

Modulation Mining—Computer-Aided Exploration of Functional Chord Forms

Music & Science

Volume 6: 1–22

© The Author(s) 2023

DOI: 10.1177/20592043221149928

journals.sagepub.com/home/mns



Markus Lepper¹ , Baltasar Trancón y Widemann^{1,2},
and Michael Oehler³ 

Abstract

The different labelling systems defined by the functional theories of harmony have been developed from about 1870 to 1950, without the help of computers. The complexity of harmonic labels spans from pure triads over added characteristic dissonances, over alterations of the fifth, up to non-chord notes like suspensions. Computer-based evaluation of functional label expressions shows that thereby their cardinality increases up to several thousands. For this, we present the basic theory, a concrete program implemented in Prolog, and some empirical results. The software is capable of analysing historic published analyses by inductively collecting all appearing labels, as well as theories as such, where the set of labels is given deductively, by regular expressions. A major application is the mining for possible modulation chords, i.e. different functional labels which result in (enharmonically) the same pitch classes. That this strategy had actually been applied by composers manually is explained by significant examples from the Romantic period.

Keywords

Functional harmonic theory, Romantic harmony, Riemannian analysis, tonal modulation, Prolog, regular expressions

Submission date: 6 June 2022; Acceptance date: 15 December 2022

Aspects of Functional Harmonic Theory

Historic Origin

The origins of *functional harmonic analysis* were established by Jean-Phillipe Rameau ([1722]1965). The basic idea is that particular patterns of interval structures and of chord sequences have a characteristic effect on the listener, which is independent of most other aspects of a musical work. Influential publications by Weber (1817), Fétis ([1844] 2008), Hauptmann (1873) and many others followed. The functional analysis in the narrow sense was defined by Hugo Riemann. He developed in a series of publications (Riemann 1877, 1880, 1895, 1918) an evolving theory which brings together different aspects: the underlying acoustic phenomena, chords' structures and relations, and tonality, culminating in the inner representation by the receiving mind. The different stages of his theory were accompanied by an evolving *symbol system*, in which combinations of letters, numbers and graphical signs express the relations between chords and tonal functions. The symbols can be used to *label* concrete examples

of written music for notating their assumed inner logic or a possible way of reception, see for instance Figure 3. They can also be used stand-alone to speak more precisely about abstract harmonic patterns, independent from a concrete musical realization and even independent from a particular musical key, as in Table 1.

Successively, many different functional theories have been developed, each with its own symbol system. They are all based on one of Riemann's original proposals, but vary in different ways (Marschner (1894), Oettingen (1913), Erpf ([1927]1969), Karg-Elert (1931), Keller (1957)). For a survey, see Imig (1970). The only ones which survived in pedagogical practice (Imig 1970,

¹ semantics gGmbH, Berlin, Germany

² Nordakademie, Elmshorn, Germany

³ Universität Osnabrück, MT DML, Niedersachsen, Germany

Corresponding author:

Michael Oehler, Universität Osnabrück, Germany.

Email: michael.oehler@uos.de



p. 223) are the systems based on the variants by Grabner (1923) and Maler (1931), hereinafter called *GM-style*.

Different Semantics of Functional Terms and Their Relations

As discussed in detail in Lepper et al. (2022a), different semantics can be assigned to a sequence of labels from a functional labelling system. In most use cases, the author of a labelling sequence intends to apply several different semantics simultaneously, for expressing a combined interpretation. Let two extreme positions be called S-m and S-p. Semantics S-m try to model the mental reception of the labelled music fragment or the abstract chord sequence.

To the classically attuned ear, the identification of the tonic immediately evokes a hierarchization of the constituent tones and chords, intuition about their context free proximity, a syntax that governs component ordering, and [...] semantics, [...] which symbolize a rich network of overlapping metaphors: [...] energy, gravity, attraction, magnetism, discharge, orientation, centre, departure and arrival, return, passing, neighboring, leading, deception, completion/incompletion, suspension, finality, stability/instability, and so forth (Cohn 2012, p. 178).

All these aspects can comprise the semantics S-m, when the author of an analysis selects a particular functional label for a particular event, sounding or notated.

Let (in any pitch organizing system) a *pitch class* stand for the equivalence class of pitches up to the octave. Then the semantics S-p are just a collection of pitch classes which make up the chord (or at least a part of it) to which the label is attached—an automatic translation coming with no further claims for meaning.

Basic Principles of Functional Chord Labels

Functional chord symbols consist of two parts: a sequence of letters gives some pitch class as the *root* of the chord, relative to the currently valid tonic centre. The characters **T**, **t**, **D**, **d**, **S** and **s** stand for the chords of the three *fundamental functions* tonic, dominant and subdominant, with major or minor third, respectively.

The characters can be followed by a sequence of numbers and modifiers which give the *components* of the chord. Each such component, including the root, is a pitch class corresponding to one or more pitches contained in the concrete sounding musical event. Each number specifies one component by the numeric name of the interval from a pitch representing the root class upward to a pitch representing the component. The modifiers qualify this interval as major, minor, diminished, augmented, etc.

For a compact notation, the pitch class of the root, the third and the fifth are assumed to sound implicitly; the quality of the third (major or minor) is indicated by upper or lower case of the last character. These implicit sounds can be suppressed by the dedicated modifier *'* appended

to the root expression or to the numbers '3' or '5'. All other numbers indicate intervals which either sound additionally or replace one of the implicit pitch classes.

Modulations

All textbooks analysed by us agree that central interest lies on the analysis of *modulations*. (Distler (1940), Lemacher and Schroeder (1958), de la Motte (1976), Andreas and Friedrichs (1986), Kretschmer (1987), Krämer (1997) and others) These are the harmonic processes (or chord sequences) which cause a change of the feeling of the current tonic centre by the listener. In most labelling systems they are notated by stacking two lines of symbols: the first describing the reception/interpretation relative to the ruling diatonic centre when entering this passage; the second relative to the new diatonic centre reached at its end, see Figure 3.

The role of a modulation can vary: for example, it can be intended to be noticed explicitly as such, or contrarily as a mere subconscious effect, see de la Motte (1976) for a discussion. Independently of this formal and dramatical role, only with respect to the sounding pitch classes, textbooks agree that there are three types of modulations called *diatonic*, *chromatic* and *enharmonic* (Geller (2002), Hussong (2005), Acker (2009)). We have not yet found a single consistent definition of chromatic modulation, but the others are always precisely defined:

The simpler *diatonic modulation* takes a simple triad which fulfils a particular function in one particular first diatonic context and re-interprets the same triad with the same root pitch class as a different function in the target context. This operation can only be applied to a simple triad of form **T** or **t**, because a characteristic dissonance (these notions will be introduced below with our example labelling system i1) would tie the chord to a particular function, i.e. distance from the tonic. (The same process can be described using scale degree theory, i.e. Roman Numeral notation—indeed in many contexts and concerns both symbol systems are exchangeable.) Figure 1 shows examples of how to reach every step of a chromatic scale by functional root expressions.¹ The postfix operators **p**, **P**, **g** and **G** are used to navigate by a *diatonic* third to the chords 'between' the main functions ('*Nebenfunktionen*'), as can be seen in Figure 2. They do not contribute to new chord forms and thus are not needed in the rest of the article. Here they illustrate diatonic modulation: Figure 3 gives examples of how to modulate 'as far as possible' by identifying chords from the extreme ends of Figure 2.

Enharmonic Modulations, Functional PC Set Homonyms and Keyboard Patterns


Enharmonic modulation is quite different: it does not re-interpret the roles of the same triad and root pitch with respect to two different diatonic contexts but rather the role of the chord's components with respect to two (nearly always) different root pitches. As long as no non-

chord notes are involved, this requires at least one of the chords to have a complex **D**, **S**, or **s** form. In nearly all cases the correct notation requires an enharmonic change of notation for the same pitch class—hence the name.²

The upcoming predominance of equally tempered twelve-tone tuning and the keyboard for theory, demonstration, teaching and composing in the eighteenth century allowed these new harmonic patterns, in which one particular set of pitch classes (or one particular keyboard grip) is interpreted as different functions relative to different root pitches and thus to different diatonic contexts. This can happen successively to modulate from one such context to another, or simultaneously to add an additional aura to a seemingly unambiguous function.

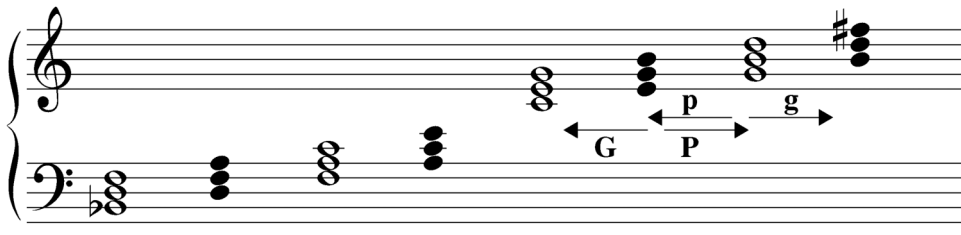
In contrast to notated pitches, the keys of the standard keyboard represent *enharmonic* pitches. Their classes up to the octave are *enharmonic pitch classes*, called *EPCs* by Hentschel et al. (2021a). Sets of these classes are the subject of pitch class set theory (Forte 1973) and are called *pc sets*.

As a consequence, in the mind of musicians at least two classification grids for chords are applied *concurrently*: the functional interpretations, representing all the energies and tendencies listed above, versus the mere mathematical logic of the resulting ‘neutral’ pc sets.³ These can be reified as concrete *keyboard patterns*, collapsing representatives of all pitch classes into one octave. Many composers (including Beethoven and Wagner) used the keyboard as a kind of ‘harmonic



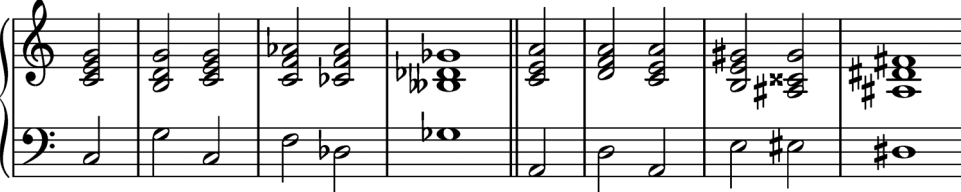
C:T	sG	DD	tP	Tg	S	DDg	D	sP	Tp	SS	Dg
a:tP	{Tg}	s	ssG	D	tG	Sg	dP	{Dg}	t	sG	DD

Figure 1. Examples of how to reach every step of the chromatic scale by functional expressions. Terms in braces {...} describe the enharmonically exchanged pitch class.



	C:Sp	S	Sg=Tp	T	Tg=Dp	D	{Dg}
	C:ii	IV	vi	I	iii	V	{vii}
a:{sG}	s	sP=tG	t	tP{=dG	d	dP}	
a:{bII}	iv	VI	i	III	v	VII	

Figure 2. All functions of major and minor key arranged by the circle of fifths. Terms in braces {...} do not belong to both diatonic keys (even not to the ‘extended tonality’ which mixes major and minor mode = ‘Mischung’ by Schenker) because they need (a) additional accidentals or (b) the minor chord on the fifth scale step.



C:T	D	T	s	a:t	s	t	D
			Gb:Dg	D7	t		d#:sG
							D7
							t

Figure 3. Examples of ‘utmost far’ diatonic modulations.

abacus’ for ‘manual’ exploration of modulations and harmonic proceedings. In this process, the mental image of functional relations and the functionally neutral patterns of keyboard grips, corresponding to pc sets, permanently co-exist and possibly conflict or co-operate.⁴

Therefore it is a valuable preparatory step for the harmonic analysis of late Classical and all Romantic music to find and classify all those *different* terms from a functional harmonic symbol system (functional label together with a diatonic reference pitch) which produce *identical* pc sets. In this article these terms are called (*functional pc set*) *homonyms*. The related theory and software for their automated detection and classification set the theme of the *funCode-pcSets* sub-project presented in this article. Since the diatonic reference pitch class for every functional symbol and thus the resulting pc set can be transposed arbitrarily, homonyms are determined by the *classes* of pitch class sets up to their (cyclic) transposition. These are called *forms* of the pc sets by Collins (2004).

