第 1 章

画像・映像信号処理とは

画像・映像情報メディアのディジタル化は，高品質，高効率，高機能，高セキュリティなどの利益をもたらす．これらの技術を実現し，発展させていくためには，画像・映像信号処理についての理解は欠かせない．本章では，本書全体を概観するために，画像・映像信号処理の概要と本書のねらいについて述べる．第2章以降で具体的に解説を進める．なお，本書において「画像」は静止画像を，「映像」は動画像を示す言葉として使用する．

1.1　画像・映像信号処理の基礎

画像・映像信号処理とは，「画像・映像情報を伝送，記録，解析，加工する技術」である．その種類も圧縮，強調，改善，再構成，記述，認識など，多岐にわたる．

● 画像・映像信号処理の流れ

図 1.1 に画像・映像信号処理の流れを示す．まず，撮像機器のセンサによって光の分布が電気信号

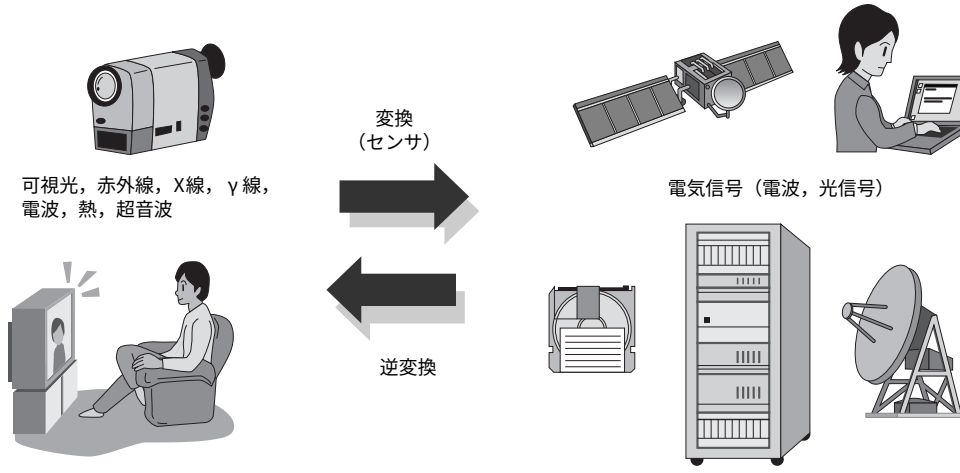

図 1.1
画像・映像信号処理の流れ

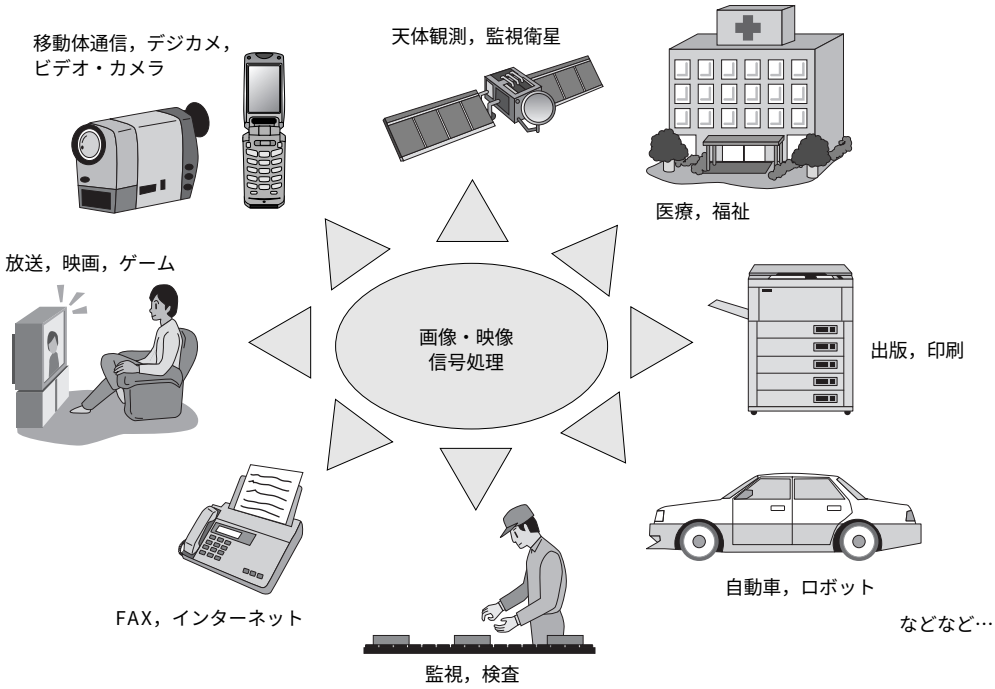

図1.2　画像・映像信号処理の応用例

に変換される．いったん電気信号に変換されれば，その光の分布の様子を伝送，記録，解析，加工することは容易である．さらに標本化・量子化を施せばディジタル信号となり，コンピュータ上での操作が可能となる．高品質・高効率な記録・伝送のほか，暗号化などによりセキュリティを高めることもできる．表示機器によって信号を再び光の分布に戻すことで，原画像・原映像を異なる場所，異なる時刻に再現できる．

なお，画像・映像信号は可視光に限らない．変換するセンサさえ存在すれば，紫外線やX線など，あらゆる物理量の分布が信号処理の対象となる．また，信号が水平方向・垂直方向を軸とする分布や配列として与えられれば「画像処理」，水平方向・垂直方向・時間を軸とする分布や配列として与えられれば「映像処理」とみなせる．

● 画像・映像信号処理の応用例

