Capítulo 2

Tabla de ondas y muestreadores

En el capítulo 1 tratamos a las señales de audio como si fluyeran de manera continua con una velocidad de muestreo. La velocidad de las muestras no es realmente una cualidad de la señal de audio, si no que especifica qué tan rápido fluyen las muestras al interior del computador. Pero las señales de audio, en el fondo, no son si no secuencias de números y en la práctica aquí no es un requerimiento que tengan que ser "tocadas" secuencialmente. Otro punto de vista complementario es que ellas pueden ser almacenadas en la memoria y, más tarde, pueden ser leídas en cualquier orden -hacia adelante, hacia atrás, atrás y adelante o demanera totalmente aleatoria. Un rango inexhaustivo de nuevas posibilidades se abre.

Por muchos años (aproximadamente entre 1950 y 1990), la cinta magnética sirvió como el principal medio para almacenar sonidos. Las cintas pasaban hacia atrás y hacia adelante a través de captadores magnéticos para reproducir las señales en tiempo real. Más o menos desde 1995, la forma predominante para almacenar sonidos ha sido guardarlos como señales de audio digital, que pueden ser reproducidas con mayor libertad y facilidad que las cintas magnetofónicas. Muchas formas de utilización que datan de la era de la cinta magnética todavía son de uso corriente, incluyendo el corte, la duplicación, el cambio de velocidad y el reversar en el tiempo. Otras técnicas tales como la *conformación de ondas* han tenido su propio desarrollo en la era digital.

Suponga que tenemos almacenadas señales de audio digital, las cuales son sólo una secuencia de muestras (por ejemplo, números) $x[n]$ para $n = 0,...,N-1$, donde N es la longitud de la secuencia. Entonces, si tenemos una señal de entrada $y[n]$ (la cual podemos imaginar fluyendo en el tiempo real), podemos utilizar sus valores como índices para leer los valores de la señal almacenada $x[n]$. Esta operación, llamada *lectura de tabla de onda* nos da una nueva señal $z[n]$, que se calcula como:

$$z[n] = x[\mathrm{y}[n]]$$

Representamos esquemáticamente esta operación como se muestra en la figura 2.1.

Viene dos complicaciones. La primera que los valores de entrada $y[n]$, podrían quedar por fuera del rango $0,...,N-1$, en cuyo caso la tabla de onda $x[n]$ no tiene valores y la expresión para la salida $z[n]$ queda indefinida. En esta situación podemos escoger *recortar* la entrada, es decir, sustituir por 0 cualquier valor negativo y por $N-1$ cualquier valor N o mayor que este. Alternativamente, podemos preferir envolver completamente la entrada de extremo a extremo. Adoptaremos aquí la convención de que las muestras por fuera del rango están siempre recortadas. Cuando necesitemos envolver completamente, introduciremos otra operación de procesamiento de señal que lo hará por nosotros.

La segunda complicación es que los valores de entrada no necesariamente son enteros; en otras palabras, ellos podrían caer entre los puntos de la tabla de onda. Por el momento, sin embargo, sólo redondearemos hacia el valor entero inferior más cercano del valor de la entrada. A esto lo llamamos lectura de tabla de onda *sin interpolación*, y su definición completa es:

$$z[n] = \begin{cases} x[\lfloor y[n] \rfloor] & \text{si } 0 \le y[n] < N-1 \\ x[0] & \text{si y}[n] < 0 \\ x[N-1] & \text{si y}[n] \ge N-1 \end{cases}$$

(donde $\lfloor y[n] \rfloor$ significa “el entero más grande que no excede el valor de $y[n]$”).

Pictóricamente, utilizamos $y[0]$ (un número) como un sitio sobre el eje horizontal de la tabla de onda que se muestra en la figura 2.1, y la salida $z[0]$ el valor que obtengamos en el eje vertical; igual para $y[1]$ y $z[1]$ y así sucesivamente. El rango “natural” para la entrada $y[n]$ es $0 \leq y[n] < N$. Este rango es diferente del usual de una señal de audio en la salida del computador, la cual va de -1 a 1 en nuestras unidades. Veremos más adelante que el rango de valores de entrada utilizable desde 0 hasta N para la lectura sin interpolación, se estrecha ligeramente si se utiliza la búsqueda con interpolación.

La figura 2.2 (parte a) muestra una tabla de onda y el resultado de utilizar dos diferentes señales de entrada como índices de búsqueda dentro de esta. La tabla de onda contiene 40 puntos, los cuales se numeran de 0 a 39. En la parte (b), una *onda diente de sierra* se utiliza como señal de entrada $y[n]$. Una onda diente de sierra no es nada más que una función de rampa que se repite de extremo a extremo. En este ejemplo el rango de la diente de sierra es de 0 a 40 (lo cual se muestra en el eje vertical). La onda diente de sierra de esta manera lee la tabla de onda de izquierda a derecha -comenzando en el punto 0 hasta el punto final 39- haciéndolo nuevamente cada vez que se repite. En los cincuenta punto que se muestran en la figura 2.2 (parte b) la onda diente de sierra hace dos ciclos y medio. Su período es de veinte muestras, o en otras palabras, la frecuencia (en ciclos por segundo) es $R/20$.

La parte (c) de la figura 2.2 muestra el resultado de aplicar la lectura de tabla de onda utilizando la tabla $x[n]$, para la señal $y[n]$. Ya que la entrada de la diente de sierra simplemente lee el contenido de la tabla de onda de izquierda a derecha, repetidamnte a una velocidad constante de precesión, el resultado será una nueva señal periódica cuya forma de onda (su forma) es derivada de $x[n]$ y cuya frecuencia está determinada por la onda diente de sierra $y[n]$.

Las partes (d) y (e) muestran un ejemplo donde la tabla de onda es leída de manera no uniforme; debido a que la señal de entrada va incrementalmente de 0 a N y luego se a regresa a 0, vemos que la tabla de onda aparece primero hacia adelante, conserva su valor en su punto final y luego va hacia atrás. La tabla es leída de izquierda a derecha y luego, más rápidamente, es leída de derecha a izquierda. Al igual que en el ejemplo previo, la señal de entrada controla la velocidad de precesión mientras las amplitudes de salida son las mismas de la tabla de onda.

2.1 El oscilador tabla de onda

La figura 2.2 sugiere una manera fácil para sintetizar cualquier forma de onda fija deseada a la frecuencia que queramos, utilizando el diagrama de bloque que se muestra en la figura 2.3. El bloque superior es un oscilador -no el oscilador sinusoidal que vimos anteriormente, si no uno que produce ondas diente de sierra en lugar de aquél. Sus valores de salida, tal como se indica a la izquierda del bloque, deben ir desde 0 hasta el tamaño de la tabla de onda N. Este se utiliza como índice dentro del bloque de lectura de la tabla de onda (que se introdujo en la figura 2.1), resultando una forma de onda periódica. La figura 2.3 (parte b) adiciona un generador de envolvente y un multiplicador para controlar la amplitud de la salida de la misma manera a como lo hace para el oscilador sinusoidal mostrado en la figura 1.7. Usualmente se utiliza una tabla de onda con amplitud 1 (pico o RMS), de tal manera que la amplitud de la salida es precisamente de la misma magnitud que la salida del generador de envolvente.

Los osciladores tabla de onda son utilizados con frecuencia para sintetizar sonidos con espectro estático, específico. Para hacer esto usted puede pre-calcular N muestras de cualquier tipo de onda de período N (frecuencia angular

$(2\pi/N)$ adicionándole los elementos de la serie de Fourier. El cálculo para el ajuste de la tabla de onda al principio podría ser significante, pero este puede realizarse de manera anticipada al proceso de síntesis, el cual a su vez podría tener lugar en tiempo real.

