DOCUMENTATION FOR GAMMA_GRADED_CHARACTER_RINGS

ALEXANDER ZIEGLER

1. Introduction

Let G be a finite group interpreted as a discrete group scheme over $\mathbb{C}$ and $\mathsf{R}(\mathsf{G})$ its ring of virtual representations. Suppose one wants to compute the Chow ring $\mathrm{CH}^*(\mathsf{BG})$ of the motivic étale classifying space of a finite group G in the sense of [Tot14, Section 2.2]. Possible first approximations are $\mathsf{gr}^*_{\mathrm{geom}}\mathsf{R}(\mathsf{G})$ or, a bit further removed, $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$, the associated graded algebras to the geometric and the γ-filtration on $\mathsf{R}(\mathsf{G})$. We will not elaborate on the precise meaning of the term "approximation" here but refer the reader to [Zie25, Section 4].

Computing finitely many graded pieces $\mathsf{gr}^i_\gamma\mathsf{R}(\mathsf{G})$, including the ring structure of $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$, is a task that is straightforward while still incredibly labor intensive, a task well suited for a computer algebra system. Usually, we can even compute $\mathsf{gr}^i_{\mathrm{geom}}\mathsf{R}(\mathsf{G})$ by restricting to subgroups. In the following document, we will describe the usage of the functions implemented in `gamma_graded_character_rings` in order to compute $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$ or under good circumstances $\mathsf{gr}^*_{\mathrm{geom}}\mathsf{R}(\mathsf{G})$. At the end we also give a sketch of how these functions work. All further details can be found in the comments contained in the files `gamma.sage` and `gamma-filtration.g`. The mathematical background is laid out in [Zie25]. The code is written in [SAGE] and [GAP4]. The main functions use the function `trim` from [GS] to reduce some generating sets of ideals.

2. Intended Usage

To access the tools presented, simply open [SAGE] from the directory containing the files `gamma.sage`, `gamma-filtration.g` and load the file `gamma.sage`. This section is a rundown on how to use the functions

- `presentation`,
- `presentation_with_classes`,
- `prepare_tables`,
- `presentation_with_restriction` and
- `presentation_with_restriction_and_classes`

which are the intended interface to the code in `gamma_graded_character_rings`. Laying things out roughly,

`presentation`

computes a generating set of $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$, and all degree i relations governing these generators, where i is bounded by some n specified by the user. To increase performance significantly, one can manually reduce the set of generators that is being considered.

`presentation_with_restriction`

Date: Juni 2025.

computes $\mathrm{gr}^n_\gamma R(G)$ where n is some natural number specified by the user, as well as non-zero terms in $\mathrm{gr}^n_\gamma R(G)$ that restrict to zero on all members of some family of subgroups $H_i \leqslant G$ specified by the user. This can be helpful to identify non-zero terms that might vanish in $\mathrm{gr}^n_{\mathrm{geom}} R(G)$, by restricting to subgroups where $\mathrm{gr}^n_{\mathrm{geom}} R(H_i) = \mathrm{gr}^n_\gamma R(H_i)$ and using the fact that the maps $\mathrm{gr}^n_\gamma R(\_) \to \mathrm{gr}^n_{\mathrm{geom}} R(\_)$ are a natural transformation of contravariant functors.

For $F^i \subseteq R(G)$ that form the coarsest filtration on $R(G)$ such that $\Gamma^i \subseteq F^i$, $F^i \cdot F^j \subseteq F^{i+j}$ and F^i contains some elements of $R(G)$ specified by the user

`presentation_with_classes`

computes a generating set of the associated graded algebra $\bigoplus_i F^i/F^{i+1}$, degree i relations governing these generators, where i is bounded by some n specified by the user. To increase performance significantly, one can manually reduce the set of generators that is being considered. Ideally $F^i = F^i_{\mathrm{geom}}$ where the latter denotes the i-th step in the geometric filtration on $R(G)$.

`presentation_with_restriction_and_classes`

computes the group F^n/F^{n+1} where n is some natural number specified by the user, as well as non-zero terms in F^n/F^{n+1} that restrict to zero in $R(H_i)/F^{n+1}_{H_i}$. Here, the F^i are a filtration on $R(G)$ as described above. Similarly, the $F^{n+1}_{H_i}$ are the smallest ideals in $R(H_i)$ thhttps://link.springer.com/book/10.1007/978-3-662-71861-2at contain the image of F^{n+1} under restriction and some additional elements of $R(H_i)$ as specified by the user. The purpose of this is the same as `presentation_with_restriction` but for a modified filtration.

We will now go into more detail on how to use each function, including some examples.

Function 2.1. The function

`presentation(tbl, n, upper_bound, modified_strong_gens)`

takes as parameters

- `tbl` - a character table of a finite group G given as a [GAP4]-object,
- `n` - a positive integer that is the highest degree in which relations are computed,
- `upper_bound` - a positive integer that is an upper bound (with respect to divisibility) for the additive order of elements of $\mathrm{gr}^i_\gamma R(G)$ where $i \leqslant n$,
- `modified_strong_gens` - either `[false]` or `[true, x]` where `x` is an ascending list of integers that correspond to generators of $\mathrm{gr}^*_\gamma R(G)$.

The function returns a list whose three entries are in order

- if `modified_strong_gens=[false]` a list of generators of $\mathrm{gr}^*_\gamma R(G)$ which only depends on `tbl`, otherwise if `modified_strong_gens=[true,x]` the sublist of this list of generators of entries at positions specified in `x` (e.g. positions 1,2 for `x=[1,2]`),
- a list of abstract Chern classes corresponding to the generators given in the form `[1,[i,j]]` which represent the `j`-th Chern class of the `i`-th character in `tbl.Irr()`,
- a generating set of the ideal containing all relations, up to degree `n`, of a presentation of $\mathrm{gr}^*_\gamma R(G)$ with respect to the aforementioned (sub-)list of generators.

We will make two important remarks on how to increase the performance of this and the following functions. Their memory requirements grow exponentially with the number of generators being considered. It is very important to make use of `modified_strong_gens=[false]` to exclude redundant generators.