画像・映像信号処理の応用例は多岐にわたる．**図1.2**に一部の例を掲載する．いずれも画像や映像を扱う応用だが，目的が異なれば撮像の条件や要求される表示の形態も異なる．従って，要求される処理は用途に応じておのずと変わる．

ある用途で優れた方法は，別の用途では役に立たないこともある．複雑な処理を施すよりも，単純な処理の方が視覚的に優れた性能を示すこともある．人間が良いと感じるか否かは，単純な問題ではない．

これが画像・映像信号処理の難しい部分であり，また，経験や発想を生かせる楽しい部分でもある．

● 画像・映像信号の入出力

なぜ，用途に応じて画像・映像信号処理への要求に違いが起こるのか．画像・映像の入出力信号に着目して，その例を示そう．

(1) 撮像機器――入力部

撮像機器は光などの分布を電気信号に変換する．すなわち，画像・映像と処理システムの接点にあたる．現在，主流の撮像機器は，CCD(Charge Coupled Device)やCMOS(Complementary Metal Oxide Semiconductor)などの固体素子を採用し，従来の撮像管による撮像機器と比べて，小型化・軽量化されている．

放送や映画，出版などのプロ向け撮像機器では，高価な光学系を構成し，CCD/CMOSセンサも複数利用して，色や解像度などが高品位な画像・映像を取得する．三板式CCD/CMOSセンサ・カメラがこれにあたる．

一方，安価な民生用カメラや携帯電話付属カメラは，CCD/CMOSセンサ1枚のみで構成される場合がほとんどである．空間解像度を犠牲にし，色情報をカラー・フィルタを通して取得するため，1枚のカラー画像を得るために「デモザイキング」と呼ばれる補間処理が必要となる．

監視カメラや車載カメラは，色情報や高解像度を必要としない場合もある．劣悪な環境にカメラが置かれていれば，ノイズ除去や強調処理が必要となるだろう．固定カメラか可動カメラかの違いによっても動きに関する処理が変わるだろう．

身体の断層画像を直接撮影することはできない．CT スキャナは身体の周りをX 線や核磁気共鳴を利用してスキャンする．その情報から断層画像を得るために，再構成処理が必要となる．

