Harmonic Analysis of Music With CCG
First Year Review

Mark Wilding
mark.wilding@ed.ac.uk

22nd June 2010

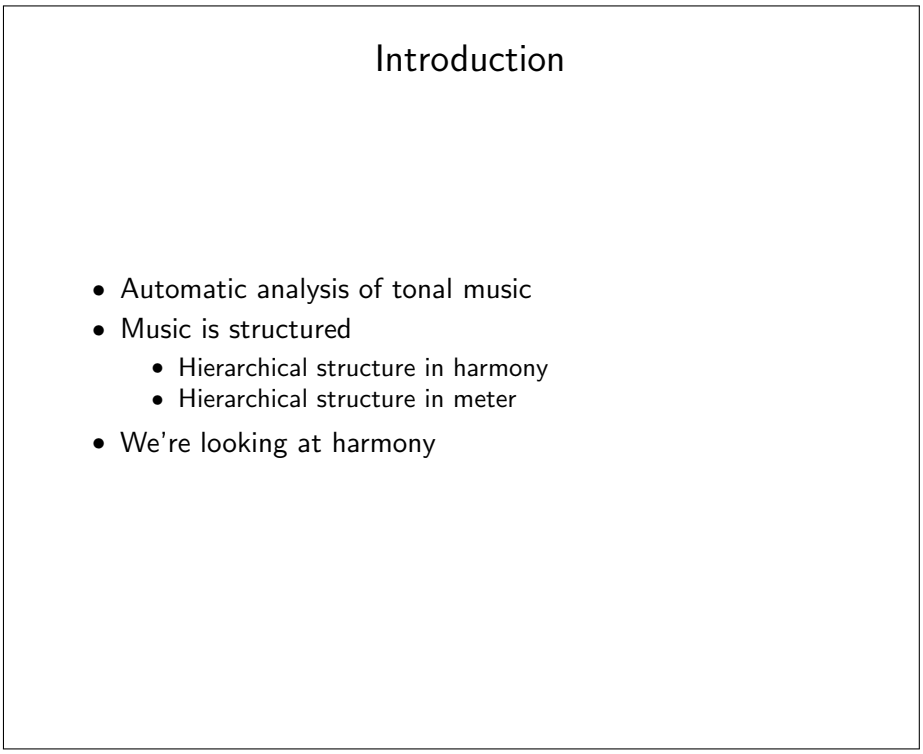
This document is based on my first year review presentation. It covers all of the topics that I covered in the presentation, including the slides from the presentation.

The presentation was given after submission of my thesis proposal, so you may wish to refer to that document for more detail on any particular topic.

During the presentation, I gave examples of chord progressions played on a keyboard. In this document I reproduce the chords that I played, so I recommend playing these on a piano or guitar, or whatever your preferred polyphonic instrument, as you encounter them. I also played some audio and video examples. I you've not received these with the document, get in touch with me and I'll supply them. There aren't many of them but I think they do a lot to help illuminate the theory in a way that's rather difficult in words.

Contents

1 Introduction



The subject of this project is the analysis of Western tonal music. Music exhibits structure in various different aspects. Two examples of this (though there are others) are harmonic structure and metrical structure.

Both of these form hierarchical structures. Harmonic structure is the structure of harmony: chords, cadences and keys and the relations between these and the notes played.

Metrical structure relates to the rhythms of played notes. Short note lengths are grouped into beats, beats into sub-bar units, these into bars and so on.

In this project, we're interested in analysing the structure of harmony automatically.

Music's Relationship to Language

- Harmonic structure similar to language's syntactic structure
- Common analogy
- Evidence of shared specialized cognitive processing
- Katz and Pesetsky (in draft):

> All formal differences between language and music are a consequence of differences in their **fundamental building blocks**... In all other respects, language and music are identical.

The structures that we see in harmony are similar to those analysed in natural language (that is, verbal human languages). The analogy between music and language is an old one that people have been drawing, in more or less abstract terms, for centuries. It's a very common thing to hear an author of a text on music theory refer to the language of music and more specific correspondances are not difficult to find.

Today there is some evidence, albeit as yet inconclusive, of specialized cognitive resources shared between human language and music processing. That is to say that there are parts of the brain specialized for the purposes of language processing which are also used in similar ways for the structural processing of music.

Katz and Pesetsky make a bold claim in their paper *The Identity Thesis for Language and Music* (in draft). They assert that the *only* differences between the two lie in their fundamental building blocks and that the structural processing that combines these in the process of interpretation is identical.

Harmonic Analysis

- Interpretation is hard
- Highly ambiguous surface form
- Needed for many musical tasks: e.g. transcription, harmonization
- We currently handle chord sequences

The interpretation of harmonic structure is not an easy task. The surface form is highly ambiguous with respect to the structures that underlie it. Depending on the task, one might take this surface form to be an audio signal, or a record of notes played. Even if one goes as far as to consider it to be the chord sequence the ambiguity, though less, is still great.

Despite this, structural interpretation is important. In fact, it is a prerequisite to many tasks in music processing, if there is to be any chance of handling the complexity of harmony that occurs commonly in, for example, jazz standards. This statement applies equally to human and computational music processing.

Common music processing tasks that rest on harmonic analysis are transcription (taking played notes and writing a score in conventional musical notation), harmonization of a melody and improvisation on a theme. In fact, it's important in any music generation task.

To give a specific example, given a melody you may wish to provide an accompaniment for it. Let's say you're happy with playing from a chord sequence. In order to produce this, you need an analysis of the harmonic structure underlying the melody.

In this work, for the moment we stick to analysing chord sequences, but in due course we hope to move on to handling music at the note level.

Background

- Bernstein's lectures, 1973
- Transformational grammars for music
- *A Generative Theory of Tonal Music* (GTTM, Lerdahl & Jackendoff): grammatical rules, transformations, preference rules
- Many approaches use preference rules
- Some others have used NL-style grammars
- Bod applied DOP to music
- Many statistical approaches: HMMs, n-grams

I do not attempt a comprehensive outline of work done in automatic harmonic analysis here, but merely intend to outline some of the different strands that have emerged.

