[SQUEAKING]

[RUSTLING]

[CLICKING]

MICHAEL CUTHBERT: I want to wrap up a tiny bit of problem set 7. But before we do, I want us to jump ahead to the second part where I say algorithmic composition for a Markov data set. If you could get your-- what do you call it-- your Jupyter Notebooks going. And now, we're getting to the point in the semester where I'm just going to say you can import everything from music21, but I will give you a few restrictions from time to time. So I don't really care what you're doing.

And in the past, I've tried to have everyone download a pre-prepared set. And it's always had problems. So we're just going to do something that takes about 3 or 4 minutes of your processor time and get that started now.

So this will probably take a second or two. You're learning a new term, corpus.search Jig. I can't remember if it's important if this is capital or not. But let's say I put it on. And this will go through music-- so music21 comes with about-- I can't remember what the number is-- something like 14,000 pieces when you do it, just so you can do it.

And this will get everything that calls itself jig. I don't know if there's any word that begins with jig that is not a jig, but this might be garbage in, garbage out. I'm getting 327. Depending on what version you're on, you might be slightly higher or slightly lower. It won't influence that too much. Great.

So give me a quick-- actually, the easiest thing so I can tell is, yeah, up or a snap of a finger when you've got it. And then I can hear.

[SNAPPING]

OK, good. All of our computers are slightly different speeds. But I think you can also start typing the next expressions while you're doing it. So Jigs is something called a metadata bundle. That means it's a collection of everything whose metadata, whose titles, authors, something match a particular search term. You can look through. You can find things by key, by starting meter, and things like that. But often searching by title or composer is one of the first things.

And what you can do is if we look at this as a very short introduction to this, jigs is 327 things. We can look at the first one with jigs(0). And we can get that it comes from some particular piece, song 47, in a collection of Irish tunes called airdsAirs_book1. So it's not a piece.

In order to get that, we can say-- I'm going to go j1. I'm going to equals jigs(1), because jigs(0) has some funny things on it. And jigs(1).parse, calling it as a method, and that will actually load it, because, otherwise, loading 327 pieces would take a little too long. And then I can do show, hopefully. "A Spanish Jigg"! Everyone get "A Spanish Jigg" on theirs?

OK, good. So what we're going to do is we're going to just get every jig that is in G major. So I'm going to call it-- because I have my verbose variable names. I'm going to call it jigs_in_g with underscores. You might just call it jg, whatever you want.

And I'll just say for. I'll say j_meta in jigs. So I remember this a thing. I'll say j. Sorry. I'll go slower. I'll make this bigger. Slower and bigger are always important. I have a whole big screen if I'm not using it, so a little bit better.

So I'll parse each one, the metadata. So we'll say j-- is that one parsed? And now, I'll do-- just kidding, jk for jigs key equals j.analyze the key. And here, just for now, so we can all get caught up, I'm going to print them all.

I'm just letting everybody. And then I'm going to be grabbing in a second every piece that is in G major. Why? Because it looks like there's a lot of jigs in G major. So it seems like a good thing to grab. Do we feel-- we'll just keep snapping when you're caught up.

[SNAPPING]

OK, good, good. We'll eventually synchronize on a rhythm for the things. Good. Now, I'll say tonic_name. I'll just be verbose again. I keep jumping back and forth. Is that keys .tonic, which returns a pitch object. And a pitch object has a .name.

And what we'll do is we'll get all the ones in G major or in G minor. But no, actually, maybe we should do-- nah, we'll just do it all in-- if tonic_name equals G-- I don't think there's a lot in G minor-- jigs_in_g.append(j).

And that's where we're going to put this aside for a bit. But you can do the snap when you're done, because this will take a while to do. And then I'll move on to the next thing. Don't snap when it's done. Snap when you're done programming this, so that we have it for later. Good.

Now, by the way, mine's running a lot faster than yours, not because MIT has bought me a fancy computer, but because when you parse something once in music21, it stores it as a cache somewhere on your hard drive. And so it goes faster the second time. So if you're feeling like, ah, I need to upgrade my computer, because mine was so much slower than his. No, it's just I may have-- like they do on the cooking shows, I might have already baked this once this morning, just to make sure it works. OK, good.

Put that aside for a little bit. I want to open up-- actually, well, I don't need to put that aside too much-- your problem set 7. Where did that go? Here it is. Sorry about that, y'all. I'm usually a little bit more organized. Great.

Problem set 7-- oh, here we go, 7 and 8. So I want to just talk through some of the things that you've just been working on. A couple people have requested and been granted extra time for the last couple questions, so I'm not going to go through them. And besides, we all have different answers on the open-ended ones.

Actually, actually, yeah, no, we'll do this. So the major_scales one, how hard did we find this particular one? Easy, hard, a little bit-- what was the-- what do you think a test case that I'm going to run on that you can already tell me? Yeah.

AUDIENCE: You're going to do something [? pathological, ?] like E-flat, flat 54.