Functional pc set homonyms can become important in at least two aspects: they can (a) be employed by the author explicitly, as the *tertium comparationis* to change the functional context, as the *common chord* in a modulation. But they can also (b) influence the reception of a piece of music by inducing the reminiscence of the other diatonic sphere, possibly without transgressing the threshold to consciousness.⁵ We call these the *pivot role* and the *auratic role*, respectively.

Our agenda for the exploration of functional homonyms is (1) to manually analyse some simple and small functional labelling systems, and in the process (2) to derive classification grids and strategies. Then (3) these are implemented by a computer program, which is (4) finally applied to a realistic labelling system, which is beyond manual analysis due to its prohibitive size. As a by-product, we get a classification grid for the types of homonyms, see Table 1, which will be explained later, when the first examples have been constructed.

Modelling Functional Symbol Systems in the FunCode Project

The *funCode* project, which is the larger context of the results presented in this paper, is a first attempt to reexamine the historically defined labels of functional harmonic theory as consistent mathematical objects (Lepper et al. 2022a). This is done by applying rigorous techniques from computer science, namely compiler theory, to the semantics S-p. Being defined in terms of numbers, it is much more easily accessible to mathematical formalization than S-m. But a formalized model of a core area, free of contradictions and ambiguities, appears to be a good starting point for a wider discussion as well. The main aspects of the *funCode* approach in general are:

- ‘Semantics S-p’ in a precise sense means mapping rules from sequences of symbols to sequences of sets of pitch classes. (Not necessarily enharmonic pc sets!)

- *FunCode* is not the invention of yet another symbol system—it enhances the well-proven GM-style system through a simplified and consistent notation which is easy to read and write, both by humans and computers. This *funCode* notation will be used throughout this paper. The GM-style goes back to Grabner (1923) and Maler (1931) and is the most common textbook system in continental European functional theory. (Imig 1970, p. 223).
- This re-formulation is not only for the execution by computer systems. A more important goal is to assign *mathematically sound semantics* to all possible syntactical combinations.
- Therefore, a subset of the programming language Prolog has been chosen for the implementation, because it comes with precise and explicit mathematical semantics. Carefully chosen Prolog clauses can be directly read as the axioms of a mathematical specification. This is not so easy with other programming languages, which may be better suited for fast and efficient programming. Lepper et al. (2022b) publish the source text in the style of ‘literate programming’, which combines program listings and explanatory text.
- These semantic definitions are intended as a translation target for other symbol systems from historical music theory, to make them comparable and discussable by humans on a precise basis, and possibly even mutually convertible by automated processing.

Mining for Homonyms in the FunCode-pcSets Sub-Project

According to the derivation of pitches and chords in classical functional theory from psychological and physical hypotheses, the pitch classes in the *funCode* core project had been given as coordinates in the Euler net (*‘Tonnetz’*). According to the new research question in this sub-project, namely mining of homonyms, we now use *enharmonic* pitch classes.⁶

Intention and Operation of the Prolog Implementation

The symbol systems of functional theories were developed from about 1870 to 1950, without any help from automated processing by computers. The *manual* analysis executed by their authors could only cover the chords of the first, most basic construction steps and cannot be considered complete. The work reported here applies programmed execution (by Prolog software) to different functional symbol systems.

The basic definitions of the interval numbers, modifiers and the set of the allowed combinations of root symbols and intervals (for chord notes and non-chord notes) are given to the algorithm as parameters. It calculates all possible chord forms, sorts them according to the normal form of the

resulting pc set and delivers statistical data, based on several measures of chord complexity.

There are two use cases: (A) The set of expressions can be defined in a *normative* way, by prescribing the set of all possible terms by a regular expression. This can be appropriate when processing a particular theory. The examples in the article follow this first strategy.

The other use case (B) follows the *inductive* approach by collecting all the expressions found in a particular analysis or group of analyses and adding them step-by-step to the Prolog database. In this case, a *normalization* of the chord component list is required, to eliminate duplicates.⁷ The applicable evaluations are the same in both cases:

- For a particular pc set and labelling system, all possible interpretations as function terms can be retrieved.
- For a particular labelling system, all generated pc sets and all pairs of homonyms can be listed, in detail or summarized.
- All these inquiries can be filtered by value ranges for different complexity measures, to separate practically relevant combinations from extreme artefacts.
- All the results can immediately be visualized for the user, or retrieved as Prolog data for further processing.
- Importing Prolog evaluation results into a language which does not know backtracking (such as Python, R or Matlab, all widely used in music theory) can only be done by *encapsulated search*. This can be done, for instance, by starting SWI Prolog via its command line interface and employing its libraries for JSON or CSV export.

The software is publicly available under the CC-BY-NC-SA licence on a public source hosting platform, under the project name `funCode`.

Structure of the Following Text

The following section presents a sequence of labelling systems with increasing complexity (i1 to i3), for which manual analysis of homonyms is still possible. This analysis is carried out in detail to demonstrate the pre-requisites and consequences of homonyms. A new classification scheme of homonyms (Table 1) is developed as a by-product.

The next section presents the labelling system i4, which introduces the use of non-chord notes. It contains 1,903,364 pairs of homonyms and thus requires automated analysis, which is demonstrated in practice. As a pre-requisite, the programming needs a new classification scheme for suspensions and retardations, a further by-product (Table 7).

The last sections discuss the psychological effects of functional homonyms in general and present a summary of the results.

Stepwise Increasing Complexity of Functional Chords; Manual Analysis Still Possible

Simplest Labelling System i1: Characteristic Dissonances Only

There is a wide variety of functional theories and corresponding labelling systems. All of them describe (possibly in slightly different sequential order) four ways of deriving more complex chords, starting from the plain major and minor triad:

- α Addition of further pitches in the distance of a third—the characteristic dissonances.
- β Cancelling of chord components, especially of the root.
- γ Chromatic alteration.
- δ Replacing chord notes by non-chord notes.

Operation α adds chord components beyond the root, third and fifth, which are nevertheless considered genuine chord tones. These are the *characteristic dissonances*. They link particular chord forms to particular fundamental functions and go back to Rameau ([1722]1965).

Each chord in a dominant role must be a chord with a major third, because only in major mode can it serve as the leading tone into the root note of the tonic.⁸ Such a dominant can additionally contain a minor seventh, in `FunCode` written as ‘**D7-**’, and further a major or minor ninth, written as ‘**D7-9+**’ and ‘**D7-9-**’, as its characteristic dissonances. Because it is the frequent default case, `FunCode` allows ‘**D7-**’ to be abbreviate to ‘**D7**’.⁹

The subdominant can appear in major or minor mode and can additionally carry a sixth, called ‘*sixte ajoutée*’. The question of which kinds of sixths are allowed with a subdominant of a particular mode is controversial among theorists. All textbooks analysed by us (Distler (1940), Lemacher and Schroeder (1958), de la Motte (1976), Andreas and Friedrichs (1986), Kretschmer (1987), Krämer (1997) and others) agree on allowing the major sixth **6+**. A first design decision (DD-1) is whether to allow the augmented sixth **6++** (as is done by some of the reviewed textbooks) and whether to allow the minor sixth (DD-2) (by none of these).¹⁰

For all these characteristic dissonances, voice leading rules are defined which prescribe their ‘normal application’—e.g. ‘the seventh of the dominant goes down to the third of the tonic’ or ‘the *sixte ajoutée* goes up to the third of the tonic.’ This article abstracts from these rules and all application contexts as far as possible.

`FunCode` supports the different syntactic and semantic rules by its parametrization mechanism. A first simple example of a sensible labelling system constructed only with the operations of type α is called ‘i1’ and contains the chord forms shown in Figure 4.¹¹ The corresponding regular expression which can be fed into our

implementation is shown in Table 2. While it seems understood that only one of both ninths is allowed simultaneously, this is indeed already a design decision (DD-3) which also excludes very moderate modernistic constructions as in Figure 12(a).

The current text is about chord forms only—the functional context of the chords is ignored. Therefore, the letters **D**, **S** and **s** stand just for chords with their characteristic dissonances; **T** and **t** stand for the major and the minor mode of the pure triad without these. This must not be mixed up with their usage when analysing and labelling a concrete piece of music: then a pure triad can appear on every diatonic step, for instance labelled with the complex root expressions from Figure 1, and a dominant can appear as pure triad **D**. Both cannot appear in Figure 4, because it does not add a chord *form* different from **T**.

With the simple collection *i1*, manual analysis is still possible; this shows that even such a simple system already contains two pairs of homonyms: (**S56++**, **D7-**) and (**s56++**, **S56+**), as shown in Figure 5.

Categories of Homonyms

The basis for any enharmonic modulation is thus a pair of homonyms, as defined above. (The German term, sometimes also used in English literature, is ‘*Mehrdeutigkeit*’, which can be translated as ‘ambiguity’.)

Every pair of homonyms falls into one of the different categories defined in Table 1. The top line of this table shows symbolically that the two pc sets into which the functional expressions evaluate are always the same. A global property is whether one or two of the functional expressions involved are of type **t** or **T**. In this case we switch to diatonic modulation, because a chord of this type can describe any step of the chromatic scale. So we assume that both terms are **D**, **S** or **s**, and thus are tied to a particular role (=to a particular scale degree) by the contained characteristic dissonances.

In cases A-1 to A-4, the functional expressions are different. In cases A-1 and A-2, the pitch classes, to which the expression is applied as its root, are the same. In case A-2, the root symbols (ignoring upper or lower case) are different. Therefore, the pair can be used for a modulation between two

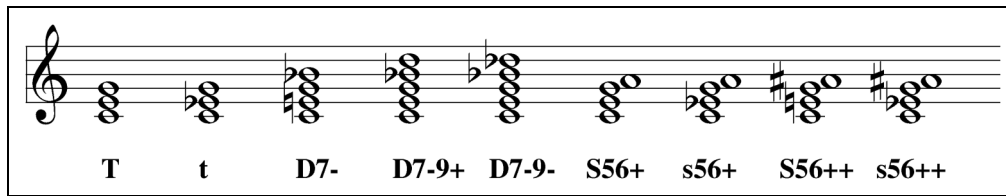


Figure 4. All chord forms from *i1*, a first most simple example labelling system.

Table 1. Categories of Ambiguities, with Examples.

pc set	=				
func. expr.	≠				=
root pitch class	=		≠	=	
root symbol (up to case)	=	≠	=	≠	(=)
	A-1	A-2	A-3	A-4	A-5
<i>i1</i> by α	–	D7 \cong S56++	S56+ \cong s56++	–	–
<i>i2</i> by β	–	–	D/7 \cong D/5/79-	D/79+ \cong s56+	D/79- = Dv (rotate by 3 or 6) D/5/7 (by 6)
<i>i3</i> by γ	–	–	D7 \cong D/5-79- (= Gr) D5-79+ \cong D5+79+ \cong D5-5+7 \cong D/5-5+79+ eight more pairs	D/5-79- \cong S56++	D5-5+79+ (by 2) D5-7 (= Fr) \cong D/5+79+ (by 6)
(<i>i3b</i> , few more β and γ)	–	total 6	total 31	total 23	total 17
<i>i4</i> with non-chord notes	S57- \cong S56++ total 218,352	total 74,652	total 1,005,335	total 604,321	total 704

tonics at the distance of two fifths: the same keyboard pattern (pc set) serves as dominant for one tonic and as subdominant for the other. The homonyms (**S56++**, **D7-**) from i1 fall into A-2: $c + e + g + a\sharp$ is subdominant to $b + d + g + b$, and $c + e + g + b\flat$ is dominant to $f + a + c$, see Figure 5. In both interpretations c is the root pitch class of the chord.

Case A-1, in which the root symbols (up to case) are identical, cannot happen with genuine chord pitches only, because the characteristic dissonances are defined unambiguously—hence their name. Case A-1 will later be filled with more complex chord sets, by *non-chord notes* enharmonically identical with other pitches, like **S7-** (which is a *suspension* to **S56+**) versus **S56++**.

