**Sintético** se define como aquello que se forma componiendo desde las partes al todo.

Un **sintetizador** de sonido es un dispositivo que permite realizar un proceso de síntesis, generando el sonido a partir de la combinación de elementos simples como señales periódicas y/o funciones matemáticas.

Con un sintetizador podemos:

* Generar una infinidad de sonidos totalmente nuevos.
* Imitar sonidos ya existentes como los de instrumentos acústicos con mayor o menor fiabilidad.

**Síntesis analógica**

Se realiza solo con circuitos analógicos.

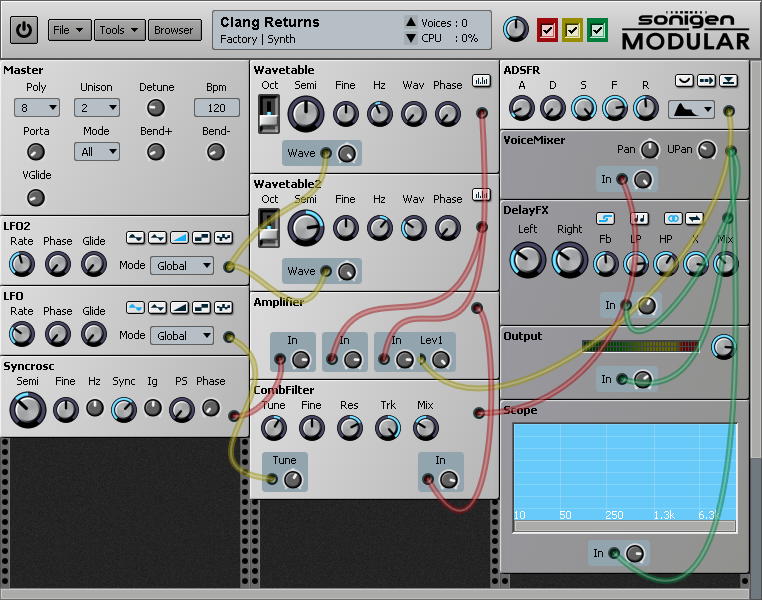
Hacia 1964-65, Robert Moog y Don Buchla inventan cada uno por separado los primeros sintetizadores controlados por voltaje.

En 1968 aparece el disco de Walter/Wendy Carlos, Switch on Bach, con temas de J. S. Bach interpretados por el **Moog.**

En 1969 Keith Emerson (ELP) y muchos otros teclistas pop y rock comienzan a utilizar sintetizadores (por ejemplo Lucky Mab).

En 1970 se lanza el portátil **Minimoog**, que se convierte en uno de los sintetizadores más populares (ELP, Stevie Wonder, Blood Sweat and Tears, Frank Zappa, Yes, Pink Floyd, etc en el pop-rock, y por Jan Hammer, Chick Corea, Herbie Hancock, etc en el jazz).

El Minimoog se fabrica hasta entrados los 80s.

****

**Síntesis digital**

Se realiza con computadores y software especifico.

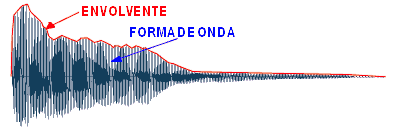
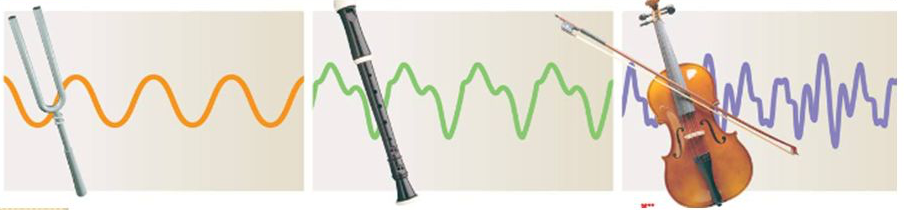
En 1957, Max Matthews realiza la primera síntesis digital en los laboratorios de Bell usando el lenguaje de síntesis sonora Music I que dará́ paso a Music II (58), III (60), IV (62), V y otros posteriores como Music 10, Music 360, CMix, cmusic, Common Lisp Music y **CSound** creado por B. Vercoe en 1986

