On $\int \mathrm{td}_X^{1/2}$ of O'Grady's exceptional hyperkähler varieties

Pieter Belmans

July 8, 2021

Abstract

In [3, Example 5.6(4)] (the first version) an exercise was left to the reader, to determine $\int \mathrm{td}_X^{1/2}$ for O'Grady's 6- resp. 10-dimensional hyperkähler varieties. The values turn out to be $\frac{2}{3}$ resp. $\frac{4}{15}$.

These coincide with the numbers for Kum_3- resp. $\mathrm{K3}^{[5]}$-type. After informing Chen Jiang of this he gave an abstract proof using the equality of Riemann–Roch polynomials, see [4, Lemma 5.7] (the second version).

In this note I detail the explicit computation, along with the code to do it, in the hope that it (in particular the code) is useful to someone.

Added later: it turns out that the result in [1, §3]. Also, there is the excellent Sage library `chow` [5] which does all the setting up and computations for you, abstracting away things into a flexible library. I've added the computation using `chow` for good measure (and advertising the library).

1 On $\int \mathrm{td}_X^{1/2}$ of hyperkähler varieties

The following is a combination of the lower bound from [2] and the upper bound from [3, Corollary 5.5].

Theorem 1 (Hitchin–Sawon, Jiang)**.** *Let X be a hyperkähler variety of dimension $2n > 2$. Then*

$$0 < \int \mathrm{td}_X^{1/2} < 1. \tag{1}$$

For $n = 1$ we have that $\int \mathrm{td}_{\mathrm{K3}} = 1$. The values for the infinite families of hyperkähler varieties are known, and given in [10, Proposition 19 and 21].

Proposition 2 (Sawon)**.** We have

$$\int \mathrm{td}^{1/2}_{\mathrm{K3}^{[n]}} = \frac{(n+3)^n}{4^n n!}, \tag{2}$$

$$\int \mathrm{td}^{1/2}_{\mathrm{Kum}^n} = \frac{(n+1)^{n+1}}{4^n n!}. \tag{3}$$

	c_2^3	$c_2 c_4$	c_6
$K3^{[3]}$	36800	14720	3200
Kum^3	30208	6784	448
OG6	30720	7680	1920

Table 1: Chern numbers for 6-dimensional hyperkähler varieties

The new[1] result in this note is the computation of $\int \mathrm{td}_X^{1/2}$ for O'Grady's 6- and 10-dimensional hyperkähler varieties; per the exercise left to the reader in [3, Example 5.6(4)] (the first version).

Proposition 3. We have

$$\int \mathrm{td}_{\mathrm{OG6}}^{1/2} = \frac{2}{3} \tag{4}$$

$$\int \mathrm{td}_{\mathrm{OG10}}^{1/2} = \frac{4}{15} \tag{5}$$

We'll collect some details in the computation section; but it is just an exercise in the explicit expression of the Todd class and expressing the Chern character using Chern classes, performed entirely within Sage.

Evaluating the formulas from Proposition 2 for $n = 3$ and $n = 5$ respectively gives the following (curious?)[2] coincidence.

Corollary 4. We have the following equalities:

$$\int \mathrm{td}_{\mathrm{OG6}}^{1/2} = \int \mathrm{td}_{\mathrm{Kum}^3}^{1/2} \tag{6}$$

$$\int \mathrm{td}_{\mathrm{OG10}}^{1/2} = \int \mathrm{td}_{\mathrm{K3}^{[5]}}^{1/2} \tag{7}$$

The important fact here is that the Chern numbers of these varieties are distinct, see Tables 1 and 2. These are collected from

- [8, page 128] for $K3^{[3]}$ and Kum^3
- [7, Appendix A] for Kum^5
- Jieao Song has computed the values for $K3^{[5]}$
- [6, Proposition 6.8] for OG6
- [9, Appendix A] for OG10

and these moreover allow for an independent check that the intermediate steps of the implementation are correct.