Much more frequent will be the ambiguities from cases A-3 and A-4, where the root pitch classes are different. In case A-3 (identical root symbol), their interval immediately gives the interval between the tonics involved in a modulation; in case A-4 (root symbols **S** or **s** versus **D**), this interval must be added to the abovementioned double-fifth to get the modulation distance. The pair (**s56++**, **S56+**) falls into A-3, see Figure 5: $f + a\flat + c + d\sharp$ with root f is the same keyboard grip (pc set) as $a\flat + c + e\flat + f$ with root $a\flat$; one is related as a subdominant to the tonic c , the other to $e\flat$.¹²

The category A-5 is fundamentally different and will be explained in the next section.

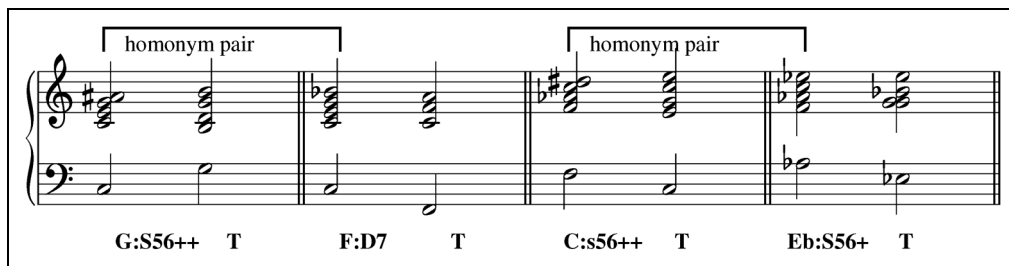


Figure 5. All homonym pairs in labelling system i1.

Table 2. Regular expressions for the example instances.

1 using α = characteristic dissonances :
$t T s 5(6+ 6++) S 5(6+ 6++) D 7- (9- 9+)$
2 using β = cancellations :
$t T (3/) s 5(6+ 6++) S 5(6+ 6++)$ $ D (1/) (5/) 7- (9- 9+)$
3 using γ = alterations of D5 only :
$t T (3/) s 5(6+ 6++) S 5(6+ 6++)$ $ D (1/) (5/ 5- 5+ 5-5+) 7- (9- 9+)$
3b few more β and γ , for demonstration only :
$T (1/) (3/) (5/) t (1/) (5/)$ $ s (1/ 1+ 1-) 5(6+ 6++)$ $ S (1/ 1+) (3/) 5(6- 6+ 6++)$ $ D (1/) (5/ 5- 5+ 5-5+) 7- (9- 9+)$
4 adding non-chord pitches (suspensions, neighbour notes, etc.) :
$T ((7^\wedge (1) (2- 2+)) ((1) (2- 2+)) 1/)$ $((2^\wedge 2^\wedge) (3) (4 4+)) ((3) (4 4+)) 3/)$ $((4^\wedge) (5) (6- 6+)) ((5) (6+ 6-)) 5/)$ $ t ((7^\wedge (1) (2- 2+)) ((1) (2- 2+)) 1/)$ $((2^\wedge) 3 (4)) (3 4))$ $((4^\wedge) (5) (6- 6+)) ((5) (6+ 6-)) 5/)$ $ s ((7^\wedge (1) (2- 2+)) ((1) (2- 2+)) 1/)$ $((2^\wedge 2^\wedge) (3) (4 4+)) ((3) (4 4+)) 3/)$ $(4^\wedge 5 4^\wedge5)$ $(6+ 6++ 7- 7+)$ $ s ((7^\wedge (1) (2- 2+)) ((1) (2- 2+)) 1/)$ $((2^\wedge) 3 (4)) (3 4))$ $(4^\wedge 5 4^\wedge5)$ $(6+ 6++ 7- 7+)$ $ D ((7^\wedge (1) (2- 2+)) ((1) (2- 2+)) 1/)$ $((2^\wedge 2^\wedge) (3) (4 4+)) ((3) (4 4+)))$ $((4^\wedge) (5) (6- 6+)) ((5) (6- 6+)) 5/ 5- 5+ 5-5+)$ $(6^\wedge 7-) (9- 9+ 10-)$

Example Labelling System i2: Applying Derivation β ; Cancellation of Chord Components

In functional theory, the *cancellation of a chord component* means that a pitch class can be removed from the concrete sounding event without affecting the psychological effect of the complex chord constructed so far. This operation (β in our agenda) requires a plethora of design decisions. For instance (DD-4), if one allows **T5/** as well as **t/**, then a major third can be labelled with the intuition that the listener (still) hears the two lowest notes of a major chord, and the second label implies that they hear the two top notes of a minor chord: identical S-p semantics, but different S-m. Do we want these kinds of differences to be expressible? In any case, new ambiguities are created.

It is understood that characteristic dissonances cannot be cancelled, because then they would not have to be introduced in the beginning. And the third of the dominant cannot be cancelled, as it is the leading note required for the dominant to work as such.

The most important cancellation is **D/7**, that of the root of the dominant seventh chord (German ‘*Verkürzung*’).¹³ Figure 6 shows the chords to add to **i1** to get the next example labelling system **i2**. See Table 2 for the regular expression and Table 1 for the new ambiguities arising. Among others, we made the following design decisions: (DD-5) = there is only **T3/** and not **t3/**, which means that we cannot label the difference between ‘no third is sounding, but the listener assumes minor’ and ‘no third is sounding, but the listener assumes major’—again S-p versus S-m.¹⁴

(DD-6) = we did not give any cancellations to the subdominants. (DD-7) = the dominants got all possible cancellations, namely the free combinations of that of the root, which is indispensable because it is the most important, and that of the fifth.

Due to the more complex chord structure with ‘holes in the middle’ we get a further ambiguity with the same root symbol **D** and different root pitches in the category A-3: **e + g + bb** as **D/7** applied to root **c** related to the tonic **f**, and **e + g + a#** as **D/5/79-** with root **f#** related to the tonic **b**, see Figure 7.

A unique case is **e + g + bb + d** as **D/79+** with root **c** as dominant to **f** versus **s56+** with root **g** as subdominant to **d**, because this re-interpretation does *not cause* enharmonic change of notation in the foreground (but in the background the coordinates in the Euler net change nevertheless, see Lepper et al. (2022a)).

But even more important are the first instances of A-5, where the functional expression does *not change at all*. This implies that the roots to which it is applied are different. And because the resulting pc set is nevertheless the same, it follows that the chord structure is *symmetric by rotation*.

The first examples are the chord **D/5/7**, which is merely the diminished fifth **e + bb**, and which can be re-interpreted as **a# + e**. The second example is more important: **D/79-**, the so called ‘chord of the diminished seventh’ (German: ‘*vermindertener Septakkord*’), in functional contexts often abbreviated as **Dv**. Given its symmetric nature, it is a central means for small scale modulations up to large scale plans of tonic keys, starting in the late Baroque (an early pointed example is the choral and recitative ‘*Er ist auf Erden kommen arm*’ from the *Weihnachtsoratorium* BWV 248) and accumulating in the Romantic era. Due to its symmetry, it serves as a central axis in all neo-Riemannian constructions (Gollin and Rehding 2011). (The other such axis, the augmented triad, needs non-chord pitches, as in **T6-** or **t7^**, or combining of two functions, as in **t/&D5/**.)

Please note that the category of ambiguity A-5 opened with these two chords is fundamentally different from the categories with different functional expressions—a distinction not always made clear by textbooks.

Figure 6. All additional chords for **i2** by applying cancellation β to some of **i1**.

Figure 7. Two more pairs of homonyms from the expression language **i2**, thanks to cancellations. (The small note heads in the lower system are added only for explanation and *not contained* in the homonyms.)

Example Labelling System i3: Applying Derivation γ ; Alteration

The term ‘alteration’ in English literature is used in significantly different ways, and even more so (*‘Alterierung’*) in German. In the widest sense it just refers to a chromatic change of an arbitrary chord component. In the narrowest sense, which is used in this article, it is restricted to a chromatic change which *strengthens* a *step-wise* tendency which is *already inherent* in the chord.

Therefore **T** and **t**, having no tendency at all (at least in all functional standard models), cannot be subject to alteration. The same holds for **S//5** and **s//5**, which is the root of the tonic, the target of all tendency.¹⁵

It is a matter of taste as to whether the **S56++** is just a special kind of *sixte ajoutée* or whether it should be called an ‘altered **S56+**’.

A similar issue applies to whether the root of a subdominant can be raised (DD-8) or lowered (DD-9). Lowering can only happen when the following tonic is minor—otherwise the lowered root would enharmonically only be an anticipation of the tonic’s third.¹⁶

The root of **D** cannot be altered because it has no immanent *step-wise* tendency which could be strengthened; the third must stay major; a **9++** would be a minor third and a **9–** the root, so these chromatic changes cannot strengthen the chord’s tendency either.

A major design decision (DD-10) is whether the **D//7-** can be altered to **7–**. Hewitt (2000, p. 283) analyses the A major chord on ‘*todgeweihtes Haupt*’ in the first act of *Tristan* as **c:sP (D/5/7–9–)** **s** and finds a convincing voice leading from the diminished 7 down to the minor third of the target, see Figure 11(j).

But the main candidate for alteration is the *fifth of the dominant*. In the standard resolution, it can go in both directions; therefore it is a candidate for **D5-7**, **D5+7**, and even **D5-5+7**. Figure 8 shows the pitch classes of the first resulting chords. Please note that **D5-5+7-9+** is the most compact functional interpretation of a whole-tone scale. When the **5-** is the lowest note, the interval of the ‘augmented sixth’ appears, see the last three chords. In English literature they carry historical names ‘(Chord of the) French, Italian, and German Sixth’ and have been used in the Classical and Romantic era for modulations.¹⁷

This alteration of the fifth opens a plethora of new ambiguities. A manual analysis of these chords is still possible, but only because both the segments limited by the seventh and

the third will be filled by a free combination of alternatives which produce *exactly the same* intervallic structures. Table 3 shows the resulting variants. (The term **D79-** can be excluded because it is no candidate for homonyms: in **i1** to **i3** there is no other term which produces the interval of a minor second.) Chord structures are given as *keyboard patterns*, the numbers giving the distance to the next key to press.

Due to the construction principle, all combinations of the diagonal are rotationally symmetric by six keys/halftones, i.e. by the interval **5-** or **4+**. When the half-pattern is rotationally symmetric in itself (restricted to a half-scale) by *n*, the number of the possible rotations of the whole is $2 * n$. The diagonal generates five homonyms in the category A-5: one with shift width 2, one with 3 and the remaining three with width 6.

The two half patterns <24> and <42> are mutual rotations, therefore also the doubled patterns in their fields on the diagonal, see the frames in the Table. This is a special case because these expressions cover both categories A-3 and A-5 from Table 1: The pc set **c + d + f# + g#** is

Table 3. Dominant chords from example set i3 using γ , the alteration of fifth, shown as keyboard patterns = chromatic keyboard distances. In the number stacks from the headline, the (major) third is the lowest pressed key—in all others it is the (minor) seventh.

