Advanced quantization in Logic Pro: what it does to MIDI note timing

Frans Absil*

August 2013

Abstract

This document explains advanced quantization in the Logic Pro software sequencer. Using a MIDI note example the effect of setting the most important quantization parameters is demonstrated.

Keywords: software sequencer, MIDI note quantization, advanced quantization

1 Introduction

Sequencer software, such as Apple's Logic Pro [1, 2], record and process MIDI data . Using a MIDI keyboard as an input device leads to note starting points not falling on a regular timing grid. This aspect may be part of a deliberate playing style or lack of keyboard skills (as is the case with me).

In the latter situation, the sequencer *quantization* option becomes a big time saver. The automatic *basic quantization* process will move the note starting points to the nearest line on a quantization grid. The quantization time interval is a musical note length, or a number of ticks (the smallest timing unit in the sequencer software). The note pitch, duration, and velocity are irrelevant for the quantization process. The quantization will lead to a more regular rhythm, with a mechanical or robotic effect in case of extreme, i.e., full quantization. This will sound unmusical in most situations.

A more refined *advanced quantization* process yields more natural results in terms of musical style. A number of parameters will determine the amount of quantization and the subset of MIDI notes that will undergo quantization.

This document will use a MIDI note region example to illustrate the effect of advanced quantization in Logic Pro. The software manual provides limited description only. Online internet forums discuss the issue in qualitative terms. Here we will demonstrate what happens to the MIDI note starting points in quantitative terms, i.e., in tick numbers and (for some readers, unfortunately) in mathematical terms.

2 Basic quantization

Basic quantization means that, based on a chosen quantization value, all MIDI note timing errors are corrected; the starting time of the notes (the MIDI Note On message) will coincide

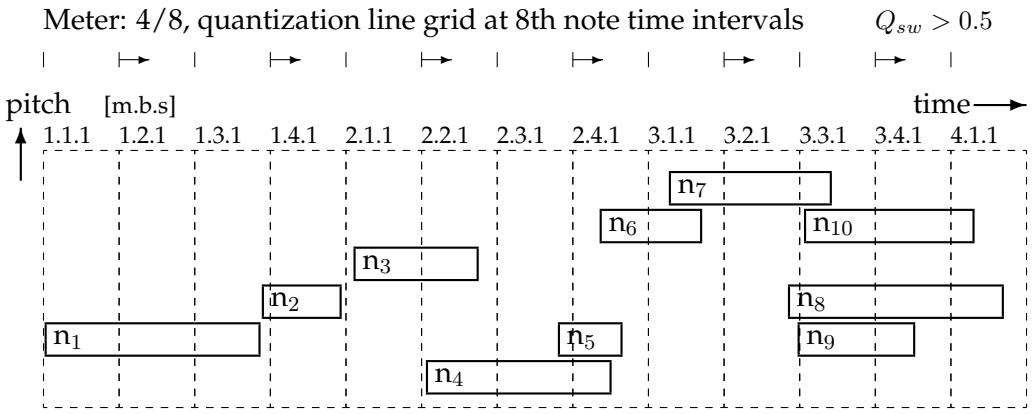

Figure 1: The MIDI sequencer editor window in piano roll style, with a regular grid (measures and beats, here for a 4/8 meter) and MIDI note information (starting point, pitch and duration) before quantization. Time runs along the horizontal axis, pitch along the vertical axis. The grid lines are shown on the top for a quantization value $\frac{1}{8}$ (8th note). The delay shift of the even quantization lines in case of a swing effect is indicated. The MIDI example contains ten notes.

with the nearest quantization line on the grid.

The MIDI note starting point in Logic Pro is written as `m.b.s.t`, with `m` the measure number, `b` the beat number, `s` the beat subdivision number and `t` the number of ticks. A *tick* is the smallest timing unit in the sequencer. Logic Pro divides the whole note into 3840 ticks; this corresponds to 480 ticks for the 8th note ($\frac{1}{8}$). Quantization frequently is applied to a grid based on 8th notes or shorter (regular subdivision into units of 480 ticks or less). The quantization value, `Quantize`,[1] is based on note durations, such as a quarter ($\frac{1}{4}$) note, a 16th note, a triplet unit (e.g., $\frac{1}{12}$-th). This document will specify the note timing in tick units, leaving out the irrelevant measure and beat numbers.

