調律(チューニング)

シンプルな1本のサイン波(線)を考える。サンプル化されたデジタル表現の場合、音の周波数が正確に表現できるのは、サイン波の半波長がサンプル間隔の(正の)整数倍の時だけである。

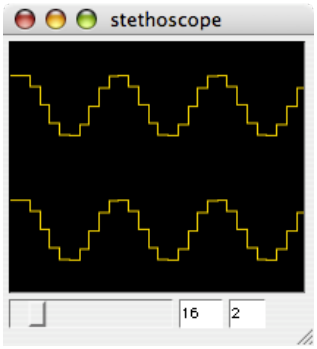
```
// 周波数の高い(波長の短い)方から20個書き出してみる。

20.do({|i| (i+1).post; ":".post; (44100/(i+1)).post; "[Hz]".postln; });
```

```
1:44100[Hz]
2:22050[Hz]
3:14700[Hz]
4:11025[Hz]
5:8820[Hz]
6:7350[Hz]
7:6300[Hz]
8:5512.5[Hz]
9:4900[Hz]
10:4410[Hz]
11:4009.0909090909[Hz]
12:3675[Hz]
13:3392.3076923077[Hz]
14:3150[Hz]
15:2940[Hz]
16:2756.25[Hz]
17:2594.1176470588[Hz]
18:2450[Hz]
19:2321.0526315789[Hz]
20:2205[Hz]
```

高周波になるにしたがって、正確に表現できる周波数が疎らになっていくことがわかる。

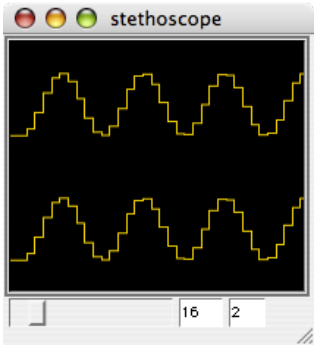
実際に音を聴いてみる。

```
// 4410[Hz]のサイン波を発生する
~out = { SinOsc.ar(4410, 0, 0.5, 0).dup };
```



4410[Hz]のサイン波の場合、1周期の波形は10個のサンプルで表現されている。この場合、4410はサンプリング周波数の整数分の1(1/10)なので、すべての波形は同じかたちをしている。

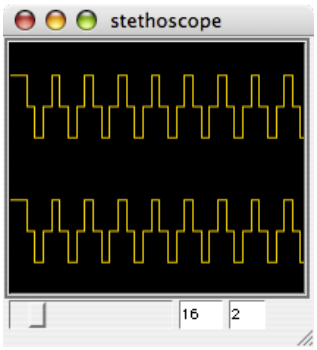
```
// 4500[Hz]のサイン波を発生する
~out = { SinOsc.ar(4500, 0, 0.5, 0).dup };
```



周波数を少し変化させ、サンプリング周波数の整数分の1からずれると、波形のかたちは周期ごとに少しずつ変化し始める。
しかしこのあたりの周波数の場合、実際の音にこの不整合の影響はほとんど現れてこない。

次にもっと高い周波数で試してみる。

```
// 11025[Hz]のサイン波を発生する
~out = { SinOsc.ar(11025, 0, 0.5, 0).dup };
```

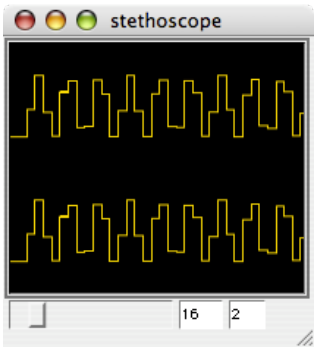


11025(=44100/4)[Hz]の場合、1周期の波形が4つのサンプルで表現されている。
周波数をもう少し高くして、サンプリング周波数の整数分の1からずらしてみる。

```
// 11025[Hz]のサイン波を発生する
~out = { SinOsc.ar(11050, 0, 0.5, 0).dup };
```

11025[Hz]の時には安定していた波形がゆっくりと変動し始める

```
// 12000[Hz]のサイン波を発生する
~out = { SinOsc.ar(12000, 0, 0.5, 0).dup };
```

