UnitLib

The library of normalized unit groups of modular group algebras

Version 2.0

August 2006

Alexander Konovalov
Elena Yakimenko

Alexander Konovalov — Email: konovalov@member.ams.org
— Homepage: http://homepages.vub.ac.be/~okonoval/
— Address: Department of Mathematics
Zaporozhye National University
Zaporozhye, 69063 Ukraine
Current address:
Department of Mathematics
Vrije Universiteit Brussel
Pleinlaan 2, Brussels
B-1050 Belgium

Elena Yakimenko — Email: k-algebra@zsu.zp.ua
— Address: Department of Mathematics
Zaporozhye National University
Ul.Zhukovskogo, 66, Zaporozhye
69600 Ukraine

Abstract

The UnitLib package extends the LAGUNA package and provides the library of normalized unit groups of modular group algebras of all finite p-groups of order not greater than 243 over the field of p elements.

Copyright

Acknowledgements

We are very grateful to the members of the GAP team: Thomas Breuer, Stefan Kohl, Frank Lübeck for useful suggestions.

We acknowledge very much the Centre for Interdisciplinary Research in Computational Algebra of the University of St Andrews for possibility to use its computer facilities.

The main amount of computations was performed on the Computational Cluster of Kiev National Taras Shevchenko University, created with the support of Intel corporation (http://www.cluster.kiev.ua).

Contents

Chapter 1

Introduction

1.1 General aims

Let KG be a group algebra of a finite p-group G over the field K of characteristic p, and let $V(KG)$ be the normalized unit group of KG. The pc-presentation of the group $V(KG)$ can be computed using the GAP package LAGUNA (http://ukrgap.exponenta.ru/laguna.htm), but for groups of orders 64 and bigger this computation will already take a lot of time.

The UnitLib package is an extension of the LAGUNA package that is aimed on this problem. It contains the library of normalized unit groups of modular group algebras of finite p-groups over the field of p elements. This allows the user to retrieve the pre-computed group from the library instead of the long-time computation. The group created with UnitLib will have the same properties and attributes as the one computed with LAGUNA.

The current version of UnitLib provides the library of normalized unit groups $V(KG)$ for all p-groups of order not greater than 243. If you need to work with groups of bigger orders, please write to the authors, because we may already have them computed or can compute them for you.

1.2 Installation and system requirements

UnitLib is designed for GAP4.4 and no compatibility with previous releases of GAP4 is guaranteed.

Libraries of normalized unit groups of groups of orders less than 243, except for the order 128, will be available in any operating system.

The library for groups of order 128 was compressed using `gzip` program and, therefore, will be available only in UNIX-type systems (including UNIX-installation in Mac OS X and Cygwin installation in Windows).

To work with the library for groups of order 243 you will also need the `Curl` program (http://curl.haxx.se) to retrieve the data from the UnitLib homepage, and the GAP package Qaos (http://www.gap-system.org/Packages/qaos.html) which provides the GAP function `Curl` to work with `cURL`.

If you need to work with groups of order 128 or 243 in Windows environment or you can not use `Curl`, please write to the authors. We will be able to give you a version of UnitLib with locally stored non-compressed data.

Because the UnitLib is an extension of the LAGUNA package, you must have the LAGUNA package installed. You can obtain it from the GAP homepage or from its homepage http://ukrgap.exponenta.ru/laguna.htm.

To use the UnitLib online help it is necessary to install the GAP4 package GAPDoc by Frank Lübeck and Max Neunhöffer, which is available from the GAP homepage or from http://www.math.rwth-aachen.de/~Frank.Luebeck/GAPDoc/.

