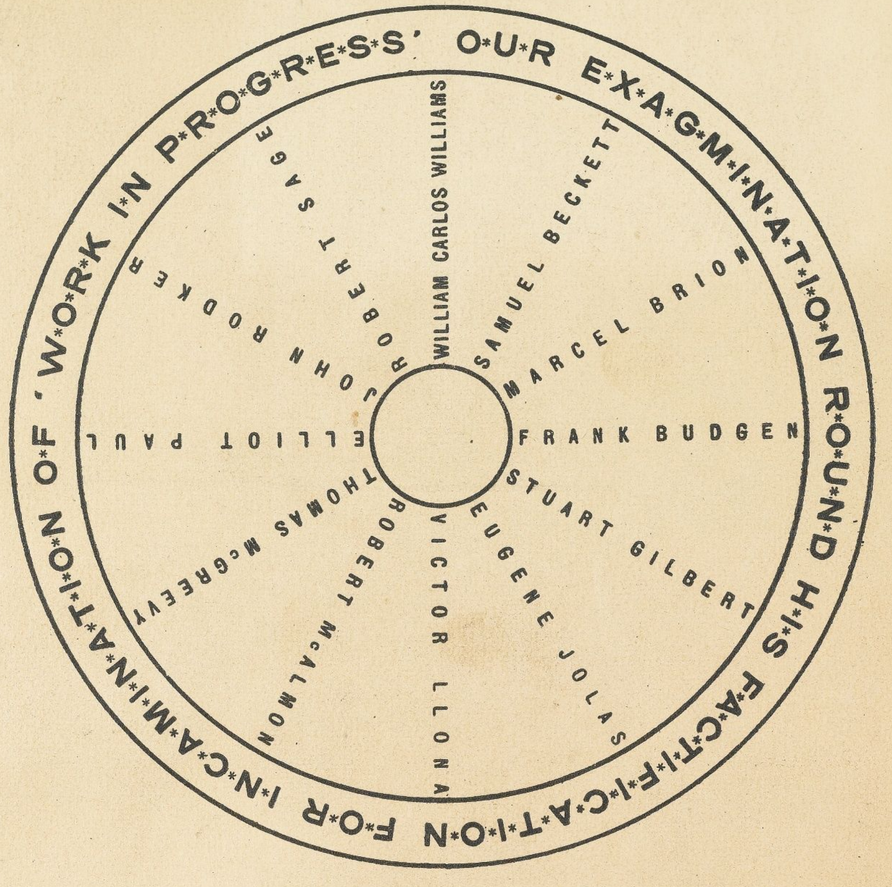
Music Analysis and Synthesis

Joint Mathematics Meetings
January 8, 2025

John Leo (Halfaya Research)
https://www.halfaya.org



SHAKESPEARE AND COMPANY

12, RUE DE L'ODÉON, PARIS

M CM XX IX

SIGMAA Special Session on Mathematics and the Arts, I

Wednesday, January 8, 2025

9:00 AM - 12:00 PM

610 (Level Six, Seattle Convention Center Arch at 705 Pike)

9:00 AM

Music Analysis and Synthesis

John Leo, *Halfaya Research, Bellevue, WA*

610 (Level Six, Seattle Convention Center Arch at 705 Pike)

9:30 AM

Mathematical Modeling of Music: Parameterized Motifs and Hierarchical Structures in Composition

Jeff Flaster, *Melodic Music LLC*

610 (Level Six, Seattle Convention Center Arch at 705 Pike)

History

JMM 2024 (San Francisco)

- Chinese Remainder Theorem

JMM 2025 (Seattle)

- Music Analysis and Synthesis

JMM 2026 (Washington, DC) [planned]

- Coinduction [fiction + art, with Amy Zhu)

Outline

1. Past work:
   - Music Tools
   - Analysis and Synthesis of Counterpoint
2. Recent work in progress:
   - Lean and proofs
   - Equivalence

Music Tools

Brief History

- Originally written in Haskell and then Agda / Cubical Agda.

  Agda is written in Haskell and has a Haskell FFI (Foreign Function Interface), which gives access to high quality libraries for MIDI, (Music)XML, and interfaces to SMT (Satisfiability Modulo Theories) solvers.

