Guía para la “Teoría y la Técnica de la Música Electrónica”

Basada en el borrador de Miller Puckette escrito en diciembre de 2006

Capítulo 1
Sinusoides, amplitud y frecuencia

La música electrónica se hace casi siempre a través de un computador sintetizando o procesando *señales de audio digital*. Estas son secuencias de números:

$$..., x[n-1], x[n], x[n+1], ...$$

donde el índice n es llamado el *número de la muestra*, cuyo rango puede estar entre algunos o todos los números enteros. Un solo número de la secuencia es denominado *muestra*. Un ejemplo de una señal de audio digital es la *Sinusoide*:

$$x[n] = a\cos(\omega n + \phi)$$

donde a es la *amplitud*, ω es la *frecuencia angular* y ϕ es la *fase* inicial. La fase es función del número de muestra n, igual a $(\omega n + \phi)$. La fase inicial es la fase en la muestra cero ($n = 0$).

Las sinusoides juegan un papel clave en el procesamiento de audio, pues si se cambia una de ellas a la izquierda o a la derecha por cualquier número de muestras, se consigue otra sinusoide. Esto hace fácil calcular el efecto de todo tipo de operaciones con sinusoides. Nuestros oídos utilizan esta misma propiedad especial para ayudarnos a analizar los sonidos que escuchamos, lo cual explica porqué la sinusoides, y la combinación de sinusoides pueden ser utilizadas para alcanzar muchos efectos musicales.

Las señales de audio digital no tienen ninguna relación intrínseca con el tiempo, pero para escucharlas debemos escoger una *velocidad de muestras*, la cual se designa usualmente con la variable R, designando el número de muestras que hay en un segundo. El tiempo t está relacionado con la muestra n por $Rt = n$, o $t = n/R$. Una señal sinusoidal con frecuencia angular ω tiene una frecuencia en el tiempo real dada por

$$f = (\omega R)/(2\pi)$$

en Hertz (ciclos por segundo), ya que un ciclo son 2π radianes y un segundo equivale a R muestras

La amplitud de una señal en el mundo real puede ser expresada como una variación en el tiempo del voltaje o de la presión de aire, pero las muestras de una señal de audio digital son números sin unidades. Casualmente asumiremos aquí que hay una seguridad numérica amplia de tal manera que podemos ignorar el redondeo de errores, y que el formato numérico tiene un rango ilimitado, de tal manera que las muestras pueden tomar el valor que queramos. Sin embargo la mayoría de los equipos que manejan audio digital trabajan únicamente en un rango de valores fijo para la entrada y para la salida, la mayoría usualmente entre -1 y 1. Los procesadores modernos de programas de computación de audio digital usualmente utilizan una representación de punto flotante para las señales. Esto nos permite usar las unidades que sean más convenientes para cada tarea, en tanto la salida final de audio se mantenga dentro del rango que maneja el equipo [Mat69, págs.4-10].

1.1. Medidas de la amplitud.

La propiedad más importante de una señal de audio digital es su amplitud. Desafortunadamente la amplitud de una señal de audio digital no tiene una definición canónica. Estrictamente hablando, todas las muestras en una señal de audio digital son ellas mismas amplitudes, además de hablar de la amplitud a de una sinusoide como un todo. Es usual tener medidas de amplitud para las señales de audio digital en general. La amplitud es mejor considerarla aplicada a una *ventana*, a un rango fijo de muestras de una señal. Por ejemplo la ventana que comienza con la muestra M y tiene una longitud N en una señal de audio $x[n]$ consiste en las muestras,

$$x[M],x[M + 1],....,x[M + N - 1]$$

Las dos medidas de amplitud utilizadas con más frecuencia son la *amplitud pico*, la cual es simplemente la muestra más grande (en valor absoluto) en la ventana:

$$A_{\text{pico}}\{x[n]\} = |x[n]|, \quad n = M,...,M + N - 1$$

y la amplitud *media cuadrática* (RMS).

$$A_{\text{RMS}}\{x[n]\} = \sqrt{P\{x[n]\}}$$

donde $P\{x[n]\}$ es la potencia media, definida como:

$$P\{x[n]\} = \left(|x[M]|^2 + ... + |x[M + N - 1]|^2\right)$$

(En esta última fórmula los signos de valor absoluto no son necesario por el momento pues estamos trabajando con señales de valor real, pero más tarde serán importantes cuando consideremos las señales de valor comlejo). Ni la amplitud pico ni la amplitud RMS pueden ser cero, y cualquiera de las dos pueden ser cero si la señal misma es cero para cualquier valor de n de la ventana.

La amplitud RMS puede ser igual a la amplitud pico, pero nunca podrá excederla; así mismo, la amplitud RMS puede ser tan pequeña como $1/\sqrt{N}$ veces la amplitud pico, pero nunca menor que este valor.

Bajo condiciones razonables -si la ventana contiene por lo menos algunos periodos y si la frecuencia angular está bien por debajo de un radián por muestra <un valor por debajo de un radián por muestra es una fracción de radián por muestra; bien por debajo, una fracción muy pequeña>- la amplitud pico de la sinusoide de la página “1” es aproximadamente a y su amplitud RMS es cercana a $a/\sqrt{2}$. La figura 1.2 muestra las amplitudes pico y RMS de dos señales de audio digital.