The quantization example in this document is shown in Figure 1. This piano style editor window shows the MIDI note pitch, start time and duration as a function of time. The numbers for this example before quantization are shown in the first columns of Table 1. The third column shows the note start timing error, relative to the neighbouring quantization line, here for a 16th note (240 ticks) quantization grid. In the example there are six late notes, three early notes, only one note is perfectly timed (n_9). There is one 2-note chord (n_5 and n_6, poorly timed), and one 3-note chord (n_8, n_9 and n_{10}).

The swing value, `Q-Swing`, is a percentage (between 0 and 100%) and corrects the position of the even (every second) quantization lines. Values below 50% will move these even quantization lines earlier, values above 50% will delay them. Typical values for swing music lie between 55% and 85%.[2]

Suppose there are N notes in the MIDI region (here $N = 10$). The starting time of the i-th note (with $i = 1, 2, \ldots N$) is Δt_i away from the quantization grid line. Positive values, $\Delta t_i > 0$, are late notes (after the quantization grid line), negative values $\Delta t_i < 0$ indicate early notes (with a pre-delay).

[1] Words in teletype, such as `Quantize` correspond to the text labels on the menu buttons in Logic Pro.

[2] The swing corrections are tempo-dependent. High swing quantization values (say, above 70%, corresponding to dotted 8th – 16th note rhythmic patterns) are used for slow tempos, such as a moderate swing. Lower swing quantization values (nearer to 50%, corresponding to more even 8th notes) are used for up tempo swing.

Table 1: The MIDI note example contains ten notes. The meter is 4/8, quantization grid is spaced at regular 16th notes ($\frac{1}{16} = 240$ ticks).

	Before		**After Quantization**					
			Q-Strength: 100%		Q-Strength: 75%		Q-Strength: 50%	
MIDI Note	**Note on** [m.b.s.t]	Δt_i [ticks]	$A_Q = 1$ pos	$\Delta t_{Q,i}$ [ticks]	$A_Q = 0.75$ pos	$\Delta t_{Q,i}$ [ticks]	$A_Q = 0.5$ pos	$\Delta t_{Q,i}$ [ticks]
n_1	1.1.1.010	10	0	[-10]	3	[-7]	5	[-5]
n_2	1.3.2.220	-20	0	[20]	-5	[15]	-10	[10]
n_3	2.1.1.030	30	0	[-30]	8	[-22]	15	[-15]
n_4	2.2.1.020	20	0	[-20]	5	[-15]	10	[-10]
n_5	2.3.2.200	-40	0	[40]	-10	[30]	-20	[20]
n_6	2.4.1.090	90	0	[-90]	23	[-67]	45	[-45]
n_7	3.1.1.070	70	0	[-70]	18	[-52]	35	[-35]
n_8	3.2.2.210	-30	0	[30]	-7	[-23]	-15	[15]
n_9	3.3.1.000	0	0	[0]	0	[0]	0	[0]
n_{10}	3.3.1.020	20	0	[-20]	5	[-15]	10	[-10]

Since only the note starting position (MIDI Note On) is relevant for the quantization process we display the notes in a different way, that represents the difference in tick units from the nearest quantization line. This is shown in Figure 2, where the original notes before quantization are shown as open circles.[3]

The effect of basic quantization is shown in Figure 2.a (on the left). After quantization, all poorly timed note attacks will lie on the nearest line in the quantization grid (closed circles).

3 Advanced quantization

In the Logic Pro main Arrange window, somewhat lower in the MIDI region information panel, we find the advanced quantization options and parameters. The effect of three of these parameters on the quantization process is illustrated.