UnitLib is distributed in standard formats (zoo, tar.gz, tar.bz2, -win.zip) and can be obtained the GAP homepage or from http://homepages.vub.ac.be/~okonoval/unitlib.htm. To unpack the archive unitlib-2.0.zoo you need the program unzoo, which can be obtained from the GAP homepage http://www.gap-system.org/ (see section 'Distribution'). To install UnitLib, copy this archive into the pkg subdirectory of your GAP4.4 installation. Then the subdirectory unitlib containing the package will be created in the pkg directory after the command unzoo -x unitlib-2.0.zoo

Chapter 2

UnitLib functions

Since the main purpose of UnitLib is the storage of large amount of data, it has only two main user functions: to read the description of $V(KG)$ for the given catalogue number of G in the Small Groups Libary of the GAP system, and to save the description of $V(KG)$ if the user would like to store it for further usage for the group that is not contained in the library.

Examples below contain some functions from the LAGUNA package [BKRS], see their description in the LAGUNA manual.

To use the UnitLib package first you need to load it as follows:

Example

```
gap> LoadPackage("unitlib");
------------------------------------------------------------------------------
Loading  UnitLib 1.0 (Library of normalized unit groups of modular group algebras)
by Alexander Konovalov (http://homepages.vub.ac.be/~okonoval/) and
   Elena Yakimenko (k-algebra@zsu.zp.ua).
------------------------------------------------------------------------------
true
gap>
```

In case of non-UNIX system, a warning will be displayed about the non-availability of the library of normalized unit groups for groups of order 128.

2.1 MainFunctions

2.1.1 PcNormalizedUnitGroupSmallGroup

◊ `PcNormalizedUnitGroupSmallGroup( s, n )` (function)

Returns: PcGroup

Let `s` be a power of prime p and `n` is an integer from `[ 1 .. NrSmallGroups(s) ]`. `PcNormalizedUnitGroupSmallGroup(s,n)` returns the normalized unit group $V(KG)$ of the modular group algebra KG, where G is `SmallGroup(s,n)` and K is a field of p elements.

Example

```
gap> PcNormalizedUnitGroupSmallGroup(128,161);
<pc group of size 170141183460469231731687303715884105728 with 127 generators>
```

The result returned by PcNormalizedUnitGroupSmallGroup will be equivalent to the following sequence of commands:

Example

```
gap> G := SmallGroup( s, n );
gap> p := PrimePGroup( G );
gap> K := GF( p );
gap> KG := GroupRing( K, G );
gap> PcNormalizedUnitGroup( KG );
```

Nevertheless, PcNormalizedUnitGroupSmallGroup is not just a shortcut for such computation. It reads the description of the normalized unit group from the UnitLib library and then reconstructs all its other necessary attributes and properties. Thus, if you would like to obtain the group algebra KG or the field K and the group G, you should extract them from $V(KG)$, which should be constructed first.

Example

```
gap> V:=PcNormalizedUnitGroup(GroupRing(GF(2),SmallGroup(8,3)));
<pc group of size 128 with 7 generators>
gap> V1:=PcNormalizedUnitGroupSmallGroup(8,3);
<pc group of size 128 with 7 generators>
gap> V1=V;     # two isomorphic groups but not identical objects
false
gap> IdGroup(V)=IdGroup(V1);
true
gap> IsomorphismGroups(V,V1);
[ f1, f2, f3, f4, f5, f6, f7 ] -> [ f1, f2, f3, f4, f5, f6, f7 ]
gap> KG:=UnderlyingGroupRing(V1);  # now the correct way
<algebra-with-one over GF(2), with 3 generators>
gap> V1=PcNormalizedUnitGroup(KG); # V1 is an attribite of KG
true
gap> K:=UnderlyingField(KG);
GF(2)
gap> G:=UnderlyingGroup(KG);
<pc group of size 8 with 3 generators>
```

If the order s is not a power of prime, an error message will appear. If s is bigger than 128, you will get a warning telling that the library does not contain $V(KG)$ for G of such order, and you can use only data stored by yourself in your unitlib/userdata directory with the help of the function SavePcNormalizedUnitGroup (2.1.2).