1.2. Unidades de amplitud

Dos amplitudes usualmente se comparan mejor utilizando la razón entre ellas que utilizando su diferencia. Decir que la amplitud de una señal es mayor que otra por un factor de dos podría ser mejor información que decir que es mayor por 30 milivoltios. Esto es válido para cualquier tipo de amplitud (la amplitud pico y la amplitud RMS). Para facilitar la comparación, usualmente expresamos las amplitudes en unidades logarítmicas llamadas *decibeles*. Si a es la amplitud de una señal (pico o RMS), entonces podemos definir el nivel d en decibeles (dB) como:

$$d = 20*\log_{10}(a/a_0)$$

donde a_0 es la amplitud de referencia. Esta definición está ajustada de tal manera que, si incrementamos la potencia de la señal por un factor de diez (y así la amplitud se incrementa por un factor de $\sqrt{10}$), el logaritmo se incrementará en 1/2, y el valor en decibeles irá hacia arriba (aditivamente) en 10.

Un incremento en la amplitud por un factor de dos corresponde a un incremento de 6.02 decibeles aproximadamente; doblar la potencia corresponde a un incremento de 3.01 dB. La relación entre la amplitud lineal y la amplitud en decibeles está graficada en la figura 1.3.

Utilizando a_0 para denotar la amplitud de referencia, una señal con amplitud lineal más pequeña que a_0 tendrá una amplitud negativa en decibeles: $a_0/10$ da -20 dB, $a_0/100$ da -40, etc. Una amplitud lineal de cero es más pequeña que cualquier valor en dB, así que da un nivel en dB de $-\infty$.

En audio digital una opción conveniente para tomar como referencia, asumiendo que el equipo tenga una amplitud máxima de uno, es:

$$a_0 = 10^{-5} = 0.00001$$

con lo que la amplitud máxima son 100 dB, y 0 dB es el silencio a un nivel de audición razonable. De manera conveniente, el rango dinámico de audición humana -la razón entre un volumen dañino y un silencio inaudible- es de 100 dB aproximadamente.

La amplitud está relacionada de una manera inexacta con la intensidad de un sonido. En general dos señales con la misma amplitud pico o RMS, no necesariamente las escucharemos con igual intensidad. Al amplificar una señal en 3 dB, por ejemplo, apenas sí se podría decir que el sonido se siente un poco más intenso.

Se ha estudiado mucho la supuesta naturaleza logarítmica del oído humano (y de los otros sentidos), lo cual explica parcialmente porqué los decibeles son tan útiles en la escala de amplitud [RMW02, pág.99].

La amplitud también está relacionada de una manera inexacta con las *dinámicas* musicales. La dinámica es mejor pensarla como una medida del esfuerzo que como intensidad o potencia. Su rango está sobre nueve valores: silencio, ppp, pp, p, mp, mf, f, ff, fff. Estos están correlacionados de una manera más lejana con la amplitud de una señal que con su intensidad [RMW02, págs.110-111].

1.3. Controlando la amplitud

Sin embargo la operación más frecuente en los sonidos electrónicos es cambiar sus amplitudes. Por ejemplo, una estrategia simple para la síntesis de sonidos es combinando sinusoides, lo cual puede ser generado evaluando la fórmula del primer capítulo

$$x[n] = a\cos(\omega n + \phi)$$

muestra por muestra. Pero la sinusoide tiene una amplitud nominal constante a, y nos gustaría ser capaces de variarla en el tiempo.

En general, al multiplicar la amplitud de una señal $x[n]$ por un factor $y \geq 0$, usted puede simplemente multiplicar cada muestra por y, dando como resultado una nueva señal $y*x[n]$. Cualquier medida de amplitud pico o RMS de $x[n]$ será mayor o menor según el factor y. De una manera más general, usted puede cambiar la amplitud en una cantidad $y[n]$ que varíe muestra por muestra. Si $y[n]$ no es negativa y si varía con cierta lentitud, la amplitud del producto $y[n]*x[n]$ (en una ventana fija desde M hasta $M + N - 1$) será el valor de $x[n]$, multiplicado

por el valor de $y[n]$ en la ventana (el cual asumimos no cambia mucho en las N muestras de la ventana).

En el caso más general donde tanto $x[n]$ como $y[n]$ tienen valores negativos y positivos y/o cambian rápidamente, el efecto de su multiplicación no puede ser descrito como el simple cambio de amplitud de uno de ellos; esta situación es considerada posteriormente, en el capítulo 5.

1.4 Frecuencia

Las frecuencias, al igual que las amplitudes, son usualmente medidas sobre la escala logarítmica, con el fin de enfatizar las proporciones entre ellas, lo que usualmente provee una mejor descripción de la relación entre las frecuencias, que la que hace la diferencia entre ellas. La razón entre las frecuencias de dos notas musicales determina el intervalo musical entre ellas.

La escala musical occidental divide la *octava* (el intervalo musical asociado con una relación de 2:1) en doce sub-intervalos iguales, cada uno de los cuales de esta manera corresponde a una razón de $2^{1/12}$. Por razones históricas este intervalo es denominado un *semitono*. Una escala logarítmica conveniente para las notas musicales es simplemente contar el número de semitonos desde una nota de referencia -permitiendo fracciones que posibilitan especificar notas que no corresponden con las notas musicales de la escala occidental. La escala de notas logarítmica más utilizada es la de “notas MIDI”, en la cual la nota 69 está asignada a una frecuencia de 440 ciclos por segundo -el A sobre el C medio. Para hacer la conversión entre una nota MIDI m y una frecuencia en ciclos por segundo f, se aplican las fórmulas de conversión Nota/Frecuencia:

$$m = 69 + 12*\log_2(f/440)$$

$$f = 440*2^{(m-69)/12}$$

El C medio, correspondiente a la nota MIDI m = 60, sería f = 261,626 ciclos por segundo.