3.1 Setting the quantization strength

Using *quantization strength* we prevent the robotic effects of all notes starting exactly on the quantization grid lines. Instead we apply a gradual, limited effect that shifts the notes towards these lines somewhat, depending on the value of the strength setting.

The quantization strength, `Q-Strength`, has a value between 0 and 100%; this can be represented by the setting $A_Q = [0, 1]$ (i.e., it has a value between zero and one).[4] For the i-th note the proportional quantization correction is

$$\Delta t_{Q,i} = A_Q \, \Delta t_i \qquad \text{for} \quad i = 1, 2, \ldots, N. \tag{1}$$

[3] Verify that the two diagrams (Figures 1 and 2) and Table 1 represent the same data, but in different form. Hopefully, these representations will help in understanding the effect of different quantization parameter settings.

[4] When not specified, Logic Pro assumes that `Q-Strength` is 100%, i.e., the default value, that leads to full quantization.

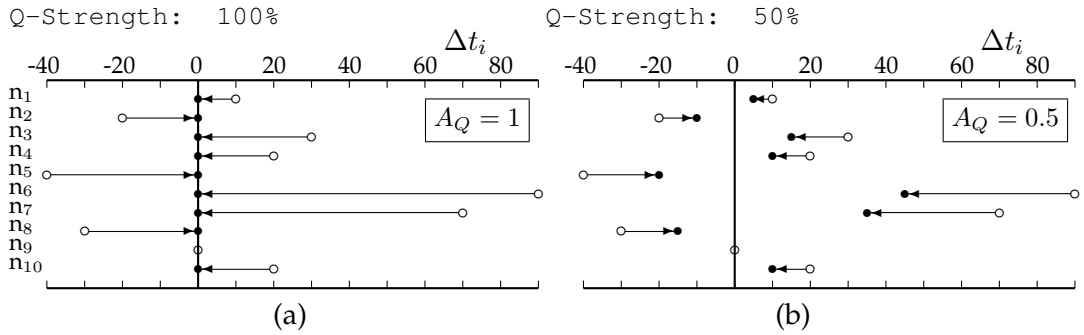

Figure 2: The difference between the note starting position and the nearest quantization grid line in tick units, before (open circle) and after quantization (closed circle). (a): Full quantization (strength 100%), (b): partial quantization (strength 50%).

Figure 2 shows what will happen to the original MIDI notes in the case of full (i.e., $A_Q = 1$) and partial ($A_Q = 0.5$) quantization. In the former case all notes will be shifted until they lie on the nearest quantization grid line, corresponding to perfect timing. In the case of partial quantization ($A_Q < 1$) the notes will move towards the quantization line. The two cases shown in the figure make understanding of all other intermediary cases easy; there is a simple proportionality to the value of `Q-Strength`.

3.2 Setting the quantization range

The quantization range parameter `Q-Range` acts as a filter on the MIDI notes in the region. It will select a subset of notes that will be quantized. The quantization window range, here indicated by Δt_{QR}, can have positive and negative values. The effect of either negative of positive settings is different and will be illustrated below.

3.2.1 Quantizing the outliers: negative quantization range

The idea behind *negative* quantization range settings, $\Delta t_{QR} < 0$, is fairly simple. Only large errors will be corrected and shifted towards the quantization grid lines. The starting points of the early and late outliers move, proportional to the setting of the quantization strength. So in this case we have

$$\Delta t_{Q,i} = A_Q \, \Delta t_i \quad \text{if} \quad |\Delta t_i| > \Delta t_{QR}, \quad \text{for} \quad i = 1, 2, \ldots, N. \tag{2}$$

Figure 3 demonstrates the effect of a negative quantization range setting, combined with either full or partial quantization. When the quantization range `Q-range` is -35 ticks, i.e., $\Delta t_{QR} = -35$ ticks, only the MIDI notes, for which $|\Delta t_i| > |\Delta t_{QR}|$, will be affected. This subset contains three notes, that lie outside the range window. In the case of full quantization these will lie on the grid lines after quantization. When the quantization range window is narrowed, $\Delta t_{QR} = -25$ ticks (see Fig. 3.b) and partial quantization is applied, the subset of five notes moves somewhat closer to the nearest grid line.

