Gigasampler's Velocity Response Curves

Christian Schoenebeck <schoenebeck@software-engineering.org>

November 20, 2004

The keystroke velocity of a triggered key on the keyboard is essential for the volume of the resulting tone. For this a total and mostly injective function $f \in V \to A$ is used. In case of MIDI the set of velocity values is $V = \{0, 1, \ldots, 127\}$ and the result set of volume attenuation factors is $A \subset \{a \in \mathbb{R} \mid 0 \leq a \leq 1\}$. This function is called *Velocity Response Curve.* In contrast to the past where one simple and constant function was used, today's synthesizers and samplers offer a variety of possible functions to be able to provide a more realistic feeling while playing a simulated, natural instrument and to constrain synthetic sounds in their dynamic range.

The Gigasampler format offers three function types called *nonlinear*, *linear* and *special.* To increase the possibilities for a sound programmer, these functions were not realized by the Gigasampler developers as simple monadic functions but as three figure functions $f \in (V, D, S) \to A$, where $V = \{0, 1, \ldots, 127\}$ is as always the velocity of the keystroke, $D = \{0, 1, 2, 3, 4\}$ is the so called *Response Depth* and $S = \{0, 1, \ldots, 127\}$ is the so called *Curve Scaling.* The first argument can of course only be determined at runtime, the other two arguments are already constantly defined by the instrument patch file, thus is already given by the sound programmer. The resulting *Velocity Response Curves* can differ quite a lot between various Gigasampler instruments. Of course there is no specification of Gigasampler's velocity functions publicly available, so we had to approximate them. Figure 1 shows some measurements made by Mark Knecht, which represent snapshots of the original velocity curves of Gigasampler.

For approximation I decided to use a square function of form:

$$f(v, d, s) = c_1 v^2 + c_2 d^2 + c_3 s^2 + c_4 v + c_5 d + c_6 s + c_7 \tag{1}$$

which reduces the search of an appropriate function to a *Linear Least Squares Error Problem*, which can be solved by using the following approach:

$$A^T A \vec{c} = A^T \vec{b}$$

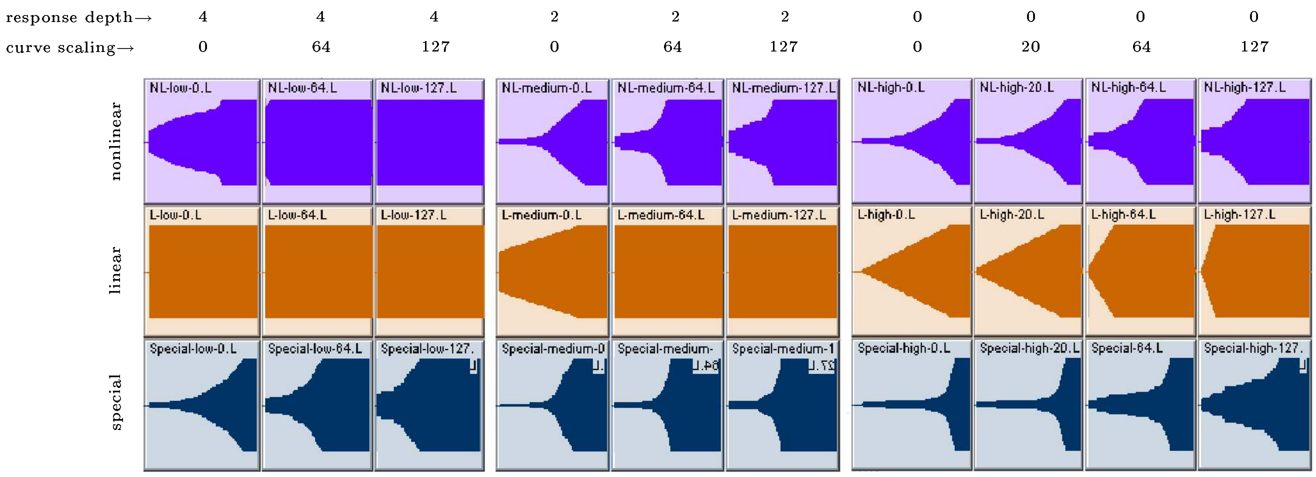

Figure 1: measured Velocity Response Curves of Gigasampler

where

$$\vec{b} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

with $y_1, y_2, \ldots, y_n$ as ordinates, thus the measured snapshot values of the original curves. In our case, that is, with the chosen function (1), the sought coefficient vector is

