BASIC USAGE OF `modcone_genus_2` IN SAGEMATH

Riccardo Zuffetti

Summary

The SageMath program `modcone_genus_2` is useful for working with the modular cones introduced in [Zuf22b]. For fixed k, it provides empirical evidences that the modular cone $\mathcal{C}_k$ is polyhedral. For $k \leq 38$, such polyhedrality is certified. This note gives a short explanation on how to use `modcone_genus_2`.

Last update: April 4, 2022.

Contents

1. Notation and background

Let k be an even integer. We denote by M_2^k, resp. $M_2^k(\mathbb{Q})$, the space of Siegel modular forms of genus 2 and weight k, resp. the $\mathbb{Q}$-space of the ones with rational Fourier coefficients. The space of elliptic modular forms of weight k is denoted by M_1^k, with analogous meaning for $M_1^k(\mathbb{Q})$.

Recall that the Fourier coefficients of a Siegel modular form in M_2^k are indexed over the set Λ_2 of symmetric half-integral positive semi-definite matrices of order two, namely

$$\Lambda_2 = \big\{ T = \left(\begin{smallmatrix} n & r/2 \\ r/2 & m \end{smallmatrix}\right) : n, r, m \in \mathbb{Z} \text{ and } T \geq 0 \big\}.$$

For every $F \in M_2^k$, we write its Fourier expansion as

$$F(Z) = \sum_{T \in \Lambda_2} c_T(F) \cdot e^{2\pi i \operatorname{tr}(TZ)}, \qquad Z \in \mathbb{H}_2.$$

The subset of positive definite matrices in Λ_2 is denoted by Λ_2^+. We say that a matrix $T = \left(\begin{smallmatrix} n & r/2 \\ r/2 & m \end{smallmatrix}\right) \in \Lambda_2$ is *reduced*[1] if $|r| \leq n \leq m$.

The Siegel modular forms with Fourier coefficients supported only on Λ_2^+ are called *Siegel cusp forms.* We denote by S_2^k (resp. $S_2^k(\mathbb{Q})$) the subspace of Siegel cusp forms of M_2^k (resp. $M_2^k(\mathbb{Q})$). The notation S_1^k and $S_1^k(\mathbb{Q})$ is the analog for elliptic cusp forms.

We fix once and for all a basis of $M_2^k(\mathbb{Q})$ of the form

$$F_1, \dots, F_{\ell'}, E_{2,1}^k(f_1), \dots, E_{2,1}^k(f_\ell), E_2^k \tag{1.1}$$

where $F_1, \dots, F_{\ell'}$ and $f_1, \dots, f_\ell$ are respectively a basis of $S_2^k(\mathbb{Q})$ and $S_1^k(\mathbb{Q})$, and where E_2^k and $E_{2,1}^k(f)$ are respectively the (normalized) Siegel Eisenstein series of weight k and the Klingen Eisenstein series attached to the cusp form $f \in S_1^k$. The subspace of Klingen Eisenstein series is denoted by N_2^k. This space has complex dimension equal to the one of S_1^k.

The dual space $M_2^k(\mathbb{Q})$ is generated by the so-called *coefficient extraction functionals* c_T, defined for every $T \in \Lambda_2$ as

$$c_T \colon M_2^k(\mathbb{Q}) \longrightarrow \mathbb{Q}, \qquad F \longmapsto c_T(F).$$

[1]There are several equivalent definitions of *reduced matrix* in the literature. For instance, it is possible to define them as the ones satisfying the relation $|r| \leq m \leq n$ instead.

Every functional c_T can be rewritten over the basis (1.1) as the tuple of rational numbers

$$c_T = \Big(c_T(F_1),\dots,c_T(F_{\ell'}),c_T\big(E^k_{2,1}(f_1)\big),\dots,c_T\big(E^k_{2,1}(f_\ell)\big),c_T(E^k_2)\Big)\in\mathbb{Q}^{1+\ell+\ell'}. \tag{1.2}$$

In [Zuf22b] and [Zuf22a], we defined the *modular cone* $\mathcal{C}_k$ as the (convex) cone in $M^k_2(\mathbb{Q})^*$ generated by all functionals c_T attached to positive definite matrices, namely

$$\mathcal{C}_k = \langle c_T : T\in\Lambda_2^+\rangle_{\mathbb{Q}_{\geq 0}}. \tag{1.3}$$

We used such cone to investigate the properties of cones of codimension 2 special cycles on orthogonal Shimura varieties.

We also classified all accumulation rays of $\mathcal{C}_k$. Among them, there are the rays generated by the points V_m, where $m\in\mathbb{Z}_{>0}$; we refer to [Zuf22b, Section 5.1] for a detailed introduction. Such rays played a key role to prove that the accumulation cone of $\mathcal{C}_k$ is rational and polyhedral.

The program modcone_genus_2 may be used to compute the functionals c_T as in (1.2) and the accumulation rays generated by V_m. It also provides empirical evidences to [Zuf22b, Conjecture 1], i.e. that the modular cone $\mathcal{C}_k$ is polyhedral whenever $k\equiv 2 \bmod 4$ and $k>4$, and certifies such polyhedrality for all $k\leq 38$. The goal of the following sections is to illustrate the features of such program.