- Currently porting to Lean 4.

  Lean 4 is written in Lean 4 (bootstrapped by C++), but the FFI is still unstable and requires a C interface.

  Allows use of Mathlib, an extremely powerful mathematics library.

Music Tools

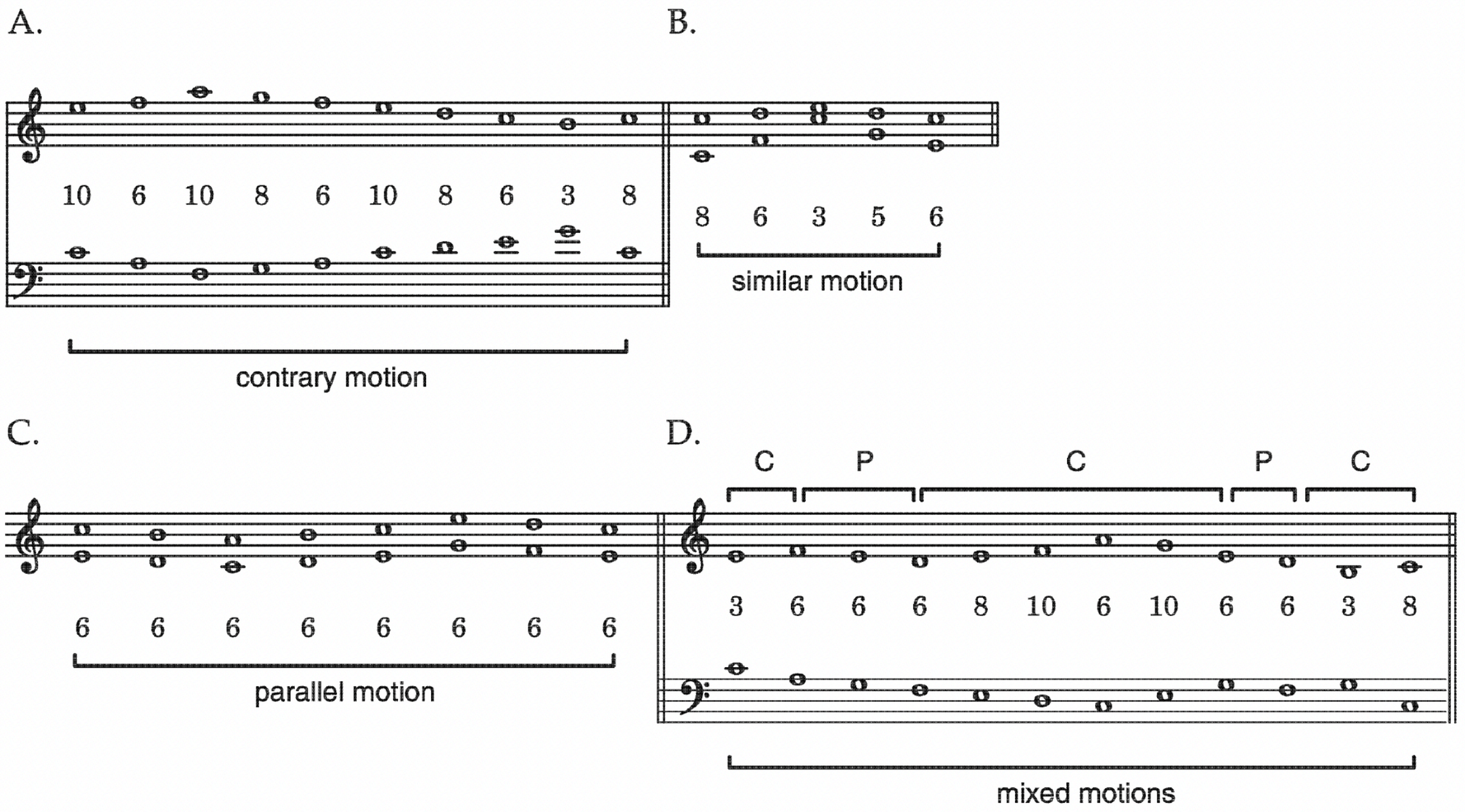
Functionality

- Small, composable tools for representation, analysis and synthesis of music.
- Reads and writes MusicXML; also outputs MIDI.
- Interfaces with an SMT solver (Z3) for synthesis.
- Previous presentations:
  - *Logical Soundness* [FARM (Functional Art and Music) 2021]
    links at https://www.halfaya.org/music/icfp
    - *Counterpoint Analysis and Synthesis* [FARM 2022]
      https://www.youtube.com/watch?v=znjisTSoKCE

