Max/MSP exercises 3b

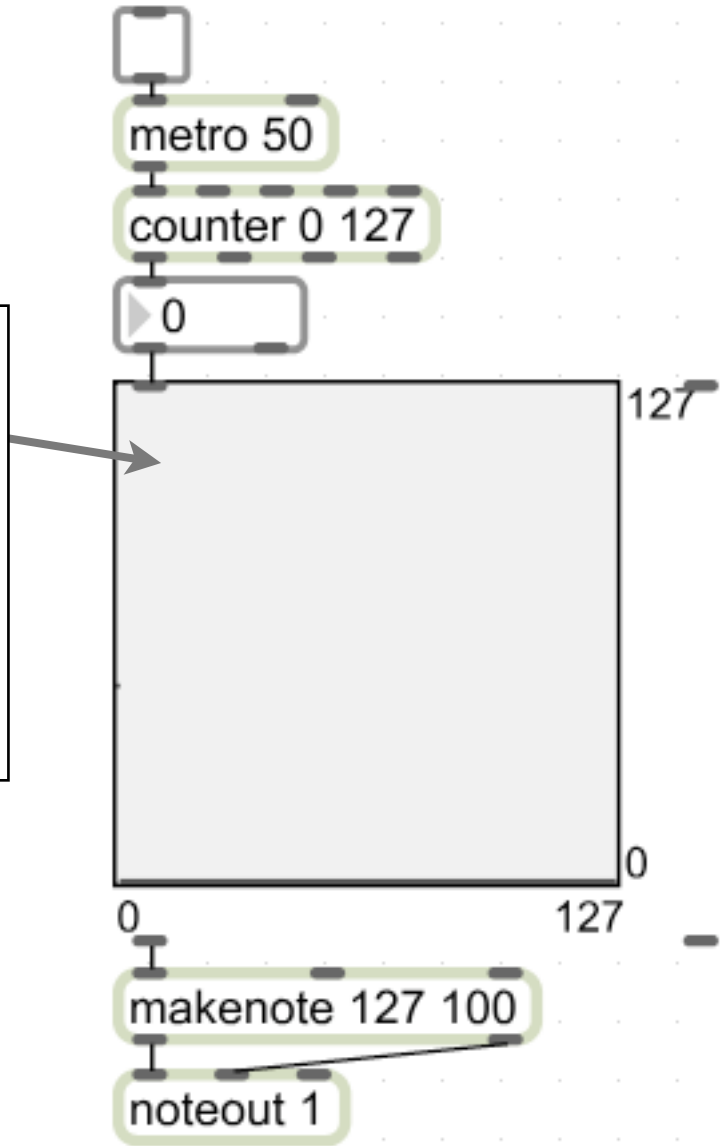
Ex. 1

In the first week we made a step sequencer for playing drum patterns. Now let's consider making some tunes.

1. Copy this routine. Switch on the [metro], then draw some wiggly lines into the [itable] object.

This is the [itable] object. It looks like this in the object palette:

Like [coll], [itable] has an index and stored value. Its index is stored on the 'x' axis and its stored value on the 'y' axis. This patch reads through the table from left-to-right.

