# 章末問題解答

## 1 章

【1】 **(1)** $T_s = 1/f_s = 1\,\mathrm{ms}$

**(2)** $f_n = f_s/2 = 500\,\mathrm{Hz}$

**(3)** $\{1210\,1210\,1210\cdots\}$

**(4)** 500 Hz より高い周波数帯域

**(5)** 150 Hz

【2】 **(1)** $(01101.1010)_2$

**(2)** $(10000.00101)_2 \simeq (10000.0011)_2$

**(3)** $(00111.1\dot{0}11\dot{0})_2 \simeq (00111.1011)_2$

**(4)** $(11011.0\dot{0}01\dot{1})_2 \simeq (11011.0010)_2$

## 2 章

【1】 **(1)**

$$v(nT_s)=x(nT_s)+0.98v\{(n-1)T_s\}$$
$$y(nT_s)=0.01\,[v(nT_s)+v\{(n-1)T_s\}]$$

**(2)** 1 次 IIR フィルタ

**(3)** 遅延器への入力信号 $v(nT_s)$ は $v(nT_s)=(0.98)^n$ であるので，因果性を考慮すると，インパルス応答は

$$h(nT_s)=\begin{cases} 0.01 & (n=0) \\ 0.01\left\{(0.98)^n+(0.98)^{n-1}\right\} & (n \geqq 1) \end{cases}$$

となる。

**(4)** 図 2.5 と図 2.6 に示したプログラム `2iirView.cpp` を以下に示すように変更する。

`iirexc.exe`

① 遅延器の個数を設定する行を

```
#define M 2
```

から

```
#define M 1
```

に変更する。

② 関数 Wave(CDC* pDC ) の乗算係数を設定する行を

```
a[0]=0.049283;
a[1]=0.098566;
a[2]=0.049283;
b[1]=1.2813;
b[2]=-0.47844;
```

から

```
a[0]=0.01;
a[1]=0.01;
b[1]=0.98;
```

に変更する。

③ 出力信号波形の表示倍率を 10 倍

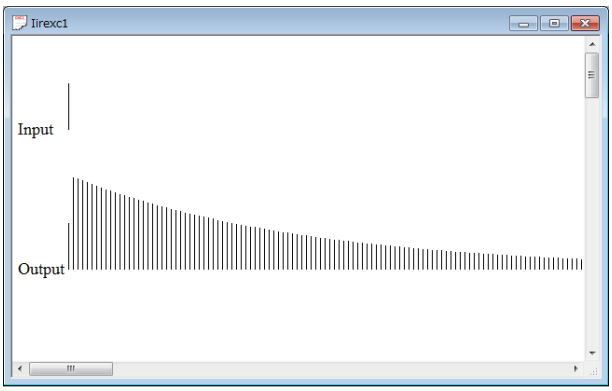
```
SignalMag1(pDC, i, x);
SignalMag2(pDC, i, 10.0*y);
```

から 100 倍

```
SignalMag1(pDC, i, x);
SignalMag2(pDC, i, 100.0*y);
```

に変更する。

インパルス応答を解図 **2.1** に示す。




解図 **2.1** ディジタルフィルタのインパルス応答

**(5)**

$$\sin\{\omega(n-k)T_s\} = \frac{e^{j\omega(n-k)T_s} - e^{-j\omega(n-k)T_s}}{j2}$$

を用いると，出力信号は

$$\begin{aligned}
y(nT_s) =& \sum_{k=0}^{n} \sin\{\omega(n-k)T_s\} h(kT_s) \\
=& \frac{0.01}{j2} \left[ \sum_{k=0}^{n} (0.98)^k \left\{ e^{j\omega(n-k)T_s} - e^{-j\omega(n-k)T_s} \right\} \right. \\
& \left. + \sum_{k=1}^{n} (0.98)^{k-1} \left\{ e^{j\omega(n-k)T_s} - e^{-j\omega(n-k)T_s} \right\} \right] \\
=& \frac{0.01}{j2} \left\{ e^{j\omega nT_s} \sum_{k=0}^{n} (0.98e^{-j\omega T_s})^k - e^{-j\omega nT_s} \sum_{k=0}^{n} (0.98e^{j\omega T_s})^k \right. \\
& \left. + \frac{e^{j\omega nT_s}}{0.98} \sum_{k=1}^{n} (0.98e^{-j\omega T_s})^k - \frac{e^{-j\omega nT_s}}{0.98} \sum_{k=1}^{n} (0.98e^{j\omega T_s})^k \right\} \\
=& \frac{0.01}{1.9604 - 1.96\cos\omega T_s} \{0.02(1+\cos\omega T_s)\sin\omega nT_s \\
& -1.98\sin\omega T_s \cos\omega nT_s + 1.98(0.98)^n \sin\omega T_s\}
\end{aligned}$$

となる。

**(6)** 十分に時間が経過すると，**(5)** で求めた出力信号 $y(nT_s)$ の右辺第 3 項は

$$1.98(0.98)^n \sin\omega T_s \to 0$$

となるので，定常応答は

$$\begin{aligned}
y(nT_s) =& \frac{0.01}{1.9604 - 1.96\cos\omega T_s} \{0.02(1+\cos\omega T_s)\sin\omega nT_s \\
& -1.98\sin\omega T_s \cos\omega nT_s\} \\
=& 0.01\sqrt{\frac{2(1+\cos\omega T_s)}{1.9604 - 1.96\cos\omega T_s}} \sin\left\{\omega nT_s \right. \\
& \left. - \tan^{-1}\frac{1.98\sin\omega T_s}{0.02(1+\cos\omega T_s)}\right\}
\end{aligned}$$