2. The modular cone in genus 2

The program modcone_genus_2 has been written using *SageMath Version 8.1*, released on December 7, 2017. To use the program, it is necessary to install the SageMath package degree2 from [Tak17]; see [Tak17, Readme file, Installation] for the procedure to do so.

We remark that degree2, hence also modcone_genus_2, may not work for more recent versions of SageMath. Anyway, it is possible to download Version 8.1 and run it side-by-side with a more recent one.

The first step to use the program is to launch it as `load("modcone_genus_2.sage")`.

2.1. Basis of $M^k_2(\mathbb{Q})$.

```
compute_and_save_bases(k , prec = 30)
```

This command saves the following two files in the folder on which SageMath is working.

- eis$k$_prec_30: a dictionary that may be regarded as the Siegel Eisenstein series E^k_2. As explained in Example 2.1, it is possible to extract the Fourier coefficient of E^k_2 associated to any matrix in Λ_2 with diagonal entries at most prec. The value prec is by default equal to 30.
- basis$k$_prec_30: it may be regarded as a basis of the space $S^k_2\oplus N^k_2$ of Klingen Eisenstein series and cusp forms, namely $F_1,\dots,F_\ell,E^k_{2,1}(f_1),\dots,E^k_{2,1}(f_\ell)$. Each of such modular forms is represented as a list of matrices in Λ_2 with diagonal entries at most prec, together with their associated Fourier coefficients. The saved file may be summoned as illustrated in Example 2.2.

Note that any matrix $\begin{pmatrix} n & r/2\\ r/2 & m\end{pmatrix}\in\Lambda_2$ must be written as (n,r,m) when extracting the Fourier coefficients of the modular forms in the basis computed above.

Example 2.1. Let $k=22$. We launch compute_and_save_bases(22), obtaining two files denominated eis22_prec_30 and basis22_prec_30. We load the former as

```
eis=load("eis22_prec_30")
```

which may be regarded as the Siegel Eisenstein series E^{22}_2. We can compute the Fourier coefficient $c_T(E^{22}_2)$ associated to any matrix $T\in\Lambda_2$ whose diagonal entries are at most 30. For instance, we may compute the Fourier coefficient associated to the zero matrix and to $\begin{pmatrix} 1 & 1/2\\ 1/2 & 1\end{pmatrix}$ as follows.

```
v=eis["fc_dct"]
v[(0,0,0)]
v[(1,1,1)]
```

We obtain respectively 1 and 21294576422083465536/1180857452724874941654444953.

Example 2.2. As in the previous example, we launch `compute_and_save_bases(22)`. We may summon the saved space $S_2^{22} \oplus N_2^{22}$ of Klingen Eisenstein series and Siegel cusp forms as follows.

```
SN=KlingenEisensteinAndCuspForms(22,30)
SN.load_basis_from("basis22_prec_30")
```

We compute the dimension of such space.

```
SN.dimension()
SN.dimension_of_cuspforms()
```

We obtain as output respectively 5 and 4, deducing that $\dim S_2^{22} = 4$ and that $\dim N_2^{22} = 1$. The basis of $S_2^{22} \oplus N_2^{22}$ can be summoned as

```
b=SN.basis()
```

The output `b` is a list of five dictionaries. The first four represent a basis of cusp forms for $S_2^{22}(\mathbb{Q})$. The remaining one is a Klingen Eisenstein series generating $N_2^{22}(\mathbb{Q})$.

2.2. Coefficient extraction functionals in $M_2^k(\mathbb{Q})^*$.

`c_T(`k`, `n`, `r`, `m`, prec = 30)`

This command loads the files generated by `compute_and_save_bases(`k`,prec=30)`, and gives as output a list. The latter is the vector representing the coefficient extraction functional c_T, where $T = \left(\begin{smallmatrix} n & r/2 \\ r/2 & m \end{smallmatrix}\right) \in \Lambda_2^+$ depends on the input, over the basis (1.1) of $M_2^k(\mathbb{Q})$ computed as explained in Section 2.1. We remark that both the input n and m must not exceed the value of precision `prec`.

2.3. Lists of coefficient extraction functionals.

`compute_and_save_coefficient_extraction_functionals(`k` , prec = 30)`

This command loads the files generated by `compute_and_save_bases(`k`,prec=30)`, and saves a file named `coeff_TOT`k`_prec_30`. The latter is a list containing the following three objects. In fact, the most important is the last one.

