ディジタル信号処理シリーズ

Scilabで学ぶ
信号数学，信号解析，信号処理

見本

【改訂新版】

やり直しのための工業数学 信号処理＆解析編

●三谷 政昭 著

CD-ROM付
Scilab/解析ツールを収録!

CQ出版社

第1章 信号処理& 解析の初歩の初歩

信号処理＆解析というと，難しい数式が目いっぱい出てくる，手に負えないもののように思い込んでいて，初めから避けている人が多いのではないだろうか．しかし，シミュレータやプログラムを用いて解き方を体験すれば，数式が苦手でも大丈夫．取りあえず，何もしないで『これは難解だ？』とか『あれはダメだ！』という先入観を捨て去ってもらいたい．

私たちの身の周りの物理現象や身体から発せられる情報の多くは，アナログ信号の形で伝えられる．例えば，私たちが日常的に目で見る映像や耳にするサウンドはいろいろな情報を持っているが，映像は撮影カメラ，音声はマイクを通して電圧や電流に変換され，時間とともに連続的に変換するアナログの電気信号になる．このようなアナログの電気信号をそのままの形で，抵抗，コンデンサ，トランジスタ，OPアンプ(Op-Amp，演算増幅器)などから構成される電気電子回路(**アナログ・システム**)で微分・積分処理するのが**アナログ信号処理**である．一方，アナログの電気信号をディジタルの数値データの並びに変換し，パソコンや演算プロセッサなどの**ディジタル・システム**を用いて加減乗除(+，−，×，÷)の単純な四則演算で信号処理するのが**ディジタル信号処理**である．

いずれの信号処理についても，入力信号および出力信号の変動の様子を目で見て手を動かして体感すること，そして直観的な理解とイメージを皆さんに植え付けることによって，「信号解析に対するアレルギー」を取り去ってもらおうというわけだ．

本章では，“まずはチャレンジ！　そしてダメなら次の一手を考える”という精神で，絵本を見る感覚で，信号処理＆解析の醍醐味を体験してもらおう．あたかも自分で実験しているかのような気分に浸りつつ，信号処理＆解析への興味と理解を深めていただく．また，「信号処理＆解析で何ができるのか？」という素朴な疑問に対し，目に見える形で表現して分かりやすく解説する．

1.1　画像表示＆２次元フーリエ解析を体験してみよう(Scilab)

信号処理＆解析と言えば，「数式ばかりで，ああーん，イヤになっちゃっう」とか，「物理的な意味がさっぱり分からん，数式は丸暗記するだけ」のように“数式恐怖症”に苛まれている皆さんがほとんどかもしれない．

「まえがき」にも記したように，**付録A**を参考に“Scilab/Xcos”を必ずインストールし，さらには**付**

録CのScilabの基本命令と使用方法について，おっくうがらずに手を動かして確認しながら丁寧に読み進んでいただきたい．

Scilab（サイラブ）はINRIA（Institut National de Recherche en Informatique et Automatique；フランス国立情報学自動制御研究所）が開発・提供する科学技術計算ソフトウェアである（**図1.1**）．付随するXcos（エックスコス）はブロック線図（ブロック・ダイアグラム，Block Diagram）を用いたシミュレーション機能を有している．グラフ表示も多彩で，しかもプログラム作成も簡単にできるという特徴がある．

▶Scilab

機能的にはMATLAB（MATrix LABoratory）とほぼ同等と考えてよく，多数の科学技術分野（音/画像処理，最適化，制御理論，線形代数，システム理論など）における計算処理用の関数群から構成される．また，インタプリタ型のプログラミング言語の機能（2次元や3次元のグラフィックス表示，行列計算など）を持ち，既存の関数群を組み合わせてプログラム作成すれば，新しい機能が容易に実現される．さらには，C言語やFortranなどの別言語で作成したプログラムも組み込めるので，過去のソフトウェア資産を活用できる．

▶Xcos（詳細は後述，**1.8**，**1.9**を参照）

機能的にはSimulinkもどきと見なせる．シミュレーション用の多種類の機能ブロック（主な適用分野：電気工学，信号処理，連続時間/離散時間システム，熱水力学など）が用意されており，ブロック線図を描く（機能ブロックを相互に接続する）だけで，Scilab言語による複雑なプログラムを必要としない．もちろん所望の機能ブロックがないときは，ユーザ定義関数として作成可能である．