This function performs much better for smaller `n`, so usually we will execute the function as

```
tmp=presentation(tbl, 1, upper_bound, [false])
```

to get a generating set `tmp[0]` of $\mathsf{gr}^*_\gamma \mathsf{R}(\mathsf{G})$ (which is independent of `n`) and some degree 1 relations. Using these relations, we can find redundancies among the generators. From this we can determine a list of integers `i` such that the `tmp[i]` generate $\mathsf{gr}^*_\gamma \mathsf{R}(\mathsf{G})$ but they do not exhibit any redundancies in degree 1. Let `l1` be the ascending list of these integers `i` and execute

```
tmp1=presentation(tbl, 2, upper_bound, [true,l1]).
```

This yields again redundancies which we can use to determine a sublist `l2` of `l1` that correspond to generators with no redundancies in degree 2 and we continue with

```
tmp2=presentation(tbl, 3, upper_bound, [true,l2])
```

and so on. An example of this strategy can be seen in Example 2.2.

Choosing a smaller value for `upper_bound` usually does not make a big difference, and it makes no difference if `upper_bound` is the power of a prime number. This parameter is used to compute the torsion of every generator of $\mathsf{gr}^n \mathsf{R}(\mathsf{G})$ by multiplying the generator with every divisor of `nupper_bound` in ascending order and checking whether it vanishes. Thanks to [Che20, Proposition 2.6] the order of G is always a sensible choice for `upper_bound`.

Example 2.2. In this example we'll take G to be the finite group given by

```
G=gap.SmallGroup(64,138)
```

which can be identified with $\mathrm{Syl}_2(\mathrm{GL}(4,2))$. We first define in [SAGE]

```
tbl=G.CharacterTable().
```

Denote the list of irreducible complex representations provided by `tbl.Irr()` as $(\phi_1, \ldots, \phi_n)$. Then

```
presentation(tbl,1,64,[false])
```

returns a list of generators of $\mathsf{gr}^*_\gamma \mathsf{R}(\mathsf{G})$

```
[x4, x5, x6, x7, x9, x11, x13, x15, x17, x19, x21, x23, x25]
```

identified with a list of abstract Chern classes

```
[[1, [6, 1]], [1, [7, 1]], [1, [8, 1]], [1, [9, 1]], [1, [11, 1]],
[1, [13, 1]], [1, [15, 1]], [1, [9, 2]], [1, [11, 2]], [1, [13, 2]],
[1, [15, 2]], [1, [15, 3]], [1, [15, 4]]]
```

as well as relations

```
[2*x13,2*x11,2*x9,x7+x9+x11+x13,x6+x11,x5+x11+x13,x4+x9].
```

This should be interpreted as, $\mathsf{gr}^*_\gamma \mathsf{R}(\mathsf{G})$ is generated by Chern classes

$$\begin{gathered}
\mathsf{x4} := c_1(\phi_6), \mathsf{x5} := c_1(\phi_7), \mathsf{x6} := c_1(\phi_8),\\
\mathsf{x7} := c_1(\phi_9), \mathsf{x9} := c_1(\phi_{11}), \mathsf{x11} := c_1(\phi_{13}), \mathsf{x13} := c_1(\phi_{15}),\\
\mathsf{x15} := c_2(\phi_9), \mathsf{x17} := c_2(\phi_{11}), \mathsf{x19} := c_2(\phi_{13}), \mathsf{x21} := c_2(\phi_{15}),\\
\mathsf{x23} := c_3(\phi_{15}), \mathsf{x25} := c_4(\phi_{15})
\end{gathered}$$

subject to degree 1 relations

$$\begin{gathered}
2c_1(\phi_{15}), 2c_1(\phi_{13}), 2c_1(\phi_{11})\\
c_1(\phi_9) + c_1(\phi_{11}) + c_1(\phi_{13}) + c_1(\phi_{15}),\\
c_1(\phi_8) + c_1(\phi_{13}), c_1(\phi_7) + c_1(\phi_{13}) + c_1(\phi_{15}),\\
c_1(\phi_6) + c_1(\phi_{11})
\end{gathered}$$

and relations in higher degree that have yet to be determined. We simply take the relations from the third list and substitute the Chern classes given by the second list for the generators at the same place in the first list.

Only the first seven generators correspond to first Chern classes. The relations tell us $\mathsf{gr}^1_\gamma \mathsf{R}(\mathsf{G}) \cong \langle \mathtt{x4}, \mathtt{x5}, \mathtt{x6}\rangle_{\mathbb{F}_2}$ is just an $\mathbb{F}_2$-vector space of dimension 3. The remaining generators of degree 1 are redundant. Thus we can improve performance by forcibly dropping these generators. Recall that the previous use of `presentation` returned the list of generators

```
[x4, x5, x6, x7, x9, x11, x13, x15, x17, x19, x21, x23, x25]
```

of which we now know that the sublist

```
[x4, x5, x6, x15, x17, x19, x21, x23, x25]
```

already generates $\mathsf{gr}^*_\gamma \mathsf{R}(\mathsf{G})$. The position of entries in this sublist as entries in the original list above is

```
[0,1,2,7,8,9,10,11,12].
```

We call now

```
presentation(tbl,2,64,[true,[0,1,2,7,8,9,10,11,12]])
```

and get relations

```
[4*x21, 4*x19, 4*x17, x15 + x17 + x19 + 2*x21, 2*x6, 2*x5, 2*x4,
x5^2 + x5*x6, x4*x5 + x4*x6 + x5*x6 + x6^2].
```

Notably `x19` is redundant and from the present relations we can compute

$$\begin{aligned}\mathsf{gr}^2_\gamma \mathsf{R}(\mathsf{G}) &\cong (\mathbb{Z}/2\mathbb{Z})^4 \oplus (\mathbb{Z}/4\mathbb{Z})^3 \\ &\cong \langle \mathtt{x4}^2, \mathtt{x5}^2, \mathtt{x6}^2, \mathtt{x4}\cdot\mathtt{x6}\rangle_{\mathbb{Z}/2\mathbb{Z}} \oplus \langle \mathtt{x15}, \mathtt{x17}, \mathtt{x21}\rangle_{\mathbb{Z}/4\mathbb{Z}}.\end{aligned}$$

As `x19` is a position 9 in our original list of generators we exclude 9 from the parameter `modified_strong_gens` and call on

```
presentation(tbl, 3, 64, [true,[0,1,2,7,8,10,11,12]])
```

to obtain relations

```
[2*x23, 4*x21, 4*x17, 4*x15, 2*x6, 2*x5, 2*x4, x5*x23,
x4*x23 + x6*x23, x5*x21 + x6*x21 + x23, x4*x17, x6*x15 + x6*x17,
x4*x15 + x5*x15, x5^2 + x5*x6, x4*x5 + x4*x6 + x5*x6 + x6^2,
x5*x15*x17, x6^2*x17, x5*x6*x17,
x5*x6^2 + x5*x15 + x5*x17 + x6*x21 + x23, x4^2*x6 + x4*x6^2,
x4^3 + x6^3 + x5*x15 + x6*x17 + x4*x21 + x6*x21].
```