3.2.2 Quantizing the near misses: positive quantization range

This case is somewhat of a mystery. The Logic Pro documentation [1] reads:

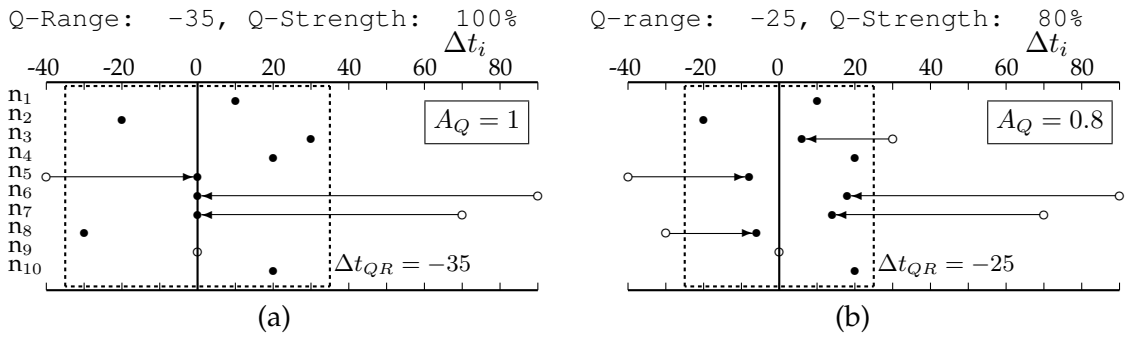

Figure 3: The effect of negative quantization range setting `Q-Range`. (a): $\Delta t_{QR} = -35$ ticks and full quantization (100%, $A_Q = 1$), (b): $\Delta t_{QR} = -25$ ticks and partial quantization (80%, $A_Q = 0.8$). Only the outliers, i.e., those lying outside the quantization range window (shown as a dashed box), will move towards the quantization grid lines.

> "Q-range: A very musical quantization strategy that requires a certain amount of technical musical prowess. Q-Range is ideal for recordings that already have the right groove but are too hurried or laid back in places. It allows you to retain the original feel, but positions the rhythmic center precisely in the groove. A value of 0 means that every note or transient marker is quantized. If you enter negative Q-Range values, only notes or transient markers that fall outside the set range are moved to ideal quantization grid positions, whereas those closer to an ideal position remain unquantized. This moves the most poorly played notes or transient markers – those outside the range – to perfect timing positions on the quantization grid, or at least moves them toward these positions, depending on the Q-Strength setting. Tip: To obtain the best Q-Range results, use a low, even quantize value, such as 1/4 note. Set the Q-Range parameter to compensate for the maximum error in the recording."

The interpretation of this text has been puzzling me.[5] An essential element here is the *feel* or *groove*, that has to do with both the relative starting position of the MIDI notes in a pattern (internal subtle rhythm and timing variations of individual notes) and the average timing with respect to the quantization lines (relaxed, laidback timing vs. hurried playing style).

In mathematical terms, elements of the groove are both the bias t_B of the set of N MIDI notes in the region and the standard deviation σ_N. The bias represents the mean value of the timing errors

$$t_B = \frac{1}{N}\sum_{i=1}^{N} \Delta t_i, \tag{3}$$

and the standard deviation is

$$\sigma_N = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (\Delta t_i - t_B)^2}. \tag{4}$$

[5] I am fairly sure that the person writing the documentation talked to the software developers, but did not understand the details of this quantization process and made up this vague description. Or, it is a well-kept Apple trade secret, not intended for disclosure. The documentation does not discuss positive Q-Range settings; that is left to online Internet forums.

For a relaxed timing the bias has a positive value $\Delta t_B > 0$, i.e., the notes start later than the quantization beats (after the click). Another aspect of the groove is the freedom of timing; a high standard deviation (always a positive number) indicates a large spread around the bias. Yet another element is the distance between the individual notes; these also contribute to the feel of the music.