となる。

**(7)** 直流と扱える最高周波数における振幅は，$\omega = 0$ と $\omega = \pi/T_s$ を

$$0.01\sqrt{\frac{2(1+\cos\omega T_s)}{1.9604 - 1.96\cos\omega T_s}}$$

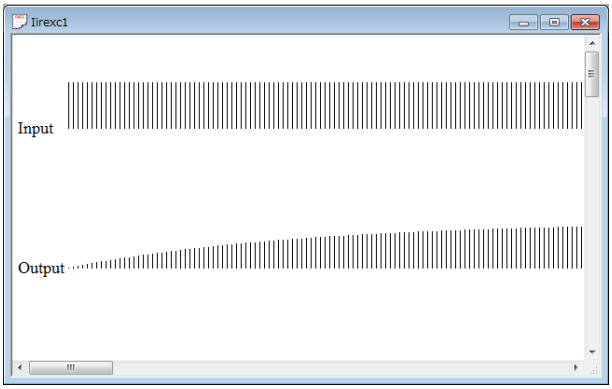
に代入すると，それぞれ 1.0 倍と 0.0 倍になる。

**(8)** 単位ステップ $u(nT_s)$ を図 2.18 の 1 次 IIR フィルタに入力すると，**(1)** で求めた差分方程式より，遅延器への入力信号 $v(nT_s)$ は，初項 1，公比 0.98 の等比級数 $v(nT_s) = 1 + 0.98 + (0.98)^2 + \cdots + (0.98)^n$ となる。等比級数の和を求めると $v(nT_s) = 50\{1 - (0.98)^{n+1}\}$ となるので，ステップ応答 $y(nT_s)$ は，同差分方程式を用いて

$$y(nT_s) = 0.5\{2 - (0.98)^n - (0.98)^{n+1}\}$$

となる。

**(9)** **(4)** で作成したプログラムにおいて，入力信号であった単位インパルス関数を単位ステップ関数に，出力信号波形の表示倍率を 100 倍から 1 倍にそれぞれ変更すればよい。ステップ応答を解図 **2.2** に示す。

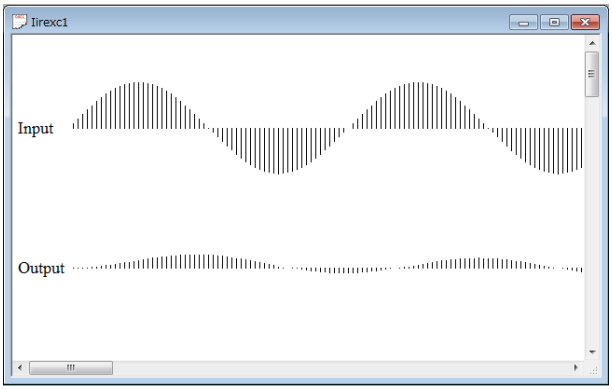



解図 **2.2** ディジタルフィルタのステップ応答

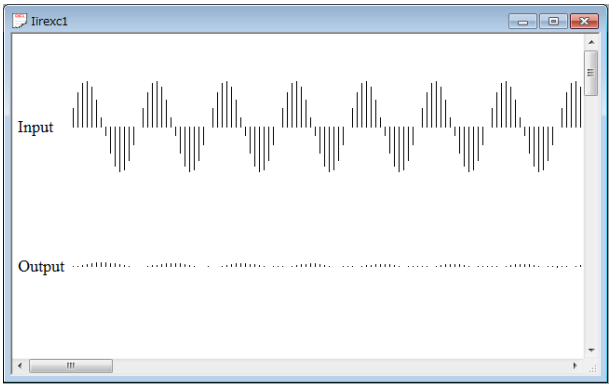

`iirexc.exe`

**(10)** **(4)** で作成したプログラムにおいて，入力信号であった単位インパルス関数を正弦波に変更するとともに，出力信号波形の表示倍率を 100 倍から 1 倍に変更すればよい。正弦波の応答を解図 **2.3** と解図 **2.4** に示す。


`iirexc.exe`




解図 **2.3** ディジタルフィルタの過渡応答 (正弦波 500 Hz)




解図 **2.4** ディジタルフィルタの過渡応答 (正弦波 2 kHz)

## 3 章

**【1】** **(1)** 2 章章末問題**【1】(1)** より，差分方程式は

$$v(nT_s)=x(nT_s)+0.98v\{(n-1)T_s\}$$
$$y(nT_s)=0.01\,[v(nT_s)+v\{(n-1)T_s\}]$$

であった。$V(z)=\mathcal{Z}[v(nT_s)]$，$X(z)=\mathcal{Z}[x(nT_s)]$，$Y(z)=\mathcal{Z}[y(nT_s)]$ とすると，上式の $z$ 変換は

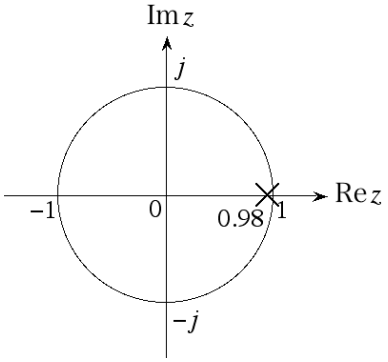
$$V(z)=X(z)+0.98z^{-1}V(z)$$
$$Y(z)=0.01V(z)+0.01z^{-1}V(z)$$

となる。上式と式 (3.44) を用いると，伝達関数は

$$H(z)=\frac{Y(z)}{X(z)}=\frac{V(z)}{X(z)}\cdot\frac{Y(z)}{V(z)}$$
$$=\frac{1}{1-0.98z^{-1}}\cdot 0.01(1+z^{-1})=\frac{0.01(1+z^{-1})}{1-0.98z^{-1}}$$

となる。

**(2)** **(1)** で求めた伝達関数の分母を $1-0.98z^{-1}=0$ とすると，極は $z=0.98$ となる。解図 **3.1** に極の位置を示す。




解図 **3.1**　極の位置

**(3)** 極は正の実数軸上の単位円内に存在するので，インパルス応答は $(0.98)^n$ のように指数関数的に減少すると推定できる。**(1)** より，理論的に求めたインパルス応答は

$$h(nT_s) = \begin{cases} 0.01 & (n=0) \\ 0.01\left\{(0.98)^n + (0.98)^{n-1}\right\} & (n \geqq 1) \end{cases}$$

であったので，指数関数的に減少することに関しては，極の位置から推定したインパルス応答とほぼ一致することが確認できる。

**(4)** 表 3.1 より，正弦波の $z$ 変換は

$$\begin{aligned} X(z) &= \mathcal{Z}[x(nT_s)] = \frac{z^{-1}\sin\omega T_s}{1 - 2z^{-1}\cos\omega T_s + z^{-2}} \\ &= \frac{1}{j2}\frac{z^{-1}(e^{j\omega T_s} - e^{-j\omega T_s})}{(1 - e^{j\omega T_s}z^{-1})(1 - e^{-j\omega T_s}z^{-1})} \end{aligned}$$