- `coeff[0]` is a dictionary. Its keys are the determinants of the reduced matrices in Λ_2^+ that have diagonal entries at most `prec`. To every key, the dictionary associates a maximal list of reduced matrices in Λ_2^+ with diagonal entries at most `prec`, determinant equal to the key, and such that the associated coefficient extraction functionals are *pairwise different*.
- `coeff[1]` is a list of numbers. The j-th entry is the number of (pairwise different) coefficient extraction functionals c_T associated to matrices T appearing as values attached to the smallest first j keys of `coeff[0]`.
- `coeff[2]` is a list. Its entries are the (pairwise different) coefficient extraction functionals $c_T \in \mathbb{Q}^{1+\ell+\ell'}$ arising from the matrices indexed in `coeff[0]`. They are ordered with respect to $\det T$, hence the first entry of `coeff[2]` is the functional c_T associated to the matrix $T = \left(\begin{smallmatrix} 1 & 1/2 \\ 1/2 & 1 \end{smallmatrix}\right)$.

Example 2.3. Let $k = 22$. Suppose we have launched `compute_and_save_bases(22)` once, so that the files `eis22_prec_30` and `basis22_prec_30` are already saved in our current folder. We launch `compute_and_save_coefficient_extraction_functionals(22)`, obtaining a new file saved as `coeff_TOT22_prec_30`. We load it as follows.

```
coeff=load("coeff_TOT22_prec_30")
```

The functionals c_T are saved in the list `coeff[2]`. The first entry of such list is the functional

(2.1) `(0, 1, 1, 1, 48384, 212945764220834655536/1180857452724874941654449 53),`

namely the functional c_T associated to the matrix $T = \begin{pmatrix} 1 & 1/2 \\ 1/2 & 1 \end{pmatrix}$, rewritten as a tuple in $\mathbb{Q}^6$ with respect to the basis (1.1) of $M_2^{22}(\mathbb{Q})$ computed as explained in Section 2.1.

2.4. An algorithm to check empirically whether the modular cone $\mathcal{C}_k$ is polyhedral.

Let $k > 4$ be such that $k \equiv 2 \bmod 4$. In [Zuf22b] [Zuf22a] we computed all accumulation rays of the modular cone $\mathcal{C}_k$ with respect to the set of generators used to define it. In [Zuf22b, Theorem 7.5], we provided a sufficient condition for the polyhedrality of $\mathcal{C}_k$, conjecturing that it is always satisfied.

The program `modcone_genus_2` can be used to provide empirical evidences to the polyhedrality of $\mathcal{C}_k$. We illustrate here the idea of such empirical checks, postponing to Section 2.5 the technical explanation of the program. To do so, we need to introduce the following auxiliary cones in $M_2^k(\mathbb{Q})^*$.

Definition 2.4. Let D be the determinant of some matrix in Λ_2^+, and let d be a positive integer. We denote by $\mathcal{C}_k(D, d)$ the polyhedral cone in $M_2^k(\mathbb{Q})^*$ generated by all functionals c_T associated to *reduced* matrices $T \in \Lambda_2^+$ with determinant at most D and diagonal entries at most d, in short

$$\mathcal{C}_k(D, d) = \left\langle c_T : T = \begin{pmatrix} n & r/2 \\ r/2 & m \end{pmatrix} \in \Lambda_2^+ \text{ such that } |r| \le n \le m \le d \text{ and } \det T \le D \right\rangle_{\mathbb{Q}_{\ge 0}}.$$

Saying that $\mathcal{C}_k$ can be generated by a finite number of functionals c_T is equivalent of saying that

(2.2) $\qquad$ if D and d are sufficiently large, then $\mathcal{C}_k(D, d) = \mathcal{C}_k$.

For k and d fixed, the program `modcone_genus_2` may be used to compute $\mathcal{C}_k(D, d)$ for every $D \le d^2$. It also checks whether the sequence of cones

$$\mathcal{C}_k(3/4, d), \qquad \mathcal{C}_k(1, d), \qquad \dots, \qquad \mathcal{C}_k(d^2, d),$$

stabilizes, i.e. whether there exists a D_0 such that $\mathcal{C}_k(D_0, d) = \mathcal{C}_k(D, d)$ for every $D_0 \le D \le d^2$. We refer to Section 2.5 and Example 2.5 for an explicit output in the case of $k = 22$.

2.5. Modular cones in $M_2^k(\mathbb{Q})^*$.

`compute_and_save_cone(`k` , prec = 30)`

This command loads the file saved as `coeff_TOT`k`_prec_30`, see Section 2.3, and produces the following outputs.

- It saves a file named `coneC`k`_prec_30`. It is $\mathcal{C}_k(\texttt{prec}^2, \texttt{prec})$, i.e. the cone in $M_2^k(\mathbb{Q})^*$ generated by all coefficient extraction functionals c_T associated to matrices with diagonal entries at most `prec`, where the latter is by default 30. The cone is computed using the SageMath command `Polyhedron`; see Example 2.6 for the properties that one can extract from the saved file.
- It runs an algorithm to provide empirical evidences to (2.2), as follows. For every determinant D of some reduced matrix in Λ_2^+ with diagonal entries at most `prec`, it computes the cone $\mathcal{C}_k(D, \texttt{prec})$ introduced in Section 2.4. The latter is then printed on the SageMath console. The program also checks whether $\mathcal{C}_k(D, \texttt{prec})$ equals the analogous cone associated to the subsequent determinant; see Example 2.5.

