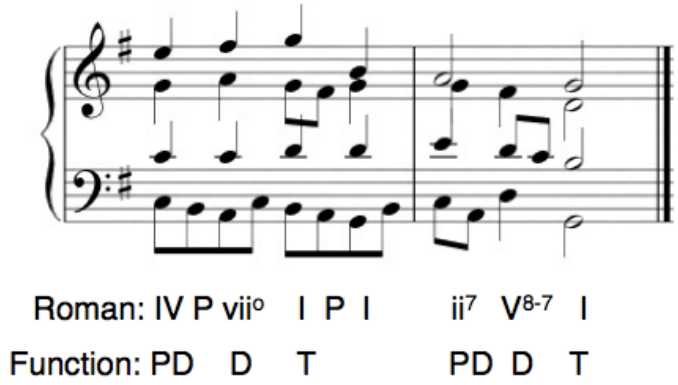
Example 1: BWV 115.6, mm. 3–4



Example 1 shows measures 3–4 of BWV 115.6, the chorale *Straf' mich nicht in deinem Zorn*. The top row analyzes the chords from a traditional Roman numeral perspective, with a "P" indicating passing chords. The second row analyses these chords' functions, showing two rotations through the traditional Predominant-Dominant-Tonic cycle.

The passage exemplifies several assumptions underlying function theory. First, and most basically, the theory treats tonic, dominant, and predominant as primitive, unrefinable categories that act as equivalence classes. In other words, the lower row interprets the musical surface as elaborations on or prolongations of these three functions, and groups two bars as rotating through the same general equivalence classes, with – for instance – the IV and the ii^7 as participating within identical functions.

Several other more subtle equivalencies are at play in the analysis as well. For one, two different chords are bundled under the same "passing" symbol, suggesting that these chords exhibit equivalent functions. Additionally, by treating the addition of the seventh in the penultimate chord as elaborating a single Roman numeral and a single function, we equivocate the D major triad with the D major-minor seventh chord.

The passage also allows for introspection towards how functional categories are assigned. In general, we can imagine the criteria used to categorize chords as tonic, dominant, or predominant – to be accounted for by two polar forces: a chords' *context* and its scale-degree *content*.[1] At the one extreme of the dipole, chord prototypes – harmonic ideals – define functions. From this perspective, tonic, predominant, and dominant are all best represented by the triads I, IV, and V; musical surfaces can be understood as transformations or elaborations of those primary chords. Using this logic, the second measure's ii^7 chord can be recognized as a predominant chord because it shares more scale degrees with IV than any of the other functional pillars.

At the other end of the dipole, harmonic function is conceived of analogous to a theatrical "role": the part remains essentially the same while different actors fill that role From this perspective, the three functions are fundamentally identified with how they act in music – where they occur within a phrase, what other functions they progress to or transition from, and other contextual cues. Using this logic, the both measures begin as

[1] (Harrison 1994: 39-40).

predominant because of their positions preparing dominant chords, which themselves are defined by their positions preparing tonic chords, and so on.

While three-function theory is as useful, intuitive and explanatory as it is widely accepted, this article investigates and formalizes a corpus-based model of harmonic function, allowing us to question and test these assumptions within various data sets. Using a Hidden Markov Model (HMM), an algorithm that identifies contextual categories of objects within streams of observations, we track chord behaviors in a corpus of Bach chorales, common-practice free composition, and pop/rock music from Billboard Top 40. We will investigate whether a purely contextual model – i.e., one that does not take a chord's pitch or scale-degree content into consideration – is sufficient to create functional chord classes within this corpus and by studying the results of a purely contextual model and its differences with our music-theoretic intuitions, we will begin to see how chord content influences traditional notions of harmonic function. Furthermore, we will investigate whether the three traditional function are indeed nonrefinable categories, focusing particularly on the role that passing chords, non-tertian sonorities, and that applied dominants play in the model. In what follows, we first present our model, discuss the results drawn from the Bach chorale corpus, and then discuss how the results change when using a larger common-practice corpus and when refocusing on a corpus of contemporary popular music.

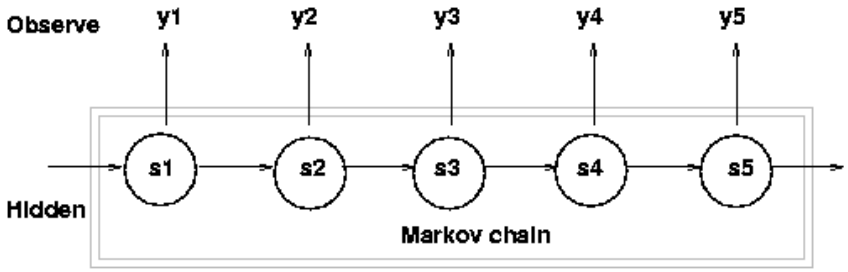
How the Model Works

Hidden Markov Models (HMMs) process some string of symbols – or, *observations* – and then group those symbols into equivalence classes – or, *hidden states* – based on contextual regularities within that string. In other words, given an observable series, the model associates each event in that series with an unobservable "hidden" state.

To harness some intuitions concerning observations and hidden states, consider this simple example: after missing last night's game on television, you arrive at your local coffee shop wondering whether your home team won. When you look around, you are struck by how many people are wearing your team's jerseys, and come to the conclusion that they are celebrating a win. Here, using observations, you can make an educated guess about some hidden piece of information about your team's performance. The jerseys are the observation, the team's performance the hidden state.

