Integration Theory: An Exponential Approach to Harmonic Analysis and Composition

Andrew Simpson

April, 2025

Abstract

This paper introduces Integration Theory, a novel mathematical framework for harmonic analysis and composition in Western music theory. By applying a dimensional augmentation operation to traditional diatonic harmony, we transform the conventional one-dimensional approach to harmonization into a comprehensive two-dimensional harmonic space. The resulting structure reveals 49 distinct harmonic contexts for any given melody note, compared to the traditional seven modal options. We present formal mathematical definitions of the Integration Operation, prove its completeness and exclusivity properties, and demonstrate its practical applications in composition. Empirical validation through compositional implementation confirms the theory's musical validity and creative utility.

1. Introduction

Traditional Western music theory approaches harmonization primarily through a one-dimensional framework, offering seven modal options (Ionian, Dorian, Phrygian, Lydian, Mixolydian, Aeolian, and Locrian) for any given melody note within a key (Ellard & Johnstone, 2018). While this approach has served composers for centuries, it represents a limited subset of the available harmonic possibilities within diatonic music.

This paper introduces Integration Theory, which extends this one-dimensional approach into a two-dimensional harmonic space through a mathematical operation analogous to calculus integration—transforming a linear input into an area of possibilities. The resulting structure provides precisely 49 distinct harmonic contexts for any given melody note, each musically valid within diatonic theory.

2. Theoretical Framework

2.1 Definitions

Let us define the following terms:

1. **Melody Note (N)**: *Any note in the chromatic scale C, C♯/D♭, D, D♯/E♭, E, F, F♯/G♭, G, G♯/A♭, A, A♯/B♭, B.*
2. **Diatonic Scale**: *A seven-note scale constructed using a specific pattern of whole and half steps. The major scale follows the pattern W-W-H-W-W-W-H, where W represents a whole step and H represents a half step.*
3. **Mode**: *A scale derived from a parent scale by changing the starting note while maintaining the same set of notes. For a major scale, we define seven modes:*
    - *Mode 1 (Ionian): Starting on the 1st scale degree*
    - *Mode 2 (Dorian): Starting on the 2nd scale degree*
    - *Mode 3 (Phrygian): Starting on the 3rd scale degree*
    - *Mode 4 (Lydian): Starting on the 4th scale degree*
    - *Mode 5 (Mixolydian): Starting on the 5th scale degree*
    - *Mode 6 (Aeolian): Starting on the 6th scale degree*
    - *Mode 7 (Locrian): Starting on the 7th scale degree*
4. **Key Squared (K^2)**: *A specific collection of keys, ordered so that their linear sequence itself resembles a key, derived from a melody note through the Integration Operation, which serves as the foundation for generating the two-dimensional harmonic space.*

2.2 Dimensional Augmentation in Harmonic Theory

In traditional music theory, the harmonic options for a melody note N are represented by the set of modes that contain N. This can be visualized as a one-dimensional array of seven options.

Integration Theory augments this one-dimensional approach to a two-dimensional matrix by:

1. Identifying a specific key related to the melody note (Key Squared)
2. Generating a 7×7 matrix where each cell represents a distinct harmonic context containing the melody note

The transformation from 1D to 2D harmonic space can be formally expressed as a dimensional augmentation operation, analogous to the mathematical process of integration in calculus, where a one-dimensional function is transformed into a two-dimensional area (Larson & Edwards, 2010).

3. The Integration Operation

3.1 The Integration Theorem

Theorem 1 (Integration): *For any melody note N in the chromatic scale, the complete set of diatonic harmonic contexts containing N can be generated through the following operation:*

1. *Define the Key Squared Ionian, $K^2(Ion)$, as the note located at interval -4 semitones (descending major 3rd) from N.*
2. *Define the Primary Scale S_1 as the major scale with $K^2(Ion)$ as its tonic: $S_1 = \{K^2(Ion), K^2(Ion)+2, K^2(Ion)+4, K^2(Ion)+5, K^2(Ion)+7, K^2(Ion)+9, K^2(Ion)+11\}$*

3. *For each note $n \in S_1$, define a Secondary Scale S(n) as the major scale with n as its tonic.*
4. *The complete Integration of N, denoted $\int K(N)dN$, is the 7×7 matrix where:*
    - *Row 1 contains the notes of S_1*
    - *Column 1 contains the notes of S_1*
    - *Cell (i,j) contains the (i-1)th mode of the scale $S(S_1[j])$*

This operation can be expressed algorithmically as:

```
Algorithm 1: The Integration Operation
   Input: Melody note N
   Output: 7×7 matrix of harmonic options ∫K(N)dN

   1. K²(Ion) ← N - 4 (semitones)
   2. S₁ ← MajorScale(K²(Ion))
   3. Initialize 7×7 matrix ∫K(N)dN
   4. For j from 1 to 7:
        a. S_j ← MajorScale(S₁[j])
        b. For i from 1 to 7:
                  i. ∫K(N)dN[i,j] ← Mode(S_j, i)
   5. Return ∫K(N)dN
```

