第3章　MATLAB入門

3.1　introduction

3.1.1　特徴

- 変数は基本的にマトリックスである。
- 組み込み関数が豊富である。

このため他の言語と比較して、より簡単で分かりやすいプログラムを作ることができる。

3.2　コマンド入力の基本

3.2.1　コマンドの入力と出力

```
>> a=1


a =
     1
```

3.2.2　計算結果を表示しないようにするには、最後に ; （セミコロン）をつける。

```
>> b=2;
>>
```

3.2.3　% を書くと、行のそれ以降はコメントとみなされる。

```
>> %コメントの書き方
>> x=a/b %ここからはコメント


x =
    0.5000
```

3.2.4　継続行は最後に 3 つ以上の$\boxed{.}$（ピリオド）を置く。

```
>> a=1+...
2....
+3

a =
     6
```

3.3　行列の作り方

3.3.1　行の区切りは$\boxed{;}$で、列の区切りは$\boxed{,}$または$\boxed{\text{space}}$

```
>> a=[1, 2, 3; 4, 5, 6; 7, 8, 9]

a =
     1     2     3
     4     5     6
     7     8     9

>> a=[1 2 3; 4 5 6; 7 8 9]

a =
     1     2     3
     4     5     6
     7     8     9
```

3.3.2　ゼロ行列

```
>> n=zeros(3, 4)

n =
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

3.3.3　すべての要素が 1

```
>> m=ones(3, 4)

m =
     1     1     1     1
     1     1     1     1
     1     1     1     1
```

3.3.4　単位行列

```
>> c=eye(3, 3)

c =
     1     0     0
     0     1     0
     0     0     1
```

3.4　等間隔ベクトル

3.4.1　変数名=[初期値: 間隔: 最終値]

```
>> a=[1:0.5:3]

a =
    1.0000    1.5000    2.0000    2.5000    3.0000
```

3.4.2　変数名=linspace(初期値, 最終値, 要素数)

```
>> a=linspace(1, 3, 5)

a =
    1.0000    1.5000    2.0000    2.5000    3.0000
```

3.5　対数間隔ベクトル

3.5.1　変数名=logspace(最初の冪, 最後の冪, 要素の数)

```
>> b=logspace(-1, 1, 5)

b =
    0.1000    0.3162    1.0000    3.1623   10.0000
```

3.6　乱数行列の作り方

3.6.1　0 から 1 までの一様乱数

```
>> d=rand(3,3)

d =
    0.9501    0.4860    0.4565
    0.2311    0.8913    0.0185
    0.6068    0.7621    0.8214
```

3.6.2　平均値が 0、標準偏差が 1 のガウス乱数

```
>> d=randn(3,3)

d =
   -0.4326    0.2877    1.1892
   -1.6656   -1.1465   -0.0376
    0.1253    1.1909    0.3273
```

3.7　演算

3.7.1　スカラー演算

```
>> b=1+sin(pi/2)*2*3^2/6

b =
     4
```

3.8　行列の加算、減算

```
X =                          Y =
     2     0     0                0     0     3
     0     2     0                0     3     0
     0     0     2                3     0     0

>> X+Y

ans =
     2     0     3
     0     5     0
     3     0     2

>> X-Y

ans =
     2     0    -3
     0    -1     0
    -3     0     2
```

3.8.1　転置行列

```
a =                          b =
     1     2     3                1     2     3
     4     5     6
     7     8     9

>> a'

ans =
     1     4     7
     2     5     8
     3     6     9

>> b'

ans =
     1
     2
     3
```

3.8.2 行列の乗算

```
>> X*Y

ans =
     0     0     6
     0     6     0
     6     0     0

>> X.*Y

ans =
     0     0     0
     0     6     0
     0     0     0

>> Y^2

ans =
     9     0     0
     0     9     0
     0     0     9

>> Y.^2

ans =
     0     0     9
     0     9     0
     9     0     0

>> exp(eye(3,3))

ans =
    2.7183    1.0000    1.0000
    1.0000    2.7183    1.0000
    1.0000    1.0000    2.7183
```

3.8.3 行列の除算