であるので，式 (3.22) のたたみ込み演算を用いると，出力信号 $y(nT_s)$ の $z$ 変換 $Y(z)$ は

$$\begin{aligned} Y(z) &= H(z)X(z) = \frac{0.01(1 + z^{-1})}{1 - 0.98z^{-1}} \\ &\quad \cdot \frac{1}{j2}\frac{z^{-1}(e^{j\omega T_s} - e^{-j\omega T_s})}{(1 - e^{j\omega T_s}z^{-1})(1 - e^{-j\omega T_s}z^{-1})} \end{aligned}$$

となる。部分分数展開，逆 $z$ 変換の線形性，表 3.1 を使用して $Y(z)$ の逆 $z$ 変換を求める。$Y(z)$ を部分分数

$$\begin{aligned} Y(z) &= \frac{0.01}{j2}\frac{1 + z^{-1}}{1 - 0.98z^{-1}}\frac{z^{-1}(e^{j\omega T_s} - e^{-j\omega T_s})}{(1 - e^{j\omega T_s}z^{-1})(1 - e^{-j\omega T_s}z^{-1})} \\ &= D_0 + \frac{D_1}{1 - 0.98z^{-1}} + \frac{D_2}{1 - e^{j\omega T_s}z^{-1}} + \frac{D_3}{1 - e^{-j\omega T_s}z^{-1}} \end{aligned}$$

に展開する。ここで，係数 $D_0$，$D_1$，$D_2$，$D_3$ はそれぞれ

$$D_0 = Y(z)\Big|_{z=0} = 0$$

$$\begin{aligned} D_1 &= (1 - 0.98z^{-1})Y(z)\Big|_{z=0.98} \\ &= \frac{0.01}{j2}\frac{1.98(e^{j\omega T_s} - e^{-j\omega T_s})}{(0.98 - e^{j\omega T_s})(0.98 - e^{-j\omega T_s})} \end{aligned}$$

$$D_2 = (1 - z^{-1}e^{j\omega T_s})Y(z)\Big|_{z=e^{j\omega T_s}} = \frac{0.01}{j2}\frac{e^{j\omega T_s} + 1}{e^{j\omega T_s} - 0.98}$$

$$D_3 = (1 - z^{-1}e^{-j\omega T_s})Y(z)\Big|_{z=e^{-j\omega T_s}} = -\frac{0.01}{j2}\frac{e^{-j\omega T_s}+1}{e^{-j\omega T_s}-0.98}$$

と求まる。求めた係数を代入すると

$$\begin{aligned}Y(z)=&\frac{0.01}{j2}\left\{\frac{1.98(e^{j\omega T_s}-e^{-j\omega T_s})}{(0.98-e^{j\omega T_s})(0.98-e^{-j\omega T_s})}\frac{1}{1-0.98z^{-1}}\right.\\&+\frac{e^{j\omega T_s}+1}{e^{j\omega T_s}-0.98}\frac{1}{1-e^{j\omega T_s}z^{-1}}\\&\left.-\frac{e^{-j\omega T_s}+1}{e^{-j\omega T_s}-0.98}\frac{1}{1-e^{-j\omega T_s}z^{-1}}\right\}\end{aligned}$$

を得る。表 3.1 を利用して $Y(z)$ の両辺を逆 $z$ 変換すると，出力信号 $y(nT_s)$ を

$$\begin{aligned}y(nT_s)=&\frac{0.01}{j2}\left\{\frac{1.98(e^{j\omega T_s}-e^{-j\omega T_s})}{(0.98-e^{j\omega T_s})(0.98-e^{-j\omega T_s})}(0.98)^n\right.\\&\left.+\frac{e^{j\omega T_s}+1}{e^{j\omega T_s}-0.98}e^{j\omega nT_s}-\frac{e^{-j\omega T_s}+1}{e^{-j\omega T_s}-0.98}e^{-j\omega nT_s}\right\}\\=&\frac{0.01}{1.9604-1.96\cos\omega T_s}[1.98(0.98)^n\sin\omega T_s\\&-0.98\sin\{\omega(n+1)T_s\}+0.02\sin\omega nT_s+\sin\{\omega(n-1)T_s\}]\\=&\frac{0.01}{1.9604-1.96\cos\omega T_s}\{0.02(1+\cos\omega T_s)\sin\omega nT_s\\&-1.98\sin\omega T_s\cos\omega nT_s+1.98(0.98)^n\sin\omega T_s\}\end{aligned}$$

と求めることができる。

**(5)** **(4)** で求めた $y(nT_s)$ において，過渡応答を表す右辺第 3 項は

$$\lim_{n\to\infty}1.98(0.98)^n\sin\omega T_s = 0$$

となるので，伝達関数が安定なディジタルフィルタに正弦波を入力すると，出力信号は発散しないことが確認できる。また，同時に，**(4)** で求めた $y(nT_s)$ は，2 章章末問題 **(5)** で求めた出力信号と一致することも確認することができる。

**(6)** **(1)** で求めた伝達関数に $z = e^{j\omega T_s}$ を代入すると，システム関数 $H(\omega)$ は

$$\begin{aligned}H(\omega)=&\frac{0.01(1+z^{-1})}{1-0.98z^{-1}}\Big|_{z=e^{j\omega T_s}}=\frac{0.01(1+e^{-j\omega T_s})}{1-0.98e^{-j\omega T_s}}\\=&\frac{0.01(1+\cos\omega T_s-j\sin\omega T_s)}{1-0.98\cos\omega T_s+j0.98\sin\omega T_s}\end{aligned}$$

となる。または，上式を有理化すると，システム関数 $H(\omega)$ を

$$\begin{aligned}H(\omega)=&\frac{0.01(1+\cos\omega T_s-j\sin\omega T_s)}{1-0.98\cos\omega T_s+j0.98\sin\omega T_s}\\=&0.01\frac{0.02(1+\cos\omega T_s)-j1.98\sin\omega T_s}{1.9604-1.96\cos\omega T_s}\end{aligned}$$