In our example MIDI region, the values for the bias and the standard deviation before quantization are

$$t_B = \frac{1}{N}\sum_{i=1}^{N}\Delta t_i = \frac{1}{10}(10-20+30+\ldots+0+20) = \frac{150}{10} = 15 \text{ ticks},$$

$$\sigma_N = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\Delta t_i - t_B)^2} = \sqrt{\frac{1}{10}\left((-5)^2+25^2+15^2+\ldots+15^2+5^2\right)} = 31.3 \text{ ticks}.$$

So, on average, the notes are played 15 ticks late, relative to the grid, and their spread around this mean value is characterized by the number of 31.3 ticks.

And now, here is my best guess of what the quantization process is doing for a *positive* setting of the range $\Delta t_{QR} > 0$. The basic idea now is that large timing errors were a deliberate musician's choice; therefore these will be not be quantized, but shifted after the quantization, in order to maintain the feel of the MIDI region. However, the notes within the range (early or late), i.e., closest to the quantization grid lines, will be moved towards the nearest quantization value, according to

$$\Delta t_{Q,i} = A_Q\,\Delta t_i \quad \text{if} \quad |\Delta t_i| < \Delta t_{QR}, \qquad \text{for} \quad i = 1, 2, \ldots, N. \tag{5}$$

This amount of correction is similar to the case for negative Δt_{QR}, Equation (2), but now it will be applied to a different subset of notes, i.e., those with the smallest timing errors.

The effect of positive quantization range setting is displayed in Figure 4. When $\Delta t_{QR} = 15$ ticks (left diagram) only one note (n_1) undergoes full quantization, since n_9 was already perfectly aligned with the grid. The subset increases to four moving notes when the quantization range window increases to 25 ticks (right diagram). Note how the other notes will also shift, in the direction of and proportional to the average quantization correction. In the example this yields a pre-delay and in the right diagram the shift is much higher (between 10 to 30 ticks) than in the left diagram (always less than 10 ticks and decreasing for the later notes, as the distance to the quantized note gets greater).

This does not explain the detailed algorithm behind this specific part of the quantization process, responsible for the shift of the outliers in the direction of the average correction. There are so many variables involved; for another MIDI note sequence the outcome will be totally different and I could not infer the detailed shift mechanism.

The next diagram illustrates what happens when we increase `Q-Range` for this specific example, see Figure 5. As expected, when the quantization range window increases, more notes are quantized (see the clusters around $\Delta t_i = 0$). Note how the other notes are also shifted, first to the left (pre-delay), and then, for wider range windows, to the right (delay). However, the behaviour of notes n_6, n_7, n_9 and n_{10} is most peculiar. The former two are moving away from the timing center for $\Delta t_{QR} > 30$ ticks, which seems contrary to the original feel of these notes. And note n_{10}, included in the quantization set for $\Delta t_{QR} = 25$ ticks (as expected), escapes from this cluster for a wider quantization range. Note n_9, which was perfectly timed, starts to move even earlier! So, I doubt whether the implemented algorithm in Logic Pro can stand a