図1.1　Scilabのホームページ（http://www.scilab.org/，2012年1月現在）

それでは，解説書で敬遠しがちな画像(2次元データ)を取り上げ，Scilabを利用して画像や周波数スペクトル成分を表示し，思ったより簡単なんだと信号処理&解析に対する違和感を取り去ってもらうことから始めよう．ただし，処理の煩雑さを避けるために濃度(輝度，明暗度ともいう)データで表されるモノクロ画像を考えることにする．なお，カラー画像の場合は，R(赤)，G(緑)，B(青)の光の三原色に分ければよく，それぞれの色情報が濃度データを有するので3枚のモノクロ画像と見なせる．

早速，モノクロ画像を読み込んで表示し，濃度データの変化のようすを調べてみよう(**実行例1.1**，**図1.2**)．

実行例1.1(モノクロ画像の読み込み，画像表示)

```
--> g1=imload(); ↵          ..... ①
--> imdisp(g1,1); ↵         ..... ②
--> imdisp3d(g1,2); ↵       ..... ③
```

【実行例1.1の説明】

① 画像データ(拡張子「.img」)を読み込んで，変数g1に格納する

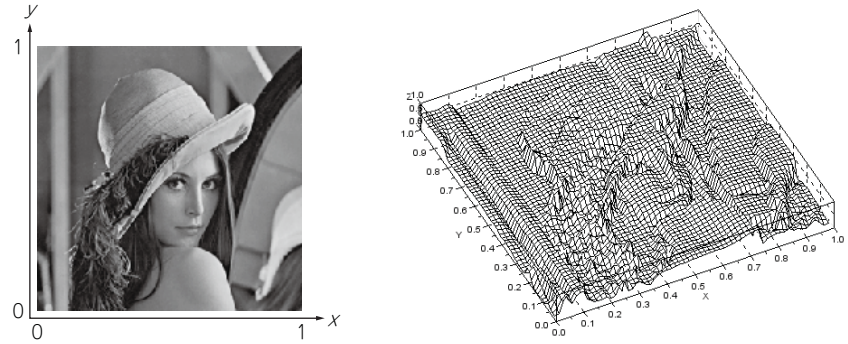
② モノクロ画像(変数g1)をウィンドウ画面1に表示する[**図1.2**(**a**)]

③ モノクロ画像(変数g1)の濃度変化を，ウィンドウ画面2に3D表示する[**図1.2**(**b**)]

[**関数コマンド1.1**(画像ファイルの読み込み)]

```
imload()
```

imload命令を実行すると，新しいウィンドウ画面(フォルダ名「image」)が開く．ファイル・タイプとして「全ての*.imgファイル」を選択すれば，モノクロ標準画像ファイル(拡張子「.img」)の一覧表が現れるので，読み出したいファイルを選択して[開く(O)]をクリックする(**図1.3**)．**図1.3**では「LENNA.img」を選択したが，ほかの画像も試してもらいたい．なお，ファイル・タイプとして「全ての*.spmファイル」を選択すれば，画像スペクトル値の保存ファイル(拡張子「.spm」)を読み出すことができる(後述)．



(**a**) 画像表示　　(**b**) 濃度変化の3D表示

図1.2　[実行例1.1](画像ファイル「LENNA.img」)

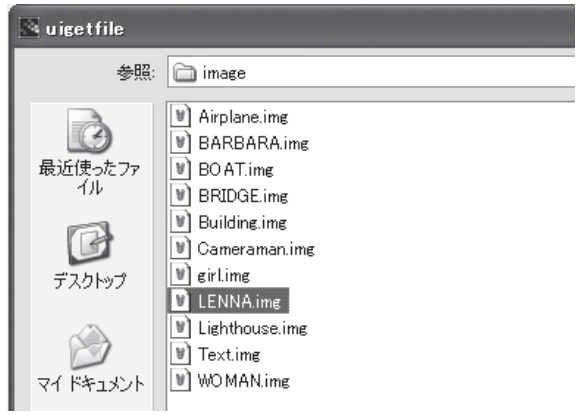


図1.3　モノクロ標準画像ファイル(拡張子「.img」)**のウィンドウ表示例**

[**関数コマンド1.2**(モノクロ画像の濃度表示)]

```
imdisp（画像変数，ウィンドウ番号）
```

黒レベルは0，白レベルは1として，濃度(グレイ・レベル値)を0～1の範囲で画像化して表示する．なお，

```
imload（ウィンドウ番号）
```

と入力すると，imloadとimdispの二つの命令を同時に実行できる．

[**関数コマンド1.3**(モノクロ画像の濃度変化の3D表示)]

