Journal of Software for Algebra and Geometry

admcycles - a Sage package for calculations in the tautological ring of the moduli space of stable curves

VINCENT DELECROIX, JOHANNES SCHMITT AND JASON VAN ZELM

vol 11 2021

JSAG **11** (2021), 89–112
https://doi.org/10.2140/jsag.2021.11.89

The Journal of Software for
Algebra and Geometry

admcycles - a Sage package for calculations in the tautological ring of the moduli space of stable curves

VINCENT DELECROIX, JOHANNES SCHMITT AND JASON VAN ZELM

ABSTRACT: The tautological ring of the moduli space of stable curves has been studied extensively in the last decades. We present a SageMath implementation of many core features of this ring. This includes lists of generators and their products, intersection numbers and verification of tautological relations. Maps between tautological rings induced by functoriality, that is pushforwards and pullbacks under gluing and forgetful maps, are implemented. Furthermore, many interesting cycle classes, such as the double ramification cycles, strata of k-differentials and hyperelliptic or bielliptic cycles are available. We show how to apply the package, including concrete example computations.

1. INTRODUCTION. A crucial tool in the study of the singular cohomology of the moduli space $\overline{\mathcal{M}}_{g,n}$ of stable curves is the tautological ring

$$\mathrm{RH}^*(\overline{\mathcal{M}}_{g,n}) \subset H^*(\overline{\mathcal{M}}_{g,n}) = H^*(\overline{\mathcal{M}}_{g,n}, \mathbb{Q}).$$

It is a $\mathbb{Q}$-subalgebra of the singular cohomology of $\overline{\mathcal{M}}_{g,n}$ with an explicit, finite set of generators (indexed by decorated graphs $[\Gamma, \alpha]$) admitting combinatorial descriptions of operations like cup products and intersection numbers. For a detailed introduction to the tautological ring, see, e.g., [Faber and Pandharipande 2000; Arbarello et al. 2011; Pandharipande 2018].

Since computations with the generators $[\Gamma, \alpha]$ quickly become untractable by hand, it is natural to implement them in a computer program. With `admcycles` we present such an implementation using the open source mathematical software [SageMath]. It is based on an earlier implementation by Aaron Pixton. It features intersection products and numbers between the classes $[\Gamma, \alpha]$ and verification of linear relations between these generators using the known generalized Faber–Zagier relations [Pixton 2012; Pandharipande et al. 2015; Janda 2017]. For the gluing and forgetful morphisms between (products of) the moduli spaces $\overline{\mathcal{M}}_{g,n}$ it implements pullbacks and pushforwards of the generators $[\Gamma, \alpha]$ of the tautological ring.

Many geometric constructions of cohomology classes on $\overline{\mathcal{M}}_{g,n}$ (such as the Chern classes λ_d of the Hodge bundle $\mathbb{E}$ over $\overline{\mathcal{M}}_{g,n}$) give classes contained in the tautological ring and can thus be written as linear combinations of classes $[\Gamma, \alpha]$. For many examples of such classes, the package `admcycles`

MSC2010: 14H10, 97N80.
Keywords: moduli of curves, tautological ring, intersection theory, double ramification cycle.
`admcycles` version 1.3.1

implements known formulas or algorithms to calculate them and thus allows further computations, such as intersections or comparisons to other cohomology classes. In particular, `admcycles` contains

- a formula for double ramification cycles $\mathrm{DR}_g(A)$ from [Janda et al. 2017] ,
- a conjectural formula for the strata $\overline{\mathcal{H}}_g^k(\boldsymbol{m})$ of k-differentials from [Farkas and Pandharipande 2018; Schmitt 2018],
- (generalized) lambda classes, the Chern classes of derived pushforwards $R^\bullet\pi_*\mathcal{O}(D)$ of divisors D on the universal curve $\pi : \mathcal{C}_{g,n} \to \overline{\mathcal{M}}_{g,n}$, as discussed in [Pagani et al. 2020],
- admissible cover cycles,[1] such as the fundamental classes of loci of hyperelliptic or bielliptic curves with marked ramification points, as discussed in [Schmitt and van Zelm 2020]

Instead of discussing the details of the algorithms in `admcycles`, this document serves as a user manual for the package, with an emphasis on concrete example computations. These computations are also available in an interactive online format on CoCalc (without need for registration) here.

One way to explore `admcycles` is to go through these examples and refer back to the text below for additional explanations and background. While the code in the examples is mostly self-explanatory, some basic familiarity with SageMath and the Python programming language (e.g., as explained in the official SageMath tutorial) is helpful.

***Applications of* `admcycles`.** By now the package `admcycles` has been used in a variety of contexts. Its original purpose was computing new examples of admissible cover cycles in [Schmitt and van Zelm 2020], e.g., computing the class of the hyperelliptic locus in $\overline{\mathcal{M}}_5$ and $\overline{\mathcal{M}}_6$ and the locus of bielliptic cycles in $\overline{\mathcal{M}}_4$. It was also used to verify results about Hodge integrals on bielliptic cycles in [Pandharipande and Tseng 2019] and on loci of cyclic triple covers of rational curves in [Owens and Somerstep 2019].

Buryak and Rossi [2021] used `admcycles` to explore formulas for intersection numbers involving double ramification cycles and lambda classes. The implementation of generalized lambda classes led to the discovery of previously missing terms in the computations of [Pagani et al. 2020] when doing comparisons with double ramification cycles. The package was also used in [Chen et al. 2019] to verify computations of Masur–Veech volumes in terms of intersection numbers on $\overline{\mathcal{M}}_{g,n}$. It was used to check a new recursion for intersection numbers of ψ-classes presented in [Grosse et al. 2019] and formulas for double Hurwitz numbers in terms of intersection numbers in [Borot et al. 2020] and [Do and Lewański 2020]. In [Castorena and Gendron 2020], which computes a fundamental class of a stratum of meromorphic differentials in genus 3, some errors have been found and corrected after comparing the result with the output of `admcycles`. More recently, in [Bae and Schmitt 2020] some code based on `admcycles` was used to compute ranks of Chow groups of moduli stacks $\mathfrak{M}_{0,n}$ of prestable curves. Molcho et al. [2021] applied the package to verify the completeness of the generalized Faber–Zagier relations in two new cases on $\overline{\mathcal{M}}_{4,1}$ and $\overline{\mathcal{M}}_{5,1}$ and used this to show that for $g \geq 7$ the class λ_g is not contained in the subring of the

[1] Computing these cycles was the original purpose of `admcycles`, hence the name of the package.

cohomology of $\overline{\mathcal{M}}_g$ generated by classes of cohomological degree at most 4. Very recently, Canning and Larson [2021] used `admcycles` for computing the rational Chow rings of the spaces $\mathcal{M}_g$ for $g = 7, 8, 9$.

Other implementations. Apart from `admcycles` (and the code of Pixton on which it is based) there have been several other implementations of the tautological ring, starting with Faber's program [1999] for computing intersection numbers of divisors and Chern classes of the Hodge bundle. Yang [2008] presents a program computing intersection pairings of tautological classes on various open subsets of $\overline{\mathcal{M}}_{g,n}$. The package [mgn] by Johnson implements general intersections of the $[\Gamma, \alpha]$ and also verification of linear relations between these generators against the known generalized Faber–Zagier relations.

Based on `admcycles` there is the new SageMath-package `diffstrata` (included in `admcycles` since version 1.1) by Costantini, Möller and Zachhuber. It implements the tautological ring and intersection products on the smooth compactification of the strata of differentials presented in [Bainbridge et al. 2019a]. Computations with `diffstrata` are used in [Costantini et al. 2020a] to evaluate formulas for Euler characteristics of strata of differentials in examples. Similar to the present paper, a detailed description of the package `diffstrata` is given in [Costantini et al. 2020b].

1.1. ***Conventions.*** Let $\overline{\mathcal{M}}_{g,n}$ be the moduli space of stable curves and $\pi : \overline{\mathcal{M}}_{g,n+1} \to \overline{\mathcal{M}}_{g,n}$ be the forgetful morphism of the marking $n+1$, which can be seen as the universal curve over $\overline{\mathcal{M}}_{g,n}$. Let $\sigma_i : \overline{\mathcal{M}}_{g,n} \to \overline{\mathcal{M}}_{g,n+1}$ be the section of π corresponding to the i-th marked point ($i = 1, \dots, n$). For ω_π the relative dualizing line bundle of π on the space $\overline{\mathcal{M}}_{g,n+1}$ and $i = 1, \dots, n$ we define the ψ-class

$$\psi_i = c_1(\sigma_i^* \omega_\pi) \in H^2(\overline{\mathcal{M}}_{g,n}).$$

For $a = 0, 1, 2, \dots$ we define the (Arbarello–Cornalba) κ-class

$$\kappa_a = \pi_*((\psi_{n+1})^{a+1}) \in H^{2a}(\overline{\mathcal{M}}_{g,n}).$$

Finally, given a stable graph Γ of genus g with n legs, let

$$\xi_\Gamma : \overline{\mathcal{M}}_\Gamma = \prod_{v \in V(\Gamma)} \overline{\mathcal{M}}_{g(v),n(v)} \to \overline{\mathcal{M}}_{g,n}$$

be the gluing map associated to Γ. For a class $\alpha \in H^*(\overline{\mathcal{M}}_\Gamma)$ given as a product of κ and ψ-classes on the factors $\overline{\mathcal{M}}_{g(v),n(v)}$, define

$$[\Gamma, \alpha] = (\xi_\Gamma)_* \alpha \in H^*(\overline{\mathcal{M}}_{g,n}).$$

Such decorated boundary strata form a generating set (as a $\mathbb{Q}$-vector space) of the tautological ring $\mathrm{RH}^*(\overline{\mathcal{M}}_{g,n})$.

