点・線・面の定義

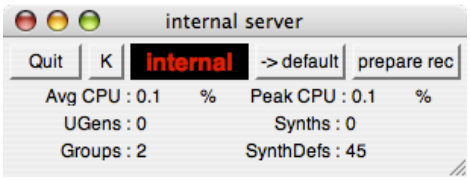
最初に、グラフィックス(視覚表現)における点、線、面に相当する、3つの基本的な音響要素(sound element)を定義する。

本書では

点 = クリック
線 = サイン波
面 = ノイズ

という幾何学的なメタファーをベースに、ボトムアップで構成的に音をつくりあげていくことを考える。この3つの基本的なメタファーによる音の視覚的イメージをコンポジションに活用することが、幾何学的/構成的アプローチの第一の目的である。

```
// 音を出すための準備
(
// インターナル・サーバーをデフォルトに指定して起動
Server.default = Server.internal;
s = Server.default;
if(not(s.serverRunning), {s.boot});
// プロキシ空間を用意し変数pに代入しておく
p = ProxySpace.push(s);
// フェイドアウト時間を1秒に設定
p.fadeTime = 1;
)
```



インターナル・サーバーのウィンドウ

```
// 出力用プロキシ(out)を用意
~out.ar(2);
// モニター開始
~out.play;
// 音量を設定
~out.vol = 1.0;
// 波形の出力
~out.scope;
```

幾何学的なアプローチと共に、本書では音や音楽/音響構造を視覚化することを重視する。音を視覚化することで、聴覚をサポートするというよりも、より深く音を聴きとるためのツールとして視覚表現を援用する。

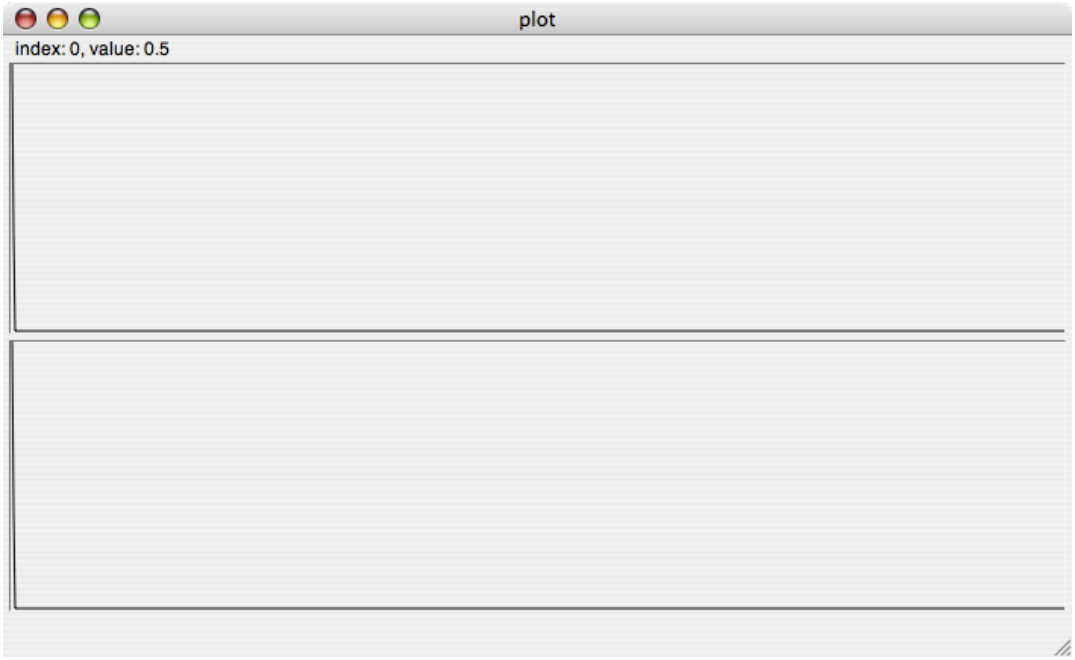
音を時間領域で視覚化したものが波形である。周波数に対する各成分の分布を表したスペクトルは、周波数領域で視覚化された音である。波形とスペクトルの両者を見ながら音を聴くことで、音と知覚の関係をより詳細に探求することができる。