Counterpoint Rules

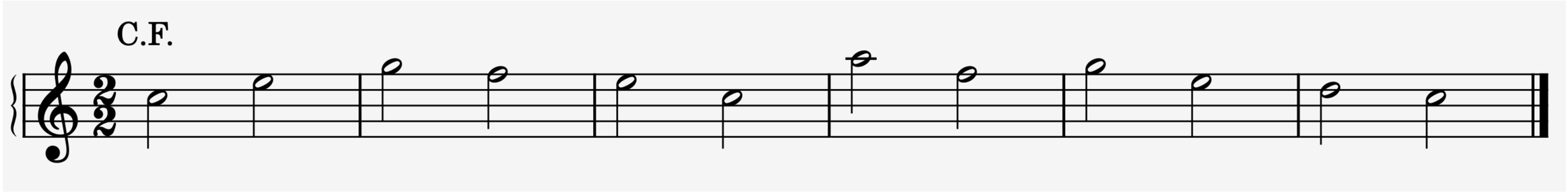
from Laitz: *The Complete Musician* (2012)

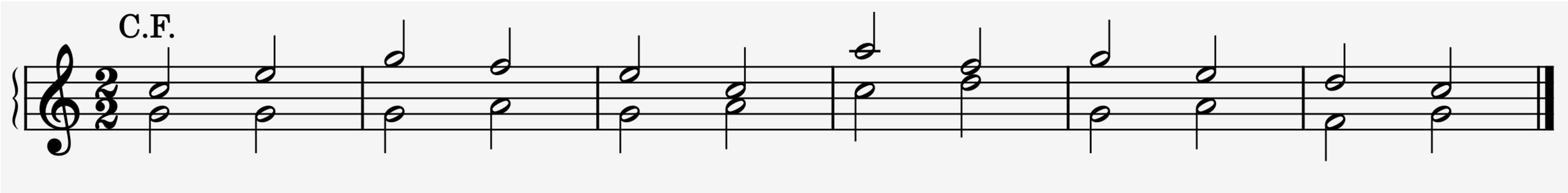
EXAMPLE 2.9 Contrapuntal Motions



Counterpoint Example

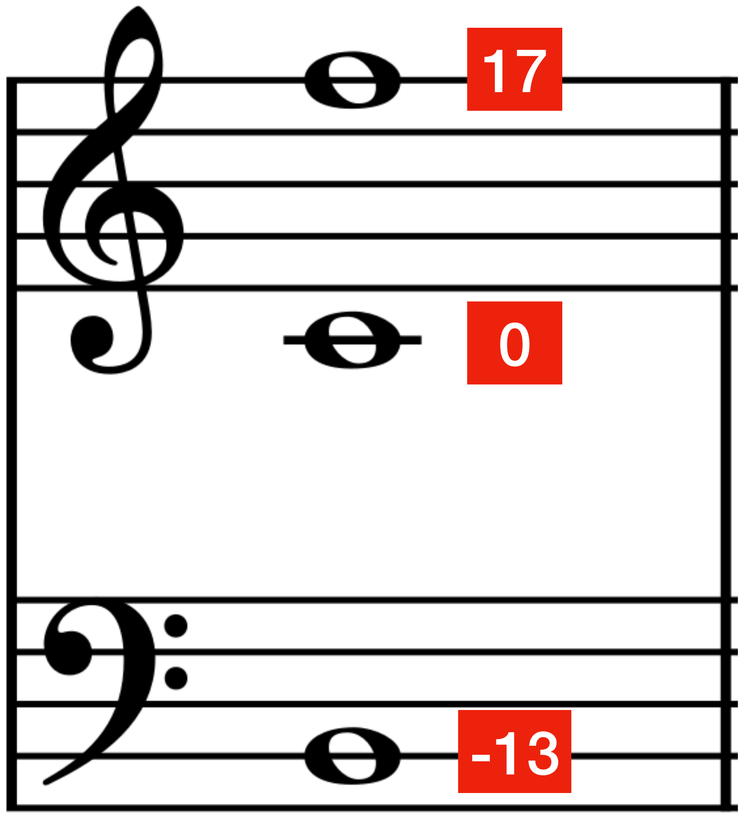
First Species (exercise 146 by Haydn for Beethoven)