```
A =                         B =
     1     2                     3     2
     3     1                     2     1

>> A/B

ans =
    3.0000   -4.0000
   -1.0000    3.0000

>> A*inv(B)

ans =
    3.0000   -4.0000
   -1.0000    3.0000

>> A\B

ans =
    0.2000         0
    1.4000    1.0000

>> inv(A)*B

ans =
    0.2000         0
    1.4000    1.0000

>> A./B

ans =
    0.3333    1.0000
    1.5000    1.0000

>> A.\B

ans =
    3.0000    1.0000
    0.6667    1.0000
```

3.9 グラフィックス

3.9.1 $y = f(x)$ グラフの描画

```
>> x = [0:0.2:10];

>> y=besselj(0,x);
>> plot(x, y)
>> hold on

>> z=besselj(1,x);
>> plot(x, z, 'r')

>> grid on;

>> title('Bessel Functions',  'fontsize', 18)
>> xlabel('x', 'fontweight',  'bold', 'fontsize', 14)
>> ylabel('J_n(x)', 'fontweight',  'bold', 'fontsize', 14)
>> text(1.2, 0.8, ' J_0(x)', 'fontweight', 'bold', 'fontsize', 14)
>> text(3.3, 0.25, ' J_1(x)', 'fontweight', 'bold', 'fontsize', 14)
```

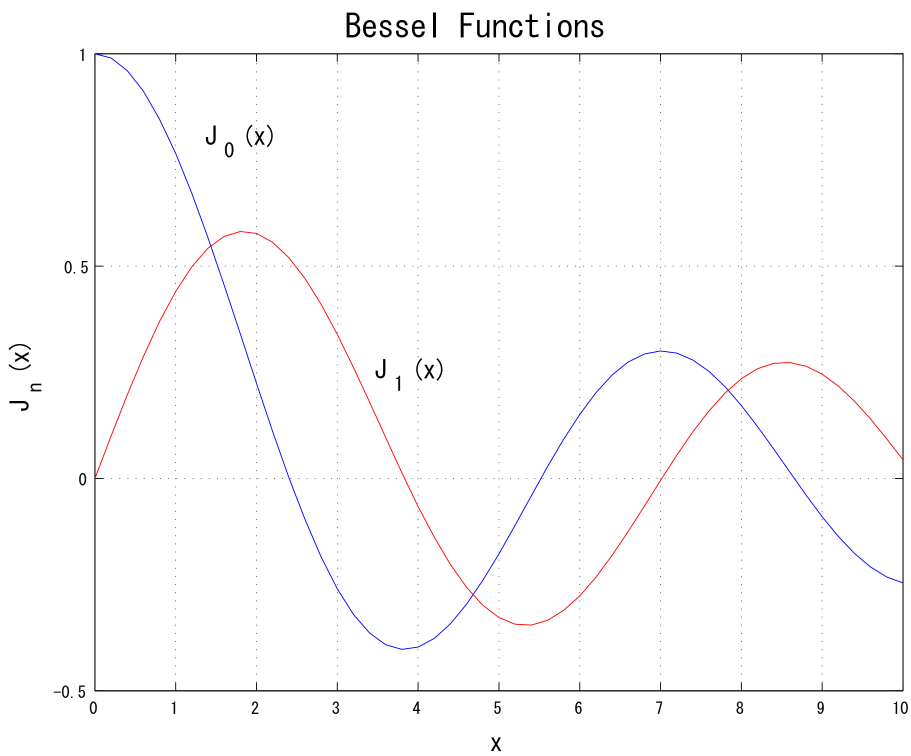

図 3.1: ベッセル関数

3.9.2 グラフに関する属性

表 3.1: グラフの属性

'fontsize'	フォントのサイズ、12, 14, ...
'fontweight'	フォントの太さ、'normal', 'bold',....
'color'	線、テキストのカラー、'r', 'b', 'g',....
'linewidth'	線の太さ、1, 2,....
'linestyle'	線の種類、'-', '.',.....

3.9.3 種々の 2 次元プロット関数

表 3.2: 種々の 2 次元プロット関数

loglog	loglog(x, y)	データ y の両対数プロット
semilogx	semilogx(x, y)	データ y の x 軸対数プロット
semilogy	semilogy(x, y)	データ y の y 軸対数プロット
plotyy	plotyy(x1, y1, x2, y2)	(x1,y1) の座標軸を左、(x2,y2) の座標軸を右に示す
polar	polar(theta, r)	極座標プロット
bar	bar(x, y)	棒グラフ
barh	barh(x, y)	水平棒グラフ
errorbar	errorbar(x, y, erb)	エラーバーつきグラフ
pie	pie(y)	円グラフ
contour	contour(Z)	2 次元配列データのコンタープロット

3.9.4 複数グラフの配置

