# Twisted Hodge numbers for Hilbert schemes of points

***Release v1.0***

**Pieter Belmans**

**Dec 20, 2024**

# CONTENTS

Compute twisted Hodge numbers for Hilbert schemes of points

We implement the twisted Hodge number formula for Hilbert schemes of points on surfaces, as

- stated in Conjecture E of [2309.06244],
- proven in Theorem 1.1 of [2412.09975]

It reads

$$\sum_{n\geq 0}\sum_{p=0}^{2n}\sum_{q=0}^{2n}\mathrm{h}^{p,q}(\mathrm{Hilb}^n\, S, L_n)x^p y^q t^n = \prod_{k\geq 1}\prod_{p=0}^{2}\prod_{q=0}^{2}\left(1-(-1)^{p+q}x^{p+k-1}y^{q+k-1}t^k\right)^{-(-1)^{p+q}\mathrm{h}^{p,q}(S,L^{\otimes k})}.$$

Here, $L$ is a line bundle on a smooth projective surface $S$ (or a compact complex surface), $\operatorname{Hilb}^nS$ is the Hilbert scheme of $n$ points on $S$, and $L_n$ is the induced line bundle on the Hilbert scheme. One is referred to [2412.09975] for more details.

This formula computes the twisted Hodge numbers of $L_n$ on the Hilbert scheme, i.e.,

$$\mathrm{h}^{p,q}(\mathrm{Hilb}^n\, S, L_n)$$

An interesting example where things can be computed also using an explicit description of the Hilbert scheme is $\operatorname{Hilb}^2\mathbb{P}^2$, where the Hochschild cohomology (or rather, Hochschild–Kostant–Rosenberg decomposition) is computed, which corresponds to the twisted Hodge diamond using the anticanonical line bundle, cf. [Section 4.2, 2309.06244]:

```
sage: from twisted_hilbert import *
sage: S = CompleteIntersectionSurface([], 3) # anticanonical twist
sage: TwistedHilbertSchemeDiamond(S, 2).as_parallelogram()
  1
  0   8
  0   10   38
  0   0    35   80
  0   0    0    28   55
      0    0    0    0
           0    0    0
                0    0
                     0
```

- [2309.06244] Pieter Belmans, Lie Fu, Andreas Krug, Hochschild cohomology of Hilbert schemes of points on surfaces arXiv:2309.06244
- [2412.09975] Lie Fu: Twisted Hodge numbers and deformation theory of Hilbert schemes of points on surfaces via Hodge modules [arXiv:2412.09975](https://arxiv.org/abs/2412.09975)

AUTHORS:

- Pieter Belmans (2024-12-12): initial version

CHAPTER

# ONE

# TWISTED HODGE DIAMONDS

**class** twisted_hilbert.**TwistedHilbertSchemeDiamond**(*S:* TwistedSurfaceDiamonds, *n*)

Construct twisted Hodge diamond Hilbert schemes of $n$ points on $S$

INPUT:

- S – twisted Hodge diamonds for powers of a line bundle
- n – number of points

EXAMPLES:

Anticanonically twisted Hodge diamond of $\operatorname{Hilb}^2\mathbb{P}^2$:

```
sage: from twisted_hilbert import *
sage: S = CompleteIntersectionSurface([], 3)
sage: H = TwistedHilbertSchemeDiamond(S, 2)
sage: H
twisted Hodge diamond for Hilb^2 S
sage: H.pprint()
                    0
                0       0
            0       0       0
        0       0       0       0
  1       10      0       0       0
        8       35      0       0
            38      28      0
                80      0
                    55
```

Anticanonically twisted Hodge diamond of $\operatorname{Hilb}^2S$ where $S$ is a bielliptic surface with canonical bundle of order 2:

```
sage: TwistedHilbertSchemeDiamond(BiellipticSurface(2), 2).pprint()
                    0
                0       0
            0       2       0
        1       4       4       1
  1       3       8       3       1
        1       4       4       1
            0       2       0
                0       0
                    0
```

