THE UNIVERSITY *of* EDINBURGH

Criterion for the Accumulation of ν-Walls

Luke Naylor

Doctor of Philosophy
University of Edinburgh
2024

Lay Summary

In mathematics, geometric spaces can be studied via the vector bundles over them. Possibly the most famous example of this is the 'hairy ball Theorem' with the image a hairy ball with 'tufts' at each pole. In mathematical terms, this is a statement about the existence of a non-vanishing section of the tangent bundle of the sphere. This can also expressed via a numeric invariant called the 'degree', which encodes how many points a section must vanish at, or in the case of the hairy ball, how many 'tufts' you inevitably get when trying to comb it. Compare this with a hairy doughnut, which you could comb without any tufts. This gives mathematicians a way to distinguish between different spaces, in particular by comparing a numerical invariant between their corresponding tangent spaces.

In algebraic geometry, invariants such as these do not completely determine a vector bundle. This has led to the construction of 'moduli spaces' which parametrise vector bundles with the same invariants. However these typically only parametrise a subset of 'stable' bundles. Somewhat analoguously to introducing complex numbers which can sometimes be used to solve problems about real numbers, mathematicians have been moving towards studying a generalisation of vector bundles called the 'derived category of coherent sheaves'. This thesis studies generalisations of the notion of 'stable' in this more abstract setting, and in particular when different notions of 'stable' coincide or differ.

Abstract

In this thesis we consider walls in the slice of geometric stability conditions for Chern characters on a principally abelian polarised surface. It is known that there is a vertical line of stability conditions in our choice of parametrisation, determined entirely numerically, which must intersect every wall, apart from a single vertical one. Furthermore if this line has rational horizontal coordinate β_-, then there can only be finitely many walls. In principal, when this horizontal coordinate is not rational, then there could be infinitely many walls. This has also been shown for certain examples and classes of Chern characters, but not exhaustively for all Chern characters where the aforementioned vertical line of stability conditions has irrational horizontal coordinate.

The main conclusion of this thesis is that all other classes of Chern characters not considered also have infinitely many walls. That is, a Chern character v on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ has finitely many walls if and only if $\beta_-(v)$ is rational. The other aspect of this thesis is developing methods to computing possibilities for walls more efficiently.

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*Luke Naylor*)

To my wife Kim.

Acknowledgements

Above all, I would like to show appreciation my supervisor Antony Maciocia. His experience and guidance on directions of research worth pursuing or ignoring was provided with patience and professionalism right up to the end of me writing this thesis. I am thankful for his knowledge to set me on this path of research, and the respect to support me on my own direction.

I would also like to thank Husniyah, Patrick, Hannah, Arman, Lucas, Dora, and Alisa. My PhD cohort, and the inaugural group to a training program that is now the GlaMS CDT. You all made the postgraduate researcher community here in Edinburgh a pleasure to be a part of. I will fondly remember the journey we made together, through the PhD, and the pandemic.

Before enrolling on the course, certain individuals played a vital role influencing my decision on whether or not to pursue the PhD. I value the grounding and certainty this gave me, instead of feeling like a train following tracks. Most notably are: my supervisor for a previous summer project Adam Paxton, my previous line manager Chris Purvis, and my father. Not all advice was followed, but it was all listened to.

Finally, I would like to thank those who have been close to me on a personal level and at a young age. My family have always helped me in my pursuits, and my mother has always been a strong believer in me which I appreciate. My teachers at the Cherwell academy were an inspiration. Mr Jamieson giving me confidence for my undergraduate applications, Mr Dyer for confidence in the entry exams, and Ms Tonks for planting the idea of living in Scotland. Last and not least, my wife, the person I have spent the most time with in my adult life. You have given me more support than I could have asked for, during and before this endeavour.

Notation

Symbol	Meaning
$\Delta(v)$, $\Delta(v_1, v_2)$	Bogomolov form on $v \in H^{\text{even}}_{\text{alg}}(X)$, also coincides with the Mukai pairing on an abelian surface (Definition 2.2, Theorem 1.6).
$\Delta^{\ell^2}(v)$	'Normalised' version of Bogomolov form on $v \in H^{\text{even}}_{\text{alg}}(X)$ when ℓ generates $\mathrm{NS}(X)$, given by $\Delta^{\ell^2}((r, c\ell, d\ell^2)) = c^2 - 2rd$, so that $\Delta(v) = \Delta^{\ell^2}(v)\ell^2$.
$\mu^{\ell}(v)$	Scaled Mumford slope of $v \in H^{\text{even}}_{\text{alg}}(X)$ when ℓ generates $\mathrm{NS}(X)$, given by $\mu^{\ell}((r, c\ell, \ldots)) = \frac{c}{r}$.
$\beta_-(v)$	Quantity associated to a Chern character $v \in H^{\text{even}}_{\text{alg}}(X)$ given by Definitions 1.29 and 1.30.
$\mathfrak{st}_G(x)$	Stabiliser subgroup of G for an element x of a set that G acts on.
$\mathrm{lcm}(n_1, n_2, \ldots)$, $\gcd(n_1, n_2, \ldots)$	Lowest common multiple, and greatest common divisor of integers n_i.
$\mathrm{SL}(n, k)$	Special Linear group of $n \times n$ matrices over a field k with. determinant 1
$\mathrm{GL}^+(2, \mathbb{R})$, $\widetilde{\mathrm{GL}}^+(2, \mathbb{R})$	General linear group of 2×2 real matrices with strictly positive determinant, and its universal cover.
$\mathrm{PGL}^{(+)}(2, k)$, $\mathrm{PSL}^{(+)}(2, k)$	Projectivisations of $\mathrm{GL}^{(+)}(2, k)$ and $\mathrm{SL}^{(+)}(2, k)$ respectively (quotiented by k^*).
$\mathrm{Symm}(2, k)$, γ	Set of symmetric 2×2 matrices over a field k, identified with $H^{\text{even}}_{\text{alg}}(\mathbb{T})$ via γ as per Definion 2.4.
$\mathcal{A}$	A generic abelian category.
$\mathcal{T}$	A generic triangulated category.
$\mathcal{D}^b(\mathcal{A})$, $\mathcal{D}^b(X)$	The bounded derived category over an abelian category $\mathcal{A}$, and $\mathrm{Coh}(X)$ respectively.
$\mathcal{O}_X$, $\mathcal{O}_x$	The structure sheaf associated to a ringed spaces X and the skyscraper sheaf supported on a closed point $x \in X$.
$\mathrm{ch}(E)$	Chern character of a coherent sheaf E (Definition 1.13).
$\mathrm{Coh}(X)$	The category of coherent sheaves on a complex scheme X.
$\mathfrak{Re}(z)$, $\mathfrak{Im}(z)$	Real and imaginary parts of $z \in \mathbb{C}$.
$\mathrm{NS}(X)$	$\mathrm{Pic}(X)/\mathrm{Pic}^0(X)$ The Néron–Severi group of X.
$\mathbb{T}$, L, ℓ	Generic principally polarised abelian surface with corresponding choice of polarisation L with $\ell \coloneqq c_1(L)$ and $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$.
$\mathcal{P}_{\hat{x}}$	Degree 0 line bundle on an abelian variety X corresponding to the point on the dual variety $\hat{x} \in \hat{X}$.
$\widetilde{\mathbb{C}^*}$	Universal cover of $\mathbb{C} \setminus \{0\}$.
$\mathcal{Z}$, $\mathcal{P}$	Generic central charge and slicing associated to a Bridgeland stability condition, as defined in Definition 1.11.
$\mathcal{B}^{\beta}$	Heart of the t-structure associated to the stability condition $\sigma_{\alpha,\beta}$ given by Definition 1.23.

Table of Contents

Introduction

The overall goal pursued in this thesis is to develop methods to compute possibilities for walls of Bridgeland stability conditions, and make the computations more accessible. This is only considered on the slice of stability conditions defined by Bridgeland on K3 surfaces (later generalised to other smooth projective complex surfaces), and more specifically in the case of Picard rank 1. A complete picture is only developed specifically for principally polarised abelian surfaces with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$.

The immediately relevant question when pursuing this goal is whether, for a given Chern character v, there are finitely many walls on the slice of stability condition considered. If there are finitely many, then we could in principle develop a tool to produce a list of possibilies. If there are infinitely many, then this is not possible, but we could maybe instead produce a formula for generating them sequentially, provided they are countable. Then the possibilities for walls could be computed up to an arbitrary number on demand.

Content Overview

For a Chern character v to have any Bridgeland stable objects (and for it to make sense to search for walls), we must have non-negative Bogomolov form $\Delta(v) \geq 0$. Among such Chern characters, there is a known numerical condition on a Chern character of a smooth projective complex surface which ensures that there are only finitely many walls. In this case, there is a pre-existing SageMath library which computes possibilities for walls, however it suffers from slow run time. The main content of this thesis starts at Part II 'Finite Walls' which explores this case and develops formulae for bounding possibilities and an alternative algorithm for computing possibilities for walls. Strictly, the set of possibilities generated by the new algorithm is slightly larger in some cases, as it currently does not use any criteria related to the Todd class to eliminate possibilities (integrality of the Euler characteristic and non-triviality of a certain extension group). The results of Part II are valid for all smooth projective complex surfaces, but generates fewer 'false positives' for a principally abelian polarised surface. The key victory of this new algorithm is the vast improvement in run time as shown in the following table comparing the existing library [Sch20b] with the new one accompanying this thesis [Nay23b].

Choice of v on $\mathbb{P}^2$	$(3, 2\ell, -2)$	$(3, 2\ell, -\frac{15}{2})$
[Sch20b, `tilt.walls_left`] exec time	∼20s	>1hr
[Nay23b] exec time	∼50ms	∼50ms

Part III 'Infinite Walls' is largely independent of the previous part and takes on a significantly different tone, using more theory as opposed to computing solutions to numerical inequalities. This part also deals with finding genuine walls, as opposed to just a set of possibilities. Other Chern characters v with $\Delta(v) \geq 0$ which do not satisfy the condition for Part II are considered, but specifically on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$.

We shall see that in all such cases, an infinite sequence of genuine walls exist. This is a phenomenon observed before, but not exhaustively for all cases (clarified in the background introduction to Part III and opening of Chapter 6). Finally concluding a simple numerical condition determining whether a Chern character on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, has finitely many walls with the following theorem, which was already proven in [Yos16] but this thesis provides an alternative presentation with some more explicit constructions, which could help build a program to apply it.

Theorem 9.6 (Criterion for the accumulation of $\boldsymbol{\nu}$-walls on principally polarised abelian surfaces)**.** *Let $\mathbb{T}$ be a principally polarised abelian complex surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, and $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ be a Chern character with non-negative rank $\mathrm{ch}_0(v) > 0$, non-negative Bogomolov discriminant $\Delta(v) \geq 0$.*

Then:

$$v \text{ has finitely many } \nu\text{-walls} \iff \beta_-(v) \in \mathbb{Q}$$

where $\beta_-((r, c\ell, d)) = \frac{c-\sqrt{c^2-rd}}{r}$.

The preliminaries for the main two Parts II and III are covered in Part I 'Preliminaries', as well as some other general context to the field. This mainly consists of an introduction to stability conditions on triangulated categories, some parametrisations in the case of $\mathcal{D}^b(X)$ of a smooth projective surface X, and Fourier-Mukai transforms.

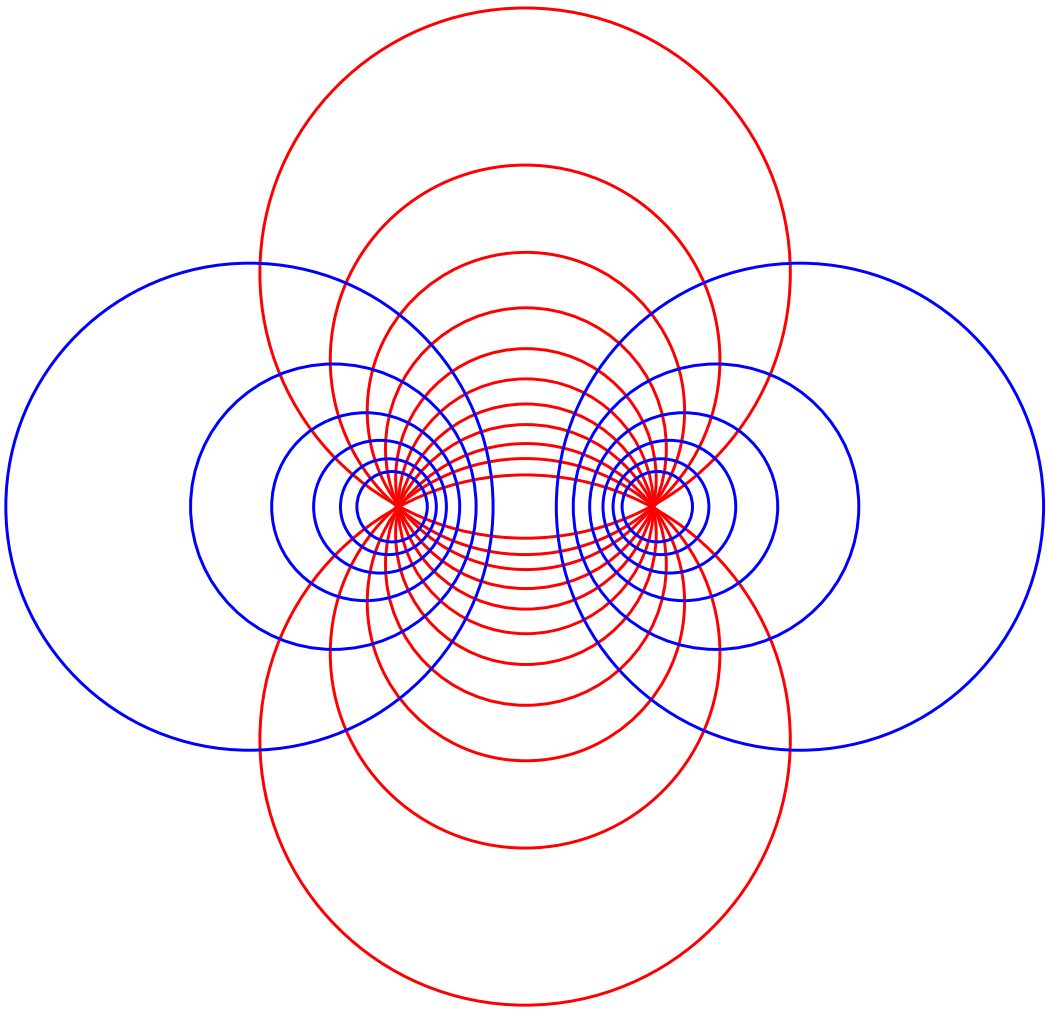
Reading Guide

The appendix contains code performing symbolic algebra calculations for proofs throughout this thesis, and also code for a library to compute possibilities for walls on stability conditions. The purpose of the former is to reduce explicit calculations in the main document, and sections of the appendix will be referenced from proofs in the main document for details. The latter, library code, is included for completeness but may be better viewed from the source provided in the corresponding appendix.

Coloured bars are used to visually indicate parts of the text which are most likely to be referenced from elsewhere, or are useful for understanding at a higher level. This includes theorems, definitions, examples, and problem statements. The mathematical context of the statements in a lemma or proposition should be specified in the containing section, but theorems and definitions should be self-contained statements.

Part I

Preliminaries



Chapter 1

Stability Conditions

When classifying vector bundles on a variety X, the first differentiators between them are numerical invariants such as rank and degree, among other higher dimensional invariants which can be encoded by invariants such as Chern classes. But these invariants do not necessarily determine the isomorphism class of a vector bundle. For a finer classification of vector bundles, there is a desire for 'moduli spaces' to parametrise isomorphism classes of vector bundles of a given set of invariants. This is hard to achieved for all isomorphism classes, but can be done when restricting to 'stable' vector bundles, such as Mumford stability on curves and Gieseker stability on higher dimensional spaces. These two stability conditions also satisfy a 'Harder-Narasimhan filtration' property, which allows any vector bundle to be decomposed into a filtration of semi-stable vector bundles, which they themselves can be decomposed into stable ones with a 'Jordan–Hölder' filtration. The latter objects being parametrised by moduli spaces that can be constructed.

There has been a wider movement to study the much larger space of the derived category of coherent sheaves $\mathcal{D}^b(X) \coloneqq \mathcal{D}^b(\mathrm{Coh}(X))$ on a variety X, instead of just vector bundles. This has motivated 'Bridgeland stability conditions' on $\mathcal{D}^b(X)$, as analogues of stability for vector bundles, to enable the construction of moduli spaces of stable derived objects. As opposed to the vector bundle case, 'Brigeland stability conditions' are referred to in plural here because we do not consider a single stability condition but in fact a whole space of them. A benefit of these notions of stability conditions on triangulated categories is that they fit better with equivalences of categories, with autoequivalences acting on the space of stability conditions. At the same time, classical Gieseker stability can also be recovered as a limit of Bridgeland stability conditions. This allows us to relate classically stable objects to Bridgeland stable derived objects, which is a strategy used in Part III of this thesis.

1.1 Transitioning to Stability on Triangulated Categories

Here we cover the transition from classical notions of stability, to Bridgeland stability conditions on triangulated categories. For a full account of the theory, we refer to [Bri07]. This section focuses on the transition specifically with a goal of exposition and illustration instead of proof. Specifically, we will expand on the opening example introduced in the second paragraph of [Bri07].

1.1.1 Stability on Abelian Categories

First, let us define the classical notions of stability.

Definition 1.1 (rank and degree)**.** Given a smooth projective variety X with a polarisation L, The rank and degree of a coherent sheaf F are defined as

$$\operatorname{rank}(F) \coloneqq \operatorname{ch}_0(F) \quad \text{and} \quad \deg(F) \coloneqq \operatorname{ch}_1(F) \cdot L^{\dim(X)-1}.$$

Definition 1.2 (Mumford Stability)**.** Let X be a smooth projective variety with polarisation L (choice of ample line bundle). Given a torsion-free sheaf $F \in \operatorname{Coh}(X)$, we say that F is **Mumford stable** (also called **μ-stable** or **slope stable**) if its **Mumford slope** is greater than that of any of its strict non-zero sub-objects:

$$\mu(E) < \mu(F) \qquad \text{whenever } 0 \neq E \subsetneq F.$$

Here the **Mumford slope** on a sheaf G is defined as

$$\mu(G) \coloneqq \frac{\deg(G)}{\operatorname{rank}(G)}.$$

Furthermore, if the above condition is weakened to allow for equality ($\mu(E) \leq \mu(F)$), then we say that F is **Mumford semistable**.

Remark. The "torsion free" restriction could be removed from the above Definition 1.2 if we extended μ to evaluate to $+\infty$ for torsion sheaves. Then sheaves which are non torsion free, but not torsion themselves, are immediately unstable as they are destabilised by their torsion subsheaf.

Definition 1.3 (Gieseker Stability)**.** Let X be a smooth projective variety with polarisation L (choice of ample line bundle). Given a torsion-free sheaf $F \in \operatorname{Coh}(X)$, we say that F is **Gieseker stable** (sometimes just called **stable**) if its **reduced Hilbert polynomial** is greater than that of any of its strict non-zero sub-objects:

$$p(E) < p(F) \qquad \text{whenever } 0 \neq E \subset F.$$

Where the **reduced Hilbert polynomial** is the Hilbert polynomial, which is defined as $P(G, m) = \chi(G \otimes L^m)$, scaled to be monic.

Furthermore, if the above condition is weakened to allow for equality ($p(E) \leq p(F)$), then we say that F is **Gieseker semistable**.

Remark. For a principally polarised surface with $\operatorname{NS}(\mathbb{T}) = \mathbb{Z}\ell$, which we shall work exclusively over in Part III, the reduced Hilbert polynomial is given in terms of the Chern character by

$$p(F) = x^2 + \frac{\operatorname{ch}_1(F) \cdot L}{\operatorname{ch}_0(F)} x + \frac{\operatorname{ch}_2(F)}{\operatorname{ch}_0(F)}.$$

So comparing Mumford slopes will equivalent to comparing the x coefficient of the reduced Hilbert polynomial.

In order to make the transition to stability on triangulated categories clearer, we will recognise Mumford stability as a Bridgeland stability condition. This will be done by restating it in terms of a 'central charge' function' instead of a slope function. The purpose of this will

be that the slope will not generalise well to the bounded derived category. Instead, we will use 'phases' which will represent choices of complex arguments of the central charge function, and will allow for 'wrapping' multiple times around the origin.

Definition 1.4 (Stability Function on an Abelian Category)**.** Let $\mathcal{A}$ be an abelian category, then a **stability function** is a group homomorphism $\mathcal{Z}\colon K(\mathcal{A}) \to \mathbb{C}$ with the following property:

$$\mathcal{Z}([a]) \in \{z \in \mathbb{C}\colon \mathfrak{Im}(z) > 0\} \cup \mathbb{R}_{<0},$$

for any $a \in \mathcal{A} \setminus \{0\}$.

The **phase** $\varphi(a)$ of $a \in \mathcal{A}\setminus\{0\}$, corresponding to $\mathcal{Z}$ is defined as the complex argument (chosen positive), divided by π. So we have:

$$\mathcal{Z}([a]) = |\mathcal{Z}([a])|\exp(i\pi\varphi(a)) \qquad 0 < \varphi(a) \leq 1,$$

for any $a \in \mathcal{A} \setminus \{0\}$.

An element $a \in \mathcal{A} \setminus \{0\}$ is **stable** with respect to this stability function $\mathcal{Z}$ if:

$$0, a \neq b \hookrightarrow a \Rightarrow \varphi(b) < \varphi(a).$$

Furthermore, if the inequality is weakened to allow equality in the above ($\varphi(b) < \varphi(a)$), then we say a is **semistable** with respect to this stability function.

Given a stability function $\mathcal{Z}$ on an abelian category $\mathcal{A}$. Since $\mathcal{Z}$ sends any element of $\mathcal{A} \setminus \{0\}$ into $\{\mathfrak{Im} > 0\} \cup \mathbb{R}_{<0}$, then comparing 'phases' of such elements, is the same as comparing the 'slopes' corresponding to this stability function:

$$\mu_{\mathcal{Z}}(a) = -\frac{\mathfrak{Re}(\mathcal{Z}(a))}{\mathfrak{Im}(\mathcal{Z}(a))} \qquad a \in \mathcal{A} \setminus \{0\},$$

where we allow $\mu_{\mathcal{Z}}(a) = +\infty$ when $\mathcal{Z}(a) \in \mathbb{R}_{<0}$.

Remark. Comparing slopes and comparing phases is only equivalent when the objects of interest are mapped by $\mathcal{Z}$ to the upper-half plane. This will no longer be the case as we generalise to triangulated categories, which is why it is necessary to introduce this 'phase'.

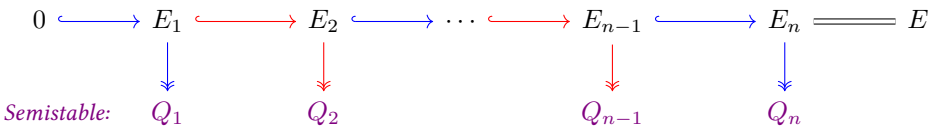
Example 1.1 (Stability Function for Mumford Stability)**.** *For the abelian category of abelian sheaves* $\mathrm{Coh}(X)$ *on a smooth projective curve* X*, we can define a stability function:*

$$\mathcal{Z}(a) = -\deg(a) + i\,\mathrm{rank}(a),$$

where the corresponding slope associated to this is precisely the Mumford slope given in Definition 1.2. So then, a coherent sheaf on X *is Mumford stable if and only if it is stable with respect to this stability function as in Definition 1.4.*

However, the stability function for Mumford stability has another desirable property:

Lemma 1.1 (Harder-Narasimhan Filtration for Mumford Stability)**.** *Let X be a smooth projective curve, and $E \in \mathrm{Coh}(X)$. Then there exists a unique filtration of objects $E_i \in \mathrm{Coh}(X)$ such that the successive quotients are Mumford semistable, with decreasing Mumford slopes:*

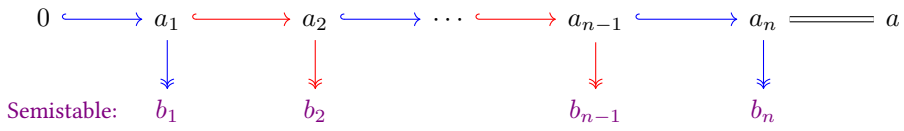


$$\mu(Q_1) > \mu(Q_2) > \cdots > \mu(Q_{n-1}) > \mu(Q_n)$$

Proof. [HL10, Theorem 1.3.4] gives this result, keeping in mind that Gieseker and Mumford stability coincide for smooth projective curves. □

Remark. This is somewhat similar to the Jordan-Hölder filtration for finite groups, expressing any object as an extension of a specific smaller class that will be easier to classify. However the main difference is that the ordering in the Harder-Narasimhan filtration is unique, due to the decreasing slopes; whereas the order of the successive quotients in the Jordan-Hölder filtration is not. There is also a "Jordan-Hölder" filtration in the context of stability conditions, breaking semistable objects into stable ones, but it is not relevant for this thesis.

Definition 1.5 (Stability Condition on an Abelian Category)**.** A **stability condition** on an abelian category $\mathcal{A}$ is given by a stability function $\mathcal{Z}$ (Definition 1.4) which satisfies the "Harder-Narasimhan" condition: that any $a \in \mathcal{A} \setminus \{0\}$ has a unique filtration where the successive quotients are semistable with respect to $\mathcal{Z}$, with decreasing phases (with respect to $\mathcal{Z}$):



$$\varphi(b_1) > \varphi(b_2) > \cdots > \varphi(b_{n-1}) > \varphi(b_n).$$

Finally, we require the **central charge** to be **numerical** in the sense that it factors through some $v\colon \mathcal{K}(\mathcal{A}) \to \Lambda$ to a finite rank lattice Λ, and that it satisfies the **support property**:

$$\inf\left\{\frac{|\mathcal{Z}(v(E))|}{||v(E)||} : 0 \neq E \in \mathcal{A} \text{ is semistable}\right\} > 0,$$

for a choice of norm $||\cdot||$ on the lattice Λ.

Remark. Chern characters will be the lattice elements in all explicit stability conditions in this thesis.

Remark. In the above definition, we could have replaced the phases φ with the slopes $\mu_{\mathcal{Z}}$ corresponding to the stability function.

1.1.2 Stability on Triangulated Categories

Triangulated Categories and t-Structures

Here is a brief summary on the topic of triangulated categories and t-structures. The relevance of these in our context is that the bounded derived category of coherent sheaves on a variety

($\mathcal{D}^b(X)$) is a triangulated category, and has a t-structure with heart $\mathrm{Coh}(X)$ (but also has other t-structures which is key to the ideas behind Bridgeland stability). For a more complete reference on the topic, Chapter 4 from [Gel96] is a good text to study.

The following notion will only be used in the context of triangulated categories defined slightly later in Definition 1.7. However, is stated first to avoid bloating definitions by nesting extra concepts.

Definition 1.6 (Triangles)**.** Given an additive category, with an autoequivalence T, a **triangle** is given by a triples of the following:

- Objects $a, b, c \in \mathcal{T}$ and morphisms.
- Morphisms $f\colon a \to b$, $g\colon b \to c$, and $h\colon c \to T(a)$.

Furthermore, each of the successive compositions $g \circ f$, $h \circ g$, and $f[1] \circ h$ are 0.

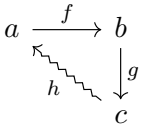
A **morphism of triangles** $a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} T(a)$ and $a' \xrightarrow{f'} b' \xrightarrow{g'} c' \xrightarrow{h'} T(a')$ is given by isomorphisms $e_a\colon a \to a'$, $e_b\colon b \to b'$, and $e_c\colon c \to c'$ such that the following commutes.

$$\begin{array}{ccccccc} a & \xrightarrow{f} & b & \xrightarrow{g} & c & \xrightarrow{h} & T(a) \\ \downarrow{\scriptstyle e_a} & & \downarrow{\scriptstyle e_b} & & \downarrow{\scriptstyle e_c} & & \downarrow{\scriptstyle T(e_a)} \\ a' & \xrightarrow{f'} & b' & \xrightarrow{g'} & c' & \xrightarrow{h'} & T(a') \end{array}$$

Furthermore, this is an **isomorphism of triangles** if the e_* are isomorphisms.

We can denote triangles by the following diagrams:

$$a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} T(a) \qquad\qquad a \xrightarrow{f} b \xrightarrow{g} c \overset{h}{\rightsquigarrow}$$



where $x \rightsquigarrow y$ is a shorthand for $x \to T(y)$.

Definition 1.7 (Triangulated Category)**.** An additive category $\mathcal{T}$ is called a triangulated category, if it has the following associated data a. and b., with which the axioms i. to vi. are satisfied:

a. A fully faithful automorphism T called the **shift functor (we denote $a[n]$ in place of $T^n(a)$ for $n \in \mathbb{Z}$, $a \in \mathcal{T}$)**.

b. A class of **exact triangles**, a subclass of **triangle**s.

In particular, the **exact triangles** form a class satisfying the axioms:

i. $a \xrightarrow{\mathrm{id}_a} a \to 0 \to a[1]$ is an exact triangle for any $a \in \mathcal{T}$.

ii. The class of exact triangle is closed under isomorphism of triangles (as in Definition 1.6).

iii. Any morphism $a \xrightarrow{f} b$ in $\mathcal{T}$ can be completed, not necessarily uniquely, to an exact triangle $a \xrightarrow{f} b \to c \rightsquigarrow$.
(a construction for c using this axiom can be called the '**cone**' $\mathrm{cone}(f)$).

iv. A triangle $a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} a[1]$ is exact if and only if $b \xrightarrow{g} c \xrightarrow{h} a[1] \xrightarrow{-f[1]} b[1]$ is.

v. If we have two triangles $a \xrightarrow{f} b \xrightarrow{g} c \xrightarrow{h} a[1]$ and $a' \xrightarrow{f'} b' \xrightarrow{g'} c' \xrightarrow{h'} a[1]$ and morphisms $e_a\colon a \to a'$, $e_b\colon b \to b'$, such that the following commutes:

$$\begin{array}{ccccccc} a & \xrightarrow{f} & b & \xrightarrow{g} & c & \xrightarrow{h} & T(a) \\ \downarrow{\scriptstyle e_a} & & \downarrow{\scriptstyle e_b} & & & & \downarrow{\scriptstyle T(e_a)} \\ a' & \xrightarrow{f'} & b' & \xrightarrow{g'} & c' & \xrightarrow{h'} & T(a'), \end{array}$$

then it can be completed, not necessarily uniquely, with a morphism $e_c\colon c \to c'$ in $\mathcal{T}$, to a morphism of triangles.

vi. The exact triangles satisfy a so-called **octahedral axiom** which states the following:

given exact triangles

- $a \xrightarrow{u} b \xrightarrow{j} c \xrightarrow{k} a[1]$
- $b \xrightarrow{v} d \xrightarrow{l} x \xrightarrow{i} b[1]$
- $a \xrightarrow{vu} d \xrightarrow{m} e \xrightarrow{n} a[1]$,

then there exists an exact triangle $c \xrightarrow{f} e \xrightarrow{g} x \xrightarrow{h} a[1]$, such that

$$l = gm, \quad k = nf, \quad h = j[1]i, \quad ig = u[1]n, \text{ and } \quad l = mv.$$

Remark. The relation to octahedra in the last axiom is clarified in [Gel96, Chapter 4, Section 1].

In the context of this thesis, the triangulated categories of concern are the derived category $\mathcal{D}^b(X) := \mathcal{D}^b(\mathrm{Coh}(X))$ of coherent sheaves $\mathrm{Coh}(X)$ on a smooth projective surface X.

Proposition 1.2. *The bounded derived category $\mathcal{D}^b(\mathcal{A})$ of an abelian category $\mathcal{A}$, is a triangulated category.*

Proof. [Gel96, Chapter 4, Section 2]. □

Elements of $\mathcal{D}^b(X)$ being represented by chain complexes in $\mathrm{Coh}(X)$, reflects a special relationship of $\mathrm{Coh}(X)$ in $\mathcal{D}^b$[1]. The following definition of a *t-structure* on a triangulated category is an axiomatisation of this relationship. This will also allow us to consider other full subcategories of $\mathcal{D}^b(X)$, with a similar relationship. One method for generating other t-structures with different hearts will be covered in Proposition 1.3, and will later be used in the construction of certain stability conditions on $\mathcal{D}^b(X)$.

Definition 1.8 (t-structure and heart)**.** Let $\mathcal{T}$ be a triangulated category, then a **t-structure** on $\mathcal{T}$ is given by a pair of strictly full subcategories $(\mathcal{T}^{\leq 0}, \mathcal{T}^{\geq 0})$ satisfying the following condition, where we denote $\mathcal{T}^{\leq n} := \mathcal{T}^{\leq 0}[-n]$, $\mathcal{T}^{\geq n} := \mathcal{T}^{\geq 0}[-n]$:

a. $\mathcal{T}^{\leq 0} \subset \mathcal{T}^{\leq 1}$ and $\mathcal{T}^{\geq 1} \subset \mathcal{T}^{\geq 0}$.

b. $\mathrm{Hom}(a, b) = 0$ for $a \in \mathcal{T}^{\leq 0}$, $b \in \mathcal{T}^{\geq 1}$.

[1] We can view $E \in \mathrm{Coh}(X)$ as an element of $\mathcal{D}^b(X)$ given by a complex concentrated in degree 0

c. Any $x \in \mathcal{T}$ fits into an exact triangle $a \to x \to b \to a[1]$ for some $a \in \mathcal{T}^{\leq 0}$, $b \in \mathcal{T}^{\geq 1}$.

Given a **t-structure** on a triangulated category $\mathcal{T}$, we define a corresponding **heart** as $\mathcal{A} := \mathcal{T}^{\geq 0} \cap \mathcal{T}^{\leq 0}$.

Remark. A key fact about hearts of a t-structure is that they are abelian categories. This definition can be found in [Gel96, Chapter 4, Section 2] with related material. Note that what is referred to here as a 'heart' is referred in that text as a 'core'.

Example 1.2. *$\mathcal{D}^b(X)$ has a t-structure with heart $\mathrm{Coh}(X)$. [Gel96, Chapter 4, Section 2, Proposition 3]*

The reason for this notation for the two subcategories of the t-structure is also consider the shifts of them, as used in the following lemma.

Lemma/Definition 1.8.1 (Truncation functors)**.** Let $\mathcal{T}$ be a triangulated category with t-structure $(\mathcal{T}^{\leq 0}, \mathcal{T}^{\geq 0})$, then there exist two *'truncation functors'* $\tau_{\leq n} \colon \mathcal{T} \to \mathcal{T}^{\leq n}$, and $\tau_{\geq n} \colon \mathcal{T} \to \mathcal{T}^{\geq n}$ that are right (resp. left) adjoint to the corresponding inclusion functor.

Any $x \in \mathcal{T}$ fits into an exact triangle

$$\tau_{\leq 0} x \to x \to \tau_{\geq 1} x \rightsquigarrow,$$

and any other exact triangle of the form $a \to x \to b \to a[1]$ with $a \in \mathcal{T}^{\leq 0}$, $b \in \mathcal{T}^{\geq 1}$, is canonically isomorphic to the one above.

Proof. [Gel96, Chapter 4, Section 2, Lemma 5] □

A tool for constructing other t-structures, using an existing t-structure on a triangulated category $\mathcal{T}$ is via taking 'torsion theories' on the heart from the former t-structure. This theory is not used later in this section about transitioning to stability condition on triangulated categories. However this is a natural question to ask at this point and is used in the construction of stability conditions.

Definition 1.9 (torsion theory on an abelian category)**.** Let $\mathcal{A}$ be an abelian category, then a **torsion theory** on $\mathcal{A}$ is a pair of full subcategories $(\mathcal{T}, \mathcal{F})$ of $\mathcal{A}$ satisfying

- $\mathrm{Hom}(t, f) = 0$ for any $t \in \mathcal{T}$, $f \in \mathcal{F}$.
- Any $a \in \mathcal{A}$ fits in an exact sequence $t \hookrightarrow a \twoheadrightarrow f$ for some $t \in \mathcal{T}$, $f \in \mathcal{F}$.

Proposition 1.3. *Suppose that we have a* 'torsion theory' *$(\mathcal{T}, \mathcal{F})$ on $\mathcal{A}$, then there exists a t-structure on $\mathcal{D}^b(\mathcal{A})$ which has the following as its heart (instead of $\mathcal{A}$):*

$$\mathcal{B} := \left\{ E \in \mathcal{D}^b(\mathcal{A}) \colon \mathcal{H}^{-1}(E) \in \mathcal{F}, \mathcal{H}^0(E) \in \mathcal{T}, \mathcal{H}^i(E) = 0 \text{ for other } i \right\}.$$

Proof. [HRS96, Proposition 2.1] □

Bridgeland Stabilities

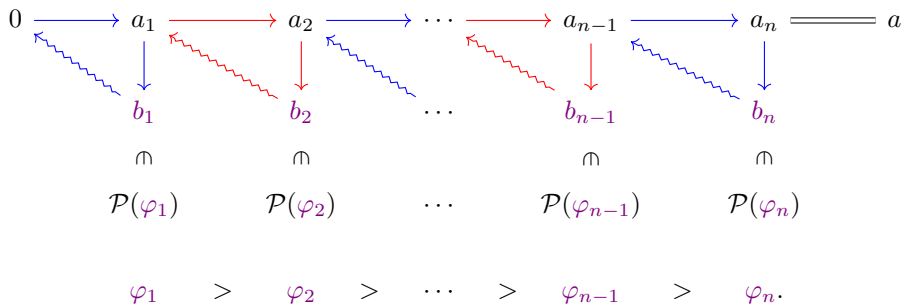
Definition 1.10 (Stability Condition on a t-Structure)**.** Let $\mathcal{T}$ be a triangulated category with a t-structure giving a heart $\mathcal{A}$. Then a stability condition on this t-Structure is given

by a pair $(\mathcal{Z}, \mathcal{P})$. Where $\mathcal{Z}$ is a group homomorphism:

$$\mathcal{Z}\colon \mathcal{K}(\mathcal{T}) \to \mathbb{C}.$$

And $\mathcal{P}$, called a **slicing** of $\mathcal{T}$, is a family of subcategories of $\mathcal{T}$ (containing objects we interpret as 'semistable') parametrised by $\varphi \in \mathbb{R}$ (we call 'phases') with the properties:

- If $a \in \mathcal{P}(\varphi) \setminus \{0\}$, then $\mathcal{Z}(a) = r\exp(i\pi\varphi)$ with $r \in \mathbb{R}_{>0}$
  (i.e. the phases are choices of complex arguments of $\mathcal{Z}(a)$ for 'semistable' objects a).
- $\mathcal{P}(\varphi) \subseteq \mathcal{A}$ for all $\varphi \in (0, 1]$.
- $\mathcal{P}(\varphi + 1) = \mathcal{P}(\varphi)[1]$ for all $\varphi \in \mathbb{R}$.
- If $\varphi_1 > \varphi_2$ are real numbers, and $a_i \in \mathcal{P}(\varphi_i)$, then $\hom(a_1, a_2) = 0$
  (i.e. a 'semistable' object cannot map non-trivially to another stable object of lower phase).
- Finally, there is a collection of exact triangles analogous to the Harder-Narasimhan filtration seen before, for any $a \in \mathcal{T} \setminus \{0\}$:



Finally, we require the **central charge** to be **numerical** in the sense that it factors through some $v\colon \mathcal{K}(\mathcal{A}) \to \Lambda$ to a finite rank lattice Λ, and that it satisfies the **support property**:

$$\inf\left\{\frac{|\mathcal{Z}(v(E))|}{||v(E)||} : 0 \neq E \in \mathcal{P}(\varphi),\ \ \varphi \in \mathbb{R}\right\} > 0,$$

for a choice of norm $||\cdot||$ on the lattice Λ.

Remark. For a stability function on an abelian category $\mathcal{A}$, we associated phases to all elements of $\mathcal{A}$. However, here, the phases are only associated to the 'semistable' objects which include some elements of $\mathcal{A}$, and also their shifts.

Lemma 1.4 (Connection to stability condition on the heart)**.** *Let $\mathcal{T}$ be a triangulated category which has a t-structure with heart $\mathcal{A}$. Then the data for a stability condition on this t-structure is equivalent to the data for the data for a stability condition on the heart $\mathcal{A}$.*

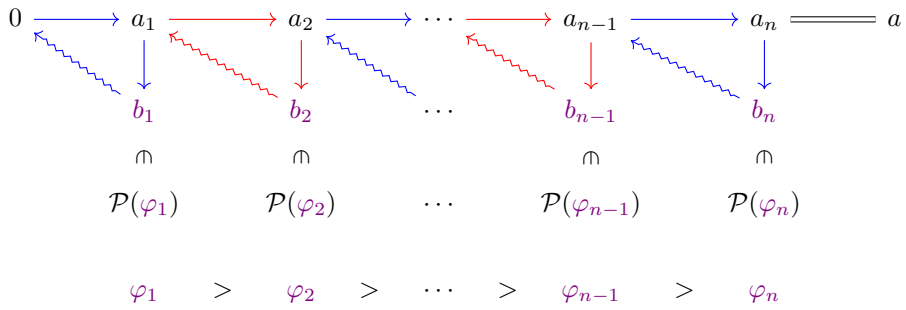
Furthermore, if $\mathcal{P}$ is the slicing from a stability condition on the t-structure, then the semistable objects in $a \in \mathcal{A}$ with $\mathcal{Z}(a) \in \exp(i\pi\varphi)\mathbb{R}_{>0}$, $0 < \varphi \leq 1$ (with respect to the corresponding stability condition on the abelian category), are precisely the objects in $\mathcal{P}(\varphi)$.

Proof. This is a special case of [Bri07, Proposition 5.3].

□

Definition 1.11 (Stability Condition on a Triangulated Category)**.** Let $\mathcal{T}$ be a triangulated category. A stability condition on $\mathcal{T}$ is given by a pair $(\mathcal{Z}, \mathcal{P})$ where $\mathcal{Z}$ is a group homomorphism $\mathcal{Z}\colon K(\mathcal{T}) \to \mathbb{C}$, and $\mathcal{P}$ is a family of subcategories of $\mathcal{T}$ parametrised by $\varphi \in \mathbb{R}$ with the properties:

- If $a \in \mathcal{P}(\varphi) \setminus \{0\}$, then $\mathcal{Z}(a) = r\exp(i\pi\phi)$ with $r \in \mathbb{R}_{>0}$.
- For all $\varphi \in \mathbb{R}$, we have $\mathcal{P}(\varphi + 1) = \mathcal{P}(\varphi)[1]$.
- If $\varphi_1 > \varphi_2$ are real numbers, and $a_i \in \mathcal{P}(\varphi_i)$, then $\hom(a_1, a_2) = 0$.
- Finally, there is a collection of exact triangles analoguous to the Harder-Narasimhan filtration seen before, for any $a \in \mathcal{T} \setminus \{0\}$:



Finally again, we require the **central charge** to be **numerical** in the sense that it factors through some $v\colon K(\mathcal{A}) \to \Lambda$ to a finite rank lattice Λ, and that it satisfies the **support property**:

$$\inf\left\{\frac{|\mathcal{Z}(v(E))|}{||v(E)||} : 0 \neq E \in \mathcal{P}(\varphi),\ \ \varphi \in \mathbb{R}\right\} > 0,$$

for a choice of norm $||\cdot||$ on the lattice Λ.

The next theorem highlights two viewpoints on Bridgeland stability conditions, each useful in different contexts.

Theorem 1.5 (Connection to stability on a t-structure)**.** *Given a triangulated category $\mathcal{T}$, then the data for a Bridgeland stability condition on $\mathcal{T}$ is equivalent to the data for a bounded t-structure, and a stability condition on the heart.*

Proof. This is one of the main results in [Bri07]. Briefly, starting from a Bridgeland stability condition $(\mathcal{Z}, \mathcal{P})$ on $\mathcal{T}$, one can recover the t-structure using the slicing, with the heart being generated by $\mathcal{P}(t)$ for $t \in (0, 1]$. Vice-versa, given a t-structure on $\mathcal{T}$ with heart $\mathcal{A}$, a stability condition on $\mathcal{A}$ gives a stability condition on the t-structure in the way stated in Definition 1.10, as per Lemma 1.4. □

Each stability condition will give a moduli space of semistable objects. The topic of interest in this thesis, as clear from the title, is finding 'walls'. These walls correspond to boundaries in the space of stability conditions where the moduli spaces of semistable objects change as we cross it.

Definition 1.12 (Stability wall)**.** Given $v \in H^{\text{even}}_{\text{alg}}(X)$, a **stability wall** is a codimension one subspace of the space of stability conditions such that for each point of the wall, there is a neighbourhood where there exists $E \in \mathcal{D}^b(X)$ with $\text{ch}(E) = v$ such that E is semistable with respect to the stability condition on one side of the wall, and not on the other.

Remark. The support property for stability conditions is related to a local finiteness property for the walls [BM11, Appendix B].

Bridgeland Stability Conditions on a Smooth Projective Surface

In the context of this thesis, we will be considering Bridgeland stability conditions on the derived category $\mathcal{D}^b(\mathcal{S}) \coloneqq \mathcal{D}^b(\mathrm{Coh}(\mathcal{S}))$ of coherent sheaves on a smooth projective surface $\mathcal{S}$. Our definitions of Bridgeland stability conditions required them to be *numerical*, that is, the central charge should factor through a finite rank lattice. This lattice is chosen to be $H^{\mathrm{even}}_{\mathrm{alg}}(\mathcal{S})$ via the Chern character map, defined now.

Definition 1.13 (Chern character)**.** On a smooth projective surface $\mathcal{S}$, we define the Chern character map

$$\begin{aligned} \mathrm{ch}\colon \mathrm{Coh}(\mathcal{S}) &\to H^{\mathrm{even}}_{\mathrm{alg}}(\mathcal{S}) \cong \mathbb{Z} \oplus \mathrm{NS}(\mathcal{S}) \oplus \mathbb{Z} \\ E &\mapsto \left(c_0(E), c_1(E), -c_2(E) + \frac{1}{2} c_1(E)^2 \right), \end{aligned}$$

where c_i are the Chern classes [Har13, Appendix A.3].

Remark. The Chern character map is additive on short exact sequences, so factors through the Grothendieck group $\mathcal{K}(\mathrm{Coh}(\mathcal{S}))$.

Next is a common property for Bridgeland stability conditions in the context of algebraic geometry. All stability conditions in this thesis will be *geometric*. In fact, in the case of a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, the main surface considered in this thesis, all stability conditions are geometric, as shown in [Bri08]. For context, distinguishing between varieties with only geometric stability conditions and those which have non-geometric stability conditions is a current topic of interest, but not relevant in this thesis.

Definition 1.14 (Geometric stability condition)**.** Given a smooth projective surface $\mathcal{S}$, a stability condition on $\mathcal{D}^b(\mathcal{S})$ is said to be **geometric** if for each closed point $x \in \mathcal{S}$, the skyscraper sheaf $\mathcal{O}_x$ is stable, and they all have the same phase.

In the context of Bridgeland stability on sheaves, the next theorem is a useful numerical restriction on the Chern character, which is useful for eliminating possibilities for destabilising sub-objects, giving walls. This is a key in Part II.

Theorem 1.6 (Bogomolov-Gieseker inequality)**.** *Let $E \in \mathcal{D}^b(\mathcal{S})$, for a smooth projective surfaces $\mathcal{S}$, be semistable with respect to some geometric Bridgeland stability condition. Then it satisfies the* Bogomolov-Gieseker inequality*:*

$$0 \leq \Delta(E) \coloneqq \mathrm{ch}_1(E)^2 - 2\,\mathrm{ch}_0(E)\,\mathrm{ch}_2(E).$$

Proof. [MS19, Theorem 6.14] □

Natural Actions on Bridgeland Stability Conditions

Recall that Theorem 1.5 gave two viewpoints on Bridgeland stability conditions. The viewpoint of the stability condition on the heart of the associated t-structure allows us to consider 'semistabilising' or 'destabilising' subobjects and sequences, due to the heart being an abelian category. This will be the main viewpoint in the main Parts II and III of this thesis. However

the second broader viewpoint on the whole triangulated category allows us to see some natural actions on the space of Bridgeland stability conditions. This section aims to give a heuristic overview of these actions. A complete proof of the well-definedness of these actions is given in [Bri07, Lemma 8.2].

These natural actions on the stability conditions are seen by considering the actions on the central charges of the stability conditions $\mathcal{Z}: K(\mathcal{T}) \to \mathbb{C}$ which are compatible with the slicing, to be discussed throughout this section. The two natural group actions come from pre-composing, with an autoequivalence of the triangulated category, or post-composing with an $\mathbb{R}$-linear automorphism of $\mathbb{C}$.

Autoequivalences of the Triangulated Category

First let us consider the natural action from pre-composing the central charge.

Definition 1.15 (Action of autoequivalences on Bridgeland Stabilities)**.** Given a triangulated category $\mathcal{T}$, an autoequivalence ψ, and a stability condition $\sigma = (\mathcal{Z}, \mathcal{P})$ on $\mathcal{T}$, we define the stability condition $\psi \cdot \sigma = (\mathcal{Z}', \mathcal{P}')$ as follows:

$$\mathcal{Z}'(a) = \mathcal{Z}(\psi^{-1}(a)),$$

$$\mathcal{P}'(t) = \psi(\mathcal{P}(t)).$$

This action is defined in such a way that ψ maps semi-stable objects with respect to σ to semi-stable objects with respect to $\psi \cdot \sigma$ and preserves the central charge. Harder-Narasimhan filtrations for σ are also mapped to Harder-Narasimhan for $\psi \cdot \sigma$. The fact that ψ preserves exact triangles will directly imply that this definition satisfies the properties of a Bridgeland stability condition.

$\widetilde{\mathrm{GL}}^+(2, \mathbb{R})$

Next we consider the natural action from post-composing the central charge, to give a different stability condition with the same semi-stable objects, but with different values for the central charge (and consequently phases). To preserve the linearity of the central charge, anything used to post-compose it must be linear itself. Also, this linear map must preserve the orientation of the complex plane to ensure that sub-objects of semi-stable objects do not have larger phase. This leads us to consider post-composing the central charge with an element of $\mathrm{GL}^+(2, \mathbb{R})$. Finally when taking the phases into account, which can be considered as choices for complex arguments of the central charges of semi-stable objects, rotating the complex plane anti-clockwise continuously should continuously increase the phases. So after continuously rotating the complex plane, finishing with a full rotation, the phases should be increased by 2 despite the final corresponding linear map being the identity. This point forces us to use the universal cover $\widetilde{\mathrm{GL}}^+(2, \mathbb{R})$ to be compatible with the phase aspect of stability conditions.

Definition 1.16 (Action of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ on Bridgeland Stabilities)**.** Given a Bridgeland stability condition $\sigma = (\mathcal{Z}, \mathcal{P})$, and an element $g \in \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$, we define the stability condition $g \cdot \sigma = (\mathcal{Z}', \mathcal{P}')$ as follows:

$$\mathcal{Z}'(a) = g\mathcal{Z}(a),$$

where $g \in \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ acts on $\mathbb{C}$ via its projection onto $\mathrm{GL}^+(2,\mathbb{R})$, and

$$\mathcal{P}'(t) = \mathcal{P}({f_g}^{-1}(t)).$$

Here $f_g \colon \mathbb{R} \to \mathbb{R}$ is a continuous increasing bijection with $f_g(t+1) = f_g(t)+1$ determined by g with the property $g\exp(i\pi t) \in \exp(i\pi f_g(t))\mathbb{R}_{>0}$.

Remark. Elements $g \in \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ are distinguished from other elements with the same projection onto $\mathrm{GL}^+(2,\mathbb{R})$ by this function f_g. For other such elements, the corresponding function f_g will differ by a constant even integer. This corresponds to having the same central charge and semi-stable objects, but different choices of complex arguments for them.

One significant subgroup of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ is $\widetilde{\mathbb{C}^*}$ which can be thought of as rotations and scalings of the central charge. The action of this subgroup is often used to 'normalise' geometric stability conditions so that skyscraper sheaves have central charge -1, with phase 1. The elements of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ which preserve the central charges of objects with integer-valued phase include the following 'skewing' and 'stretching' of the upper-half $\mathbb{C}$-plane. As well as not modifying the semi-stable objects, by preserving integer-valued phases, these actions also preserve the heart of the t-structure associated to the slicing.

Definition 1.17 (Horizontal 'skewing' of the central charge)**.** Given a numerical stability condition $\sigma = (\mathcal{Z}, \mathcal{P})$, we can **'skew'** the image of the central charge by applying the action of $\mathrm{skew}_x := \begin{pmatrix} 1 & x \\ 0 & 1 \end{pmatrix}$. So then the central charge would be given by:

$$\begin{pmatrix} 1 & x \\ 0 & 1 \end{pmatrix} \cdot \mathcal{Z}(a) = \mathfrak{Re}(\mathcal{Z}(a)) + i\left(\mathfrak{Im}(\mathcal{Z}(a)) + x\mathfrak{Re}(\mathcal{Z}(a))\right).$$

Definition 1.18 (Vertical 'stretching' of the central charge)**.** Given a numerical stability condition $\sigma = (\mathcal{Z}, \mathcal{P})$, we can **'stretching'** the image of the central charge by applying the action of $\mathrm{stretch}_y := \begin{pmatrix} 1 & 0 \\ 0 & y \end{pmatrix}$ for $y \in \mathbb{R}_{>0}$. So then the central charge would be given by:

$$\begin{pmatrix} 1 & 0 \\ 0 & y \end{pmatrix} \cdot \mathcal{Z}(a) = \mathfrak{Re}(\mathcal{Z}(a)) + iy\mathfrak{Im}(\mathcal{Z}(a)).$$

1.2 Stability Condition Parametrisations on Smooth Projective Surfaces

When working in the context of stabilities on the derived category of coherent sheaves on a smooth projective surface $\mathcal{S}$, there are a few existing parametrisations for certain slices of stability conditions. Some of them shall be explored in the next sections. They shall also be

specified slightly more in the case when $\mathcal{S}$ has Picard rank 1, that is, its Néron–Severi group is generated by a single element. In these cases the formulae reduce to purely numerical expressions. And then finally, the differences between these parametrisations is discussed, as well as how to express one in terms of the other.

1.2.1 Parametrisation of Hearts for Stability Condition

Using the tilting method from Proposition 1.3, we can now construct alternative hearts to t-structures on $\mathcal{D}^b(\mathcal{S})$ for a smooth projective surface $\mathcal{S}$. All the stability conditions given by the parametrisations in the upcoming Section 1.2.2 will involve the hearts in the following Definition 1.19. This does not mean that all hearts of stability conditions are among the ones considered here, however for any geometric stability condition σ there exists a $g \in \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ such that the heart associated to $g \cdot \sigma$ is one given by Definition 1.19[2]. These particular hearts are convenient because the elements only have non-zero cohomology in two positions.

Definition 1.19 (Tilt categories)**.** Let $\mathcal{S}$ be a smooth projective surface. Define the following **tilt categories**:

$$\mathcal{B}^{H,B} \coloneqq \left\{ E \in \mathcal{D}^b(\mathcal{S}) \colon \mathcal{H}^{-1}(E) \in \mathcal{T}_{H,B},\ \mathcal{H}^0(E) \in \mathcal{F}_{H,B},\ \mathcal{H}^i(E) = 0 \text{ o.w.} \right\},$$

where:

- $\mathcal{T}_{H,B}$ consists of sheaves where the torsion-free parts have μ_ω-semistable Harder-Narasimhan factors of slope $\mu_H > B \cdot H$.
- $\mathcal{F}_{H,B}$ consists of torsion-free sheaves such that each of their μ_H-semistable Harder-Narasimhan factors have slope $\mu_H \leq B \cdot H$.

Remark. The hearts $\mathcal{B}^{H,B}$ associated to the stability conditions above do not change when scaling H by a scalar $\mathbb{R}$ factor. Furthermore, for any fixed H, the heart only depends on $H \cdot B$.

Following this remark, we will specialise this notation in the case of Picard rank 1 surfaces to only have one degree of freedom in the parameters.

Definition 1.20 (Tilt categories on Picard rank 1 surfaces)**.** Let $\mathcal{S}$ be a smooth, projective, Picard rank 1 surface with choice of ample line bundle L where $\ell \coloneqq c_1(L)$ generates $\mathrm{NS}(\mathcal{S})$. We define:

$$\mathcal{B}^\beta \coloneqq \mathcal{B}^{\ell,\beta\ell} = \left\{ E \in \mathcal{D}^b(\mathcal{S}) \colon \mathcal{H}^{-1}(E) \in \mathcal{T}_\beta,\ \mathcal{H}^0(E) \in \mathcal{F}_\beta,\ \mathcal{H}^i(E) = 0 \text{ o.w.} \right\},$$

where:

$$\begin{aligned} \mathcal{F}_\beta \coloneqq \mathcal{F}_{\ell,\beta\ell} &= \left\{ E \in \mathcal{D}^b(\mathcal{S}) \colon \mu^\ell(F) < \beta \text{ whenever } F \hookrightarrow E \right\} \\ \mathcal{T}_\beta \coloneqq \mathcal{T}_{\ell,\beta\ell} &= \left\{ E \in \mathcal{D}^b(\mathcal{S}) \colon \mu^\ell(F) \geq \beta \text{ whenever } E \twoheadrightarrow F \right\} \\ \mu^\ell(A) &\coloneqq \frac{c_1(A)\ell}{c_0(A)\ell^2} \in \mathbb{Q}. \end{aligned}$$

Remark. Definitions for the Mumford slope with respect to a polarisation typically do not have the ℓ^2 term in the denominator above. However, expressing a slope this way gives

[2] Follows from Theorem 1.9

$\mu^\ell((r, c\ell, d\ell^2)) = c/r$, which allows to make calculations independently of the quantity ℓ^2 when Chern characters are expressed this way.

1.2.2 Parametrisations for Geometric Stability Conditions

One fact to keep in mind when relating the different parametrisations is the following.

Proposition 1.7. *A geometric stability condition, on a smooth projective surface, is determined, up to even shifts, by its central charge.*

Proof. [Bri08, Proposition 10.3] □

First, let us consider the stability conditions originally constructed by Bridgeland on K3 surfaces. These have since been generalised to smooth projective surfaces.

Definition 1.21 (Mukai vector and pairing)**.** Let $E \in \mathcal{D}^b(\mathcal{S})$, for a smooth projective surface $\mathcal{S}$, we define the **Mukai vector** of E as follows:

$$v(E) \coloneqq \mathrm{ch}(E)\sqrt{\mathrm{td}_{\mathcal{S}}} \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathcal{S}),$$

with the **Mukai pairing** $\langle \_, \_ \rangle$ defined by:

$$\langle v(E), v(F) \rangle \coloneqq -\sum_i (-1)^i \dim \mathrm{Ext}^i(E, F).$$

The above definition will be specialised to principally polarised abelian surfaces with $\mathrm{NS} = \mathbb{Z}\ell$ in the next chapter with Definition 2.2.

Definition 1.22 (Parametrisation B [Bri08, Section 6])**.** Given choices

$$B, H \in \mathrm{NS}_{\mathbb{R}}(\mathcal{S}) \coloneqq \mathrm{NS}(\mathcal{S}) \otimes \mathbb{R},$$

with H being ample. There exists a stability condition $\sigma^B_{B,H}$ with a central charge:

$$\mathcal{Z}^B_{B,H}(E) \coloneqq \langle \exp(B + iH), v(E) \rangle,$$

where $v(E) \coloneqq \mathrm{ch}(E)\sqrt{\mathrm{td}(\mathcal{S})}$ is the Mukai vector for E, and the heart of the associated t-structure is $\mathcal{B}^{B,H}$, given by Definition 1.19.
This all holds provided that $\mathcal{Z}^B_{B,H}(E) \notin \mathbb{R}_{\leq 0}$ for any spherical sheaf E on $\mathcal{S}$.

The expression for $\mathcal{Z}^B_{B,H}(E)$ expanded gives:

$$\mathcal{Z}^B_{B,H}(E) = \left(v_1(E) \cdot B + \frac{1}{2} v_0(E)(H^2 - B^2) - v_2(E) \right) + iH \cdot (v_1(E) - v_0(E)B). \quad (1.1)$$

Proposition 1.8 (Properties of parametrisation B - Bridgeland)**.** *The slice of stability conditions given by Definition 1.22 satisfy the following properties:*

- *Each $\sigma^B_{B,H}$ is a geometric stability condition such that $\mathcal{Z}^B_{B,H}(\mathcal{O}_x) = -1$ and assigned a phase of 1.*
- *Any other geometric stability condition on $\mathcal{S}$ is given by $g \cdot \sigma^B_{B,H}$ for unique B, H and $g \in \widetilde{GL}^+(2, \mathbb{R})$.*

Proof. [Bri08, Proposition 10.3] □

Next, let us consider the slice of stability conditions that will be used in all other chapters in this thesis. In this chapter, this parametrisation will be referred to with the label 'A', however outside this label will be implicit.

Definition 1.23 (Parametrisation A [ABL07, Section 2, 'Our charges']). Let $\mathcal{S}$ be a smooth projective surface with a choice of ample line bundle H such that $\ell := c_1(H)$ generates $\mathrm{NS}(X)$. Given $\alpha, \beta \in \mathbb{R}$ and $\alpha > 0$. There exists a stability condition $\sigma^A_{\alpha,\beta}$ with a central charge given by:

$$\mathcal{Z}^A_{\alpha,\beta}(E) := -\operatorname{ch}_2\left(\exp(-\beta\ell - i\alpha\ell)\cdot \operatorname{ch}(E)\right),$$

and where the heart of the associated t-structure is $\mathcal{B}^\beta$ as given by Definition 1.20.

Remark. In the case of a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, which we will exclusively work over in Part III, these stability conditions are identical to the ones given by parametrisation B in Definition 1.22. This is because the Todd class is trivial and so the Chern character and Mukai vector of objects are identical.