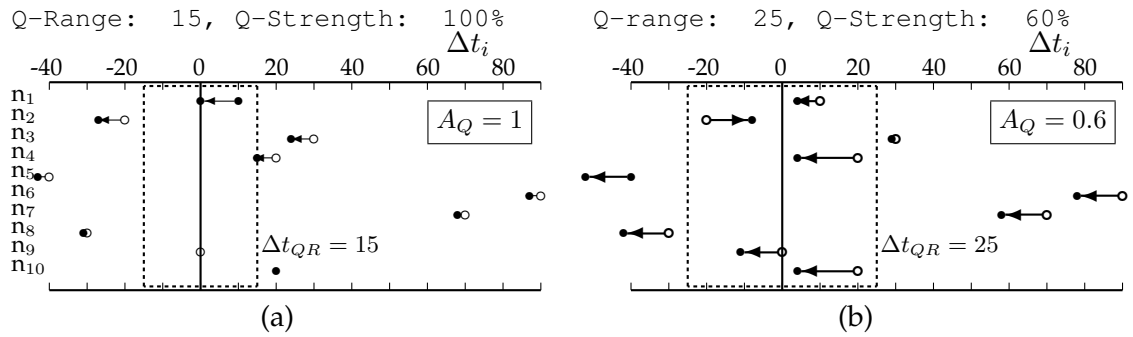

Figure 4: The effect of positive quantization range setting `Q-Range`. (a): $\Delta t_{QR} = 15$ ticks and full quantization (100%, $A_Q = 1$), (b): $\Delta t_{QR} = 25$ ticks and partial quantization (60%, $A_Q = 0.6$). The notes lying inside the quantization range window will be quantized, the other notes will undergo a shift in order to maintain the groove.

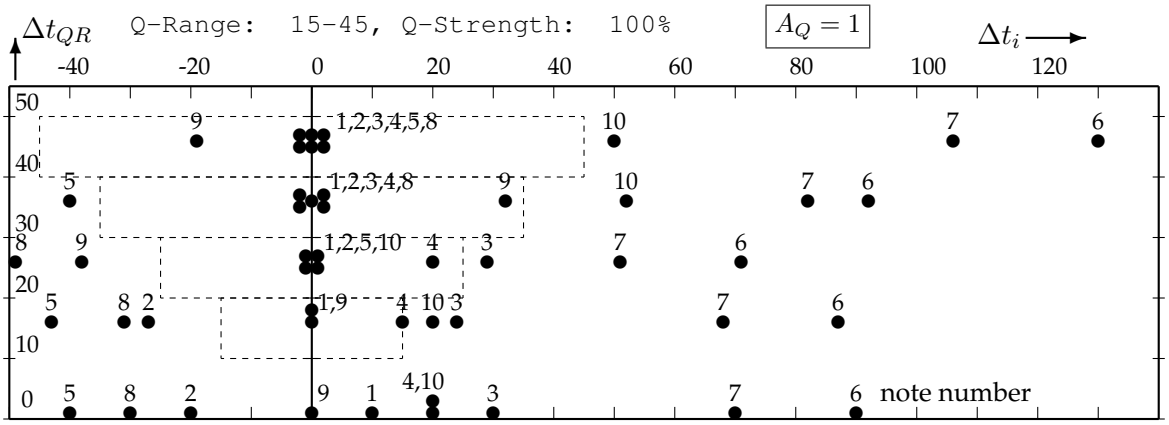

Figure 5: Increasing the quantization range window (positive values, `Q-Range` > 0, vertical axis) for full quantization (`Q-Strength` $= 100\%$). Note numbers and quantization range windows (dashed boxes) indicated.

wide range of parameter settings, when positive quantization range values are used.[6]

3.3 Setting the quantization flam value

The flam value, `Q-Flam`, is relevant for simultaneously sounding MIDI notes (in chords), either before or after quantization. Selecting positive flam values will yield a fast upward arpeggio effect, negative flam values a downward arpeggio. With N_c notes in the chord, the i-th note in the chord will move by

$$\Delta t_{F,i} = i\,\Delta t_{\text{Q-Flam}} \qquad \text{for} \quad i = 1, \ldots, N_c. \tag{6}$$

[6] If anybody knows the answer, please enlighten me, and I will update this document. Maybe the quantization algorithm has been updated in Logic Pro X (the example was created and tested, using version 9). There exist many alternative advanced quantization algorithms in the scientific literature, but I could not trace them to the Apple software developers.