Anticanonically twisted Hodge diamond of $\operatorname{Hilb}^2S$ where $S$ is a bielliptic surface with canonical bundle of order 3:

```
sage: TwistedHilbertSchemeDiamond(BiellipticSurface(3), 2).pprint()
                  0
               0     0
            0     0     0
         1     1     1     0
      1     2     2     1     0
         1     1     1     0
            0     0     0
               0     0
                  0
```

twisted_hilbert.twisted_hilbert.**HilbertSchemeDeformations**(*HKR:* TwistedHodgeDiamond)

The degree 0, 1 and 2 cohomology of the tangent bundle of a Hilbert scheme

This implements Theorem 1.5 of [2412.09975]. It it expected that the user inputs the anticanonical Hodge diamond, describing the Hochschild cohomology of the surface.

For this one doesn't have to be able to compute all the twisted Hodge diamonds required to describe the twisted Hodge diamond giving all of the Hochschild cohomology of the Hilbert scheme, a single twisted Hodge diamond, for the anticanonical twist, suffices.

EXAMPLES:

Our favourite example is still $\operatorname{Hilb}^2\mathbb{P}^2$:

```
sage: from twisted_hilbert import *
sage: HilbertSchemeDeformations(CompleteIntersectionSurface([], 3)[1])
[8, 10, 0]
```

**class** twisted_hilbert.**TwistedHodgeDiamond**(*parent*, *M*)

Container structure for twisted Hodge diamonds.

EXAMPLES:

The twisted Hodge diamond for the projective plane and anticanonical bundle:

```
sage: from twisted_hilbert import *
sage: H = TwistedHodgeDiamond.from_matrix([[10, 0, 0], [8, 0, 0], [1, 0, 0]])
sage: H
twisted Hodge diamond
sage: H.pprint()
            0
         0     0
      1     0     0
         8     0
            10
```

Notice how (twisted) Hodge diamond are printed in a funny way, with $\mathrm{h}^{0,0}$ at the bottom.

**__init__**(*parent*, *M*)

Constructor for a TwistedHodgeDiamond (not to be called directly)

INPUT:

- `M` – matrix encoding twisted Hodge diamond

This function should not be called directly, use the class method *TwistedHodgeDiamond.from_matrix()* instead.

**classmethod from_matrix**(*M*)

Construct a twisted Hodge diamond from a matrix

INPUT:

- `M` – square matrix encoding twisted Hodge diamond

EXAMPLES:

The twisted Hodge diamond for the projective plane and anticanonical bundle:

```
sage: from twisted_hilbert import *
sage: H = TwistedHodgeDiamond.from_matrix([[10, 0, 0], [8, 0, 0], [1, 0, 0]])
sage: H.pprint()
              0
          0       0
      1       0       0
          8       0
              10
```

**dimension**()

Dimension of the variety underlying the twisted Hodge diamond

EXAMPLES:

```
sage: from twisted_hilbert import *
sage: EnriquesSurface()[0].dimension()
2
sage: TwistedHilbertSchemeDiamond(EnriquesSurface(), 3).dimension()
6
```

**pprint**()

Pretty print the twisted Hodge diamond

EXAMPLES:

The twisted Hodge diamond for the projective plane and anticanonical bundle:

```
sage: from twisted_hilbert import *
sage: H = TwistedHodgeDiamond.from_matrix([[10, 0, 0], [8, 0, 0], [1, 0, 0]])
sage: H.pprint()
              0
          0       0
      1       0       0
          8       0
              10
```

**as_parallelogram**()

Return the twisted Hodge diamond as polyvector parallelogram

It is up to the user to make sure that the twisted Hodge diamond is computed using the anticanonical bundle.

EXAMPLES:

Our favourite example is still $\operatorname{Hilb}^2\mathbb{P}^2$:

```
sage: from twisted_hilbert import *
sage: H = TwistedHilbertSchemeDiamond(CompleteIntersectionSurface([], 3), 2)
sage: H.as_parallelogram()
  1
  0   8
  0   10   38
  0   0    35   80
  0   0    0    28   55
      0    0    0    0
           0    0    0
                0    0
                     0
```

**__str__**()

Pretty print a twisted Hodge diamond

This gets called when you specifically print the object.