Patterns from 3+ up to 7- →	5/	5	5-	5+	5+5-
Patterns from 7- up to 3+ ↓		3	4	2	2
	6	3	2	4	2
		3	4	2	2
D/7 6	6	3	2	4	2
	6	6	6	6	6
		3	4	2	2
	6	3	2	4	2
D/79- 3	3	3	3	3	3
3	3	3	3	3	3
		3	4	2	2
	6	3	2	4	2
D7 4	4	4	4	4	4
2	2	2	2	2	2
		3	4	2	2
	6	3	2	4	2
D/79+ 2	2	2	2	2	2
4	4	4	4	4	4
		3	4	2	2
	6	3	2	4	2
D79+ 2	2	2	2	2	2
2	2	2	2	2	2
2	2	2	2	2	2
2	2	2	2	2	2

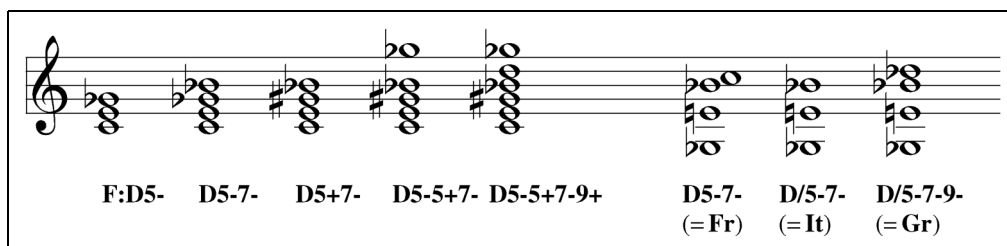


Figure 8. Examples for alteration of the dominant’s fifth and their historic namings.

enharmonically a **D5-7** ('French Sixth') with roots both *d* and *ab* and a **D/5+79+** with roots *bb* and *e*.¹⁸

All fields mirrored at the diagonal yield the same key pattern when applied to two roots at the distance **4+** or **5-**. This table shows clearly the origin of the ambiguity between **D/5-79-** ('German Sixth') and the simple **D7**, which is widely used in the Classical and Romantic eras for (surprising) modulations. All pairs mirrored at the diagonal can be seen as the bases for different variants of *tritone substitution*, which for instance is found ubiquitously in the music of Bela Bartok and in jazz and its relatives.

A special case is that the rotationally symmetric half pattern can become adjacent to the distance 2 from the other half-pattern, making each of the four combinations rotations of each other. So they are the first entry in category A-3 where *four* functional terms are involved.

Therefore we get eight pairs of homonyms and one quadruple. From *i2* to *i3* we have only changed the rules producing dominant forms; therefore Tables 1 and 3 indeed show all resulting homonyms completely.¹⁹

Automated Analysis

Applying Automated Analysis to the Simple Models *i1* to *i3*

Table 4 documents the fundamental functions to operate the software. Programming and testing has been done for SWI-Prolog (threaded, 64 bits, version 8.3.29, <https://www.swi-prolog.org>), but the code is not too specific and should run also on other implementations.

A labelling system is defined by one or more calls to *addFunctions(Id, R)*. Here *Id* is some arbitrary atom to identify the labelling system, which allows for handling several of these in parallel. *R* is the source text of a regular expression such as those in Table 2. This function expands the regular expressions to pc set classes and complexity values and sorts them accordingly. Finally, a call to *expandHomonyms(Id)* analyses and makes explicit all homonym combinations.

Each pc set class is represented by that pc set which gives the smallest number when its bit set representation is read as an unsigned binary integer. This pc set is called

Table 4. Fundamental commands to operate the Prolog implementation.

<ul style="list-style-type: none"> • <i>addFunctions(Id, R)</i> expands the given regular expression <i>R</i> and binds all resulting functional labels to the atomic identifier <i>Id</i>. • <i>expandHomonyms(Id)</i> expands all functional expressions currently bound to <i>Id</i> into to pc sets; sorts, analyses and indexes them. • <i>allClassNumbers(Id, R)</i> returns in <i>R</i> the list of all generated pc set class numbers. • <i>numberOfTerms(Id, R)</i> returns in <i>R</i> the number of all functional expressions bound to <i>Id</i>. • <i>writeText(Id)</i> displays all functional expressions bound to <i>Id</i>, sorted by pc set classes. • <i>writeText(Id, N)</i> displays all functional expressions resulting in the pc set class given by its class number <i>N</i>. • <i>allText(Id, Min, C_i, C_a, R)</i> returns in <i>R</i> a list of text representations for all pc sets with the minimum number of functional terms. The list contains those terms within the given range of complexity. • <i>writeHomonyms(Id)</i> displays all pairs of interpretation for the same pc set class, ordered by their class numbers. • <i>writeHomonyms(Id, J, K, C₁, C₂, C₃, C₄)</i> as above, but filters by tonic distance intervals (<i>J</i> is a list of integers from 0 to 6, giving the required half tone distances), categories (<i>K</i> is list of integers according to Table 1) and complexities (<i>C₁</i> is a list of minimal complexities (<i>c₁, c₂, c₃, c₄</i>) as defined in the main text; they filter each functional label individually; <i>C₂</i> a list of maximal complexities; <i>C₃</i> and <i>C₄</i> give minima and maxima for the <i>sum</i> of both labels.) All filters can be completely opened by supplying the special atomic value 'all'. • <i>writeHomonyms(Id, [N])</i> • <i>writeHomonyms(Id, [N], J, K, C₁, C₂, C₃, C₄)</i> As above, but restricted to the class numbers in the list <i>[N]</i>. • <i>writeHomonymsSurvey(Id)</i> • <i>writeHomonymsSurvey(Id, C₁, C₂, C₃, C₄)</i> displays the number of homonym pairs by the tonic displacements and the categories as defined in Table 1, without and with filtering. • <i>writeSymmetrics(Id)</i> displays the number of rotation symmetric pc sets bound to <i>Id</i>, sorted by the shift interval. • <i>writeSetNormalized(L)</i> takes a list <i>L</i> of atoms representing pc classes and prints a normalized representation. • <i>writeFunction(L)</i> parses a function source text and displays the resulting pc set.

the *normal form* of the pc set class; the corresponding number is called its (*class*) *number*.²⁰

When loading the accompanying Prolog source `funCode-pcSets`, the loading of data corresponding to the example systems `i1` to `i4` is implied. The compound request `allClassNumbers(Id, R)`, `length(R, N)` returns the number of all created normalized pc set classes. The request `numberOfTerms(Id, R)` returns the number of functional expressions. The values for our examples are `i1: 7,9; i2: 15, 19; and i3: 22, 37`.

Interactive input `writeText(Id)` displays all chords bound to `Id`, sorted by pc set classes. This is shown for `i2` in Table 5. The auxiliary command `writeText(Id, N)` shows only the fragment of the table for the pc set class with the given class number `N`.

Every headline shows a pc set class in its normal form. The first visualization is the list of pitch classes as one-digit numbers in base 12, with `t` standing for 10 and `e` for 11, written without any separator, which is a common way of notating in pc set theory. Our format

Table 5. Output of pc sets and corresponding functional expressions when executing `writeText(i2)`.

```

?- writeText(i2) .
--- {05} = 021h = 33 <57> -----
T3/ (51000)
--- {06} = 041h = 65 <66> Symm = 6 -----
D1/5/7- (22000)
--- {026} = 045h = 69 <246> -----
D5/7- (21000)
--- {036} = 049h = 73 <336> -----
D1/5/7-9- (22000)
D1/7- (81000)
--- {0236} = 04Dh = 77 <2136> -----
D5/7-9- (21000)
--- {046} = 051h = 81 <426> -----
D1/5/7-9+ (22000)
--- {0246} = 055h = 85 <2226> -----
D5/7-9+ (21000)
--- {037} = 089h = 137 <345> -----
t (00000)
--- {047} = 091h = 145 <435> -----
T (00000)
--- {0258} = 125h = 293 <2334> -----
s56+ (50000)
D1/7-9+ (T1000)
--- {0358} = 129h = 297 <3234> -----
s56++ (50000)
S56+ (80000)
--- {0368} = 149h = 329 <3324> -----
S56++ (80000)
D7- (80000)
--- {0369} = 249h = 585 <3333> -----
  Symm = 3
D1/7-9- (21000)
--- {02369} = 24Dh = 589 <21333> -----
D7-9- (20000)
--- {02469} = 255h = 597 <22233> -----
D7-9+ (20000)
true

```

uses braces '{..}' instead of brackets '[..]' corresponding to the mathematical notation of a *set*. Then follows the bit set interpreted as a positive integer number, in hexadecimal and decimal notation. Then follows the *keyboard pattern* as a sequence of key distances—one-digit numbers in the same number format. We use angle brackets as is common for sequences, and the cycle of the octave is closed, so there are as many keyboard distance values as members in the pc set.

If the pc set class is in itself rotationally symmetric, then the corresponding rotation angle (= shift interval) is shown at the end of the line.

The functional expressions appear below the pc set. They are followed by five one-digit numbers, again without separator and in base 12. The first integer `d` is the distance of the resulting pc set to the normal form of the pc set class as printed in the headline: when the functional expression is applied with the pitch class '0' as its root pitch class, then the resulting pc set must be shifted upward (cyclically) by `d` steps to get the normal form of the pc set class. Then follow the complexities `c1` to `c4`, see the dedicated section below. (Due to the parametrization `fun_tacetSensibilis(i2)`, in `i2` some functional expressions appear with `c1 > 0`.)

A comparison of Tables 6 and 1 shows the identity of programmed and manual analyses, thus far.

The auxiliary command `allClassNumbers(Id, R)` delivers the list of all class numbers bound to `Id`. The interactive auxiliary command `writeSetNormalized(L)` takes a list `L` of atoms representing pc classes (numbers 0 to 11, allowing also the atoms `t` and `e`) and shows a text with the numerical representations of this set and its normal form.

The command `writeHomonyms(Id)` prints all pairs of homonyms, sorted by their pc set. This is displayed as a headline, as described above, followed by one line for each pair. Each line shows two functional expressions connected by an arrow which is labelled with the *tonic distance* measured in half-tone steps upward. (The complexity values of both expressions follow; these will be described in the next section.)

The printed interval is that between two tonics to which the expressions are related in functional theory. It lies between 0 and 6, because for larger values the two expressions will be swapped. It is a primely theoretic value; in a concrete musical setting this distance may differ: (a) each of both chords which happens to have the chord form **T** or **t** can of course stand on *any* scale degree, see Figure 1; if (b) the target chord has form **D**, in real-world modulations it often takes the function of the *applied* dominant **DD**, so that the true tonic is again seven half tone steps lower. Nevertheless it is sensible to inquire for all homonym pairs with a particular distance of these nominal tonics.

The auxiliary function `writeHomonymsSurvey(Id)` gives the number of homonym pairs by the tonic distances and the categories as defined in Table 1.

The interactive command `writeSymmetrics(Id)` writes a table which shows how many different pc set classes with inner rotational symmetry are generated by

Table 6. Output of homonym pairs when executing `writeHomonyms(i3)`.

```

?- writeHomonyms(i3).
==== { 06} = 041h = 65 = <66> Symm = 6 =====
D1/5/7-                -- (6) --> D1/5/7-                (2000)                (2000)
==== { 026} = 045h = 69 = <246> =====
D1/5-7-                -- (6) --> D5/7-                (2100)                (1000)
==== { 036} = 049h = 73 = <336> =====
D1/5/7-9-             -- (6) --> D1/7-                (2000)                (1000)
==== { 046} = 051h = 81 = <426> =====
D1/5/7-9+             -- (6) --> D1/5+7-                (2000)                (2100)
==== { 0246} = 055h = 85 = <2226> =====
D1/5-5+7-             -- (6) --> D5/7-9+                (2200)                (1000)
==== { 0248} = 115h = 277 = <2244> =====
D1/5-7-9+             -- (6) --> D5+7-                (2100)                (1100)
==== { 0258} = 125h = 293 = <2334> =====
D1/5+7-9-             -- (6) --> D1/7-9+                (2100)                (1000)
s56+                  -- (3) --> D1/7-9+                (0000)                (1000)
D1/5+7-9-             -- (3) --> s56+                (2100)                (0000)
==== { 0358} = 129h = 297 = <3234> =====
s56++                 -- (3) --> S56+                (0000)                (0000)
==== { 0268} = 145h = 325 = <2424> Symm = 6 =====
D5-7-                 -- (6) --> D5-7-                (1100)                (1100)
D1/5+7-9+             -- (6) --> D1/5+7-9+                (2100)                (2100)
D5-7-                 -- (2) --> D1/5+7-9+                (1100)                (2100)
D1/5+7-9+             -- (4) --> D5-7-                (2100)                (1100)
==== { 0368} = 149h = 329 = <3324> =====
D1/5-7-9-             -- (6) --> D7-                (2100)                (0000)
D7-                   -- (2) --> S56++                (0000)                (0000)
S56++                 -- (4) --> D1/5-7-9-                (0000)                (2100)
==== { 02468} = 155h = 341 = <22224> =====
D5+7-9+               -- (4) --> D5-5+7-                (1100)                (1200)
D5-7-9+               -- (6) --> D5-5+7-                (1100)                (1200)
D5-7-9+               -- (2) --> D5+7-9+                (1100)                (1100)
D5-5+7-               -- (2) --> D1/5-5+7-9+                (1200)                (2200)
D1/5-5+7-9+           -- (6) --> D5+7-9+                (2200)                (1100)
D1/5-5+7-9+           -- (4) --> D5-7-9+                (2200)                (1100)
==== { 0369} = 249h = 585 = <3333> Symm = 3 =====
D1/7-9-               -- (6) --> D1/7-9-                (1000)                (1000)
D1/7-9-               -- (3) --> D1/7-9-                (1000)                (1000)
==== { 02469} = 255h = 597 = <22233> =====
D1/5-5+7-9-           -- (6) --> D7-9+                (2200)                (0000)
==== { 02468t} = 555h = 1365 = <222222> Symm = 2 =====
D5-5+7-9+             -- (6) --> D5-5+7-9+                (1200)                (1200)
D5-5+7-9+             -- (4) --> D5-5+7-9+                (1200)                (1200)
D5-5+7-9+             -- (2) --> D5-5+7-9+                (1200)                (1200)
true

