Charles R. Sullivan
Princeton University
Princeton, New Jersey 08540 USA

Extending the Karplus-Strong Algorithm to Synthesize Electric Guitar Timbres with Distortion and Feedback

Rock guitarists often make use of distortion and feedback to create special effects. This paper describes a software synthesis algorithm that makes these effects available to the computer musician. Plucked strings simulated by the Karplus-Strong algorithm (Karplus and Strong 1983) are incorporated into a system with distortion and feedback. For use with this system, a new implementation of the Karplus-Strong algorithm is developed. The implementation is similar to the implementations of Jaffe and Smith (1983), but a different set of filters is used, which offers the advantages of smooth, continuous frequency changes for glissandi and more flexible control of timbre. Because of these advantages, it is also useful in conventional applications of the Karplus-Strong algorithm without distortion and feedback.

The Karplus-Strong Algorithm

The basic form of the Karplus-Strong algorithm is shown in Fig. 1. A delay line is initially filled with random numbers. The output of the delay line is fed through a low-pass filter back into the input. The output of the low-pass filter is the output of the instrument.

If the output of the delay line were fed back directly into the input instead of through a low-pass filter, the output of the algorithm would be periodic, with a period corresponding to the length of the delay line, and it would have many strong harmonics. It would be a musical note, although a rather uninteresting and unpleasant one. In the actual algorithm, the initial random values are passed repeatedly through the filter. Each time, the filter attenuates the signal so that the amplitude of the note decays exponentially. Because a low-pass filter is used, the higher harmonics decay faster. This results in a surprisingly natural plucked string sound. An external excitation of the string, other than plucking, can be simulated by adding an input, as shown in Fig. 2. This will be used later for simulating feedback in an electric guitar.

Jaffe and Smith (1983), whose methods have been used widely and successfully, discuss a variety of possibilities within and beyond this basic scheme. In the following sections an alternative to their filtering and tuning methods is developed.

Developing a Low-Pass Filter

In the following sections I will make use of digital filter theory. For an introduction to digital filter theory, see Smith (1985). I will use the convention of specifying a frequency, ω, in radians per sample. In terms of the continuous-time frequency, f, in *Hz*, $\omega = 2\pi f/f_s$, where f_s is the sampling frequency, also in *Hz*. Thus $\omega = \pi$ corresponds to the Nyquist frequency (one half the sampling frequency). There are many types of low-pass digital filters. Independent control of the response at low and high frequencies, monotonically decreasing response, and linear phase are desirable characteristics for the low-pass filter to be used in the Karplus-Strong algorithm.

Independent control of the filter's amplitude response at high and low frequencies gives the user of the algorithm the ability to control the decay rates of the fundamental and the harmonics of the signal independently. This gives an extra measure of con-

Computer Music Journal, Vol. 14, No. 3, Fall 1990,

Fig. 1. Basic form of the Karplus-Strong algorithm. The delay line is initially filled with random values.

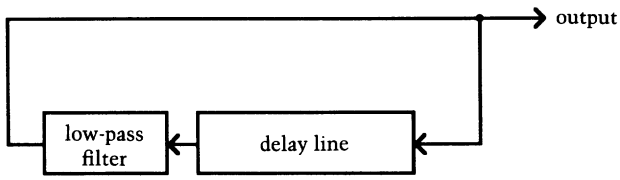
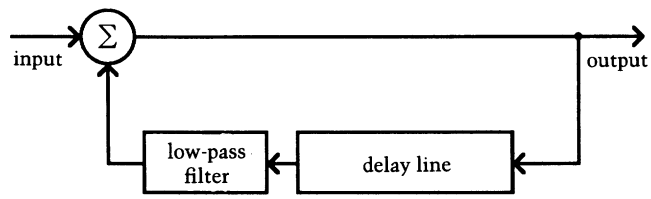


Fig. 2. An input added to the Karplus-Strong algorithm to simulate an external excitation of the string, other than plucking.



trol of the resulting timbre. Monotonically decreasing amplitude response will give the most predictable results; it guarantees that higher frequencies will always decay faster than lower frequencies.

A linear-phase filter is one with constant time delay at all frequencies. Linear phase is desirable because if the total time delay through the delay line and filter is different for different frequencies, the harmonics will be out of tune with the fundamental. Experiments with some nonlinear-phase filters resulted in sounds that were perceived as changing in pitch very slightly as the note decayed. This was because the initial perception of the pitch was based predominantly on the strong harmonics at the beginning of the note, but based more on the fundamental as the harmonics decayed. Because the fundamental was not exactly in tune with the harmonics, the result sounded like a string going out of tune towards the end of the note. No careful listening tests were done to determine at what point this effect was audible and at what point it became objectionable, but it is assumed that a filter closer to linear phase is preferable.

One filter that can satisfy the criteria of linear phase, monotonically decreasing response, and independently adjustable response at two different frequencies is the three-point FIR (Finite Impulse Response) linear-phase filter described by the difference equation

$$y_n = a_0 x_n + a_1 x_{n-1} + a_0 x_{n-2},$$

where the x_n values are the inputs for sample n; y_n is the output sample n; and the a_n values are multiplier factors. The phase response of this filter is simply a constant delay of one sample. Its amplitude response is

$$|H_1(\omega)| = |2a_0 \cos\omega + a_1|, \qquad \text{(Eq. 1)}$$

where ω is the frequency in radians per sample. If $a_1 \geq 2a_0 \geq 0$, the response is monotonically decreasing or constant. The attenuation, and thus the decay time, may be specified at two frequencies: the fundamental and a representative higher frequency. For convenience the Nyquist frequency is chosen for this higher frequency. If the specified attenuation at the higher frequency is greater than or equal to the specified attenuation at the lower frequency, it will be possible to choose coefficients to meet these specifications and satisfy the condition $a_1 \geq 2a_0 \geq 0$.