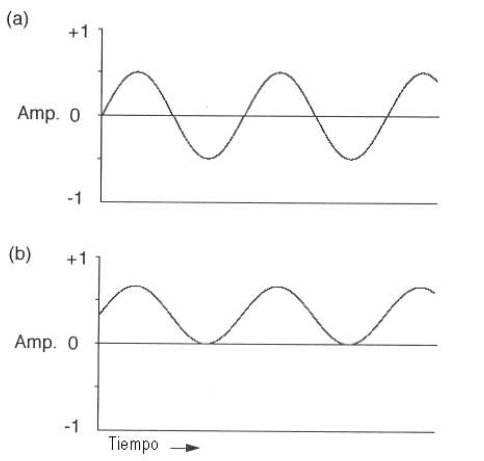
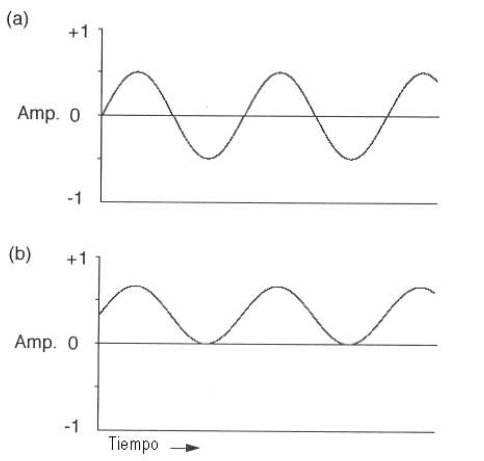
Los primeros compositores en utilizarlo, en los 60s son James Tenney (Dialogue, 1963), Jean Claude Risset, Richard Moore, Barry Vercoe, John Chowning (inventor de la síntesis FM en 1977), Gerald Bennet, Charles Dodge, etc.

**Forma de onda y envolvente**

Generalmente la síntesis de sonido se consigue mediante dos pasos:

* Generar una forma de onda con un contenido armónico o timbre interesante. Para esto se usa algún tipo de síntesis.
* Generar una envolvente que modifique la amplitud de la onda anterior en el tiempo. Para esto se usa un generador de envolvente ADSR.

****

**Señales unipolares y bipolares**

Bipolar alterna valores positivos y negativos.

Unipolar es siempre positiva, se considera **la suma** de una señal bipolar y un valor constante.

****

**Una onda** tiene como parámetros la amplitud, la frecuencia y fase, cualquiera de ellos se puede modular (modificar)**.**

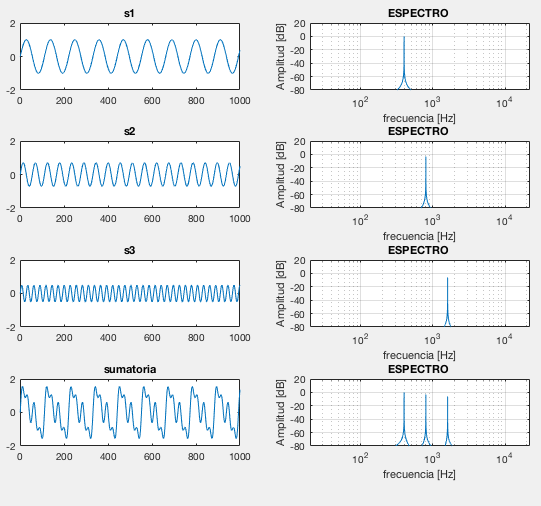
**Portadora,** representa la señal que será modulada, es decir aquella a la que se le modificara un parámetro en función de la señal moduladora.

**Moduladora,** representa la información a transmitir que modificara algún aspecto de la portadora.

**Modulada** es la señal después del proceso de modulación.

|  |  |  |
| --- | --- | --- |
| **Tipo** | **Definición** | **Gráficos** |
| **Síntesis Aditiva.** | Se suman ondas senoidales de distintas frecuencias y amplitudes para formar una onda compleja.  Las frecuencias pueden estar relacionadas armónicamente o no.  **Su espectro tiene** tantas barras como frecuencias se sumen |  |
| **Síntesis Substractiva.** | Se genera una señal compleja (triangular, diente de sierra, cuadrada u otras pero no senoidal) y luego se  filtra con un LPF para modificar su contenido armónico restando frecuencias.  **Su espectro** tiene tantas barras como frecuencias tenga la onda original después de filtrada |  |
| **Modulación de anillo**  **(Ring Modulation**  **o RM)** | Multiplicación de dos señales bipolares, una portadora y una moduladora.  Ambas pueden Senoidales, triangulares, diente de sierra cuadradas u otras.  **Su espectro** tiene dos barras con la suma y la diferencia de las frecuencias fm+fp y fm-fp Con amplitud mitad de las originales.  Si fp=1000 fm=10000 |  |
| **Modulación de amplitud (AM)** | Corresponde a la multiplicación de una señal bipolar llamada portadora, con una señal unipolar llamada moduladora.  **Su espectro** tiene tres barras : una en fp y dos bandas laterales con la suma y diferencia fm+fp y fm-fp de amplitud mitad de las originales  **En el ejemplo fp=** |  |
| **Modulación de frecuencia (FM)**  **Equivalente a la modulación de fase** | La frecuencia de una señal bipolar portadora, es modulada (suma) por otra señal bipolar moduladora.  I es el índice de modulación.  La modulación de fase es equivalente a modulación de frecuencia.  **Su espectro** la fm y una serie de frecuencias distribuidas de forma simétrica alrededor de ella a todos los múltiplos enteros + y - de fp |  |
| **Modulación Waveshaping** | Se hace pasar la señal a modular (Asinwt) por una función de transferencia in/out no líneal f(x), de manera que se producen distorsiones en la forma de onda.  Matemáticamente equivale a multiplicar cada valor de la amplitud de la señal Asinwt por el que corresponde en f(x)  **Su espectro** depende del tipo de función f(x) usada y produce en general una alta cantidad de frecuencias. |  |