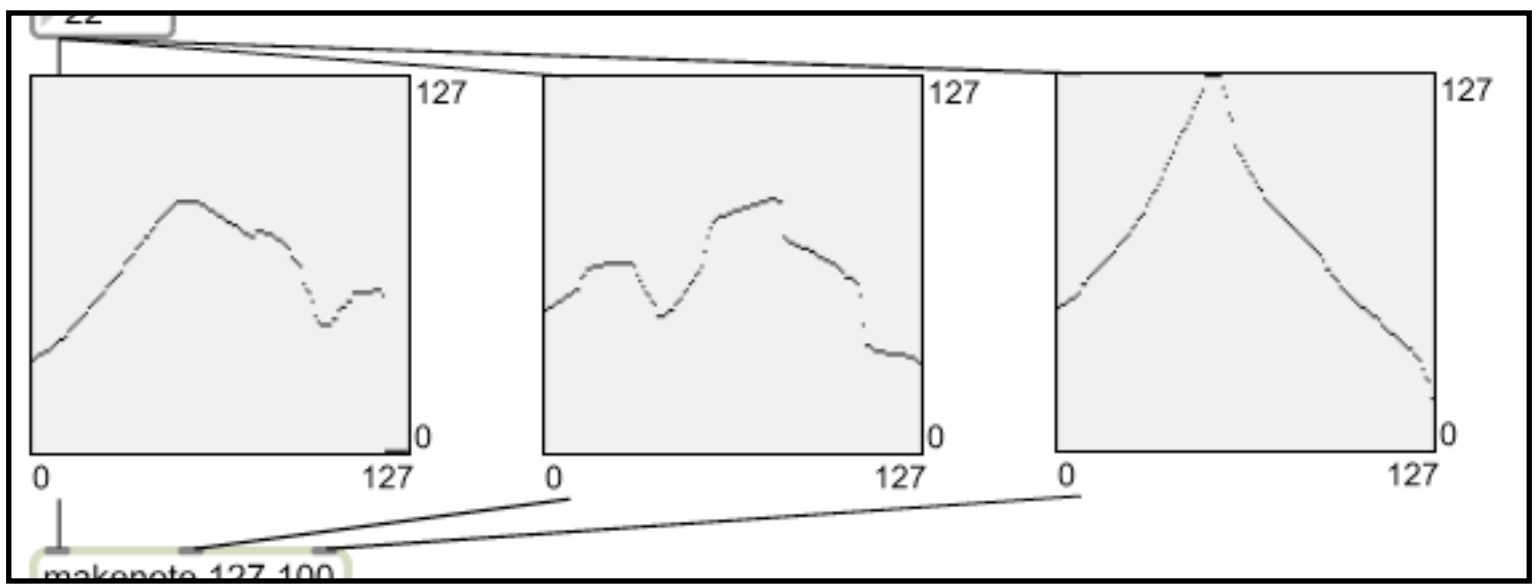


Ex.1 (cont)

We could further [itable]s to dictate the velocities and note lengths for each of the notes in this first [itable].

2. Use a [pgmout] object to find a sustained instrument (17 or 18 are decent organ sounds). This will enable you to hear the changing note lengths more effectively.

3. Add two more [itable] objects. Connect the [number box] to the first inlet of each of these, and their first outlets to the second (velocity) and third (note length) inlets of [makenote] as per the following:



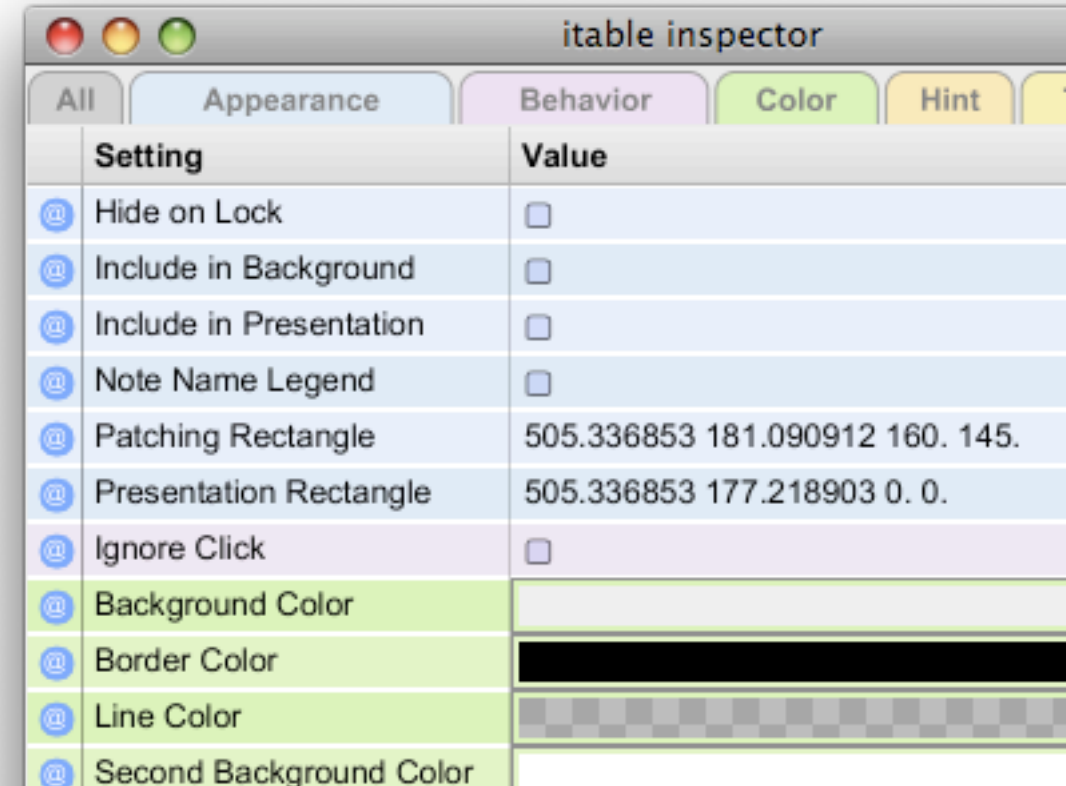
Ex.1 (cont)

Note that the note lengths remain very short. This is because the range of [itable] is limited to 128 (so we only have a range of 0-127ms note duration). To remedy this, we can use the object's inspector window.

4. Click on the [itable] used for note lengths and press -I (or Ctrl-I in Windows). The inspector window contains information about the object that you can change -- very much like adding arguments to it (in fact that is essentially what you are doing).