Note: The degree of the gluing map ξ_Γ to its image is given by the size $|\mathrm{Aut}(\Gamma)|$ of the automorphism group of Γ. Therefore many authors prefer to define $[\Gamma, \alpha]$ as $1/|\mathrm{Aut}(\Gamma)| \cdot (\xi_\Gamma)_* \alpha$ (so that $[\Gamma, 1]$ equals the class of the boundary stratum of $\overline{\mathcal{M}}_{g,n}$ associated to Γ). However, throughout this paper and in the package `admcycles`, we take the convention of *not* dividing by the size $|\mathrm{Aut}(\Gamma)|$ of the automorphism group of Γ.

2. GETTING STARTED. The `admcycles` package works on top of SageMath which is an open source software for mathematical computations. We describe how to install SageMath and `admcycles` on a computer and how to use the available online services.

2.1. `admcycles` ***in the cloud.*** The simplest way to play with `admcycles` without installing anything beyond a web browser is to use one of [SageMathCell] or the website [CoCalc]. The former provides a basic interface to SageMath. The latter requires registration and allows one to create worksheets that can easily be saved and shared. As mentioned before, it is possible to explore the computations presented below on share.cocalc.com without the need to register.

2.2. ***Obtaining SageMath.*** SageMath is available on most operating systems. Depending on the situation one can find it in the list of softwares available from the package manager of the operating system. Alternatively, there are binaries available from the SageMath website . Lastly, one can compile it from the source code. More information on the installation process can be found here.

2.3. ***Installation of the*** `admcycles` ***package.*** The package `admcycles` is available from the Python Package Index (PyPI) where detailed installation instructions are available for a range of systems. Note that the best performance (in particular for functions like `DR_cycle`) is obtained using version 9.0 of SageMath or newer.

The package `admcycles` is being developed on GitLab where one can find the latest development version and a link to report bugs. This is also the place to look at to suggest features or improvements.

2.4. ***First step with*** `admcycles`***.*** Once successfully installed, to use `admcycles` one should start a SageMath-session and type

```
sage: from admcycles import *
```

In the sample code, we reproduce the behavior of the SageMath console that provides the `sage:` prompt on each input line. When using the online SageMathCell or a Jupyter worksheet, there is no need to write `sage:`. In all our examples, this `sage:` prompt allows one to distinguish between the input (command) and the output (result). *All other examples below assume that the line*

```
from admycles import *
```

has been executed before.

In addition to this manual, the package has an internal documentation with more information concerning the various functions. To access additional information about some function or object `foo`, type `foo?` during the SageMath session; e.g.,

```
sage: TautologicalRing?
```

3. TAUTOLOGICAL RING AND CLASSES. The main objects in admcycles to manipulate tautological classes are `TautologicalRing` and `TautologicalClass`.

3.1. ***Creating tautological rings.*** A convenient way to start a computation in the tautological ring of $\overline{\mathcal{M}}_{g,n}$ is to construct the appropriate ring itself by calling the function `TautologicalRing(g, n)`.

```
sage: R = TautologicalRing(1, 1); R
TautologicalRing(g=1, n=1, moduli='st') over Rational Field
```

As we explain in Section 3.2, the object `R` above then allows easy access to many of the standard tautological classes on $\overline{\mathcal{M}}_{g,n}$. As an example, we show how to compute the integral

$$\int_{\overline{\mathcal{M}}_{1,1}} \psi_1 = \tfrac{1}{24}$$

using the ring `R` we created above (see Section 3.3 for more details):

```
sage: R.psi(1).evaluate()
1/24
```

Instead of working with the tautological ring of all of $\overline{\mathcal{M}}_{g,n}$, it is also possible to work on open subsets of the moduli space, such as the locus of compact type curves. This can be specified with the parameter `moduli`:

```
sage: Rct = TautologicalRing(3, 1, moduli='ct')
```

The available moduli types are:

- `'st'`: all stable curves (default).
- `'tl'`: treelike curves (all cycles in the stable graph have length 1).
- `'ct'`: compact type (stable graph is a tree).
- `'rt'`: rational tails (there exists a vertex of genus g).
- `'sm'`: smooth curves.

As an example of how this affects the behavior of the tautological ring, we can compute the so-called *socle degree*, i.e., the highest nonvanishing (complex) degree of the tautological ring of the corresponding subset of $\overline{\mathcal{M}}_{g,n}$.

```
sage: Rst = TautologicalRing(3, 1, moduli='st')
sage: Rst.socle_degree()
7
sage: Rsm = TautologicalRing(3, 1, moduli='sm')
sage: Rsm.socle_degree()
2
```

We will see in more detail in Section 3.4 how specifying the moduli affects computations.

3.2. ***Creating tautological classes.*** Each tautological class in admcycles has type `TautologicalClass`. We list in this section the different ways to enter tautological classes in the program. Depending on the example, some are more convenient than others.

As explained in Section 3.1 all computations happen in a given tautological ring (with a fixed base ring and fixed moduli). Once a tautological ring `R` for $\overline{\mathcal{M}}_{g,n}$ has been created as explained in Section 3.1,

the fundamental class, boundary divisors as well as ψ, κ and λ-classes are predefined methods of the ring `R`.

- `R.fundamental_class()` returns the fundamental class of $\overline{\mathcal{M}}_{g,n}$.
- `R.separable_boundary_divisor(h,A)` gives the pushforward $\xi_*[\overline{\mathcal{M}}_\Gamma]$ of the boundary gluing map
$$\xi : \overline{\mathcal{M}}_\Gamma = \overline{\mathcal{M}}_{h,A\cup\{p\}} \times \overline{\mathcal{M}}_{g-h,(\{1,\dots,n\}\setminus A)\cup\{p'\}} \to \overline{\mathcal{M}}_{g,n},$$
where `A` can be a list, set or tuple[2] of numbers from 1 to n.
- `R.irreducible_boundary_divisor()` gives the pushforward $(\xi')_*[\overline{\mathcal{M}}_{g-1,n+2}]$ of the boundary gluing map
$$\xi' : \overline{\mathcal{M}}_{g-1,n+2} \to \overline{\mathcal{M}}_{g,n}$$
identifying the last two markings to a node. Note that, since ξ' has degree 2 onto its image, this gives *twice* the fundamental class of the boundary divisor of irreducible nodal curves.
- `R.psi(i)` gives the ψ-class ψ_i of marking i on $\overline{\mathcal{M}}_{g,n}$.
- `R.kappa(a)` gives the (Arbarello–Cornalba) κ-class κ_a on $\overline{\mathcal{M}}_{g,n}$.
- `R.lambdaclass(d)` gives the class λ_d on $\overline{\mathcal{M}}_{g,n}$, defined as the d-th Chern class $\lambda_d = c_d(\mathbb{E})$ of the Hodge bundle $\mathbb{E}$, the vector bundle on $\overline{\mathcal{M}}_{g,n}$ with fiber $H^0(C, \omega_C)$ over the point $(C, p_1, \dots, p_n) \in \overline{\mathcal{M}}_{g,n}$.

These tautological classes can be combined in the usual way by operations `+`, `-`, `*` and raising to an integral power `^`.

```
sage: R1 = TautologicalRing(3, 4)
sage: t1 = 3*R1.separable_boundary_divisor(1,(1,2)) - R1.psi(4)^2
sage: R2 = TautologicalRing(2, 1)
sage: t2 = -1/3*R2.irreducible_boundary_divisor() * R2.lambdaclass(1)
```

For user convenience, alternative functions are available to create the basic tautological classes (over the rationals and for the full moduli of stable curves), without having to create the tautological ring before. Each of these functions require extra arguments `g` and `n` to specify the genus and the number of marked points.

- `fundclass(g, n)`
- `sepbdiv(g1, A, g, n)`
- `irrbdiv(g, n)`
- `psiclass(i, g, n)`
- `kappaclass(a, g, n)`
- `lambdaclass(d, g, n)`

[2] Be careful that tuples of length 1 must be entered as `(a,)` in Python, instead of `(a)`.

```
sage: tt1 = 3 * sepbdiv(1, (1,2), 3, 4) - psiclass(4, 3, 4)^2
sage: t1 == tt1
True
sage: tt2 = -1/3*irrbdiv(2, 1) * lambdaclass(1, 2, 1)
sage: t2 == tt2
True
```

To enter more complicated classes coming from decorated boundary strata, it is often convenient to first list all such decorated strata forming the generating set of $\mathrm{RH}^{2r}(\overline{\mathcal{M}}_{g,n})$ in a specified degree r using `R.list_generators(r)` and then select the desired ones from the list (see below for an explanation of the notation). As a shortcut one can also directly use the function `tautgens(g,n,r)` to produce this list without having to create the ring `R` before.

```
sage: R = TautologicalRing(2, 0)
sage: R.list_generators(2)
[0] : Graph :      [2] [[]] []
Polynomial : (kappa_2)_0
[1] : Graph :      [2] [[]] []
Polynomial : (kappa_1^2)_0
[2] : Graph :      [1, 1] [[2], [3]] [(2, 3)]
Polynomial : (kappa_1)_0
[3] : Graph :      [1, 1] [[2], [3]] [(2, 3)]
Polynomial : psi_2
[4] : Graph :      [1] [[2, 3]] [(2, 3)]
Polynomial : (kappa_1)_0
[5] : Graph :      [1] [[2, 3]] [(2, 3)]
Polynomial : psi_2
[6] : Graph :      [0, 1] [[3, 4, 5], [6]] [(3, 4), (5, 6)]
Polynomial : 1
[7] : Graph :      [0] [[3, 4, 5, 6]] [(3, 4), (5, 6)]
Polynomial : 1
```

The list itself is created by `R.generators(r)`, from which one can then select the classes:

```
sage: L = R.generators(2)
sage: t3 = 2*L[3]+L[4]
sage: t3
Graph :      [1] [[2, 3]] [(2, 3)]
Polynomial : (kappa_1)_0
Graph :      [1, 1] [[2], [3]] [(2, 3)]
Polynomial : 2*psi_2
```

The output above should be interpreted as follows: each `TautologicalClass` consists of a sum of decorated boundary strata (represented by data type `decstratum`), which consist of a graph (datatype `StableGraph`) and a polynomial in κ and ψ-classes (datatype `KappaPsiPolynomial`).