$$\vec{c} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_7 \end{pmatrix}$$

The matrix A is calculated by inserting the abscissae, that is by insering the respective function arguments $v_1, d_1, s_1, v_2, d_2, s_2, \ldots, v_n, d_n, s_n$, used by the measurement of the snapshots, into our chosen function form (1):

$$A = \begin{pmatrix} {v_1}^2 & {d_1}^2 & {s_1}^2 & v_1 & d_1 & s_1 & 1 \\ {v_2}^2 & {d_2}^2 & {s_2}^2 & v_2 & d_2 & s_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ {v_n}^2 & {d_n}^2 & {s_n}^2 & v_n & d_n & s_n & 1 \end{pmatrix}$$

The matrix A^T is the transposed matrix of A. With $A^T A$ we obtain the square matrix A', in our case $A' \in \mathbb{R}^{7\times7}$. With $A^T\vec{b}$ we obtain the vector $\vec{b}'$ with same dimension than A' has rows and columns, thus in our case $\vec{b}' \in \mathbb{R}^7$. Now the linear equation system given by

$$A'\vec{c} = \vec{b}'$$

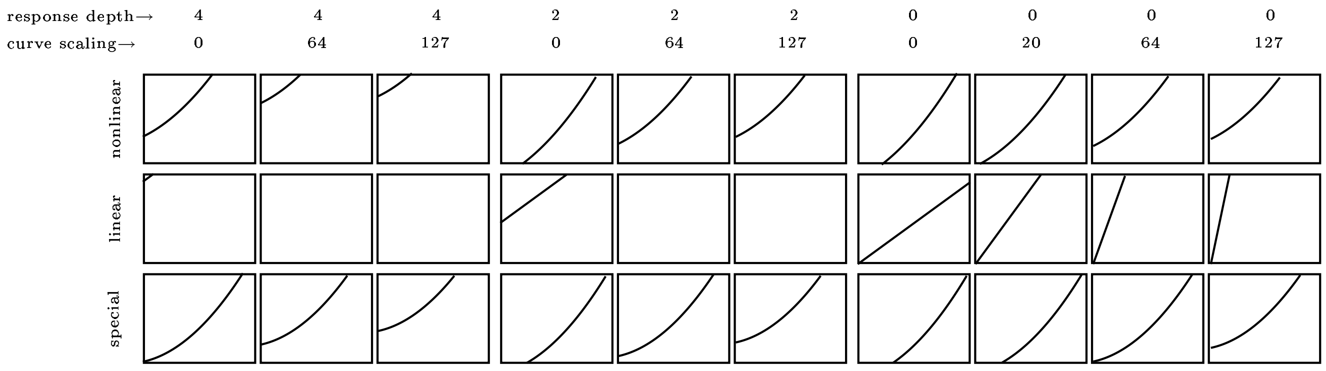

Figure 2: approximated Velocity Response Curves (compare with fig. 1)

is not overdetermined anymore and as we have inserted appropriate measurement values of course, this means that A' is regular and the linear equation system has thus an unambiguous, granted solution $\vec{c}$ which can be calculated e.g. by using the Gauss Elimination Algorithm. This can be conveniently done e.g. by using the computer programs Maxima or Maple. And here are the calculated approximated functions as of today (November 20, 2004) in use by LinuxSampler's `gig::Engine` (or more specific; contained in `libgig`):

The approximated square function for *nonlinear* is:

$$f_{nlin}(v,d,s) = 0.0005143775427v^2 + 0.05318278732d^2 - 0.0003560390502s^2 + 0.04683631221v - 0.9484386143d + 0.08030068910s - 0.1666235937$$

The approximated square function for *special* is:

$$f_{spec}(v,d,s) = 0.000005794551347v^2 + 0.008361491099d^2 - 0.000004303475615s^2 + 0.002085522765v + 0.01313747345d + 0.003220916836s - 0.1630504921$$

For comparison with the original measurements in figure 1 the approximated functions can be seen in figure 2. Note that the *linear* function in the middle row doesn't reflect the function used currently in LinuxSampler, means the *linear* function shown in figure 2 differs slightly from the one used in `libgig` currently. Sorry, I was too lazy at the moment to fix it. ;-)

CU
Christian