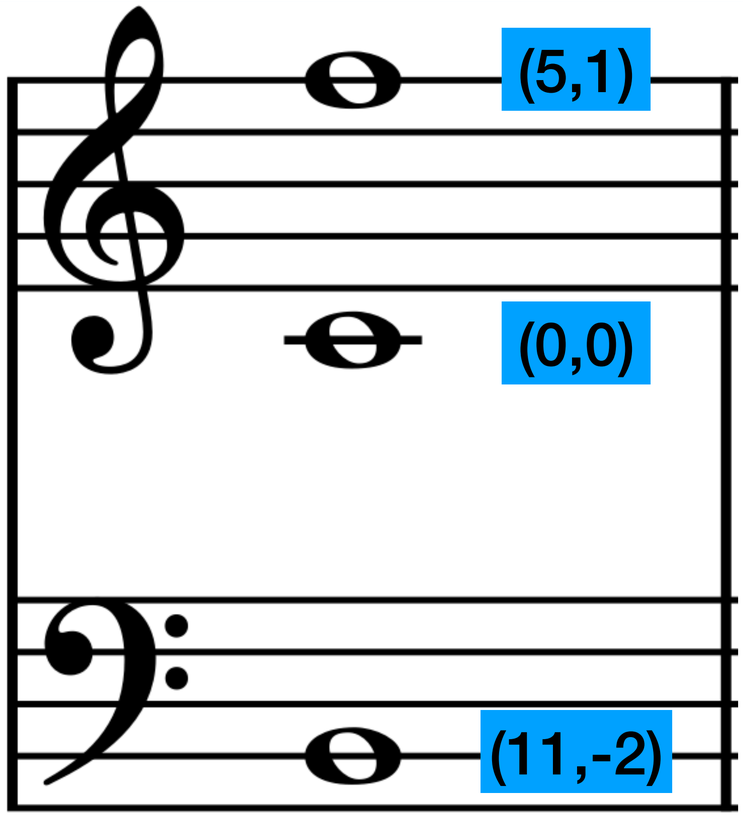


Equivalence in Music

Absolute vs Relative Pitch

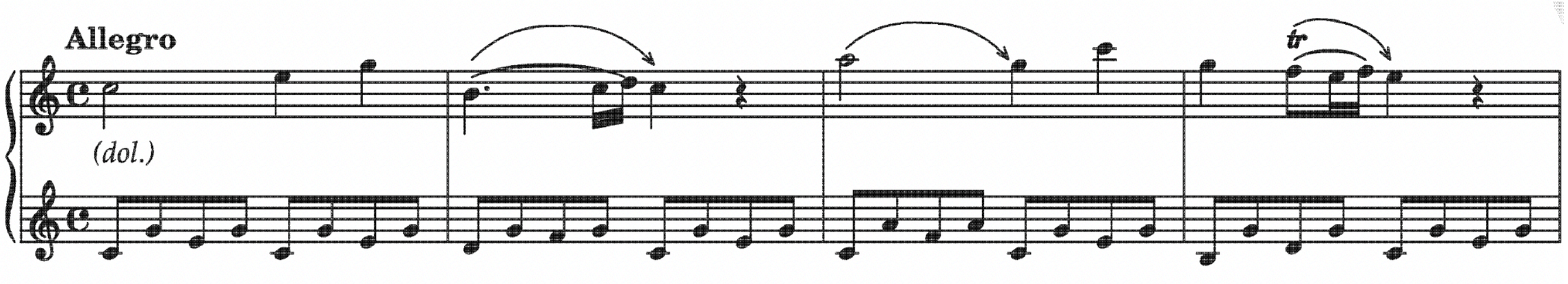


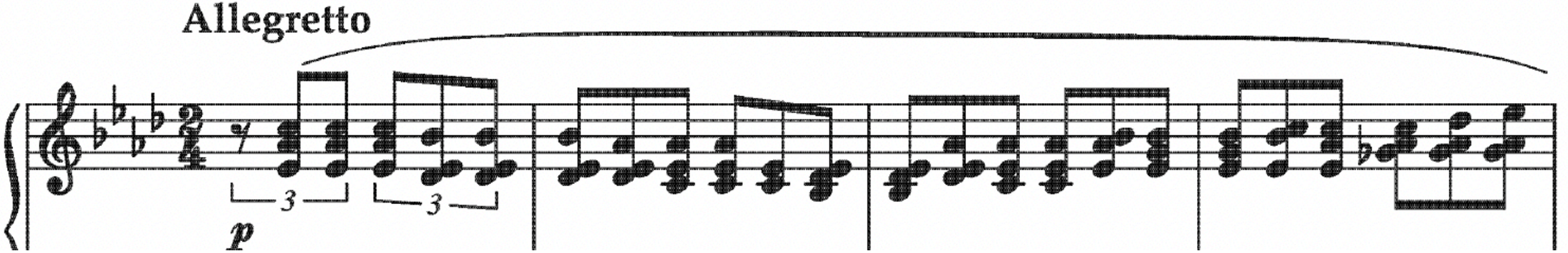
(Pitch within octave, Octave)