There are some redundant relations in degrees > 3. Our code uses the [GS]-function `trim` to turn a highly redundant set of previously determined relations into a smaller, not quite so redundant set of relations. Despite this, there are often some leftover redundancies. After removing redundant relations, the seven new degree 3 relations are

```
x5*x21 + x6*x21 + x23, x4*x17, x6*x15 + x6*x17, x4*x15 + x5*x15,
x5*x6^2 + x5*x15 + x5*x17 + x6*x21 + x23,
x4^2*x6 + x4*x6^2, x4^3 + x6^3 + x5*x15 + x6*x17 + x4*x21 + x6*x21.
```

We invite the reader to compare these seven relations to those determined in [Zie25, Proof of Lemma 8.6 Eq. (9) to (15)]. We can conclude from the seven relations in degree 3 and the two relations in degree 2 that $\mathsf{gr}^3_\gamma \mathsf{R}(\mathsf{G}) \cong (\mathbb{Z}/2\mathbb{Z})^5$. One could continue to search for relations of higher degree, given sufficient computational power.

Function 2.3. The upcoming functions deal with subgroups and their character tables. This requires implicit data on the embeddings of these subgroups. This function gives a quick way to generate this data in the format required by the upcoming functions,

`prepare_tables(subgroup_list, tbl, subgroups_index)`

whose parameters are

- `subgroup_list` - an object of the form

  `gap.ConjugacyClassesSubgroups(G)`

  for a group `G` as a [GAP4]-object,
- `tbl` - the character table of `G`,
- `subgroups_index` - a list of positive indices `i` corresponding to entries `subgroup_list[i]`.

The function returns a pair `[tbl,sub_tbl]` where `tbl` is the character table from the first parameter and `sub_tbl` is a list such that `sub_tbl[i]` is the character table of a representative of the conjugacy class of subgroups `subgroup_list[i]`.

Function 2.4. A natural way to understand how much $\mathrm{gr}^i_\gamma R(G)$ and $\mathrm{gr}^i_{\mathrm{geom}} R(G)$ can differ at most are restrictions to subgroups where both coincide. Such an argument is realized by

`presentation_with_restriction(tbls, n,`
`upper_bound,modified_strong_gens)`

whose parameters are

- `tbls` - a pair `[tbl,sub_tbl]` of a character table `tbl` of a finite group G given as a [GAP4]-object and a list of character tables `sub_tbl` of subgroups of G (possibly generated by the function `prepare_tables`),
- `n` - a positive integer that is the degree in which relations are computed,
- `upper_bound` - a positive integer that is an upper bound (with respect to divisibility) for the additive order of elements of $\mathrm{gr}^n_\gamma R(G)$,
- `modified_strong_gens` - either `[false]` or `[true, x]` where `x` is an ascending list of integers that correspond to generators of $R(G)$.

The parameter `n,upper_bound,modified_strong_gens` work as for the function `presentation`.

The function returns a list whose five entries are

- if `modified_strong_gens=[false]` a list of generators of $\mathrm{gr}^*_\gamma R(G)$, otherwise if `modified_strong_gens=[true,x]` the sublist of entries at positions specified in `x`,
- a list of corresponding Chern classes given in the form `[1,[i,j]]` which represent the `j`-th Chern class of the `i`-th character in `tbl.Irr()`,
- a generating set of the ideal containing all relations in degree `n` of a presentation of $\mathrm{gr}^*_\gamma R(G)$ with respect to the aforementioned (sub-)list of generators (also multiples of relations of lower degree and sometimes some redundant relations of higher degree),
- a list of all non-zero terms in $\mathrm{gr}^n_\gamma R(G)$ that vanish when mapped via restriction to the product of all $\mathrm{gr}^n_\gamma R(H_i)$ where the H_i are the underlying groups of the entries in `sub_tbl`,
- a list of all isomorphism classes of groups underlying the entries of `sub_tbl` given as pairs of positive integers `[i,j]` that represent the corresponding entry in the Small Groups library.

Be careful with the relations returned in the third entry of the list. This function returns all relations witnessed in degree n which might be multiples of relations of lower degree. If you want a nice description of relations governing $\mathrm{gr}^*_\gamma R(G)$ you should use `presentation` instead.

Also know that the kernel of the combined restriction map is returned as a list of all non-zero terms that restrict to 0. Many of these non-zero terms will be equivalent modulo Γ^{n+1}. At first glance, it might not look like they form a subgroup of $\mathrm{gr}^n R(G)$, but they do if you check these equivalences. The list might also be very long for the same reason.

Example 2.5. In the previous example, we have computed a generating set and all corresponding relations up to degree 3 for $\mathrm{gr}^*_\gamma R(G)$ with G given by

`G:=gap.SmallGroup(64,138).`

We would also like to know whether the map $\mathrm{gr}^*_\gamma R(G) \to \mathrm{gr}^*_{\mathrm{geom}} R(G)$ is an isomorphism up to degree 3. This map is, independently of G, always an isomorphism in degrees 0 and 1. In fact, if we take Γ^i to be the i-th step in the γ-filtration and likewise F^i_{geom} the i-th step in the geometric filtration, it is known for all $i \in \mathbb{N}$ that

$$F^i_{\mathrm{geom}} \subseteq \Gamma^i$$

where this inclusion is equality for $i = 0, 1, 2$. Therefore, it suffices to check that $\mathrm{gr}^*_\gamma R(G) \to \mathrm{gr}^*_{\mathrm{geom}} R(G)$ is injective up to degree 3, as this is equivalent to $F^i_{\mathrm{geom}} = \Gamma^i$ for $i = 3, 4$.

We will do so by using the fact that $\mathrm{gr}^*_\gamma R(\_) \to \mathrm{gr}^*_{\mathrm{geom}} R(\_)$ is natural with respect to restriction to subgroups. For this to work, we will have to restrict to subgroups $H \leqslant G$ for which we already know $\mathrm{gr}^*_\gamma R(\_) = \mathrm{gr}^*_{\mathrm{geom}} R(\_)$.