3.2 The Harmonic Exclusivity Theorem

Theorem 2 (Harmonic Exclusivity): *For any melody note N, the set of harmonic contexts generated by the Integration Operation $\int K(N)dN$ constitutes exactly the set of all and only the diatonic harmonic contexts in which N appears.*

Proof (Constructive)

1. For any note N, $K^2(N)$ = N - 4 semitones, placing N as the 3rd scale degree of the $K^2(N)$ major scale.
2. The K^2 Ionian set generates 7 keys, each with 7 modes, producing 49 harmonic modes.
3. Since N is contained in the $K^2(N)$ major scale of major scales, all 49 modes contain N.
4. Any diatonic context containing N must be one of 49 possible combinations (7 scale positions × 7 modes).

Therefore, $\int K(N)dN$ produces exactly all diatonic keys containing N. By dimensional expansion, $\int K(N)dN$ also produces exactly all diatonic modes containing N.

Proof (Exclusionary)

1. Of the 12 possible key centers, exactly 7 contain a given note N diatonically.
2. These 7 keys correspond to N appearing as each of the 7 possible scale degrees.
3. The Integration Operation $K^2(N)$ = N - 4 semitones generates a K^2 Ionian set of 7 notes.
4. These 7 notes, as key centers, produce exactly the 7 keys containing N.
5. The remaining 5 key centers do not contain N diatonically.

Therefore, ∫K(N)dN contains all and only the diatonic keys with N. By dimensional expansion, ∫K(N)dN also contains all and only the diatonic modes with N.

3.3 Example: Integration of C

To illustrate the Integration Operation, we apply it to the melody note C:

1. $K^{♭♭}$(Ion) = C - 4 semitones = A ♭ ♭
2. S_1 = MajorScale(A ♭) = {A ♭ , B ♭ , C, D ♭ , E ♭ , F, G}
3. Secondary Scales:
    - S(A ♭) = {A ♭ , B ♭ , C, D ♭ , E ♭ , F, G}
    - S(B ♭) = {B ♭ , C, D, E ♭ , F, G, A}
    - S(C) = {C, D, E, F, G, A, B}
    - S(D ♭) = {D ♭ , E ♭ , F, G ♭ , A ♭ , B ♭ , C}
    - S(E ♭) = {E ♭ , F, G, A ♭ , B ♭ , C, D}
    - S(F) = {F, G, A, B ♭ , C, D, E}
    - S(G) = {G, A, B, C, D, E, F♯}
4. The resulting 7×7 matrix ∫K(C)dC = A ♭ ♭ contains all modes of these scales, organized with keys as rows and modes as columns.

This generates exactly 49 distinct harmonic contexts, each containing the note C in a unique modal interval.

3.4 Integration of Multiple Notes: The Common Keys Approach

While the Integration Operation is powerful when applied to single melody notes, its utility expands considerably when handling multiple notes simultaneously. This section explores how Integration Theory extends to multiple-note inputs through the Common Keys approach.

3.4.1 Theoretical Framework for Multiple Notes

The Common Keys approach identifies diatonic harmonic frameworks that can accommodate multiple melody notes simultaneously. This is achieved by finding the intersection of the K^2 Ionian sets produced by each individual note.

Definition 3.4.1: *For melody notes {N_1, N_2, ..., $N_□$}, the Common Keys (CK) are defined as:*

$CK(\{N_1, N_2, ..., N_□\}) = K^2\_Ion(N_1) \cap K^2\_Ion(N_2) \cap ... \cap K^2\_Ion(N_□)$

Where $K^2\_Ion(N_i)$ represents the set of seven notes in the first row/column of the Integration matrix for note N_i.

This intersection operation produces a subset of keys that contain all input notes in diatonic contexts.

3.4.2 Modal Analysis of Note Groups

Integration Theory extends beyond identifying Common Keys to provide modal analysis of note groups. When a specific modal context is desired, the system maps Common Keys to their respective modes:

Definition 3.4.2: *For Common Keys CK and a modal designation m (where m ∈ {1=Ion, 2=Dor, 3=Phr, 4=Lyd, 5=Mix, 6=Aeo, 7=Loc}), the Modal Analysis MA is defined as:*

MA(CK, m) = {K_m(X) | X ∈ CK}

Where K_m(X) represents key X interpreted in mode m.

