Summary of New Features in Magma V2.28

May 2023

1 Introduction

This document provides a terse summary of the new features released as part of Magma versions V2.28 (May 2023).

A small number of new features were exported in patch releases prior to the main release of V2.28 in May 2023 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 V2.27-x.

2 Highlights

Algebraic Geometry

- *Schemes*
  - For computing rational points on a scheme, a parameter `SkipSingularityCheck` has been added to the function `RationalPoints` so that the singularities check can be skipped at the beginning so that the function does not fail on singular inputs.

Arithmetic Geometry

- *Elliptic Curves Over* **Q**
  - The `SelmerGroup` algorithm has been improved so that it runs much faster on certain types of input.
- *Elliptic Curves Over Number Fields*
  - The function `MordellWeilGroup` for an elliptic curve defined over a number field where the coefficients are rational has been made more stable and efficient.

Arithmetic Fields

- *Class Groups of Number Fields*

  - A major new parallel version of the main algorithm to compute the class group of a number field or an order has been developed for V2.28. The algorithm is fully distributed, so one can use all cores on the local node, but also add workers from other nodes.

    The relation matrix for the class group algorithm is now also represented internally by the sparse matrix type, which allows much larger factor bases than could be handled previously (e.g., over 30000). This is particularly important when there are many parallel workers, each of which handles its own local relation matrix.

- *Quadratic Fields*

  - The algorithm to compute a fundamental unit of a real quadratic field (function `FundamentalUnit`) has been greatly improved so that is typically much faster than previously. Crashes for certain types of input field have also been fixed.

- *p-adic Rings and their Extensions*

  - A `SeriesPrinting` attribute has been added to the p-adic rings and fields and their extensions. When this attribute has been set to `true` for a ring or field, elements of that ring or field will be printed as a series in the uniformizing element.

Basic Rings and Fields

- *Rational Field*
  - A new function `HJContinuedFraction(r)` has been added. This takes a rational r and returns the sequence C of partial quotients of the Hirzebruch-Jung continued fraction expansion of r. An inverse function `HJContinuedFractionValue(C)` has been added, which takes such an expansion C and returns the corresponding rational number r.
- *Finite Fields*
  - Arithmetic has been significantly sped up for computations in the fields GF(p^2) for large p.
  - The function `IrreduciblePolynomial(GF(p), n)` has been improved to return a binomial polynomial always for large p when possible; certain cases had been missed previously.
  - The method for storing finite fields has been greatly improved so that a slowdown is now avoided when very many fields of different characteristics are created at the same time.
- *Polynomial Rings*
  - Parallel versions of both the Karatsuba and asymptotically-fast Cantor algorithms have been developed for fast multiplication of high-degree univariate polynomials defined over finite fields of characteristic 2. This speeds up arithmetic for very high-degree polynomials defined over GF(2) but also for medium to high-degree polynomials defined over larger finite fields of characteristic 2. It also can speed up the factorisation of sufficiently high-degree polynomials over any finite field of characteristic 2.

    As an example, on a dual Intel Xeon Gold 6146 (3.20GHz) node, the product of two random degree-1048576 (2^{20}) polynomials with coefficients in the finite field GF(2^{1000}) takes 86.3 seconds (single thread) or 10.2 seconds (24 threads); parallel speedup is 8.5.

Coding Theory

- *Linear Codes over Finite Fields*
  - Compared to previous released versions of the Magma BKLC database, 469 missing codes have been added, and the lower bound on the minimum distance has been improved for 1802 codes.

Commutative Algebra

- *Gröbner Bases*

  - The Faugere F_4 algorithm has been greatly sped up (in both serial and parallel modes) in the case that the base field is GF(p), where for p is in the range of 23.5 to 63 bits.
  - The Faugere F_4 algorithm has been sped up in the case that the base field is the rational field or a number field.
  - The parallel version of the dense variant of Faugere F_4 algorithm in 'HFE' mode, (selected by `SetNthreads` and the `HFE` parameter) has been greatly improved as follows:

    1. The critical pair management phase has been fully parallelised for the first time (including the handling of the field polynomials).
    2. The parallelisation of the linear algebra phase has been improved, by use of more asynchronous techniques.