To explain the notation above, let us look at the example of generator `L[3]`.

```
Graph :      [1, 1] [[2], [3]] [(2, 3)]
Polynomial : 1*psi_2^1
```

Its stable graph is represented by three lists.

(1) The first list `[1, 1]` are the genera of the vertices, so there are two vertices, both of genus 1. Note that vertices are numbered by 0, 1, 2, . . . , so in the above case, the vertices are numbers 0 and 1.

(2) The second list gives the legs (that is markings or half-edges) attached to the vertices, so vertex 0 carries the half-edge 2 and vertex 1 the half-edge 3.

(3) The third list gives the edges, that is half-edge pairs that are connected; in the above case, the two half-edges 2 and 3 form an edge, connecting the two vertices.

If we wanted to enter this `StableGraph` manually, we could use its constructor as follows:

```
sage: G = StableGraph([1,1],[[2],[3]],[(2,3)]); G
[1, 1] [[2], [3]] [(2, 3)]
```

The polynomial in κ and ψ is `1*psi_2^1` in this case, so the half-edge 2 on the first vertex carries a ψ-class. For the generator `L[4]` the polynomial looks like `1*(kappa_1^1)_0`, meaning that vertex 0 carries a class $\kappa_1^1 = \kappa_1$.

Finally, it is possible to manually enter tautological classes by constructing a stable graph `gamma` and calling the main constructor `R(gamma,kappa,psi)` of the tautological ring.

```
sage: R = TautologicalRing(3,2)
sage: g = StableGraph([2,0], [[1,3],[2,4,5,6]], [(3,4),(5,6)])
sage: R(g, kappa=[[],[1]], psi={1:2})
Graph :      [2, 0] [[1, 3], [2, 4, 5, 6]] [(3, 4), (5, 6)]
Polynomial : (kappa_1)_1*psi_1^2
sage: R(g, kappa=[[1,1],[]])
Graph :      [2, 0] [[1, 3], [2, 4, 5, 6]] [(3, 4), (5, 6)]
Polynomial : (kappa_1*kappa_2)_0
```

In the above call, the arguments `kappa` and `psi` are both optionals and specify the κ and ψ decorations on the stable graph `gamma`. We refer to the documentation of admcycles for more details.

3.3. ***Basic operations.*** Apart from the usual arithmetic operations, we can take forgetful pushforwards and pullbacks of tautological classes and also compute the degree of tautological zero-cycles. In particular, we can compute intersection numbers. Below, for the forgetful map $\pi : \overline{\mathcal{M}}_{1,3} \to \overline{\mathcal{M}}_{1,2}$ forgetting the marking 3 we verify the relations

$$\pi_* \psi_3^2 = \kappa_1 \quad \text{and} \quad \pi^* \psi_2 = \psi_2 - D_{0,\{2,3\}},$$

where $D_{0,\{2,3\}}$ is the class of the boundary divisor in $\overline{\mathcal{M}}_{1,3}$ where generically the curve splits into two components of genera 0, 1 connected at a node with the component of genus 0 carrying markings 2, 3.

```
sage: s1 = TautologicalRing(1, 3).psi(3)^2
sage: s1.forgetful_pushforward([3])
Graph :      [1] [[1, 2]] []
Polynomial : (kappa_1)_0
sage: s2 = TautologicalRing(1, 2).psi(2)
sage: s2.forgetful_pullback([3])
Graph :      [1] [[1, 2, 3]] []
Polynomial : psi_2
Graph :      [1, 0] [[1, 4], [2, 3, 5]] [(4, 5)]
Polynomial : -1
```

Using the method `evaluate` of `TautologicalClass`, we also compute intersection numbers of ψ-classes on $\overline{\mathcal{M}}_{g,n}$, the so-called *correlators* or *descendent integrals*. Here, given numbers $k_1, \dots, k_n$ summing to $3g-3+n$ one can define these correlators $\langle \tau_{k_1} \cdots \tau_{k_n} \rangle_{g,n}$ as

$$\langle \tau_{k_1} \cdots \tau_{k_n} \rangle_{g,n} = \int_{\overline{\mathcal{M}}_{g,n}} \psi_1^{k_1} \cdots \psi_n^{k_n} . \tag{1}$$

Below we compute the intersection number

$$\langle \tau_0 \tau_1 \tau_2 \rangle_{1,3} = \int_{\overline{\mathcal{M}}_{1,3}} \psi_1^0 \psi_2 \psi_3^2 = \tfrac{1}{12}$$

and check that it agrees with the prediction $\langle \tau_0 \tau_1 \tau_2 \rangle_{1,3} = \langle \tau_0 \tau_2 \rangle_{1,2} + \langle \tau_1^2 \rangle_{1,2}$ by the string equation.

```
sage: R1 = TautologicalRing(1, 3)
sage: s3 = R1.psi(2) * R1.psi(3)^2
sage: s3.evaluate()
1/12
sage: R2 = TautologicalRing(1, 2)
sage: s4 = R2.psi(2)^2 + R2.psi(1) * R2.psi(2)
sage: s4.evaluate()
1/12
```

Instead of multiplying ψ-classes and evaluating by hand, we can also use the function `psi_correlator`, which takes as input the numbers $k_1, \dots, k_n$ and outputs the correlator (1).

```
sage: psi_correlator(0,1,2)
1/12
```

Note that in the current version of `admcycles`, the list of tautological generators $[\Gamma_i, \alpha_i]$ in a tautological class is *not* automatically simplified by combining equivalent terms (since in general this requires testing graph isomorphisms between the Γ_i). When performing arithmetic operations with complicated tautological classes, this simplification can be manually triggered using the function `simplify`, as demonstrated below. For this toy example, we create two different but isomorphic stable graphs, convert them to tautological classes and form their sum `s`. After applying the method `simplify` they are recognized as equal, so that we obtain a shorter sum.

```
sage: gamma1 = StableGraph([1,2],[[3],[4]],[(3,4)]).to_tautological_class()
sage: gamma2 = StableGraph([2,1],[[5],[6]],[(5,6)]).to_tautological_class()
sage: s = gamma1 + gamma2; s
Graph :      [1, 2] [[3], [4]] [(3, 4)]
Polynomial : 1
Graph :      [2, 1] [[5], [6]] [(5, 6)]
Polynomial : 1
sage: s_simple = s.simplify(); s_simple
Graph :      [1, 2] [[2], [3]] [(2, 3)]
Polynomial : 2
```

In a future version of `admcycles` (after improving our algorithms for graph isomorphisms), we plan to automate this process.

3.4. ***A basis of the tautological ring and tautological relations.*** One can compute, using the function `generating_indices(g,n,r)`, the indices (for the list `tautgens(g,n,r)`) of a basis of $\mathrm{RH}^{2r}(\overline{\mathcal{M}}_{g,n})$, assuming that the generalized Faber–Zagier relations (see [Pixton 2012; Pandharipande et al. 2015; Janda 2017]) between the additive generators $[\Gamma, \alpha]$ give a complete set of relations between them. For many concrete examples of (g, n, r), this conjecture can be checked using `admcycles` via the function `FZ_conjecture_holds(g,n,r)` (see [Molcho et al. 2021, Appendix B] and the documentation of the function for more details). For the computation we show below, let us verify that the generalized Faber–Zagier relations for $\mathrm{RH}^{2\cdot 2}(\overline{\mathcal{M}}_{2,0})$ are complete:

```
sage: FZ_conjecture_holds(2,0,2)
True
```

If the relations are complete as discussed above, `Tautvecttobasis` converts a vector with respect to the whole generating set into a vector in this basis. The function `TautologicalClass.basis_vector(r)` converts a `TautologicalClass` into such a vector.

Continuing the example from Section 3.2 we see:

```
sage: generating_indices(2,0,2)
[0, 1]
sage: t3.basis_vector(2)
(-48, 22)
```

This means that the generators `L[0]` and `L[1]` form a basis of $\mathrm{RH}^4(\overline{\mathcal{M}}_2)$ and the `TautologicalClass` `t3=2*L[3]+L[4]` is equivalent to `-48*L[0]+22*L[1]`.

It is also possible to directly verify tautological relations using the built-in function `is_zero` of `TautologicalClass`. It checks if the tautological class is contained in the ideal generated by the 3-spin relations [Pandharipande et al. 2015] (what we call the generalized Faber–Zagier relations above). Below we verify the known relation $\kappa = \psi - \delta_0 \in R^1(\overline{\mathcal{M}}_{1,n})$ for $n = 4$. Here ψ is the sum of all ψ_i and δ_0 is the sum of all separating boundary divisors, i.e., those having a genus 0 component. For this, we list all stable graphs with one edge via `list_strata(g,n,1)`. We exclude the graph `gamma` with a self-loop by requiring that the number of vertices `gamma.numvert()` is at least 2. Then we can convert these graphs `bd` to tautological classes by using `to_tautological_class`.

```
sage: R = TautologicalRing(1, 4)
sage: bgraphs = [bd for bd in list_strata(1,4,1) if bd.num_verts() > 1]
sage: del0 = sum(bd.to_tautological_class() for bd in bgraphs)
sage: psisum = sum(R.psi(i) for i in range(1,5))
sage: rel = R.kappa(1) - psisum + del0
sage: rel.is_zero()
True
```

As a shorthand for `is_zero` one can also simply compare to the integer `0` as follows:

```
sage: rel == 0
True
```