We use `G` and `tbl` from the previous example and invoke

```
CG=G.ConjugacyClassesSubgroups(),
tbls=prepare_tables(CG, tbl, [41,49,55]])
```

to generate the character tables of certain subgroups. The subgroups $H_i \leqslant G$ given by representatives of the entries of `CG` at positions 41, 49, 55 are all isomorphic to `gap.SmallGroup(8,3)` which is isomorphic to the dihedral group D_8 and $\mathrm{Syl}_2(\mathrm{GL}(3,2))$. Their Chow rings are generated by Chern classes (see [Tot14, Chapter 13] or [Zie25]) which implies $\mathrm{gr}^*_\gamma R(H_i) = \mathrm{gr}^*_{\mathrm{geom}} R(H_i)$. We have determined these groups to be the ones that make our argument work by trial and error.

In the previous example we have already seen that we may run our functions with

```
modified_strong_gens=[true,[0,1,2,7,8,10,11,12]]
```

to remove redundant generators of $\mathrm{gr}^*_\gamma R(G)$. We execute the function

```
presentation_with_restriction(tbls, 2, 64,
        [true,[0,1,2,7,8,10,11,12]])
```

which returns a list whose fourth entry is the empty list. This means

$$\ker\left(\mathrm{gr}^2_\gamma R(G) \to \prod_{i \in [41,49,55]} \mathrm{gr}^2_\gamma R(H_i)\right) = 0$$

where we denote by H_i a representative of

`G.ConjugacyClassesSubgroups()[i].`

Applying the commutative square

$$\begin{tikzcd}
\mathrm{gr}^2_\gamma R(G) \arrow[r] \arrow[d, hook] & \mathrm{gr}^2_{\mathrm{geom}} R(G) \arrow[d] \\
\prod_{i \in [41,49,55]} \mathrm{gr}^2_\gamma R(H_i) \arrow[r, "\sim"] & \prod_{i \in [41,49,55]} \mathrm{gr}^2_{\mathrm{geom}} R(H_i)
\end{tikzcd}$$

we see that the top map is injective and we already knew that it is surjective. This means

$$\mathrm{gr}^2_\gamma R(G) = \mathrm{gr}^2_{\mathrm{geom}} R(G).$$

The function

```
presentation_with_restriction(tbls, 3, 64,
        [true,[0,1,2,7,8,10,11,12]])
```

again returns a list whose fourth entry is the empty list. By the same argument as in degree 2 we get $\mathrm{gr}^3_\gamma R(G) = \mathrm{gr}^3_{\mathrm{geom}} R(G)$.

In the sequel we will deal with modifications of the γ-filtration in order to determine the geometric filtration. For this we need to describe elements of $R(G)$, where G is some finite group, in terms of abstract Chern classes, as this is the language the code operates in. Previously we have described the j-th Chern class of the i-th character in the character table of G as a nested list `[1,[i,j]]`. We want this to also be a stand-in for the canonical lift of the Chern class to $R(G)$ which is $C_j(\phi_i) := \lambda^j(\phi_i - \deg\phi_i + j - 1)$ where ϕ_i is the i-th character in the character table of G. Furthermore we will deal with integer polynomials of these (lifts of) Chern classes. We model an integer monomial

$$z \prod_{k=1}^{n} C_{j\_k}(\phi_{i\_k})$$

as a nested list `[z,[i_1,j_1],...,[i_n,j_n]]`. We model an integer polynomial of Chern classes just as the list of the monomials that are its summands.

Function 2.6. Take $A_i \subseteq R(G)$ to be a family of finite sets indexed over $i \in \mathbb{N}$ that is the empty set for all but finitely many i. Let F^i be the coarsest filtration on $R(G)$ such that $\Gamma^i \subseteq F^i$, $F^i \cdot F^j \subseteq F^{i+j}$ and $A_i \subseteq F^i$. Ideally $F^i = F^i_{\mathrm{geom}}$ where the latter denotes the i-th step in the geometric filtration on $R(G)$. One can compute the associated graded algebra to the F^i via

```
presentation_with_classes(tbl, n, upper_bound,
modified_strong_gens, added_strong_gens, added_degrees)
```

which takes as parameters

- `tbl` - a character table of a finite group G given as a [GAP4]-object,
- `n` - a positive integer that is the highest degree in which relations are computed,
- `upper_bound` - a positive integer that is an upper bound (with respect to divisibility) for the order of elements of the abelian group F^i/F^{i+1} where $i \leqslant n$,
- `modified_strong_gens` - either `[false]` or `[true, x]` where `x` is an ascending list of integers that correspond to generators of $\mathrm{gr}^*_\gamma R(G)$,
- `added_strong_gens` - a list of elements of $R(G)$ represented as polynomials of canonical lifts of Chern classes which should be understood as the elements of A_i above
- `added_degrees` - a list of positive integers whose j-th entry describes the degree of the j-th entry of `added_strong_gens`, or more precisely the largest index i such that the lifted Chern class described by the j-th entry of `added_strong_gens` is an element of A_i.

The parameter `modified_strong_gens`, works as usual. To find suitable candidates for `upper_bound`, one could use the fact that elements in Γ^i/Γ^{i+1} are $\#G$-torsion which implies elements in Γ^i/Γ^{i+k} are $(\#G)^k$-torsion. We'll see an example in Example 2.8.

The function returns a list whose three entries are

- if modified_strong_gens=[false] a list of generators of the associated graded algebra of the F^i including a generator for each entry of the parameters added_strong_gens, otherwise if modified_strong_gens=[true,x]} the sublist of entries at positions specified in x together with all generators corresponding to entries of added_strong_gens,
- a list of corresponding Chern classes given in the form [1,[i,j]] which represent the j-th Chern class of the i-th character in tbl.Irr() or just

  ['generator that is not a Chern class', j]

  for every generator of degree j that is not a polynomial of Chern classes, i.e. every generator corresponding to an entry of added_strong_gens,
- a generating set of the ideal containing all relations, up to degree n, of the presentation of the associated graded algebra for the F^i with respect to the aforementioned (sub-)list of generators.

As before, the list of generators that is computed is independent of n, so we can do the usual trick of computing generators in low degree and then excluding the redundant generators, to increase the performance of this function. For an example, see Example 2.8.