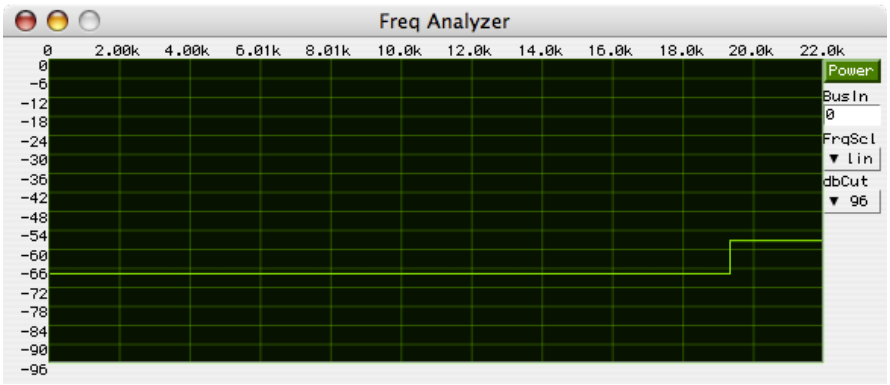
波形とスペクトルを見ながら音を聴く

```
// スペクトル・アナライザを起動
FreqScope.new(200, 0);

// 点==クリック
~out = { Impulse.ar(0.5.dup, 0, 0.5, 0) }.plot;
```

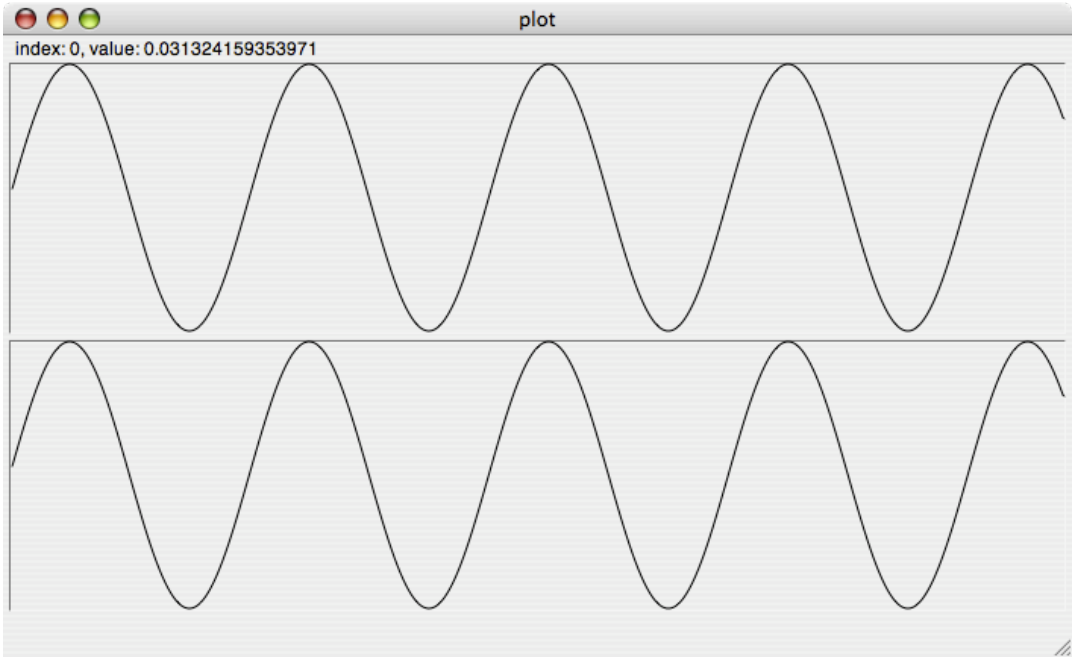
このコードを実行すると、プチッとした小さな音が2秒間隔で聴こえる。このプチッという音をクリック(click)という。クリック音は、デジタルオーディオにおける1サンプル、すなわち1つの数によって表現された音である。(理想的な)クリックは(数学的に)すべての周波数の音を含む。したがってスペクトル・アナライザには、水平の線が一瞬表われる。






```
// 線==サイン波(正弦波)
~out = { SinOsc.ar(440.dup, 0, 0.5, 0) }.plot;
```