Example 2.5. Let $k = 22$. We launch `compute_and_save_cone(22)`, obtaining the following printed text. We explain it piece by piece.

```
CASE OF WEIGHT k=22.
With just the first determinant, we get:  A 1-dimensional polyhedron in QQ^6 defined as
the convex hull of 1 vertex and 1 ray.
```

Here the program computes the cone generated by the functionals c_T, where $T \in \Lambda_2^+$ are reduced matrices with diagonal entries at most 30 and with minimal determinant. The minimal determinant is $D = 3/4$, and the only T as above is $T = \left(\begin{smallmatrix} 1 & 1/2 \\ 1/2 & 1 \end{smallmatrix}\right)$. The resulting cone is $\mathcal{C}_k(3/4, 30)$, which is made only of one ray. It lies in $\mathbb{Q}^6$, since $\dim M_2^k(\mathbb{Q})^* = 6$.

```
Now we begin to increase the determinants.
Different cone!  We get:  A 2-dimensional polyhedron in QQ^6 defined as the convex hull
of 1 vertex and 2 rays.
Number of determinants considered:  2.
```

The program computes the cone generated by the functionals c_T, where $T \in \Lambda_2^+$ are reduced matrices with diagonal entries at most `prec`$= 30$ and with determinant at most the second smallest one. The resulting cone is 2-dimensional with two extremal rays.

```
Different cone!  We get:  A 3-dimensional polyhedron in QQ^6 defined as the convex hull
of 1 vertex and 3 rays.
Number of determinants considered:  3.
Different cone!  We get:  A 4-dimensional polyhedron in QQ^6 defined as the convex hull
of 1 vertex and 4 rays.
Number of determinants considered:  4.
```

The program iterates the previous construction twice, increasing the determinants D considered. At every iteration, the arising cone is different from the previous one. Its dimension and number of extremal rays grow.

```
Same cone as above.
Number of determinants considered:  5.
```

At this iteration, the program tells us that the arising cone is the same as the one of the previous iteration.

```
Different cone!  We get:  A 5-dimensional polyhedron in QQ^6 defined as the convex hull
of 1 vertex and 5 rays.
Number of determinants considered:  6.
Different cone!  We get:  A 6-dimensional polyhedron in QQ^6 defined as the convex hull
of 1 vertex and 6 rays.
Number of determinants considered:  7.
Different cone!  We get:  A 6-dimensional polyhedron in QQ^6 defined as the convex hull
of 1 vertex and 7 rays.
Number of determinants considered:  8.
```

The program iterates the previous construction again. At every iteration, it finds a different cone. We can observe that at the 7-th iteration the cone become full-dimensional. Still, the number of extremal rays grows.

```
Same cone as above.
Number of determinants considered:  1545.
A 6-dimensional polyhedron in QQ^6 defined as the convex hull of 1 vertex and 7 rays
```

The program iterates the construction, and tells us that the arising cone is the same as the previous one. In fact, it remains the same even when it adds all remaining functionals available, i.e. until it reaches the last 1545-th determinant saved in `coeff_TOT22_prec_30`.

Example 2.6. Let $k = 22$. Suppose we have launched `compute_and_save_cone(22)` once, so that the file `coneC22_prec_30` is saved in our current folder. We load it as follows.

```
C=load("coneC22_prec_30")
C
```

The output says that `C` is a 6-dimensional polyhedron in $\mathbb{Q}^6$ defined as the convex hull of 1 vertex and 7 rays, meaning that it is a full-dimensional cone in $\mathbb{Q}^6$ with 7 extremal rays. A list of generators of such extremal rays may be extracted with the command `C.rays_list()`. For instance, one can see that the functional (2.1) generates one of those rays.

3. Back in genus 1

Let k be a positive integer such that $k \equiv 2 \bmod 4$. In [BM19], Bruinier and Möller considered cones of functionals on the space of elliptic modular forms $M_1^k(\mathbb{Q})$. In this section we illustrate how to use `modcone_genus_2` to construct such polyhedral cones.

We denote the Fourier expansion of any modular form $f \in M_1^k(\mathbb{Q})$ by

$$f(\tau) = \sum_{n \geq 0} c_n(f) \cdot e^{2\pi i n \tau}.$$

In this setting, the coefficient extraction functionals $c_n \in M_1^k(\mathbb{Q})^*$ are defined as the linear maps

$$c_n \colon M_1^k(\mathbb{Q}) \longrightarrow \mathbb{Q}, \qquad f \longmapsto c_n(f).$$

We may represent such functional over a basis of the form

$$f_1, \dots, f_\ell, E_1^k, \tag{3.1}$$

where $f_1, \dots, f_\ell$ is a basis of $S_1^k(\mathbb{Q})$ and E_1^k is the (normalized) elliptic Eisenstein series of weight k. In this way the functional c_n may be considered as the tuple of rational numbers

$$c_n = \Big(c_n(f_1), \dots, c_n(f_\ell), c_n(E_1^k) \Big) \in \mathbb{Q}^{1+\ell}.$$

The *modular cone of genus* 1 is the cone $\mathcal{C}_{1,k}$ defined in $M_1^k(\mathbb{Q})^*$ as

$$\mathcal{C}_{1,k} \coloneqq \langle c_n : n \geq 1 \rangle_{\mathbb{Q}_{\geq 0}}. \tag{3.2}$$

By [BM19, Theorem 3.4], such cone is polyhedral. This is deduced by showing that the dual of E_1^k, represented over the basis (3.1) as the vector $(0, \dots, 0, 1)$, generates the only accumulation ray of $\mathcal{C}_{1,k}$, and that such ray is internal in $\mathcal{C}_{1,k}$.