Function 2.7. Take $A_i \subseteq R(G)$ to be a family of finite sets that is empty for all but finitely many i and F^i to be the coarsest filtration on $R(G)$ such that $\Gamma^i \subseteq F^i$, $F^i \cdot F^j \subseteq F^{i+j}$ and $A_i \subseteq F^i$. Ideally $F^i = F^i_{\text{geom}}$. Furthermore let $H_j \leqslant G$ be a family of subgroups together with families of finite sets $A_i^{H_j} \subseteq R(H_j)$ giving rise to filtrations $F^i_{H_j}$ on $R(H_j)$ analogously. One can check how far F^i and F^i_{geom} might differ via restrictions to the H_j by invoking the function

```
presentation_with_restriction_and_classes(tbls, n, upper_bound,
    modified_strong_gens, added_strong_gens, added_degrees,
        sub_added_rels, sub_added_degrees)
```

which takes as parameters

- tbls - a pair [tbl,sub_tbl] of a character table tbl of a finite group G given as a [GAP4]-object and a list of character tables sub_tbl of subgroups of G (possibly generated by the function prepare_tables),
- n - a positive integer that is the degree in which relations are computed,
- upper_bound - a positive integer that is an upper bound (with respect to divisibility) for the additive order of elements of F^i/F^{i+1} where $i \leqslant n$,
- modified_strong_gens - either [false] or [true, x] where x is an ascending list of integers that correspond to generators of $\mathrm{gr}^*_\gamma R(G)$,
- added_strong_gens - a list of elements of $R(G)$ represented as polynomials of canonical lifts of Chern classes which should be understood as the elements of A_i above
- added_degrees - a list of positive integers whose j-th entry describes the degree of the j-th entry of added_strong_gens, or in the language above the largest i such that the j-th entry of added_strong_gens is an element of A_i,
- sub_added_rels - a list whose j-th entry is a list of elements of $R(H_j)$ represented as polynomials of lifted Chern classes, for H_j the group underlying the character table sub_tbl[j], which should be understood as the elements of $A_i^{H_j}$ above,
- sub_added_degrees - a list of lists of positive integers whose j-th entry describes the degree of the j-th entry of sub_added_rels analogously to how added_degrees describes the degrees of entries of added_strong_gens

The parameter modified_strong_gens, works as usual. To find suitable candidates for upper_bound one could use the fact, that elements in Γ^i/Γ^{i+1} are $\#G$-torsion which implies elements in Γ^i/Γ^{i+k} are $(\#G)^k$-torsion. We'll see an example later.

If the $A_i^{H_j}$ are empty then sub_added_rels and sub_added_degrees have to be a list of empty lists for every subgroup H_j. This is a bit annoying so alternatively to a list of empty lists they can be chosen to be just [], the empty list.

The function returns a list whose five entries are

- if modified_strong_gens=[false] a list of generators of the associated graded algebra of the F^i including a generator for each entry of the parameter added_strong_gens, if modified_strong_gens=[true,x] the sublist of entries at positions specified in x together with all generators corresponding to entries of added_strong_gens,
- a list of corresponding Chern classes given in the form [1,[i,j]] which represent the j-th Chern class of the i-th character in the character table of tbl or just

  ['generator that is not a Chern class', j]

  for every generator of degree j that is not a polynomial of Chern classes,
- a generating set of the ideal containing all relations in degree n (sometimes also some redundant relations of higher degree) of the presentation of the associated graded algebra for the F^i with respect to the aforementioned (sub-)list of generators,
- a list of all non-zero terms in F^n/F^{n+1} that vanish when restricted to all $F^n_{H_j}/F^{n+1}_{H_j}$ where the H_j are the underlying groups of the entries in sub_tbl,
- a list of all isomorphism classes of groups underlying the entries of sub_tbl given as pairs of positive integers [i,j] that represent the corresponding entry gap.SmallGroup(i,j) in the Small Groups library.

The notes on the pathologies of the returned objects given in Function 2.4 applies here as well.

We now give an example on how we can use

Example 2.8. In this example we take G to be the group G=gap.SmallGroup(32,8). We first define in [SAGE]

```
G=gap.SmallGroup(32,8),
tbl=G.CharacterTable().
```

Then presentation(tbl,1,32,[false]) returns a list of generators

```
[x6, x4, x7, x9, x10, x12, x13, x14]
```

identified with a list of abstract Chern classes

```
[[1, [8, 1]], [1, [6, 1]], [1, [9, 1]], [1, [11, 1]], [1, [9, 2]],
 [1, [11, 2]], [1, [11, 3]], [1, [11, 4]]]
```

and returns an ideal of relations

```
(x9, 2*x7, 4*x6, x4 + x7).
```

We see that x9 and x7 are redundant and

$$\mathrm{gr}^1_\gamma R(G) \cong \mathrm{gr}^1_{\mathrm{geom}} R(G) \cong \mathbb{Z}/2\mathbb{Z} \oplus \mathbb{Z}/4\mathbb{Z}.$$

We execute

```
presentation(tbl, 2, 32, [true, [0, 1, 4, 5, 6, 7]])
```

and get an ideal of relations

```
(4*x12, 4*x10, 4*x6, 2*x4, 2*x6^2,
x4^2 + x4*x6 + x6^2 + 2*x10 + 2*x12).
```

Thus $\mathsf{gr}^2_\gamma \cong (\mathbb{Z}/2\mathbb{Z})^2 \oplus (\mathbb{Z}/4\mathbb{Z})^2$. For upcoming restriction argments we define

```
CG=G.ConjugacyClassesSubgroups(),
tbls=prepare_tables(CG, tbl, [9,10,12,13]]).
```

The subgroups $\mathsf{H_i} \leqslant \mathsf{G}$ given by representatives of the entries of `CG` at 9, 10, 12, 13 are all finite abelian meaning for each of them the geometric and the γ-filtration coincide. By invoking

```
presentation_with_restriction(tbls, 2, 32, [true,[0,1,4,5,6,7]])
```

we get a list of non-zero terms that vanish under all restrictions

$$\mathsf{gr}^2_\gamma \mathsf{R(G)} \to \mathsf{gr}^2_\gamma \mathsf{R(H_i)} = \mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(H_i)}$$

which is

```
[2*x10, x4^2, x4^2 + 2*x10, x4*x6 + x6^2 + 2*x12,
x4*x6 + x6^2 + 2*x10 + 2*x12, x4^2 + x4*x6 + x6^2 + 2*x12]].
```

Suppose

$$x \in \ker(\mathsf{gr}^2_\gamma \mathsf{R(G)} \to \mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)})$$

then for all i

$$x \in \ker(\mathsf{gr}^2_\gamma \mathsf{R(G)} \to \mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(H_i)}).$$

Thus the the terms above give an upper bound on the kernel of the surjective map $\mathsf{gr}^2_\gamma \mathsf{R(G)} \to \mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)}$. We will now use this to compute $\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)} \cong \mathsf{CH}^2(\mathsf{BG})$.