El MIDI en sí es un viejo protocolo de equipos de cómputo el cual desafortunadamente ha insinuado un gran convenio de diseño de programas de computación. En los equipos de cómputo, el MIDI permite únicamente notas enteras entre 0 y 127. Sin embargo, la escala en general está bien definida para cualquier número “MIDI”, incluso los números negativos; por ejemplo una “nota MIDI” de -4 es una cantidad de vibración presentable. La escala de notas no puede, sin embargo, describir frecuencias iguales o menores que cero ciclos por segundo. (Para una descripción clara del MIDI, sus posibilidades y limitaciones, vea [Bal03, cap.6-8]).

Un semitono conforma una razón de aproximadamente 1,059 a 1, un seis por ciento aproximadamente de incremento en la frecuencia. Los semitonos están además divididos en *cents*, siendo cada cent un centésimo de un semitono. En la práctica, se considera que con tres cents se hace una diferencia perceptible mínima en el cambio de afinación de una nota musical. En el C medio esto viene a equivaler a 1/2 ciclo por segundo aproximadamente.

1.5. Sintetizando una sinusoide

En la mayoría de los paquetes programación de síntesis audio más ampliamente usados (Csound, Max/MSP, y Pd, por ejemplo), las operaciones de audio están especificadas como redes de *unidades generadoras* [Mat69], las cuales pasan señales de audio entre ellas mismas. El usuario del paquete de programas especifica la red, a veces llamada *un parche*, la cual esencialmente corresponde al algoritmo de síntesis que se utilizará, y entonces su preocupación se enfoca en cómo controlar las diferentes unidades generadoras en el tiempo. En esta

sección utilizaremos diagramas de bloque abstractos para describir los parches, pero en la sección de “ejemplos” (página “17”), escogeremos un ambiente de implementación específico y mostraremos algunos detalles que dependen del programa de cómputo.

Para mostrar cómo producir una sinusoide con amplitud variable en el tiempo, necesitaremos introducir dos unidades generadoras. Primero, necesitamos una sinusoide pura la cual está hecha con un oscilador. La figura 1.5 (parte a) muestra una representación pictórica de un oscilador sinusoidal como un ícono. La entrada es una frecuencia (en ciclos por segundo), y la salida es una sinusoide cuya amplitud pico es uno.

La figura 1.5 (parte b) muestra cómo multiplicar la salida de un oscilador sinusoidal por un factor de escala apropiado $y[n]$ para controlar su amplitud. Ya que la amplitud pico del oscilador es 1, la amplitud pico del producto es $y[n]$, asumiendo que $y[n]$ cambia con suficiente lentitud y no tiene valores negativos.

La figura 1.6 muestra cómo la sinusoide de la Figura 1.1 es afectada por el cambio de amplitud de dos controladores de señal diferentes $y[n]$. El control de la señal en la parte (a) tiene una discontinuidad y de esa manera da como resultado la sinusoide de amplitud controlada mostrada en (b). Las partes (c) y (d) muestran una posibilidad de variación menos brusca para $y[n]$ y su resultado. La intuición sugiere que el resultado mostrado en (b) no sonará como una sinusoide que varía en su amplitud, si no como una sinusoide interrumpida por un “pop” audible luego del cual continúa con menos intensidad. En general, por razones que no pueden ser explicadas en este capítulo las señales $y[n]$ que controlan la amplitud en forma de rampa suave entre un valor y otro son menos propensas a dar resultados parásitos (tales como el “pop”), que las que cambian abruptamente.

Por el momento podemos dar dos reglas generales sin justificarlas. La primera, que las sinusoides puras son las señales más sensibles a los efectos parásitos de los cambios rápidos de amplitud. Así que cuando usted quiera examinar una transición de amplitud, si esta trabaja para las sinusoides probablemente trabajará para otras señales igual de bien. Segundo, que dependiendo de la señal cuya amplitud usted esté cambiando, el control de amplitud requerirá entre de 0 a 30 milisegundos de tiempo “rampa” -cero para las señales más clementes (tales como el ruido blanco), y 30 para las menos (como una sinusoide). Todo esto depende también en una forma complicada, de los niveles de escucha y del contexto acústico.

Funciones apropiadas de control $y[n]$ pueden ser hechas utilizando un *generador de envolvente*. La figura 1.7 muestra una red en la cual un generador de envolvente se utiliza para controlar la amplitud de un oscilador. Los generadores de envolvente varían ampliamente en su diseño, pero nos enfocaremos en su tipo más simple, el cual genera segmentos de línea como se muestra en la figura 1.6 (parte c). Si un segmento de línea está específicamente diseñado para hacer de rampa entre dos valores de salida a y b sobre N número de muestras comenzando en la muestra M, la salida es:

$$y[n] = a + ((b - a)*(n - M)/N), \quad M \le n \le M + N - 1$$

La salida puede tener cualquier número de segmentos tales como este conectados en los extremos sobre el rango completo del número de muestras n; los segmentos planos, horizontales, pueden hacerse dejando que $a = b$.

Adicional al cambio de las amplitudes de los sonidos, el control de amplitud es usual, especialmente en las aplicaciones en tiempo real, simplemente para encender sonidos y para enmudecerlos: se enmudecen haciendo que la rampa de amplitud vaya suavemente a cero. La mayoría de los paquetes de programas de

cómputo para síntesis también proveen fórmulas para detener por completo los módulos desde los cómputos de las muestras, pero aquí utilizaremos control de amplitud en su lugar.

El generador de envolvente existe desde la era análoga [Str95, pág.64] [Cha80, pág.90], al igual que el resto de componentes de la figura 1.7; los osciladores de frecuencia controlable fueron denominados osciladores controlados por voltaje o VCO's, y el paso de multiplicación se hizo utilizando un amplificador controlado por voltaje o VCA [Str95, págs.34-35] [Cha80, pág.84-89]. Los generadores de envolvente se describen con mayor detalle en la sección 4.1.