    Several computations in this mode are thus significantly faster than before. As an example, a new record has been reached for the 80-variable Patarin HFE Challenge 1, which can now be solved in 7.1 seconds on a compact Dell Optiplex 7000 desktop containing a single Intel Core i7-12700 processor, with 8 threads selected (which reach up to 4.5GHz speed).

- *Multivariate Polynomial System Solving*

  - The function to compute the variety of a an ideal I of a multivariate polynomial ring, defined over a finite field K of size q, now has a new parameter `Eval := e`, for a small positive integer e. This selects a new algorithm which works by evaluating the input polynomials of I at the last e variables to all possible constants in K (so there are q^e cases enumerated) and then computing the variety of each simplified ideal by Gröbner basis techniques, and finally combining all the results. A fully parallel distributed version of this evaluation-based algorithm has also been developed.

- *Ideals of Multivariate Polynomial Rings*

  - The algorithm to compute the primary or radical decomposition of a zero-dimensional ideal defined over rational function field over a finite field has been significantly improved for harder cases.

Group Theory

- *Finite Groups*
  - The computation of maximal subgroups has been extended by Derek Holt so that it now works for groups having composition factors of type PSU$(5, q)$, G2(q), Sz(q), Ree(q), 3D4(3), and Co1.
- *Permutation Groups*
  - A new algorithm for computing the normaliser of a subgroup of a symmetric permutation group was developed by Stephan Elsenhans. This is much faster than the current method, particularly for permutation groups of small degree.
- *Matrix Groups*
  - A major new version of the `CompositionTree` code for computing with very large finite permutation and matrix groups was provided by Eamonn O'Brien.

Linear Algebra and Module Theory

- *Linear Algebra over Finite Fields*
  - Matrix multiplication over a class of finite fields GF(p^e) where $p > 7$, e is composite and $p^e > 2^{20}$ has been improved (including with parallelisation).
  - Matrix multiplication over GF(p) for medium-sized p has been improved in the AVX2 executable.
  - The recursive echelon algorithm (called when computing echelon forms, rank, determinant, inverse etc. for large matrices) has been improved for matrices defined over non-prime finite fields of medium size.
- *Linear Algebra over* **Z** *and* **Q**
  - The Smith Normal Form algorithm for dense integer matrices has been improved, and is particularly faster under parallelisation.
  - Some improvements have been made to nullspace computation for matrices over **Z** or **Q**, particularly when the nullity is small.
  - The computation of the nullspace of sparse matrices over the rational field has been sped up for certain types of input. This speeds up some types of computation with modular forms.
- *Lattices*
  - The function `Orthonormalize` has been greatly improved so that it is more stable. It also gives a correct informative error message if there are precision problems for a given real field (but such problems are much less likely to happen now).

Modular Forms

- *Modular Forms*
  - Some significant speedups have been achieved in the modular forms algorithms, in particular for computing newforms. For example, computing the newforms associated to the space of modular forms on $\Gamma_0(3025)$ of weight 4 over $\mathbf{Z}$ and with a character of order 8 is now about 15 times faster than with V2.27.

- *Modular Spaces*
  - Some significant speedups have been achieved in the algorithm to compute newform decompositions. For example, computing the newform decomposition of the cuspidal subspace of the modular symbols space of level 62208, weight 2 and with a character of order 8 is now about 3.5 times faster than with V2.27.

Representation Theory

- *KG-Modules*
  - A new intrinsic `LowDimensionalModules(G,K,n)`, developed by Derek Holt, computes the representations of isomorphism classes of KG-modules having dimension up to n, for a finite group G and finite field K.

System

- *Language*
  - A new function `NumberOfResults()` or `Nresults()` has been added, which returns the number of results expected by the caller of the current function or intrinsic (0 means that the result(s) will be printed).

3 Algebraic Geometry

3.1 Schemes

New Features:

- For computing rational points on a scheme, a parameter `SkipSingularityCheck` has been added to the function `RationalPoints` so that the singularities check can be skipped at the beginning so that the function does not fail on singular inputs.

Bug Fixes:

- An internal crash in `BaseScheme` called out of `Extend` for schemes has been fixed (V2.27-3).
- Error checking for `Dimension` of schemes has been clarified to reflect that it can only be computed for schemes over fields over which Gröbner bases can be computed (V2.27-6).
- A bug in `Extend` when alternate equations become undefined has been fixed (V2.27-7).
- Bugs in `Genus4GonalMap` and `Genus5GonalMap` have been fixed (V2.27-7).

