第6章

音発生用の正弦波発振器のアナログ処理を四則演算で置き換えよう

アナログ処理の簡単なディジタル化手法

これまでは，アナログ信号を対象とするフィルタ回路について解説してきた．一方，通信・放送，画像，音声，制御などのあらゆるアナログ信号がディジタル化されるに伴い，ディジタル信号処理技術が脚光を浴びている．ディジタル信号処理は，「アナログ信号を数値データに置き換えて，四則演算する」だけなのであるが，従来のアナログ処理では不可能であったアプリケーションも実現できるようになり，信号処理分野での基本技術として大いに評価されている．ここでは，微分方程式や積分方程式で記述したものや，音発生用の正弦波発振器（着メロ）のアナログ処理を四則演算で置き換える，すなわちディジタル処理で実現するための基本的な考え方を紹介する．（筆者）

1 微分方程式のディジタル化

いま，**図1**のアナログ・フィルタ（*RC*回路でローパス・フィルタ）を数値演算処理で実現すること（ディジタル化）を考えてみましょう．**第3章**の**4**「信号の入出力と微積分方程式」より，アナログ入力信号$x(t)$とアナログ出力信号$y(t)$との関係は，

$$CR\frac{dy(t)}{dt}+y(t)=x(t) \quad \cdots\cdots(1)$$

という微分方程式で表されます．

さて，**図1**のアナログ・フィルタをディジタル数値演算で置き換えるポイントは簡単なことで，入力および出力信号を単純明快にディジタル化するだけなのです．つまり，式(1)の微分方程式をディジタル化するには，$t=t_0$の時刻でサンプリングするので，$t=t_0$を代入して，

$$CR\left.\frac{dy(t)}{dt}\right|_{t=t_0}=-y(t_0)+x(t_0) \quad \cdots\cdots(2)$$

となる関係が得られます．左辺の$t=t_0$の時刻における微分値は，

$$\left.\frac{dy(t)}{dt}\right|_{t=t_0}=\lim_{\Delta\to 0}\frac{y(t_0)-y(t_0-\Delta)}{\Delta} \quad \cdots\cdots(3)$$

で定義されます（**図2**）．

ここで，アナログ出力信号$y(t)$がT［秒］ごとにサンプリングされるとすれば，連続時間tをT［秒］ごとの飛び飛びの時刻で考えることになるので，$t_0=kT$，$\Delta=T$とおけば，式(3)は，

$$\left.\frac{dy(t)}{dt}\right|_{t=kT}\cong\lim_{T\to 0}\frac{y(kT)-y(kT-T)}{T} \quad \cdots\cdots(4)$$

と置き換えられます．このとき，サンプリング間隔Tを0にした極限値が微分に相当しますが，サンプリングによりディジタル化されているので極限をとらないことにすれば，

$$\left.\frac{dy(t)}{dt}\right|_{t=kT}\cong\frac{y[k]-y[k-1]}{T} \quad \cdots\cdots(5)$$

ただし，$y[k]=y(kT)$, $y[k-1]=y(kT-T)=y\big((k-1)T\big)$

と表されます．よって，式(2)，(5)より，

$$CR\frac{y[k]-y[k-1]}{T}=-y[k]+x[k],\ x[k]=x[kT]$$

となり，

$$y[k]=ay[k-1]+bx[k] \quad \cdots\cdots(6)$$

ただし，$a=\frac{1}{1+\tau}$, $b=\frac{\tau}{1+\tau}$, $\tau=\frac{T}{CR}$

と変形されます．ここで，式(6)の表現形式は，差分方程式と

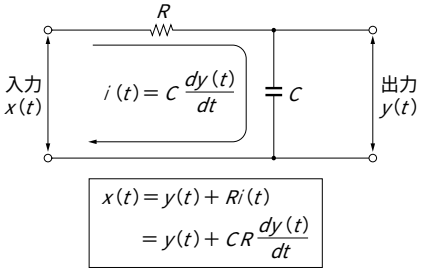

図1
アナログ・フィルタの微分方程式による表現

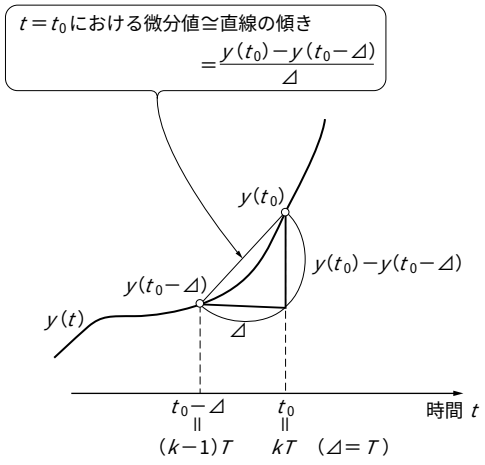

図2
時間微分の定義

KEYWORD ── 微分方程式のディジタル化，積分方程式のディジタル化，音発生用発振器のディジタル化

呼ばれています(次章で詳述).