One problem with this filter is that with some coefficient values it has a gain equal to or greater than one at zero frequency, or *dc*. This means that any *dc* content in the original random numbers or generated by round-off error during the computations involved in filtering may not decay, and so can remain at the end of the note after everything else has decayed. The switch from this constant *dc* output to zero output at the end of a note would produce an annoying click. Even worse, the *dc* level could grow over the course of the note.

Jaffe and Smith (1983) deal with this problem by quickly attenuating the signal, including *dc*, at the end of the note. One can also eliminate the average value of the initial random numbers before the note starts. These techniques are usually adequate for filters with a gain of one at zero frequency, such as the ones used by Jaffe and Smith. However, with filters with gain greater than one, they are not adequate. Jaffe and Smith's method will not work well because fast attenuation of a large amount of *dc* in the output can result in a thump. Even if *dc* is removed from the initial values, *dc* created by round-off error can grow to substantial levels. The amount of *dc* can even get large enough to cause overflow or loss of precision in the filtering calculations.

In order to avoid these *dc* problems, the above-described low-pass filter requires a high-pass or *dc*-

blocking filter in addition to subtracting *dc* from the initial values. The high-pass filter described by the difference equation

$$y_n = \hat{a}_0 x_n + \hat{a}_1 x_{n-1} + b_1 y_{n-1}$$

has zero response at zero frequency and essentially no effect on frequencies at or above the fundamental frequency if the coefficients are chosen as follows:

$$\hat{a}_0 = \frac{1}{1 + \omega_{co}/2}$$
$$\hat{a}_1 = -\hat{a}_0$$
$$b_1 = \hat{a}_0(1 - \omega_{co}/2), \qquad \text{(Fig. 2)}$$

where ω_{co} is a cutoff frequency chosen well below the fundamental frequency of the note to be played. For example, making ω_{co} a factor of ten lower worked well. The amplitude response of this filter is

$$|H_2(\omega)| = \frac{1}{\sqrt{1 + (\omega_{co}/2)\cot(\omega/2)}}.$$

Exact Tuning

There is still one problem with this algorithm. It does not allow a continuous choice of frequencies. Only those frequencies whose periods are integral multiples of the sampling period can be used. Some way of allowing the delay to be adjusted more continuously is desired. One can achieve this by the use of an all-pass filter (Jaffe and Smith 1983), but there are two unfortunate side effects. First, such an all-pass filter is not linear-phase. Second, the delay must be adjusted continuously for a glissando—or at least in very small steps—while the note is being played. This can be done with an all-pass filter by gradually changing the delay it introduces. However, the all-pass filter's phase response gets much less linear as its delay is increased beyond one sample. Thus for glissandi between distant pitches, severe problems with the tuning of harmonics are encountered during some parts of the note. Another way to produce a glissando would be to gradually increase the delay time from the all-pass filter until it reached a delay of one sample, at which point the delay line's length would be increased and the all-pass filter set back to produce a small delay. This would keep the harmonics in tune better, but the sudden change would produce a glitch; a smooth glissando would not be produced.

Another way of producing continuously variable tuning is to use interpolation between samples at the end of the delay line. A simple linear interpolation can be represented by

$$y_n = c_0 x_n + c_1 x_{n-1},$$

where c_0 is varied between one and zero to achieve delays between zero and one sample, and $c_1 = 1 - c_0$. Readers familiar with Jaffe and Smith's implementation will note that this is the same as the two-point averaging, low-pass filter they used to make the note decay. If the same two-point filter is used for tuning, it will still have attenuation and will make the note decay faster. In Jaffe and Smith's implementation, the coefficients were chosen to achieve the desired decay rate. Here, however, the coefficients must be chosen to give the desired delay time. The decay rate would then be uncontrolled, except that the coefficients of the three-point averaging filter, $H_1(\omega)$, can be chosen with the knowledge of what the response of the two-point filter will be. The three-point filter is made to provide only the additional attenuation needed.

Thus the user's specifications of decay time can be met, except in the case when a long decay time at high frequencies is specified, and the frequency specified is such that the interpolation produces a large high-frequency attenuation. In such a case, the three-point filter would have to boost high frequencies to meet the specifications. This is not recommended, because the composite response could then have peaks in response that would cause some harmonics to decay more slowly than the fundamental or even to grow. Instead of this, a faster high frequency decay time is allowed in such a case so that only the fundamental frequency decay time is exactly as specified.

Glissandi can now be created in a straightforward way. For example, to go down in pitch, c_0 is gradually decreased until it is zero and c_1 is one. Now the filter is simply a delay of one sample. The delay of the delay line can be increased by one sample,

Fig. 3. Phase delay of filters that may be used for exact tuning. Linear interpolation ($H_3[\omega]$) is compared with the all-pass filter used by Jaffe and Smith ($H_c[\omega]$). Filters set for a delay of 0.25 sample at $\omega_0 = 0.25\pi$, (a); filters set for a delay of 0.45 sample at $\omega_0 = 0.25\pi$ (b); and filters set for a delay of 0.6 sample at $\omega_0 = 0.25\omega$ (c).