```
>> x = [0:0.2:10];

>> y=besselj(1,x);
>> subplot(2, 2, 1)
>> plot(x, y)
>> text(0.15, -0.2, ' J_1(x)', 'fontweight', 'bold', 'fontsize', 14)

>> y=besselj(2,x);
>> subplot(2, 2, 2)
>> plot(x, y)
>> text(0.15, -0.2, ' J_2(x)', 'fontweight', 'bold', 'fontsize', 14)

>> y=besselj(3,x);
>> subplot(2, 2, 3)
>> plot(x, y)
>> text(0.15, -0.2, ' J_3(x)', 'fontweight', 'bold', 'fontsize', 14)

>> y=besselj(4,x);
>> subplot(2, 2, 4)
>> plot(x, y)
>> text(0.15, -0.2, ' J_4(x)', 'fontweight', 'bold', 'fontsize', 14)
```

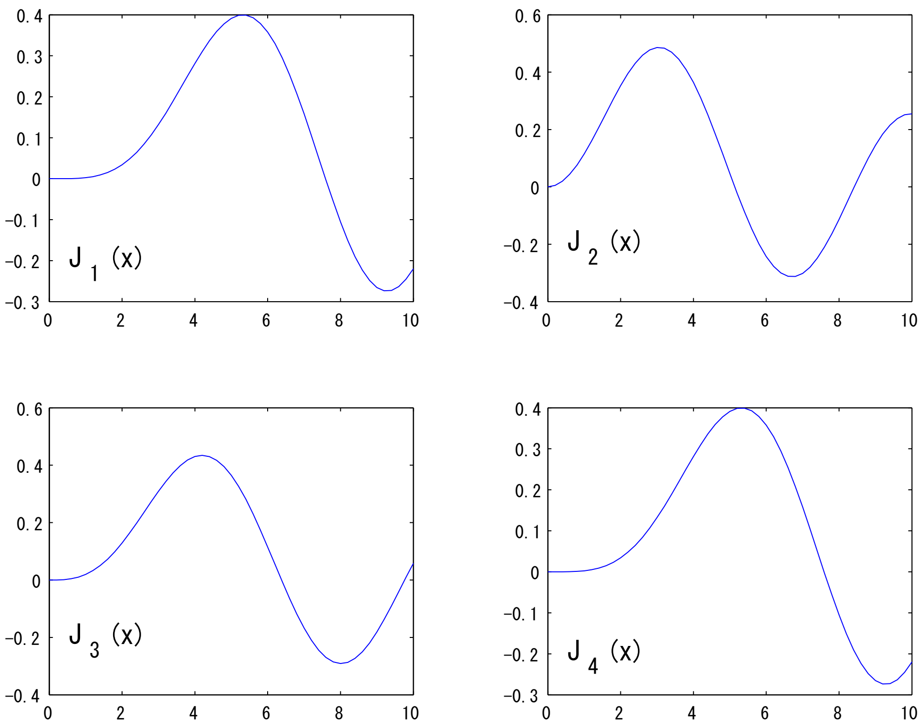

図 3.2: グラフの配置

3.10　応用例：線型応答

3.10.1　例題：RC 積分回路の応答

RC 積分回路で $RC = 1[s]$ としたとき、次を図示せよ。

(1) 伝達関数の Bode 線図

```
>> num=1;
>> den=[1, 1];

>> sys=tf(num, den)

