Dispense di elaborazione analogica e numerica del segnale sonoro per la musica elettronica.

Oscillatori numerici

Molti modi di generare suoni periodici nei sistemi a tempo campionato.

Indice generale

1 Note sul copyright

2 Introduzione

Una delle funzioni elementari, basilari, sia per la sintesi di suoni, sia per l'analisi, è costituita dagli *oscillatori,* in grado di produrre suoni periodici di caratteristiche desiderate, cioè *forme d'onda periodiche*[1]. Nei sistemi numerici gli oscillatori prendono la forma di *algoritmi*.

In questo testo prenderemo in considerazione due tipi di algoritmi: gli oscillatori sinusoidali, e gli oscillatori a forma d'onda arbitraria "a tabella".

Tra gli oscillatori sinusoidali, verranno presi in esame quelli complessi, capaci di implementare un *fasore*, e quelli reali.

Indipendentemente dal tipo, ogni oscillatore deve obbligatoriamente avere almeno una variabile d'ingresso in frequenza, e talvolta anche in fase. Per quanto sia diffuso trovare oscillatori che ammettono come ingresso anche l'ampiezza dell'oscillazione, questo può essere assente senza costituire un problema: è sempre possibile moltiplicare l'uscita per un guadagno fisso, oppure variabile quando si voglia operare una modulazione di ampiezza.

Nel seguito, per semplicità di trattazione, tutti gli oscillatori produrranno un segnale di ampiezza unitaria, lasciando ad una operazione come quella appena descritta il compito di modificarla.

Infine, si prenderanno in esame i *banchi di oscillatori,* in grado di produrre pettini armonici a partire da un solo oscillatore sinusoidale, con un minore numero di calcoli rispetto all'equivalente numero di oscillatori indipendenti.

Si presuppongono note le proprietà fondamentali dei sistemi a tempo discreto, o campionati nel tempo, in particolare il teorema di Nyquist e la problematica legata alle *frequenze fantasma*, ovvero lo *aliasing* o *foldover*.

Nel testo non si affronteranno invece gli oscillatori analogici - argomento interessante ma piuttosto complesso – che non soffrono della problematiche appena citate.

1 Dotate pertanto di un *pitch* e quindi intonate. Gli oscillatori generano dunque delle *note*.

3 Oscillatori sinusoidali a frequenza fissa e variabile

Si definisce "oscillatore sinusoidale" un algoritmo che implementa la seguente funzione:

$$output(t)=e^{i\varphi(t)+\phi} \qquad \text{1 - oscillatore complesso}$$

oppure:

$$output(t)=\sin(\varphi(t)+\phi) \qquad \text{2 - oscillatore reale}$$

Ricordiamo le relazioni tra fase e pulsazione (e frequenza):

$$\omega(t)=\frac{d\varphi}{dt} \text{ ovvero } \varphi(t)=\int_0^t \omega(\tau)\,d\tau \text{ con } \omega=2\pi\nu$$

dove ω è la pulsazione (in radianti al secondo) e ν è la frequenza (in Hz).

Un oscillatore che debba operare a frequenza variabile deve dunque implementare correttamente l'espressione:

$$output(t)=e^{i(\int_0^t \omega(\tau)d\tau)+\phi} \text{ oppure } output(t)=\sin(\int_0^t \omega(\tau)\,d\tau+\phi) \qquad 3$$

Il fatto che un oscillatore operi eseguendo *l'integrale* della pulsazione è assolutamente essenziale. Un oscillatore che implementasse le seguenti formule:

$$output(t)=e^{i\omega(t)t+\phi} \text{ oppure } output(t)=\sin(\omega(t)t+\phi) \qquad 4$$

non garantirebbe *continuità di fase* in caso di variazioni di frequenza, e sarebbe quindi in grado di operare correttamente solo *a frequenza costante*[2].

A quali inconvenienti si possa andare incontro variando la frequenza direttamente è facile immaginare notando che, ad esempio, se la pulsazione sta diminuendo, la fase ad un determinato istante $\omega(t_n)\cdot t_n$ può risultare più piccola di quella dell'istante precedente $\omega(t_{n-1})\cdot t_{n-1}$, ottenendo una rotazione *in senso orario,* cioè inversa a quella che si otterrebbe per una diminuzione inferiore o per un aumento della pulsazione.

E' in effetti possibile implementare oscillatori come in 4, ad esempio utilizzando esplicitamente le funzioni trigonometriche (o esponenziali complesse) basandosi su di una variabile "tempo" progressiva, ottenendo un oscillatore a frequenza fissa, oppure a frequenza molto lentamente variabile[3]. Calcolando invece previamente la fase come integrale della frequenza (come indicato in 3) si otterrebbe invece un oscillatore a frequenza variabile.

---

2 A frequenza costante infatti non c'è differenza tra le due espressioni: $\int_0^t \omega\tau\,d\tau=\omega t$

3 In tal caso un errore di continuità di fase sarebbe sempre presente, ma sarebbe piccolo.

In altre parole, la seguente procedura di calcolo non garantisce continuità di fase:

$$s_n = \sin(2\pi\nu_n \cdot t_n) \quad \text{oppure} \quad s_n = e^{i2\pi\nu_n \cdot t_n}$$

La seguente procedura garantisce invece continuità:

$$\varphi_n = \varphi_{n-1} + 2\pi\nu_n \Delta t \;\rightarrow\; s_n = \sin(\varphi_n) \quad \text{oppure} \quad s_n = e^{i\varphi_n}$$

dove ν_n è una frequenza variabile nel tempo

L'operazione a sinistra prende il nome di *integrazione della frequenza*.

Le due figure seguenti illustrano con un esempio le differenze tra le due implementazioni nel caso di un oscillatore reale (come in 2 e nella parte destra di 4).

Fig. 1 - Discontinuità di fase dovuta a variazione brusca di frequenza
(salto brusco di frequenza da 20 a 5 Hz)

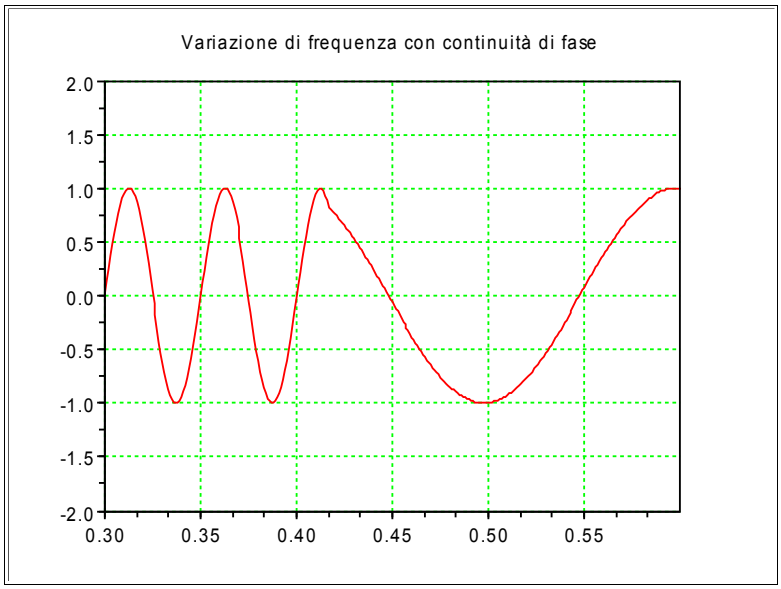


Fig. 2 - Assenza di discontinuità di fase nelle medesime condizioni (salto brusco di frequenza da 20 a 5 Hz).

Le due figure illustrano come queste considerazioni siano della massima importanza e debbano essere tenute sempre presenti, pena gravi errori di implementazione, *con conseguenze udibili*.

Possiamo riassumere quanto analizzato in un enunciato che è utile tenere a mente:

> ***La fase di un oscillatore è uguale al prodotto della frequenza per il tempo solo se la frequenza è costante.***

4 Oscillatore complesso, o "in quadratura"

Il più semplice oscillatore sinusoidale è derivabile direttamente dalla nozione di fasore: si tratta semplicemente di implementare un punto che gira attorno all'origine di un piano complesso, generando una coppia seno-coseno. I due segnali sono "in quadratura", secondo il gergo delle telecomunicazioni, in quanto sfasati di 90° l'uno dall'altro. Un oscillatore del genere prende – per questa ragione - anche il nome di "oscillatore in quadratura". Vedremo dunque come non vi sia bisogno del calcolo esplicito della funzione seno per ottenere perfette sinusoidi, ma siano sufficienti le moltiplicazioni e le somme (con segno).

Un fasore è un segnale complesso, analitico, siffatto[4]:

$$output(t) = e^{i(\varphi(t)+\phi)} \qquad 5$$

dove φ è la fase, in generale funzione del tempo (variabile nel tempo, *time-variant*, o tempo-variante).

4 Vedi: Lorenzo Seno, *Richiamo di algebra e matematica, e loro nessi con la musica*. Dispense 2006.

Esso dà luogo ad una coppia di segnali, costituiti dalla parte reale e da quella immaginaria:

$$output(t)=\cos(\varphi(t))+i\sin(\varphi(t)) \qquad 6$$

La derivata temporale (variazione nell'unità di tempo) della fase è la *frequenza,* che si misura in Hz, ovvero "giri al secondo". In genere, per la frequenza si usa la lettera greca "ni", ν, (ma pronunciata da fisici e ingegneri come "nu"). Un altro modo per esprimere la stessa grandezza è la pulsazione, indicata generalmente con ω, misurata nelle unità trigonometriche "radianti al secondo" (dove 2π radianti corrispondono ad un giro).

In un sistema a tempo discreto, il problema di implementare un fasore si riduce a quello di calcolare il nuovo punto a partire dalla conoscenza di quello all'istante precedente, e della frequenza e fase istantanei (competenti cioè all'istante di campionamento corrente).