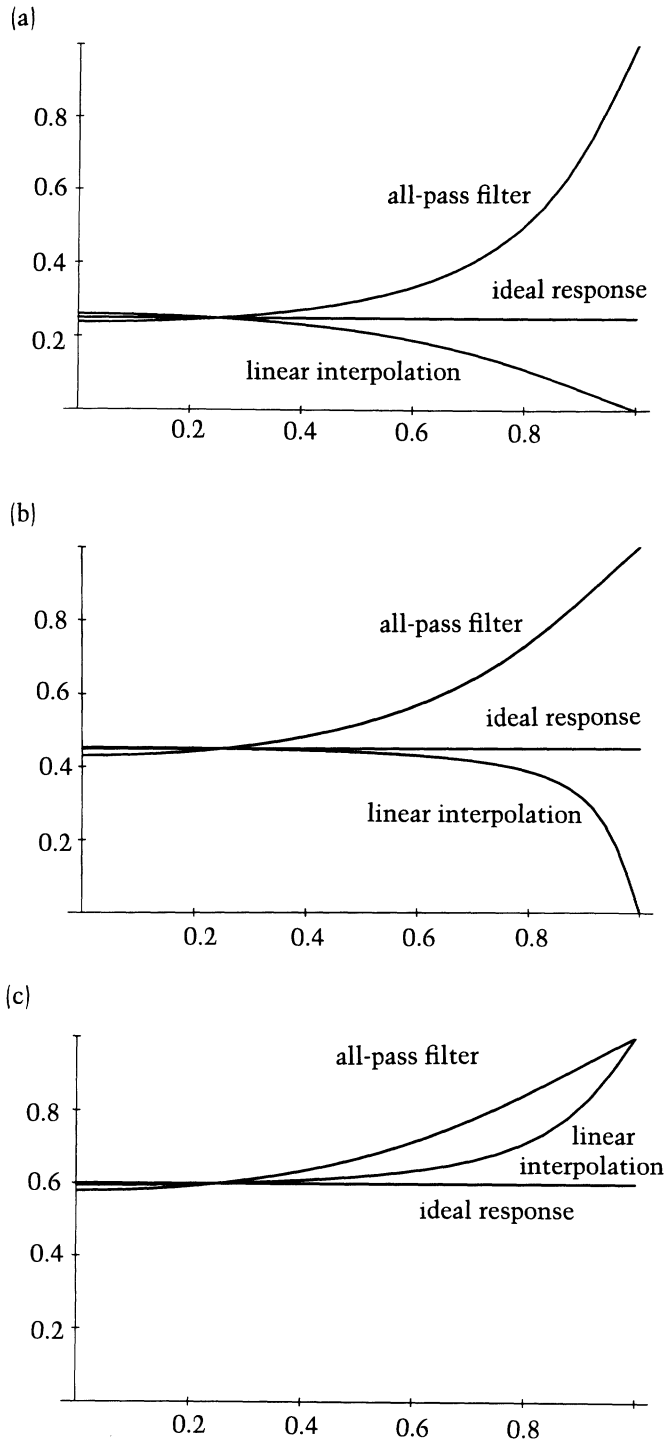
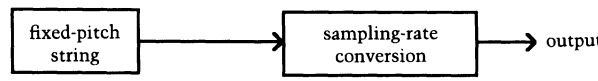


Fig. 4. Sampling-rate conversion for pitch adjustment of the Karplus-Strong algorithm.



and c_0 set back to one, without producing a glitch, because a delay of one sample in either place is equivalent.

The two-point averaging filter for interpolation is not always linear-phase, but it is not far off. It is, in fact, closer to linear phase than the all-pass filter used by Jaffe and Smith. Figure 3 shows a comparison of the phase delay of the two filters.

The all-pass filter's difference equation is

$$y_n = cx_n + x_{n-1} - cy_{n-1},$$

which has z-transform

$$H_c(z) = \frac{c + z^{-1}}{1 + cz^{-1}}.$$

Its phase delay is

$$-\frac{\arg[H_c(\omega)]}{\omega} = \frac{1}{\omega}\tan^{-1}\left(\frac{-c\sin\omega}{1 + c\cos\omega}\right) - \frac{1}{\omega}\tan^{-1}\left(\frac{-\sin\omega}{c + \cos\omega}\right)$$

The z-transform of the two point averaging filter (linear interpolation) is,

$$H_3(z) = c_0 + c_1 z^{-1},$$

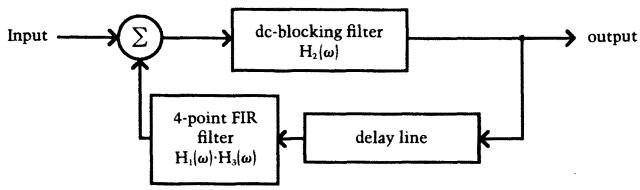
and its phase delay is,

$$-\frac{\arg[H_c(\omega)]}{\omega} = \frac{1}{\omega}\tan^{-1}\left(\frac{c_1\sin\omega}{c_0 + c_1\cos\omega}\right).$$

An altogether different approach to achieving continuously variable tuning is to always use the same length delay line and to change the pitch of the resulting sound through sampling-rate conversion, as shown in Fig. 4.

This is the approach taken in the CSOUND system (Vercoe 1988). One difficulty with this approach is that aliasing can produce undesired foldover frequencies in the output. Filtering could be used to reduce this, but a satisfactory filter might require much computation, and, ideally, a different filter would need to be designed for each new out-

Fig. 5. Implementation of the Karplus-Strong algorithm. The input is optional.



put pitch. Aliasing problems could also be avoided by starting the model with the sample buffer filled with a band-limited signal rather than with broadband noise, so that no filtering would be required.

An Implementation

The Karplus-Strong algorithm was implemented with the low-pass and high-pass filters described above and with linear interpolation for tuning. Although the computation of the filter coefficients is somewhat complex, for notes without glissando or vibrato it is only done once at the beginning of the note, and so is unimportant compared to computations that must be done once for each sample. To reduce the once-per-sample computation a bit, the coefficients for the two-point interpolation and three-point, low-pass filters can be combined via convolution to yield the coefficients of a single four-point filter, $H_{total}(\omega)$, with the same response as the first two cascaded, $H_1(\omega).H_3(\omega)$. The block diagram of this algorithm is shown in Fig. 5 and a procedure for the coefficient computation is given in the Appendix.

Compared to the implementation proposed by Jaffe and Smith, this implementation uses more computations per sample: seven multiplications compared to three. In some cases this may preclude the use of this implementation, but in other cases the complete elimination of *dc*, closer approximation of linear-phase, independent control of decay time at two frequencies, and ease of producing glissandi will be worth it.