Remark. Note that definitions for the Mumford slope with respect to a polarisation typically does not have the ℓ^2 term in the denominator above. However, expressing a slope this way gives $\mu^\ell((r, c\ell, d\ell^2)) = c/r$, which allows to make calculations independently of the quantity ℓ^2 when Chern characters are expressed this way.

Proposition 1.9 (Characterisation of parametrisation A). *Let X be a smooth projective surface, then the stability conditions given by parametrisation A (Definition 1.23) satisfy the following properties:*

- *They are geometric stability conditions where the skyscrapers all have phase 1.*
- *The central charge satisfies* $\mathcal{Z}((0,\ell,0))^2 + 2\mathcal{Z}((1,0,0))\ell^2 = 0$.

Proof. The first statement holds as these stability conditions are a subset of the ones given by parametrisation B. The second statement follows from the observation that if the numerical condition holds for a central charge $\mathcal{Z}$, then $\mathcal{Z} = \mathcal{Z}_{\alpha,\beta}$ where we can now recover α and β from $\beta + i\alpha = \mathcal{Z}((0,\ell,0))/\ell^2$. □

Remark. If the first condition from Proposition 1.9 holds, then the second condition is equivalent to $\mathcal{Z}((0,\ell,0))^2 - 2\mathcal{Z}((1,0,0))\mathcal{Z}((0,0,\ell^2)) = 0$. This latter equation is preserved by multiplying the central charge by any complex number. Therefore, that condition characterises stability conditions in the same orbit as some $\sigma^A_{\alpha,\beta}$ under the action of $\widetilde{\mathbb{C}^*} \subset \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$.

Definition 1.24 (Parametrisation F [FLZ22, Theorem 3.4]). Given an ample line bundle H on a smooth projective surface $\mathcal{S}$, and for certain $\alpha, \beta \in \mathbb{R}$. There exists a stability condition $\sigma^F_{\alpha,\beta,H}$ with a central charge given by:

$$\mathcal{Z}^F_{\alpha,\beta,H}(E) := \left(-\operatorname{ch}_2(E) + \alpha H^2 \operatorname{ch}_0(E)\right) + i\left(H\operatorname{ch}_1(E) - \beta H^2 \operatorname{ch}_0(E)\right).$$

The heart of the associated t-structure is $\mathcal{B}^{\beta H}$.

Parametrisation B gave all geometric stability conditions up to the action of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$, however we could also consider a larger slice of numerical stability conditions, which still

assign a phase of 1 to all skyscraper sheaves (and so geometric), but generate all geometric stability conditions up to the action of $\widetilde{\mathbb{C}^*} \subset \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ instead.

To build this larger family of stability conditions, we would include the ones given by parametrisation A, but also others generated by applying the actions of $g \in \widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ which preserve the phase of the skyscrapers (equal to 1). These correspond to $g \in \mathrm{GL}^+(2,\mathbb{R})$ which fix $\mathbb{R} \subseteq \mathbb{C}$, such as skew_x and $\mathrm{stretch}_y$ from Definitions 1.17 and 1.18. We understand those as elements in the universal cover which fix elements of integer-valued phase.

One description, given in [Del23, Theorem 5.5], for this larger family is given by the following parametrisation for the central charges. This is done at the cost of not corresponding to stability conditions for all combinations of values for the parameters.

Definition 1.25 (Parametrisation D [Del23, Theorem 5.5])**.** Let X be a smooth projective surface, $\alpha, \beta \in \mathbb{R}$, and $\omega, B \in \mathrm{NS}_{\mathbb{R}}(X)$ with $\omega^2 > 0$. Then we can define a function

$$\mathcal{Z}^D_{\alpha,\beta,\omega,B}(E) \coloneqq (\alpha - i\beta)^2\omega^2 \operatorname{ch}_0(E) + (B + i\omega)\operatorname{ch}_1(E) - \operatorname{ch}_2(E).$$

Remark. Notice that this family also have 2 extra $\mathbb{R}$ degrees of freedom (given by α and β), on top of the family $\sigma^A_{\omega,B}$.

Not all functions given above are necessarily central charges of stability conditions. For any given β, B, ω, they only correspond to a stability condition for sufficiently large α. This is specified in [Del23, Theorem 5.10] with conditions stemming from the restrictions on the stability conditions in Definition 1.22.

In the case where X has Picard rank 1, we can write the above $\mathcal{Z}^D_{\alpha,\beta,\omega,B}$ as

$$\begin{aligned}\mathcal{Z}^D_{\alpha,\beta,h\ell,b\ell}((r, c\ell, d\ell^2)) &= \left((h^2(\alpha - i\beta)^2)r + (b + ih)c - d\right)\ell^2 \\ &= z_1 r + z_2 c - d\ell^2 \qquad\qquad (\text{for some } z_i \in \mathbb{C}).\end{aligned}$$

The latter being more explicitly the generic form for a $\mathbb{Z}$-linear map $H^{\mathrm{even}}_{\mathrm{alg}}(X) \to \mathbb{C}$ mapping $(0,0,1) \mapsto -1$. This form of central charge also appears in [BM11, Theorem 2.5], as well as restrictions on the $z_i \in \mathbb{C}$ for them to be central charges of some geometric stability condition (on $\mathbb{P}^2$ in this case).

1.3 Conversions Between Parametrisations

This section calculates explicitly how to express stability conditions from the earlier parametrisations, into one of the others, for the case of smooth projective Picard rank 1 surfaces. Parametrisation B from Definition 1.22 depends on the Todd class of the surface, whereas all the others are expressed in terms of the Chern characters. So any conversions to and from that parametrisation would depend on the surface.

1.3.1 Parametrisation A to D

The stability conditions described by parametrisation A form a subset of those described by D.

$$\begin{aligned}\mathcal{Z}^A_{\alpha,\beta}((r,c\ell,d\ell^2)) &= \left(\frac{1}{2}(\beta+i\alpha)^2 r + (\beta+i\alpha)c - d\right)\ell^2 \\ &= \left(\alpha^2\left(\frac{\beta}{4\alpha}-\frac{-i}{4}\right)^2 r + (\beta+i\alpha)c - d\right)\ell^2 \\ &= \mathcal{Z}^D_{\frac{\beta}{4\alpha},\frac{-1}{4},\alpha,\beta}((r,c\ell,d\ell^2))\end{aligned}$$

Given that geometric stability conditions are determined by their central charges, up to a double shift ($[2n]$) of the slicing (Proposition 1.7), we can equate the corresponding stability conditions because they both assign a phase of 1 to skyscraper sheaves:

$$\sigma^A_{\alpha,\beta} = \sigma^D_{\frac{\beta}{4\alpha},\frac{-i}{4},\alpha,\beta} \qquad \alpha\in\mathbb{R}_{>0}, \beta\in\mathbb{R}.$$

1.3.2 Parametrisation D to A

Remark. The computer algebra calculations corresponding to this section can be found in Appendix C.1.2

Consider a geometric stability condition $\sigma^D_{\alpha,\beta,h\ell,b\ell}$ which has central charge $\mathcal{Z}^D_{\alpha,\beta,h\ell,b\ell}$ from Definition 1.25. By Proposition 1.9, this should correspond to some stability condition $\sigma^A_{?,?}$, up to an action of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$. And since they assign a phase of 1 to skyscraper sheaves, that $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ action should be expressible as a combination of skew_x and $\mathrm{stretch}_y$ from Definitions 1.17 and 1.18.

So consider an arbitrary 'skewing' and 'stretching' of the central charge, in the hopes of reversing this action:

$$\mathcal{Z} \coloneqq \mathrm{stretch}_y \cdot \mathrm{skew}_x \cdot \mathcal{Z}^D_{\alpha,\beta,h\ell,b\ell}.$$

This corresponds to a stability condition assigning phase 1 to skyscrapers, so it is one of the form $\sigma^A_{?,?}$ if and only if the following condition holds, at which points we can read off the parameters as the imaginary and real parts of $\mathcal{Z}((0,\ell,0)$.

$$\mathcal{Z}(0,\ell,0)^2 + 2\mathcal{Z}(1,0,0)\ell^2 = 0 \tag{1.2}$$

This equates to:

$$-2\,\beta h^2 x + h^2x^2 - 2i\,\beta h^2 y + 2i\,h^2xy - h^2y^2 + 2\,\alpha h^2 + 2\,bhx + 2i\,bhy + b^2 = 0.$$

Considering the imaginary part gives $-2\,(\beta h - hx - b)hy = 0$. Given that $h > 0$ from the definition of parametrisation D, and that $y > 0$ by definition of $\mathrm{stretch}_y$ (so that it has positive determinant), this fixes x:

$$x = \beta - \frac{b}{h}.$$

Then considering the real part in the condition from Equation (1.2), and substituting the fixed value of x, we get:

$$y^2 = -\beta^2 + 2\,\alpha + \frac{2\,b\beta}{h}.$$

Therefore, provided that the right-hand-side is positive, x and y can be found to satisfy the

Stability condition	Parametrisation A	Parametrisation D
$\sigma^A_{\alpha,\beta}: \quad \alpha \in \mathbb{R}_{>0},\ \beta \in \mathbb{R}$	$\sigma^A_{\alpha,\beta}$	$\sigma^D_{\frac{\beta}{4\alpha},\frac{-1}{4},\alpha,\beta}$
$\sigma^D_{\alpha,\beta,h\ell,b\ell}: \quad \alpha,\beta,h,b \in \mathbb{R}$ $\alpha, h > 0 \ \ \alpha > \frac{\beta^2}{2} - \frac{b\beta}{h}$	$\begin{pmatrix} 1 & -\frac{\beta - b/h}{y} \\ 0 & \frac{1}{y} \end{pmatrix} \cdot \sigma^A_{hy,\ \beta h}$ where $y \coloneqq \sqrt{-\beta^2 + 2\,\alpha + \frac{2\,b\beta}{h}}$	$\sigma^D_{\alpha,\beta,h\ell,b\ell}$

Table 1.1: Conversion Table

condition in Equation (1.2). Finally, consider $\mathcal{Z}((0,\ell,0))$ for x, y satisfying this condition:

$$\mathrm{stretch}_y \cdot \mathrm{skew}_{\beta - \frac{b}{h}} \cdot \mathcal{Z}^D_{\alpha,\beta,h\ell,b\ell}(0,\ell,0) = \beta + iy,$$

where $y \coloneqq \sqrt{-\beta^2 + 2\,\alpha + \frac{2\,b\beta}{h}}$. Which then allows us to conclude

$$\mathrm{stretch}_y \cdot \mathrm{skew}_{\beta - \frac{b}{h}} \cdot \sigma^D_{\alpha,\beta,h\ell,b\ell} = \sigma^A_{hy,\ \beta h},$$

rearranging to

$$\begin{aligned} \sigma^D_{\alpha,\beta,h\ell,b\ell} &= \mathrm{skew}_{-\beta + \frac{b}{h}} \cdot \mathrm{stretch}_{\frac{1}{y}} \cdot \sigma^A_{hy,\ \beta h} \\ &= \begin{pmatrix} 1 & -\frac{\beta h - b}{h\sqrt{\frac{2\,b\beta - \left(\beta^2 - 2\,\alpha\right)h}{h}}} \\ 0 & \frac{1}{\sqrt{\frac{2\,b\beta - \left(\beta^2 - 2\,\alpha\right)h}{h}}} \end{pmatrix} \cdot \sigma^A_{hy,\ \beta h}, \end{aligned}$$

provided that $-\beta^2 + 2\,\alpha + \frac{2\,b\beta}{h} > 0$.

1.4 A Closer Look at Parametrisation A

Outside this chapter, only smooth projective surfaces with Picard rank 1 will be considered, and specifically principally polarised abelian surfaces with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ for large parts. The parametrisation for the stability conditions used will be the one from Definition 1.23. The B superscript will be dropped on $\mathcal{Z}_{\alpha,\beta}$ and $\sigma_{\alpha,\beta}$. For this parametrisation, we shall be using ν for the slope associated (the analogue of μ for Mumford stability), which we shall use in practice for comparing phases of objects in the heart of the stability condition $\sigma_{\alpha,\beta}$.

Definition 1.26 (ν-slope)**.**

$$\nu_{\alpha,\beta} \coloneqq \frac{-\mathfrak{Re}(\mathcal{Z}_{\alpha,\beta})}{\mathfrak{Im}(\mathcal{Z}_{\alpha,\beta})}$$

So for this section, let X be a smooth projective surface and L be a choice of ample line bundle such that $\ell \coloneqq c_1(L)$ generates $\mathrm{NS}(X)$. We use $(\beta,\alpha) \in \mathbb{R} \times \mathbb{R}_{>0}$ to parametrise the stability conditions $\sigma^B_{\alpha,\beta}$ so we can use the upper-half real plane to illustrate this slice of stability conditions.

Another relevant piece of notation that is common in the context of this slice of stability conditions is 'twisted' Chern characters.

Definition 1.27 (Twisted Chern Character)**.** Given an object $E \in \mathcal{D}^b(X)$ and $B \in \mathrm{NS}_{\mathbb{R}}(X)$, the **twisted Chern character** for E is given by:

$$\begin{aligned}\mathrm{ch}_0^B(E) &\coloneqq \mathrm{ch}_0(E)\\ \mathrm{ch}_1^B(E) &\coloneqq \mathrm{ch}_1(E) - B\,\mathrm{ch}_0(E)\\ \mathrm{ch}_2^B(E) &\coloneqq \mathrm{ch}_2(E) - B\,\mathrm{ch}_1(E) + \frac{B^2}{2}\,\mathrm{ch}_0(E).\end{aligned}$$

In the case when there is a chosen fixed $\ell \in \mathrm{NS}_{\mathbb{R}}(X)$ which generates the whole group, then we extend this notation to allow real valued superscripts $\beta \in \mathbb{R}$:

$$\begin{aligned}\mathrm{ch}_0^\beta(E) &\coloneqq r\\ \mathrm{ch}_1^\beta(E) &\coloneqq c - \beta r\\ \mathrm{ch}_2^\beta(E) &\coloneqq d - \beta c + \frac{\beta^2}{2} r,\end{aligned}$$

where $\mathrm{ch}(E) = (r, c\ell, d\ell^2)$. This extension is related to the previous notation by the equality $\mathrm{ch}_i^{\beta\ell}(E) = \mathrm{ch}_i^\beta(E)\ell^i$.

Remark. The latter notation only involves expressions with real numbers, so will be favoured when considering Picard rank 1 surfaces.

Remark. If there exists a line bundle L such that $\mathrm{ch}(L) = \exp(B)$, then we have

$$\mathrm{ch}^{nB}(E) = \mathrm{ch}(E \otimes L^{-n}).$$

Expressing the central charge from parametrisation A (Definition 1.23) in terms of these twisted Chern characters gives:

$$\mathcal{Z}^A_{\alpha,\beta}(E) = \left[\left(\frac{\alpha^2}{2}\,\mathrm{ch}_0(E) - \mathrm{ch}_2^\beta(E)\right) + i\alpha\,\mathrm{ch}_1^\beta(E)\right]\ell^2.$$

Remark. The α in the imaginary component could have been removed. This would be equivalent to applying $\mathrm{stretch}_{\alpha^{-1}}$, from Definition 1.18, to the stability condition), which does not change the heart or the semi-stable objects. The corresponding hearts of the stability conditions would be unchanged and Proposition 1.9 would still hold for this new family, apart from the final statement. For an object $E \in \mathcal{B}^\beta$, we can view $\mathrm{ch}^\beta_{\geq 1}(E)$ being similar to the Chern character for coherent sheaves on a curve, but for objects in $\mathcal{B}^\beta$, in the sense that it is additive, and $\mathrm{ch}_1^B(E) \geq 0$ (just as coherent sheaves have non-negative rank).

1.4.1 Characteristic Curves of Stability Conditions Associated to Chern Characters

We can draw 2 characteristic curves for any given Chern character v with $\Delta(v) \geq 0$ and positive rank. These are given by the real or imaginary components of $\mathcal{Z}_{\alpha,\beta}(v)$ being 0.

Definition 1.28 (Characteristic Curves V_v and Θ_v)**.** Given a Chern character v, with positive rank and $\Delta(v) \geq 0$, we define two characteristic curves on the (α, β)-plane:

$$\begin{aligned} V_v&: \ \mathfrak{Im}(\mathcal{Z}_{\alpha,\beta}(v)) = 0 \\ \Theta_v&: \ \mathfrak{Re}(\mathcal{Z}_{\alpha,\beta}(v)) = 0. \end{aligned}$$

1.4.2 Geometry of the Characteristic Curves

These characteristic curves for a Chern character v with $\Delta(v) \geq 0$ are not affected by flipping the sign of v so it is only necessary to consider non-negative rank. As discussed in Section 1.4.4, making this choice has Gieseker stable coherent sheaves appearing in the heart of the stability condition $\mathcal{B}^\beta$ as we move 'left' (decreasing β).

Positive Rank Case

The following facts can be deduced from the formulae for $\mathcal{Z}_{\alpha,\beta}(v)$ with the restrictions on v.

Lemma 1.10 (Geometry of Characteristic Curves in Positive Rank Case)**.** *Let $v \in H^{\text{even}}_{\text{alg}}(X)$ be a Chern character with positive rank and $\Delta(v) \geq 0$:*

- *V_v is a vertical line at $\beta = \mu(v)$*
- *Θ_v is a hyperbola with asymptotes angled at $\pm\pi/4$ rad crossing where V_v meets the β-axis: $(\mu^\ell(v), 0)$*
- *Θ_v is oriented with left-right branches (as opposed to up-down). The left branch shall be labelled Θ_v^- and the right Θ_v^+.*
- *The gap along the β-axis between either branch of Θ_v and V_v is $\sqrt{\Delta^{\ell^2}(v)}/\mathrm{ch}_0(v)$.*
- *When $\Delta(v) = 0$, Θ_v degenerates into a pair of lines, but the labels $\Theta_v^\pm$ will still be used for convenience.*

These are illustrated in Figure 1.1.

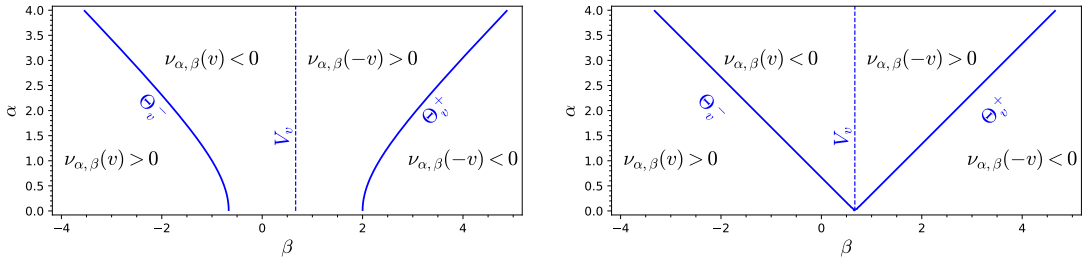
Proof. Take $v = (r, c\ell, d\ell^2)$ with $r > 0$ and $c^2 \geq 2rd$. Equation (1.1) gives:

$$\mathcal{Z}_{\alpha,\beta}(v) = \left[\left(c\beta + \frac{r}{2}(\alpha^2 - \beta^2) - d\right) + i\alpha\,(c - r\beta)\right]\ell^2.$$

Hence $(\beta, \alpha) \in V_v$ if and only if $\beta = \frac{c}{r} = \mu^\ell(v)$. So then $(\beta, \alpha) \in \Theta_v$ if and only if

$$\begin{aligned} \frac{2d}{r} &= 2\beta\frac{c}{r} + \alpha^2 - \beta^2 \\ &= \alpha^2 - \left(\beta - \frac{c}{r}\right)^2 + \frac{c^2}{r^2}, \\ \text{so} \qquad -\frac{c^2 - 2cd}{r^2} &= \alpha^2 - \left(\beta - \frac{c}{r}\right)^2, \\ \text{i.e.} \qquad -\frac{\Delta^{\ell^2}(v)}{r^2} &= \alpha^2 - (\beta - \mu^\ell(v))^2. \end{aligned}$$

We can recognise this as the equation of a parabola with the required properties in the case where $\Delta(v) > 0$, degenerating to a pair of lines (with required properties) when $\Delta(v) = 0$. □



(a) $\Delta(v) > 0$ (b) $\Delta(v) = 0$: hyperbola collapses

Figure 1.1: Characteristic curves ($\mathrm{ch}_i^{\alpha,\beta}(v) = 0$) of stability conditions associated to Chern characters v with $\Delta(v) \geq 0$ and positive rank.

Definition 1.29 ($\beta_\pm$). Given a formal Chern character v with positive rank, we define $\beta_\pm(v)$ to be the β-coordinate of where $\Theta_v^\pm$ meets the β-axis:

$$\beta_\pm(R, C\ell, D\ell^2) = \frac{C \pm \sqrt{C^2 - 2RD}}{R}$$

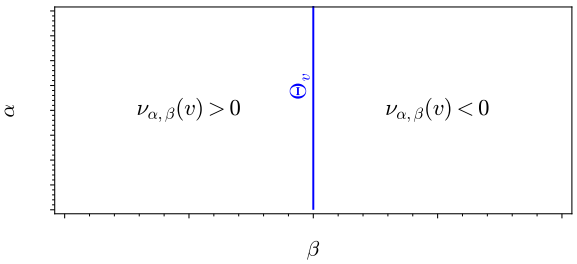
In particular, this means $\beta_\pm(v)$ are the two roots of the quadratic equation $\mathrm{ch}_2^\beta(v) = 0$. This definition will be extended to the rank 0 case in Definition 1.30.

Rank Zero Case

Similarly to Lemma 1.10, we clarify the shape of the characteristic curves in the special case when considering Chern characters of rank 0.

Lemma 1.11 (Geometry of Characteristic Curves in Rank 0 Case). *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(X)$ be a Chern character with positive rank, $\mathrm{ch}_1(v) \cdot \ell > 0$ and $\Delta(v) = 0$:*

- $V_v = \emptyset$.
- Θ_v *is a vertical line at* $\beta = \frac{D}{C}$ *where* $v = \left(0, C\ell, D\ell^2\right)$.



Proof. In the case of rank 0, the central charge simplifies to the following, from which the results can be read off: $\mathcal{Z}_{\alpha,\beta}((0, C\ell, D\ell^2)) = [(C\beta - D) + i\alpha C]\,\ell^2$. □

We can view the characteristic curves for $\left(0, C\ell, D\ell^2\right)$ with $C > 0$ as the limiting behaviour of those of $\left(\varepsilon, C\ell, D\ell^2\right)$. Indeed:

$$\mu\left(\varepsilon, C\ell, D\ell^2\right) = \frac{C}{\varepsilon} \longrightarrow +\infty$$
$$\text{as } 0 < \varepsilon \longrightarrow 0.$$

So we can view V_v as moving off infinitely to the right, with Θ_v^+ even further. As for Θ_v^-,

$$\beta_-\left(\varepsilon, C\ell, D\ell^2\right) = \frac{C - \sqrt{C^2 - 2D\varepsilon}}{\varepsilon} \longrightarrow \frac{D}{C}$$
$$\text{as } 0 < \varepsilon \longrightarrow 0 \qquad \text{(via L'Hôpital),}$$

so we can view the hyperbola branch Θ_v^-, for v with negligeable rank, as approaching the vertical line at $\beta = \frac{D}{C}$ which Θ_v is for $v = (0, C\ell, D)$. For this reason, I will refer to the whole of Θ_v in the rank zero case as Θ_v^- to be able to use the same terminology in both positive rank and rank zero cases. With this in mind, let us extend Definition 1.29.

Definition 1.30 (Extending β_- to rank 0 case)**.** Given $v \in H^{\text{even}}_{\text{alg}}(X)$ with rank 0 and $\text{ch}_1(v) > 0$, we define $\beta_-(v)$ to be the β-coordinate of point where Θ_v meets the β-axis:

$$\beta_-(0, C\ell, D\ell^2) = \frac{D}{C}$$
$$\beta_+(0, C\ell, D\ell^2) = +\infty.$$

Remark. This defines the characteristic curves for all positive v with $\Delta(v) \geq 0$ and one of $\text{ch}_0(v)$ or $\text{ch}_1(v)$ being non-zero. Other positive Chern characters with $\Delta(v) \geq 0$ are of the form $(0, 0, n)$. The semistable objects of these Chern characters are skyscraper sheaves and extensions, which are semistable for all considered stability conditions.

1.4.3 Useful Identities

Here we establish some helper Lemmas to be referenced in calculations in other parts.

Lemma 1.12 (Comparison test with $\beta_\pm$)**.** *Let $\beta \in \mathbb{R}$ and $v \in H^{\text{even}}_{\text{alg}}(X)$ with $\Delta(v) \geq 0$ and $\text{ch}_0(v) \geq 0$ for some smooth projective surface X. Then we have the following:*

$$\text{ch}_2^\beta(v) < 0 \iff \beta \in (\beta_-, \beta_+)$$
$$\text{ch}_2^\beta(v) = 0 \iff \beta = \beta_\pm.$$

In particular, if $\text{ch}^\beta(v) \geq 0$, we have

- $\beta \leq \mu^\ell(v) \Rightarrow \beta \leq \beta_-(v)$
- $\beta \geq \mu^\ell(v) \Rightarrow \beta \geq \beta_+(v)$,

as well as the corresponding statements with strict equalities.

Proof. In the case $\text{ch}_0(v) > 0$, as noted in Definition 1.29, we have that $\beta_\pm(v)$ are the roots of the quadratic $\text{ch}_2^\beta(v)$ in β. Given that this quadratic has $\frac{\text{ch}_0(v)}{2} > 0$ as β^2 coefficient, the statement of the lemma follows from standard parabola properties.

In the case $\text{ch}_0(v) = 0$, we have $\text{ch}_2^\beta((0, C\ell, D\ell^2)) = D - \beta C$ which is negative if and only if $\beta \in (\frac{D}{C}, +\infty) = (\beta_-(0, C\ell, D\ell^2, \beta_+(0, C\ell, D\ell^2))$. In this case $\beta = \beta_+(v) \Rightarrow \text{ch}_2^\beta(v) < 0$ is vacuously true given that $\beta_+(v) = +\infty$.

□

Lemma 1.13 (Twisted Chern character at β_-)**.** *Consider a Chern character $v \in H^{\text{even}}_{\text{alg}}(X)$, of positive rank and $\Delta(v) \geq 0$. Then the twisted Chern character at $\beta_-(v)$ is*

$$\text{ch}^{\beta_-(v)}(v) = \left(\text{ch}_0(v), \sqrt{\Delta^{\ell^2}(v)}\ell, 0\right).$$

Proof. We have $\mathrm{ch}_0^\beta(v) = \mathrm{ch}_0(v)$ by definition. Definition 1.29 commented on $\mathrm{ch}_2^{\beta_-(v)}(v) = 0$. Take $v = (R, C\ell, D\ell^2)$.

For $R > 0$, we have

$$\beta_- = \frac{C - \sqrt{\Delta^{\ell^2}(v)}}{R},$$
$$\text{and } \sqrt{\Delta^{\ell^2}(v)} = C - R\beta_- = \mathrm{ch}_1^{\beta_-}(v).$$

For $R = 0$, we have $\mathrm{ch}_1^{\beta_-}(v) = \mathrm{ch}_1(v)$ by definition, which is equal to $\sqrt{C^2}$, since $C > 0$ due to v being positive. □

1.4.4 Relevance of Characteristic Curves to Finding Walls

Relevance of V_v

For the positive rank case, by definition of the first tilt $\mathcal{B}^\beta$, objects of Chern character v can only be in $\mathcal{B}^\beta$ on the left of V_v, where $\mathfrak{Im}(\mathcal{Z}_{\alpha,\beta}(v)) > 0$, and objects of Chern character $-v$ can only be in $\mathcal{B}^\beta$ on the right, where $\mathrm{ch}_1^{\alpha,\beta}(-v) > 0$. Because of this, when using these characteristic curves, only positive ranks are considered, as negative rank objects are implicitly considered on the right hand side of V_v. Note that shifts of semistable objects in the heart are technically also semistable objects by the definition of a Bridgeland stability condition, however the shifts are not in the heart themselves. Hence, they cannot destabilise objects in our heart. This is why we shall make sure to keep distinguishing stable objects of Chern characters v and $-v$.

In the rank zero case, this still applies if we consider V_v to be 'infinitely to the right' ($\mu(v) = +\infty$). In particular, any coherent sheaf E with $\mathrm{ch}_0(E) = 0$ and $\mathrm{ch}_1(E) > 0$ is contained in $\mathcal{B}^\beta$ for all β.

Relevance of Θ_v

Since $\mathfrak{Re}(\mathcal{Z}_{\alpha,\beta}(v))$ is the numerator of the tilt slope $\nu_{\alpha,\beta}$. The curve Θ_v, where this is 0, firstly divides the $(\alpha$-$\beta)$-half-plane into regions where the signs of tilt slopes of objects of Chern character v (or $-v$) are fixed. Secondly, it gives more of a fixed target for some E to be a semistabiliser of F, in the following sense: if (α, β), is on $\Theta_{\mathrm{ch}(F)}$, then E can only be a semistabiliser of F if $\nu_{\alpha,\beta}(E) = 0$, and hence $\mathfrak{Re}(\mathcal{Z}_{\alpha,\beta}(\mathrm{ch}(E))) = 0$. In fact, this allows us to use the characteristic curves of some $\mathrm{ch}(E)$ and $\mathrm{ch}(F)$ (with $\Delta(E), \Delta(F) \geq 0$ and positive ranks) to determine the location of the wall where E could possibly semistabilise F. This is done by finding the intersection of $\Theta_{\mathrm{ch}(E)}$ and $\Theta_{\mathrm{ch}(F)}$, the point (β, α) where $\nu_{\alpha,\beta}(\mathrm{ch}(E)) = \nu_{\alpha,\beta}(\mathrm{ch}(F)) = 0$, and a potential wall point on $\Theta_{\mathrm{ch}(F)}$, and hence (potentially) the apex of the circular wall with centre $(\beta, 0)$, as per the following Theorem 1.14.

> **Theorem 1.14** (Bertram's Nested Wall Theorem). *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathcal{S})$ satisfy $\Delta(v) \geq 0$. Then the intersections of walls for v with the slice of stability conditions $\{\sigma^A_{\alpha,\beta}\}$, form nested, non-intersecting, semicircular walls centred on the β-axis and bounded above in size. There is also possibly a single vertical wall along V_v.*
>
> *Furthermore, the 'apex' of each semicircular wall (point with maximal α-coordinate) lies on Θ_v.*

Remark. Given the known properties of the characteristic curves (Θ_v being a hyperbola with asymptotes intersection the β-axis at $\pm\pi/4$ rad), one can deduce that the walls are nested and non-intersecting if we are given only the facts about their centre and apex.

Corollary 1.15. *Given $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathcal{S})$ with $\Delta(v) \geq 0$, $\mathcal{Z}^A_{\alpha,\beta}$ lies on a wall if and only if there exists $F \in \mathcal{B}^\beta$ with a non-zero stable sub-object E with $\nu_{\alpha,\beta}(E) = \nu_{\alpha,\beta}(F)$ and $\mathrm{ch}(E) \notin \mathbb{Q}v$.*

Given *Bertram's Nested Wall Theorem*, we know that there is at least one unbounded chamber outside the semi-circular walls. In particular, if V_v is a wall, then there is one unbounded chamber on each side. Otherwise there is a single unbounded chamber.

The next theorem allows us to relate the semistable objects with respect to the stability conditions in the unbounded chambers with more classical Gieseker semistable objects. Within the bounded chambers, there will be no guarantee that the semistable objects are even sheaves, that is, derived objects with non-zero cohomology only in degree 0.

Theorem 1.16 (Left Limit stability)**.** *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathcal{S})$ satisfy $\Delta(v) \geq 0$, and take $\sigma^A_{\alpha,\beta}$, to be a stability condition in the unbounded chamber to the left of V_v ($\beta < \mu^\ell(v)$). Then the semistable objects with respect to $\sigma^A_{\alpha,\beta}$ inside the corresponding heart are precisely Gieseker semistable sheaves of Chern character v.*

Proof. [Bri08, Proposition 14.2] □

Chapter 2

Fourier-Mukai Transforms

Shigeru Mukai first introduced a "Fourier functor" in [Muk81], which is now known as the Fourier-Mukai transform. The Fourier transform in analysis gives a non-trivial isometry between $L^2(V)$ and $L^2(V^*)$, used for many purposes including transforming differential equation problems to an alternative, possibly easier, one. Similarly, the Fourier-Mukai transform in its original form, gives a non-trivial equivalence between the derived category of an abelian variety X and that of its dual $\hat{X}$, which also induces an isometry on the Mukai lattices. But also, as will be seen in Section 2.4, Fourier-Mukai transforms map objects that are stable with respect to one stability condition, to objects that are stable with respect to another stability condition. In Part III, this will be used to equate the existence of some Bridgeland stable objects and short exact sequences of a given form to a corresponding question for Gieseker stable sheaves, in order to prove the existence of certain walls.

2.1 The Functor

Definition 2.1 (Fourier-Mukai Transform)**.** An **integral functor** $X \to Y$ is given by a **kernel** $K \in \mathcal{D}^b(X \times Y)$ and is defined as

$$\begin{aligned} \Phi_K^{X\to Y} &: \mathcal{D}^b(X) \to \mathcal{D}^b(Y) \\ \Phi_K(E) &= Rp_{2*}\left(K \overset{L}{\otimes} {p_1}^*(E)\right), \end{aligned}$$

$$\text{where } p_i \text{ are the projections} \qquad X \xleftarrow{p_1} X \times Y \xrightarrow{p_2} Y.$$

If an **integral functor** gives an equivalence of categories, then it is called a **Fourier-Mukai Transform**.

Example 2.1 (Original Fourier-Mukai Transform)**.** *Let X be an abelian surface and $\hat{X}$ be its dual variety, and $\mathcal{P}$ be the Picard sheaf on $X \times \hat{X}$, then $\Phi_{\mathcal{P}}$ is a Fourier-Mukai transform. Morally, the Picard sheaf $\mathcal{P}$ encodes a way that $\hat{X}$ parametrises $\mathrm{Pic}^0(X)$ and vice-versa (so we write $\mathcal{P}_{\hat{x}}$ for the line bundle on X, that $\hat{x} \in \hat{X}$ represents, and vice versa). This property*

of the kernel induces these identities on the functor:

$$\begin{aligned} \Phi_{\mathcal{P}}(\mathcal{O}_x) &= \mathcal{P}_x \\ \Phi_{\mathcal{P}}(\mathcal{P}_{\hat{x}}) &= \mathcal{O}_{\hat{x}}[-\dim(X)] \\ \textit{in particular,}\quad \Phi_{\mathcal{P}}(\mathcal{O}_X) &= \mathcal{O}_0[-\dim(X)] \end{aligned}$$

A parallel to the Fourier transform in analysis can be seen here, with the Fourier transform which maps periodic functions to distributions concentrated at a single point corresponding to the frequency. Here, elements of $\mathrm{Pic}^0(X)$ are mapped to sheaves supported at a single point corresponding to that element. This example can be studied more in the original paper [Muk81] or in the standard book on the subject [Huy06b, Chapter 9, Section 2].

2.2 Isometries of the Mukai Lattice

Definition 2.2 (Mukai Lattice)**.** Although differing in other settings, on a principally polarised abelian surface $\mathbb{T}$ with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, the **Mukai lattice** is identified with

$$H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \cong \mathbb{Z} \oplus \mathbb{Z}\ell \oplus \mathbb{Z},$$

on which we also associate a bilinear form $\Delta(\cdot,\cdot)$ defined by:

$$\Delta(v,w) = \mathrm{ch}_1(v)\,\mathrm{ch}_1(w) - \mathrm{ch}_2(v)\,\mathrm{ch}_0(w) - \mathrm{ch}_0(v)\,\mathrm{ch}_2(w).$$

Explicitly with coordinates, this is given by

$$\Delta\left((r_1, c_1\ell, d_1), (r_2, c_2\ell, d_2)\right) = 2c_1c_2 - r_1d_2 - r_2d_1.$$

The following adjectives are used for an element $v = (r, c\ell, d) \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$:

- **Irreducible** when $\gcd(r,c,d) = 1$.
- **Positive** in one of the following cases:
  - $r > 0$
  - $r = 0$ and $c > 0$
  - $r = 0$, $c = 0$ and $d > 0$.
- **Isotropic** when $\Delta(v) \coloneqq \Delta(v,v) = 0$.

Remark. This bilinear form is sometimes denoted by $\langle\cdot,\cdot\rangle$ instead. However in this setting on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, $\langle v, v\rangle = \Delta(v)$ so we will recycle the notation for the Bogomolov-Gieseker form.

Notice that $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ that are positive, isotropic and irreducible are of the form $(a^2, ab\ell, b^2)$ for some coprime integers a, b. This leads us to the next definition:

Definition 2.3 (Identification of primitive positive isotropic Chern characters with a subset of $\mathbb{Z}^2/\{\pm 1\}$)**.** Let $\mathcal{C}$ be the irreducible, positive, isotropic Chern characters. We define the following injection Γ:

$$\Gamma\colon \mathcal{C} \xrightarrow{\sim} S := \left\{ \begin{pmatrix} a \\ b \end{pmatrix} : a, b \in \mathbb{Z},\ (a,b) = 1 \right\} / \{\pm 1\}$$
$$(a^2, ab\ell, b^2) \mapsto \left[\begin{pmatrix} a \\ b \end{pmatrix} \right].$$

Remark. The bilinear form of the Mukai lattice restricted to $\mathcal{C}$ corresponds to the amplitude squared of the cross product in $\mathbb{Z}/\{\pm 1\}$, that is:

$$\Delta(w_1, w_2) = (ad - bc)^2, \quad \text{where} \quad \Gamma(w_1) = \left[\begin{pmatrix} a \\ b \end{pmatrix} \right] \quad \text{and} \quad \Gamma(w_1) = \left[\begin{pmatrix} c \\ d \end{pmatrix} \right].$$

Remark. With this identification, comparing Mumford slopes of positive, principle, isotropic Chern characters is equivalent to comparing some angles in $\mathbb{Z}^2$ for certain choices of representatives.

Here we describe the isometries of the Mukai lattice as a group. Culminating in the following proposition.

Proposition 2.1 (Isometries of the Mukai Lattice)**.** *Let $\mathbb{T}$ be a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, then the group of isometries on the Mukai lattice $H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$, that is the group of $\mathbb{Z}$-linear bijections preserving the bilinear form $\Delta(\cdot,\cdot)$, is isomorphic to*

$$\mathrm{PGL}(2,\mathbb{Z}) \times \{\pm 1\},$$

where the action is given by (γ given by Definition 2.4):

$$([g], \pm 1) \cdot v = \pm \gamma^{-1}\left(g\gamma(v)g^T\right).$$

Proof. Deferred to the end of this section. □

Remark. This group contains a subgroup $\mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\}$ which could be interpreted as 'orientation preserving' isometries, to be illustrated further in the lemmas to come. The isometry induced by the derived dual, for example, corresponds to the element $\left(\left[\left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)\right], +1\right)$ which is not in this subgroup. We will see later that this subgroup is given by Fourier-Mukai transforms.

Definition 2.4 (Identification of Chern characters/Mukai Lattice with symmetric integer matrices)**.** We define the bijection γ as follows representing Chern characters by symmetric integer matrices:

$$\gamma\colon H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \xrightarrow{\sim} \mathrm{Symm}(2,\mathbb{Z})$$
$$(r, c\ell, d) \mapsto \begin{pmatrix} r & c \\ c & d \end{pmatrix}.$$

The $\mathbb{Z}$-linearity of the isometries of the Mukai lattice implies that they are determined entirely by their restrictions to the set of isotropic, irreducible Chern characters. The next lemma pushes this a little further but will also explain the $\{\pm 1\}$ term in the group product expression in Proposition 2.1.

Lemma 2.2 (Restriction of isometries of $\mathcal{C}$). *For any isometry of the Mukai lattice ψ, either ψ fixes $\mathcal{C}$ (positive, isotropic, irreducible Chern characters), or $-\psi$ does (defined by $v \mapsto -\psi(v)$).*

In particular, the isometries G of the Mukai lattice is isomorphic to

$$\begin{aligned} G^0 \times \{\pm 1\} &\cong G \\ (\psi, \pm 1) &\mapsto \pm\psi, \end{aligned}$$

where G^0 is the subgroup of isometries that fix $\mathcal{C}$.

Proof. The isometry ψ certainly fixes isotropic and irreducible Chern characters. However, that set contains both positive and non-positive elements, so it remains to show that ψ maps all positive ones to only positive ones, or to only non-positive ones.

The key observation is that for any two distinct positive, isotropic, irreducible $v_1 = ({a_1}^2, a_1 b_1 \ell, {b_1}^2)$ and $v_2 = ({a_2}^2, a_2 b_2 \ell, {b_2}^2)$, we have $\Delta(v_1, v_2) = (a_1 b_2 - a_2 b_1)^2$ is a strictly positive square integer. So by $\mathbb{Z}$-linearity of Δ, for any two distinct isotropic and irreducible v_1, v_2 (not necessarily positive), the sign of $\Delta(v_1, v_2)$ determines whether the v_i have the same sign (both positive or both not). Therefore, ψ preserving Δ implies that it maps $\mathcal{C}$ to either $\mathcal{C}$ or $-\mathcal{C}$ as required.

Finally, for the group isomorphism, G^0 is indeed a subgroup of G. The earlier part of the proof shows that G^0 with $\{\pm \mathrm{Id}\}$ generates the whole of G. But since $-\mathrm{Id}$ commutes with any $\mathbb{Z}$-linear function these two subgroups form an inner direct product of G, giving the required isomorphism in the statement of the lemma.

□

Lemma 2.3 (Determining Elements of S). *Consider the set S from Definition 2.3 along with three of its elements:*

$$S = \left\{ \begin{pmatrix} a \\ b \end{pmatrix} : a, b \in \mathbb{Z}, \gcd(a, b) = 1 \right\} / \{\pm 1\}$$

$$e_1 \coloneqq \left[\begin{pmatrix} 1 \\ 0 \end{pmatrix} \right], e_2 \coloneqq \left[\begin{pmatrix} 0 \\ 1 \end{pmatrix} \right], e_3 \coloneqq \left[\begin{pmatrix} 1 \\ 1 \end{pmatrix} \right] \in S.$$

Any element $u \in S$ is determined up to two possibilities by the values $\Delta(u, e_1)$ and $\Delta(u, e_2)$ (or one possibility in edge cases). In addition, the extra value $\Delta(u, e_3)$ determines u uniquely.

Proof. Take a generic $u = \left[\begin{pmatrix} a \\ b \end{pmatrix} \right] \in S$. $\Delta(u, e_1) = b^2$, $\Delta(u, e_2) = a^2$ so a and b can be recovered from these values up to a combination of signs. In particular, the only other element of S with the same corresponding values is $\left[\begin{pmatrix} -a \\ b \end{pmatrix} \right]$. If $a = 0$ or $b = 0$, then in fact these two elements of S are equal and so u is uniquely determined. Otherwise, taking e_3 into consideration:

$$\begin{aligned} \Delta\left(\left[\begin{pmatrix} a \\ b \end{pmatrix} \right], e_3 \right) &= (b - a)^2, \\ \Delta\left(\left[\begin{pmatrix} -a \\ b \end{pmatrix} \right], e_3 \right) &= (b + a)^2. \end{aligned}$$

These two values are distinct so $\Delta(u, e_i)$ for $i = 1, 2, 3$ determines u uniquely.

□

Lemma 2.4 (Isometries of S). *The bijections of S (from Definition 2.3) which preserve the bilinear form Δ are given by $\mathrm{PGL}(2,\mathbb{Z})$, where the corresponding action is given by matrix multiplication $[g]\cdot[u]=[gu]$.*

Proof. Let ψ be such a bijection on S and take e_i, $i=1,2,3$ as in Lemma 2.3.

$$\text{Take} \qquad \left[\begin{pmatrix}a\\c\end{pmatrix}\right] \coloneqq \psi(e_1) \quad \text{and} \quad \left[\begin{pmatrix}b\\d\end{pmatrix}\right] \coloneqq \psi(e_2).$$

Since ψ preserves Δ, the values $\Delta(\psi(e_3),\psi(e_1))=1$ and $\Delta(\psi(e_3),\psi(e_2))=1$ are determined. So by Lemma 2.3, there is at most one other $u\in S$ satisfying $\Delta(u,e_1)=\Delta(u,e_2)=1$. Here are two such elements:

$$\begin{aligned}\Delta\left(\psi(e_1),\begin{pmatrix}a\pm b\\c\pm d\end{pmatrix}\right) &= (a(c\pm d)-c(a\pm b))^2\\ &= (ad-cb)^2=\Delta(\psi(e_1),\psi(e_2))=1,\\ \Delta\left(\psi(e_2),\begin{pmatrix}a\pm b\\c\pm d\end{pmatrix}\right) &= (b(c\pm d)-d(a\pm b))^2\\ &= (ad-cb)^2=\Delta(\psi(e_1),\psi(e_2))=1.\end{aligned}$$

Therefore, we must have $\psi(e_3)=\left[\begin{pmatrix}a\pm b\\c\pm d\end{pmatrix}\right]$. Without loss of generality, use the plus signs, switching the choice of representative for $\psi(e_2)$ if necessary. Define the matrix $A_\psi\coloneqq\begin{pmatrix}a&b\\c&d\end{pmatrix}$, so that $\psi(e_i)=[Ae_i]$ for $i=1,2,3$. It remains to show that the same is true for other elements of S. This follows from noting that $[u]\mapsto[Au]$ is a bijection on S preserving Δ. So $\Delta([Au],\psi(e_i))=\Delta(\psi([u],\psi(e_i)))$ for $i=1,2,3$. Lemma 2.3 will also hold when replacing e_i by $\psi(e_i)$ by isometry, so we can conclude $[Au]=\psi(u)$.

Strictly, at this point we have a group surjection from $\mathrm{GL}(2,\mathbb{Z})$ onto the bijections of S preserving Δ. This surjection given by $A\mapsto([u]\mapsto[Au])$. Considering the kernel of this surjection finally concludes that $\mathrm{PGL}(2,\mathbb{Z})$ is isomorphic to that latter group.

□

Finally, we have the tools to consider the main proposition of this section.

Proof of Proposition 2.1. By Lemma 2.2, the group of isometries G of the Mukai lattice is isomorphic to $G^0\times\{\pm1\}$ where G^0 is the subgroup of isometries which fix $\mathcal{C}$ (positive, irreducible, isotropic Chern characters). The elements of G^0 fix $\mathcal{C}$ so restrict to bijections of $\mathcal{C}$ which preserve Δ. Lemma 2.4, gives that these bijections of $\mathcal{C}$ are given by $\mathrm{PGL}(2,\mathbb{Z})$ as described. Conversely, $\mathcal{C}$ spans the Mukai lattice over $\mathbb{Z}$, so these restrictions determine the isometries of the Mukai lattice uniquely. Therefore G^0 is a subgroup of $\mathrm{PGL}(2,\mathbb{Z})$, but in fact they are equal because any of these bijections of $\mathcal{C}$ can be lifted to isometries of the Mukai lattice, as will be shown next.

Take any $[g]\in\mathrm{PGL}(2,\mathbb{Z})$, which represents a bijection ψ of $\mathcal{C}$ preserving Δ. Notice that for any $v\in\mathcal{C}$, we have:[1]

$$\Gamma(v)\Gamma(v)^T=\begin{pmatrix}a\\b\end{pmatrix}\begin{pmatrix}a&b\end{pmatrix}=\begin{pmatrix}a^2&ab\\ab&b^2\end{pmatrix}=\gamma(v).$$

[1] Strictly, we take a representative of $\Gamma(v)$ in this expression, however this is well defined because the minus signs cancel by multiplication.

So we can express the action of $\mathrm{PGL}(2,\mathbb{Z})$ on any $v \in \gamma(\mathcal{C})$ via the identification with $\mathrm{Symm}(2,\mathbb{Z})$ instead $\mathbb{Z}^2/\{\pm I_2\}$:

$$\begin{aligned}\gamma(\psi(v)) &= \Gamma(\psi(v))\Gamma(\psi(v))^T \\ &= (g\Gamma(v))(g\Gamma(v))^T \\ &= g\gamma(v)g^T \\ \psi(v) &= \gamma^{-1}(g\gamma(v)g^T).\end{aligned}$$

This formula for ψ extends linearly (over $\mathbb{Z}$) to the whole Mukai lattice as written, and indeed preserves the bilinear form Δ.

This identifies G^0 as $\mathrm{PGL}(2,\mathbb{Z})$ and describes the corresponding action on the Mukai lattice. Finally this gives the action of $G \cong G^0 \times \{\pm 1\}$ as the one in the statement of the lemma.

□

2.3 Cohomological Fourier-Mukai Transform

Typically, $\mathrm{SL}(2,\mathbb{Z})$ is considered to be represented by some Fourier-Mukai transforms modulo shifts, as covered in [Huy06b, Chapter 9, Section 3]. However, Chern characters are not quite respected by shifts (there is a sign change). Due to this, we shall be considering a slight adjustment to the typical 'cohomological Fourier-Mukai transform', where we consider the action on Chern characters, modulo 2 shifts.

Theorem 2.5 (Cohomological Fourier-Mukai transforms)**.** *Let $\mathbb{T}$ be a principally polarised abelian complex surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, and Φ be a Fourier-Mukai transform. Then it descends to a map Chern characters as specified in the diagram below. In particular, it induces an isomorphism on the Mukai Lattice, called the corresponding* cohomological Fourier-Mukai transform (CFMT) *Φ^H.*

$$\begin{array}{ccc} \mathcal{D}^b(\mathbb{T}) & \xrightarrow{\Phi} & \mathcal{D}^b(\mathbb{T}) \\ \downarrow{\scriptstyle v} & & \downarrow{\scriptstyle v} \\ H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) & \xrightarrow{\Phi^H} & H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \end{array}$$

Furthermore, the cohomological Fourier-Mukai transforms *Φ^H form a normal subgroup of the isometries of the Mukai lattice:*

$$\left\{\Phi^H\right\} \cong \mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\} \lhd \mathrm{PGL}(2,\mathbb{Z}) \times \{\pm 1\} \cong \{H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \text{ isometry}\},$$

with the action on $H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ given by

$$([g], \pm 1) \cdot v = \pm\gamma^{-1}\left(g\gamma(v)g^T\right),$$

where operations on the right hand side are matrix multiplications.

Finally, if $\Phi^H = ([g], +1)$, then Φ^H preserves positive, irreducible, isotropic Chern characters and acts on their representations via Γ (from Definition 2.3) by plain matrix multiplication with g:

$$\Gamma(\Phi^H(w)) = \left[g\begin{pmatrix}a\\b\end{pmatrix}\right] \quad \text{where } w \in \mathcal{C},\ \Gamma(w) = \left[\begin{pmatrix}a\\b\end{pmatrix}\right].$$

Remark. The above group $\mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\}$ does not give all the isometries of the Mukai lattice. See Proposition 2.1 and its following remark.

Proof. The cohomological Fourier-Mukai transforms Φ^H are well defined and induce isometries of the Mukai lattice from [Huy06b, Chapter 5, Section 2]. These isometries being identified in Proposition 2.1, they form a subgroup of $\mathrm{PGL}(2,\mathbb{Z}) \times \{\pm 1\}$ acting on $H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ and $\mathcal{C}$ as described in the statement of the theorem.

This subgroup contains the elements

$$(\otimes L)^H = \left(\left[\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}\right], +1\right), \quad \Phi^H_{\mathcal{P}} = \left(\left[\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}\right], +1\right), \quad \text{and } ([1])^H = \left(\left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right], -1\right),$$

which generate $\mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\}$. Conversely the fact that this group is contained in $\mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\}$ is the equivalent of [HMS08, Theorem 2] or [YY09, Lemma 6.18].

□

Lemma 2.6. *Suppose $E \in \mathrm{Coh}(\mathbb{T} \times \mathbb{T})$ gives rise to a Fourier-Mukai transform. Then the corresponding element of $\{\pm 1\}$ in $\Phi^H_{E[n]}$ is given by the parity of n.*

More precisely:

$$p_2(\Phi^H_{E[n]}) = (-1)^n,$$

where

$$\begin{aligned} p_2 \colon \mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\} &\twoheadrightarrow \{\pm 1\} \\ ([g], \pm 1) &\mapsto \pm 1. \end{aligned}$$

Remark. Any Fourier-Mukai transform on abelian varieties has one of these considered forms [Huy06b, Proposition 9.53].

Proof. Considering Lemma 2.2, which was used in the construction of the isometries of the Mukai lattice given in Proposition 2.1, we have

$$p_2(\Phi^H_{E[n]}) = \begin{cases} 1 & \text{if } \Phi^H_{E[n]} \text{ fixes } \mathcal{C} \\ -1 & \text{if } \Phi^H_{E[n]} \text{ maps } \mathcal{C} \text{ to } -\mathcal{C} \end{cases}$$

where $\mathcal{C}$ (from Definition 2.3) is the set of positive irreducible isotropic Chern characters. That lemma also gives that those two possibilities are exhaustive.

An easy test for these possibilities is to consider the image of $\mathcal{O}_x$ since $\mathrm{ch}(\mathcal{O}_x) \in \mathcal{C}$, which is a sheaf [Huy06b, Example 5.7 vi)]. So $\Phi_{E[n]}(\mathcal{O}_x)$ is a sheaf shifted n times, and so is positive if and only if n is even giving the formula as expressed in the statement of the lemma.

□

2.4 Action on Stabilities

The action of Fourier-Mukai transforms on stability conditions has been studied before in [Huy06a, Corollary 5.3]), [Mea12a, Lemma 3.2.7]), to give a statement similar to Theorem 2.7 in this section. Here, we reprove this but leveraging the identification given by Γ from Definition 2.3 to simplify the calculation to potentially allow easier future adjustments of this method. More precisely, this identification Γ is generalised slightly with the following definition.

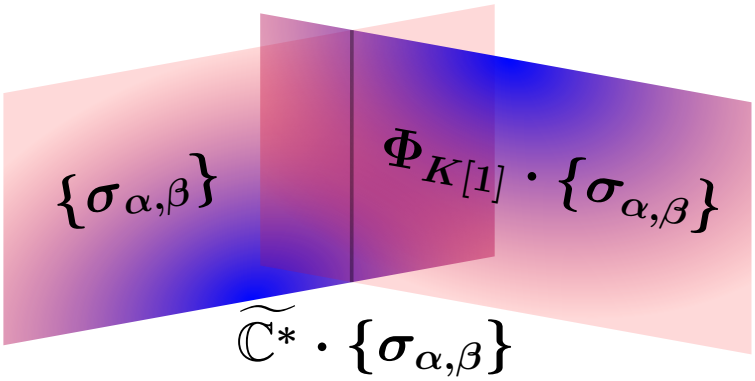


Figure 2.1: Visual aid for the action of a fixed FMT on slice of stability conditions within a larger space of stability conditions. Generally, only a single vertical line is mapped back onto the same slice.

Definition 2.5 (Extending $\boldsymbol{\Gamma}$)**.** Γ can be extended to a bijection

$$\begin{aligned}\Gamma\colon \left\{v \in H^{\text{even}}_{\text{alg}}(\mathbb{T}) \otimes_{\mathbb{Z}} \mathbb{C}\colon \Delta(v) = 0\right\} &\xrightarrow{\sim} \frac{\mathbb{C}^2 \setminus \{0\}}{\{\pm 1\}} \\ z\exp(w\ell) = (z, zw\ell, zw^2) &\mapsto \left[\begin{pmatrix}\sqrt{z}\\ \sqrt{z}w\end{pmatrix}\right] \\ a^2\exp\left(\frac{b}{b}\ell\right) = (a^2, ab\ell, b^2) &\mapsto \left[\begin{pmatrix}a\\ b\end{pmatrix}\right].\end{aligned}$$

Note that the action $([g], \pm 1)$ on $H^{\text{even}}_{\text{alg}}(\mathbb{T})$ extended linearly to $H^{\text{even}}_{\text{alg}}(\mathbb{T}) \otimes_{\mathbb{Z}} \mathbb{C}$ restricts to $\left\{v \in H^{\text{even}}_{\text{alg}}(\mathbb{T}) \otimes_{\mathbb{Z}} \mathbb{C}\colon \Delta(v) = 0\right\}$ in a way such that the corresponding action on $\frac{\mathbb{C}\setminus\{0\}}{\pm 1}$ is given by

$$([g], \pm 1) \cdot [u] = \left[\sqrt{\pm 1}gu\right].$$

Theorem 2.7 (FMT action on stabilities)**.** *Let X be a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, and $K \in \mathrm{Coh}(X \times X)$ be a kernel of a Fourier-Mukai transform. Take a, b, c, d as:*

$$\Phi_K^H = \left(\left[\begin{pmatrix}a & b\\ c & d\end{pmatrix}\right], +1\right) \in \mathrm{PSL}(2, \mathbb{Z}) \times \{\pm 1\}.$$

Then the action of autoequivalences of of $D^b(X)$ on the space of stability conditions satisfies the following identity:

$$\Phi_{K[1]} \cdot \sigma_{\alpha, \frac{-a}{b}} = \sigma_{\frac{1}{\alpha b^2}, \frac{d}{b}}.$$

Remark. Notice that the β-coordinates of these stability conditions are related to the image and pre-image of skyscraper sheaves:

$$\frac{-a}{b} = \mu(\Phi^{-1}_{K[1]}(\mathcal{O}_x)) \qquad \frac{d}{b} = \mu(\Phi_{K[1]}(\mathcal{O}_x)).$$

Indeed $\Phi^{-1}_{K[1]}(\mathcal{O}_x)$ has $+\infty$ tilt slope for the stability conditions $\sigma_{\alpha, \frac{-a}{b}}$, like skyscraper sheaves. Vice-versa $\Phi_{K[1]}(\mathcal{O}_x)$ has $+\infty$ tilt slope for the stability conditions $\sigma_{\alpha, \frac{d}{b}}$.

We can view part of the action of $\Phi_{K[1]}$ on the stability conditions as taking the stable objects of infinite $\nu_{\alpha,\frac{-a}{b}}$-slope with irreducible Chern characters (corresponding to 'minimal objects' in other sources [Huy06a, Proposition 2.2]), to the corresponding objects for $\sigma_{\frac{1}{\alpha b^2},\frac{d}{b}}$, sending skyscrapers to non-skyscrapers, and vice-versa.

$$\begin{array}{ccc} M(b^2,-ab,a^2)[1] & \cup & \{\mathcal{O}_x : x \in X\} \\ & \searrow \quad \swarrow & \\ & \Phi_{K[1]} & \\ & \swarrow \quad \searrow & \\ M(d^2,bd,d^2)[1] & \cup & \{\mathcal{O}_x : x \in X\} \end{array}$$

Proof. Following [Mea12a, Theorem 3.1.10] this amounts to the following calculation.

$$\begin{aligned} \Phi^H_{K[1]}\left(\exp\left(\left(-\frac{a}{b}+\alpha i\right)\ell\right)\right) &= \Gamma^{-1}\left(\left(\left[\begin{pmatrix} a & b \\ c & d\end{pmatrix}\right],-1\right)\cdot\left[\begin{pmatrix} 1 \\ -\frac{a}{b}+\alpha i\end{pmatrix}\right]\right) \\ &= \Gamma^{-1}\left(i\left[\begin{pmatrix} \alpha b i \\ c-\frac{ad}{b}+\alpha d i\end{pmatrix}\right]\right) \\ &= \Gamma^{-1}\left(\left[\begin{pmatrix} -\alpha b \\ -\frac{i}{b}-\alpha d\end{pmatrix}\right]\right) \\ &= \Gamma^{-1}\left(\left[\alpha b\begin{pmatrix} 1 \\ \frac{1}{\alpha b^2}i+\frac{d}{b}\end{pmatrix}\right]\right) \\ &= (\alpha b)^2\exp\left(\left(\frac{d}{b}+\frac{1}{\alpha b^2}i\right)\ell\right) \end{aligned}$$

□

With this result, we introduce the following notation for convenience.

Definition 2.6. For a Fourier-Mukai transform of the form $\Phi_{E[1]}$, where $E \in \mathrm{Coh}(\mathbb{T}\times\mathbb{T})$, define the following quantities:

$$\begin{aligned} \beta_{\mathrm{dom}}(\Phi_{E[1]}) &= \mu(\Phi^{-1}_{E[1]}(\mathcal{O}_x) = -\frac{a}{b} \\ \beta_{\mathrm{codom}}(\Phi_{E[1]}) &= \mu(\Phi_{E[1]}(\mathcal{O}_x) = \frac{d}{b} \end{aligned} \qquad \text{where } \Phi^H_E = \begin{pmatrix} a & b \\ c & d\end{pmatrix}.$$

Remark. We may extend these to functions on all Fourier-Mukai transforms by taking the unique shift of the Fourier-Mukai transform which is of the form above.
This was defined so that we can write:

$$\Phi_{E[1]}\cdot\sigma_{\alpha,\beta_{\mathrm{dom}}(\Phi_{E[1]})} = \sigma_{\frac{1}{\alpha b^2},\beta_{\mathrm{codom}}(\Phi_{E[1]})}.$$

With respect to the illustrations of the upper-half plane parametrising geometric stability conditions, this means that Fourier-Mukai transforms of a certain form map a certain vertical line to another, whilst flipping it vertically.