EXAMPLES:

The twisted Hodge diamond for the projective plane and anticanonical bundle:

```
sage: from twisted_hilbert import *
sage: H = TwistedHodgeDiamond.from_matrix([[10, 0, 0], [8, 0, 0], [1, 0, 0]])
sage: print(H)
            0
        0       0
    1       0       0
        8       0
            10
```

**__eq__**(*other*)

Compare two twisted Hodge diamonds

INPUT:

- `other` – the other twisted Hodge diamond

EXAMPLES:

Twisted Hodge diamonds for bielliptic surfaces are (not) the same:

```
sage: from twisted_hilbert import *
sage: BiellipticSurface(2)[0] == BiellipticSurface(3)[0]
True
sage: BiellipticSurface(2)[1] == BiellipticSurface(3)[1]
False
```

**__getitem__**(*key*)

Return $\mathrm{h}^{p,q}(X,L)$

This is

$$\dim \mathrm{H}^q(X, \Omega_X^p \otimes L)$$

corresponding to the entry indexed by $p$ and $q$ in the matrix, where we take *key*=*(p,q)*.

INPUT:

- `key`: tuple of indices for the twisted Hodge diamond

EXAMPLES:

The twisted Hodge diamond for the projective plane and anticanonical bundle:

```
sage: from twisted_hilbert import *
sage: H = TwistedHodgeDiamond.from_matrix([[10, 0, 0], [8, 0, 0], [1, 0, 0]])
sage: H[0, 0]
10
sage: H[1, 0]
8
```

**__hash__ = None**

**__weakref__**

list of weak references to the object

CHAPTER

# TWO

# SURFACES

**class** twisted_hilbert.**TwistedSurfaceDiamonds**

Encodes twisted Hodge diamonds of surface and powers of a line bundle

This makes it possible to implement both a class that knows all about a surface and one that only contains a finite amount of data.

**classmethod from_list**(*diamonds*)

Construct a *TwistedSurfaceDiamonds* object from a list of matrices

This is the basic approach, and limits the calculation of twisted Hodge numbers to however many entries are provided.

INPUT:

- `diamonds` – list of 3x3 matrices with twisted Hodge numbers

EXAMPLES:

Twisted Hodge numbers of the projective plane:

```
sage: from twisted_hilbert import *
sage: P2 = [None, None, None]
sage: P2[0] = matrix([[1, 0, 0], [0, 1, 0], [0, 0, 1]])
sage: P2[1] = matrix([[10, 0, 0], [8, 0, 0], [1, 0, 0]])
sage: P2[2] = matrix([[28, 0, 0], [35, 0, 0], [10, 0, 0]])
sage: S = TwistedSurfaceDiamonds.from_list(P2)
```

**__getitem__**(*k*)

Get the twisted Hodge diamond for the line bundle `L^k`

**__weakref__**

list of weak references to the object

**class** twisted_hilbert.**BiellipticSurface**(*order*)

*TwistedSurfaceDiamonds* for a bielliptic surface

The line bundle is the anticanonical line bundle

**__init__**(*order*)

**__getitem__**(*k*)

Get the twisted Hodge diamond for the kth power of the anticanonical bundle

INPUT:

- `k` – power of the anticanonical bundle

The calculations are based on the basic results of Section 4.3 of [2309.06244], and provide an independent verification of the conclusions.

EXAMPLES:

The bigraded version of Lemma 4.7:

```
sage: from twisted_hilbert import *
sage: BiellipticSurface(2)[1].as_parallelogram()
  1
  1   1
  0   2   0
      1   1
          1
sage: BiellipticSurface(3)[1].as_parallelogram()
  1
  1   1
  0   1   0
      0   0
          0
sage: BiellipticSurface(4)[1].as_parallelogram()
  1
  1   1
  0   1   0
      0   0
          0
sage: BiellipticSurface(6)[1].as_parallelogram()
  1
  1   1
  0   1   0
      0   0
          0
```