3.2 Algebraic Curves

Bug Fixes:

- A failure in EightDescent for a certain class of curves has been fixed (V2.27-5).
- A bug in `LPolynomial(C)` applied to a certain class of curves has been fixed (V2.27-8).

4 Arithmetic Geometry

4.1 Elliptic Curves

4.1.1 Elliptic Curves over the Rational Field

New Features:

- The `SelmerGroup` algorithm has been improved so that it runs much faster on certain types of input.

Bug Fixes:

- A hang in `TwoSelmerGroup` for elliptic curves with very large coefficients has been fixed (V2.27-4).
- A crash in `TwoDescent` for elliptic curves has been fixed (V2.27-5).
- A crash in `IsogenousCurves` for certain types of input has been fixed (V2.27-7).

4.1.2 Elliptic Curves over Number Fields

New Features:

- The function `MordellWeilGroup` for an elliptic curve defined over a number field where the coefficients are rational has been made more stable and efficient.

Bug Fixes:

- A crash in `MordellWeilGroup` for an elliptic curve defined over a number field has been fixed (V2.27-8).

4.2 Hyperelliptic Curves and Jacobians

Bug Fixes:

- A bug in HeightConstant for Jacobians of hyperelliptic curves has been fixed (V2.27-7).
- A hang in Chabauty has been fixed (V2.27-7).
- A crash in HyperellipticCurveFromIgusaClebsch for certain types of input has been fixed (V2.27-7).
- A bug which caused `ReducedModel` to hang on some hyperelliptic curves has been fixed (V2.27-8).

5 Arithmetic Geometry (Modular Forms)

5.1 Modular Forms

New Features:

- Some significant speedups have been achieved in the modular forms algorithms, in particular for computing newforms. For example, computing the newforms associated to the space of modular forms on $\Gamma_0(3025)$ of weight 4 over $\mathbf{Z}$ and with a character of order 8 is now about 15 times faster than with V2.27.

5.2 Modular Symbols

Bug Fixes:

- Some significant speedups have been achieved in the algorithm to compute newform decompositions. For example, computing the newform decomposition of the cuspidal subspace of the modular symbols space of level 62208, weight 2 and with a character of order 8 is now about 3.5 times faster than with V2.27.

6 Arithmetic Fields (Global)

6.1 Algebraic Number Fields

New Features:

- A major new parallel version of the main algorithm to compute the class group of a number field or an order has been developed for V2.28. As usual, one simply selects the new parallel algorithm by first calling `SetNthreads(N);` where N is the number of desired threads and then the function `Classgroup` can be called on a number field or order as usual. One can also add workers from other nodes (so the algorithm is fully distributed); see the chapter on Parallelism in the Handbook.

  The relation matrix for the class group algorithm is now also represented internally by the sparse matrix type, which allows much larger factor bases than could be handled previously (e.g., over 30000). This is particularly important when there are many workers each of which handles its own local relation matrix.

Changes and Removals:

- The default algorithm to compute the class group of number fields does not now use the initial sieving phase by default; to force this on, one should give the parameter `Al := "Sieve"` to functions `ClassGroup`, `ClassNumber`, etc.

Bug Fixes:

- A crash in the computation of the absolute field of a number field with very large coefficients has been fixed. (V2.27-3)
- `RelativeField(F, F)` now returns a linear extension of `F` as documented. (V2.27-9)
- Testing some not yet known to be principal integral ideals of non maximal orders for principality has been fixed. (V2.27-8)
- An oversight in which a computed maximal order was not stored on the field has been fixed, resulting in no recomputation of known orders. (V2.27-8)
- The handling of empty sequences on the RHS of the `ideal<|>` constructor has been improved. (V2.27-6)
- A fix has been made to the computation of `GCD`s of elements of quotients of orders. (V2.27-6)

6.2 Quadratic Fields

New Features:

- The algorithm to compute a fundamental unit of a real quadratic field (function `FundamentalUnit`) has been greatly improved so that is typically much faster than previously. Crashes for certain types of input field have also been fixed.