(2) 表示機器――出力部

表示機器は画像・映像信号と人間の接点にあたる．現在，映像の表示機器として電子ビーム系のCRT(Cathode Ray Tube)に変わり，液晶ディスプレイ(LCD：Liquid Crystal Display)やプラズマ・ディスプレイ(PDP：Plasma Display Panel)などのフラット・パネル系が勢力を増している．

放送では，インターレース(飛び越し)走査と呼ばれる映像フォーマットが採用されている．垂直時間解像度を保ちつつ1チャネルの伝送に必要となる帯域を半分にするという技術である．放送がディジタル化されてもインターレース走査は健在である．インターレース走査では，空間の標本位置がフィールド周期という時間間隔でズレている．インターレース走査は，画素に空間的広がりや残像のある電子ビーム系のCRTに比べて，発光を画素単位で制御するLCDやPDPにおいて視覚上の問題を生じる．空間的な標本位置のズレを解消するためのインターレース-プログレッシブ(IP)変換(デインターレース処理)という補間技術が必要となる(**図1.3**)．

画像の印刷についても，レーザ記録，静電記録，感熱記録，インクジェット記録など，手法は多岐にわたる．白と黒の2階調[注1-1]，あるいは限られた色の組み合わせで中間の明るさや微妙な色を表現しなければならない．インクかトナーかの違いも考慮しなければならない．視覚的劣化を抑えつつ限られた階調や色へ変換するハーフ・トーニングという技術が必要となる．

注1-1：階調数は，表現できる明るさの種類を示す．

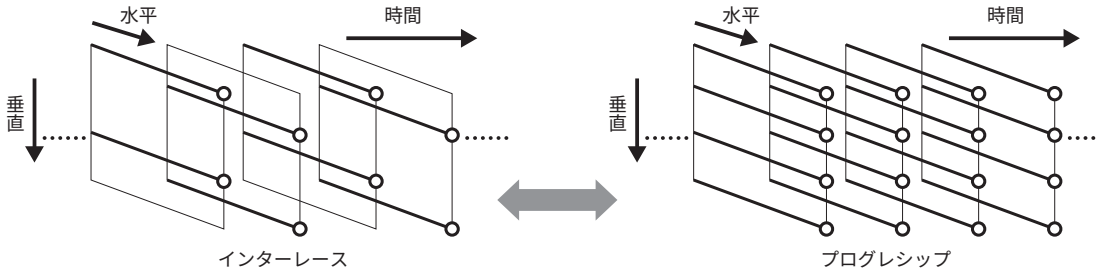

図1.3　インターレース-プログレッシブ変換

最後に，画像・映像信号処理の結果が画像・映像であるとは限らないことにも注意しておこう．特徴を表す数値や内容を示す判別結果が出力となることもある．指紋画像や顔画像による人物認証の結果は，個人を特定する名前やIDで十分である．

1.2　本書のねらい

学ぶべき画像・映像信号処理は用途ごとに変わるのではないか――確かにそうである．しかし，汎用的に役立つ基礎知識や広く利用される要素技術が存在する．本書では，そのような画像・映像信号処理全般において重要と思われる，

- 画素処理
- フィルタリング
- 周波数解析
- 標本化とその変換

について，直感的に理解できることを目標にして解説した．そのため，数値シミュレータMATLABのプログラム例とシミュレーション結果を豊富に掲載した．また，学習に役立つように多くの実習や演習課題も準備した．ぜひとも取り組んで，じっくりと深く理解してもらいたい．なお，**図1.4**に本書の読み進め方の例を示しておく．

● 本書の構成

第2章では，MATLABに慣れていない読者のためにMATLABの基本的な操作について解説した．従って，MATLABの利用経験がある読者は読み飛ばしても差し支えない．

第3章では，画像・映像信号処理について学ぶための準備として，1次元信号処理について解説した．復習という位置付けで，基礎的な内容については詳細の説明を省き，結論を中心にその概要を述べた．MATLABを利用した信号処理シミュレーションの導入も兼ねているので，第4章以降の理解に役立てていただきたい．