**Síntesis en MatLab**



%% sintesis aditiva

clf,clc

fs=44100;

n=0:1/fs:1;

s1=sin(2\*pi\*400\*n);

s2=0.7\*sin(2\*pi\*800\*n);

s3=0.5\*sin(2\*pi\*1600\*n);

sr=s1+s2+s3;

subplot(421);plot(s1(1:1000));ylim([-2 2]);title('s1')

subplot(423);plot(s2(1:1000));ylim([-2 2]);title('s2')

subplot(425);plot(s3(1:1000));ylim([-2 2]);title('s3')

subplot(427);plot(sr(1:1000));ylim([-2 2]);title('sumatoria')

subplot(422);plotfft(s1,fs,0)

subplot(424);plotfft(s2,fs,0)

subplot(426);plotfft(s3,fs,0)

subplot(428);plotfft(sr,fs,0)

%% sintesis substractiva

clf,clc

fs=44100;

n=0:1/fs:1;

s=sawtooth(2\*pi\*500\*n);

b=[1 2 1];

a=[1 -1.8 0.9];

y=filter(b,a,s);

subplot(421);plot(s(1:500));title('onda compleja')

subplot(425);plot(y(1:500));title('filtrada')

subplot(422);plotfft(s,fs,1)

subplot(426);plotfft(y,fs,1)

%% RM

clf,clc

fs=44100;

n=0:1/fs:1;

fp=10000;

sp=sin(2\*pi\*fp\*n);

sm=sin(2\*pi\*1000\*n);

sr=sp.\*sm;

subplot(321);plot(sp(1:100));ylim([-2 2]);title('portadora')

subplot(323);plot(sm(1:100));ylim([-2 2]);title('moduladora')

subplot(325);plot(sr(1:100));ylim([-2 2]);title('RM')

subplot(322);plotfft(sp,fs,1)

subplot(324);plotfft(sm,fs,1)

subplot(326);plotfft(sr,fs,1)

%% AM

clf,clc

fs=44100;

n=0:1/fs:1;

fp=10000;

sp=sin(2\*pi\*fp\*n);

sm=sin(2\*pi\*3000\*n);

sr=sp.\*(1+sm);

subplot(321);plot(sp(1:100));ylim([-2 2]);title('portadora')

subplot(323);plot(sm(1:100));ylim([-2 2]);title('moduladora')

subplot(325);plot(sr(1:100));ylim([-2 2]);title('AM')

subplot(322);plotfft(sp,fs,1)

subplot(324);plotfft(sm,fs,1)

subplot(326);plotfft(sr,fs,1)

%% FM

clf,clc

fs=44100;

n=0:1/fs:1;

fp=10000;

I=25;

sp=sin(2\*pi\*fp\*n);

sm=sin(2\*pi\*5000\*n);

sr=sin(2\*pi\*fp\*n+I\*sm);

subplot(321);plot(sp(1:100));ylim([-2 2]);title('portadora')

subplot(323);plot(sm(1:100));ylim([-2 2]);title('moduladora')

subplot(325);plot(sr(1:100));ylim([-2 2]);title('FM')

subplot(322);plotfft(sp,fs,1)

subplot(324);plotfft(sm,fs,1)

subplot(326);plotfft(sr,fs,1)