Transfer function:
  1
-----
s + 1

>> bode(sys) % or bode(num, den)
```

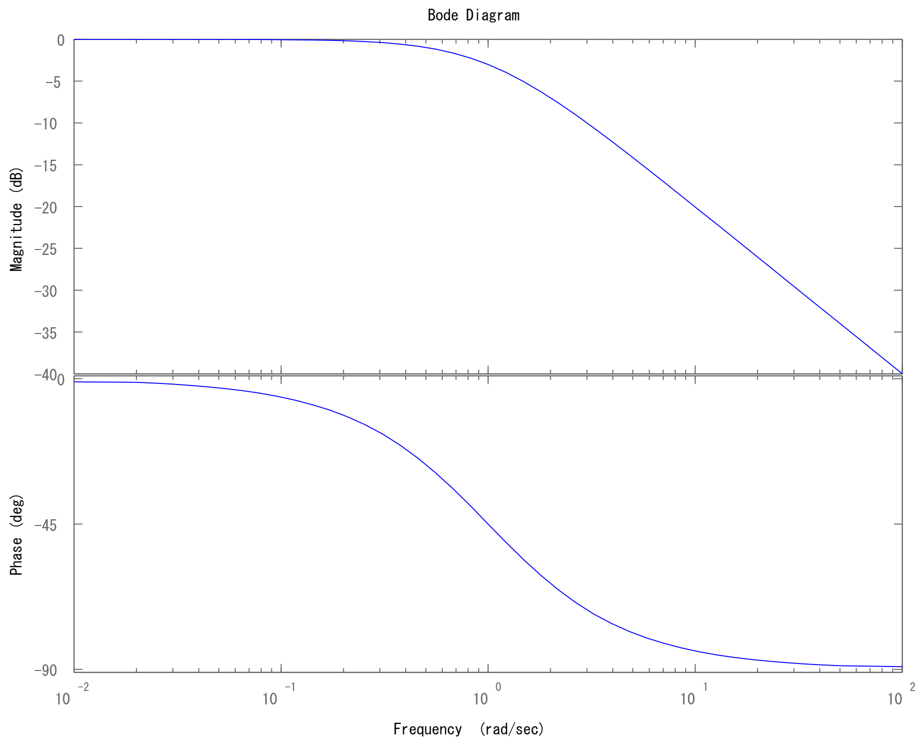

図 3.3: RC 積分回路の Bode 線図

(2) 伝達関数の Nyquist 線図

```
>> nyquist(sys) % or nyquist(num, den)
```

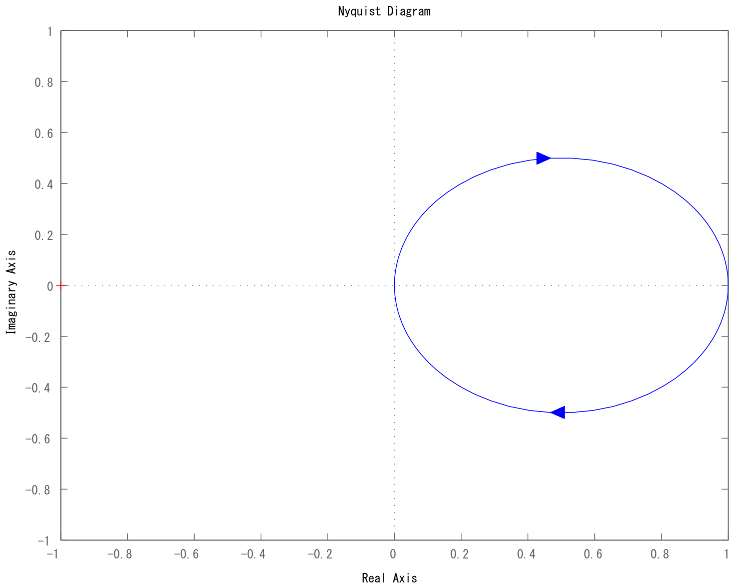

図 3.4: *RC* 積分回路の Nyquist 線図

(3) インパルス応答

```
>> impulse(sys) % or impulse(num, den)
```

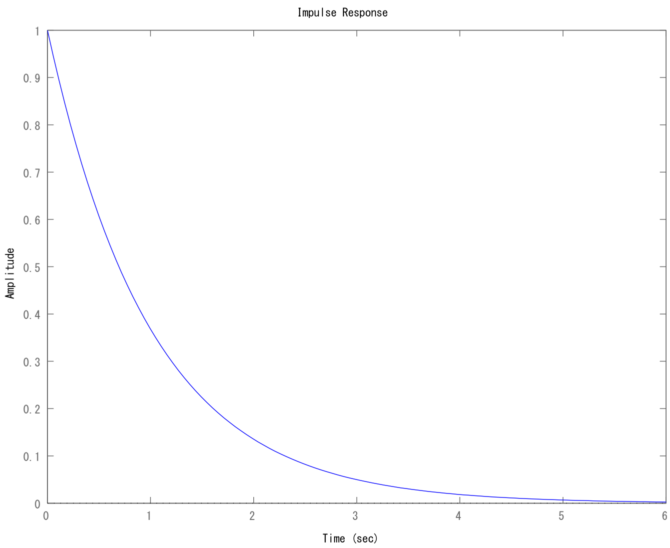

図 3.5: *RC* 積分回路のインパルス応答

(4) ステップ応答