[1]After sharing the first version of this note with the author of [3] he was able to give a proof based on the equality of Riemann–Roch polynomials from [9], see [4, Lemma 5.7] (the second version).

[2]They are implied by the equality of Riemann–Roch polynomials, so I leave it up to the reader to decide how curious the equality is. The equality of the Riemann–Roch polynomials is proven in a somewhat ad hoc way after all.

	c_2^5	$c_2^3 c_4$	$c_2^2 c_6$	$c_2 c_8$	$c_2 c_4^2$	$c_4 c_6$	c_{10}
$K3^{[5]}$	126867456	52697088	12168576	1774080	21921408	5075424	176256
Kum^5	84478464	26220672	8141472	3141504	979776	142560	2592
OG10	127370880	53071200	12383280	1791720	22113000	5159700	176904

Table 2: Chern numbers for 10-dimensional hyperkähler varieties

Acknowledgements I thank Chen Jiang for interesting discussions about this integral and finding an abstract proof for the equality, Olivier Debarre for spotting a typo, and Thorsten Beckmann and Jieao Song for an interesting followup discussion.

2 The computation

By [3, §2.4] the square root of the Todd class of a hyperkähler variety X is given by

$$\mathrm{td}_X^{1/2} = \exp\left(-\sum_{k=1}^{\infty} \mathrm{b}_{2k}(2k)!\,\mathrm{ch}_{2k}(X)\right), \tag{8}$$

where b_{2k} is the $2k$*th modified Bernoulli number*, in the generating function

$$\sum_{k=0}^{+\infty} \mathrm{b}_{2k} x^{2k} = \frac{1}{2}\ln\left(\frac{\mathrm{e}^{x/2}-\mathrm{e}^{-x/2}}{x/2}\right) \tag{9}$$

or in a closed form expression given as

$$\mathrm{b}_{2k} := \begin{cases} \frac{\ln(2)}{2} & k=0 \\ \frac{\mathrm{B}_{2k}}{2(2k)^2\Gamma(2k)} & k \geq 1 \end{cases} \tag{10}$$

where B_{2k} is the $2k$*th Bernoulli number.*

The homogeneous part of degree $2n$ in $\mathrm{td}_X^{1/2}$ for a $2n$-dimensional hyperkähler variety X^{2n} for $n = 3$ (resp. $n = 5$) is given by

(11)

$$\begin{aligned}(\mathrm{td}_{X^6}^{1/2})_6 &= -\frac{1}{82944}\,\mathrm{ch}_2^3 - \frac{1}{5760}\,\mathrm{ch}_2\,\mathrm{ch}_4 - \frac{1}{504}\,\mathrm{ch}_6 \\ (\mathrm{td}_{X^{10}}^{1/2})_{10} &= -\frac{1}{955514880}\,\mathrm{ch}_2^5 - \frac{1}{19906560}\,\mathrm{ch}_2^3\,\mathrm{ch}_4 - \frac{1}{2764800}\,\mathrm{ch}_2\,\mathrm{ch}_4^2 - \frac{1}{580608}\,\mathrm{ch}_2^2\,\mathrm{ch}_6 \\ &\quad - \frac{1}{120960}\,\mathrm{ch}_4\,\mathrm{ch}_6 - \frac{1}{11520}\,\mathrm{ch}_2\,\mathrm{ch}_8 - \frac{1}{264}\,\mathrm{ch}_{10}\end{aligned}$$

so that after the substitution of the products of the ch_{2k} by the Chern numbers we end up with

(12)

$$\begin{aligned}(\mathrm{td}_{X^6}^{1/2})_6 &= \frac{31}{967680}\,\mathrm{c}_2^3 - \frac{11}{241920}\,\mathrm{c}_2\,\mathrm{c}_4 + \frac{1}{60480}\,\mathrm{c}_6 \\ (\mathrm{td}_{X^{10}}^{1/2})_{10} &= \frac{73}{3503554560}\,\mathrm{c}_2^5 - \frac{1073}{15328051200}\,\mathrm{c}_2^3\,\mathrm{c}_4 + \frac{311}{7664025600}\,\mathrm{c}_2\,\mathrm{c}_4^2 + \frac{61}{1277337600}\,\mathrm{c}_2^2\,\mathrm{c}_6 \\ &\quad - \frac{1}{45619200}\,\mathrm{c}_4\,\mathrm{c}_6 - \frac{53}{1916006400}\,\mathrm{c}_2\,\mathrm{c}_8 + \frac{1}{95800320}\,\mathrm{c}_{10}\end{aligned}$$