Mientras la síntesis aditiva directa de ondas complejas, como se mostró en el capítulo 1, es en principio infinitamente flexible como una técnica para producir tímbricas que varían en el tiempo, la síntesis de tabla de onda es mucho menos costosa en términos de computación, pero requiere el cambio de tablas de onda para variar el timbre. Una técnica intermedia, más flexible y costosa que la simple síntesis de tabla de onda pero menos flexible y menos costosa que la síntesis aditiva, es crear mezclas que varíen con el tiempo entre un pequeño número de tablas de onda. Si el número de tablas de onda es de dos únicamente, se trata en efecto de un cruzamiento atenuado entre las dos tablas de onda, tal como se diagrama en la figura 2.4. Suponga que deseamos usar alguna señal $0 \leq x[n] \leq 1$ para controlar las fuerzas relativas entre los dos tipos de onda, de tal manera que si $x[n] = 0$ obtenemos la primera y si $x[n] = 1$ obtenemos la segunda. Denotando las dos señales para el cruzamiento atenuado como $y[n]$ y $z[n]$, calculamos la señal

$$(1 - x[n])y[n] + x[n]z[n]$$

o de manera equivalente y usualmente más eficiente para calcular,

$$y[n] + x[n](z[n] - y[n])$$

Este cómputo se diagrama en la figura 2.4.

Cuando se utiliza esta técnica para el cruzamiento atenuado entre osciladores de tabla de onda, es deseable mantener las fases de los parciales correspondientes iguales a través de las tablas de onda, de tal manera que sus amplitudes se combinen aditivamente cuando se estén mezclando. De otro lado, si se están utilizando tablas de onda arbitrarias (extraídas, por ejemplo, de un sonido grabado) habrá un efecto de fase cuando las diferentes formas de onda se mezclen.

Este esquema puede extenderse en cadena sucesiva para poder moverse a lo largo de un camino continuo entre una sucesión de timbres. Alternativamente, o en combinación con una cadena extensa, el cruzamiento atenuado puede utilizarse para interpolar entre dos timbres diferentes, por ejemplo como una función de la dinámica musical. Para hacer esto usted deberá preparar dos o más formas de ondas de una sola voz sintetizada ejecutada con dinámicas diferentes, e interpolar entre ellas de manera sucesiva como una función de la salida dinámica que usted requiera.

2.2 Muestreo

El “muestreo” no es más que grabar una señal en vivo dentro de una tabla de onda, y luego ejecutarla de nuevo. (En las muestras comerciales la tabla de onda entera es usualmente denominada “muestra” pero para evitar confusión usaremos la palabra “muestra” para significar un solo número en una señal de audio digital.)

En su forma más sencilla, un “muestreador” es simplemente un oscilador de tabla de onda, como se muestra en la figura 2.3, Sin embargo, en una discusión anterior imaginamos al oscilador funcionando a una frecuencia tan alta que la percibimos como una nota, por lo menos desde los 30 Hz aproximadamente. Para el caso del muestreo, la frecuencia es casi siempre menor de 30 Hz y así el período, de por lo menos 1/30 de segundo, es suficientemente largo como para escuchar los ciclos individuales como eventos separados.

Regresando a la figura 2.2, suponga que en lugar de una tabla de onda de 40

puntos $x[n]$ es un segundo de grabación con una velocidad de muestras original de 44100, de tal manera que tiene 44100 puntos; y digamos que $y[n]$ en la parte (b) de la figura tiene un período de 22050 muestras. Esto nos lleva a una frecuencia de 2 Hertz. Pero lo que escuchamos no es un sonido o tono musical a 2 ciclos por segundo (es demasiado lento para escucharlo como un tono) si no que estamos escuchando la grabación original $x[n]$ ejecutada de manera repetida al doble de la velocidad. Acabamos de re-inventar el muestreo.

En general, si asumimos que la velocidad de las muestras R de la grabación es la misma que la velocidad de las muestras a la salida, que la tabla de ondas tiene N número de muestras e indexamos ésta con una onda diente de sierra cuyo período consta de M número de puntos, la muestra es acelerada o desacelerada por un factor de N/M, igual a Nf/R si f es la frecuencia en Hertz de la diente de sierra <esta fórmula hace que varíe la afinación original de la tabla de onda cuando la diente de sierra cambia de frecuencia, pues si tiene una frecuencia mayor, ocupará menos muestras y la afinación de la tabla de onda subirá, y viceversa>. Si denotamos el factor de transposición por t (de tal manera que por ejemplo, $t = 3/2$ significa una transposición de una quinta perfecta hacia arriba), y si denotamos la transposición en semitonos por h, entonces obtenemos las Fórmulas de Transposición para Tablas de Onda en Lazo

$$t = N/M = Nf/R$$
$$h = 12\log_2(N/M) = 12\log_2(Nf/R)$$

Con frecuencia se conoce la transposición deseada en semitonos (h) y la fórmula se debe resolver para f o para N:

$$f = (2^{h/12}R)/N$$
$$N = (2^{h/12}R)/f$$

Hasta ahora hemos utilizado una diente de sierra como onda de entrada $y[t]$, pero tal como se sugiere en las partes (d) y (e) de la figura 2.2, podemos utilizar cualquier onda que queramos como señal de entrada. En general, la transposición será dependiente del tiempo y está controlada por la manera en que cambia la señal de entrada.

La transposición múltiple t y la transposición en semitonos h están determinadas entonces por las Fórmulas de Transposición Momentáneas para las Tablas de Onda:

$$t[n] = |y[n] - y[n - 1]|$$
$$h[n] = 12\log_2|y[n] - y[n - 1]|$$

(Aquí las barras que cierran "|" significan valor absoluto). Por ejemplo si $y[n] = n$, entonces $z[n] = x[n]$ de tal manera que escucharemos la tabla de onda con su tono original y esto es lo que la fórmula predice ya que, en este caso,

$$y[n] - y[n - 1] = 1$$

De otro lado si $y[n] = 2n$ entonces la tabla de onda se transpone una octava hacia arriba, lo cual es consistente con

$$y[n] - y[n - 1] = 2 \quad <2n - (2n - 2)>$$

Si los valores de $y[n]$ decrecen con n, usted escuchará la muestra hacia atrás, pero la fórmula de transposición dará todavía un multiplicador positivo. Todo esto de acuerdo con la primera Fórmula de Transposición para las Tablas de Ondas en Lazo; si una diente de sierra va de 0 a N, f veces por segundo, la diferencia de las muestras sucesivas es precisamente Nf/R excepto en la muestra

al principio de cada nuevo ciclo.

Es bien sabido que al transponer una grabación también se transpone su timbre -este es el efecto “chipmunk”. No solamente cualquier periodicidad (tal como cuando se eleva a un tono) es transpuesta, si no también las frecuencias de los sobretonos. Algunos timbres, notablemente aquellos de sonidos vocales, tienen rangos de frecuencia carcaterísticos en los cuales los sobretonos son más fuertes que los otros más cercanos.

Tales rangos de frecuencia también son transpuestos y esto es escuchado como un cambio en el timbre. En un lenguaje que será más preciso en la sección 5.1, decimos que la *envolvente espectral* es transpuesta junto con el tono o los tonos.

Tanto en esta como en la sección precendente, consideramos la ejecución de las tablas de onda de manera periódica. En la sección 2.1 la reproducción es repetida tan velozmente, que se eleva a un “tono”, es decir entre 30 y 4000 veces por segundo, aproximadamente la tesitura de un piano. En esta sección asumiremos una tabla de onda de un segundo de longitud, y en este caso factores de transposición “razonables” (menos de cuatro octavas arriba) podrán elevar a una velocidad de repetición por debajo de 30, usualmente mucho menor y hacia abajo, tan grave como se quiera.

