Tame Genus Package

Joshua Maglione

Colorado State University
jmaglion@math.kent.edu

Peter A. Brooksbank

Bucknell University
pbrooksb@bucknell.edu

James B. Wilson

Colorado State University
James.Wilson@ColoState.Edu

Version 1.0
October 31, 2017

Copyright 2015–2017 Joshua Maglione, Peter A. Brooksbank, James B. Wilson

Contents

CHAPTER 1

Introduction

The goal of this package is to provide MAGMA [**BCP**] with high performance functionality for groups with tame genus. When appropriate, all intrinsics dealing with groups also work for tensors (`TenSpcElt`). This package includes intrinsics for automorphism and isomorphism computations, canonical labeling, and constructing random groups of genus 1 or 2 of exponent $\leq p^2$. Mathematical details of these algorithms can be found in [**BMW**].

Users will need to attach the latest versions of `eMAGma` [**MW**] and `StarAlge` [**BW**] to use all the features of this package.

Citing TameGenus. To cite the Tame Genus package, please use the following

Joshua Maglione, Peter A. Brooksbank, and James B. Wilson, *Tame Genus*, version 1.0, GitHub, 2017. https://github.com/algeboy/TameGenus.

For AMSRefs:

```
\bib{eMAGma}{misc}{
   author={Maglione, Joshua},
   author={Brooksbank, Peter A.},
   author={Wilson, James B.},
   title={Tame Genus},
   publisher={GitHub},
   year={2017},
   edition={version 1.0},
   note={\texttt{https://github.com/algeboy/TameGenus}},
}
```

1.1. Verbose Printing and Version

We have included intrinsics to allow for verbose printing. Currently, there is only one level of printing, so either it is on of off.

`SetVerbose(MonStgElt, RngIntElt) : ->`

`SetVerbose` is a built in Magma function, but with this package, this intrinsic accepts the string `"TameGenus"` and an integer in $\{0, 1\}$.

We have included an intrinsic to check which version of TameGenus you have attached in Magma.

`TameGenusVersion() : -> MonStgElt`

Returns the version number for the TameGenus package attached in Magma.

CHAPTER 2

Constructors

We introduce intrinsics to construct nonabelian groups of genus ≤ 2. We provide some algorithms to construct random groups, and we make no attempt at asserting any analysis of these algorithms. In particular, these random constructors may not select groups in a uniform distribution.

```
TGRandomGroup(q, n, g : parameters) : RngIntElt, RngIntElt, RngIntElt -> GrpPC
    Exponentp : BoolElt : true
```

Given $q = p^m$, $n > 0$, and $g > 0$, returns a p-group G with genus $\leq g$ and order $q^{n+g} = p^{m(n+g)}$. If q is not a prime, then the centroid of the group will be a proper field extension of $\mathbb{F}_p$, isomorphic to $\mathbb{F}_q$. If $\circ : V \times V \rightarrowtail W$ is the commutation tensor of G, then $\dim_{\mathcal{C}(\circ)}(V) = n$ and $\dim_{\mathcal{C}(\circ)}(W) = g$, where $\mathcal{C}(\circ)$ is the centroid of $\circ$. The algorithm is based on the Universal Coefficients Theorem, see [**LGM**, Chapter 9] the statement and proof. There is one optional parameter: `Exponentp`.

> `Exponentp`**:** The default is set to `true`. Set to false if you want the returned group to have exponent $\leq p^2$.

```
RandomGenus2Group(q, d : parameters) : RngIntElt, [RngIntElt] -> GrpPC
    Exponentp : BoolElt : true
```

Given $q = p^m$ and $d = [d_1, \dots, d_k]$, where $d_i > 0$, it returns a genus 2 group G whose commutation tensor $\circ : V \times V \rightarrowtail W$ has centroid $\mathcal{C}(\circ) \cong \mathbb{F}_q$ and whose $\perp$-decomposition has blocks of dimensions $d_1, \dots, d_k$ (over $\mathcal{C}(\circ)$). If $d_i = 1$, then this increases the dimension of the radical by 1 (over $\mathcal{C}(\circ)$). There is one optional parameter: `Exponentp`.

> `Exponentp`**:** The default is set to `true`. Set to false if you want the returned group to have exponent $\leq p^2$.

```
RandomGenus1Group(q, d, r : parameters) : RngIntElt, RngIntElt, RngIntElt -> GrpPC
    Exponentp : BoolElt : true
```

Given $q = p^m$, $d > 0$, and $r \geq 0$, it returns a group G with genus 1 and of order q^{2d+r+1}. The center of G has order q^{r+1}, and G is dm-generated. There is one optional parameter: `Exponentp`.