One final modification was added. Although the system as it stands allows variations in relative decay time of harmonics, it does not allow variation of the relative initial amplitudes of different harmonics. Passing the initial random values through a low-pass filter before starting the note has been suggested for this purpose. Jaffe and Smith used such a filter, with filter coefficients chosen to give the desired timbre. In this implementation, the filter instead always has the same frequency response, but the user of the algorithm selects the number of times the initial values are passed through the filter before the start of the note in order to control the timbre. The filter used was another three-point averaging filter,

$$y_n = a(x_n + x_{n-1} + x_{n-2}),$$

which has response

$$H_4(\omega) = a(2\cos\omega + 1).$$

The value of a affects only the overall gain, and not the shape of the frequency response. If

$$a = |2\cos\omega_0 + 1|^{-1},$$

the amplitudes of the harmonics are reduced, but the amplitude of the fundamental is not affected. More passes through this filter make it sound more like the string was plucked with a soft finger instead of a hard pick.

The results of this implementation were encouraging. The combination of a variable decay rate for harmonics and a variable initial harmonic content allow the production of a wide variety of plucked string sounds. Fast decay rates and a large initial harmonic content produce sounds reminiscent of a plucked violin. Slow decay rates and a relatively small initial harmonic content produce a sound more like an electric guitar or an electric bass. Changing filter coefficients in the course of a note allows switching to a fast decay rate just before the note ends for a smooth termination when the decay time is much longer than the duration of the note. A sudden change in the length of the delay line in the middle of a note produces an effect similar to changing fingering of a note on a guitar without plucking the string again.

A repeatedly changing delay time can produce a smooth glissando effect. Ideally, the delay time should be reset once per sample to create a glissando. This would require many computations, because the complete calculation of filter coefficients must be done whenever the delay time is changed.

Computation could be saved if the delay changed less often. For example, it could be changed 1000 times per second, but this would produce objectionable frequency components at 1000 Hz and its harmonics, or at modulation products of these and the frequencies in the sound of the string. To avoid this problem, the coefficients are changed once per cycle of the current pitch, so that the extra frequency components are at the same frequencies as the sound of the string and are not noticeable with ordinary glissandi and vibrato. Note that this approach would not work with extremely fast vibrato or frequency modulation, where the modulation rate is near the frequency of the string.

Distortion

Distortion is often used by electric guitarists to modify their sound. The technique was originated by guitarists who turned up the volume on their amplifiers high enough that clipping or other nonlinear distortion occurred and discovered that a rich sound resulted.

If a single note is played through some distortion, the output is that same note, but with added harmonic content. Because distortion is nonlinear, the result of distorting the sum of several simultaneous notes is different from the sum of the individually distorted notes. The signal resulting from distorting the sum can contain frequency components at frequencies not present in the input, such as the sums and differences of the input frequencies. Because guitars normally sum the output of the pickups for each of the strings before distortion is added, these new frequency components are produced and are an important part of the sound we associate with distortion on electric guitars.

If the distortion can be represented by a transfer function whose output depends only on the present input, and not on previous inputs, the output will have the same periodicity as the input. Thus, if the input is the sum of two periodic functions, the output will be periodic with a fundamental frequency equal to the greatest common divisor of the fundamental frequencies of the two inputs. Depending on the exact kind of distortion used and the exact waveforms of the two inputs, the output will contain frequency components in differing amounts at the greatest common divisor frequency and all its harmonics.

Musically, this can work out either very nicely or very poorly. Consider a C major triad. For the sake of simplicity assume that it is played with just intonation and that the sum will be periodic at a frequency two octaves below the root. The output will consist only of this low C and its harmonics, and so the chord will be filled out with some of the following notes: C two octaves down, C one octave down, G a fourth down,the original triad, and possibly something close to a B♭, the seventh harmonic of the subfundamental. Many higher harmonics of these notes will also be added. This accounts for the rich sound one gets with simple major chords played through distortion.

On the other hand, if a more complex chord is played, the distortion products may not be at such desirable frequencies. A complex chord is periodic at a very low frequency or may not be periodic at all, depending on the tuning system used. Distorting it creates distortion products at so many different frequencies that the result becomes hard to distinguish from a burst of noise, and the identity of the original chord can be lost. For example, if a B diminished seventh chord (B, D, F, A♭) is constructed from a C major scale with just intonation, the frequencies of B, D, and F are $15f_c/8$, $18f_c/8$, and $8f_c/3$, where f_c is the frequency of the C. Using the fact that a minor third is a ratio of 6/5, the frequency of the A♭ can be chosen as 6/5 the frequency of the F, or $16f_c/5$. The greatest common divisor of all these frequencies is $f_c/120$. The frequencies of the different distortion products are then spaced a minimum of $f_c/120$ apart in frequency. Near the B this is about 7.7 cents, so that there are roughly 13 possible distortion products per equally tempered half step.

Adding Distortion Digitally

The easiest kind of distortion to produce is simple clipping, as shown in Fig. 6.

The input is fed straight through, unless it is above some threshold, in which case the output is

Fig. 6. Simple clipping function for adding distortion.