In 1973, Leonard Bernstein delivered a fascinating series of six lectures at Harvard. Among many other things, he introduced the idea of using transformational grammars, a new idea at the time, for musical analysis. Some years later, Lerdahl and Jackendoff picked up these ideas and wrote a book, *A Generative Theory of Tonal Music*, in which they proposed a linguistic-type model for music. This combined grammatical rules, transformations and preference rules to provide a structured analysis.

An approach that many have taken has been to use systems of preference rules to model human intuitions about structure. Various people have applied natural language-type grammars to the task, from the 80s to the present. Rens Bod, for example, has applied the grammatical parsing methodology of *data-oriented parsing* to music.

There have been many purely statistical approaches explored: models like HMMs and n-grams that base interpretation of the immediate context of a note or chord. These in general perform poorly when it comes to identifying the sort of higher-level structure that we're interested in.

Few people have applied modern linguistic processing techniques in ernest to the analsysis of music. The work we're doing follows on from Mark Steedman's previous work on musical grammars.

2 Musical Syntax

Structure In Harmony

- Chords classified as *tonic*, *dominant* or *subdominant*
- Dominant creates expectation of resolution to tonic ♫
- Building block of harmonic structure
- ♫ Recursive
- ♫ Coordinated
- ♫ Substitutions
- ♫ *Call Me Irresponsible*

Chords in tonal music are classified as having one of three *functions* indicating how they relate to the current key, or tonal context. This can be *tonic*, *dominant* or *subdominant*.

A dominant chord creates a tension. It raises an expectation, calling for a resolution of the tension to a particular other chord. A tonic chord provides this resolution: specifically, the chord whose root lies a perfect fifth below the dominant chord's. The dominant chord itself may be voiced in many different ways, creating differing degrees of tension, but (given the harmonic context) always creating this expectation of a resolution. Some simple examples in the key of C major are as follows (play a C chord to establish the key in your mind first):

$$\mathrm{G}^{7}\ \mathrm{C},\ \text{or}$$
$$\mathrm{G}^{aug}\ \mathrm{C}$$

This pattern of tension and resolution resulting from a dominant chord followed by tonic is the basic building block of structure in harmony. A similar pattern, though less common, is created by the subdominant chord followed by the tonic.

Various operations allow complex structures to be built out of this. It can apply recursively: the resolution of the dominant chord, expected to be a tonic, might turn out itself to be a dominant chord, itself calling for another resolution. Thus, longer strings of dominant function chords like the following are built:

Dm^7 G^7 C, or
E^7 A^7 Dm^7 G^7 C

These sequences of dominant chords are subject to something like coordination in language. One's resolution can be delayed, then eventually provided as the resolution to another dominant chord.

(Dm^7 G^7) (Dm^7 G^7) C
(A^7 Dm^7 G^7) ($E\flat^7$ Dm^7 G^7) C

Then certain chords can be substituted: particular chords can be used in the place of others in certain contexts. The *tritone subsitution* is a common replacement of, for example, the G^7 in Dm^7 G^7 C with $D\flat^7$:

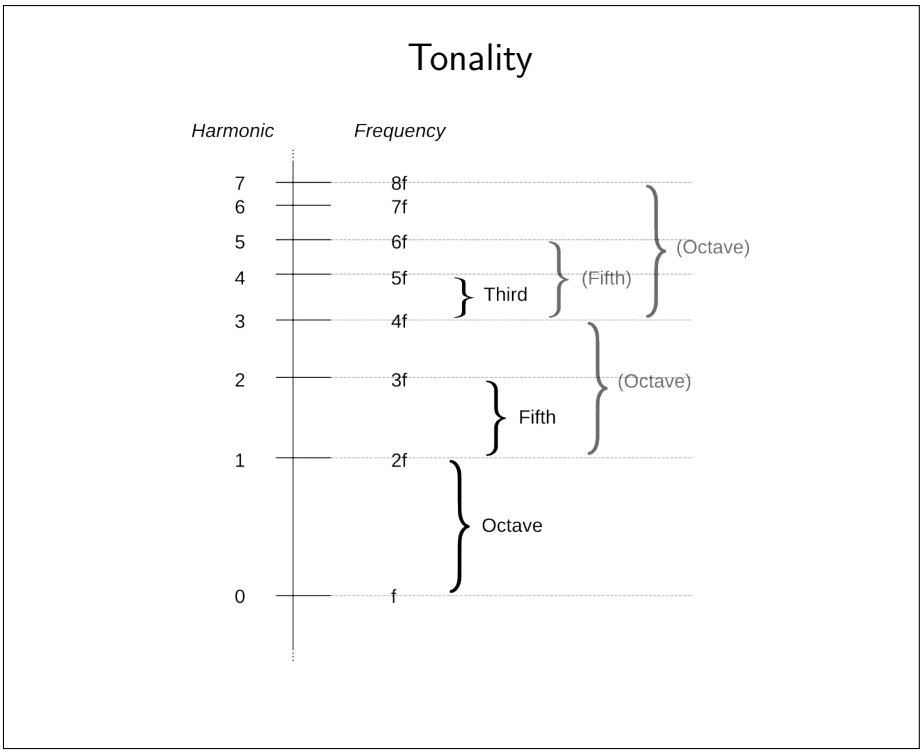
Dm^7 $D\flat^7$ C

A particularly good example of these operations in practice is the jazz standard *Call Me Irresponsible*. This has a complex harmonic structure, featuring many coordinations. In fact, the eventual resolution of the first cadence is only reached halfway through the song, and then again at the end. The result is a gradual build-up of tension as the resolution is delayed again and again.

The video `call_me.avi` shows a syntactic tree for this song, accompanied by the music, sung by Dinah Washington.