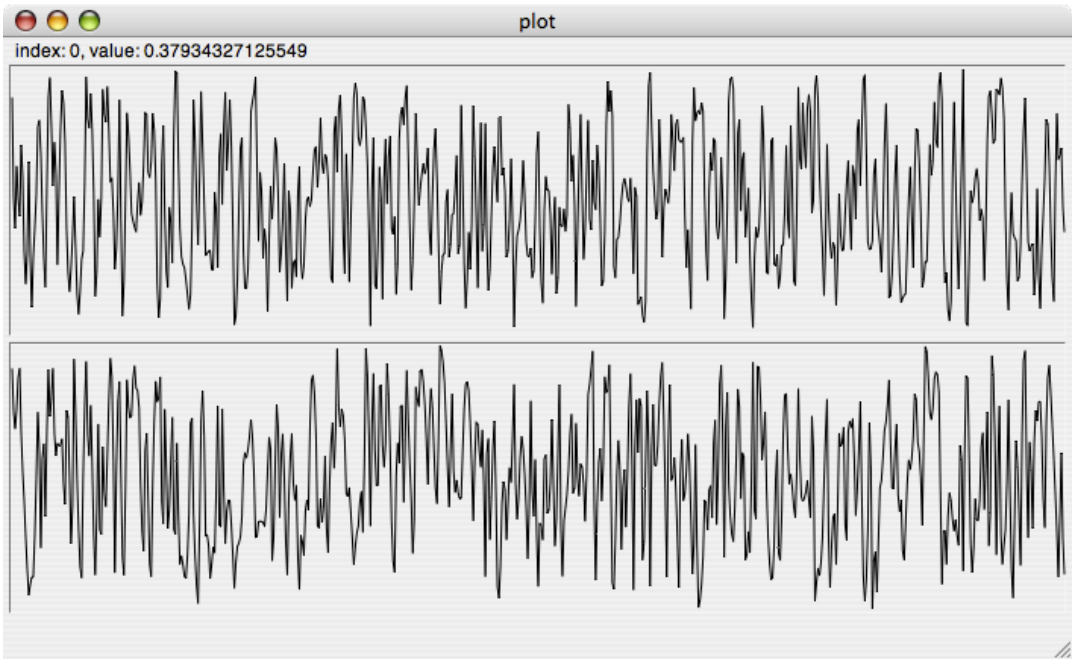
次にこのコードを実行すると、ポーッという持続音が聴こえる。これは441Hzのサイン波(sine wave)である。サイン波は三角関数で表現された波形による音である。テレビやラジオの時報でも用いられている。サイン波の波形は周期的であるため、これはいわば数字が整然と周期的に変動しながら連続しているという状態に相当する。スペクトル上では、(ほぼ)縦の一本線になる。

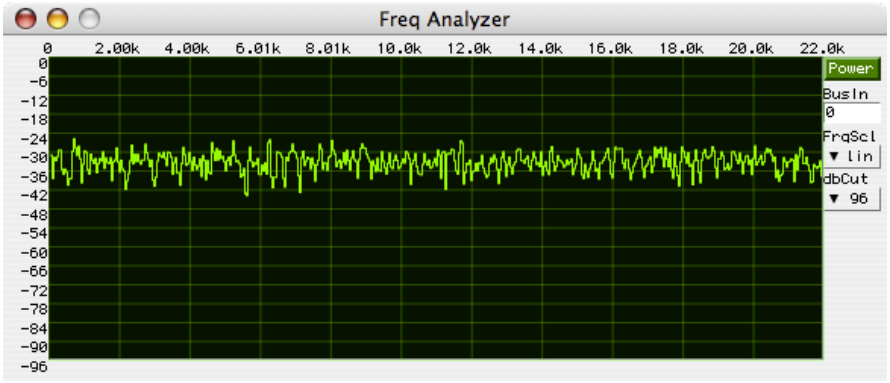



```
// 面==ホワイトノイズ

~out = { WhiteNoise.ar(0.5.dup, 0) }.plot;
```