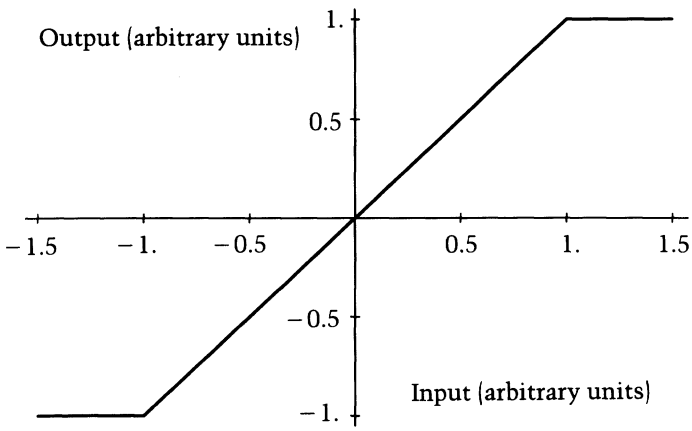
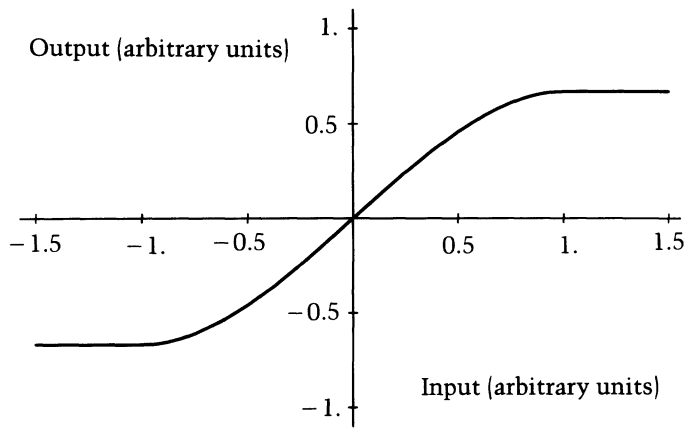


Fig. 7. Soft clipping function for adding distortion. Allows a broader range of distortion effects.



equal to that threshold. The problem here is that as the input level is gradually increased, there is no distortion at all until the threshold is crossed, at which point a lot of high harmonic content is suddenly added. Some of this harmonic content may be above the Nyquist frequency, in which case it will come out of a digital implementation at the sampling frequency minus the original frequency due to aliasing. While the resulting harsh sound may be desirable in an all-out distortion, hard-core rock context, more control over intermediate sounds would be desirable.

A much more gradual transition is achieved if the function,

$$f(x) = \begin{cases} 2/3; & x \geq 1 \\ x - x^3/3; & -1 < x < 1 \\ -2/3; & x \leq -1, \end{cases}$$

shown in Fig. 7 is used.

This function allows a wide range of distortion effects from undistorted sounds at low input levels to the grossly distorted sounds that result when the absolute value of x is almost always above one. The difference between this and simple clipping is somewhat akin to the difference between the hard clipping usually observed in transistor amplifiers and the softer clipping usually observed in tube amplifiers, which guitarists often prefer. This soft-clipping distortion function can either be implemented directly or with a look-up table. Direct implementation requires only three multiplications, is quite simple, and does not have the quantization noise problems that a small look-up table without interpolation could have at low input levels.

Feedback

When an electric guitar is near the speaker that is producing its sound, the sound from the speaker can excite the strings of the guitar. If there is sufficient gain in the amplifier, this can produce a growing oscillation at one of the frequencies of the strings or at one of their harmonics. The growth of the oscillation is limited by nonlinearities, such as distortion in the amplifier. Feedback can be an annoyance if unwanted noises are accidentally produced when the amplifier is turned up too much. However, it can also be useful as it allows a note to be sustained indefinitely once a steady level limited by distortion is reached. In addition, the metamorphosis from the note or chord originally played to a steady tone sustained by feedback is a unique timbre that can be quite interesting.

Although feedback can be hard to control, there are several ways that guitarists can influence what sounds they get. The system of an electric guitar with feedback is illustrated in Fig. 8.

An increase in gain in the preamplifier before the point where the distortion occurs increases the gain around the feedback loop, making it easier for oscillation to occur and for the oscillation to grow

Fig. 8. An electric guitar with distortion and feedback.

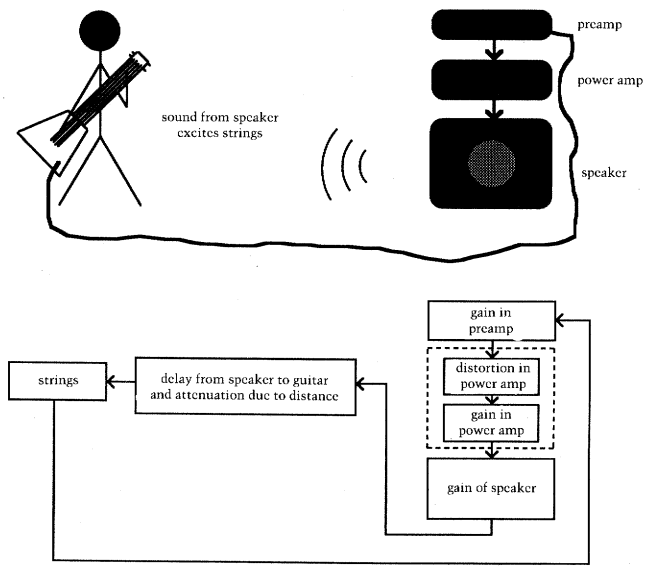
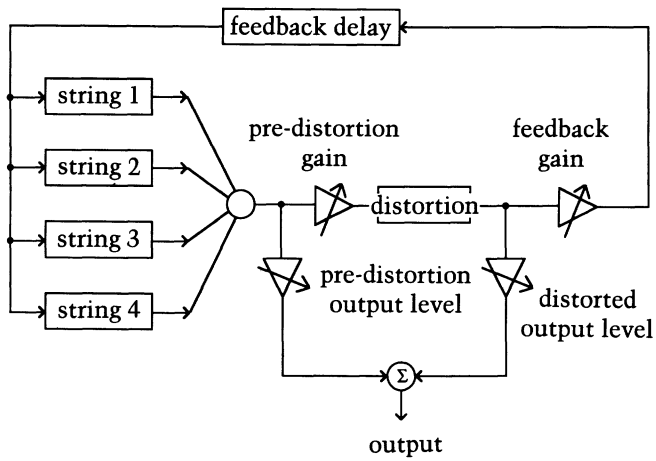


Fig. 9. Software synthesis system for electric guitar sounds.



faster. After the distortion, an increase in gain in the power amplifier has the same effect, and it also directly increases the steady-state output level of the oscillation. Moving the guitar closer to the speaker increases the gain around the loop, having the same effect as an increase in gain in the preamplifier. It also changes the delay time around the loop. Because the distance between the guitar and the speaker is usually about as long as, or longer than, the wavelengths of sound involved, it can be significant in determining the frequencies at which the oscillation due to feedback occurs. Oscillation is favored at frequencies at which the delayed sound coming form the amplifier is in phase with the motion of the string.