```

the expressions linked to *Id*. Possible shift values are the divisors of 12. The property of rotational symmetry naturally belongs to the pc sets, not to their isomorphic functional expressions. In the implementation, the Prolog database facts `innerRotation(N, R)` link any pc set (used by any functional expression and given by its class number *N*) to the minimal positive rotation angle or to the empty list [] in *R*.²¹

Example Labelling System i4: Replacing Chord Notes by Non-Chord Notes

The last operation δ replaces chord notes by *non-chord pitches*. In every concrete context, those pitches correspond

to notes appearing in historically defined voice leading patterns, like *suspension*, *anticipation*, *passing note* or *neighbour note*. The functional labelling abstracts from these situations and simply lists all pitch classes which appear in a particular region of execution time or notation. The chord forms analysed in this article are merely ‘frozen snapshots’ of unspecified, arbitrary voice leading processes.²²

In a psychological interpretation of functional labelling, the non-chord notes are perceived as a second, subordinated level of tension: *first* the listener expects the chord to be ‘cleared’ (by resolving the suspension, by the passing note reaching its target, etc.), and *then* the chord can proceed as a whole, as required by its functional role. Correspondingly, in reduction analysis (Schenker 1935;

Table 7. Non-chord notes for the different chord types, expressed by interval specifications. **2+** and **2^**, **4+** and **4^**, **6+** and **6^** as well as **7+** and **7^** stand for the same pitch class, but suppress the lower or upper neighbour, respectively. Brackets show alternatives. Boxes show *mandatory* characteristic dissonances.

	<i>pc</i>	T	t	S	s	D
7^	11	replaces 1				
2-	1	[replaces 1				
2+	2	[replaces 1				
2^	2	[replaces 3	replaces 3	[replaces 3	replaces 3	[replaces 3
2~	3	[replaces 3		[replaces 3		[replaces 3
4	5	[replaces 3	replaces 3	[replaces 3	replaces 3	[replaces 3
4+	6	[replaces 3		[replaces 3		[replaces 3
4^	6	replaces 5				[replaces 5
5-	6					own for 5
5+	8					own for 5
6-	8	[replaces 5				replaces 5
6+	9	[replaces 5		[own		replaces 5
6++	10			[own		
6^	9					replaces 7-
7-	10			[replaces 6-/+		[own
7+	11			[replaces 6+/++		
9-	1					[own
9+	2					[own
10-	3					replaces 9+/-

Lerdahl and Jackendoff 1983) and all theories based on perception hierarchies (on which McFee et al. (2017) give a recent survey), the non-chord notes are the first to be eliminated. Nevertheless the additional ‘aura’ added to the sound by briefly touching other chords and functions should not be neglected. Furthermore, historic development shows an increasing emancipation of these notes from their original contexts, for instance in jazz harmony, where **Dsus** is a chord form on its own, realizing a ‘frozen’ suspension.

Adding non-chord notes increases the numbers of homonyms beyond the feasibility of manual analysis. This is where computers become indispensable. Table 7 shows symbolically a sensible disposition of non-chord notes. The modifier ^ is used with the same interval meaning as +, but suppressing its upper neighbour instead of the lower. Hentschel et al. (2020, release note v.2.30) call this ‘retardation’, in contrast to ‘suspension’, which always comes from above. This nomenclature is gracefully adopted.²³

The table says that every chord form can get the retardation **7^** and at most one of the suspensions **2+** or **2-** for its root pitch. In a most complex version, these three can be combined, as described by the regular expression $(7^|)(|1|)(2-|2+|)$. It is a fundamental design decision (DD-11) whether the suspension pitch class and its target pitch class itself shall be allowed to sound simultaneously. This is a case explicitly forbidden in Classical

music (de la Motte 1976, p. 69) but increasingly common in Romantic harmonies, see in the *Tristan Prelude* Figure 11 (b) m. 16 **a:D78^ .9+**; (c) m. 17 **tG43 33**; (d) m. 19 **C:D7-6^ 77**; and (e) m. 26 **D3-4^5 .55**.

The next chord note, the third, can be replaced in a similar fashion—the possibilities for minor and major are different because of the different distances: An augmented fourth **4+** can only suspend a *major* third, see Figure 11(g). Again the design decision (DD-5) models the combinations without any sounding third only once, here with the upper case root symbols.

The fifth is even more complicated. **4^** is a possible retardation with all chord types, but **6-** and **6+** are non-chord notes (suspensions) in the cases of **T**, **t** and **D** and are chord notes (characteristic dissonances) in the case of **S** and **s**. Furthermore, **D** type chords can have altered fifth **5-** and **5+** with the same pitch classes as **4^** and **6-**, but the opposite tendency for resolution. So in our toy model i4, we do not allow these to be combined but instead combine **5-5+**, **4^6-** and **4^6+**. These rules cannot be expressed by the simple graphical bracket in Table 7 but only by the regular expression in Table 2.

Characteristic dissonances are chord-owned pitches, so they can be the subject of suspensions or retardations. This leads to **7-** and **7+** with **s** and **S**, and to **D6^** and **D710-**. (Figure 11(h) shows a typical modern variant of the ancient ‘English cadence’: **e:d/10- D. 9+ t3_** etc.)

Putting these pieces together we get the regular expression for set i4 from Table 2; the implementation yields `allClassNumbers(i4, L)`, `length(L, 351)` and `numberOfTerms(i4, 31632)`.

Non-chord notes *always* have an auratic effect, which can be more or less significant. A most impressive example is the passing note g in the bass of Figure 13(b), anticipating the minor mode of the next measures, dialectically in an *upward* movement, and referring back to ‘Brünnhild’s awakening’ more than 10 minutes ago. But non-chord notes can also be used in a pivot role: the examples in Figure 9 are just artificial, but the second one only makes explicit the auratic relations from Figure 11(a). All the more important is a systematic exploration of the resulting homonyms, which cannot be sensibly done without automated processing.

Filtering and Analysing Chord Forms and Homonyms

Figure 10 shows a sensible context of an expression chosen from i4 at random, and a homonym interpretation of its pc set to demonstrate a sensible pivot role. But labelling system i4 contains 1,903,364 pairs of homonyms, see the sum of the figures in the last line of Table 1. For a systematic exploration of this space, automated filter operations are necessary. We define some measures of complexity on the functional expressions:

- c1 The number of own chord pitches which are affected (by a non-chord note, by alteration or by cancellation).
- c2 The number of additional non-chord pitch classes which sound in the chord.

- c3 The number of own chord pitches which sound together with a related non-chord pitch.
- c4 The number of pitches resulting from more than one chord component.
- c5 The number of chord notes which are affected by *two* non-chord notes (mostly one from above and one from below).
- c6 The number of chord notes from c5, which are themselves sounding (= the intersection of the contributors to c3 and c5).

Whether cancellation counts for c1 is controlled by the Boolean style parameter `fun_tacetSensibilis/1`; whether an altered fifth counts as a non-chord note is controlled by `fun_quintaSensibilis/1`. Measures c5 and c6 are not calculated in the current implementation.

Measure c4 can be increased by combinations of suspension and retardation as in 7^ versus 7+, and of characteristic dissonances and suspensions as in 2- versus 9-, 2+ versus 9+ . (It is always a major design decision (DD-12) as to whether these combinations shall be supported!) With alteration we get 4+ versus 5- and 5+ versus 6-. Very doubtful cases are 2++ versus 3-, 6+ versus 7--, or 6^ versus 7--.

Measure c5 = 1 is the characterization of the ‘*doppelte LeittonEinstellung*’ (‘double framing by leading notes’, Maler (1931)) and the resulting ‘*Scheinfunktionen*’ (‘pseudo functions’), see next section for examples.

Surprisingly a limitation of c6 can exclude even only moderately modern situations, see Figure 12(b).

Some logical implications may be useful. Under $\neg(\text{fun_tacetSensibilis} \vee \text{fun_quintaSensibilis})$

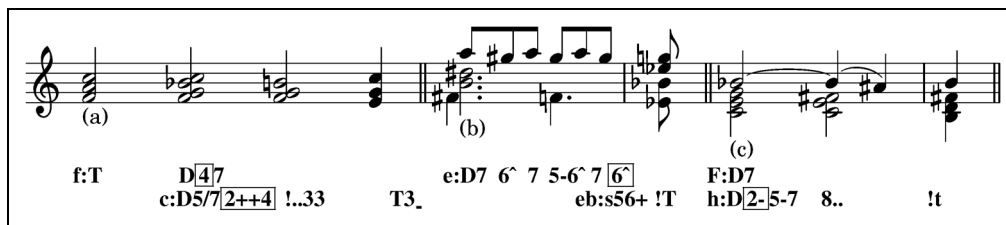


Figure 9. Non-chord notes used in modulation pivots, marked by the boxes. The eureka operator ‘!’ indicates that its prefix is recognized at this point by retrograde re-interpretation.

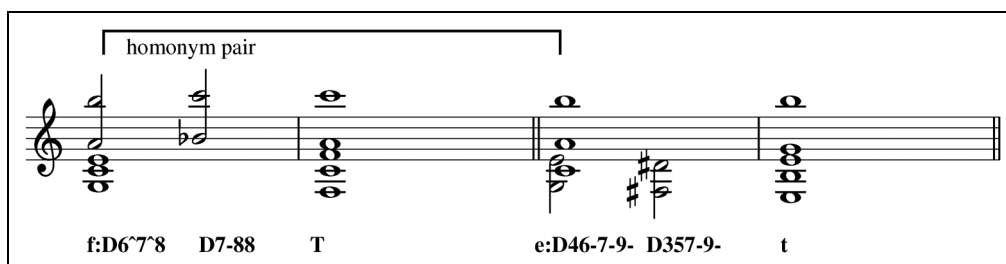


Figure 10. A homonym pair from the labelling system i4. The first expression has been chosen at random from the complexity class (2,2,1,0), to demonstrate that the use in the pivot role is still sensible. (This full-fledged FunCode labelling shows the numbers 3, 5 and 8 in the resolution chords only to indicate the voice leading. In contrast to the very first 8, they do not alter the chord form.)

and a maximum of two sounding non-chord notes per own note, it holds that

$$\begin{aligned}
 c1 &\leq c2 \leq 2 * c1 \\
 c5 &\leq c3 \leq c1 \\
 c6 &\leq c5 \\
 c1 = c3 &\Rightarrow c5 = c6
 \end{aligned}$$

The filter criterion $c1$ is especially useful: a triad with $c1 = 3$ or a tetrad with $c1 = 4$, where every original pitch class is substituted by some suspension, is critical: It will only be effective as its original function under special circumstances, and is even harder to recognize consciously.