It is also possible to express tautological classes in a basis of the tautological ring of suitable open subsets of $\overline{\mathcal{M}}_{g,n}$, e.g., to verify that some relation holds on the locus of compact type curves. This works with

the optional argument `moduli` of `TautologicalRing` that was described in Section 3.1. We recall that moduli can be one of `'st'` (stable), `'tl'` (treelike), `'ct'` (compact type), `'rt'` (rational tails) or `'sm'` (smooth). The functions `basis_vector` and `is_zero` depend very much on the underlying moduli. For instance, we can verify the relation

$$\lambda_1 = \tfrac{B_2}{2}\kappa_1 = \tfrac{1}{12}\kappa_1 \in H^2(\mathcal{M}_g)$$

following from Mumford's computation [1983] in the case $g = 3$:

```
sage: R = TautologicalRing(3, 0, moduli='sm')
sage: R.kappa(1).basis_vector()
(1)
sage: R.lambdaclass(1).basis_vector()
(1/12)
```

It is also possible to start with a class on a bigger moduli (e.g., the default locus `'st'` of all stable curves) and check whether it vanishes on a smaller subset using the optional parameter `moduli` of the functions `is_zero` or `basis_vector`:

```
sage: R = TautologicalRing(2, 0)
sage: u = R.lambdaclass(2)
sage: u.is_zero()
False
sage: u.is_zero(moduli='ct')
True
sage: u.basis_vector()
(-3/2, 1/2)
sage: u.basis_vector(moduli='ct')
()
```

The vanishing here was expected as on $\mathcal{M}^{\mathrm{ct}}_{2,0}$ the tautological ring in degree 2 vanishes:

```
sage: R = TautologicalRing(2, 0, moduli='ct')
sage: R.socle_degree()
1
```

In practice, much of the time in some computations is spent on calculating generalized Faber–Zagier relations between tautological cycles on $\overline{\mathcal{M}}_{g,n}$. However, once computed, the relations can be saved to a file and reloaded in a later session using the functions `save_FZrels()` and `load_FZrels()`. Careful: the function `save_FZrels()` creates (and overwrites previous version of) a file `new_geninddb.pkl` which, depending on the previous computations, can be quite large.

3.5. ***Pulling back tautological classes to the boundary.*** Recall that for a stable graph Γ we have a gluing map

$$\xi_\Gamma : \overline{\mathcal{M}}_\Gamma = \prod_{i=1}^{m} \overline{\mathcal{M}}_{g(v_i),n(v_i)} \to \overline{\mathcal{M}}_{g,n} \tag{2}$$

taking one stable curve for each of the vertices $v_1, \ldots, v_m$ of Γ and gluing them together according to the edges of Γ. By [Graber and Pandharipande 2003, Appendix A], the pullback of a tautological class

under ξ_Γ is contained in the tensor product of the tautological rings of the factors $\overline{\mathcal{M}}_{g(v_i),n(v_i)}$ above, and this operation is implemented in `admcycles`.

Below we pull back a generator of $\mathrm{RH}^4(\overline{\mathcal{M}}_4)$ to the boundary divisor with genus partition $4 = 2+2$. This produces an element of type `prodtautclass`, a tautological class on a product of moduli spaces, in this case $\overline{\mathcal{M}}_{2,1} \times \overline{\mathcal{M}}_{2,1}$. Two elements on the same product of spaces can be added and multiplied and further operations like pushforwards under (partial) gluing maps are supported. More details are given in the documentation of the class `prodtautclass`.

Below, we want to express the pullback to $\overline{\mathcal{M}}_{2,1} \times \overline{\mathcal{M}}_{2,1}$ in terms of a basis of $H^2(\overline{\mathcal{M}}_{2,1} \times \overline{\mathcal{M}}_{2,1})$ obtained from the preferred bases of the factors $H^*(\overline{\mathcal{M}}_{2,1})$ given by `generating_indices`. We can either represent the result as a list of matrices (giving the coefficients in the tensor product bases) or as a combined vector (using the option `vecout=true`).

```
sage: bdry=StableGraph([2,2],[[1],[2]],[(1,2)])

sage: generator=tautgens(4,0,2)[3]; generator
Graph :      [1, 3] [[2], [3]] [(2, 3)]
Polynomial : psi_3

sage: pullback=bdry.boundary_pullback(generator)

sage: pullback.totensorTautbasis(2)
[
                            [-3]
                            [ 1]
                  [0 0 0]   [-3]
                  [0 0 0]   [ 7]
[-3  1 -3  7  1], [0 0 0], [ 1]
]

sage: pullback.totensorTautbasis(2,vecout=true)
(-3, 1, -3, 7, 1, 0, 0, 0, 0, 0, 0, 0, 0, -3, 1, -3, 7, 1)
```

3.6. ***Pushing forward classes from the boundary.*** The pushforward under the map ξ_Γ in (2) sends a product of tautological classes on the factors $\overline{\mathcal{M}}_{g(v_i),n(v_i)}$ to a tautological class of $\overline{\mathcal{M}}_{g,n}$. This operation is implemented by the function `boundary_pushforward` of `StableGraph`.

That is to say, if `Gamma` is a `StableGraph` and if `[c1, ..., cm]` is a list whose i-th element `ci` is a `TautologicalClass` on the i-th factor $\overline{\mathcal{M}}_{g(v_i),n(v_i)}$ of $\overline{\mathcal{M}}_\Gamma$, then

```
Gamma.boundary_pushforward([c1, ..., cm])
```

is the pushforward of the product of the `ci`. Here, the markings for the class `ci` are supposed to go from 1 to $n(v_i)$, where the j-th marking corresponds to leg number j on the i-th vertex of `Gamma`.

As an illustration, we verify that the package correctly computes the excess intersection formula proved in [Graber and Pandharipande 2003] for the self-intersection of a boundary divisor in $\overline{\mathcal{M}}_{3,3}$.

```
sage: B=StableGraph([2,1],[[4,1,2],[3,5]],[(4,5)])

sage: Bclass = B.boundary_pushforward() # class of undecorated boundary divisor

sage: si1 = B.boundary_pushforward([fundclass(2,3),-psiclass(2,1,2)])

sage: si1
Graph :      [2, 1] [[4, 1, 2], [3, 5]] [(4, 5)]
Polynomial : -psi_5

sage: si2 = B.boundary_pushforward([-psiclass(1,2,3),fundclass(1,2)])
```

```
sage: si2
Graph :      [2, 1] [[4, 1, 2], [3, 5]] [(4, 5)]
Polynomial : -psi_4

sage: (Bclass*Bclass-si1-si2).is_zero()
True
```

Note that, e.g., for the term `si2` we needed to hand the function the term `-psiclass(1,2,3)` in the first vertex, since in the graph `B` the half-edge `4` is leg number `1` in the list of legs at the first vertex (and we have $(g(v_1), n(v_1)) = (2, 3)$ for this vertex).

4. SPECIAL CYCLE CLASSES. Beyond the already mentioned standard tautological classes ψ_i, κ_a and λ_d and boundaries from Section 3.2, `admcycles` provides more advanced constructions that we describe now. The corresponding functions are summarized here:

`TautologicalRing` method	standalone function	manual section
`double_ramification_cycle`	`DR_cycle`	Section 4.1
`theta_class`	`ThetaClass`	Section 4.1
`differential_stratum`	`Strataclass`	Section 4.2
`generalized_lambda`	`generalized_lambda`	Section 4.3
`hyperelliptic_cycle`	`Hyperell`	Section 4.4
`bielliptic_cycle`	`Biell`	Section 4.4

A convenient way to find out about tautological class constructions is to use the *tab completion* feature of SageMath. When you enter a part of a name and press the tab key (denoted `<TAB>` below) the program will show you all available completions. It can be used to discover the names in the admcycles module.

```
sage: import admcycles

sage: admcycles.<TAB>
admcycles.Biell         admcycles.DR_phi               ...
admcycles.DR            admcycles.DRpoly               ...
admcycles.DR_cycle      admcycles.FZ_conjecture_holds  ...
admcycles.DR_cycle_old  admcycles.GRRcomp              ...
```

Similarly one can discover the methods of `TautologicalRing` starting with the letter `d`:

```
sage: R = TautologicalRing(2, 2)

sage: R.d<TAB>
R.differential_stratum      R.dump
R.dimension                 R.dumps
R.double_ramification_cycle
```

4.1. ***Double ramification cycles.*** A particularly interesting family of cycles on $\overline{\mathcal{M}}_{g,n}$ is given by the double ramification cycles. Fixing g, n they are indexed by nonnegative integers $k, d \geq 0$ and a tuple $A = (a_1, a_2, \ldots, a_n)$ of integers summing to $k(2g-2+n)$.

The classical double ramification cycle (for $k = 0, d = g$)

$$\mathrm{DR}_g(A) \in H^{2g}(\overline{\mathcal{M}}_{g,n})$$

has been defined as the pushforward of the virtual fundamental class of a space of maps to rubber $\mathbb{P}^1$

relative to $0, \infty$ with tangency conditions at $0, \infty$ specified by the vector A (see [Li and Ruan 2001; Li 2002; 2001; Graber and Vakil 2005]). In [Janda et al. 2017] it is shown that this cycle is tautological and an explicit formula in terms of tautological generators is provided.

More precisely, for g, n, k, d and A with A a partition of $k(2g-2+n)$, the paper constructs an explicit tautological class

$$P_g^{d,r,k}(A) \in \mathbb{Q}[r] \otimes_{\mathbb{Q}} \mathrm{RH}^{2d}(\overline{\mathcal{M}}_{g,n})$$

with coefficients being polynomials in a formal variable r. We obtain a usual tautological class $P_g^{d,k}(A) \in \mathrm{RH}^{2d}(\overline{\mathcal{M}}_{g,n})$ by setting $r = 0$ in these polynomial coefficients. Then it is shown ([Janda et al. 2017, Theorem 1]) that in the special case $k = 0,\ d = g$, this gives a formula for the double ramification cycle

$$\mathrm{DR}_g(A) = 2^{-g} P_g^{g,k}(A).$$

While this demonstrates that the cycle $P_g^{d,k}(A)$ is useful for $k = 0, d = g$, it has many interesting properties for other values of k, d:

- For k arbitrary and $d = 1$, the restriction of $2^{-1} P_g^{1,k}(A)$ to the compact-type locus $\mathcal{M}_{g,n}^{\mathrm{ct}}$ gives the pullback of the theta divisor on the universal Jacobian $\mathcal{J}$ over $\mathcal{M}_{g,n}^{\mathrm{ct}}$ under the extension of the Abel–Jacobi section

$$\begin{aligned} \mathcal{M}_{g,n} &\to \mathcal{J}, \\ (C, p_1, \ldots, p_n) &\mapsto (\omega_C^{\log})^{\otimes k}\left(-\sum_{i=1}^{n} a_i p_i\right); \end{aligned}$$

  see [Hain 2013; Grushevsky and Zakharov 2014].