3 Tonality and Interpretation

3.1 Tonality



Western tonality (that is, the system of keys, scales and modes) is related to the naturally occurring pitch intervals between the low components of the harmonic series. These are represented in the diagram as marks on a vertical line. Each mark represents a harmonic that occurs as a natural resonance of the bottom mark, or *fundamental frequency*. Whenever a physical body is made to vibrates at the fundamental frequency, it also vibrates to different degrees at the harmonic frequencies marked on this line. This is, in fact, an infinite line in both directions, but the more distant frequencies are heard much less clearly.

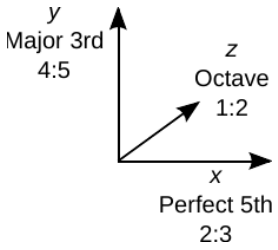
You may wish to listen to part of the harmonic series, starting at the fundamental frequency and working upwards beyond the top of this diagram: listen to `harmonic_series.ogg`.

Western tonality is based on combinations of the lowest three distinct intervals in this series, marked on the diagram with their conventional musical names. The most common chord in tonal harmony is the major chord, which consists of a note, another a third above it and one a fifth above it. So this chord is drawn precisely from the low components of the harmonic series.

The common scales that we use are derived from small combinations of these intervals into chords. The whole major scale is derived from three major chords: three chords separated by a perfect fifth. The video `just_intonation.mp4` shows these and plays the chords and then the full scale tuned according to these intervals. The result is the all of the notes of the scale tuned according to this system, *just intonation*, are related to the bottom note by a small combination of these intervals.

Tonal Space

- Longuet-Higgins formalized this as a 3D space
- Discrete space
- Project along z-dimension to ignore octaves



Christopher Longuet-Higgins drew together the work of theorists on tonality such as Helmholtz and Rameau. He formalised the theories outlined above as a three-dimensional discrete space. Each point in the space represents a pitch and no two points represent precisely the same pitch.

The basis vectors of the space are the three basic intervals: the octave, the major third and the perfect fifth. Equivalently, one step in one of the three dimensions corresponds to a change in pitch in the ratios 2:1, 3:2 and 5:4 respectively. Because any combination of these pitch ratios corresponds to a frequency multiplier expressed as a product of primes, there can be no two distinct combinations of the intervals that results in the same pitch.

At this point, I should mention a property of the first of these intervals – the octave. In any natural sound, this will be by far the loudest harmonic. In fact, the relationship between a note and its octave is so close that we can often barely distinguish the two and tend to consider the two notes in some sense equivalent. It is common when studying intervals to ignore any octaves involved in an interval and consider the interval modulo the octave. This is equivalent to projecting the three-dimensional space along the z-axis (2:1) to produce a two-dimensional space.

Tonal Space

$\sharp G^-$	$\sharp D^-$	$\sharp A$	$\sharp E$	$\sharp B$	$\sharp\sharp F^+$	$\sharp\sharp C^+$	$\sharp\sharp G^+$	$\sharp\sharp D$
E^-	B^-	$\sharp F$	$\sharp C$	$\sharp G$	$\sharp D$	$\sharp A^+$	$\sharp E^+$	$\sharp B$
C^-	G^-	D^-	A	E	B	$\sharp F^+$	$\sharp C^+$	$\sharp G$
$\flat A^-$	$\flat E^-$	$\flat B^-$	F	C	G	D	A^+	E^+
$\flat F^-$	$\flat C^-$	$\flat G^-$	$\flat D^-$	$\flat A$	$\flat E$	$\flat B$	F^+	C^+
$\flat D^{--}$	$\flat\flat A^-$	$\flat\flat E^-$	$\flat\flat B^-$	$\flat F$	$\flat C$	$\flat G$	$\flat D$	$\flat A$

A portion of the resulting two-dimensional space can be seen here. I will attempt to higlight some important properties of this space.

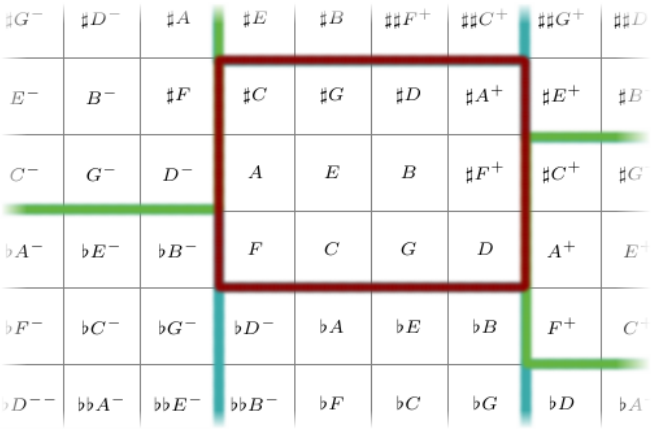
First note that this is an infinite space of which we see only a small section. Analysis in this space is conducted in terms of intervals between notes, but you could certainly attach an absolute pitch to each point simply by making an arbitrary choice of pitch to assign to one point: for example, you might conventionally declare the point named A near the centre as having a frequency of $440Hz$.

Conventional musical notation distinguishes notes separated by three major thirds (three vertical steps). For instance, the $A\flat$ close to the centre is given a distinct name from the $G\sharp$ three points above it. However, it does not give us any way to distinguish the two pitches seperated by four perfect fifths and an inverted major third (a minor sixth). Since these have distinct pitches, we name them differently: we name the central one with the conventional name and that to the left and up, which is slightly flatter, using the same name with an appended $^-$. Likewise that to the right and down, which is slightly sharper, receives a $^+$.

What makes this space useful is that is gives us a clear representation of how any two pitches are related in terms of these basic three (or now two) intervals, which are so important to human perception of pitch and resonance. If we make the (perhaps slightly overly simplistic) assumption that the intervals have equal weight, we can see how closely harmonically related a pair of points are from their distance in the space.

We consider this space to be the domain for our harmonic analysis of music. A harmonic analysis can be expressed as a mapping of the notes played onto points of the space.