Example 2.2 (Action of standard FMT on whole plane (modulo $\widetilde{\mathbb{C}^*}$ action))**.** *In the above theorem, we only consider when stability condition in the slice of stability conditions we parametrise, are mapped to other stability conditions on that same slice. This turns out to generally only happen at a single vertical line, however other stability conditions are mapped*

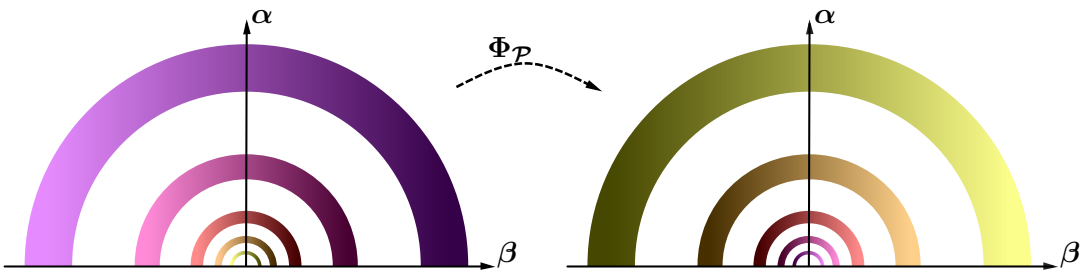


Figure 2.2: Action of standard FMT on whole plane of stability conditions, up to $\widetilde{\mathbb{C}^*}$ action.

to valid stability conditions, but not in our slice (illustrated in Figure 2.1). This slice does generate all stability conditions up to $\widetilde{GL}^+(2,\mathbb{R})$ action as seen in Proposition 1.9.

Suppose we took the original Fourier-Mukai transform induced by the Picard sheaf, apply it to stability conditions in our slice, then 'project' along a $\widetilde{GL}^+(2,\mathbb{R})$ action onto the slice again. We would get

$$\Phi_{\mathcal{P}} \cdot \sigma_{r\sin(\theta),r\cos(\theta)} = \lambda \cdot \sigma_{\frac{1}{r}\sin(\pi-\theta),\frac{1}{r}\sin(\pi-\theta)} \qquad \lambda \in \widetilde{\mathbb{C}^*},$$

so that in polar coordinates, the action is very simple:

$$(r,\theta) \mapsto \left(\frac{1}{r}, \pi-\theta\right). \tag{2.1}$$

This action is illustrated in Figure 2.2 and follows by the calculation:

$$\begin{aligned}
\Phi^H\left(\exp\left((r\cos(\theta)+r\sin(\theta)i)\,\ell\right)\right) &= \Gamma^{-1}\left(\left(\left[\begin{pmatrix}0 & 1\\ -1 & 0\end{pmatrix}\right],1\right)\cdot\left[\begin{pmatrix}1\\ r\cos(\theta)+r\sin(\theta)i\end{pmatrix}\right]\right)\\
&= \Gamma^{-1}\left(\left[\begin{pmatrix}r\cos(\theta)+r\sin(\theta)i\\ -1\end{pmatrix}\right]\right)\\
&= \Gamma^{-1}\left(\left[(r\cos(\theta)+r\sin(\theta)i)\begin{pmatrix}1\\ \frac{-\cos(\theta)+i\sin(\theta)}{r}\end{pmatrix}\right]\right)\\
&= -r^2\left(\cos(2\theta)+\sin(2\theta)\right)\exp\left(\left(\frac{\cos(\pi-\theta)+i\sin(\pi-\theta)}{r}\right)\ell\right).
\end{aligned}$$

Motivated by this example, we now pursue this action of cohomological Fourier-Mukai transforms on stability conditions module the action of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$. Following Proposition 1.8, we have that stability conditions modulo $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ is parametrised by the upper-half complex plane. We have also seen that the group of cohomological Fourier-Mukai transform is $\mathrm{PSL}(2,\mathbb{Z})\times\{\pm 1\}$ with the second direct summand accounting for shifts, which are give a trivial action on stability conditions modulo $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ (since a half rotation of the central charge has the same effect). This will induce an action of $\mathrm{PSL}(2,\mathbb{Z})$ on the upper-half complex plane. Outside of algebraic geometry, this group is known as the 'modular group' [23], which is known to act on the upper-half complex plane by Möbius transformation. This next theorem identifies these two actions.

Theorem 2.8 (Relating CFMT with the modular group). *Let $\mathbb{T}$ be a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, and a cohomological Fourier-Mukai transform*

$$\Phi^H = \left(\left[\begin{pmatrix} a & b \\ c & d \end{pmatrix} \right], \pm 1 \right) \in \mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\},$$

then the action of Φ^H on stability conditions modulo $\widetilde{GL}^+(2,\mathbb{R})$, on the representatives given by the parametrisation in Definition 1.23, is given by

$$\Phi^H \cdot \left[\sigma_{\mathfrak{Im}(z),\mathfrak{Re}(z)}\right] = \left[\sigma_{\mathfrak{Im}(z'),\mathfrak{Re}(z')}\right] \qquad \textit{where} \qquad z' = \frac{c+dz}{a+bz}.$$

Remark. This induces a hyperbolic geometry on $\mathrm{Stab}(\mathbb{T})/\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ where all walls are straight lines. The cohomological Fourier-Mukai transforms of interest in Part III are examples of hyperbolic Möbius transformations. These transformations have 'attractive' and 'repulsive' fixed points and iterative applications on the vertical line of points equidistant from them gives a family of circles which form part of the larger pencil of 'Apollonian circles' (Figure 2.3).

Proof. First, note that the action of autoequivalences of the triangulated category on the space of stability conditions (Definition 1.15) commutes with the action of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ (Definition 1.16). So the action of cohomological Fourier-Mukai transforms on stability conditions, does descend to an action on stability conditions modulo $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$. Furthermore, Proposition 1.8 gives us that the stability conditions given by the parametrisation in Definition 1.23 can be chosen as representatives[2].

Next, all stability conditions on abelian surfaces are geometric (consequence of [FLZ22, Corollary 2.17]), so Proposition 1.7 gives that the stability conditions are determined by their central charge up to even shifts, so modulo $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$, they are determined uniquely. Therefore, we need only consider the action of Φ^H on central charges $\mathcal{Z}\colon H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \to \mathbb{C}$ modulo $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ (which reduces to an action of $\mathrm{GL}^+(2,\mathbb{R})$). The central charges of the stability conditions $\sigma_{\alpha,\beta}$ are given by $\mathcal{Z}_{\beta+i\alpha}$ when using the notation defined in the following:

$$\frac{\left\{v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \otimes_{\mathbb{Z}} \mathbb{C} \colon \Delta(v) = 0\right\}}{\mathbb{C}^*} \to \frac{\hom_{\mathbb{Z}}(H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}),\mathbb{C})}{\mathrm{GL}^+(2,\mathbb{R})}$$
$$[v] \mapsto [\langle v, \_ \rangle].$$

Next note that Γ from Definition 2.5 descends to the bijection

$$[\Gamma]\colon \frac{\left\{v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T}) \otimes_{\mathbb{Z}} \mathbb{C} \colon \Delta(v) = 0\right\}}{\mathbb{C}^*} \to \mathbb{CP}^2$$
$$[\exp(z\ell)] \mapsto \begin{pmatrix} 1 \\ z \end{pmatrix}$$
$$[(0,0,1)] \mapsto \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

[2] On a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, parametrisations 'A' and 'B' are identical

These can be composed to give

$$\begin{aligned} f\colon \mathbb{CP}^2 &\to \frac{\hom_{\mathbb{Z}}(H^{\text{even}}_{\text{alg}}(\mathbb{T}),\mathbb{C})}{\mathrm{GL}^+(2,\mathbb{R})} \\ [u] &\mapsto \left[\left\langle \exp(\Gamma^{-1}(u)\ell), \_ \right\rangle\right], \end{aligned}$$

so that the equivalence class $f\left(\left[\begin{pmatrix}1 & \beta+i\alpha\end{pmatrix}^T\right]\right)$ contains the central charge for $\sigma_{\alpha,\beta}$. Finally, note that Φ^H descends to a projective linear map $\left[\Phi^H\right]$ on $\mathbb{CP}^2$ (which are naturally identified with Möbius transformations).

With this notation, we now have a clean way of describing the action of Φ^H on the central charges of the stability conditions (modulo $\mathrm{GL}^+(2,\mathbb{R})$), which are contained in the image of f. This action shall be related to the action of $\left[\Phi^H\right]$ on $\mathbb{CP}^2$:

$$\begin{aligned} \Phi^H \cdot f([u]) &= \left[\left\langle u, (\Phi^H)^{-1}(\_)\right\rangle\right] \\ &= \left[\left\langle \Phi^H(u), \_\right\rangle\right] \\ &= f\left(\left[\Phi^H\right]([u])\right). \end{aligned}$$

Now consider the action $\left[\Phi^H\right]$ on the pre-image of the central charges for $\sigma_{\alpha,\beta}$. Take $z = \beta + i\alpha$:

$$\begin{aligned} \left[\Phi^H\right]\left(\left[\begin{pmatrix}1\\ z\end{pmatrix}\right]\right) &= \left[\begin{pmatrix}a+bz\\ c+dz\end{pmatrix}\right] \\ &= \left[\begin{pmatrix}1\\ \dfrac{c+dz}{a+bz}\end{pmatrix}\right]. \end{aligned}$$

This therefore proves the action of Φ^H on the equivalence classes of stability conditions in the statement of the theorem.

□

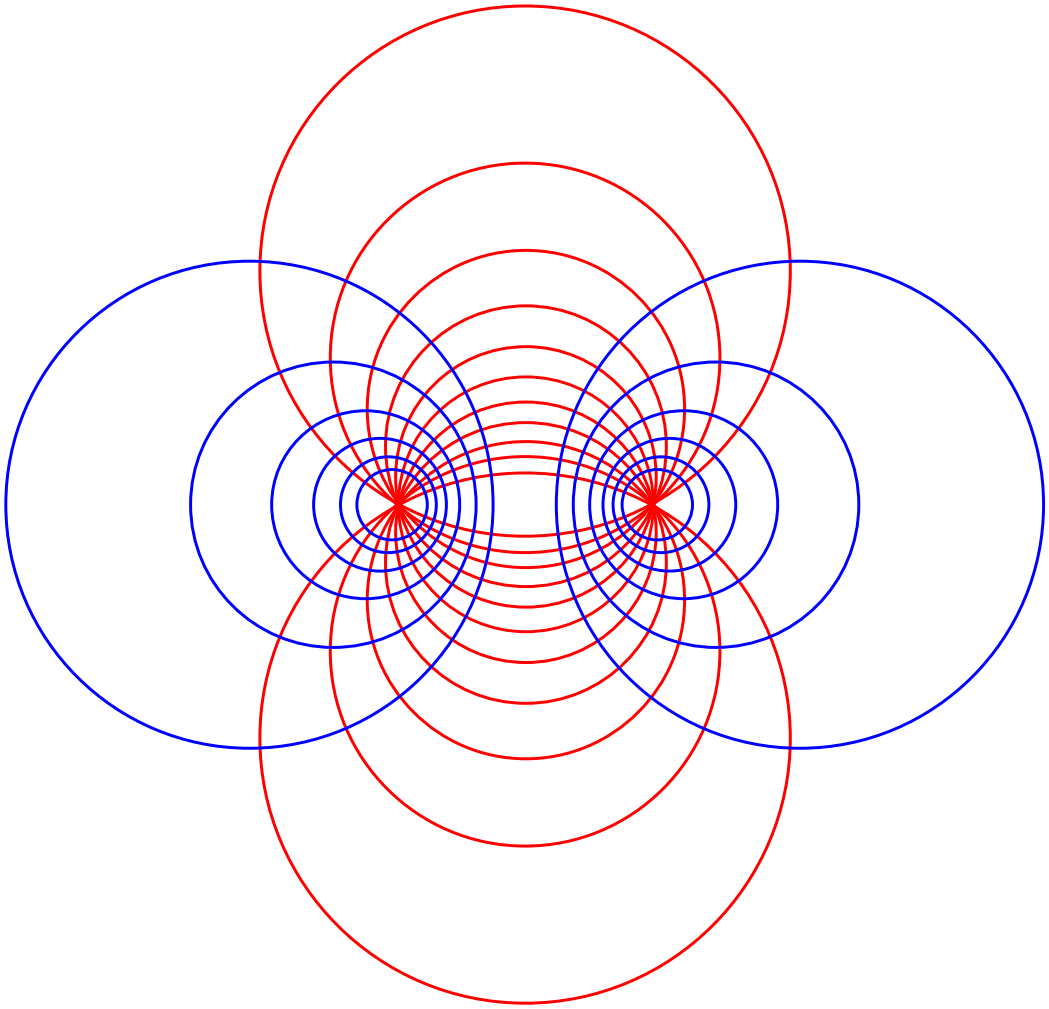


Figure 2.3: By WillowW (original); Pbroks13 (redraw) - Own work based on: Apollonian circles.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4171350
The configuration of these circles is reminiscent to walls in the $\{\sigma_{\alpha,\beta}\}$ half plane. Indeed they are connected as CFMT acts on the $\{\sigma_{\alpha,\beta}\}$ upper-half plane in the same way as the Modular group acts as Möbius transformations on the upper-half complex plane. A subset of these Möbius transformations, hyperbolic ones, can be used to generate the Apollonian circles.

Part II

Finite Walls

```
$ ./pseudo_tilt --semistabilizers 3 2 -4
Chern Character v = (3, 2ℓ, -2ℓ²)
Δ(v) = 16ℓ²
β_(v) = -2/3

pseudo-semistabilizers:
(1, 0ℓ, 0ℓ²)
(4, -2ℓ, 1/2ℓ²)
(2, 0ℓ, 0ℓ²)
(8, -4ℓ, 1ℓ²)
(4, -1ℓ, 0ℓ²)
(7, -3ℓ, 1/2ℓ²)
(16, -9ℓ, 5/2ℓ²)
(25, -15ℓ, 9/2ℓ²)
```

Introduction

In this part, we consider smooth projective surfaces of Picard rank 1, with typical examples being principally polarised abelian surfaces with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, and the projective plane $\mathbb{P}^2$. For the rest of this thesis, the parametrisation used for the slice of stability conditions where we search for walls is the one given by Definition 1.23. These will be illustrated frequently with diagrams of the upper-half plane. The purpose of Part II is the computation of walls when there are known to be finitely many.

Background

It is known that for any rational β_0, the vertical line of stability conditions $\{\sigma_{\alpha,\beta_0} : \alpha \in \mathbb{R}_{>0}\}$ only intersects finitely many walls of a Chern character v: [LQ14, Thm 1.1] [Mac14, Prop 4.2] [MYY13, Lemma 5.20]. There were two main strategies to prove this, the first was to consider numerical restrictions on Chern characters of possible semistabilisers along this line and show there are finitely many solutions. The second strategy being more theoretical, was to first that show that the circular walls cannot be arbitrarily large, then construct a Fourier-Mukai transform which in effect transforms these walls to walls for a different Chern character intersecting a similar vertical line, but inverting the sizes of the corresponding circular walls. By symmetry and a theoretical result about local finiteness of walls, this also gave the finiteness of walls intersecting that vertical line.

A consequence of this is that if $\Delta^{\ell^2}(v)$ is a square integer, and so (equivalently) $\beta_\pm(v)$ are rational, then two vertical lines can be chosen at $\beta = \beta_\pm(v)$ to intersect all circular walls. In this case, we deduce that v has only finitely many circular walls, and a possible extra vertical wall at V_v. Hence, finitely many in total. This sufficient condition for finiteness not only holds for real walls realised by actual objects in $\mathcal{D}^b(X)$ but also 'pseudo-walls'. Here pseudo-walls will be defined as 'potential' walls, induced by hypothetical Chern characters of semistabilisers which satisfy certain numerical conditions which would be satisfied by any real semistabiliser (Definition 3.1).

For the problem of computing these semistabilisers, one of the main motivating goals for this thesis, there exists a SageMath [The22] library [Sch20b], which tackles this problem. This library also contains an explicit bound for possible ranks of a certain class of semistabilisers of a Chern character v satisfying the above condition of $\Delta^{\ell^2}(v)$ being a square integer. However the performance of this library scales poorly, as discussed in Section 5.1.1, and the given bound can be refined.

Part Summary

Let v be a Chern character such that $\Delta^{\ell^2}(v)$ is a square integer, hence has only finitely many walls. For any semistabilizing sequence $E \hookrightarrow F \twoheadrightarrow G$ in $\mathcal{B}^\beta$ with $\mathrm{ch}(F) = v$, destabilising F as we move 'inwards' across a circular wall left of $V_{\mathrm{ch}(F)}$, there are some numerical conditions imposed on their Chern characters (to be introduced in Lemma 3.3). Chapter 3 clarifies these statements and clearly states the numerical problems we seek to solve. These numerical restrictions give straightforward bounds on possible values for $\mathrm{ch}_1(E)$ and $\mathrm{ch}_2(E)$ for any fixed value of $\mathrm{ch}_0(E)$. Providing any bound on $\mathrm{ch}_0(E)$, as has been done by [Sch20a], then proves the finiteness of walls. Chapter 4 explores refinements on $\mathrm{ch}_0(E)$. Finally Chapter 5 uses these refinements to develop a newer algorithm for computing possibilities of semistabilisers giving rise to circular walls left of V_v as an alternative to one of the features provided by [Sch20b].

Chapter 3

Setting and Problems

3.1 Definitions: Clarifying 'pseudo'

Throughout this Part, as noted in the introduction, we will be exclusively working over surfaces X with Picard rank 1, with a choice of ample line bundle L such that $\ell := c_1(L)$ generates $NS(X)$. We take $m := \ell^2$ as this will be the main quantity which will affect the results.

Definition 3.1 (Pseudo-semistabilisers)**.** Given a Chern Character v, and a given stability condition $\sigma_{\alpha,\beta}$, a **pseudo-semistabilising** u is a 'potential' Chern character:

$$u = \left(r, c\ell, \frac{e}{\operatorname{lcm}(m,2)}\ell^2\right) \qquad r, c, e \in \mathbb{Z},$$

which has the same tilt slope as v: $\nu_{\alpha,\beta}(u) = \nu_{\alpha,\beta}(v)$.
Furthermore the following inequalities are satisfied:

- $\Delta(u) \geq 0$
- $\Delta(v-u) \geq 0$
- $0 \leq \mathrm{ch}_1^\beta(u) \leq \mathrm{ch}_1^\beta(v)$.

Remark. We could introduce a slightly stronger definition including an extra condition on e in terms of r and c to ensure that u could arise from integral Chern classes. However, this will not affect finiteness questions considered later and this also condition turns out to be vacuous for principally polarised abelian surfaces with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$.

Remark. Note u does not need to be a Chern character of an actual sub-object of some object in the stability condition's heart with Chern character v.

Other sources may also have extra conditions such as the integrality of Euler characteristics or the non-triviality of certain extension groups. These conditions depend on the Todd class of the variety in question and could be considered but are left out for now as they do not have a great impact on the finiteness of pseudo-walls. In the case of a principally polarised abelian surface, the main example in this thesis, the Euler characteristic condition is vacuous. The extension group condition can be shown to only be redundant among the other conditions for sufficiently large rank of u (when $\mathrm{ch}_0(u) \geq \mathrm{ch}_0(v)/2$), so does not affect the finiteness of pseudo-semistabilisers.

Later, Chern characters will be written $(r, c\ell, d\ell^2)$ because operations (such as multiplication) are more easily defined in terms of the coefficients of the ℓ^i. However, at the end, it will become important again that $d \in \frac{1}{\operatorname{lcm}(m,2)}\mathbb{Z}$.

Definition 3.2 (Pseudo-walls)**.** Let u be a pseudo-semistabiliser of v, for some stability condition. Then the **pseudo-wall** associated to u is the set of all stablity conditions where u is a pseudo-semistabiliser of v.

Lemma 3.1 (Sanity check for Pseudo-semistabilisers)**.** *Given a stability condition $\sigma_{\alpha,\beta}$, and a semistabilising sequence $E \hookrightarrow F \twoheadrightarrow G$ in $\mathcal{B}^\beta$ for F, then $\mathrm{ch}(E)$ is a pseudo-semistabiliser of $\mathrm{ch}(F)$.*

Proof. Suppose $E \hookrightarrow F \twoheadrightarrow G$ is a semistabilising sequence with respect to a stability condition $\sigma_{\alpha,\beta}$.

$$\mathrm{ch}(E) = -\mathrm{ch}(\mathcal{H}^{-1}_{\mathrm{Coh}}(E)) + \mathrm{ch}(\mathcal{H}^{0}_{\mathrm{Coh}}(E))$$

Therefore, $\mathrm{ch}(E)$ is of the form $(r, c\ell, \frac{e}{\mathrm{lcm}(m,2)}\ell^2)$ provided that this is true for any coherent sheaf. For any coherent sheaf H, we have the following:

$$\mathrm{ch}(H) = \left(c_0(H), c_1(H), -c_2(H) + \frac{1}{2}c_1(H)^2\right).$$

Given that ℓ generates the Neron-Severi group, $c_1(H)$ can then be written $c\ell$.

$$\mathrm{ch}(H) = \left(c_0(H), c\ell, \left(-\frac{c_2(H)}{\ell^2} + \frac{c^2}{2}\right)\ell^2\right)$$

This fact along with c_0, c_2 being an integers on surfaces, and $m := \ell^2$ implies that $\mathrm{ch}(H)$ (hence $\mathrm{ch}(E)$ too) is of the required form.

Since all the objects in the sequence are in $\mathcal{B}^\beta$, we have $\mathrm{ch}_1^\beta \geq 0$ for each of them. Due to additivity ($\mathrm{ch}(F) = \mathrm{ch}(E) + \mathrm{ch}(G)$), we can deduce $0 \leq \mathrm{ch}_1^\beta(E) \leq \mathrm{ch}_1^\beta(F)$.

$E \hookrightarrow F \twoheadrightarrow G$ being a semistabilising sequence means $\nu_{\alpha,\beta}(E) = \nu_{\alpha,\beta}(F) = \nu_{\alpha,\beta}(F)$. But also, that this is an instance of F being semistable, so E must also be semistable (otherwise the destabilising subobject would also destabilise F). Similarly G must also be semistable too. E and G being semistable implies they also satisfy the Bogomolov inequalities (Theorem 1.6): $\Delta(E), \Delta(G) \geq 0$. Expressing this in terms of Chern characters for E and F gives: $\Delta(\mathrm{ch}(E)) \geq 0$ and $\Delta(\mathrm{ch}(F) - \mathrm{ch}(E)) \geq 0$.

□

3.2 Characteristic Curves for Pseudo-semistabilisers

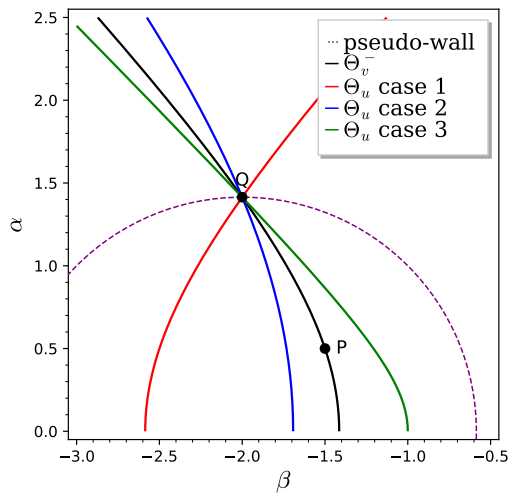
These characteristic curves introduced in Section 1.4.1 are convenient tools to think about the numerical conditions that can be used to test for pseudo-semistabilisers, and for solutions to the problems (1,2) tackled in this Part (to be introduced later). In particular, these problems will be translated to a list of numerical inequalities on its solutions u.

The next Lemma is a key to making this translation and revolves around the geometry and configuration of the characteristic curves involved in a semistabilising sequence.

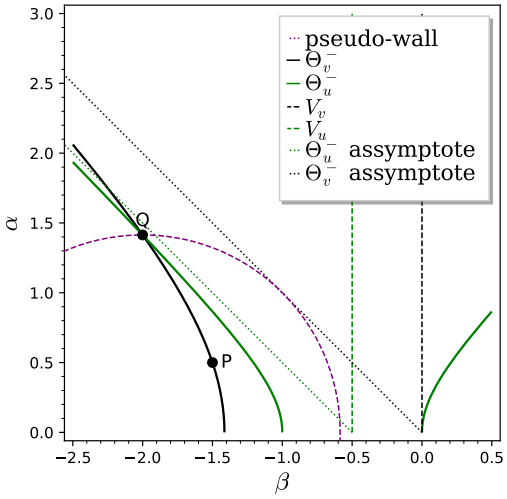
Lemma 3.2 (Numerical tests for left-wall pseudo-semistabilisers)**.** *Let v and u be Chern characters with $\Delta(v), \Delta(u) \geq 0$, and v is positive, with one of $\mathrm{ch}_0(v)$ or $\mathrm{ch}_1(v)$ non-zero. Let $P = (\alpha_0, \beta_0)$ be a point on Θ_v^-.*
The following conditions:

a) u is a pseudo-semistabiliser of v at some point on Θ_v^- above P, and

b) u destabilises v going 'inwards', that is, $\nu_{\alpha,\beta}(u) < \nu_{\alpha,\beta}(v)$ outside the pseudo-wall, and $\nu_{\alpha,\beta}(u) > \nu_{\alpha,\beta}(v)$ inside,



(a) Three ways the characteristic hyperbola for u can intersect the left branch of the characteristic hyperbola for v



(b) Closer look at characteristic curves for valid case

are equivalent to the following more numerical conditions:

1. *u has positive rank,*
2. *$\beta_0 < \mu(u) < \mu(v)$, i.e. V_u is strictly between P and V_v,*
3. $\mathrm{ch}_1^{\beta_0}(u) < \mathrm{ch}_1^{\beta_0}(v)$,
4. $\Delta(v-u) \geq 0$, *and*
5. $\mathrm{ch}_2^{\alpha_0,\beta_0}(u) > 0$.

Proof. First, consider the case where $\mathrm{ch}_0(v) > 0$. Let u, v be Chern characters such that $\Delta(u), \Delta(v) \geq 0$, and v has positive rank.

For the forwards implication, assume that the suppositions of the Lemma are satisfied. Let Q be the point on Θ_v^- (above P) where u is a pseudo-semistabiliser of v. Firstly, consequence 4 is part of the definition for u being a pseudo-semistabiliser at a point with same β value of P (since the pseudo-wall surrounds P). Also, u is a pseudo-semistabiliser along the pseudo-wall which surround P, so by definition of pseudo-semistabilisers, $0 \leq \mathrm{ch}_1^{\beta}(u) \leq \mathrm{ch}_1^{\beta}(v)$ for all (α, β) on the pseudo-wall. In particular, this holds for β in a neighbourhood of β_0, so we can conclude $0 < \mathrm{ch}_1^{\beta_0}(u) < \mathrm{ch}_1^{\beta_0}(v)$ (consequence 3). Notice that $0 < \mathrm{ch}_1^{\beta_0}(u)$ follows from consequences 1 and 2, this is why it is not included in consequence 3. If u were to have 0 rank, its tilt slope would be decreasing as β increases, contradicting supposition b. So u must have strictly non-zero rank, and we can consider its characteristic curves (or that of $-u$ in case of negative rank). $\nu_Q(v) = 0$, and hence $\nu_Q(u) = 0$ too. This means that $\Theta_{\pm u}$ must intersect Θ_v^- at Q. Considering the shapes of the hyperbolae alone, there are 3 distinct ways that they can intersect, as illustrated in Fig 3.1a. These cases are distinguished by whether it is the left, or the right branch of Θ_u involved, as well as the positions of the base. However, considering supposition b, only case 3 (green in figure) is possible. This is because we need $\nu_P(u) > 0$ (or $\nu_P(-u) > 0$ in case 1 involving Θ_u^+), to satisfy supposition b. Recalling how the sign of $\nu_{\alpha,\beta}(\pm u)$ changes (illustrated in Figure 1.1), we can eliminate cases 1 and 2.

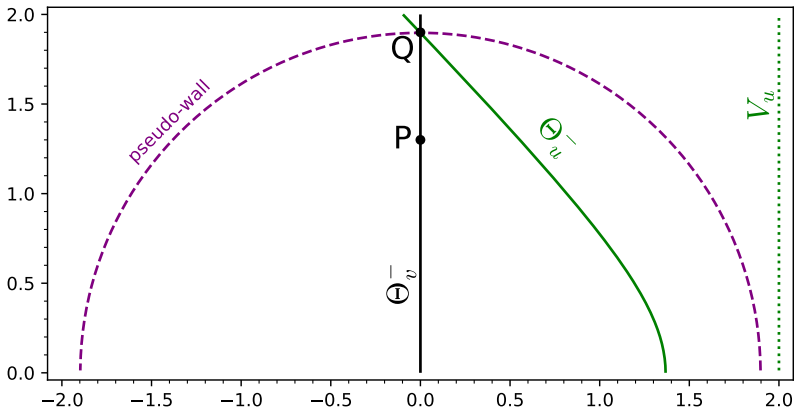


Figure 3.2: characteristic curves configuration for pseudo-semistabiliser of v with rank 0 destabilising 'downwards'

Fixing attention on the only possible case (2), illustrated in Fig 3.1b. P is on the left of $V_{\pm u}$ (first part of consequence 2), so u must have positive rank (consequence 1) to ensure that $\mathrm{ch}_1^{\beta_0} \geq 0$ (since the pseudo-wall passed over P). Furthermore, P being on the left of V_u implies $\mathrm{ch}_1^{\beta_0}(u) > 0$, and therefore $\mathrm{ch}_2^P(u) > 0$ (consequence 5) to satisfy supposition b. Next considering the way the hyperbolae intersect, we must have Θ_u^- taking a base-point to the right Θ_v, but then, further up, crossing over to the left side. The latter fact implies that the assymptote for Θ_u^- must be to the left of the one for Θ_v^-. Given that they are parallel and intersect the β-axis at $\beta = \mu(u)$ and $\beta = \mu(v)$ respectively. We must have $\mu(u) < \mu(v)$ (second part of consequence 2), that is, V_u is strictly to the left of V_v.

Conversely, suppose that the consequences 1-5 are satisfied. Consequence 2 implies that the assymptote for Θ_u^- is to the left of Θ_v^-. Consequences 4 and 5, along with $\beta_0 < \mu(u)$, implies that P must be in the region left of Θ_u^-. These two facts imply that Θ_u^- is to the right of Θ_v^- at $\alpha = \alpha(P)$, but crosses to the left side as $\alpha \to +\infty$, intersection at some point Q above P. This implies that the characteristic curves for u and v are in the configuration illustrated in Figure 3.1b. We then have $\nu(u) = \nu(v)$ along a circle to the left of V_u reaching its apex at Q, and encircling P. This along with consequence 3 implies that u is a pseudo-semistabiliser at the point on the circle with $\beta = \beta_0$. Therefore, it is also a pseudo-semistabiliser further along the circle at Q (supposition a). Finally, consequence 4 along with P being to the left of V_u implies $\nu_P(u) > 0$ giving supposition b.

Now consider the case where $\mathrm{ch}_0(v) = 0$ but $\mathrm{ch}_1(v) > 0$. Let u, v be Chern characters with $\Delta(u), \Delta(v) \geq 0$, and $\mathrm{ch}_0(v) = 0$ but $\mathrm{ch}_1(v) > 0$. So Θ_v^- is the vertical line at $\beta = \beta_-(v)$, and $\mu(v) = +\infty$.

For the forward implication, assume suppositions a and b hold. Let Q be the point above P on Θ_v^- where u is a pseudo-semistabiliser of v. So Θ_u intersects Θ_v^- at Q. Suppose, seeking a contradiction that $\mathrm{ch}_0(u) = 0$, then $\Theta_u = \Theta_u^-$ is also a vertical line, and must then be the same line as Θ_v^- to intersect Q. This would imply $\mathbb{Q}u = \mathbb{Q}v$ and then u would not destabilise v at any stability condition. So we must have either $\mathrm{ch}_0(u) < 0$, in which case Θ_u^+ is the branch of Θ_u going through Q; or $\mathrm{ch}_0(u) > 0$, in which it is Θ_u^- instead. The latter case is the only one which could satisfy supposition b, about u destabilising v going 'down' Θ_v^-. Which then forces the configuration of characteristic curves shown in Figure 3.2. The positions of

the characteristic curves ensures the numerical conditions 1, 2 and 5. The other conditions follow from the definition of u being a pseudo-semistabiliser of v.

Conversely, suppose that the numerical conditions 1-5 are satisfied, then this forces the configuration of characteristic curves shown in Figure 3.2. This ensures that u is a pseudo-semistabiliser of v at u destabilising going down Θ_v^-. □

Remark. Given a fixed positive v with $\Delta(v) \geq 0$, for any u satisfying the numerical conditions of this Lemma, with some given fixed rank ($\mathrm{ch}_0(u)$), Condition 2 gives us bounds for $\mathrm{ch}_1(u)$. For any fixed values for $\mathrm{ch}_0(u)$ and $\mathrm{ch}_1(u)$, the last three conditions will give bounds for $\mathrm{ch}_2(u)$. These bounds exist regardless of any extra assumptions about v. However any bounds on $\mathrm{ch}_0(u)$ are less immediate from these numerical conditions. The semistabiliser E for the largest 'left' wall for v must destabilise a Gieseker stable sheaf F with $\mathrm{ch}(F) = v$. We would have a short exact sequence in the heart of a stability condition on the wall given by

$$0 \to E \hookrightarrow F \twoheadrightarrow G \to 0,$$

for some G in the heart. Considering cohomology over $\mathrm{Coh}(X)$, and using the fact that F is a sheaf, we get an exact sequence in $\mathrm{Coh}(X)$ given by:

$$0 \to \cancelto{0}{\mathcal{H}^{-1}(E)} \to 0 \to \mathcal{H}^{-1}(G) \to \mathcal{H}^0(E) \to \mathcal{H}^0(F) \to \mathcal{H}^0(G) \to 0.$$

So we do have that E must be a sheaf, but not necessarily a subsheaf of F, and so $\mathrm{ch}_0(E)$ is not necessarily smaller than $\mathrm{ch}_0(F)$. Furthermore, for smaller walls, the semistabiliser may not even be a sheaf.

When choosing the base-point of Θ_v^-: $P = (\beta_-(v), 0)$, we will see in Part III that there can be infinitely many walls when $\beta_-(v)$ is irrational, hence infinitely many u satisfying the above. Therefore any construction of a bound on the ranks of possible u must rely on the rationality of β_0 in this case.

3.3 The Problem: Finding Pseudo-walls

As hinted in the introduction to this Part II, the main motivation of the results in this article are not only the bounds on pseudo-semistabiliser ranks; but also applications for finding a list (comprehensive or subset) of pseudo-walls.

After introducing the characteristic curves of stability conditions associated to a fixed Chern character v, we can now formally state the problems that we are trying to solve for.

3.3.1 Problem statements

> **Problem Statement 1** (sufficiently large 'left' pseudo-walls)**.** Fix a Chern character v with non-negative rank (and $\mathrm{ch}_1(v) > 0$ if rank 0), and $\Delta(v) \geq 0$. The goal is to find all pseudo-semistabilisers u which give circular pseudo-walls containing some fixed point $P \in \Theta_v^-$ with the added restriction that u 'destabilises' v moving 'inwards', that is, $\nu(u) > \nu(v)$ inside the circular pseudo-wall.

This will give all pseudo-walls between the chamber corresponding to Gieseker stability and the stability condition corresponding to P. The purpose of the final 'direction' condition is because, up to that condition, semistabilisers are not distinguished from their corresponding quotients: Suppose $E \hookrightarrow F \twoheadrightarrow G$, then the tilt slopes $\nu_{\alpha,\beta}$ are strictly increasing, strictly decreasing, or equal across the short exact sequence (consequence of the see-saw principle).

In this case, $\mathrm{ch}(E)$ is a pseudo-semistabiliser of $\mathrm{ch}(F)$, if and only if $\mathrm{ch}(G)$ is a pseudo-semistabiliser of $\mathrm{ch}(F)$. The numerical inequalities in the definition for pseudo-semistabiliser cannot tell which of E or G is the subobject. However, what can be distinguished is the direction across the wall that $\mathrm{ch}(E)$ or $\mathrm{ch}(G)$ destabilises $\mathrm{ch}(F)$ (they will each destabilise in the opposite direction to the other). The 'inwards' semistabilisers are preferred because we are moving from a typically more familiar chamber (the stable objects of Chern character v in the outside chamber will only be Gieseker stable sheaves).

Remark. Also note that this last restriction does not remove any pseudo-walls found, and if we do want to recover 'outwards' semistabilisers, we can simply take $v - u$ for each solution u of the problem.

Problem Statement 2 (all 'left' pseudo-walls)**.** Fix a Chern character v with non-negative rank (and $\mathrm{ch}_1(v) > 0$ if rank 0), $\Delta(v) \geq 0$, and $\beta_-(v) \in \mathbb{Q}$. The goal is to find all pseudo-semistabilisers u which give circular pseudo-walls on the left side of V_v. Where u destabilises v going 'down' Θ_v^- (in the same sense as in Problem 1.

This is a specialisation of Problem 1 with the choice $P = (\beta_-, 0)$, the point where Θ_v^- meets the β-axis. This is because all circular walls left of V_v intersect Θ_v^- (once). The $\beta_-(v) \in \mathbb{Q}$ condition is to ensure that there are finitely many solutions. As mentioned in the introduction to this Part, this is known, however this will also be proved again implicitly in Chapter 5, where an algorithm is produced to find all solutions.

This description still holds for the case of rank 0 case if we consider V_v to be infinitely far to the right (see Section 1.4.2). Note also that the condition on $\beta_-(v)$ always holds for v rank 0.

3.3.2 Numerical Formulations of the Problems

The problems introduced in this section are phrased in the context of stability conditions. However, these can be reduced down completely to purely numerical problem using Lemma 3.2.

Theorem 3.3 (Numerical Tests for Sufficiently Large 'left' Pseudo-walls)**.** *Given a Chern character v with non-negative rank (and $\mathrm{ch}_1(v) > 0$ if rank 0), and $\Delta(v) \geq 0$, and a choice of point $P = (\alpha_0, \beta_0)$ on Θ_v^-. Solutions $u = (r, c\ell, d\ell^2)$ to Problem 1 are precisely given by $r, c \in \mathbb{Z}$, $d \in \frac{1}{\mathrm{lcm}(m,2)}$ satisfying the following conditions:*

1. $r > 0$,
2. $\Delta(u) \geq 0$,
3. $\Delta(v - u) \geq 0$,
4. $\mu(u) < \mu(v)$,
5. $0 < \mathrm{ch}_1^{\beta_0}(u) < \mathrm{ch}_1^{\beta_0}(v)$,
6. $\mathrm{ch}_2^{\alpha_0,\beta_0}(u) > 0$.

Proof. Consider the context of v being a Chern character with non-negative rank (and with $\mathrm{ch}_1(v) > 0$ if rank 0) and $\Delta \geq 0$, and u being a Chern character with $\Delta(u) \geq 0$. Lemma 3.2 gives that the remaining conditions for u being a solution to Problem 1 are precisely equivalent to the remaining conditions in this lemma.

□

Theorem 3.4 (Numerical Tests for All 'left' Pseudo-walls)**.** *Given a Chern character v with non-negative rank (and $\mathrm{ch}_1(v) > 0$ if rank 0), and $\Delta(v) \geq 0$, such that $\beta_- := \beta_-(v) \in \mathbb{Q}$. Solutions $u = (r, c\ell, d\ell^2)$ to Problem 2 are precisely given by $r, c \in \mathbb{Z}$, $d \in \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}$ satisfying the following conditions:*

1. $r > 0$,
2. $\Delta(u) \geq 0$,
3. $\Delta(v-u) \geq 0$,
4. $\mu(u) < \mu(v)$,
5. $0 < \mathrm{ch}_1^{\beta_-}(u) < \mathrm{ch}_1^{\beta_-}(v)$,
6. $\mathrm{ch}_2^{\beta_-}(u) > 0$.

Proof. This is a specialisation of the previous Theorem 3.3, using $P = (\beta_-, 0)$. □

Chapter 4

Bounds on Semistabiliser Ranks

With Problems 1 and 2 stated in the previous chapter, as well as their numerical characterisations given by Lemma 3.3 and Corollary 3.4, we consider bounds on the ranks of their solutions. First we cover global bounds on all ranks of solutions to either problem, but then for Problem 2, or Problem 1 in cases where $\beta(P)$ [1] is rational, we consider much stricter bounds for different subsets of solutions as these results can be used in algorithms to compute these solutions more efficiently. Discussing these algorithms will be left to the next chapter.

4.1 Existing Bound on Semistabiliser Ranks

The proof for the following Theorem 4.1 was hinted at in [Sch20a], but the value appears explicitly in [Sch20b] as shown in the following Listing 4.1. The latter citation is a SageMath [The22] library for computing certain quantities related to Bridgeland stabilities on Picard rank 1 varieties. It also includes functions to compute pseudo-walls and pseudo-semistabilisers for tilt stability.

Listing 4.1: Snippet from [Sch20b] `stability_conditions.tilt.walls_left`

```
# Find the maximal rank for a destabilizing subobject or quotient and
# the minimal radius squared of our walls.
if s == beta_minus(v):
    p = 0
    n = Integer(s.denominator())
    m = Integer(1/var.gens[2])
    r_max = (n * ch(v, s)[1] - 1) ** 2 * m / gcd(2 * n ** 2, m)
```

Theorem 4.1 (Bound on r - Benjamin Schmidt). *Let X be a smooth projective Picard rank 1 surface with choice of ample line bundle L such that $\ell := c_1(L)$ generates $\mathrm{NS}(X)$ and take $m := \ell^2$.*

Given a Chern character v with $\Delta(v) \geq 0$ and positive rank (or $\mathrm{ch}_0(v) = 0$ but with $\mathrm{ch}_1(v) > 0$) such that $\beta_- := \beta_-(v) \in \mathbb{Q}$, the rank r of any solution u of Problem 2 is bounded above by:

$$r \leq \frac{m\left(n\,\mathrm{ch}_1^{\beta_-}(v) - 1\right)^2}{\gcd(m, 2n^2)}.$$

Proof. The Bogomolov form applied to the twisted Chern character is the same as the untwisted one.

[1] Notation for the β coordinate of the point P

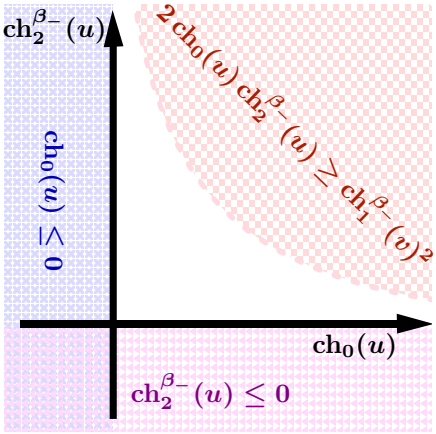
So $0 \leq \Delta(u)$ (condition 2 from Corollary 3.4) yields:

$$2\,\mathrm{ch}_0(u)\,\mathrm{ch}_2^{\beta_-}(u) \leq \mathrm{ch}_1^{\beta_-}(u)^2. \tag{4.1}$$

Furthermore, condition 5 from Corollary 3.4 gives:

$$0 < \mathrm{ch}_1^{\beta_-}(u) < \mathrm{ch}_1^{\beta_-}(v). \tag{4.2}$$

The induced restrictions on possible pairs $\mathrm{ch}_0^{\beta_-}(u)$ and $\mathrm{ch}_2^{\beta_-}(u)$, as well as conditions 1 and 6 from Corollary 3.4 are illustrated here on the right, with the invalid regions shaded.



Currently, the unshaded region in the diagram above, corresponding to possible values for $\mathrm{ch}_0(u)$ and $\mathrm{ch}_2^{\beta_-}(u)$ that satisfy the currently considered restrictions, is unbounded. This is where the rationality of β_- comes in. If $\beta_- = \frac{a_v}{n}$ for some $a_v, n \in \mathbb{Z}$, then we must have $\mathrm{ch}_2^{\beta_-}(u) \in \frac{1}{\mathrm{lcm}(m,2n^2)}\mathbb{Z}$. In particular, since $\mathrm{ch}_2^{\beta_-}(u) > 0$ we have $\mathrm{ch}_2^{\beta_-}(u) \geq \frac{1}{\mathrm{lcm}(m,2n^2)}$, which then in turn gives a bound for the rank of u:

$$\begin{aligned}
\mathrm{ch}_0(u) &\leq \frac{\mathrm{ch}_1^{\beta_-}(u)^2}{2\,\mathrm{ch}_2^{\beta_-}(u)} \\
&\leq \frac{\mathrm{lcm}(m, 2n^2)\,\mathrm{ch}_1^{\beta_-}(u)^2}{2} \\
&= \frac{mn^2\,\mathrm{ch}_1^{\beta_-}(u)^2}{\gcd(m, 2n^2)}.
\end{aligned} \tag{4.3}$$

This can then immediately be bound using Equation (4.2). Alternatively, given that $\mathrm{ch}_1^{\beta_-}(u)$, $\mathrm{ch}_1^{\beta_-}(v) \in \frac{1}{n}\mathbb{Z}$, we can tighten this bound on $\mathrm{ch}_1^{\beta_-}(u)$ given by that equation to:

$$n\,\mathrm{ch}_1^{\beta_-}(u) \leq n\,\mathrm{ch}_1^{\beta_-}(v) - 1,$$

allowing us to bound the expression in Equation (4.3) to the following:

$$\mathrm{ch}_0(u) \leq \frac{m\left(n\,\mathrm{ch}_1^{\beta_-}(v) - 1\right)^2}{\gcd(m, 2n^2)}.$$

□

Example 4.1 ($v = (3, 2\ell, -2)$ on $\mathbb{P}^2$). *Taking $\ell = c_1(\mathcal{O}(1))$ as the standard polarization on $\mathbb{P}^2$, so that $m = 1$, $\beta_- = -\frac{2}{3}$, giving $n = 3$ and $\mathrm{ch}_1^{-\frac{2}{3}}(F) = 4$.*

Using the above Theorem 4.1, we get that the ranks of tilt semistabilisers for v are bounded above by 144. However, when computing all tilt semistabilisers for v on $\mathbb{P}^2$, the maximum rank that appears turns out to be 25. This will be a recurring example to illustrate the performance of later theorems about rank bounds.

Example 4.2 (extravagant example: $\boldsymbol{v = (29, 13\ell, -3/2)}$ on $\mathbb{P}^2$). *Taking $\ell = c_1(\mathcal{O}(1))$ as the standard polarization on $\mathbb{P}^2$, so that $m = 1$, $\beta_- = -\frac{3}{29}$, giving $n = 29$ and $\mathrm{ch}_1^{-\frac{3}{29}}(F) = 16$.*

Using the above Theorem 4.1, we get that the ranks of tilt semistabilisers for v are bounded above by 215296. However, when computing all tilt semistabilisers for v on $\mathbb{P}^2$, the maximum rank that appears turns out to be 49313.

4.2 Tighter Bounds

To get tighter bounds on the rank of solutions u to the Problems 1 and 2, we will need to consider each of the values which $\mathrm{ch}_1^\beta(u)$ can take. Doing this will allow us to eliminate possible values of $\mathrm{ch}_0(u)$ for which each possible value of $\mathrm{ch}_1^\beta(u)$ leads to the failure of at least one of the inequalities, as opposed to only eliminating possible values of $\mathrm{ch}_0(u)$ for which all corresponding $\mathrm{ch}_1^\beta(u)$ fail one of the inequalities (which is what was implicitly happening before in the proof of Theorem 4.1). To pursue this, we shall restate the earlier numerical characterisations of the problems from Lemma 3.3 and Corollary 3.4, in a way which better fits our direction of travel.

Lemma 4.2. *Consider the Problem 1 (or 2), with choice of point $P = (\alpha_0, \beta_0)$ on Θ_v^- (or $\alpha_0 = 0$ and $\beta_0 = \beta_-(v)$ resp.).*
If u is a solution to the Problem then u satisfies:

$$q \coloneqq \mathrm{ch}_1^{\beta_0}(u) \in \left(0, \mathrm{ch}_1^{\beta_0}(v)\right) \qquad \text{and} \qquad \mathrm{ch}_0(u) > \frac{q}{\mu(v) - \beta_0}. \tag{4.4}$$

Conversely, any $u = (r, c\ell, d\ell^2)$ with $r, c \in \mathbb{Z}$ and $d \in \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}$ satisfying the above Equations (4.4) is a solution to the Problem if and only if the following are satisfied:

- $\Delta(u) \geq 0$,
- $\Delta(v - u) \geq 0$,
- $\mathrm{ch}_2^{\alpha_0,\beta_0}(u) \geq 0$.

Proof. Recalling Theorems 3.3 and 3.4, solutions u to the problem are given by $u = (r, c\ell, d\ell^2)$ with $r, c \in \mathbb{Z}$ and $d \in \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}$ which satisfy six numerical conditions. The first line of Equation (4.4) is equivalent to numerical condition 5. The second line is a rearrangement of numerical condition 4, assuming $r > 0$ which is given by the first numerical condition. Therefore any solution u satisfies Equation (4.4).

But then Theorems 3.3 and 3.4, also give that $u = (r, c\ell, d\ell^2)$ with $r, c \in \mathbb{Z}$ and also $d \in \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}$, satisfying Equation (4.4), is in fact a solution provided the remaining numerical conditions 1, 2, 3 and 6 are satisfied. This is in essence the second part of the lemma. □

Corollary 4.3. *Consider the Problem 1 (or 2), with choice of point $P = (\alpha_0, \beta_0)$ on Θ_v^- (or $\alpha_0 = 0$ and $\beta_0 = \beta_-(v)$ resp.), and suppose that β_0 is rational, and written $\beta_0 = \frac{a_v}{n}$ for some coprime integers a_v, n with $n > 0$.*

Then any solution u satisfies:

$$\mathrm{ch}_1^{\beta_0}(u) = \frac{b_q}{n}, \qquad a_v r \equiv -b_q \pmod{n}, \quad \text{and} \qquad r > \frac{q}{\mu(v) - \beta_0},$$

$$\text{for some } b_q \in \left\{1, 2, \ldots, n\,\mathrm{ch}_1^{\beta_0}(v) - 1\right\}.$$

And any $u = (r, c\ell, d\ell^2)$ with $r, c \in \mathbb{Z}$ and $d \in \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}$ satisfying these equations is a solution to the Problem if and only if, again, the following are satisfied:

- $\Delta(u) \geq 0$,
- $\Delta(v-u) \geq 0$,
- $\mathrm{ch}_2^P(u) \geq 0$.

Proof. This is a specialisation of Lemma 4.2 with a modification to the statement

$$q \coloneqq \mathrm{ch}_1^{\beta_0}(u) \in \left(0, \mathrm{ch}_1^{\beta_0}(v)\right),$$

for the case where β_0 is rational. Taking $\beta_0 = \frac{a_v}{n}$ we have:

$$q \coloneqq \mathrm{ch}_1^{\beta_0}(u) = c - \frac{a_v}{n} r \in \frac{1}{n}\mathbb{Z}.$$

So $q = \frac{b_q}{n}$ for some $b_q \in \left\{1, 2, \ldots, n\,\mathrm{ch}_1^{\beta_0}(v) - 1\right\}$ and then $nc - a_v r = b_q$ and so $a_v r \equiv -b_q$ modulo n.

□

4.2.1 Bounds on 'd'-values for Solutions of Problems

Let v be a Chern character with $\Delta(v) \geq 0$, $\mathrm{ch}_0(v) > 0$, or $\mathrm{ch}_0(v) = 0$ and $\mathrm{ch}_1(v) > 0$, and consider Problem 1 or 2. Take $P = (\alpha_0, \beta_0) \in \Theta_v^-$ as the choice made in Problem 1 (or $\beta_0 = \beta_-$ and $\alpha_0 = 0$ for Problem 2). Lemma 4.2 states that any solution

$$u \coloneqq (r, c\ell, d\ell^2) \qquad \text{where } r, c \in \mathbb{Z} \text{ and } d \in \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}, \tag{4.5}$$

to the problem satisfies

$$q \coloneqq \mathrm{ch}_1^{\beta_0}(u) \in \left(0, \mathrm{ch}_1^{\beta_0}(v)\right),$$

and also gives a lower bound for r when considering u with a fixed q. This Section studies the extra numerical conditions that such u must satisfy as given by Lemma 4.2, and will express them as bounds on d in terms of r (for a fixed q). These bounds will later be used in Sections 4.2.2 and 4.2.3 to construct upper bounds on r (in a similar way to how a bound on $\mathrm{ch}_0(u)$ was found in the proof of Theorem 4.1 by considering bounds on $\mathrm{ch}_0^{\beta_0}(u)$ in terms of the former).

Radius of the pseudo-wall: $\mathrm{ch}_2^P(u) > 0$

In the context of Problem 2, this condition, when rearranged to a bound on d, amounts to:

$$\mathrm{ch}_2^{\beta_-}(u) > 0 \qquad \text{and} \qquad d > \frac{1}{2}{\beta_-}^2 r + \beta_- q.$$

In the case where we are tackling Problem 1, with $P = (\alpha_0, \beta_0) \in \Theta_v^-$. Then the equality ${\alpha_0}^2 = \frac{2\,\mathrm{ch}_2^{\beta_0}(v)}{R}$ follows from $\mathrm{ch}_2^P(v) = 0$. Using this as a substitution into the condition $\mathrm{ch}_2^P(u) = -\frac{1}{2}\,{\alpha_0}^2 r + \mathrm{ch}_2^{\beta_0}(u) > 0$ yields:

$$R\mathrm{ch}_2^{\beta_0}(u) - r\mathrm{ch}_2^{\beta_0}(v) > 0. \tag{4.6}$$

Expanding $\mathrm{ch}_2^{\beta_0}(u)$ in terms of r, c, d, and rearranging for d then yields:

$$d > \frac{1}{2}\,{\beta_0}^2 r + \beta_0 q + \frac{r\mathrm{ch}_2^{\beta_0}(v)}{R}. \tag{4.7}$$

Semistability of the Semistabiliser: $\Delta(u) \geq 0$

Expressing $\Delta(u) \geq 0$ in terms of $q = \mathrm{ch}_1^{\beta_0}(u) = c - r\beta_0$, we get:

$$0 \leq (\beta_0 r + q)^2 - 2\,dr. \tag{4.8}$$

Rearranging to express this as a bound on d, we get the following. Recall that $r > 0$ is ensured by Equations (4.4).

$$d \leq \frac{1}{2}\,{\beta_0}^2 r + \beta_0 q + \frac{q^2}{2\,r} \tag{4.9}$$

Semistability of the Quotient: $\Delta(v-u) \geq 0$

Expressing $\Delta(v-u) \geq 0$ in term of q and rearranging as a bound on d yields:

$$d \leq \frac{1}{2}\,{\beta_0}^2 r + \beta_0 q + \mathrm{ch}_2^{\beta_0}(v) - \frac{\left(q - \mathrm{ch}_1^{\beta_0}(v)\right)^2}{2\,(R-r)}, \qquad \text{when } r > R. \tag{4.10}$$

If $r = R$, then $\Delta(v-u) = (C-c)^2 \geq 0$ is always true, and for $r < R$ the expression on the right hand side of Equation (4.10) gives a lower bound for d instead. However it is weaker than lower bound given by $\mathrm{ch}_2^P(u) > 0$ if u already satisfies Equations (4.4) as will be shown now:

Since $r, R - r > 0$, we have:

$$\max(\beta_0, \mu(u)) < \mu(v) < \mu(v-u). \tag{4.11}$$

The first inequality coming from $P \in \Theta_v^-$ and Equation (4.4), and the second inequality following by the see-saw principle.

$$\begin{aligned}
\left(\frac{\mathrm{ch}_1^{\beta_0}(v-u)}{\mathrm{ch}_0(v-u)}\right)^2 &= (\mu(v-u) - \beta_0)^2 \\
&> (\mu(v) - \beta_0)^2 && \text{by Equation (4.11)} \\
&= \left(\frac{\mathrm{ch}_1^{\beta_0}(v)}{\mathrm{ch}_0(v)}\right)^2 \\
&\geq 2\frac{\mathrm{ch}_2^{\beta_0}(v)}{\mathrm{ch}_0(v)} && \text{since } \Delta(v) \geq 0 \text{ and } \mathrm{ch}_0(v) > 0
\end{aligned}$$

$$\text{So} \quad \frac{\left(q - \mathrm{ch}_1^{\beta_0}(v)\right)^2}{(R-r)^2} > 2\frac{\mathrm{ch}_2^{\beta_0}(v)}{R} \qquad \text{and} \quad \mathrm{ch}_2^{\beta_0}(v) - \frac{\left(q - \mathrm{ch}_1^{\beta_0}(v)\right)^2}{2\,(R-r)} < \frac{r\,\mathrm{ch}_2^{\beta_0}(v)}{R},$$

showing that the unique terms of Equation (4.7) are greater than those of Equation (4.10).

All Bounds on d Together for Problem 2

In the context of Problem 2, with $\beta_0 = \beta_-(v)$, we have $\mathrm{ch}_2^{\beta_-(v)}(v) = 0$, simplifying the bounds on d calculated in this Section. So we conclude that final 3 conditions from Corollary 4.3 for a potential solution to the problem of the form in Equation (4.5), amounts to the following:

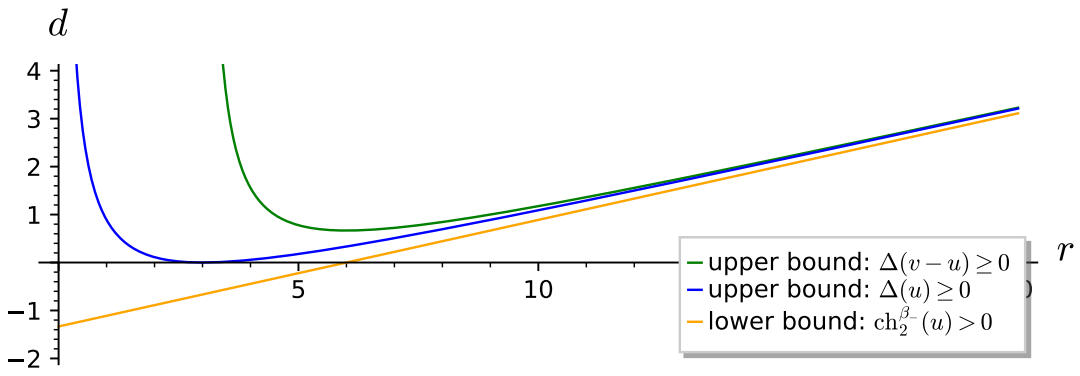


Figure 4.1: Bounds on $d := \mathrm{ch}_2(u)$ in terms of $r := \mathrm{ch}_0(u)$ for a fixed value $\mathrm{ch}_1^{\beta}(v)/2$ of $q := \mathrm{ch}_1^{\beta}(u)$. Where $\mathrm{ch}(v) = (3, 2\ell, -2\ell^2)$.

$$d > \frac{1}{2}{\beta_0}^2 r + \beta_0 q, \qquad \text{when } r > 0, \tag{4.12}$$

$$d \leq \frac{1}{2}\,{\beta_0}^2 r + \beta_0 q + \frac{q^2}{2\,r}, \qquad \text{when } r > 0, \tag{4.13}$$

$$d \leq \frac{1}{2}\,{\beta_0}^2 r + \beta_0 q - \frac{\left(\mathrm{ch}_1^{\beta_0}(v) - q\right)^2}{2\,(R-r)}, \qquad \text{when } r > R. \tag{4.14}$$

Recalling that $q := \mathrm{ch}_1^{\beta}(u) \in (0, \mathrm{ch}_1^{\beta}(v))$, it is worth noting that the extreme values of q in this range lead to the tightest bounds on d. Small values of q brings Equation (4.13) closer to Equation (4.12), and larger values of q brings Equation (4.14) closer to Equation (4.12).

For a generic case, when $0 < q := \mathrm{ch}_1^{\beta}(u) < \mathrm{ch}_1^{\beta}(v)$, the bounds on d in terms of r is illustrated in Figure (4.1). The question of whether there are pseudo-semistabilisers of arbitrarily large rank, in the context of the graph, comes down to whether there are points $(r, d) \in \mathbb{Z} \oplus \frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z}$ (with large r) that fit above the yellow line (ensuring positive radius of wall) but below the blue and green (ensuring $\Delta(u), \Delta(v-u) > 0$). These lines have the same assymptote at $r \to \infty$ (Equations (4.13), (4.14), (4.12)). As mentioned in the introduction to this Part, the finiteness of these solutions is entirely determined by whether β_- is rational or irrational. This will be pursued in Section 4.2.3.

All Bounds on d Together for Problem 1

Unlike for Problem 2, the bounds on $d = \mathrm{ch}_2(u)$ induced by the final three conditions of Lemma 4.2 have different constant and linear terms, so that the graphs for upper bounds do not share the same assymptote as the lower bound (and they will turn out to intersect).

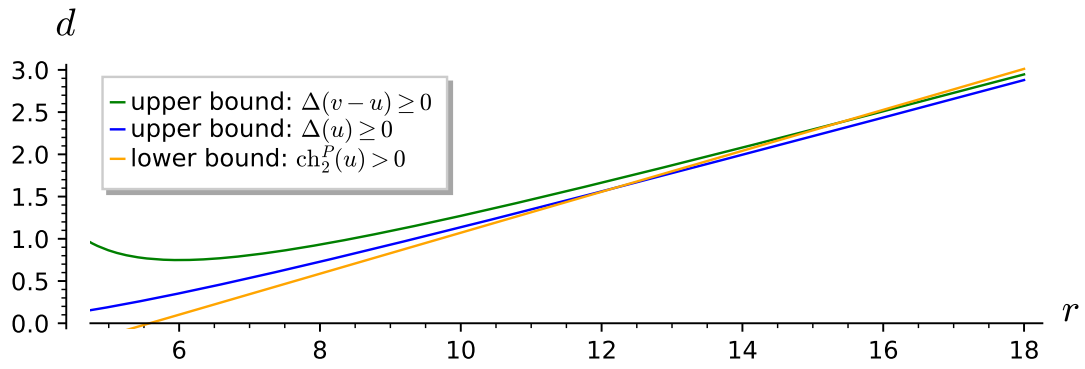


Figure 4.2: Bounds on $d := \mathrm{ch}_2(u)$ in terms of $r := \mathrm{ch}_0(u)$ for a fixed value $\mathrm{ch}_1^\beta(v)/2$ of $q := \mathrm{ch}_1^\beta(E)$. Where $\mathrm{ch}(v) = (3, 2\ell, -2\ell^2)$ and P is chosen as the point on Θ_v such that $\beta(P) := -2/3 - 1/99$ in the context of Problem 1.

$$d > \frac{1}{2}{\beta_0}^2 r + \beta_0 q + \frac{r\mathrm{ch}_2^{\beta_0}(v)}{R}, \qquad \text{when } r > 0, \tag{4.15}$$

$$d \leq \frac{1}{2}{\beta_0}^2 r + \beta_0 q + \frac{q^2}{2\,r}, \qquad \text{when } r > R, \tag{4.16}$$

$$d \leq \frac{1}{2}{\beta_0}^2 r + \beta_0 q + \mathrm{ch}_2^{\beta_0}(v) - \frac{\left(q - \mathrm{ch}_1^{\beta_0}(v)\right)^2}{2\,(R-r)}, \qquad \text{when } r > R. \tag{4.17}$$

Notice that as functions of r, the linear term in Equation (4.15) is strictly greater than those in Equations (4.16) and (4.17) in the context of the Problem. This is because $R := \mathrm{ch}_0(v)$ and $\mathrm{ch}_2^{\beta_0}(v)$ are all strictly positive:

- $R > 0$ from the setting of Problem 1.
- $\mathrm{ch}_2^{\beta_0}(v) > 0$ by Lemma 1.12 because $\beta_0 < \beta_-$ due to the choice of P being a point on Θ_v^-.