%% waveshaping

clf;clear;clc

Fs=44100; % define frec de muestreo

Ts=1/Fs; % calcula periodo de muestreo

t=0:Ts:(1-Ts); % crea un vector de tiempos

f=5; % define frecuencia

in=sin(1000\*pi\*f\*t); %sintetiza la onda sin

TH=0.7;

for i=1:length(in)

if in(i)>=TH

out(i)=TH;

elseif in(i)<=-TH

out(i)=-TH;

else

out(i)=in(i);

end

end

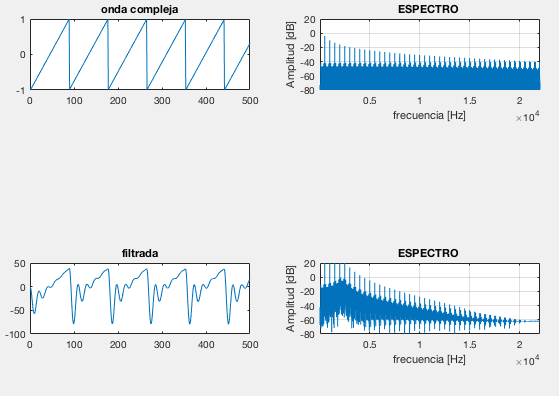
subplot(321);plot(in(1:100));ylim([-1 1]);title('in')

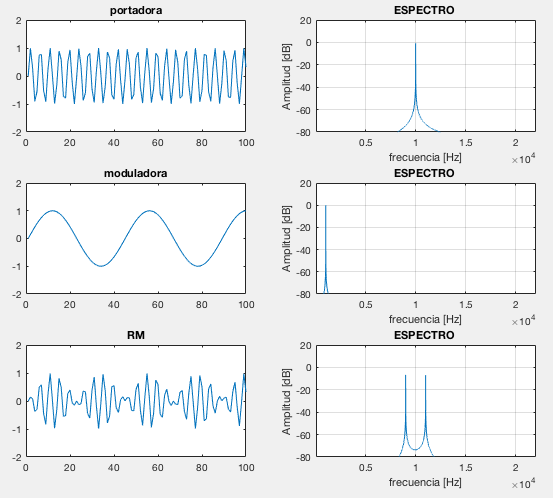
subplot(325);plot(out(1:100));ylim([-1 1]);title('out')

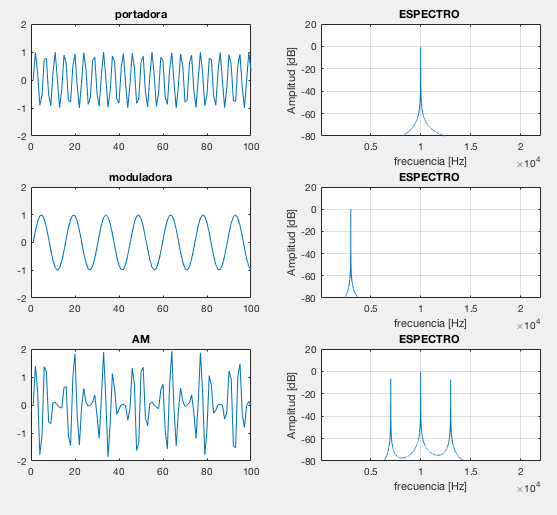
subplot(322);plotfft(in,Fs,1)

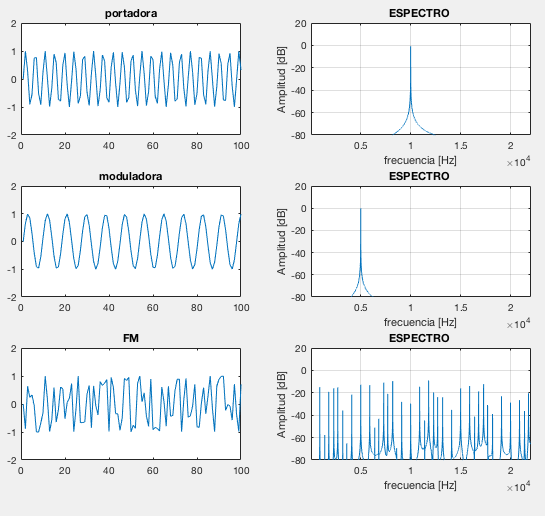
subplot(326);plotfft(out,Fs,1)

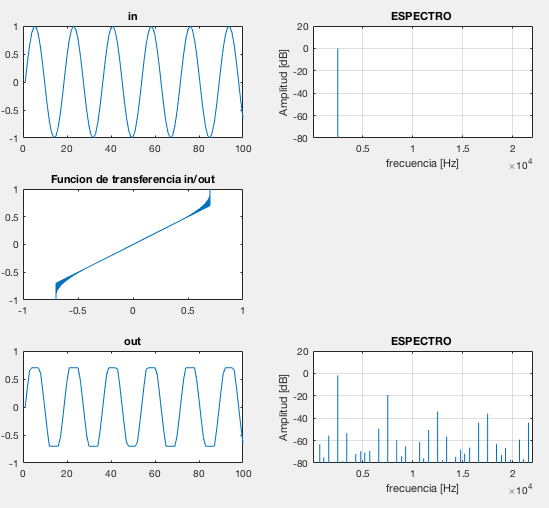
subplot(323);plot(out,in);title('Funcion de transferencia in/out')



****

****

****

****

**Envolventes ADSR en MatLab**

L**os sonidos de instrumentos reales cambian en el tiempo.** Por ejemplo una nota de piano tiene un volumen y un timbre que se desarrollan en el tiempo y que varían durante la producción de cada nota. Su volumen va disminuyendo y su ‘brillo’ inicial se va ‘opacando’ amortiguando progresivamente la presencia de los armónicos altos.