2.1.2 SavePcNormalizedUnitGroup

◊ SavePcNormalizedUnitGroup(G) (property)

Returns: true

Let G be a finite p-group of order s from the Small Groups Library of the GAP system, constructed with the help of SmallGroup(s,n). Then SavePcNormalizedUnitGroup(G) creates the file with the name of the form us_n.g in the directory unitlib/userdata, and returns true if this file was successfully generated. This file contains the description of the normalized unit group $V(KG)$ of the group algebra of the group G over the field of p elements.

If the order of G is greater than 128, after this you can construct the group $V(KG)$ using PcNormalizedUnitGroupSmallGroup (2.1.1) similarly to the previous section. The preliminary warning will be displayed, telling that for such orders you can use only those groups that were already computed and saved by you to the unitlib/userdata directory. If there will be no such file there, you will get an error message, otherwise the computation will begin.

If the order of G is less or equal to 128, then the file will be created in the unitlib/userdata directory, but UnitLib will continue to use the file with the same name from the appropriate directory in unitlib/data. You can compare these two files to make it sure that they are the same.

WARNINGS:

1. It is important to apply this function to the underlying group G and not to the normalized unit group $V(KG)$.

2. The user should use as an argument only groups from the Small Groups Library of the GAP system, constructed with the help of SmallGroup(s,n), otherwise the consistency of data may be lost.

Example

```
gap> SavePcNormalizedUnitGroup( SmallGroup( 256, 56092 ) );
true
gap> PcNormalizedUnitGroupSmallGroup( 256, 56092 );
WARNING : the library of V(KG) for groups of order
256 is not available yet !!!
You can use only groups from the unitlib/userdata directory
in case if you already computed their descriptions
(See the manual for SavePcNormalizedUnitGroup)

Description of V(KG) for G=SmallGroup(256,
56092) accepted, started its generation
<pc group of size
57896044618658097711785492504343953926634992332820282019728792003956564819968
  with 255 generators>
```

2.2 Service tools

2.2.1 UNITLIBBuildManual

◊ UNITLIBBuildManual( ) (function)

This function is used to build the manual in the following formats: DVI, PDF, PS, HTML and text for online help. We recommend that the user should have a recent and fairly complete TEX distribution. Since UnitLib is distributed together with its manual, it is not necessary for the user to use this function. Normally it is intended to be used by the developers only. This is the only function of UnitLib which requires UNIX/Linux environment.

2.2.2 UNITLIBBuildManualHTML

◊ UNITLIBBuildManualHTML( ) (function)

This fuction is used to build the manual only in HTML format. This does not depend on the availability of the TEX installation and works under Windows and MacOS as well. Since UnitLib is distributed together with its manual, it is not necessary for the user to use this function. Normally it is intended to be used by the developers only.

Chapter 3

Details of implementation

In this chapter we describe the approach used to store the normalized unit group of the group algebra in the library, and to reconstruct the group $V(KG)$ from the stored information.

3.1 Saving the data

To compute the pc-presentation of the normalized unit group of the modular group algebra of a finite p-group we used the function `PcNormalizedUnitGroup` from the LAGUNA package. It uses the algorithm descibed in [Bov98].

When this group is computed, the main idea is to use GAP function `CodePcGroup` that returns the code for the polycyclic generating sequence of the group, and then to create the group from this code using the GAP function `PcGroupCode`.

The resulting code could be very long, and to compress it we used the GAP function `HexStringInt` than returns a string that represents the code with hexa-decimal digits (using A-F as digits 10-15). The inverse translation then can be performed with the GAP function `IntHexString`. This allowed to save almost 20 MB of space on groups of order 128 and reduce the total size of their database to 90 MB.

For groups of order 128 we decided to compress the library with the `gzip` program, and then uncompress each file "on fly" when it is requested. This allowed us to reduce the size of their part in the library from 90 to 12 MB, which is already quite reasonable. Of course, there is some little overhead arising from the uncompression and subsequent translation from hexa-decimal notation, but it is neglectible comparatively with the total time of the computation of $V(KG)$ from scratch.