```
>> step(sys) % or step(num, den)
```

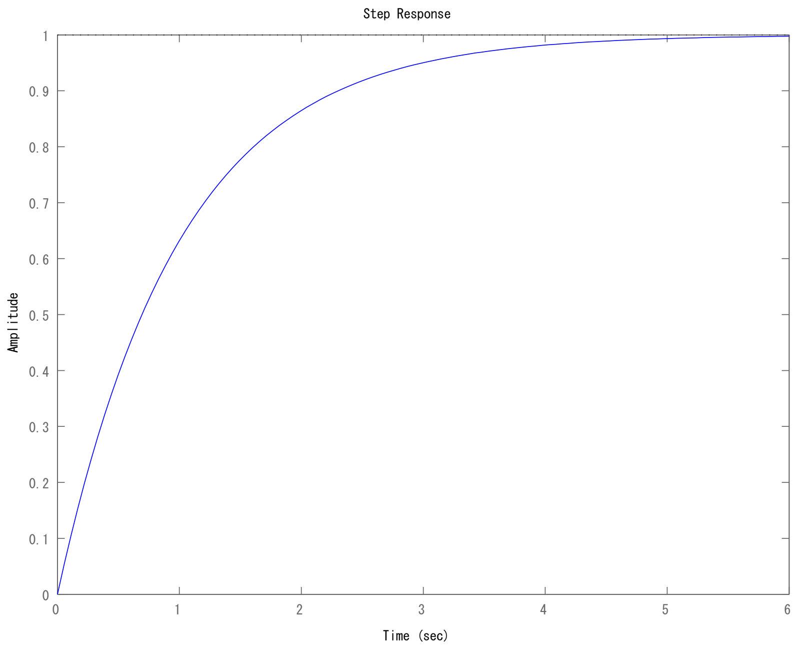

図 3.6: *RC* 積分回路のステップ応答

3.10.2　例題：調和振動子の応答

調和振動子のステップ応答が Q の値と共にどのように変わるか図示せよ。ただし、m=1 [kg], $\omega_0 = 1$ [rad/s] とする。

```
>> tt=[0: 0.01: 10];
>> y=[];

>> for Q=[0.2, 0.5, 0.8, 2]
y=[y, step(1, [1, 1/Q, 1], tt)];
end

>> plot(tt, y);grid on
```

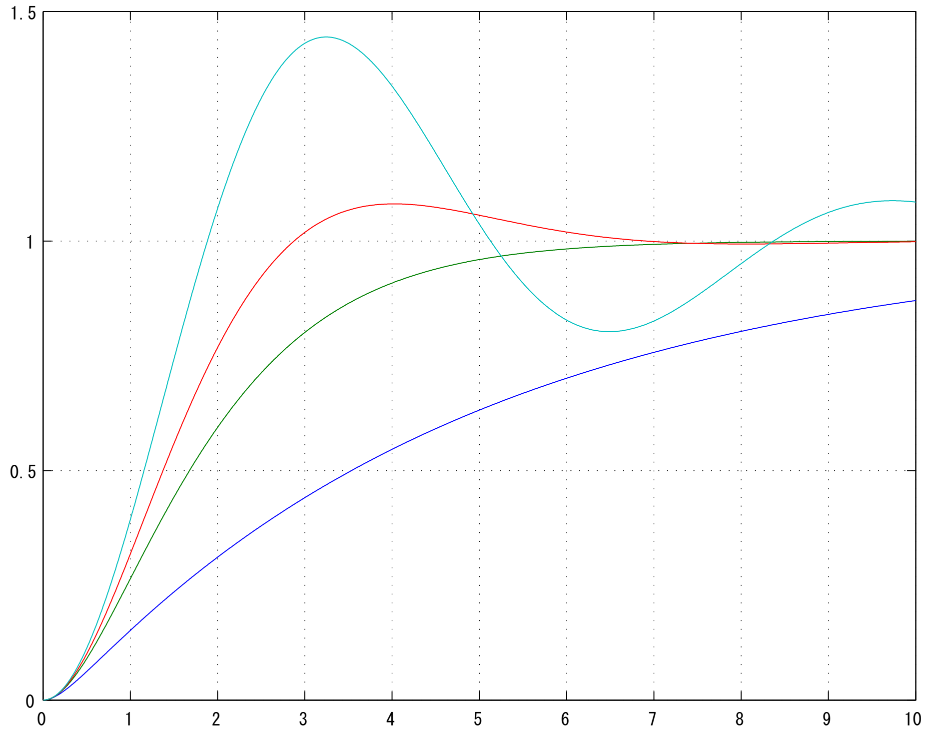

図 3.7: 調和振動子の過渡応答

3.11　応用例：データ処理

3.11.1　例題：積算平均効果

積算平均効果を実際に確かめよ。