Now we can substitute the Chern numbers and we are done. These computations are all performed in Sage, and one can easily modify the code to see intermediate steps (such as the expression of the Chern character in terms of the Chern classes, etc.)

References

[1] Thorsten Beckmann. *Derived categories of hyper-Kähler manifolds: extended Mukai vector and integral structure.* arXiv: `2103.13382v2 [math.AG]`.

[2] Nigel Hitchin and Justin Sawon. "Curvature and characteristic numbers of hyper-Kähler manifolds". In: *Duke Math. J.* 106.3 (2001), pp. 599–615. MR: 1813238.

[3] Chen Jiang. *Positivity of Riemann–Roch polynomials and Todd classes of hyperkähler manifolds.* arXiv: `2008.04685v1 [math.AG]`.

[4] Chen Jiang. *Positivity of Riemann–Roch polynomials and Todd classes of hyperkähler manifolds.* arXiv: `2008.04685v2 [math.AG]`.

[5] Manfred Lehn and Christophe Sorger. *Chow.* URL: `https://www.math.sciences.univ-nantes.fr/~sorger/en/chow/`.

[6] Giovanni Mongardi, Antonio Rapagnetta, and Giulia Saccà. "The Hodge diamond of O'Grady's six-dimensional example". In: *Compos. Math.* 154.5 (2018), pp. 984–1013. MR: 3798592.

[7] Marc Nieper-Wisskirchen. "On the Chern numbers of generalised Kummer varieties". In: *Math. Res. Lett.* 9.5-6 (2002), pp. 597–606. MR: 1906063.

[8] Marc Nieper-Wißkirchen. *Chern numbers and Rozansky-Witten invariants of compact hyper-Kähler manifolds.* World Scientific Publishing Co., Inc., River Edge, NJ, 2004, pp. xxii+150. ISBN: 981-238-851-6. MR: 2110899.

[9] Ángel David Ríos Ortiz. *Riemann-Roch Polynomials of the known Hyperkähler Manifolds.* arXiv: `2006.09307v2 [math.AG]`.

[10] Justin Sawon. *Rozansky-Witten invariants of hyperkähler manifolds.* arXiv: `math/0404360v1 [math.DG]`.