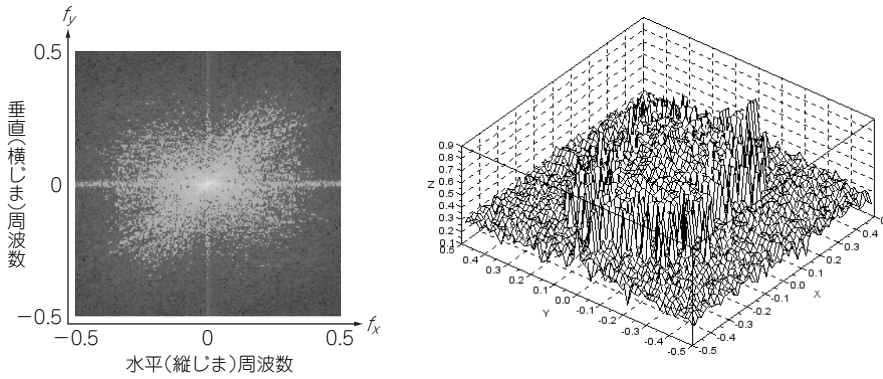
```
imdisp3d（画像変数，ウィンドウ番号）
```

画像(0～1の範囲の濃度)を，メッシュに分けて3D表示する．ここで，表示が分かりにくいときは，表示ウィンドウの左上のアイコン(,)をクリックして，回転/拡大(ズーム)するとよい．また，表示を元に戻すときはアイコン( )をクリックする．なお，メニュー・バーからツールを選択しても同様の処理が可能である．

次に，2次元フーリエ変換して画像スペクトルを求めてみよう(**実行例1.2**，**図1.4**)．

実行例1.2(2次元フーリエ変換による画像スペクトル表示)

```
--> g1=imload(); ↵ ..... ①
--> imdisp(g1,1); ↵ ..... ②
--> g2=MFT(g1); ↵ ..... ③
--> imspec(g2,2); ↵ ..... ④
--> imspec3d(g2,3); ↵ ..... ⑤
--> imsave(g2); ↵ ..... ⑥
--> g3=IMFT(g2); ↵ ..... ⑦
--> imdisp(g3,4); ↵ ..... ⑧
```



(a) 振幅スペクトル表示　　(b) 3D表示

図1.4　[実行例1.2]（画像ファイル「LENNA.img」）

【実行例1.2の説明】

①，② 画像(ファイル名「LENNA.img」)を読み込んで変数g1に格納した後，ウィンドウ画面1に表示する．`imload(1) ;` ↵ と入力してもよい

③ 画像(変数g1)の2次元フーリエ変換したスペクトル値を変数g2に格納する

④ 画像スペクトル(変数g2)の振幅特性を濃度として表し，ウィンドウ画面2に表示する[**図1.4**(**a**)]

⑤ 画像スペクトル(変数g2)の振幅特性の濃度変化を，ウィンドウ画面3に3D表示する[**図1.4**(**b**)]

⑥ 画像スペクトル(変数g2)をファイル・データとして格納する

⑦ 画像スペクトル(変数g2)の2次元フーリエ逆変換した値を変数g3に格納する

⑧ 画像データ(変数g3)をウィンドウ画面4に表示して，もとの画像データが復元されていることを確認する

[**関数コマンド1.4**(画像の2次元フーリエ変換によるスペクトル値の計算)]

MFT （画像変数）

モノクロ画像を2次元フーリエ変換して，スペクトル値(複素数)を算出する

[**関数コマンド1.5**(画像スペクトルの振幅特性の濃淡表示)]

imspec （画像スペクトル変数，ウィンドウ番号）

黒レベルは0，白レベルは1とし，画像スペクトルの振幅(絶対)値の大小に応じて0～1の範囲の濃度，すなわち輝度に対する明暗で置き換え，大きい値ほど明るくなるように画像化して表示する．ここで，**図1.5**に示すように，横軸と縦軸はそれぞれ，水平周波数(縦じまの細かさ)，垂直周波数(横じまの細かさ)の正規化した値(− 0.5 ～ 0.5)を表し，白っぽいほどスペクトル値が大きいことを意味する．例えば，**図1.6**(**a**)と(**b**)を比較してみると，大まかな画像には低い周波数成分が，細かい画像には高い周波数成分が多く含まれていることが分かる．なお，関数コマンドimload