There is one more thing that needs to be stored together with this code to make it sure that we will correctly indentify the underlying group G of the group algebra KG with its image in the pc-presentation of the normalized unit group $V(KG)$.

The group G is extracted from the GAP Small Groups Library, so it is always the same, unless its description in the library will be changed (and it will be an important task of UnitLib maintaner to update the package!), and here we are safe from inconsistencies.

But the next stage is the computation of generators of the normalized unit group $V(KG)$, and the first step is the dimension basis of the group G, that can be computed using the LAGUNA function `DimensionBasis`. To avoid the influence of possible changes in GAP or usage of random methods, we store (in compacted form) the information about the dimension basis of G in the UnitLib.

All further procedures are implemented inside the LAGUNA package, and their result is uniquely determined and predictable.

For most groups all information is stored in a single file. However, this is not the case for groups of order 243, where we have about 30 MB of data for 67 groups. For these groups we provide a solution on the base of Web-services. The information about the dimension basis is stored locally, while the codes for polycyclic generating sequences are available from the UnitLib homepage, and the package will access them using the `Curl` function from the Qaos package [FP].

For the reader interested in more details, we included in package archive the file `unitlib/lib/genlib.g` with the function `CreatePcNormalizedUnitGroupsLibrary`, that creates library files for groups of a given prime power order, and the file `unitlib/lib/genpar.g` with the function `ParCreatePcNormalizedUnitGroupsLibrary`, which is the parallel version of the previous function and must be used with the ParGAP package [Coo].

3.2 Reading the data

To reconstruct the normalized unit group $V(KG)$ from the library, we need only to know the catalogue number of the underlying group G in the GAP Small Groups Library.

We use the same numbering as in the GAP Small Group Library, so UnitLib finds the appropriate library file(s) and reads from it the code for the polycyclic generating sequence of $V(KG)$ and the information about the dimension basis of G used for the computation of this code.

Then $V(KG)$ is created from the code using the GAP function `PcGroupCode`. We also create G using the GAP Small Groups Library.

Now to "glue" $V(KG)$ with the underlying group G properly, the value of the attribute `DimensionBasis` of G is setted in accordance with the data retrieved from the library. This will guarantee the correct construction of `NaturalBijectionToPcNormalizedUnitGroup` and `NaturalBijectionToNormalizedUnitGroup` by the LAGUNA package.

It remains now to make only several technical steps, such as constructing the group algebra KG over the appropriate field K, and storing KG in the attribute `UnderlyingGroupRing` of $V(KG)$ and $V(KG)$ in the attribute `PcNormalizedUnitGroup` of KG.

Chapter 4

An example of UnitLib usage

We will finish with several examples of UnitLib usage to give an idea how to work with the package.

In the first example we retrieve from the library the normalized unit group of the group algebra of the dihedral group of order 128 over the field of two elements, compute its center and express one of its generators in terms of group algebra elements:

Example

```
gap> IdGroup(DihedralGroup(128));
[ 128, 161 ]
gap> V := PcNormalizedUnitGroupSmallGroup( 128, 161 );
<pc group of size 170141183460469231731687303715884105728
  with 127 generators>
gap> C := Center( V );
<pc group with 34 generators>
gap> gens := MinimalGeneratingSet( C );;
gap> KG := UnderlyingGroupRing( V );
<algebra-with-one over GF(2), with 7 generators>
gap> f := NaturalBijectionToNormalizedUnitGroup( KG );;
gap> gens[8]^f;
(Z(2)^0)*f3+(Z(2)^0)*f4+(Z(2)^0)*f7+(Z(2)^0)*f3*f4+(Z(2)^
0)*f3*f5+(Z(2)^0)*f3*f6+(Z(2)^0)*f3*f7+(Z(2)^0)*f4*f5+(Z(2)^
0)*f4*f6+(Z(2)^0)*f4*f7+(Z(2)^0)*f3*f4*f5+(Z(2)^0)*f3*f4*f6+(
Z(2)^0)*f3*f4*f7+(Z(2)^0)*f3*f5*f6+(Z(2)^0)*f3*f5*f7+(Z(2)^
0)*f3*f6*f7+(Z(2)^0)*f4*f5*f6+(Z(2)^0)*f4*f5*f7+(Z(2)^
0)*f4*f6*f7+(Z(2)^0)*f3*f4*f5*f6+(Z(2)^0)*f3*f4*f5*f7+(Z(2)^
0)*f3*f4*f6*f7+(Z(2)^0)*f3*f5*f6*f7+(Z(2)^0)*f4*f5*f6*f7+(Z(2)^
0)*f3*f4*f5*f6*f7
```