This is calculated through two equivalent methods:

1. For each Common Key X, compute its Integration matrix and select the first cell of row m
2. For each Common Key X, identify the mth interval of its major scale

3.4.3 The Common Keys Algorithm

```
Algorithm 2: Common Keys and Modal Analysis
Input:
  - Note groups {NG₁, NG₂, ..., NG□} where each NGᵢ contains one or
more notes
  - Modal designations {m₁, m₂, ..., m□} where each mᵢ ∈
{1,2,3,4,5,6,7}
  - Optional: Key constraint K
Output:
  - Common Keys and Modal Analysis for each note group

1. For each note group NGᵢ:
   a. If K is provided, verify all notes in NGᵢ are within key K
   b. For each note N in NGᵢ:
      i. Compute K²(N) = N - 4 (semitones)
      ii. Generate K²_Ion(N), the K² Ionian set for note N
   c. Compute CKᵢ = ∩(K²_Ion(N)) for all N in NGᵢ
   d. Apply modal analysis: MAᵢ = MA(CKᵢ, mᵢ)
   e. Perform interval analysis for each note relative to each
      modal chord
2. Return {(CK₁, MA₁), (CK₂, MA₂), ..., (CK□, MA□)}
```

3.4.4 Multiple Note Group Processing

Integration Theory can handle multiple note groups simultaneously, with each group requiring its own modal analysis. This powerful capability allows for the analysis of entire melodic phrases or compositions with varying modal contexts.

The system supports:

1. Multiple notes within each group (melodic fragments)
2. Multiple note groups (phrases or sections)

3. Individual modal designations for each group
4. Optional key constraints

This approach produces a comprehensive harmonic analysis that respects both the melodic content and the desired modal progression.

3.4.5 Implementation in Key Squared Generator

The Key Squared Generator (Simpson, 2024a) implements the Common Keys algorithm with several advanced features:

1. Chromatic input support when no key is specified
2. Modal analysis via chord progression specifications
3. Detailed interval analysis showing the function of each note within potential harmonies
4. 'Perfect Table' output summarizing all harmonic options

The unabridged version of the Key Squared Generator offers more functionality, including optional Random Chord Progression (RCP) generation capability and primitive Trimetric Theory analysis (Simpson, 2025a).

3.4.6 Compositional Implications

The Common Keys approach reveals several important properties:

1. Harmonic Constraint: As the number of input notes increases, the number of Common Keys typically decreases, revealing the natural harmonic constraints of the diatonic system.
2. Modal Coloration: Different modal designations produce distinct harmonic colors while maintaining melodic integrity.
3. Functional Insight: Interval analysis reveals the function of each melody note within potential harmonies, informing compositional choices.

3.5 Example: Advanced Integration Analysis

To illustrate the sophisticated capabilities of Integration Theory with multiple inputs and modal analysis, we present the complete analysis of a complex melody input processed through the Key Squared Generator (Simpson, 2024a).

3.5.1 Input Configuration

- Key: Not provided (allowing chromatic inputs)
- Melody: Four note groups:
    - Group 1: G# D# C
    - Group 2: E F
    - Group 3: G A#
    - Group 4: A

- Chord Progression: 3, 7, 4, 1 (Phrygian, Locrian, Lydian, Ionian)

3.5.2 Common Keys Analysis

Note Group 1: G# D# C

- Individual K^2 Ionian sets:
    - K^2(G#): {E, F#, G#, A, B, C#, D#}
    - K^2(D#): {B, C#, D#, E, F#, G#, A#}
    - K^2(C): {G#, A#, C, C#, D#, F, G}
- Common Keys: C#, D#, G# (intersection of all three K^2 Ionian sets)
- Mode Assignment: Phrygian (3rd mode)
- Modal Chords: F, G, C
- Interval Analysis:
    - G#: 3rd of F, 2nd of G, 6th of C
    - D#: 3rd of C, 7th of F, 6th of G
    - C: 5th of F, 1st of C, 4th of G

Note Group 2: E F

- Individual K^2 Ionian sets:
    - K^2(E): {C, D, E, F, G, A, B}
    - K^2(F): {C#, D#, F, F#, G#, A#, C}
- Common Keys: C, F
- Mode Assignment: Locrian (7th mode)
- Modal Chords: B, E
- Interval Analysis:
    - E: 1st of E, 4th of B
    - F: 5th of B, 2nd of E

Note Group 3: G A#

- Individual K^2 Ionian sets:
    - K^2(G): {D#, F, G, G#, A#, C, D}
    - K^2(A#): {F#, G#, A#, B, C#, D#, F}
- Common Keys: A#, D#, F, G#
- Mode Assignment: Lydian (4th mode)
- Modal Chords: D#, G#, A#, C#
- Interval Analysis:
    - G: 3rd of D#, 7th of G#, 6th of A#, 4th of C#
    - A#: 5th of D#, 1st of A#, 2nd of G#, 6th of C#

Note Group 4: A

- K^2(A): {F, G, A, A#, C, D, E}

- Common Keys: A, A#, C, D, E, F, G (all seven notes of F major)
- Mode Assignment: Ionian (1st mode)
- Modal Chords: A, A#, C, D, E, F, G
- Interval Analysis:
    - A: 1st of A, 5th of D, 3rd of F, 7th of A#, 6th of C, 4th of E, 2nd of G

3.5.3 Compositional Interpretation

This analysis demonstrates how Integration Theory handles complex melodic inputs with specific modal requirements:

1. Harmonic Constraints:
    - Note Group 1 (G# D# C) has 3 Common Keys
    - Note Group 2 (E F) has 2 Common Keys
    - Note Group 3 (G A#) has 4 Common Keys
    - Note Group 4 (A) has all 7 Common Keys
2. Modal Mood:
    - The progression from Phrygian to Locrian to Lydian to Ionian is an inverted turnaround and the moods associated with these chord changes are preserved and enhanced by modal Integration.
    - The Integration of a chord progression is in
3. Compositional Flexibility:
    - The composer can select from multiple valid harmonic options for each section
    - Interval analysis provides insight into the function of each note within potential harmonies

This example illustrates the power of Integration Theory to handle complex compositional scenarios with precise computational rigor, providing composers with a comprehensive yet bounded set of harmonic options that honor both melodic content and modal intent.