This means that the lower bound for d will be larger than either of the two upper bounds for sufficiently large r, and hence those values of r would yield no solution to Problem 1. A generic example of this is plotted in Figure 4.2, and bounds for r following from this idea will be pursued in Section 4.2.2.

4.2.2 Bounds on Semistabiliser Rank r in Problem 1

As discussed at the end of Section 4.2.1 (and illustrated in Figure 4.2), there are no solutions u to Problem 1 with large $r = \mathrm{ch}_0(u)$, since the lower bound on $d = \mathrm{ch}_2(u)$ is larger than the upper bounds. Therefore, we can calculate upper bounds on r by calculating for which values, the lower bound on d is equal to one of the upper bounds on d (i.e. finding certain intersection points of the graph in Figure 4.2).

Theorem 4.4 (Problem 1 upper Bound on r). *Let u be a solution to Problem 1 and take $q \coloneqq \mathrm{ch}_1^B(u)$. Then $r \coloneqq \mathrm{ch}_0(u)$ is bounded above by the following expression:*

$$\frac{\sqrt{2}\sqrt{R}\min\left(q, \sqrt{2}\sqrt{R}\sqrt{\mathrm{ch}_2^{\beta_0}(v)} - q + \mathrm{ch}_1^{\beta_0}(v)\right)}{2\sqrt{\mathrm{ch}_2^{\beta_0}(v)}}. \tag{4.18}$$

Proof. Lemma 4.2 gives us that any solution u must be of the form in Equation (4.5) and satisfy Equations (4.4) as well as the three conditions $\mathrm{ch}_2^P(u) > 0$, $\Delta(u) \geq 0$, and $\Delta(v-u) \geq 0$. Section 4.2.1 equates these latter three conditions (provided Equations (4.4)) to upper bounds on d given by Equations (4.16) and (4.17); and one lower bound given by Equation (4.15).

Solving for the lower bound in Equation (4.15) being less than the upper bound in Equation (4.16) yields:

$$r < \sqrt{\frac{1}{2}}q\sqrt{\frac{R}{\mathrm{ch}_2^{\beta_0}(v)}}. \tag{4.19}$$

Similarly, but with the upper bound in Equation (4.17), gives:

$$r < -\frac{\sqrt{2}\sqrt{R}q}{2\sqrt{\mathrm{ch}_2^{\beta_0}(v)}} + \frac{\sqrt{2}\sqrt{R}\mathrm{ch}_1^{\beta_0}(v)}{2\sqrt{\mathrm{ch}_2^{\beta_0}(v)}} + R. \tag{4.20}$$

Therefore, r is bounded above by the minimum of these two expressions which can then be factored into the expression given in the Lemma.

□

The above Lemma 4.4 gives an upper bound on r in terms of q. But given that we have $0 < q < \mathrm{ch}_1^{\beta_0}(v)$, we can take the maximum of this bound, over q in this range, to get a simpler (but weaker) bound in the following Lemma 4.5.

Theorem 4.5 (Problem 1 global upper Bound on r). *Let u be a solution to Problem 1. Then $r \coloneqq \mathrm{ch}_0(u)$ is bounded above by the following expression:*

$$\frac{\sqrt{2}\sqrt{R}\mathrm{ch}_1^{\beta_0}(v)}{4\sqrt{\mathrm{ch}_2^{\beta_0}(v)}} + \frac{1}{2}R. \tag{4.21}$$

Proof. The first term of the minimum in Lemma 4.4 increases linearly in q, and the second decreases linearly. So the maximum is achieved with the value of $q = q_{\max}$ where they are equal. Solving for the two terms in the minimum to be equal yields:

$$q_{\max} = \frac{1}{2}\sqrt{2}\sqrt{R}\sqrt{\mathrm{ch}_2^{\beta_0}(v)} + \frac{1}{2}\mathrm{ch}_1^{\beta_0}(v).$$

Substituting $q = q_{\max}$ into the bound in Lemma 4.4 gives the bound as stated in the current lemma.

□

Remark. $q_{\max} > 0$ is immediate from the expression, but $q_{\max} \leq \mathrm{ch}_1^B(v)$ is equivalent to $\Delta(v) \geq 0$, which is true by assumption in this setting.

4.2.3 Bounds on Semistabiliser Rank r in Problem 2

Now, the inequalities from the above Section 4.2.1 will be used to find, for each given value of $q = \mathrm{ch}_1^{\beta}(E)$, how large r needs to be in order to leave no possible solutions for d in the context of Problem 2. At that point, there are no solutions $u = (r, c\ell, d\ell^2)$ to the Problem.

In the context of Problem 2, $\beta_-(v)$ is assumed to be rational. Considering Corollary 4.3, for any solution u, we have $q := \mathrm{ch}^{\beta_-(v)}(u) = \frac{b_q}{n}$ where $\beta_-(v) = \frac{a_v}{n}$ in lowest terms and b_q is an integer between 1 and $n\,\mathrm{ch}_1^{\beta_0}(v) - 1$ (inclusive), and $a_v r \equiv -b_q \pmod{n}$. The Corollary then gives a lower bound for r, and states that any u of the form from Equation (4.5) satisfying these conditions so far, is a solution to Problem 2 if and only if it satisfies the conditions $\mathrm{ch}^{\beta_-(v)}(u) > 0$, $\Delta(u) \geq 0$, and $\Delta(v-u) \geq 0$.

Substituting more specialised values of q and $\beta_0 = \beta_-(v)$ into the condition $\mathrm{ch}^{\beta_0}(u) > 0$ (Equation (4.12)) we get:

$$\frac{1}{\mathrm{lcm}(m,2)}\mathbb{Z} \ni \quad d > \frac{(a_v r + 2\,b_q)a_v}{2\,n^2} \quad \in \frac{1}{2n^2}\mathbb{Z}. \tag{4.22}$$

This fact will be leveraged to give tighter lower bounds in a similar way to the proof of Theorem 4.1.

Theorem 4.6 (First bound on r for Problem 2)**.** *Let X be a smooth projective surface with Picard rank 1 and choice of ample line bundle L with $c_1(L)$ generating $\mathrm{NS}(X)$ and $m := \ell^2$. Let v be a fixed Chern character on this surface with positive rank (or rank 0 and $c_1(v) > 0$), and $\Delta(v) \geq 0$. Then the ranks of the pseudo-semistabilisers u for v, which are solutions to Problem 2, with $\mathrm{ch}_1^{\beta_-(v)}(u) = q$ are bounded above by the following expression.*

$$\min\left(\frac{1}{2}\,\mathrm{lcm}(m,2n^2)q^2,\ \frac{1}{2}\left(\mathrm{ch}_1^{\beta_-}(v) - q\right)^2 \mathrm{lcm}(m,2n^2) + R\right),$$

where $R := \mathrm{ch}_0(v)$.

Proof. Both d and the lower bound in (Equation (4.22)) are elements of $\frac{1}{\mathrm{lcm}(m,2n^2)}\mathbb{Z}$. So, if any of the two upper bounds on d come to within $\epsilon_v := \frac{1}{\mathrm{lcm}(m,2n^2)}$ of this lower bound, then there are no solutions for d. Hence any corresponding r cannot be a rank of a pseudo-semistabiliser for v.

To avoid this, we must have, considering Equations (4.13), (4.14), (4.12):

$$\epsilon_v = \frac{1}{\mathrm{lcm}(m,2n^2)} < \frac{q^2}{2\,r}, \tag{4.23}$$

$$\epsilon_v = \frac{1}{\mathrm{lcm}(m,2n^2)} < -\frac{\left(\mathrm{ch}_1^{\beta_0}(v) - q\right)^2}{2\,(R-r)}. \tag{4.24}$$

This is equivalent to:

$$r \leq \min\left(\frac{1}{2}\,\mathrm{lcm}(m,2n^2)q^2, \frac{1}{2}\left(\mathrm{ch}_1^{\beta_0}(v) - q\right)^2 \mathrm{lcm}(m,2n^2) + R\right). \tag{4.25}$$

□

Corollary 4.7 (Second, global bound on r for Problem 2)**.** *Let X be a smooth projective surface with Picard rank 1 and choice of ample line bundle L with $c_1(L)$ generating $\mathrm{NS}(X)$ and $m := \ell^2$.*

Let v be a fixed Chern character on this surface and $R \coloneqq \mathrm{ch}_0(v)$. Then the ranks of the pseudo-semistabilisers u of v, which are solutions to Problem 2, are bounded above as follows:

$$r \leq \frac{1}{2}R + \frac{\Delta(v)\mathrm{lcm}(m,2n^2)}{8\,m} + \frac{R^2 m}{2\,\Delta(v)\mathrm{lcm}(m,2n^2)}, \qquad \text{if } R < \frac{\Delta(v)\,\mathrm{lcm}(m,2n^2)}{2m},$$
$$r \leq \frac{\Delta(v)\,\mathrm{lcm}(m,2n^2)}{2m}, \qquad \text{if } R \geq \frac{\Delta(v)\,\mathrm{lcm}(m,2n^2)}{2m}.$$

Proof. The ranks of the pseudo-semistabilisers for v are bounded above by the maximum over $q \in [0, \mathrm{ch}_1^{\beta_-}(v)]$ of the expression in Theorem 4.6. Noticing that the expression is a maximum of two quadratic functions in q ($\beta_0 = \beta_-(v)$ in this context):

$$f_1(q) \coloneqq \frac{1}{2}\,\mathrm{lcm}(m,2n^2)q^2, \qquad f_2(q) \coloneqq \frac{1}{2}\left(\mathrm{ch}_1^{\beta_0}(v) - q\right)^2 \mathrm{lcm}(m,2n^2) + R.$$

These have their minimums at $q = 0$ and $q = \mathrm{ch}_1^{\beta_-}(v)$ respectively, with values 0 and $R > 0$ respectively. So provided that $f_2\left(\mathrm{ch}_1^{\beta_-}(v)\right) < f_1\left(\mathrm{ch}_1^{\beta_-}(v)\right)$, the maximum is achieved at their intersection. Otherwise, the maximum is achieved at $\mathrm{ch}_1^{\beta_-}(v)$. So we have that

$$r \leq f_1(q_{\max}), \qquad \text{if } f_2\left(\mathrm{ch}_1^{\beta_-}(v)\right) < f_1\left(\mathrm{ch}_1^{\beta_-}(v)\right),$$
$$\text{where } q_{\max} \text{ is the } q\text{-value where the } f_i \text{ intersect,}$$
$$\text{and } r \leq f_1\left(\mathrm{ch}^{\beta_-}(v)\right), \qquad \text{if } f_2\left(\mathrm{ch}_1^{\beta_-}(v)\right) \geq f_1\left(\mathrm{ch}_1^{\beta_-}(v)\right).$$

In the first case, solving for $f_1(q) = f_2(q)$ yields

$$q = \frac{1}{2}\,\mathrm{ch}_1^{\beta_0}(v) + \frac{R}{\mathrm{ch}_1^{\beta_0}(v)\mathrm{lcm}(m,2n^2)}.$$

Evaluating f_1 at this q-value gives:

$$\frac{1}{8}\,\mathrm{ch}_1^{\beta_0}(v)^2\mathrm{lcm}(m,2n^2) + \frac{1}{2}\,R + \frac{R^2}{2\,\mathrm{ch}_1^{\beta_0}(v)^2\mathrm{lcm}(m,2n^2)}.$$

Finally, noting that $\Delta(v) = \left(\mathrm{ch}_1^{\beta_-(v)}(v)\right)^2 \ell^2$, we get the bounds as stated in the statement of the corollary.

□

Example 4.3 ($v = (3, 2\ell, -2)$ on $\mathbb{P}^2$). *Just like in Example 4.1, take $\ell = c_1(\mathcal{O}(1))$ as the standard polarization on $\mathbb{P}^2$, so that $m = 2$, $\beta = -\frac{2}{3}$, giving $n = 3$.*

Using the above Corollary 4.7, we get that the ranks of tilt semistabilisers for v are bounded above by $\frac{2401}{64} \approx 37.5$, which is much closer to real maximum 25 than the original bound 144.

Example 4.4 (extravagant example: $v = (29, 13\ell, -3/2)$ on $\mathbb{P}^2$). *Just like in Example 4.2, take $\ell = c_1(\mathcal{O}(1))$ as the standard polarization on $\mathbb{P}^2$, so that $m = 2$, $\beta = -\frac{3}{29}$, giving $n = 29$.*

Using the above Corollary 4.7, we get that the ranks of tilt semistabilisers for v are bounded

> *above by* $\frac{55130625}{1024} \approx 53838.5$*, which is much closer to real maximum* 49313 *than the original bound 215296.*

These bound can be refined a bit more by considering restrictions from the possible values that r take. Furthermore, the proof of Theorem 4.6 uses the fact that, given an element of $\frac{1}{2n^2}\mathbb{Z}$, the closest non-equal element of $\frac{1}{2}\mathbb{Z}$ is at least $\frac{1}{2n^2}$ away. However this a conservative estimate, and a larger gap can sometimes be guaranteed if we know this value of $\frac{1}{2n^2}\mathbb{Z}$ explicitly.

The expressions that will follow will be a bit more complicated and have more parts which depend on the values of q and β, even their numerators a_v, b_q specifically. The upcoming Theorem 4.8 is less useful as a 'clean' formula for a bound on the ranks of the pseudo-semistabilisers, but has a purpose in the context of writing a computer program to find pseudo-semistabilisers. Such a program would iterate through possible values of q, then iterate through values of r within the bounds (dependent on q), which would then determine c, and then find the corresponding possible values for d.

Next, we seek to find a larger ϵ to use in place of ϵ_v in the proof of Theorem 4.6:

Lemma/Definition 4.0.1 (A better alternative to $\boldsymbol{\epsilon_v}$: $\boldsymbol{\epsilon_{v,q}}$)**.** Suppose $d \in \frac{1}{\mathrm{lcm}(m,2n^2)}\mathbb{Z}$ satisfies the condition in Equation (4.22). That is:

$$d > \frac{(a_v r + 2\,b_q)a_v}{2\,n^2} \qquad \text{for some integers } a_v, b_q, n \text{ with } (a_v, n) = 1,$$

and r satisfies $a_v r + b_q \equiv 0 \pmod{n}$, then we have:

$$d - \frac{(a_v r + 2b_q)a_v}{2n^2} \geq \epsilon_{v,q} \geq \epsilon_v > 0, \tag{4.26}$$

where $\epsilon_{v,q}$ is defined as follows:

$$\epsilon_{v,q} \coloneqq \frac{k_q}{\mathrm{lcm}(m, 2n^2)},$$

with $k_{v,q}$ being the least $k \in \mathbb{Z}_{>0}$ satisfying

$$k \equiv -a_v b_q \frac{m}{\gcd(m, 2n^2)} \mod \gcd\left(\frac{n^2\gcd(m,2)}{\gcd(m,2n^2)}, \frac{mna_v}{\gcd(m,2n^2)}\right).$$

Remark. The quantity m is determined by the variety, whereas a_v and n are determined by the Chern character v for which we are trying to find pseudo-semistabilisers. So the gcd expression we are taking the modulus with respect to is considered constant in the context of the problem we are solving for (i.e. Problem 2). However b_q depends on the choice of q, that is the value of $\mathrm{ch}_1^{\beta_-(v)}(u)$ for which we are searching for solutions u, hence why $k_{v,q}$ is denoted to depend on q on top of v and the context of the problem.

Proof. Consider the following sequence of logical implications. The one-way implication follows from $a_v r + b_q \equiv 0 \pmod{n}$, and the final logical equivalence is just a simplification of the expressions.

$$\frac{x}{\operatorname{lcm}(m,2)} - \frac{(a_v r + 2b_q)a_v}{2n^2} = \frac{k}{\operatorname{lcm}(m,2n^2)} \quad \text{for some } x \in \mathbb{Z} \tag{4.27}$$

$$\begin{aligned}
\Leftrightarrow \quad -(a_v r + 2b_q)a_v \frac{\operatorname{lcm}(m,2n^2)}{2n^2} &\equiv k \\
&\mod \frac{\operatorname{lcm}(m,2n^2)}{\operatorname{lcm}(m,2)} \\
\Rightarrow \quad -b_q a_v \frac{\operatorname{lcm}(m,2n^2)}{2n^2} &\equiv k \\
&\mod \gcd\left(\frac{\operatorname{lcm}(m,2n^2)}{\operatorname{lcm}(m,2)}, \frac{n a_v \operatorname{lcm}(m,2n^2)}{2n^2}\right) \\
\Leftrightarrow \quad -b_q a_v \frac{m}{\gcd(m,2n^2)} &\equiv k \\
&\mod \gcd\left(\frac{n^2 \gcd(m,2)}{\gcd(m,2n^2)}, \frac{mna_v}{\gcd(m,2n^2)}\right)
\end{aligned} \tag{4.28}$$

In our situation, we want to find the least $k > 0$ satisfying Equation (4.27). Since such a k must also satisfy Equation (4.28), we can pick the smallest $k_{q,v} \in \mathbb{Z}_{>0}$ which satisfies this new condition (a computation only depending on q and β, but not r). We are then guaranteed that $k_{v,q}$ is less than any k satisfying Equation (4.27), giving the first inequality in Equation (4.26). Furthermore, $k_{v,q} \geq 1$ gives the second part of the inequality: $\epsilon_{v,q} \geq \epsilon_v$, with equality when $k_{v,q} = 1$.

□

Theorem 4.8 (Third bound on r for Problem 2)**.** *Let X be a smooth projective surface with Picard rank 1 and choice of ample line bundle L with $c_1(L)$ generating $\mathrm{NS}(X)$ and $m := \ell^2$. Let v be a fixed Chern character on this surface with positive rank (or rank 0 and $c_1(v) > 0$), and $\Delta(v) \geq 0$. Then the ranks of the pseudo-semistabilisers u for v, which are solutions to Problem 2, with $\mathrm{ch}_1^{\beta_-(v)}(u) = q$ are bounded above by the following expression:*

$$\min\left(\frac{\operatorname{lcm}(m,2n^2)q^2}{2\,k_{v,q}}, \frac{\left(\mathrm{ch}_1^{\beta_-}(v) - q\right)^2 \operatorname{lcm}(m,2n^2)}{2\,k_{v,q}} + R\right),$$

where $k_{v,q}$ is defined as in Definition/Lemma 4.0.1, and $R = \mathrm{ch}_0(v)$.

Proof. Following the same proof as Theorem 4.6, $\epsilon_{v,q} = \frac{k_{v,q}}{\operatorname{lcm}(m,2n^2)}$ can be used instead of $\epsilon_v = \frac{1}{\operatorname{lcm}(m,2n^2)}$ as it satisfies the same required property, as per Definition/Lemma 4.0.1.

□

Although the general form of this bound is quite complicated, it does simplify a lot when m is small.

Corollary 4.9 (Third bound on r on $\mathbb{P}^2$ and principally polarised abelian surfaces)**.** *Suppose we are working over $\mathbb{P}^2$ or a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ (or any other surfaces with $m = \ell^2 = 1$ or 2). Let v be a fixed Chern character, with $\beta_- := \beta_-(v) = \frac{a_v}{n}$ rational and expressed in lowest terms. Then the ranks r of the pseudo-semistabilisers u for v*

with, which are solutions to Problem 2, $\mathrm{ch}_1^{\beta-}(u) = q = \frac{b_q}{n}$ are bounded above by the following expression:

$$\min\left(\frac{n^2q^2}{k_{v,q}},\ \frac{\left(\mathrm{ch}_1^{\beta-}(v)-q\right)^2n^2}{k_{v,q}}+R\right),$$

where $R = \mathrm{ch}_0(v)$ and $k_{v,q}$ is the least $k \in \mathbb{Z}_{>0}$ satisfying $k \equiv -a_v b_q \pmod n$.

Proof. This is a specialisation of Theorem 4.8, where we can drastically simplify the lcm and gcd terms by noting that m divides both 2 and $2n^2$, and that a_v is coprime to n. □

Example 4.5 ($v = (3, 2\ell, -2)$ on $\mathbb{P}^2$). *Just like in Examples 4.1 and 4.3, we shall take $\ell = c_1(\mathcal{O}(1))$ as the standard polarization on $\mathbb{P}^2$, so that $\beta = -\frac{2}{3}$, giving $n = 3$ and $\mathrm{ch}_1^{-\frac{2}{3}}(F) = 4$. The (non-exclusive) upper bounds for $r := \mathrm{ch}_0(u)$ of a tilt semistabiliser u of v in terms of the possible values for $q := \mathrm{ch}_1^{\beta-}(u)$ are as follows.*

$q = \mathrm{ch}_1^{\beta-}(u)$	0	$\frac{1}{3}$	$\frac{2}{3}$	1	$\frac{4}{3}$	$\frac{5}{3}$	2	$\frac{7}{3}$	$\frac{8}{3}$	3	$\frac{10}{3}$	$\frac{11}{3}$	4
Theorem 4.6	0	1	4	9	16	25	36	28	19	12	7	4	3
Theorem 4.8	0	0	4	3	8	25	12	15	19	6	5	4	3

It is worth noting that the bounds given by Theorem 4.8 reach, but do not exceed, the actual maximum rank 25 of the pseudo-semistabilisers of v in this case. As a reminder, the original loose bound from Theorem 4.1 was 144.

Example 4.6 (extravagant example: $v = (29, 13\ell, -3/2)$ on $\mathbb{P}^2$). *Just like in examples 4.2 and 4.4, take $\ell = c_1(\mathcal{O}(1))$ as the standard polarization on $\mathbb{P}^2$, so that $\beta = -\frac{3}{29}$, giving $n = 29$ and $\mathrm{ch}_1^{-\frac{3}{29}}(F) = 16$. This example was chosen because the n value is moderatly large, giving more possible values for $k_{v,q}$, in Definition/Lemma 4.0.1. This allows for a larger possible difference between the bounds given by Theorems 4.6 and 4.8, with the bound from the second being up to 29 times smaller, for any given q value. The (non-exclusive) upper bounds for $r := \mathrm{ch}_0(u)$ of a tilt semistabiliser u of v in terms of the first few smallest possible values for $q := \mathrm{ch}_1^{\beta}(u)$ are as follows.*

$q = \mathrm{ch}_1^{\beta}(u)$	0	$\frac{1}{29}$	$\frac{2}{29}$	$\frac{3}{29}$	$\frac{4}{29}$	$\frac{5}{29}$	$\frac{6}{29}$	$\frac{7}{29}$	$\frac{8}{29}$	$\frac{9}{29}$	$\frac{10}{29}$	$\frac{11}{29}$	$\cdots$
Theorem 4.6	0	1	4	9	16	25	36	49	64	81	100	121	$\cdots$
Theorem 4.8	0	0	0	1	1	1	2	2	2	3	100	30	$\cdots$

However the reduction in the overall bound on r is not as drastic, since all possible values for $k_{v,q}$ in $\{1, 2, \ldots, 29\}$ are iterated through cyclically as we consider successive possible values for q. And for each q where $k_{v,q} = 1$, both theorems give the same bound. Calculating the maximums over all values of q yields 53824 for Theorem 4.6, and 49313 for Theorem 4.8.

Chapter 5

Computing Solutions to Problems 1 and 2

5.1 Existing Pseudo-Wall Finding Method

The SageMath Library [Sch20b] provides functions which calculates all solutions to Problems 1 or 2. Here is an outline of the algorithm which does this. Simplifications will be made in the presentation to concentrate on the case we are interested in: Problem 2, finding all pseudo-walls when $\beta_-(v) \in \mathbb{Q}$ for a v with positive rank. In the next Section 5.2 from the previous chapter, a different algorithm will be presented making use of theorems from Section 4.2 with the goal of cutting down the run time.

Finding possible r and c

To do this, first calculate the upper bound $r_{\max}$ on the ranks of tilt semistabilisers, as given by Theorem 4.1.

Recalling Consequence 2 of Lemma 3.2, we can iterate through the possible values of $\mu(u) = \frac{c}{r}$ taking a decreasing sequence of all fractions between $\mu(v)$ and β_-, whos denominators are no large than $r_{\max}$ (giving a finite sequence). This can be done with Farey sequences [Niv66, Chapter 6], for which there exist formulae to generate.

These $\mu(u)$ values determine pairs r, c up to multiples, we can then take all multiples which satisy $0 < r \leq r_{\max}$.

We now have a finite sequence of pairs r, c for which there might be a solution $(r, c\ell, d\ell^2)$ to our problem. In particular, any $(r, c\ell, d\ell^2)$ satisfies Consequence 2 of Lemma 3.2, and the positive rank condition. What remains is to find the d values which satisfy the Bogomolov inequalities and Consequence 3 of Lemma 3.2 ($\mathrm{ch}_2^{\beta-}(u) > 0$).

Finding d for fixed r and c

$\Delta(u) \geq 0$ induces an upper bound $\frac{c^2}{2r}$ on d, and the $\mathrm{ch}_2^{\beta-}(u) > 0$ condition induces a lower bound on d. The values in the range can be tested individually, to check that the rest of the conditions are satisfied.

5.1.1 Limitations

The main downside of this algorithm is that many r,c pairs which are tested end up not yielding any solutions for the problem. In fact, solutions u to our problem with high rank must

Listing 5.1: `tilt_stability::left_pseudo_semistabilizers::considered_b_for_beta`

```rust
/// Returns i64 iterator which gives all b s.t.
///   b/n ∈ (0, ch₁^β(v)) ∩ 1/n ℤ     where n = β's denominator
///
/// These correspond to all possible values ch₁^β(u) = b/n can take if u is a
/// pseudo-semistabilizer for v at some stability condition with this fixed β value.
fn considered_b_for_beta<const M: u32>(
    v: &ChernChar<M>,
    β: &Rational64,
) -> impl Iterator<Item = i64> {
    let c: i64 = v.c.into();
    let r: i64 = v.r;
    1..(β.denom() * c - β.numer() * r)
}
```

have $\mu(u)$ close to $\beta_-(v)$:

$$0 \leq \mathrm{ch}_1^{\beta_-}(u) \leq \mathrm{ch}_1^{\beta_-}(u),$$
$$0 \leq \mu(u) - \beta_- \leq \frac{\mathrm{ch}_1^{\beta_-}(v)}{r}.$$

In particular, it is the $\mathrm{ch}_1^{\beta_-}(v-u) \geq 0$ condition which fails for r,c pairs with large r and $\frac{c}{r}$ too far from β_-. This condition is only checked within the internal loop. This, along with a conservative estimate for a bound on the r values (as illustrated in Example 4.1) occasionally leads to slow computations.

Here are some benchmarks to illustrate the performance benefits of the alternative algorithm which will later be described in Section 5.2.

Choice of v on $\mathbb{P}^2$	$(3, 2\ell, -2)$	$(3, 2\ell, -\frac{15}{2})$
[Sch20b, `tilt.walls_left`] exec time	∼20s	>1hr
[Nay23b] exec time	∼50ms	∼50ms

5.2 Computing Solutions to Problem 2

Alongside this thesis, there is a library [Nay23b] to compute the solutions to Problem 2, using the theorems above. The source code is also shown in Appendix A, but is better viewed digitally from source, or via the documentation [Nay24b]

5.2.1 Algorithm

The algorithm yields solutions $u = (r, c\ell, d\ell^2)$ to the problem as follows.

Iterating Over Possible $q = \mathrm{ch}^{\beta_-}(u)$

Given a Chern character v, the domain of the problem are first verified: that v has positive rank, that it satisfies $\Delta(v) \geq 0$, and that $\beta_-(v)$ is rational. Take $\beta_-(v) = \frac{a_v}{n}$ in simplest terms. Iterate over $q = \frac{b_q}{n} \in (0, \mathrm{ch}_1^{\beta_-}(v)) \cap \frac{1}{n}\mathbb{Z}$. The code used to generate the corresponding values for b_q is shown in Listing 5.1.

Lemma 4.3 gives us any solution u has $\mathrm{ch}_1^{\beta_-}(u) = q = \frac{b_q}{n}$ for one of the values through which we are iterating. We can therefore reduce the problem of finding solutions to the

Listing 5.2: `tilt_stability::left_pseudo_semistabilizers::find_all`

```
81  /// Given a ChernChar v,
82  /// returns an iterator over all possible pseudo-semistabilizers of v
83  /// giving pseudo-walls for of the vertical line β=μ(v)
84  /// if β_ is rational,
85  ///
86  /// otherwise returns Error.
87  pub fn find_all<const M: u32>(
88      v: &ChernChar<M>,
89  ) -> Result<Vec<ChernChar<M>>, &'static str>
90  {
91      let problem_data = problem_2_data(v)?;
92
93      Ok(considered_b_for_beta(problem_data.v(), problem_data.beta())
94          .map(|b|
95              // fix ch, β  = b/n ℓ of pseudo-semistabilizer
96              fixed_q_beta::ProblemData{b, beta_data: &problem_data}
97              .find_all()
98          ).try_collect::<Vec<_>>()?
99          .concat()
100     )
101 }
```

more specialised problem of finding the solutions u with each fixed possible $q = \mathrm{ch}_1^{\beta}(u)$ (i.e. choice of b). The code representing this appears in Listing 5.2. Line 16 refers to creating an objects representing the context the specialised problem for the fixed q value, with the next line 'solving' the specialised problem, which is defined next. The code around this deals with applying this specialisation to each q value and collect up the results.

Iterating Over Possible $r = \mathrm{ch}_0(u)$ for Fixed $q = \mathrm{ch}^{\beta-}(u)$

Let $q = \frac{b_q}{n}$, for which we are now solving the more specialised problem of finding solutions u with $\mathrm{ch}_1^{\beta-}(u) = q$. Corollary 4.3 gives a lower bound for $r \coloneqq \mathrm{ch}_0(u)$, and $a_v r \equiv b_q \pmod{n}$. Furthermore, we can use Theorem 4.8 to have an upper bound on r If $a_v \neq 0$, we can used an integer representative for its inverse modulo n (as they are coprime by definition), and use it to generate the values of r within the bounds satisfying the modular arithmetic condition. If $a_v = 0$, then we necessarily have $n = 1$ in which case the modular arithmetic condition on n is vacuous, so we can just consider all r within the bounds. The code related to this process is in the `possible_r` method to the `fixed_q_beta::ProblemData` structure which can be found in Appendix A.3.1.

Fixing r and q also determines $c \coloneqq \mathrm{ch}_1(u)$, and so we can generate the corresponding values of c, as we generate the r values. It now remains to solve the problem for each of the combinations of fixed values for q and r (and consequently c) considered. This is shown in Listing 5.3.

Iterating Over Possible $d = \mathrm{ch}_2(u)/\ell^2$ for Fixed $r = \mathrm{ch}_0(u)$ and $q = \mathrm{ch}^{\beta-}(u)$

At this point we are considering a specialisation of the problem where we are searching for solutions u for fixed given values of $q = \mathrm{ch}^{\beta_-(v)}(u)$ and $r \coloneqq \mathrm{ch}_0(u)$ (and hence $c = \mathrm{ch}_1(u)$). have fixed $\mathrm{ch}_0(u) = r$ and $\mathrm{ch}_1(u) = c = q + r\beta_-$. Corollary 4.3 gives us that these solutions

Listing 5.3: `tilt_stability::left_pseudo_semistabilizers ::fixed_q_beta::ProblemData::find_all`

```
128     pub fn find_all(&self) -> Result<Vec<ChernChar<M>>, &'static str> {
129         Ok(self.possible_r()?
130             .map(|(r,c)|
131                 // fix ch₀=r and ch₁=cℓ of pseudo-semistabilizer
132                 // ChernChars of pseudo-semistabilizers with fixed b, r, c
133                 fixed_r::ProblemData{r, c, fixed_q_beta_context: self}
134                 .find_all())
135             .try_collect::<Vec<_>>()?
136             .concat())
137     }
```

Listing 5.4: `tilt_stability::left_pseudo_semistabilizers ::fixed_q_beta::fixed_r::ProblemData::find_all`

```
37      pub fn find_all(&self)
38      -> Result<Vec<ChernChar<M>>, &'static str> {
39          Ok(self.possible_chern2()?
40              .map(move |d| ChernChar::<M>::new( self.r, self.c, d ))
41              .collect())
42      }
```

are precisely

$$u = (r, c\ell, d\ell^2) \qquad d \in \frac{1}{\mathrm{lcm}(m, 2n^2)}\mathbb{Z},$$

such that three numerical conditions are met ($\mathrm{ch}^{\beta-}(u) > 0$, $\Delta(u) \geq 0$, $\Delta(v-u) \geq 0$). Section 4.2.1 from the previous chapter showed that these conditions are equivalent to bounds on d given by the equations in Section 4.2.1 It therefore remains to just pick the values for $d \in \frac{1}{\mathrm{lcm}(m,2n^2)}\mathbb{Z}$ within the bounds. Listing 5.4 is the code for solving this specialisation of the problem, where the possible d values are computed in Listing 5.5. The explicit code for the bounds can be found in Appendix A.3.1.

5.2.2 Benchmarking Different Bounds

The bounds of the ranks of solutions to Problem 2 given by Theorems 4.1, 4.6, 4.8, have been shown in passing to be tighter than the previous one. However, in principle, it could be possible that this does not translate to a decrease in computational time to find the solutions to the problem. This could be due to a range of potential reasons:

- Unexpected optimisations from the compiler for a certain form of the program.
- Increased complexity to computing the formulae for the tighter bounds.
- Modern CPU architecture such as branch predictors [24] may offset the overhead of considering ranks that turn out to be too large to have any solutions.

For relatively small Chern characters (as those appearing in examples so far), the difference in performance between the program [Nay23b] when patched with the results of the different theorems above, do not show any significant difference in performance, with the earlier, weaker theorems occasionally producing the results marginally faster.

Listing 5.5: `tilt_stability::left_pseudo_semistabilizers::fixed_q_beta::fixed_r::ProblemData::possible_chern2`

```rust
    /// Returns possible ch₂ of semistabilizers u of v,
    /// where u satisfies the assumptions of ContextData
    pub fn possible_chern2(&self)
    -> Result<impl Iterator<Item = Chern2<M>>, &'static str> {
        let lowerbound = bound_on_d::lower::radius_condition(self);

        let upperbound = [
            bound_on_d::upper::bgmlv1(self),
            bound_on_d::upper::bgmlv2(self),
          ]
          .iter()
          .filter_map(|x| *x) // Remove None's
          .min()
          .ok_or("logical error, all upper bounds are None")?
          // This error should never happen^
        ;

        // Return Chern2 values d\ell^2 such that d is between the bounds
        Ok(
            inclusive_range(
                least_greater_chern2(lowerbound),
                greatest_lesser_or_eq_chern2(upperbound)
            )
        )
    }
}
```

Note that this program patched with Theorem 4.1 will be using the same bound as was used in the previously existing program [Sch20b]. However the difference of performance can be of several orders of magnitude as illustrated in the table in Section 5.1.1. This will be attributed to the difference in programming language and algorithm, the latter having already been discussed in that same section.

In order to see a difference between the different patches, we use the Chern character $v = (45, 54\ell, -41\frac{\ell^2}{2})$ for a smooth projective surface X with a generator ℓ for $\mathrm{NS}(X)$ such that $\ell^2 = 1$ or 2 (such as a principally polarised surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ or $\mathbb{P}^2$). This example was chosen for the large rank $\mathrm{ch}_0(v) = 45$, but also because of the large Bogomolov discriminant $\Delta(v) = 4761\ell^2$, which are both indicators of the size of the bounds on the pseudo-semistabiliser ranks.

As shown in Figure 5.1, there can be a significant improvement by using Theorems 4.6 and 4.8 which specialise to different values of $\mathrm{ch}_1^{\beta_-(v)}(u)$ of solutions u of Problem 2. the program to eliminate.

As for the difference between Theorems 4.6 and 4.8, the biggest indicator is the 'n'-value, that is, the denominator of $\beta_-(v)$. For this example, it is 15. The bound from Theorem 4.8 is roughly $1/k_{v,q}$ times that of Theorem 4.6. Note that $k_{v,q}$ iterates through all its possible values $\{1, 2, \ldots, n\}$ cyclically. So we could expect the average tighter bound to be approximately that of the average looser times the mean average of $1/k_{v,q}$:

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{i}.$$

This certainly tends to 0 for large n. But in the current example, with $n = 15$, this gives

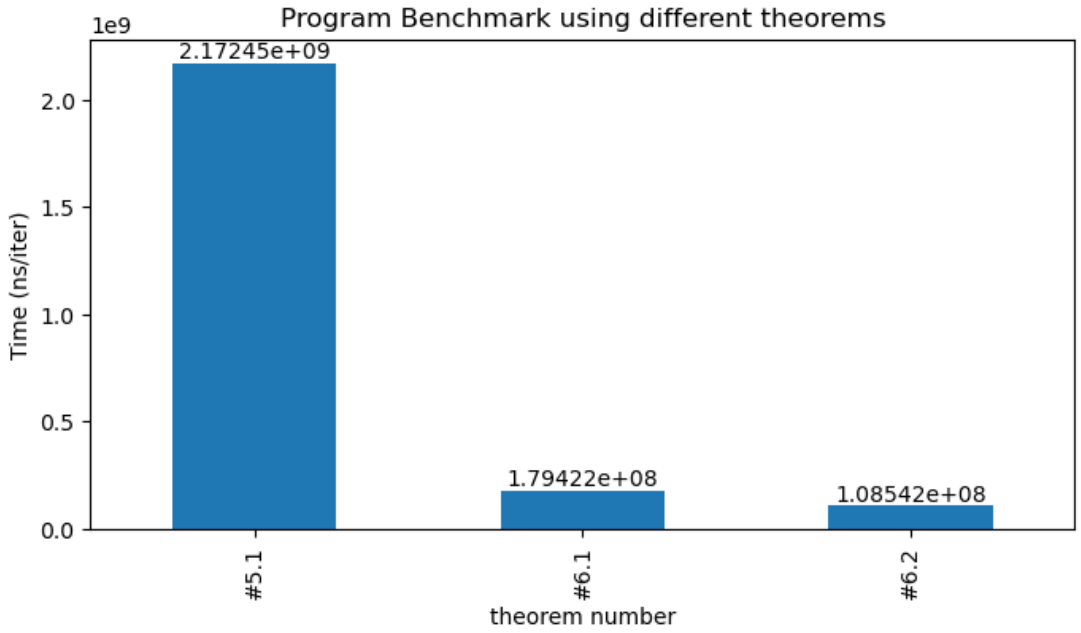
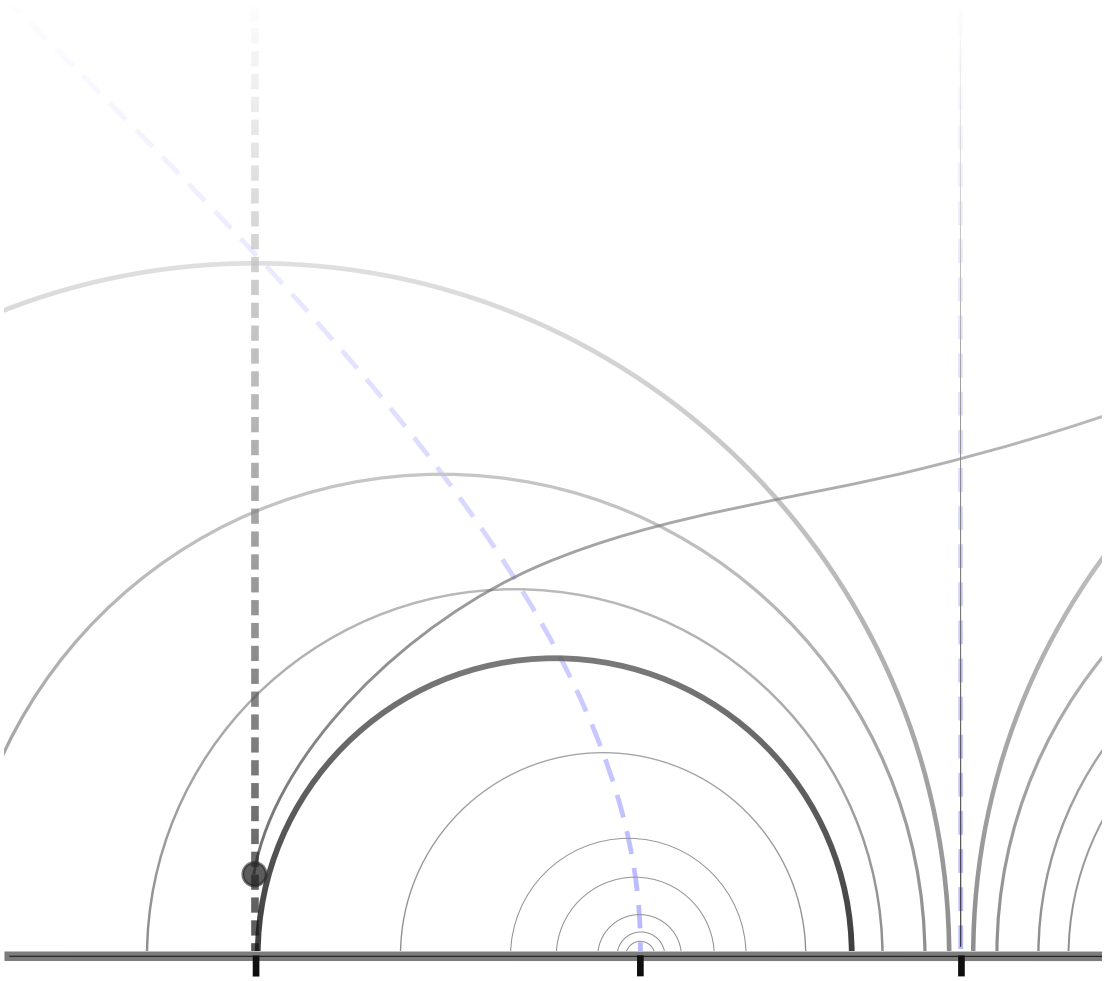


Figure 5.1: Comparing the performance of program [Nay23b] with different patches corresponding to the results of Theorems 4.1, 4.6, 4.8 when computing solutions to Problem 2 for $(45, 54\ell, -41\frac{\ell^2}{2})$

us approximately 0.2 for the ratio of the average tighter bound versus the average looser. However, the actual ratio in the benchmark shown in Figure 5.1 between the two instances of the program patched with the two corresponding bounds is around 0.6 instead. This is not as good as the improvement on the bound, however it is still not insignificant for examples with larger 'n'-value, where the execution time will potentially be in the order of minutes, or even hours.

Part III

Infinite Walls



Introduction

Part II covered the cases of Chern characters v with $\Delta(v) \geq 0$ and a square integer value for $\Delta^{\ell^2}(v)$. In that case there are finitely many walls for v, and the problem of computing them was discussed. This Part considers all other Chern characters v with $\Delta(v) \geq 0$ but specifically on a principally polarised abelian surface $(\mathbb{T}, L)$ with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ We will see that there are always infinitely many walls for such v.

Background

This phenomenon of infinite walls has been observed independently by Meachan in [Mea12b] and by Yanagida and Yoshioka in [YY14]. Yanagida and Yoshioka introduced the notion of semi-homogeneous presentations for coherent sheaves in [YY09] as a tool to transform a general stable sheaf of Chern character v, satisfying a certain "numerical condition" into an ideal sheaf of points I_S fitting into an exact sequence

$$0 \to I_S \to \mathcal{O}_X \to \mathcal{O}_S \to 0.$$

In $\mathcal{B}^0$, this becomes a destabilizing sequence for $I_S[-1]$:

$$0 \to \mathcal{O}_s \to I_S[-1] \to \mathcal{O}_X[-1] \to 0$$

giving a codimension 0 wall along the vertical line V_{I_S}. Then iterative use of a Fourier-Mukai transform generates infinitely many other codimension 0 walls which can also be transferred back to the original v. But this phenomena is restricted to positive Chern characters v which are in the same orbit as one of the form $(1, 0, -\Delta)$ for some $\Delta \geq 0$, under the action of cohomological Fourier-Mukai transforms. This is related to the above mentioned "numerical condition". However this does not cover all possible Chern characters v for which $\Delta(v) \geq 0$ and $\beta_{\pm}(v) \notin \mathbb{Q}$ (see 'Part Summary').

Meachan's approach also used Fourier-Mukai transforms iteratively to generate walls, this time explicitly on examples v with rank 1. These Fourier-Mukai transforms were also used to describe the moduli spaces of Bridgeland stable objects. But in fact, these cases considered also fall within these orbits of the Chern characters of form $(1, 0, -\Delta)$:

$$\begin{aligned}(1, c\ell, \chi) \otimes L^{\otimes -c} &= (1, c\ell, \chi) \otimes (1, \ell, 1)^{\otimes -c} \\ &= (1, c\ell, \chi) \otimes (1, -c\ell, c^2\ell^2/2) \\ &= (1, 0, -(c^2\ell^2 - 2\chi)/2) \qquad \text{Note } 0 \leq \Delta(v) = c^2\ell^2 - 2\chi\end{aligned}$$

Part Summary

In Chapter 6, some ideas behind the work by Yanagida and Yoshioka are used to generate an infinite sequence of walls for a larger class of Chern characters. This is done at the cost of constructing objects with a weaker version of the type of presentation which they considered (so called "semi-homogeneous" presentations), whereas Yanagida and Yoshioka constructed a stronger version of this type of the presentation for a general element (of the moduli space of stable sheaves). In particular, the walls generated for this larger class of Chern characters are no longer codimension 0 walls. However, as discussed in the introduction to Chapter 6, this class of Chern characters does not include all v with $\Delta(v) \geq 0$ and $\beta_{\pm}(v) \notin \mathbb{Q}$, even up to the action of cohomological Fourier-Mukai transforms. This chapter is not referenced in the rest of Part III and is mainly for illustrative purposes and to draw connection to previous work.

In the remaining chapters, the full class of possible Chern characters of sheaves on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ will be considered, relying less on ideas around semi-homogeneous presentations, but instead directly finding Fourier-Mukai transforms which fix the Chern character numerically (discussed in the introduction for Chapter 7).

The cohomological Fourier-Mukai transforms fixing a given Chern character will be explored in Chapter 7. This will then be used to consider the phenomenon of infinite walls for Bogomolov non-critical Chern characters in Chapter 8. This class of Chern characters, unlike the ones considered in Chapter 6, does include all possible Chern characters v with $\Delta(v) \geq 0$ and $\beta_{\pm}(v) \notin \mathbb{Q}$ (up to the action of CFMT). This is shown at the beginning of Chapter 9, where Fourier-Mukai transforms are then used to transfer the walls generated for Bogomolov non-critival Chern characters to others in the same orbit. This concludes with Theorem 9.6 which equates the existence of infinitely many walls for a Chern character v with the condition $\beta_{\pm}(v) \notin \mathbb{Q}$, whenever $\Delta(v) \geq 0$.

Chapter 6

Weaker Semi-Homogeneous Presentations

The later chapters in this Part III will all contribute towards the main conclusion, in Theorem 9.6, about the existence of infinitely many walls for any Chern character v on a principally polarised abelian surface whenever $\beta_{\pm}(v)$ is irrational (or equivalently $\Delta^{\ell^2}(v)$ is not a square integer). This chapter considers this but for a specific class of examples, inspired by the class considered in [YY14] of Chern characters satisfying a certain 'numerical condition'. The main purpose of this chapter is exploring an illustrative example of a natural follow up from the work by Yanagida and Yoshioka to preview a simpler version of the main argument before considering all Chern characters more systematically.

The class of Chern characters to be considered ($(R, 0\ell, -D)$ for R, $D \in \mathbb{Z}_{>0}$), does not generate all valid Chern characters with $\Delta(v) > 0$ up to action by cohomological Fourier-Mukai transforms (CFMT). This can be seen because the action of any CFMT on a Chern character $(r, c\ell, d)$ on a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, does not change the quantity $\gcd(r, d, 2)$, or the quantity $c \mod \gcd(r, d, 2c)$. In particular any Chern character in the same orbit as $(2, \ell, -2)$ must have odd c_1, not zero, hence cannot be of the form $(R, 0, -D)$. Ultimately, a different class of Chern characters shall be considered in Chapter 8. This class, of Bogomolov non-critical Chern characters, will be shown to cover all Chern characters v with $\Delta(v) \geq 0$ up to action of CFMT in Chapter 9 in order to complete the final Theorem 9.6.

6.1 Weaker 'Numerical Condition' and Semi-Homogeneous Presentations

In [YY14], a 'numerical condition' was considered on Chern characters. A Chern character v was said to satisfy the 'numerical condition' if we have:

$$v = \pm(w_1 - Dw_2) \qquad D \in \mathbb{Z}_{>0}, \tag{6.1}$$

for some positive isotropic irreducible Chern characters w_i with $\Delta(w_1, w_2) = -1$.

Remark. The minus sign in Equation (6.1) is necessary to have $\Delta(v) \geq 0$. More generally, for any positive isotropic irreducible Chern characters w_1 and w_2, we have $\Delta(w_1, w_2) = -n^2$ for some integer n.

A Chern character that obviously satisfies the numerical condition is $v := (1, 0, -D)$, as it equals $(1, 0, 0) - D(0, 0, 1)$. Instances of Coherent sheaves K with such a Chern character

are kernels of epimorphisms

$$K \hookrightarrow \mathcal{P}_{\hat{x}} \twoheadrightarrow \bigoplus_{i=1}^{D} \mathcal{O}_{x_i}. \tag{6.2}$$

This short exact sequence is an example of a 'semi-homogeneous presentation' ([YY14, Definition 4.1]), with the terms in the sequence other than K being semi-homogeneous sheaves. In fact, such kernels account for a $2+2D$ dimensional family (2 degrees of freedom for the choice of $\hat{x}$ and 2 degrees for each of the D choices of, distinct, x_i). The moduli of stable sheaves has dimension $2+\Delta(v) = 2+2D$ and these kernels will account for the general stable sheaf of Chern character $v = (1, 0, -D)$.

With the terms in Equation (6.2) being semistable sheaves, it is a stable sequence in the unbounded chamber left of V_v (for $\beta < 0 = \mu^\ell(v) = \mu^\ell(\mathcal{P}_{\hat{x}})$). For $\beta \geq 0$, the following is a short exact sequence in $\mathcal{B}^\beta$:

$$\bigoplus_{i=1}^{D} \mathcal{O}_{x_i} \hookrightarrow K[1] \twoheadrightarrow \mathcal{P}_{\hat{x}}[1].$$

This sequence destabilises $K[1]$ for $\beta > 0$ as the skyscrapers have infinite slope, making V_v a genuine vertical wall. Since the general semi-stable object of Chern character v in the left unbounded chamber is one of these Ks, which are destabilised by the wall at V_v, and is a so called 'codimension 0 wall' [YY14, Definition 5.16]. That paper related this codimension 0 wall to an infinite sequence of other codimension 0 walls v via Fourier-Mukai transform for the case when D is a non-square integer[1] Each of these codimension 0 walls correspond to another distinct way that v satisfies the numerical condition. The largest circular wall left of V_v, for example, corresponds the numerical condition solution associated to a cokernel semi-homogeneous presentation of the general stable sheaf of Chern character v:

$$E_1 \hookrightarrow E_2 \twoheadrightarrow K \qquad \mathrm{ch}(K) = v,\ E_i \text{ semi-homogeneous sheaves.} \tag{6.3}$$

The semistability of semi-homogeneous sheaves will ensure $\mu(E_1) < \mu(E_2) < \mu(K) = 0$. So that the following is a short exact sequence in $\mathcal{B}^\beta$ for $\beta \in (\mu(E_1), \mu(E_2))$:

$$E_2 \hookrightarrow K \twoheadrightarrow E_1[1],$$

and induces the said wall in that region. Furthermore, codimension 0 walls for other Chern characters satisfying the numerical condition are also related the vertical wall for $(1, 0, -D)$.[2]

In contrast, here we will consider a weaker form of the 'numerical condition':

Definition 6.1 (Weaker numerical condition)**.** We say that $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ satisfies the **weaker numerical condition with R, $D \in \mathbb{Z}_{>0}$** if we can write:

$$v = Rw_1 - Dw_2, \tag{6.4}$$

and for some positive, isotropic, and irreducible w_i with $\Delta(w_1, w_2) = -1$.

Remark. The original 'numerical condition' did not need to specify its integer parameter D as part of the condition because it is determined by $\Delta(v)$.

[1] Here the statement is specialised to the case of the abelian surface being principally polarised such that $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$. This is a translation of the condition $\sqrt{\frac{\ell}{n}} \in \mathbb{Q}$ in the original setting.

[2] The implied method and order of proof here reflects the later original content of this chapter, not the method pursued in [YY14] which followed a different order.

The analogue of $(1, 0, -D)$ for the original numerical condition shall be $v = (R, 0, -D)$ for the weaker one, with canonical objects for these Chern characters being kernels K of the form:

$$K \hookrightarrow \bigoplus_{j=1}^{R} \mathcal{P}_{\hat{x_j}} \twoheadrightarrow \bigoplus_{i=1}^{D} \mathcal{O}_{x_i}. \tag{6.5}$$

We can consider this to be a *weak semi-homogeneous kernel presentation* for K. Such K shall, as before, be destabilised by the vertical wall at $\beta = 0$, but they form a family of lower dimension than the whole moduli space of stable sheaves in general. So the vertical wall will not have codimension 0 when $R, D > 1$, which can also be deduced from [YY14, Lemma 5.21]. However we shall still be relating other ways that v satisfies the weaker numerical condition with R, D to other walls for v.

6.2 Alternative Solutions to the Weaker Numerical Condition

Consider $v = (R, 0, -D)$ for $R, D \in \mathbb{Z}_{>0}$, we hope to find non-trivial ways that v satisfies the weaker numerical condition with R, D. This is to find analogues of the semi-homogeneous presentations discussed in the previous section to then give walls for v. In order to do this, the following Problem 3 states the goals set out for section.

> **Problem Statement 3** (Pseudowalls and Pseudosemistabilisers of interest)**.** Given a fixed, positive, degree 0, Chern character v satisfying $\Delta(v) > 0$. That is, $v = (R, 0, -D)$, for some $R, D \in \mathbb{Z}_{>0}$.
> We want to determine Chern characters w_2 such that:
>
> - w_2 is positive, irreducible, and isotropic.
> - $v - Rw_2$ is also isotropic.
> - Rw_2 is a pseudo-semistabiliser for v giving rise to a pseudowall left of V_v.
>
> Then, locate these pseudo-walls.

The following Lemma 6.1 will prove, among other things, that this problem also corresponds to finding alternative solutions to the weaker numerical condition with R, D.

Lemma 6.1 (First 2 conditions of Problem 3)**.** *Let $v = (R, 0, -D)$, $R, D \in \mathbb{Z}_{>0}$. Suppose w_2 is a irreducible, positive, isotropic Chern character such that $v - Rw_2$ is also isotropic.*

Then $v - Rw_2 = Dw_1$ for some irreducible, positive, isotropic w_1 satisfying $\Delta(w_1, w_2) = -1$ and the w_i are of the following form:

$$w_2 = (a^2, \pm ab\ell, b^2) \qquad a, b \in \mathbb{Z}_{\geq 0} \text{ satisfy } 1 = a^2 - \tfrac{Rb^2}{D} \text{ (hence coprime),}$$
$$w_1 = (c^2, \pm ca\ell, a^2) \qquad \text{and } bc = a^2 - 1 \text{ (hence } c \in \mathbb{Z} \text{ coprime to } a).$$

Furthermore, these solutions satisfy $\mu(w_1) < \mu(w_2) < \mu(v)$, when using the - instead of + sign in the formulae. Otherwise, when using +: $\mu(w_1) > \mu(w_2) > \mu(v)$.

Proof. Take a general irreducible, positive, isotropic Chern character $w_2 = (a^2, \pm ab, b^2)$

(where $a, b \in \mathbb{Z}_{\geq 0}$ are coprime), and suppose that $v - Rw_2$ is isotropic. Then we have

$$\begin{aligned} 0 &= \Delta(v - Rw_2) \\ &= \Delta(v) + \cancel{R^2\Delta(w_2)} - 2R\Delta(w_2, v) && (\Delta \text{ is a bilinear form}) \\ &= 2R\left(D - \Delta(w_2, v)\right) \\ &= 2R\left(D + Rb^2 - Da^2\right) \\ \iff 1 &= a^2 - \frac{R}{D}b^2 && (R > 0). \end{aligned} \tag{6.6}$$

Take $R' \coloneqq \frac{R}{\gcd(R,D)}$, $D' \coloneqq \frac{D}{\gcd(R,D)}$.

Since a, b are coprime from Equation (6.6), and so are D' and R', we must have D' dividing b since $a^2 - 1 = \frac{R'}{D'}b^2$ is an integer. Hence $a^2 - 1$ is divisible by b and we can define the integer

$$c \coloneqq \frac{a^2 - 1}{b} = \frac{bR}{D},$$

which we can use for substitutions in the following:

$$\begin{aligned} v - Rw_2 &= \left((1 - a^2)R, \mp abR\ell, -(D + b^2R)\right) \\ &= \left(-\frac{b^2R}{D}R, \pm(-D)a\left(\frac{bR}{D}\right)\ell, -(D + (a^2 - 1)D)\right) \\ &= -D\left(c^2, \pm ac\ell, a^2\right)) \\ &= -Dw_1 \qquad \text{where } w_1 \coloneqq \left(c^2, \pm ac\ell, a^2\right)). \end{aligned}$$

In fact, the value of $\Delta(w_1, w_2)$ is fixed by these conditions:

$$\begin{aligned} \Delta(w_1, w_2) &= 2(\pm ab)(\pm ca) - a^2a^2 - b^2c^2 \\ &= -(a^2 - bc)^2 \\ &= -1 && (\text{by definition of } c). \end{aligned}$$

Now notice that Equation (6.6) implies that $a > 1$ since $R, D > 0$. Consequently, c is also positive. So then $\frac{b}{a}$ and $\frac{a}{c}$ are both positive. Furthermore, from the definition of c, we have:

$$\frac{c}{a} = \frac{a^2 - 1}{ab} < \frac{a^2}{ab} = \frac{a}{b},$$

from which we can conclude that $\mu(w_1) < \mu(w_2) < \mu(v)$, when using the - instead of + sign in the formula for w_2. Otherwise, when using +: $\mu(w_1) > \mu(w_2) > \mu(v)$.

□

Now to relate this lemma to the problem at hand, we have the following corollary:

Corollary 6.2 (Existence of the pseudowalls of interest). *The solutions w_2 to Problem 3 are given by $w_2 = (a^2, -ab, b^2)$ for $a, b \in \mathbb{Z}_{\geq 0}$ satisfying $1 = a^2 - \frac{Rb^2}{D}$. $v - Rw_2 = Dw_1$ where $w_1 = (c^2, -ca, a^2)$ with $c = \frac{(a+1)(a-1)}{b}$.*

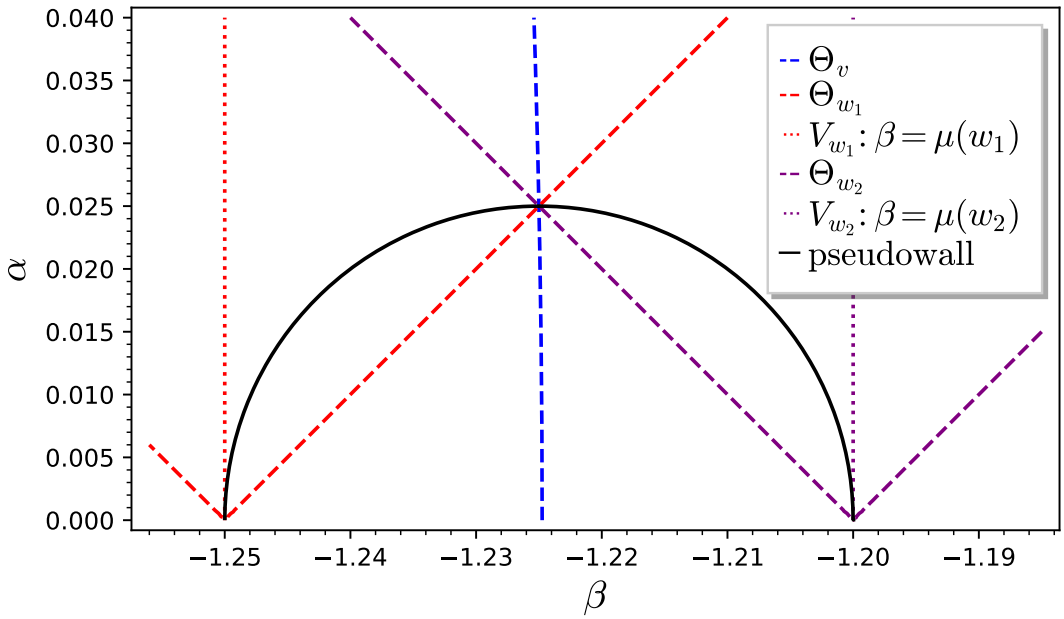


Figure 6.1: Characteristic curves and pseudowall when both destabiliser (moving inwards) and quotient are isotropic

Finally, the corresponding pseudowall induced by Rw_2 for v is positioned and sized as follows:

$$\begin{aligned} \textit{left side: } \beta &= -\frac{ab}{(a+1)(a-1)}, \\ \textit{right side: } \beta &= -\frac{b}{a}, \\ \textit{radius} &= \frac{b}{2\,(a+1)(a-1)a}, \\ \textit{centre: } \beta &= -\frac{(2\,a^2-1)b}{2\,(a+1)(a-1)a}. \end{aligned}$$

Proof. By Lemma 6.1, solutions to Problem 3, from the first two conditions, are of the form $w_1 = (a^2, \pm ab\ell, b^2)$ for some $a, b \in \mathbb{Z}_{\geq 0}$ satisfying $1 = a^2 - \frac{Rb^2}{D}$. And $v - Rw_2 = Dw_1$ where $w_1 = (c^2, \pm ca, a^2)$ with $c = \frac{(a+1)(a-1)}{b}$. Furthermore, when considering a solution with the minus sign instead of plus, we have $\mu(-Dw_1) < \mu(Rw_2) < \mu(v)$. Since $\Delta(w_i) = 0$, the characteristic curves Θ_{Rw_2} and Θ_{-Dw_1} degenerate to straight lines at $\pm 45°$ intersecting the β-axis at $\mu(Rw_2)$ and $\mu(-Dw_1)$ respectively. So they must intersect at at point $Q = (\alpha_Q, \beta_Q)$ where

$$\begin{aligned} \mu(-Dw_1) < \beta_Q = \frac{\mu(Rw_2) + \mu(-Dw_1)}{2} < \mu(Rw_2) < \mu(v), \\ 0 < \alpha_Q = \frac{\mu(Rw_2) - \mu(-Dw_1)}{2}. \end{aligned}$$

which implies that Rw_2 is a pseudo-semistabiliser of v at Q:

- $\nu_Q(Rw_2) = \nu_Q(Dw_1) = 0$, and consequently, $\nu_Q(v) = \nu_Q(Rw_2 - Dw_1) = 0$ too.