4.1 Oscillatore complesso a frequenza costante

Per semplicità di trattazione inizieremo considerando il caso in cui la frequenza e la fase siano costanti, e la fase iniziale sia zero.

All'istante 0 la fase è zero, e quindi:

$$s_0=e^{i\cdot 0} \quad ovvero \quad s_0=1+i\cdot 0 \qquad 7$$

All'istante successivo il fasore deve ruotare di un angolo φ tale che, applicando tale rotazione ad ogni istante di campionamento, il fasore ruoti a ν giri/sec, ovvero ad ω rad/sec. Se ν_s è la frequenza di campionamento, abbiamo che $T=1/\nu_s$ è il periodo di campionamento, e in un secondo il fasore deve fare ν giri.

In un secondo vi sono $N=1/T$ periodi di campionamento. Quindi:

$$N\cdot\varphi=\omega \quad \text{e dunque} \quad \nu_s\cdot\varphi=\omega$$

Questo conduce a:

$$\varphi=\frac{\omega}{\nu_s} \quad \text{ovvero} \quad \varphi=2\pi\frac{\nu}{\nu_s} \qquad 8$$

Il rapporto all'estrema destra, tra frequenza e frequenza di campionamento, prende anche il nome di *frequenza numerica*, che indichiamo con *f*. $0\le f<0.5$, stante che per il teorema del campionamento in un sistema campionato nel tempo non si possono esprimere frequenze superiori a quella di Nyquist, pari alla metà di quella di campionamento.

$$\varphi=2\pi f$$

Dunque all'istante successivo il nuovo segnale sarà:

$$s_1 = s_0 \cdot e^{i\varphi}$$

E in generale sarà: $s_n = s_{n-1} \cdot e^{i\varphi}$ ovvero $s_n = s_{n-1} \cdot (\cos(\varphi) + i \sin(\varphi))$

Naturalmente il prodotto qui indicato è un prodotto complesso.

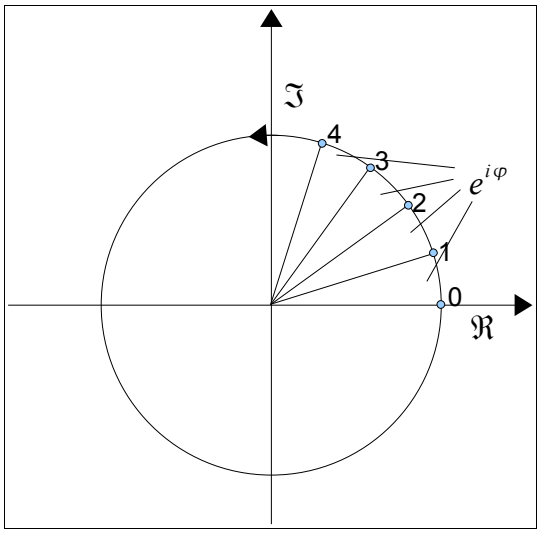


Fig. 3 - Oscillatore complesso: rotazioni successive.

Un oscillatore del genere richiede quindi:

1. Il calcolo una volta per tutte delle due quantità

$$C = \cos(\varphi) \quad e \quad S = \sin(\varphi) \quad con \quad \varphi = 2\pi f \qquad 9$$

2. Ad ogni istante la memorizzazione del valore del fasore come numero complesso, oppure come coppia di reali: $(\Re_n, \Im_n)$
3. Ad ogni istante il calcolo del nuovo valore del fasore a partire dal valore precedente memorizzato e dai valori di *C* e *S* (che esprimono la frequenza):

$$s_n = (\Re_n + i\Im_n) = (\Re_{n-1} + i\Im_{n-1}) \cdot (C + iS)$$

Sviluppando il prodotto complesso abbiamo:

$$(\Re_n + i\Im_n) = (\Re_{n-1} C - \Im_{n-1} S) + i(\Re_{n-1} S + \Im_{n-1} C) \qquad 10$$

E quindi, in termini di coppie reali (da mettere in uscita e memorizzare per il calcolo successivo):

$$\begin{aligned} \Re_n &= (\Re_{n-1} C - \Im_{n-1} S) \\ \Im_n &= (\Re_{n-1} S + \Im_{n-1} C) \end{aligned} \qquad 11$$

per un totale di due somme e quattro moltiplicazioni ad ogni istante di campionamento.

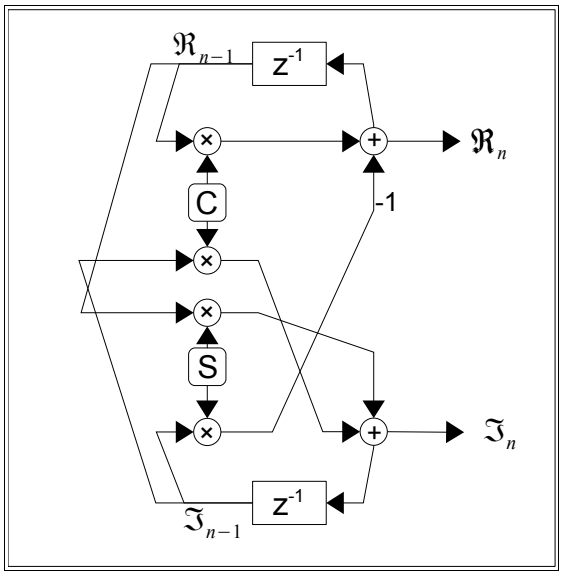


Fig. 4 - *Patch* che realizza la 11.
I valori iniziali (che si trovano a sinistra) devono essere correttamente inizializzati: ad esempio: $\Re_0=1 \quad \Im_0=0$

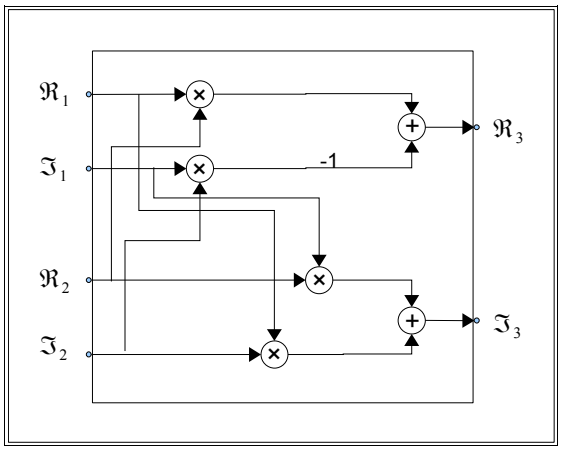


Fig. 5 - *Patch* che realizza il prodotto complesso $\Re_3+i\,\Im_3=(\Re_1+i\,\Im_1)\cdot(\Re_2+i\,\Im_2)$

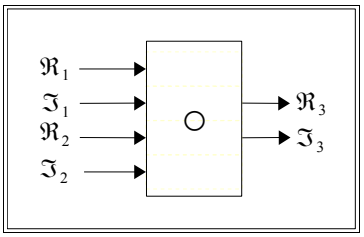


Fig. 6 -Simbolo per il prodotto complesso

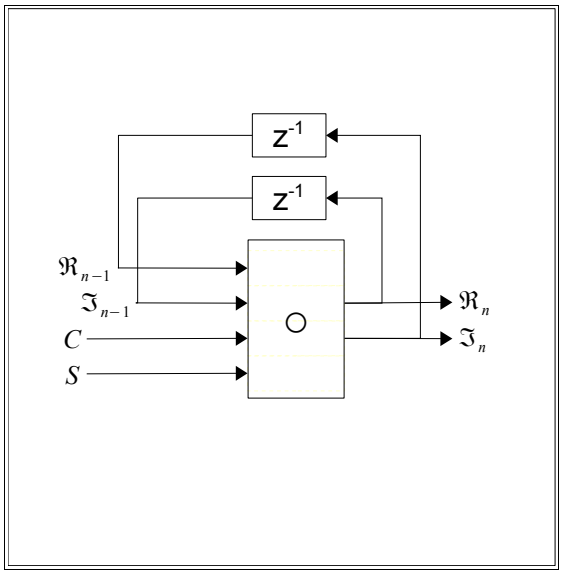


Fig. 7 - *Patch* che realizza sempre la 11, molto più concisa grazie alla simbologia complessa.

4.2 Oscillatore complesso a frequenza variabile

Non è difficile a questo punto verificare che così operando si integra implicitamente la fase, operando quindi in continuità di fase[5]. E' quindi immediato generalizzare la formula al caso in cui sia frequenza sia fase siano variabili nel tempo, avendo cioè una sequenza di frequenze e fasi:

$$f_0, f_1, \ldots, f_n \qquad \phi_0, \phi_1, \ldots, \phi_n$$

La rotazione che istante per istante si deve applicare è:

$$e^{i(\varphi_n+(\phi_n-\phi_{n-1}))} = e^{i(\varphi_n+\Delta\phi_n)} \qquad 12$$

5 Questo deriva direttamente dal fatto che l'operazione di rotazione del fasore è di fatto una integrazione (si opera per differenza rispetto alla fase precedente), e garantisce quindi che la rotazione avvenga comunque in senso orario per frequenze positive, indipendentemente dall'entità e dal segno della variazione della pulsazione.

Ma questo equivale a calcolare i *C* e *S* nel seguente modo:

$$\begin{aligned} C &= \cos(2\pi f_n + \Delta\phi_n) \\ S &= \sin(2\pi f_n + \Delta\phi_n) \\ \Delta\phi_n &= (\phi_n - \phi_{n-1}) \end{aligned} \qquad 13$$

Naturalmente, nel caso in cui frequenza o fase siano variabili nel tempo non si può più asserire che i *C* ed *S* possono essere calcolati una volta per tutte, e dato che il loro calcolo, come appare chiaro dalla 13, richiede il calcolo di un coseno e di un seno, il vantaggio di operare nel modo anzidetto sembra vanificarsi: tanto varrebbe calcolare direttamente il coseno e il seno relativo alla nuova posizione, in modo esplicito, integrando sulla frequenza in modo numerico.