1.6. Superponiendo señales

Si una señal $x[n]$ tiene una amplitud pico o RMS A (en alguna ventana fija), entonces la señal escalada $k*x[n]$ (donde $k \geq 0$) tiene una amplitud kA. La potencia media de la señal escalada cambia por un factor de k^2. La situación se vuelve más complicada cuando dos señales diferentes se añaden simultáneamente; conocer las amplitudes de las dos no es suficiente para conocer la amplitud de la suma. Las dos medidas de amplitud deben por lo menos obedecer el triángulo de las desigualdades; para cualquiera de dos señales $x[n]$ y $y[n]$,

$$A_{\text{pico}}\{x[n]\} + A_{\text{pico}}\{y[n]\} \geq A_{\text{pico}}\{x[n] + y[n]\}$$

$$A_{\text{RMS}}\{x[n]\} + A_{\text{RMS}}\{y[n]\} \geq A_{\text{RMS}}\{x[n] + y[n]\}$$

Si fijamos una ventana desde M hasta $M + N - 1$ como es usual, podemos escribir la salida de la potencia media de la suma de las dos señales:

$$P\{x[n] + y[n]\} = P\{x[n]\} + P\{y[n]\} + 2*\text{COV}\{x[n],y[n]\}$$

donde hemos introducido la covariancia de dos señales:

$$\text{COV}\{x[n],y[n]\} = (x[M]y[M] + \ldots + x[M + N - 1]y[M + N - 1])/N$$

La covariancia puede ser positiva, cero o negativa. Sobre una ventana suficientemente grande, la covariancia de dos sinusoides con frecuencias diferentes es insignificante comparada con la potencia media. Dos señales que no tengan covariancia se les llama *no correlacionadas* (la correlación es la covariancia normalizada que se encuentra entre -1 y 1). En general, para dos señales no correlacionadas, la potencia de la suma es igual a la suma de las potencias:

$$P\{x[n] + y[n]\} = P\{x[n]\} + P\{y[n]\}, \text{ siempre que } \text{COV}\{x[n],y[n]\} = 0$$

Al ponerlo en términos de amplitudes, se tiene:

$$(A_{RMS}\{x[n] + y[n]\})^2 = (A_{RMS}\{x[n]\})^2 + (A_{RMS}\{y[n]\})^2$$

Esta es la ya conocida relación pitagórica. De esta manera las señales no correlacionadas pueden ser tomadas como vectores que conforman un ángulo recto; las señales correlacionadas positivamente forman un ángulo agudo entre ellas y las correlacionadas negativamente forman un ángulo obtuso.

Por ejemplo, si dos señales no correlacionadas tienen ambas una amplitud RMS de valor a, su suma tendrá una amplitud $\sqrt{2}a$. De otro lado, si las dos señales son iguales, -la mayor correlación posible- la suma tendrá una amplitud $2a$, el cual es el valor máximo permitido, según el triángulo de las desigualdades.

1.7 Señales periódicas

Se dice que una señal $x[n]$ se repite en un período τ si

$$x[n + \tau] = x[n]$$

para todo n. Tal señal deberá repetirse en períodos de 2τ y así sucesivamente; el τ más pequeño en el cual se repite la señal se denomina *período* de la señal. En la discusión de los períodos de las señales de audio digital, rápidamente llegamos a la dificutad de describir señales cuyo "período" no es un entero, de tal manera que la ecuación anterior queda sin sentido. Por el momento ignoraremos efectivamente esta dificultad suponiendo que la señal puede ser interpolada de alguna manera entre las muestras de tal manera que esté bien definida ya sea n un entero o no.

Una sinusoide tiene un período (en muestras) de $2\pi/\omega$ donde ω es la frecuencia angular. De una manera más general, cualquier suma de sinusoides con frecuencias $2\pi k/\omega$ para enteros de k, se repetirá después de $2\pi/\omega$ número de muestras. Tal suma es llamada *Serie de Fourier*:

$$x[n] = a_0 + a_1\cos(\omega n + \phi_1) + a_2\cos(2\omega n + \phi_2) + \ldots + a_p\cos(p\omega n + \phi_p)$$

Más aún, si asumimos ciertas condiciones técnicas (en efecto que la señal únicamente contiene frecuencias hasta una frontera finita), podemos representar cualquier señal peródica con dicha suma. Esta es la variante discreta en el tiempo del análisis de Fourier, que reaparecerá en el capítulo 9.

Las frecuencias angulares de las sinusoides anteriores son todas múltiplos enteros de ω. Estas son llamadas los *armónicos* de ω, el cual es llamado a su vez la *fundamental*. En términos de su afinación, los armónicos ω, 2ω,... son intervalos de 0, 1200, 1902, 2400, 2786, 3102, 3369, 3600, ..., cents sobre la fundamental; esta secuencia de tonos es a veces llamada *serie armónica*. Las primeras seis notas de la serie son muy aproximadamente múltiplos de 100; en otras palabras, los primeros seis armónicos de una nota en la escala occidental caen muy cercanos a otras notas (pero no siempre de manera exacta) de la misma escala; la tercera y la sexta se desvían únicamente en 2 cents y la quinta se desvía por 14 cents.

Puesto de otra manera, la razón de frecuencias 3:2 (una quinta justa en la terminología occidental) equivale casi exactamente a siete semitonos, 4:3 (una cuarta justa) está muy cercano a cinco semitonos, y las razones 5:4 y 6:5 (tercera mayor y menor) están muy cercanas a de los cuatro y los tres semitonos, respectivamente.

Se muestra una serie de Fourier (con sólo tres términos) en la figura 1.8. las primeras tres gráficas son de sinusoides, cuyas frecuencias están en razones 1:2:3. El período común está marcado sobre los ejes horizontales. Cada sinusoide tiene una amplitud y una fase inicial diferentes. La suma de las tres en la parte inferior no es una sinusoide, pero sigue manteniendo la periodicidad compartida por las tres sinusoides componentes.