More specifically, the algorithm balances *emission* and *transition* probabilities to underlay observations with hidden states. As seen in FIGURE XXX, emission probabilities show how likely an observation is to be associated with – or, emitted by – a hidden state. Transition probabilities, then, capture how likely the hidden states are to be adjacent to one another. In the figure, each hidden state *s* emits some observation *y*. The arrows between each *s* and *y* represent the emission probabilities. The transition probabilities are determined by how probable each *s* is given the *s* that occurred before it, here shown by the horizontal arrows. For instance, we can talk about how likely s_3 is to occur given the s_2 that happened before it, and then how probable s_2 was to occur given the s_1 that preceded it. These overlapping probabilities are referred to as a *Markov Chain*, as noted in the figure.

Figure XXX (*remake later*):



If emission and transition probabilities are known, the algorithm can analyze a series of observations, assigning the most likely hidden state given the series. This process is analogous to that undertaken in Example 1's functional analysis: given our intuitions about what (observed) chords are associated with what (hidden) functions (i.e., approximating the emission probabilities) and given the proscriptions about what functions proceed to and from what other functions (i.e., approximating the transition probabilities), we assign functions to the harmonies.

If the probabilities are not known, the algorithm can estimate them, essentially finding the best way to fit for some number of states to an observation series. Here, two steps are required. First, one must set the number of hidden states; while one can compare the models resulting from different numbers of states, the HMM process itself requires this parameter to be fixed throughout. Second, the process uses a technique called *expectation maximization* to find the optimal fit between hidden states and the observations. Expectation maximization begins with randomly assigning transition and emission probabilities, and then tweaking those assignments to maximize the fit between the model and the observation string, throwing out worse solutions and retaining better solutions until an optimization is found.

As an example, consider Toy Corpus #1 in Example XXXa. In the 2-state solution, the model creates a category *A* that has a 100% probability of emitting I chords, a 50% probability to transition to itself, and a 50% probability of transitioning to state *B*. *B*, one the other hand, emits V half the time and V^7 the other half of the time; *B* transitions to *A* 100% of the time. The probability for the sequence would be the product of each emission and each transition (shown in the example). Compare this model Example XXXb, and the solution labeled "Poor 2-state." Here, our toy solution now groups V^7 with I in state A, and both states move between one another with equal probability. The transition and emission probabilities of observed and hidden states given the model are lower (worse) than those in the Good 2-state solution. Because the good solution has a higher probability than the poor solution, an expectation maximization process would discard the latter in favor of the former. In our modeling, we will prefer solutions like Example XXXa over Example XXXb for this reason.

Example XXXa: Toy Corpus #1, with Good 2-state solution

Observations:	I	V^7	I	I	V	I	I	I	V^7	I	I	V	I	I	I	V^7	I				
Good 2-State:	A	B	A	A	B	A	A	A	B	A	A	B	A	A	A	B	A				
Emission Probs:	1	.5	1	1	.5	1	1	1	.5	1	1	.5	1	1	1	.5	1	=	.0625		
Transition Probs:		.5	1	.5	.5	1	.5	.5	.5	1	.5	.5	1	.5	.5	.5	1	=	.00049		

Emission Probabilities:

State		A	B
Emitted Chord	I	100%	0%
	V^7	0%	50%
	V	0%	50%

Transition Probabilities:
(probability of row moving to column)

	A	B
A	50%	50%
B	0%	100%

Example XXXb: Toy Corpus #1 with Poor 2-State Solution:

Observations:	I	V^7	I	I	V	I	I	I	V^7	I	I	V	I	I	I	V^7	I				
Poor 2-State:	A	A	A	B	B	A	A	B	B	A	B	B	A	A	B	A	A				
Emission Probs:	.8	.2	.8	.8	.2	.8	.8	.2	.8	.8	.2	.8	.8	.8	.2	.2	.8	=	.0000055		
Transition Probs:		.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	.5	=	.0000153		

Emission Probabilities:

State		A	B
Emitted Chord	I	80%	20%
	V^7	20%	0%
	V	0%	80%

Transition Probabilities:
(probability of row moving to column)

	A	B
A	50%	50%
B	50%	50%

While the comparison between the Good and Poor 2-state models illustrates how the algorithm chooses between different possible solutions, our process also must choose between different numbers of hidden states. That is, how would we know that 2 states optimally underpin the observation string, as opposed to 3, 4, or more?

Consider the two hypothetical Good 4-state solutions in Example XXXc. Both distinguish between I, V^7, and V as different functional entities (states A, B, and C), and both identify the fourth state, D, as one which emits I chords that occur in particular relationships to states B and C. In the first model, D occurs before B or C, and in the second it occurs after. If the emission and transition probabilities were constant between

both models and the latter only differed from the former in whether D proceeds to or from B and C, the probability of the sequence would be identical between both models. It would not be clear which model represented the better option for an expectation maximization procedure; because there is no obvious solution, the algorithm would sometimes result in the first solution, and sometimes the second. This lack of consistency suggests that 4 is not an ideal number of states to underlay this observation sequence. In our modeling, we will prefer solutions like Example XXXa over Example XXXc for this reason.

Example XXXc: Toy Corpus #1 with two Good 4-State Solutions:

Observations:	I V^7 I I V I I I V^7 I I V I I I V^7 I
Good 4-State #1:	A C A D B A A D C A D B A A D C A
Good 4-State #2:	A C A A B D A A C D A B D A A C D

In sum, optimal numbers of states are those which return consistent models – i.e., there is a single optimal solution toward which the expectation maximization procedure can aim. Optimal solutions, then, are those assign the highest relative probability to a sequence of observations when applying the model's transition and emission probabilities. (Appendix XXX produces the math formalization of Hidden Markov Models and the expectation maximization procedures used in this study.) In what follows, we find the ideal numbers of hidden states and the properties of the optimal solutions that result from applying this procedure to various musical corpora.