Nel caso in cui non si debba però variare la fase si possono però memorizzare i valori delle coppie (C, S) per un elevato numero di frequenze *f*. Ad esempio, volendo lavorare in Hz, la memorizzazione di una tabella di coppie di valori ogni 1/10 Hz da 0 a 20 KHz comporta 400.000 valori floating, un numero che può essere considerato piccolo rispetto alle dimensioni attuali delle memorie. Operare in Hz non è però sempre razionale in campo musicale, perché comporta un eccesso di dettaglio alle alte frequenze[6]. Lavorando in *cent*[7] (risoluzione che anche i microtonali considerano sufficiente), si hanno 11.959 cent tra 20Hz e 20 KHz, e quindi memorizzare le coppie (C, S) per ogni cent significa memorizzare 23.916 costanti *floating point*, un valore che può essere considerato piccolo.

Si deve notare che una tabella del genere (nella quale si entrerebbe con il valore in cent, per ottenere i corrispondenti *C* ed *S)*, è una tabella *universale*, una volta fissata la frequenza di campionamento, ed è in grado di servire un numero qualsivoglia di oscillatori indipendenti.

La complessità di calcolo di un oscillatore del genere, a fase costante, torna quindi a ridursi a quattro moltiplicazioni, due somme, e due accessi ad una tabella di costanti floating.

4.3 Oscillatore complesso a fase variabile

Anche il caso a fase variabile (modulata) può essere risolto in modo tabellare. Si osservi la 12:

$$e^{i(\varphi+\Delta\phi)} = e^{i\varphi}\, e^{i\Delta\phi}$$

Il problema può quindi essere scomposto in due prodotti complessi:

$$e^{i\varphi} e^{i\Delta\phi} = (\cos(\varphi) + i\sin(\varphi)) \cdot (\cos(\Delta\phi) + i\sin(\Delta\phi)) = (C + iS) \cdot (C_\Delta + i S_\Delta) \qquad 14$$

I valori C_Δ e S_Δ possono anch'essi essere tabulati (in scala lineare, cioè in

6 Il nostro sistema uditivo è sensibile agli *intervalli* musicali. Dunque ad esempio la risoluzione di 1 Hz è certamente insufficiente a 20 Hz (dove corrisponde a circa un tono) ma è esageratamente fine a 10 KHz, dove corrisponde a meno di due millesimi di semitono.

7 Un *cent* è un centesimo di semitono equabile, ovvero la 1200ª parte di un'ottava.

Hertz). La tabulazione di questi valori presenta ancora meno problemi delle frequenze, perché in genere si tratta di valori piccoli. Questa problematica è legata alla modulazione di frequenza, quando si desideri operare sulle formule dirette utilizzando l'ingresso in fase. I valori massimi di deviazione di frequenza sono in questo caso definiti dall'indice di modulazione[8]. In tal caso le frequenze dovranno essere tabulate in scala lineare (Hz).

Dalla 14 si ottengono le formule per l'oscillatore con frequenza e fase variabili.

Dalla: $(C+iS)\cdot(C_\Delta+iS_\Delta)=(CC_\Delta-SS_\Delta)+i(CS_\Delta+SC_\Delta)$

abbiamo:
$$\begin{aligned}\Re_n&=(\Re_{n-1}(CC_\Delta-SS_\Delta)-\Im_{n-1}(CS_\Delta+SC_\Delta))\\ \Im_n&=(\Re_{n-1}(CS_\Delta+SC_\Delta)+\Im_{n-1}(CC_\Delta-SS_\Delta))\end{aligned} \qquad (15)$$

L'algoritmo richiede ad ogni variazione di frequenza o fase 8 moltiplicazioni e 4 somme[9].

4.4 Continuità di fase e costanza di ampiezza

Operando nel modo anzidetto per variare frequenza e fase dell'oscillatore sono garantite continuità di fase e, in linea di principio, costanza della ampiezza, dato che essa è l'ipotenusa di un seno e di un coseno.

8 Tutta la materia troverà una più specifica trattazione nel testo relativo alla modulazione di frequenza.

9 Nel calcolo si tenga conto della simmetria della formula, che non richiede la ripetizione dei calcoli tra parentesi tonde interne tra prima e seconda riga.

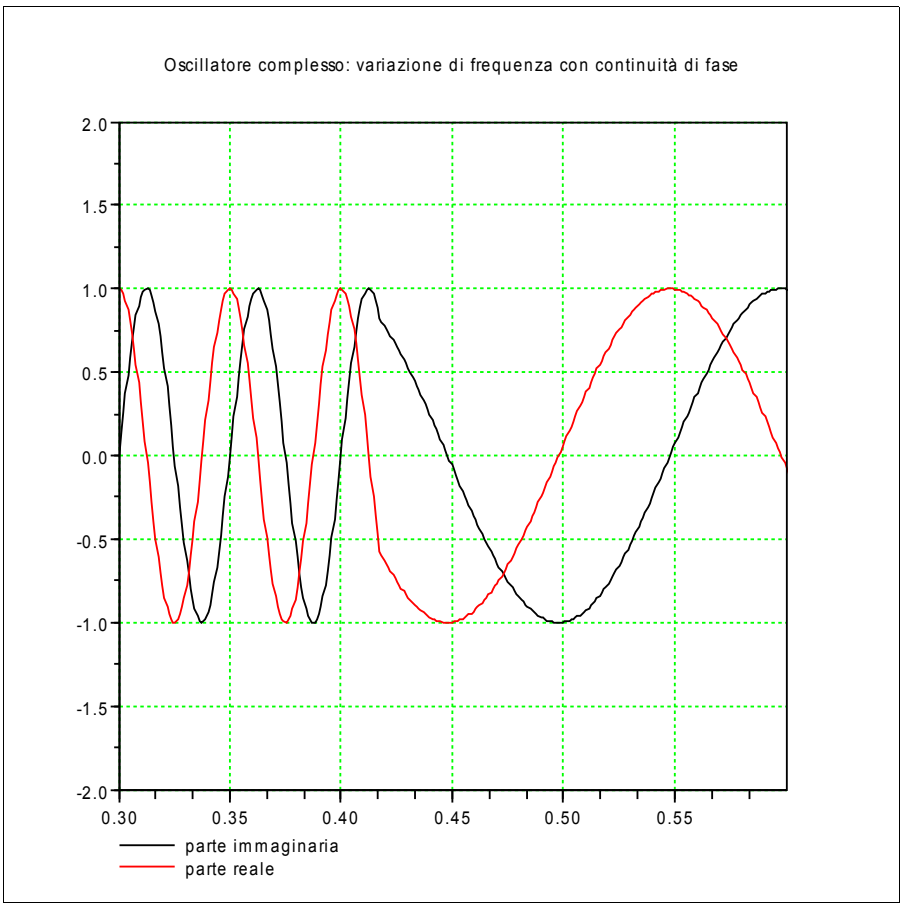


Fig. 8 - Continuità di fase nell'oscillatore complesso per brusca variazione da 20 a 5 Hz.

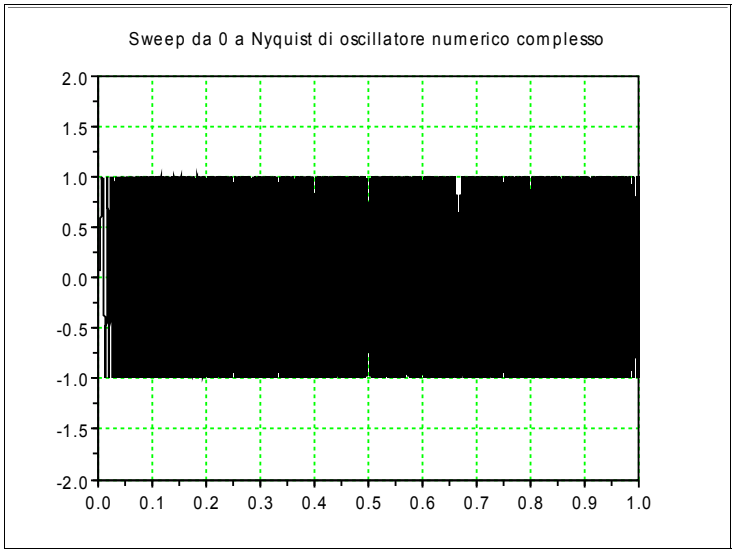


Fig. 9 - Lo *sweep* mostra costanza dell'ampiezza.

4.5 Il problema della rinormalizzazione.

Gli oscillatori qui descritti soffrono di un inconveniente (al quale è però facile porre rimedio con un dispendio di calcolo trascurabile) legato alla approssimazione dei calcoli dovuta al troncamento ad un numero finito di decimali.

Questi oscillatori, come detto, eseguono ad ogni istante di campionamento una rotazione di un fasore, moltiplicando il vecchio fasore per uno che esprime la rotazione desiderata:

$$e^{i\varphi_n} = e^{i\varphi_{n-1}} \cdot e^{i\Delta\varphi}$$

Si noti che con il passare del tempo questo equivale ad applicare *n* volte la rotazione alla fase iniziale all'istante 0, se la frequenza è costante:

$$e^{i\varphi_n} = e^{i\varphi_0} \cdot e^{in\Delta\varphi} = e^{i\varphi_0} \cdot e^{i\Delta\varphi} \cdot e^{i\Delta\varphi} \cdot \ldots \cdot e^{i\Delta\varphi} = e^{i\varphi_0} \cdot \left(e^{i\Delta\varphi}\right)^n \qquad 16$$

Si noti ancora che tutti i termini a destra hanno modulo unitario, e quindi tale è il modulo del termine corrente a sinistra: c'è un implicito $\rho = 1$ dappertutto, a moltiplicare tutti i termini. Il modulo risultante resta costantemente = 1 perché $1^n = 1$ qualunque sia *n*.