- For k arbitrary and $d = g$, various geometric definitions of a double ramification cycle have been put forward and an equality with $2^{-g} P_g^{g,k}(A)$ was conjectured in [Farkas and Pandharipande 2018; Schmitt 2018] (see [Holmes and Schmitt 2019, Section 1.6] for an overview of the various definitions). Recently, this conjecture was proven in [Bae et al. 2020] based on earlier results of [Holmes and Schmitt 2019].

- For k arbitrary and $d > g$, the class $P_g^{d,k}(A)$ vanishes by [Clader and Janda 2018].

In `admcycles`, the formula for $P_g^{d,k}(A)$ has been implemented. The function `DR_cycle(g,A,d,k)` returns the cycle $2^{-d} P_g^{d,k}(A)$. The factor 2^{-d} was chosen such that `DR_cycle(g,A)` indeed gives the cycle $\mathrm{DR}_g(A)$. With the option `rpoly=True`, it is even possible to compute the cycle $2^{-d} P_g^{d,r,k}(A)$ whose coefficients are polynomials in the variable r.

As an application, we can verify the result from [Holmes et al. 2019] that DR cycles satisfy the multiplicativity property

$$\mathrm{DR}_g(A) \cdot \mathrm{DR}_g(B) = \mathrm{DR}_g(A) \cdot \mathrm{DR}_g(A+B) \in H^{4g}(\mathcal{M}_{g,n}^{tl})$$

on the locus $\mathcal{M}_{g,n}^{tl}$ of treelike curves but *not* on the locus of all stable curves, in the example given in [Holmes et al. 2019, Section 8].

```
sage: A=vector((2,4,-6)); B=vector((-3,-1,4))
sage: diff = DR_cycle(1,A)*DR_cycle(1,B)-DR_cycle(1,A)*DR_cycle(1,A+B)
sage: diff.is_zero(moduli='tl')
True
sage: diff.is_zero(moduli='st')
False
```

In fact, using that the cycle $\mathrm{DR}_g(A)$ is polynomial in the entries of the vector A (i.e., a tautological class with polynomial coefficients), we can check multiplicativity for all vectors A, B in the case $g = 1, n = 3$. To gain access to the polynomial-valued DR cycle, we define a polynomial ring and call `DR_cycle` with a vector A having as coefficients the generators of this ring:

```
sage: R.<a1,a2,a3,b1,b2,b3> = PolynomialRing(QQ,6)
sage: A = vector((a1,a2,a3)); B = vector((b1,b2,b3))
sage: diff = DR_cycle(1,A)*DR_cycle(1,B)-DR_cycle(1,A)*DR_cycle(1,A+B)
sage: diff.is_zero(moduli='tl')
True
```

As a second application, we can verify the formula from [Buryak and Rossi 2021, Theorem 2.1] for intersection numbers of two DR cycles with λ_g on $\overline{\mathcal{M}}_{g,3}$ in the case $g = 1$:

```
sage: intersect = DR_cycle(1,A)*DR_cycle(1,B)*lambdaclass(1,1,3)
sage: f = intersect.evaluate(); factor(f)
(1/216) * (a2*b1 - a3*b1 - a1*b2 + a3*b2 + a1*b3 - a2*b3)^2
sage: g = f.subs({a3:-a1-a2,b3:-b1-b2}); factor(g)
(1/24) * (a2*b1 - a1*b2)^2
```

The formula of the cycle $P_g^{d,r,k}(A)$ in [Janda et al. 2017] is obtained as a simplification (modulo r) of a cycle

$$r^{2d-2g+1}\epsilon_* c_d(-R^*\pi_*\mathcal{L}) \tag{3}$$

appearing in [Janda et al. 2017, Corollary 4, Proposition 5] (see there for the notation). The cycle (3) is often called a *Chiodo class* and it is relevant for certain computations (see [Borot et al. 2020; Do and Lewański 2020]). Since the latest version of `admcycles`, the cycle (3) can be obtained using the optional parameters `chiodo_coeff = True` and `r_coeff` of `DR_cycle`, which evaluates the expression (3) at the value `r_coeff` of r.

```
sage: g=2; A=(5,-1); d=2; k=1
sage: Chiodo = DR_cycle(g,A,d,k,chiodo_coeff=True,r_coeff=7)
```

As a special case of this formula, we can obtain the cycle class $\theta_{g,n} \in R^*(\overline{\mathcal{M}}_{g,n})$ described in [Norbury 2017], which is accessible via the function `ThetaClass`.

```
sage: T = ThetaClass(1,1)
sage: T == 3*psiclass(1,1,1)
True
```

codim	$\boldsymbol{m}=0$	$\boldsymbol{m}=k\cdot\boldsymbol{m}'$ for $\boldsymbol{m}'\in\mathbb{Z}^n_{\geq 0}$	$\boldsymbol{m}\neq k\cdot\boldsymbol{m}'$ for $\boldsymbol{m}'\in\mathbb{Z}^n_{\geq 0}$
$k=0$	0	0	g
$k=1$	0	$g-1$	g
$k>1$	0	$g-1$ and g	g

Table 1. Dimension theory of $\overline{\mathcal{H}}^k_g(\boldsymbol{m})$. Note that for $k>1$ and $\boldsymbol{m}=k\cdot\boldsymbol{m}'$ with $\boldsymbol{m}'\in\mathbb{Z}^n_{\geq 0}$, the set $\overline{\mathcal{H}}^1_g(\boldsymbol{m}')\subset\overline{\mathcal{H}}^k_g(\boldsymbol{m})$ is a union of components of codimension $g-1$ in $\overline{\mathcal{M}}_{g,n}$, with all other components of $\overline{\mathcal{H}}^k_g(\boldsymbol{m})$ having pure codimension g.

4.2. ***Strata of k-differentials.*** Let $g,n,k\geq 0$ with $2g-2+n>0$ and let $\boldsymbol{m}=(m_1,\dots,m_n)\in\mathbb{Z}^n$ with $\sum_i m_i=k(2g-2)$. Consider the subset

$$\mathcal{H}^k_g(\boldsymbol{m})=\left\{(C,p_1,\dots,p_n)\in\mathcal{M}_{g,n}:\omega_C^{\otimes k}\left(\sum_{i=1}^n m_i p_i\right)\cong\mathcal{O}_C\right\}\subset\mathcal{M}_{g,n}.$$

Denote by $\overline{\mathcal{H}}^k_g(\boldsymbol{m})$ the closure of $\mathcal{H}^k_g(\boldsymbol{m})$ inside $\overline{\mathcal{M}}_{g,n}$. Since the above equality of line bundles is equivalent to the existence of a meromorphic k-differential η on C with zeros and poles exactly at the points p_i with multiplicities m_i, the subsets $\overline{\mathcal{H}}^k_g(\boldsymbol{m})$ are called *strata of k-differentials*.

These strata are of interest in algebraic geometry, the theory of flat surfaces and Teichmüller dynamics and have been studied intensely in the past. Elements appearing in the boundary have been classified in [Bainbridge et al. 2018; 2019b] and a smooth, modular compactification has been constructed in [Bainbridge et al. 2019a]. The dimension of $\overline{\mathcal{H}}^k_g(\boldsymbol{m})$ depends on $k,\boldsymbol{m}$ as in Table 1 (see, e.g., [Farkas and Pandharipande 2018; Schmitt 2018]).

For $k\geq 1$, [Farkas and Pandharipande 2018; Schmitt 2018] present conjectural relations between the fundamental classes $[\overline{\mathcal{H}}^k_g(\boldsymbol{m})]$ and the formulas for the double ramification cycles proposed by Pixton (see Section 4.1). The conjectures were recently proven in [Bae et al. 2020] based on results from [Holmes and Schmitt 2019]. As explained in the papers, these conjectures can be used to recursively determine all cycles

- $[\overline{\mathcal{H}}^k_g(\boldsymbol{m})]\in\mathrm{RH}^{2g}(\overline{\mathcal{M}}_{g,n})$ for $k\geq 1$ and $\boldsymbol{m}\neq k\boldsymbol{m}'$ for some $\boldsymbol{m}'\in\mathbb{Z}^n_{\geq 0}$,
- $[\overline{\mathcal{H}}^1_g(\boldsymbol{m})]\in\mathrm{RH}^{2g-2}(\overline{\mathcal{M}}_{g,n})$ for $k=1$ and $\boldsymbol{m}\in\mathbb{Z}^n_{\geq 0}$.

These recursive algorithms have been implemented in the function `Strataclass(g,k,m)`, where as above `m` is a tuple of n integers summing to $k(2g-2)$.