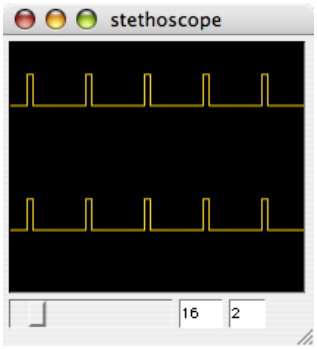


周波数を少しずらすと、波形は各周期ごとに大きくかたちが変ってくる。

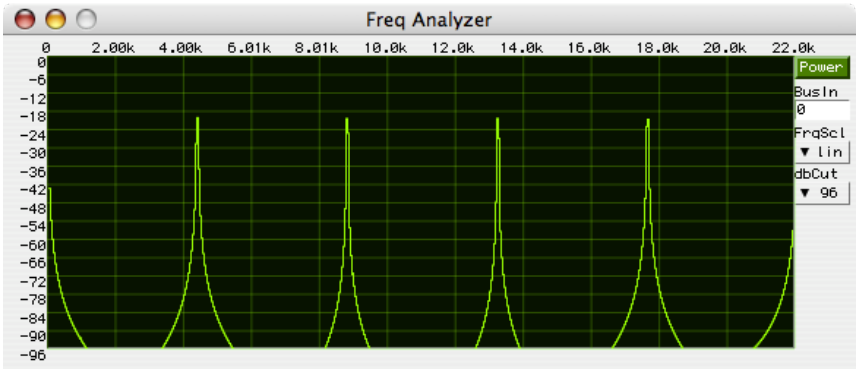
次にクリック(点)で試してみる。

```
// 4410[Hz]のクリック列を発生する
~out = { Impulse.ar(4410, 0, 0.5, 0).dup };
```

4410[Hz]の点列の場合、10サンプル毎に1回、規則正しくクリックが立ちあがる。

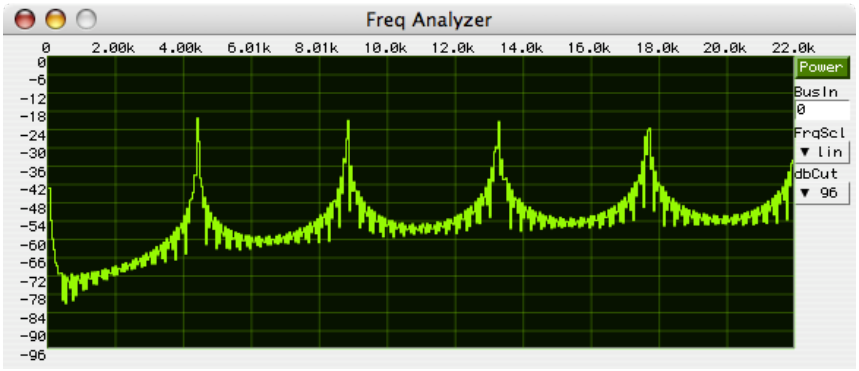


この波形(音)のスペクトルは以下のようになる。4410[Hz]の整数倍のピークが綺麗に立ち上っている。



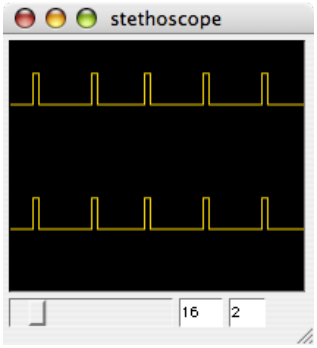

```
// 4420[Hz]のクリック列を発生する
~out = { Impulse.ar(4420, 0, 0.5, 0).dup };
```