Ora supponiamo che la nostra rotazione non sia a modulo esattamente unitario, ma sia:

$$\rho\, e^{i\Delta\varphi}$$

con $\rho \approx 1$ ma (molto lievemente) diverso da 1.

Il nostro termine risultante a sinistra della 16 diventerebbe:

$$\rho^n e^{i\rho_n}$$

un termine cioè con modulo lievemente crescente o lievemente decrescente con *n*, a seconda se ρ sia rispettivamente maggiore o minore di 1.

In altre parole il nostro fasore risultante, anziché ruotare nel cerchio di raggio unitario, *spiralizzerebbe* in modo crescente o decrescente.

Questo è quello che provoca l'errore di arrotondamento nei calcoli. La spiralizzazione è di norma decrescente (dato il tipo di arrotondamento in uso nelle unità *floating-point*), ma è molto lenta: con una aritmetica a 32 bit ci vogliono ore perché il fenomeno diventi avvertibile. Con una ad 80 bit[10] il fenomeno non ha rilevanza pratica.

Dato che però anche nelle CPU moderne quali i Pentium può risultare conveniente utilizzare una aritmetica a 32 bit[11], è bene porre rimedio all'inconveniente.

Per eliminare l'inconveniente basta *rinormalizzare* il valore corrente. Questo non ha un modulo esattamente unitario, per quanto detto, e cumula un errore di modulo con l'andare del tempo.

Questo significa che:

---

10 Quale è quella delle FPU IEEE delle CPU della classe Pentium.

11 Ad esempio, per utilizzare l'unità di calcolo vettoriale SSE, che permette di eseguire 4 calcoli in parallelo.

$$\Re_n^2+\Im_n^2\neq 1$$

Basta però ricalcolare ad esempio la parte immaginaria in modo da avere modulo unitario, e sostituire questo nuovo valore corretto a quello originale, per evitare l'accumulo dell'errore:

$$\Im_n=\sqrt{(1-\Re_n^2)}$$

Si tratta di un calcolo che comporta anch'esso un errore, ma trattandosi di un calcolo *non iterativo*, l'errore resta costante (oltre che minuscolo) e non si accumula.

L'operazione comporta un quadrato (una moltiplicazione), una somma e una radice quadrata, operazione quest'ultima piuttosto dispendiosa in termini di calcolo. Tuttavia la lentezza del fenomeno non richiede che la normalizzazione sia eseguita ad ogni istante di campionamento: si può procedere con tempi dell'ordine dei secondi o anche dei minuti, quindi ogni centinaia di migliaia (o milioni) di campioni. Il costo computazionale associato alla normalizzazione diventa quindi percentualmente irrisorio e del tutto trascurabile.

5 Oscillatori reali

Esiste una possibilità analoga all'oscillatore complesso, che fa però uso della sola rappresentazione reale. Questo "oscillatore reale" è talvolta pomposamente chiamato "oscillatore del 2°ordine", perché fa uso della memorizzazione di due valori precedenti all'istante corrente e può essere visto come soluzione dell'equazione che descrive l'oscillatore del 2° ordine. In realtà esso può essere derivato direttamente dalle formule di duplicazione della trigonometria. Riportiamo qui per completezza la derivazione della formula iterativa.

Un oscillatore (supponiamolo per comodità cosinusoidale) a frequenza numerica f, e pulsazione numerica $\varphi=2\pi f$ con fase iniziale ϕ, è una sequenza del genere:

$$s_0=\cos(\phi),\quad s_1=\cos(\varphi+\phi),\quad s_2=\cos(2\varphi+\phi),\quad \ldots\quad s_n=\cos(n\varphi+\phi)$$

In generale, possiamo esprimere il termine generico in funzione dei due precedenti:

$s_n=\cos(2\varphi+\phi_{n-2})$ dove ϕ_{n-2} è la fase di due istanti precedenti, ovvero:

$$s_{n-1}=\cos(\varphi+\phi_{n-2}) \qquad\qquad s_{n-2}=\cos(\phi_{n-2})$$

Il termine generico è dunque il coseno di una somma:

$$s_n=\cos(2\varphi+\phi_{n-2})=\cos(2\varphi)\cos(\phi_{n-2})-\sin(2\varphi)\sin(\phi_{n-2})$$

Quello precedente è:

$$s_{n-1}=\cos(\varphi+\phi_{n-2})=\cos(\varphi)\cos(\phi_{n-2})-\sin(\varphi)\sin(\phi_{n-2})$$

Dalle formule di duplicazione applicate al termine $\cos(2\varphi)$:

$$\cos(2\varphi)=2\cos(\varphi)^2-1 \quad \sin(2\varphi)=2\sin(\varphi)\cos(\varphi)$$

$$s_n=\cos(2\varphi+\phi_{n-2})=(2\cos(\varphi)^2-1)\cos(\phi_{n-2})-2\sin(\varphi)\cos(\varphi)\sin(\phi_{n-2})$$

Riorganizzando la formula abbiamo:

$$s_n=\cos(2\varphi+\phi_{n-2})=2\{\cos(\varphi)[\cos(\varphi)\cos(\phi_{n-2})-\sin(\varphi)\sin(\phi_{n-2})]\}-\cos(\phi_{n-2})$$

Il termine tra parentesi quadre è pari a s_{n-1} :

$$s_n=\cos(2\varphi+\phi_{n-2})=2(\cos(\varphi)\,s_{n-1})-\cos(\phi_{n-2})$$

L'ultimo termine a destra è pari a s_{n-2} :

$$s_n=\cos(2\varphi+\phi_{n-2})=2\cos(\varphi)s_{n-1}-s_{n-2} \qquad 17$$

La 17 ci fornisce la formula iterativa per l'oscillatore. Chiamando $C=\cos(\varphi)$ il termine di frequenza, calcolabile (e tabulabile) una volta per tutte, abbiamo:

$$s_n=2C\,s_{n-1}-s_{n-2} \qquad 18$$

L'algoritmo deve essere inizializzato con due valori iniziali:

$$s_0=\cos(\phi) \qquad s_1=\cos(\varphi+\phi) \qquad 19$$

dove ϕ è la fase iniziale.

Per $n = 2,3, \ldots \infty$ è a questo punto utilizzabile la 18.

Si noti che la formula produce una cosinusoide. La circostanza è però irrilevante, dato che possiamo scegliere qualunque fase iniziale: ad esempio pari a $\pi/2$, ottenendo così una sinusoide.

5.1 Continuità di fase, costanza di ampiezza, e altre caratteristiche

Tra questo oscillatore e quello complesso esistono differenze notevoli che derivano dal differente meccanismo di memorizzazione della fase (un solo valore nel caso complesso, *due* istanti precedenti[12] in questo caso. Qui dunque la coerenza di fase deve essere mantenuta su due istanti oltre che su quello corrente.

12 In entrambi i casi si tratta di due valori reali: nel caso complesso, di un valore complesso riferito all'istante precedente, nel caso reale, di due valori reali riferiti a due istanti precedenti. Per procedere in modo iterativo, se non si vuole memorizzare esplicitamente la fase (cosa che implicherebbe il calcolo di una funzione trigonometrica per ottenere il segnale), ma limitarsi a valori precedenti dell'uscita, ci si scontra con l'ambiguità di quadrante delle funzioni periodiche e simmetriche quali sono quelle trigonometriche. L'oscillatore complesso individua il punto precedente nel cerchio goniometrico perché ne memorizza le due coordinate cartesiane. Quello reale memorizza la "storia" del segnale, e agisce nell'ipotesi che la fase sia sempre crescente. I due valori precedenti equispaziati permettono a questo punto di dirimere l'ambiguità di quadrante. La necessità quindi di memorizzare due valori in calcoli iterativi che non implichino la memorizzazione esplicita della fase è quindi *essenziale e inevitabile*, e legata alla periodicità e simmetria delle funzioni trigonometriche.

1. L'algoritmo dell'oscillatore reale non garantisce automaticamente continuità di fase e costanza di ampiezza per frequenze variabili nel tempo. Per ottenerla, è necessario reinizializzare i due valori “storici” in modo tale da garantire la continuità di fase con l'istante precedente. I calcoli necessari sono gravosi e implicano il calcolo di funzioni trascendenti.
2. Anche questo algoritmo, essendo iterativo, soffre dell'inconveniente della spiralizzazione. La rinormalizzazione richiede anch'essa la reimpostazione dei due valori “storici” con continuità di fase rispetto al passato, con le stesse conseguenze enunciante nel punto 1.

Queste caratteristiche limitano l'uso di questo oscillatore ai casi di frequenza costante o molto raramente variabile.