Equal Temperament



- Spread 12 notes evenly across octave

The natural tuning system presented above is known as *just intonation*. It presents some practical problems as a basis for tuning musical instruments. The tuning of the notes of the scale depends on knowing the key that the music is played in. If an instrument is tuning using these intervals, music can only be played in the particular key that was chosen to tune in.

As soon as a chord from outside this key is played, unpleasant intervals are heard in it. This restricts the composer to using only one key within a piece and only chords with intervals found in our three basic chords, if he is to preserve the purity of his harmony and avoid the unpleasant remoter intervals.

Over the course of centuries, theorists grappled with this problem and proposed a huge array of alternative tuning systems that minimised the unpleasantness of theoretically remote intervals and gave greater freedom to move between keys. Eventually a simplistic solution became commonly used – *equal temperament* (ET), which is almost ubiquitously used today, despite much early opposition to the its unsatisfactory distortions of, in particular, the major third. Today we are accustomed to the sound.

In terms of the tonal space, the use of ET is effectively a limitation of the infinite space to a small space of 12 notes (central box), along with a slight distortion of all the intervals. If this box is repeated across the space, both vertically and horizontally as shown, each point is mapped by ET to the point in our central distorted box that lies in the same place within the box. Note, for example, that moving up a (not quite) perfect fifth from D brings you to A^+, which is now identical to the A found on the left edge of the central box.

Tonal Ambiguity

- Harmonic relation between notes is ambiguous under ET
- Musical notation partly distinguishes
- Must disambiguate to transcribe scores
- Harmonic relations dependent on key
- Important for generation tasks
- Corresponds to harmonic analysis done by:
  - musicologists
  - jazz performers (when improvising)
  - you (unconciously)

Music played on ET instruments leaves the true harmonic relations between the notes ambiguous. Interestingly, musical notation still partly distinguishes this ambiguity, as I mentioned when describing the notation of the tonal space. The black note on a keyboard that falls between G and A may be written as $A\flat$ or $G\sharp$.

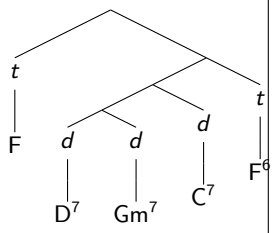
One consequence of this is that we need to disambiguate the harmonic relations behind the notes played (at least partly) to be able to write notes on a score. Furthermore, this disambiguation is a part of the process of identifying the key. It's important for any musical generation task, such as the generation of a chord sequence to accompany a melody.

Disambiguation in the tonal space corresponds to the the sort of analsysis done by a musicologist studying the harmony of a piece of music; and to the that done by jazz performers when improvising; and even to that done unconciously by everyone when listening to music.

3.2 Syntax

Syntax

- Need harmonic structure to establish harmonic relations between chords
- Map notes in ET space back onto infinite space
- Syntax → harmonic structure
- Semantics → tonal space interpretation



And so back to the issue of musical syntax. By understanding the harmonic structure of a piece, we are able to work out the harmonic relations between its notes and chords. In other words, we can map the notes played in the little equal temperament box back onto the underlying infinite tonal space.

In our musical grammar, we analyse the harmonic structure as musical syntax. Our semantic interpretation, or at least the start of it, involves a mapping back into the full tonal space. There we can see face to face the true harmonic relations between the notes.

4 Our Grammar

Our Grammar: Syntax

- Modification of NL CCG
- Atomic categories of form:

$$X{-}Y$$

- Important syntactic information: start and end chord roots
- E.g. simple tonic chord on C interpreted as:

$$C{-}C$$

- Just one chord – start and end

I'm going to assume a basic understanding of CCG here, though not a great deal is required. Our grammar formalism for music is based on standard CCG as used for NL purposes, though we've extended it with some special category notation and features for music.

A category is assigned to each word in the input (in this case a chord) and the sequence of categories are combined using a set of combinatory rules to produce a complete derivation.

Let's begin with the simple atomic category. Unlike in normal CCG, the atomic category is not quite atomic: it's split into two parts. These tell us the chords that begin and end the span covered by the category. This is the important information we need to know about the span to know how it can combine with the surrounding categories.

Here's the simplest example: the simple tonic chord, in this case rooted on C. The category is trivial: the span begins and ends on the same chord, so has C in both parts.

Our Grammar: Syntax

- Complex categories built up with slashes
- ♫ Simple dominant chord on G expects resolution to C

$$G{-}X \;/\; C{-}X$$

- Next category must begin at C

$$\dfrac{\dfrac{\mathsf{G}^7}{G{-}X \;/\; C{-}X} \quad \dfrac{\mathsf{C}}{C{-}C}}{G{-}C}\,>$$

- Interprets this chord as a movement from G to C

Complex categories are built up as usual in CCG, with forward and backward slashes. Recall the dominant chord: it creates a tension and expectation of resolution to a tonic rooted a fifth below.

Here's the complex category for that interpretation of a chord. It's a forward slash, since it's looking for an argument that follows it. It only specifies that the category that follows need to start on the resolution of this chord – in this case a C. Given such a category, it will produce a category that begins on G (this chord's root) and ends wherever the subsequent span takes us.

You can see here the simple derivation of a cadence built from a dominant tonic and its resolution. CCG's basic primitive rule – application – combines these two categories to interpret both chords together.

These are the simplest of the lexical items. The lexicon contains more complex categories for interpreting different structures and substitutions. In addition to the basic CCG rules of application (as used above) and composition, a couple of special musical rules allow us to build complex structures and combine multiple cadences into a piece.

Our Grammar: Semantics

- Semantics: path through tonal space
- λ-calculus terms on lexical entries
- Represented as a list of points in the space
- Tonic: single point in space

  $C{-}C \quad : \quad C :: Nil \qquad [C]$
- Dominant: one point moving to another

  $G{-}X \;/\; C{-}X \quad : \quad \lambda x.G :: x \qquad [G] + x$

  $\lambda x.G :: x \quad C :: Nil \quad \Rightarrow \quad G :: C :: Nil \qquad [G, C]$

The semantics of a chord sequence under a particular interpretation of substitutions and functions is the path through the tonal space that is followed by the chords' true underlying roots and the functions of the chords. We represent the semantics as a list of points in the space. Each point also specifies the function of the chord (not shown in the examples on this slide).