と求めることもできる。

**(7)** **(6)** で求めたシステム関数より，周波数特性 (振幅特性と位相特性) は

$$\begin{aligned}H(\omega)=&0.01\sqrt{\frac{(1+\cos\omega T_s)^2+\sin^2\omega T_s}{(1-0.98\cos\omega T_s)^2+0.98^2\sin^2\omega T_s}}\\&\angle-\tan^{-1}\frac{\sin\omega T_s}{1+\cos\omega T_s}-\tan^{-1}\frac{0.98\sin\omega T_s}{1-0.98\cos\omega T_s}\\=&0.01\sqrt{\frac{2(1+\cos\omega T_s)}{1.9604-1.96\cos\omega T_s}}\\&\angle-\tan^{-1}\frac{\sin\omega T_s}{1+\cos\omega T_s}-\tan^{-1}\frac{0.98\sin\omega T_s}{1-0.98\cos\omega T_s}\end{aligned}$$

となる。または

$$\begin{aligned}H(\omega)=&0.01\frac{\sqrt{0.02^2(1+\cos\omega T_s)^2+1.98^2\sin^2\omega T_s}}{1.9604-1.96\cos\omega T_s}\\&\angle-\tan^{-1}\frac{1.98\sin\omega T_s}{0.02(1+\cos\omega T_s)}\\=&0.01\sqrt{\frac{2(1+\cos\omega T_s)}{1.9604-1.96\cos\omega T_s}}\angle-\tan^{-1}\frac{1.98\sin\omega T_s}{0.02(1+\cos\omega T_s)}\end{aligned}$$

と求めることもできる。

**(8)** **(7)** で求めたシステム関数を使用して振幅特性を計算する。図 3.7～ 図 3.9 のプログラムを以下のように変更する。

`iirexc.exe`

① グローバル変数

```
static char yaxes[][4]={"  0","-10","-20","-30"};
```

を

```
static char yaxes[][5]={"   0"," -40"," -80","-120"};
```

に変更する。

② 関数 LineMag() の

```
pDC->LineTo(100.+70./3.*x,50.-8.*y);
```

を

```
pDC->LineTo(100.+70./3.*x,50.-80./40.*y);
```

に変更する。

③ 関数 Magnitude() においてシステム関数の分子に複素数を導入するために，配列宣言

```
double numerator,numerator_real;
double numerator_imaginary;
```

を

```
double denominator;
```

の後に付け加える。

④ 乗算係数

```
a[0] = 0.1;
b[1] = 0.9;
```

を

```
a[0] = 0.01;
a[1] = 0.01;
b[1] = 0.98;
```

に変更する。

⑤ システム関数の分母では複素数を使用しないので

```
denominator_real=1.0-b[1]*cos(omega*ts);
denominator_imaginary=b[1]*sin(omega*ts);
denominator=pow(denominator_real,2.0);
denominator+=pow(denominator_imaginary,2.0);
```

を

```
denominator=1.0+b[1]*b[1]-2.0*b[1]*cos(omega*ts);
```

に変更する。

⑥ システム関数の分子の実数部と虚数部に値を設定するために，⑤で変更したシステム関数の分母を計算する行

```
denominator=1.0+b[1]*b[1]-2.0*b[1]*cos(omega*ts);
```

の後に

```
numerator_real=(a[0]-a[1]*b[1])+(a[1]-a[0]*b[1])*
cos(omega*ts);
numerator_imaginary=-(a[1]+a[0]*b[1])*sin(omega*ts);
```

を付け加える。

⑦ システム関数の分子の大きさを計算するために，⑥ で付け加えた

システム関数の分子の実数部と虚数部に値を設定する行

```
numerator_real=(a[0]-a[1]*b[1])+(a[1]-a[0]*b[1])*
cos(omega*ts);
numerator_imaginary=-(a[1]+a[0]*b[1])*sin(omega*ts);
```

の後に

```
numerator=pow(numerator_real,2.0);
numerator+=pow(numerator_imaginary,2.0);
```

を付け加える。

⑧ 振幅を計算する行を

```
amplitude=a[0]/sqrt(denominator);
```

から

```
amplitude=sqrt(numerator)/denominator;
```

に変更する。

振幅特性を解図 **3.2** に示す。図 2.18 の IIR フィルタは，図 2.9 のディジタルフィルタに比べ高域な周波数帯において大きな減衰量を確保できることがわかる。**(7)** より，振幅特性 $|H(\omega)|$ は

$$|H(\omega)| = 0.01\sqrt{\frac{2(1+\cos\omega T_s)}{1.9604 - 1.96\cos\omega T_s}}$$

であるので，標本化定理より，扱える最高角周波数 $\pi/T_s$ における理論的な振幅値は

$$\left|H\left(\frac{\pi}{T_s}\right)\right| = 0$$

となる。上式を電圧利得表示すると

$$20\log_{10}\left|H\left(\frac{\pi}{T_s}\right)\right| = -\infty$$

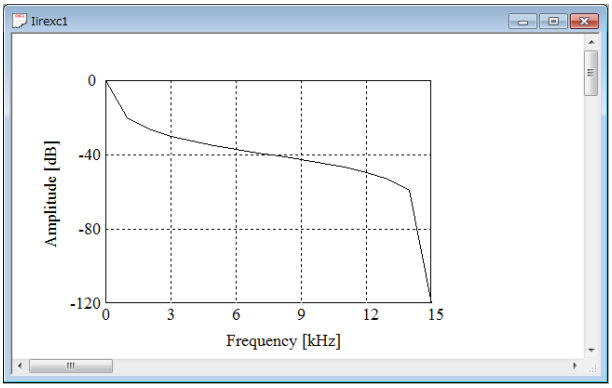
となる。パソコンによる振幅特性の表示においても，プログラム中の標本化周期の有効桁数を高めると理論値に近い振幅値を得ることができる。

**(9)** **(7)** で示したシステム関数を使用して位相特性を計算する。**(8)** で作成したプログラムを以下のように変更する。

iirexc.exe

① グローバル変数

```
static char yaxes[][5]={"   0"," -40"," -80","-120"};
```




解図 **3.2** ディジタルフィルタの振幅特性

を

```
static char yaxes[][5]={"  0","-0.6","-1.2","-1.8"};
```

に変更する。

② 関数 LineMag() の

```
pDC->LineTo(100.+70./3.*x,50.-80./40.*y);
```

を

```
pDC->LineTo(100.+70./3.*x,50.-80./0.6*y);
```

に変更する。

③ 関数名

```
Magnitude( CDC* pDC )
```

を

```
Phase( CDC* pDC )
```

に変更する。

④ 関数 Phase() においてシステム関数の分子の大きさを計算する行

```
numerator=pow(numerator_real,2.0);
numerator+=pow(numerator_imaginary,2.0);
```