> `Exponentp`**:** The default is set to `true`. Set to false if you want the returned group to have exponent $\leq p^2$.

```
Genus2Group(f) : RngUPolElt -> GrpPC
Genus2Group(f) : RngMPolElt -> GrpPC
```

Given either a univariate or homogeneous multivariate polynomial in x (and y), it returns a group G whose tensor from communtation has a Pfaffian equivalent to f.

2.1. Examples

CHAPTER 3

Automorphism groups

We have two intrinsics for constructing automorphism groups: one with group inputs and the other with tensor inputs.

```
TGAutomorphismGroup(G : parameters) : GrpPC -> GrpAuto
    Cent : BoolElt : true
    Method : RngIntElt : 0
    Order : BoolElt : false
```

Given a p-group G, of class 2, exponent p, and genus ≤ 2, this returns $\mathrm{Aut}(G)$. This currently does not work with $p = 2$. This intrinsic provides three optional parameters: `Cent`, `Method`, and `Order`.

- `Cent`: This parameter is a flag for the algorithm to rewrite the tensor over its centroid. In order to rewrite over the centroid, we assume the centroid is a local ring, see [**MW**] for details. The default is set to `true`, so by default, the algorithm computes the centroid and rewrites the tensor over the residue field. If you know the centroid is trivial, set `Cent` to `false` to save time. WARNING: if the centroid is a proper field extension of $\mathbb{F}_p$, then the full automorphism group may not be constructed (this is a bug we will fix in the near future).
- `Method`: The default is 0, and it accepts any input from $\{0, 1, 2\}$. This will set the method for handling the sloped part of the tensor. If you want to use the adjoint-tensor method, set `Method` to 1, and if you want to use the Pfaffian method, set `Method` to 2. The default will try to find the optimal method based on the input; this is not optimally tuned.
- `Order`: Finally, `Order` is included so you have the order of the returned automorphism group. The data structure of `GrpAuto` makes it hard to compute the order of the group. However, when computing $\mathrm{Aut}(G)$, we construct a (linear) representation of the group. We can take advantage of the LMG functions [**BHLGO**] in Magma to compute the order of the representation. The default is set to `false`. WARNING: if set to `true` this step could take the majority of the computation time.

```
TGPseudoIsometryGroup(T : parameters) : TenSpcElt -> GrpMat
    Cent : BoolElt : true
    Method : RngIntElt : 0
```

Given an alternating tensor $T : V \times V \rightarrowtail W$, where V and W are $\mathbb{F}_q$-vector spaces, returns the pseudo-isometry group $\Psi\mathrm{Isom}(T) \leq \mathrm{GL}(V) \times \mathrm{GL}(W)$. This currently does not work for even q. This intrinsic provides two optional parameters: `Cent` and `Method`.

- `Cent`: This parameter is a flag for the algorithm to rewrite the tensor over its centroid. In order to rewrite over the centroid, we assume the centroid is a local ring, see [**MW**] for details. The default is set to `true`, so by default, the algorithm computes the centroid and rewrites the tensor over the residue field. If you know the centroid is trivial, set `Cent` to `false` to save time. WARNING: if the centroid is a proper field extension of $\mathbb{F}_q$, then the full pseudo-isometry group may not be constructed (this is a bug we will fix in the near future).
- `Method`: The default is 0, and it accepts any input from $\{0, 1, 2\}$. This will set the method for handling the sloped part of the tensor. If you want to use the adjoint-tensor method, set `Method` to 1, and if you want to use the Pfaffian method, set `Method` to 2. The default will try to find the optimal method based on the input; this is not optimally tuned.

3.1. Examples

We list some examples to illustrate the functionality of the automorphism intrinsics.

CHAPTER 4

Isomorphisms

```
TGIsIsomorphic(G, H : parameters) : GrpPC, GrpPC -> BoolElt
    Cent : BoolElt : true
    Constructive : BoolElt : true
    Method : RngIntElt : 0
```

Given class 2, exponent p, p-groups G and H of genus ≤ 2, decides if $G \cong H$. If $G \cong H$, then an isomorphism is provided unless specified otherwise. Currently this does not work for $p = 2$. There are three optional parameters: `Cent`, `Constructive`, and `Method`.

- `Cent`: This parameter is a flag for the algorithm to rewrite the tensors over their centroids. In order to rewrite over the centroid, we assume the centroid is a local ring, see [**MW**] for details. The default is set to `true`, so by default, the algorithm computes the centroid and rewrites the tensor over the residue field. If you know the centroid is trivial, set `Cent` to `false` to save time. WARNING: if the centroid is a proper field extension of $\mathbb{F}_p$, then the full coset of (potential) isomorphisms are not fully exhausted. That is, it is possible the algorithm concludes `false` when $G \cong H$ (this is a bug we will fix in the near future).
- `Constructive`: The default is `true`. Set to `false` if you do not want the algorithm to explicitly construct an isomorphism.
- `Method`: The default is 0, and it accepts any input from $\{0, 1, 2\}$. This will set the method for handling the sloped part of the tensor. If you want to use the adjoint-tensor method, set `Method` to 1, and if you want to use the Pfaffian method, set `Method` to 2. The default will try to find the optimal method based on the input; this is not optimally tuned.

```
TGIsPseudoIsometric(T, S : parameters) : TenSpcElt, TenSpcElt -> BoolElt
    Cent : BoolElt : true
    Constructive : BoolElt : true
    Method : RngIntElt : 0
```