- $\mu(-Dw_1) < \beta_Q < \mu(Rw_2)$ ensures $\mathrm{ch}_1^{\beta_Q}(-Dw_1) > 0$ and $\mathrm{ch}_1^{\beta_Q}(Rw_2) > 0$.

Bertram's nested wall Theorem 1.14 gives us that the pseudo-wall associated to Rw_2 is centred on the β-axis at $\beta = \beta_Q$ and the left and right sides reach up to $\mu(-Dw_1)$ and $\mu(Rw_2)$ respectively. This is illustrated in Figure 6.1. The position of the pseudo-wall can then be determined:

$$\begin{aligned}
\text{left side: } \beta = \mu(-Dw_1) &= -\frac{ab}{(a+1)(a-1)},\\
\text{right side: } \beta = \mu(Rw_2) &= -\frac{b}{a},\\
\text{radius} &= \frac{\mu(Rw_2) - \mu(-Dw_1)}{2} = \frac{b}{2\,(a+1)(a-1)a},\\
\text{centre: } \beta &= \frac{\mu(Rw_2) + \mu(-Dw_1)}{2} = -\frac{(2\,a^2-1)b}{2\,(a+1)(a-1)a}.
\end{aligned}$$

Computer algebra for these calculations can also be found at Appendix C.6. In particular, this pseudowall are on the left of $V_v = \{\beta = 0\}$. By symmetry, the possibilities for w_2 with the + sign instead of - give pseudowalls on the right hand side. Therefore the solutions to Problem 3 are precisely the ones specified in Lemma 6.1 with the - sign as required with the specified position.

□

The equation on a, b in the above lemma and corollary can be related to a Pell's equation, whose solutions are well known and in particular can be generated in certain cases as shall be done next.

Lemma/Definition 6.1.1 (Sequence of the pseudowalls of interest)**.** Let $v = (R, 0, -D)$, with $R, D \in \mathbb{Z}_{>0}$, and such that $\Delta(v) = RD$ is non-square. Then there are *sequences of positive integers* $(a_n)_{n\geq 1}$ *and* $(b_n')_{n\geq 1}$ such that the solutions to Problem 3 are:

$$w_2^n = \left(a_n^2, -D' a_n b_n', D'^2 b_n'^2\right) \qquad D' := \frac{D}{\gcd(R, D)},$$

and then $v - Rw_2^n = Dw_1^n$ where $w_1^n = (c_n^2, -c_n a_n, a_n^2)$, with $c_n := \frac{(a_n+1)(a_n-1)}{D' b_n'}$.

The pseudowall C_n for v is the one induced by Rw_2^n. These pseudowalls decrease in size monotonically, and accumulate to the point $(\beta_-(v), 0)$. These sequences are expressed in terms of *a minimal solution* (a_1, b_1') to the equation $1 = a^2 - (RD')b'^2$ via:

$$\begin{pmatrix} a_n & D'R'b_n' \\ b_n' & a_n \end{pmatrix} = \begin{pmatrix} a_1 & D'R'b_1' \\ b_1' & a_1 \end{pmatrix}^n.$$

Proof. Take $R' := \frac{R}{\gcd(R,D)}$, $D' := \frac{D}{\gcd(R,D)}$.
Corollary 6.2, states that solutions to Problem 3 are $w_2 = (a^2, -ab\ell, b^2)$ where $a, b \in \mathbb{Z}_{\geq 0}$ such that $1 = a^2 - \frac{Rb^2}{D}$, and $v - Rw_2 = Dw_1$ where $w_1 = (c^2, -ca\ell, a^2)$ and $c = \frac{a^2-1}{b}$. As seen in the proof of that corollary, we also have that D' divides b for any such solution. Substituting $b = D'b'$, we can write these solutions as $w_2 = (a^2, -D'ab'\ell, D'^2 b^2)$ (and correspondingly, $w_1 = (c^2, ac\ell, a^2)$ with $c = \frac{a^2-1}{D'b'}$) where $1 = -D'R'b'^2 + a^2$. Since $RD = R'D'\gcd(R, D)^2$ is non-square, the same is true for $R'D'$. By [AA15, Theorem 3.2.1], the values of a and b' giving the solutions to Problem 3 can be generated from a minimal

solution $a_1, b_1^{'}$ by the formula in the statement of Lemma/Definition giving sequences $a_n, b_n^{'}$ determining the solutions w_2^n to the problem as stated.

□

Lemma 6.3. *The top of C_n is vertically aligned with the left side of C_{2n}. That is, the β-coordinate of the point on C_n with maximum α-coordinate, is equal to the infimum of the β-coordinates of the points on C_{2n}.*

Proof. The computer algebra for the calculations can be found at Appendix C.6. Note that $a_n, b_n^{'}$ are related to $a_{2n}, b_{2n}^{'}$ via:

$$\begin{pmatrix} a_{2n} & D^{'}R^{'}b_{2n}^{'} \\ b_{2n}^{'} & a_{2n} \end{pmatrix} = \begin{pmatrix} a_n & D^{'}R^{'}b_n^{'} \\ b_n^{'} & a_n \end{pmatrix}^2 \qquad \text{giving} \qquad \begin{cases} a_{2n} &= 2\,a_n^2 - 1 \\ b_{2n}^{'} &= 2\,a_n b_n^{'}. \end{cases}$$

Using the formulae from Corollary 6.2 to find the left side of C_{2n}, and the centre of C_n shows that they are equal to $-\frac{(2\,a_n^2-1)D^{'}b_n^{'}}{2\,(a_n+1)(a_n-1)a_n}$.

□

6.3 Infinite Wall Phenomena for $(R, 0, -D)$

Lemma/Definition 6.1.2 (FMTs of interest - Yanagida)**.** For each $n \in \mathbb{Z}_{\geq 1}$, there is a $K_n \in \mathrm{Coh}(\mathbb{T} \times \mathbb{T})$ such that:

$$(\Phi_{K_n})^H = \left(\left[\begin{pmatrix} a_n & c_n \\ b_n & a_n \end{pmatrix} \right], +1 \right),$$

where $b_n = D^{'}b_n^{'}$, and $c_n = \frac{{a_n}^2-1}{b_n}$. Furthermore the action on stability conditions satisfies the identity.

$$\Phi_{K_n[1]}\left(\sigma_{\alpha,\,\mu(w_1^n)}\right) = \sigma_{\frac{1}{\alpha {c_n}^2},\,-\mu(w_1^n)}.$$

Proof. Since ${a_n}^2 - b_n c_n = 1$, this is an application of Theorem 2.5, with the second statement given by Theorem 2.7.

□

Proposition 6.4 (Semistable sequence in unbounded chamber on the right of V_v)**.** *Given $R, D \in \mathbb{Z}_{>0}$ and $v := (R, 0, -D)$. For $\beta > 0$ and α sufficiently large, then the following is an exact sequence in $\mathcal{B}^\beta$, where all three terms are $\sigma_{\alpha,\,\beta}$-stable:*

$$0 \to \bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[1] \hookrightarrow E^\vee[1] \twoheadrightarrow \bigoplus_{j=1}^{D} \mathcal{O}_{x_j} \to 0,$$

with $\mathrm{ch}\left(E^\vee[1]\right) = -v$, for any semistable E coherent sheaf which is the kernel of an epimorphism $\bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i} \twoheadrightarrow \bigoplus_{j=1}^{D} \mathcal{O}_{x_j}$.

Remark. The sequence destabilises $E^\vee[1]$ when crossing V_v left-wards. In essence, the coming Theorem 6.5 will relate V_v as a wall, to an infinite sequence of other walls for v when RD is non-square.

Proof. Suppose E is a Gieseker semistable sheaf that fits into the exact sequence

$$0 \to E \hookrightarrow \bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i} \twoheadrightarrow \bigoplus_{j=1}^{D} \mathcal{O}_{x_j} \to 0,$$

so that $\mathrm{ch}(E) = (R, 0, -D)$. Applying the derived dual functor to the corresponding exact triangle and shifting the triangle twice gives:

$$\bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[1] \to E^\vee[1] \to \bigoplus_{j=1}^{D} \mathcal{O}_{x_j} \rightsquigarrow \bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[2].$$

By [JM19, Lemma 5.17], these are semistable objects in the unbounded chamber to the right of V_v, and so gives the required short exact sequence when taking cohomology with respect to $\mathcal{B}^\beta$ for $\beta \geq 0$.

□

Theorem 6.5 (Infinite walls examples with $(R, 0, -D)$). *Let $\mathbb{T}$ be a principally polarised surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$. Let $v = (R, 0, -D)$ for some $R, D \in \mathbb{Z}_{>0}$, and suppose that $\Delta^{\ell^2}(v) = RD$ is non-square. For any semistable kernel (in $\mathrm{Coh}(\mathbb{T})$) of an epimorphism of the form $\bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i} \twoheadrightarrow \bigoplus_{j=1}^{D} \mathcal{O}_{x_j}$ for some $\hat{x}_i \in \hat{\mathbb{T}}$ and $x_j \in \mathbb{T}$, there is a sequence of walls $(C_n)_{n\geq 1}$ for v, left of V_v which accumulate to the point $(\beta_-(v), 0)$ induced by the sequences:*

$$0 \to \left(\Phi_{K_n[1]}\right)^{-1} \left(\bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[1] \right) \to \left(\Phi_{K_n[1]}\right)^{-1} \left(E^\vee[1]\right) \to \left(\Phi_{K_n[1]}\right)^{-1} \left(\bigoplus_{j=1}^{D} \mathcal{O}_{x_j} \right) \to 0.$$

Proof. Take E as described. By Proposition 6.4, the following sequence consists of semistable objects on the unbounded chamber on the right-hand-side of V_v:

$$0 \to \bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[1] \hookrightarrow E^\vee[1] \twoheadrightarrow \bigoplus_{j=1}^{D} \mathcal{O}_{x_j} \to 0.$$

Considering Lemma/Definition 6.1.2, $\sigma_{\frac{1}{\alpha c_n{}^2}, -\mu(w_1^n)}$ is in the unbounded chamber right of V_v, for $0 < \alpha \ll 1$, because $\mu(w_1^n) < 0$. Also, $\sigma_{\alpha, \mu(w_1^n)}$ approaches arbitrarily close, for arbitrarily small α to the pseudowall associated to Rw_2^n as a pseudo-semistabiliser of v, whose position was determined in Lemma/Definition 6.1.1. Therefore the following sequence consists of semistable objects arbitrarily close to the pseudowall.

$$0 \to \left(\Phi_{K_n[1]}\right)^{-1} \left(\bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[1] \right) \to \left(\Phi_{K_n[1]}\right)^{-1} \left(E^\vee[1]\right) \to \left(\Phi_{K_n[1]}\right)^{-1} \left(\bigoplus_{j=1}^{D} \mathcal{O}_{x_j} \right) \to 0$$

Finally, $\mathrm{ch}\left(\left(\Phi_{K_n[1]}\right)^{-1} \left(\bigoplus_{i=1}^{R} \mathcal{P}_{\hat{x}_i}[1]\right)\right) = Rw_1^n$ from the definition of K_n, so this sequence instantiates the pseudowall C_n associated to Rw_1^n as a genuine wall v. Finally, Lemma/Definition 6.1.1 gives us that these walls C_n decrease in size monotonically, and accumulate to the point $(\beta_-(v), 0)$.

□

Chapter 7

Stabiliser Subgroups of CFMT

This chapter takes a number theoretical detour to establish a necessary fact about the existence of Cohomological Fourier-Mukai transforms (CFMT) fixing a given Chern character (up to shifts). This revolves around calculations related to $\mathrm{SL}(2,\mathbb{Z})$, with the main result being Lemma 7.3. Then this is reformulated to the context of this thesis in Theorem 7.5.

In this thesis, we consider Fourier-Mukai transforms (FMT) which fix a Chern character numerically (modulo sign change), before considering the transforms themselves and the associated sheaf theory. Compare this with [YY14], which relies on the predictability of the images of semi-homogeneous sheaves under the transforms (semi-homogeneous sheaves themselves, or a shift). This then directed the construction of 'semi-homogeneous presentations' for generic sheaves of a Chern character v satisfying the 'numerical condition'. This presentation was used to define the Fourier-Mukai transforms which related the codimension 0 walls to the vertical wall along V_v.

The direction pursued in this thesis, is to first consider CFMT which act on stability conditions in a way which take points arbitrarily close to a pseudowall, to an unbounded chamber where the stable objects are heavily related to Gieseker stable sheaves. This will then equate the existence of the destabilising sequence for that pseudowall to the existence of a short exact sequence of stable sheaves of given Chern characters. In the case of [YY14], this latter question corresponded to the existence of semi-homogeneous presentations, but in this thesis, will correspond to the existence of sequences of a possibly different form (Lemma 8.1). A benefit of this approach is that everything up to the final step of constructing the sequence of sheaves, relies on the CFMT group. Then the final step relies on knowledge of stable sheaves on our variety. This will potentially allow for this strategy to be replicated on other varieties more easily, whereas [YY14] relies on semi-homogeneous sheaves heavily throughout, which is a concept very specific to abelian varieties.

7.1 Stabiliser Subgroups of $\mathrm{PSL}(2,\mathbb{Z})$

Although we are ultimately interested in $\mathrm{PSL}(2,\mathbb{Z})$, due to its relation to cohomological Fourier-Mukai transforms, working over $\mathbb{Q}$ is a necessary detour for a later proof.

Definition 7.1 (Left action of $\mathbf{PSL(2,\mathbb{Q})}$ on $\mathbf{Symm(2,\mathbb{Q})}$)**.** For $g \in \mathrm{PSL}(2,\mathbb{Q})$ and $A \in \mathrm{Symm}(2,\mathbb{Q})$, we can define

$$g \cdot A = gAg^T,$$

where g^T is the matrix transpose of g and the products on the right-hand side are matrix products.

We similarly define the corresponding action of $\mathrm{PSL}(2,\mathbb{Z})$ on $\mathrm{Symm}(2,\mathbb{Z})$.

Lemma 7.1 (Stabiliser for diagonal matrix)**.** *Consider the action of* $\mathrm{PSL}(2,\mathbb{Q})$ *on* $\mathrm{Symm}(2,\mathbb{Q})$*. Then the stabilisers of diagonal matrices with non-zero diagonal entries is given by the following:*

$$\mathfrak{st}_{\mathrm{PSL}(2,\mathbb{Q})}\left(\begin{pmatrix} q_1 & 0 \\ 0 & q_2 \end{pmatrix}\right) = \left\{ \begin{pmatrix} x & -yq_1 \\ yq_2 & x \end{pmatrix} : x, y \in \mathbb{Q},\ x^2 + (q_1q_2)y^2 = 1 \right\}.$$

Proof. Take a generic element $g = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{PSL}(2,\mathbb{Q})$ and $A = \begin{pmatrix} q_1 & 0 \\ 0 & q_2 \end{pmatrix} \in \mathrm{Symm}(2,\mathbb{Q})$, with $q_i \neq 0$. Note that the latter condition ensures that A is invertible as a matrix over $\mathbb{Q}$

$$\begin{aligned}
& A = g \cdot A \\
\iff \quad & A = gAg^T \\
\iff \quad & g^{-1} = Ag^TA^{-1} \\
\iff \quad & \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = \begin{pmatrix} q_1 & 0 \\ 0 & q_2 \end{pmatrix} \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} q_1^{-1} & 0 \\ 0 & q_2^{-1} \end{pmatrix} \\
\iff \quad & \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = \begin{pmatrix} a & \frac{q_1}{q_2}c \\ \frac{q_2}{q_1}b & d \end{pmatrix} \\
\iff \quad & \begin{cases} a = d \\ q_1c + q_2b = 0 \end{cases} \\
\iff \quad & g = \begin{pmatrix} x & -q_1y \\ q_2y & x \end{pmatrix} \qquad \text{for some } x, y \in \mathbb{Q}.
\end{aligned}$$

Therefore, $g \in \mathrm{PSL}(2,\mathbb{Q})$ being a stabiliser for x is equivalent to it being in the form in the last expression. Conversely, an element of that form is in $\mathrm{PSL}(2,\mathbb{Q})$ iff $x^2 + (q_1q_2)y^2 = 1$. We can then conclude the statement of the lemma.

□

The above alone would allow us to identify the Cohomological Fourier-Mukai Transforms which fix a Chern character of degree 0. But for other Chern characters, we would need to perform a "formal twisting", as we remark on following this Lemma.

Lemma 7.2 (Reducing problem to formal Chern characters of degree 0)**.** *Let $g \in \mathrm{PSL}(2,\mathbb{Z})$, and $t \in \mathrm{PSL}(2,\mathbb{Q})$. Then g being a stabiliser for $A \in Symm(2,\mathbb{Z})$ is equivalent to having $tgt^{-1} \in \mathrm{PSL}(2,\mathbb{Q})$ being a stabiliser for $tAt^T \in Symm(2,\mathbb{Q})$. Furthermore, t can be chosen so that tAt^T is diagonal.*

Explicitly, this choice of t is

$$t \coloneqq \begin{pmatrix} 1 & 0 \\ -\frac{C}{R} & 1 \end{pmatrix}, \qquad \textit{when} \qquad A = \begin{pmatrix} R & C \\ C & D \end{pmatrix} \in \mathrm{Symm}(2,\mathbb{Z}). \tag{7.1}$$

Proof. Let $g \in \mathrm{PSL}(2,\mathbb{Z})$, and $t \in \mathrm{PSL}(2,\mathbb{Q})$.

$$\begin{aligned} & g \cdot A = A \\ \iff \quad & gAg^T = A \\ \iff \quad & gt^{-1}tAt^T(t^T)^{-1}g^T = t^{-1}tAt^T(t^T)^{-1} \\ \iff \quad & tgt^{-1}tAt^T(t^T)^{-1}g^Tt^T = tAt^T \\ \iff \quad & (tgt^{-1}) \cdot (tAt^T) = (tAt^T) \end{aligned}$$

Finally, with the choice of t given in Equation (7.1) we have

$$\begin{pmatrix} 1 & 0 \\ -\frac{C}{R} & 1 \end{pmatrix} \begin{pmatrix} R & C \\ C & D \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\frac{C}{R} & 1 \end{pmatrix}^T = \begin{pmatrix} R & 0 \\ 0 & D - \frac{C^2}{R} \end{pmatrix},$$

which is diagonal as required.

□

Remark (Relation to twisting). Note that in the above Lemma 7.2, when R divides C, and A corresponds to a Chern character of an object E, then tAt^T corresponds to the Chern character of the twisting of $E \otimes L^{\otimes -C/R}$. However, in other cases we will need to consider these 'formal' Chern characters and twistings. 'Formal' here means involving $\mathbb{Q}$ and not necessarily representing real geometric objects.

Lemma 7.3 (Stabiliser subgroups of $\mathbf{PSL(2,\mathbb{Z})}$). *Let* $\begin{pmatrix} R & C \\ C & D \end{pmatrix} \in \mathrm{Symm}(2,\mathbb{Z})$, *then its stabiliser subgroup of* $\mathrm{PSL}(2,\mathbb{Z})$ *is*

$$\left\{ \begin{pmatrix} x - C'y & -R'y \\ D'y & x + C'y \end{pmatrix} : x, y \in \mathbb{Z},\ x^2 - (C'^2 - R'D')y^2 = 1 \right\},$$

unless both R' and D' are even, in which case we get a slightly large group:

$$\left\{ \frac{1}{2} \begin{pmatrix} x - C'y & -R'y \\ D'y & x + C'y \end{pmatrix} : x, y \in \mathbb{Z},\ x^2 - (C'^2 - R'D')y^2 = 4 \right\},$$

where

$$(R', C', D') \coloneqq \frac{1}{\gcd(R,C,D)} \cdot (R,C,D).$$

Remark. As elements of the Modular group, these are examples of "hyperbolic" Möbius transformations, since the traces of the matrices are greater than 2 [Nee23, Section 3.7.5]. Visual representations of their actions can be found at [Nee23, Section 3.8.3], showing 'attractive' and 'repulsive' fixed points.

Proof. Due to the $\mathbb{Z}$-linearity of the $\mathrm{PSL}(2,\mathbb{Z})$ action, the stabiliser subgroup of any element $A \in \mathrm{Symm}(2,\mathbb{Z})$ is equal to that of any of its $\mathbb{Z}_{\neq 0}$-multiples. We can therefore assume without loss of generality that $\gcd(R,C,D) = 1$ (and hence $(R,C,D) = (R',C',D')$).

Take $t \coloneqq \begin{pmatrix} 1 & 0 \\ -\frac{C}{R} & 1 \end{pmatrix}$, so

$$\begin{aligned}
\mathfrak{st}_{\mathrm{PSL}(2,\mathbb{Z})}(A) &= t\left[t^{-1}\mathrm{PSL}(2,\mathbb{Z})t \cap \mathfrak{st}_{\mathrm{PSL}(2,\mathbb{Q})}(t^T A t)\right] t^{-1} && \text{by Lemma 7.2}\\
&= \mathrm{PSL}(2,\mathbb{Z}) \cap t\left(\mathfrak{st}_{\mathrm{PSL}(2,\mathbb{Q})}(t^T A t)\right) t^{-1}\\
&= \mathrm{PSL}(2,\mathbb{Z}) \cap && \text{by Lemma 7.1}\\
&\quad t\left\{\begin{pmatrix} x & -Ry \\ \left(D-\frac{C^2}{R}\right)y & x \end{pmatrix} : x,y\in\mathbb{Q},\ x^2-(C^2-RD)y^2=1\right\} t^{-1}\\
&= \mathrm{PSL}(2,\mathbb{Z}) \cap && (7.2)\\
&\quad \left\{\begin{pmatrix} -Cy+x & -Ry \\ Dy & Cy+x \end{pmatrix} : x,y\in\mathbb{Q},\ x^2-(C^2-RD)y^2=1\right\}.
\end{aligned}$$

The computer algebra performing the above calculations can be found at Appendix C.9.4. Now to prove equality between the final line and the statement of the Lemma. Consider matrices of the form

$$\begin{pmatrix} -Cy+x & -Ry \\ Dy & Cy+x \end{pmatrix} \qquad x,y\in\mathbb{Q} \quad x^2-(C^2-RD)y^2=1. \tag{7.3}$$

They have determinant 1, so they are contained in $\mathrm{PSL}(2,\mathbb{Z})$ iff their entries are all integral. Given that $\gcd(R,C,D)=1$ by assumption, we have:

$$\gcd(R,2C,D) = \begin{cases} 2 & \text{if } R \text{ and } D \text{ both even} \\ 1 & \text{otherwise.} \end{cases}$$

These two cases corresponding to the cases in the statement of the Lemma.

$$\begin{aligned}
&\text{matrix in Equation (7.3) is in } \mathrm{PSL}(2,\mathbb{Z})\\
&\iff x \pm Cy,\ Dy,\ Ry \in \mathbb{Z}\\
&\iff x - Cy,\ 2Cy,\ Dy,\ Ry \in \mathbb{Z}\\
&\iff x - Cy,\ \gcd(R,2C,D)y \in \mathbb{Z}
\end{aligned}$$

Assuming one of R or D odd:

$$\begin{aligned}
&\iff x - Cy,\ y \in \mathbb{Z}\\
&\iff x,\ y \in \mathbb{Z}
\end{aligned}$$

Assuming R, D both even:

$$\begin{aligned}
&\iff x - Cy,\ 2y \in \mathbb{Z}\\
&\iff y \in \frac{1}{2}\mathbb{Z},\ x - Cy \in \mathbb{Z}\\
&\iff x, y \in \frac{1}{2}\mathbb{Z} \quad \text{(proof deferred)}
\end{aligned}$$

One of R or D odd: the logical implications above tell us that elements of the form in Equation (7.3) are in $\mathrm{PSL}(2,\mathbb{Z})$ iff $x^2-(C^2-RD)y^2=1$ and we have $x,y\in\mathbb{Z}$. Thus giving equality between $\mathfrak{st}_{\mathrm{PSL}(2,\mathbb{Z})}(A)$ and the first subgroup of $\mathrm{PSL}(2,\mathbb{Z})$ in the statement of the lemma.

***R* and *D* both even:** first, to cover the deferred proof from above for

$$y \in \frac{1}{2}\mathbb{Z},\ x - Cy \in \mathbb{Z} \iff x, y \in \frac{1}{2}\mathbb{Z}$$

under the current assumptions. The implication $\Rightarrow$ follows from C being and integer. The converse will follow from $x^2-(C^2-RD)y^2=1$. Suppose $x,y\in\frac{1}{2}\mathbb{Z}$, then we have the

equality $(2x)^2 - (C^2 - RD)(2y)^2 = 4$.

$$\begin{aligned}(2x)^2 - (C^2 - RD)(2y)^2 &\equiv 0 \pmod 4 \\ (2x)^2 - (2y)^2 &\equiv 0 \pmod 4 \text{ since } C \text{ is odd and } R, D \text{ even} \\ (2x) \pm (2y) &\equiv 0 \pmod 2 \\ (2x) \pm (2Cy) &\equiv 0 \pmod 2 \\ x \pm Cy &\in \mathbb{Z}\end{aligned}$$

Coming back to computing $\mathfrak{st}_{\mathrm{PSL}(2,\mathbb{Z})}(A)$, matrices of the form in Equation (7.3) are therefore in $\mathrm{PSL}(2,\mathbb{Z})$ iff $x, y \in \frac{1}{2}\mathbb{Z}$. Reparametrising new x, y as the old $2x, 2y \in \mathbb{Z}$, we can equate Equation (7.2) to the second subgroup in the Lemma statement as required.

□

7.2 Relation to Cohomological Fourier-Mukai Transforms

Let $\mathbb{T}$ be a principally polarised complex surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$. Recall from Theorem 2.5, there is a surjective group homomorphism from the automorphisms of $\mathcal{D}^b(\mathbb{T})$ given by Fourier-Mukai transforms, to the group of cohomological Fourier-Mukai transforms, given by $\mathrm{PSL}(2,\mathbb{Z}) \times \{\pm 1\}$, through which the action on Chern characters factors through. We also know from Theorem 2.7, that Fourier-Mukai transforms Φ such that $\Phi^H = ([g], -1)$, map stability conditions in a certain way. We now consider, for a given Chern character $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$, the cohomological Fourier-Mukai transforms Φ^H such that $\Phi^H = ([g], -1)$, and $\Phi^H(v) = -v$. This will follow directly from results in the previous section, and more general results about any cohomological Fourier-Mukai transform fixing v up to sign change could also be derived. However the statements would involve negative Pell's equations which are not known to have constructive solutions. The conclusion about the finiteness of the CFMTs of interest in Theorem 7.5 would still hold for this larger solution set.

Proposition 7.4. *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ for a principally polarised abelian surface $\mathbb{T}$ such that $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, with positive rank $\mathrm{ch}_0(v) > 0$ and $\Delta(v) > 0$. Given any Fourier-Mukai transform of the form $\Phi_{K[1]}$ with $K \in \mathcal{D}^b(\mathbb{T} \times \mathbb{T})$, such that $\Phi^H_{K[1]}(v) = -v$ (or equivalently, that $\Phi^H_K(v) = -v$), then the following is satisfied:*

$$\beta_{\mathrm{dom}}(\Phi_{K[1]}) < \beta_-(v) \quad \textit{or} \quad \beta_{\mathrm{dom}}(\Phi_{K[1]}) > \beta_+(v),$$

$$\beta_{\mathrm{codom}}(\Phi_{K[1]}) < \beta_-(v) \quad \textit{or} \quad \beta_{\mathrm{codom}}(\Phi_{K[1]}) > \beta_+(v).$$

Proof. Due to the $\mathbb{Z}$-linearity of the action of CFMT, we can assume without loss of generality that v is irreducible, so $v = (R, C, D)$ for some coprime integers R, C, D. By Lemma 2.6, $\Phi_{K[1]} = ([g], -1)$ for some $g \in \mathrm{PSL}(2,\mathbb{Z})$. So the condition $\Phi^H_{K[1]}(v) = -v$ is equivalent to the condition $g \in \mathfrak{st}(\gamma(v))$, which is classified by Lemma 7.3. In particular, in either case of the Lemma, g must be of the form

$$\begin{pmatrix} x - Cy & -Ry \\ Dy & x + Cy \end{pmatrix} : \quad x, y \in \mathbb{Q},\ x^2 - \Delta^{\ell^2}(v)y^2 = 1.$$

The condition on x and y ensures that

$$\left(\frac{x}{y}\right)^2 = \Delta^{\ell^2}(v) + \frac{1}{y^2} > \Delta^{\ell^2}(v),$$

so considering $\beta_{\mathrm{dom}}(\Phi_{K[1]})$:

$$\begin{aligned}\beta_{\mathrm{dom}}(\Phi_{K[1]}) &= \frac{-(x+Cy)}{-Ry} \\ &= \frac{x}{Ry} + \mu^{\ell}(v), \qquad (7.4)\\ \text{so} \quad \frac{x}{y} &= R\beta_{\mathrm{dom}}(\Phi_{K[1]}) - C \\ &= -\operatorname{ch}_1^{\beta_{\mathrm{dom}}(\Phi_{K[1]})}(v).\end{aligned}$$

$$\text{Hence,} \quad \Delta^{\ell^2}(v) < \left(\frac{x}{y}\right)^2 = \left(\operatorname{ch}_1^{\beta_{\mathrm{dom}}(\Phi_{K[1]})}(v)\right)^2. \qquad (7.5)$$

Recall that equality in $\left(\operatorname{ch}_1^{\beta}(v)\right)^2 = \Delta^{\ell^2}(v)$ is only achieved by $\beta = \beta_{\pm}(v)$. In fact by Lemma 1.12, the last Equation (7.5) implies that we have one of $\beta_{\mathrm{dom}}(\Phi_{K[1]}) < \beta_-(v)$ or $\beta_{\mathrm{dom}}(\Phi_{K[1]}) > \beta_+(v)$. Revisiting Equation (7.4), we can deduce the conditions for either case (since $\beta_-(v) < \mu^{\ell}(v) < \beta_+(v)$):

$$\begin{cases}\beta_{\mathrm{dom}}(\Phi_{K[1]}) < \beta_-(v), & \text{if } x, y \text{ have opposite signs,}\\ \beta_{\mathrm{dom}}(\Phi_{K[1]}) > \beta_+(v), & \text{if } x, y \text{ have same signs.}\end{cases}$$

The corresponding statement on $\beta_{\mathrm{codom}}(\Phi_{K[1]})$ follows from exactly the same calculations, but with a single sign change in front of Cy throughout, concluding:

$$\begin{cases}\beta_{\mathrm{codom}}(\Phi_{K[1]}) > \beta_-(v), & \text{if } x, y \text{ have opposite signs,}\\ \beta_{\mathrm{codom}}(\Phi_{K[1]}) < \beta_+(v), & \text{if } x, y \text{ have same signs.}\end{cases}$$

□

Theorem 7.5 (Criterion for finiteness of CFMT of interest). *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ for a principally polarised complex abelian surface $\mathbb{T}$, with positive rank $\mathrm{ch}_0(v) > 0$ and $\Delta(v) > 0$.*

a. *Then the set of cohomological Fourier-Mukai transforms of the form $\Phi^H = ([g], -1)$ with $g \in \mathrm{PSL}(2,\mathbb{Z})$ such that $\Phi^H(v) = -v$ is finite if and only if $\Delta(v)/2$ is a square integer.*

b. *Furthermore, in the infinite case we can construct distinct $E_n \in \mathcal{D}^b(\mathbb{T}\times\mathbb{T})$, $n \in \mathbb{N}$ which are kernels of Fourier-Mukai transforms such that:*

$$\begin{aligned}\Phi_{E_n[1]}(v) &= -v \\ \beta_{\mathrm{dom}}(\Phi_{E_n[1]}) &< \beta_-(v) \qquad \textit{(strictly increasing)}\\ \lim_{n\to\infty} \beta_{\mathrm{dom}}(\Phi_{E_n[1]}) &= \beta_-(v).\end{aligned}$$

Remark. $\Delta(v)/2$ being a square integer is equivalent to $\beta_{\pm}(v)$ being rational. As previously mentioned in the beginning of this chapter, the Fourier-Mukai transforms Φ referenced in Theorem 2.7 (which preserve stability in a certain way) give CFMT of the form $([g], -1)$.

Proof. Again, without loss of generality, we can assume that v is irriducible due to the $\mathbb{Z}$-linearity of the actions of the CFMT. So let $v = (R, C\ell, D)$ with R, C, D being coprime

integers.

Statement a.

The condition on $g \in \mathrm{PSL}(2,\mathbb{Z})$ in the statement (just like in Proposition 7.4) is equivalent $g \in \mathfrak{st}(\gamma(v))$. So the finiteness of the set of these CFMT of interest is equivalent to the finiteness of $\mathfrak{st}(\gamma(v))$. Lemma 7.3 relates elements of $\mathfrak{st}(\gamma(v))$ to solutions of a Pell's equation (or generalised Pell's equation):

$$x^2 - (C^2 - RD)y^2 = \begin{cases} 1 & \text{if one of } R \text{ or } D \text{ is odd} \\ 4 & \text{if both } R \text{ and } D \text{ are even (and so } C \text{ is odd).} \end{cases} \tag{7.6}$$

In either case, these equations are known to have infinitely many solutions if and only if $\Delta^{\ell^2}(v) = C^2 - RD$ is not square. If it were a square, we could factorise the left hand side and consider the possible factorisations of 1 or 4, and verify that none of them yield nontrivial solutions (where $(x,y) \neq (1 \text{ or } 2, 0)$). The infinite cases are given by [AA15, Thm 3.2.1] in the case where one of R or D is odd. The other case, where both R and D are even, taking the doubles of each of the corresponding solutions of the Equation with 1 on the right hand side immediately gives us infinitely many solutions with 4 instead. However all solutions could also be found algorithmically as per [AA15, Section 4.4.2]. Note in fact that the first paragraph of the last citation refers to our context:

$$\begin{aligned} C^2 - RD &\equiv C^2 && \pmod 4 && \text{(as } R \text{ and } D \text{ both even),} \\ &\equiv 1 && \pmod 4 && \text{(as } C \text{ is odd).} \end{aligned}$$

Statement b.

In the case where one of R or D is odd and $C^2 - RD$ is not a perfect square, there is a minimal solution (x_1, y_1) of positive integers to Equation (7.6) (right hand side equal to 1), from which all other solutions (x_n, y_n) of positive integers can be generated:

$$\begin{pmatrix} x_n & \Delta^{\ell^2}(v)y_n \\ y_n & x_n \end{pmatrix} = \begin{pmatrix} x_1 & \Delta^{\ell^2}(v)y_1 \\ y_1 & x_1 \end{pmatrix}^n,$$

which satisfy $x_n/y_n \to \sqrt{\Delta^{\ell^2}(v)}$ monotonically (due to the solutions getting arbitrarily large monotonically). So then taking

$$g_n := \begin{pmatrix} (-x_n) - Cy_n & -Ry_n \\ Dy_n & (-x_n) + Cy_n \end{pmatrix},$$

and corresponding Fourier-Mukai Kernels $E_n \in \mathcal{D}^b(\mathbb{T} \times \mathbb{T})$ such that $\Phi^H_{E_n} = ([g_n], 1)$. Due to our constructions of g_n (with $(-x_n)$ and y_n having opposite signs), by Proposition 7.4 we have:

$$\begin{aligned} \Phi_{E_n[1]}(v) &= -v \\ \beta_{\mathrm{dom}}(\Phi_{E_n[1]}) &< \beta_-(v). \end{aligned}$$

Finally, for the limit of $\beta_{\mathrm{dom}}(\Phi_{E_n[1]})$:

$$\begin{aligned}\beta_{\mathrm{dom}}(\Phi_{E_n[1]}) &= \frac{-((-x_n)+Cy_n)}{-Ry_n}\\ &= \mu^{\ell}(v) - \frac{x_n}{Ry_n}\\ \lim_{n\to\infty}\beta_{\mathrm{dom}}(\Phi_{E_n[1]}) &= \mu^{\ell}(v) - \frac{\Delta^{\ell^2}(v)}{R}\\ &= \beta_{-}(v).\end{aligned}$$

In the case where R and D are both even, we could first generate a sequence of positive integer solutions to $x^2 - \Delta^{\ell^2}(v)y^2 = 1$, then double them to have a sequence of positive integer solutions to $x^2 - \Delta^{\ell^2}(v)y^2 = 4$. At that point, the same process produces the Fourier-Mukai transforms desired. Alternatively, using a refinement from [AA15, Section 4.4.2] would also give this result, but constructing a larger solution set along the way.

□

Chapter 8

Infinite Wall Criterion for Bogomolov non-Critical

In order to prove the sufficient condition for infinitely many walls in Theorem 9.6, this chapter shall first cover the case of Bogomolov non-critical Chern characters. Following this, Chapter 9 will then generalise the result to all other Chern characters.

8.1 Existence of Required Stable Sequences of Sheaves

Definition 8.1 (Bogomolov Criticality)**.** Let $v \in H^{\text{even}}_{\text{alg}}(\mathbb{T})$, with positive rank $\text{ch}_0(v) > 0$. Then we say that v is **Bogomolov critical** if $\Delta(v) \geq 0$, but $\Delta(v + (0, 0, 1)) < 0$.

Remark. When v has positive rank, then increasing $\text{ch}_2(v)$ will always decrease the value of $\Delta(v) = \text{ch}_1(v)^2 - 2\,\text{ch}_0(v)\,\text{ch}_2(v)$. In particular, if v has positive rank, and $\Delta(v) = 0$, then v is Bogomolov critical.

Lemma 8.1 (Short exact sequence for non-critical)**.** *Suppose that $v \in H^{\text{even}}_{\text{alg}}(\mathbb{T})$ has positive rank, $\Delta(v) \geq 0$ but is not Bogomolov critical. Then there exists a short exact sequence in $\text{Coh}(\mathbb{T})$ of Gieseker semistable objects, of the form:*

$$0 \to E \hookrightarrow F \twoheadrightarrow \mathcal{O}_x \to 0,$$

where $\text{ch}(E) = v$.

Proof. Let $w = v + \text{ch}(\mathcal{O}_x)$.

If $\Delta(w) = 0$, take $w = nu$ for $n \in \mathbb{N}$ and u irreducible. Take $F' \in M_L^{GSS}(u)$, which is non empty by [Muk78, Section 6]. Take

$$F := \bigoplus_{i=1}^{n} (\mathcal{P}_{\hat{x_i}} \otimes F') \qquad \text{for any choice of distinct } \hat{x_i} \in \hat{\mathbb{T}}.$$

Next choose epimorphisms from each direct summand to $\mathcal{O}_x$ giving an epimorphism $F \twoheadrightarrow \mathcal{O}_x$. Let E be the kernel, then we shall show it is Gieseker semistable, creating the short exact sequence required. Suppose, seeking a contraction, that E has a Gieseker destabiliser D, then D is also a sub-object of F which is Gieseker semistable. The small difference in $\text{ch}(E)$ and $\text{ch}(F)$ will force $\text{ch}(D) = mu$ for some $1 \leq m < n$: $\mu(D) \geq \mu(E)$ to destabilise E, and

$\mu(D) \geq \mu(F)$ to not destabilise F, but $\mu(E) = \mu(F)$ enforcing equality in both inequalities.

$$\begin{aligned} \text{Taking } u =: (r, c\ell, \chi), \quad & \text{we have:} \\ w &= (nr, nc\ell, n\chi) \\ v &= (nr, nc\ell, n\chi - 1) \\ \mathrm{ch}(D) &= (mr, mc\ell, \chi_u) \qquad\qquad \text{for some } \chi_u \in \mathbb{Z},\, m \in \mathbb{N}. \end{aligned}$$

Then, for K to destabilise E yet not destabilise F we need the lexicographic ordering:

$$\left(\frac{c}{r}, \frac{\chi}{r} - \frac{1}{rn}\right) < \left(\frac{c}{r}, \frac{\chi_u}{mr}\right) \leq \left(\frac{c}{r}, \frac{\chi}{r}\right).$$

This is equivalent to

$$\begin{aligned} \frac{\chi}{r} - \frac{1}{rn} < \frac{\chi_u}{mr} \leq \frac{\chi}{r}, \\ m\chi - \frac{m}{n} < \chi_u \leq m\chi. \end{aligned}$$

We have $m \leq n$ (K can not have larger rank than E as a sub-object), so the only integer value possible for χ_u satisfying the above is $\chi_u = m\chi$. This fixes $\mathrm{ch}(K) = mu$.

As stated in [YY09, Fact 2.10], which follows from [Muk78, Prop 6.15], K is S-equivalent to a direct sum of stable objects (semihomogeneous sheaves) of Chern character u. In particular, this gives a Gieseker semistabilising object of Chern character u which must also destabilise E. Therefore, without loss of generality, the destabilising object K has $\mathrm{ch}(K) = u$.

Next, we consider the inclusion $i\colon K \hookrightarrow F$ to show that the composition with $F \twoheadrightarrow \mathcal{O}_x$ must be non-zero, contradicting the fact that the former factors through the kernel of the latter. Taking $p_i\colon F \to \mathcal{P}_{\hat{x_i} \otimes F'}$ as the projections onto the direct summands, we have

$$p_i \circ i \in \hom(K, \mathcal{P}_{\hat{x_i}} \otimes F') \cong \begin{cases} \mathbb{C} & \text{if } K \cong \mathcal{P}_{\hat{x_i}} \otimes F', \\ 0 & \text{otherwise.} \end{cases}$$

by [HL10, Prop 1.2.7]. At least one $p_i \circ i$ must be non-zero since i is non-zero, and at most one can be non-zero as none of the direct summands of F are isomorphic to each other. So, up to scaling, K includes into F as a direct summand. The morphism $F \twoheadrightarrow \mathcal{O}_x$ was constructed so that post-composition with any inclusion of a direct summand (such as K) is non-zero. However this contradicts the fact that this inclusion factors through the kernel, and therefore, K cannot exist and E must therefore be Gieseker semistable giving the short exact sequence required.

If $\Delta(w) > 0$ then there exists a (strictly) Gieseker stable F with $\mathrm{ch}(f) = w$ (by a combination of [Yos01, Theorem 0.1] and [Yos99, Lemma 2.3]). Take any epimorphism $F \to \mathcal{O}_x$ and let E be the kernel. As seen in the previous case, any Gieseker destabiliser of E must semistabilise F, but F is stable. Therefore E is Gieseker stable and we have the required short exact sequence for the statement of the Lemma.

□

8.2 Generating Walls for Bridgeland Stabilities

Lemma 8.2 (Short exact sequence for non-critical right of $\boldsymbol{V_v}$). *Suppose that $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ has positive rank, $\Delta(v) \geq 0$ but is not Bogomolov critical. Then there exists a short exact sequence*

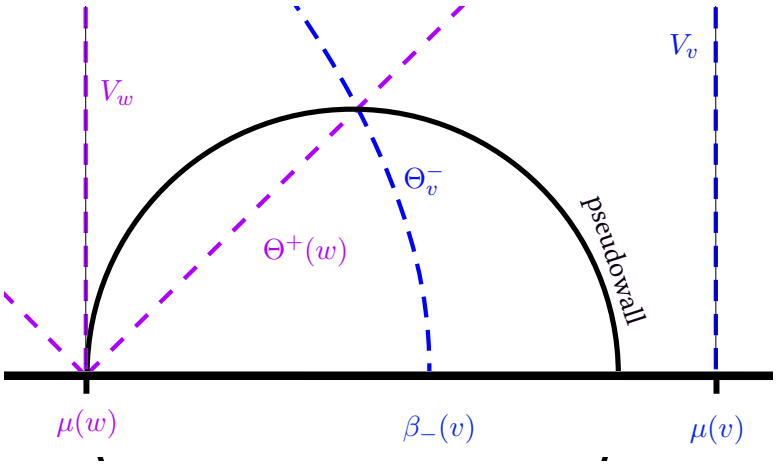


Figure 8.1: Pseudowall position

of stable objects in the unbounded chamber on the right of V_v, of the form:

$$0 \to F \hookrightarrow E \twoheadrightarrow \mathcal{O}_x \to 0,$$

where $\mathrm{ch}(E) = -v$.

Proof. Consider $v^\vee$, which satisfies the conditions of Lemma 8.1, and so we can take a short exact sequence of Gieseker stable sheaves

$$0 \to G \hookrightarrow H \twoheadrightarrow \mathcal{O}_x \to 0,$$

where $\mathrm{ch}(G) = v^\vee$. Applying the derived dual functor to the corresponding exact triangle and shifting the triangle twice gives:

$$H^\vee[1] \to G^\vee[1] \to \mathcal{O}_x \rightsquigarrow H^\vee[1].$$

By [JM19, Lemma 5.17], these are semistable objects in the unbounded region to the right of V_v, and so gives the required short exact sequence when taking cohomology with respect to $\mathcal{B}^\beta$ for sufficiently large β. □

Proposition 8.3 (Isotropic walls for Bogomolov non-critical)**.** *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ with positive rank, $\Delta(v) \geq 0$ but is not Bogomolov critical. For any Fourier-Mukai Transform of the form Φ_K, $K \in \mathrm{Coh}(\mathbb{T} \times \mathbb{T})$, satisfying $\Phi_K(v) = v$, and the inequality*

$$\beta_{\mathrm{dom}}\left(\Phi_{K[1]}\right) = \mu\left(\Phi^{-1}_{K[1]}(\mathcal{O}_x)\right) < \mu(v), \tag{8.1}$$

then

$$0 \to \Phi^{-1}_{K[1]}(F) \hookrightarrow \Phi^{-1}_{K[1]}(E) \twoheadrightarrow \Phi^{-1}_{K[1]}(\mathcal{O}_x) \to 0, \tag{8.2}$$

where E and F are from Lemma 8.2, gives rise to a wall for v, left of V_v, destabilising $\Phi^{-1}_{K[1]}(E)$ going 'down' Θ_v^-.

Remark. We will call such walls 'isotropic' because the destabiliser going 'upwards' is isotropic.

Proof. **Part 1: The pseudowall**
Consider the Chern characters of the hypothetical destabilising sequence given in Equation (8.2). All objects in the sequence satisfy the Bogomolov inequality because their pre-

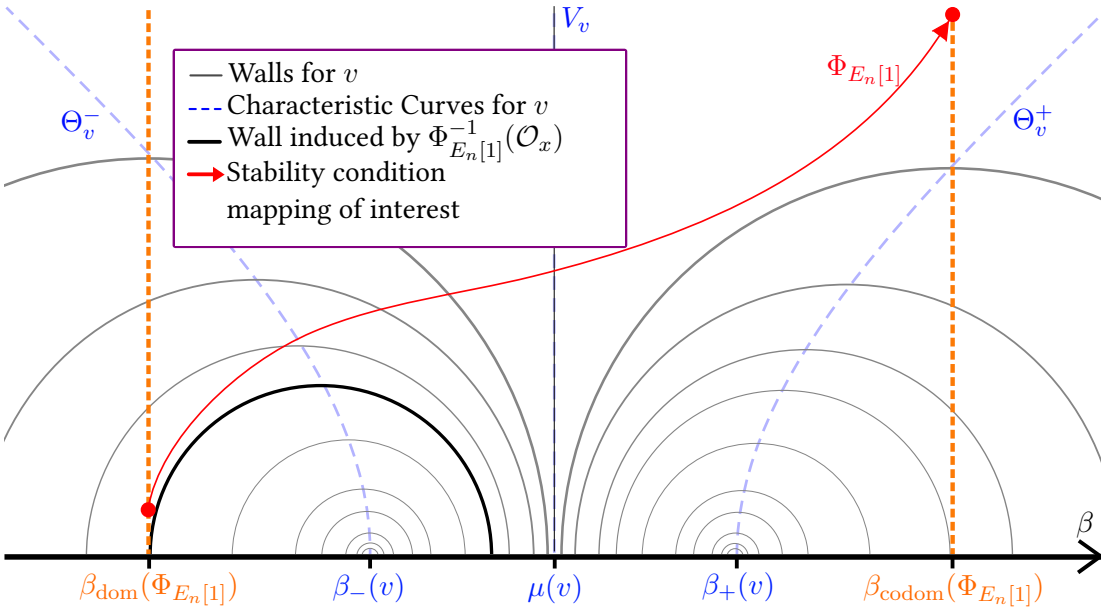


Figure 8.2: $\Phi^{-1}_{E_n[1]}$ takes a short exact sequence of stable objects in the unbounded chamber, right of V_v, to a stable sequence arbitrarily close to its corresponding pseudowall. Hence proving it is a real wall.

images under $\Phi_{K[1]}$ do (as they are stable objects in a chamber by Lemma 8.2). The condition from Equation (8.1) can be strengthened using Proposition 7.4 to:

$$\mu\left(\Phi^{-1}_{K[1]}(\mathcal{O}_x)\right) < \beta_-(v).$$

The pseudo-wall

$$\left\{(\beta,\alpha)\colon \alpha>0,\ \nu_{\alpha,\beta}(\Phi^{-1}_{K[1]}(\mathcal{O}_x)) = \nu_{\alpha,\beta}(\Phi^{-1}_{K[1]}(F))\right\},$$

forms a half circle centred on the β-axis and reaches its apex (maximum α-coordinate) on $\Theta_{\Phi^{-1}_{K[1]}(\mathcal{O}_x)}$, with $\nu_{\alpha,\beta}(\Phi^{-1}_{K[1]}(\mathcal{O}_x)) < \nu_{\alpha,\beta}(\Phi^{-1}_{K[1]}(F))$ on the inside. Since $\Delta(\Phi^{-1}_{K[1]}(\mathcal{O}_x)) = 0$, the hyperbola $\Theta_{\Phi^{-1}_{K[1]}(\mathcal{O}_x)}$ degenerates to two lines intersecting the β-axis at $\pm\pi/4$ rad at the base of $V_{\Phi^{-1}_{K[1]}(\mathcal{O}_x)}$. This forces the left side of this circle is tangential to this vertical line, $V_{\Phi^{-1}_{K[1]}(\mathcal{O}_x)}$, shown in Figure 8.1. Which, incidentally, is the set of points of the upper-half plane corresponding to the stability conditions that $\Phi_{K[1]}$ sends to the line

$$\left\{(\beta_{\mathrm{codom}}(\Phi_{K[1]}),\alpha)\colon \alpha>0\right\},$$

by Theorem 2.7.

Part 2: The pseudowall is real

The argument for this part is illustrated in Figure 8.2. It is sufficient to show that the objects of the sequence in Equation (8.2) are stable with respect to some stability conditions arbitrarily close to the pseudowall. The chosen stability conditions for this purpose will be

$$\left\{\sigma_{\alpha,\beta_{\mathrm{dom}}(\Phi_{K[1]})}\colon 1 \gg \alpha > 0\right\}.$$

Having the objects of the sequence stable with respect to these stability conditions is equiv-

alent to having F, E and $\mathcal{O}_x$ stable with respect to stability conditions

$$\left\{\sigma_{\alpha,\beta_{\mathrm{codom}}(\Phi_{K[1]})}\colon \alpha \gg 1\right\}.$$

The latter can be ensured by taking sufficiently large α, so that $\sigma_{\alpha,\beta_{\mathrm{codom}}(\Phi_{K[1]})}$ are in the unbounded chamber, right of V_v. Note that it is the right side of V_v by Lemma 9.2 since $\beta_{\mathrm{dom}}(\Phi_{K[1]}) < \mu(v)$. Finally, Lemma 8.2 gives us that F, E, and $\mathcal{O}_x$ are stable with respect to these stability conditions, as required.

□

Finally, to close this chapter, the following Theorem 8.4 provides a weaker version of the main Theorem (9.6) of this thesis. Recall from Bertram's nested wall Theorem 1.14, that circular walls for a Chern character $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ with $\Delta(v) \geq 0$, left of V_v could in principle accumulate (formally[1]) to the point $(\beta_-(v), 0)$. This Theorem provides the proof that this accumulation does indeed happen for a large class of v where $\beta_-(v)$ is irrational. Chapter 9 will then use this Theorem to transfer the destabilising sequences for these walls to destabilising sequences for the class of currently unconsidered v.

Theorem 8.4 (Criterion for the accumulation of wall to $\boldsymbol{\beta_-(v)}$ when irrational and $\boldsymbol{v}$ not Bogomolov Critical)**.** *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ with positive rank, $\Delta(v) \geq 0$ but is not Bogomolov critical.*

If $\beta_-(v)$ is not rational, then there is an infinite sequence of walls left of V_v accumulating to $(\beta_-(v), 0)$.

Proof. By Proposition 7.4, there is a sequence of Fourier-Mukai kernels $E_n \in \mathrm{Coh}(\mathbb{T} \times \mathbb{T})$, such that $\beta_{\mathrm{dom}}(\Phi_{E_n[1]}) < \mu(v)$ and $\lim_{n\to\infty} \beta_{\mathrm{dom}}(\Phi_{E_n[1]}) = \beta_-(v)$. Proposition 8.3 gives us, for each n, the existence of a wall for v, left of V_v, and tangential to the vertical line $\left\{\sigma_{\alpha,\beta_{\mathrm{dom}}(\Phi_{E_n[1]})}\colon \alpha > 0\right\}$. Finally, the limit $\beta_{\mathrm{dom}}(\Phi_{E_n[1]}) \to \beta_-(v)$ ensures that the walls accumulate to $(\beta_-(v), 0)$.

□

[1] This point in question does not give a stability condition, but in our parametrisation, the half circles accumulate in $\mathbb{R} \times \mathbb{R}_{\geq 0}$

Chapter 9

Transferring Isotropic Walls via FMTs

This final chapter of the thesis brings together the results of this Part III, to prove the main Theorem 9.6. This will be done by using Fourier-Mukai transforms to prove the existence of the destabilising sequences giving the walls, from the existence of those in the cases already considered in Chapter 8.

9.1 Non-Bogomolov Critical Transform

Previously, the infinite wall phenomenon was explored for Chern characters v, which were not Bogomolov critical, on a principally polarised surface $\mathbb{T}$ with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$. Concluding in Theorem 8.4. This was necessary to construct an object E, with $\mathrm{ch}(E) = v$ which fits in an exact sequence of Gieseker-stable coherent sheaves of the form:

$$0 \to E \hookrightarrow F \twoheadrightarrow \mathcal{O}_x \to 0.$$

Which was then used to generate the sequence of walls.

However there do exist elements of $H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ which are Bogomolov critical, yet $\beta_-(v)$ is still irrational. So none of the theorems so far allow us to conclude on whether there are finitely many walls. Furthermore, such examples may also have a Bogomolov critical (standard) Fourier-Mukai transform. Such as the following example 9.1.

Example 9.1 (Bogomolov critical Chern character and transform)**.** *Consider the choice $v = (6, 9\ell, 13) \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$. We have $\Delta(v) = 3$ which is not a square hence could potentially have infinitely many walls. However it is Bogomolov critical: $\Delta(v + \mathcal{O}_x) = -3$, and Fourier-Mukai transform is also: $\Delta(\Phi_{\mathcal{P}}(v) + \mathcal{O}_x) = -10$.*

However, we may still hope (and will indeed succeed at) applying the previous trick to prove existence of an infinite sequence of walls for some image under some different Fourier-Mukai transform. At that point, the Fourier-Mukai transform could then be used to relate this back to our original Chern character. The first step to find such a transform is covered in the next Theorem 9.1.

Theorem 9.1 (Existence of Non-Bogomolov Critical Transform)**.** *Let $\mathbb{T}$ be a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$ where $\ell := c_1(L)$ for a choice of ample line bundle L, and a positive $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ with $\Delta(v) > 0$.*

Then there exists a Fourier-Mukai Transform $\Phi\colon \mathbb{T} \to \mathbb{T}$ such that $\Phi^H(v)$ is positive and not Bogomolov critical (even if v is).

Proof. The required construction can be found in the proof of [Yos16, Proposition 4.2], which in our case is specialised to the following. Take $v = (r, c\ell, d)$ with $r > 0$ and $\Delta(v) > 0$. We claim by strong induction on r that v has a transform $(r', c'\ell, d')$ such that $r' \geq 0$ and $d' < 0$.

Suppose that $r = 0$, then $v \cdot \exp(n\ell) = (0, c\ell, d + 2cn)$ so n can be chosen such that $d + 2cn < 0$. For this n, we can choose $\Phi := \otimes L^{\otimes n}$ as required.

Suppose that the claim holds for all smaller r, then

$$\begin{aligned} v \cdot \exp(n\ell) &= (r, c\ell, d) \cdot (1, n\ell, n^2) \\ &= \left(r, (c + rn)\ell, r(n + c/r)^2 - \frac{c^2 - rd}{r^2}\right). \end{aligned}$$

Now n can be chosen so that $-1 \leq n + c/r < 0$. For this n we have

$$r(n + c/r)^2 - \frac{c^2 - rd}{r^2} < r,$$

because $\Delta(v) > 0$. If this value is negative, we can choose $\Phi := \otimes L^{\otimes n}$ as required. Otherwise, if it is non-negative, we can still conclude the claim by using the inductive hypothesis on

$$\Phi^H_{\mathcal{P}}(v \otimes L^{\otimes n}) = \left(r(n + c/r)^2 - \frac{c^2 - rd}{r^2}, -(c + rn)\ell, r\right),$$

which is still a positive Mukai vector even if the rank is 0 because of the choice of n guarantees that $-(c + rn) > 0$.

Now v has a transform $v' = (r', c'\ell, d')$ such that $r' \geq 0$ and $d' < 0$, we notice that v' is Bogomolov non-critical since $\Delta(v' + \mathrm{ch}(\mathcal{O}_x)) = 2(c'^2 - r'(d' + 1)) \geq 2c'^2 \geq 0$.

□

9.2 Transferring Isotropic Walls via FMTs

This section applies Theorem 2.7 in order to prove existence of walls, which we will call "isotropic", for a transform of a Chern character $\Phi^H(v)$ from the walls for the original Chern character v.

To do this, the specialisation of Theorem 2.7 to Φ^H, but also $\Phi^H \circ g$ for $g \in \mathfrak{st}_{\mathrm{SL}(2,\mathbb{Z})}(v)$ will be studied closer. In particular, relating the positions of the geometric stability conditions being acted on with the position of the characteristic curves V_v, Θ_v for v (and also for $\Phi^H(v)$).

Lemma 9.2 (Transferring stability condition to either side of V_*)**.** *Consider a kernel $E \in \mathrm{Coh}(\mathbb{T} \times \mathbb{T})$ of a Fourier-Mukai transform, and a Chern character $v \in H^{\mathrm{even}}(\mathbb{T})$.*

Suppose that the ranks $\mathrm{ch}_0(v)$ and $\mathrm{ch}_0(\Phi_E(v))$ are non-zero and of the same sign, then $\beta_{\mathrm{dom}}(\Phi_{E[1]})$ is not on the same side of V_v as $\beta_{\mathrm{codom}}(\Phi_{E[1]})$ is of $V_{\Phi_E(v)}$. That is:

$$\begin{aligned} \textit{either } \beta_{\mathrm{dom}}(\Phi_{E[1]}) \leq \mu(v) \textit{ and } \mu(\Phi(v)) \leq \beta_{\mathrm{codom}}(\Phi_{E[1]}), \\ \textit{or } \beta_{\mathrm{dom}}(\Phi_{E[1]}) \geq \mu(v) \textit{ and } \mu(\Phi_E(v)) \geq \beta_{\mathrm{codom}}(\Phi_{E[1]}). \end{aligned}$$

Otherwise, if the ranks $\mathrm{ch}_0(v)$ and $\mathrm{ch}_0(\Phi(v))$ are non-zero and have alternate signs, then they are on the same side:

$$\begin{aligned} \beta_{\mathrm{dom}}(\Phi_{E[1]}) \leq \mu(v) \textit{ and } \beta_{\mathrm{codom}}(\Phi_{E[1]}) \leq \mu(\Phi_E(v)), \\ \textit{or } \beta_{\mathrm{dom}}(\Phi_{E[1]}) \geq \mu(v) \textit{ and } \beta_{\mathrm{codom}}(\Phi_{E[1]}) \geq \mu(\Phi_E(v)). \end{aligned}$$

Remark. This Lemma statement involves both Φ_E and $\Phi_{E[1]}$ ($= \Phi_E[1]$). It could have been stated just in terms of one, or the other. However I shall continue to use Φ_E when used to act on Chern characters since this can be expressed as an action of $\mathrm{SL}(2,\mathbb{Z})$ on $\mathrm{Symm}(2,\mathbb{Z})$ (from Definition 7.1), directly admiting results from elsewhere in this document without sign change. The Fourier-Mukai transform $\Phi_{E[1]}$ is used when acting on stability conditions, in order to use identity involving the stability conditions in our slice of interest (from Theorem 2.7) directly. Note also that $\mu(\Phi_{E[n]}(v))$ is independent of n so either shifts could have been writter there.