第4章から画像・映像信号処理の本格的な解説が始まる．第4章では，MATLABで画像・映像を取り扱うための留意点や機能を解説した．また，ヒストグラム操作のような画素処理や色空間変換といった話題にも触れている．

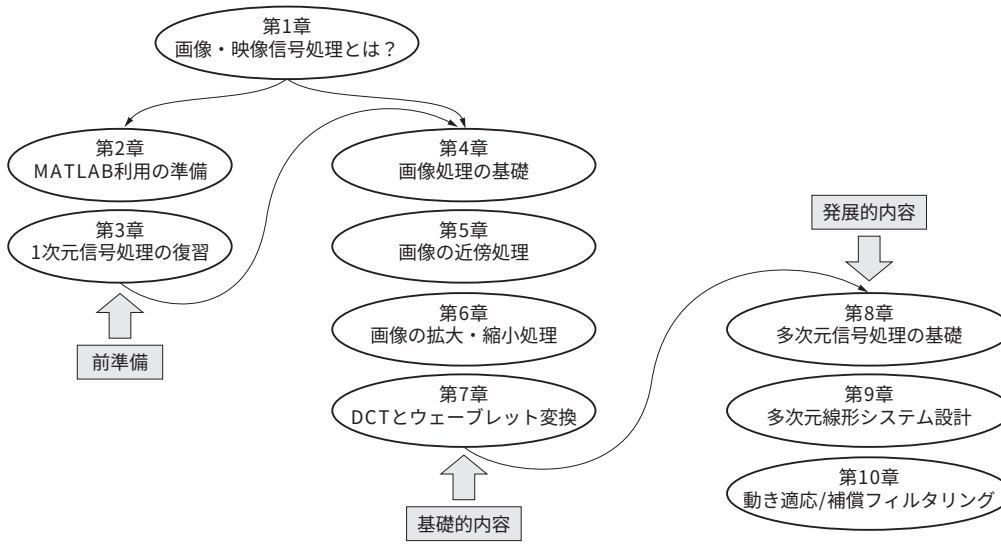

図1.4　本書の読み進め方

第5章では，画像の近傍処理や映像のフレーム間処理について解説した．近傍処理はマスク処理，あるいはフィルタリングとも呼ばれ，もっとも基本的な画像・映像処理技術である．この章では，画像処理の分野で典型的に利用される技法とそれらの特徴について解説した．

第6章では，画像の拡大・縮小について解説した．映像のフレーム・レート変換の例題についても掲載した．マルチレート信号処理の導入も兼ねており，拡大・縮小処理の周波数領域での振る舞いやシステムの設計法，効果的実現法についても解説した．

第7章では，離散コサイン変換(DCT：Discrete Cosine Transform)とウェーブレット変換について解説した．DCTとウェーブレット変換は，符号化をはじめ，画像・映像信号処理において欠かせない要素技術となっている．本章では，第6章で紹介するマルチレート信号処理を用いて，DCTとウェーブレット変換をフィルタリングの観点からも統一的に解説した．

画像・映像信号処理は多次元信号処理である．第8章以降では，1次元信号処理にはない多次元特有の問題について扱った．多くの読者にとっては，第7章までの内容で十分だろう．ただし，面白くなるのはここからである．第8章は，多次元信号処理の基礎的事項についてまとめている．特に，変数のベクトル表記を導入し，周波数解析と標本化について1次元信号処理との類似点と相違点をまとめている．

第9章は「多次元線形システム設計」と題して，主に非分離型のフィルタ設計法を紹介している．各手法についてじっくりと理解するというよりも，多次元フィルタを設計したいとき，リファレンスとして利用すると良いだろう．後半では，非分離型のマルチレート信号処理についても紹介した．これは発展的内容といえよう．