As a small application, we can check that the stratum class $[\overline{\mathcal{H}}^1_2((3,-1))]$ vanishes (the stratum is empty since by the residue theorem there can be no meromorphic differential with a single, simple pole). Also, the stratum $\overline{\mathcal{H}}^1_2((2))$ exactly equals the class of the locus of genus 2 curves with a marked Weierstrass point, which can be computed by the function `Hyperell` (see below for details).

```
sage: L=Strataclass(2,1,(3,-1)); L.is_zero()
True

sage: L=Strataclass(2,1,(2,)); (L-Hyperell(2,1)).is_zero()
True
```

4.3. ***Generalized lambda classes.*** Let $\pi : \mathcal{C}_{g,n} \to \overline{\mathcal{M}}_{g,n}$ be the universal curve and assume $n \geq 1$. Every divisor of $\mathcal{C}_{g,n}$, up to pullback of divisors on $\overline{\mathcal{M}}_{g,n}$, takes the form

$$D = l\tilde{K} + \sum_{p=1}^{n} d_p \sigma_p + \sum_{\substack{h \leq g, \\ 1 \in S \subset [n]}} a_{h,S} C_{h,S}$$

for some integers l, d_p, $a_{h,S}$. Here $\tilde{K} = c_1(\omega_\pi)$ is the first Chern class of the relative dualizing sheaf, σ_p is the class of the p-th section and

$$C_{h,S} = \xi_*[\overline{\mathcal{M}}_{h,S\cup\{\bullet\}} \times \overline{\mathcal{M}}_{g-h,[n]\setminus S\cup\{\star,x\}}] \in \mathrm{CH}^1(\overline{\mathcal{M}}_{g,[n]\cup\{x\}}) = \mathrm{CH}^1(\mathcal{C}_{g,n}).$$

The fact that every divisor D on $\mathcal{C}_{g,n}$ can be written in this form up to pullbacks from $\overline{\mathcal{M}}_{g,n}$ follows from the identification $\mathcal{C}_{g,n} \cong \overline{\mathcal{M}}_{g,n+1}$ and the computation of the Picard group of the moduli spaces of stable curves due to Harer [1983] and Arbarello and Cornalba [1987]. In [Pagani et al. 2020] a formula is given for the Chern character $\mathrm{ch}(R^\bullet\pi_*\mathcal{O}(D))$. This Chern character can be computed up to degree `dmax` using `generalized_chern_hodge(l,d,a,dmax,g,n)`. It takes as input an integer `l`, a list `d=[d1,...,dn]` of the integers d_i and a list of triples `a=[[h1,S1,ahS1],...,[hn,Sn,ahSn]]` where the `ahSi` are the integers $a_{h,S}$ above (given in any order). It is enough to just include the triples `[h,S,ahS]` for which $a_{h,S}$ is nonzero.

Using `generalized_lambda(i,l,d,a,g,n)` the Chern class $c_i(-R^\bullet\pi_*\mathcal{O}(D))$ can be computed directly. In particular when $l = 1$ and the d_p and $a_{h,S}$ are zero, this equals the normal λ class.

```
sage: g=3;n=1
sage: l=1;d=[0];a=[]
sage: s=lambdaclass(2,g,n)
sage: t=generalized_lambda(2,l,d,a,g,n)
sage: (s-t).is_zero()
True
```

Let $d_1, ..., d_n$ be integers such that $\sum_{i=1}^n d_i$ is divisible by $2g-2$ and let $\phi \in V^0_{g,n}$ be an element of the stability space $V^0_{g,n}$ defined in [Kass and Pagani 2019, Definition 3.2]. This ϕ is an assignment which given a stable curve $(C, p_1, \ldots, p_n) \in \overline{\mathcal{M}}_{g,n}$ associates a real number $\phi(C, p_1, \ldots, p_n)_{C'}$ to every irreducible component C' of C. These numbers must sum to zero as C' runs through the components of C; they only depend on the stable graph of C and must be compatible with degenerations of curves. Given this data, Kass and Pagani construct a compactification $\overline{\mathcal{J}}_{g,n}(\phi)$ of the universal Jacobian over $\overline{\mathcal{M}}_{g,n}$.

Let now $l = \sum_{i=1}^n d_i/(2g-2)$ and let $a_{h,S}$ be integers such that

$$D(\phi) = l\tilde{K} + \sum d_i\sigma_i + \sum a_{h,S}(\phi) C_{h,S}$$

is ϕ-stable on the locus of stable curves with one node (for definitions, see [Kass and Pagani 2019] or [Pagani et al. 2020]). For the shifted[3] vector $A = (d_1 + l, \ldots, d_n + l)$, [Holmes et al. 2018] proves

[3]This shift is due to the fact that the literature on double ramification cycles uses the "log-convention", i.e., the entries of the input sum to $l(2g-2+n)$.

an equality

$$\mathrm{DR}_g(A)|_{U(\phi)} = c_g(-R^\bullet\pi_*\mathcal{O}(D(\phi)))|_{U(\phi)} \tag{4}$$

on the largest open locus $U(\phi) \subset \overline{\mathcal{M}}_{g,n}$ where the Abel–Jacobi section

$$s_{l,d}(\phi) : \overline{\mathcal{M}}_{g,n} \dashrightarrow \overline{\mathcal{J}}_{g,n}(\phi),\ (C, p_1, \ldots, p_n) \mapsto \omega_C^{\otimes l}\left(-\sum_{i=1}^{n} d_i p_i\right)$$

extends to a morphism. In particular, $U(\phi)$ always includes $\mathcal{M}^{\mathrm{ct}}_{g,n}$ and equals $\overline{\mathcal{M}}_{g,n}$ if and only if l, d are trivial or $l(2g-2) = 0$ and $d = [0, ..., \pm 1, ..., \mp 1, ..., 0]$. See [Pagani et al. 2020, Section 4.3] for more details.

The function `DR_phi(g,d)` computes $c_g(-R^\bullet\pi_*\mathcal{O}(D(\phi)))$. We can verify equality (4).

```
sage: g=2;d=[1,-1]

sage: (DR_cycle(g,d)-DR_phi(g,d)).is_zero()
True
```

We also see that equality does not always hold over all of $\overline{\mathcal{M}}_{g,n}$ but it does hold over $\mathcal{M}^{\mathrm{ct}}_{g,n}$.

```
sage: g=2;d=[2,-2]

sage: (DR_cycle(g,d)-DR_phi(g,d)).basis_vector()
(12, -4, 14, 7, -40, -10, -14, -12, 28, -4, 6, -1, 4, 0)

sage: (DR_cycle(g,d)-DR_phi(g,d)).basis_vector(moduli='ct')
(0, 0, 0, 0, 0)
```

4.4. *Admissible cover cycles.*

Hyperelliptic and bielliptic cycles. Before we go into details of how to specify general admissible cover cycles, let us mention the important cases of hyperelliptic and bielliptic cycles.

Recall that a smooth curve C is called *hyperelliptic* if C admits a double cover $C \to \mathbb{P}^1$ and is called *bielliptic* if it admits a double cover $C \to E$ of *some* smooth genus 1 curve E. In both cases we have an involution $C \to C$ that exchanges the two sheets of the cover. Given $g, n, m \geq 0$ with $n \leq 2g+2$ and $2g-2+n+2m > 0$, we have the locus $\overline{H}_{g,n,2m} \subset \overline{\mathcal{M}}_{g,n+2m}$ which is the closure of the locus of smooth curves $(C, p_1, \ldots, p_n, q_1, q_1', \ldots, q_m, q_m')$ such that C is hyperelliptic with $p_1, \ldots, p_n$ fixed points of the hyperelliptic involution and the pairs q_i, q_i' being exchanged by this involution. An analogous definition gives the locus $\overline{B}_{g,n,2m} \subset \overline{\mathcal{M}}_{g,n+2m}$ as the closure of the set of bielliptic curves with $n \leq 2g-2$ fixed points and m pairs of points forming orbits under the bielliptic involution.

Then the fundamental class of the (reduced) loci $\overline{H}_{g,n,2m}$ and $\overline{B}_{g,n,2m}$ can (in many cases) be computed by the functions `Hyperell(g,n,m)` and `Biell(g,n,m)` of our program.

As an example, we compute the class $[\overline{H}_3] \in \mathrm{RH}^2(\overline{\mathcal{M}}_3)$ and verify that we obtain the known result

$$[\overline{H}_3] = 9\lambda - \delta_0 - 3\delta_1,$$

where δ_0 is the class of the divisor of irreducible nodal curves and δ_1 is the divisor of curves with a separating node between a genus 1 and a genus 2 component.

```
sage: H = Hyperell(3,0,0)
sage: H.basis_vector()
(3/4, -9/4, -1/8)
sage: R = TautologicalRing(3, 0)
sage: H2 = 9*R.lambdaclass(1)-(1/2)*R.irreducible_boundary_divisor()
          -3*R.separable_boundary_divisor(1,())
sage: H2.basis_vector()
(3/4, -9/4, -1/8)
```

Here we need to divide `irrbdiv()` by two, the degree of the corresponding gluing map.

Creating and identifying general admissible cover cycles. Generalizing the case of hyperelliptic and bielliptic cycles, we can consider loci of curves C admitting a cover $C \to D$ to a second curve D such that the cover is Galois with respect to a fixed finite group G. Cycles defined via such covers were studied in [Schmitt and van Zelm 2020]. In general, such an admissible cover cycle is specified by the genus g of the curve C, the finite group G, and monodromy data (we refer the reader to [Schmitt and van Zelm 2020, Section 1.3] for the precise definitions). Currently, intersections are only implemented for cyclic groups. Below we will study bielliptic curves in genus 2, which are double covers of elliptic curves branched over two points. As a first step we enter the monodromy data.

```
sage: G=PermutationGroup([(1,2)])
sage: list(G)
[(), (1,2)]
sage: H=HurData(G,[G[1],G[1]])
```

The function `HurData` takes the group G as the first argument and as the second a list of group elements $\alpha \in G$, each of which corresponds to the G-orbit of some marking $p \in C$. Here α is a generator of the stabilizer of p under the group action $G \curvearrowright C$, which gives the monodromy around p. In other words, the natural action of the stabilizer $G_p = \langle \alpha \rangle$ on a tangent vector $v \in T_pC$ is given by

$$\alpha.v = \exp(2\pi i/\mathrm{ord}(h))v.$$

Thus in the example above, we have two markings, both with stabilizer generated by `G[1]=(1,2)` which acts by multiplication by -1 on the tangent space.