Dejando la cuestión de la fase a un lado, podemos utilizar un banco de osciladores sinusoidales para sintetizar sonidos periódicos o incluso cambiar su forma suavemente a través de una sucesión de sonidos periódicos, especificando la frecuencia fundamental y las (posiblemente variantes en el tiempo) amplitudes de los parciales. La figura 1.9 muestra un diagrama de bloques para hacer esto.

Este es un ejemplo de *síntesis aditiva*; y de manera más general el término puede ser aplicado a redes en las cuales la frecuencia de los osciladores son controlables de manera independiente. Los primeros días de la música por computador sonaron con la síntesis aditiva.

1.8 Acerca de los ejemplos con programas de cómputo

Los ejemplos de este libro utilizan Pure Data (Pd), y para entenderlos usted debe aprender lo mínimo de Pd mismo. Pd es un ambiente para realizar rápidamente aplicaciones musicales en el computador, con el fin principal de las ejecuciones de música en vivo. Pd puede ser utilizado para otros medios, pero no iremos a eso aquí.

Existen algunos otros ambientes de audio DSP por parches junto con Pd. El más ampliamente utilizado es ciertamente Csound de Barry Vercoe [Bou00], el cual difiere de Pd en que está basado en el texto (no basado en GUI). Esto es ventajoso en algunos aspectos y desventajoso en otros. Csound está mejor adaptado que Pd para procesos de vaciado y maneja la polifonía mucho mejor que como lo hace Pd. De otro lado, Pd tiene una estructura de control de tiempo real más desarrollada que Csound. Genealógicamente, Csound proviene de los llamados lenguajes Music N [Mat69, págs.115-172].

Otra alternativa de código abierto de amplio uso es el SuperCollider de James McCartney, el cual está también más orientado al texto que Pd, pero al igual que Pd, está explícitamente diseñado para el uso en tiempo real. SuperCollider tiene construcciones lingüísticas poderosas que lo hacen una herramienta más adecuada que Csound para tareas como la escritura de lazos o el mantenimiento de estructuras de datos complejas.

Finalmente, Pd tiene un hermano utilizado ampliamente, el programa comercial de Cycling74 Max/MSP (los demás mencionados aquí son todos de código abierto). Tanto principiantes como gerentes de sistemas a cargo de laboratorios de cómputo mutiusuarios y multipropósito encontrarán Max/MSP mejor soportado y documentado que Pd. Es posible aprender en Pd y aplicar lo aprendido en Max/MSP y viceversa, e incluso es posible conectar parches de uno a otro, pero los dos no son verdaderamente compatibles.

Introducción rápida a Pd

Los documentos de Pd son llamados *parches*. Ellos corresponden muy aproximadamente a las cajas de los diagramas de bloques abstractos mostrados anteriormente en este capítulo, pero en el detalle son diferentes, debido a que Pd es un ambiente de implementación, no un lenguaje de especificación.

Un parche Pd, tal como los mostrados en la figura 1.10, consiste en un conjunto de *cajas* conectadas en red. El borde de las cajas le dice cómo interpretar su texto y cómo funcionan. En la parte (a) de la figura vemos tres tipos de cajas. De arriba a abajo son:

- una *caja de mensaje*. Las cajas de mensajes, con un borde en forma de bandera, interpretan el texto como un mensaje para enviar, siempre y cuando la caja sea activada (por un mensaje de entrada o por un dispositivo de señalización). El mensaje en este caso consiste simplemente del número “21”.

- una *caja de objetos*. Las cajas de objetos tienen un borde rectangular; interpretan el texto para crear objetos cuando usted carga un parche. Las cajas de objetos pueden contener cientos de objetos de diferentes tipos -incluyendo osciladores, generadores de envolventes, y otros módulos de procesamiento de señal que se presentarán más tarde- dependiendo del texto en su interior. En este caso la caja contiene un sumador. En gran parte de los parches Pd, la mayoría de las cajas son del tipo “objeto”. La primera palabra escrita en un objeto especifica su *clase*, la cual en este caso es precisamente “+”. Cualquier palabra adicional (separada por un espacio) que aparezca en la caja es llamada *argumento de creación*, y especifica el estado inicial del objeto cuando se crea.

- una *caja de número*. Las cajas de número son un tipo particular de *caja GUI*. Otros tipos incluyen botones para oprimir e interruptores de palanca;

estos vendrán más adelante en los ejemplos. La caja de número tiene un borde con forma de tarjeta perforada, con una muesca en su esquina superior derecha. Mientras que la apariencia de un objeto o caja de mensaje queda fija cuando el parche está corriendo, el contenido (texto) de la caja de número cambia para reflejar el valor actual de la caja. Usted puede utilizar también una caja de números como control haciendo click y arratrando arriba y abajo, o escribiendo valores en ella.

En la figura 1.10 (parte a) al hacer click en la caja de mensaje, esta envía el mensaje "21" a una caja de objeto que le suma 13. las líneas que conectan las cajas llevan datos de una caja a la siguiente; las salidas de las cajas están en la parte inferior y las entradas en la parte superior.

La figura 1.10 (parte b) muestra un parche Pd que fabrica una sinusoide con frecuencia y amplitud controlables. Las líneas de conexión del parche son de dos tipos aquí; las delgadas se usan para llevar *mensajes* esporádicos, y las gruesas (que están conectando el oscilador, el multiplicador y el objeto de salida `dac~`) llevan señales de audio digital. Ya que Pd es un programa de tiempo real, las señales de audio fluyen continuamente. De otro lado, los mensajes esporádicos aparecen en específicos pero probablemente impredecibles instantes del tiempo.