今度はザーッという音が聴こえる。これはホワイトノイズ(whitc noise)と呼ばれる音で、すべての周波数の線が含まれている。互いに無関係なランダムな数字の羅列に相当する音でもある。スペクトル上では(ほぼ)横の一本線になる。





次にこの3つの音響要素をいくつかの特徴的なパラメータで表現する。音をパラメータで表現すると、音を抽象化することができる。コードによる音の表現には、「パラメータによる音の抽象」という特徴がある。

パラメータで音を抽象する

```
// 音響要素のパラメトリックな定義

// 点列(points)の定義
(
SynthDef(\points, {arg amp=0.5, freq=1, pan=0, sustain=10;
    var env;
    env = Env.linen(0, sustain, 0, amp, 0);
    Out.ar(0,
        Pan2.ar(
        Impulse.ar(freq) * EnvGen.ar(env, doneAction:2),
        pan))
    }).send(s);
)
```

点(列)には4つの特徴的なパラメータがある。振幅(amp)は個々の点の大きさを表わす。最大振幅が1なので、このパラメータが0.5であれば、最大振幅の半分の振幅(高さ)ということになる。周波数(freq)は点の時間的な間隔である。周波数が1であれば1秒間隔、周波数が2であれば、点は1/2=0.5秒間隔、0.2であれば点は1/0.2=5秒間隔ということになる。パン(pan)は音の左右のバランスを表現する音の空間情報である。左が-1で右が1となる。0の場合は音が中央に位置する。4つめのパラメータの継続(sustain)は点列の継続時間(秒)である。点列の周波数と継続時間で点の数とコントロールできる。

```
// デフォルトのパラメータ(amp=0.5, freq=1, pan=0, sustain=10)で点列を鳴らしてみる
// 出力に点列の定義をアサインする
~out = ~points;
// 音の出力
~points = \points;

// パラメータの値を変更して(freq=1->10)点を鳴らしてみる
~points.set(\freq, 10);
~points = \points;

// さらに周波数をあげていくとどうなるか?
~points.set(\freq, 210);
~points = \points;

~points.set(\freq, 2109);
~points = \points;

~points.set(\freq, 11050);
~points = \points;


// 線(line)の定義
(
SynthDef(\line, {arg amp=0.5, freq=441, pan=0, sustain=10;
    var env;
```

```
    env = Env.linen(0, sustain, 0, amp, 0);
    Out.ar(0,
        Pan2.ar(
        SinOsc.ar(freq) * EnvGen.ar(env, doneAction:2),
        pan))
    }).send(s);
)
```

線のパラメータは点列と同じである。振幅(amp)は線の大きさ、周波数(freq)は線の高さ、すなわちサイン波の振動数、パン(pan)が左右の位置情報、継続(sustain)が線の継続時間(秒)である。

```
// デフォルトのパラメータ(freq=441, amp=0.5, pan=0, sustain=10)で線を鳴らしてみる
~out = ~line;
~line = \line;

// パラメータの値を変更して(freq=441->4410)線を鳴らしてみる
~line.set(\freq, 441);
~line = \line;

// 面(plane)の定義
(
SynthDef(\plane, {arg amp=0.5, pan=0, sustain=10;
    var env;
    env = Env.linen(0, sustain, 0, amp, 0);
    Out.ar(0,
        Pan2.ar(
        WhiteNoise.ar * EnvGen.ar(env, doneAction:2),
        pan))
    }).send(s);
)
```