Let $\mathcal{C}_{1,k}(m)$ be the polyhedral cone in $M_1^k(\mathbb{Q})^*$ defined as the convex span of the first m coefficient extraction functionals used to define $\mathcal{C}_{1,k}$, namely

$$\mathcal{C}_{1,k}(m) = \langle c_n : 1 \leq n \leq m \rangle_{\mathbb{Q}_{\geq 0}}.$$

Since the modular cone of genus 1 is polyhedral, there exists a value $n_0 = n_0(k)$ such that $\mathcal{C}_{1,k}$ equals $\mathcal{C}_{1,k}(n_0)$.

3.1. **Coefficient extraction functionals.**

`gen1_c_n(`k`, `n`)`

This command returns the coefficient extraction functional $c_n \in M_1^k(\mathbb{Q})$ rewritten over the basis (3.1) computed by SageMath via `ModularForms(1,k).basis()`

3.2. **Modular cones of genus 1.**

`gen1_compute_and_save_cone(`k`, max = 100)`

This command computes the cone $\mathcal{C}_{1,k}$(`max`), producing the following output.

- It saves a file named `coneC`k`_max_100`. It is $\mathcal{C}_{1,k}$(`max`), represented over the basis (3.1) computed by SageMath via `ModularForms(1,k).basis()`. Such convex cone is also given as output. The value `max` is by default 100. The cone is computed using the SageMath command `Polyhedron`.
- It checks whether the dual of E_1^k, namely the vector $(0, \dots, 0, 1)$, is either contained in the interior, contained on the boundary, on not contained in $\mathcal{C}_{1,k}$(`max`)

There are other two functions defined analogously. The first is

```
gen1_compute_and_save_cone_Hecke(k, max = 100),
```

which is as `gen1_compute_and_save_cone`, but the cone is rewritten with respect to the basis of (normalized) Hecke forms $M_1^k(\mathbb{R})$ extracted using `CuspForms(1,k).newforms(names="a")`. The cone is saved as `coneCk_max_100_Hecke`.

The second one is

```
gen1_compute_cone_step_by_step(k, max = 100),
```

which is as `gen1_compute_and_save_cone`, but it does not return and save any cone. Instead, it computes the cone $\mathcal{C}_{1,k}(n)$ for every $n \leq$ `max`, comparing it with the subsequent one.

3.3. How to certify the computation of modular cones of genus 1.

```
gen1_check_if_internal_Hecke(k, m, max = 100, accumulation = False)
```

The entry m must be at most `max`. This command loads the polyhedral cone $\mathcal{C}_{1,k}(\texttt{max})$ written with respect to a basis of Hecke forms, previously saved as `coneCk_max_100_Hecke`, and certifies whether $\mathcal{C}_{1,k}(m)$ equals $\mathcal{C}_{1,k}$, i.e. whether the weight k modular cone of genus 1 is generated by the first m coefficient extraction functionals. The optional input `accumulation` will be relevant in Section 4.3.

The idea of the algorithm is as follows. Suppose that the basis $f_1, \dots, f_\ell$ of $S_1^k(\mathbb{R})$ is made of (normalized) Hecke forms. We denote by $\sigma_s(n)$ the sum of the s-th powers of the positive divisors of n. By [Del74, Théorème 18] and the trivial inequality $\sigma_0(n) \leq 2\sqrt{n}$, the Fourier coefficients of the Hecke forms may be bounded as

$$|c_n(f_j)| \leq \sigma_0(n) \cdot n^{(k-1)/2} \leq 2n^{k/2}, \qquad \text{for every } n \in \mathbb{N}.$$

Recall that we may rewrite the normalized functional $c_n/c_n(E_1^k)$ over the chosen basis of Hecke forms as

$$\frac{c_n}{c_n(E_1^k)} = \Big(\frac{c_n(f_1)}{c_n(E_1^k)}, \dots, \frac{c_n(f_\ell)}{c_n(E_1^k)}, 1\Big).$$