Que una conexión lleve mensajes o señales depende de la conexión de la caja de la cual proviene; así, por ejemplo, el objeto `+` da como salidas, mensajes, pero el objeto `*~` da como salida una señal. Las entradas de un objeto dado pueden aceptar o no señales (pero siempre aceptan mensajes, incluso si es para solamente convertirlos en señales). Como convención, las cajas de objetos con entradas o salidas de señal son nombradas todas con una tilde al final ("~") como en "`*~`" y "`osc~`".

Cómo encontrar y hacer funcionar los ejemplos

Para hacer funcionar los parches, primero debe descargar, instalar y hacer funcionar Pd. Las instrucciones para hacer esto aparecen en la documntación HTML de Pd en línea, que puede encontrar en http://crca.ucsd.edu/~msp/software.htm.

Este libro debe aparecer en http://crca.ucsd.edu/~msp/techniques.htm, posiblemente con alguna revisiones. Escoja la revisión que corresponde al texto que está leyendo (o tal vez sólo la última) y descargue el archivo que contiene la revisión asociada de los ejemplos (puede descargar también un archivo de la versión HTML de este libro para un acceso más fácil en su máquina). Los ejemplos deberán quedar todos en un directorio aparte, ya que algunos de ellos dependen de otros archivos en ese directorio y podrían no cargarse correctamente si ha movido algunas cosas.

Si quiere copiar uno de los ejemplos a otro directorio para construir en el (para lo cual es bienvenido), deberá incluir también el directorio de ejemplos en la ruta de búsqueda de Pd (vea la documentación de Pd) o mirar qué otros n archivos necesita y copiarlos también. Una buena manera de saber cuáles son es poner a funcionar Pd desde el archivo reubicado y ver si Pd se queja de algo que no puede encontrar.

Deben aparecer docenas de archivos en las carpetas de los "ejemplos", incluyendo los ejemplos mismos y los archivos de soporte. Los nombres de los archivos de los ejemplos comienzan todos con una letra (A para el capítulo 1, B para el 2, etc) y un número como en "A01.sinewave.pd".

Los parches de ejemplo también se distribuyen con Pd pero tenga precaución porque puede encontrar una versión diferente de los ejemplos que no corresponda con el texto que está leyendo.

1.9 Ejemplos

Escalado de amplitud constante

El ejemplo A01.sinewave.pd, que se muestra en la figura 1.11, contiene esencialmente el parche más simple posible para hacer un sonido, con sólo tres cajas de objetos (también hay comentarios, y dos cajas de mensajes para encender y apagar el procesador de audio "DSP" de Pd). Las tres cajas de objetos son:

osc~: oscilador sinusoidal. La entrada de la izquierda y la salida son señales de audio digital. La entrada toma la frecuencia (que se puede variar en el tiempo) en Hertz. La salida es una sinusoide en una frecuencia específica. Si no hay nada conectado a la entrada de la frecuencia, el argumento de creación (440 en este ejemplo) se utiliza como la frecuencia. La salida tiene su pico de amplitud uno. Usted puede ajustar una fase inicial enviando mensajes (no señales de audio) a la entrada derecha. La entrada izquierda (la frecuencia) también puede ajustarse para enviar mensajes y ajustar la frecuencia, ya que a cualquier entrada que tome una señal de audio, también pueden enviársele mensajes que son convertidos automáticamente en la señal de audio digital deseada.

*~: multiplicador. Existe de dos formas. Si se especifica una creación de argumento (como en este ejemplo, que es 0,05), esta caja multiplica una señal de audio digital (en la entrada izquierda) por el número; los mensajes en la entrada derecha pueden también actualizar ese número. Si no se da ningún argumento, esta caja multiplica las dos señales entrantes de audio digital.

dac~: dispositivo de salida de audio. Dependiendo de su equipo de cómputo, este podría no actuar como un Convertidor Análogo/Digital tal y como su nombre lo sugiere; pero en general, le permite enviar cualquier señal de audio digital a la(s) salida(s) de audio de su computador. Si no hay creación de argumentos, el comportamiento por defecto es entregar salida a los canales uno y dos de la tarjeta de sonido; usted puede especificar números de canales alternativos (uno o muchos) utilizando argumentos para su creación. Pd mismo puede configurarse para utilizar dos o más canales de salida, o podría no tener el dispositivo de audio abierto; consulte la documentación de Pd para estos detalles.

Las dos cajas de mensajes muestran una particularidad en la manera en que los mensajes son analizados en las cajas de mensajes. Al principio en la figura 1.10 (parte a), el mensaje consistía únicamente del número 21. Cuando se hace click en él, esa caja envía el mensaje "21" a su salida y por lo tanto, a cualquier objeto conectado a ésta. En este ejemplo en particular, el texto de la caja de mensaje comienza con un punto y coma. Este es un cierre entre mensajes (así, el primer mensaje está vacío), después del cual la siguiente palabra se toma como el nombre del receptor del siguiente mensaje. De esta manera el mensaje aquí es "dsp 1" (o "dsp 0") y el mensaje que se va a enviar, no para ninguno de los objetos conectados -no hay ninguno, de todas maneras- si no para el objeto llamado "pd". Este objeto en particular se provee de manera invisible por el programa Pd y usted puede enviar varios mensajes para controlar el estado global de Pd, en este caso encendiendo el procesamiento de audio ("1") y apagándolo ("0").