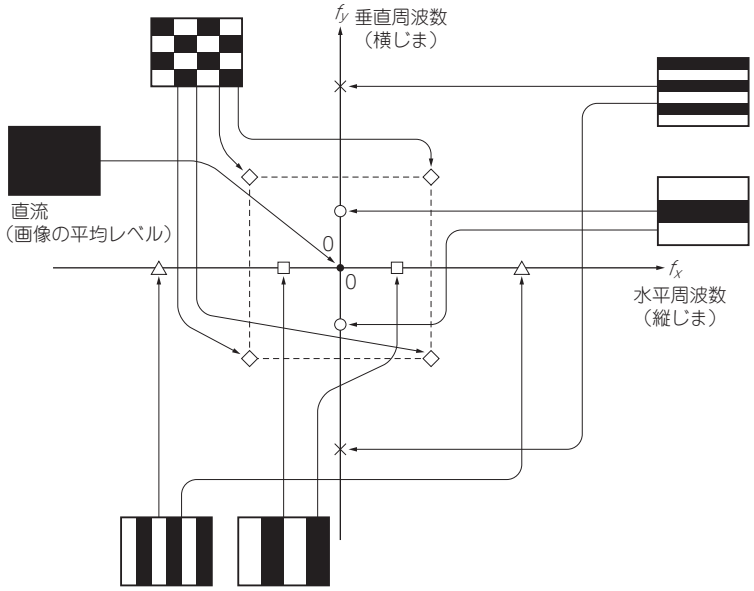
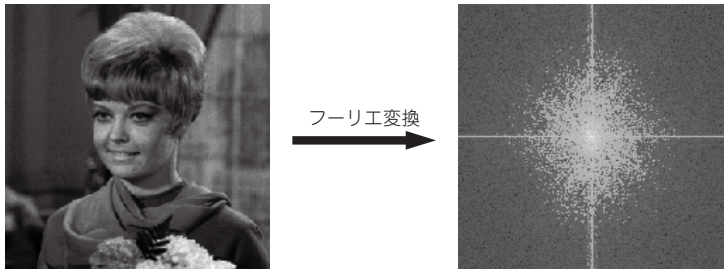
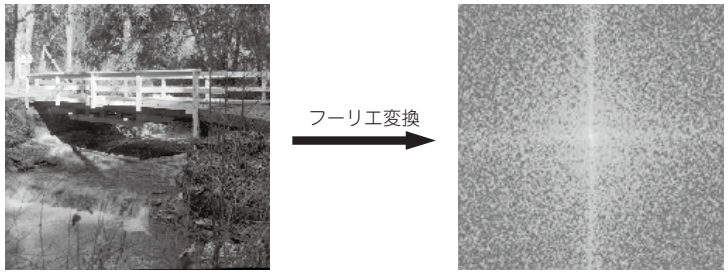


図1.5　2次元周波数と画像の関係



（低い周波数成分が多い）

（**a**）大まかな濃度変化の画像例（画像ファイル「`girl.img`」）



（高い周波数成分が多い）

（**b**）細かな濃度変化の画像例（画像ファイル「`BRIGDE.img`」）

図1.6　画像のスペクトル値の比較

を利用して，

　imload（ウィンドウ番号）

と入力すると，imloadとimspecの二つの命令を同時に実行できる．その際，ファイル・タイプとして「全ての*.spmファイル」を選択すれば，画像スペクトル値を保存したファイル（拡張子「.spm」）の一覧表が現れるので，読み出したいファイルを選択して 開く(O) をクリックする．

［**関数コマンド1.6**（画像スペクトルの濃度変化の3D表示）］

imspec3d（画像スペクトル変数，ウィンドウ番号）

画像スペクトル（0～1の範囲の濃度）を，メッシュに分けて3D表示する．ここで，表示が分かりにくいときは，表示ウィンドウの左上のアイコン（  ，  ）をクリックして，回転/拡大（ズーム）するとよい．また，表示を元に戻すときはアイコン（  ）をクリックする．なお，メニュー・バーからツールを選択しても同様の処理が可能である．