**Para sintetizar un sonido de manera realista** necesitamos una forma de controlar el volumen para que cambie en el tiempo y una forma de controlar la frecuencia de corte del filtro para que también cambie en el tiempo durante la duración de cada nota.

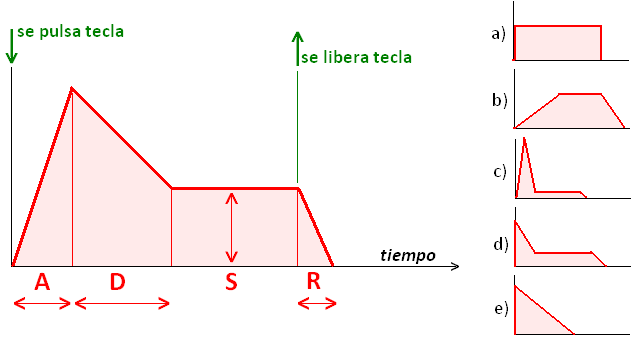
Un generador de envolvente ADSR crea ‘formas’ o ‘contornos’  cada vez que pulsamos una tecla repitiendo su forma, nos permite cambiar en el tiempo algún parámetro de la onda generando así sonidos que sean dinámicos y no estáticos.

En síntesis substractiva es normal generar el sonido a partir de un oscilador de ondas complejas que luego pasa por un filtro pasa bajos resonante.

El efecto de la envolvente sobre el sonido depende de donde la apliquemos. Si se aplica a:

* El control de ganancia en un amplificador (en un VCA) tenemos un cambio de volumen a lo largo del desarrollo de la nota.
* El control de la frecuencia de corte del filtro resonante (en el VCF) tenemos un cambio de brillo a lo largo del desarrollo de la nota.

**Envolvente ADSR aplicada a la amplitud**

Para generar una envolvente ADSR usamos los siguientes parámetros:

* A: Attack: Tiempo de ataque, el nivel va desde cero hasta el máximo. Momento en que se pulsa la tecla.
* D: Decay: Tiempo de Decaimiento, el nivel va desde el máximo a un valor medio que llamamos nivel de sostenido
* S: Sustain; Tiempo de Sostenido. Tiempo que se mantiene constante el nivel mientras permanece la tecla presionada.
* R: Release Tiempo de Relajación o liberación, el nivel va del nivel de sostenido a cero. Momento en que se suelta la tecla.

En la figura vemos algunas posibles ‘formas’ de ADSR que pueden obtenerse y su parecido con instrumentos musicales cuando se aplica a como envolvente de amplitud:

1. A y R cero (muy rápidos). Es muy básica, solo se puede aplicar para crear sonidos de órgano o para algunos leads simples.
2. A y R grandes(lentos). Podría ser adecuado para pads, cuerdas, fondos…
3. A y D pequeños (cortos). Crea un sonido percusivo. Ajustando los tiempos de ataque, decay y release, podemos animar las sucesivas repeticiones de la onda y dotarlas de variedad y riqueza.
4. A y D pequeños (cortos), S grande (largo). Sonidos tipo lead, es una mejora frente al tipo a.
5. A pequeño S cero R grande . Es una aproximación a sonidos como piano o guitarra pulsada.

function [env,Tt]=ADSR(At,Dt,Da,St,Rt,fs)

% calcula la envolvente ADSR dado:

% At= tiempo de ataque al valor 1 en ms

% Dt= tiempo de decaimiento al valor Da en ms

% Da= amplitud del decaimiento [0 1]

% St= tiempo de sustain en el valor Da en ms

% Rt= tiempo de release al valor 0 en ms

% fs= frec de muestreo

% todo a muestras

At=At\*fs/1000;

Dt=Dt\*fs/1000;

St=St\*fs/1000;

Rt=Rt\*fs/1000;

Tt=At+Dt+St+Rt;

% armando envolvente

env1=linspace(0,1,At);

env2=linspace(1,Da,Dt);

env3=Da\*ones(St,1)';

env4=linspace(Da,0,Rt);

env=[env1 env2 env3 env4];

end

La función llamada ADSR de MatLab que permite generar una envolvente ADSR que se guarda en el vector env y se calcula entregando lo indicado en la propia función:

At= tiempo de ataque al valor 1 en ms

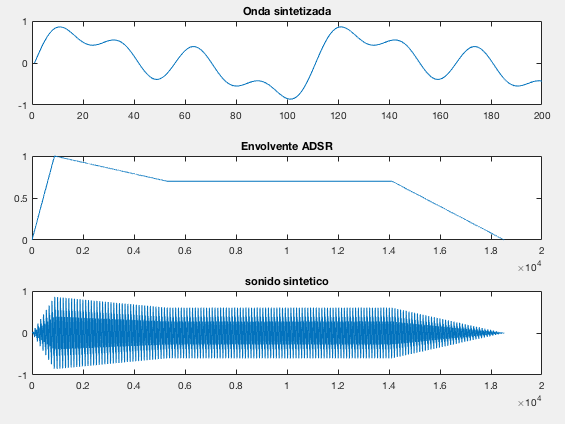
Dt= tiempo de decaimiento al valor Da en ms

Da= amplitud del decaimiento [0 1]

St= tiempo de sustain en el valor Da en ms

Rt= tiempo de release al valor 0 en ms

fs= frec de muestreo

Aquí tenemos un script que genera una onda por síntesis aditiva de tres senoidales , y luego hace uso de la función ADSR anteriormente mostrada generando en el vector out con una onda sintetizada con su envolvente.

% aditiva

clf,clc,clear

fs=44100;

n=0:1/fs:1;

s1=0.5\*sin(2\*pi\*400\*n);

s2=0.4\*sin(2\*pi\*800\*n);

s3=0.3\*sin(2\*pi\*1600\*n);

sr=s1+s2+s3;

subplot(311);plot(sr(1:200));title('Onda sintetizada')

[env,Tt]=ADSR(20,100,0.7,200,100,fs);

subplot(312);plot(env);title('Envolvente ADSR')

out=env.\*sr(1:Tt);

subplot(313);plot(out);title('sonido sintetico')

sound(out,fs)

**Tarea**

Dados los sonidos de la nota la3 de violín y un piano disponibles en

<https://todoelectroacustica.blogspot.cl/2018/05/diseno-de-filtros-digitales_17.html>

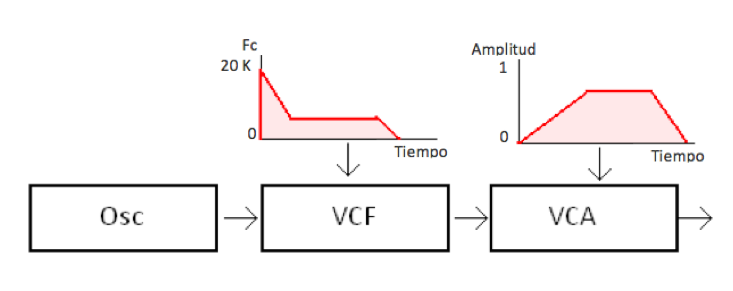
1. Cargue el sonido del violín en una variable monofónica y normalice la amplitud.
2. Visualice la forma de onda y el espectro.
3. Genere por síntesis aditiva una onda lo mas parecida posible a la del violín usando al menos 15 armónicos, obtenga sus frecuencias y amplitudes analizando el espectro.
4. Genere una envolvente ADSR de amplitud similar a un violín, analizando su envolvente de amplitud .
5. Sintetice el sonido del violín con su forma de onda y envolvente. Grafique el sonido real y el sintético en onda y espectro.
6. Grabe un archivo .wav con el sonido sintético del violín.
7. Repita los pasos anteriores para el sonido del piano.
8. Debe entregar los archivos .m y .wav para su evaluación en una carpeta con su nombre.

**Envolvente ADSR aplicada al filtro.**

Si usamos la envolvente para controlar la frecuencia de corte del pasa bajos resonante tendremos un cambio en la cantidad de armónicos del sonido, normalmente quitando agudos a medida que la nota evoluciona.

Sin embargo el hecho de quitar armónicos superiores repercute en una baja del nivel total del sonido, por lo que podemos decir que también afecta a la envolvente de amplitud.

Es recomendable hacer primero la etapa de filtrado para posteriormente proceder a modificar la amplitud considerando el efecto del filtrado.

Un sintetizador debería disponer entonces de dos generadores ADSR uno para el filtrado y otro para la envolvente. Uno controla el VCF y otro el VCA.

En la figura siguiente vemos dos envolventes distintas, una aplicada al VCF o otra al VCA.

La del VCF parte en una frecuencia de corte alta dejando pasar mucho brillo al inicio y luego haciendo mas opaco el sonido para hacerlo muy opaco al final.