Control de amplitud en decibeles

El ejemplo A02.amplitude.pd muestra cómo hacer un control de amplitud crudo; los elementos activos se muestran en la figura 1.12 (parte a). Hay una clase de objeto nueva:

dbtorms: conversión de decibeles a amplitud lineal. "RMS" es un término errado; debería llamarse "dbtoamp", ya que este objeto realmente convierte decibeles a cualquier unidad de amplitud lineal, sea esta RMS, pico u otra. Una entrada de 100 dB es normalizada a una salida de 1. Los valores más grandes que 100 están bien (120 nos dará 10), pero valores menores o iguales a cero darán salida cero (una entrada de cero, de otro lado, podría tener como salida un pequeño valor

positivo). Este es un objeto de control, pues los números que entran y salen son mensajes, no señales (Un objeto correspondiente, dbtorms~, es el correlativo de señal. Sin embargo, como objeto de señal este es costoso en tiempo de CPU y la mayoría de las veces encontraremos una u otra manera de evitar su utilización.)

Las dos cajas de números están conectadas a la entrada y a la salida del objeto dbtorms. La función de entrada es como un control; trabaje con el ratón en esta (click y arrastre arriba y abajo) para cambiar la amplitud. Esta ha sido ajustada para que quede en el rango de 0 a 80, con el fin de proteger sus parlantes y sus oídos, y es prudente construir tales guardas en sus propios parches.

La otra caja de números muestra la salida del objeto dbtorms. Es inútil trabajar esta caja con el mouse ya que su salida no está conectada a ningún lado; aquí sólo se muestra su entrada. Las cajas de números pueden ser útiles como control, indicador o ambos, aunque si las está utilizando para ambas cosas, puede haber un trabajo extra por hacer.

Control de amplitud suavizado con un generador de envolvente

Según se muestra en la figura 1.6 una de las maneras de suavizar los cambios de amplitud en una señal, sin clicks, es multiplicarla por la salida de un generador de envolvente tal como se muestra en el diagrama de bloques de la figura 1.7. Esto se puede implementar en Pd utilizando el objeto line~:

line~: generador de envolvente. La salida es una señal que ejecuta una pendiente lineal desde un valor a otro, en el tiempo, según lo determinen los mensajes recibidos. La entrada toma mensajes que especifican los valores finales (entrada izquierda) y los tiempos que se demora (entrada derecha). Debido a una regla general para los mensajes de Pd, un par de números enviados a la entrada izquierda son suficientes para para especificar de manera conjunta un valor final y un espacio de tiempo. El tiempo se mide en milisegundos (tomando en cuenta la velocidad de las muestras), y el valor final no tiene unidades, o sea que su rango de salida podría ser según la entrada a la que estuviese conectado.

El ejemplo A03.line.pd demuestra la utilización de un objeto line~ para controlar la amplitud de una sinusoide. La parte activa se muestra en la figura 1.12 (parte b). Las seis cajas de mensajes están todas conectadas al objeto line~, y son activadas al hacer click en ellas; la primera caja de la parte superior, por ejemplo, especifica que la rampa de line~ (comenzando donde fuere que estuviera su valor de salida antes de recibir el mensaje) al valor 0,1 en un lapso de dos segundos. Luego de que transcurren estos dos segundos, a no ser de que otro mensaje haya llegado mientras tanto, la salida permanecerá en 0,1. Los mensajes pueden llegar antes de que culmine el tiempo de los dos segundos, en cuyo caso el objeto line~ abandona su vieja trayectoria y toma la nueva.

Dos mensajes pueden llegar a line~ al mismo tiempo o con tan poco espacio de tiempo entre ellos que no se efectúe el cómputo DSP; en este caso, el primer mensaje no tiene efecto, debido a que line~ no habrá cambiado su salida aún para seguir ese primer mensaje, y su salida corriente, sin cambio, se utiliza entonces como punto de partida para el segundo segmento. Una excepción a esta regla es que si line~ llega a un valor cero, el valor de la salida es ajustado inmediatamente al nuevo valor y los segmentos subsiguientes comenzarán desde ese nuevo valor; así, al enviar dos pares, el primero con un valor de tiempo cero y el segundo par con un valor de tiempo diferente de cero, uno puede especificar independientemente los valores iniciales y finales de un segmento en la salida de line~.

El tratamiento de la entrada derecha de line~ es inusual para los objetos Pd pues olvida los valores viejos; un mensaje con un solo número tal como “0.1” es siempre equivalente al par “0.1 0”. Casi cualquier otro objeto retendrá el valor previo por la entrada derecha, en lugar de reajustarlo a cero.

El ejemplo A04.line2.pd muestra la salida del objeto line~ gráficamente. Utilizando los mensajes de las diversas cajas, usted puede recrear los efectos mostrados en la figura 1.6.

Triada mayor

El ejemplo A05.output.subpatch.pd, cuyos ingredientes activos se muestran en la figura 1.12 (parte c), presenta tres sinusoides con frecuencias en razones 4:5:6, de tal forma que las dos más bajas están separadas por una tercera mayor, las dos superiores por una tercera menor y la superior y la inferior por una quinta. La frecuencia más baja es 440, igual al A sobre el C medio, o MIDI 69. Las demás están cuatro y siete semitonos más altas, respectivamente. Las tres tienen amplitudes iguales.

El control de amplitud en este ejemplo está manejado por un nuevo objeto llamado output~. Este no es un objeto de Pd, si no que es un parche Pd alojado en un archivo, "output.pd". (Puede ver lo que hay en output~ abriendo el menú de propiedades de la caja y seleccionando "open".) Usted tiene dos controles, uno para la amplitud en dB (100 significa "ganancia unitaria"), y un botón "mute". El proceso de audio de Pd es encendido automáticamente cuando ajusta el nivel de entrada -este podría no ser el mejor comportamiento en general, pero es apropiado para estos parches de ejemplo <en la versión extendida este parche está modificado -mejorado>. El mecanismo para incluir un parche Pd dentro de otro como una caja de objetos se discute en la sección 4.7.