We know $\mathsf{gr}^1_{\mathrm{geom}} \mathsf{R(G)} \cong \mathsf{CH}^1(\mathsf{BG}) \cong \mathsf{H}^2(\mathsf{BG}, \mathbb{Z})$ and by [Tot14, Lemma 15.1] that

$$\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)} \cong \mathsf{CH}^2(\mathsf{BG}) \to \mathsf{H}^4(\mathsf{BG}, \mathbb{Z})$$

is injective. Using [Ell24] we can compute all squares of 2-torsion elements of $\mathsf{H}^2(\mathsf{BG}, \mathbb{Z})$, only one of which vanishes. This element has to be the image of `2*x6`. We can conclude, that `x4^2` can not vanish in $\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)}$.

Suppose that `x4^2+2*x10` does not vanish in $\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)}$. By inspecting the upper bound on

$$\ker(\mathsf{gr}^2_\gamma \mathsf{R(G)} \to \mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)})$$

determined above and noting that `x4^2` can not vanish in $\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)}$, we get that $\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)}$ contains a subgroup isomorphic to $(\mathbb{Z}/2\mathbb{Z})^4$ while [Ell24] tells us that

$$\mathsf{H}^4(\mathsf{BG}, \mathbb{Z}) \cong \mathbb{Z}/2\mathbb{Z} \oplus (\mathbb{Z}/4\mathbb{Z})^2$$

does not contain such a subgroup. This means `x4^2+2*x10` has to vanish.

As all remaining possible relations are equivalent to the two relations considered above via some degree 1 relations, we can conclude that

$$\ker(\mathsf{gr}^2_\gamma \mathsf{R(G)} \to \mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)}) \cong \mathbb{Z}/2\mathbb{Z}$$

is generated by `x4^2+2*x10`. From this we can deduce

$$\mathsf{gr}^2_{\mathrm{geom}} \mathsf{R(G)} \cong \mathbb{Z}/2\mathbb{Z} \oplus (\mathbb{Z}/4\mathbb{Z})^2.$$

This means $\mathsf{gr}^3_{\mathrm{geom}} \mathsf{R(G)}$ contains a generator given by the canonical lift (as a polynomial of Chern classes) of the term `x4^2+2*x10` from Γ^2/Γ^3 to $\mathsf{R(G)}$, and that lift can not be expressed in terms of Chern classes. We know for all i that $\mathsf{gr}^i_\gamma \mathsf{R(G)}$ is $\#\mathsf{G}$-torsion meaning `x4^2+2*x10` is $(\#\mathsf{G})^2$-torsion modulo Γ^4. We also know $\Gamma^4 + \Gamma^1(\texttt{x4^2+2*x10})) \leqslant \mathsf{F}^4_{\mathrm{geom}}$.

Define now a multiplicative filtration on $\mathsf{R(G)}$

$$\mathsf{F}^i := \begin{cases} \Gamma^i & \text{for } i = 0,1,2 \\ \Gamma^i + \Gamma^{i-2}(\texttt{x4^2+2*x10}) & \text{else} \end{cases}$$

and we get $\mathsf{F}^i = \mathsf{F}^i_{\mathrm{geom}}$ for $i \leqslant 3$ and $\mathsf{F}^i \leqslant \mathsf{F}^i_{\mathrm{geom}}$ else.

As an approximation of $\mathsf{F}^3_{\mathrm{geom}}/\mathsf{F}^4_{\mathrm{geom}}$, we compute $\mathsf{F}^3/\mathsf{F}^4$ via

```
presentation_with_classes(tbl,3,32^2,
[true,[0,1,4,5,6,7]],[[[1,[6,1],[6,1]],[2,[9,2]]]],[3])
```

and get a list of relations governing the associated graded to the F^i up to degree 3

```
(2*x15, x13, 4*x12, 4*x10, 4*x6, 2*x4, 2*x10*x12, 2*x6*x12, x4*x12,
2*x10^2, 2*x6*x10, x4*x10, 2*x6^2, x4*x6 + x6^2 + 2*x12,
x4^2 + 2*x10, x6^2*x12 + 2*x12^2, x6^2*x10, x6^3)
```

where `x15` corresponds to the new generator given by the lift of `x4^2+2*x10`. Notably `x13` is redundant. To compute from this $\mathsf{F}^3_{\text{geom}}/\mathsf{F}^4_{\text{geom}}$, we execute

```
presentation_with_restriction_and_classes(tbls, 3,32^2,
[true,[0,1,4,5,7]], [9,10,12,13], [[[1,[6,1],[6,1]],[2,[9,2]]]],
[3], [], [])
```

and get the following terms that restrict to 0 under $\mathsf{F}^3/\mathsf{F}^4 \to \mathsf{gr}^3_{\text{geom}}\mathsf{R}(\mathsf{H}_i)$ for all i

```
[x6*x10, x4^2*x6 + x6*x10, x4*x6^2 + x6*x10,
x4^2*x6 + x4*x6^2 + x6*x10].
```

All of these relations are equivalent modulo F^3. It is remains to be checked whether `x6*x10` actually vanishes modulo $\mathsf{F}^4_{\text{geom}}$ and thus whether

$$\mathsf{gr}^3_{\text{geom}}\mathsf{R}(\mathsf{G}) \cong \mathsf{F}^3/\mathsf{F}^4 \cong (\mathbb{Z}/2\mathbb{Z})^3$$

or

$$\mathsf{gr}^3_{\text{geom}}\mathsf{R}(\mathsf{G}) \cong \mathsf{F}^3/(\mathsf{F}^4 + (\mathtt{x6*x10})) \cong (\mathbb{Z}/2\mathbb{Z})^2.$$

2.9. **Groups of high order.** This section covers a trick to get meaningful results for big groups. Say, we have a large character table `tbl` of a finite group G as a [GAP4]-object and we want to use it to determine some information on $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$ by invoking

```
presentation(tbl, n, upper_bound, [false]).
```

This computes a generating set of $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$ with many redundant generators and then relations between the generators. The memory requirements for this grow exponentially in the number of generators that are being considered, so we really want to get rid of these redundancies. We usually do this by taking `[true,x]` as the fourth input, where `x` a list of numbers corresponding to a sublist of the computed generators. To get relations, we do not need this sublist to be a generating set. Recall that the computed generators do not depend on the third input `n`. Compute by

```
tmp=presentation(tbl, 1, upper_bound, [false])[0]
```

a list of generators of $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$, then

```
presentation(tbl, n, upper_bound, [true,[2,3,4]])[0]
```

computes all relations (up to degree `n`) between the entries `tmp[2]`,`tmp[3]`,`tmp[4]`, even if they do not generate $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$. Thus, we can check subsets of our generating set to find redundancies, without checking the entire set. We will now show this at an example for a group G of order 729, where we are able to compute $\mathsf{gr}^1_\gamma\mathsf{R}(\mathsf{G}), \mathsf{gr}^2_\gamma\mathsf{R}(\mathsf{G})$ and an upper bound on the rank of $\mathsf{gr}^3_\gamma\mathsf{R}(\mathsf{G})/3$.