Since $c_n(E_1^k) = 2\sigma_{k-1}(n)/\zeta(1-k)$ and $\sigma_{k-1}(n) \geq n^{k-1}$ for every $n \geq 1$, we may bound the distance between $c_n/c_n(E_1^k)$ and the dual $(0, \dots, 0, 1)$ of the Eisenstein series E_1^k as

$$\Big\|\Big(\frac{c_n(f_1)}{c_n(E_1^k)}, \dots, \frac{c_n(f_\ell)}{c_n(E_1^k)}, 0\Big)\Big\| = \frac{|\zeta(1-k)|}{2\sigma_{k-1}(n)}\Big(\sum_{j=1}^{\ell} c_n(f_j)^2\Big)^{1/2} \leq \Big(\frac{\ell \cdot \zeta(1-k)^2}{n^{k-2}}\Big)^{1/2}. \tag{3.3}$$

Let $r(k,n)$ be the number appearing on the right-hand side of (3.3). Note that $r(k,n) \to 0$ when n diverges. We have just shown that the normalized functional $c_n/c_n(E_1^k)$ lies in the ball $B(k,n)$ of radius $r(k,n)$ centered in the dual $(0, \dots, 0, 1)$ of the Eisenstein series E_1^k, for every $n \geq 1$, in short

$$\frac{c_n}{c_n(E_1^k)} \in B(k,n) = \big\{P : d\big(P, (0, \dots, 0, 1)\big) \leq r(k,n)\big\} \subset M_1^k(\mathbb{R})^*.$$

The command `gen1_check_if_internal_Hecke` checks whether the ball $B(k,m)$ is contained in the *interior* of $\mathcal{C}_{1,k}(\texttt{max})$. Whenever this happens, since $m <$ `max`, the cone $\mathcal{C}_{1,k}(\texttt{max})$ equals the whole modular cone $\mathcal{C}_{1,k}$.

Example 3.1. Let $k = 50$. Suppose that we have already computed the truncated modular cone $\mathcal{C}_{1,50}(100)$ once. We summon it as

```
PP=load("coneC50_max_100_Hecke")
```

We certify that such cone equals the weight 50 modular cone of genus 1, i.e. $\mathcal{C}_{1,50}$. To do so, we launch

```
gen1_check_if_internal_Hecke(50, 11)
```

obtaining as output

```
Yes, all functionals c_n, where n >= 11, are contained in the *interior* of the
saved cone!
```

4. The accumulation cones

From now on, we fix an integer $k > 4$ such that $k \equiv 2 \bmod 4$, and work over $\mathbb{R}$ and $\mathbb{Q}$ interchangeably. In fact, every accumulation ray is generated by some element of $M_2^k(\mathbb{Q})^*$.

As proved in [Zuf22b], the accumulation cone $\mathcal{A}_k$ of the modular cone $\mathcal{C}_k$ may be generated as

$$\mathcal{A}_k = \langle V_m : m > 0 \rangle_{\mathbb{R}_{\geq 0}}.$$

The points V_m written over the basis (1.1) of $M_2^k(\mathbb{R})$ are

$$V_m = \Big(0, \dots, 0, \frac{\zeta(1-k)}{2}\alpha_m(1, f_1), \dots, \frac{\zeta(1-k)}{2}\alpha_m(1, f_\ell), 1\Big),$$

where $\alpha_m(1, f_j)$ is defined as in [Zuf22b, Proposition 3.23]. If we forget about the first $\dim S_2^k(\mathbb{R})$ entries, the points V_m may be written over the associated basis (3.1) of $M_1^k(\mathbb{R})$. We may furthermore assume that the chosen basis $f_1, \dots, f_\ell$ of $S_1^k(\mathbb{R})$ is made of Hecke forms.

We denote by $\mathcal{A}_k(t)$ the truncated accumulation cone generated by the first t points V_m, in short

$$\mathcal{A}_k(t) = \langle V_m : 1 \leq m \leq t \rangle_{\mathbb{R}_{\geq 0}}.$$

4.1. The accumulation rays generated by V_m.

`V_m(`k`, `m`, prec = 30)`

This command loads the file saved as `basis`k`_prec_30`, see Section 2.1, and produces as output the vector V_m written with respect to the basis (1.1). The value m must be at most `prec`, which is by default 30.

Example 4.1. Let $k = 22$. Suppose we have launched `compute_and_save_basis(22)` once, so that the file `basis22_prec_30` is saved in our current folder. We compute the vectors V_1 and V_2 as follows.

```
V1=V_m(22,1)
V2=V_m(22,2)
```

We may check whether V_1 and V_2 are contained in the truncated cone $\mathcal{C}_{22}(900, 30)$ as follows.

```
C=load("coneC22_prec_30")
C.contains(V1)
C.contains(V2)
```

We obtain for both vectors the output `True`, meaning that $V_1, V_2 \in \mathcal{C}_{22}(900, 30)$.

We may check whether V_1 and V_2 are contained in the *interior* of $\mathcal{C}_{22}(900, 30)$ as follows.

```
C.interior_contains(V1)
C.interior_contains(V2)
```

As output, we obtain `False` for V_1 and `True` for V_2.

4.2. The computation of accumulation cones.

`accumulation_cone_Hecke(`k`, max = 100)`

This command computes the truncated accumulation cone $\mathcal{A}_k$(`max`), rewritten with respect to the basis (3.1) of Hecke forms of $M_1^k(\mathbb{R})^*$, as illustrated at the beginning of Section 4. The basis of $S_1^k(\mathbb{R})$ is extracted using `CuspForms(1,`k`).newforms(names="a")`. The output is a convex cone computed via the command `Polyhedron`, and is saved as `coneC`k`_100_Hecke_with_Vm`.