式(6)のディジタル・フィルタは高校で学習する無限級数の漸化式に類似するもので,

$$y[1]=ay[0]+bx[1]$$
$$y[2]=ay[1]+bx[2]$$
$$y[3]=ay[2]+bx[3]$$
$$\vdots$$

のように,次々と計算を進めて出力値が得られるのですから,**図1**に示すアナログ・フィルタのディジタル版ということになります.

このように,微分方程式を四則演算で置き換えることにより,アナログ素子(抵抗,コンデンサ,コイルなど)で構成されるフィルタ回路を,四則演算による数値計算処理で実現できるわけで,何となく不思議な感じがしますね(詳細は次章で解説).

いま,

$$\tau=\frac{T}{CR}=1$$

とすれば,式(6)の係数は$a=b=0.5$となり,

$$y[k]=0.5y[k-1]+0.5x[k] \quad \cdots\cdots(7)$$

で計算されます.式(7)の計算処理をScilabプログラム(これ以後,DSPプログラムと略す)で作成して,いろいろな波形(キーボードから入力)に対する応答出力を調べてみましょう.そこで,**プログラム例1**をDSPプログラムで作成し,適当なファイル名(たとえば,prog61.sce)を付けてディレクトリCQfilterに保存してください.このとき,ファイル保存するディレクトリをまちがえないように注意が必要です.

プログラム例1(ディジタル・フィルタ)

```
function[youtput] = FILTER(xinput)      ……①
//
//******************** 初期化 *****************
        y1=0;                           ……②
//*****************************************
//
   for k = 1:1:length(xinput)
        xin = xinput(k);                ……③
//
//**** 四則計算によるディジタル出力の算出 ****
   yout = 0.5*y1 + 0.5*xin;             ……④
   y1 = yout;                           ……⑤
//*****************************************
//
   youtput(k) = yout;                   ……⑥
   end;
endfunction
```

[プログラム例1の説明]

ディジタル・フィルタのDSPプログラムは,この例を下敷きにして作成します.ここで,入力信号$x[k]$の変数名はxin,出力信号$y[k]$の変数名はyoutで,いろいろな関数コマンドを利用可能とするために,変更はできないことに注意してください.

また,DSPプログラムにおいては,

② 出力信号の初期化処理($y[-1]=0$)
④ 出力信号$y[k]$の計算
⑤ 出力信号を遅らせる処理$y[k-1]$

の三つの計算が実行されます.

関数コマンドfilinを実行すると,保存したDSPプログラムに基づき,入力波形をキーボードから入力することにより出力値が計算され,入力信号(上段,ピンク色)と出力信号(下段,緑色)の二つの波形グラフが表示されます(**実行例1**,**図3**).**図3**に示す具体的な数値例を参考に,自由にいろいろな値を入力して試してみましょう.

実行例1

```
-->sfrq=100;↵                  ……①
-->tmax=50;↵                   ……②
-->filin('prog61',1);↵         ……③
```

[実行例1の説明]

[関数コマンド1(入力信号(キーボード入力)に対する応答計算)]

filin(ファイル名,ウィンドウ画面)

ここで,ファイル名として拡張子.sceをはずして入力します.

① サンプリング周波数を設定する
② 画面表示サンプル数を設定する
③ 入力波形をキーボードから入力することにより,フォルダCQfilterに保存したファイル名prog61.sceのディジタル・フィルタの出力信号値が計算され,入出力波形をグラフ表示する

2 積分方程式のディジタル化

こんどは,**図1**の回路の抵抗RとコンデンサCを入れ替えたアナログ・フィルタをディジタル化してみましょう(**図4**).

まず,抵抗Rに流れる電流は,オームの法則より,

$$i(t)=\frac{y(t)}{R} \quad \cdots\cdots(8)$$

であり,電流$i(t)$がコンデンサに流れて蓄積されることから,静電容量C[F]を用いて,キルヒホッフの法則より,

$$y(t)+\frac{1}{C}\int_{-\infty}^{t} i(\tau)d\tau=x(t) \quad \cdots\cdots(9)$$

となります.式(8)を式(9)に代入すれば,最終的に入出力信号の関係として,