5. Scroll to where you find the entry for Table Range and change it to 1000. Your [itable] y-axis will now read 0-999.

6. Investigate the other properties of [itable] that you can change via the inspector window.



Ex.1 (cont)

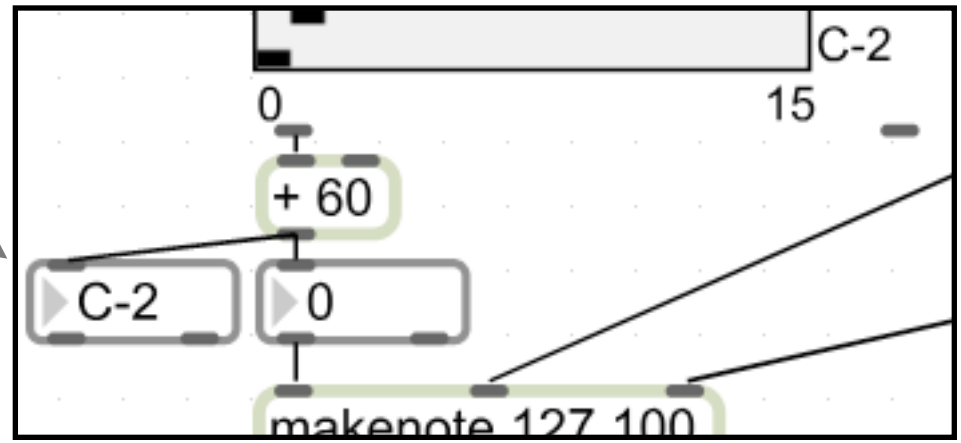
Having a table size of 127 isn't particularly useful to us in our pursuit of tune generation; we'd more likely want shorter bar lengths of 8, 16 or 32 beats. And we may want to reduce the range of available notes to a couple of objects.

7. In the inspector window of the [itable] for note numbers (i.e. the left-most one):
- change Table Size to 16 (for beats)
- change Table Range to 26 (for semitones -- this will give us just over two octaves)
- tick the check-box for Note Name Legend (the y-axis will now have note name labels instead of numbers -- C-2 to C#0).

Ex.1 (cont)

8. Our note range for this [itable] is therefore rather low, so we'll transpose it up by five octaves (by adding 60 semitones):

the [number] box can also be made to display MIDI note names: just change the Display Format in its inspector window to 'MIDI'



Of course this means our MIDI note-name label on [itable] shows the wrong octave. But at least we get the right degree of the scale...

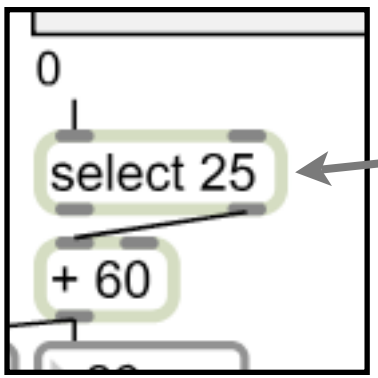
9. A few more things to change before the whole thing works satisfactorily:
- add a [number] box above [metro] to enable you to reduce the tempo
- change the values in [counter] to only count 16 beats (remember [itable] values start at 0, so you'll need to take that into account)
- change the Table Size of both the velocity and note duration [itable]s

Ex.2

Our tune is a bit relentless as there are no rests. So we'll deal with that issue next.

You will recall that we chose a 26 note range in Ex1.7 (0-25). Which leaves us with an extra, somewhat redundant, C# on top of our two octaves. So we'll get Max to ignore this note completely, treating it as a rest.

1. Add a [select] object with an argument of 25 (remember that this is the 26th note (0-25)) beneath your note-number [itable] as follows:



Normally we would be using [select] to send output when it *recognises* a number. But here we are taking the *right-most* outlet of [select] which spits out anything it *doesn't* recognise. This means that the number 25 is completely ignored. Only the other numbers (the notes we want to hear) are let through.

Any 'notes' you insert right at the top of the note-number [itable] will now trigger rests.

Ex.3

1. Download the patch 'polytuneseq.maxpat' from the MUST1002 website and open it. This is a development of the instrument you have just made (omitting the velocity and note-length parameters).

2. Explore the patch as follows:
   a) turn on the metro;
   b) turn on each of the voices in turn, noting the use of different channels and instruments (see [pgmout] and [noteout] objects);
   c) turn on each of the changes toggles in turn, noting that the tunes now change on every bar. You should end up with some quite complex textures between the various voices.
   d) double-click the [table]s labelled 'tune1', 'tune2' etc. These work in much the same way as the [itable] object, but are editable within a pop-up window.
   e) double-click one of the [patcher voice] objects and explore the sub-patch. Read the labels carefully from top to bottom. Much of it should be either familiar or self-explanatory.