Equivalence in Music

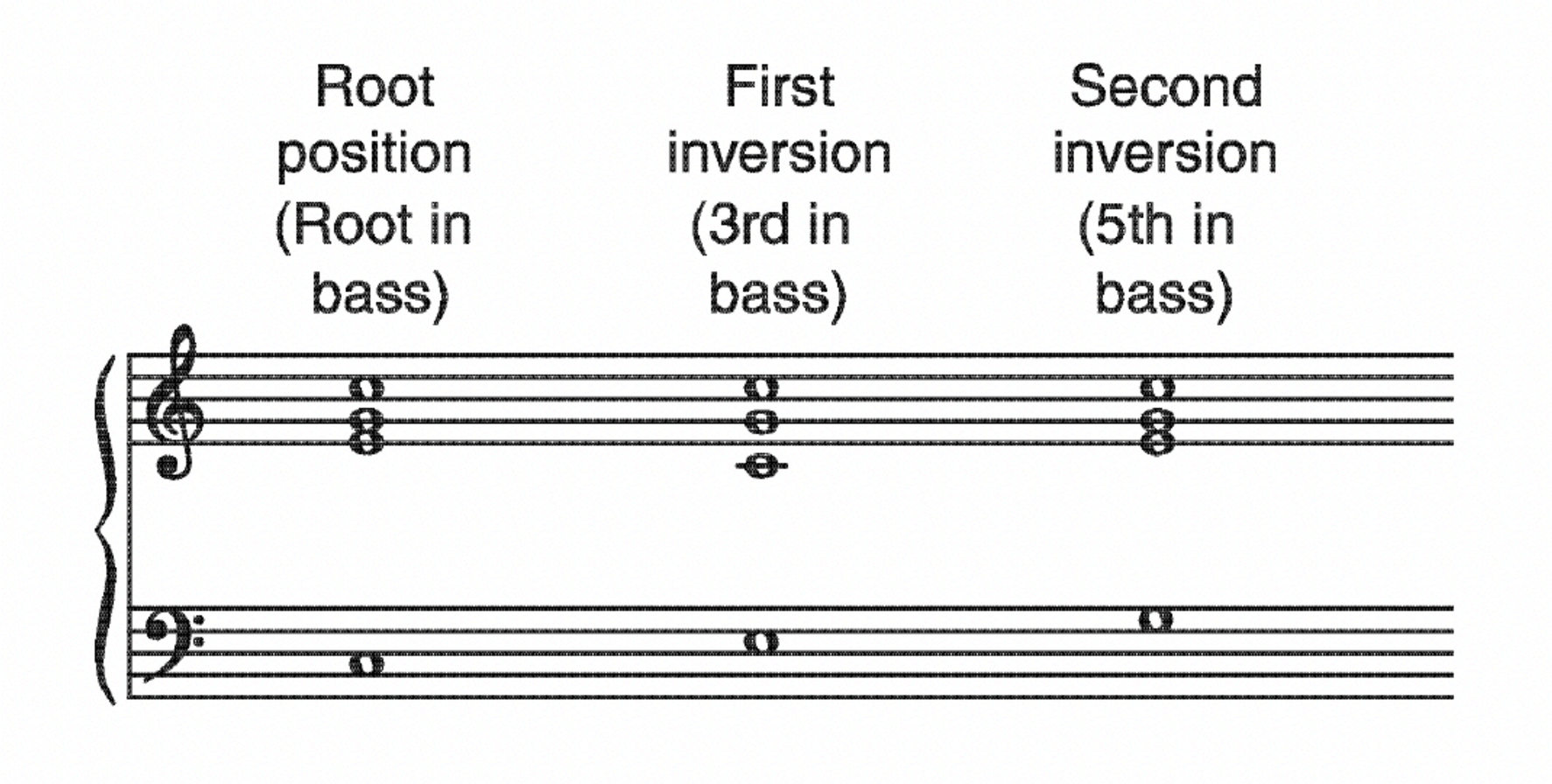
Horizontal vs Vertical





Equivalence in Music

Chord Inversions



Equality and Equivalence in Type Theory

Intensional vs Extensional

- Intensional Type Theory (Lean, Agda, Coq/Rocq etc.)
  Two kinds of equality:
  - Definitional Equality — automatic
  - Propositional Equality — must be proven and manually applied
- Extensional Type Theory

  One kind of equality — might have to be proven but automatically applied
  - Problem: Type checking becomes undecidable