Each lexical category gets a semantics. The λ-calculus allows us to specify how these should be pieced together during derivation to produce the final path.

Here I add the semantics to the two categories I've shown the syntax for already. The tonic chord's semantics is trivial again: it's a single chord with a single root, so the semantics is a one-item list.[1]

The dominant category receives a functional semantics. The lambda-abstracted variable will be filled in derivation by the argument's semantics. This category just prepends its own tonal space point to this list.

At the bottom of the slide is the combination of the two categories' semantics into the result by application.

Now is a good time, if you have it, to watch the video example of a tonal space path in `basin_street2.mp4`. The recording is of Louis Armstrong playing and singing the Basin St. Blues, with a live tonal space path. This example is especially interesting because the cadence of each line leads us to a new point in the space which is equated with the original key's tonic by ET, but which in fact should be slightly flatter if justly intoned.

[1] The list representation is of the standard form for functional lists: a head item combined by the :: (`cons`) operator to another list. I show the semantics in an alternative list notation on this slide as well.

Ambiguity

- Huge lexical ambiguity
- Generally use only a few chord types
- Lots of possible structures
- Big search space for parsing
- Sparse data
- Need some good modeling

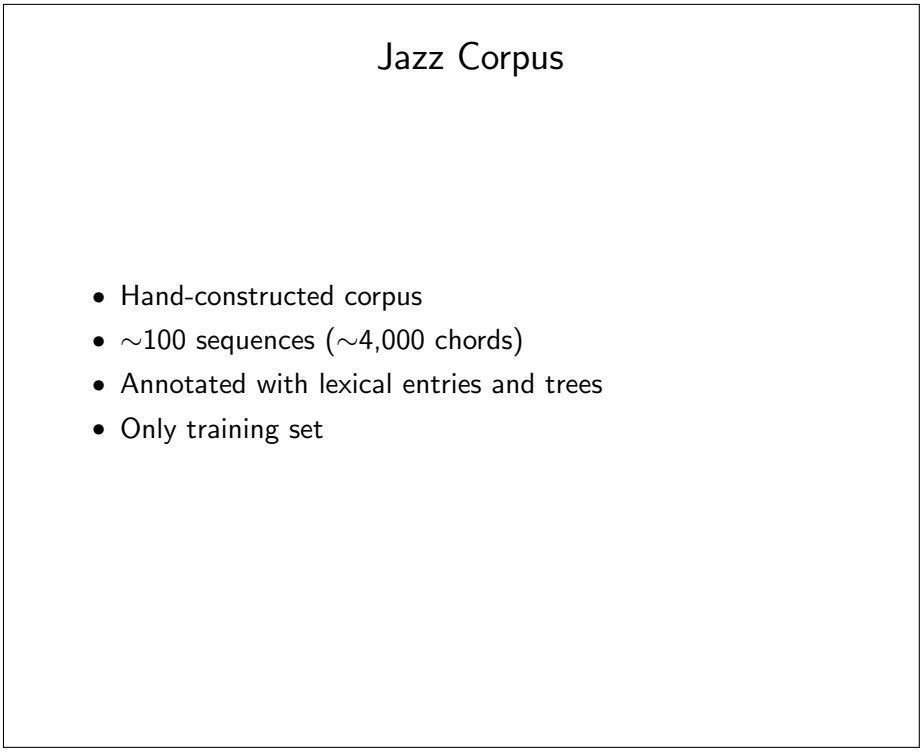
Harmony has a huge ambiguity of interpretation at the lexical level. That is to say, any one chord could play many different roles in the structure.

In ET, there simple aren't very many notes and the are only a few chord types that get commonly used. In fact, most of the chord types found in jazz lead sheets can be condensed into a small group of chord type classes, within which types can be freely interchanged.

It is perhaps informative to think about this in relation to NL and its ambiguity at the same level. Imagine a language with only 12 words. It has a bit of inflectional morphology, but this is highly ambiguous – a given affix doesn't very much narrow down the possible syntactic types. On top of this, words can commonly be interchanged for particular other words in specific contexts. And then, there a variety of recursive syntactic structures that can be used.

In any case, the search space involved in parsing is huge. There are many rare interpretations that a possible and do occur, but which are very rarely seen in a data set. As a result, to be able to parse anything practically, the grammar's going to need some pretty good statistical modeling.

5 Corpus



I am in the process of constructing a corpus of jazz chord sequences, annotated with interpretations. It is quite a small corpus, but we hope sufficient to train some models on to start with. It consists currently of about 100 chord sequences, amounting to roughly 4,000 chords.

Each chord is annotated with a lexical entry from the lexicon of the jazz grammar. In addition to this, a small amount of further information annotated on each sequence means that the annotation contains a full canonical CCG derivation tree for each sequence[2].

