3　音を重ねる

別々に録音した演奏を重ねることができると便利である。音の重ね合わせは簡単に実現できる。重ね合わせは時間領域の加算となる。

実際に足してみる。

```
>> [p1,fs1]=wavread('p1.wav');
>> [p2,fs2]=wavread('p2.wav');
>> pmix = p1 + p2;
??? エラー ==> plus
行列の次元は一致しなければなりません。
```

運がよくない限り、このようなエラーが出てしまうだろう。p1 と p2 は、列ベクトルであり、このようなエラーが出る場合には両方のベクトルの要素数が異なっている。

そのようなエラーを回避するためには、ベクトルの長さを揃えなければならない。

ベクトルの長さを揃える方法を考える。

2 つの長さの異なる列ベクトルを作成する。

```
>> fs=8000;
>> t1=(0:1/fs:1)';
>> t2=(0:1/fs:0.9)';
>> s1=sin(2*pi*440*t1);
>> s2=cos(2*pi*523*t2);
>> smix=s1+s2;
??? エラー ==> plus
行列の次元は一致しなければなりません。
```

s1,s2 の長さを揃えるには、長い方を切り詰めて短くするか、短い方に 0 の要素を追加して長くするかのいずれかである。

まず長い方を切り詰めて短くする場合を考える。この場合、s2 の方が短いので、s1 の最後の部分を切り詰めることにする。

```
>> ss1=s1(1:_______);
>> smix=ss1+s2;
>> soundsc(smix,fs)
```

問題なく聞くことができる。

次に短い方に 0 の要素を追加することを考える。

0 が要素のの列ベクトルを作るには、zeros という関数を用いる。

```
>> zeros(5,1)
ans =
     0
     0
     0
     0
     0
```

これを用いると、次のようになる。

```
>> ss2=[s2; zeros(_____________________,1)];
>> smix2=s1+ss2;
>> soundsc(smix2,fs)
```

重ねた音は、soundsc 関数では問題なく再生できるが、sound 関数では歪(ひず)んでしまう。また、保存するときも、そのままでは歪んでしまう。

サウンドデータは、wavwrite 関数で.wav ファイルに保存できる。

```
>> wavwrite(smix,'smix.wav')
警告: Data clipped during write to file:smix.wav
```

このような警告が出た場合には歪んでしまっている。

この警告に書かれている 'clip' という現象は、時間波形の変位が -1 以下、もしくは 1 以上になった場合に生じてしまう。

歪みが生じないようにするには、重ねた音の変位が -1 から 1 の範囲にはいるようにすればよい。

```
>> smix=smix/_____________;
```

このようにすれば歪まなくなる。

4 振幅の加工

音の大きさを単に変更する場合、音声データ全体を定数倍すればよい。

```
>> smix=s1+2*s2;
```

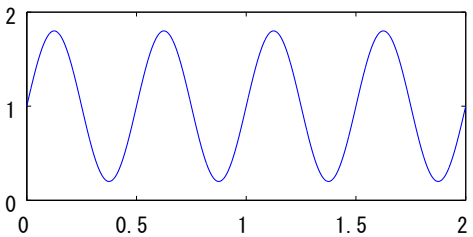
このようにすると、s2 の成分が強められる。

音の大きさを徐々に変化させたい場合は、倍率を時間によって変化させればよい。

例えば、次のようにすると、音は徐々に小さくなる。

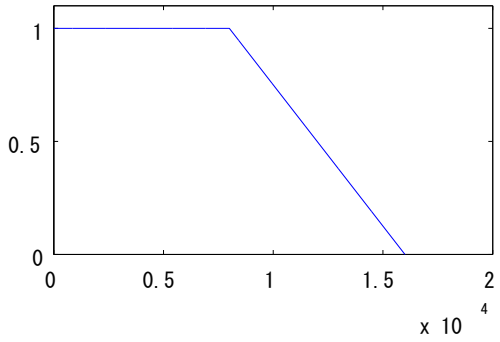
```
>> fs=8000;
>> t=(0:1/fs:2)';
>> s=sin(2*pi*440*t);
>> sound(s,fs)
>> e=1-t/2;
>> plot(t,e)
>> s1=s.*e;
>> plot(s1)
>> soundsc(s1,fs)
```

この e の関数を変えると、変化の仕方もいろいろと変わる。



このような関数を作成して、上記の e のように信号にかけあわせると、いわゆるトレモロという効果が得られる。

また、部分ごとに大きさを変えることもできる。1 秒のところまではそのままで、その後、徐々に小さくするには、次のような関数を作成すればよい。



このような形で特定のフレーズや音符だけを強調することもできる。