El número 30 es significativo por otra razón: es aproximadamente el máximo número de eventos separados que puede discernir el oído en un segundo; por ejemplo, 30 fonemas vocales, o notas melódicas o ataques de un redoblante están cerca de lo que más podemos esperar discernir en un segundo antes de que nuestra habilidad para distinguirlos falle.

Existe un continuo entre muestreadores y osciladores de tabla de onda, de tal forma que el parche de la figura 2.3 puede ser tratado como un muestreador (si la frecuencia de repetición es de menos de 20 Hertz aproximadamente) o como un oscilador de tabla de onda (si la frecuencia es de aproximadamente 40 Hertz). Es posible movernos continuamente entre los dos regímenes. Es más, no es necesario ejecutar una tabla de onda completa en un lazo; con un poco más de aritmética podemos escoger subsegmentos de la tabla de onda, y estos pueden cambiar en longitud y localización continuamente cuando se ejecuta la tabla de onda.

La práctica de ejecutar muchos pequeños segmentos de una tabla de onda en rápida sucesión es usualmente llamada *síntesis granular*.

La figura 2.5 muestra cómo construír un muy simple muestreador de lazo. En la figura, si la frecuencia es f y el tamaño del segmento en número demuestras es s, el factor de transposición de la salida está dado por $t = fs/R$, donde R es la velocidad de muestra a la cual fue grabada la tabla de onda (la cual no necesita ser igual a la velocidad de muestras a la cual trabaja el diagrama de bloque.) En la práctica, esta ecuación deberá resolverse o para f o para s para obtener una transposición deseada.

En la figura, un oscilador diente de sierra controla la localización de la lectura de la tabla de onda, pero los valores inferiores y superiores de la diente de sierra no están especificados de manera estática tal como lo estaban en la figura 2.3; en lugar de esto, el oscilador de diente de sierra simplemente tiene valores de 0 a 1 y este rango se ajusta para seleccionar un segmento deseado de las muestras en la tabla de onda.

Sería bueno especificar la localización del segmento l sea desde su extremo izquierdo (su frontera más baja) o desde el punto medio del segmento; en cualquier caso especificamos la longitud s como un parámetro separado. En el primer caso, comenzamos multiplicando la diente de sierra por s,de tal manera que el rango quede de 0 a s; luego adicionamos l de tal manera que ahora el

rango vaya de l a $l + s$. Para especificar la localización a partir del punto medio del segmento, primero sustraemos 1/2 de la diente de sierra (así su rango queda de -1/2 a 1/2), y luego, tal como se hizo antes, multiplicamos por s (de tal manera que el rango queda entre $-s/2$ y $s/2$) y añadimos l para que el rango quede de $l-s/2$ a $l+s/2$.

En el muestreador de lazo, debemos mantener la continuidad entre el comienzo y el final de los segmentos de la tabla de onda; nos ocuparemos de esto en la próxima sección.

Un detalle adicional es que, si el tamaño del segmento y su localización están cambiando en el tiempo (podrían ser señales de audio digital también, por ejemplo), afectarán el factor de transposición, y la afinación o el timbre de la señal de salida podrían ondular hacia arriba y hacia abajo como consecuencia de esto. La forma más simple para evitar este problema es sincronizar los cambios en los valores de s y l con las discontinuidades regulares de la diente de sierra; ya que la señal salta de manera discontinua, la transposición no está realmente definida aquí de ninguna manera, y, si está haciendo envolvente para esconder la discontinuidad, el efecto de cambio en s y l también queda oculto.

2.3 Muestreadores de envolvente

En las secciones previas consideramos la lectura de una tabla de onda ya fuera de manera esporádica o de manera repetida para hacer un muestreador. En la mayoría de las aplicaciones reales debemos trabajar para que las muestras inicien y paren de manera limpia, de tal manera que la señal de salida no salte de manera discontinua al comienzo y al final de las muestras. Esta discontinuidad puede sonar como un “click” o como un “thump” dependiendo de la tabla de onda.

La manera más fácil de hacerlo, asumiendo que la tabla de onda se ejecutará desde el principio hasta el fin, es simplemente prepararla con anticipación de tal manera que su amplitud entre gradualmente al inicio y se atenúe de igual manera al final. Esto es posible hacerlo incluso cuando se graban las muestras de la tabla de onda en vivo, multiplicando la señal de entrada por una envolvente segmento de línea, que tenga la misma longitud de la grabación.

En muchas situaciones es inconveniente o imposible pre-envolver la tabla de onda -por ejemplo si quisiéramos reproducir sólo una parte de ésta o si quisiéramos cambiar la pendiente de la envolvente dinámicamente. En la sección 1.5 vimos cómo controlar la envolvente de los osciladores sinusoidales utilizando la multiplicación por una función rampa (también conocida como generador de envolvente), y reconstruimos esta noción en los osciladores de tabla de onda de las figuras 2.3 y 2.4. Esto también funciona para iniciar y finalizar las muestras evitando discontinuidades, pero con una gran diferencia: toda vez que en la síntesis de tabla de onda éramos libres de asumir las formas de ondas lineales de extremo a extremo, de tal manera que podíamos escoger cualquier tiempo para esta envolvente, en el caso del muestreo, al utilizar formas de onda no preparadas, estamos obligados a tener valor cero del generador de envolvente al momento de alcanzar el final de la tabla de onda por primera vez. Esta situación se grafica en la figura 2.6.

En situaciones donde una tabla de onda arbitraria debe repetirse tantas veces como se requiera, la forma más sencilla de hacer que el lazo trabaje de manera continua es arreglar las cosas para que el cambio de amplitud se sincronice con el lazo, utilizando una tabla de onda aparte (la envolvente). Esto se puede implementar según se muestra en la figura 2.7. Un solo oscilador diente de sierra se utiliza para calcular los índices de lectura de las dos tablas, la que contiene el sonido grabado y la que hace la forma de la envolvente. De lo que más debemos preocuparnos es que las entradas para ambas tablas de onda tengan su propio rango correspondiente.

En muchas situaciones es deseable combinar dos o más copias del muestreador de lazo de la tabla de onda a la misma frecuencia y a una relación de fase específica. Esto puede hacerse de tal manera que cuando una de ellas en particular llegue al final del segmento, otra u otras estén en la mitad de ese segmento, de tal manera que todas en conjunto hacen un sonido continuo. Para lograrlo, requerimos una manera de generar dos o más ondas diente de sierra con la relación de fase deseada para utilizarlas en lugar del oscilador de la parte superior de la figura 2.7. Podemos comenzar con una sola onda diente de sierra y luego producir las otras con una relación de fase fija con la primera. Si queremos una diente de sierra que esté, digamos, *a* ciclos por delante de la primera, simplemente añadimos el parámetro *a* y luego tomamos la parte fraccional, que es la nueva diente de sierra que queríamos, tal como se muestra en la figura 2.8.

2.4 Estiramiento del timbre

El oscilador tabla de onda de la sección 2.1, que extendimos en la sección 2.2 para encerrar ondas de tablas de onda arbitrarias tales como sonidos grabados, puede adicionalmente extenderse de una manera complementaria, que denominaremos *estiramiento del timbre* por razones que desarrollaremos en esta sección. Hay muchas otras maneras de extender la síntesis de tablas de onda como por ejemplo la modulación de la frecuencia y la conformación de onda, pero las dejaremos para capítulos posteriores.