Example 2.10. The following example was computed using 8GB of RAM. We had to restart our [SAGE]-session regularly to free up enough RAM to continue. None of the computation steps took more than 3 hours, using an Intel(R) Core(TM) i5-10310U CPU @ 1.70GHz as a processor.

One should keep in mind, that for a group `G` as [GAP4]-object the character table `CharacterTable(G)` depends on what things have already been determined about `G` (e.g. the conjugacy classes) in the session. To make sure the character tables are the same, we suggest to generate the group and character table immediately after starting the session or using `gap.PrintCharacterTable()` to save the exact structure of the table.

We take $\mathsf{G} = \mathrm{Syl}_3\mathrm{GL}(4,3)$ which is provided by the Small Groups library as `G:=SmallGroup(729,307)`. If you intend to save the structure of the character table using `gap.PrintCharacterTable()`, you should instead construct the group as the group of strict upper triangular matrices over $\mathbb{F}_3$, as `gap.PrintCharacterTable()` does not save the relations governing the abstract generators of a group defined from a finite presentation.

Now

```
tmp=presentation(tbl,1,729,[false])
```

returns a list `tmp[0]` of 45 generators identified with a list `tmp[1]` of 45 abstract Chern classes and returns an ideal generated by relations

```
[x53,x50,x47,x44,x41,x38,x35,x32,x29,x26,3*x25,3*x24,3*x22].
```

Here the generators

```
x26, x29, x32, x35, x38, x41, x44, x47, x50, x53
```

correspond to the entries of `tmp[0]` at positions `range(3,13)`. Thus we get

$$\mathsf{CH}^1(\mathsf{BG}) \cong \mathsf{gr}^1_{\mathrm{geom}}\mathsf{R}(\mathsf{G}) \cong \mathsf{gr}^1_\gamma\mathsf{R}(\mathsf{G}) \cong (\mathbb{Z}/3\mathbb{Z})^3.$$

By comparing the generators in `tmp[0]` to their corresponding abstract Chern classes in `tmp[1]` we see that the degree 2 generators in `tmp[0]` are at the positions `range(13,23)`. With all this in mind, we want to search for interesting degree 2 relations, so we ignore the redundant generators of degree 1 and the generators in degree above 2. Sadly, 8 GB of RAM do not suffice to to execute

```
tmp=presentation(tbl,2,729,[true,[0,1,2]+list(range(13,23))]).
```

Instead we check the degree 2 generators step by step for redundancies. Start with

```
tmp=presentation(tbl,2,729,[true,[0,1,2,13,14,15,16]]).
```

and get relations

```
[3*x65, 3*x62, 3*x59, x56 - x59, 3*x25, 3*x24, 3*x22].
```

where `x59` is the generator `tmp[0][14]`. So we can continue with

```
tmp=presentation(tbl,2,729,[true,[0,1,2,13,15,16,17]]).
```

to get relations

```
3*x68, 3*x65, 3*x62, 3*x56, 3*x25, 3*x24, 3*x22.
```

As a next step we would then call

```
tmp=presentation(tbl,2,729,[true,[0,1,2,13,15,16,17,18]]).
```

and get that `tmp[0][18]` is a redundant generator. Continuing in this manner will tell us that the new degree 2 generators can be taken to be entries of `tmp[0]` at positions `[13,15,16,17,21]`. They are all 3-torsion and together with the twofold products of the three degree 1 generators they form a basis of $\mathsf{gr}^2_\gamma\mathsf{R}(\mathsf{G})$ as a $\mathbb{F}_3$-vector space of dimension 11.

We want to get a nice upper bound on the rank of $\mathsf{gr}^3_\gamma\mathsf{R}(\mathsf{G})/3$ as this gives rise to a nice upper bound on its quotient $\mathsf{gr}^3_{\mathrm{geom}}\mathsf{R}(\mathsf{G})/3 \cong \mathsf{CH}^3(\mathsf{BG})/3$. We begin by calling on

```
tmp=presentation(tbl,3,729,[true,[0,1,2,13]]).
```

and see that the generators

```
[x25, x22, x24, x56]
```

are subject to the relations

```
[3*x56, 3*x25, 3*x24,3*x22, x24^3 - x25^3 + x24*x56 - x25*x56,
x22^2*x24 - x22^2*x25 - x24*x25^2 + x25^3,
x22^3 - x25^3 + x22*x56 - x25*x56]]
```

This means the subgroup of $\mathsf{gr}^3_\gamma \mathsf{R}(\mathsf{G})$ generated by degree 3 products of the generators `x25, x22, x24, x56` is of rank 10 with a basis consisting of `x22*x56` and all threefold products of `x25, x22, x24` except for `x25^3`. We continue with

`tmp=presentation(tbl,3,729,[true,[0,1,2,15]])`.

and again get that the subgroup of $\mathsf{gr}^3_\gamma \mathsf{R}(\mathsf{G})$ generated by degree 3 products of the generators `x25, x22, x24, x62` is of rank 10 with a basis consisting of `x22*x62` and all threefold products of `x25, x22, x24` except for `x25^3`. We get similar results when doing this for any of the (non-redundant) degree 2 generators and conclude that the subgroup of $\mathsf{gr}^3_\gamma \mathsf{R}(\mathsf{G})$ generated by threefold products of degree 1 elements and products of degree 1 with degree 2 elements is of rank at most 13. We get 8 degrees of freedom from the threefold products of degree 1 generators and then one more degree of freedom for each degree 2 generator.