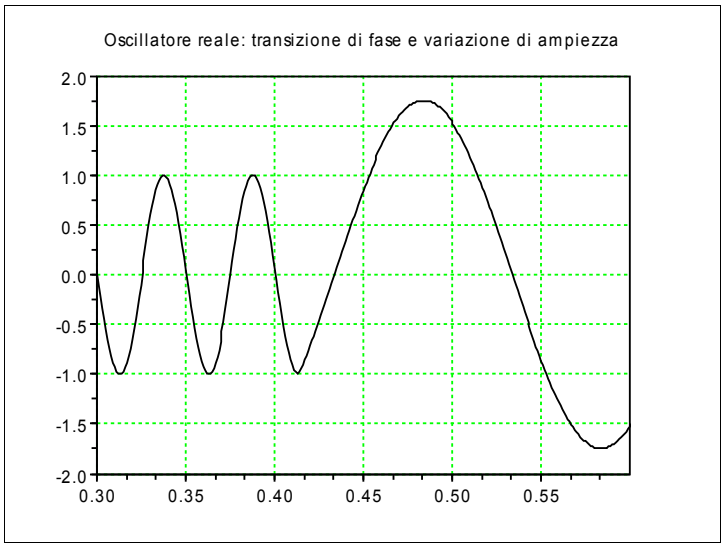


Fig. 10 - L'oscillatore reale mostra una transizione di fase irregolare e un cambiamento di ampiezza per una brusca variazione di frequenza da 20 a 5 Hz

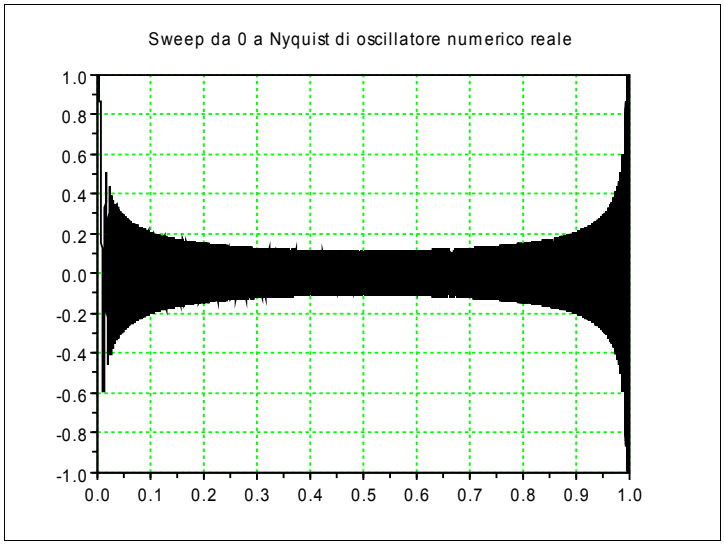


Fig. 11 - Andamento di ampiezza di uno *sweep* di un oscillatore reale.

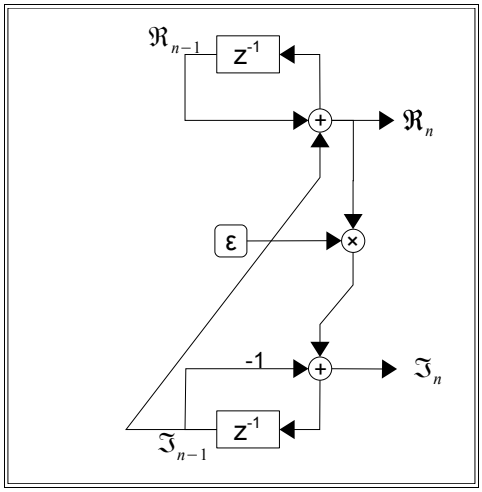
6 Varianti

6.1 Oscillatore a "cerchio magico"

Una "variante" dell'oscillatore complesso, detta "*Magic Circle*" fa uso di uno schema di calcolo lievemente differente, e non soffre della necessità di rinormalizzazione. In compenso non garantisce costanza di ampiezza per variazioni della frequenza.

Lo schema richiede sempre la memorizzazione e il calcolo di due valori in quadratura, ma utilizzando uno schema lievemente diverso:

$$\begin{aligned}\Re_n &= \Re_{n-1} + \epsilon\,\Im_{n-1} \\ \Im_n &= -\epsilon\,\Re_n + \Im_{n-1}\end{aligned} \quad \text{dove } \epsilon = 2\sin(\varphi/2)\ .$$



Patch per il Cerchio Magico. I valori iniziali devono essere inizializzati:
ad esempio $\Re_0 = 1 \quad \Im_0 = 0$

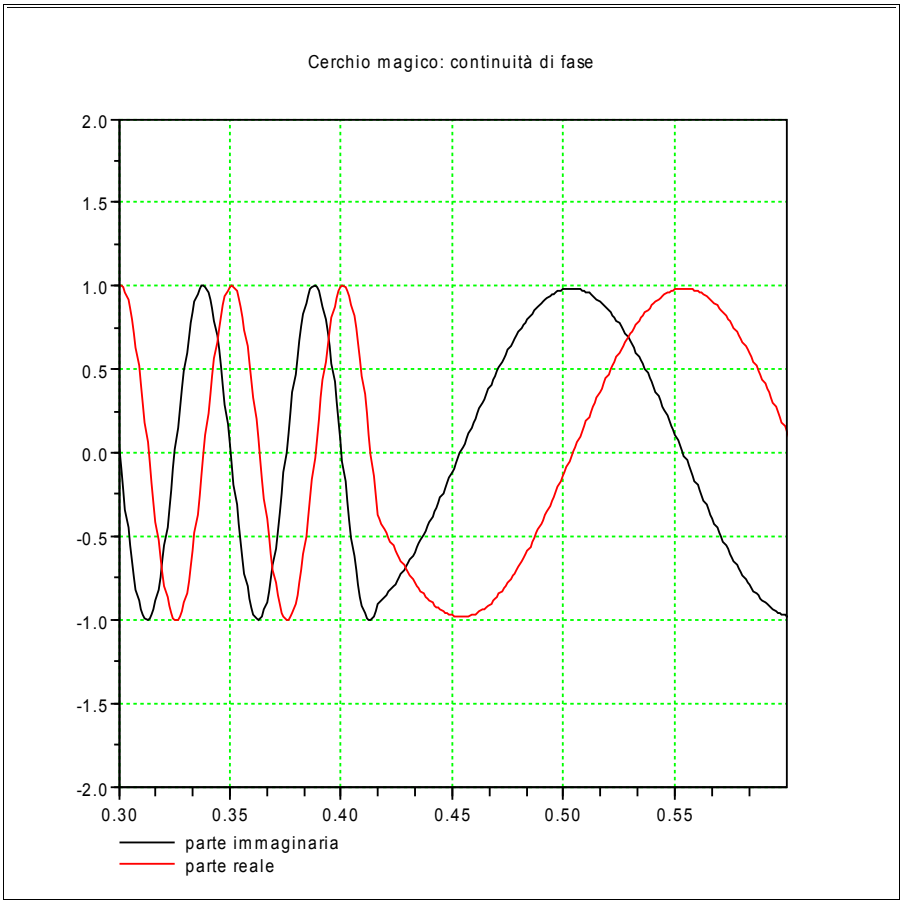


Fig. 12 - Transizione con continuità di fase per una brusca variazione di frequenza da 20 a 5 Hz.

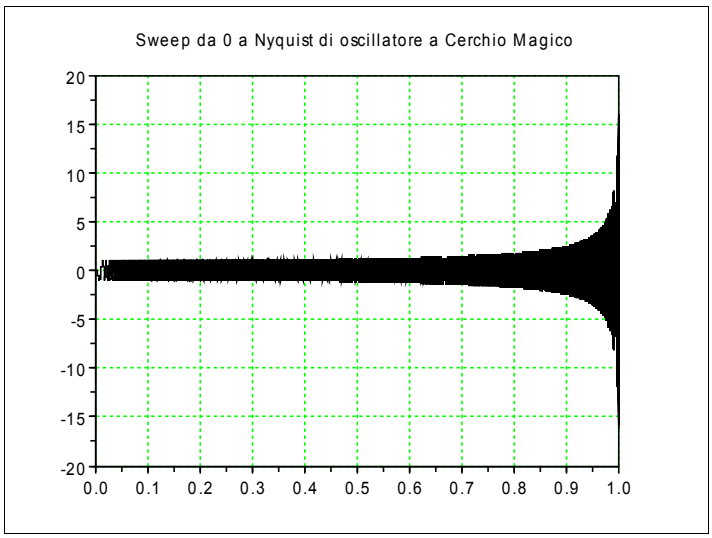


Fig. 1 - Lo *sweep* mostra una dipendenza dalla frequenza, soprattutto per frequenze alte.

Il cerchio magico ha continuità di fase, ma a differenza dell'oscillatore complesso, mostra una dipendenza dell'ampiezza dalla frequenza.

Le variazioni di ampiezza *non dipendono* però dalla *variazione* di frequenza, ma *dalla frequenza stessa*. Questa dipendenza è sensibile a partire dalla metà della frequenza di Nyquist, come mostra la figura, e diviene molto alta a frequenza prossime a Nyquist. Dato che la dipendenza è statica, è pensabile introdurre delle compensazioni, inserendo un guadagno funzione della frequenza in modo inverso a come mostrato. La necessità di una correzione del genere rende però dubbia l'utilità di questa variante rispetto all'oscillatore complesso.

6.2 Oscillatore sinusoidale "a guida d'onda"

L'oscillatore a guida d'onda (*Second-Order Digital Waveguide Oscillator)* è dovuto a Julius Orion Smith III e Perry R. Cook del CCRMA della Stanford University[13]. Esso è stato elaborato in vista di una sua implementazione nel silicio, con tecnologie VLSI (*Very Large Scale Integration)*.

Esso è qui solo citato per completezza, perché il vantaggio del suo uso è oggi dubbio, stante la complessità del calcolo di rinormalizzazione per variazioni di frequenza e il numero non piccolo di operazioni comunque richieste. Si deve tenere presente infatti che nella implementazione in silicio in logica cablata, il costo di una moltiplicazione è (e sopratutto, *era* al tempo di cui stiamo parlando) molto superiore a quello di una somma. L'algoritmo quindi riduce le moltiplicazioni a due, a scapito di un notevole numero di somme, e di un complesso calcolo di rinormalizzazione in caso di variazione di frequenza.

7 Oscillatori tabellari

Mentre la realizzazione di un oscillatore complesso (o di un oscillatore reale) è concepibile e fattibile nel dominio analogico, gli oscillatori tabellari sono creature esclusive del mondo numerico[14].

L'idea base è quella di generare una forma d'onda leggendone i valori da una tabella di memoria esplorata sequenzialmente:

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x(n)	0	0,41	0,74	0,95	0,99	0,87	0,59	0,21	-0,21	-0,59	-0,87	-0,99	-0,95	-0,74	-0,41	0

Ad esempio, la tabella contiene un seno calcolato in 16 valori equispaziati, ma avrebbe potuto trattarsi di una forma d'onda qualsiasi.