6.3 Abelian Extensions of Number Fields

Bug Fixes:

- A bug in computing the `ArtinMap` of an abelian extension constructed from a map from the trivial group has been fixed. (V2.27-4)

6.4 Galois Groups

Bug Fixes:

- The intrinsic IsEasySnAn has been fixed in cases where the input polynomial has non-trivial denominators. (V2.27-9)
- A crash in the computation of Subfields and GaloisGroup of a relative extension of function fields has been fixed. (V2.27-7)
- The GeometricGaloisGroup computation now performs more checks on the values at which the input polynomial is specialised. (V2.27-6)

7 Arithmetic Fields (Local)

7.1 p-adic Rings and their Extensions

New Features:

- A SeriesPrinting attribute has been added to the p-adic rings and fields and their extensions. When this attribute has been set to true for a ring or field, elements of that ring or field will be printed as a series in the uniformizing element. (V2.27-4)

Bug Fixes:

- Enumerating the elements of an extension of a p-adic ring with precision 1 has been fixed to ensure there are no duplicates in the enumeration. (V2.27-7)
- p-adic rings constructed as a Completion of the integers were unintentionally returned as fields. (V2.27-5)

7.2 Newton Polygons

Bug Fixes:

- The input type to NewtonPolygon has been generalised to RngMPolEltGL. (V2.27-6)

8 Basic Rings and Fields

8.1 Integer Ring

Bug Fixes:

- Some hangs in the ECPP primality proving algorithm have been fixed.
- A message dumped by ECPP has been suppressed (V2.27-6).

8.2 Rational Field

New Features:

- A new function `HJContinuedFraction(r)` has been added. This takes a rational r and returns the sequence C of partial quotients of the Hirzebruch-Jung continued fraction expansion of r. An inverse function `HJContinuedFractionValue(C)` has been added, which takes such an expansion C and returns the corresponding rational number r.

8.3 Finite Fields

New Features:

- Arithmetic has been significantly sped up for computations in the fields GF(p^2) for large p.
- The function `IrreduciblePolynomial(GF(p), n)` has been improved to return a binomial polynomial always for large p when possible; certain cases had been missed previously.
- The method for storing finite fields has been greatly improved so that a slowdown is now avoided when very many fields of different characteristics are created at the same time.

Changes:

- For a finite field extension with the Zech representation but a non-primitive defining polynomial, the internal generator of the Zech-based field was not chosen uniquely; this been changed so that sorting involving elements of such a field now always behaves the same, irrespective of the random seed.

8.4 Real and Complex Fields

Bug Fixes:

- Some hangs in polynomial root finding over the complex field in the case that roots had very small imaginary part or were very close to each other have been fixed (V2.27-5, 2.27-6).

8.5 Polynomial Rings

New Features:

- Parallel versions of both the Karatsuba and asymptotically-fast Cantor algorithms have been developed for fast multiplication of high-degree univariate polynomials defined over finite fields of characteristic 2. This speeds up arithmetic for very high-degree polynomials defined over GF(2) but also for medium to high-degree polynomials defined over larger finite fields of characteristic 2. It also can speed up the factorisation of sufficiently high-degree polynomials over any finite field of characteristic 2. This feature is selected by first doing the procedure calls `SetParallelCantor(true);` and `SetNthreads(k);` to select the desired number of threads, and then doing the operations which use high-degree polynomial multiplication.

  As an example, on a dual Intel Xeon Gold 6146 (3.20GHz) node, the product of two random degree-1048576 (2^{20}) polynomials with coefficients in the finite field GF(2^{1000}) takes 86.3 seconds (single thread) or 10.2 seconds (24 threads); parallel speedup is 8.5.

Bug Fixes:

- A bad slowdown in resultant computation for some types of multivariate polynomials has been fixed (V2.27-5).
- The input type to `NewtonPolygon` has been generalised to `RngMPolEltGL` (V2.27-6).
- The input type to `CoefficientNumerator` and `CoefficientDenominator` has been generalised to `GenMPolBElt` (V2.27-6).
- `&*` when applied to polynomials over a non commutative ring has been fixed (V2.27-8).
- Crashes in univariate polynomial factorisation for very large degree and coefficient size have been fixed.

9 Coding Theory

9.1 Linear Codes over Finite Fields

New Features:

- Compared to previous released versions of the Magma BKLC database, 469 missing codes have been added, and the lower bound on the minimum distance has been improved for 1802 codes.