La idea central del estiramiento del timbre es reconsiderar la idea del oscilador tabla de onda como un mecanismo para ejecutar una tabla de onda (o parte de ésta) almacenada de extremo a extremo. No hay razón para que el final de un ciclo deba coincidir con el comienzo del siguiente. En lugar de esto podríamos buscar la manera de espaciar las copias de la onda insertando silencios alternativamente; o, yendo en la dirección opuesta, las copias de la onda podrían estar tan juntas que se traslapen. El único parámetro disponible en la sección 2.1 -la frecuencia- ha sido utilizado hasta aquí para controlar dos aspectos separados de la salida: el período al cual comenzamos las nuevas copias de la onda, y la longitud de cada copia individual. La idea del estiramiento del timbre es controlar las dos de manera independiente.

La figura 2.9 muestra el resultado de ejecutar una tabla de onda de tres maneras diferentes. En cada caso la onda de salida tiene un período de 20; en otras palabras, la frecuencia de salida es $R/20$, siendo R la velocidad de salida de las muestras <si R = 44100, la frecuencia de salida es 2205 Hz; así el período de 20 muestras se hace en 1/2205 segundos> En la parte (a) de la figura, cada copia de la onda es ejecutada en 20 muestras, de tal manera que la forma de la onda queda exactamente dentro del ciclo, sin espacios ni traslapes. En la parte (b) aunque el período sigue siendo 20, la onda se comprime en el medio a la mitad del período (10 muestras); o, en otras palabras, el *ciclo de trabajo* -la cantidad relativa de tiempo en el cual la onda llena el ciclo- es del 50 por ciento. Durante el 50 por ciento restante la salida es cero.

En la parte (c) la onda es estirada a 40 muestras, y ya que ésta se repite todavía cada 20 muestras, las ondas se traslapan dos a uno. El ciclo de trabajo es, de esta manera, del 200 por ciento.

Suponga ahora que el ciclo de tarea del 100 por ciento de la onda tiene una serie de Fourier (sección 1.7) igual a:

$$x_{100}[n] = a_0 + a_1\cos(\omega n + \phi_1) + a_2\cos(2\omega n + \phi_2) + \ldots$$

donde ω es la frecuencia angular (igual a $\pi/10$ en nuestro ejemplo ya que el período es 20 <es decir, se recorre la distancia de 2π en 20 muestras, así la frecuencia angular es $2\pi/20 = \pi/10$>.) Para simplificar este ejemplo no nos

preocuparemos de dónde debe terminar la serie, y la dejaremos ir al infinito.

Nos gustaría relacionar esta con las series de Fourier de las otras dos ondas del ejemplo, con el fin de mostrar cómo cambiando el ciclo de trabajo, cambia el timbre. Para el caso del ciclo con el 50 por ciento de ciclo de trabajo (denominamos la señal $x_{50}[n]$) observamos que la onda, si la replicamos con una fase equivalente a la mitad del período y adicionamos otra, nos da exactamente la onda original al doble de la frecuencia:

$$x_{100}[2n] = x_{50}[n] + x_{50}[n + \pi/\omega]$$

<como si a la onda que ocupa la mitad del período le pusiéramos otra en la otra mitad vacía del período; se obtiene una onda una octava más aguda -al doble de la frecuencia> donde ω es la frecuencia angular (y así π/ω es la mitad del período) de ambas señales. De esta manera, si escribimos la serie de Fourier de $x_{50}[n]$ como:

$$x_{50}[n] = b_0 + b_1\cos(\omega n + \theta_1) + b_2\cos(2\omega n + \theta_2) + \ldots$$

y sustituimos las series de Fourier para todos los tres términos de arriba, tenemos:

$$a_0 + a_1\cos(2\omega n + \phi_1) + a_2\cos(4\omega n + \phi_1) + \ldots$$

$$= b_0 + b_1\cos(\omega n + \theta_1) + b_2\cos(2\omega n + \theta_2) + \ldots$$
$$+ b_0 + b_1\cos(\omega n + \pi + \theta_1) + b_2\cos(2\omega n + 2\pi + \theta_2) + \ldots$$

$$= 2b_0 + 2b_2\cos(2\omega n + \theta_2) + 2b_4\cos(4\omega n + \theta_4) + \ldots$$

y así

$$a_0 = 2b_0,\ a_1 = 2b_2,\ a_2 = 2b_4$$

sucesivamente: en cuanto a los parciales pares de x_{50}, se obtienen estirando los de x_{100} el doble, al alejarse. <recordar que en la gráfica se compara la magnitud de a_1 con la de b_2, no con la de b_1> (No sabemos nada aún de los parciales impares de x_{50}, los cuales podrían estar en línea con los parciales pares o no, dependiendo de factores que no podemos controlar todavía. Basta con decir, por el momento, que si las ondas se conectan suavemente con el eje horizontal en ambos extremos, los parciales impares actuarán globalmente como los pares. Para hacer esto con mayor exactitud necesitaremos análisis de Fourier, el cual es desarrollado en el capítulo 9.)

De manera similar, x_{100} y x_{200} están relacionadas de la misma manera:

$$x_{200}[2n] = x_{100}[n] + x_{100}[n + \pi/\omega]$$

de tal manera que, si las amplitudes de las series de Fourier de x_{200} se denotan como c_0, c_1, ..., obtenemos:

$$c_0 = 2a_0,\ c_1 = 2a_2,\ c_2 = 2a_4,\ \ldots$$

de tal manera que los parciales de x_{200} son aquellos de x_{100} comprimidos a la mitad, y hacia la izquierda.

Vemos que comprimir la onda por un factor de 2 tiene el efecto de estirar la serie de Fourier por dos, y de otro lado, estirar la onda por un factor de dos comprime la serie de Fourier por dos. Siguiendo el mismo argumento, en general al estirar la onda por un factor cualquiera posible f comprime los sobretonos, en frecuencia, por el recíproco $1/f$ -por lo menos aproximadamente, y esta

aproximación es buena si la onda se “comporta bien” en sus extremos. (Como lo veremos más adelante, la onda puede forzarse para comportarse razonablemente bien envolviéndola como se muestra en la figura 2.7.)

La figura 2.10 muestra el espectro de las tres ondas -o en otras palabras la onda en sus tres ciclos de trabajo- de la figura 2.9. En la figura se enfatiza la relación entre los tres espectros, trazando curvas a través de cada espectro, las cuales, por inspección, tienden a convertirse en la misma curva, únicamente estiradas de manera diferente; cuando el ciclo de trabajo aumenta, la curva se comprime a la izquierda (todas la frecuencias caen) y se amplifican (se estira hacia arriba).

Las curvas continuas tienen una interpretación muy simple. Imagine comprimir la onda en un muy pequeño ciclo de trabajo, por decir del 1 por ciento. El contorno se estirará por un factor de 100. Yendo hacia atrás, esto nos permitiría interpolar entre cada par de puntos consecutivos del 100 por ciento del contorno del ciclo de trabajo (el original) 99 puntos nuevos. Como ya se ve en la figura el ciclo de tarea del 50 por ciento define la curva con el doble de resolución de la curva original. En el límite, si el ciclo de trabajo se hace muy pequeño, el espectro se llena más y más densamente; el límite es el espectro “real” de la onda.

Este espectro “real” se escucha únicamente a muy bajos ciclos de trabajo, sin embargo. En el ejemplo del 200 por ciento de ciclo de trabajo desaparece el pico en el espectro ideal (continuo) debido a que el pico cae por debajo del primer armónico. En general los ciclos de trabajo altos son una muestra ideal de la curva a más bajas resoluciones <Al estirar la onda a ciclos de trabajo cada vez más bajos -menos del 100%-, se escuchan armónicos cada vez más altos, estos se fortalecen y el espectro se amplía; pero al subir los ciclos de trabajo por encima del 100% los armónicos altos pierden fortaleza y los más bajos son los que se fortalecen>.