MICHAEL CUTHBERT: E-flat, flat-- what's that?

AUDIENCE: You're going to do something [? pathological. ?]

MICHAEL CUTHBERT: Major. Yeah, I'm going to probably start with E-sharp major, or even-- yeah, so certain things like that to make sure. Or even a normal thing, like F-sharp major, what am I going to be looking for? What's the leading tone in F-sharp major?

[INTERPOSING VOICES]

MICHAEL CUTHBERT: E-sharp, F natural, E double sharp. Who goes for E-sharp? F natural? E double sharp? Yeah, I'm going to be looking for E-sharp, because a major scale, one of the things about it is it uses every one of the lines and spaces once and only once, regard-- using some sort of sharp or flat. Great.

Now, let's talk a little bit about the second question. Any questions that you had on the routes? Depending on how much you chose to do this by hand, which when we didn't combine the two problem sets, I'd like to use this one as something to do by hand. It gets quite a bit harder. What are some of the techniques that anybody remembers using on this? Yeah.

AUDIENCE: You want everything to be between 0 and 11.

MICHAEL CUTHBERT: Ah, convert everything to 0 and 11.

AUDIENCE: Well, for now, I guess.

MICHAEL CUTHBERT: For now. Great. And then-- so we have the pitch class space. Maybe I should-- do I have a side? Yeah, there we go. Oops. So 0, 1-- is this what you're thinking of?

AUDIENCE: Yep.

MICHAEL CUTHBERT: 2. Great. So that can be one part of it. And then what do yo-- what do we do-- how are we going to use this? Somebody else. Yeah, go ahead.

AUDIENCE: [INAUDIBLE] I had an idea, but I also added like a set( ), so just repeated notes.

MICHAEL CUTHBERT: So if there's repeated notes-- yeah, so we're going to use the set( ) operation-- good-- to remove things that are ir there multiple times. What's the-- yeah, so for root-- for the root of the chord, does it matter what octave something's in?

AUDIENCE: No.

MICHAEL CUTHBERT: No. So root is a kind of equivalence class that implies octave equivalence, right? Great. So we're going to be using a set. We're going to be discarding octaves. No octaves. Great. What are other things that you did in this? I just want to spin it around. Yes. Go ahead.

AUDIENCE: Oh, mine is similar, but I used the diatonic note numbers instead.

MICHAEL CUTHBERT: So you used diatonic numbers instead. Diatonic note numbers, so that you have something like-- and I can't believe there's a brain cell in my head that remembers this. So you had something like 31, 32, 33, 34. Great. The number C4 as 29 is pretty arbitrary.

I just like-- if you go to the Media Lab, they have a Bösendorfer Imperial Grand, one that has the extra keys down here. And the lowest note on that one, I assigned to 1, and just went up from there. So C4 ends up being 29. Good. And so we can use that for certain parts.

Who used diatonic note numbers or something like that in their example-- in your answer? 1, 2, 3, 4. Who used something like pitch classes? That's good. Good. Who used both? Anybody? My implementation ends up using both of them. So I'm going to say both of these can be a right way to work. Good.

Now, what are some other techniques that we did on rootness? Yeah.

AUDIENCE: Should I check for fifths?

MICHAEL CUTHBERT: Check for fifths-- good. Check for fifths. What's the best-- which one of these, do you think, works better for checking for fifths?

Do you want to take a guess?

AUDIENCE: I wouldn't want to say either, just because they're both, like, number-based. So you could have some sort of augmented thing, and it'd still return the same difference.

MICHAEL CUTHBERT: Yeah. So it has a lot to do with whether we care about spelling equivalents or not. So we can always have a doubly augmented fourth. It sounds the same as a perfect fifth, but it repulses me because it is a dissonance, right? Actually, I'm not repulsed by dissonance, but it is.

And what do we have the problem if we're only looking at diatonic note numbers? Give me a fifth that might-- actually, you're supposed to-- well, change-- does it change the root, which fifth you find?

AUDIENCE: No, not really.

MICHAEL CUTHBERT: Good. So we checked for fifths. Anybody check for any other intervals? Yeah.

AUDIENCE: I did the dumb thing and checked for two thirds.

MICHAEL CUTHBERT: Check for two thirds in a row, two thirds. So if you have one third, and you have two thirds, then you might have something. Yeah, you're right. That was a very dumb thing to say that it was a dumb thing, because I think it's a great way of doing it.

So if you can have two thirds, then presumably the bottom 1 is a root. Where's the problem come? Yeah.

AUDIENCE: A major triad and a minor triad, or a major third, minor third, minor third, major third.

MICHAEL CUTHBERT: OK, great. So minor third versus major third. I want you to think in your head, and then tell your neighbor yes or no on the count of 3, on whether-- does a major third or minor third, does this affect the root? 1, 2, 3.

AUDIENCE: No.

MICHAEL CUTHBERT: No. OK. So we might-- so we're not going to do that for right now. But I'm going to put this in the save for later pile, because a root is only one of the things we're going to want to do. Save for later. But no, so it's not going to do-- what's an issue that we might still have? But that's going to be important with this.