To identify the admissible cover cycle (inside the moduli space $\overline{\mathcal{M}}_{g,n}$ with n the total number of marked points from the monodromy data) in terms of tautological classes, one can use the function `Hidentify`. It pulls back the admissible cover cycle to all boundary divisors and (recursively) identifies the pullback itself in terms of tautological classes. It compares this pullback to the pullback of a basis of the tautological ring. Often this pullback map is injective in cohomology so that one can then write the admissible cover cycle in terms of the basis using linear algebra. Sometimes, it is necessary to additionally intersect with some monomials in κ and ψ-classes.

To apply `Hidentify` one gives the genus and the monodromy data as arguments. The standard output format is an instance of the class `TautologicalClass`. For users familiar with Pixton's implementation of the tautological ring, there is the option `vecout=true` which returns instead a vector with respect to

the generating set of the tautological ring provided by this program.

```
sage: vbeta = Hidentify(2, H, vecout=true)
sage: vector(vbeta)
(517/4, -33, 11/4, 243/4, -125/4, 15/2, 41/4, 125, 99/4, -41, -1137/4, -285/4, 0, 0, 0, 0, 0,
 0, -57/8, -3/8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
```

This output means, specifically, that inside $\overline{\mathcal{M}}_{2,2}$ the locus of bielliptic curves with the two points fixed by the involution being the marked points is given (in the generating set `gens=all_strata(2,3,(1,2))` produced by Pixton's program) as $517/4 \cdot$ `gens[0]` $- 33 \cdot$ `gens[1]` $+ \ldots$.

If we instead wanted to have bielliptic curves with two marked fixed-points of the involution and one pair of markings that are exchanged by the involution, we would need to use the monodromy data

```
sage: H2=HurData(G,[G[1],G[1],G[0]])
```

in which case `Hidentify(2,H2)` would live inside $\mathrm{RH}^8(\overline{\mathcal{M}}_{2,4})$.

If we only want to remember a subset of the markings, we can use the optional parameter `marking` to give this subset. For instance, the command `Hidentify(2,H,markings=[])` would give the pushforward of `Hidentify(2,H)` in $\overline{\mathcal{M}}_{2,2}$ to the space $\overline{\mathcal{M}}_2$ under the forgetful morphism (see also Section 4.4).

Example: specifying and identifying $[\bar{B}_2]$ *by hand.* The locus $\bar{B}_2 \subset \overline{\mathcal{M}}_2$ of bielliptic curves is a divisor. A bielliptic genus 2 curve is ramified over two points. In the following we use the methods of the previous section to identify its cycle class.

Now when treating admissible cover cycles in general, our program a priori handles the cycle where all possible ramification points are marked. In this case, this is the cycle $[\bar{B}_{2,2,0}] \in \mathrm{RH}^6(\overline{\mathcal{M}}_{2,2})$ of bielliptic curves C with the two ramification points p_1, p_2 marked. By specifying `markings=[]` when calling `Hidentify`, we tell it to remember none of the markings, in other words to push forward under the map $\pi : \overline{\mathcal{M}}_{2,2} \to \overline{\mathcal{M}}_2$ forgetting the markings.

```
sage: G = PermutationGroup([(1,2)])
sage: H = HurData(G, [G[1],G[1]])
sage: B22 = Hidentify(2, H, markings=[])
sage: B22.basis_vector(1)
(30, -9)
```

We compare the result with the known formulas for $[\bar{B}_2]$. For δ_0 the class of the irreducible boundary of $\overline{\mathcal{M}}_2$ and δ_1 the class of the boundary divisor with genus-splitting $(1, 1)$, it is known that $[\bar{B}_2] = \frac{3}{2}\delta_0 + 6\delta_1$ (see [Faber 1996]). If we want to enter this combination of δ_0 and δ_1, we have to be careful about conventions, though: the corresponding gluing maps $\xi : \overline{\mathcal{M}}_{1,2} \to \overline{\mathcal{M}}_2$ and $\xi' : \overline{\mathcal{M}}_{1,1} \times \overline{\mathcal{M}}_{1,1} \to \overline{\mathcal{M}}_2$ both have degree 2. This corresponds to the fact that the associated stable graphs both have an automorphism group of order 2. Hence we have to divide by a factor of two and obtain

```
sage: B22_formula = 3/4*irrbdiv(2,0) + 3*sepbdiv(1,(),2,0)
sage: B22_formula.basis_vector(1)
(15/2, -9/4)
```

We see that up to a factor of 4 the two vectors `(30, -9)` and `(15/2, -9/4)` agree. Where does this factor come from?

For this, recall that the cycle `B22` above is equal to $\pi_*[\overline{B}_{2,2,0}]$. Since for the generic bielliptic curve C there are two choices of orderings for marking p_1, p_2, this explains a factor of 2. On the other hand, the hyperelliptic involution $\sigma : C \to C$ on C exchanges p_1 and p_2. Thus $\sigma \in \mathrm{Aut}(C)$, but $\sigma \notin \mathrm{Aut}(C, p_1, p_2)$. This missing automorphism factor explains another factor of 2 in the pushforward under π, so in fact $[\overline{B}_2] = \frac{1}{4}\pi_*[\overline{B}_{2,2,0}]$.

Note that since the cycles of bielliptic loci are implemented via the function `Biell`, we could have taken a shortcut above.

```
sage: B = Biell(2,0,0)

sage: B.basis_vector()
(15/2, -9/4)
```

As an application, we can check the Hurwitz–Hodge integral

$$\int_{[\overline{B}_{2,2,0}]} \lambda_2\lambda_0 = \int_{\pi_*[\overline{B}_{2,2,0}]} \lambda_2 = \frac{1}{48}$$

predicted by [Pandharipande and Tseng 2019].

```
sage: (B22 * lambdaclass(2,2,0)).evaluate()
1/48
```

The corresponding integrals for $g = 3, 4$ have also been verified like this, but the amount of time and memory needed grows drastically.

We can also check the Hurwitz–Hodge integral

$$\int_{[\overline{\mathcal{H}}_{2,\mathbb{Z}/3\mathbb{Z},((1,2,3)^2,(1,3,2)^2)}]} \lambda_1 = \frac{2}{9}$$

of λ_1 against the locus of genus 2 curves admitting a cyclic triple cover of a genus 0 curve with two points of ramification $(1, 2, 3) \in \mathbb{Z}/3\mathbb{Z}$ and two points of ramification $(1, 3, 2) \in \mathbb{Z}/3\mathbb{Z}$, computed in [Owens and Somerstep 2019, Section 5].

```
sage: G = PermutationGroup([(1,2,3)]); sorted(list(G))
[(), (1,2,3), (1,3,2)]

sage: H = HurData(G,[G[1],G[1],G[2],G[2]]) #n=2, m=2

sage: t = Hidentify(2,H,markings=[])

sage: (t*lambdaclass(1,2,0)).evaluate()
2/9
```

Note that while originally the cycle $[\overline{\mathcal{H}}_{2,\mathbb{Z}/3\mathbb{Z},((1,2,3)^2,(1,3,2)^2)}]$ lives in $\overline{\mathcal{M}}_{2,4}$, since we intersect with λ_1 which is a pullback from $\overline{\mathcal{M}}_2$ we can specify `markings=[]` above to compute the pushforward `t` of this cycle to $\overline{\mathcal{M}}_2$ before intersecting. This significantly reduces the necessary computation time.

ACKNOWLEDGEMENTS. We are indebted to Aaron Pixton for letting us use and modify a previous implementation of operations in the tautological ring by him as well as pointing out several issues in `admcycles` which have now been fixed. We thank Frédéric Chapoton, Samuel Lelièvre and Jonathan Zachhuber for many valuable contributions. We are very grateful to the anonymous referees for many helpful comments, greatly improving the clarity of the paper and suggesting many improvements of the code.

We thank Harald Schilly and the team of CoCalc as well as Andrey Novoseltsev and the team of SageMathCell for making `admcycles` available on these platforms.

Delecroix was a guest of the Max Planck Institute and then of the Hausdorff Institut for Mathematics during the development of the project.

Schmitt was supported by the grant SNF-200020162928 and has received funding from the European Research Council (ERC) under the European Union Horizon 2020 research and innovation programme (grant agreement No 786580). During the last phase of the project, he profited from the SNF Early Postdoc Mobility grant 184245 and also wants to thank the Max Planck Institute for Mathematics in Bonn for its hospitality.

Van Zelm was supported by the Einstein Foundation Berlin during the course of this work.

SUPPLEMENT. The online supplement contains version 1.3.1 of `admcycles`.

REFERENCES.