［**関数コマンド1.7**（画像スペクトルあるいは画像データの書き込み保存）］

imsave（画像スペクトルあるいは画像データ変数）

imsave命令を実行すると，新しいウィンドウ画面（フォルダ名「image」）が開く．ファイル・タイプとして「全ての*.spmファイル」を選択すれば，画像スペクトル・ファイル（拡張子「.spm」）の一覧表が現れるので，保存ファイル名（拡張子「.spm」を必ず付ける）を入力して 開く(O) をクリックする（**図1.7**）．**図1.7**では，「LENNA.spm」というファイル名を付けて格納している．

また，モノクロ画像データを格納するときは，ファイル・タイプとして「全ての*.imgファイル」を選択後，保存ファイル名（拡張子「.img」を必ず付ける）を入力して 開く(O) をクリックすればよい．

［**関数コマンド1.8**（画像スペクトル値の2次元フーリエ逆変換による画像の濃度計算）］

IMFT（画像スペクトル変数）

MFT命令の逆操作で，画像スペクトル値（複素数）を2次元フーリエ逆変換して画像データを算出する．

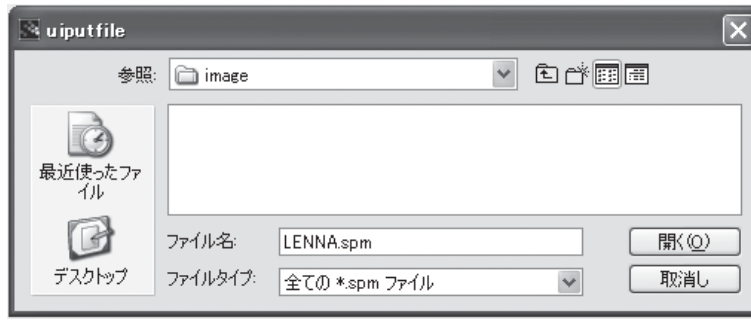
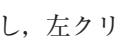
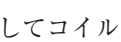
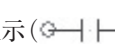


図1.7　画像スペクトル（拡張子「.spm」）の格納時のウィンドウ表示例

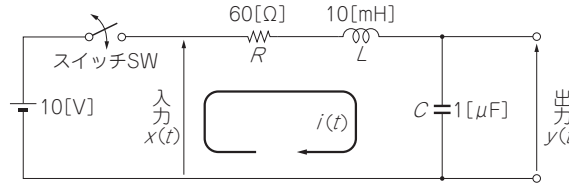
1.2　アナログ信号処理(電気・電子回路シミュレータ)を体験してみよう[＊1]

ここでは，アナログ信号処理の簡単な例として，**図1.8**に示す抵抗R，コイルL，コンデンサCの直列回路(これ以後，RLC回路)を考えてみよう(**3.7**，**付録B**を参照)．取りあえず，**図1.8**のスイッチSWをON/OFFしたときの回路電流とコンデンサの端子電圧の時間的変化(過渡応答特性)，ならびに入力に対する出力の周波数特性(利得，位相)をシミュレーションする手順を紹介するので，一つずつ順を追って進めていただきたい．

① ハイブリッド・シミュレータを立ち上げ，フリースペース(実験室)を開いて，素子や測定プローブを配置するためのブレッド・ボード画面を表示する(**図1.9**)．

② 抵抗(アイコン； )の真上にカーソルを移動し，左クリックする．抵抗表示()を移動して回路図上に置く．次に，スペース･キーを押してコイル表示()にし，回路図上に置く．続けて，スペース・キーを押してコンデンサ表示()にし，キーボードから“R”キーを押して回転し，回路図上に置く(**図1.10**)．なお，抵抗は10[kΩ]，コイルは100[mH]，コンデンサは0.01[μF]が素子値としてデフォルトで設定されているが，再設定する必要がある(⑤で後述)．

③ 続いて，アイコン( ，  ， )の真上にカーソルを移動し，左クリックして，順にDC電源(直



$$x(t)=LC\frac{d^2y(t)}{dt^2}+CR\frac{dy(t)}{dt}+y(t)$$

(入出力関係を表す微分方程式)

