5　音程の変更

5.1　構造体

作成した音を楽器音のように使うには、音程を変えられる方が便利である。そこで、音程を変えられるように前章の wavetable_synth1 を拡張して wavetable_synth2 という関数を作ることを考える。

その前に、Wavetable 合成の汎用性を増すために、音源波形をファイルで管理する方法を考える。音源波形は、wavetable_synth1 で利用した wavetable のように配列で表現する。この配列に関しては、データ以外にサンプリング周波数が必要である。また、名称も記録されていると便利である。

これらを一括して管理し、ファイルにも保存する方法に構造体を使う方法がある。MATLAB の構造体とは、多次元の配列であり、文字列のフィールド名を用いて対応する値を参照したり変更したりできるデータ構造である。

```
>> piano.wavetable = wavetable;
>> piano.fs = 48000;
>> piano.name = 'piano';
>> piano

piano =

         name: 'piano'
    wavetable: [68x4 double]
           fs: 48000
```

wavetable は手作業で作成した。何度も同じ作業をするのを避けるためには、このデータをファイルに保存するのがよい。MATLAB では、ワークスペースの変数をファイルに保存できる。そのための関数は save である。

```
>> save wavetable_piano piano
```

このようにすると、wavetabe_piano.mat というファイルに構造体 piano が保存される。

練習 5 lookfor コマンドや help コマンドを利用して、save した .mat ファイルをワークスペースに読み込む関数を調べてみよ。

5.2　リサンプルによるピッチの加工

Wavetable 合成では、1 周期分のデータを用いて楽器音を生成する。したがって、このデータの長さで、生成される音程が決まる。

ここでは、このデータの長さを変更することで、音程を変更することを考える。

前回資料で作成した wavetable_synth1 の冒頭部分

```
tau = length(table); % 音源の周期を基本周期とする
```

を次のように変更することで、周波数を変更してみる。

```
tau = length(table); % 音源の周期を基本周期とする
tau = round(tau/2);
parts = size(format,1);
for i = 1:parts,
    tab(:,i) = table(1:tau,i);
end
table = tab;
```

このようにすると、1 オクターブ高い音が生成される。しかし、生成された波形を拡大して観察してみるとわかるが、このような強引な方法では、音色が変化してしまうし、音程を (1) することができない。音色を変化させず、自由に音程を上下に変化させるためには、音源波形の形を保ちつつ周期を変化させなければならない。

練習 6 下線部 (1) を埋めよ。また、なぜ、そうなるか理由を考えてみよ。

MATLAB で音源波形の形を保ちつつ周期を変化させる一番簡単な方法は、resample を利用する方法である。(簡単ではあるが、計算量の観点からは、必ずしも最善の方法ではないことに留意すること)。

音程に対応する周波数を計算できれば (詳細は、2007 年の資料「MATLAB 入門 (1)」(http://www.slp.k.hosei.ac.jp/~itou/lecture/2007/ProjectB/matlab-04.pdf) を参考にすること) resample を利用して、wavetable_synth1 の冒頭部分を次のように修正すると音程を指定して生成できるようになる (resample の使い方は、help で調べること)。

```
function wave = wavetable_synth2(num, table, format, fs)
%WAVETABLE_SYNTH1 wavetable 合成法の簡単な実装
% wavetable に格納されたデータのピッチ通りに
% ADSR エンベロープで振幅を変化させて合成する
% num は、69 が 440Hz の A を表し、半音ごとに1変化する値
% format は以下の行列
% [attack_time(秒) attack_level(1.0を最大とする相対値);
%  decay_time(秒) decay_level(1.0を最大とする相対値);
%  sustain_time(秒) sustain_level(1.0を最大とする相対値);
%  release_time(秒) 0]
tau = round(------------------------------------); % 作成する音の基本周期
base = length(table); % 音源の周期
for i = 1:4,
    tab(:,i) = resample(table(:,i), -------, ------);
end
table = tab;
```

この wavetable_synth2 と前節の構造体 piano を利用すると次のようにして音を生成できる。

```
>> soundsc(wavetable_synth2(60,piano.wavetable,[0.025 1; 0.04 0.5; 0.3 0.015; 1.0 0], piano.fs),
piano.fs);
>>
```

(入力時には改行しないこと。また引数はあくまで例なので、各自適切な値を指定すること)