So you've gotten rid of octaves. So let's say we're going to write everything in the same octave number. For a second-- can I blank this for a second? You still have it over there if you need it.

So if we're saying everything is in octave 4, and I'm going to rewrite that F, A, C, the F major triad, F, A, C. Is that going to work? Third, third. Yeah.

AUDIENCE: Do you mean like if we use the pitch class, then this upper C will be 0? So you won't be able to find it. Just [INAUDIBLE].

MICHAEL CUTHBERT: So it will be-- yeah, where will it be? Let's say we put everything in octave 4.

AUDIENCE: It'll look like the one below it.

MICHAEL CUTHBERT: It'll be below it. So what is this now?

AUDIENCE: It's still a major.

MICHAEL CUTHBERT: Still a major triad. What would we call this, by the way?

AUDIENCE: Second inversion.

MICHAEL CUTHBERT: First inversion? Second inversion, second inversion-- 64. Numbering instead of-- I've just switched between two different systems that are kind of equivalences of each other. I switched from calling this 33 system, which we use sometimes, to the calling this 53 system, which we use more often, where one system where you measure everything from the note below, and one thing where you measure everything from the base.

Boy, isn't this terrible in music that we do this? Do you remember the first time that somebody said to you something like, oh, I was in F major, and I was on scale degree 2 on a 57 chord. And you actually mentally heard in your head, I was on scale degree 2 in a Roman numeral V, superscript 7 thing. Somebody figure out what part of our brain is doing this kind of translation on the fly, that the exact same numbers are being translated in different ways in our memory.

So this is not one that we talk about very much, this particular translation. But we do it quite a bit, and I just did it maybe too fast. So yeah, 6, 4, or the other way you could do it is say it's a third stacked on top of a fourth. 3 plus 4 in music equals?

AUDIENCE: 6.

MICHAEL CUTHBERT: 6. Yep. This is also something we have to worry about. That was a parentheses within a parentheses. How do we get around this problem? Yeah.

AUDIENCE: I basically added 12 to the lowest-- I sorted it and added 12 to the lowest note twice and then checked.

MICHAEL CUTHBERT: OK. So you sort, and then if this doesn't match 33, then what did you do?

AUDIENCE: I bumped the lowest note up an octave.

MICHAEL CUTHBERT: You bumped the lowest note up the octave. Will that always be enough?

AUDIENCE: No.

MICHAEL CUTHBERT: So what do you have to decide to do next? So let's-- sometimes you have-- yeah, go ahead.

AUDIENCE: I took the initial chord, and I duplicated it, the whole chord, an octave above. So I guaranteed that I will see a pattern that will be either major, minor, augmented, or diminished.

MICHAEL CUTHBERT: OK. So you take whatever notes you see. Now I'm going to put it in first inversion, except there are chords beside C major, Michael. So we'll put it in first inversion, A, C, F. Call that a 63 chord. And if you just keep duplicating things up-- is that right-- you will eventually at some point have this 3-- ah! 53 stack, the stack of thirds. There will be a 53 embedded in there. Does that-- great hair, or jut-- good, super.

These are all techniques that work really well here. What are the techniques that will work if you have sevenths, or if you might be given something that is not a triad at all? How would you do this with sevenths again? Did anybody do that as their extension? You did? Yeah. What did you--

AUDIENCE: The way I did it is I worked off of what-- I guess, first, I had a function that kind of reduced the input to see whether or not it could kind of work as a triad or as a semi chord.

MICHAEL CUTHBERT: So first you had-- I'm just repeating for the microphone. First, you had a function that reduced it as a triad or a seventh to see if it worked with the previous thing. And then?

AUDIENCE: And then I kind of-- I guess I combined both functions. And so, first, I had to iterate through a list of potential interval distances. And then just for each potential candidate I have a root, and then iterate through each potential candidate and see-- check whether or not the interval distances are the same after adjusting for if it was a bigger [INAUDIBLE].

MICHAEL CUTHBERT: So you checked each one for-- each candidate as a root, and then you checked to see if the interval distances matched something. Great, super. There was a second hand for doing it. How-- yeah.

AUDIENCE: I basically did the same thing.

MICHAEL CUTHBERT: Pretty similar?

AUDIENCE: Yeah.

MICHAEL CUTHBERT: So increase the-- great, super. This, by the way, is not one of the great unsolved problems in algorithms, but I do not know if it has been proven what the most algorithmically efficient way of finding a root in anything, especially when we have chords like-- here's one of our favorites in C major. What do we want to call that chord?

AUDIENCE: Diminished seventh.

MICHAEL CUTHBERT: Not close, but--

AUDIENCE: Dominant seventh.

MICHAEL CUTHBERT: Dominant seventh chord. It should be four pitches. Which one's missing?

AUDIENCE: D.

MICHAEL CUTHBERT: D, which is the what of the chord?