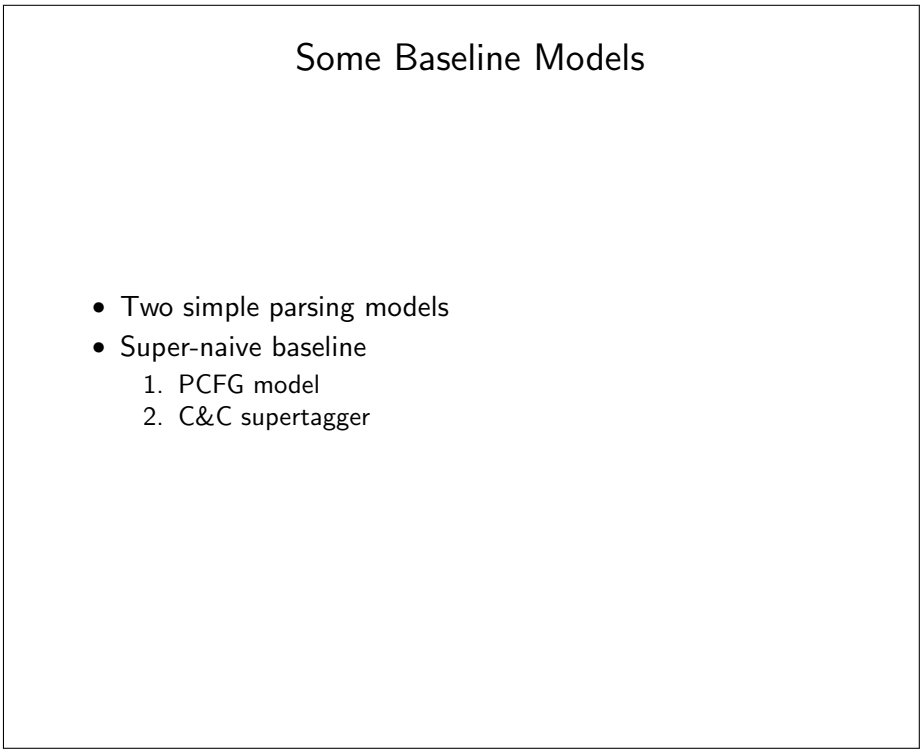
Currently, the corpus consists of one set, which is being used as a training set. In future we hope to add a separate test set, but for the time being we evaluate using cross-validation on this single set.

There are still some gaps in the annotation – in general just one or two chords in any one sequence with no interpretation. This is either because I am uncertain of the correct interpretation, or, in most cases, because new categories need to be added to the grammar before it can produce the required interpretation. Although the gaps are small, removing any sequence with incomplete annotations (and therefore no complete gold standard derivation) leaves only about 65 sequences in the corpus.

[2]The only information required beyond the categories is markers of coordination. The corpus contains this information and the implicit derivation tree can be easily constructed.

6 Baseline Models

6.1 Models



I've implemented two simple statistical parsing models to establish a baseline. Both models are very naive. Their main purpose is to demonstrate the application of this kind of model to the task of parsing chord sequences.

The first model is a generative model of derivations that is similar to a probabilistic context-free grammar (PCFG). The second performs a full parse on a limited selection of lexical categories, selected using a supertagging technique.

Aside: Absolute Pitch

- Absolute pitch irrelevant to interpretation
- More or less irrelevant to perception
- Models should be insensitive to absolute pitch

Before describing the models in more detail, I'd like to make a brief aside regarding absolute pitch.

In general, the absolute pitch at which a piece is played or notated carries almost no information that's relative to interpretation. The only thing of interest for interpretation is the relative pitch between notes, chords, keys, etc.

It is clear that absolute pitch is irrelevant for a model of musical cognition: most humans do not have perfect pitch, so without a reference point have very limited ability even to recognise the absolute pitch of a note.

If our models are based on the absolute pitch of the input, we artificially increase data sparsity, since we distinguish between inputs that would not be distinguished in analysis or by human listeners. Therefore it makes sense to design any model to be insensitive to the absolute pitch of the input.

Model 1: The PCFG Model

- Generative model based on Hockenmaier (2001)
- Frequentist model of derivations
- Only naive smoothing
- Beam applied to chart during parsing
- All pitches relative to parent in expansions

The first model is a simple generative model based on Julia Hockenmaier's PCFG-style models for parsing CCG for natural language. It is a model of derivation trees. It assigns probabilities to expansions at nodes in the tree based on counts of expansions in the training set.

The model uses only very naive smoothing for unseen expansions – effectively add-one smoothing. Parsing is performed bottom-up. A harsh beam is applied to each arc in the chart, keeping only the most probable categories.

The chord root symbols in the categories are taken relative to the pitch of the parent of the expansion when computing the probability of an expansion. This means that the model is insensitive to the absolute pitch of the chords, taking into account relative pitch only at each node in the tree.

Model 2: Supertagger Model

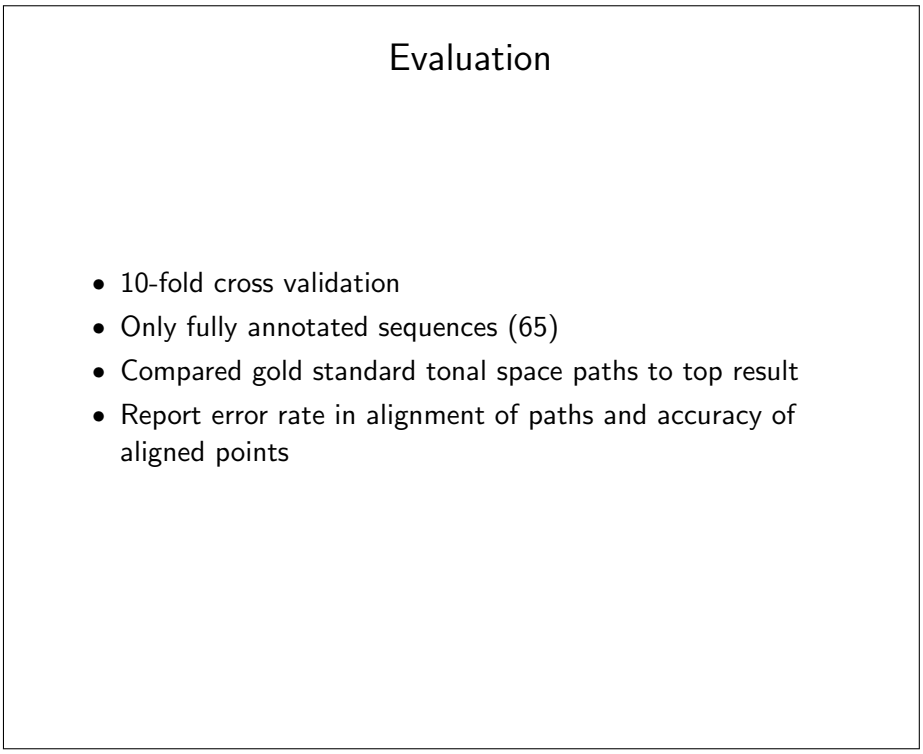
- Uses supertagger component of the C&C parser (Clark and Curan, 2002)
- Chooses categories for chords using log-linear model
- Parser asks for less probable tags if it can't get a full parse
- Observations: chord types and intervals between chord roots

The second model uses the supertagger component of the C&C parser of Clark and Curran. A supertagger model works by using a short-range statistical model of lexical categories to select a subset of the lexical items for each word in a sentence. These are then parsed bottom-up to provide a full interpretation. If no full parse can be found with this small subset of possible lexical categories (tags), lower probability ones are added to the chart until a parse is found.