[Arbarello and Cornalba 1987] E. Arbarello and M. Cornalba, "The Picard groups of the moduli spaces of curves", *Topology* **26**:2 (1987), 153–171. MR Zbl

[Arbarello et al. 2011] E. Arbarello, M. Cornalba, and P. A. Griffiths, *Geometry of algebraic curves, II*, Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences] **268**, Springer, 2011. MR Zbl

[Bae and Schmitt 2020] Y. Bae and J. Schmitt, "Chow rings of stacks of prestable curves II", 2020. arXiv

[Bae et al. 2020] Y. Bae, D. Holmes, R. Pandharipande, J. Schmitt, and R. Schwarz, "Pixton's formula and Abel–Jacobi theory on the Picard stack", 2020. arXiv

[Bainbridge et al. 2018] M. Bainbridge, D. Chen, Q. Gendron, S. Grushevsky, and M. Möller, "Compactification of strata of Abelian differentials", *Duke Math. J.* **167**:12 (2018), 2347–2416. MR Zbl

[Bainbridge et al. 2019a] M. Bainbridge, D. Chen, Q. Gendron, S. Grushevsky, and M. Möller, "The moduli space of multi-scale differentials", 2019. arXiv

[Bainbridge et al. 2019b] M. Bainbridge, D. Chen, Q. Gendron, S. Grushevsky, and M. Möller, "Strata of k-differentials", *Algebr. Geom.* **6**:2 (2019), 196–233. MR

[Borot et al. 2020] G. Borot, N. Do, M. Karev, D. Lewański, and E. Moskovsky, "Double Hurwitz numbers: polynomiality, topological recursion and intersection theory", 2020. arXiv

[Buryak and Rossi 2021] A. Buryak and P. Rossi, "Quadratic double ramification integrals and the noncommutative KdV hierarchy", *Bull. Lond. Math. Soc.* **53**:3 (2021), 843–854. MR Zbl

[Canning and Larson 2021] S. Canning and H. Larson, "The Chow rings of the moduli spaces of curves of genus 7, 8, and 9", 2021. arXiv

[Castorena and Gendron 2020] A. Castorena and Q. Gendron, "On the locus of genus 3 curves that admit meromorphic differentials with a zero of order 6 and a pole of order 2", 2020. arXiv

[Chen et al. 2019] D. Chen, M. Möller, and A. Sauvaget, "Masur–Veech volumes and intersection theory: the principal strata of quadratic differentials", 2019. With an appendix by G. Borot, A. Giacchetto, and D. Lewanski. arXiv

[Clader and Janda 2018] E. Clader and F. Janda, "Pixton's double ramification cycle relations", *Geom. Topol.* **22**:2 (2018), 1069–1108. MR Zbl

[CoCalc] SageMath, "CoCalc Collaborative Computation Online", available at https://cocalc.com/.

[Costantini et al. 2020a] M. Costantini, M. Möller, and J. Zachhuber, "The Chern classes and the Euler characteristic of the moduli spaces of abelian differentials", 2020. arXiv

[Costantini et al. 2020b] M. Costantini, M. Möller, and J. Zachhuber, "diffstrata – a Sage package for calculations in the tautological ring of the moduli space of Abelian differentials", 2020. arXiv

[Do and Lewański 2020] N. Do and D. Lewański, "On the Goulden–Jackson–Vakil conjecture for double Hurwitz numbers", 2020. arXiv

[Faber 1996] C. Faber, "Intersection-theoretical computations on $\overline{M}_g$", pp. 71–81 in *Parameter spaces* ((Warsaw, 1994)), edited by P. Pragacz, Banach Center Publ. **36**, Polish Acad. Sci. Inst. Math., Warsaw, 1996. MR Zbl

[Faber 1999] C. Faber, "Algorithms for computing intersection numbers on moduli spaces of curves, with an application to the class of the locus of Jacobians", pp. 93–109 in *New trends in algebraic geometry* (Warwick, 1996), edited by C. P. Klaus Hulek, Fabrizio Catanese and M. Reid, London Math. Soc. Lecture Note Ser. **264**, Cambridge Univ. Press, 1999. MR Zbl

[Faber and Pandharipande 2000] C. Faber and R. Pandharipande, "Logarithmic series and Hodge integrals in the tautological ring", pp. 215–252 , 2000. MR

[Farkas and Pandharipande 2018] G. Farkas and R. Pandharipande, "The moduli space of twisted canonical divisors", *J. Inst. Math. Jussieu* **17**:3 (2018), 615–672. MR

[Graber and Pandharipande 2003] T. Graber and R. Pandharipande, "Constructions of nontautological classes on moduli spaces of curves", *Michigan Math. J.* **51**:1 (2003), 93–109. MR

[Graber and Vakil 2005] T. Graber and R. Vakil, "Relative virtual localization and vanishing of tautological classes on moduli spaces of curves", *Duke Math. J.* **130**:1 (2005), 1–37. MR

[Grosse et al. 2019] H. Grosse, A. Hock, and R. Wulkenhaar, "A Laplacian to compute intersection numbers on $\overline{\mathcal{M}}_{g,n}$ and correlation functions in NCQFT", 2019. arXiv

[Grushevsky and Zakharov 2014] S. Grushevsky and D. Zakharov, "The zero section of the universal semiabelian variety, and the double ramification cycle", *Duke Math J.* **163**:5 (2014), 889–1070. arXiv

[Hain 2013] R. Hain, "Normal functions and the geometry of moduli spaces of curves", pp. 527–578 in *Handbook of moduli*, *I*, edited by G. Farkas and I. Morrison, Adv. Lect. Math. (ALM) **24**, International Press, Somerville, MA, 2013. MR Zbl

[Harer 1983] J. Harer, "The second homology group of the mapping class group of an orientable surface", *Invent. Math.* **72**:2 (1983), 221–239. MR Zbl

[Holmes and Schmitt 2019] D. Holmes and J. Schmitt, "Infinitesimal structure of the pluricanonical double ramification locus", 2019. arXiv

[Holmes et al. 2018] D. Holmes, J. L. Kass, and N. Pagani, "Extending the double ramification cycle using Jacobians", *Eur. J. Math.* **4**:3 (2018), 1087–1099. MR Zbl

[Holmes et al. 2019] D. Holmes, A. Pixton, and J. Schmitt, "Multiplicativity of the double ramification cycle", *Doc. Math.* **24** (2019), 545–562. MR Zbl

[Janda 2017] F. Janda, "Relations on $\overline{M}_{g,n}$ via equivariant Gromov–Witten theory of $\mathbb{P}^1$", *Algebr. Geom.* **4**:3 (2017), 311–336. MR

[Janda et al. 2017] F. Janda, R. Pandharipande, A. Pixton, and D. Zvonkine, "Double ramification cycles on the moduli spaces of curves", *Publ. Math. Inst. Hautes Études Sci.* **125** (2017), 221–266. MR

[Kass and Pagani 2019] J. L. Kass and N. Pagani, "The stability space of compactified universal Jacobians", *Trans. Amer. Math. Soc.* **372**:7 (2019), 4851–4887. MR Zbl

[Li 2001] J. Li, "Stable morphisms to singular schemes and relative stable morphisms", *J. Differential Geom.* **57**:3 (2001), 509–578. MR

[Li 2002] J. Li, "A degeneration formula of GW-invariants", *J. Differential Geom.* **60**:2 (2002), 199–293.

[Li and Ruan 2001] A.-M. Li and Y. Ruan, "Symplectic surgery and Gromov–Witten invariants of Calabi–Yau 3-folds", *Invent. Math.* **145**:1 (2001), 151–218. MR Zbl

[mgn] D. Johnson, "mgn – a Sage program for computing products, Faber-Zagier relations, and top intersections on the moduli space of stable curves", available at https://pypi.org/project/mgn/.

[Molcho et al. 2021] S. Molcho, R. Pandharipande, and J. Schmitt, "The Hodge bundle, the universal 0-section, and the log Chow ring of the moduli space of curves", 2021. arXiv

[Mumford 1983] D. Mumford, "Towards an enumerative geometry of the moduli space of curves", pp. 271–328 in *Arithmetic and geometry*, *II*, edited by M. Artin and J. Tate, Progr. Math. **36**, Birkhäuser, Boston, 1983. MR Zbl

[Norbury 2017] P. Norbury, "A new cohomology class on the moduli space of curves", 2017. arXiv

[Owens and Somerstep 2019] B. Owens and S. Somerstep, "Boundary Expression for Chern Classes of the Hodge Bundle on Spaces of Cyclic Covers", 2019. arXiv

[Pagani et al. 2020] N. Pagani, A. T. Ricolfi, and J. van Zelm, "Pullbacks of universal Brill–Noether classes via Abel–Jacobi morphisms", *Math. Nachr.* **293**:11 (2020), 2187–2207. MR

[Pandharipande 2018] R. Pandharipande, "A calculus for the moduli space of curves", pp. 459–487 in *Algebraic geometry*: *Salt Lake City* 2015, edited by T. de Fernex et al., Proc. Sympos. Pure Math. **97**, Amer. Math. Soc., Providence, RI, 2018. MR

[Pandharipande and Tseng 2019] R. Pandharipande and H.-H. Tseng, "Higher genus Gromov–Witten theory of $\mathrm{Hilb}^n(\mathbb{C}^2)$ and CohFTs associated to local curves", *Forum Math. Pi* **7** (2019), e4, 63. MR

[Pandharipande et al. 2015] R. Pandharipande, A. Pixton, and D. Zvonkine, "Relations on $\overline{M}_{g,n}$ via 3-spin structures", *J. Amer. Math. Soc.* **28**:1 (2015), 279–309. MR

[Pixton 2012] A. Pixton, "Conjectural relations in the tautological ring of $\overline{M}_{g,n}$", 2012. arXiv

[SageMath] The Sage Developers, "Sage Mathematics Software (version 9.0)", available at http://www.sagemath.org.

[SageMathCell] The Sage Developers, "SageMathCell, embeddable web interface for the Sage Mathematics Software System", available at https://sagecell.sagemath.org. (2020).

[Schmitt 2018] J. Schmitt, "Dimension theory of the moduli space of twisted k-differentials", *Doc. Math.* **23** (2018), 871–894. MR Zbl

[Schmitt and van Zelm 2020] J. Schmitt and J. van Zelm, "Intersections of loci of admissible covers with tautological classes", *Selecta Math.* (*N.S.*) **26**:5 (2020), Paper No. 79, 69. MR Zbl

[Yang 2008] S. Yang, "Calculating intersection numbers on moduli spaces of pointed curves", 2008. arXiv

RECEIVED: 16 Mar 2020 REVISED: 30 Jun 2021 ACCEPTED: 15 Jul 2021

VINCENT DELECROIX:

vincent.delecroix@u-bordeaux.fr
Laboratoire Bordelais de Recherche en Informatique, CNRS - Université de Bordeaux, Talence, France

JOHANNES SCHMITT:

schmitt@math.uni-bonn.de
Mathematical Institute, University of Bonn, Bonn, Germany

JASON VAN ZELM:

jasonvanzelm@outlook.com
Humboldt Universität zu Berlin, Berlin, Germany

Journal of Software for Algebra and Geometry vol 11, no 1, 2021