Proof. Take $v = (R, C\ell, D)$ and $\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \Phi_E^H$.
Consider the sign of product of the offsets of $\beta_{\mathrm{dom}}(\Phi)$ from $\mu(v)$ and that of $\beta_{\mathrm{codom}}(\Phi)$ from $\mu(\Phi(v))$. This product is non-negative iff the offsets have the same sign (allowing 0 for either sign):

$$\begin{aligned}
&\left(\beta_{\mathrm{dom}}(\Phi_{E[1]}) - \mu(v)\right)\left(\beta_{\mathrm{codom}}(\Phi_{E[1]}) - \mu(\Phi_E(v))\right) \\
&= \left(-\frac{C}{R} - \frac{a}{b}\right)\left(\frac{Ra + Cb}{(Ra^2 + 2\,Cab + Db^2)b}\right) \\
&= -\frac{(Ra + Cb)^2}{(Ra^2 + 2\,Cab + Db^2)Rb^2} \\
&= \frac{-(Ra + Cb)^2}{b^2 R\,\mathrm{ch}_0(\Phi_E(v))}.
\end{aligned}$$

In particular, the sign of this product (allowing 0 for either sign) is determined by whether R and $\mathrm{ch}_0(\Phi_E(v))$ have the same sign. This then allows us to conclude the possibilities for the signs of the offsets between β_{dom} (resp. β_{codom}) from $\mu(v)$ (resp. $\mu(\Phi(v))$) as stated in the lemma.

□

Proposition 9.3 ($\boldsymbol{\beta_{\mathrm{dom}}}$ when precomposing by stabilisers of $\boldsymbol{v}$). *Let $v = (R, C\ell, D)$ be a Chern character with positive rank and $\Delta(v) > 0$. Let $E, F_n \in \mathrm{Coh}(\mathbb{T}\times\mathbb{T})$ be kernels of a Fourier-Mukai transforms, where:*

- *$\Phi_E^H(v)$ has positive rank.*
- *$\Phi_{F_n}^H$ are stabiliser elements for v, hence, of the form:*

$$\Phi_{F_n}^H = \begin{pmatrix} -Cy_n + x_n & Ry_n \\ -Dy_n & Cy_n + x_n \end{pmatrix} \qquad \begin{array}{r} \text{where } x_n, y_n \in \mathbb{Q}, \\ \text{with } {x_n}^2 - \Delta^{\ell^2}(v){y_n}^2 = 1. \end{array}$$

- *The F_n are chosen so that $\frac{x_n}{y_n} \to \pm\sqrt{\Delta^{\ell^2}(v)}$ as $n \to \infty$ (choosing one of $+$ or $-$).*
  Or equivalently: $\beta_{\mathrm{dom}}(\Phi_{F_n[1]}) \to \beta_{\pm}(v)$.

Then considering the compositions $\Phi_E \circ \Phi_{F_n}$ (which all map v to the same Chern character) we have:

$$\begin{aligned} \beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}) &\to \beta_{\mp}(v) \\ \beta_{\mathrm{codom}}(\Phi_E \circ \Phi_{F_n}) &\to \beta_{\pm}(\Phi_E(v)) \end{aligned} \qquad \text{as } n \to \infty.$$

Proof. **Part 1: $\boldsymbol{\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}) \to \beta_{\mp}(v)}$**

Modulo the sign (β_- or β_+), this is equivalent to showing:

$$R(\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}) - \mu(v)) \to \mp\sqrt{\Delta^{\ell^2}(v)}$$

$$\Phi_E^H = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{SL}(2,\mathbb{Z}) \tag{9.1}$$

$$\Phi_{F_n}^H = \begin{pmatrix} -Cy_n + x_n & Ry_n \\ -Dy_n & Cy_n + x_n \end{pmatrix} \tag{9.2}$$

$$\Phi_E^H \Phi_{F_n}^H = \begin{pmatrix} -Cay_n - Dby_n + ax_n & Ray_n + Cby_n + bx_n \\ -Ccy_n - Ddy_n + cx_n & Rcy_n + Cdy_n + dx_n \end{pmatrix}. \tag{9.3}$$

$$\begin{aligned} R(\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}) - \mu(v)) &= R\left(-\frac{C^2by_n - DRby_n + Rax_n + Cbx_n}{(Ray_n + Cby_n + bx_n)R}\right) \\ &= \left(-\frac{C^2b - DRb + Ra\frac{x_n}{y_n} + Cb\frac{x_n}{y_n}}{Ra + Cb + b\frac{x_n}{y_n}}\right) \\ &= -\frac{C^2b - DRb + (Ra + Cb)\frac{x_n}{y_n}}{Ra + Cb + b\frac{x_n}{y_n}} \end{aligned}$$

$$\begin{aligned} \lim_{n\to\infty}(\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}) - \mu(v)) &= -\frac{C^2b - DRb \pm (Ra + Cb)\sqrt{\Delta^{\ell^2}(v)}}{Ra + Cb \pm b\sqrt{\Delta^{\ell^2}(v)}} \\ &= -\frac{b\sqrt{\Delta^{\ell^2}(v)}^2 \pm (Ra + Cb)\sqrt{\Delta^{\ell^2}(v)}}{Ra + Cb \pm b\sqrt{\Delta^{\ell^2}(v)}} \\ &= \mp\frac{\pm b\sqrt{\Delta^{\ell^2}(v)} + (Ra + Cb)}{Ra + Cb \pm b\sqrt{\Delta^{\ell^2}(v)}}\sqrt{\Delta^{\ell^2}(v)} \\ &= \mp\sqrt{\Delta^{\ell^2}(v)} \end{aligned}$$

The computer algebra code verifying this can be found in Appendix C.8.2.

Justification for the limit:

A subtlety skipped over in the calculation above when computing the limit is that I assume that the limit of the denominator is not 0. That is, that we never have:

$$Ra + Cb \pm b\sqrt{\Delta^{\ell^2}(v)} = 0.$$

This is equivalent to:

$$(Ra + Cb)^2 = b^2\Delta^{\ell^2}(v).$$

Which rearranges to:

$$\begin{aligned}
0 &= R^2a^2 + 2RCab + C^2b^2 - C^2b^2 + b^2RD \\
&= -R\Delta\left(v, (b^2, -ab\ell, a^2\ell^2)\right) \\
&= -R\Delta\left(v, (\Phi_E^H)^{-1}(\mathrm{ch}(\mathcal{O}_x))\right) \\
&= -R\Delta\left(\Phi_E^H(v), \mathrm{ch}(\mathcal{O}_x)\right) \\
&= R\,\mathrm{ch}_0\left(\Phi_E^H(v)\right).
\end{aligned}$$

This is a contradiction, as R and $\mathrm{ch}_0\left(\Phi_E^H(v)\right)$ are both positive by assumption.

Part 2: $\beta_{\mathbf{codom}}(\Phi_E \circ \Phi_{F_n}) \to \beta_{\pm}(\Phi_E(v))$

A similar calculation can be done (which can also be found at C.8.2), but this can also be spotted by considering the inverse and applying Lemma 9.2.

$$\begin{aligned}
\beta_{\mathrm{codom}}(\Phi_E \circ \Phi_{F_n}) &= \beta_{\mathrm{dom}}((\Phi_E \circ \Phi_{F_n})^{-1}) \\
&= \beta_{\mathrm{dom}}(((\Phi_E \circ \Phi_{F_n} \circ \Phi_E^{-1}) \circ \Phi_E)^{-1}) \\
&= \beta_{\mathrm{dom}}(\Phi_E^{-1} \circ (\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}))
\end{aligned}$$

Since $\Phi_E^{-1}(\Phi_E(v)) = v$ and $\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}$ is a stabiliser for $\Phi_E(v)$, we can apply the first part of this Proposition to show that

$$\beta_{\mathrm{dom}}(\Phi_E^{-1} \circ (\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1})) \to \beta_{\pm}(\Phi_E(v)),$$

provided that $\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}) \to \beta_{\mp}(\Phi_E(v))$.

But in principle, given that $\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}$ are distinct (cohomologically), we can only guarantee that $\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1})$ can be spliced into subsequences that each converge to one of $\beta_-(\Phi_E(v))$ or $\beta_+(\Phi_E(v))$. However Lemma 9.2 ensures that a tail of

$$\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}) = \beta_{\mathrm{codom}}(\Phi_E \circ \Phi_{F_n} \circ \Phi_E^{-1}),$$

is strictly greater than $\mu(\Phi_E(v))$ when $\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n} \circ \Phi_E^{-1}) \to \beta_-(v)$ (or is strictly smaller when $\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}) \to \beta_+(v)$). Recall that $\beta_-(u) < \mu(u) < \beta_+(u)$ for $u = v$ or $\Phi_E(v)$ (as $\Delta(v) = \Delta(\Phi_E(v)) > 0$).

We can then conclude that

$$\beta_{\mathrm{dom}}(\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1}) = \beta_{\mathrm{codom}}(\Phi_E \circ \Phi_{F_n} \circ \Phi_E^{-1}) \to \beta_{\mp}(\Phi_E(v)),$$

$$\beta_{\mathrm{codom}}(\Phi_E \circ \Phi_{F_n}) = \beta_{\mathrm{dom}}(\Phi_E^{-1} \circ (\Phi_E \circ \Phi_{F_n}^{-1} \circ \Phi_E^{-1})) \to \beta_{\pm}(\Phi_E(v)).$$

□

Lemma 9.4 (Transferring all left-walls to right-walls via FMTs when $\beta_-(v) \notin \mathbb{Q}$)**.** *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ be a Chern character with positive rank $\mathrm{ch}_0(v) > 0$, non-negative Bogomolov discriminant $\Delta(v) \geq 0$, and irrational $\beta_-(v)$. Then v has infinitely many walls left of V_v if and only if there are infinitely many walls on the right of V_v.*

Remark. As will be shown in Theorem 9.6, only the infinite case is possible. This Lemma is a stepping stone to that result.

Proof. Consider Proposition 9.3 applied with the identity as the choice of Fourier-Mukai transform, and the sequence of transforms fixing v given by Theorem 7.5.

Then we have a sequence of Fourier-Mukai transforms Φ_n, such that $\beta_{\mathrm{dom}}(\Phi_n) \to \beta_-(v)$ and $\beta_{\mathrm{codom}}(\Phi_n) \to \beta_+(v)$. Suppose there are k walls on the left side of V_v. Using the fact that $\beta_{\mathrm{dom}}(\Phi_n) \to \beta_-(v)$, we can choose n such that these k walls are intersected at k distinct point by the vertical line of stability conditions:

$$\left\{\sigma_{\alpha,\beta_{\mathrm{dom}}(\Phi_n)} : \alpha \in \mathbb{R}_{>0}\right\}.$$

Given the action on stability conditions given in Definition 2.6, There are therefore k wall points on the line

$$\left\{\sigma_{\alpha,\beta_{\mathrm{codom}}(\Phi_n)} : \alpha \in \mathbb{R}_{>0}\right\},$$

which is on the right-hand side of V_v. Therefore, if there are infinitely many walls on the left, then there are also infinitely many on the right. Using the second fact $\beta_{\mathrm{codom}}(\Phi_n) \to \beta_+(v)$, we can make the same argument in the other direction.

□

Proposition 9.5 (Transferring isotropic walls via FMTs for Chern characters $\boldsymbol{v}$ with irrational $\boldsymbol{\beta_-(v)}$)**.** *Let $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ be a Chern character with positive rank $\mathrm{ch}_0(v) > 0$, non-negative Bogomolov discriminant $\Delta(v) \geq 0$, and irrational $\beta_-(v)$.*

If we have a Fourier-Mukai transform of the form Φ_E with $E \in \mathrm{Coh}(\mathbb{T} \times \mathbb{T})$, such that $\Phi^H_E(v)$ has positive rank too; then v has infinitely many walls if and only if $\Phi^H_E(v)$ does.

Remark. The condition $\Delta(v) \geq 0$ with $\beta_-(v) \notin \mathbb{Q}$ is equivalent to $\Delta^{\ell^2}(v)$ being a positive, non-square integer. Φ^H_E being an isometry of the Mukai lattice implies that $\Phi^H_E(v)$ satisfies all the same conditions as v.

Proof. Due to the symmetry of the situation, and Lemma 9.4, it is sufficient to show that if v has at least k walls on the left side of V_v, then $v' := \Phi^H_E(v)$ has at least k walls on the right side of $V_{v'}$.

In a similar way to the proof in Lemma 9.4, we can apply Proposition 9.3 with the Φ_E, and the sequence of transforms fixing v given by Theorem 7.5.

Then we have a sequence of Fourier-Mukai transforms Φ_n, such that $\beta_{\mathrm{dom}}(\Phi_n) \to \beta_-(v)$ and $\beta_{\mathrm{codom}}(\Phi_n) \to \beta_+(v')$. This gives a bijection between the walls of v intersecting

$$\left\{\sigma_{\alpha,\beta_{\mathrm{dom}}(\Phi_n)} : \alpha \in \mathbb{R}_{>0}\right\},$$

and the walls of v' intersecting

$$\left\{\sigma_{\alpha,\beta_{\mathrm{codom}}(\Phi_n)} : \alpha \in \mathbb{R}_{>0}\right\}.$$

We can choose n such the first line intersect arbitrarily many walls for v left of V_v. And conversely, n can be chosen so that the second line intersections arbitrarily many walls for v' right of $V_{v'}$. Giving us the Proposition statement.

□

9.3 Clarifying the Criterion for the Accumulation of ν-walls

Finally we can tie all the results in the past chapters together in order conclude the main theorem which gave the title to this thesis:

Theorem 9.6 (Criterion for the accumulation of $\boldsymbol{\nu}$-walls on principally polarised abelian surfaces)**.** *Let $\mathbb{T}$ be a principally polarised abelian complex surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, and $v \in H^{\mathrm{even}}_{\mathrm{alg}}(\mathbb{T})$ be a Chern character with non-negative rank $\mathrm{ch}_0(v) > 0$, non-negative Bogomolov discriminant $\Delta(v) \geq 0$.*

Then:

$$v \text{ has finitely many } \nu\text{-walls} \iff \beta_-(v) \in \mathbb{Q}$$

where $\beta_-((r, c\ell, d)) = \frac{c-\sqrt{c^2-rd}}{r}$.

Proof. In Part II, it was shown constructively that $\beta_-(v) \in \mathbb{Q}$ implies v has finitely many pseudowalls, and hence finitely many walls by Lemma 3.1. But this was already well known as a consequence to [LQ14, Thm 1.1], [Mac14, Prop 4.2], [MYY13, Lemma 5.20].

For the converse, suppose that $\beta_-(v) \notin \mathbb{Q}$, and in particular, must have strictly positive rank. Theorem 9.1 gives us the existence of a Fourier-Mukai transform of the form Φ_E such that $\Phi_E^H(v)$ has positive rank, non-negative Bogomolov discriminant, and is not Bogomolov critical. Theorem 8.4 gives us that $\Phi_E^H(v)$ has infinitely many walls. Finally Proposition 9.5 allows us to conclude that v must also have infinitely many walls.

□

9.3.1 Comparison with Existing Result from Yoshioka

Earlier comparisons in this thesis with works involving Yoshioka Kōta, such as in Chapter 6, referred to results for Chern characters under the assumption that they satisfy a so-called "numerical condition". However, there also exists a result [Yos16, Corollary 4.3] which is a stronger version of Theorem 9.6 in this thesis, applying a larger class of Picard rank 1 abelian surfaces. That paper even considers the example $(2, \ell, -2)$ for that result in [Yos16, Section 4.2] which is exactly the counterexample used in this thesis, at the start of Chapter 6, for a Chern character which does not satisfy the numerical condition but still has infinitely many walls according to Theorem 9.6.

The argument in that paper for that particular result follows a similar strategy to the one in this thesis:

1. Infinite subgroup of Cohomological Fourier-Mukai transforms stabilising the Chern character in the given class: Theorem 7.5 and [Yos16, Lemma 2.7].
2. Existence of a transform which has at least one wall: Theorem 9.1 and [Yos16, Proposition 4.2].
3. Conclude that the class of Chern characters has infinitely many walls: Theorem 9.6 and [Yos16, Corollary 4.3].

Despite these similarities, this thesis aims to provide an alternative perspective on some particular details, with more illustrations about the action of the Fourier-Mukai transforms on the space of stability conditions. Certain parts of this thesis also provide slightly more explicit statements compared to their counterparts in [Yos16] which was done to help in the development of a future computer program to calculate the sequence of walls in the case where there are infinitely many.

Conclusion

The main outcomes of this thesis were Theorem 9.6, and a vastly better performing algorithm for computing possibilities for tilt walls when there are finitely many on a smooth projective Picard rank 1 surface. While Theorem 9.6 had been proved before ([Yos16, Corollary 4.3]), Part III provides some alternative arguments with many statements being more explicit in order to facilitate the writing of a future program to implement the results.

Other than the theoretical conclusion in the thesis, there are also some other outcomes when it comes to accessibility of associated computations and ideas for further development leading on from the work here. These are discussed in the next sections.

Associated Computational Libraries

On the computational side, the library developed as part of this thesis [Nay23b], was designed to help with more accessible quick computations to find possibilities for pseudo-walls and pseudo-semistabilisers. For immediate and graphical use, in the case of principally polarised abelian surfaces with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, there is a webapp hosted on [Nay24c]. Performing computations in Python or SageMath with these results can be done with the Python wrapper [Nay23a]. However for other surfaces, currently the only way of using code associated to this thesis is to use the Rust crate [Nay23b] directly. A project to allow the reader to try this out without setting up the Rust toolchain on their computer is the repository [Nay24a]. This repository contains a configuration for GitHub Codespaces to run the software in the cloud (green 'code' button ⟩'codespaces' ⟩'create codespace on main'). At the time of writing, 60 hours of usage per month is provided for free by GitHub.

Further Afield

This section discusses potential generalisations or extensions of this work which could be applied in the context of strong stability conditions on other surfaces or threefolds.

Other Surfaces

Naturally, there is a question of how much this generalises to other smooth projective surfaces. Part II about finite walls in the case $\beta_-(v) \in \mathbb{Q}$ generalises to smooth projective Picard rank 1 surfaces, so this question comes down to when infinite sequences of walls can be generated in other cases. For abelian surfaces X with $\mathrm{NS}(X) = \mathbb{Z}\ell$, it has been shown for $\ell^2 \leq 8$, by [Yos16], that a Chern character v has infinite walls if and only if $\beta_-(v)$ is irrational. For $\ell^2 > 8$, that same paper also provides a counterexample (slight modification of [Yos16, Lemma 3.48]) where $\beta_-(v)$ is irrational, yet v has no walls. In these cases, it comes down to whether $\beta_-(v)$ has any wall at all. The infinite wall phenomenon has also recently been observed for abelian surfaces with Picard rank greater than 1 [CNY24, Section 4.1].

For other smooth projective surfaces, the strategy in this thesis will only be applicable if the group of autoequivalences of the derived category of coherent sheaves is large enough for there to be infinite stabiliser subgroups, and if a single wall can be constructed in the cases where we want to show the existence of infinitely many walls.

Validating the Existence of Walls with Gieseker Stable Sequences

The method used to explicitly construct the destabilising sequence corresponding to a wall for which we wanted to show the existence, heavily relies on the quotient of the destabilising sequence having a isotropic Chern character. This was necessary so that the stability conditions we were mapping could be arbitrarily close to the hypothetical wall, so we could then equate the existence of the wall to the existence of a short exact sequence of Gieseker stable sheaves with specific Chern characters. See Figure 8.1. However when working with stability conditions modulo $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$, where wall points are still well defined, Theorem 2.8 gives a more complete picture. This viewpoint could be used to relate the existence of other walls, where the destabiliser has larger Bogomolov form, to the existence of Gieseker stable sequences of sheaves with specific Chern characters.

Hyperbolic Geometry on $\mathrm{Stab}(X)/\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$

Theorem 2.8 showed that, for a principally polarised abelian surface $\mathbb{T}$ with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$, there exists a complex manifold structure on $\mathrm{Stab}(\mathbb{T})/\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$, where autoequivalences of $\mathcal{D}^b(\mathbb{T})$ act via holomorphic functions. In this case more specifically, this came from the standard action of the modular group. This has also been observed on K3 surfaces [Kaw19, Theorem 3.3], and a more general metric on stability conditions of a triangulated category has also been shown to coincide with this hyperbolic metric in the case of smooth projective surfaces in [Woo11, Section 4]. This viewpoint was not instrumental for the results of this thesis, but in hindsight could have vastly simplified the argument presented, with compelling visuals.

This viewpoint could potentially also be applied elsewhere, as was done implicitly in [Fey18, Section 4] which benefitted from a reparametrisation of the half plane of stability conditions, almost like how the Klein disk relates to the Poincaré half-plane as hyperbolic models, so that the walls became straight lines segments. One specific idea would be to consider the fundamental domain of stability conditions under the action of stabiliser subgroup of the autoequivalences of $\mathcal{D}^b(X)$ which fixes a given Chern character. In the context of this thesis, a fundamental domain can be chosen as the area bounded between two consecutive walls generated in Proposition 7.4. In other contexts, where less is known, restricting attention to a smaller region of stability conditions could be beneficial.

Applications to λ-Stability on Threefolds

On threefolds, tilt walls are also considered, potentially benefiting from an adaptation of the computations from this thesis. But even when considering λ-stability, there is a potential for there to be applications for the computational libraries from the 'associated computational libraries' section. This is due to [JM, Lemma 4.5] (with the 'real' analogue from [Sch20a, Theorem 6.1]), which states that for any pseudo λ-wall for a Chern character v intersecting the characteristic curve Θ_v, there is a pseudo ν-wall going through that intersection point. This would allow us to compute the possibilities for truncations of λ-semistabilisers that give such walls, and then after studying other restrictions such as those in [BMT13], could yield finitely many solutions in certain cases.

The idea in Part III of considering the actions of stabilisers of a given Chern character v, on the space of λ-stability conditions could also be pursued.

Bibliography

[24] In: *Branch Predictor*. Feb. 19, 2024. URL: https://en.wikipedia.org/w/index.php?title=Branch_predictor&oldid=1209022299 (visited on 04/01/2024).

[AA15] Titu Andreescu and Dorin Andrica. *Quadratic Diophantine Equations*. Vol. 40. Developments in Mathematics. New York, NY: Springer, 2015. ISBN: 978-0-387-35156-8 978-0-387-54109-9. DOI: 10.1007/978-0-387-54109-9. URL: https://link.springer.com/10.1007/978-0-387-54109-9 (visited on 06/10/2024).

[ABL07] Daniele Arcara, Aaron Bertram, and Max Lieblich. *Bridgeland-Stable Moduli Spaces for K-Trivial Surfaces*. Aug. 16, 2007. DOI: 10.48550/arXiv.0708.2247. arXiv: 0708.2247 [math]. URL: http://arxiv.org/abs/0708.2247 (visited on 08/03/2024). Pre-published.

[BMT13] Arend Bayer, Emanuele Macri, and Yukinobu Toda. "Bridgeland Stability Conditions on Threefolds I: BG Type Inequalities". In: (2013).

[BM11] Arend Bayer and Emanuele Macrì. "The Space of Stability Conditions on the Local Projective Plane". In: *Duke Mathematical Journal* 160.2 (Nov. 2011), pp. 263–322. ISSN: 0012-7094, 1547-7398. DOI: 10.1215/00127094-1444249. URL: https://projecteuclid.org/journals/duke-mathematical-journal/volume-160/issue-2/The-space-of-stability-conditions-on-the-local-projective-plane/10.1215/00127094-1444249.full (visited on 05/23/2024).

[Bri07] Tom Bridgeland. "Stability Conditions on Triangulated Categories". In: *Annals of mathematics* 166.2 (2007), pp. 317–345. ISSN: 0003-486X. URL: https://www-jstor-org.eux.idm.oclc.org/stable/20160065?sid=primo.

[Bri08] Tom Bridgeland. "Stability Conditions on K3 Surfaces". In: *Duke Mathematical Journal* 141.2 (2008), pp. 241–291. DOI: 10.1215/S0012-7094-08-14122-5. URL: https://doi.org/10.1215/S0012-7094-08-14122-5.

[CNY24] Izzet Coskun, Howard Nuer, and Kota Yoshioka. *Weak Brill-Noether on Abelian Surfaces*. Aug. 12, 2024. DOI: 10.48550/arXiv.2408.06095. arXiv: 2408.06095 [math]. URL: http://arxiv.org/abs/2408.06095 (visited on 08/13/2024). Pre-published.

[Del23] Hannah Dell. *Stability Conditions on Free Abelian Quotients*. Version 2. Aug. 1, 2023. DOI: 10.48550/arXiv.2307.00815. arXiv: 2307.00815 [math]. URL: http://arxiv.org/abs/2307.00815 (visited on 05/23/2024). Pre-published.

[Fey18] Soheyla Feyzbakhsh. "Bridgeland Stability Conditions, Stability of the Restricted Bundle, Brill-Noether Theory and Mukai's Program". In: (Nov. 29, 2018). URL: https://era.ed.ac.uk/handle/1842/31485 (visited on 08/08/2024).

[FLZ22] Lie Fu, Chunyi Li, and Xiaolei Zhao. *Stability Manifolds of Varieties with Finite Albanese Morphisms*. Jan. 20, 2022. DOI: 10.48550/arXiv.2103.07728. arXiv: 2103.07728 [math]. URL: http://arxiv.org/abs/2103.07728 (visited on 05/23/2024). Pre-published.

[Gel96] S. I. Gel'fand. *Methods of Homological Algebra / Sergei I. Gelfand, Yuri I. Manin.* Berlin ; Springer, 1996. ISBN: 978-3-540-54746-4.

[HRS96] Dieter Happel, Idun Reiten, and Sverre O. Smalo. *Tilting in Abelian Categories and Quasitilted Algebras*. Providence, UNITED STATES: American Mathematical Society, 1996. ISBN: 978-1-4704-0160-3. URL: http://ebookcentral.proquest.com/lib/ed/detail.action?docID=3113737 (visited on 06/14/2024).

[Har13] Robin Hartshorne. *Algebraic Geometry*. Springer Science & Business Media, June 29, 2013. 511 pp. ISBN: 978-1-4757-3849-0. Google Books: 7z4mBQAAQBAJ.

[Huy06a] Daniel Huybrechts. *Derived and Abelian Equivalence of K3 Surfaces*. Apr. 6, 2006. DOI: 10.48550/arXiv.math/0604150. arXiv: math/0604150. URL: http://arxiv.org/abs/math/0604150 (visited on 05/09/2024). preprint.

[Huy06b] Daniel Huybrechts. *Fourier-Mukai Transforms in Algebraic Geometry*. Oxford, UNITED KINGDOM: Oxford University Press, Incorporated, 2006. ISBN: 978-0-19-151635-1. URL: http://ebookcentral.proquest.com/lib/ed/detail.action?docID=3052126 (visited on 07/19/2024).

[HL10] Daniel Huybrechts and Manfred Lehn. *The Geometry of Moduli Spaces of Sheaves*. 2nd ed. Cambridge Mathematical Library. Cambridge: Cambridge University Press, 2010. ISBN: 978-0-521-13420-0. DOI: 10.1017/CBO9780511711985. URL: https://www.cambridge.org/core/books/geometry-of-moduli-spaces-of-sheaves/E69325DA1892E9BA762E354C4C64E337 (visited on 04/17/2024).

[HMS08] Daniel Huybrechts, Emanuele Macrì, and Paolo Stellari. "Stability Conditions for Generic K3 Categories". In: *Compositio Mathematica* 144.1 (Jan. 2008), pp. 134–162. ISSN: 1570-5846, 0010-437X. DOI: 10.1112/S0010437X07003065. URL: https://www.cambridge.org/core/journals/compositio-mathematica/article/stability-conditions-for-generic-k3-categories/A959FD370AB714D7920DC46E7603CAE6 (visited on 07/29/2024).

[JM19] Marcos Jardim and Antony Maciocia. "Walls and Asymptotics for Bridgeland Stability Conditions on 3-Folds". In: (2019). URL: https://arxiv.org/abs/1907.12578.

[JM] Marcos Jardim and Antony Maciocia. "Walls and Asymptotics for Bridgeland Stability Conditions on 3-Folds". In: *Épijournal de géométrie algébrique* 6 (). ISSN: 2491-6765. DOI: 10.46298/epiga.2022.6819.

[Kaw19] Kotaro Kawatani. "A Hyperbolic Metric and Stability Conditions on $\mathrm{K}3$ Surfaces with $\rho = 1$". In: *Communications in Analysis and Geometry* 27.6 (Oct. 2019), pp. 1325–1354. ISSN: 1944-9992. DOI: 10.4310/CAG.2019.v27.n6.a5. URL: https://www-intlpress-com.eux.idm.oclc.org/site/pub/pages/journals/items/cag/content/vols/0027/0006/a005/abstract.php (visited on 08/12/2024).

[LQ14] Jason Lo and Zhenbo Qin. "Mini-Walls for Bridgeland Stability Conditions on the Derived Category of Sheaves over Surfaces". In: *The Asian journal of mathematics* 18.2 (2014), pp. 321–344. ISSN: 1093-6106. URL: https://arxiv.org/abs/1103.4352.

[Mac14] Antony Maciocia. "Computing the Walls Associated to Bridgeland Stability Conditions on Projective Surfaces". In: (Mar. 31, 2014). URL: https://arxiv.org/abs/1202.4587.

[MS19] Emanuele Macrì and Benjamin Schmidt. *Lectures on Bridgeland Stability*. Oct. 30, 2019. DOI: 10.48550/arXiv.1607.01262. arXiv: 1607.01262 [math]. URL: http://arxiv.org/abs/1607.01262 (visited on 04/17/2024). Prepublished.

[Mea12a] Ciaran Meachan. "Moduli of Bridgeland-Stable Objects". The University of Edinburgh, 2012. URL: https://era.ed.ac.uk/handle/1842/6230.

[Mea12b] Ciaran. Meachan. *Moduli of Bridgeland-Stable Objects / Ciaran Meachan.* Place of publication not identified: [publisher not identified], 2012.

[MYY13] Hiroki Minamide, Shintarou Yanagida, and Kōta Yoshioka. "Some Moduli Spaces of Bridgeland's Stability Conditions". In: *International Mathematics Research Notices* 2014.19 (June 2013), pp. 5264–5327. ISSN: 1073-7928. DOI: 10.1093/imrn/rnt126. eprint: https://academic.oup.com/imrn/article-pdf/2014/19/5264/18895160/rnt126.pdf. URL: https://doi.org/10.1093/imrn/rnt126.

[23] *Modular Group*. In: *Wikipedia*. Nov. 8, 2023. URL: https://en.wikipedia.org/w/index.php?title=Modular_group&oldid=1184112985 (visited on 08/07/2024).

[Muk78] Shigeru Mukai. "Semi-Homogeneous Vector Bundles on an Abelian Variety". In: *Kyoto journal of mathematics* 18.2 (1978). ISSN: 2156-2261. DOI: 10.1215/kjm/1250522574. URL: https://projecteuclid.org/journals/kyoto-journal-of-mathematics/volume-18/issue-2/Semi-homogeneous-vector-bundles-on-an-abelian-variety/10.1215/kjm/1250522574.full.

[Muk81] Shigeru Mukai. "Duality between D(X) and D(\hat X) with Its Application to Picard Sheaves". In: *Nagoya mathematical journal* 81 (1981), pp. 153–175. ISSN: 2152-6842. DOI: 10.1017/S002776300001922X. URL: https://projecteuclid.org/journals/nagoya-mathematical-journal/volume-81/issue-none/Duality-between-DX-and-Dhat-X-with-its-application-to/nmj/1118786312.full.

[Nay23a] Luke Naylor. *Lnay/Pseudo_tilt_py*. May 24, 2023. URL: https://github.com/lnay/pseudo_tilt_py (visited on 05/11/2024).

[Nay23b] Luke Naylor. *Pseudo-Wall Finder, Try by Scanning QR Code at Top-Right*. GitLab, 2023. URL: https://gitlab.com/pseudowalls/tilt.rs.

[Nay24a] Luke Naylor. *Lnay/Tilt-Rs-Notebook*. Jan. 23, 2024. URL: https://github.com/lnay/tilt-rs-notebook (visited on 05/11/2024).

[Nay24b] Luke Naylor. *Pseudo_tilt - Rust*. July 18, 2024. URL: https://pseudowalls.gitlab.io/tilt.rs/pseudo_tilt/ (visited on 07/18/2024).

[Nay24c] Luke Naylor. *Pseudowalls Page on Luke Naylor's Site*. 2024. URL: https://lukideangeometry.xyz/pseudowalls (visited on 05/11/2024).

[Nee23] Tristan Needham. *Visual Complex Analysis / Tristan Needham ; Foreword by Roger Penrose*. 25th anniversary edition. Oxford: University Press, 2023. ISBN: 978-0-19-196494-7. URL: https://academic.oup.com/book/45765.

[Niv66] Ivan Niven. "An Introduction to the Theory of Numbers / Ivan Niven, Herbert S. Zuckerman." In: *An Introduction to the Theory of Numbers*. Second edition. New York ; Wiley, 1966.

[The22] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.6.0)*. manual. 2022.

[Sch20a] Benjamin Schmidt. "Bridgeland Stability on Threefolds: Some Wall Crossings". In: *Journal of algebraic geometry* 29.2 (2020), pp. 247–283. ISSN: 1056-3911.

[Sch20b] Benjamin Schmidt. *Stability_conditions, SageMath Library*. GitHub, 2020. URL: https://github.com/benjaminschmidt/stability_conditions.

[Woo11] Jon Woolf. "Some Metric Properties of Spaces of Stability Conditions". In: *arXiv.org* (Aug. 12, 2011). DOI: 10.1112/blms/bds056. URL: https://www.proquest.com/docview/2083872669?pq-origsite=primo&parentSessionId=I1PlwqQX6QLa4XZgWZnq3Hi9aitjEryzbB5Vf%2BxaUlE%3D&sourcetype=Working%20Papers (visited on 08/12/2024).

[YY09] Shintarou Yanagida and Kōta Yoshioka. "Semi-Homogeneous Sheaves, Fourier-Mukai Transforms and Moduli of Stable Sheaves on Abelian Surfaces". In: *Journal für die reine und angewandte Mathematik* 2013.684 (684 2009), pp. 31–86. ISSN: 0075-4102. DOI: 10.48550/arxiv.0906.4603. URL: https://arxiv.org/abs/0906.4603.

[YY14] Shintarou Yanagida and Kōta Yoshioka. "Bridgeland's Stabilities on Abelian Surfaces". In: *Mathematische Zeitschrift* 276.1-2 (2014), pp. 571–610. ISSN: 0025-5874.

[Yos01] Kōta Yoshioka. "Moduli Spaces of Stable Sheaves on Abelian Surfaces". In: *Mathematische annalen* 321.4 (4 2001), pp. 817–884. ISSN: 0025-5831. DOI: 10.1007/s002080100255. URL: https://link-springer-com.eux.idm.oclc.org/article/10.1007/s002080100255.

[Yos16] Kōta Yoshioka. "Bridgeland's Stability and the Positive Cone of the Moduli Spaces of Stable Objects on an Abelian Surface". In: *Development of Moduli Theory — Kyoto 2013*. Vol. 69. Mathematical Society of Japan, Jan. 1, 2016, pp. 473–538. DOI: 10.2969/aspm/06910473. URL: https://projecteuclid.org/ebooks/advanced-studies-in-pure-mathematics/Development-of-Moduli-Theory--Kyoto-2013/chapter/Bridgelands-stability-and-the-positive-cone-of-the-moduli-spaces/10.2969/aspm/06910473 (visited on 01/21/2025).

[Yos99] Kota Yoshioka. "Irreducibility of Moduli Spaces of Vector Bundles on K3 Surfaces". In: (1999). DOI: 10.48550/arxiv.math/9907001. URL: https://arxiv.org/abs/math/9907001 (visited on 04/23/2024).

Appendices

Appendix A

Computing Pseudowalls Program

This appendix contains the source code of the `pseudo_tilt` program, to accompany Part II, to compute solutions to Problem 2 (currently only for a principally polarised abelian surface with $\mathrm{NS}(\mathbb{T}) = \mathbb{Z}\ell$). The code is hosted on GitLab at `https://gitlab.com/pseudowalls/tilt.rs`, the revision shown here is commit 6689d9a0. The reader may find it more ergonomic to view the code via the online docs (only available for latest version).

A.1 Library, Utils CLI Frontend

Listing A.1: `main`

```rust
#![allow(uncommon_codepoints)]

use bitflags::bitflags;
use pseudo_tilt::chern_character::{ChernChar, Δ};
use pseudo_tilt::tilt_stability::left_pseudo_semistabilizers::find_all;
use pseudo_tilt::tilt_stability::β_;
use std::env;

const M: u32 = 2;

bitflags! {
    #[derive(Clone, Copy, Debug, PartialEq, Eq, Hash)]
    struct RunOptions: u8 {
        const SEMISTABILIZERS = 0b1;
        const WALLS = 0b10;
        const UNSAFE = 0b100;
        const DATA = 0b1000;
    }
}

fn parse_args() -> Result<(ChernChar<M>, RunOptions), &'static str> {
    let mut opts = RunOptions::empty();

    // Collect first three integer-parsible arguments, handling other
    // valid arguments on the way: --<option>
    let int_args: Vec<i64> = env::args()
        .skip(1) // skip executable name
        .filter_map(|arg| match arg.parse::<i64>() {
            Err(_e) => {
                match arg.as_str() {
                    "--semistabilizers" => {
                        opts |= RunOptions::SEMISTABILIZERS;
                    }
```

```
                    "--walls" => {
                        opts |= RunOptions::WALLS;
                    }
                    "--unsafe" => {
                        opts |= RunOptions::UNSAFE;
                    }
                    "--data" => {
                        opts |= RunOptions::DATA;
                    }
                    _ => {
                        eprintln!("Non-integer argument: {}", arg);
                    }
                }
                return None;
            }
            Ok(n) => {
                return Some(n);
            }
        })
        .take(3)
        .collect();

    if int_args.len() != 3 {
        eprintln!("Usage: pseudo_tilt [options] r c d");
        eprintln!("    r, c, d integers");
        eprintln!("Options:");
        eprintln!("  --semistabilizers");
        eprintln!("  --walls");
        eprintln!("  --unsafe");
        eprintln!("  --data");
        eprintln!("");
        eprintln!("Incorrect order or format might be ignored without warning");

        return Err("Invalid CLI args");
    }

    if !opts.intersects(RunOptions::SEMISTABILIZERS | RunOptions::WALLS) {
        return Err("One of --semistabilizers or --walls must be specified");
    }

    if opts.contains(RunOptions::WALLS) {
        return Err("--walls not implemented yet");
    }

    if opts.contains(RunOptions::UNSAFE) {
        return Err("--unsafe no longer implemented");
    }

    return Ok((
        ChernChar::<M>::from ((
            int_args[0],
            int_args[1],
            int_args[2],
        )),
        opts,
    ));
}

fn main() -> Result<(), &'static str> {
    // Attempt to parse arguments
    let (v, run_opts) = parse_args()?;

    if !run_opts.contains(RunOptions::DATA) {
        // Preliminary info about given Chern character:
```

```
        println!("Chern Character v = {}", v);
        println!("Δ(v) = {}", Δ(&v));

        if let Some(β) = β_(&v) {
            println!("β_(v) = {}", β);
        } else {
            println!("β_(v) is irrational");
        }
    }

    // Act on flags specified:

    if run_opts.contains(RunOptions::SEMISTABILIZERS) {
        if run_opts.contains(RunOptions::DATA) {
            for u in find_all(&v)? {
                let r = u.r;
                let c: i64 = u.c.into();
                let d: i64 = u.d.raw_int_repn();
                println!("{}\t{}\t{}", r, c, d);
            }
        } else {
            println!("");
            println!("pseudo-semistabilizers:");
            for u in find_all(&v)? {
                println!("{u}");
            }
        }
    }
    Ok(())
}
```

Listing A.2: `lib`

```
#![feature(iterator_try_collect)]
#![feature(test)]
#![allow(uncommon_codepoints)]
#![allow(soft_unstable)]

pub mod chern_character;
pub mod tilt_stability;
pub mod utils;
```

Listing A.3: `utils`

```
use num::{rational::Rational64, integer::Roots};
use std::cmp::Ordering::{Equal, Greater, Less};

/// Returns least integer strictly greater than `frac`
pub fn least_greater_int(frac: Rational64) -> i64 {
    if frac.is_integer() {
        return frac.to_integer() + 1;
    }
    // Otherwise, not an int

    // Sign of `frac` needs to be checked because `to_integer` rounds towards
    // 0 instead of 'up' or 'down' consistently
    match frac.cmp(&Rational64::from_integer(0)) {
        Greater => frac.to_integer() + 1,
        Less => frac.to_integer(),
        Equal => 1, // should already be covered by integer case
    }
}

```

```rust
/// Returns greatest integer less than, or equal to, `frac`
pub fn greatest_lesser_or_eq_int(frac: Rational64) -> i64 {
    if frac.is_integer() {
        return frac.to_integer();
    }
    // Otherwise, not an int

    // Sign of `frac` needs to be checked because `to_integer` rounds towards
    // 0 instead of 'up' or 'down' consistently
    match frac.cmp(&Rational64::from_integer(0)) {
        Greater => frac.to_integer(),
        Less => frac.to_integer() - 1,
        Equal => -1, // should already be covered by integer case
    }
}

/// Crude placeholder implementation of modular inverse
/// Returns m s.t. (n*m) % modulus == 1 if exists, else None
pub fn modulo_inverse(n: i64, modulus: i64) -> Result<i64, &'static str> {
    if n < 0 || modulus <= 1 {
        // THOUGHT: maybe better to have upfront typechecking via `modulus`
        // crate types
        return Err("both arguments to modulo_inverse must be positive");
    }
    let mut inverse = 1;
    loop {
        let next = (inverse * n) % modulus;
        if next == 1 {
            return Ok(inverse);
        }
        if next == 0 {
            return Err("modulo inverse does not exist");
        }
        inverse = next;
    }
}

// Custom alternative to rarges like (1..), which allow for infinite
// sequences, but with negative step

struct LinearSeq<VAL, STEP> where VAL: std::ops::AddAssign<STEP> + Copy {
    current: VAL,
    step: STEP,
}

impl<VAL, STEP: Copy> Iterator for LinearSeq<VAL, STEP>
where VAL: std::ops::AddAssign<STEP> + Copy {
    type Item = VAL;

    fn next(&mut self) -> Option<Self::Item> {
        let result = self.current;
        self.current += self.step;
        Some(result)
    }
}

pub fn linear_seq<VAL, STEP: Copy>(start: VAL, step: STEP)
-> impl Iterator<Item = VAL>
where VAL: std::ops::AddAssign<STEP> + Copy {
    LinearSeq {
        current: start,
        step,
    }
}
```

```rust

// FUTURE: implement something like this for Chern2 directly
// (in chern_character::terms)
pub fn checked_sqrt(q: Rational64) -> Option<Rational64> {
    let numer = q.numer();
    let denom = q.denom();
    match (checked_sqrt_i64(*numer), checked_sqrt_i64(*denom)) {
        (Some(a), Some(b)) => Some((a, b).into()),
        _ => None,
    }
}

fn checked_sqrt_i64(n: i64) -> Option<i64> {
    if n < 0 {
        return None;
    }
    let potential_sqrt = n.sqrt();
    if potential_sqrt * potential_sqrt == n {
        Some(potential_sqrt)
    } else {
        None
    }
}
```

A.2 Chern Character Submodule

Listing A.4: `chern_character`

```rust
#![allow(dead_code)]

pub mod terms;

use num::rational::Rational64;
use std::fmt::{Display, Formatter};
use serde::{Serialize, Deserialize};
use terms::{Chern0, Chern1, Chern2, cup_product};

/// Represents Chern character of Picard rank 1 surface
#[derive(PartialEq, Eq, Clone, Copy, Debug, Serialize, Deserialize)]
pub struct ChernChar<const M: u32> {
    pub r: terms::Chern0,
    pub c: terms::Chern1,
    pub d: terms::Chern2<M>,
}

impl<const M: u32> ChernChar<M> {
    pub fn new(r: Chern0, c: Chern1, d: Chern2<M>) -> ChernChar<M> {
        ChernChar::<M> { r, c, d }
    }
}
impl<const M: u32> Default for ChernChar<M> {
    fn default() -> ChernChar<M> {
        ( 1, 0, 0 ).into()
    }
}

impl<const M: u32> From<(i64, i64, i64)> for ChernChar<M> {
    fn from(tuple:(i64,i64,i64)) -> ChernChar<M> {
        let (r, c, d) = tuple;
        let r: Chern0 = r;
        let c: Chern1 = c.into();
        let d: Chern2<M> = d.into();
        ChernChar::<M> { r, c, d }
```

```rust
    }
}

impl<const M: u32> Display for ChernChar<M> {
    fn fmt(&self, f: &mut Formatter<'_>) -> std::fmt::Result {
        write!(f, "({}, {}, {})", self.r, self.c, self.d)
    }
}

// Define addition for Chern Characters on the same variety
impl<const M: u32> std::ops::Add for ChernChar<M> {
    type Output = Self;
    /// Addition of Chern Characters (done component-wise)
    fn add(self, other: Self) -> Self {
        Self {
            r: self.r + other.r,
            c: self.c + other.c,
            d: self.d + other.d,
        }
    }
}
impl<const M: u32> std::ops::Sub for ChernChar<M> {
    type Output = Self;
    /// Addition of Chern Characters (done component-wise)
    fn sub(self, other: Self) -> Self {
        Self {
            r: self.r - other.r,
            c: self.c - other.c,
            d: self.d - other.d,
        }
    }
}

// Multiplication for Chern Characters on the same variety
impl<const M: u32> std::ops::Mul for ChernChar<M> {
    type Output = Self;
    /// Multiplication of Chern Characters
    /// (corresponds to tensor product of underlying objects)
    fn mul(self, other: Self) -> Self {
        Self {
            r: self.r * other.r,
            c: self.r * other.c + other.r * self.c,
            d: self.r * other.d + other.r * self.d
                + cup_product(self.c, other.c),
        }
    }
}

// Left scalar multiplication for ChernChar
impl<const M: u32> std::ops::Mul<ChernChar<M>> for i64 {
    type Output = ChernChar<M>;
    /// multiplication done componentwise
    fn mul(self, other: Self::Output) -> Self::Output {
        let scalar: i64 = self.into();
        Self::Output::new (
            scalar * other.r,
            scalar * other.c,
            scalar * other.d,
        )
    }
}

/// Mumford slope (as a coefficient of ℓ²):
/// ```equation
```

```rust
///      c
/// μ  = -   for v = (r, c·ℓ, d·ℓ²)
///      r
///```
pub fn μ<const M: u32>(v: &ChernChar<M>) -> Rational64 {
    (v.c.into(), v.r).into()
}

/// Bogomolov discriminant (as a coefficient of ℓ²):
/// ```equation
/// Δ  = c² - 2rd          for v = (r, c·ℓ, d·ℓ²)
///    = c² - rd'          for v = (r, c·ℓ, d'·½ℓ²)
///```
#[allow(non_snake_case)]
pub fn Δ<const M: u32>(v: &ChernChar<M>) -> Chern2<M> {
    cup_product::<M>(v.c, v.c) - 2 * v.r * v.d
}


#[cfg(test)]
mod tests {
    use super::*;

    #[test]
    fn display() {
        let r: Chern0 = 1.into();
        let c: Chern1 = 2.into();
        let d: Chern2<2> = (2*2*2).into();
        let v = ChernChar::<2> { r, c, d };
        assert_eq!(format!("{}", v), "(1, 2ℓ, 4ℓ²)");
    }

    #[test]
    fn chern_slope() {
        const M: u32 = 2;
        let v: ChernChar<M> = (1, 1, 1).into();
        assert_eq!(μ(&v), 1.into());
        let v: ChernChar<M> = (2, 4, 5).into();
        assert_eq!(μ(&v), 2.into());
        let v: ChernChar<M> = (3, -1, -3).into();
        assert_eq!(μ(&v), (-1, 3).into());
    }

    #[test]
    fn bogomolov_discriminant() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (1, 1, 1).into();
        assert_eq!(format!("{}", v), "(1, 1ℓ, 1/2ℓ²)");
        assert_eq!(Δ(&v), 0.into());
        let v: ChernChar::<M> = (2, 4, -5).into();
        assert_eq!(Δ(&v), ((16 + 10)*(M as i64)).into());
        let v: ChernChar::<M> = (3, -1, -3).into();
        assert_eq!(Δ(&v), ((1 + 9)*(M as i64)).into());
    }

    #[test]
    fn scalar_mult() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (2, 4, 5).into();
        let scalar: i64 = 3;
        let scalar_mult: ChernChar::<M> = (6, 12, 15).into();
        assert_eq!(scalar_mult, scalar * v);
    }

```

```
    #[test]
    fn chern_mult() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (2, 4, 5).into();
        let o = ChernChar::<M>::default(); // structure sheaf

        // check multiplication by structure sheaf does nothing
        assert_eq!(v, o * v);
        assert_eq!(v, v * o);

        // try non-trivial multiplication
        let v_times_v: ChernChar::<M> = (
            2 * 2,
            2 * 4 + 2 * 4,
            5 * 2 + 5 * 2 + 2 * 4 * 4,
        ).into();
        assert_eq!(v_times_v, v * v);
    }
}
```

Listing A.5: `chern_character::terms`

```
use std::fmt::{Display, Formatter};
use num::rational::Rational64;
use serde::{Serialize, Deserialize};
use crate::utils::{greatest_lesser_or_eq_int, least_greater_int};

pub type Chern0 = i64;

// Chern 1 and type conversions //
// ===== = === ==== =========== //
#[derive(Clone, Copy, Debug, PartialEq, Eq, Serialize, Deserialize)]
pub struct Chern1 {
    coeff: i64, // multiple of ℓ
}

impl From<i64> for Chern1 {
    fn from(coeff: i64) -> Self {
        Self{coeff}
    }
}

impl From<Chern1> for i64 {
    fn from(ch1: Chern1) -> i64 {
        ch1.coeff
    }
}

// Chern 2 and type conversions //
// ===== = === ==== =========== //

// M must be a positive even integer
#[derive(Clone, Copy, Debug, PartialEq, Eq, Serialize, Deserialize)]
pub struct Chern2<const M: u32> {
    coeff: i64, // multiple of ℓ²/M
}

impl<const M: u32> From<i64> for Chern2<M> {
    fn from(coeff: i64) -> Self {
        Self{coeff}
    }
}

impl<const M: u32> Chern2<M> {
```

```
    pub fn ℓ2_coeff(self) -> Rational64 {
        (self.coeff, M as i64).into()
    }
    pub fn raw_int_repn(self) -> i64 {
        self.coeff
    }
}

pub fn least_greater_chern2<const M: u32>(rat: Rational64) -> Chern2<M> {
    least_greater_int(rat * (M as i64)).into()
}

pub fn greatest_lesser_or_eq_chern2<const M: u32>(rat: Rational64) -> Chern2<M> {
    greatest_lesser_or_eq_int(rat * (M as i64)).into()
}

pub fn inclusive_range<const M: u32>(start: Chern2<M>, end: Chern2<M>)
-> impl Iterator<Item=Chern2<M>> {
    (start.coeff..=end.coeff)
        .map(|c| c.into())
}

// Multiplications between Chern Terms //
// =============== ======= ===== ===== //

// define multiplication between Chern0 and Chern1
impl std::ops::Mul<Chern1> for Chern0 {
    type Output = Chern1;
    fn mul(self, other: Chern1) -> Self::Output {
        Self::Output{ coeff: self * other.coeff }
    }
}
// define multiplication between Chern0 and Chern2
impl<const M: u32> std::ops::Mul<Chern2<M>> for Chern0 {
    type Output = Chern2<M>;
    fn mul(self, other: Chern2<M>) -> Self::Output {
        Self::Output{ coeff: self * other.coeff }
    }
}

// This couldn't be done with an operator because generic parameter of output
// not determined by inputs
pub fn cup_product<const M: u32>(a: Chern1, b: Chern1) -> Chern2<M> {
    Chern2::<M>{ coeff: a.coeff * b.coeff * (M as i64) }
}

// Addition/Subtraction among Chern Terms //
// ================== ===== ===== ===== //

// define addition of Chern terms
impl std::ops::Add for Chern1 {
    type Output = Chern1;
    fn add(self, rhs: Chern1) -> Self::Output {
        Self { coeff: self.coeff + rhs.coeff }
    }
}
impl std::ops::Sub for Chern1 {
    type Output = Chern1;
    fn sub(self, rhs: Chern1) -> Self::Output {
        Self { coeff: self.coeff - rhs.coeff }
    }
}
impl std::ops::AddAssign for Chern1 {
    fn add_assign(&mut self, rhs: Chern1) {
```

```rust
        self.coeff += rhs.coeff;
    }
}

impl<const M: u32> std::ops::Add for Chern2<M> {
    type Output = Chern2<M>;
    fn add(self, rhs: Self) -> Self::Output {
        Self { coeff: self.coeff + rhs.coeff }
    }
}
impl<const M: u32> std::ops::Sub for Chern2<M> {
    type Output = Chern2<M>;
    fn sub(self, rhs: Self) -> Self::Output {
        Self { coeff: self.coeff - rhs.coeff }
    }
}

// Display for Chern Terms //
// ======= === ===== ===== //

impl Display for Chern1 {
    fn fmt(&self, f: &mut Formatter<'_>) -> std::fmt::Result {
        write!(f, "{}ℓ", self.coeff)
    }
}
impl<const M: u32> Display for Chern2<M> {
    fn fmt(&self, f: &mut Formatter<'_>) -> std::fmt::Result {
        write!(f, "{}ℓ²", self.ℓ2_coeff())
    }
}

#[cfg(test)]
mod tests {
    use super::*;

    #[test]
    fn chern_terms() {
        let r: Chern0 = 1.into();
        let c: Chern1 = 2.into();
        let d: Chern2::<2> = (2*2*2).into();
        assert_eq!(r * r, r);
        assert_eq!(r * c, c);
        assert_eq!(cup_product::<2>(c, c), d);
        assert_eq!(Rational64::from_integer(2*2), d.ℓ2_coeff());
    }

    #[test]
    fn display() {
        let r: Chern0 = 1.into();
        let c: Chern1 = 2.into();
        let d: Chern2<2> = (2*2*2).into();
        assert_eq!(format!("{}", r), "1");
        assert_eq!(format!("{}", c), "2ℓ");
        assert_eq!(format!("{}", d), "4ℓ²");
    }

    #[test]
    fn chern2_ranges() {
        let start = least_greater_chern2::<2>((-1,3).into());
        let end = greatest_lesser_or_eq_chern2::<2>((1,3).into());
        let range = inclusive_range(start, end).collect::<Vec<_>>();
        assert_eq!(
            range,
            vec![
```

```
                0.into(),
            ]
        );
        let start = least_greater_chern2::<4>((-1,3).into());
        let end = greatest_lesser_or_eq_chern2::<4>((1,3).into());
        let range = inclusive_range(start, end).collect::<Vec<_>>();
        assert_eq!(
            range,
            vec![
                (-1).into(),
                0.into(),
                1.into(),
            ]
        );
    }
}
```

A.3 Tilt Stability Submodule

Listing A.6: `tilt_stability`

```rust
use super::chern_character::{ChernChar, Δ};
use num::CheckedDiv;
use num::rational::Rational64;
use crate::utils::checked_sqrt;

pub mod left_pseudo_semistabilizers;
pub mod wall;
pub mod twisted;

extern crate test;

/// Returns β_ for given Chern character v, that is:
/// ```equation
///              _________
///       c - √ c² - 2rd
/// β_  = --------------   for v = (r, c·ℓ, d·ℓ²) and r≠0
///              r
///
///        d
/// β_  = ---              for v = (0, c·ℓ, d·ℓ²) and c≠0
///        c
///
/// β_  = +∞               otherwise
///```
/// ..., but only if rational (ϵℚ), otherwise returns None
pub fn β_<const V: u32>(v: &ChernChar<V>) -> Option<Rational64> {
    if v.r == 0 {
        // Rank 0 case uses alternative formula, and returns None
        // when divides by 0
        return v.d.ℓ2_coeff()
            .checked_div(&Rational64::from_integer(v.c.into()));
        // ^ β_ = +∞ considered here with potential `None` return
    }
    // term inside of sqrt in β_ formula:
    let sqrt_part: Rational64 = checked_sqrt(Δ(&v).ℓ2_coeff())?;

    Some(
        (Rational64::from_integer(v.c.into()) - sqrt_part) / v.r
    )
}


```

```rust
#[cfg(test)]
mod tests {
    use super::*;
    use test::Bencher;

    #[test]
    fn beta_minus() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (3, 2, -4).into();
        assert_eq!(Δ(&v), (16 * M as i64).into());
        assert_eq!(β_(&v), Some(Rational64::from(-2) / 3));
        // rank 0 case
        let v: ChernChar::<M>  = (0, 2, -4).into();
        assert_eq!(Δ(&v), (4 * M as i64).into());
        assert_eq!(β_(&v), Some(Rational64::from(-1)));
    }
    // TODO test considered_q
    #[test]
    fn wall() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (1, 0, 0).into();
        let u: ChernChar::<M> = (1, 1, 1).into();
        assert_eq!(wall::centre(&v, &u), Some(Rational64::from_integer(1) / 2));
        assert_eq!(wall::radius2(&v, &u), Some(Rational64::from_integer(1) / 4));
        // rank 0 case
        let v: ChernChar::<M> = (0, 1, 0).into();
        let u: ChernChar::<M> = (1, 1, 1).into();
        assert_eq!(wall::centre(&v, &u), Some(Rational64::from_integer(0)));
        assert_eq!(wall::radius2(&v, &u), Some(Rational64::from_integer(1)));
        let v: ChernChar::<M> = (0, 1, 0).into();
        let u: ChernChar::<M> = (4, 2, 1).into();
        assert_eq!(wall::centre(&v, &u), Some(Rational64::from_integer(0)));
        assert_eq!(wall::radius2(&v, &u), Some(Rational64::from_integer(1)/4));
    }

    #[test]
    fn structure_sheaf_example() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (1, 0, 0).into();
        let semistabilizers = left_pseudo_semistabilizers::find_all(&v);
        assert_eq!(semistabilizers, Ok(vec![]));
    }

    #[test]
    fn nontrivial_example() {
        const M: u32 = 2;
        let v: ChernChar::<M> = (3, 2, -4).into();
        let semistabilizers = left_pseudo_semistabilizers::find_all(&v);
        assert_eq!(
            semistabilizers,
            Ok(vec![
                (1,0,0).into(),
                (4,-2,1).into(),
                (2,0,0).into(),
                (8,-4,2).into(),
                (4,-1,0).into(),
                (7,-3,1).into(),
                (16,-9,5).into(),
                (25,-15,9).into(),
                (3,0,-1).into(),
                (3,0,0).into(),
                (12,-6,3).into(),
                (2,1,-2).into(),
                (2,1,-1).into(),
```

```
                (2,1,0).into(),
                (5,-1,0).into(),
                (8,-3,1).into(),
                (11,-5,2).into(),
                (4,0,-1).into(),
                (4,0,0).into(),
                (7,-2,0).into(),
                (10,-4,1).into(),
                (19,-10,5).into(),
                (3,1,-2).into(),
                (3,1,-1).into(),
                (3,1,0).into(),
                (6,-1,-1).into(),
                (5,0,-2).into(),
                (4,1,-3).into()
            ])
        );
    }

    #[bench]
    fn many_solutions(b: &mut Bencher) {
        const M: u32 = 2;
        let v: ChernChar::<M> = (45,54,-41).into();
        b.iter(|| left_pseudo_semistabilizers::find_all(&v));
    }
}
```

Listing A.7: `tilt_stability::twisted`

```
//! Namespace for functions giving twisted Chern character parts.

use crate::chern_character::ChernChar;
use num::Rational64;

/// Returns ch₁ β (v) as a coefficient of ℓ
pub fn chern_1<const M: u32>(v: &ChernChar<M>, β: Rational64) -> Rational64 {
    -β * v.r + i64::from(v.c)
}

/// Returns ch₂ β (v) as a coefficient of ℓ²
pub fn chern_2<const M: u32>(v: &ChernChar<M>, β: Rational64) -> Rational64 {
    v.d.ℓ2_coeff() - β * i64::from(v.c) + β.pow(2) * v.r / 2
}
```

Listing A.8: `tilt_stability::wall`

```
//! Namespace for functions giving info about pseudowalls.
//! Typically taking two Chern characters as arguments
//! (corresponding to object being destabilized and the destabilizer)
use crate::chern_character::ChernChar;
use num::rational::Rational64;
use num::CheckedDiv;
use std::cmp::Ordering;

/// calculates the beta coordinate for centre of circle where u and v have same tilt
/// slope
pub fn centre<const M: u32>(u: &ChernChar<M>, v: &ChernChar<M>)
-> Option<Rational64> {
    let divisor: i64 = i64::from(v.c) * u.r - v.r * i64::from(u.c);
    (v.d.ℓ2_coeff() * u.r - u.d.ℓ2_coeff() * v.r)
        .checked_div(&Rational64::from_integer(divisor))
}