Figure 9 shows how the model shown in Fig. 8 can be implemented on a software synthesis system using the string and distortion algorithms discussed above. This system produces sounds that subjectively resemble those of a real electric guitar used very closely with feedback and distortion.

The amount of distortion can be controlled by controlling the initial amplitude of the random sequence or by adjusting the gain before the distortion unit. The output can be taken either before or after the distortion or the two outputs can be mixed, allowing further control over the amount and timbre of distortion. If the output is taken only from before the distortion, the distortion only serves to limit the growth of the feedback. If frequencies other than the resonant frequencies of the strings are produced by distortion and fed back, they will be filtered out by the strings; thus no significant amounts will appear in the output.

The gain around the feedback loop can be adjusted, as can the delay time. It is easiest to specify delay time in terms of the pitch whose period is equal to the delay time. This pitch and its harmonics will then be favored for feedback. The plucked-string algorithm with adjustable decay time at low and high frequencies is useful because adjustments to the decay time of the strings at high and low frequencies help determine whether high or low harmonics will be favored in feedback. The DC-blocking, high-pass filter in the string is useful in preventing a growing DC offset due to feedback.

Because the simulation allows exact specification of all these parameters and allows adjustment of them over the course of a note, with a bit of trial and error it is possible to get feedback to happen exactly when it is desired and at exactly the right pitches—something that is very difficult with a real guitar. It is even possible to get feedback at a frequency that is not a note in the original chord or one of its harmonics. This can be done by starting the system with plucked strings at all the pitches desired in the initial chord and an additional string set up to oscillate at the desired

Fig. 10. Deriving the output from a different point in the delay line. Unlike a different pickup position on an electric guitar, this does not significantly alter the timbre: m *can be any positive integer less than* n.

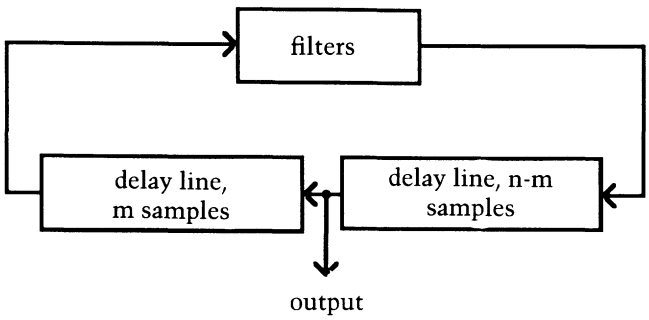
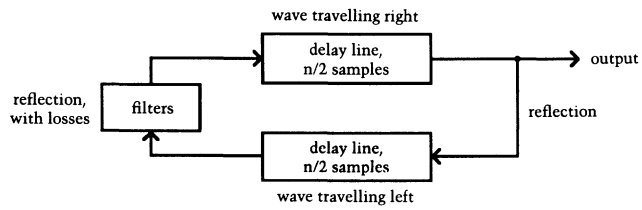


Fig. 11. The Karplus-Strong algorithm viewed as two delay lines, each representing the wave travelling left in one direction on a string.



feedback frequency, but initially filled with zeros instead of random numbers. The feedback delay length can then be chosen to favor this frequency, so that the initial chord will change into an oscillation at this frequency. As with the string algorithm alone, notes can be ended smoothly by shortening decay times of the strings just before the note is to end, but the gain around the feedback loop should also be set to zero. Frequency changes during a note and glissandi can be created in the same way as before.

What's Missing

The algorithm described here produces realistic electric guitar sounds with feedback and distortion. There are, however, a few aspects of electric guitar sound that have been ignored. These include the position of the pickup, the position at which the strings are plucked, the resonances of the guitar body, the frequency response of the amplifier and speaker used, and any other processing that may be done to the signal. I will address these issues below.

Pickup Position

An electric guitar has a pickup that electromagnetically senses the displacement of the string. The pickup only senses the displacement of the string in an area small enough that it can be considered to be a point. If the string is excited in a mode that has a node at that point, the pickup will detect nothing. Thus the placement of the pickup can considerably influence the amounts of different harmonics present in the output. Guitars are usually fitted with several pickups mounted in different positions so that the player can choose the desired mixture of harmonics by electrically mixing the outputs of the different pickups.

The Karplus-Strong algorithm is essentially a loop, a delay with the output tied back around to the input. One might be tempted to try taking the output from a different point in this loop, as in Fig. 10, but this would have no effect on the resulting mixture of harmonics.

On a real string, the waves do not go around in a circle, but rather go back and forth. The displacement of the string is really the superposition of two waves travelling in opposite directions, reflecting off the ends. The delay line in the Karplus-Strong algorithm is equivalent to two delay lines, each half the length of the original one, as shown in Fig. 11.

Each of these can be considered to represent the wave moving in one direction. Actually, there should be a multiplication by -1 at each direction change, because an inversion occurs on a string when the wave reflects from a fixed support, but because the two sign inversions cancel each other out, the multiplication is not necessary. The displacement of the string at a given point is the sum of the displacements due to waves travelling in both directions, represented by the sum of the values in the two delay lines, or rather the difference, because the sign inversions are omitted at the reflection points.

Figure 12 shows the Karplus-Strong algorithm with such an output representing the displacement of the string. The output is the signal minus a delayed version of itself. This is equivalent to the original system with a comb filter added on the output, as shown in Fig. 13.