AUDIENCE: Second.

AUDIENCE: Fifth.

MICHAEL CUTHBERT: Fifth of the chord, yeah, the fifth of the chord missing, most common note in the chord to leave out, unless you're Chopin. And then he likes to leave out that one-- leave out the third. Usually, you can't leave out the bass. Otherwise, you don't have a 5. Although, there's a music theorist, Hugo Riemann, who would disagree with you on that. And of course, you can't leave out the seventh. Otherwise, you don't have that. But yeah.

So how to deal with this and how to do these things is still something that we're working on And I was giving a presentation a few months ago on how things were calculated. And in the middle of the presentation, I go, oh, my gosh, that's not algorithmically efficient. And so somewhere in the music21 thing, I quickly said, to do: remember how to make this faster. And I've forgotten how to make it faster. So maybe we'll go back into that. Great.

Then the last topic-- sorry, let's put that back. So Roman numerals, what are some of the differences between the Roman numerals and just finding the root? Yeah, go ahead.

AUDIENCE: So I took the tonic that was given in native scale of it. And once I had the scale, I found the root of the chord, and then saw what position it was in the scale.

MICHAEL CUTHBERT: Great. So you created a scale based on-- what was given?

AUDIENCE: The tonic.

MICHAEL CUTHBERT: The tonic-- yeah, I only gave the tonic. I didn't give the key because we're still-- created a major scale based on the tonic. And you saw which position it was in. That's really a great, great way of doing it. What if I didn't give it in this, but what if it wasn't in the scale? How would you do that? Anyone know if things like flat 3 or flat 7? Yeah.

AUDIENCE: Just match the step.

MICHAEL CUTHBERT: Just match the step, and then figure out the subtraction of the difference. So great. So we can find a position in the scale. If you were wondering why question 1 was on major scales, that's exactly one of the approaches I thought you could do. Anybody else have a different methodology for this? Yeah.

AUDIENCE: If it wasn't in the scale, I just returned an error, saying it was invalid.

MICHAEL CUTHBERT: Yep. And it's invalid in this assignment. Was it not? Yep. Good. Any other things to take all your information? Now, it was Jake, right? Who brought in major third, minor third? Yep, now I think it's time to get that in. What do you-- how do you convert that? What does this information tell us about the Roman numeral? Yeah.

AUDIENCE: Uppercase, lowercase.

MICHAEL CUTHBERT: Uppercase, lowercase. Good. Or there's one other thing to put on. What's the Roman numeral that often takes a special symbol?

[INTERPOSING VOICES]

AUDIENCE: --7.

MICHAEL CUTHBERT: Diminished 7. Yeah, so the seventh scale degree, quite often the 5 with the O-- 7 with an O after it, the diminished 7. Good. And then inversions, those become kind of the problem there.

So I want to give a 2-minute peek on the code that you've now unlocked. Sorry. There is a way to make this bigger. Nope. There we go. And just to give a little bit of a peek at music21's-- there we go. Sorry. A tiny bit too big. Tell me-- yeah, that should be the right size. Sorry a lot of docs over time-- things.

romanNumeralFromChord-- so this is something-- so given a chord object and a key object, and then possibly a preference for do we prefer secondary dominants? Can you have in major-- can you have a major chord on two, or is that the dominant of the dominant, 5 of 5? And try to give an appropriate name.

So I don't know why my first one is so complicated. I guess back then it was just like, hey, I can do this. Let's see if we can come up with something. So if you have a chord, E, C, G, B flat, and then a few extra octave things, just to throw off. In F major, that should be a V65.

Who remembers their inversions of seventh chords? So root position, 5, 7, then-- first inversion, V--

AUDIENCE: 65.

MICHAEL CUTHBERT: 65. Second inversion?

AUDIENCE: 43.

MICHAEL CUTHBERT: V43. And third inversion?

AUDIENCE: V42.

MICHAEL CUTHBERT: V42. Some people also teach the tradition 2. And some people-- so 42 is the most commonly used one. I've seen some people have the less-- the even more compact 2, and some people the more verbose 6/4/2.

Any baseball fans? I always think of double plays as-- I mean, second inversion. That was-- because they always-- never mind. So you can have certain things. And then one of the big issues is that people call Roman numerals VI and VII in minor, according to different naming traditions.

So some people say that the harmonic minor scale always dictates what the letter name is. And some people say that if you say it's diminished, like diminished 7, you already know it's been raised, and you say it's major. You know it's been lowered, and you don't need a sharp or a flat in front of it.

So there's a whole bunch of different conventions. And unfortunately, that's something we haven't standardized. For anyone who's had two different music theory teachers, you might have already encountered that.

So let's get down to the code to show some of the things that I found. And maybe you'll come up with a better one. Who has taken 302 or a class that has encountered augmented sixth chords? Yeah. So you know these country codes, Italian, French, Swiss, some people call it, some people English, German. You don't need to worry about that for this class since we didn't have it.