La del VCA inicia suavemente luego se mantiene y finalmente se extingue.

Generar la envolvente de frecuencia de corte del filtro en Matlab es igual para la envolvente de amplitud, sin embargo lograr que el filtro responda de manera variable durante el desarrollo de la nota es todo un desafío de programación.

Abordaremos aquí una manera de hacerlo dividiendo el audio en pequeños fragmentos o Frames los que serán procesados cada uno con filtro de fc distinta ( según la envolvente deseada) para luego ser recompuestos en un solo archivo de audio filtrado.

Tenemos dos aproximaciones a este problema, una versión sin superposición de Frames y otra con superposición.

clc,clear,clf

%% crea forma de onda

fs=44100; dur=1; % duracion del sonido, o aprete de tecla

n=linspace(0,dur,fs)'; % crea vector de tiempos de 1 seg

x=sawtooth(2\*pi\*100\*n,0.5); % USER crea una onda triangular de 1 seg

L=length(x);

% crea ADSRfc, ADSRfc(At,Dt,Da,St,Rt,fs,fmax) At+Dt\*St+Rt=dur (en ms)

envfc=ADSRfc(100,200,0.05,300,400,fs,5000); %

% define frames y alarga audio y envolvente

F=128; % USER define tamaÒo del frame

N=ceil(L/F); % numero de frames aproximado al entero superior.

x=[x;zeros(N\*F-L,1)]; % alargando x para numero entero de frames

envfc=[envfc;zeros(N\*F-L,1)]; % alargando envf para numero entero de frames

% extrae frames

xf=[]; envfcfr\_ac=[];

for i=1:N-1

xfF=x((1+i\*F):(i+1)\*F); % extrae un frame de audio

envfcfr=envfc((1+i\*F):(i+1)\*F); % extrae un frame de envfc

fc=envfcfr(1);Wc =fc\*2/fs; % calcula la fc del frame

[b,a] = butter(4,Wc); % calculando un filtro segun ADSR

xfF=filter(b,a,xfF); % aplicando filtro al audio

xf=[xf;xfF]; % concatena y reconstruye x

envfcfr\_ac=[envfcfr\_ac ;envfcfr ]; % concatena y reconstruye env

end

sound(xf,fs)

subplot(211);plot(xf)

subplot(212);spectrogram(xf,256,128,256,fs,'yaxis');colorbar off

**Sin superposición de frames**

podemos usar un código como el mostrado, donde creamos una señal triangular de 1 segundo y una envolvente de frecuencia de 1 segundo usando la función propia ADSRfc.

Luego definimos el tamaño en samples de un Frame, ajustamos la duración del audio y entramos a un ciclo que extrae Frames de audio, crea y aplica un filtro a cada Frame para finalmente volver a unir cada Frame y generar el audio resultante.

Lamentablemente este código genera muchos clicks debido a que cada Frame simplemente se une con el anterior generando discontinuidades en la onda.

function [envf]=ADSRfc(At,Dt,Da,St,Rt,fs,fmax)

% calcula la envolvente ADSR de la fc para el filtro:

% debe ser de la misma logitud del audio a modificar.

% At= tiempo de ataque al valor 1 en ms

% Dt= tiempo de decaimiento al valor Da en ms

% Da= amplitud del decaimiento [0 1]

% St= tiempo de sustain en el valor Da en ms

% Rt= tiempo de release al valor 0 en ms

% fs= frec de muestreo

% fmax= frec maxima de la envolvente del filtro

% todo a muestras

At=At\*fs/1000;

Dt=Dt\*fs/1000;

St=St\*fs/1000;

Rt=Rt\*fs/1000;

% armando envolvente

env1=linspace(0,1,At);

env2=linspace(1,Da,Dt);

env3=Da\*ones(St,1)';

env4=linspace(Da,0,Rt);

envf=fmax\*[env1 env2 env3 env4]';

end

Cada rectángulo representa un Frame del audio.

Frame

Audio

**Con superposición y enventanado de Frames**

Para mejorar el código anterior debemos usar superposición de Frames con enventanado.

Frame

Audio

Si tomamos el caso anterior y superponemos cada Frame con el anterior una cierta cantidad de muestras, habrían zonas en que se sumarian las muestras

de Frames adyacentes lo que aumentaría el nivel de esas muestras y además puede seguir produciendo los clicks del caso anterior.

Para solucionar esta situación se usa el enventanado que consiste en

multiplicar cada Frame por una ventana de forma no rectangular,

Frame

sino que una con inicio y termino suave

La estrategia que usaremos en este código es usar ventanas tipo Hanning con un traslape o superposición de medio Frame, generando así las ventanas impares (rojas) y pares (azules).