点列の周期をサンプル数の整数倍から少しずらすと、音色が濁りノイズ成分が混入してくる。

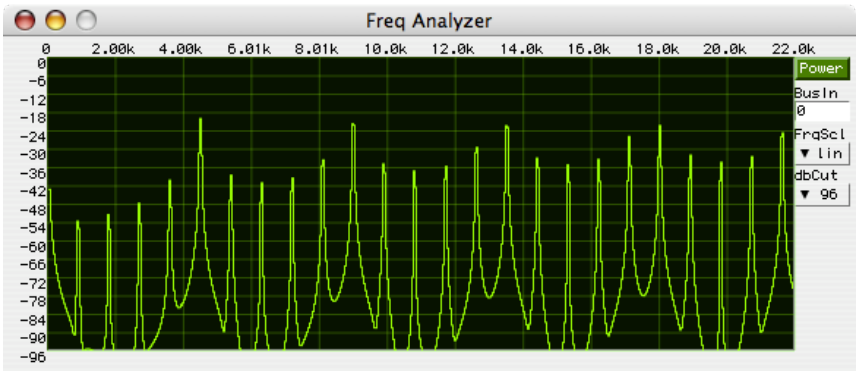



```
// 4500[Hz]のクリック列を発生する
~out = { Impulse.ar(4500, 0, 0.5, 0).dup };
```

さらに周波数をずらしてみる。周波数が一定であるにもかかわらず、点列が不等間隔になっていることがわかる。

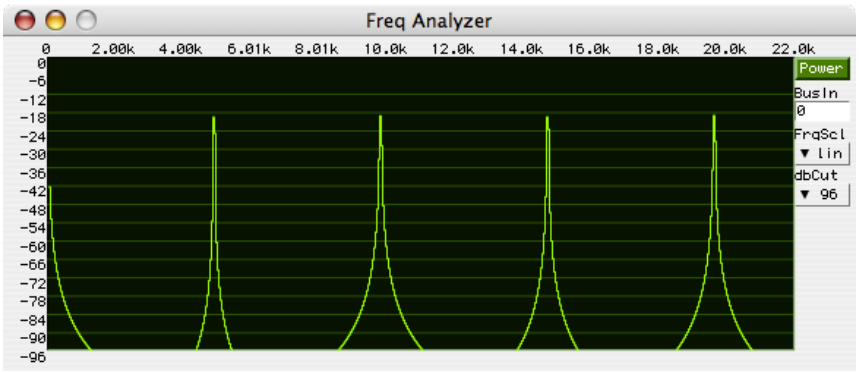


この波形のスペクトルをみると、さまざまな周波数成分が混在していることがわかる。



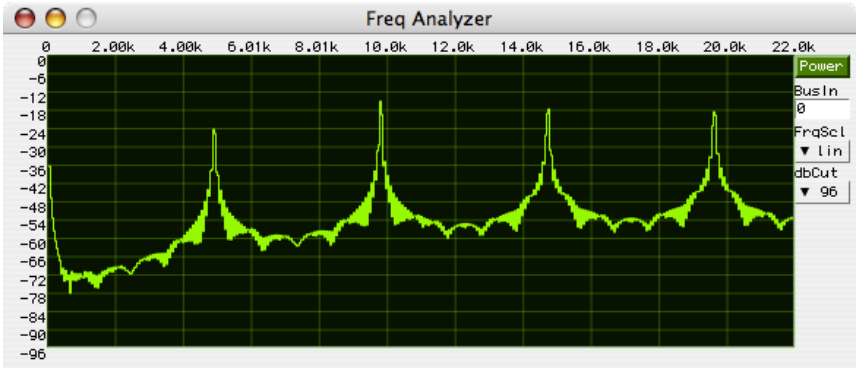

```
// 4900[Hz]のクリック列を発生する
~out = { Impulse.ar(4900, 0, 0.5, 0).dup };
```