But the real borderline depends heavily on genre, epoch and style: replacing more than half of all notes is often found in Romantic music, see the chord marked (x) in Figure 11. (*‘Lausch, Geliebter’* from Tristan II.)

For interactive exploration, the implementation provides versions of the abovementioned display functions which prepend a call to a filter function. `writeHomonyms(Id, I, K, C1, C2, C3, C4)` takes I = a set of intervals filtering the tonic distances, K = a set of integers **0** to **5** filtering the categories, C_1 a minimum complexity which must be reached by both functional expressions involved, and C_2 the corresponding maximum. C_3 and C_4 are the minimum and maximum for the *sum* of both complexities. For convenience, every filter position can take the special value `all` for no effect. The same meanings hold for the parameters of `writeHomonymsSurvey(Id, C1, C2, C3, C4)`.

Therefore `writeHomonyms(Id, all, all, compl(0, 1, 0, 0), compl(11, 1, 11, 11), all, all)` prints all pairs of homonyms where each side has exactly one non-chord note, while `writeHomonyms(Id, all, all, all, all, compl(0, 1, 0, 0), compl(11, 1, 11, 11))` prints those with exactly one non-chord note in total, i.e. summing up both sides.

The request `writeHomonymsSurvey(i4, all, all, compl(1,1,0,0), compl(1,99,0,0))` thus lists all modulations with exactly one affected chord note together in both functional interpretations and at least one sounding non-chord note, see Table 8, covering for instance case (c) in Figure 9.

`writeSetNormalized([2, 6, 9])` shows the structure of a major triad, and `writeHomonyms(Id, N)` displays all

homonyms valid for a particular pc set class given by its class number N in the labelling system *Id*.

Possible Musical Effects of Homonyms and Ambiguities

As mentioned above, there are two main possible effects of all ambiguities: they may be used explicitly by the composer as a pivot chord in a modulation, or they can add a second aura to a sound, a volatile impression of a ‘parallel universe’ in which the same chord would have a different future.

Whether this impression reaches the level of consciousness is dependent on the full context. A simple experiment is offered by the $c\sharp$ fugue in WTC I, see Figure 13(a): In the normal flow of playing and listening, the combination $e+f\sharp$ is unambiguous. But if a halt is made, after at most one second the perception will change to $e + g$, to e-minor—the acoustical equivalent to a ‘rabbit–duck illusion’. A similar test can be made with the **G:D7** appearing at Figure 11(x).

This effect can even raise to an important role in the overall architecture of a larger work: the chord $b\sharp + e + g\sharp$ at Figure 11(i) is in its context doubtlessly a dominant with cancelled root and two retardations **b:D/2⁴7**. But read enharmonically it is also a simple **a:t**—the tonic of the Tristan prelude, desperately missed for more than a century!

In general, `writeHomonyms(i4, 145)` delivers all homonyms of the major triad, and `writeHomonyms(i4, 137)` of the minor.

Somehow in the middle of explicit modulation and implicit aura are the abovementioned ‘double framings by leading tones’ (*‘doppelte Leittoneneinstellungen’*). A typical example are the two opening chords in the finale of Mahler’s First Symphony $f + ab + b\sharp + db$, going to $f + ab + c$, see Figure 12(c). This is heard as **Gb:D7 <f:t4⁶-155**.²⁴

The beginning of the finale of Mahler’s Sixth Symphony is even a ‘double double take’: $ab + c + eb + f\sharp$ may sound as a **Db:D7**; but the contemporary listener knew that this could be a ‘German Sixth’, namely **c:DD/5-79-**. (The version with **3_** had become so frequent in his ‘break throughs’ that Büsing (2012, p. 32) even proposed to call it the *‘Brucknerscher Sekundakkord’*.) So the listener is

Table 8. Numbers of homonyms filtered by minimal and maximal complexities.

```
? - writeHomonymsSurvey(i4, all, all, compl(1,1,0,0), compl(1,99,0,0)).
Survey of homonyms
Complexity limits {min,max} X{ each, sum} = compl(0,0,0,0) / compl(11,11,11,11) / compl(1,1,0,0)
/compl(1,99,0,0) :
```

cat	0	1	2	3	4	5	6	sum
A-1	56	0	2	2	2	0	5	67
A-2	0	2	16	0	3	2	0	23
A-3	0	18	65	41	48	18	24	214
A-4	30	51	24	57	45	45	14	266
A-5	0	0	0	0	0	0	0	0

true

Haupt! (D/5/7--9-) Tod- s DDS-6^ .7 D76- .5 Gb:T 2^4^6- ..5 35. G:D3_7
 Tod- ge- weih- tes

Figure 11. Examples for extreme pitch combinations in *Tristan und Isolde*: (a) a widely accepted interpretation of the beginning; (b) synchronicity of altered and non-altered root as retardation of a 9+; (c) synchronicity of suspension and suspended; (d) major sixth as retardation of a dominant seventh; (e) synchronicity of a fifth and its retardation; (g) augmented fourth going down; (h) synchronicity of a major third and minor tenth; (i) the missing tonic chord a minor as a *Scheinfunktion*; (j) a seemingly remote mediant explained as applied dominant with *diminished seventh*, according to Hewitt (2000, p. 283); (k) support for the **DD5-6+** interpretation of (a); (x) ‘*Scheinfunktion*’ / auratic effect in act II, ‘*Lausch, Geliebter!*’.

D9-9+ (a) t4+6- 55 (c)

Figure 12. Moderately modern examples for more than one suspension to the same note.

Figure 13. Two very different examples of an auratic e minor.

aware of the chord’s double meaning. But indeed it is neither of both, but the sequence is again **c:t4^6- !55**.

That a labelling system like **i4** is required to analyse more elaborate functional harmonics, and that the level of **i3** will not suffice, is obvious in situations like (f) in Figure 11, m. 37 of the *Tristan* prelude, where something ‘sounds like’ a chord of the lowered fifth (here: French Sixth) but is indeed only a short retardation.

Only **i4**, not **i3**, provides the means to reason about the most-discussed chord ever, see Figure 11(a). The functional interpretation by Kurth (1920) (not without alternatives, but shared by most theorists) is that of **a:DD5-6^ .7-** an applied dominant with lowered fifth (in the bass) and an additional retardation into its characteristic dominant seventh.²⁵ (The slightly simpler parallel on ‘*todgeweihtes Herz*’ supports this interpretation, see Figure 11(k).)

So this chord is not covered by **i3**, but only when non-chord notes are considered. It is (in the narrow sense) not a functional *chord*, but a snapshot of a voice leading event. It is nevertheless ‘accidentally’ a ‘half-diminished seventh chord’ and thus a homonym to **s56+** and others. It could be a subdominant to **d#** minor, the most remote key to a minor. This gives the mysterious aura.

Our implementation can put you on that trail by executing `writeSetNormalized([5, 8, 3, 11])` and `writeHomonyms(i4, 293)`.

Related Work, Future Work, and Conclusion

The different systems of functional harmonic music analysis go back to ideas of Rameau ([1722]1965) and have been developed mainly by Riemann (1877, 1880, 1895, 1918). From this starting point, different families of theories evolved. Each comes with a syntax and semantics for chord labelling. The labelling systems of the GM-style family (going back to Grabner (1923) and Maler (1931), the prevailing style in continental Europe’s textbooks) were defined from 1930 to 1970 without the help of computers. The aim of the *funCode* project is only to clarify the semantics of these symbol systems as such. They have been in wide use for nearly a century in professional music theory, with only partial definition of their syntax and without any explicit definition of their semantics. Such clarifications are

indispensable pre-requisites for digital encoding and are helpful for preventing misunderstandings among humans.

In functional theory, a quest of fundamental interest (in theory, but even more in practice) is for homonyms, i.e. pairs of functional symbols which produce isomorphic pc sets / keyboard patterns. These pairs can be used (a) explicitly as pivots for modulation or (b) to explain auratic undertones in seemingly one-dimensional chord sequences. We showed that it is possible to find these homonyms by manual analysis for very simple labelling systems (in this article and in the software **i1** to **i3**), but that realistic complex labelling systems are beyond manual analysis (example system **i4**).

The software presented in this article (part of the *funCode* project) allows a labelling system to be defined deductively by one regular expression or by collecting different labels from some historic analysis. The labelling system then can be asked for all functional interpretations of a given pc set and searched for all pairs of homonyms, filtered by complexity bounds.

These results are loosely but naturally related to two other areas of recent research: first, projects of *manual* annotation of existing encoded music corpora (Temperley and de Clercq 2013; Neuwirth et al. 2018; Hentschel et al. 2021b) are necessarily concerned with harmonic annotation encoding in general. Several systems have been proposed (Harte et al. 2005; Hentschel et al. 2020; Nápoles López and Fujinaga 2020), which are based not on functional but on scale degree theory. These systems address only the syntax and not the semantics, which are considered to be understood or to be taken from informal textbooks. Hentschel et al. (2021a) define a format for using different semantics and encodings in parallel, which can perhaps be enhanced to support the ‘relative sections’ required for functional labelling. Remarkably, Temperley and de Clercq (2013, p. 194) follow the functional approach in one central point, namely that they ‘simply treat a “key” in rock as a single pitch-class’.

Second, there are projects for *automated* extraction of harmonic information from musical data in various formats, among others by Raphael and Stoddard (2004), Rohrmeyer (2007), Illescas et al. (2007), Harte (2010), De Haas et al. (2013), Jacoby et al. (2015) and White and Quinn (2018). Some of the publications have the word ‘functional’ in their title and indeed are definitions of chord forms, of the relations

between root pitches, of search patterns and strategies, etc., all inspired by the fundamental functions **T**, **D** and **S**. But functional harmonic theory as such (which includes relative sections, ‘*Nebenfunktionen*’, cancellation, virtual functions, ambiguities, etc.) is not involved. The computational results are always labellings of the input stream with (linearly or hierarchically organized) scale degree / Roman numeral symbols.

Necessarily, all the abovementioned works of automated processing include a definition of the semantics of the applied labelling system, namely implicitly by the code which extracts the labels from the input data. But only one of them gives an *explicit* definition of non-functional chord names, namely Harte (2010, p. 103). All other publications assume the meaning of the Roman numeral symbols to be understood, or refer to informal textbooks; Jacoby et al. (2015, p. 9) speak of the ‘familiar Roman numerals’. To give exact semantics to scale degree systems may turn out to be a similarly demanding task as for functional systems: Their syntax is simpler, but the context additionally contains the major/minor mode.

The sheer numbers shed light on the different characters of the two families of labelling systems: the dozens of alternatives for a particular pc set, even when choosing narrow complexity limits (the first chord in Figure 10 is part of 28 homonym pairs, even with a low maximal complexity of **compl(2,2,0,0)** on both sides!), clearly show that labelling with functional symbols is a creative utterance of the author of the analysis. Their choice for a particular label is always also a statement about semantics S-m, which include ‘gravity, attraction, magnetism’ etc., see the catalogue by Cohn (2012) cited above. It is a statement about the middle-ground of the piece of art, about some derivation steps in construction and/or reception. But Temperley and de Clercq (2013) point out that even when using scale degree labels, ‘a certain degree of interpretation and subjectivity is involved’.

Nevertheless, both activities could be connected: the sequence of pc sets extracted from an encoded piece of music could be fed into our software and turned into a sequence of *sets* of functional symbols, from which sensible combinations could be selected manually or automatically.

In considering future work, the homonyms collected so far can possibly be taken as a basis for *empirical psychological* studies. For instance, we found in spontaneous tests with piano that listeners more familiar with Classical/Romantic music and thus familiar with tonal organizations around the symmetric **Dv** found the two interpretations of $e + g + bb$ in Figure 7 both indistinctively convincing, while others, more familiar with folk and popular music, would only follow the first.

Furthermore, the *design decisions* listed in the text above only as an informal collection could be made more systematic by specifying them as Prolog code, for synthesis, analysis and comparison of labelling systems.