In the second example we will check the conjecture about the coincidence of the lower and upper Lie nilpotency indices of the modular group algebras for all non-abelian groups of order 64.

It is known that these indices coincide for p-groups with $p > 3$ [BP92], but in the general case the problem remains open.

The indices $t_L(G)$ and $t^L(G)$ can be computed using the LAGUNA package. While the upper Lie nilpotency index can be expressed only in terms of the underlying group G, the lower Lie nilpotency index is determined by the formula $t_L(G) = \mathrm{cl}\, V(KG) + 1$ [Du92], and can be computed immediately whenever $V(KG)$ is known.

In the program below we enumerate all groups of size 64 and check the conjecture (we do not exclude from consideration other cases when the conjecture is known to be true for $p = 2$, because this is beyond the task of this manual).

Example

```
gap> for n in [ 1 .. NrSmallGroups( 64 ) ] do
> if not IsAbelian( SmallGroup( 64, n ) ) then
>   V := PcNormalizedUnitGroupSmallGroup( 64, n );
>   if LieLowerNilpotencyIndex( KG ) <>
>      LieUpperNilpotencyIndex( KG ) then
>     Print( n," - counterexample !!! \n" );
>     break;
>   fi;
> fi;
> od;
gap>
```

Thus, the test was finished without finding a counterexample.

In the next example we will answer the question about possible nilpotency classes of normalized unit groups of modular group algebras of nonabelian groups of order 128:

Example

```
gap> cl := [];
gap> for n in [ 1 .. NrSmallGroups( 128 ) ] do
> if not IsAbelian( SmallGroup( 128, n ) ) then
>   V := PcNormalizedUnitGroupSmallGroup( 128, n );
>   AddSet( cl, NilpotencyClassOfGroup( V ) );
> fi;
> od;
gap> cl;
[ 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 16, 32 ]
```

With UnitLib you can perform the computation from the last example in several hours on a modern computer. Without UnitLib you will spend the same time to compute only several normalized unit groups $V(KG)$ for groups of order 128 with the help of the LAGUNA package. Note that without LAGUNA such computation would not be feasible at all.

References

[BKRS] V. Bovdi, A. Konovalov, R. Rossmanith, and C. Schneider. LAGUNA — Lie AlGebras and UNits of group Algebras. GAP4 package (http://ukrgap.exponenta.ru/LAGUNA.htm). 6

[Bov98] Adalbert Bovdi. Generators of the units of the modular group algebra of a finite p-group. In *Methods in ring theory (Levico Terme, 1997)*, volume 198 of *Lecture Notes in Pure and Appl. Math.*, pages 49–62. Dekker, New York, 1998. 10

[BP92] Ashwani K. Bhandari and I. B. S. Passi. Lie-nilpotency indices of group algebras. *Bull. London Math. Soc.*, 24(1):68–70, 1992. 12

[Coo] G. Cooperman. ParGAP — Parallel GAP. GAP4 package (http://www.gap-system.org/Packages/pargap.html). 11

[Du92] Xian Kun Du. The centers of a radical ring. *Canad. Math. Bull.*, 35(2):174–179, 1992. 12

[FP] S. Freundt and S. Pauli. QaoS — Interfacing the QaoS database from GAP. GAP4 package (http://www.gap-system.org/Packages/qaos.html). 11

Index