Bug Fixes:

- Some spurious printing in `IsIsomorphic` for codes has been fixed (V2.27-5).
- A crash involving the deletion of codes after `SymplecticDual` had been called has been fixed (V2.27-8).
- A bug in computing automorphisms of a linear code over a finite field with more than 2 elements has been fixed.
- The minimum distance of the code $[77, 46, 17]_8$ was incorrectly given as 18, due to a typo in the related publication. This code and derived codes have been corrected.

10 Commutative Algebra

10.1 Ideal Theory and Gröbner Bases

New Features:

- The Faugere F_4 algorithm has been greatly sped up (in both serial and parallel modes) in the case that the base field is GF(p), where for p is in the range of 23.5 to 63 bits.
- The Faugere F_4 algorithm has been sped up in the case that the base field is the rational field or a number field.
- The parallel version of the dense variant of Faugere F_4 algorithm in 'HFE' mode, (selected by `SetNthreads` and the `HFE` parameter) has been greatly improved as follows:

  1. The critical pair management phase has been fully parallelised for the first time (including the handling of the field polynomials).
  2. The parallelisation of the linear algebra phase has been improved, by use of more asynchronous techniques.

  Several computations in this mode are thus significantly faster than before. As an example, a new record has been reached for the 80-variable Patarin HFE Challenge 1, which can now be solved in 7.1 seconds on a compact Dell Optiplex 7000 desktop containing a single Intel Core i7-12700 processor, with 8 threads selected (which reach up to 4.5GHz speed).
- The `Variety(I)` function, for an ideal I of a multivariate polynomial ring defined over a finite field K of size q, now has a new parameter `Eval := e`, for a small positive integer e. This selects a new algorithm which works by evaluating the input polynomials of I at the last e variables to all possible constants in K (so there are q^e cases enumerated) and then computing the variety of each simplified ideal by Gröbner basis techniques, and finally combining all the results.

  A fully parallel distributed version of this evaluation-based algorithm has also been developed. As usual, one simply selects the new parallel algorithm by first calling `SetNthreads(N);` where N is the number of desired threads and then the function `Variety(I: Eval := e)` can be called, as above.
- The algorithm to compute the primary or radical decomposition of a zero-dimensional ideal defined over rational function field over a finite field has been significantly improved for harder cases.

Bug Fixes:

- Crashes in the multi-threaded F_4 algorithm with a large number of variables have been fixed (V2.27-2, V2.27-8).
- An incorrect result when computing the lexicographical Gröbner basis of a certain type of ideal over the integer ring has been fixed (V2.27-2).
- An bug (causing a hang or incorrect results) when computing Gröbner bases of certain types of ideal over medium-size prime finite fields has been fixed (V2.27-2).
- A slowdown in the F_4 algorithm for certain types of ideals has been fixed (V2.27-4).
- A hang in Gröbner basis computation over the rational field for a certain type of input has been fixed (V2.27-4).
- A crash in `PrimaryDecomposition` over rational function fields in higher characteristic (arising from computing divisors of points on schemes) has been fixed.
- A bug in `Kernel` for matrices defined over a field of fractions of an affine algebra has been fixed.

11 Group Theory

11.1 Finite Groups

New Features:

- The computation of maximal subgroups has been extended by Derek Holt so that it now works for groups having composition factors of type PSU$(5, q)$, G2(q), Sz(q), Ree(q), 3D4(3), and Co1.
- The new individual intrinsics `SzMaximals`, `ReeMaximals`, and `G2Maximals` are provided.
- The maximal subgroups of a permutation or matrix groups are now cached after being computed.
- New functions for finding permutation representations of extensions of modules by groups were developed.
- New intrinsics have been provided to calculate a permutation representation of a split extension. See `SplitExtension(GrpPerm, G, M)` where G is a group and M is a KG-module.
- The cohomology functions were revised so that they work when the module is the trivial abelian group with invar= [].