を削除する。

⑤ 位相特性を計算するために，振幅とその電圧利得表示を計算する行

```
amplitude=sqrt(numerator)/sqrt(denominator);
amplitude=20.0*log10(amplitude);
```

を

```
phase=atan(numerator_imaginary/numerator_real);
```

に変更する。

⑥ 位相特性を表示するために，電圧利得表示された振幅を表示する行

```
LineMag( pDC, frequency, amplitude );
```

を

```
LineMag( pDC, frequency, phase );
```

に変更する。

⑦ **(8)** で作成したプログラム名が dspView.cpp であるならば，ファイル dspView.h の関数

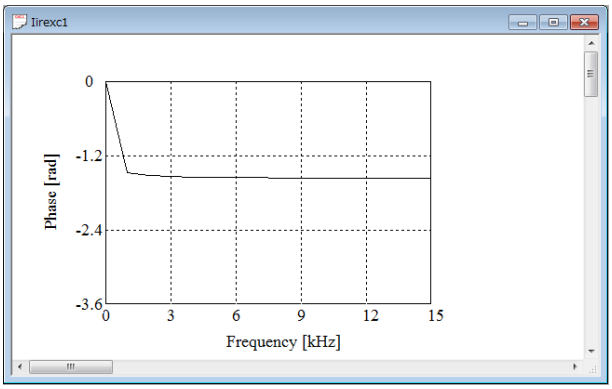
```
void Magnitude( CDC* );
```

を

```
void Phase( CDC* );
```

に変更する。

位相特性を解図 **3.3** に示す。出力信号は，高域な周波数帯において約 $\pi/2$ rad 遅れて出力されることがわかる。




解図 **3.3** ディジタルフィルタの位相特性

**(10)** **(8)** で作成したプログラムを以下のように変更する。


iirexc.exe

① グローバル変数

```
static char xaxes[][3]={"0","3","6","9","12","15"};
```

を

```
static char xaxes[][4]={"-60","-40","-20","0","20",
"40","60"};
```

に変更する。

② 関数 LineMag() の

```
pDC->LineTo(100.+70./3.*x,50.-80./40.*y);
```

を

```
pDC->LineTo(100.+65./20.*(x+60.),50.-80./40.*y);
```

に変更する。

③ 関数 Magnitude() の内容を以下に示すように変更する。
入力する正弦波の周波数範囲を設定する行を

```
for (int i=0; i<=15; i++){
```

から

```
for (int i=-24; i<=24; i++){
```

に変更する。

④ 周波数の単位を kHz から Hz に変換する行を

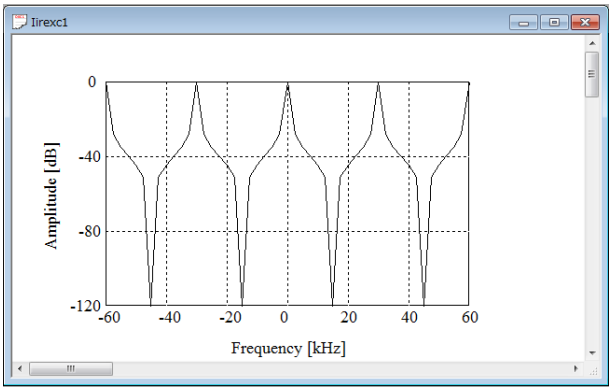
```
frequency=double(i)*1000.0;
```

から

```
frequency=double(i)*2500.0;
```

に変更する。この結果，正弦波の周波数を $-60 \sim 60$ kHz までの範囲で 2.5 kHz ずつ変化させることになる。

実行結果を解図 **3.4** に示す。正および負の周波数においても，標本化周波数 30 kHz ごとに振幅特性が繰り返すことが確認できる。**(8)** の場合と同様に，プログラム中の標本化周期の有効桁数を高めると理論値に近い振幅特性を得ることができる。




解図 **3.4** $-60$ kHz から 60 kHz の周波数帯の振幅特性

**4 章**

**【1】** **(1)** 離散時間信号系列より，周期は $4T_s$ であることがわかる。したがって，離散時間信号 $f(nT_s)$ の周期 $T$ は $T = NT_s = 4 \times 10^{-3}$ 秒 であるので，基本周波数 $f_0$ は $f_0 = 1/T = 250\,\mathrm{Hz}$ となる。図 4.2 と式 (4.13) を用いると，$f(nT_s)$ の DFT は

$$\begin{pmatrix} F(0) \\ F(250) \\ F(500) \\ F(750) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{pmatrix} \begin{pmatrix} f(0) \\ f(T_s) \\ f(2T_s) \\ f(3T_s) \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{pmatrix} \begin{pmatrix} -1.5 \\ 0.5 \\ -1.5 \\ -3.5 \end{pmatrix} = \begin{pmatrix} -6 \\ -j4 \\ 0 \\ j4 \end{pmatrix}$$

となる。

**(2)** 式 (4.6) を用いると

$$|F(0)| = 6, \quad |F(250)| = 4, \quad |F(500)| = 0, \quad |F(750)| = 4$$

である。

**(3)** $N = 4$ であるので，式 (4.19) の $k = 1$ のとき $F(750) = F(250)^*$ が成立することを確認すればよい。$F(250) = -j4$ と $F(750) = j4$ であるので，対称性が成り立つことがわかる。

**(4)** 周期 $4T_s$ の離散時間信号 $f(nT_s)$ を $2T_s$ 推移させた離散時間信号 $f\{(n+2)T_s\}$ の DFT は行列表現を用いて

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^4 & W_4^6 \\ 1 & W_4^3 & W_4^6 & W_4^9 \end{pmatrix} \begin{pmatrix} f(2T_s) \\ f(3T_s) \\ f(4T_s) \\ f(5T_s) \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{pmatrix} \begin{pmatrix} -1.5 \\ -3.5 \\ -1.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} -6 \\ j4 \\ 0 \\ -j4 \end{pmatrix}$$