```

```rust
/// Calculates the radius squared of centre of circle where u and v have same tilt
/// slope
///   returning None instead of any infinite or negative value
pub fn radius2<const M: u32>(u: &ChernChar<M>, v: &ChernChar<M>)
-> Option<Rational64> {
    let v_c = i64::from(v.c);
    let u_c = i64::from(u.c);
    let beta_coeff: Rational64 = v.d.ℓ2_coeff() * 2 * u.r - u.d.ℓ2_coeff() * 2 * v.r;
    let beta2_coeff: i64 = u_c * v.r - v_c * u.r;
    let const_coeff: Rational64 = u.d.ℓ2_coeff() * 2 * v_c
        - v.d.ℓ2_coeff() * 2 * u_c;

    let radius2 = -const_coeff.checked_div(&beta2_coeff.into())?
        + (beta_coeff.checked_div(&beta2_coeff.into())? / 2).pow(2);

    match radius2.cmp(&Rational64::from_integer(0)) {
        Ordering::Greater => Some(radius2),
        _ => None,
    }
}
```

A.3.1 Left Pseudowalls Submodule

Listing A.9: `tilt_stability::left_pseudo_semistabilizers`

```rust
//! Module containing all logic related to computing the pseudo-semistabilizers for a
//! fixed Chern character v giving pseudo-walls to the left of the vertical line
//! β=μ(v).
//!
//! The main exposed functions are `find_all` and `find_all_unsafe` both
//! return the pseudo-semistabilizers described above.
//!  - `find_all` returns a vector of all pseudo-semistabilizers.
//!  - `find_all_unsafe` returns a lazy iterator, which takes ownership of the
//!  input ChernChar. This may be faster but that has not been observed.
//!
//!  The former is safer in principal, because by collecting all results on
//!  return, errors can be thrown if any of the computations fail.
//!  Since the latter is lazy, unexpected errors when iterating over the
//!  output, will cause a `panic`, and the error cannot be caught.
//!
//!  This *should* not happen, but the safe version is so fast anyway, there
//!  is little point to using `find_all_unsafe`.
use crate::chern_character::ChernChar;
use super::β_;
use crate::utils::modulo_inverse;
use num::rational::Rational64;

mod fixed_q_beta;

/// Returns i64 iterator which gives all b s.t.
///   b/n ∈ (0, ch, β (v) ) ∩ 1/n ℤ     where n = β's denominator
///
/// These correspond to all possible values ch, β (u) = b/n can take if u is a
/// pseudo-semistabilizer for v at some stability condition with this fixed β value.
fn considered_b_for_beta<const M: u32>(
    v: &ChernChar<M>,
    β: &Rational64,
) -> impl Iterator<Item = i64> {
    let c: i64 = v.c.into();
    let r: i64 = v.r;
    1..(β.denom() * c - β.numer() * r)
}

pub type ProblemType = u8;
```

```rust
pub const PROBLEM1: ProblemType = 1;
pub const PROBLEM2: ProblemType = 2;

/// data relating to point around which we are searching for walls
/// (currently only (β_, ϑ))
struct ProblemData<'a, const M: u32, const P: ProblemType> {
    beta: Rational64,
    a_inv: Result<i64, &'static str>,
    v: &'a ChernChar<M>,
}

impl<'a, const M: u32, const P: ProblemType> ProblemData<'a, M, P> {
    fn a(&self) -> &i64 {
        self.beta.numer()
    }
    fn n(&self) -> &i64 {
        self.beta.denom()
    }
    fn v(&self) -> &'a ChernChar<M> {
        self.v
    }
    fn beta(&self) -> &Rational64 {
        &self.beta
    }
}

fn problem_2_data<const M: u32>(v: &ChernChar<M>)
-> Result<ProblemData<M, PROBLEM2>, &'static str>{
    let beta: Rational64 = β_(v)
        // Error out now if β_ is not rational
        // (and hence infinite semistabilizers/walls)
        .ok_or("β_ is not rational: ∞ walls")?;
    let n = beta.denom();
    let a = beta.numer();
    // Should not panic since a and n should be coprime:
    let a_inv = modulo_inverse(a.rem_euclid(*n), *n);

    Ok(ProblemData::<M, PROBLEM2>{ beta, a_inv, v })
}


/// Given a ChernChar v,
/// returns an iterator over all possible pseudo-semistabilizers of v
/// giving pseudo-walls for of the vertical line β=μ(v)
/// if β_ is rational,
///
/// otherwise returns Error.
pub fn find_all<const M: u32>(
    v: &ChernChar<M>,
) -> Result<Vec<ChernChar<M>>, &'static str>
{
    let problem_data = problem_2_data(v)?;

    Ok(considered_b_for_beta(problem_data.v(), problem_data.beta())
        .map(|b|
            // fix ch₁ β  = b/n ℓ of pseudo-semistabilizer
            fixed_q_beta::ProblemData{b, beta_data: &problem_data}
            .find_all()
        ).try_collect::<Vec<_>>()?
        .concat()
    )
}
```

Fixed $\mathrm{ch}_1^\beta(u)$ Submodule

Listing A.10: `tilt_stability::left_pseudo_semistabilizers::fixed_q_beta`

```rust
//! Module containing all logic which works under the assumption that we are
//! searching for pseudo-walls for a given v, through a fixed β (=a/n) value,
//! and a fixed possible first twisted chern for u (q=b/n) inducing the wall.
//!
//! ```equation
//!        a                           b
//!    β = - (in smallest terms)   q = -
//!        n                           n
//! ```
use crate::chern_character::ChernChar;
use crate::chern_character::terms::{Chern0, Chern1};
use crate::tilt_stability::twisted;
use crate::utils::{least_greater_int, linear_seq};
use num::rational::Rational64;
use num::{CheckedDiv, Integer};
use std::cmp::Ordering;
use super::ProblemData as GeneralProblemData;
use super::{ProblemType, PROBLEM2};

mod fixed_r;

pub struct ProblemData<'a, const M: u32, const P: ProblemType> {
    pub b: i64,
    pub beta_data: &'a GeneralProblemData<'a, M, P>,
}

impl<'a, const M: u32, const P: ProblemType> ProblemData<'a, M, P> {
    pub fn q_val(&self) -> Rational64 {
        (
            self.b,
            *self.beta_data.n()
        ).into()
    }

    /// (inclusive) upper bound for the ranks of pseudo-semistabilizers u
    /// satisfying the assumptions of ContextData
    pub fn r_max(&self) -> Chern0 {
        if P == PROBLEM2 {
            let v = self.beta_data.v;
            let beta = self.beta_data.beta;
            let n = self.beta_data.n();
            let a = self.beta_data.a();
            let b = self.b;
            // TODO express this in terms of bgmlv discriminant
            let twisted_chern_1 = twisted::chern_1(v, beta);
            let q = self.q_val();
            let modulo = ((2*n.pow(2)).lcm(&(M as i64)) / (M as i64)).gcd(&(
                n * a * (2*n.pow(2)).lcm(&(M as i64))
                / (2*n.pow(2))
            ));
            let k_vq = (
                -a * b
                * (M as i64 / (2 * n.pow(2)).gcd(&(M as i64)))
            ).rem_euclid(modulo);
            // need least positive integer with specific mod `modulo` value:
            let k_vq = match k_vq {
                0 => modulo,
                _ => k_vq,
```

```
            };

            let bound_1 = q.pow(2)
                * (2 * n.pow(2)).lcm(&(M as i64)) / k_vq / 2;
            let bound_2 =  (twisted_chern_1 - q).pow(2)
                * (2 * n.pow(2)).lcm(&(M as i64)) / k_vq / 2
                + v.r;

            bound_1.min(bound_2).to_integer()
        } else {
            panic!("Not implemented");
        }
    }

    pub fn possible_r(&self)
    -> Result<impl Iterator<Item=(Cherno, Chern1)> + 'a, &'static str> {
        let n = self.beta_data.n();
        let b = self.b;
        let a = self.beta_data.a();
        let a_inv = self.beta_data.a_inv;
        let v = self.beta_data.v();
        // strict lower bound on r to guarantee μ(u) < μ(v):
        let r_lower_bound = Rational64::from_integer(b * v.r)
            .checked_div(&Rational64::from_integer(
                Into::<i64>::into(v.c) * n - a * v.r
            ));

        // non-strict integer lower bound on r to guarantee μ(u) < μ(v):
        let r_int_lower_bound: i64 = match r_lower_bound.map(least_greater_int) {
            Some(bound) => match bound.cmp(&0) {
                Ordering::Greater => bound,
                _ => return Err("Internal logical error: Negative lower bound on r"),
            },
            // Corresponds to case Δ(u)=0, so no circular-left-walls
            None => return Ok(
                (1..=0).step_by(1).zip(linear_seq(0.into(), 0.into()))
            ),
            // this should be emptyˆ
        };

        // smallest r s.t. a r ≡ -b (mod n) that's at least r_int_lower_bound
        let rstart: i64 = match n.cmp(&1) {
            Ordering::Greater => {
                r_int_lower_bound
                + (-r_int_lower_bound - a_inv? * b).rem_euclid(*n)
            }
            Ordering::Equal => r_int_lower_bound,
            Ordering::Less => return Err("n must be positive"),
        };

        //  c value corresponding to rstart (with the given q=b/n value)
        let cstart: Chern1 = match (b + rstart * a).is_multiple_of(n) {
            true => (b + rstart * a) / n,
            false => {
                return Err("Internal logical error: underlying modular arithmetic")
            }
        }.into();

        let rmax: Cherno = self.r_max();

        // will panic if n is too big:
        let r_vals = (rstart..=rmax)
            .step_by((*n).try_into().unwrap());
        // will panic if a is too big:
```

```
        let c_vals = linear_seq::<Chern1, Chern1>(cstart, (*a).into());

        Ok(r_vals.zip(c_vals))
    }

    pub fn find_all(&self) -> Result<Vec<ChernChar<M>>, &'static str> {
        Ok(self.possible_r()?
            .map(|(r,c)|
                // fix ch₀=r and ch₁=cℓ of pseudo-semistabilizer
                // ChernChars of pseudo-semistabilizers with fixed b, r, c
                fixed_r::ProblemData{r, c, fixed_q_beta_context: self}
                .find_all())
            .try_collect::<Vec<_>>()?
            .concat())
    }
}
```

Fixed $\mathrm{ch}_0(u)$ Submodule

Listing A.11: `tilt_stability::left_pseudo_semistabilizers::fixed_q_beta::fixed_r`

```
//! Module containing all logic which works under the assumption that we are
//! searching for pseudo-walls for a given v, through a fixed β (=a/n) value, fixed
//! possible first twisted chern for u (q=b/n), and a fixed rank for u, inducing the
//! wall.
//!
//! ```equation
//!        a                          b
//!    β = - (in smallest terms)   q = -
//!        n                          n
//! ```
use crate::chern_character::ChernChar;
use crate::chern_character::terms::{
    Chern0,
    Chern1,
    Chern2,
    least_greater_chern2,
    greatest_lesser_or_eq_chern2,
    inclusive_range
};
use super::ProblemData as GeneralProblemData;
use super::super::ProblemType;

mod bound_on_d;

/// Data relating made for semistabilizers that we are searching for
/// This consists of fixed values of ch₀(u), ch₁(u), on top of
/// assumptions represented by `fixed_q_beta::ContextData`
pub struct ProblemData<'a, const M: u32, const P: ProblemType> {
    pub r: Chern0,
    pub c: Chern1,
    pub fixed_q_beta_context: &'a GeneralProblemData<'a, M, P>,
}

impl<'a, const M: u32, const P: ProblemType> ProblemData<'a, M, P> {
    /// Finds all pseudo_semistabilizers giving 'left' pseudo-walls for v which
    /// satisfy the assumptions in `ContextData`
    pub fn find_all(&self)
    -> Result<Vec<ChernChar<M>>, &'static str> {
        Ok(self.possible_chern2()?
            .map(move |d| ChernChar::<M>::new( self.r, self.c, d ))
            .collect())
```

```
    }

    /// Returns possible ch₂ of semistabilizers u of v,
    /// where u satisfies the assumptions of ContextData
    pub fn possible_chern2(&self)
    -> Result<impl Iterator<Item = Chern2<M>>, &'static str> {
        let lowerbound = bound_on_d::lower::radius_condition(self);

        let upperbound = [
            bound_on_d::upper::bgmlv1(self),
            bound_on_d::upper::bgmlv2(self),
          ]
          .iter()
          .filter_map(|x| *x) // Remove None's
          .min()
          .ok_or("logical error, all upper bounds are None")?
          // This error should never happen^
        ;

        // Return Chern2 values dℓ²  such that d is between the bounds
        Ok(
            inclusive_range(
                least_greater_chern2(lowerbound),
                greatest_lesser_or_eq_chern2(upperbound)
            )
        )
    }
}
```

Listing A.12: `tilt_stability::left_pseudo_semistabilizers ::fixed_q_beta::fixed_r::bound_on_d`

```
//! Namespace for functions which give bounds for d where
//! u = (r, _ℓ , d·½ℓ²) is a semistabilizer for v at β with ch₁ β (u) = qℓ.
//! The conditions here match those assumed of the parent module.

// FUTURE: Revisit use of function generics here when it's possible to
// specialize over const parameter values
use super::ProblemData as FixedRProblemData;
use crate::tilt_stability::left_pseudo_semistabilizers::{
    PROBLEM1,
    PROBLEM2,
    ProblemType
};

pub mod lower {
    //! Lower bounds for d as per described in parent module
    use num::rational::Rational64;
    use crate::tilt_stability::twisted;
    use super::{PROBLEM1, PROBLEM2, ProblemType, FixedRProblemData};

    pub fn radius_condition<const M: u32, const P: ProblemType>(
        problem: &FixedRProblemData<'_, M, P>
    ) -> Rational64 {
        if P == PROBLEM2 {
            radius_condition_prob2(problem)
        } else if P == PROBLEM1 {
            let r = problem.r;
            let beta = problem.fixed_q_beta_context.beta_data.beta;
            let v = problem.fixed_q_beta_context.beta_data.v;
            radius_condition_prob2(problem) + twisted::chern_2(v, beta) + r / v.r
        } else {
            panic!("Not imlemented");
```

```rust
        }
    }
    pub fn radius_condition_prob2<const M: u32, const P: ProblemType>(
        problem: &FixedRProblemData<'_, M, P>
    ) -> Rational64 {
        let beta = problem.fixed_q_beta_context.beta_data.beta;
        let q = problem.fixed_q_beta_context.q_val();
        let r = problem.r;
        beta.pow(2) * r / 2 + beta * q
    }
}

pub mod upper {
    //! Upper bounds for d as per described in parent module
    use super::lower;
    use crate::tilt_stability::twisted;
    use num::rational::Rational64;
    use super::{PROBLEM1, PROBLEM2, ProblemType, FixedRProblemData};

    /// Returns upper bound on d induced
    /// by the Bogomolov inequality 0 ≤ Δ(u)
    /// where u = (r, _ℓ , d·½ℓ²) with ch₁ β (u) = qℓ
    ///
    /// Returns value as rational if it indeed induces an upper bound,
    /// otherwise None (if say, it induces a lower bound instead)
    pub fn bgmlv1<const M: u32, const P: ProblemType>(
        problem: &FixedRProblemData<'_, M, P>
    ) -> Option<Rational64> {
        // The bound is the same for both probem types
        let q = problem.fixed_q_beta_context.q_val();
        let r = problem.r;
        if r <= 0 {
            return None;
        }

        Some(lower::radius_condition_prob2(problem) + q.pow(2) / (2 * r))
    }

    /// Returns upper bound on d induced
    /// by the Bogomolov inequality 0 ≤ Δ(v-u)
    /// where u = (r, _ℓ , d·½ℓ²) with ch₁ β (u) = qℓ
    ///
    /// Returns value as rational if it indeed induces an upper bound,
    /// otherwise None (if say, it induces a lower bound instead)
    pub fn bgmlv2<const M: u32, const P: ProblemType>(
        problem: &FixedRProblemData<'_, M, P>
    ) -> Option<Rational64> {
        let q = problem.fixed_q_beta_context.q_val();
        let r = problem.r;
        let v = problem.fixed_q_beta_context.beta_data.v;
        let beta = problem.fixed_q_beta_context.beta_data.beta();
        if r <= v.r {
            return None;
        }
        let problem2_bound = lower::radius_condition_prob2(problem)
            + (twisted::chern_1(&v, *beta) - q).pow(2) / (2 * (r - v.r));

        if P == PROBLEM2 {
            Some(problem2_bound)
        } else if P == PROBLEM1{
            Some(
                problem2_bound
                + twisted::chern_2(v, *beta)
            )
```

```
        } else {
            panic!("Not implemented");
        }
    }

    /// Returns upper bound on d induced
    /// by the Bogomolov inequality Δ(u) + Δ(v-u) ≤ Δ(v)
    /// where u = (r, _ℓ , d·½ℓ²) with ch₁ β (u) = qℓ
    ///
    /// Returns value as rational if it indeed induces an upper bound,
    /// otherwise None (if say, it induces a lower bound instead)
    pub fn bgmlv3_upperbound_on_d<const M: u32, const P: ProblemType>(
        problem: &FixedRProblemData<'_, M, P>
    ) -> Option<Rational64> {
        if P == PROBLEM2 {
            let q = problem.fixed_q_beta_context.q_val();
            let r = problem.r;
            let v = problem.fixed_q_beta_context.beta_data.v;
            let beta = problem.fixed_q_beta_context.beta_data.beta();
            if 2 * r >= v.r {
                return None;
            }

            Some(
                lower::radius_condition(problem)
                    + (twisted::chern_1(&v, *beta) - q) * q / (v.r - 2 * r),
            )
        } else {
            panic!("Not implemented");
        }
    }
}
```

Appendix B

Pseudowalls Computer Algebra Library

This appendix contains the source code of the `pseudowalls` Python library, which is used throughout this thesis to make computations relating to stability conditions. The code is hosted on GitLab at https://gitlab.com/pseudowalls/pseudowalls, the revision shown here is commit 17e4163. The reader may find it useful to view the examples in the notebooks linked in the README for the repository to learn its usage.

B.1 Main Module

Initialisation file for the library as a whole, exposing the key items of the submodules.

Listing B.1: `pseudowalls.__init__`

```
from .chern_character import Chern_Char, Twisted_Chern_Char, exponential_chern,
    generic_chern_char
from .integral_chern import Integral_Chern, generic_integral_chern
from . import stability

__all__ = [
    "Chern_Char",
    "Twisted_Chern_Char",
    "exponential_chern",
    "generic_chern_char",
    "Integral_Chern",
    "generic_integral_chern",
    "stability"
]

```

B.2 Chern Character Modules

Listing B.2: `pseudowalls.integral_chern`

```
from sympy.core.add import Add as SympyAdd
from .chern_character import Chern_Char
from .utils.symb_util import latex, symbols, factorial


class Integral_Chern(Chern_Char):

    def __init__(self, *ch):
        adjusted = (chi/factorial(i) for i, chi in enumerate(ch))
```

```
        Chern_Char.__init__(self, *adjusted)

    def _parametrization(self):
        def brackets_needed(exponent, expression):
            if hasattr(expression, "_sympy_"):
                # Tweak for Sage expressions
                expression = expression._sympy_()
            return (
                exponent > 0
                and hasattr(expression, "func")
                and expression.func is SympyAdd
            )

        return (
          r"\begin{array}{l} "
          + r" \\ ".join(
            # ch_i:
            r"\mathrm{ch}_{"+latex(i)+r"} ="
            + (r"\left(" if brackets_needed(i, vi) else "")
            + latex(vi*factorial(i))
            + (r"\right)" if brackets_needed(i, vi) else "")
            # add l î/i! when i>0:
            + (
                r"\frac{"
                + f"\\ell { {{i}}}"
                + r"}{" + str(i) + "!}"
                if i > 0 else ""
              )
            for i, vi in enumerate(self.ch)
          )
          + r" \end{array}"
        )

##############################
#  Alternative Constructors #
##############################


def generic_integral_chern(g, name="a"):
    ch = symbols(
        " ".join(
            name+f"_{i}"
            for i in range(g+1)
        ),
        integer=True
    )

    return Integral_Chern(*ch)

```

Listing B.3: `pseudowalls.chern_character`

```
from .utils.symb_util import symbols, latex, factorial

from sympy.core.add import Add as SympyAdd

############################
#  Chern Character type  #
############################

class Chern_Char:
    """
    Class for representing Chern Characters
    of a variety X of Picard Rank 1
```

```
    """
    # This objects of this class definitely must be
    # considered immutable from the outside
    # TODO find out if this can be enforced

    def __init__(self, *ch):
        r"""
        Given a generator $L \in NS(X)$ with $\dim(X)=g$,
        The initialiser takes $g+1$ elements, where the ith
        element $ch_{i-1}$, the coefficient of $\[L\] { i-1}$
        in $\mathrm{chern}_{i-1} = ch_i \[L\] { i-1}$
        """
        self.ch = ch

    def _title(self):
        return r"Chern Character:"

    def _parametrization(self):
        def brackets_needed(exponent, expression):
            if hasattr(expression, "_sympy_"):
                # Tweak for Sage expressions
                expression = expression._sympy_()
            return (
                exponent > 0
                and hasattr(expression, "func")
                and expression.func is SympyAdd
            )

        return (
          r"\begin{array}{l} "
          + r" \\ ".join(
            # ch_i:
            r"\mathrm{ch}_{"+str(i)+r"} ="
            + (r"\left(" if brackets_needed(i, vi) else "")
            + latex(vi)
            + (r"\right)" if brackets_needed(i, vi) else "")
            + (r"\ell { "+str(i)+"}" if i > 0 else "")
            for i, vi in enumerate(self.ch)
          )
          + r" \end{array}"
        )

    def _repr_latex_(self):
        return (
          self._title()
          + self._parametrization()
        )

    def _latex_(self):
        return (
          r"\text{"
          + self._title()
          + r"} \\ "
          + self._parametrization()
        )

    def __mul__(self, other):
        assert len(self.ch) == len(other.ch), f"""
        Chern characters can only be multiplied if they have
        the same length:
            LHS length = {len(self.ch)}
            RHS length = {len(other.ch)}
        """

```

```
        return Chern_Char(*(
            # ch_i of the product:
            sum(
                self.ch[j]*other.ch[i-j]
                for j in range(i+1)
            )
            for i in range(len(self.ch))
        ))

    def __neg__(self):
        return Chern_Char(*( - ch for ch in self.ch))

    def __add__(self, other):
        assert len(self.ch) == len(other.ch), f"""
        Chern characters can only be added if they have
        the same length:
            LHS length = {len(self.ch)}
            RHS length = {len(other.ch)}
        """

        return Chern_Char(*(
            # ch_i of the sum:
            a + b
            for a, b in zip(self.ch, other.ch)
        ))

    def __sub__(self, other):
        assert len(self.ch) == len(other.ch), f"""
        Chern characters can only be subtracted if they have
        the same length:
            LHS length = {len(self.ch)}
            RHS length = {len(other.ch)}
        """

        return Chern_Char(*(
            # ch_i of the sum:
            a - b
            for a, b in zip(self.ch, other.ch)
        ))

    def twist(self, beta):
        g = len(self.ch) - 1
        twisted = exponential_chern(-beta, g) * self

        result = Twisted_Chern_Char(beta, *twisted.ch)

        # twist inherits the chosen "parametrisation"
        # i.e. the way it's written out
        # Behold the world's most despicable hack:
        result._parametrization = lambda: self.__class__._parametrization(result)

        return result

    def Q_tilt(self):
        v0, v1, v2, *_ = self.ch
        return v1**2 - 2*v0*v2


###################################
#  Twisted Chern Character type  #
###################################

class Twisted_Chern_Char(Chern_Char):
    def __init__(self, beta, *ch):
```

```
        super().__init__(*ch)
        self.beta = beta

    def _title(self):
        return (
          r"Twisted Chern Character for "
          + r"$\beta={"
          + latex(self.beta)
          + r"}$"
        )

    def twist(self, beta):
        g = len(self.ch) - 1
        twisted = exponential_chern(self.beta-beta, g) * self

        return Twisted_Chern_Char(beta, *twisted.ch)


##############################
#  Alternative Constructors #
##############################

def exponential_chern(x, g):
    return Chern_Char(*(
        x**i / factorial(i)
        for i in range(g+1)
    ))


def generic_chern_char(g, name="ch"):
    ch = symbols(
        " ".join(
            name+f"_{i}"
            for i in range(g+1)
        ),
        rational=True
    )

    return Chern_Char(*ch)

```

Listing B.4: `pseudowalls.integral_chern`

```
from sympy.core.add import Add as SympyAdd
from .chern_character import Chern_Char
from .utils.symb_util import latex, symbols, factorial


class Integral_Chern(Chern_Char):

    def __init__(self, *ch):
        adjusted = (chi/factorial(i) for i, chi in enumerate(ch))
        Chern_Char.__init__(self, *adjusted)

    def _parametrization(self):
        def brackets_needed(exponent, expression):
            if hasattr(expression, "_sympy_"):
                # Tweak for Sage expressions
                expression = expression._sympy_()
            return (
                exponent > 0
                and hasattr(expression, "func")
                and expression.func is SympyAdd
            )
```

```
        return (
          r"\begin{array}{l} "
          + r" \\ ".join(
            # ch_i:
            r"\mathrm{ch}_{"+latex(i)+r"} ="
            + (r"\left(" if brackets_needed(i, vi) else "")
            + latex(vi*factorial(i))
            + (r"\right)" if brackets_needed(i, vi) else "")
            # add l î/i! when i>0:
            + (
                r"\frac{"
                + f"\\ell {{i}}}"
                + r"}{" + str(i) + "!}"
                if i > 0 else ""
              )
            for i, vi in enumerate(self.ch)
          )
          + r" \end{array}"
        )

##############################
#  Alternative Constructors #
##############################


def generic_integral_chern(g, name="a"):
    ch = symbols(
        " ".join(
            name+f"_{i}"
            for i in range(g+1)
        ),
        integer=True
    )

    return Integral_Chern(*ch)

```

B.3 Stability Module

This module deals with the parametrisation of the central charges of stability conditions in the slices we consider.

Listing B.5: `pseudowalls.stability`

```
from .utils.stability_util import stab_from_central_charge, stab_from_rank_degree
from .chern_character import Chern_Char
from .utils.symb_util import I, Symbol, Rational


@stab_from_rank_degree
class Mumford:

    def rank(self, v):
        return v.ch[0]

    def degree(self, v):
        return v.ch[1]


@stab_from_central_charge
class Tilt:

```

```
    def __init__(
      self,
      alpha=Symbol("alpha", real=True, positive=True),
      beta=Symbol("beta", real=True)
    ):
        self.alpha = alpha
        self.beta = beta

    def central_charge(self, v):
        return - v.twist(self.beta + I*self.alpha).ch[2]


@stab_from_rank_degree
class SecondTilt:

    def __init__(
      self,
      alpha=Symbol("alpha", real=True, positive=True),
      beta=Symbol("beta", real=True),
      s=Symbol("s", real=True)
    ):
        self.alpha = alpha
        self.beta = beta
        self.s = s

    def rank(self, v):
        return Tilt(self.alpha, self.beta).degree(v)

    def degree(self, v):
        _, ch1, _, ch3, *_ = v.twist(self.beta).ch
        SIXTH = Rational(1, 6)

        return ch3 - (self.s + SIXTH) * self.alpha**2 * ch1

# Alternative aliases for these classes

Lambda = SecondTilt
nu = Tilt
```

B.4 Utility Modules

These modules are less mathematical in nature, but are concerned with metaprogramming and monkey-patching to interface with the symbolic algebra functionality in both SageMath and SymPy.

Listing B.6: `pseudowalls.utils.__init__`

```
pass
```

Listing B.7: `pseudowalls.utils.stability_util`

```
from sympy import I
try:
    from sage.symbolic.constants import I
    def real_part(s):
        return s.real_part()
    def imag_part(s):
        return s.imag_part()
    # SAGE_INSTALLED = True
except:
```

```
    def real_part(s):
        return s.as_real_imag()[0]
    def imag_part(s):
        return s.as_real_imag()[1]
    # SAGE_INSTALLED = False

def stab_from_rank_degree(cls):
    class stab_class(cls):
        def central_charge(self, v):
            return (
            - self.degree(v)
            + I*self.rank(v)
            )

        def slope(self, v):
            return (
            self.degree(v)
            / self.rank(v)
            )

        def wall_eqn(self, v, w):
            return (
                self.degree(v) * self.rank(w)
                - self.degree(w) * self.rank(v)
            )

    return stab_class

def stab_from_central_charge(cls):
    class stab_class(cls):
        def rank(self, v):
            return imag_part(self.central_charge(v))

        def degree(self, v):
            return - real_part(self.central_charge(v))

        def slope(self, v):
            return self.degree(v)/self.rank(v)

        def wall_eqn(self, v, w):
            return (
                self.degree(v) * self.rank(w)
                - self.degree(w) * self.rank(v)
            )

    return stab_class

```

Listing B.8: `pseudowalls.utils.symb_util`

```
try:
    from sage.all import latex
    from sage.all import var as _sage_var
    from sage.symbolic.constants import I

    def symbols(*args, **kwargs):
        sage_kwargs = {}
        if "real" in kwargs and kwargs["real"] \
           or "rational" in kwargs and kwargs["rational"]:
            sage_kwargs["domain"] = "real"
        if "positive" in kwargs and kwargs["positive"]:
            sage_kwargs["domain"] = "positive"
        if "integer" in kwargs and kwargs["integer"]:
            sage_kwargs["domain"] = "integer"
```

```
15
16          return _sage_var(*args, **sage_kwargs)
17
18      def Symbol(*args, **kwargs):
19          sage_kwargs = {}
20          if "real" in kwargs and kwargs["real"] \
21             or "rational" in kwargs and kwargs["rational"]:
22              sage_kwargs["domain"] = "real"
23          if "positive" in kwargs and kwargs["positive"]:
24              sage_kwargs["domain"] = "positive"
25          if "integer" in kwargs and kwargs["integer"]:
26              sage_kwargs["domain"] = "integer"
27
28          return _sage_var(*args, **sage_kwargs)
29
30      SAGE_INSTALLED = True
31      from sympy import factorial, Rational # Maybe find SageMath equivs
32  except:
33      from sympy import Symbol, symbols, latex, factorial, Rational, I
34      SAGE_INSTALLED = False
```

Appendix C

Jupyter Notebooks

C.1 Stability Condition Parametrisations

[1]:
```
from pseudowalls import *
from pseudowalls.utils.stability_util import stab_from_central_charge
import copy
%display latex
```

[2]:
```
var("alpha beta", domain="real")
```

[2]: (α, β)

[3]:
```
v = generic_chern_char(2, "v")
v
```

[3]: Chern Character:
$\mathrm{ch}_0 = v_0$
$\mathrm{ch}_1 = v_1\ell^1$
$\mathrm{ch}_2 = v_2\ell^2$

C.1.1 Bridgeland Parametrisation

The central charge for Parametrisation "A" of stability conditions is characterised by assigning -1 to $(0, 0, 1)$, and a numerical condition:

$$\mathcal{Z}(0, \ell, 0)^2 + 2\mathcal{Z}(1, 0, 0)$$

[4]:
```
ts = stability.Tilt()
```

Function to "normalise" a stability condition object, corresponding to applying a $\widetilde{\mathbb{C}^*}$ element so that skyscrapers have phase 1 and $\mathcal{Z} = -1$:

[5]:
```
def normalize_skyscraper_image(stab):
    stab = copy.deepcopy(stab) # remove risk of unintended mutation

    a = -stab.central_charge(Chern_Char(0,0,1))
    assert a!=0, "Image of skyscraper must be non-zero"

    @stab_from_central_charge
    class _normalised_stab:
        def __init__(self):
            pass
        def central_charge(self,v):
            return stab.central_charge(v)/a

    return a, _normalised_stab()
```

Given a "normalised" stability condition, check whether it corresponds to one given by parametrisation A, and if so, provide the parameters (assuming geometric, up to 2 shifts).

[6]:
```
def get_bridgeland_parameters(stab):
    skyscraper = Chern_Char(0,0,1)
```

```
    just_ch1 = Chern_Char(0,1,0)
    struct_sheaf = Chern_Char(1,0,0)
    assert stab.central_charge(skyscraper) == -1, "not normalized around image of
↪skyscraper"

    just_ch1_image = stab.central_charge(just_ch1)
    alpha = just_ch1_image.imag()
    beta = just_ch1_image.real()

    assert 0 == (stab.central_charge(struct_sheaf)*2 + just_ch1_image^2).expand(),
↪"not a stab given by B"
    return alpha, beta
```

Test:

[7]:
```
get_bridgeland_parameters(stability.Tilt())
```

[7]: (α, β)

[8]:
```
get_bridgeland_parameters(stability.Tilt(2.0, 1.0))
```

[8]: $(2.00000000000000, 1.00000000000000)$

Function to verify whether stability is given by parametrisation A (assuming geometric, up to 2 shifts), without returning parameters.

[9]:
```
def bridg_param_condition(stab):
    struct_sheaf = Chern_Char(1,0,0)
    just_ch1 = Chern_Char(0,1,0)
    return stab.central_charge(struct_sheaf)*2 + stab.central_charge(just_ch1)^2 ==0
```

[10]:
```
bridg_param_condition(ts).expand()
```

[10]: $0 = 0$

Actions on "normalised" stabilities

Function to "skew" a central charge of a stability condition, corresponding to $\text{skew}_x \cdot \sigma$ in the notation of the main document. Where $x =$`skewing` and $\sigma =$`stab` are the function arguments.

[11]:
```
def skew_stab(skewing, stab):
    stab = copy.deepcopy(stab) # remove risk of unintended mutation

    a = -stab.central_charge(Chern_Char(0,0,1))
    assert a!=0, "Image of skyscraper must be non-zero"

    @stab_from_central_charge
    class _skewed_stab:
        def __init__(self):
            pass
        def central_charge(self,v):
            z = stab.central_charge(v)
            re = z.real()
            im = z.imag()
            return re + skewing*im + I*im

    return _skewed_stab()
```

Tests:

[12]:
```
(
    skew_stab(0, ts).central_charge(v)
    - ts.central_charge(v)
).expand()
```

[12]: 0

[13]:
```
skew_stab(1, ts).central_charge(v)
```

[13]:

$\frac{1}{2}\,\alpha^2 v_0 - (i+1)\,\alpha\beta v_0 - \frac{1}{2}\,\beta^2 v_0 + (i+1)\,\alpha v_1 + \beta v_1 - v_2$

Function to "squash" a central charge of a stability condition, corresponding to $\mathrm{stretch}_y \cdot \sigma$ in the notation of the main document. Where $y =$`squashing` and $\sigma =$`stab` are the function arguments. I apologise for the inconsistency in naming.

[14]:
```
def squash_stab(squashing, stab):
    stab = copy.deepcopy(stab) # remove risk of unintended mutation

    a = -stab.central_charge(Chern_Char(0,0,1))
    assert a!=0, "Image of skyscraper must be non-zero"

    @stab_from_central_charge
    class _squashed_stab:
        def __init__(self):
            pass
        def central_charge(self,v):
            z = stab.central_charge(v)
            re = z.real()
            im = z.imag()
            return re + I*im*squashing

    return _squashed_stab()
```

Test:

[15]:
```
(
    squash_stab(1, ts).central_charge(v)
    - ts.central_charge(v)
).expand()
```

[15]: 0

C.1.2 Consider Dell parametrisation

Now consider parametrisation "D", for which we define a stability condition:

[16]:
```
var("b h alpha beta", domain="real")

@stab_from_central_charge
class dell_stab:
    def __init__(self):
        pass
    def central_charge(self,v):
        ch0, ch1, ch2 = v.ch
        return (alpha - I*beta)*h^2 * ch0 + (b + I*h) * ch1 - ch2
```

[17]:
```
dell_stab().central_charge(v)
```

[17]: $(\alpha - i\,\beta)h^2 v_0 + (b + i\,h)v_1 - v_2$

skew and stretch Dell central charge to Bridgeland form

Condition for an arbitrary skewing+stretching of Dell central charge to be of parametrisation A:

[18]:
```
var("x y", domain="real")

(xy_problem :=
    bridg_param_condition(
        squash_stab( y,
            skew_stab( x,
                dell_stab()
            )
        )
    )
)
```

[18]: $-2\,\beta h^2 x - 2i\,\beta h^2 y + 2\,\alpha h^2 + (hx + i\,hy + b)^2 = 0$

Quantity which must be 0 for the skewing+stretching to be of form from parametrisation A:

[19]:
```
xy_problem.lhs().imag().factor()
```

[19]: $-2\,(\beta h - hx - b)hy$

Solve for x:

[20]:
```
[x_val] = ( xy_problem
    .lhs()
    .imag()
    .factor()
    # h>0 by definition of param
    # y>0 by definition of squashing (to keep positive determinant)
    ._div_(-2*h*y)
    .solve(x)
)
x_val.expand()
```

[20]: $x = \beta - \frac{b}{h}$

Substitute x value back into problem and solve for y:

[21]:
```
[y2_sol] = ( xy_problem
    .lhs()
    .real()
    .subs(x_val)
    .expand()
    .solve(y^2)
)
y2_sol.expand()
```

[21]: $y^2 = -\beta^2 + 2\,\alpha + \frac{2\,b\beta}{h}$

Retrieve converting matrix

Now that we know how must skewing and stretching is needed to convert parametrisation D to A, determine the element of $\widetilde{\mathrm{GL}}^+(2,\mathbb{R})$ (that preserves phase 1) that does the conversion:

[22]:
```
conversion = (
    matrix([
        [1, 0],
        [0, y]
    ])
    * matrix([
        [1, x],
        [0, 1]
    ])
)
conversion
```

[22]: $\begin{pmatrix} 1 & x \\ 0 & y \end{pmatrix}$

[23]:
```
y_val = y==sqrt(y2_sol.rhs().expand())
```

[24]:
```
(conversion_D_to_B :=
conversion.subs(x_val.expand()).subs(y_val).inverse()
)
```

[24]: $\begin{pmatrix} 1 & -\frac{\beta-\frac{b}{h}}{\sqrt{-\beta^2+2\,\alpha+\frac{2\,b\beta}{h}}} \\ 0 & \frac{1}{\sqrt{-\beta^2+2\,\alpha+\frac{2\,b\beta}{h}}} \end{pmatrix}$

Finally retrieve the "A" parameters for this skewing and squashing of the "D" parametrisation:

[25]:
```
retrieved_parameters = squash_stab( y_val.rhs(),
        skew_stab( x_val.rhs(),
            dell_stab()
        )
```

```
    ).central_charge(Chern_Char(0,1,0))
retrieved_parameters
```

[25]:
$$\beta h + i\sqrt{-\beta^2 + 2\,\alpha + \frac{2\,b\beta}{h}}h$$

C.2 Characteristic Curves

C.2.1 Utilities

[1]:
```
# Requires extra package:
#! sage -pip install "pseudowalls==0.0.3" --extra-index-url https://gitlab.com/api/
 ↪v4/projects/43962374/packages/pypi/simple
%display latex

from pseudowalls import *

Δ = lambda v: v.Q_tilt()
mu = stability.Mumford().slope
ts = stability.Tilt

var("beta", domain="real")

def beta_minus(v):
    beta = stability.Tilt().beta
    solutions = solve(
        stability.Tilt(alpha=0).degree(v)==0,
        beta)
    return min(map(lambda s: s.rhs(), solutions))

class Object(object):
    pass
```

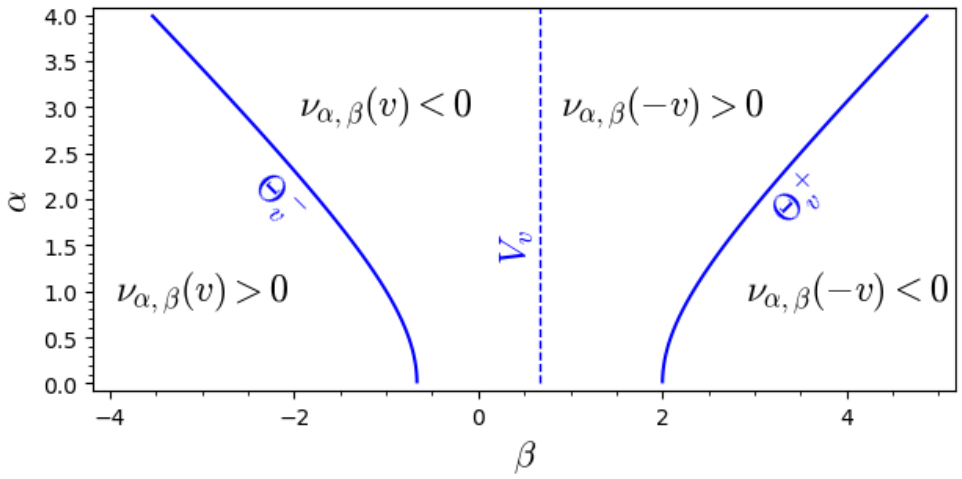
C.2.2 Characteristic Curves Plots

[2]:
```
def charact_curves(v):
    alpha = stability.Tilt().alpha
    beta = stability.Tilt().beta
    coords_range = (beta, -4, 5), (alpha, 0, 4)
    text_args = {"fontsize":"xx-large", "clip":True}
    black_text_args = {"rgbcolor": "black", **text_args}
    p = (
        implicit_plot(stability.Tilt().degree(v), *coords_range )
        + line([(mu(v),0),(mu(v),5)], linestyle = "dashed")
        + text(r"$\Theta_v^+$",[3.5, 2], rotation=45, **text_args)
        + text(r"$V_v$", [0.43, 1.5], rotation=90, **text_args)
        + text(r"$\Theta_v^-$", [-2.2, 2], rotation=-45, **text_args)
        + text(r"$\nu_{\alpha, \beta}(v)>0$", [-3, 1], **black_text_args)
        + text(r"$\nu_{\alpha, \beta}(v)<0$", [-1, 3], **black_text_args)
        + text(r"$\nu_{\alpha, \beta}(-v)>0$", [2, 3], **black_text_args)
        + text(r"$\nu_{\alpha, \beta}(-v)<0$", [4, 1], **black_text_args)
    )
    p.xmax(5)
    p.xmin(-4)
    p.ymax(4)
    p.axes_labels([r"$\beta$", r"$\alpha$"])
    return p

v1 = Chern_Char(3, 2, -2)
v2 = Chern_Char(3, 2, 2/3)
```

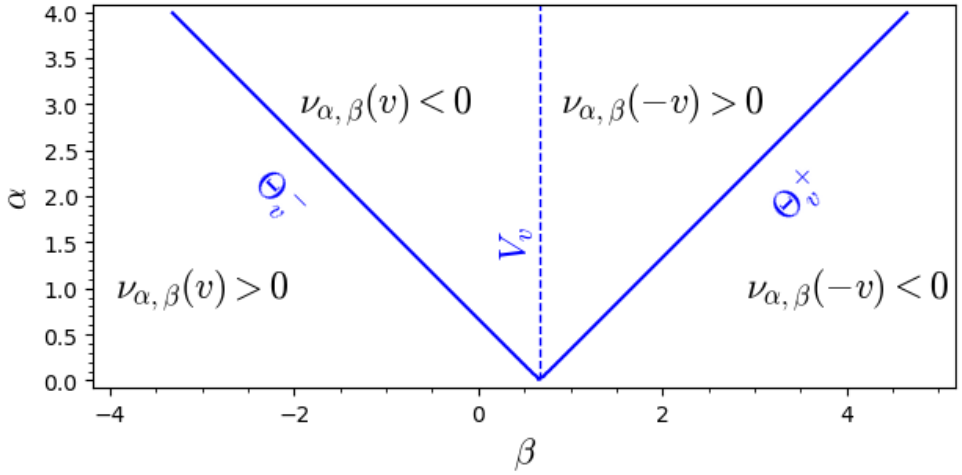
[3]:
```
typical_characteristic_curves = charact_curves(v1)
typical_characteristic_curves
```

[3]:



[4]:
```
degenerate_characteristic_curves = charact_curves(v2)
degenerate_characteristic_curves
```

[4]:



Characteristic Curves for Pseudo-semistabilizers

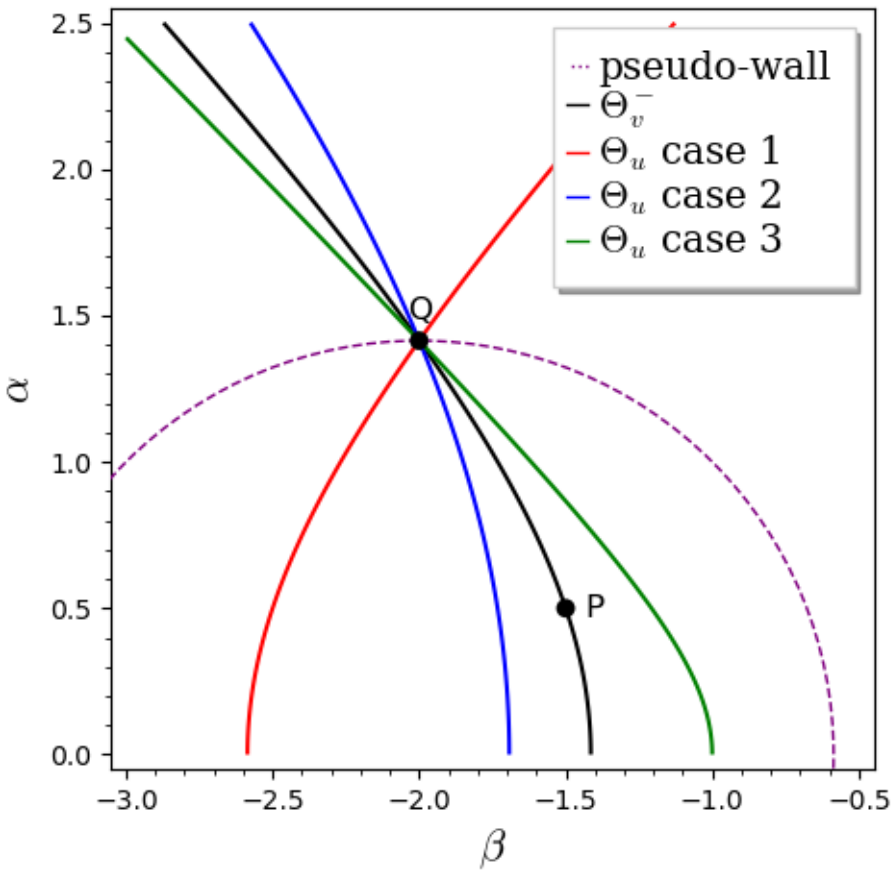
[5]:
```
def hyperbola_intersection_plot():
    var("alpha beta", domain="real")
    coords_range = (beta, -3, -1/2), (alpha, 0, 2.5)
    delta1 = -sqrt(2)+1/100
    delta2 = 1/2
    pbeta=-1.5
    text_args = {"fontsize":"large", "clip":True}
    black_text_args = {"rgbcolor":"black", **text_args}
    p = (
        implicit_plot( beta^2 - alpha^2 == 2,
            *coords_range , rgbcolor = "black", legend_label=r"a")
        + implicit_plot( (beta+4)^2 - (alpha)^2 == 2,
            *coords_range , rgbcolor = "red")
        + implicit_plot( (beta+delta1)^2 - alpha^2 == (delta1-2)^2-2,
            *coords_range , rgbcolor = "blue")
```

```
        + implicit_plot( (beta+delta2)^2 - alpha^2 == (delta2-2)^2-2,
            *coords_range , rgbcolor = "green")
        + point([-2, sqrt(2)], size=50, rgbcolor="black", zorder=50)
        + text("Q",[-2, sqrt(2)+0.1], **black_text_args)
        + point([pbeta, sqrt(pbeta^2-2)], size=50, rgbcolor="black", zorder=50)
        + text("P",[pbeta+0.1, sqrt(pbeta^2-2)], **black_text_args)
        + circle((-2,0),sqrt(2), linestyle="dashed", rgbcolor="purple")
        # dummy lines to add legends (circumvent bug in implicit_plot)
        + line([(2,0),(2,0)] , rgbcolor = "purple", linestyle="dotted",
            legend_label=r"pseudo-wall")
        + line([(2,0),(2,0)] , rgbcolor = "black",
            legend_label=r"$\Theta_v^-$")
        + line([(2,0),(2,0)] , rgbcolor = "red", legend_label=r"$\Theta_u$ case 1")
        + line([(2,0),(2,0)] , rgbcolor = "blue", legend_label=r"$\Theta_u$ case 2")
        + line([(2,0),(2,0)] , rgbcolor = "green", legend_label=r"$\Theta_u$ case
↪3")
    )
    p.set_legend_options(loc="upper right", font_size="x-large",
        font_family="serif")
    p.xmax(coords_range[0][2])
    p.xmin(coords_range[0][1])
    p.ymax(coords_range[1][2])
    p.ymin(coords_range[1][1])
    p.axes_labels([r"$\beta$", r"$\alpha$"])
    return p
```

Considering the 3 different ways that Θ_u could intersect Θ_v^-

[6]:
```
hyperbola_intersection_plot()
```

[6]:

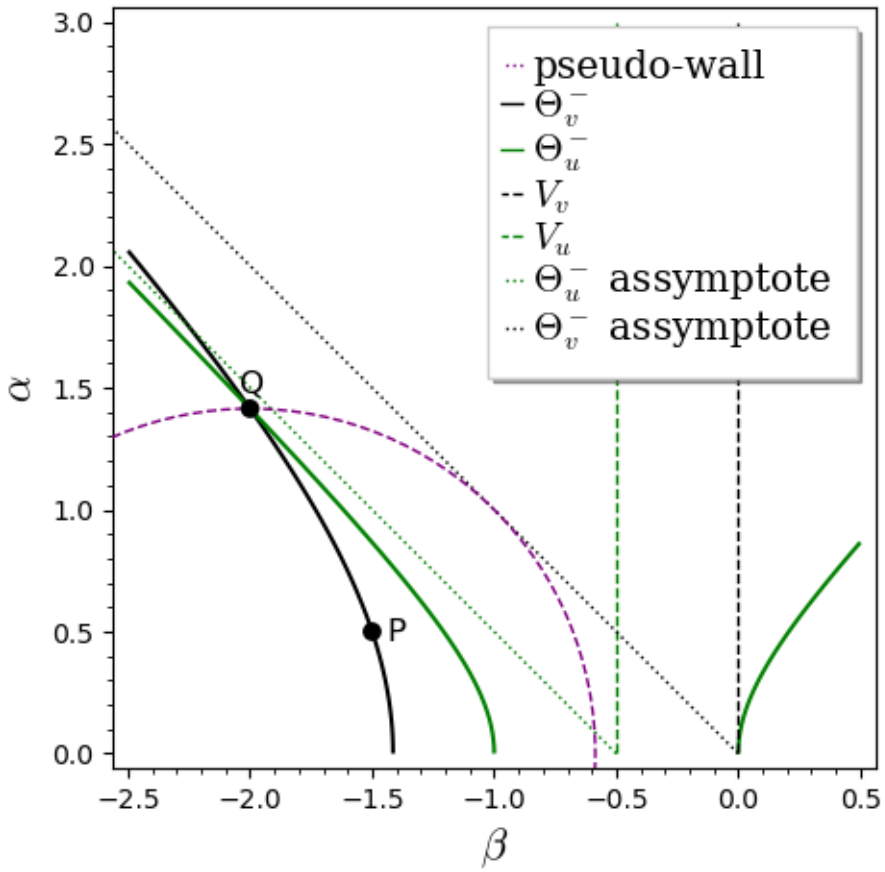


[7]:
```
def correct_hyperbola_intersection_plot():
    var("alpha beta", domain="real")
    coords_range = (beta, -2.5, 0.5), (alpha, 0, 3)
    delta2 = 1/2
    pbeta=-1.5
    text_args = {"fontsize":"large", "clip":True}
    black_text_args = {"rgbcolor":"black", **text_args}
    p = (
        implicit_plot( beta^2 - alpha^2 == 2,
            *coords_range , rgbcolor = "black", legend_label=r"a")
        + implicit_plot((beta+delta2)^2 - alpha^2 == (delta2-2)^2-2,
            *coords_range , rgbcolor = "green")
        + point([-2, sqrt(2)], size=50, rgbcolor="black", zorder=50)
        + text("Q",[-2, sqrt(2)+0.1], **black_text_args)
        + point([pbeta, sqrt(pbeta^2-2)], size=50, rgbcolor="black", zorder=50)
        + text("P",[pbeta+0.1, sqrt(pbeta^2-2)], **black_text_args)
        + circle((-2,0),sqrt(2), linestyle="dashed", rgbcolor="purple")
        # dummy lines to add legends (circumvent bug in implicit_plot)
        + line([(2,0),(2,0)] , rgbcolor = "purple", linestyle="dotted",
            legend_label=r"pseudo-wall")
        + line([(2,0),(2,0)] , rgbcolor = "black",
            legend_label=r"$\Theta_v^-$")
        + line([(2,0),(2,0)] , rgbcolor = "green",
            legend_label=r"$\Theta_u^-$")
        # vertical characteristic lines
        + line([(0,0),(0,coords_range[1][2])],
            rgbcolor="black", linestyle="dashed",
            legend_label=r"$V_v$")
        + line([(-delta2,0),(-delta2,coords_range[1][2])],
            rgbcolor="green", linestyle="dashed",
            legend_label=r"$V_u$")
        + line([(-delta2,0),(-delta2-coords_range[1][2],coords_range[1][2])],
            rgbcolor="green", linestyle="dotted",
            legend_label=r"$\Theta_u^-$ assymptote")
        + line([(0,0),(-coords_range[1][2],coords_range[1][2])],
            rgbcolor="black", linestyle="dotted",
            legend_label=r"$\Theta_v^-$ assymptote")
    )
    p.set_legend_options(loc="upper right", font_size="x-large",
        font_family="serif")
    p.xmax(coords_range[0][2])
    p.xmin(coords_range[0][1])
    p.ymax(coords_range[1][2])
    p.ymin(coords_range[1][1])
    p.axes_labels([r"$\beta$", r"$\alpha$"])
    return p
```

Full picture of characteristic curves, assymptotes, pseudo-walls for the scenario described in main problem (object u destabilizing v going down Θ_v^- before reaching some point P)

[8]:
```
correct_hyperbola_intersection_plot()
```

[8]:



C.3 Rank Zero Case

[1]:
```
from pseudowalls import *
%display latex

alpha = stability.Tilt().alpha
beta = stability.Tilt().beta
```

[2]:
```
var("C D r c d", domain="real")
```

[2]: (C, D, r, c, d)

[3]:
```
v = Chern_Char(0,1,0)
v
```

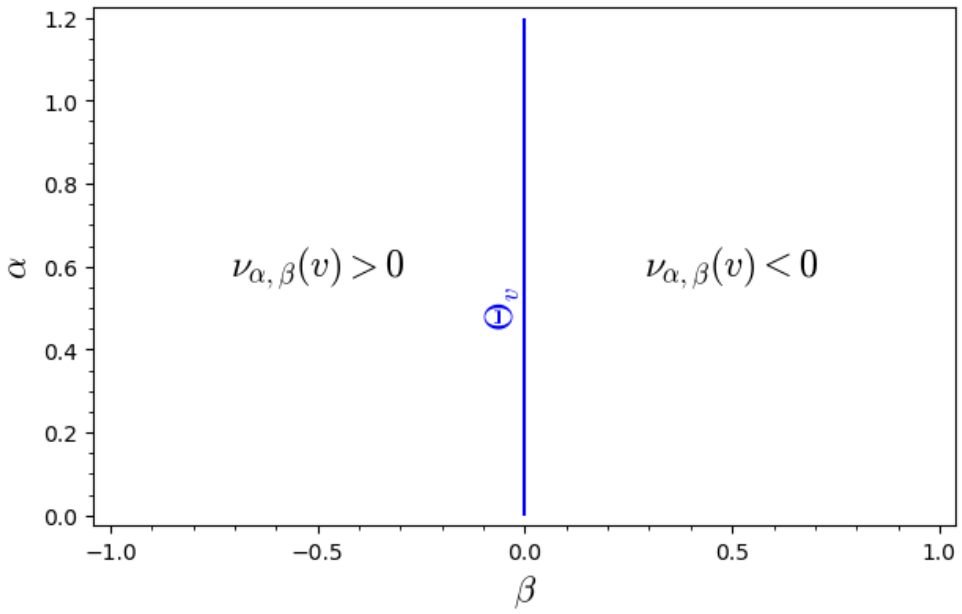
[3]: Chern Character:
$\mathrm{ch}_0 = 0$
$\mathrm{ch}_1 = 1\ell^1$
$\mathrm{ch}_2 = 0\ell^2$

[4]:
```
alpha_range = alpha, 0, 1.2
beta_range = beta, -1, 1
black_text_args = {"fontsize":"xx-large", "clip":True, "rgbcolor":"black"}

Theta_v_plot = implicit_plot(stability.Tilt().degree(v), beta_range, alpha_range)
Theta_v_plot += text(r"$\nu_{\alpha,\beta}(v) < 0$", (0.5, 0.6), **black_text_args)
Theta_v_plot += text(r"$\nu_{\alpha,\beta}(v) > 0$", (-0.5, 0.6), **black_text_args)
Theta_v_plot += text(r"$\Theta_v$", (-0.05, 0.5), rotation=90, fontsize="xx-large")
Theta_v_plot.axes_labels([r"$\beta$", r"$\alpha$"])
```

```
Theta_v_plot
```

[4]:



[5]:
```
alpha_range = alpha, 0, 2
beta_range = beta, -2, 2

u_1 = Chern_Char(1,2,1.8)
u_1
```

[5]:
Chern Character:
$\mathrm{ch}_0 = 1$
$\mathrm{ch}_1 = 2\ell^1$
$\mathrm{ch}_2 = 1.80000000000000\ell^2$

[6]:
```
u_2 = Chern_Char(1,-2,1.8)
u_2
```

[6]:
Chern Character:
$\mathrm{ch}_0 = 1$
$\mathrm{ch}_1 = -2\ell^1$
$\mathrm{ch}_2 = 1.80000000000000\ell^2$

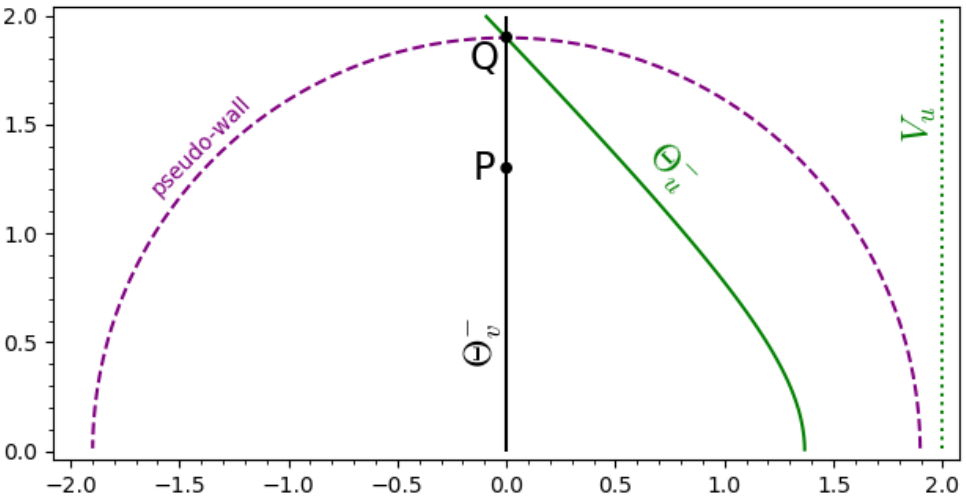
[14]:
```
stability.Tilt().rank(u_1)
```

[14]:
$-\alpha\beta + 2\,\alpha$

[36]:
```
( pseudo_semistab_char_curves_rank_zero :=
implicit_plot(stability.Tilt().wall_eqn(u_1, v)/alpha, beta_range, alpha_range,
 ↪linestyle="dashed", color="purple") \
+ implicit_plot(stability.Tilt().degree(u_1), beta_range, alpha_range,
 ↪color="green") \
+ implicit_plot(stability.Tilt().rank(u_1)/alpha, beta_range, alpha_range,
 ↪color="green", linestyle="dotted") \
+ text(r"$\Theta_v^{-}$", (-0.1, 0.5), rotation=90, fontsize="xx-large",
 ↪color="black") \
+ text(r"$V_u$", (1.9, 1.5), rotation=90, fontsize="xx-large", color="green") \
+ text(r"$\Theta_u^{-}$", (0.75, 1.3), rotation=-45, fontsize="xx-large",
 ↪color="green") \
+ point([0, 1.9], size=30, rgbcolor="black", zorder=50)
+ text("Q",[-0.1, 1.8], **black_text_args)
```

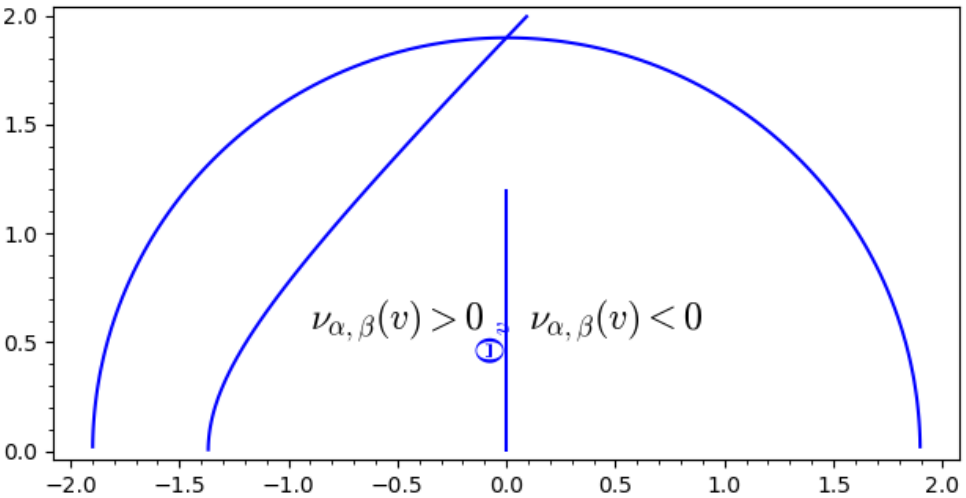
```
+ point([0, 1.3], size=30, rgbcolor="black", zorder=50)
+ text("P",[-0.1, 1.3], **black_text_args)
+ implicit_plot(stability.Tilt().degree(v), beta_range, alpha_range, color="black")
+ text("pseudo-wall",[-1.4, 1.4], color="purple", rotation=45)
)
```