But one of the other things that comes up a lot is that you need to be able to change certain things that might come up. So that's a 57 chord is 753 or 7531, if you want to call it, but we only call it 7. 65, seventh chord first inversion is 6/5/3, but we call it 65. So you just need a bunch of substitution things. And these aren't all of them, because we just grabbed them from a lot of places to make sure ninth works. That's always a nice thing to do.

So what we find is the first thing you want to know is, what's the root of the chord? And finding the root, as I said, is already an interesting algorithm. And then figure out how many semitones is there between that root and whatever is the third, and find out if it's on the basis of major or minor, because that's going to determine uppercase or lowercase.

We'll skip anything that says, if there isn't a key. Put that in. Then created this conceptual device called a figure tuple, which is the notion of something like, if you had a chord with-- oh, I don't know-- you might-- sorry. That is not a French vocal clef.

You might have-- call that flat 5, flat 3. In some ways, this, one of these figures, call this a figure tuple, which tells you-- oop! Trip hazard. What's the thing over the base? It's just sometimes easier to keep track of how many semitones is it altered and what do we call it? So you might think that negative 1 always means flat, but there are some cases where that's implied by other parts of the thing.

Then correct the flat VI to VI in minor, depending on people's spelling. Then we end up trying to figure out all these things on how we can figure-- how we can transpose the tonic so that we can deal with those things that are outside the scale. So like what we said, if you have a root that is not part of the scale, then we just really quickly construct a whole other scale down a semitone or up a semitone. And so we can do all the other calculations, and then just throw flats on everything afterwards.

Probably guess what this does-- capitalize if it's major, lowercase if it's minor. And then a lot of these things are slow. So a lot of-- if we've already seen this before, get it from the cache.

And then something to turn 1 to i, 2 to ii, 3 to whatever-- convert the number from 1 to 3,999 to a Roman numeral. I don't know why I stopped at 3,999. It's because I'm a historian. And the numbers above 3,999 were never standardized, no matter what your teachers tell you.

Then we try to figure out what kind figure it's going to be to try to get-- what kind of inversion it's going to be. So what's the string for that? And that's where a lot of these-- what is it? 643 becomes 64 and so on.

Then there's-- then we'll try to think about-- so we take-- what's the prefix? Is it flat too, whatever? Add to that the Roman letter. Is it I, II, III, IV, V, VI, VII? And then the inversion. And then everything after that is every special case that we've encountered over time that turns out to be an inconsistency in the standard of Roman numerals.

This is not a history class, but there's a really fascinating thing about Roman numerals analysis. And that is it is a mashing together two different, incompatible analysis types that over time got put together into one, but so that things like the uppercase and lowercase duplicating the function of putting a flat 3 on something.

And so what happens if you're in C major, and then you see lowercase v flat 3? So let's say you're in C major and you see lowercase v flat 3. What do you think that should be? G as the base, as the root. Yep.

AUDIENCE: B-flat.

MICHAEL CUTHBERT: B-flat and D. What happens if I leave off the flat 3?

AUDIENCE: The same.

MICHAEL CUTHBERT: Same. So is this a cautionary? You can add it, or you can put it out. But then there are other cases-- what if I'm in C major and I put I7 versus I flat 7? But the same thing with V7 doesn't need the flat. V flat 7, same thing?

So there are all these inconsistencies in the world of Roman numeral analysis. And it really begins to get exposed when you start doing things that we're going to be looking into right now. So we're going to get to the algorithmic composition in just a bit. But what I'd like you to do for a second is-- oops. Which one's here?

We'll go back into our favorite Bach piece, because corpus-- sorry. Switching gears, laptops open again. We did this once already, bwv66.6. And we'll remember chordify. And we'll say chord_bach today, chord_bach=bach.chordify().

And I remember this one as basically an F-sharp minor. So we'll say, fm=key.Key('f#'), F-sharp. And what we'll do on this is just for-- so actually, snap this up when you've got this. And also, by the way, if I'm typoing, this one, I don't have a template for. So if I typo, definitely correct me on this.

So for chord in ch, I often say, I don't write the word chord because I don't want to lose access to my chord module-- chord_bach chord.Chord for each chord. We'll say rn=roman.romanNumeralFromChord ch-- and we're in the key of F minor. So you could just write the string fm there, but it'll be a lot slower.

And then we'll just say ch-- so actually, give a second on that. So you're going to iterate for the recurse. The square brackets mean all the chords in chord_bach. Get the chord out. Analyze the Roman numeral from chord. If I were a really, really mean prof, I would say your next assignment is to reimplement that. But you can see the gray hairs here. That's exactly from that-- from implementing that one. That and chordify are the two reasons I'm getting old.

And now what we're going to do is we'll say the chord's lyric-- we did this once, right? ch.lyric=rn.figure. I think we did this, but we did it rather fast. So I want to make sure that.