```
>> clf;
>> t=[0: 0.002: 1];
>> N=size(t, 2)
N =
   501
>> y=(15*t).*exp(-10*t).*sin(20*pi*t);
>> z=zeros(1, N);

>> for k=1:1000
x=y+(rand(1, N)-0.5);
z=z+x;
if k==1
    subplot(5, 1, 1), plot(t, z/k); ylabel('k=1');
    text(0.8, -0.5, num2str(std(z/k-y)^2), 'fontweight', 'bold');
elseif k==10
    subplot(5, 1, 2), plot(t, z/k); ylabel('k=10');
    text(0.8, -0.5, num2str(std(z/k-y)^2), 'fontweight', 'bold');
elseif k==100
    subplot(5, 1, 3), plot(t, z/k); ylabel('k=100');
    text(0.8, -0.5, num2str(std(z/k-y)^2), 'fontweight', 'bold');
elseif k==1000
    subplot(5, 1, 4), plot(t, z/k); ylabel('k=1000');
    text(0.8, -0.5, num2str(std(z/k-y)^2), 'fontweight', 'bold');
end
subplot(5, 1, 5), plot(t, y); ylabel('signal');
end;
```

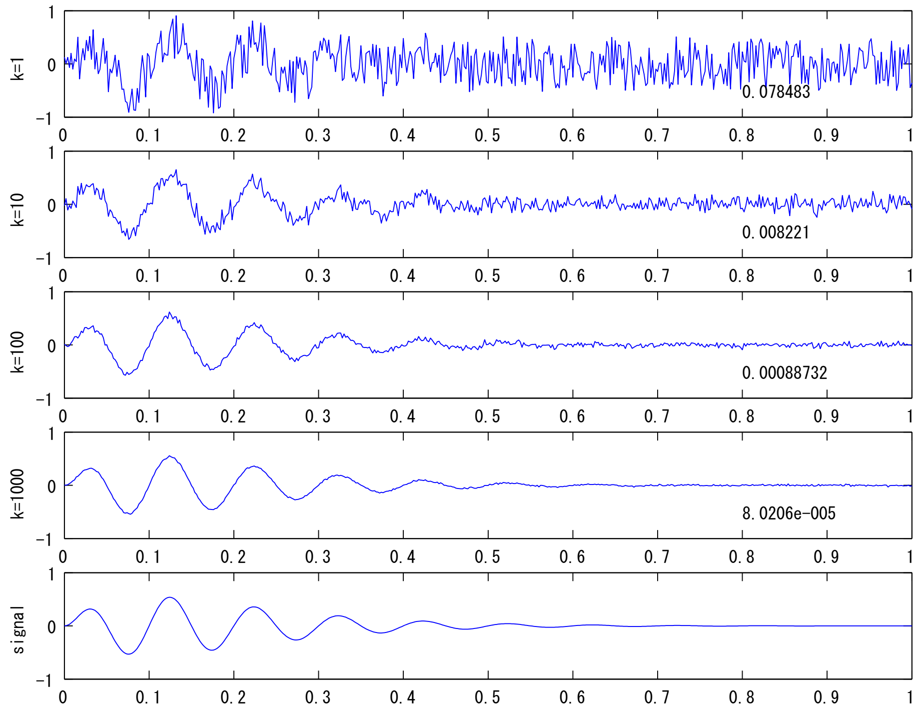

図 3.8: 積算平均効果のテスト