4. Software Implementation

The Integration Theory framework has been implemented in a suite of Python applications that demonstrate both the theoretical principles and practical applications of the theory.

4.1 Primary Integration Engine: Key Squared Generator

The core implementation (Simpson, 2024a) consists of a Python application that:

1. Accepts user input for melody notes, optional key preferences, and modal chord progressions
2. Performs the Integration Operation as defined in Theorem 1
3. Outputs a comprehensive table of the 49 harmonic contexts
4. Provides analytical tools for exploring the harmonic relationships
5. Offers customizable input options, including optional key and multiple note group entries.

The implementation follows the algorithm defined in Section 3.1, with optimizations for computational efficiency and musical usability.

4.2 Pedagogical Applications

To facilitate understanding and exploration of Integration Theory, three game-based applications have been developed.:

1. **Solitaire Squared**: A card game application that generates unique harmonic progressions based on gameplay, with victory conditions that produce viable compositional frameworks based on Integration Theory (Simpson, 2024b).
2. **Integration: The Siege of Scale World**: A role-playing game where musical challenges are solved using Integration Theory concepts, reinforcing the practical application of the theoretical framework. The objective is to use Integration Theory to form diplomatic coalitions in an effort to rescue hostages abducted by aliens and secure the lands of Scale World (Simpson, 2024c).
3. **Integration: Quest**: A tutorial-style, text-based exploration game application that guides users through the principles of Integration Theory with interactive exercises of Common Keys (Simpson, 2024d).

Each application reinforces different aspects of the theory while providing practical tools for composition and analysis. Links can be found in the References Section.

4.3 Example Computational Output: Integration of C

The following data is produced by the Key Squared Generator. The data displays example output from the following user data:

1. Input Key = C (optional)
2. Melody Note/Note Group = C
3. Chord Progression/Desired Harmonic Output = Ionian

```
Input Information:

Key: C

Melody: C

Chord Progression: 1



Grid for Key = C (Key^2 = G#):

            G#    A#    C     C#    D#    F     G

Ionian      G#    A#    C     C#    D#    F     G

Dorian      A#    C     D     D#    F     G     A

Phrygian    C     D     E     F     G     A     B
```

```
Lydian      C#    D#    F     F#    G#    A#    C

Mixolydian  D#    F     G     G#    A#    C     D

Aeolian     F     G     A     A#    C     D     E

Locrian     G     A     B     C     D     E     F#


Perfect Table:

Melody   Common Keys              Mode      Modal Chords              Interval 1,3,5     Interval
2,4,6,7

-------------------------------------------------------------------------------------------------------
-----------

C        A#, C, C#, D#, F, G, G#  Ionian    A#, C, C#, D#, F, G, G#   C(1:C, 5:F, 3:G#)  C(2:A#, 7:C#,
6:D#, 4:G)
```

The resulting 7×7 matrix $\int K(C)dC = A\flat^{2}$ is characteristic of the Integration Operation. By convention, Columns represent the set of Keys, and the Rows the set of Modes, corresponding to the input data.

The ‘Perfect Table’ provides further organization and analysis of the harmonic output of the Integration Operation.

4.4 Example Computational Output: Integration of Complex Input

The Key Squared Generator was used to generate the following example. The data displays Integrated results from complex melody input, consisting of four comma-separated note groups, further computed according to additional modal input specifications:

1. Input Key = ‘none’
2. Melody Notes/Note Groups = G# D# C, E F, G A#, A
3. Chord Progression/Desired Harmonic Output = Phrygian, Locrian, Lydian, Ionian

```
Enter Key (optional):

Enter Melody: g# d# c, e f, g a#, a

Enter Chord Progression (comma-separated numbers): 3, 7, 4, 1


Input Information:

Key: Not provided

Melody: G# D# C, E F, G A#, A

Chord Progression: 3, 7, 4, 1
```

Common Keys Analysis:

Note Group 1:

G# E F# G# A B C# D#

D# B C# D# E F# G# A#

C G# A# C C# D# F G

Common Keys: C# D# G#

Note Group 2:

E C D E F G A B

F C# D# F F# G# A# C

Common Keys: C F

Note Group 3:

G D# F G G# A# C D

A# F# G# A# B C# D# F

Common Keys: A# D# F G#

Note Group 4:

A F G A A# C D E

Common Keys: A A# C D E F G

Perfect Table:

Melody	Common Keys	Mode	Modal Chords	Interval 1,3,5	Interval 2,4,6,7
G# D# C	C#, D#, G#	Phrygian	F, G, C	G#(3:F)	G#(2:G, 6:C)
				D#(3:C)	D#(7:F, 6:G)

				C(5:F, 1:C)	C(4:G)
E F	C, F	Locrian	B, E	E(1:E) F(5:B)	E(4:B) F(2:E)
G A#	A#, D#, F, G#	Lydian	D#, G#, A#, C#	G(3:D#) A#(5:D#, 1:A#)	G(7:G#, 6:A#, 4:C#) A#(2:G#, 6:C#)
A	A, A#, C, D, E, F, G	Ionian	A, A#, C, D, E, F, G	A(1:A, 5:D, 3:F)	A(7:A#, 6:C, 4:E, 2:G)

The Modal Chords column in the Perfect Table displays the total set of possible diatonic modes which contain all notes in each note group. Moreover, the Interval analysis columns offer detailed interval location information for each note in the corresponding note group.

The results of this example show the complex computational power of the Key Squared Generator to handle multiple groups of melody notes and modal parameters.

4.5 Operational Mechanics

The Key Squared Generator processes harmonic analysis differently depending on user inputs, illustrating Integration Theory's application under various constraints:

1. **Key-Determined Integration**: When a key is specified, the system generates the complete Key² grid for that key and constraints analysis within this framework:
    - When the melody note equals the key note, all seven Key² Ionian options remain valid, providing maximum harmonic flexibility
    - When the melody note differs from the key note, the harmonic options are constrained by the interval relationship between them, reducing the available Common Keys to those reflecting this relationship
2. **Unconstrained Integration**: When no key is specified, the system performs direct Integration on each melody note independently:
    - The K² Ionian row is generated for each melody note
    - Common Keys are determined through intersection analysis of these rows
    - This approach allows for more extensive harmonic exploration unbound by key constraints

This operational structure creates a versatile analytical framework that accommodates both key-constrained composition and free-form harmonic exploration. When analyzing multiple note groups, the system performs independent Integration operations for each group, then assesses modal compatibility based on the specified chord progression, maintaining coherence across the complete harmonic analysis.

4.6 Example: Computation of Key-Determined and Unconstrained Integration

The following data is produced by the Key Squared Generator. The data compares the output results from three trial runs of various input parameters:

1. When melody note is equivalent to the input key
2. When melody note is not the input key, but is found within it
3. When no input key is entered

1. Melody note is equivalent to the input key:

```
Enter Key (optional): c
Enter Melody: c
Enter Chord Progression (comma-separated numbers): 1

Grid for Key = C (Key^2 = G#):

             G#    A#    C     C#    D#    F     G

Ionian       G#    A#    C     C#    D#    F     G
Dorian       A#    C     D     D#    F     G     A
Phrygian     C     D     E     F     G     A     B
Lydian       C#    D#    F     F#    G#    A#    C
Mixolydian   D#    F     G     G#    A#    C     D
Aeolian      F     G     A     A#    C     D     E
Locrian      G     A     B     C     D     E     F#

Input Information:
Key: C
Melody: C
Chord Progression: 1

Perfect Table:
Melody   Common Keys               Mode     Modal Chords              Interval 1,3,5      Interval
2,4,6,7
-------------------------------------------------------------------------------------------------------
-----------
C        A#, C, C#, D#, F, G, G#   Ionian   A#, C, C#, D#, F, G, G#   C(1:C, 5:F, 3:G#)   C(2:A#, 7:C#,
6:D#, 4:G)
```

2. Melody note is not the input key, but is found within it:

```
Enter Key (optional): a#
Enter Melody: c
Enter Chord Progression (comma-separated numbers): 1

Grid for Key = A# (Key^2 = F#):

             F#    G#    A#    B     C#    D#    F

Ionian       F#    G#    A#    B     C#    D#    F
```

```
Dorian       G#    A#    C     C#    D#    F     G
Phrygian     A#    C     D     D#    F     G     A
Lydian       B     C#    D#    E     F#    G#    A#
Mixolydian   C#    D#    F     F#    G#    A#    C
Aeolian      D#    F     G     G#    A#    C     D
Locrian      F     G     A     A#    C     D     E

Input Information:
Key: A#
Melody: C
Chord Progression: 1

Perfect Table:
Melody   Common Keys          Mode       Modal Chords         Interval 1,3,5         Interval 2,4,6,7
-------------------------------------------------------------------------------------------------------
---
C        A#, C#, D#, F, G#    Ionian     A#, C#, D#, F, G#    C(5:F, 3:G#)           C(2:A#, 7:C#,
6:D#)
```

3. No input key is entered:

```
Enter Key (optional):
Enter Melody: c
Enter Chord Progression (comma-separated numbers): 1

Input Information:
Key: Not provided
Melody: C
Chord Progression: 1

Common Keys Analysis:

Note Group 1:
C       G# A# C C# D# F G
Common Keys: A# C C# D# F G G#

Perfect Table:
Melody   Common Keys               Mode      Modal Chords              Interval 1,3,5     Interval
2,4,6,7
-------------------------------------------------------------------------------------------------------
-----------
C        A#, C, C#, D#, F, G, G#   Ionian    A#, C, C#, D#, F, G, G#   C(1:C, 5:F, 3:G#)  C(2:A#, 7:C#,
6:D#, 4:G)
```

If a key is entered by the user, the Key Squared Generator produces the Integrated grid from the entered key, and all melody notes are compared according to their locations on the grid. Therefore, if a key is entered, all melody notes must conform to the key.