Bug Fixes:

- A missing check that N is normal in G for the call `Subgroups(G, N)` has been fixed (V2.27-4).
- A bug in `MinimalDegreePermutationRepresentation`, when handling extraspecial and symplectic type p-groups for odd p), has been fixed.
- A bug in `MaximalSubgroups`, in finding generators of PSL$(2, p^3)$, has been fixed.
- A bug in `MaximalSubgroups` when applied to CSU$(6, 3)$ was fixed.
- A bug when computing subgroup conjugacy in COPlus$(4, 4)$ has been fixed.
- A bug in the intrinsic `GroupName` has been fixed.

11.2 Matrix Groups over Finite Fields

New Features:

- A major new version of the `CompositionTree` code for computing with very large finite permutation and matrix groups was provided by Eamonn O'Brien.

Bug Fixes:

- A bug in `MinimalDegreePermutationRepresentation` has been fixed (V2.27-8).

11.3 Permutation Groups

New Features:

- A new algorithm for computing the normaliser of a subgroup of a symmetric permutation group was developed by Stephan Elsenhans. It can be invoked via `SymmetricNormaliser` with the parameter `Al := "Auto"`, while the old algorithm can be selected with the option `Al := "Backtrack"`. Transitive groups of degree 40 are sped up by a factor of 3 on average. Former hard cases of intransitive groups with very similar orbit groups are now handled much faster.

11.4 Finite Soluble Groups

Bug Fixes:

- A bug in `Complements` for certain types of soluble groups has been fixed.

12 Lattices and Quadratic Forms

12.1 Lattices

New Features:

- The function `Orthonormalize` has been greatly improved so that it is more stable. It also gives a correct informative error message if there are precision problems for a given real field (but such problems are much less likely to happen now).

Bug Fixes:

- An error in `GenusRepresentatives` has been fixed (V2.27-7).

12.2 Representation Theory

Changes and Removals:

- A crash in `CharacterTable` applied to soluble groups with very large number of classes has been fixed (V2.27-7).

13 Linear Algebra and Module Theory

13.1 Matrices

New Features:

- The computation of the nullspace of sparse matrices over the rational field has been sped up for certain types of input. This speeds up some types of computation with modular forms.
- Matrix multiplication over a class of finite fields GF(p^e) where $p > 7$, e is composite and $p^e > 2^{20}$ has been improved (including with parallelisation).
- Matrix multiplication over GF(p) for medium-sized p has been improved in the AVX2 executable.
- The Smith Normal Form algorithm for dense integer matrices has been improved, and is particularly faster under parallelisation.
- Some improvements have been made to nullspace computation for matrices over **Z** or **Q**, particularly when the nullity is small.
- The recursive echelon algorithm (called when computing echelon forms, rank, determinant, inverse etc. for large matrices) has been improved for matrices defined over non-prime finite fields of medium size.

Bug Fixes:

- A incorrect result for `Kernel` applied to matrices over quaternion algebras has been fixed (V2.27-2).
- An incorrect result in `FactoredCharacteristicPolynomial` for a special type of integer matrix has been fixed (V2.27-2).
- A bug when computing random matrices over the finite field `GF(2^32)` has been fixed (V2.27-3).
- A hang in matrix inverse over cyclotomic fields for non-invertible matrices has been fixed (V2.27-4).
- A bug in the algorithm for computing the Hermite Normal Form of a certain type of sparse integer matrix has been fixed (V2.27-6).
- A bug in `FrobeniusFormAlternating` where the result did not satisfy the divisibility condition has been fixed (V2.27-9).

13.2 Sparse Matrices

Bug Fixes:

- A crash in `SetEntry` when dynamically extending a sparse matrix while setting an entry to zero has been fixed.

14 Linear Associative Algebras

14.1 Matrix Algebras

Bug Fixes:

- A case in the function `PrimitiveIdempotents` where the result did not contain normal matrices has been fixed (V2.27-7).

14.2 Quaternion Algebras

Bug Fixes:

- An crash in `RightIdealClasses` for certain ideals of orders of quaternion algebra has been fixed (V2.27-2).

15 Representation Theory

15.1 Group Algebras

New Features:

- The error message has been made more informative in the case that coercion of a sequence into a group algebra A fails (if the vector representation is not used in A).

16 System

New Features:

- A new function `NumberOfResults()` or `Nresults()` has been added, which returns the number of results expected by the caller of the current function or intrinsic (0 means that the result(s) will be printed).

Bug Fixes:

- Some crashes in the `save/restore` feature running under Mac OS X have been fixed (V2.27-5).