となる。一方

$$W_4^{-2k} = e^{j\pi k} = \cos(\pi k) + j\sin(\pi k)$$

を $k$ 行 $k$ 列の要素とする対角行列を用いて式 (4.24) を行列表現すると，**(1)** の結果を用いて

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} -6 \\ -j4 \\ 0 \\ j4 \end{pmatrix} = \begin{pmatrix} -6 \\ j4 \\ 0 \\ -j4 \end{pmatrix}$$

となる。これらの結果が一致することから循環推移が成り立つことが確認できる。

**(5)** $f_0 = 250\,\mathrm{Hz}$ であるので，標本化周期 $T_s$ は $T_s = 1/(Nf_0) = 1\times10^{-3}$ 秒となる。図 4.5 と式 (4.36) を用いると，$F(250k)$ の IDFT は

$$\begin{aligned} \begin{pmatrix} f(0) \\ f(1\times10^{-3}) \\ f(2\times10^{-3}) \\ f(3\times10^{-3}) \end{pmatrix} &= \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4^{-1} & W_4^{-2} & W_4^{-3} \\ 1 & W_4^{-2} & W_4^{-4} & W_4^{-6} \\ 1 & W_4^{-3} & W_4^{-6} & W_4^{-9} \end{pmatrix} \begin{pmatrix} F(0) \\ F(250) \\ F(500) \\ F(750) \end{pmatrix} \\ &= \frac{1}{4} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{pmatrix} \begin{pmatrix} -6 \\ -j4 \\ 0 \\ j4 \end{pmatrix} = \begin{pmatrix} -1.5 \\ 0.5 \\ -1.5 \\ -3.5 \end{pmatrix} \end{aligned}$$

のように計算できる。このように，IDFT によって DFT 前の離散時間信号が復元できることが確認できた。

**【2】 (1)** 二つの正弦波の周波数は離れているが，周波数成分の大きさが大きく異なる離散時間信号のスペクトルをハミング窓とブラックマン窓を用いて分析する。

- ハミング窓　　ハミング窓を使用する場合には，プログラム `speana.cpp` に以下の変更を加える†。

`spanaexc.exe`

† 振幅スペクトルを画面上に描画するには，記載したほかにいくつかの追加・変更が必要です。詳しくは，4 章フォルダ内の `ReadMe(4 章)` ファイルを読んでください。

① ハミング窓関数 void Hamming( COMPLEX x[] ) を使用するので

```
public:
  fft(int num);
  void difFFT(COMPLEX x[]);
  void Hanning( COMPLEX x[] );
```

の後に

```
void Hamming( COMPLEX x[] );
```

を付け加える。

② ハミング窓を乗算するために，ハミング窓関数

```
void fft::Hamming( COMPLEX x[] ){
  double omega = 2.0*pi/(double)n;
  for (int i=0; i<n; i++)
    x[i].real *= (0.54-0.46*cos(omega*(double)i));
}
```

を void fft::Hanning( COMPLEX x[] ) の後に打ち込む。

③ int _tmain(int argc, _TCHAR* argv[]) を以下に示すように変更する。

式 (4.71) に示した二つの周波数を使用するために

```
double f1 = 53.0;
double f2 = 79.0;
double a0 = 0.0;
double a1 = 0.5;
double a2 = 1.0;
```

を

```
double f1 = 30.0;
double f2 = 90.0;
double a0 = 0.0;
double a1 = 1.0;
double a2 = 0.0025;
```

に変更する。

④ ハミング窓を乗算するために

```
ft.Hanning( x );
```

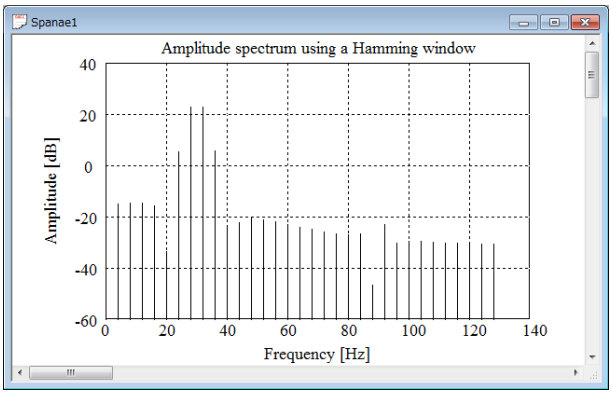
を

```
ft.Hamming( x );
```

に変更する。

解図 **4.1** に実行結果を図示する (132 ～ 252 Hz の表示は省略して

いる)。ハミング窓によるスペクトル分析では，90 Hz 付近に振幅の小さな正弦波が存在することをまったく検出できないことから，小さな振幅成分の検出にハミング窓は不適当であることがわかる。




解図 **4.1** ハミング窓を使用したプログラムの実行結果

- ブラックマン窓　　ブラックマン窓を使用する場合には，**【2】(1)** のハミング窓を用いたスペクトル解析で使用したプログラムに以下の変更を加える。


`spanaexc.exe`

① ブラックマン窓関数を使用するので

```
public:
  fft(int num);
  void difFFT(COMPLEX x[]);
  void Hanning( COMPLEX x[] );
  void Hamming( COMPLEX x[] );
```

の後に

```
  void Blackman( COMPLEX x[] );
```

を付け加える。

② ブラックマン窓を乗算するために，ブラックマン窓関数

```
void fft::Blackman( COMPLEX x[] ){
  double omega = 2.0*pi/(double)n;

  for (int i=0; i<n; i++)
    x[i].real *= (0.42-0.5*cos(omega*(double)i)
  +0.08*cos(2.0*omega*(double)i));
}
```

を `void fft::Hamming( COMPLEX x[] )` の後に打ち込む。

③ ブラックマン窓を乗算するために，`int _tmain(int argc, _TCHAR* argv[])` の

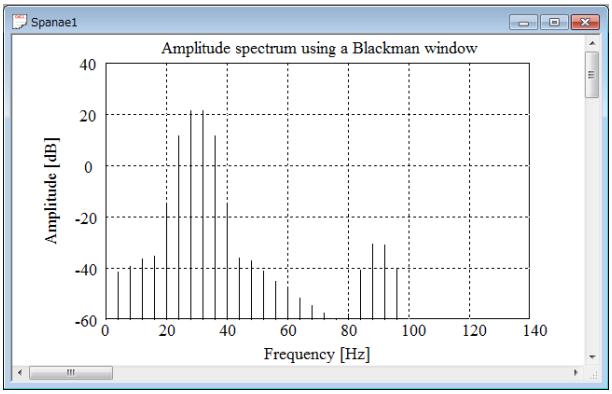
```
ft.Hamming( x );
```

を

```
ft.Blackman( x );
```

に変更する。

解図 **4.2** に実行結果を図示する (132 ～ 252 Hz の表示は省略している)。ブラックマン窓によるスペクトル分析では，90 Hz 付近の振幅の小さな正弦波を検出できることから，二つの正弦波の周波数が離れている場合，小さな振幅成分の検出にブラックマン窓は適していることがわかる。