Example 4.2. Let $k = 50$. We write

```
P=accumulation_cone_Hecke(50)
```

obtaining the truncated accumulation cone $\mathcal{A}_{50}(100)$. If we print `P`, we see that such cone is full-dimensional in $M_1^{50}(\mathbb{R})^*$, of dimension 4 and with 4 extremal rays.

Suppose that we have already computed the truncated modular cone $\mathcal{C}_{1,50}(100)$ once, over a basis of Hecke forms, as explained in Section 3.2. We summon it as

```
PP=load("coneC50_max_100_Hecke")
```

This is in fact the whole weight 50 modular cone of genus 1, as we certified in Example 3.1.

In [Zuf22b], we proved that the accumulation cone $\mathcal{A}_k$ contains always the rank one modular cone $-\mathcal{C}'_k$. In this setting, i.e. with $k = 50$ and over a basis of $M_1^{50}(\mathbb{R})$ instead of $M_2^{50}(\mathbb{R})$, the previous sentence is implied by the inclusion $-\mathcal{C}_{1,50}(100) \subseteq \mathcal{A}_{50}(100)$. The latter can be checked as follows.

We construct the cone $-\mathcal{C}_{1,50}(100)$ as

```
v=[]
for r in PP.rays_list():
  v=v+[list(-vector(r))]
minusPP=Polyhedron(rays=v)
```

We may then check that $-\mathcal{C}_{1,50}(100) \subseteq \mathcal{A}_{50}(100)$ writing

```
minusPP.intersection(P)==minusPP
```

which gives `True` as output.

Note that the accumulation cone $\mathcal{A}_{50}$ is *strictly larger* than $-\mathcal{C}_{1,50}(100)$, i.e. $-\mathcal{C}_{1,50}(100)$ is not equal to $\mathcal{A}_{50}$. We check this by writing

```
P.intersection(minusPP)==minusPP
P==minusPP
```

which gives respectively `True` and `False` as output.

4.3. **How to certify the computation of accumulation cones.**

`gen1_check_if_internal_Hecke(`k`, `m`, max = 100, accumulation = True)`

The entry m must be at most `max`. This command loads the truncated accumulation cone $\mathcal{A}_k$(`max`) written with respect to a basis of Hecke forms, saved as `coneC`k`_max_100_Hecke_with_Vm`, and certifies whether $\mathcal{A}_k(m) = \mathcal{A}_k$, i.e. whether $\mathcal{A}_k$ is generated by the points V_s with $s \leq m$.

Since the idea of the implemented algorithm is the same as in Section 3.3, we skip many details. Recall that we may rewrite the point V_m over the chosen basis of Hecke forms as

$$V_m = \Big(\frac{\zeta(1-k)}{2}\alpha_m(1,f_1),\dots,\frac{\zeta(1-k)}{2}\alpha_m(1,f_\ell),1\Big).$$

By [Zuf22b, Lemma 5.9], the points V_m converge to $P_\infty := (0,\dots,0,1)$ when m diverges. We may bound the distance between V_m and P_∞ as

$$\begin{aligned}&\Big\|\Big(\frac{\zeta(1-k)}{2}\alpha_m(1,f_1),\dots,\frac{\zeta(1-k)}{2}\alpha_m(1,f_\ell),0\Big)\Big\| = \frac{|\zeta(1-k)|}{2}\Big(\sum_{j=1}^{\ell}\alpha_m(1,f_j)^2\Big)^{1/2} = \\ &= \frac{|\zeta(1-k)|}{2g_k(m)}\Big(\sum_{j=1}^{\ell} g(f_j,m)^2\Big)^{1/2} \leq \frac{|\zeta(1-k)|}{m^{k-1}}\Big(\ell m^k\Big(\sum_{g^2|m}1\Big)^2\Big)^{1/2} \leq \Big(\frac{\ell\cdot\zeta(1-k)^2}{m^{k-3}}\Big)^{1/2}.\end{aligned} \tag{4.1}$$

Let $r'(k,m)$ be the number appearing on the right-hand side of (4.1). Note that $r'(k,m) \to 0$ when m diverges. We have just shown that the point V_m lies in the ball $B(k,m)$ of radius $r'(k,m)$

centered in P_∞, for every $m \geq 1$, in short

$$V_m \in B(k,m) = \{P : d(P, P_\infty) \leq r'(k,m)\} \subset M_1^k(\mathbb{R})^*.$$

The command `gen1_check_if_internal_Hecke` checks whether the ball $B(k,m)$ is contained in the *interior* of $\mathcal{A}_k(\texttt{max})$. Whenever this happens, since $m \leq \texttt{max}$, the cone $\mathcal{A}_k(\texttt{max})$ equals the whole accumulation cone $\mathcal{A}_k$.