To summarize, this example illustrates how parameterization is handled within Integration:

1. If a key is specified, in the case where the melody note is equivalent to the specified key note, provided no other notes are in the note group, then all common key possibilities are valid.
2. If a key is specified, where the melody note is not equivalent to the key note, then the particular interval of the melody note within the key, and further comparison with all other note group notes (if specified), will determine the number of common keys.

3. If a key is not specified, then the grid is not displayed in full; instead, the first row or column, Key² Ionian scale, of each melody note is compared within each note group to determine the number of common keys.

5. Validation through Composition

The theoretical framework has been empirically validated through extensive compositional application:

1. **Integration (Album, 2024)**: A complete album of unique compositions has been created using direct application of Integration Theory, demonstrating its practical utility in generating harmonically rich and musically satisfying compositions (Simpson, 2024e).
2. **Solitaire Squared (Album, 2025)**: A second collection of compositions has been developed, each derived from the output of the Integration Solitaire game application, further validating the theory's applicability across different compositional approaches (Simpson, 2025b).

All composition, music, performance, recording, editing, and producing was facilitated by the author, under the auspices of a solo music project titled, A Micron.

Each composition of Integration (Album) is accompanied by data from the Key Squared Generator which was used in composition. Each composition of Solitaire Squared (Album) is accompanied by victory card data from the Solitaire Squared card game application, which was used in composition (Simpson, 2025c).

Each composition serves as an empirical test case, confirming that the theoretical harmonic contexts produced by the Integration Operation yield musically coherent and arguably aesthetically pleasing results.

6. Future Research Directions

The mathematical properties of Integration Theory suggest several promising avenues for further investigation:

1. **Cyclic Integration Structure**: The behavior of repeated integration operations follows a pattern resembling rotation in the complex plane: if we denote a note N as $e^{i\theta}$, then Integration transforms it to $K^2 = e^{i(\theta+2\pi/3)}$, and a second integration yields $K^3 = e^{i(\theta+4\pi/3)}$, with a third integration returning to the original position. This cyclic structure ($N \rightarrow K^2 \rightarrow K^3 \rightarrow N$) suggests a fundamental three-phase symmetry in tonal space that transcends traditional boundaries between dimensions. The employment of coordinate transformations may aid in this analysis.
2. **Unified Field Theory**: The connection between Integration Theory and Trimetric Theory (Simpson, 2025a) points toward the possibility of a unified theoretical framework encompassing both diatonic and chromatic approaches to harmony. Integration Theory operates primarily in diatonic space, while Trimetric Theory extends into chromatic territory, suggesting complementary aspects of a single underlying system. The implementation of Integration and Trimetric Theories to traditional altered minor scales, particularly the melodic ascending minor and the harmonic minor scales, offers promising avenues for further examination in this regard.

3. **Dimensional Transformations**: Each integration operation appears to transform the dimensional properties of the music theory space: from the 0-dimensional note (as a point in 1-dimensional key space) to the 2-dimensional Key Squared to what may be a 3-dimensional structure at K^3, before collapsing back to the origin. This suggests a möbius-like topology to the theoretical space that warrants further exploration. Preliminary research indicates this third dimension could consist of the chromatic scale starting at the note which results from the differentiation of the 0-dimensional note; however, more research is needed.

These research directions could extend the theoretical foundation established in this paper while revealing fundamental organizational principles in music theory that may have broader implications for our understanding of harmonic relationships.

7. Conclusion

Integration Theory represents a significant advancement in the mathematical understanding of diatonic harmony by transforming the conventional one-dimensional approach into a comprehensive two-dimensional framework. Through the Integration Operation, we have demonstrated that for any melody note, exactly 49 distinct harmonic contexts exist within diatonic theory.

The formal mathematical properties of the Integration Operation, particularly the Harmonic Exclusivity Theorem, provide a rigorous foundation for this approach. The practical implementation through software applications and compositional validation confirms the theory's utility in both analytical and creative contexts.

This dimensional augmentation of harmonic theory opens new possibilities for composition, analysis, and music education, offering a more complete understanding of the harmonic relationships inherent in Western music theory.

References

Ellard, G., & Johnstone, S. (2018). Music theory and composition: A practical approach. Rowman & Littlefield.

Larson, R., & Edwards, B. (2010). Calculus (9th ed.). Brooks/Cole Cengage Learning.