解図 **4.2** ブラックマン窓を使用したプログラムの実行結果

**(2)** 二つの正弦波の大きさには大きな差はないが，周波数が接近している離散時間信号のスペクトルをハミング窓とブラックマン窓を用いて分析する。

- ハミング窓　　**【2】(1)** のハミング窓を使用したプログラムに以下の変更を加える。

spanaexc.exe

`int _tmain(int argc, _TCHAR* argv[])` を以下に示すように変更する。

式 (4.70) と (4.72) の振幅と式 (4.72) の周波数の未知の離散時間信号を発生させるために

```
double f1 = 30.0;
double f2 = 90.0;
double a0 = 0.0;
double a1 = 1.0;
double a2 = 0.0025;
```

を

```
double f1 = 30.0;
double f2 = 41.0;
double a0 = 0.0;
double a1 = 1.0;
double a2 = 0.25;
```

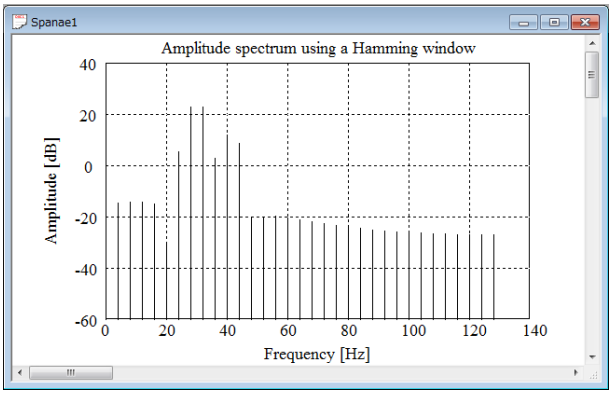
に変更する。

解図 **4.3** に実行結果を図示する (132 ～ 252 Hz の表示は省略している)。ハミング窓によるスペクトル分析によって，離散時間信号が 30 Hz と 40 Hz 付近の二つの周波数成分から成る信号であることがわかる。このように，ハミング窓は周波数分解能が高く接近した二つの周波数を分離できることが確認できる。

- ブラックマン窓　　**【2】(2)** のハミング窓を用いたスペクトル解析で使用したプログラムに，**【2】(1)** のブラックマン窓を使用したスペクトル解析と同じ変更を加える。

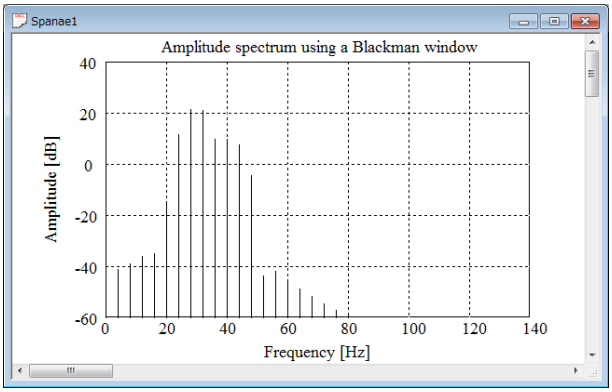
spanaexc.exe

解図 **4.4** に実行結果を図示する (132 ～ 252 Hz の表示は省略している)。ブラックマン窓によるスペクトル分析では 30 Hz と 40 Hz 付近の二つの周波数成分を分離できず，対象とした離散時間信号が二つの周波数成分から成るのか否か分析することはできない。このように，ブラックマン窓は周波数分解能が低いので，隣接した複数




解図 **4.3** ハミング窓を使用したプログラムの実行結果

の周波数成分から成る離散時間信号のスペクトル分析にはハミング窓が有効である。




解図 **4.4** ブラックマン窓を使用したプログラムの実行結果

## 5 章

**【1】** **(1)** 図 5.15 と図 5.16 に示したプログラム `lpfirlpfdsn.cpp` に以下に示す変更を加える。

`lpfirbpfdsn.exe`

① 直線位相 FIR フィルタの次数を $2M = 64$ に設定するために,
`int _tmain(int argc, _TCHAR* argv[])` の
```
int m = 16;
```
を
```
int m = 32;
```
に変更する。

② `ibpf( COMPLEX f[], int n )` では, 式 (5.139) に基づき 150($= N$) 等分した希望振幅特性の標本値を `f[i]` に記憶する。`ibpf( COMPLEX f[], int n )` の `n` は分割数を示している。標本値は実数なので, 虚数部 `f[i].imag` は零に設定する。

```
void ibpf( COMPLEX f[], int n ){
  for (int i=8; i<=17; i++)
    f[i].real = 1.0;
  for (int i=133; i<=142; i++)
    f[i].real = 1.0;
  f[7].real = 0.5;
  f[18].real = 0.5;
  f[132].real = 0.5;
  f[143].real = 0.5;
  for (int i=0; i<=6; i++)
    f[i].real = 0.0;
  for (int i=19; i<=131; i++)
    f[i].real = 0.0;
  for (int i=144; i<n; i++)
    f[i].real =0.0;
  for (int i=0; i<n; i++)
    f[i].imag = 0.0;
}
```

③ `HammingforLPFIR( COMPLEX sf[], int m )` では, 式 (5.131) に基づき, 希望振幅特性の標本値 `sf[i]` にハミング窓 `(0.54-0.46*cos(omega))` を乗算する。

```
void HammingforLPFIR( COMPLEX sf[], int m ){
  double pi=acos(-1.0);
  for (int i=0; i<=m; i++){
    double omega = 2.0*pi*(double)(m+i)/(double)(2*m);
    sf[i].real *= (0.54-0.46*cos(omega));
  }
}
```

④ 理想帯域通過特性を 150 等分し，求めた sfr[i] にハミング窓 (0.54-0.46*cos(omega)) を乗算するために，int _tmain(int argc, _TCHAR* argv[]) の

```
ilpf( lf, n, ifc );
idft( lf, sf, n, m );
```

