphic algebra over the lower coefficient field, together with a bijective map.
- The `RestrictionOfScalars` of an order of an associative algebra over a number field can also be computed.
- The intrinsics `Discriminant` and `FactoredDiscriminant` are now provided for orders in general associative algebras.
- An intrinsic `ReducedDiscriminant` has been included.
- An improvement by M. Kirschmer to the algorithm for isomorphism testing for ideals in quaternion algebras over number fields has been included.

Changes and Removals:

- Some conventions have had to be changed regarding the `Discriminant` of orders in some types of algebras. The new convention is that `Discriminant` returns the determinant of the (non-reduced) trace form, except when the algebra is a quaternion algebra (of type `AlgQuat`) it returns the reduced discriminant. This changes the behaviour (only) for orders in non-quaternion algebras over $\mathbf{Q}$ created by the general `MaximalOrder` function. In addition, a new function `ReducedDiscriminant` is also provided where appropriate.

Bug Fixes:

- A fix has been made to the memory management in the powering of elements (V2.19-9).
- The intrinsic `Order` given a sequence of algebra elements and a sequence of ideals of the coefficient ring has been fixed for `AlgGrp`.

17.2 Basic Algebras

New Features:

- The package developed by Jon Carlson for computing the automorphism group and testing isomorphism of basic algebras has been improved.

17.3 Matrix Algebras

Changes and Removals:

- The code for constructing the basic algebra of a matrix algebra over a finite field has been improved in a number of ways with the result that it is now much faster in the case of matrix algebras having large degree.

17.4 Quaternion Algebras

New Features:

- An intrinsic `IsBass` has been implemented which determines whether an order O of a quaternion algebra is *Bass*, that is, has the property that every order which contains O is Gorenstein.

Bug Fixes:

- The function `DefiniteClassNumber(D, N)` which implementes a formula for the class number of an Eichler order in a definite quaternion algebra was wrong (or gave a runtime error) in many instances.

18 Representation Theory

18.1 Group Algebras

New Features:

- The code for constructing the basic algebra of a matrix algebra over a finite field has been improved in a number of ways with the result that it is now much faster in the case of matrix algebras having large degree. For example, it has been used to construct the basic algebras of the group algebra of the Mathieu group M_{23} in all finite characteristics.
- The package developed by Jon Carlson for computing the automorphism group and testing isomorphism of basic algebras has been improved.

18.2 Character Theory

Bug Fixes:

- A crash (runtime error) while computing `SchurIndices` has been fixed. Bug reported by Alex Bartel. (V2.19-4)
- A bug in `IsIrreducible` applied to a group character constructed by induction has been fixed. Bug reported by Derek Holt. (V2.19-7)