El estrechamiento del timbre es una técnica extremadamente poderosa para generar sonidos con un espectro variable. En combinación con las posibilidades de mezclar las ondas (sección 2.1) y de poder escoger entre un sin fin de ondas variables de sonidos grabados en muestras (sección 2.2), es posible generar todo tipo de sonidos. Por ejemplo, el diagrama de bloque de la figura 2.7 nos da una manera de grabar y estirar formas de onda de una tabla de onda grabada. Cuando el parámetro “frecuencia” f es suficientemente alto como para escucharse como un tono, el parámetro “tamaño” s puede tomarse para controlar el estiramiento del timbre, vía la fórmula $t = fs/R$ de la sección 2.2 <tomar este parámetro para controlar el estiramiento del timbre no cumple exactamente con el cometido inicial del estiramento, pues al dar a s un valor menor que el de N (número de muestras de la tabla) se recorta la tabla, y el cambio del espectro sonoro obedecerá tanto a dicho recorte como al acortamiento del ciclo de trabajo>, con lo que ahora reinterpretamos t como el factor por el cual se estira el timbre <se había presentado en esa sección 2.2 como “factor de transposicón”>.

2.5 Interpolación

Como se mencionó anteriormente, los esquemas de interpolación son utilizados normalmente para incrementar la exactitud de la lectura de tabla. Haremos aquí un cálculo algo simplificado de los efectos de los tamaños de la tabla de onda y de los esquemas de interpolación en el resultado de la lectura de tabla.

Para hablar del error en la lectura de tabla de onda, debemos ver la tabla de onda como una versión en muestras de una función subyacente. Cuando buscamos un valor de esa función subyacente que está entre los puntos de la tabla de onda, el error es la diferencia entre el resultado de la lectura de la tabla de onda y el valor “ideal” de la función en ese punto. Los estudios más relevantes del error en la lectura de tabla de onda asumen que la función subyacente es una sinusoide (ver página 1). Podemos entender entonces lo que sucede con otras

tablas de onda considerándolas como superposiciones (sumas) de sinusoides.

La seguridad de la lectura de una tabla de onda conteniendo una sinusoide depende de dos factores: de la calidad del esquema de interpolación, y del período de la sinusoide. En general mientras más largo es el período de la sinusoide, el resultado es más exacto.

En el caso de una tabla de onda sintetizada, podemos conocer sus componentes sinusoidales al especificarlos -en cuyo caso basta con escoger un tamaño de tabla de onda apropiado al momento de calcular la tabla de onda, para que se ajuste con el algoritmo de interpolación y encontrar así el nivel deseado de exactitud. En el caso de sonidos grabados, el análisis de exactitud podrían dirigirnos a ajustar la velocidad de las muestras de la grabación, ya sea ajustando la salida o a hacer un nuevo muestreo.

El error de interpolación para una tabla de onda sinusoidal puede tener dos componentes: el primero, que la señal continua (el resultado teórico de leer la tabla de onda continuamente en el tiempo, tal como si la velocidad de las muestras fuera infinita) podría no ser una sinusoide pura; y el segundo, que la amplitud podría ser errónea. (Es posible tener errores de fase también, pero únicamente por descuido.)

En el desarrollo únicamente consideraremos esquemas de interpolación polinomial tales como el redondeo, la interpolación lineal y la interpolación cúbica. Estos esquemas dan valores al evaluar polinomios (de grado cero, uno y tres respectivamente) en los intersticios entre los puntos de la tabla de onda. La idea es que para cualquier índice x, escojamos un punto de referencia cercano x_0, y hagamos que la salida se calcule por algún polinomio:

$$y_{INT}(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)^2 + \ldots + a_n(x - x_0)^n$$

Usualmente escogemos el polinomio que pasa a través de los puntos $n + 1$ más cercanos de la tabla de onda. Interpolación de 1 punto (polinomio de grado cero) significa dejar a_0 igual al punto más cercano de la tabla de onda. Para la interpolación de dos puntos, dibujamos un segmento de línea entre los dos puntos de la tabla de onda a cada lado del punto deseado x. Podemos hacer que x_0 sea el entero más cercano a la izquierda de x (lo que escribimos como $\lfloor x \rfloor$) y entonces la fórmula para la interpolación lineal es:

$$y_{INT}(x) = y[x_0] + (y[x_0 + 1] - y[x_0]).(x - x_0)$$

el cual es un polinomio, como en la fórmula previa, con

$$a_0 = y[x_0]$$
$$a_1 = y[x_0 + 1] - y[x_0]$$

En general usted puede ajustar un polinomio de grado n -1 a través de cualquier número de puntos n, en tanto sus valores x sean todos diferentes.

La figura 2.11 muestra el efecto de utilizar interpolación lineal (dos puntos) para rellenar una sinusoide de periodo 6. De los tres trazos que se dibujan, desde arriba hacia abajo, aparecen: la sinusoide original, el resultado de la interpolación lineal entre los 6 puntos del período utilizado para representar la sinusoide y, finalmente, otra sinusoide, ligeramente más pequeña en amplitud, la cual es la que mejor coincide con la onda de los seis segmentos. El error introducido al reemplazar la sinusoide original por la versión con interpolación lineal tiene dos componentes: el primero, un cambio en la amplitud (que es poco perceptible) y el segundo, una distorsión en la forma de la onda (la cual es muy perceptible).

La gráfica inferior en la figura muestra la diferencia entre la onda interpolada

y la sinusoide que mejor le encaja. Esta es una señal residual de todas aquellas energías que están en los sobretonos de la sinusoide original. Si el número de puntos se incrementa, el error disminuye en magnitud. Ya que el error es la diferencia entre una sinusoide y una secuencia de aproximación de segmentos, la magnitud del error es aproximadamente proporcional al cuadrado de la diferencia de fase entre cada par de puntos, o en otras palabras, inversamente proporcional al cuadrado del número de puntos de la tabla de onda. Puesto de otra manera, el error en la tabla de onda decrece 12 dB cada vez que la tabla dobla su tamaño. (Esta regla es buena únicamente para tablas con 4 puntos o más.)

La interpolación de cuatro puntos (cúbica) trabaja de manera similar. La fórmula para la interpolación es:

$$y_{INT}(x) = -f(f-1)(f-1)/6.y[x_0 - 1] + (f+1)(f-1)(f-2)/2.y[x_0] - (f+1)f(f-2)/2.y[x_0 + 1] + (f+1)f(f-1)/6.y[x_0 + 2]$$

donde $f = x - x_0$ es la parte fraccional del índice. Para tablas con 4 o más puntos, doblar el número de puntos de la tabla tiende a mejorar el error RMS en 24 dB. La tabla 2.1 muestra el error RMS calculado para sinusoides en varios períodos, para 1, 2 y 4 puntos de interpolación. (Una cantidad ligeramente diferente se mide en [Moo90, pág.164]. Aquí, los errores en amplitud y fase se pueden añadir llegando a resultados ligeramente más pesimistas. Ver también [Har87].)

El dominio de entrada disponible para la lectura de la tabla depende del número de puntos de interpolación. En general, cuando usamos k-puntos de interpolación en una tabla con N puntos, el rango está sobre un intervalo de $N + 1 - k$ puntos. Si $k = 1$ (es decir, sin ningún tipo de interpolación), el dominio es de 0 a N (incluyendo el punto final en 0 pero excluyendo el que queda en N) asumiendo que los valores de entrada están truncados (tal como se hace para leer tabla no interpolada en Pd). El dominio es de -1/2 a N - 1/2 si, en cambio, redondeamos la entrada al entero más cercano en lugar de interpolar. En cualquier caso el dominio se estira sobre una longitud de N puntos.