L'emissione del segnale corrispondente è piuttosto semplice. Ad ogni istante di campionamento:

$$n = (n+1) \, mod \, 16 \quad \rightarrow \quad s_n = x(n)$$

L'operazione *mod* indica il resto della divisione tra il termine a sinistra e quello a destra.

---

13 Julius O. Smith III, Perry R. Cook, *The Second-Order Digital Waveguide Oscillator,* CCRMA, Standford. Data sconosciuta ma posteriore al 1990.

14 Una versione *continua* dell'oscillatore tabellare è costituita dai *loop* di nastro magnetico (o i solchi circolari dei grammofoni) della *musique concrète*.

Di fatto, il risultato percorre i numeri da 0 a 15 in modo ciclico, tornando a zero ogni volta che n raggiunge 16 :

(0 *mod* 16 0; 1 *mod* 16 = 1; 2 *mod* 16 = 2; ...15 *mod* 16 = 15; 16 *mod* 16 = 0).

L'operazione *mod* ci fornisce una *rampa numerica* (un dente di sega) tra 0 e 15, che ci permette di fare la scansione della tabella.

In generale, se la tabella è lunga *N:*

$$n=(n+1)\,mod\,N \quad \rightarrow \quad s_n=x(n) \qquad 20$$

L'operazione *modulo* è particolarmente veloce, soprattutto in aritmetica intera, e può diventare implicita[15] se *N* è proprio la lunghezza di un "tipo" intero non segnato a disposizione nel sistema[16]. In tal caso, basta eseguire l'operazione incremento e automaticamente il valore del registro sarà riportato a zero una volta raggiunto il valore massimo.

Così facendo abbiamo una frequenza fissa: l'operazione di estrarre *N* campioni alla frequenza di campionamento f_s dura un tempo $T=N/f_s$, e dunque avremo un frequenza di ripetizione:

$$\nu=1/T=f_s/N \qquad 21$$

Chiamiamo questa *frequenza naturale* dell'oscillatore: essa dipende dalla frequenza di campionamento e dalla lunghezza della tabella.

Emerge qui il primo problema degli oscillatori tabellari (e dei sistemi campionati nel tempo*)*: dato che questi operano a frequenza di campionamento fissa e predeterminata, e dato che *N* è giocoforza un intero, *le frequenze naturali degli oscillatori a tabella sono discrete*. Se operassimo in questo modo, saremmo limitati ad un pettine di frequenze possibili, non equispaziate, più fitte alla basse frequenze e meno fitte alle alte. Inoltre, saremmo costretti ad avere, per ogni forma d'onda, una tabella per ognuna delle frequenze desiderate.

A titolo di esempio, esaminiamo alcune zone relative alla classica frequenza di campionamento di 44.100 Hz:

N	**44100**	**44099**	**44098**	**22050**	**22049**	**22048**	**220**	**219**	**218**	
ν	1,0000	1,0000	1,0000	2,000	2,000	2,000	200,45	201,37	202,29	Hz
N	**22**	**21**	**20**	**8**	**7**	**6**	**4**	**3**	**2**	
ν	2004,55	2100	2205	5512,5	6300	7350	11025	14700	22050	Hz

Come si vede, le frequenze alte hanno una grana decisamente grossa, mentre quelle basse la hanno eccessivamente fine.

Si può pensare di utilizzare una sola tabella, e di modificare la velocità di scansione. Dato che questa è legata alla frequenza di campionamento che è invariabile, possiamo pensare di procedere "saltando" dei valori. Così facendo ab-

15 Senza cioè oneri di calcolo.

16 Ad esempio, tipi del *C-language* per e CPU classe Pentium sono lo *unsigned byte* (8 bit: 0-255), lo *unsigned short integer* (16 bit: 0-65535), lo *unsigned integer* (32 bit).

brevieremmo il tempo di scansione totale, aumentando la frequenza. Per fissare le idee, supponiamo di partire da una tabella di lunghezza 4.410, la cui frequenza naturale a 44.100 Hz di campionamento è 10 Hz.

La situazione delle frequenze disponibili non migliorerebbe di molto, però: leggendo *con passo 2* (saltando cioè un campione ogni due) avremmo un raddoppio della frequenza, con passo 3 una triplicazione, e così via. Di fatto, possiamo in questo modo ottenere solo frequenze armoniche di quella naturale, il che non sarebbe accettabile a basse frequenze neanche adottando tabelle molto lunghe. Ad esempio, adottando una tabella contenente 1 sec. di forma d'onda (44.100 campioni) avremmo le armoniche di 1 Hz, il che significa una grana di 1 semitono a 20 Hz, ancora troppo per molte esigenze, quali ad esempio quelle della modulazione di frequenza.

Si potrebbe pensare di usare uno schema non regolare di salti, in modo da ottenere frequenze intermedie, ad esempio, facendo un passo 2 e un passo 3 alternati, mediamente cioè con passo 2,5, o schemi simili. In questo modo però si distorcerebbe significativamente la forma d'onda.

Lavorare a passi regolari richiederebbe tabelle molto lunghe, pari a secondi di suono. Le dimensioni delle moderne memorie permetterebbero agevolmente tabelle del genere[17], ma si è preferito e si preferisce dare al problema una risposta diversa: "liberandosi" nella natura intera della tabella, e passando a velocità di scansione espresse con una grana più fine, generalmente – almeno oggi - come numero in virgola fluttuante *(floating point)*. Questo equivale a definire posizioni frazionarie nella tabella, e quindi tempi di scansione frazionari o fluttuanti.

Anzitutto si deve notare come quel segnale a rampa ottenuto dalla 20 funga di fatto da segnale di fase all'interno di una ipotetica funzione periodica di periodo *N*[18] definita, anziché come usuale da una espressione matematica, da una tabella:

$$n = n+1 \;\rightarrow\; s_n = tab(n)$$

Ciò di cui abbiamo bisogno è dunque un segnale floating a rampa di durata selezionabile $\tau = 1/\nu$ che funga da fase periodica floating di periodo *N*:

$$rampa_n = N\left[\left(\frac{n}{\tau}\right) fmod\ 1\right] \qquad 22$$

Qui *fmod* indica la funzione *modulo* ma estesa ai floating point:

$$fmod\,(a,b) = \frac{a}{b} - floor\left(\frac{a}{b}\right)$$

In breve, $fmod\ 1$ restituisce la parte frazionaria del quoziente:

17 Per avere le massime prestazioni però è opportuno che tabelle che debbano essere scandite ad ogni intervallo di campionamento risiedano nella memoria *cache* dati della CPU, non nella memoria centrale. Le *cache* sono oggi dell'ordine del megabyte, ovvero di 256 Kfloat a 32 bit.

18 Esattamente come la funzione seno è periodica di periodo 2π

$$(xyz\,,abcdef...\ fmod\ 1)=0,abcdef...$$

Nella 22 il termine tra parentesi quadre indica un dente di sega di ampiezza 1 (da 0 a 1), della durata τ, quindi di frequenza $\nu=1/\tau$ [19]. Moltiplicato per *N* permette di fare la scansione della intera tabella alla frequenza (qualsiasi) $\nu=1/\tau$, come desiderato.

Il dente di sega 22 produce però valori *floating*, frazionari, mentre la tabella è definita solo per valori interi della fase. Resta quindi da risolvere il problema della conversione di questo valore frazionario in uno intero.

Un metodo brutale di conversione consiste semplicemente nell'eseguire l'approssimazione al più prossimo intero del valore della rampa. In questo modo talvolta un valore verrebbe emesso più volte (frequenze basse) oppure alcuni valori verrebbero saltati (frequenze alte). L'uscita si presenterebbe "a gradini", con un grado di distorsione della forma dipendente dalla frequenza di scansione.

Un metodo meno brutale consiste nell'interpolare linearmente tra i due punti adiacenti, come se fossero uniti da un segmento:

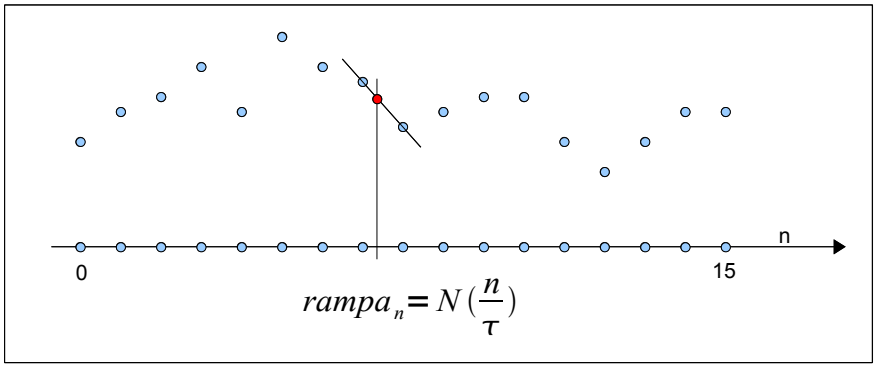


Fig. 13 - Tabella interpolata

Questo modo di procedere implica che ciò che viene generato è in realtà la forma d'onda "spezzata" ottenuta dall'interpolazione di tutti i suoi punti. Se la forma "sottostante" è ad esempio una sinusoide (perché si desidera implementare un oscillatore sinusoidale), questo comporta un certo grado di *distorsione armonica* (rispetto a quanto desiderato), perché una sinusoide "spezzata" non contiene solo la frequenza che gli è propria, ma anche un certo numero di armoniche con ampiezze non trascurabili.