The categories are chosen using a log-linear model. For this model, we simply trained the C&C supertagger component and used it to select categories at the front end of the parser. The supertagger returns possible categories, along with their probabilities. The parser adds these to the chart in batches of similar probabilities.

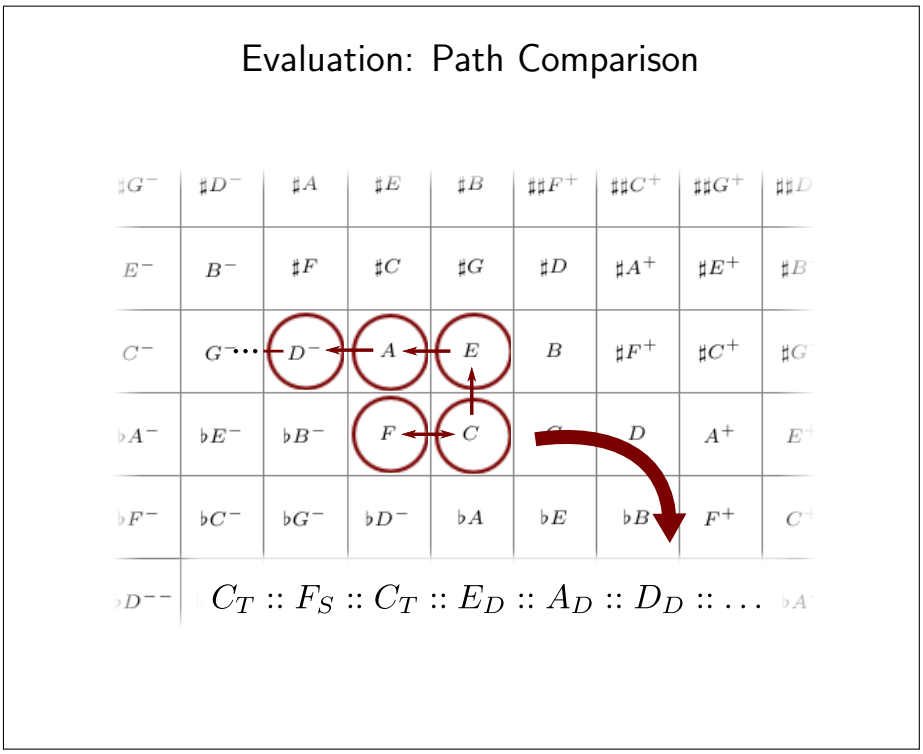
The observations of the chord sequence that we used as input to the supertagger were preprocessed to take the form of intervals between chord roots, plus the chord type, rather than the absolute chord root itself. This makes the model insensitive to absolute pitch.

6.2 Evaluation

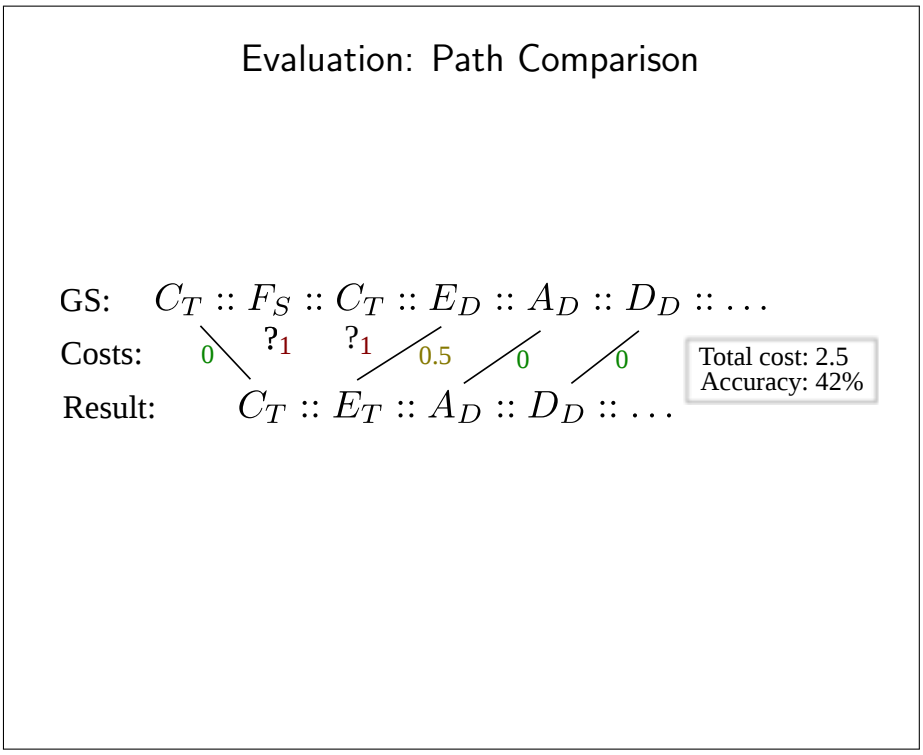


As described above, the corpus currently contains no test set, but only a single training set. An initial evaluation of the models was performed using 10-fold cross validation on the training set. We used only those sequences which are fully annotated ($\sim$65).

We evaluated by comparing the tonal space paths of the highest probability result and the gold standard annotation. We report the results as an error rate in the alignment of the two paths and the accuracy of aligned points in the path.