Para la interpolación de dos puntos, la entrada debe quedar entre el primer y el último puntos, esto es, entre 0 y N - 1. De esta manera N puntos son suficientes para definir la función sobre un dominio de longitud $N - 1$. Para interpolación de cuatro puntos, no podemos tener valores de entrada entre 0 y 1 (no podemos tener los dos puntos requeridos a la izquierda de la entrada) y tampoco podemos tenerlos para el espacio entre los dos últimos puntos ($N - 2$ y $N - 1$). Así en este caso el dominio va de 1 a $N - 2$ y tiene una longitud $N - 3$.

Las ondas periódicas almacenadas en tablas de onda requieren tratamiento especial en los extremos de la tabla. Por ejemplo, suponga que queremos almacenar una sinusoide pura de longitud N. Para leer una tabla sin interpolación, es suficiente ajustar, por ejemplo,

$$x[n] = \cos(2\pi n/N),\ n = 0, \ldots, N - 1$$

Para interpolación de dos puntos, necesitamos $N + 1$ puntos:

$$x[n] = \cos(2\pi n/N),\ n = 0, \ldots, N$$

En otras palabras debemos repetir el primer punto ($n = 0$) al final, de tal manera que el último segmento de N - 1 a N alcance a regresar al valor inicial.

Para la interpolación de cuatro puntos, el ciclo debe ajustarse para iniciar en el punto $n = 1$, ya que no podemos obtener valores de interpolación apropiados para valores de entrada menores que uno. Si, entonces, un ciclo de la tabla de onda se arregla de 1 a N, debemos suplir los puntos extra para 0 (copiado de

N), y también N + 1 y N + 2 copiados de 1 y 2, para hacer una tabla de longitud N + 3. Para la misma sinusoide de arriba, la tabla deberá contener:

$$x[n] = \cos(2\pi(n - 1)/N),\ n = 0, \ldots, N + 2$$

2.6 Ejemplos

Oscilador tabla de onda

El ejemplo B01.wavetables.pd, mostrado en la figura 2.12, implementa un oscilador tabla de onda, el cual reproduce desde una tabla de onda denominada "table10". Dos nuevos objetos Pd se muestran aquí. El primero es la tabla de onda misma, que aparece a la derecha en la figura. Usted puede arrastrar el ratón sobre la tabla de onda para cambiar su forma y escuchar el cambio en el sonido resultante. No se muestra en la figura pero se demuestra en el parche Pd la facilidad para calcular tablas de onda automáticamente con amplitudes específicas de los parciales, lo cual es a veces preferible a dibujar ondas a mano. Usted también puede leer y escribir tablas (texto o sonido) en archivos para intercambiar datos con otros programas. La otra novedad es un objeto de la clase:

tabosc4~: un oscilador tabla de onda. El "4" indica que esta clase utiliza interpolación de 4 puntos (cúbica). En el ejemplo, el nombre de la tabla, "table10", está especificado como un argumento para la creación del objeto tabosc4~. (Usted también puede cambiar dinámicamente de tablas de onda enviando el mensaje apropiado al objeto.)

Las tablas de onda utilizadas por tabosc4~ deben tener siempre un período igual a una potencia de dos; pero como se mostró antes, la tabla de onda debe tener tres puntos extras envolviendo los extremos. Las longitudes de tabla disponibles son así de la forma 2^m + 3, tales como 131, 259, 515, etc.

Los osciladores tabla de onda no están limitados a utilizarse como osciladores de audio. El parche B02.wavetable.FM.pd (que no se muestra aquí) utiliza un par de osciladores de tabla de onda en serie. La salida del primero es utilizado como la entrada del segundo, y de esta manera controla su frecuencia, la cual cambia periódicamente en el tiempo.

Lectura de tabla de onda en general

El objeto tabosc4~, aunque es práctico y eficiente, es algo especializado y para muchas de las aplicaciones descritas en este capítulo necesitaremos algo más general. El ejemplo B03.tabread4.pd (figura 2.3) demuestra la técnica de estiramiento de timbre discutida en la sección 2.4. Este es un ejemlo simple de una situación donde tabosc4~ no sería suficiente. Hay nuevos objetos aquí:

tabread4~: lectura de tabla de onda. Al igual que con tabosc4~ la tabla se lee utilizando interpolación de 4 puntos. Pero aunque tabosc4~ toma una frecuencia como entrada y automáticamente lee la forma de onda en patrón repetido, tabread4~ es más sencillo y espera el índice de lectura de tabla de onda como una entrada. Si se quiere utilizarlo para hacer algo repetitivo, como en este ejemplo, la entrada misma debe ser una onda repetida. Al igual que tabosc4~ (y todos los demás objetos de lectura y escritura de tablas), usted puede enviar mensajes para seleccionar cuál tabla utilizar.

tabwrite~: graba una señal de audio en una tabla de onda. En este ejemplo tabwrite~ se utiliza para mostrar la salida (aunque más tarde lo utilizaremos para otro tipo de cosas.) Cada vez que recibe un mensaje "bang" del ícono botón sobre éste, tabwrite~ comienza a escribir muestras sucesivas de su entrada en la tabla nombrada.

El ejemplo B03.tabread4.pd muestra cómo combinar un objeto phasor~ y uno

tabread4~ para hacer un oscilador tabla de onda. La salida de phasor~ tiene un rango de valores de 0 a 1. En este caso la tabla de onda de entrada, denominada "waveform12", es de 131 elementos de logitud. El dominio para el objeto tabread4~ es así de 1 a 129. Para ajustar el rango de phasor~ según esto, lo multiplicamos por la longitud del dominio (128) de tal manera que vaya de 0 a 128, y luego adicione 1, deslizando el intervalo de manera efectiva a la derecha, en un punto. Este reescalado se hace con los objetos *~ y +~ entre phasor~ y tabread4~.

Con solamente estas cuatro cajas habríamos reinventado en escencia el objeto tabosc4~. En este ejemplo, sin embargo, la multiplicación no es por una constante 128 si no por una cantidad variable controlada por el parámetro "squeeze". La función de las cuatro cajas a la derecha del parche es suministrar al objeto *~ los valores para escalar a phasor~. Para esto se hace uso de una nueva clase de objeto:

pack: se compone de una lista de dos o más elementos. La creación de los argumentos establece el número de estos, sus tipos (usualmente números) y sus valores iniciales. Las entradas (serán según la cantidad que se especifique en la creación de argumentos) actualizan los valores de los mensajes de los argumentos, y, si se cambia la entrada de la izquierda (o si simplemente se dispara con un mensaje "bang"), el mensaje va a la salida.

En este parche los argumentos son inicialmente 0 y 50, pero la caja de número actualizará el valor del primer argumento, de tal manera que, según se muestra, el mensaje más reciente que salió del objeto pack fue "206 50". El efecto de esto sobre el objeto line~ que está debajo es una pendiente al valor 206 en 50 milisegundas; en general la salida del objeto line~ es una señal de audio que de manera suave sigue los cambios esporádicos en los valores de la caja de número etiquetada "squeeze".

Finalmente 128 es adicionado al valor "squeeze"; si "squeeze" toma valores no negativos (tal como lo fuerza la caja de número en este parche), el multiplicador de ajuste de rango hace que phasor~ se multiplique por 128 o más. Si este valor es mayor que 128 el efecto es que la función "phasor" re-escalada gaste alguna fracción de su ciclo atorada al final de la tabla de onda (lo cual recorta su entrada a 129). El resultado es que la onda es leída sobre alguna fracción del ciclo. Como se muestra, la onda es comprimida a 128/(128+206) de un ciclo, de tal forma que el espectro se estira por un factor de casi 1/2 <de esta manera se logra la escencia del estiramiento de la tabla de onda, y realmente no se modifica directamente el parámetro *s* (y por lo tanto la onda misma no se recorta) según se proponía en el diagrama de bloques, si no que, dejando el tamaño de la muestra fijo (128 muestras) se modifica la longitud de la diente de sierra desde un mínimo dado por ese tamaño y de allí en adelante, según el número "squeeze" que se le sume, lo que da como resultado los ciclos de trabajo menores que 1>.