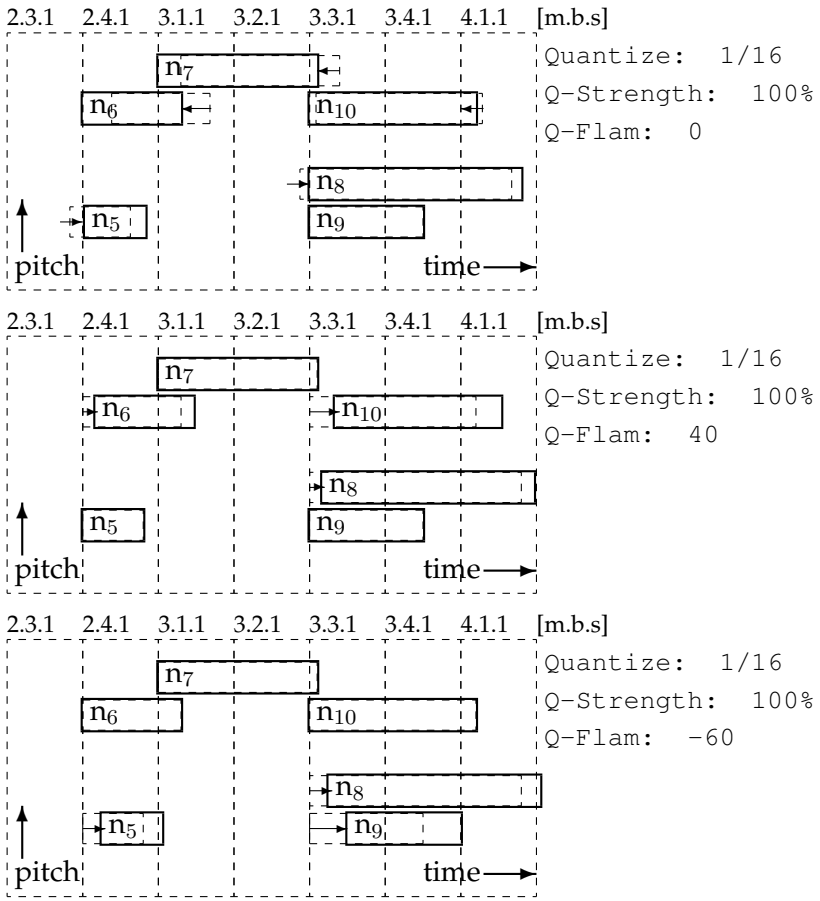

Figure 6: The effect of setting the flam value in combination with quantization. All notes are quantized to a 16th note grid at `Q-Strength` $= 100\%$ (upper plot, full quantization), then the flam effect creates either an upward arpeggio (middle plot, `Q-Flam` $= 40$ ticks) or a downward arpeggion (bottom plot, `Q-Flam` $= -60$ ticks).

Here, the combination of quantization and flam values will be shown. We return to the plot with the MIDI notes on the time axis. Remember, in the example there were two chords and we will zoom in on those notes (n_5 to n_{10}). In Figure 6 full quantization will align all chord notes in the grid lines; see the chords at position 2.4.1 and 3.3.1 (upper diagram). A positive flam setting will yield an upward arpeggio at multiples of the flam setting (here $\Delta t_{F,i} = 40, 80, \ldots$ ticks), the negative value a downward arpeggio at multiples of $\Delta t_F = -60$ ticks. Single notes, such as n_7, remain unaffected by the flam setting.

4 Conclusion

This document demonstrates the advanced quantization options in Logic Pro. With a MIDI region example we have seen the effect of full vs. partial quantization, how a quantization range window will select a subset of notes for quantization, and how the flam setting will create arpeggiated chords. Expected behaviour was confirmed by the example, but also some peculiarities were discovered. Hopefully this document helps new sequencer software users to

understand what the advanced quantization process is doing to the MIDI note timing.

Donate

This electronic document is offered free of charge **for personal use only** (see the copyright restrictions on the title page). In case you would like to support the writing and editing of this and other documents, **go to the website and make a donation**. See the website for payment instructions (find the URL in the page footer).

References

[1] Apple Inc., Cupertino CA, USA. *Logic Pro 9 User Manual*, 2011.

[2] Apple Inc., Cupertino CA, USA. *Logic Pro X User Guide for OS X*, 2013.