In addition, all symbol combinations could be collected from the influential textbooks by Distler (1940), Grabner (1923), Maler (1931), de la Motte (1976) and others, automatically evaluated for their expressiveness and thus compared.

Further research is required for automated soundness analysis of regular expressions and whether stronger formalisms are desirable for ergonomic reasons.

A classification scheme for functional ambiguities (A-1 to A-5, see Table 1) and a new systematic approach to non-chord notes (see Table 7) came out as by-products of the mathematical analysis. These are examples of the fact that mathematical re-modelling is also fertile ground beyond digital processing, for discourse by humans. And we discovered the one and only ‘tonic’ chord in the Tristan *prélude*, the chord $b\sharp + e + g\flat$ in measure 56, see Figure 11(i).

Acknowledgements

The *Tristan* examples have been copied from the *Complete Piano Score* by F. H. Schneider, published 1914 by Breitkopf & Härtel, Leipzig. The *Siegfried* example has been copied from the piano reduction by K. Klindworth, published by Schott, Mainz.

Many thanks to the reviewers and editors, who helped to improve this article considerably.

Action Editor

Frank Hentschel, Universität zu Köln, Musikwissenschaftliches Institut

Peer Review

Fabian Moss, École Polytechnique Fédérale de Lausanne, Digital Humanities Institute.

Néstor Nápoles López, McGill University, Music Research’.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.


Ethical Approval


This research did not require ethics committee or IRB approval. This research did not involve the use of personal data, fieldwork, or experiments involving human or animal participants, or work with children, vulnerable individuals, or clinical populations.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: We acknowledge support by Deutsche Forschungsgemeinschaft (DFG) and Open Access Publishing Fund of Osnabrück University (BO 5110/2-1, 491052604).

ORCID iDs

Markus Lepper  <https://orcid.org/0000-0002-9120-3908>

Michael Oehler  <https://orcid.org/0000-0003-1034-8601>

Notes

1. These root expressions are only examples to prove that every pitch is reachable. For each pitch class there are alternatives. In particular it is important that in each concrete labelling application, each functional root expression does not only

- indicate the pitch class reached (semantics S-p, as defined above), but also the *intuition of how* this point is reached (semantics S-m): Different expressions describe different mental representations in the listener. For instance, **DD** and **SP** reach the same scale degree, but ‘with different history’—they are different mental models.
- There are rare cases without this, e.g. the re-interpretations among **D/79+** and **s56+**, but we prefer to keep the name ‘enharmonic’. Indeed, also here the intervals in the Euler net change significantly, which is always the primary effect—the change of the external notation is only a consequence.
 - This can be seen as one instance of a more fundamental hypothesis, namely that in art in general, in production as well as in reception, for providing structure, measures and rules, at any point in time *more than one* system can simultaneously be active. Even more: we propose that more than one active system is the normal case and that conflicts between these are a source not of confusion or disorder but of creative stimuli for the author and increased interest by the listener. ‘[S]omething has a \gg meaning \ll only when it has a few; if we understood something just one way, we would not understand it at all.’ (Minsky 1981). For related questions of multi-syntax listening, see Cohn (2012, p. 195ff); for a composer’s view see Lepper (2015).
 - Reifying the abstract pc set as a concrete ‘keyboard pattern’ or ‘grip’ is obviously appropriate and can reflect the experiences and mental models of a pianist correctly. But it is (dependent on style, epoch and genre) only more or less related to musical practice: minor tenth and major third clash in the pc set view but are separated by at least one octave in most concrete Romantic settings; in jazz harmony, adjacent octave registers are filled with *different* chord components anyhow.
 - de la Motte (1976, p. 152) writes ‘Beachte: S_5^{\leq} und ‘ D_5^{\gt} klingen wie ein Dominantseptakkord.’ (‘Beware that **S56++** and **DDv5-** [=Gr] sound like a dominant seventh chord’). This is of course untrue: the ‘soundings’ of these chords are all totally different, as they are ruled by the functional context in which and together with which they are perceived. ‘As soon as [a pair of pitches] participates in a continuum of musical relationship, its nature becomes completely dependent on its surroundings’ (Mitchell 1962, p. 23). What de la Motte means is, that *taken out of all context*, the *keys to press on the keyboard* are the same. (Some editions erroneously print ‘3>’.)
 - Indeed the transition from Euler coordinates to enharmonic pitch classes can be seen as realised by two chained quotient constructions: first the syntonic comma is eliminated and all points of the plain projected to the axis of the fifths; then the Pythagorean comma is eliminated and ‘e \sharp ’ is identified with ‘f’. Of course, algebraic derivation of the pc set semantics of funCode from its original Euler net semantics would be much cleaner, but it is omitted here to make this article self-contained.
 - The sorting order for normalization of chord components is as follows: first comes 7 \wedge as the ‘lowest in stack’, when the traditional chord structure of ‘thirds above a root’ is imagined. Then follow all other chord components with ascending interval numbers, their modifiers in the order: no modifier, /, ..., --, -, +, ++, ..., ^, ^^, ... Such a normalization is *not possible* when using funCode for concrete analyses, because the sequential order of the code components may describe voice leading, see Lepper et al. (2022a).
 - In the concrete sounding chord, the interval of the major third may not be present due to the operation β = cancellation. But functional theory always speaks about the original triads *before* applying derivations: in all chords in Figure 6 the pitch class *e* is ‘the third’ of the chord (= fulfills the role of the third of the function) and must be a major one, even when the root does not sound. The letter ‘d’ can be used in a particular labelling, see for instance the **dp** in Figure 1. But it describes a minor chord on the fifth scale degree, not a dominant. A minor dominant ‘*ist ohne analytischen Sinn*’ (Büsing 2012, S. 10) (‘is without analytical sense’). He proposes to borrow ‘v’ from Roman numerals even in functional contexts.
 - This abbreviation is currently not supported by the Prolog implementation, especially in the regular expressions. Neither is the cancellation of the root **D/**, which must be written with the explicit interval number **D1/**.
 - Since **D7-**, **D7-9-** and **D7-9+** add further pitch classes in the distance of thirds on top of the triad, a more systematic approach is to derive **s56+** by adding a third *below* the triad. This is the explanation preferred by all authors from the so-called *dualistic* tradition, who explain the minor chord as the inverted spectrum of the major chord, and thus the **s6+** as the exact mirror of the **D7-** (Riemann (1906, p. 161), Hyer (1989, p. 17), Vogel (1962, p. 50)). But Rameau ([1722]1965) and most textbooks (Distler (1940), Lemacher and Schroeder (1958), de la Motte (1976)) speak of a ‘sixte ajoutée’. Since we are dealing with pitch *classes* only (= modulo octave), a sixth above is the same as a third below. With the minor sixth the ‘chord of the Neapolitanian Sixth’ can be written as **s5/6-**. But since the mediant chords are addressable by mere root expressions anyhow, this term is not required but can be replaced by **sG**.
Even the nature and origin of the seemingly simple 7- is discussed controversially among functional theorists. It is seen either as a frozen ‘transitional voice-leading’: ‘*Nach und nach wurde auch die Sept, obwohl Durchgang oder Vorhalt (II2) und die Non, obwohl Vorhalt oder Wechselnote, als Akkordbestandteil hingestellt, womit man zu Sept- und Nonakkorden gelangte.*’ (Schenker 1930, p. 17) Or it is related to the natural seventh by Vogel (1975, p. 93) and Hewitt (2000, p. 143). Or seen as a mixture of dominant and subdominant components, by Rameau ([1722]1965), Hauptmann (1873, p. 114), and Hyer (1989, p. 35).
Hyer (1989, p. 39) mentions that in contemporary music the chord of the seventh can also be employed as tonic ‘**T7-**’, which may not appear in our examples because it does not add any new chord form.
 - Due to the historic evolved structure of notation, there are some peculiarities: the characteristic dissonances and the non-chord notes are written with the same numbers. So ‘6+’ is written for the characteristic dissonance of a subdominant, but also as a non-chord pitch (‘suspension’) with a chord of any other kind, see Table 7. Because the latter role is so frequent, the rules of most symbol systems imply that the appearance of the ‘6’ suppresses the sounding of the default fifth, which is the correct modelling for all voice leading events (suspension, passing, neighbour note, anticipation, etc.) Therefore in the first case, when the sixth sounds *additionally* to the fifth, we must re-establish the sounding of the fifth explicitly by writing ‘**s56+**’. But this is not a semantic issue but only one of external representation, of writing.

While the difference between the two roles of interval numbers seems to be unknown to the authors of most labelling systems, it is nevertheless sensible to use the same numbers with identical semantics independently of the letter which serves as root symbol. In our implementation, the set of the chords' own intervals is configurable per root symbol; it has consequences only for the complexity criteria C1 to C3, as defined below.

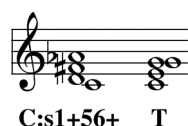
In `funCode` we could define *different modifiers* to distinguish between a sixth as suspension or as *sixte ajoutée*, one suppressing the fifth, the other not. For leaner notation, this is not done in this article.

12. The pc set is identical to that of a 'minor seventh' chord like **t7-**, which is *not a functional* expression.
13. The sequential order applied by our derivations and in nearly all textbooks does *not* correspond to the historical evolution in musical practice: the diminished triad was used *earlier than* the dominant seventh—Kretschmer (1987, p. 36) and Daniel (2000, p. 267, 274) according to Hussong (2005, p. 283, 313). Likewise, the **D/79+** was used (for instance by Mozart) before the **D79+** (Schumann) (Hussong 2005, p. 96) (de la Motte 1976, p. 180). Nevertheless the derivation **D7** \Rightarrow **D/7** had already been used by Rameau ([1722]1965) as a mere 'systematic' explanation, disregarding historic development.
14. That this distinction can indeed be highly relevant shows for instance



from Lepper, 'Acht letzte Worte'.

15. We abuse the notation **S/5** which in core `funCode` stands for a subdominant of which only the single component of the fifth is sounding, to refer to this component itself.
16. Raising is relevant, especially for sequences like



with voice leading $f\sharp - g$, despite the fact that the unaltered root would resolve downward. But in the sequence



the first chord would better be labelled an applied dominant **DD5-7**. (In `FunCode`, the postfix operator `'_'` marks the preceding interval number as the bass pitch of the chord.)

17. In the Classical era these alterations have mostly been applied to *applied* dominants **DD**. But here we abstract from these contexts of usage, which do not contribute to the chord *forms*.
18. The latter is a good example of why it may be sensible to use `funCode` even for *talking*: in German this chord would be called something like '*verkürzter tiefalterierter Sept-Groß-Non-Akkord*', meaning 'root-cancelled fifth-lowered seventh major ninth chord'—but there is no standard of naming anyhow.

Similar de la Motte (1976, p. 156) '*Durgegenklang der Mollsubdominante; umständliche Worte bei einprägsamer Bezeichnung. Am besten spricht man deshalb auch wie man schreibt: klein s großG.*' ('long-winded words for a concise concept; better speak like writing: *lower s upper G.*')

19. This can be verified by the Prolog implementation: entering the command `writeText(i3)` shows all homonyms. `writeText(i3, N)` shows all interpretations of the pc set **N**. It is easy to see that the number would increase considerably as soon as cancellation β and alteration γ were applied also to the other chord forms. Soon we reach the limit of manual analysis. For demonstration purposes the implementation provides the variant `writeText(i3b)`.
20. This kind of encoding is similar to the various notions of 'prime form' developed by musicians, but significantly different. E.g. inversion is not included as an identity transformation, see Collins (2004) for a detailed discussion. Our approach is more canonical from the standpoint of mathematics and of software technology.
21. The regular expression for `i4` is not sound in so far as the totally empty set can be produced by **T/3/5/**. This set is the only one with rotation angle = 1, beside the full chromatic cluster, which is not produced in our examples.
22. The category of the default components **1**, **3** and **5** must not be confused with that of the chord-own pitches, which may be larger by the characteristic dissonances. It is a matter of mere convenience that the former need not to be written but always sound unless explicitly cancelled or suppressed by a neighbour pitch (by suspension etc.). In the Euler net semantics of `FunCode` (Lepper et al. 2022a), this behaviour is configurable. The corresponding definition would be $suppress = \{7^{\wedge} \mapsto 1, 2+ \mapsto 1, 2- \mapsto 1, 2^{\wedge} \mapsto 3, 4 \mapsto 3, 4+ \mapsto 3, 4^{\wedge} \mapsto 5, 6- \mapsto 5, 6+ \mapsto 5\}$. For the higher chord-own notes, no such rules are necessary because they must always be notated explicitly. This difference is a historically evolved front-end phenomenon and could (more regularly) be treated by a pre-processing step which makes the default intervals explicit.