A Bruteforce implementation

```
# expressions taken from Propositions 19 and 21 of Sawon's PhD thesis
def K3n(n): return (n+3)^n / (4^n * factorial(n))
def Kum_n(n): return (n+1)^(n+1) / (4^n * factorial(n))

def todd_root(n):
    """Determine the square root of the Todd class of a hyperkaehler

    Returns the square root of the Todd class expressed in terms of Chern numbers
    together with the Chern numbers as monomials in a polynomial ring.
    """
    def modified_bernoulli(n):
        if n == 0: return ln(2) / 2
        if n % 2 == 1: return 0
        return bernoulli(n) / (2 * n**2 * gamma(n))

    def exp(t, precision=n):
        return sum([t**i / factorial(i) for i in range(precision + 1)])

    R = PolynomialRing(QQ, ["ch" + str(2*i) for i in range(1, n+1)])

    # (1) truncated power series expansion of square root of Todd class
    result = prod([exp(-modified_bernoulli(2*k) * factorial(2*k) \
           * R.gen(k-1)) for k in range(1, n+1)])

    # (2) now collect degree 2*n part
    # not really Chern numbers but didn't have better name
    chern_numbers = WeightedIntegerVectors(n, range(1, n+1))
    chern_numbers = [prod([R.gen(i)**c for (i, c) in enumerate(C)]) for C in chern_numbers]
    result = sum([result.monomial_coefficient(ch) * ch for ch in chern_numbers])

    # (3) from ch_i to c_i
    S = PolynomialRing(QQ, ["c" + str(2*i) for i in range(1, n+1)])
    morphism = []

    for k in range(2, 2*n+1, 2):
        # go from the Chern character to Chern classes using symmetric functions
        Sym = SymmetricFunctions(QQ)
        terms = Sym.elementary()(Sym.powersum()[k]).monomial_coefficients().items()

        # select only the even Chern classes on a hyperkaehler
        terms = [(E, c) for (E, c) in terms if all([i % 2 == 0 for i in E])]

        expression = 1 / factorial(k) \
                * sum([c * prod([S.gen(i // 2 - 1) for i in E]) for (E, c) in terms])
        morphism.append(expression)

    result = R.hom(morphism, S)(result)

    # (4) list the Chern numbers that can appear
    chern_numbers = WeightedIntegerVectors(n, range(1, n+1))
    chern_numbers = [prod([S.gen(i)**c for (i, c) in enumerate(C)]) for C in chern_numbers]

    return (result, chern_numbers)

print("O'Grady 6-fold")
(square_root, chern_numbers) = todd_root(3)
square_root.parent().inject_variables()
values = {c2^3: 30720, c2*c4: 7680, c6: 1920}

value = sum([square_root.monomial_coefficient(c) * values[c] for c in values])
assert value == Kum_n(3)
print(value, "\n")


print("O'Grady 10-fold")
(square_root, chern_numbers) = todd_root(5)
square_root.parent().inject_variables()
values = {c2^5: 127370880, c2^3*c4: 53071200, c2^2*c6: 12383280, c2*c8: 1791720, \
          c2*c4^2: 22113000, c4*c6: 5159700, c10: 176904}

value = sum([square_root.monomial_coefficient(c) * values[c] for c in values])
assert value == K3n(5)
print(value, "\n")


print("Generalised Kummer variety of dimension 6-fold: computation by hand")
(square_root, chern_numbers) = todd_root(3)
square_root.parent().inject_variables()
values = {c2^3: 30208, c2*c4: 6784, c6: 448}

value = sum([square_root.monomial_coefficient(c) * values[c] for c in values])
assert value == Kum_n(3)
print(value, "\n")
```

B Using Chow

```
from sage.schemes.chow.all import *

# dimension 6
X = ChowScheme(6, ["c2", "c4", "c6"], [2, 4, 6])
X.chowring().inject_variables()
td = Sheaf(X, 6, 1 + sum(X.gens())).todd_class()
integrand = (td._logg()/2)._expp().by_degrees()[6]

# O'Grady 6
values = {c2^3: 30720, c2*c4: 7680, c6: 1920}
print(integrand.reduce([m - values[m] for m in values]))

# Kum^3
values = {c2^3: 30208, c2*c4: 6784, c6: 448}
print(integrand.reduce([m - values[m] for m in values]))


# dimension 10
X = ChowScheme(10, ["c2", "c4", "c6", "c8", "c10"], [2, 4, 6, 8, 10])
X.chowring().inject_variables()
td = Sheaf(X, 10, 1 + sum(X.gens())).todd_class()
integrand = (td._logg()/2)._expp().by_degrees()[10]

# O'Grady 10
values = {c2^5: 127370880, c2^3*c4: 53071200, c2^2*c6: 12383280, c2*c8: 1791720, \
          c2*c4^2: 22113000, c4*c6: 5159700, c10: 176904}
print(integrand.reduce([m - values[m] for m in values]))

# K3^[5]
values = {c2^5: 126867456, c4*c2^3: 52697088, c4^2*c2: 21921408, c6*c2^2: 12168576, \
          c6*c4: 5075424, c8*c2: 1774080, c10: 176256}
print(integrand.reduce([m - values[m] for m in values]))
```