Oops. That's not going to do anything, right? Because I put it onto the thing. I want chord_bach.show. My apologies. And you're going to realize there's one thing I didn't do that's going to make it look really funny. I forgot to put all of the chords in closed position, but at least it works pretty well.

So I'm going to scroll to the end, so around here. If you didn't get it all, don't worry. Some of y'all are to be thanked in the class, especially John and Adam, for when I forget to put a notebook up. They remind me. And now all the previous notebooks are up. So if you miss anything, they'll be going up later.

But you can see that there's some places where you see chords that look like chords that you've seen in your life, iv, V6, i. And then there are places, again, V. That looks pretty good. Sorry. That autosave thing knocks it down.

And then chords 763. That's rather unusual. Flat VI augmented in 64 position. Y'all remember how to resolve that chord, right? No, no. I've never heard of that chord in my life.

So this is a pretty dumb way of working with things. In fact, I'm just going to go straight from this to the last part here. Then if you get to the top, you see a lot of figures that look pretty normal. No-- nothing after the III. You see 6, nothing, 6, whatever. These look a lot better, 7. Those are numbers you're used to seeing, the Arabic numerals you're used to seeing. But the Romans are way off.

Why are these Romans so weird? Because we're wearing togas. No, no. Why are the Roman numerals so weird? I don't remember this piece. Go ahead.

AUDIENCE: It starts in A.

MICHAEL CUTHBERT: It starts in A. Yeah, if we go back, and-- we don't have to-- we remember how it goes a bit, right? It starts in A major. So one of the things you can start thinking about on your computer system is if you see a lot of III, flat 6, flat 6, III or flat VII 3, 3 flat VII, and you're dealing with a classical piece of music, what have you probably gotten wrong?

AUDIENCE: The key.

MICHAEL CUTHBERT: The key. Specifically, you've put it in blank instead of blank. You've put it in-- what kind of?

AUDIENCE: Relative minor.

MICHAEL CUTHBERT: Relative minor instead of the relative major. Good. What if you accidentally analyze the piece in major instead of minor? What would you be seeing a lot of instead for 1 to 5? Yeah.

AUDIENCE: Oh, yeah. You'll see 6 minor. You'll see 2 minor and 3 major.

MICHAEL CUTHBERT: 6 minor, 2-- yeah, 6, 2's, and 3's-- so that might tell you something about, hey, you're in a different position. Great. So strange Roman numerals probably means failure of key. Strange figures-- what do these strange figures probably mean? Yeah, Adam.

AUDIENCE: It isn't a very compatible chord.

MICHAEL CUTHBERT: Isn't a very compatible chord, or it's a chord that has embedded possibly what in it?

AUDIENCE: Non-chord tones.

MICHAEL CUTHBERT: Non-chord tones, good. Everybody's favorite non-chord tones. We'll go-- Hannah, favorite non-chord tone? It doesn't have to be your favorite. Just name one.

AUDIENCE: I'm not going to lie. I don't remember what it is.

MICHAEL CUTHBERT: Don't remember any. Great, great. So that's great, because you'll be listening up, because we're going to be doing some of these.

AUDIENCE: Diminished fifth.

MICHAEL CUTHBERT: Diminished fifth-- ah, so diminished fifth is a chord tone when you say it that way. But it's not-- or non-harmonic tone is maybe the term that you've heard in other things. When you say it that way, it might be part of a chord, but it might-- it has to have a function in order to do it. So a function-- yeah, John?

AUDIENCE: Anticipation.

MICHAEL CUTHBERT: An--

--ticipation. Yeah, where you get to the chord-- maybe you get to the tonic a little bit too early. Good. Anticipation is a really good one. Anyone else have a favorite one? Yeah.

AUDIENCE: Suspension.

MICHAEL CUTHBERT: Suspension Yeah, we hold the chord-- the tone "too long." I mean, it's great. Don't-- you know, obviously, it's wonderful, but too long for the harmonic analysis. So let's say-- computationally, if your job is to get rid of non-chord tones, what are you going to do? How do you get rid of anticipation?

AUDIENCE: You just have the rest of the chord following.

MICHAEL CUTHBERT: Yeah. You just have the rest of the chord following. Great. How do you get rid of suspension?

AUDIENCE: You cut it shorter.

MICHAEL CUTHBERT: You cut it shorter. Great, great. So what's another non-chord tone? Nobody's done the one that you-- the first one that most of us learn. These are more-- yeah.

AUDIENCE: Stepwise.

MICHAEL CUTHBERT: Stepwise, yeah, between two things. What do we call that?

AUDIENCE: Passing tone.

MICHAEL CUTHBERT: Passing tone, yes, the passing tone. So great. Well, how do you usually get rid of a passing tone? Just cut it out and make the previous note longer, right? How do you identify a passing tone? It just looks like it, doesn't it? It's so easy on our neural processors or our eyes. It just sounds like it. What kinds of things-- what criteria do we need to know that something is a passing tone?