Given alternating tensors $T, S : V \times V \rightarrowtail W$, where V and W are $\mathbb{F}_q$-vector spaces, it decides if T is pseudo-isometric to S. If T is pseudo-isometric to S, then a pseudo-isometry is provided (as an element of $\mathrm{GL}(V) \times \mathrm{GL}(W)$) unless specified otherwise. Currently this does not work for even q. There are three optional parameters: `Cent`, `Constructive`, and `Method`.

- `Cent`: This parameter is a flag for the algorithm to rewrite the tensors over their centroids. In order to rewrite over the centroid, we assume the centroid is a local ring, see [**MW**] for details. The default is set to `true`, so by default, the algorithm computes the centroid and rewrites the tensor over the residue field. If you know the centroid is trivial, set `Cent` to `false` to save time. WARNING: if the centroid is a proper field extension of $\mathbb{F}_q$, then the full coset of (potential) pseudo-isometries are not fully exhausted. That is, it is possible the algorithm concludes `false` when T is pseudo-isometric to S (this is a bug we will fix in the near future).
- `Constructive`: The default is `true`. Set to `false` if you do not want the algorithm to explicitly construct an isomorphism.
- `Method`: The default is 0, and it accepts any input from $\{0, 1, 2\}$. This will set the method for handling the sloped part of the tensor. If you want to use the adjoint-tensor method, set `Method` to 1, and if you want to use the Pfaffian method, set `Method` to 2. The default will try to find the optimal method based on the input; this is not optimally tuned.

4.1. Examples

We list some examples to illustrate the functionality of the isomorphism intrinsics.

CHAPTER 5

Canonical Labeling

```
Genus(G) : GrpPC -> RngIntElt
Genus(T) : TenSpcElt -> RngIntElt
```

Given a p-group G, returns the genus of G. That is, the dimension of $[G, G]$ over its centroid. For a tensor T, it returns the dimension of its image over its centroid.

```
Genus2Signature(G) : GrpPC -> List
Genus2Signature(T) : TenSpcElt -> List
```

Given a genus 2 group or tensor, returns the canonical genus 2 signature. The list has two parts. The first entry is a sequence of odd integers corresponding to the dimensions of the flat indecomposable spaces. The second entry is a list of sequences in $\mathbb{F}_q$, the field the given data is defined over. The sequences in the second entry are the coefficients of the homogeneous polynomial in x and y in degree d:

$$c_1x^d + c_2x^{d-1}y + c_3x^{d-2}y^2 + \cdots + c_{d+1}y^d.$$

Because this is a canonical label, two groups (or tensors) are isoclinic (or pseudo-isometric) if, and only if, their genus 2 signatures are equal.

Currently, this does not work for even q. WARNING: if the centroid is a proper field extension of $\mathbb{F}_q$, the field the data is defined over, then this is not a canonical label. That is, two groups can be isomorphic but have two different genus 2 signatures. This is a bug we will fix in the near future.

5.1. Examples

We list some examples to illustrate the functionality of these intrinsics.

Bibliography

[BCP] Wieb Bosma, John Cannon, and Catherine Playoust, *The Magma algebra system. I. The user language*, J. Symbolic Comput. **24** (1997), no. 3-4, 235–265. Computational algebra and number theory (London, 1993). MR1484478

[BMW] Peter A. Brooksbank, Joshua Maglione, and James B. Wilson, *A fast isomorphism test for groups whose Lie algebra has genus 2*, J. Algebra **473** (2017), 545–590. MR3591162

[BW] Peter A. Brooksbank and James B. Wilson, *Star Algebra*, GitHub, 2017. `https://github.com/algeboy/StarAlge`.

[BHLGO] Henrik Bäärnhielm, Derek F. Holt, C. R. Leedham-Green, and E. A. O'Brien, *The Large Matrix Group Package*, Magma, 2017. `http://magma.maths.usyd.edu.au/magma/handbook/text/680`.

[LGM] C. R. Leedham-Green and S. McKay, *The structure of groups of prime power order*, London Mathematical Society Monographs. New Series, vol. 27, Oxford University Press, Oxford, 2002. Oxford Science Publications. MR1918951

[MW] Joshua Maglione and James B. Wilson, *Experimental Multilinear Algebra Group*, version 1.2.3, GitHub, 2017. Contributions from Peter A. Brooksbank, `https://github.com/algeboy/eMAGma`.