The First Example

```
inductive ℕ : Type where
  | zero : ℕ
  | succ : ℕ → ℕ

def add : ℕ → ℕ → ℕ
  | m, zero   => m
  | m, succ n => succ (add m n)

theorem add_zero (m : ℕ) : m + 0 = m := rfl

theorem zero_add (m : ℕ) : 0 + m = m := by
  induction m with
  | zero => rfl
  | succ m ih => rewrite [add_succ, ih]; rfl
```

Some Recent Work

with connections to the University of Washington

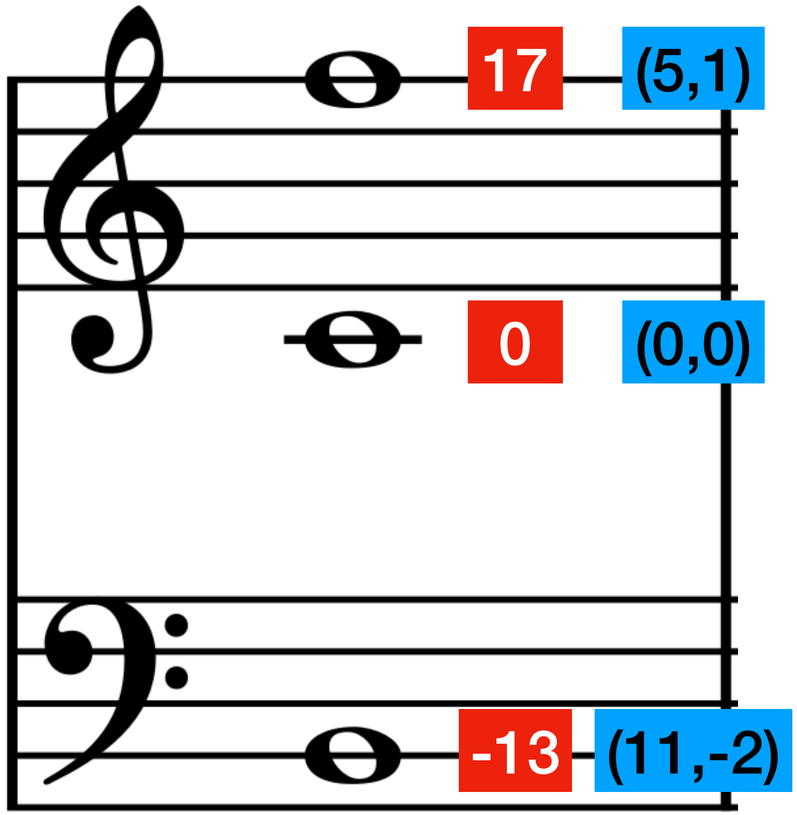
- Talia Ringer, et. al: *Proof Repair Across Type Equivalences* [PLDI 2021]
  https://dependenttyp.es/pdf/repair.pdf