The degree 3 generators in `tmp[0]` are the entries at positions `range(23,33)`. We use

`tmp=presentation(tbl,3,729,[true,[23,24,25,26,27,28]])`.

and get that the degree 3 generators

`x86, x89, x92, x95, x98, x101`

are subject to the relations

`9*x101, 9*x98, 3*x95 + 3*x98 - 3*x101,`
`x92 + x101, 3*x89 - 3*x98 - 3*x101, x86 + x89`,

We can now exclude the redundant generators `x101,x89` and continue with

`tmp=presentation(tbl,3,729,[true,[23,25,26,27,29,30]])`.

We will omit the details here, but proceeding by finding redundancies, excluding redundant generators and adding new generators until all degree 3 generators are considered will tell us that the

`x86, x92, x95, x98, x110`

generate the subgroup of $\mathsf{gr}^3_\gamma \mathsf{R}(\mathsf{G})$ generated by our original list of degree 3 generators. They are subject to the relations

`9*x110, 9*x98, 9*x95, 3*x92 + 3*x95 + 3*x98, 3*x86 + 3*x95 - 3*x98`.

Thus said subgroup is isomorphic $(\mathbb{Z}/3\mathbb{Z})^2 \oplus (\mathbb{Z}/9\mathbb{Z})^3$.

Combining our computation for generators of degree 1 and 2 with our computation for generators of degree 3, we get that $\mathsf{gr}^3\mathsf{R}(\mathsf{G})/3$ is of rank at most 18.

3. Rough Explanation of the [SAGE] Code

We want to use this section to explain very roughly how our code works. For details, we ask the curious kindly to consult the comments in the files.

Large parts of the code, in particular the [GAP4] part are dedicated to setting up the data type of an element in Γ^i represented as a pair of a virtual character and as an abstract Chern class (or monomial or polynomial) of abstract Chern classes. We need the abstract representation to give meaning to the generator and the concrete representation as character to determine sums and products. These parts of the code are mathematically straightforward and well documented by the comments.

Assuming that this data type just works, we can explain the structure of the main functions.

The most base functionality is given by

```
presentation(tbl, n, upper_bound, modified_strong_gens)
```

which is just a special case of

```
presentation_with_classes(tbl, n, upper_bound,
modified_strong_gens, added_strong_gens, added_degrees)
```

with `added_strong_gens=[]` and `added_degrees=[]`. Take G and $\mathsf{F}^i \trianglelefteq \mathsf{R}(\mathsf{G})$ as in Function 2.6. Denote by $\mathsf{gr}^*_\mathsf{F}\mathsf{R}(\mathsf{G})$ the associated graded to the F^i. This function works roughly in the following way where we treat lists as sets by abuse of notation:

`weak_gens` $\leftarrow$ a set of generators of $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$ which is just the Chern classes of all non-trivial irreducible characters, they give rise to elements in $\mathsf{gr}^*_\mathsf{F}\mathsf{R}(\mathsf{G})$ via the natural map $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G}) \to \mathsf{gr}^*_\mathsf{F}\mathsf{R}(\mathsf{G})$
`strong_gens` $\leftarrow$ a $\mathsf{gr}^*_\gamma\mathsf{R}(\mathsf{G})$ generating subset of `weak_gens` with fewer redundancies which consists of the Chern classes of a set of irreducible characters that generate $\mathsf{R}(\mathsf{G})$
if `modified_strong_gens[0]=true` **then**
    Remove entries of `strong_gens` at positions not in `modified_strong_gens[1]`
end if
`added_strong_gens_transformed` $\leftarrow$ additional generators of $\mathsf{gr}^*_\mathsf{F}\mathsf{R}(\mathsf{G})$ (in terms of characters) coming from the abstract polynomials of Chern classes listed in `added_strong_gens`
`deg_list` $\leftarrow$ the list of degrees of elements in `added_strong_gens`$\cup$`strong_gens`
`R` $\leftarrow \mathbb{Z}$[`weak_gens`]
for `i` $\in [1, ..., n+1]$ **do**
    `tmp_summand_gens` $\leftarrow$ a subset of $\mathsf{R}(\mathsf{G})$ that additively generates $\mathsf{F}^i/\mathsf{F}^{i+1}$ consisting of products of the entries of `added_strong_gens`$\cup$`strong_gens` using their degrees `deg_list`
    `tmp_gamma` $\leftarrow$ a generating set of the ideal F^{i+1} in `R`
    `tmp_torsion` $\leftarrow \{\mathrm{ord}_{\mathsf{F}^i/\mathsf{F}^{i+1}}(x) \mid x \in$ `tmp_summand_gens`$\}$
    `lin_combs` $\leftarrow \{\sum_i n_i x_i \mid x_i =$ `tmp_summand_gens[i]`$, n_i \leqslant$ `tmp_torsion[i]`$\}$
    `result` $\leftarrow$ `result` $+ \{x \in$ `lin_combs` $\mid x \equiv 0 \bmod$ `ideal(tmp_gamma)`$\}$
end for
return `result`

We omit a similar diagram for

```
presentation_with_restriction_and_classes(tbls, n, upper_bound,
    modified_strong_gens, added_strong_gens, added_degrees,
            sub_added_rels, sub_added_degrees)
```

because it is essentially as the one above but for the character table `tbls[0]` and every character table in `tbls[1]` whose structure naturally gives rise to a restriction map of representation rings. We then compute relations only in degree `n` and list all terms that do not restrict to an element in the ideal of degree `n` relations for all subgroups.

References

[Che20] Béatrice I. Chetard. "Graded character rings of finite groups". In: *Journal of Algebra* 549 (2020), pp. 291–318. ISSN: 0021-8693. DOI: https://doi.org/10.1016/j.jalgebra.2019.11.041.

[Ell24] G. Ellis. *HAP, Homological Algebra Programming, Version 1.66.* https://gap-packages.github.io/hap. GAP package. Oct. 2024.

[GAP4] *GAP – Groups, Algorithms, and Programming, Version 4.13.1.* The GAP Group. 2024. URL: %5Curl%7Bhttps://www.gap-system.org%7D.

[GS] Daniel R. Grayson and Michael E. Stillman. *Macaulay2, a software system for research in algebraic geometry.* Available at http://www2.macaulay2.com.

[SAGE] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 3.11.1).* https://www.sagemath.org. 2024.

[Tot14] Burt Totaro. *Group Cohomology and Algebraic Cycles*. Cambridge Tracts in Mathematics. Cambridge University Press, 2014.

[Zie25] Alexander Ziegler. "The Chow Ring of the 2-Sylow Subgroup of GL(4,2)". In: *ArXiv Preprint* (2025).