[36]:



[8]:
```
implicit_plot(stability.Tilt().wall_eqn(u_2, v)/alpha, beta_range, alpha_range) \
+ implicit_plot(stability.Tilt().degree(u_2), beta_range, alpha_range) \
+ Theta_v_plot
```

[8]:



C.4 Examples

Notebook for the Recurring examples that appear across Part II

[1]:
```
# Requires extra package:
#! sage -pip install "pseudowalls==0.0.3" --extra-index-url https://gitlab.com/api/
↪v4/projects/43962374/packages/pypi/simple
```

```
%display latex

from pseudowalls import *

Δ = lambda v: v.Q_tilt()
mu = stability.Mumford().slope
ts = stability.Tilt

var("beta", domain="real")

def beta_minus(v):
    beta = stability.Tilt().beta
    solutions = solve(
        stability.Tilt(alpha=0).degree(v)==0,
        beta)
    return min(map(lambda s: s.rhs(), solutions))

class Object(object):
    pass
```

C.4.1 Define Cherns in Examples

Define two recurring examples used to illustrate performance of the different theorems to find semistabilizer bounds:

[2]:
```
recurring = Object()
recurring.chern = Chern_Char(3, 2, -2)
recurring.chern
```

[2]:
Chern Character:
$\mathrm{ch}_0 = 3$
$\mathrm{ch}_1 = 2\ell^1$
$\mathrm{ch}_2 = -2\ell^2$

[3]:
```
extravagant = Object()
extravagant.chern = Chern_Char(29, 13, -3/2)
extravagant.chern
```

[3]:
Chern Character:
$\mathrm{ch}_0 = 29$
$\mathrm{ch}_1 = 13\ell^1$
$\mathrm{ch}_2 = -\frac{3}{2}\ell^2$

C.4.2 Calculate Preliminary Quantities

[4]:
```
for example in [recurring, extravagant]:
    example.betaminus = beta_minus(example.chern)
    example.twisted = example.chern.twist(example.betaminus)
    example.n = example.betaminus.denominator()
    example.m = 1 # \ell^2 = 1 on P^1
    example.bgmlv = example.chern.Q_tilt()

# Actual maximal rank of Pseudo-Semistabilizers
# (needs to be calculated elsewhere)
recurring.actual_rmax = 25
extravagant.actual_rmax = 49313
```

C.4.3 First Loose Bound

Formula for loose bound:

[5]:
```
def loose_bound(example):
    n = example.n
    twisted = example.twisted
    return ( example.m*n^2*twisted.ch[1]^2
    ) / gcd(example.m, 2*n^2)

for example in [recurring, extravagant]:
```

```
    example.loose_bound = loose_bound(example)
```

Loose bounds for the two examples:

[6]:
```
recurring.loose_bound
```

[6]: 144

[7]:
```
extravagant.loose_bound
```

[7]: 215296

C.4.4 Stronger Convenient Bound

Use expression for bound used in main document

[8]:
```
from plots_and_expressions import main_theorem1, bgmlv_v, m, R, n, lcm_m_2n2
# Delta: symbol for Δ(v)
# n: symbol for denominator for β_(v)
# R : symbol for chern_0(v)
# nu : ...
main_theorem1.corollary_r_bound
```

[8]: $$\frac{1}{2}\,R + \frac{\Delta(v)\mathrm{lcm}(m, 2n^2)}{8\,m} + \frac{R^2 m}{2\,\Delta(v)\mathrm{lcm}(m, 2n^2)}$$

[9]:
```
def corrolary_bound(example):
    return (
        main_theorem1.corollary_r_bound
        .subs(bgmlv_v==example.bgmlv)
        .subs(m==example.m)
        .subs(R==example.chern.ch[0])
        .subs(n==example.n)
        .subs(lcm_m_2n2==lcm(example.m,2*example.n^2))
    )


for example in [recurring, extravagant]:
    example.corrolary_bound = corrolary_bound(example)
```

[10]:
```
extravagant.corrolary_bound
```

[10]: $$\frac{55130625}{1024}$$

[11]:
```
float(recurring.corrolary_bound)
```

[11]: 37.515625

C.4.5 Stronger Complicated Bounds

[12]:
```
import numpy as np

def bound_comparisons(example):
    n = example.n
    a_v = example.betaminus.numerator()

    def theorem_bound(v_twisted, q_val, k):
        return int(min(
            n^2*q_val^2/k,
            v_twisted.ch[0]
            + n^2*(v_twisted.ch[1] - q_val)^2/k
        ))

    def k(n, a_v, b_q):
        n = int(n)
        a_v = int(a_v)
        b_q = int(b_q)
```

```
        k = -a_v*b_q % n
        return k if k > 0 else k + n

    b_qs = list(range(example.twisted.ch[1]*n+1))
    qs = list(map(lambda x: x/n,b_qs))
    ks = list(map(lambda b_q: k(n, a_v, b_q), b_qs))
    theorem2_bounds = [
        theorem_bound(example.twisted, q_val, 1)
        for q_val in qs
    ]
    theorem3_bounds = [
        theorem_bound(example.twisted, q_val, k)
        for q_val, k in zip(qs,ks)
    ]
    return qs, theorem2_bounds, theorem3_bounds
```

Array content: - First row: q-values - Second row: Theorem 1 bounds on $ch_0(u)$ for u solutions to prob with $ch_1^\beta = q$ - Second row: Theorem 2 bounds on $ch_0(u)$ for u solutions to prob with $ch_1^\beta = q$

[13]:
```
np.array(bound_comparisons(recurring))
```

[13]:
```
[[0 1/3 2/3 1 4/3 5/3 2 7/3 8/3 3 10/3 11/3 4]
 [0 1 4 9 16 25 36 28 19 12 7 4 3]
 [0 0 4 3 8 25 12 15 19 6 5 4 3]]
```

C.5 Other P Choice

[1]:
```
from pseudowalls import *
%display latex
```

C.5.1 Initialize Cherns

[2]:
```
var("R C D r c d", domain="real")
var("A", latex_name=r"{\alpha_0}", domain="real")
var("B", latex_name=r"{\beta_0}", domain="real")
P = A, B
```

[3]:
```
v = Chern_Char(R,C,D)
v
```

[3]:
Chern Character:
$\mathrm{ch}_0 = R$
$\mathrm{ch}_1 = C\ell^1$
$\mathrm{ch}_2 = D\ell^2$

[4]:
```
twisted_v = Twisted_Chern_Char(B,
            R,
            var("twisted_v1", latex_name = r"\mathrm{ch}_1^{\beta_0}(v)",
↪domain="real"),
            var("twisted_v2", latex_name = r"\mathrm{ch}_2^{\beta_0}(v)",
↪domain="real"),
)
twisted_v
```

[4]:
Twisted Chern Character for $\beta = \beta_0$
$\mathrm{ch}_0 = R$
$\mathrm{ch}_1 = \mathrm{ch}_1^{\beta_0}(v)\ell^1$
$\mathrm{ch}_2 = \mathrm{ch}_2^{\beta_0}(v)\ell^2$

[5]:
```
assume(twisted_v2 > 0)
```

[6]:
```
u = Chern_Char(r,c,d)
u
```

[6]:

Chern Character:
$\mathrm{ch}_0 = r$
$\mathrm{ch}_1 = c\ell^1$
$\mathrm{ch}_2 = d\ell^2$

[7]:
```
twisted_u = Twisted_Chern_Char(B,
            r,
            var("twisted_u1", latex_name = r"\mathrm{ch}_1^{\beta_0}(u)",
↪domain="real"),
            var("twisted_u2", latex_name = r"\mathrm{ch}_2^{\beta_0}(u)",
↪domain="real"),
)
twisted_u
```

[7]:
Twisted Chern Character for $\beta = \beta_0$
$\mathrm{ch}_0 = r$
$\mathrm{ch}_1 = \mathrm{ch}_1^{\beta_0}(u)\ell^1$
$\mathrm{ch}_2 = \mathrm{ch}_2^{\beta_0}(u)\ell^2$

C.5.2 Numerical Conditions

Condition of $P = (A, B)$ being on Θ_v (i.e. $ch_2^{A,B}(v) = 0$) expressed in terms of twisted Chern character for v at $\beta = B$:

[8]:
```
A2_subs = solve(
    stability.Tilt(*P).degree(twisted_v) == 0,
    A^2)[0]

A2_subs
```

[8]:
$$\alpha_0{}^2 = \frac{2\,\mathrm{ch}_2^{\beta_0}(v)}{R}$$

Condition: $ch_2^P(u) > 0$

[9]:
```
( radius_condition_before_sub := stability.Tilt(*P).degree(twisted_u) > 0 )
```

[9]:
$$-\frac{1}{2}\,\alpha_0{}^2 r + \mathrm{ch}_2^{\beta_0}(u) > 0$$

[10]:
```
radius_condition = expand(
    ( radius_condition_before_sub / r ).expand().subs(
        A2_subs
    ) * r * R
)
radius_condition
```

[10]:
$$R\mathrm{ch}_2^{\beta_0}(u) - r\mathrm{ch}_2^{\beta_0}(v) > 0$$

[11]:
```
from plots_and_expressions import c_in_terms_of_q, beta
radius_condition_d_bound = (
    radius_condition
    .subs(twisted_u.ch[2] == u.twist(B).ch[2])
    .expand()
    .add_to_both_sides(B*R*c - B^2*R*r/2 + r*twisted_v.ch[2])
    .divide_both_sides(R)
    .subs(c_in_terms_of_q.subs(beta==B))
    .expand()
)
radius_condition_d_bound
```

[11]:
$$d > \frac{1}{2}\,\beta_0{}^2 r + \beta_0 q + \frac{r\mathrm{ch}_2^{\beta_0}(v)}{R}$$

Condition: $\Delta(u) \geq 0$

[12]:
```
from plots_and_expressions import bgmlv2_d_upperbound_terms
```

[13]:
```
(bgmlv2_d_upperbound_terms.linear
 + bgmlv2_d_upperbound_terms.const
 + bgmlv2_d_upperbound_terms.hyperbolic)
```

[13]:
$$\frac{1}{2}\,{\beta_0}^2 r + \beta_0 q + \frac{q^2}{2\,r}$$

Condition: $\Delta(v-u) \geq 0$

[14]:
```
from plots_and_expressions import bgmlv3_d_upperbound_terms, ch1bv, ch2bv, q

(bgmlv3_d_upperbound_terms.linear
 + bgmlv3_d_upperbound_terms.const
 + bgmlv3_d_upperbound_terms.hyperbolic)
```

[14]:
$$\frac{1}{2}\,{\beta_0}^2 r + \beta_0 q - \frac{\left(\mathrm{ch}_1^{\beta_0}(v) - q\right)^2}{2\,(R-r)} + \mathrm{ch}_2^{\beta_0}(v)$$

consider lowerbound inducesd when $r < R$ and compare to bound given by radius condition:

[81]:
```
#from plots_and_expressions import bgmlv3_d_upperbound_alt
(
(
    radius_condition_d_bound.rhs()
    - (bgmlv3_d_upperbound_terms.linear
     + bgmlv3_d_upperbound_terms.const
     + bgmlv3_d_upperbound_terms.hyperbolic)
).factor()* (R-r) * R #.subs(q==c-r*beta)#
).factor()
```

[81]:
$$\frac{1}{2}\,Rq^2 - Rq\mathrm{ch}_1^{\beta_0}(v) + \frac{1}{2}\,R\mathrm{ch}_1^{\beta_0}(v)^2 - R^2\mathrm{ch}_2^{\beta_0}(v) + 2\,Rr\mathrm{ch}_2^{\beta_0}(v) - r^2\mathrm{ch}_2^{\beta_0}(v)$$

Specialize last 2 conditions to $\beta = \beta_0$

[16]:
```
# substitutions to replace beta -> beta_0 in relevant equations to specialize to
↪problem 1
beta_substitutions = [beta==B, ch1bv==twisted_v.ch[1], ch2bv==twisted_v.ch[2]]
```

[17]:
```
for terms in [bgmlv2_d_upperbound_terms, bgmlv3_d_upperbound_terms]:
    terms.const = terms.const.subs(beta_substitutions)
    terms.linear = terms.linear.subs(beta_substitutions)
    terms.hyperbolic = terms.hyperbolic.subs(beta_substitutions)
    terms.all = terms.const + terms.linear + terms.hyperbolic
```

[18]:
```
bgmlv3_d_upperbound_terms.linear
```

[18]:
$$\frac{1}{2}\,{\beta_0}^2 r$$

[19]:
```
bgmlv2_d_upperbound_terms.linear
```

[19]:
$$\frac{1}{2}\,{\beta_0}^2 r$$

C.5.3 Example bounds

[20]:
```
from plots_and_expressions import v_example, beta_minus
v_example
```

[20]:
Chern Character:
$\mathrm{ch}_0 = 3$
$\mathrm{ch}_1 = 2\ell^1$
$\mathrm{ch}_2 = -2\ell^2$

[21]:
```
B_example = -2/3 - 1/99 # anything less than -2/3
```

[22]:
```
v_example_twisted = v_example.twist(B_example)
v_example_twisted
```

[22]:

Twisted Chern Character for $\beta = -\frac{67}{99}$

$\mathrm{ch}_0 = 3$
$\mathrm{ch}_1 = \frac{133}{33}\ell^1$
$\mathrm{ch}_2 = \frac{265}{6534}\ell^2$

[23]:
```
q_example = 2
```

[24]:
```
example_substitutions = [
    B == B_example,
    twisted_v.ch[0] == v_example_twisted.ch[0],
    twisted_v.ch[1] == v_example_twisted.ch[1],
    twisted_v.ch[2] == v_example_twisted.ch[2],
    q == q_example
]
```

[25]:
```
example_upperbound_1 = (bgmlv3_d_upperbound_terms.linear
 + bgmlv3_d_upperbound_terms.const
 + bgmlv3_d_upperbound_terms.hyperbolic).subs(example_substitutions)
example_upperbound_1
```

[25]:

$$\frac{4489}{19602}\,r + \frac{4489}{2178\,(r-3)} - \frac{8579}{6534}$$

[26]:
```
example_upperbound_2 = (bgmlv2_d_upperbound_terms.linear
 + bgmlv2_d_upperbound_terms.const
 + bgmlv2_d_upperbound_terms.hyperbolic).subs(example_substitutions)
example_upperbound_2
```

[26]:

$$\frac{4489}{19602}\,r + \frac{2}{r} - \frac{134}{99}$$

[27]:
```
example_lowerbound_1 = radius_condition_d_bound.subs(example_substitutions).rhs()
example_lowerbound_1
```

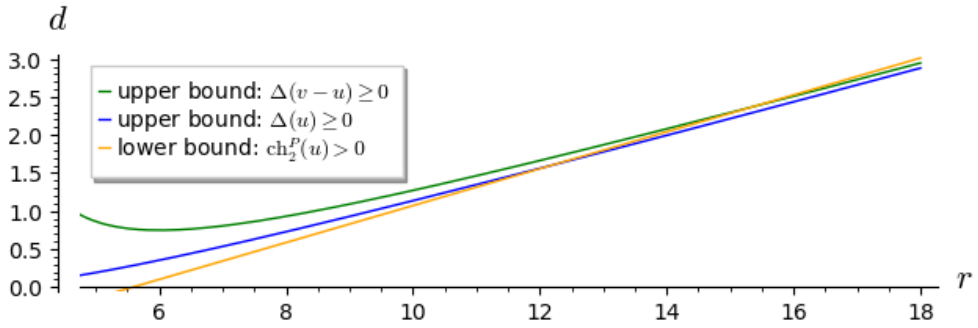
[27]:

$$\frac{2377}{9801}\,r - \frac{134}{99}$$

[28]:
```
r_range = r,0,18

example_plot = (
    plot(example_upperbound_1, (r,3,18), rgbcolor="green", legend_label=r"upper bound: $\Delta(v-u)\geq 0$")
    + plot(example_upperbound_2, r_range, legend_label=r"upper bound: $\Delta(u)\geq 0$")
    + plot(example_lowerbound_1, r_range, rgbcolor="orange", legend_label=r"lower bound: $\mathrm{ch}_2^P(u)>0$")
)
example_plot.ymin(0)
example_plot.ymax(3)
example_plot.xmin(5)
example_plot.set_aspect_ratio(1.2)
example_plot.axes_labels([r"$r$", r"$d$"])
example_plot
```

[28]:



C.5.4 Finding intersection points

Find intersection of lower bound with upper bound from bgmlv2:

[29]:
```
var("epsilon", latex_name=r"\varepsilon")
( intersections_bgmlv2 :=
    ( bgmlv2_d_upperbound_terms.all - radius_condition_d_bound.rhs() )
      .roots(r)
)
```

[29]:
$$\left[\left(-\sqrt{\frac{1}{2}}q\sqrt{\frac{R}{\mathrm{ch}_2^{\beta_0}(v)}},1\right),\left(\sqrt{\frac{1}{2}}q\sqrt{\frac{R}{\mathrm{ch}_2^{\beta_0}(v)}},1\right)\right]$$

[30]:
```
( positive_intersection_bgmlv2 :=
    intersections_bgmlv2[1][0]
)
```

[30]:
$$\sqrt{\frac{1}{2}}q\sqrt{\frac{R}{\mathrm{ch}_2^{\beta_0}(v)}}$$

Quickly verify this is indeed a root:

[31]:
```
(
    ( bgmlv2_d_upperbound_terms.all - radius_condition_d_bound.rhs() )
    .subs(r==positive_intersection_bgmlv2)
).factor()
```

[31]:
$$0$$

Find intersection of lower bound with upper bound from bgmlv2:

[32]:
```
var("epsilon", latex_name=r"\varepsilon")
( intersections_bgmlv3 :=
    ( bgmlv3_d_upperbound_terms.all - radius_condition_d_bound.rhs() )
      .roots(r)
)
```

[32]:
$$\left[\left(\frac{\sqrt{2}\sqrt{R\mathrm{ch}_2^{\beta_0}(v)}\left(q-\mathrm{ch}_1^{\beta_0}(v)\right)+2\,R\mathrm{ch}_2^{\beta_0}(v)}{2\,\mathrm{ch}_2^{\beta_0}(v)},1\right),\left(-\frac{\sqrt{2}\sqrt{R\mathrm{ch}_2^{\beta_0}(v)}\left(q-\mathrm{ch}_1^{\beta_0}(v)\right)-2\,R\mathrm{ch}_2^{\beta_0}(v)}{2\,\mathrm{ch}_2^{\beta_0}(v)},1\right)\right]$$

[33]:
```
( positive_intersection_bgmlv3 :=
    intersections_bgmlv3[1][0].expand().simplify()
)
```

[33]:
$$-\frac{\sqrt{2}\sqrt{R}q}{2\sqrt{\mathrm{ch}_2^{\beta_0}(v)}}+\frac{\sqrt{2}\sqrt{R}\mathrm{ch}_1^{\beta_0}(v)}{2\sqrt{\mathrm{ch}_2^{\beta_0}(v)}}+R$$

Quickly verify this is indeed a root too:

[34]:
```
(
    ( bgmlv3_d_upperbound_terms.all - radius_condition_d_bound.rhs() )
    .subs(r==positive_intersection_bgmlv3)
```

```
).factor()
```

[34]: 0

[35]:
```
_common_factor = sqrt(R)/sqrt(twisted_v2)/sqrt(2)
( r_bound_expression :=
    min_symbolic(
        positive_intersection_bgmlv2 / _common_factor,
        positive_intersection_bgmlv3 / _common_factor
    )
    .__mul__(_common_factor)
    .expand()
    .simplify()
)
```

[35]: $$\frac{\sqrt{2}\sqrt{R}\min\left(q, \sqrt{2}\sqrt{R}\sqrt{\mathrm{ch}_2^{\beta_0}(v)} - q + \mathrm{ch}_1^{\beta_0}(v)\right)}{2\sqrt{\mathrm{ch}_2^{\beta_0}(v)}}$$

Find which q value maximises the minimum of these two quantities

[36]:
```
(maximising_q :=
    (positive_intersection_bgmlv3 - positive_intersection_bgmlv2)
     .roots(q)[0][0]
     .expand()
     .simplify()
)
```

[36]: $$\frac{1}{2}\sqrt{2}\sqrt{R}\sqrt{\mathrm{ch}_2^{\beta_0}(v)} + \frac{1}{2}\mathrm{ch}_1^{\beta_0}(v)$$

[37]:
```
( r_max :=
    positive_intersection_bgmlv2
     .subs(q == maximising_q)
     .expand()
     .simplify()
)
```

[37]: $$\frac{\sqrt{2}\sqrt{R}\mathrm{ch}_1^{\beta_0}(v)}{4\sqrt{\mathrm{ch}_2^{\beta_0}(v)}} + \frac{1}{2}R$$

rational B, involving epsilon

[38]:
```
var("epsilon", latex_name=r"\varepsilon")
(
    ( bgmlv2_d_upperbound_terms.all - radius_condition_d_bound.rhs() - epsilon )
      .roots(r)
)
```

[38]: $$\left[\left(-\frac{R\varepsilon + \sqrt{R^2\varepsilon^2 + 2\,Rq^2\mathrm{ch}_2^{\beta_0}(v)}}{2\,\mathrm{ch}_2^{\beta_0}(v)}, 1\right), \left(-\frac{R\varepsilon - \sqrt{R^2\varepsilon^2 + 2\,Rq^2\mathrm{ch}_2^{\beta_0}(v)}}{2\,\mathrm{ch}_2^{\beta_0}(v)}, 1\right)\right]$$

C.6 Pseudo-walls for (R, 0, -D)

[1]:
```
from pseudowalls import *
%display latex

def Δ(chern):
    return chern.Q_tilt()

μ = stability.Mumford().slope

flip = lambda expression: expression.rhs() == expression.lhs()
```

Yanagida/Yoshioka found different ways to fit the Chern character $v = (1, 0, -\Delta)$ at the ends of short exact sequences in the form $\ell_1 w_1 \to \ell_2 w_2 \to v$ or $v \to \ell_1 w_1 \to \ell_2 w_2$ with w_i being primitive Chern characters with $\Delta(w_i) = 0$ and one of the ℓ_i being 1

One solution comes from $I_S \to O \to O_S$, giving an explicit sequence to go with it.

Then use theory of FMT and semi-homog sheaves to transform the latter sequence into genuine sequences with the other Chern characters.

I'm fairly certain $(2, 0, -3)$ cannot fit in such a sequence, but if you allow $\ell_1 = 3$ and $\ell_2 = 2$, then there are infinitely many solutions.

More Generally, take a Chern character v of the following form:

[2]:
```
var("R D", domain="real")
(v :=
 Integral_Chern(R, 0, -D)
)
```

[2]:

Chern Character:

$\mathrm{ch}_0 = R$

$\mathrm{ch}_1 = 0\frac{\ell^1}{1!}$

$\mathrm{ch}_2 = -D\frac{\ell^2}{2!}$

[3]:
```
var("gcd_RD", latex_name=r"\gcd(R,D)")
var("Rprime", latex_name=r"R^{'}")
var("Dprime", latex_name=r"D^{'}")
```

[3]:

$D^{'}$

[4]:
```
(RD_subs := [
    R == Rprime * gcd_RD,
    D == Dprime * gcd_RD
 ])
```

[4]:

$$\left[R = R^{'}\gcd(R, D), D = D^{'}\gcd(R, D)\right]$$

Let's try to find ways that v can fit into a sequence $Dw_1 \to Rw_2 \to v$ or $v \to Rw_2 \to Dw_1$.

To do this, take the general form of a primitive semi_homog $w_2 = (a^2, \pm ab, b^2)$ (start with +), then find when $\Delta(Rw_2 - v) = 0$, so that it can be of the form Dw_1.

[5]:
```
var("a b", domain="real")
w_2 = Integral_Chern(a^2, - a*b, b^2)
Rw_2 = Integral_Chern(R * a^2, - R * a*b, R * b^2)


(pre_pells_eq :=
    (Δ(Rw_2 - v) == 0)
    .divide_both_sides(R*D)
    .expand()
    .add_to_both_sides(a^2 - R*b^2/D)
)
```

[5]:

$$1 = a^2 - \frac{Rb^2}{D}$$

Using the fact that $R^{'}$ and $D^{'}$ are coprime, we can deduce that $D^{'}$ divides b, so we'll make the following substitution involving integers:

[6]:
```
var("bprime", latex_name=r"b^{'}")
(b_subs :=
    b == Dprime * bprime
)
```

[6]:

$$b = D^{'}b^{'}$$

[7]:
```
(pells_equation :=
    pre_pells_eq.subs(RD_subs).subs(b_subs)
)
```

[7]:

$$1 = -D^{'}R^{'}{b^{'}}^2 + a^2$$

Provided that $D^{'}R^{'}$ is not a square, this has infinitely many solutions for a, $b^{'}$ (and consequently a, b). Notice, that this is equivalent to $\Delta(v)$ not being square:

[8]:
```
Δ(v).subs(RD_subs)
```

[8]: $D^{'}R^{'}\gcd(R,D)^2$

Note: if $D^{'}R^{'}$ is square, then there are only two integer solutions: $a=\pm 1, b=b^{'}=0$

[9]:
```
(_substitution :=
    pre_pells_eq
    .solve(R) # rearrange for R'
    #.factor()
)
```

[9]: $\left[R=\frac{Da^2-D}{b^2}\right]$

Notice that w_1 for this solution is $(c^2,-ac\ell,a^2)$ where $c:=\frac{(a+1)(a-1)}{D^{'}b^{'}}$

[10]:
```
var("c", domain="real")
(c_dfn := c == (a+1)*(a-1)/b)
```

[10]: $c=\frac{(a+1)(a-1)}{b}$

[11]:
```
Dw_1 = Rw_2 - v

w_1 = Chern_Char(*map(
    lambda x: (x/D)
        .subs(_substitution)
        .subs(solve(c_dfn, b))
        .factor()
    ,
    Dw_1.ch
))
w_1
```

[11]: Chern Character:

$\mathrm{ch}_0=c^2$

$\mathrm{ch}_1=-ac\ell^1$

$\mathrm{ch}_2=\frac{1}{2}a^2\ell^2$

You get the same equation by changing - to + in the formula for w_2:

[12]:
```
Rw_2 = Integral_Chern(R * a^2, R * a*b, R * b^2)

(problem_dual :=
    (Δ(Rw_2 - v) == 0)
    .divide_both_sides(R*D)
    .expand()
    .add_to_both_sides(a^2 - R*b^2/D)
)
```

[12]: $1=a^2-\frac{Rb^2}{D}$

Iterative solution to Pell's equation

[13]:
```
def abprime_to_matrix(a_val, bprime_val):
    assert a_val.lhs() == a
    assert bprime_val.lhs() == bprime

    return matrix([
        [a, bprime*Dprime*Rprime],
        [bprime, a]
    ]).subs(a_val).subs(bprime_val)

abprime_to_matrix(a==a, bprime==bprime)
```

[13]: $\begin{pmatrix} a & D^{'}R^{'}b^{'} \\ b^{'} & a \end{pmatrix}$

[14]:
```
def abprime_to_ab(a_val, bprime_val):
    assert a_val.lhs() == a
    assert bprime_val.lhs() == bprime

    return a_val, b==bprime_val.rhs()*Dprime

abprime_to_ab(a==a, bprime==bprime)
```

[14]: $\left(a = a, b = D' b'\right)$

[15]:
```
def matrix_to_abprime(mat):
    return a==mat[0][0], bprime==mat[1][0]

matrix_to_abprime(abprime_to_matrix(a==a, bprime==bprime))
```

[15]: $\left(a = a, b' = b'\right)$

[16]:
```
def ab_to_abprime(a_val, b_val):
    assert a_val.lhs() == a
    assert b_val.lhs() == b
    return a_val, bprime==b_val.rhs()/Dprime

ab_to_abprime(a==a, b_subs)
```

[16]: $\left(a = a, b' = b'\right)$

w_2^n in terms of a_n, b_n':

[17]:
```
var("a_n b_n")
var("bprime_n", latex_name=r"b^{'}_n")
_aa, _bb = abprime_to_ab(a==a_n, bprime==bprime_n)
_a = _aa.rhs()
_b = _bb.rhs()
(nth_w2 := (_a^2,-_a*_b, _b^2))
```

[17]: $\left(a_n^2, -D' a_n b_n', D'^2 b_n'^2\right)$

[18]:
```
var("c_n")
(nth_c :=
    c_dfn.rhs().subs(b_subs).subs(bprime==bprime_n).subs(a==a_n)
)
```

[18]: $\frac{(a_n + 1)(a_n - 1)}{D' b_n'}$

[19]:
```
(nth_matrix := abprime_to_matrix(a==a_n, bprime==bprime_n))
```

[19]: $\begin{pmatrix} a_n & D' R' b_n' \\ b_n' & a_n \end{pmatrix}$

[20]:
```
var("a_1")
var("bprime_1", latex_name=r"b^{'}_1")
(first_matrix := abprime_to_matrix(a==a_1, bprime==bprime_1))
```

[20]: $\begin{pmatrix} a_1 & D' R' b_1' \\ b_1' & a_1 \end{pmatrix}$

Formulae for properties of these pseudowalls

The left side of the induced pseudowall is:

[21]:
```
(left_side := μ(w_1)
    .factor()
    .subs(c_dfn)
)
```

[21]: $-\frac{ab}{(a + 1)(a - 1)}$

And the right side is given by

[22]:
```
(right_side := μ(w_2))
```

[22]: $-\frac{b}{a}$

Giving a centre:

[23]:
```
(centre :=
    (
       (right_side + left_side) / 2
    ).factor()
)
```

[23]: $-\frac{(2\,a^2-1)b}{2\,(a+1)(a-1)a}$

[24]:
```
(radius :=
    (
        (right_side - left_side) / 2
    ).factor()
)
```

[24]: $\frac{b}{2\,(a+1)(a-1)a}$

Centre of C_n aligns with left side of C_{2n}

Suppose a_n, b_n is the solutions for a, b (and $b_n = b_n' D'$) corresponding to pseudowall C_n. We get the solutions $a = a_{2n}$, $b = b_{2n}$ for pseudowall C_{2n} by squaring the corresponding matrix:

Take the a, b' values corresponding to pseudowall C_n

[25]:
```
var("a_n b_n")
var("bprime_n", latex_name=r"b^{'}_n")
(Cn_vals := (a == a_n, bprime == bprime_n))
```

[25]: $\left(a = a_n, b' = b_n'\right)$

a, b' values corresponding to C_{2n} in terms of a_n, b_n':

[26]:
```
(C2n_vals :=
    matrix_to_abprime(
        abprime_to_matrix(*Cn_vals)^2
    )
)
```

[26]: $\left(a = D' R' {b_n'}^2 + a_n^2, b' = 2\,a_n b_n'\right)$

Noticing that a_n, b_n satisfies `problem`, we can make the following substitution into the above:

[27]:
```
(_substitution :=
    pells_equation
    .subs(Cn_vals)
    .solve(Rprime)[0] # rearrange for R'
    .factor()
)
```

[27]: $R' = \frac{(a_n+1)(a_n-1)}{D' {b_n'}^2}$

Apply this substitution to the a, b values corresponding to C_n (stored in `C2n_vals`):

[28]:
```
(C2n_vals :=
    tuple(
        x.subs(_substitution).expand()
        for x in C2n_vals
    )
)
```

[28]: $\left(a = 2\,a_n^2 - 1, b' = 2\,a_n b_n'\right)$

[29]:
```
(C2n_left_side:=
    left_side
    .subs(abprime_to_ab(*C2n_vals))
    .expand()
    .factor()
)
```

[29]:
$$-\frac{\left(2\,a_n^2-1\right)D'\,b_n'}{2\,(a_n+1)(a_n-1)a_n}$$

[30]:
```
(Cn_centre:=
    centre
    .subs(abprime_to_ab(*Cn_vals))
    .expand()
    .factor()
)
```

[30]:
$$-\frac{\left(2\,a_n^2-1\right)D'\,b_n'}{2\,(a_n+1)(a_n-1)a_n}$$

Indeed, these two values match:

[31]:
```
assert C2n_left_side == Cn_centre
```

Fixed point of FMT corresponding to (a_{2n}, b_{2n}) is the apex of C_n

[32]:
```
C2n_vals
```

[32]:
$$\left(a=2\,a_n^2-1, b'=2\,a_n b_n'\right)$$

[33]:
```
(Cn_apex_alpha :=
    radius
        .subs(abprime_to_ab(*Cn_vals))
        .expand()
)
```

[33]:
$$\frac{D'\,b_n'}{2\,(a_n+1)(a_n-1)a_n}$$

the FMT corresponding to (a_{2n}, b_{2n}) should fix a point with $\alpha = \frac{1}{c}$ for the value of c corresponding to C_{2n}:

[34]:
```
(fixed_alpha_val :=
    (1/c)
        .subs(c_dfn)
        .subs(abprime_to_ab(*C2n_vals))
        .expand()
        .factor()
)
```

[34]:
$$\frac{D'\,b_n'}{2\,(a_n+1)(a_n-1)a_n}$$

Indeed, these two values are equal:

[35]:
```
assert Cn_apex_alpha == fixed_alpha_val
```

C.6.1 (2, 0, -3) example

Specialize the above problem with the following values:

[36]:
```
example_values = [
    Rprime == 2,
    Dprime == 3,
    R == 2,
    D == 3,
]
```

[37]:
```
example_problem = pells_equation.subs(example_values)
example_problem
```

[37]:
$1 = a^2 - 6\,b'^2$
Trial and error gives the first solution with $a = 5, b = 6$

[38]:
```
Co_vals = [a==5, bprime==2]

example_problem.subs(Co_vals)
```

[38]:
$1 = 1$
Similarly to Pell's equation theory, take powers $\begin{pmatrix} 5 & \frac{2}{3}6 \\ 6 & 5 \end{pmatrix}^n$ to get other solutions $\begin{pmatrix} a & \frac{2}{3}b \\ b & a \end{pmatrix}$

[39]:
```
def abprime_pairs(N=20):
    first = abprime_to_matrix(*Co_vals).subs(example_values)
    current = first
    for _ in range(N):
        yield matrix_to_abprime(current) # contents of first column
        current *= first
```

[40]:
```
for aa,bb in abprime_pairs(4):
    print(f"{aa}\t\t {bb}")
```

```
a == 5          bprime == 2
a == 49         bprime == 20
a == 485                bprime == 198
a == 4801               bprime == 1960
```

Using the '-' flips the signs of the slopes for w_i (and $\mu(v) = 0$), giving 'left' pseudowalls
The corresponding solutions here also happen to satisfy $\Delta(w_1, w_2) = -1$

C.6.2 Plot wall with characteristic curves

[41]:
```
v
```

[41]:
Chern Character:
$\mathrm{ch}_0 = R$
$\mathrm{ch}_1 = 0\frac{\ell^1}{1!}$
$\mathrm{ch}_2 = -D\frac{\ell^2}{2!}$

[42]:
```
w_2
```

[42]:
Chern Character:
$\mathrm{ch}_0 = a^2$
$\mathrm{ch}_1 = -ab\frac{\ell^1}{1!}$
$\mathrm{ch}_2 = b^2\frac{\ell^2}{2!}$

[43]:
```
w_1
```

[43]:
Chern Character:
$\mathrm{ch}_0 = c^2$
$\mathrm{ch}_1 = -ac\ell^1$
$\mathrm{ch}_2 = \frac{1}{2}\,a^2\ell^2$

[44]:
```
(v_example := Integral_Chern(2, 0, -3))
```

[44]:
Chern Character:
$\mathrm{ch}_0 = 2$
$\mathrm{ch}_1 = 0\frac{\ell^1}{1!}$
$\mathrm{ch}_2 = -3\frac{\ell^2}{2!}$

[45]:
```
(v_theta :=
    stability.Tilt().degree(v).subs(example_values)
)
```

[45]:
$-\alpha^2 + \beta^2 - \frac{3}{2}$

[46]:
```
(w_1_theta :=
    stability.Tilt().degree(w_1)
    .subs(c_dfn)
```

```
    .subs(b_subs)
    .subs(Co_vals)
    .subs(example_values)
)
```

[46]: $-8\,\alpha^2 + 8\,\beta^2 + 20\,\beta + \frac{25}{2}$

[47]:
```
(w_2_theta :=
    stability.Tilt().degree(w_2)
    .subs(b_subs)
    .subs(Co_vals)
    .subs(example_values)
)
```

[47]: $-\frac{25}{2}\,\alpha^2 + \frac{25}{2}\,\beta^2 + 30\,\beta + 18$

[48]:
```
alpha = stability.Tilt().alpha
beta = stability.Tilt().beta
(w_1_V :=
    stability.Tilt().rank(w_1)
    .subs(c_dfn)
    .subs(b_subs)
    .subs(Co_vals)
    .subs(example_values)
    ._div_(alpha)
    .factor()
)
```

[48]: $-16\,\beta - 20$

[49]:
```
(w_2_V :=
    stability.Tilt().rank(w_2)
    .subs(b_subs)
    .subs(Co_vals)
    .subs(example_values)
    ._div_(alpha)
    .factor()
)
```

[49]: $-25\,\beta - 30$

[50]:
```
(v_V :=
    stability.Tilt().rank(v)
    .subs(example_values)/alpha
)
```

[50]: $-2\,\beta$

[51]:
```
(example_wall_eqn :=
    stability.Tilt().wall_eqn(v, w_2)
    .subs(b_subs)
    .subs(Co_vals)
    .subs(example_values)
    ._div_(alpha)
    .factor()
)
```

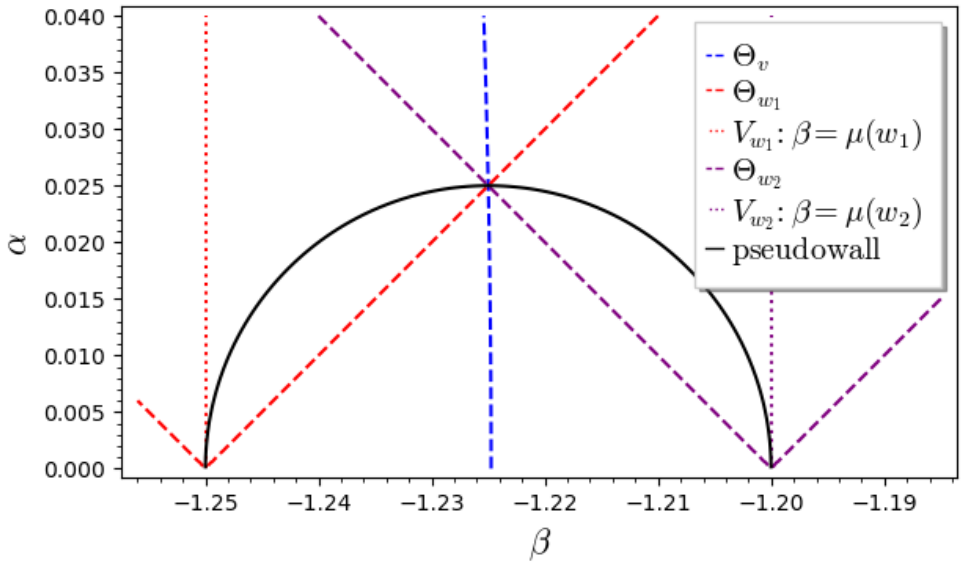
[51]: $30\,\alpha^2 + 30\,\beta^2 + \frac{147}{2}\,\beta + 45$

[52]:
```
alpha_range = alpha, 0, 0.04
beta_range = beta, -1.256, -1.185
v_plot_options = {"rgbcolor" : "blue"}
w_1_plot_options = {"rgbcolor" : "red"}
w_2_plot_options = {"rgbcolor" : "purple"}
theta_options = {"linestyle": "dashed"}
```

```
V_options = {"linestyle": "dotted"}
wall_options = {"rgbcolor": "black"}
special_wall_plot = (
    implicit_plot(v_theta, beta_range, alpha_range, **v_plot_options,
↪**theta_options)
    #+ implicit_plot(v_V, beta_range, alpha_range, **v_plot_options)
    + implicit_plot(w_1_theta, beta_range, alpha_range, **w_1_plot_options,
↪**theta_options)
    + implicit_plot(w_1_V, beta_range, alpha_range, **w_1_plot_options, **V_options)
    + implicit_plot(w_2_theta, beta_range, alpha_range, **w_2_plot_options,
↪**theta_options)
    + implicit_plot(w_2_V, beta_range, alpha_range, **w_2_plot_options, **V_options)
    + implicit_plot(example_wall_eqn, beta_range, alpha_range, **wall_options)

    + line(((-1.2,0),(-1.2,0)), **v_plot_options, **theta_options,
↪legend_label=r"$\Theta_v$")
    + line(((-1.2,0),(-1.2,0)), **w_1_plot_options, **theta_options,
↪legend_label=r"$\Theta_{w_1}$")
    + line(((-1.2,0),(-1.2,0)), **w_1_plot_options, **V_options,
↪legend_label=r"$V_{w_1} \colon\: \beta=\mu(w_1)$")
    + line(((-1.2,0),(-1.2,0)), **w_2_plot_options, **theta_options,
↪legend_label=r"$\Theta_{w_2}$")
    + line(((-1.2,0),(-1.2,0)), **w_2_plot_options, **V_options,
↪legend_label=r"$V_{w_2} \colon\: \beta=\mu(w_2)$")
    + line(((-1.2,0),(-1.2,0)), **wall_options,
↪legend_label=r"$\mathrm{pseudowall}$")
)
special_wall_plot.set_legend_options(font_size="x-large")
special_wall_plot.axes_labels([r"$\beta$", r"$\alpha$"])
special_wall_plot
```

[52]:



Plot for nested walls of interest, however $(2, 0, -3)$, but it turns out they get very very small very very quickly:

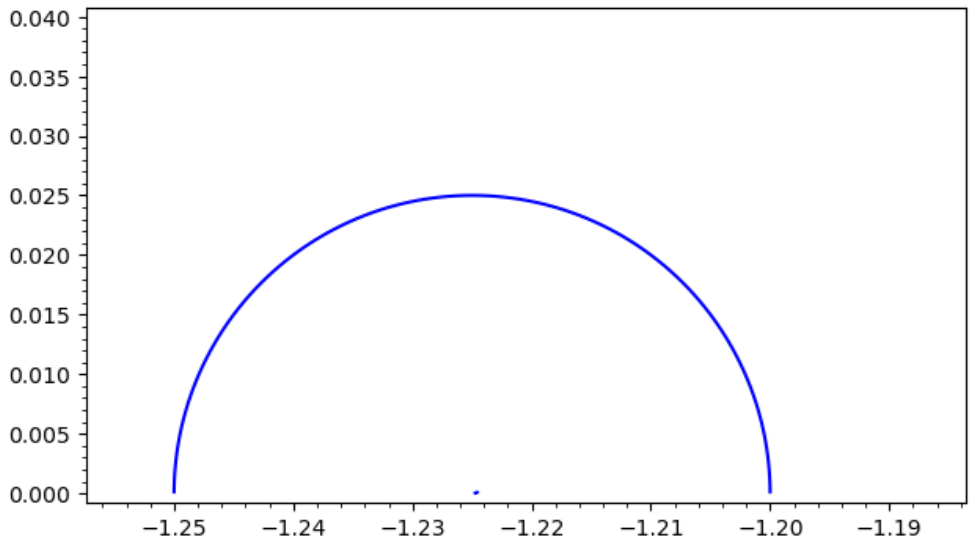
[53]:

```
alpha_range = alpha, 0, 0.04
beta_range = beta, -1.256, -1.185

pplot = line([(0,0),(0,0)])
for _aa, _bb in map(
```

```
    lambda x: abprime_to_ab(*x),
    abprime_pairs(3)
):
    assert _bb.lhs()==b
    assert _aa.lhs()==a
    _w_2 = Chern_Char(*(
        term.subs(_aa).subs(_bb)
        for term in
        w_2.ch
    ))
    pplot += implicit_plot(
        stability.Tilt().wall_eqn(v, w_2)
        .subs(_aa,_bb)
        .subs(example_values)
        ._div_(alpha)
        .expand()
        ,
        beta_range,
        alpha_range
    )
pplot.xmin(beta_range[1])
pplot.xmax(beta_range[2])
pplot.ymin(alpha_range[1])
pplot.ymax(alpha_range[2])
pplot
```

[53]:



C.7 Existence of Wall Argument for (R, 0, -D)

[1]:

```
from pseudowalls import *
from general_presentations import c_dfn
%display latex

def Δ(chern):
    return chern.Q_tilt()

μ = stability.Mumford().slope

flip = lambda expression: expression.rhs() == expression.lhs()
```

```
class Ppas_Chern(Integral_Chern):
    def _latex_(self):
        r, c, d = self.ch
        return rf"\left({latex(r)},\:{latex(c)},\:{latex(2*d)}\right)"
```

```
a == 5          bprime == 2
a == 49         bprime == 20
a == 485                bprime == 198
a == 4801               bprime == 1960
```

C.7.1 Trying to prove existence of walls

Showing translation of the stab conds under the FMTs

[2]:
```
var("a b c d")
(fmt_of_interest_inv := matrix([[a,-b],[-c,a]]))
```

[2]:
$$\begin{pmatrix} a & -b \\ -c & a \end{pmatrix}$$

[3]:
```
(structure_sheaf := Ppas_Chern(1,0,0))
```

[3]: $(1,\ 0,\ 0)$

[4]:
```
(skyscraper_sheaf := Ppas_Chern(0,0,1))
```

[4]: $(0,\ 0,\ 1)$

Define the action of cohomological FMT:

[5]:
```
def chern_to_matrix(chern):
    r,c,d = chern.ch
    return matrix([[r,c],[c,2*d]])

def matrix_to_chern(mat):
    r = mat[0][0]
    c = mat[0][1]
    chi = mat[1,1]
    return Ppas_Chern(r,c,chi)
```

[6]:
```
chern_to_matrix(skyscraper_sheaf)
```

[6]:
$$\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

[7]:
```
chern_to_matrix(structure_sheaf)
```

[7]:
$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

[8]:
```
matrix_to_chern(chern_to_matrix(structure_sheaf))
```

[8]: $(1,\ 0,\ 0)$

[9]:
```
matrix_to_chern(chern_to_matrix(skyscraper_sheaf))
```

[9]: $(0,\ 0,\ 1)$

[10]:
```
def apply_fmt(fmt, chern):
    return matrix_to_chern(fmt.transpose()*chern_to_matrix(chern)*fmt)
```

The inverse of the FMT of interest sends the skyscaper and structure sheaf to the w_i (expressions in a, b, c), check this property:

[11]:
```
apply_fmt(fmt_of_interest_inv, structure_sheaf)
```

[11]: $(a^2,\ -ab,\ b^2)$

[12]:
```
apply_fmt(fmt_of_interest_inv, skyscraper_sheaf)
```

[12]: $(c^2,\ -ac,\ a^2)$

So indeed, the cohomological FMT of interest is the inverse of the one above:

[13]:
```
(fmt_of_interest := matrix([[a,b],[c,a]]))
```

[13]:
$$\begin{pmatrix} a & b \\ c & a \end{pmatrix}$$

[14]:
```
var("alpha beta", domain="real")
```

[14]:
$$(\alpha, \beta)$$

[15]:
```
in_chern = exponential_chern(-a/c + I*alpha, g=2)
out = apply_fmt(fmt_of_interest, in_chern)
rr, cc, dd = out.ch
(
    rr.subs(c_dfn).expand(),
    cc.expand(),
    dd.expand()
)
```

[15]:
$$\left(-\frac{a^4\alpha^2}{b^2} + \frac{2i\,a^3\alpha}{b} - \frac{2i\,a^5\alpha}{(a+1)(a-1)b} + a^2 - \frac{2\,a^4}{(a+1)(a-1)} + \frac{a^6}{(a+1)^2(a-1)^2} + \frac{2\,a^2\alpha^2}{b^2} - \frac{2i\,a\alpha}{b} + \frac{4i\,a^3\alpha}{(a+1)(a-1)b} + \frac{2\,c}{(a+1)}\right.$$

[16]:
```
(out_matrix := chern_to_matrix(out).expand().factor())
```

[16]:
$$\begin{pmatrix} -\alpha^2c^2 & -(a\alpha c + i\,a^2 - i\,bc)\alpha \\ -(a\alpha c + i\,a^2 - i\,bc)\alpha & -\frac{a^2\alpha^2c^2+2i\,a^3\alpha c-2i\,a\alpha bc^2-a^4+2\,a^2bc-b^2c^2}{c^2} \end{pmatrix}$$

[17]:
```
(out_chern := matrix_to_chern(out_matrix.expand()))
```

[17]:
$$\left(-\alpha^2c^2,\ -a\alpha^2c - i\,a^2\alpha + i\,\alpha bc,\ -a^2\alpha^2 + 2i\,a\alpha b - \frac{2i\,a^3\alpha}{c} + b^2 + \frac{a^4}{c^2} - \frac{2\,a^2b}{c}\right)$$

[18]:
```
(llambda := out_chern.ch[0])
```

[18]:
$$-\alpha^2c^2$$

[19]:
```
(scaled_out_chern := matrix_to_chern(
    (out_matrix)
    ._div_(llambda)
    .expand()
))
```

[19]:
$$\left(1,\ \frac{a}{c} + \frac{i\,a^2}{\alpha c^2} - \frac{i\,b}{\alpha c},\ \frac{a^2}{c^2} + \frac{2i\,a^3}{\alpha c^3} - \frac{2i\,ab}{\alpha c^2} - \frac{a^4}{\alpha^2c^4} + \frac{2\,a^2b}{\alpha^2c^3} - \frac{b^2}{\alpha^2c^2}\right)$$

$\beta' =$

[20]:
```
( beta_prime :=
    scaled_out_chern.ch[1].real_part()
)
```

[20]:
$$\frac{a}{c}$$

$\alpha' =$

[21]:
```
( alpha_prime_unsimplified :=
    scaled_out_chern.ch[1]
    .imag_part()
    .factor()
)
```

[21]:
$$\frac{a^2 - bc}{\alpha c^2}$$

[22]:
```
( alpha_prime :=
    alpha_prime_unsimplified
    .subs(solve(c_dfn, a^2)[0])
)
```

[22]:
$$\frac{1}{\alpha c^2}$$

C.8 Transferring Walls via FMT

[1]:
```
from pseudowalls import *
%display latex

def chern_to_matrix(v):
    a,b,c = v.ch
    return matrix([
        [a, b],
        [b, 2*c]
    ])

def matrix_to_chern(mat):
    r,c,d = mat[0,0], mat[0,1], mat[1,1]
    return Integral_Chern(r,c,d)

def apply_fmt(fmt, v):
    return matrix_to_chern(
        fmt * chern_to_matrix(v) * fmt.transpose()
    )

def expand_chern(v):
    r, c, d = v.ch
    return Integral_Chern(r.expand(), c.expand(), (2*c).expand())
```

[2]:
```
var("R C D")
v = Integral_Chern(R,C,D)
v
```

[2]:
Chern Character:
$\mathrm{ch}_0 = R$
$\mathrm{ch}_1 = C\frac{\ell^1}{1!}$
$\mathrm{ch}_2 = D\frac{\ell^2}{2!}$

[3]:
```
matrix_to_chern(chern_to_matrix(v))
```

[3]:
Chern Character:
$\mathrm{ch}_0 = R$
$\mathrm{ch}_1 = C\frac{\ell^1}{1!}$
$\mathrm{ch}_2 = D\frac{\ell^2}{2!}$

[4]:
```
var("a b c d")
fmt = matrix([
    [a, b],
    [c, d]
])
fmt
```

[4]:
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

[5]:
```
v_transform = expand_chern(apply_fmt(fmt, v))
v_transform
```

[5]:
Chern Character:
$\mathrm{ch}_0 = Ra^2 + 2\,Cab + Db^2$
$\mathrm{ch}_1 = (Rac + Cbc + Cad + Dbd)\,\frac{\ell^1}{1!}$
$\mathrm{ch}_2 = (2\,Rac + 2\,Cbc + 2\,Cad + 2\,Dbd)\,\frac{\ell^2}{2!}$

[6]:
```
mu = stability.Mumford().slope
mu(v)
```

[6]:
$$\frac{C}{R}$$

[7]:
```
c_expression = (fmt.determinant()==1).solve(c)[0]
c_expression
```

[7]:
$$c = \frac{ad - 1}{b}$$

[8]:
```
mu(v_transform).subs(c_expression).factor()
```

[8]:
$$\frac{Ra^2d + 2\,Cabd + Db^2d - Ra - Cb}{(Ra^2 + 2\,Cab + Db^2)b}$$

[9]:
```
def source_beta(fmt):
    return - fmt[0,0]/fmt[0,1]

source_beta(fmt)
```

[9]:
$$-\frac{a}{b}$$

[10]:
```
def output_beta(fmt):
    return fmt[1,1]/fmt[0,1]

output_beta(fmt)
```

[10]:
$$\frac{d}{b}$$

C.8.1 Transform sending stability conds to opposite side of V_*

[11]:
```
beta_source_mu_offset = source_beta(fmt) - mu(v)
beta_source_mu_offset
```

[11]:
$$-\frac{C}{R} - \frac{a}{b}$$

[12]:
```
beta_out_mu_offset = output_beta(fmt) - mu(v_transform).subs(c_expression)
beta_out_mu_offset.factor()
```

[12]:
$$\frac{Ra + Cb}{(Ra^2 + 2\,Cab + Db^2)b}$$

I'm hoping that the offsets are always opposite signs?? i.e. that the following always has the same sign

[13]:
```
product_of_offsets = (
    beta_out_mu_offset * beta_source_mu_offset
).expand().factor()

product_of_offsets
```

[13]:
$$-\frac{(Ra + Cb)^2}{(Ra^2 + 2\,Cab + Db^2)Rb^2}$$

C.8.2 Limit of the transforms precomposed with stabilisers

[14]:
```
x, y = var("x_n y_n")
stabiliser_element_input = matrix([
    [x - C*y, R*y],
    [-D*y, x + C*y]
])
stabiliser_element_input
```

[14]:
$$\begin{pmatrix} -Cy_n + x_n & Ry_n \\ -Dy_n & Cy_n + x_n \end{pmatrix}$$

[15]:
```
pre_composition = (fmt * stabiliser_element_input).expand()
pre_composition
```

[15]:
$$\begin{pmatrix} -Cay_n - Dby_n + ax_n & Ray_n + Cby_n + bx_n \\ -Ccy_n - Ddy_n + cx_n & Rcy_n + Cdy_n + dx_n \end{pmatrix}$$

One-liner version of everything following:

[24]:
```
var("Delta")
(
    (
        ( source_beta(pre_composition) - mu(v))
```

```
        .factor()
        .subs(x==y*sqrt(C^2-D*R)) * R
    )^2
).expand().factor()
```

[24]: $C^2 - DR$

[25]:
```
beta_dom_offset = (source_beta(pre_composition) - mu(v)).factor()
beta_dom_offset
```

[25]: $$-\frac{C^2by_n - DRby_n + Rax_n + Cbx_n}{(Ray_n + Cby_n + bx_n)R}$$

[26]:
```
var("x_over_y", latex_name=r"\frac{x_n}{y_n}")
( beta_dom_offset_times_R :=
    beta_dom_offset
    # action of dividing numerator and denominator by y:
    .subs(x==x_over_y*y)
    .factor()
    * R
)
```

[26]: $$-\frac{C^2b - DRb + Ra\frac{x_n}{y_n} + Cb\frac{x_n}{y_n}}{Ra + Cb + b\frac{x_n}{y_n}}$$

[27]:
```
( beta_dom_offset_times_R_squared_plus :=
    beta_dom_offset_times_R
    .subs(x_over_y==sqrt(C^2-D*R))
    ^2
)
```

[27]: $$\frac{\left(C^2b - DRb + \sqrt{C^2 - DR}Ra + \sqrt{C^2 - DR}Cb\right)^2}{\left(Ra + Cb + \sqrt{C^2 - DR}b\right)^2}$$

[28]:
```
( beta_dom_offset_times_R_squared_minus :=
    beta_dom_offset_times_R
    .subs(x_over_y==-sqrt(C^2-D*R))
    ^2
)
```

[28]: $$\frac{\left(C^2b - DRb - \sqrt{C^2 - DR}Ra - \sqrt{C^2 - DR}Cb\right)^2}{\left(Ra + Cb - \sqrt{C^2 - DR}b\right)^2}$$

Verify that this simplify down:

[29]:
```
beta_dom_offset_times_R_squared_plus.expand().factor()
```

[29]: $C^2 - DR$

[30]:
```
beta_dom_offset_times_R_squared_minus.expand().factor()
```

[30]: $C^2 - DR$

C.9 Stabilising Matrices

[1]:
```
%display latex
```

C.9.1 Generals

I want to find some FMTs which fix an arbitrary Chern character $v = (R, C\ell, D)$ on a principally polarized abelian surface.

[2]:
```
from pseudowalls import *

class Ppas_Chern(Integral_Chern):
    def _latex_(self):
```

```
        r, c, d = self.ch
        return rf"\left({latex(r)},\:{latex(c)},\:{latex(2*d)}\right)"

def chern_to_matrix(chern):
    r,c,d = chern.ch
    return matrix([[r,c],[c,2*d]])

def matrix_to_chern(mat):
    r = mat[0][0]
    c = mat[0][1]
    chi = mat[1,1]
    return Ppas_Chern(r,c,chi)

def apply_fmt(fmt, chern):
    return matrix_to_chern(fmt*chern_to_matrix(chern)*fmt.transpose())
```

[3]:
```
var("R C D")
```

[3]: (R, C, D)

[4]:
```
(v :=
    Ppas_Chern(R, C, D)
)
```

[4]: $(R,\ C,\ D)$

[5]:
```
chern_to_matrix(v)
```

[5]: $\begin{pmatrix} R & C \\ C & D \end{pmatrix}$

[6]:
```
var("a b c d")
```

[6]: (a, b, c, d)

[7]:
```
(fmt :=
    matrix([[a, b],[c, d]])
)
```

[7]: $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$

C.9.2 Lemma: Reducing problem to formal Chern characters of degree 0

[8]:
```
v.twist( C/R )
```

[8]: Twisted Chern Character for $\beta = \dfrac{C}{R}$

$\mathrm{ch}_0 = R$

$\mathrm{ch}_1 = 0\frac{\ell^1}{1!}$

$\mathrm{ch}_2 = \left(D - \frac{C^2}{R}\right)\frac{\ell^2}{2!}$

[9]:
```
(twisting := matrix([[1, 0],[-C/R, 1]]))
```

[9]: $\begin{pmatrix} 1 & 0 \\ -\frac{C}{R} & 1 \end{pmatrix}$

[10]:
```
apply_fmt(twisting, v)
```

[10]: $\left(R,\ 0,\ D - \dfrac{C^2}{R}\right)$

Suppose F is a matrix corresponding to an FMT fixing v (`fmt`), A is the matrix representing v (`chern_to_matrix(v)`), and T is the matrix corresponding to the twisting of v which gives a c_1 of 0 (`twisting`), which is represented by the matrix $A' = TAT^T$.

$$
\begin{aligned}
A &= FAF^T \\
T^{-1}TAT^TT^{-1,T} &= FT^{-1}TAT^TT^{-1,T}F^T \\
A^{'} &= TFT^{-1}A^{'}T^{-1,T}F^TT^T \\
A^{'} &= F^{'}A^{'}F^{',T} \qquad \text{where: } F^{'} = TFT^{-1}
\end{aligned}
$$

So we can reframe the problem of fixing v with the problem of fixing it's formal twisting.
That is, we want to find FMTs of the following form...

[11]:
```
(fmt_prime :=
    (twisting * fmt * twisting.inverse()).factor()
)
```

[11]:
$$\begin{pmatrix} \frac{Ra+Cb}{R} & b \\ -\frac{CRa+C^2b-R^2c-CRd}{R^2} & -\frac{Cb-Rd}{R} \end{pmatrix}$$

Which fix the chern character

[12]:
```
( v_prime :=
    apply_fmt(twisting, v)
)
```

[12]:
$$\left(R,\ 0,\ D-\frac{C^2}{R}\right)$$

C.9.3 Lemma Stabilizer for diagonal matrix

Stabilizers of $diag(q_1, q_2)$ are of the following form:

[13]:
```
var("q_1 q_2")
var("x y")
stab_over_Q = matrix([
    [x, -q_1*y],
    [q_2*y, x]
])
stab_over_Q
```

[13]:
$$\begin{pmatrix} x & -q_1y \\ q_2y & x \end{pmatrix}$$

such that:

[14]:
```
stab_over_Q.determinant() == 1
```

[14]:
$$q_1q_2y^2 + x^2 = 1$$

C.9.4 General stabiliser lemma

Stabilisers of...

[15]:
```
tTAt = chern_to_matrix(v_prime)
tTAt
```

[15]:
$$\begin{pmatrix} R & 0 \\ 0 & D-\frac{C^2}{R} \end{pmatrix}$$

... are of form (by previous lemma)...

[16]:
```
stab_of_vprime = (
    stab_over_Q
    .subs([q_1 == tTAt[0,0], q_2 == tTAt[1,1]])
)
stab_of_vprime
```

[16]:
$$\begin{pmatrix} x & -Ry \\ \left(D-\frac{C^2}{R}\right)y & x \end{pmatrix}$$

...twist back to give stabiliser of original matrix:

[17]:
```
twist_back_of_vprime_stab = (
    twisting * stab_of_vprime * twisting.inverse()
).factor()
twist_back_of_vprime_stab
```

[17]:
$$\begin{pmatrix} -Cy + x & -Ry \\ Dy & Cy + x \end{pmatrix}$$

...then the rest of the argument is modular arithmetic...