- Anjali Pai: *Working with Equivalent Definitions in Rocq*
  https://uwplse.org/2024/08/19/Equivalent-Rocq.html

- Terry Tao, et. al: *Equational Theories* [Sept-Oct 2024]
  https://github.com/teorth/equational_theories
  makes use of *egg (e-graphs good)* [https://egraphs-good.github.io/]

Equivalence in Music

Absolute vs Relative Pitch




```
abbrev Pitch  := Int
abbrev Octave := Int
def PC := {n : Int // 0 ≤ n ∧ n < 12}
def PCOctave : Type := PC × Octave

def absoluteToRelative (p : Pitch) : PCOctave :=
  ⟨⟨p % 12,
     And.intro (Int.emod_nonneg p (by simp))
               (Int.emod_lt_of_pos p (by simp))⟩,
   p / 12⟩

def relativeToAbsolute : PCOctave → Pitch
  | ⟨p , o⟩ => o * 12 + p.1
```

Equivalence of Absolute and Relative Pitch

One direction

```
theorem PitchToPCOctaveToPitch (p: Pitch) :
     relativeToAbsolute (absoluteToRelative p) = p :=
   Int.ediv_add_emod' p 12
```

Equivalence of Absolute and Relative Pitch

The other direction

```
theorem PCOctaveToPitchToPCOctave : (pco: PCOctave) →
    absoluteToRelative (relativeToAbsolute pco) = pco := by
  intro pco
  unfold relativeToAbsolute absoluteToRelative;
  rw [← Prod.eta pco]
  apply Prod.ext_iff.2; simp
  constructor

  apply Subtype.ext; simp
  rw [Int.add_emod, Int.mul_emod_left]; simp
  rw [Int.emod_eq_of_lt pco.fst.property.1 pco.fst.property.2]

  rw [Int.add_ediv_of_dvd_left (Int.dvd_mul_left pco.snd 12)]
  rw [Int.mul_ediv_cancel pco.snd (by simp)]
  rw [Int.ediv_eq_zero_of_lt pco.fst.property.1 pco.fst.property.2]
  simp
```

```
def PC := {n : Int // 0 ≤ n ∧ n < 12}
```

Using Equivalence

Given

$$\varphi : A \cong C$$
$$\psi : B \cong D$$

and a function

$f : A \to B$, derive

$g : C \to D$ as

$$g = \psi \circ f \circ \varphi^{-1}$$

Using Equivalence in Lean

Automatic coercion

```
instance : Coe Pitch PCOctave where
  coe := absoluteToRelative

instance : Coe PCOctave Pitch where
  coe := relativeToAbsolute

def pitchClass : PCOctave → PC
  | (p, _) => p

def f5 : Pitch := 17

#eval pitchClass f5
—— 5
```

Future Work

- Finish Lean translation
- Port foreign function interfaces to SMT and MusicXML
- Continue work with equivalences and proofs
- Expand functionality to further analysis and synthesis

Tlön, Uqbar, Orbis Tertius

Jorge Luis Borges, *Ficciones* (1940)

Numismatics, pharmacology and archaeology have been revised. I gather that biology and mathematics are awaiting their avatar. […]

Then, English, French, and mere Spanish will disappear from this planet. The world will be Tlön. I take no notice. I go on revising, in the quiet of the days in the hotel at Androgué, a tentative translation into Spanish, in the style of Quevedo, which I do not intend to see published, of Sir Thomas Browne's *Urn Burial*.