を

```
ibpf( lf, n );
idft( lf, sf, n, m );
HammingforLPFIR( sf, m );
```

に変更する。

解図 **5.1** に設計した直線位相 FIR フィルタの乗算係数を示す。

**(2)** **5.2** 節の後半で示したように，**(1)** で作成した直線位相 FIR フィルタの乗算係数を求めるプログラムの int _tmain(int argc, _TCHAR* argv[]) に，式 (5.120) に基づき作成した振幅特性を求めるプログラムを付け加えればよい。設計した直線位相 FIR フィルタの振幅特性を解図 **5.2** に示す。

lpfirexc.exe

**(3)** **(1)** で使用したプログラムの LPW() に以下に示す変更を加える†。

lpfirexc.exe

① 配列宣言

```
int i;
```

を

```
double ts=0.3333e-4;
double y,x[2*m+1];
```

に変更する。

② 遅延器を初期化するために

```
for (int k=0; k<=2*m; k++)
  x[k]=0.0;
```

を

```
ibpf( lf, n );
```

† 波形を画面上に描画するには，記載したほかにいくつかの追加・変更が必要です。詳しくは，5 章フォルダ内の ReadMe(5 章) ファイルを読んでください。

```
idft( lf, sf, n, m );
HammingforLPFIR( sf, m );
```

の後に付け加える。

③ 時刻 $100T_s$ までの離散時間信号 $x(nT_s)$ を発生させるために，関数 LPW() の

```
for (int i=0; i<=m; i++){
```

を

```
for (int i=0; i<=100; i++){
```

に変更する。

④ 式 (5.140) の離散時間信号 $x(nT_s)$ を発生させるために

```
printf("alpha(%2dTs) = %lf \n",i,sfr[i]);
```

の代わりに

```
x[0]=0.6*sin(2.0*pi*500.*(double)i*ts);
x[0]+=0.5*sin(2.0*pi*2500.*(double)i*ts);
x[0]+=0.7*sin(2.0*pi*4500.*(double)i*ts);
```

を打ち込む。

⑤ 直線位相 FIR フィルタの出力信号を求めるために，④ で打ち込んだ行の後に，たたみ込み演算を実行するプログラム

```
y=0.0;
for (int k=0; k<m; k++)
  y+=sf[m-k].real*(x[k]+x[2*m-k]);
y+=sf[0].real*x[m];
```

と遅延を実行するプログラム

```
for (k=2*m; k>=1; k--)
  x[k]=x[k-1];
```

を順に打ち込む。

⑥ 入出力信号を画面上に表示するために，⑤ で打ち込んだ行の後に

```
SignalMag1(pDC, i, x[0]);
SignalMag2(pDC, i, y);
```

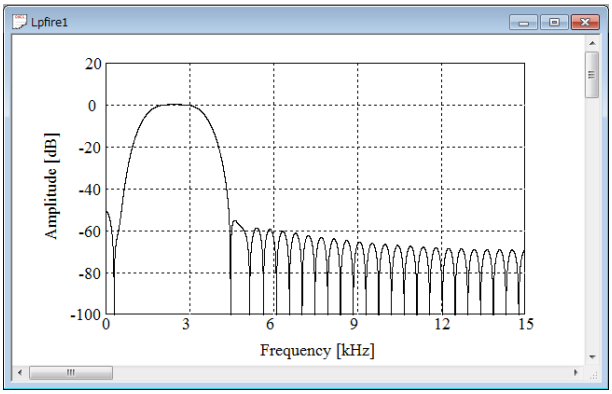
を順番に打ち込む。

出力信号波形を 解図 **5.3** に示す。500 Hz と 4.5 kHz 周波数成分の正弦波を除去し，$32T_s(= MT_s)$ 標本化周期遅れて 2.5 kHz 周波数成分の正弦波のみを取り出すことができることが確認できる。

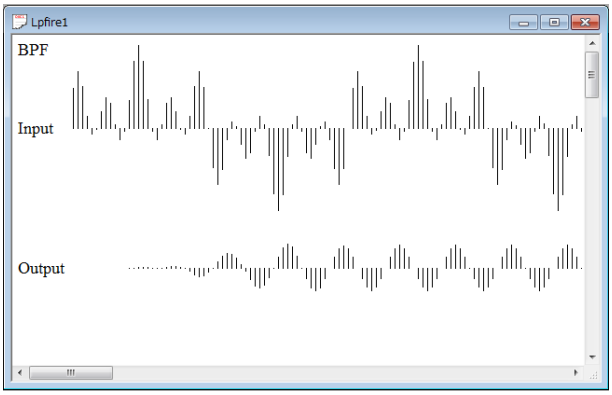
実行例 **1.1**

```
alpha( 0Ts) = 0.146667
alpha( 1Ts) = 0.125599
alpha( 2Ts) = 0.070099
alpha( 3Ts) = 0.000000
alpha( 4Ts) = -0.061023
alpha( 5Ts) = -0.094913
alpha( 6Ts) = -0.095637
alpha( 7Ts) = -0.069970
alpha( 8Ts) = -0.032849
alpha( 9Ts) = -0.000000
alpha(10Ts) = 0.018543
alpha(11Ts) = 0.021265
alpha(12Ts) = 0.013688
alpha(13Ts) = 0.004069
alpha(14Ts) = -0.001164
alpha(15Ts) = -0.000000
alpha(16Ts) = 0.005355
alpha(17Ts) = 0.010751
alpha(18Ts) = 0.012803
alpha(19Ts) = 0.010544
alpha(20Ts) = 0.005420
alpha(21Ts) = 0.000000
alpha(22Ts) = -0.003580
alpha(23Ts) = -0.004566
alpha(24Ts) = -0.003565
alpha(25Ts) = -0.001844
alpha(26Ts) = -0.000501
alpha(27Ts) = 0.000000
alpha(28Ts) = -0.000192
alpha(29Ts) = -0.000643
alpha(30Ts) = -0.000958
alpha(31Ts) = -0.000947
alpha(32Ts) = -0.000596
```

解図 **5.1** 設計した BPF の乗算係数




解図 **5.2** 設計した BPF の振幅特性




解図 **5.3** 設計した BPF の出力信号波形