Fig. 12. An output representing the displacement of the string at a given point. This output is similar to what an electric guitar pickup would sense; m can be any positive integer less than n/2.

Fig. 13. A system equivalent to the system in Fig. 12. If P is the sampling period in seconds, 2mP corresponds to t of Eq. (3) in the text.

Fig. 14. Initial values of delay elements for plucked string. The configuration of a real string when plucked a); the values of delay elements to represent this when two delay lines are used as in Fig. 9. b); and how this is converted to a set of values for a single delay line c).

Fig. 12

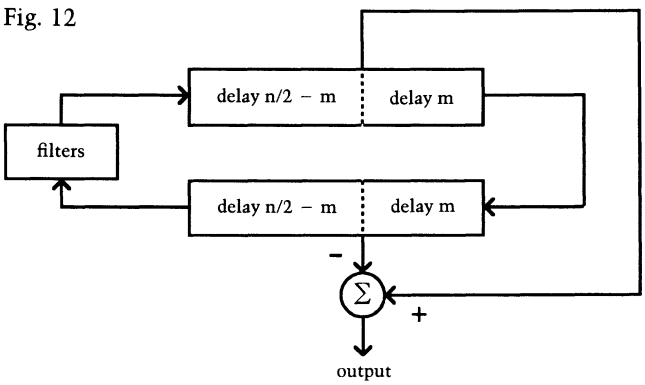
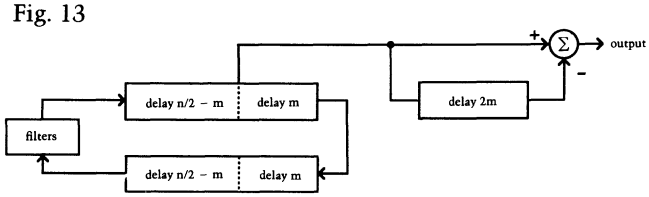
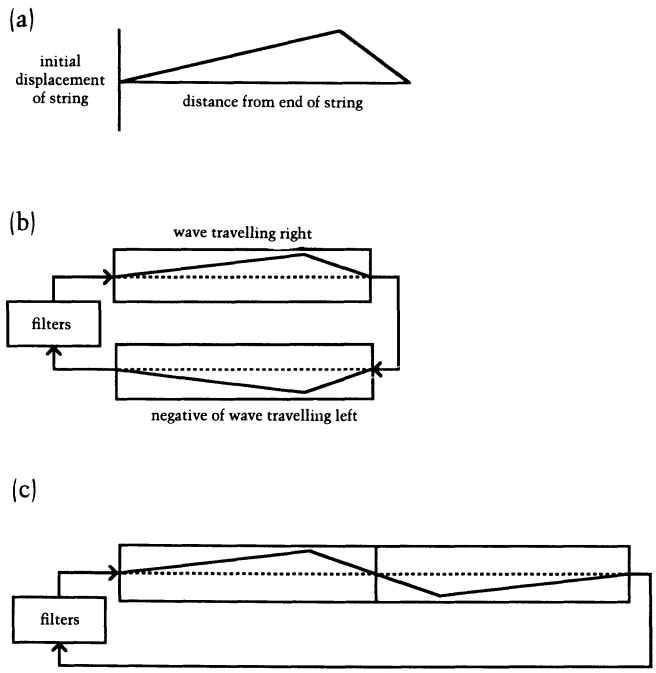


Fig. 13





The delay time for this filter is

$$t = \frac{d}{D} T \qquad (3)$$

where T is the period of the note, d is the distance from the end of the string to the pickup, and D is the length of the string. The original version is like having a sophisticated pickup that only picks up waves travelling in one direction, thus emphasizing all harmonics equally. It is an idealized version of the string-pickup system.

Plucking the string in different positions can also be simulated using this relationship between the Karplus-Strong algorithm and a physical model of a real string. Instead of filling the delay line with random numbers, it is possible to calculate an initial set of values based on the initial string deflection when a string is plucked. When a point on a string is pulled off to one side, held still, and then released, waves propagate in both directions from the disturbance. Thus, analogously, the two delay lines representing the waves travelling in the two directions should be filled to represent the same displacement. One is given the opposite sign from the other, because again, the multiplication by -1 is omitted at the ends of the string. Figure 14 shows an example of this.

Initial shapes for this can be triangles, representing the initial shape of a string that has been pulled aside and is waiting to be released, or pulses, representing the shape of a string that has just received a short impulse from a small, hard hammer. A wider or softer hammer can be simulated by the same type of filtering that was used to reduce harmonics in the random initial values.

Other Factors

Electric guitar bodies can also significantly color the instrument's sound. This is particularly true of semiacoustic guitars. Digital filters, which could be added after the output of the string algorithm to simulate this, would probably be satisfactory. However, the resonances of the guitar body actually affect how fast various harmonics decay, so a more complicated filter within the loop of the string al-

gorithm would be required to simulate the actual effect of body resonances.

Amplifiers and speakers for electric guitars are very different from high fidelity music reproduction equipment. Because their goal is not to reproduce a sound accurately, but rather to create a new sound, flat frequency response is not needed. Various equalizers in the amplifier and the speaker itself add significant coloration. If this coloration is desired from a computer synthesis system, various digital filters could be added to mimic the coloration, or the output from the computer's digital-to-analog converters could simply be plugged into a guitar amplifier.

With a real electric guitar, the various filtering effects discussed above are actually inside the feedback loop. In a simulation, it is probably best to add them outside the loop, so that they do not make control of feedback frequencies more difficult.