Para simplificar, este parche es sutilmente diferente del ejemplo de la sección 2.4 en el que las ondas son comprimidas hacia el comienzo de cada ciclo y no hacia la mitad. Esto tiene el efecto de cambiar ligeramente la fase de varios parciales de la onda cuando se estira y cuando se comprime; si el factor de compresión cambia rápidamente, la deriva correspondiente de la fase sonará como una ligera variación en el tono. Esto se puede evitar utilizando un arreglo ligeramente más complicado: sustraer 1/2 de phasor~, multiplicarlo por 128 o más y luego añadir 65 en lugar de uno.

Utilizando una tabla de onda como un muestreador

El ejemplo B04.sampler.pd (figura 2.14) muestra cómo utilizar una tabla de onda como un muestreador. En este ejemplo el índice dentro de la muestra (la tabla de onda) es controlado al accionar el ratón sobre la caja de número que se encuentra en la parte superior. Una escala conveniente para la caja de número es

de centésimas de segundo; para convertirlo a muestras (tal como lo requiere la entrada de tabread4~) multiplicamos por 44100 muestras/s por 0.01 s para obtener 441 muestras por unidad, antes de aplicar pack y line~ de la misma manera que lo utilizamos en ejemplo previo. La transposición que usted escucha depende de qué tan rápido mueva el ratón arriba y abajo. Este ejemplo ha introducido un nuevo tipo de objeto:

hip~: un filtro simple pasa-altos (corta-graves). El argumento proporciona la frecuencia que se elimina en ciclos por segundo. Utilizamos esto aquí para eliminar la salida constante (frecuencia cero) cuando la entrada se situa en una sola muestra (cada que usted no esté cambiando de manera activa el sitio de lectura con el ratón). Los filtros se discuten en el capítulo 8.

Los objetos pack y line~ en este ejemplo no se incluyen únicamente para hacer el sonido más continuo, si no que son esenciales para hacer el sonido entendible. Si el índice de lectura de la tabla de onda cambia simplemente cada que el ratón se mueve un pixel (por decir, veinte a quince veces por segundo) la gran mayoría de las muestras tendrán el mismo índice que la muestra anterior (las otras 44100+ muestras, sin contar aquellas donde el ratón se ha movido). De esta manera la velocidad de precesión será casi siempre cero. En lugar de cambiar las transposiciones, usted escuchará arena de 20 a 50 ciclos por segundo. (Trate de encontrar cómo suena esto!)

Muestreadores de lazo

En la mayoría de las situaciones, usted querrá una forma más automática para especificar los sitios de lectura de la tabla de onda, que hacerlo con el movimiento del ratón; por ejemplo, usted podría querer ser capaz de ejecutar una muestra en una transposición fija; usted podría tener varias muestras reproduciéndose a la vez (u otras cosas que requieran de atención), o usted podría querer cambiar rápidamente entre muestras o ir a sitios pre-fijados de la tabla de onda. En los pocos ejemplo siguientes desarrollaremos un lector de muestras enlazadas automático, el cual, aun cuando es una de las múltiples posibilidades, es una poderosa y muy utilizada.

Los parches B05.sampler.loop.pd y B06.sampler.loop.smooth.pd muestran cómo hacer esto: el primero de la manera más sencilla posible y el segundo (que se muestra en la figura 2.15, parte a) incorporando una segunda onda para envolver el sonido según se describe en la sección 2.3. Un nuevo tipo de objeto se introduce aquí:

cos~: calcula el coseno de 2π veces la entrada de la señal (de tal manera que de 0 a 1 hace un ciclo completo). A diferencia de la tabla que lee objetos en Pd, cos~ maneja la envolvente de tal manera que no hay limitación de rango en su entrada.

En la figura 2.15, (parte a), un objeto phasor~ proporciona los índices de la tabla de onda (a la derecha) y la fase para la función de envolvente de mitad de coseno a la izquierda. Estas dos se multiplican y el producto es filtrado por un filtro pasa-altos, y enviado a la salida de audio. La lectura de la tabla de onda es directa; el fasor es multiplicado por un parámetro “chunk size” <”tamaño de segmento”>, adicionado a 1, y utilizado como un índice para tabread4~. El parámetro tamaño de segmento se multiplica por 441 para convertirlo de centésimos de segundo a muestras. Esto corresponde exactamente al diagrama de bloque que se muestra en la figura 2.5, con una localización de segmento de 1. (la localización del segmento no puede ser 0 debido a que 1 es el índice mínimo para el cual trabaja tabread4~.)

El camino de la señal a la izquierda en el ejemplo corresponde a la técnica de lectura de tabla de onda envolvente mostrado enla figura 2.7. Aquí la onda diente de sierra está ajustada al rango (-1/4, 1/4) (al sustraer y multiplicar por 0.5), y luego lo envía a cos~. Este lee la función coseno en el rango (-π/2,

π/2), dándole de esta manera la mitad positiva de la onda.

La parte (b) de la figura 2.15 introduce un tercer parámetro, el “read point” <“punto de lectura”> el cual especifica en qué punto de la muestra debe comenzar el lazo. (En la parte (a) siempre se comenzaba al inicio). El cambio necesario es simple: adicionar el valor de control para el “punto de lectura”, en muestras, para el índice de la tabla de ondas y proceder como antes. Para evitar discontinuidades en el índice suavizamos el valor del punto de lectura utilizando los objetos pack y line~ tal como lo hicimos en el primer ejemplo del muestreador (figura 2.14).

Esto pone de relieve un importante aunque sutil asunto. La Fórmula de la Transposición Momentánea (página “33”) predice que mientras el tamaño del segmento y el punto de lectura no estén cambiando en el tiempo, la transposición es la frecuencia por el tamaño del segmento (como siempre, utilizando unidades apropiadas, Hertz y segundos, por ejemplo, de tal manera que el producto es sin dimensiones). Sin embargo, al variar el tamaño del segmento y el punto de lectura en el tiempo se afectará la transposición momentánea, usualmente de maneras muy notables, tal como se puede escuchar en el ejemplo B07.sampler.scratch.pd. El ejemplo B08.sampler.nodoppler.pd (el que se muestra en la figura) muestra una forma posible de controlar este efecto, introduciendo un nuevo tipo de objeto:

samphold~: una unidad de muestra y retención. (Esta será familiar para los usuarios de los sintetizadores análogos, pero con un giro digital; para más detalles ver la sección 3.7.) Esta almacena una sola muestra de la entrada de la mano izquierda y la envía a la salida de manera repetida hasta que actue la entrada de la derecha (que es también una señal de audio digital llamada el *gatillo*) para sobre-escribir la muestra almacenada con una nueva -de nuevo desde la entrada izquierda. La unidad adquiere una nueva muestra siempre y cuando el valor numérico del gatillo caiga desde una muestra a la siguiente. Está diseñada para que funcione bien con los objetos phasor~, para facilitar los disparos en fase con las envolventes.

El ejemplo B08.sampler.nodoppler.pd utiliza dos objetos samphold~ para actualizar los valores de tamaño de segmento y de punto de lectura, exactamente cuando el phasor~ realice la envolvente, momento en el cual la envolvente coseno es cero y así el efecto del cambio instantáneo no pueda escucharse. En esta situación podemos utilizar la Fórmula de Transposición para Tablas de Onda en Lazo, que es más sencilla, para relacionar la frecuencia, el tamaño del segmento y la transposición. Esto se demuestra en el ejemplo B09.sampler.transpose.pd (que no se muestra).