図1.8　*RLC*回路

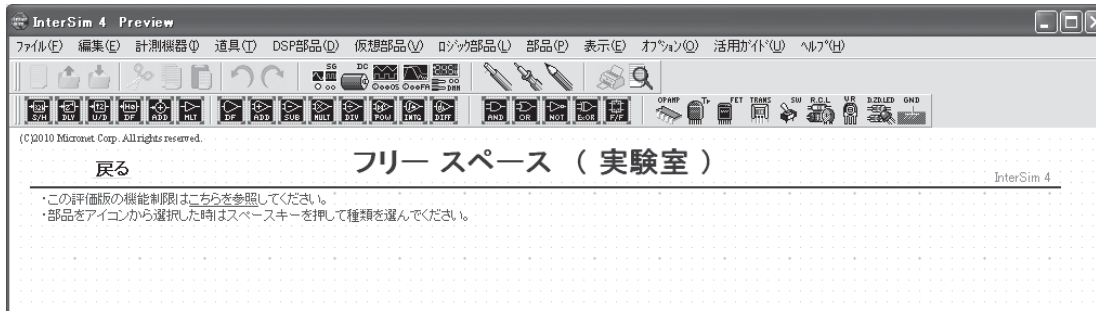
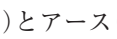


図1.9　実験室(シミュレーション画面，一部表示)

[＊1]　三谷政昭；わかる電子回路入門の入門(Ⅰ)ダイオード・トランジスタ編，(Ⅱ)オペアンプ・基礎編，(Ⅲ)オペアンプ・応用編，(Ⅳ)オペアンプ・実用編，CQ出版，2011年

流電源，  ）とアース（）とスイッチ（  ）を回路図上に置く（**図1.10**）．

④ 配線コード（アイコン；  ）あるいははんだごて（アイコン；  ）を移動し，左クリックして，**図1.10**の配置された回路素子をつないで*RLC*回路が動作するように接続する（**図1.11**）．

⑤ **図1.8**の各素子値を設定するため，回路素子の真上にカーソル（  ）あるいは鉛筆（アイコン；  ）を移動して左クリックすると，部品定数の設定ダイアログ画面が現れるので，所定の各素子値を入力する（**図1.12**）．この例では**図1.8**に基づき，コイルL1は10[mH]，コンデンサC1は1[μF]，抵抗R1は60[Ω]に設定する．