面には周波数パラメータが不要なので、面のパラメータは他の点列や面よりひとつ少ない。

```
// デフォルトのパラメータ(amp=0.5, pan=0, sustain=10)で線を鳴らしてみる
~out = ~plane;
~plane = \plane;
```

点列(クリック列)、線(サイン波)、面(ホワイトノイズ)はそれぞれ
・点列:振幅(amp)、周波数(freq)、パン(pan)、継続時間(sustain)
・線:振幅(amp)、周波数(freq)、パン(pan)、継続時間(sustain)
・面:振幅(amp)、パン(pan)、継続時間(sustain)
というパラメータを持っている。

パラメータの数のことを次元(dimension)という。点列と線のパラメータは4個、面のパラメータは3個であるから、点列と線の次元は4、面の次元は3、ということになる。

■エクササイズ:
上で定義した点・線・面の3つの音響要素のパラメータを変化させて、それらがどのように聴こえるかを確かめてみよ。

パラメータを時間的に変化させる

前述の定義には一つ問題がある。音が鳴っている間、振幅、周波数、パンといったパラメータが一定であるため、時間的にパラメータが変化する音が表現できない。そこで、この時間的にパラメータが変化する音を表現するために、振幅、パン、周波数の3つのパラメータをそれぞれ音の開始時刻(初期値)と終了時刻(終値)における値の組として表わす。その間は各パラメータの値を連続的(線形)に変化させる。これはパラメータの変化を直線近似することである。

```
// 時間変化する点列(points2)の定義
(
SynthDef(\points2, {arg amp1=0.2, amp2=0.7, freq1=2, freq2=0.5, pan1= -1, pan2=1, sustain=10;
    var env;
    env = Env.new([amp1, amp2], [sustain]);
    Out.ar(0,
        Pan2.ar(
        Impulse.ar(Line.kr(freq1, freq2, sustain)) * EnvGen.ar(env, doneAction:2),
        Line.kr(pan1, pan2, sustain)))
```

```
    }).send(s);
)

// デフォルトのパラメータで時間変化する点列を鳴らしてみる
~out = ~points2;
~points2 = \points2;

// 時間変化する線(line2)の定義
(
SynthDef(\line2, {arg amp1=0.7, amp2=0.2, freq1=4410, freq2=441, pan1= -1, pan2=1, sustain=10;
    var env;
    env = Env.new([amp1, amp2], [sustain]);
    Out.ar(0,
        Pan2.ar(
        SinOsc.ar(Line.kr(freq1, freq2, sustain)) * EnvGen.ar(env, doneAction:2),
        Line.kr(pan1, pan2, sustain)))
    }).send(s);
)

// デフォルトのパラメータで時間変化する点列を鳴らしてみる
~out = ~line2;
~line2 = \line2;

// 時間変化する面(plane2)の定義
(
SynthDef(\plane2, {arg amp1=0.7, amp2=0.2, pan1= -1, pan2=1, sustain=10;
    var env;
    env = Env.new([amp1, amp2], [sustain]);
    Out.ar(0,
        Pan2.ar(
        WhiteNoise.ar * EnvGen.ar(env, doneAction:2),
        Line.kr(pan1, pan2, sustain)))
    }).send(s);
)

// デフォルトのパラメータで時間変化する点列を鳴らしてみる
~out = ~plane2;
~plane2 = \plane2;
```

パラメータのうち「1」がついているものが初期値、「2」がついているものが終値である。
・点列：振幅1(amp1)、振幅2(amp2)、周波数1(freq1)、周波数2(freq2)、パン1(pan1)、パン2(pan2)、継続時間(sustain)
・線：振幅1、振幅2、周波数1、周波数2、パン1、パン2、継続時間
・面：振幅1、振幅2、パン1、パン2、継続時間

■エクササイズ：
上で定義した時間変化する点・線・面のパラメータを変化させて、それらがどのように聴こえるかを確かめてみよ。