5. Some results and some empirical evidences

We suppose that $k > 4$ and $k \equiv 2 \bmod 4$. In this section we certify the polyhedrality of $\mathcal{C}_k$ for k small enough. Moreover, we collect some of the empirical evidences on the polyhedrality of $\mathcal{C}_k$ for larger values of k.

5.1. How to certify the polyhedrality of $\mathcal{C}_k$ when k is small.

In [Zuf22b], we proved the following result.

Proposition 5.1 (Zuffetti). *If $k \leq 38$, then the cone $\mathcal{C}_k$ is polyhedral.*

The proof of it relies on SageMath computations that may be achieved via `modcone_genus_2`, as we briefly illustrate.

The computation of the truncated cone $\mathcal{C}_k(900, 30)$ and of V_m for all $m \leq 30$ is made as explained respectively in Section 2.5 and Section 4.1.

The check whether V_m lies on the boundary of $\mathcal{C}_k(900, 30)$ can be made using the command `Polyhedron` already implemented in SageMath following the same pattern of Example 4.1.

The check whether the points V_m with $m \leq 30$ generate the whole accumulation cone $\mathcal{A}_k$ is made as explained in Section 4.3.

In the following table we summarize the outputs computed as above. We discard the trivial case of $k = 6$, since $\dim M_2^6 = 1$. The cases of $k = 10, 14$ are the easiest, since there is no non-zero elliptic cusp form of such weight, hence there is no non-zero Klingen Eisenstein series in M_2^k.

With "minimum number of determinants to have a stable cone", we mean the minimum number of D's needed from `compute_and_save_cone` to reach a determinant D_0 such that $\mathcal{C}_k(D_0, 30)$ equals $\mathcal{C}_k(900, 30)$.

k	$\dim \mathcal{C}_k(900, 30)$	Minimum num. of det. to have a stable cone	Number of extremal rays	$m \leq 30$ such that $V_m \in \partial\mathcal{C}_k(900, 30)$	m such that $\mathcal{A}_k(m) = \mathcal{A}_k$
10	2	2	2	none	1
14	2	2	2	none	1
18	4	8	6	V_1	3
22	6	8	7	V_1, V_2	3
26	7	10	8	V_1, V_2	4
30	11	18	16	V_1, V_2	5
34	14	18	16	V_1, V_2, V_3	6
38	16	24	21	V_1, V_2, V_3	7

5.2. Certain interesting subcones of $\mathcal{C}_k$.

We proved in [Zuf22b, Lemma 7.6] that if m is squarefree, then the subcone $\mathcal{R}_m$ of $\mathcal{C}_k$ defined as

$$\mathcal{R}_m = \langle c_T : T \in \Lambda_2^+ \text{ with } m \text{ as bottom-right entry}\rangle_{\mathbb{Q}_{\geq 0}} \tag{5.1}$$

contains V_m in its interior. This discards V_m from being an extremal ray that gives roundness to $\mathcal{C}_k$.

In [Zuf22b, Example 7.8], we illustrated how such construction does not seem to be valid if m is non square-free. We conjectured that the correct way to construct $\mathcal{R}_m$, so that V_m is internal in it, is

$$\mathcal{R}_m = \left\langle c_T : T = \left(\begin{smallmatrix} n & r/2 \\ r/2 & m \end{smallmatrix}\right) \in \Lambda_2^+ \text{ such that if } t^2 | m \text{ with } t \neq 1, \text{ then } t \nmid r \right\rangle_{\mathbb{Q}_{\geq 0}}. \tag{5.2}$$

Note that the latter coincide with (5.1) if m is square-free.

The command

```
subcone_for_V_m(k, m, prec = 30)
```

loads the basis of $M_2^k(\mathbb{Q})$ constructed as in Section 2.1, and provides as output the *truncation* of the cone $\mathcal{R}_m$ constructed using only those generators whose matrix T has diagonal entries at most `prec`. Such cone is then saved as `subconeR`k`_prec_30_for_V_`m.

The command

```
subcone_for_V_m_enlarged(k, m, prec = 30)
```

works in a similar way, but whenever m is non square-free, it computes the truncation of the cone (5.1) instead of (5.2). These are the commands used to construct the table in [Zuf22b, Example 7.8].

References

[BM19] J. Bruinier and M. Möller. "Cones of Heegner divisors". In: *J. Algebraic Geom.* 28.3 (2019).

[Del74] P. Deligne. "La conjecture de Weil. I". In: *Inst. Hautes Études Sci. Publ. Math.* 43 (1974), pp. 273–307.

[Tak17] S. Takemori. *A Sage package for computation of degree 2 Siegel modular forms.* `https://github.com/stakemori/degree2/`. 2017.

[Zuf22a] R. Zuffetti. *Cones of special cycles and unfolding of the Kudla–Millson lift.* PhD thesis. 2022.

[Zuf22b] R. Zuffetti. *Cones of special cycles of codimension 2 on orthogonal Shimura varieties.* Preprint on arXiv. 2022. URL: `https://arxiv.org/abs/2202.12610`.