Simpson, A. (2025a). Trimetric Theory: Equilateral Tonal Symmetry in Quaternary Chromatic Space. https://www.amicronmusic.com/trimetry-thesis

Simpson, A. (2025b). Solitaire Squared [Album, in development]. A Micron Music. https://www.amicronmusic.com/

Simpson, A. (2025c). Codeouts [pdf]. A Micron Music. https://www.amicronmusic.com/codeouts

Simpson, A. (2024a). Key squared generator [Software application]. A Micron Music. https://www.amicronmusic.com/key-squared-generator/

Simpson, A. (2024b). Solitaire Squared [Software application]. A Micron Music. https://www.amicronmusic.com/solitaire-squared/

Simpson, A. (2024c). Integration: The Siege of Scale World [Software application]. A Micron Music. https://amicronmusic.com/key-squared-rpg/

Simpson, A. (2024d). Integration: Quest [Software application]. A Micron Music. https://amicronmusic.com/integration-quest/

Simpson, A. (2024e). Integration [Album]. A Micron Music. https://www.amicronmusic.com/product/integration/

Appendices

Appendix A: Complete Integration Tables for All 12 Chromatic Notes

Grid for Key = C (Key^2 = G#):

	G#	A#	C	C#	D#	F	G
Ionian	G#	A#	C	C#	D#	F	G
Dorian	A#	C	D	D#	F	G	A
Phrygian	C	D	E	F	G	A	B
Lydian	C#	D#	F	F#	G#	A#	C
Mixolydian	D#	F	G	G#	A#	C	D
Aeolian	F	G	A	A#	C	D	E
Locrian	G	A	B	C	D	E	F#

Grid for Key = C# (Key^2 = A):

	A	B	C#	D	E	F#	G#
Ionian	A	B	C#	D	E	F#	G#
Dorian	B	C#	D#	E	F#	G#	A#
Phrygian	C#	D#	F	F#	G#	A#	C
Lydian	D	E	F#	G	A	B	C#

Mixolydian	E	F#	G#	A	B	C#	D#
Aeolian	F#	G#	A#	B	C#	D#	F
Locrian	G#	A#	C	C#	D#	F	G

Grid for Key = D (Key^2 = A#):

	A#	C	D	D#	F	G	A
Ionian	A#	C	D	D#	F	G	A
Dorian	C	D	E	F	G	A	B
Phrygian	D	E	F#	G	A	B	C#
Lydian	D#	F	G	G#	A#	C	D
Mixolydian	F	G	A	A#	C	D	E
Aeolian	G	A	B	C	D	E	F#
Locrian	A	B	C#	D	E	F#	G#

Grid for Key = D# (Key^2 = B):

	B	C#	D#	E	F#	G#	A#
Ionian	B	C#	D#	E	F#	G#	A#
Dorian	C#	D#	F	F#	G#	A#	C
Phrygian	D#	F	G	G#	A#	C	D
Lydian	E	F#	G#	A	B	C#	D#
Mixolydian	F#	G#	A#	B	C#	D#	F
Aeolian	G#	A#	C	C#	D#	F	G
Locrian	A#	C	D	D#	F	G	A

Grid for Key = E (Key^2 = C):

	C	D	E	F	G	A	B
Ionian	C	D	E	F	G	A	B

Dorian	D	E	F#	G	A	B	C#
Phrygian	E	F#	G#	A	B	C#	D#
Lydian	F	G	A	A#	C	D	E
Mixolydian	G	A	B	C	D	E	F#
Aeolian	A	B	C#	D	E	F#	G#
Locrian	B	C#	D#	E	F#	G#	A#

Grid for Key = F (Key^2 = C#):

	C#	D#	F	F#	G#	A#	C
Ionian	C#	D#	F	F#	G#	A#	C
Dorian	D#	F	G	G#	A#	C	D
Phrygian	F	G	A	A#	C	D	E
Lydian	F#	G#	A#	B	C#	D#	F
Mixolydian	G#	A#	C	C#	D#	F	G
Aeolian	A#	C	D	D#	F	G	A
Locrian	C	D	E	F	G	A	B

Grid for Key = F# (Key^2 = D):

	D	E	F#	G	A	B	C#
Ionian	D	E	F#	G	A	B	C#
Dorian	E	F#	G#	A	B	C#	D#
Phrygian	F#	G#	A#	B	C#	D#	F
Lydian	G	A	B	C	D	E	F#
Mixolydian	A	B	C#	D	E	F#	G#
Aeolian	B	C#	D#	E	F#	G#	A#
Locrian	C#	D#	F	F#	G#	A#	C

Grid for Key = G (Key^2 = D#):

	D#	F	G	G#	A#	C	D
Ionian	D#	F	G	G#	A#	C	D
Dorian	F	G	A	A#	C	D	E
Phrygian	G	A	B	C	D	E	F#
Lydian	G#	A#	C	C#	D#	F	G
Mixolydian	A#	C	D	D#	F	G	A
Aeolian	C	D	E	F	G	A	B
Locrian	D	E	F#	G	A	B	C#