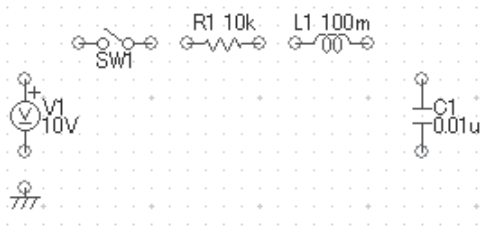


図1.10　回路素子の配置

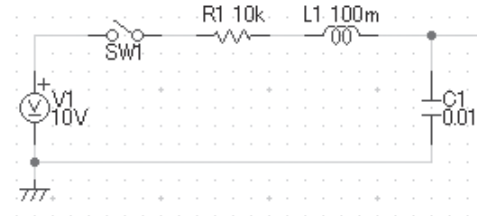


図1.11　回路素子の接続

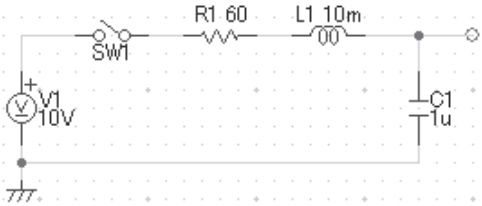


図1.12　回路素子値の設定

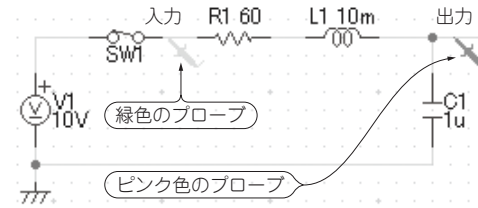


図1.13　測定プローブ（オシロスコープ）の接続

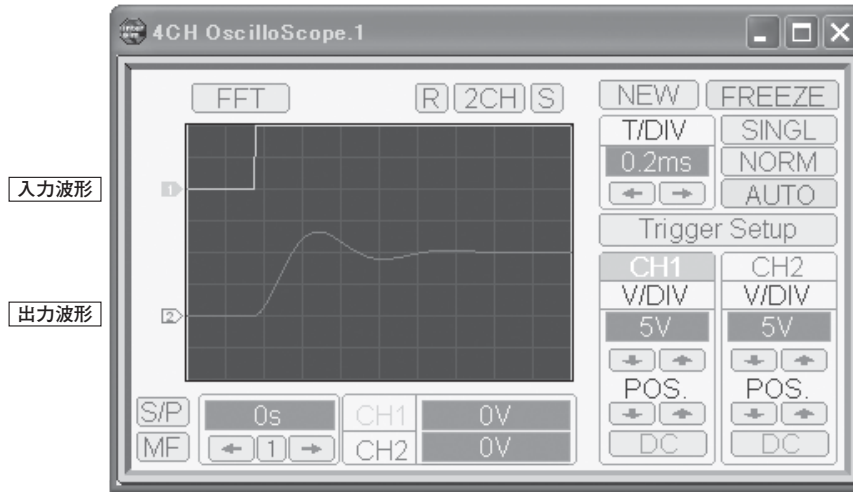


図1.14　スイッチ投入後の入出力波形

⑥ オシロスコープ画面(アイコン； )を開き，プローブ()を接続して入出力波形を表示する．ここでは，2チャネル表示とし，入力は緑色，出力はピンク色のプローブを接続した後，スイッチSW1を投入し，*RLC*回路の応答特性を確認する(**図1.13**，**図1.14**)．その際，オシロスコープの電圧/電流(縦軸で，切り替えは，V/DIVあるいはA/DIVスイッチを左クリックする)，時間(横軸)のスケール変更はNEWおよびFREEZEスイッチを適切に押して波形表示の状態を切り替える必要がある．同時に，スイッチSW1を投入するタイミングもいろいろと変えて，**図1.14**の入出力波形の表示が得られるように楽しみながら試行錯誤していただきたい．なお，新しい設定で再度シミュレーションする場合は，スイッチSW1を切断した後，NEWスイッチを押してコンデンサに蓄積された電荷をゼロ(0)にすることを忘れないようにします．

⑦ 次に，周波数アナライザ画面(アイコン； )を開き，黒く縁取りされた緑色のプローブ()を入力，中が白抜きの緑色のプローブ()を出力に接続して，周波数特性(利得は明るい緑色でデシベル[dB]表示，位相は暗い緑色)を表示する(**図1.15**，**図1.16**)．その際，周波数アナライザの周波数(横軸)，利得(縦軸)を適切に切り替える必要があり，いろいろと変更していただきたい．

図1.15　測定プローブ(周波数アナライザ)**の接続**

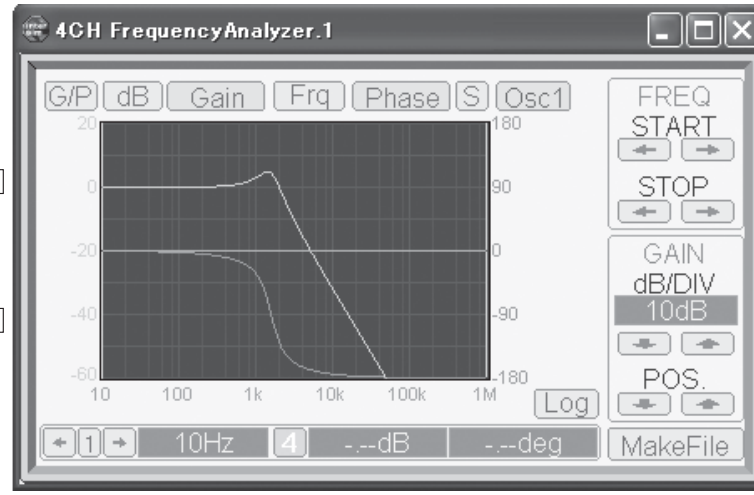


図1.16　*RLC*回路の周波数特性(ローパス特性)

見本

このPDFは，CQ出版社発売の「改訂新版 やり直しのための工業数学 信号処理&解析編」の一部見本です．

内容・購入方法などにつきましては以下のホームページをご覧下さい．
内容　　http://shop.cqpub.co.jp/hanbai/books/31/31461.htm
購入方法　　http://www.cqpub.co.jp/hanbai/order/order.htm

ディジタル信号処理シリーズ