La distorsione armonica così generata dipende dalla differenza tra valore interpolato e valore "vero", sinusoidale, e quindi, quando per caso il valore floating generato è proprio un intero, è nulla, e quindi tale anche la distorsione. In media questa è evidentemente funzione del numero di punti della tabella: più elevato *N*, più piccolo l'errore. Al limite, questo errore diventa nullo per $N\to\infty$, e può quindi essere reso piccola a piacere purché si scelga *N* sufficientemente grande.

19 Il tempo di passaggio tra *n* e *n+1* è il periodo di campionamento Δt, quindi il termine tra parentesi quadre passa da 0 ad 1 quando $n\cdot\Delta/\tau=1$, ovvero dopo un tempo $n\Delta t=\tau$, cioè in un tempo τ.

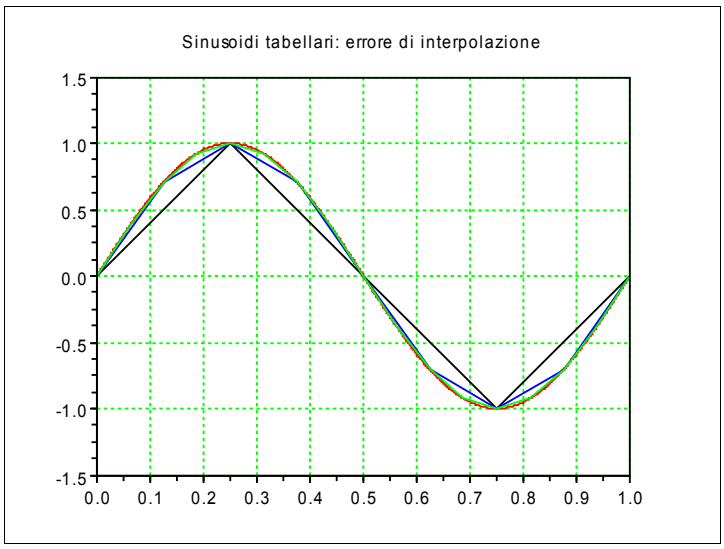


Fig. 14 - Sinusoide (---) tabellare interpolata con 4 (---), 8 (---) e 16 (---) punti

7.1 Continuità di fase e costanza di ampiezza.

La costanza dell'ampiezza non costituisce ovviamente nessun problema per l'oscillatore tabellare: l'ampiezza del segnale è definita in modo invariabile dai valori memorizzati o interpolati. Per assicurare la continuità di fase bisogna *integrare* la frequenza per ottenere la fase come già visto per gli altri oscillatori: memorizzando la fase precedente e incrementandola secondo quanto indicato dalla frequenza istantanea:

$$rampa_n = N\left[(rampa_{n-1} + \tau f_s)\ fmod\ 1\right]$$

In questo modo l'oscillatore tabellare può essere usato anche in regime di frequenza o fase variabile.

7.2 Forme d'onda e composizione spettrale

Ci si deve a questo punto domandare cosa succeda con diverse forme d'onda memorizzate. Se si memorizza una sinusoide, a parte la "triangolarizzazione" dovuta all'interpolazione, avremo in uscita una sinusoide di frequenza pari alla frequenza di scansione della tabella.

Se si memorizza invece una forma d'onda qualsiasi, questa sarà esprimibile come combinazione lineare di armoniche, alla Fourier. Dato che anche qui ci troviamo in un mondo campionato (sia pure, questa volta, a *frequenza variabile*, grazie ai nostri sforzi), anche qui abbiamo un limite di Nyquist, quindi potremo esprimere solo forme d'onda con un numero finito di armoniche.

Il limite di Nyquist si situa proprio al medesimo valore del nostro sistema campionato nel quale ci troviamo ad operare, quando l'oscillatore è utilizzato alla sua *frequenza naturale*. Modificando la frequenza di scansione, altereremo tutte le frequenze contenute nella forma d'onda in modo proporzionale, come se le moltiplicassimo tutte per lo stesso fattore. I rapporti tra le frequenze (quindi, *gli intervalli* in senso musicale) resteranno costanti, una circostanza del tutto di-

versa dallo spostamento di frequenza, nel quale alle frequenze viene invece *sommata* la stessa quantità, alterandone quindi i rapporti reciproci.

In altre parole, modificando la frequenza di scansione, noi modifichiamo "*l'altezza" (*il *pitch)* del suono, lasciandolo intonato. Si tratta in breve di una *trasposizione*.

E' opportuno a questo punto fare presente come le peculiarità dei sistemi numerici, campionati, si manifestino in questa circostanza.

Se la forma d'onda contiene frequenze fino a Nyquist, alterando verso l'alto la frequenza di scansione, rispetto a quella naturale, avremo *fold-over*. Le frequenze alte, vicine a Nyquist, sarebbero infatti spinte *oltre* Nyquist, cosa impossibile in un sistema campionato.

Il nostro oscillatore *non si comporterà dunque come l'equivalente analogico.* Quest'ultimo infatti è libero di produrre tutte le armoniche che desidera, senza *fold-over*. Le frequenze alte saranno semplicemente attenuate o tagliate da qualcuno dei filtri passa-basso impliciti nel sistema (la banda passante dell'elettronica, o presumibilmente, ancora prima, il nostro orecchio).

Se la forma d'onda è a banda limitata al di sotto di Nyquist, avremo dei margini al di sotto dei quali non si ha *fold-over*.

Ma la differenza si manifesterebbe anche per spostamenti di altezza verso il basso. Nell'oscillatore numerico, infatti, la frequenza massima presente nel segnale si abbasserà anch'essa con la fondamentale, lasciando oltre se stessa uno "spazio vuoto" di frequenze. Il altre parole, emetterà un segnale a banda bruscamente limitata, con questo limite che segue la fondamentale.

In un oscillatore analogico, comunque sia concepito e costruito, è impossibile avere forme d'onda bruscamente limitate: le frequenze alte possono essere rapidamente attenuate con la frequenza, ma non potranno mai andare rigorosamente a zero, e l'attenuazione avrà necessariamente un carattere "dolce" (in genere, una pendenza del tipo 6 dB/ottava). Quindi un oscillatore analogico emette sempre "tutte" le frequenze, indipendentemente dalla frequenza scelta per la fondamentale. Queste saranno limitate dal nostro orecchio, o da altri dispositivi analogici i quali comunque non possono produrre neanche loro cadute brusche, "a muro", delle frequenze presenti.

7.3 Dimensioni della tabella

Ci si deve domandare quali siano le dimensioni più adeguate per una tabella. Per minimizzare l'errore di interpolazione siamo spinti ad adottare tabelle lunghe, quindi con frequenze naturali basse.

Finora ci siamo occupati prevalentemente di cosa accade volendo leggere la tabella a frequenze superiori a quella naturale. Se invece adottiamo una frequenza inferiore, questo significa che il nostro puntatore "a rampa" procederà più lentamente di un punto a istante di campionamento. Sosterà quindi più volte tra un ingresso e il successivo della tabella. Se l'oscillatore non fosse stato inter-

polato, questo avrebbe provocato l'emissione dello stesso valore per più volte, e la forma d'onda sarebbe stata riprodotta “a gradini”. L'interpolazione fornisce però tutti i valori intermedi tra un campione e l'altro, e quindi anche in questo caso la lunghezza della tabella gioca a favore di una minore distorsione armonica.

Sono quindi preferibili, da tutti i punti di vista, tabelle lunghe. Valori tipici sono 512 valori per tabelle sinusoidali, ma nelle moderne architetture sono pensabili anche valori maggiori.

Se la tabella contiene però una forma d'onda non sinusoidale, bisogna tenere presente che queste considerazioni si applicano alla più alta frequenza contenuta nella forma d'onda. Quindi è bene che questa sia rappresentata da un numero di punti sufficiente in modo da non avere un errore troppo grosso di interpolazione.

Supponiamo il seguente caso:

Forma d'onda con 16 armoniche, lunga 128.

Un periodo della fondamentale è rappresentato con 128 campioni.

Un periodo della seconda armonica è rappresentato con 64 campioni.

....

Un periodo della 16ª armonica è rappresentato con 8 campioni.

Dunque la 16 armonica si trova ad essere rappresentata con un numero di campioni probabilmente insufficiente (vedi fig. 14 a pag. 22).

Naturalmente queste considerazioni vanno commisurate con gli effettivi requisiti auditivi e musicali: non è detto che la distorsione armonica dovuta all'interpolazione costituisca sempre un problema o un effetto indesiderato.

8 Banchi di oscillatori

8.1 Definizione

Un *banco di oscillatori* è un algoritmo che produce un pettine di frequenze armoniche a partire da un segnale sinusoidale alla frequenza della fondamentale. Naturalmente lo fa, rispetto ad un set di oscillatori indipendenti, con un minore numero di operazioni, ma al prezzo di imporre un rapporto rigidamente ed esattamente armonico tra le frequenze, ed una relazione fissa tra le fasi delle armoniche. Queste relazioni sono invariabili, non modulabili.

Questa circostanza ha come ovvio riferimenti diretti a questioni di rilevanza estetica e musicale, che saranno affrontati in uno specifico scritto.

In un set di oscillatori *indipendenti* siamo infatti liberi di stabilire ogni singola frequenza (che può dunque appartenere esattamente ad un pettine armonico,

oppure non appartenervi affatto, oppure ancora solo approssimativamente). Inoltre, siamo liberi di scegliere in modo del tutto arbitrario la fase iniziale per ogni oscillatore e di variarla nel tempo. Un banco di oscillatori produce invece un set armonico perfetto con relazione di fase strettamente predeterminate: è dunque in sostanza equivalente a un set di oscillatori *dipendenti*.