The Euler net semantics of the interval specifications used in the examples can be defined as $intervals = \{7+ \mapsto (1, 1), 1 \mapsto (0, 0), 2- \mapsto (-1, -1), 2+ \mapsto (2, 0), 2+ + \mapsto (1, 2), 3- \mapsto (1, -1), 3+ \mapsto (0, 1), 4- \mapsto (0, -2), 4 \mapsto (-1, 0), 4+ \mapsto (2, 1), 5 \mapsto (1, 0), 6- \mapsto (0, -1), 6+ \mapsto (-1, 1), 7- \mapsto (-2, 0), 7+ \mapsto (1, 1)\}$ Furthermore, every combination of interval number and modifier nm with $n > 7$ is treated as $(n-7)m$ and every $n(\cdot)^m$ as $n(+)^m$. Please note that in the following example systems, all $2(-/+ /++)$ variants suppress their neighbour (=1), but **9..** do not. So only the former can be used for suspensions.

23. Independently to us, they invented different interval modifiers for indicating up or down tendency, but we found no explicit specification.
24. The `funCode` encoding '`<`' indicates that an alternative interpretation track starts in parallel to the symbols so far; '`!`' indicates that this interpretation will be perceived in retrospect, not before the sign **!** has passed, see Lepper et al. (2022a).
25. According to Holtmeier (2011, p. 42), this idea goes back to Ergo (1914).

References

- Acker, H. (2009). *Modulationslehre*. Bärenreiter. ISBN 978-3-7618-2126-6.

- Andreas, H., & Friedrichs, T. (1986). *Harmonielehre*. Hamburg: Wagner.
- Büsing, O. (2012). *Harmonik als Netzwerk*. Georg Olms. ISBN 978-3-487-14844-1.
- Cohn, R. (2012). *Audacious Euphony — Chromaticism and the Triad's Second Nature*. Oxford Press. ISBN 978-0-19-977269-8.
- Collins, N. (2004). An algorithm for the direct generation of set class representatives in any pitch class space. *Music Theory Online*, 10(3). <https://mtosmt.org/issues/mto.04.10.3/mto.04.10.3.collins.php>
- Daniel, T. (2000). *Der Choralatz bei Bach und seinen Zeitgenossen*. Köln: Dohr.
- De Haas, W. B., Magalhães, J. P., Wiering, F., & Veltkamp R, C. (2013). Automatic functional harmonic analysis. *Computer Music Journal*, 37(4), 37–53. https://doi.org/10.1162/COMJ_a_00209.
- de la Motte, D. (1976). *Harmonielehre*. Bärenreiter. ISBN 3-7618-4183-3.
- Distler, H. (1940). *Funktionelle Harmonielehre*. Bärenreiter.
- Ergo, E. (1914). *Über Richard Wagners Harmonik und Melodik*. Breitkopf.
- Erpf, H. ([1927]1969). *Studien zur Harmonie- und Klangtechnik der neueren Musik*. Breitkopf & Härtel.
- Fétis, FJ ([1844] 2008). *Traité complet de la théorie et de la pratique de l'harmonie*. Maurice Schlesinger.
- Forte, A. (1973). *The Structure of Atonal Music*. Yale University Press.
- Geller, D. (2002). *Modulationslehre*. Breitkopf und Härtel. ISBN 3-7651-0368-3.
- Gollin, E., & Rehding, A. (eds.). (2011). *The Oxford Manual of Neo-Riemannian Music Theories*. Oxford Press. ISBN 978-0-19-522133-3.
- Grabner, H. (1923). *Die Funktionentheorie Hugo Riemanns und ihre Bedeutung für die praktische Analyse*. Leuckart.
- Harte, C. (2010). *Towards Automated Extraction of Harmony Information from Music Signals*. PhD Thesis, Queen Mary, University of London.
- Harte, C., Sandler, M. B., Abdallah, S. A., & Gómez, E. (2005). Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations. In: *Proceedings of the 6th International Conference on Music Information Retrieval*. London, United Kingdom: ISMIR, pp. 66–71. <https://doi.org/10.5281/zenodo.1415114>.
- Hauptmann, M. (1873). *Die Natur der Harmonik und Metrik*. Breitkopf und Härtel.
- Hentschel, J., Moss, F. C., Finkensiep, C., & Lieck, R. (2020). Standard for harmonic annotations developed at the Digital and Cognitive Musicology Lab at École Polytechnique Fédérale de Lausanne. Technical report. <https://github.com/DCMLab/standards>. [Accessed 02-Mar-2022].
- Hentschel, J., Moss, F. C., McLeod, A., Neuwirth, M., & Rohrmeier, M. (2021a). Towards a unified model of chords in western harmony. In: *proceedings of the Music Encoding Conference MEC 2021, in preparation*.
- Hentschel, J., Neuwirth, M., & Rohrmeier, M. (2021b). The annotated Mozart sonatas: Score, harmony, and cadence. Technical Report 1. <https://transactions.ismir.net/articles/10.5334/tismir.63/>.
- Hewitt, M. (2000). *The Tonal Phoenix*. Orpheus Verlag. ISBN 3-922626-96-3.
- Holtmeier, L. (2011). The reception of Hugo Riemanns music theory. In *Gollin and Rehding*, pp. 3–54.
- Hussong, H. (2005). *Untersuchungen zu praktischen Harmonielehren seit 1945*. Verlag im Internet GmbH.
- Hyer, B. (1989). *Tonal Intuitions in Tristan und Isolde*. University Microfilms International. ISBN 1-4354-5672-6.
- Illescas, P. R., Rizo, D., & Quereda, J. M. I. (2007). Harmonic, melodic, and functional automatic analysis. In: *Proceedings of the 2007 International Computer Music Conference, ICMC 2007, Copenhagen, Denmark, August 27-31, 2007*. Michigan Publishing, pp. 165–168. <https://hdl.handle.net/2027/spo.bbp2372.2007.145>.
- Imig, R. (1970). *Systeme der Funktionsbezeichnung in den Harmonielehren seit Hugo Riemann*. Gesellschaft zur Förderung der systematischen Musikwissenschaft e.V.
- Jacoby, N., Tishby, N., & Tymoczko, D. (2015) An information theoretic approach to chord categorization and functional harmony. *Journal of New Music Research*, 44(3), 219–244. <https://doi.org/10.1080/09298215.2015.1036888>
- Karg-Elert, S. (1931). *Polaristische Klang- und Tonalitätslehre*. Leuckart.
- Keller, W. (1957). *Handbuch der Tonsatzlehre*. Bosse.
- Krämer, T. (1997). *Lehrbuch der harmonischen Analyse*. Wiesbaden: Breitkopf & Härtel.
- Kretschmer, R. (1987). *Einführung in die durmolltonale Harmonik*. Innsbruck.
- Kurth, E. (1920). *Romantische Harmonik und ihre Krise in Wagners Tristan*. Haupt.
- Lemacher, H., & Schroeder, H. (1958). *Harmonielehre*. Gerig.
- Lepper, M. (2015). Das Zweite System <http://senzatempo.de/ston2015081400.html>.
- Lepper, M., Trancón y Widemann, B., & Oehler, M. (2022a). FunCode — versatile syntax and semantics for functional harmonic analysis labels. *Music and Science*, 5. <https://doi.org/10.1177/20592043221085659>
- Lepper, M., Trancón y Widemann, B., & Oehler, M. (2022b). *FunCode 1.0 Technical Report*. Universität Osnabrück. <https://doi.org/10.48693/28>
- Lerdahl, F., & Jackendoff, R. (1983). *A Generative Theory of Tonal Music*. MIT Press. ISBN 0-262120941.
- Maler, W. (1931). *Beiträge zur durmolltonalen Harmonielehre*. Leuckart.
- Marschner, F. (1894). *Die Klangschrift*. Selbstverlag.
- McFee, B., Nieto, O., Farbood, M. M., & Bello, J. P. (2017). Evaluating hierarchical structure in music annotations. *Frontiers in Psychology*, 8, <https://www.frontiersin.org/article/10.3389/fpsyg.2017.01337>. <https://doi.org/10.3389/fpsyg.2017.01337>
- Minsky, M. (1981). Music, mind, and meaning. *Computer Music Journal*, 5(3), 28–44. <https://web.media.mit.edu/~minsky/papers/MusicMindMeaning.html>. <https://doi.org/10.2307/3679983>
- Mitchell, W. J. (1962). The study of chromaticism. *Journal of Music Theory*, 6(1), 2–31. <http://www.jstor.org/stable/843257>. <https://doi.org/10.2307/843257>
- Nápoles López, N., & Fujinaga, I. (2020). Harmalysis: A language for the annotation of roman numerals in symbolic music representations. In E. De Luca & J. Flanders (Eds.), *Music encoding*

- conference proceedings 2020* (pp. 83–85). Humanities Commons. <https://doi.org/10.17613/380x-dd98>
- Neuwirth, M., Harasim, D., Moss, F. C., & Rohrmeier, M. (2018). The Annotated Beethoven Corpus (abc): A dataset of harmonic analyses of all Beethoven string quartets. *Frontiers In Digital Humanities*, 5. <https://www.frontiersin.org/articles/10.3389/fdigh.2018.00016/full>. <https://doi.org/10.3389/fdigh.2018.00016>
- Oettingen, A. (1913). *Das duale Harmoniesystem*. Siegel.
- Rameau, J. P. ([1722]1965). *Traité de l'Harmonie*. Broude.
- Raphael, C., & Stoddard, J. (2004). Functional harmonic analysis using probabilistic models. *Computer Music Journal*, 28(3), 45–52. <https://doi.org/10.1162/0148926041790676!>
- Riemann, H. (1877). *Harmonische Syntaxis. Grundriß einer harmonischen Satzbildungslehre*. Breitkopf und Härtel.
- Riemann, H. (1880). *Skizze einer neuen Methode der Harmonielehre*. Breitkopf und Härtel.
- Riemann, H. (1895). *Vereinfachte Harmonielehre*. Augener.
- Riemann, H. (1906). *Elementar-Schulbuch der Harmonielehre*. Hesse.
- Riemann, H. (1918). *Handbuch der Harmonielehre*. Breitkopf und Härtel.
- Rohrmeier, M. (2007). A generative grammar approach to diatonic harmonic structure. In C. Spyridis, A. Georgaki, G. Kouroupetroglou, & C. Anagnostopoulou (Eds.), *Proceedings of the 4th Sound and Music Computing Conference (SMC07)*. Athens: National and Kapodistrian University of Athens, pp. 97–100.
- Schenker, H. (1930). *Das Meisterwerk in der Musik – Band drei. Drei-Masken Verlag*.
- Schenker, H. (1935). *Der freie Satz*. Universal Edition.
- Temperley, D., & de Clercq, T. (2013). Statistical analysis of harmony and melody in rock music.
- Vogel, M. (1962). *Der Tristan-Akkord und die Krise der modernen Harmonielehre*. Düsseldorf.
- Vogel, M. (1975). *Die Lehre von den Tonbeziehungen*. Verlag für systematische Musikwissenschaft.
- Weber, G. (1817). *Versuch einer geordneten Theorie der Tonsetzkunst*. Schott.
- White, C. W., & Quinn, I. (2018). Chord context and harmonic function in tonal music. *Music Theory Spectrum*, 40(2), 314–335. <https://doi.org/10.1093/mts/mtty021>