Those are the worst stems ever. Actually, the clef doesn't matter, does it? No, it's still a passing tone. Which one's the passing tone? Who votes for C? Who votes for D? Who votes for E? OK, good. Lots of Ds, especially on the side of the room that I have privileged by making it much easier to see. Sorry about that. I'm not going to stay here very long, so just squint for a little bit and feel free to shout out what I'm doing for free closed captioning. Great.

So what kinds of things would you be putting if statements on if you were like, "I'll just throw it in a neural net and it'll learn"? But no, if you had to program it, what would you be looking for? Yeah.

AUDIENCE: Seconds.

MICHAEL CUTHBERT: Seconds. So we have to have a second, and then a second. Do the seconds have to be ascending? No. Can they be in opposite directions?

AUDIENCE: No.

MICHAEL CUTHBERT: No. Why not?

AUDIENCE: Because then you have a neighbor.

MICHAEL CUTHBERT: Then you have a neighbor tone, yep. So 2, 2, second, second equals passing tone. Negative 2, negative 2 equals passing tone. 2, negative 2 equals neighbor tone. And negative 2, 2 is neighbor tone.

I know some of you are already trying to pass your LeetCode things and thinking, what's the bitwise operation that's going to make it so I don't have to put all four of those? No, it's OK, it's only four. So you can look-- find these in here.

So yeah, you might be doing that. Then anything else that you think is important for saying that something is a passing tone? So I have this format. Yeah, John?

AUDIENCE: Making sure the notes surrounding it are chord tones.

MICHAEL CUTHBERT: Making sure the notes surrounding it are chord tones. Yeah. So these things have to fit into this. And this is why I was saying that diminished fifth might be a-- might not be a non-chord tone if the surrounding chord happens to be a diminished triad. And you might say, well, then all of these might be chord tones, because maybe the surrounding chord really is 4-7-6-3, but probably not.

What do we-- so now there's two ways to look at this. We can do this on the individual voice level. And so that's what we're doing-- or individual part level. If you're looking at a chorale, that's what you're going to do before chordifying. The other thing you can do-- what do you think is happening here? You can't even really see what's happening. But you can see that we have iv6, something weird happening, and then iv.

So we're going-- something's happening here, where we're going from the first inversion of a chord to the root position of the chord. And these, there might be a lot of passing motion in there. So you might be able to identify these non-chord tones at the chord level, or you might be working on the individual voice level. Which one do you think works better? Individual voice level, chord level? Individual voice? Chords? Yeah.

And the answer? We don't really know. I try to do both, because some things are easier to get from one than the other. So great. We've gone through, and we've done a question-- outline 1 and most of outline 3 on here. I jumped around because I realized that a different ordering was going to be better. So we're going to gone through 3 A and B, and then I'll leave C and D for a little bit later.

What I want to do, since we have our jigs loaded up, I want to do a little bit of 2. We'll say take two if we're doing it. No, we're not doing that level of editing. But one last little thing-- problem set 9, you're going to be getting in a little bit. And what you're going to be doing in problem set 9 is this kind of chord tone identification and removal.

So you're going to be given various-- you're going to be given various four-part chorales. And what I want to see is instead of things like this that look really hilarious, something pretty clear out, and a nice reduction of that.

One of the things you might start thinking about is that Roman numerals are a type of equivalence class on a set of pitches. That's not only about-- well, it helps to say that this is the same chord-- the same chordal function in different keys. It's also an equivalence class on the cloud of notes, including non-chord tones that we can say, these two things are the same Roman numeral, even though one of them has filled in all the passing tones or neighbor tones-- suspensions.

Did we get them all? Escape tones. That's a pretty hard one. There are a bunch of other ones. But yeah, so these are things that you're going to be doing. So I'll be giving it to you in just a little bit. So you want to be able to take something and make it quite a bit nicer.

You're going to have help on this. A couple of things that help-- you're going to have at least a week. I'm thinking next Friday. There's the danger between I don't want to give too little time, but I also don't want to go so late that you can't have your full attention on the final projects.

So what's your thing? What I'd like you to do is you're going to be doing problem set 9 with your partner that you're working on the final project with and one other team. So groups of four. The team of four, I think you're a natural group of four. So why don't you put them there?

But what I'd like you to do is there are different steps along the process. Here, I'll just-- I'll pop up PSet 9 right now, so you can see it while it's here, even though I'm doing some last second tweaks, depending on what we get through today. Great.

PSet 9, you're going to be doing reduction and key analysis of tonal music. And I want you to say-- note this. You need not get every chord right. It need not work on every chorale. I'm asking for a decent amount of work toward a solution, not a complete solution in itself. Great. So you're not going to be doing that.

So what you'd probably like to do is-- what you're going to do is I'm going to call vocabulary_reduce on a score, and you're going to give me either a score or a part, depending on how you try to choose to represent it. That is the reduced one.

You have four people on your team. You probably do not want to implement-- you definitely do not want to implement your entire solution inside this vocabulary_reduce programming in a Jupyter Notebook. Take this, divide it up, and share it among your favorite collaborative coding tools.