Muestra enlazada traslapada

Tal como se describió en la sección 2.3, a veces es deseable utilizar dos o más muestras enlazadas traslapadas para producir un sonido razonablemente continuo sin tener que envolver de manera muy aguda en los extremos del lazo. Esto es especialmente útil en situaciones donde el segmento que está enlazado es corto, una décima de segundo o menos. El ejemplo B10.sampler.overlap.pd, que se muestra en la figura 2.16 (parte a), realiza dos lazos de muestras con medio ciclo de diferencia de fase entre ellas. Los nuevos tipos de objetos son:

loadbang: proporciona una salida con el mensaje “bang” al cargar el parche. Es utilizado en este parche para asegurar que la división de la transposición por el tamaño del segmento tiene un factor de transposición válido en caso de que “tamaño del segmento” se mueva con el ratón en primer lugar.

expr: evalúa expresiones matemáticas. Las variables aparecen como $f1, $f2, y así se corresponden con las entradas de los objetos. Las operaciones matemáticas se permiten, con paréntesis para agrupar, y se suministran muchas librerías de funciones tales como la potenciación, la cual se muestra en este ejemplo como

“pow” (función de potenciación).

wrap~: envuelve el intervalo de 0 a 1. Así, por ejemplo, 1.2 lo convierte en 0.2; 0.5 queda tal cual; y -0.6 lo convierte en 0.4.

send~, s~, receive~, r~: versiones de señal de audio de send y receive. Una señal de audio enviada a un objeto send~ aparece en la salida de cualquiera y de todos los objetos receive~ con el mismo nombre. A diferencia de send y receive, usted no puede tener más de un objeto send~ con el mismo nombre (para ese tipo de conexión ver los objetos throw~ y catch~).

En el ejemplo, parte de la maquinaria de lectura de la tabla de onda se duplica, utilizando cálculos idénticos para “chunk-size-samples” (la secuencia de mensajes) y para “read-pt” (una señal de audio suavizada como antes). Sin embargo, el elemento “phase” de la señal de audio, en la otra copia, es reemplazado por “phase2”. La parte superior de la figura muestra el cálculo de las dos fases de la señal: la primera como la salida del objeto phasor~, y la segunda adicionando 0.5 y envolviéndola, adicionándole así 0.5 ciclos (π radianes) a la fase. Las dos fases de las señales son utilizadas cada una, con los mismos ajustes de rango según se hicieron, para calcular el índice en la tabla de onda y del objeto cos~, y para controlar también los dos objetos samphold~. Finalmente, los resultados de las dos copias se adicionan para enviarlas a la salida de audio.

Precesión automática del punto de lectura

El ejemplo B11.sampler.rockafella.pd, que se muestra en la parte (b) de la figura 2.16, adapta las ideas que se mostraron antes para una situación donde el punto de lectura es computado automáticamente. Aquí hacemos la precesión del punto de lectura a través de la muestra en un lazo, permitiéndonos acelerar y desacelerar la reproducción, independientemente de la transposición.

Este ejemplo muestra una debilidad del procedimiento, que consiste en que si la velocidad de precesión relativa es cercana a uno (que es el caso de la velocidad normal de escucha de la tabla de onda grabada), y si tampoco hay mucha transposición, es preferible utilizar granos más grandes y frecuencias más bajas de repetición en concordancia (manteniendo el producto constante para alcanzar la transposición deseada.) Sin embargo, si el tamaño del grano queda demasiado grande, ya no es conveniente cuantizar los cambios de control a la fase de la envolvente, pues estos podrían estar demasiado aparte como para permitir un tiempo de respuesta razonable para los cambios de control.

En este parche quitamos el objeto samphold~ que había controlado el punto de lectura (pero dejamos el del tamaño de segmento el cual es mucho más difícil de cambiar en medio del lazo). En su lugar utilizamos el porcentaje de precesión (conocido) del punto de lectura para corregir la frecuencia de la diente de sierra, de tal manera que mantenemos la transposición deseada. Esto en cambio hace que cuando el factor de transposición y la precesión están muy cercanos el uno de la otra (haciendo entonces algo muy cercano a un simple cambio de velocidad), la frecuencia caerá a un valor cercano a cero, y así habremos incrementado la naturalidad del resultado al mismo tiempo.

En este parche pasamos de manejar puntos de lectura, tamaños de segmento, etc., en número de muestras para utilizar segundos, haciendo la conversión a número de muestras (y cambiando por uno) justo antes del objeto tabread4~. La tabla de onda contiene un sonido de un segundo de duración, y asumiremos que el tamaño nominal del segmento no será mayor que 0.1 segundos, de tal manera que podemos dejar con seguridad que el rango del punto de lectura vaya de 0 a 0.9; el tamaño “real” del segmento variará, y puede llegar a ser bastante largo, debido al movimiento del punto de lectura.

El control de la precesión ajusta la frecuencia de un fasor de amplitud 0.9, y

de esta manera la precesión debe multiplicarse por 0.9 para ajustar la frecuencia del fasor (así, para obtener una precesión de uno la amplitud y la frecuencia del punto de lectura son ambas 0.9, de tal manera que la pendiente, al ser iguales la amplitud sobre la frecuencia, es uno). La salida es denominada "read-pt" como antes, y es utilizada para ambas copias del lector de la tabla de onda.

La precesión p y el tamaño del segmento c se conocen, y si denotamos la frecuencia del fasor de la parte de arriba (la original) por f, el factor de transposición queda dado por:

$$t = p + cf$$

y resolviendo para f da:

$$f = (t - p)/c = (2^{h/12} - p)/c$$

donde h es la transposición deseada en semitonos. Esta es la fórmula utilizada en el objeto `expr`.

Ejercicios

1. Si una tabla de onda con 1000 muestras se reproduce con transposición de una unidad, a una velocidad de lectura de muestras de 44100 Hertz, qué tanto dura el sonido?

2. Una tabla de onda de un segundo de duración es ejecutada en 0.5 segundos. A qué intervalo se transpone el sonido?

3. Asumiendo todavía la tabla de onda de un segundo, si la reproducimos periódicamente (en un lazo), a cuántos Hertz deberíamos hacer el lazo para transponer el sonido original en medio tono?

4. Queremos reproducir una tabla de onda (grabada a R = 44100) en un lazo a 10 veces en cada segundo, de tal manera que el sonido original almacenado en la tabla de onda sea transpuesto una quinta perfecta (ver página "12"). Qué longitud de segmento, en número de muestras debería ser el que se reproduce?

5. Suponga que desea utilizar el estiramiento de onda para una tabla de onda que contiene una onda periódica de período 100. Usted desea escuchar el espectro no transpuesto a un período de 200 muestras. Por qué factor de tarea debería usted comprimir la onda?

6. La primera mitad de una tabla de onda contiene un ciclo de una sinusoide de amplitud pico uno. La segunda mitad contiene ceros. Cuál es la fortaleza del segundo parcial de la tabla de onda.

7. Una sinusoide se almacena en una tabla de onda con período 4 de tal manera que los primeros cuatro elementos son 0, 1, 0, -1, correspondiendo a los índices 0, 1, 2 y 3. Qué valor obtenemos con la entrada 1.5: (a) utilizando interpolación de dos puntos? (b) utilizando interpolación de 4 puntos? (c) cuál es el valor de la sinusoide original aquí?

8. Si todo el contenido de una tabla de onda cae entre los valores -1 y 1, cuál es el rango de salidas posibles de la lectura de tabla de onda utilizando interpolación de 4 puntos?