Analysis consists of a path through the tonal space. This is represented as a list of tonal space points.



The two paths are aligned. Note that they may not necessarily be of the same length. It is possible for a chord to be interpreted as an elaboration of the same root as the previous chord: a common example is the *I IV I* motif (e.g. C F C), which is used as a colouration of a simple *I* (in tonic position). We can therefore not assume that two analyses have the same length, even if the input sequences were of the same length.

We compute the edit distance between the two paths. The only difference between this measure and the standard definition of word error rate is that we allow two elements to be partially aligned (as in the case of the E_D and E_T in the example. There are two pieces of information contained in the analysis for each point – the chord root and its function. Since these are rather independent it seems fair to attribute an interpretation only a half cost if one of them is correct but not the other. The cost is then averaged over the length of the gold standard sequence to get the error rate.

We also compute an accuracy score between 0 and 1 by averaging over the length of the longer of the two sequences.

Results

	C&C supertagger	PCFG
Coverage	21.4%	100.0%
Mean accuracy	90.8% (10.0%)	59.8% (11.9%)
Mean error rate	0.10 (0.11)	0.41 (0.13)

- PCFG: parses everything, but not that well
- C&C: terrible recall

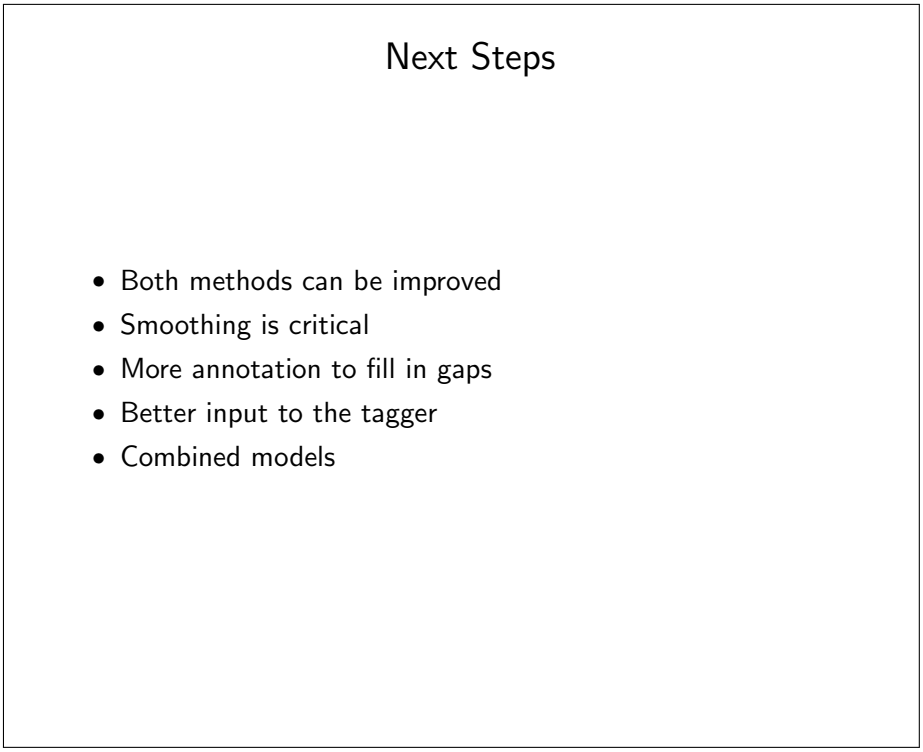
The PCFG model produces some full parse for every sequence, but doesn't perform very well. The supertagger model produces interpretations of few sequences, but performs well when it does.

The error rate, as described above, can be thought of as the number of mistakes made per point in the gold standard. The PCFG model makes 2 errors in every 5 points, whilst the supertagger model makes 1 in every 10. Numbers in brackets are standard deviations.

The PCFG model can parse all sequences. The poor performance is unsurprising: with such naive smoothing we would expect it to produce only simple analyses using the most common expansions. Interestingly, it usually gets either the function or the root right. We expect slightly better smoothing to improve greatly on this result.

The supertagger's poor coverage is also unsurprising. Sentences (chord sequences) are very long in comparison to natural language – usually over 40 chords, and up to 95. A few localised mistakes, where the tagger fails to assign a suitable category to a single chord, can prevent the full parse from succeeding. Currently, the tagger never returns very rare categories, since they only appear a few times in the training set, but these are sometimes critical to acheiving a parse. We can expect some improvement on this result simply by better use of the supertagger. We may find that the tagger's model, which is optimised for use on natural language, cannot perform well on this data.

7 Next Steps



Both of the parsing models can be improved by some simple modifications, as suggested above. Smoothing is a major issue, since our data is so sparse. We can expect to gain a lot by using improved smoothing and we can expect the quality of the smoothing to be one of the biggest concerns in producing an effective model.

One of the immediate outstanding tasks is to complete the annotation of the sequences currently in the corpus. Many sequences had to be omitted due to small gaps in their annotation. This is a time-consuming task, because it is the most difficult to interpret parts that are now left and many will require interpretations not currently provided by the lexicon.

An approach that could yield good results is a combined model: it would use a supertagger non-aggressively to narrow down the lexical categories and then parse these using a PCFG model, potentially with a less aggressive beam.

8 Plan

Plan

1. Devise evaluation methods
   - use them to do consistent evaluation of baseline methods already implemented
   - evaluate improvement of better models as they are developed

   Months 10-11
2. Enhance the baseline models with more sophisticated smoothing 11-13
3. Consider alternative annotation schemes. Possibly get more data 10-14
4. Try more sophisticated statistical models 14-20
5. Investigate annotator agreement. Experiment with other annotators 14-20
6. Investigate semi-supervised methods 20-24
7. MIDI realization models 20-34
8. Use best full models (tonal space interpretation from MIDI) to implement some applications 25-34
9. Write up 31-36

This slide sets out the general plan for the remainder of the project.