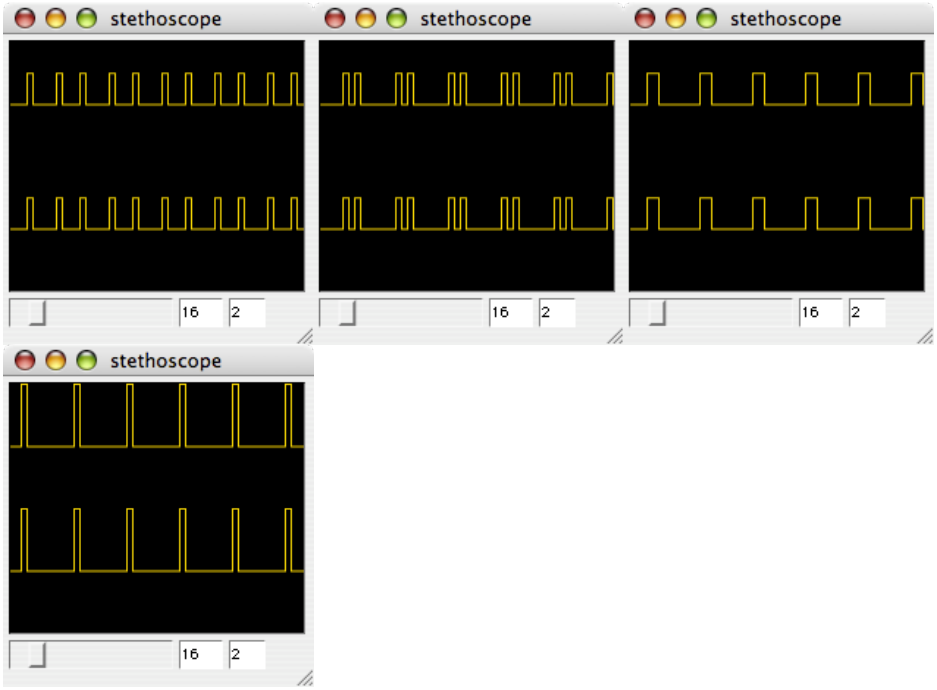
もう少し周波数を高くして、再び周期がサンプルの整数倍(9倍)に一致すると、スペクトルも音色もクリアになる。




```
// 4900[Hz]と4910[Hz]のクリック列を重ねる
~out = { Impulse.ar(4900, 0, 0.5, 0).dup + Impulse.ar(4910, 0, 0.5, 0).dup };
```

わずかに異なる周波数の点列を重ねると、以下のように複雑なスペクトルと動的な波形になる。波形のアニメーションから、2つの(周波数の)クリックが少しずつ移動し、重なったり離れたりしていることがわかる。





デジタル純正律

ある基音を起点として、音程が協和する(周波数の比が簡単な整数比になる)ように、音階の各音を順に採って決定していく音律のことを、純正律、あるいは純正調という。例えば純正律における1オクターヴの音程は周波数の比が1:2、完全五度は2:3、完全４度は3:4、長３度は4:5、短３度が5:6である。純正律に調律された音程や和音は、うなりを生じない澄んだ響きを生み出す。

デジタル表現の場合、常に基音としてのサンプリング周波数が存在する。サンプリング周波数を基準として、周波数の比が単純な整数比になると、音の響きがクリアになりスペクトルもクリアに立ちあがる。このサンプリング周波数を基準として良く響き合う音程や音律のことを「デジタル純正律(digital just temperament)」と呼ぶ。デジタル純正律とは、サンプリング周波数を基音とした周波数の比による協和不協和の関係が、音色におよぼす影響のことである。サンプリング周波数は一定なので、デジタル純正律は絶対的な周波数として提示される。

サンプリング周波数はデジタルサウンドの基音である

▼エクササイズ：響きを聴く

以下のサンプルコードを鳴らして、デジタル純正律からのズレによる音の響きの違いを確認せよ。音色だけでなく、波形やスペクトルの変化にも着目すること。

```
// 周波数用のプロキシを定義する
~freq.kr(1);
```

```
// 点列を鳴らす
~out = { Impulse.ar(~freq.kr, 0.0, 0.8).dup };

// 点列の周波数のスイープ(44100/5=8820[Hz]から44100/4=11025[Hz]まで)
~freq = { Line.kr(8820.0, 11025.0, 10, doneAction:2) };

// マウスの動きによって周波数を変化させる(ウィンドウの左端=44100/5=8820[Hz]、右端=44100/4=11025[Hz])
~freq = { MouseX.kr(8820.0, 11025.0, 1) };
```