The bigraded version of Proposition 4.9:

```
sage: S = BiellipticSurface(2)
sage: TwistedHilbertSchemeDiamond(S, 2).as_parallelogram()
  1
  1   1
  0   3   0
  0   4   4   0
  0   2   8   2   0
      0   4   4   0
          0   3   0
              1   1
                  1
sage: S = BiellipticSurface(3)
sage: TwistedHilbertSchemeDiamond(S, 2).as_parallelogram()
  1
  1   1
  0   2   0
  0   1   1   0
  0   0   2   0   0
      0   1   1   0
          0   1   0
```

(continues on next page)

(continued from previous page)

```
              0   0
                  0
sage: S = BiellipticSurface(4)
sage: TwistedHilbertSchemeDiamond(S, 2).as_parallelogram()
  1
  1   1
  0   2   0
  0   1   1   0
  0   0   1   0   0
      0   0   0   0
          0   0   0
              0   0
                  0
sage: S = BiellipticSurface(6)
sage: TwistedHilbertSchemeDiamond(S, 2).as_parallelogram()
  1
  1   1
  0   2   0
  0   1   1   0
  0   0   1   0   0
      0   0   0   0
          0   0   0
              0   0
                  0
```

**class** twisted_hilbert.**CompleteIntersectionSurface**(*d*, *i=1*)

*TwistedSurfaceDiamonds* for a complete intersection

It is possible to vary the line bundle being used to any power of $\mathcal{O}(1)$

**__init__**(*d*, *i=1*)

Construct a complete intersection

The twisted Hodge numbers for complete intersection surfaces are computed using [twisted-hodge-ci].

- [twisted-hodge-ci] Twisted Hodge numbers for complete intersections twisted-hodge-ci

INPUT:

- `d` – degree, or list of degrees
- `i` (default: 1) – power of $\mathcal{O}(1)$ to be used

EXAMPLES:

Anticanonically twisted projective plane:

```
sage: from twisted_hilbert import *
sage: CompleteIntersectionSurface([], 3)[1].pprint()
          0
      0       0
  1       0       0
      8       0
          10
```

Quadric surface with default twist:

```
sage: CompleteIntersectionSurface(2)[1].pprint()
          0
      0       0
  0       0       0
      0       0
          4
```

del Pezzo surface of degree 4 with default (and anticanonical) twist:

```
sage: CompleteIntersectionSurface([2, 2])[1].pprint()
          0
      0       0
  1       2       0
      0       0
          5
```

**__getitem__**(*k*)

Get the twisted Hodge diamond for the line bundle `L^k`

**class** twisted_hilbert.**EnriquesSurface**

*TwistedSurfaceDiamonds* for an Enriques surface

The line bundle is the (anti)canonical line bundle

EXAMPLES:

The following is Appendix B of [AJM.2017.v21.n6.a4]:

```
sage: from twisted_hilbert import *
sage: TwistedHilbertSchemeDiamond(EnriquesSurface(), 2)[3, 1]
10
sage: TwistedHilbertSchemeDiamond(EnriquesSurface(), 2).pprint()
                  0
              0       0
          0       1       0
      0       0       0       0
  1       10      66      10      1
      0       0       0       0
          0       1       0
              0       0
                  0
```

- [AJM.2017.v21.n6.a4] Taro Hayashi, Universal covering Calabi–Yau manifolds of the Hilbert schemes of points of Enriques surfaces AJM.2017.v21.n6.a4

**__getitem__**(*k*)

Get the twisted Hodge diamond for the line bundle `L^k`

**class** twisted_hilbert.**ProductSurface**(*g*, *h*)

*TwistedSurfaceDiamonds* for the product of two curves

The line bundle is the anticanonical bundle.

**__init__**(*g*, *h*)

**__getitem__**(*k*)

Get the twisted Hodge diamond for the kth power of the anticanonical bundle

- k – power of the anticanonical bundle

EXAMPLES:

The quadric is the product of two curves of genus 0:

```
sage: from twisted_hilbert import *
sage: S = ProductSurface(0, 0)
sage: T = CompleteIntersectionSurface(2, 2)
sage: T[0] == S[0]
True
sage: T[3] == S[3]
True
sage: T[-5] == S[-5]
True
```

# PYTHON MODULE INDEX

## t

# INDEX

## Symbols



## A



## B



## C



## D



## E



## F



## H



## M



## P



## T