The modern electric guitarist's equipment typically includes many electronic processing devices inserted between the guitar and the amplifier. These types of alterations to the signal, or effects, can be created by appropriate processing on a computer system or by devices intended for use with real guitars, used after digital-to-analog conversion. Reverberation, "wah-wah" pedals, sustain, and phasing are among the most popular effects. Time-reversed reverberation is one effect that can be created with a software synthesis system, and it works particularly well with electric guitar sounds. It is properly created by time-reversing the signal, adding reverberation, and then reversing the signal back to its original time sequence, so it is not available to the guitarist performing in real-time, although there are some simulations of it that work in almost real-time.

Conclusion

A refined implementation of the Karplus-Strong algorithm has been presented here. As an alternative to the methods of Jaffe and Smith (1983), it offers the advantages of smooth, continuous frequency changes for glissandi and more control of timbre. A system incorporating this implementation has been used to synthesize electric guitar sounds, providing convincing imitations of real instruments. Methods for extending it to create even more convincing imitations have also been discussed. Particularly with feedback, the system is useful in allowing performances that would be technically very difficult or even impossible on a conventional instrument. It is hoped that this algorithm will not only be used to duplicate sounds that can be produced already by electric guitars but will serve as a starting point for creating previously unimagined musical sounds.

Acknowledgments

Thanks to Paul Lansky for his help with the CMIX system and his encouragement throughout the project. Thanks to Thomas M. Parks for patiently listening to my ideas and for helping me find my bugs. For their suggestions and help in editing the manuscript, thanks also to William E. Hopkins, Eline T. Luning Prak, my family, and especially to James Beauchamp.

References

Jaffe, D. A., and J. O. Smith. 1983. "Extensions of the Karplus-Strong Plucked-String Algorithm." *Computer Music Journal* 7(2):56–69.

Karplus, K., and A. Strong. 1983. "Digital Synthesis of Plucked String and Drum Timbres." *Computer Music Journal* 7(2):43–55.

Smith, J. O. 1985. "Fundamentals of Digital Filter Theory." *Computer Music Journal* 9(3):13–23.

Vercoe, B. 1988. *CSOUND, A Manual for the Audio Processing System and Supporting Programs.* Media Laboratory, M.I.T.

Appendix: Calculation of the Filter Coefficients

The filter coefficients for the system shown in Fig. 5 are calculated as follows:

1. The delay time in samples needed for the desired frequency is given by

$$d = 2\pi/\omega_0,$$

where ω_0 is the fundamental frequency of the note.

2. The three-point, linear-phase, low-pass filter, $H_1(\omega)$, has a delay of one. The additional delay needed is $\tilde{n} = d - 1$. The length of the delay line, n, is chosen to be the integer part of $\tilde{n}$. The fractional part is then created by the delay, d_f, of the two-point interpolation filter, $H_3(\omega)$, so that $n + d_f = \tilde{n}$.
3. The coefficients for the two-point filter must be set to give delay d_f, which corresponds to a phase shift, $\phi = -d_f\omega 0$. The filter is of the form $y_n = c_0x_n + c_1x_{n-1}$, where $c_1 + c_0 = 1$, so its transfer function in terms of the z transform is $H_3(z) = c_0 + c_1z^{-1}$. The phase response is the argument of this for $z = ej\omega$. Setting the phase response to ϕ at $\omega = \omega_0$ and solving for c_0 yields

$$c_0 = \frac{1}{1 - \dfrac{1}{\cos\omega_0 + \dfrac{\sin\omega}{\tan\phi}}}$$

4. The desired amplitude response of the complete filter system, $|H_{\text{total}}(\omega)|$, must be calculated at ω_0 and at the Nyquist frequency from the desired decay times with the relation,

$$10^{-(A/20)} = |H_{\text{total}}(\omega)|^N,$$

where A is the attenuation in dB after the specified time and N is the number of cycles of the fundamental frequency in that time.
5. The desired amplitude response of the three-point filter is $|H_1(\omega)| = |H_{\text{total}}(\omega)|/|H_3(\omega)|$, both at ω_0 and at the Nyquist frequency. $|H_3(\omega)|$ can be shown to be

$$|H_3(\omega)| = \sqrt{1 - 2c_1c_0(1 - \cos\omega)}.$$

6. The coefficients for the three-point filter must be calculated from the desired response at ω_0 and at the Nyquist frequency. Using Eq. 1 from the text for the amplitude response and solving for a_1,

$$a_1 = \frac{|H_1(\omega_0)| + (\cos\omega_0).|H_1(\pi)|}{1 + \cos\omega_0}$$

and

$$a_0 = \frac{a_1 - |H_1(\pi)|}{2}.$$

7. If the condition $a_1 \geq 2a_0 \geq 0$ is met, the filter response will be constant or monotonically decreasing. If the decay produced by the two-point filter is faster than that required by the specification, a_0 will be negative according to the above expressions. To insure that the frequency response decreases monotonically, a_0 should in such cases be set to zero, and a_1 to $|H_1(\omega_0)|$, so that H_1 becomes a simple delay of one sample, with gain a_1.
8. The coefficients of the two-point and the three-point filters are combined by a convolution to form the coefficients, $\tilde{a}$, of an equivalent four-point filter

$$\{\tilde{a}_0, \tilde{a}_1, \tilde{a}_2, \tilde{a}_3\} = \{a_0, a_1, a_0\} * \{c_0, c_1\},$$

where $*$ indicates convolution, so that

$$\begin{aligned}\tilde{a}_0 &= a_0c_0,\\ \tilde{a}_1 &= a_0c_1 + a_1c_0,\\ \tilde{a}_2 &= a_1c_1 + a_0c_0,\\ \tilde{a}_3 &= a_0c_1.\end{aligned}$$

The final recursive relation for this filter is

$$y_n = \tilde{a}_0x_n + \tilde{a}_1x_{n-1} + \tilde{a}_2x_{n-2} + \tilde{a}_3x_{n-3}.$$

9. The coefficients of the dc blocking filter are calculated according to the Eq. (2) of the text, with $\omega_{co} = \omega_0/10$.