8.2 Banchi di oscillatori complessi

Torniamo all'oscillatore complesso espresso nella sua forma più semplice:

$$output(t)=e^{i\varphi(t)+\phi}$$

Facendo il quadrato di un tale oscillatore (operazione che implica una moltiplicazione complessa), abbiamo:

$$output^2(t)=(e^{i(\int_0^t \omega(\tau)d\tau+\phi)})^2=e^{i(\int_0^t 2\omega(\tau)d\tau+2\phi)} \qquad 23$$

In altri termini, il quadrato di un oscillatore complesso (un fasore) è un oscillatore (un fasore) a frequenza doppia e con fase iniziale doppia.

Ma in generale:

$$osc^n(t)=(e^{i(\int_0^t \omega(\tau)d\tau)+\phi})^n=e^{i(\int_0^t n\omega(\tau)d\tau+n\phi)} \qquad 24$$

Dunque:

(II) La potenza n-esima di un oscillatore complesso è sua volta un oscillatore complesso di frequenza n volte quella di partenza: ne è dunque la n-esima armonica. La sua fase iniziale è n volte quella iniziale della fondamentale.

Per ottenere dunque un pettine di armoniche da un oscillatore complesso, basta calcolare le potenze (complesse) dello stesso, e applicare a ciascuna la ampiezza relativa desiderata (profilo spettrale) a_n. Uno schema di calcolo possibile è il seguente:

1. fondamentale: $s(t)=e^{i(\omega t+\phi)}$
2. 2ª armonica: $s_2(t)=s(t)^2=s(t)\cdot s(t)$
3. 3ª armonica: $s_3(t)=s(t)^3=s_2(t)\cdot s(t)$
4. 4ª armonica: $s_4(t)=s(t)^4=s_3(t)\cdot s(t)$
5. 5ª armonica: $s_5(t)=s(t)^5=s_4(t)\cdot s(t)$
6. 6ª armonica: $s_6(t)=s(t)^6=s_5(t)\cdot s(t)$

.....

$$output(t)=\sum_{n=1}^{N} a_n s_n(t)$$

Come si vede, la produzione di ciascuna armonica richiede solo l'esecuzione di un prodotto complesso. Rispetto alla esecuzione di un algoritmo per un oscillatore complesso a frequenza *n* volte la fondamentale, abbiamo dunque il risparmio della memorizzazione (e dell'accesso in lettura) allo stato precedente, ed inoltre non è necessario calcolare o memorizzare i coefficienti *C* ed *S* relativi alle diverse armoniche, così come non è necessario accedervi in lettura.

Si tratta di un vantaggio indubbio, anche se non clamoroso. Il prezzo da pagare – come già detto - sta nel rigido rapporto tra frequenze (perfettamente armonico) e nel rigido rapporto di fase con la fondamentale che l'algoritmo produce.

8.2.1 Continuità di fase

Se l'oscillatore che fornisce la fondamentale è gestito con continuità di fase, anche tutto il banco lo sarà. Tuttavia è bene ricordare che anche la fase iniziale viene moltiplicata per l'indice di armonica. Pertanto, tutti gli oscillatori avranno fase iniziale uguale solo se la fase iniziale della fondamentale è zero.

Se il banco è gestito per *frame* inferiori ai 20 msec, la circostanza è irrilevante ai fini percettivi.

8.3 Banchi di oscillatori reali

Anche con oscillatori reali è possibile realizzare un banco di oscillatori. Si supponga di avere un oscillatore cosinusoidale:

$$s(t)=\cos(\varphi(t)+\phi)$$

La scelta del coseno, effettuata per comodità, non restringe la generalità dei risultati, a causa della presenza del termine di fase ϕ : possiamo infatti ottenere una sinusoide semplicemente ponendo $\phi=\pi/2$.

Dalle formule di duplicazione abbiamo:

$$\cos(2(\varphi(t)+\phi))=2\cos^2(\varphi(t)+\phi)-1$$

L'operazione, che consiste in tre prodotti e una somma:

$$s_2(t)=2s(t)^2-1$$

produce dunque una seconda armonica con fase iniziale doppia.

Per la terza armonica abbiamo:

$$\cos(3x)=4\cos^3(x)-3\cos(x)$$

In generale:

$$\cos(nx)=2\cos([n-1]x)\cos(x)-\cos([n-2]x) \qquad 26$$

Questa formula permette di esprimere una segnale di frequenza e fase *n-upla* nei termini dello stesso segnale di frequenza e fase *(n-1) e (n-2),* dando luogo alla possibilità di uno schema di calcolo iterativo simile (ma più complicato) del caso complesso.

$$s_2(t)=2s(t)^2-1$$
$$s_3(t)=4\,s(t)^3-3\mathrm{s}(t)$$
$$\cdots\cdots$$
$$s_n(t)=2s_{n-1}(t)\cdot s(t)-s_{n-2}(t)$$

Il numero di operazioni da eseguire, a parte seconda e terza armonica, è pari a due moltiplicazioni e una somma, una quantità pari alla metà di quelle necessarie per un prodotto complesso.

Per evitare complicazioni nella rinormalizzazione, come oscillatore per la fondamentale può essere utilizzato l'oscillatore complesso, prelevandone solo la parte immaginaria o quella reale.

8.3.1 Continuità di fase

Se l'oscillatore che fornisce la fondamentale è gestito con continuità di fase, anche tutto il banco lo sarà. Riguardo alla fase iniziale, le cose non vanno così semplicemente come per l'oscillatore complesso, perché la seconda armonica ha una fase diversa dalla fondamentale, come si evince dalla 25 (se $s(t)$ è una sinusoide, $s_2(t)$ è una cosinusoide).

Se l'oscillatore è gestito per *frame* inferiori ai 20 msec, valgono le stesse considerazioni svolte a pag. 22, e dunque le relazioni di fase tra le diverse armoniche non saranno percepibili.

9 Banchi di oscillatori versus oscillatori tabellari

Dalle considerazioni svolte è facile capire perché i banchi di oscillatori siano caduti in disuso. In effetti, il risultato di un banco non è sostanzialmente differente da quello di un oscillatore tabellare caricato con la forma d'onda "sintetizzata" a partire dal profilo spettrale e di fase proprio del banco. Inoltre l'oscillatore tabellare può produrre forme d'onda qualsivoglia, mentre un banco può produrre solo quelle compatibili con le sue rigide relazioni di fase. Quest'ultima considerazione deve essere temperata con la considerazione che peraltro noi siamo pressoché insensibili alla forma d'onda, ma solo al suo contenuto spettrale. Non c'è infine dubbio che un oscillatore tabellare anche interpolato presenta un minore carico di calcolo, anche per numeri piccoli di armoniche, e da questo numero indipendente.

Tuttavia va notato che il banco di oscillatori permette un controllo (anche tempo-variante) sul profilo spettrale, ottenuto semplicemente variando i coefficienti a_n o il numero di armoniche calcolate. Lo stesso effetto con un oscillatore tabellare richiederebbe l'uso di un filtro tempo-variante. E' quindi inoltre più

facile ovviare all'inconveniente descritto a pag. 22 con un banco, “sorvegliando” che la frequenza massima sia sempre la maggiore possibile purché inferiore a Nyquist.

10 Tempo varianza

Le considerazioni fin qui svolte ci permettono molto rapidamente e semplicemente di intendere cosa accada in caso si voglia lavorare con sintesi tempo-variante. Non vi è infatti alcun motivo per pensare che le variazioni di frequenza o di fase dei segnali generati abbiano necessariamente un carattere periodico.

Potremmo tranquillamente parlare di partitura in frequenza (o in altezza, se si lavora con banchi), ad esempio. Operando in sintesi additiva, non c'è motivo per variazioni di frequenza “al campione”, ma sarà assai presumibilmente sufficiente far variare frequenze o altezza per frame dell'ordine della ventina di millisecondi, trascurando a questo punto le relazioni di fase tra le armoniche o le parziali (a seconda dei casi). Al cambio di frame la variazione di frequenza va eseguita con continuità di fase, e quindi è necessario usare oscillatori che operino (correttamente) in continuità di fase. All'interno di ciascuna frame, ampiezze e frequenze resterebbero invece costanti.

10.1 Time stretching e pitch shifting

La sintesi additiva per *frame* apre una interessante possibilità.

Se si modifica la lunghezza temporale della frame, si ha quello che viene definito *time-stretching* (stiramento del tempo): è possibile accelerare o rallentare il brano senza alterarne il *pitch*[20]. Se invece modificassimo le frequenze degli oscillatori (moltiplicandole tutte per un determinato fattore, nel caso di oscillatori indipendenti, oppure alterando la frequenza dell'oscillatore fondamentale nel caso dei banchi), altereremmo il *pitch* senza alterare le durate (la prosodia). La sintesi additiva fornisce quindi in modo “naturale” la possibilità di alterare indipendentemente altezze e durate, senza artefatti.

E' quindi possibile analizzare un brano concreto in termini di oscillatori indipendenti, e poi in fase di resintesi alterare in modo tempo-variante le altezze o la prosodia (il ritmo), accelerando o rallentando, o tutte e due le cose contemporaneamente e indipendentemente. Nel caso di rallentamenti notevoli si è coniata l'espressione di “microscopio acustico”, a significare che per questa via si possono ascoltare dettagli che appartengono ad una scala temporale che normalmente sfugge alla nostra capacità di discriminazione[21].

---

20 Un registratore analogico a nastro, ad esempio, si comporta diversamente rispetto a modifiche della velocità di trascinamento: accelerando, si abbreviano le durate e si alza il *pitch*. Viceversa abbassando la velocità. E' lo stesso comportamento di un oscillatore tabellare rispetto alla velocità di scansione, o di una sequenza di campioni convertita in analogico con una frequenza di campionamento diversa da quella originale.

21 Si potrebbe anche parlare di “orecchio di uccello”, dato che questi posseggono una discriminazione temporale molto superiore alla nostra.

http://www.mnt-aq.it
Versione: 0.5 del 6 Aprile 2009
Testi, formule e figure: OpenOffice
Grafici, calcoli: Scilab
Calcolo simbolico: Maxima