I'm starting you with a chorale that's easier than the one we've been working on in class 66.6. And here are the four I'm going to test you on for 90% of your points. So I'm going to tell you that. Oops. So this is why I'm still editing. You have key analysis unlocked because we lost four classes this year. So you'll be allowed to analyze the key using analysis things.

We're going to chordify. This addPartIdAsGroup is something that lets you figure out for each pitch in your chordified score what part it originally came from, which can be very useful for figuring out what to delete. And yeah, we'll just show the first-- put it in-- actually, I'll show different steps that might happen here.

So what I've done is I've gone through one chord that if you load these things-- sorry-- one choral-- oops! Sorry. I forgot I have to restart and run all. Yep, here we go. So here's the original chorale I'm asking you to work on. And then I'm unlocking something called implode(), which you might find useful, just for seeing a chorale on two staves. It might not be too useful for you analytically.

And then what I've done is given you the first part of the answer-- how an answer, how I think this might work with all the non-chord tones labeled. And then, second one, here is after they've been removed. So you have them all labeled. Now you have all them removed.

And then the last step, you might be trying to figure out that, actually, there's some key changes in here. So we move from A major to D major to A major and so on. Why have I given you these intermediate steps, if this is what you're trying to get to? It's because you have a larger team. And so what I would like is that somebody can be working on part 4 while somebody else is working on part 3, while somebody else is working on part 2 and part 1.

So then what I'll say, the last little thing is just to show a little preview of some of the things that we might be doing later. And I know it's not on the right screen. I can make that bigger. This is something I've been working on an automatic-- well, let's let it run first-- on an AI-based automatically trying to do labeling of Roman numerals based on sets of pitches and training data.

What I've done is, right now, I'm only having it work with a few Roman numerals. And always take a guess when you're trying to figure out what things are. These, how many entries are there in each of these arrays? 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12. So these are the pitch classes above the tonic thing, so that one has-- I can't even reach up there.

So one has the most entries in, if you call that, C, C-sharp, D, E-flat, E-- that's quite a bit-- F, F-sharp, almost nothing, G, quite a bit. And so this is how many times you see that chord tone in everything in our labeled training data was labeled I, and so on. For V, you see a lot of V, scale degree 5, scale degree 7, and scale degree 2, kind of what you would expect.

And put everything into a neural net that I'm still trying to make this a little bit better, a little bit better. Run it. Hopefully, it's running. You can see run it through one time. It's starting to get 62% accurate. Keep going up.

It stays at 62, 62, 62 quite a long time. It jumps up to 67, 68. Hopefully, it's going pretty well. Eventually, it gets 87, 88, 90, and so on. So it is possible to do this kind of thing with today's computational resources kind of ending at around-- are we going to be able to get to 99 today?

AUDIENCE: How big is it?

MICHAEL CUTHBERT: So that's what I'm going to get to in just a second. So this-- oh, of course, my computer can barely handle this while projecting. I think I did 400.

Well, while it tries to scroll down to the bottom, the danger is this data set took me years to get. And it took years to put in. This is the complete piano sonatas of Mozart.

[VOCALIZING]

Complete piano sonatas of Mozart, all labeled. And when I divided it up into half training, half test, I still didn't have enough information for it to be able to get on. So I'm dividing it into 90% training, 10% test. And that seems to be about what's there. So the other thing was when I decided, oh, this sounded like a fun project for over the holiday break, so I thought I'd be able to present this on the first day of class or something.

I went in, and the entire Roman numeral set could not be trusted compared to the encoding of the things, because people slightly disagreed on where measure 14 was, or where measure 113 was, or this edition had this part of the minuet. The repeats were written out. And they weren't-- so I was going through. And when I was spot checking, the Roman numerals just looked totally wrong to me, or it would say, Roman numeral placed after end of piece.

And so, again, it was about-- I don't know. It was about 3 hours a night for about 3 weeks of just, well, let's realign this one. Let's realign. Now, it was a little bit less than that, but about 2 weeks maybe. So this ends up being one of the dangers and why I'm pushing-- even though, I think we're starting to get-- how many epochs did I do this time? I must have changed it. But we're getting pretty close.

And it takes a very long time before the computer is being able to get things up. So these are some of the reasons why I'm going to teach you some of these techniques. But then I'm going to encourage you not to use them on your final projects, even though they are kind of the coolest new thing going. Great. Any questions?

And sorry. I'm not getting-- oh, here we go. Finally, we get to the done. And my computer is still so slow. Here's the mistakes that it was making. Oh, this time it only got there. Last time, it got to all but one correct.

Yep. 177 times it saw I. It predicted it one. One time it saw it, and predicted V. You can imagine that IV and II have some confusion with the computer. And V and VII also had some confusion. VI, VI and IV have some confusion. So then the nice thing was actually going back the last time I did this and looking at every place that there was a confusion and saying, oh, about half the time I agreed with the neural net and not with the professional encoder.