Aquí tenemos el código ampliamente comentado, que presenta las siguientes etapas :

clc,clear,clf

%% crea forma de onda

fs=44100; dur=1; % duracion del sonido, o aprete de tecla

n=linspace(0,dur,fs)'; % crea vector de tiempos de 1 seg

x=sawtooth(2\*pi\*200\*n,0.5); % USER crea una onda triangular de 1 seg

L=length(x);

% crea ADSRfc, ADSRfc(At,Dt,Da,St,Rt,fs,fmax) At+Dt\*St+Rt=dur (en ms)

envfc=ADSRfc(100,200,0.05,300,600,fs,10000); %

% define frames y alarga audio y envolvente

F=512; % USER define tamaÒo del frame

N=ceil(L/F); % numero de frames aproximado al entero superior.

x=[x;zeros(N\*F-L,1)]; % alargando x para numero entero de frames

envfc=[envfc;zeros(N\*F-L,1)]; % alargando envf para numero entero de frames

% extrae frames

N=floor(2\*L/F-1); % numero de ventanas con traslape de F/2

H=hann(F); % crea ventana hanning de F samples

%

yi=[];%ventanas impares

for i=1:(N+1)/2

y=x(((i-1)\*F+1):i\*F); % extrae un frame de audio

envF=envfc((1+i\*F):(i+1)\*F); % extrae un frame de envfc

fc=envF(1);Wc =fc\*2/fs; % calcula la fc del frame

[b,a] = butter(4,Wc); % calculando un filtro segun ADSR

y=filter(b,a,y); % aplicando filtro al audio

y=y.\*H; % concatena y reconstruye audio

yi=[yi;y];

end

yp=[];%ventanas pares

for i=1:(N-1)/2

y=x(((i-1)\*F+F/2+1):(i\*F)+F/2);

envF=envfc((1+i\*F):(i+1)\*F); % extrae un frame de envfc

fc=envF(1);Wc =fc\*2/fs; % calcula la fc del frame

[b,a] = butter(4,Wc); % calculando un filtro segun ADSR

y=filter(b,a,y); % aplicando filtro al audio

y=y.\*H;

yp=[yp;y];

end

yp=[zeros(F/2,1); yp;zeros(F/2,1)];

xf=(yi+yp);

%

sound(xf,fs)

subplot(311);plot(xf)

subplot(312);spectrogram(xf,256,128,128,fs,'yaxis');colorbar off

subplot(313);plotfft(xf,fs,2)

Crea una forma de onda triangular de un segundo.

Crea una envolvente de frecuencia de corte de un segundo.

Define un tamaño de Frame

Crea una ventana tipo Hanning para aplicar a cada Frame.

Extrae Frames impares.

Crea y aplica filtro

Extrae Frames pares.

Crea y aplica filtro

Reconstruye el audio desde los Frames sumando los Frames impares y pares.

Podemos generar una función que haga el enventanado y filtrado mostrado , la que después puede ser llamada cómodamente desde otras aplicaciones.

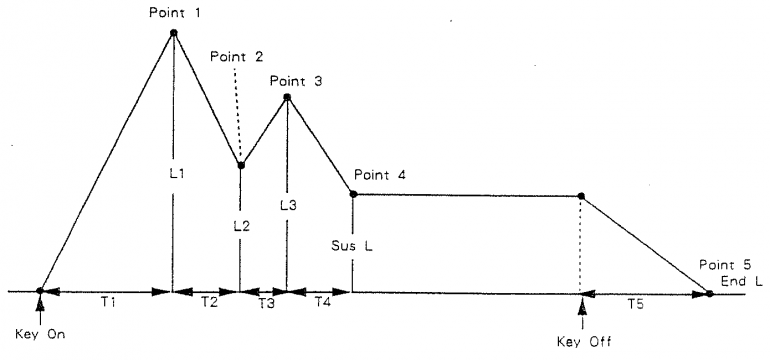
Una forma de hacer mas realistas nuestros sonidos sintéticos es a**gregarles los ‘ruidos de ataque’ o transitorios** que producen diferentes instrumentos, como por ejemplo:

* El golpe del martillo del piano,
* El contacto inicial del arco en el violín
* El contacto inicial de la uñeta en la guitarra,
* El inicio de un soplido en un tubo de zampoña
* El inicio de un soplido en la boquilla de una traversa
* etc…

Sonidos que pueden ser grabados de instrumentos reales, para quedarnos solo con el ataque

Muestras cortas, pero capaces de albergar la peculiaridad, altamente significativa en un plano perceptual, de esos primeros milisegundos inestables de diferentes familias de sonidos.

UN ADSR multi segmento



Filter ADSR