Conversión entre frecuencia y nota

El ejemplo A06.frequency.pd (figura 1.13) muestra el objeto Pd que convierte notas a frecuencia (mtof que significa "MIDI a frecuencia") y su inverso ftom, Introducimos también dos clases de objetos, send y receive.

mtof, ftom: convierten notas MIDI a unidades de frecuencia de acuerdo con las Fórmulas de Conversión Nota/Frecuencia. Las entradas y salidas son mensajes (los objetos "tilde" equivalentes existen, sin embargo, al igual que dbtorms~ son costosos en tiempo de CPU). La salida del objeto ftom es 1500 si la entrada es cero o negativa; y de otra manera, si usted da un valor de -1500 o menor a mtof su salida es cero.

receive, r: los mensajes receive no son locales. El objeto receive, el cual puede abreviarse "r", espera mensajes externos enviados por un objeto send o por una caja de mensajes que utilice redireccionamiento (el ";" discutido antes en el ejemplo A01.sinewave.pd). El argumento (tal como "frecuency" y "pitch" en este ejemplo) es el nombre al cual se enviarán los mensajes. Objetos receive múltiples pueden compartir el mismo nombre, en cuyo caso cualquier mensaje enviado con ese nombre llegará a todos éstos.

send, s: El objeto send, que puede ser abreviado "s", envía mensajes a los objetos receive.

Dos nuevas propiedades de cajas de números se utilizan aquí. Anteriormente las hemos utilizado como cajas de control y de indicación; aquí las dos cajas de números hacen ambas funciones. Si una caja de números tiene un valor numérico en su entrada, esta no sólo muestra el número si no que también repite el número en su salida. Sin embargo a una caja de número también puede ser enviado un mensaje de "ajuste", tal como se hace con "set 55" en el ejemplo. Este deberá ajustar el valor de la caja de número a 55 (y mostrarlo) pero no hará que sea 55 el valor numérico de la salida. En este caso los números que llegan desde dos objetos receive están formateados (utilizando cajas de mensaje) para leer "set 55" en lugar de "55" u otro número de la misma manera. (La palabra especial "$1" se reemplaza por el número de entrada.) Esto se hace debido a que de otra manera obtendríamos un lazo infinito: la frecuencia cambiará la nota y esta cambiará la

frecuencia y así por siempre o hasta que algo se dañe.

Más síntesis aditiva

La triada mayor (ejemplo A06.frequency.pd) muestra una manera de combinar sinusoides, sumándolas. Hay muchas otras maneras de organizar conjuntos de sinusoides, de las cuales veremos dos. El ejemplo A07.fusion.pd (figura 1.14) muestra cuatro osciladores, cuyas frecuencias están en razones 1:2:3:4, con amplitudes relativas 1, 0,1, 0,2 y 0,5. las amplitudes están ajustadas multiplicando las salidas de los osciladores (el objeto *~ debajo de los osciladores).

Los osciladores segundo, tercero y cuarto se encienden y apagan mediante el interruptor de palanca. Este es un control gráfico, al igual que la caja de número introducida anteriormente. El interruptor de palanca cambia a 1 y a 0 alternativamente cada que se hace click en él con el ratón. Este valor es multiplicado efectivamente por la suma del segundo, tercer y cuarto oscilador, hacéndolos que se enciendan y se apaguen.

Incluso cuando todos los osciladores están combinados (con el interruptor de palanca en la posición “1”), el resultado se fusiona en un solo tono, escuchándose la nota dada por el primer oscilador de la izquierda. En efecto, este parche suma una serie de Fourier de cuatro términos que generan una onda periódica compleja.

El ejemplo A08.beating.pd (figura 1.15) muestra otra posibilidad, en la cual seis osciladores están afinados en tres pares de fercuencias vecinas, por ejemplo 330 y 330,2 Hertz. Estos pares entran y salen de fase uno con otro, de tal manera que la amplitud de su suma cambia en el tiempo. Con el nombre de *beating <batido o vibrato>* este fenómeno es utilizado con fercuencia para efectos musicales.

Los osciladores se pueden combinar de otras maneras similares simplemente sumando sus salidas, y se puede obtener un amplio rango de sonidos diferentes. El ejemplo A09.frequency.mod.pd (que no se muestra aquí) muestra la síntesis por *modulación de la frecuencia*, en la cual un oscilador controla la frecuencia de otro oscilador. Esto se describirá de manera más completa en el capítulo 5.

Ejercicios

1. Una sinusoide $x[n] = a\cos(\omega n + \phi)$ tiene una fase inicial $\phi = 0$ y una frecuencia angular $\omega = \pi/10$. Cuál es su periodo en muestras? Cuál es la fase en la muestra $n = 10$?

2. Dos sinusoides tienen periodos de 20 y 30 muestras, respectivamente. Cuál es el periodo de la suma de los dos?

3. Si 0 dB corresponden a una amplitud de 1, cuántos dB corresponden a amplitudes de 1,5, 2, 3 y 5?

4. Dos señales no correlacionadas de amplitud RMS 3 y 4 se suman; cuál es la amplitud RMS de la suma?

5. Cuántas señales no correlacionadas, todas con amplitud igual, debería usted adicionar para lograr una señal 9 dB más grande en amplitud?

6. Cuál es la frecuencia del C medio a 44.100 muestras por segundo?

7. Dos sinusoides suenan un C medio (MIDI 60) y un C sostenido medio (MIDI 61). Cuál es la diferencia, en Hertz, entre sus frecuencias?

8. Cuántos cent hay en el intervalo entre el séptimo y el octavo armónico de una señal periódica?

9. Si una señal de audio $x[n]$, $n = 0,...,N - 1$ tiene una amplitud pico de 1, cuál es la amplitud RMS mínima posible? Cuál es la máxima posible?