Grid for Key = G# (Key^2 = E):

	E	F#	G#	A	B	C#	D#
Ionian	E	F#	G#	A	B	C#	D#
Dorian	F#	G#	A#	B	C#	D#	F
Phrygian	G#	A#	C	C#	D#	F	G
Lydian	A	B	C#	D	E	F#	G#
Mixolydian	B	C#	D#	E	F#	G#	A#
Aeolian	C#	D#	F	F#	G#	A#	C
Locrian	D#	F	G	G#	A#	C	D

Grid for Key = A (Key^2 = F):

	F	G	A	A#	C	D	E
Ionian	F	G	A	A#	C	D	E
Dorian	G	A	B	C	D	E	F#
Phrygian	A	B	C#	D	E	F#	G#
Lydian	A#	C	D	D#	F	G	A
Mixolydian	C	D	E	F	G	A	B

Aeolian	D	E	F#	G	A	B	C#
Locrian	E	F#	G#	A	B	C#	D#

Grid for Key = A# (Key^2 = F#):

	F#	G#	A#	B	C#	D#	F
Ionian	F#	G#	A#	B	C#	D#	F
Dorian	G#	A#	C	C#	D#	F	G
Phrygian	A#	C	D	D#	F	G	A
Lydian	B	C#	D#	E	F#	G#	A#
Mixolydian	C#	D#	F	F#	G#	A#	C
Aeolian	D#	F	G	G#	A#	C	D
Locrian	F	G	A	A#	C	D	E

Grid for Key = B (Key^2 = G):

	G	A	B	C	D	E	F#
Ionian	G	A	B	C	D	E	F#
Dorian	A	B	C#	D	E	F#	G#
Phrygian	B	C#	D#	E	F#	G#	A#
Lydian	C	D	E	F	G	A	B
Mixolydian	D	E	F#	G	A	B	C#
Aeolian	E	F#	G#	A	B	C#	D#
Locrian	F#	G#	A#	B	C#	D#	F

Appendix B: Technical Specifications of Software Implementations

Data Structures

- **Chromatic Scale**: Array of 12 semitones (['C', 'C#', 'D', 'D#', 'E', 'F', 'F#', 'G', 'G#', 'A', 'A#', 'B'])
- **Note Groups**: Nested arrays parsed from comma-separated user input

- **Modal Scales**: Dictionary mapping modes to interval patterns
- **Trimetric Scale Patterns**: Three symmetrical 9-note patterns:
    - [1, 2, 1, 1, 2, 1, 1, 2, 1] (ΔXYZ(0))
    - [2, 1, 1, 2, 1, 1, 2, 1, 1] (ΔXYZ(1))
    - [1, 1, 2, 1, 1, 2, 1, 1, 2] (ΔXYZ(2))

Core Algorithms

1. Key² Relationship

```
def find_key2(key):

    notes = generate_chromatic_scale()

    start = notes.index(key)

    return notes[(start - 4) % 12]
```

2. Common Keys Identification

```
def find_common_keys(key, note_group):

    if not key:

        # Find intersection of possible key relationships

        common_keys = set()

        for note in note_group:

            key2 = find_key2(note)

            grid_top_row = set(generate_major_scale(key2))

            # Update intersection

            common_keys = common_keys.intersection(grid_top_row) if common_keys else
grid_top_row

    else:

        # Use provided key as constraint

        key2 = find_key2(key)

        grid = [generate_major_scale(note) for note in generate_major_scale(key2)]

        common_keys = find_notes_common_to_columns(grid, note_group)



    return sorted(list(common_keys))
```

3. Harmony Generation

```
def generate_harmony(raw_harmony, chord_progression, modes):

    harmony = []

    for raw_note, chord, mode in zip(raw_harmony, cycle(chord_progression),
cycle(modes)):

        if chord == 'No Common Keys':

            harmony.append('No Harmony')

        else:

            modal_scale = generate_modal_scale(chord, mode)

            harmony_note = adjust_to_scale(raw_note, modal_scale)

            harmony.append(harmony_note)

    return harmony
```

4. Trimetric Integration

```
def integrate_trimetric_scale(scale):

    notes = scale.split()

    return notes[-3:] + notes[:-3]  # Move down a major third
```

Implementation Details

1. Input Parameters

- **Key** (optional): Single note constraining harmonic relationships
- **Melody**: Note sequence with optional comma grouping for tonal centers
- **Chord Progression**: Numbers 1-7 representing modal choices

2. Output Components

- **Analysis Tables**: Common keys, modal chords, and interval relationships
- **Grid Visualizations**: Key² grid and trimetric relationship displays
- **Random Chord Progression:** Optional randomized chord progression generation
- **File Exports**: Text analysis and multi-track MIDI generation

Web Implementation

- JavaScript-based real-time analysis
- Dynamic table and grid generation

Computational Performance

- Common key identification: O(g*n) where g = note groups, n = notes per group
- Harmony generation: O(t) where t = total notes
- Overall memory footprint: < 5MB for typical use cases