第10章は，映像信号処理における動き情報の重要性をインターレース-プログレッシブ(IP)変換を題材に解説している．固定係数，動き適応処理，動き補償処理という順序でシミュレーションを行

い，性能の向上について確認できるように構成している．

● 表記法

本書では次のような表記を利用する．

- 丸かっこ：連続変数．$x(t)$，$X(j\Omega)$，$X(z)$など
- 四角かっこ：離散変数．$x[n]$，$X[k]$など
- 太字の小文字：ベクトル．$\boldsymbol{x}$，$\boldsymbol{a}$など
- 太字の大文字：配列もしくは行列．$\boldsymbol{X}$，$\boldsymbol{A}$など

● MATLABの環境

本書に掲載したMATLAB のソース・コードは，次の環境で動作確認を行った．

- MATLAB R2006a
- Signal Processing Toolbox
- Image Processing Toolbox

本書に掲載するソース・コードは，基本的にMATLAB本体とSignal Processing Toolboxのみで動作する．ただし，Image Processing Toolboxがある場合についてのコメントも記載し，必要に応じてソース・コードも併記した．

例題で紹介したソース・コードはすべて実習でも活用できるように，`practiceXX_X.m`というファイル名で用意してある．`XX_X`の部分は章番号と各章における実習番号となっている．一部，Image Processing Toolboxの機能を用いた`practiceXX_X_ip.m`も合わせて配布している．以下のWebサイト，

`http://www.cqpub.co.jp/hanbai/books/30/30941.htm`

よりダウンロードして実際に試してほしい．

参考文献

(1) 吹抜 敬彦；画像・メディア工学，コロナ社，2002年．
(2) Rafael C. Gonzalez and Richard E. Woods；Digital Image Processing, 2nd Ed.，Prentice Hall，2001.
(3) 今村 元一；ビデオ信号の基礎とその操作法，CQ出版社，2003年．
(4) Interface編集部 編；ディジタル放送の基礎技術入門，TECH Iシリーズ，CQ出版社，2002年．
(5) Bahadir K. Gunturk, John Glotzbach, Yucel Altunbasak, Ronald W. Schafer, and Russel M. Mersereau；Demosaicking: Color Filter Array Interpolation，IEEE Signal Proc. Magazine，Vol.44，Jan. 2005.
(6) Zhaoui Sun；Video Halftoning，IEEE Trans. on Image Proc.，Vol.15，No.3，pp.678-686，Mar. 2006.
(7) Gerard De Haan, and Erwin B. Bellers；Deinterlacing . An Overview，Proc. of IEEE，Vol.86，No.9，pp.1839-1857，Sep. 1998.

第2章

MATLAB利用の準備

本書ではMATLABのプログラム例を多数掲載し，実際に実行することで，画像・映像信号処理に関する理解を深めてもらうことを目的としている．まず準備として，MATLAB の基本操作について概説したい．本書に掲載するプログラム例を理解して確かめるのに十分な機能のみ，解説する．従って，インストール方法や詳細な利用法については，MATLABのマニュアルなどを参照されたい．なお，MATLABを実行する環境がなくても，ソース・コードを読み解くことができれば，ほかのプログラミング言語による学習も可能であろう．すでにMATLAB利用の経験がある読者は，本章を読み飛ばし，第3章以降に進んでも差し支えない．

2.1　基本操作

以下では，MATLABバージョンR2006aの利用を前提として解説を進める．

● 立ち上げ方と終わり方

(1) 立ち上げ方

Windows系のOSを使用している場合，デスクトップ上にあるアイコン(**図2.1**)をダブルクリックすればよい．すると，**図2.2**のような画面が現れ，すぐに利用できる状態になる．Unix系のOS を使用している場合は，ターミナル上で，

```
matlab
```

と入力すればよい．実際は，使用環境により立ち上げ方が異なることもあるため，使用するコンピュータの管理者に問い合わせるのが良いだろう．

図2.1
MATLABのアイコン

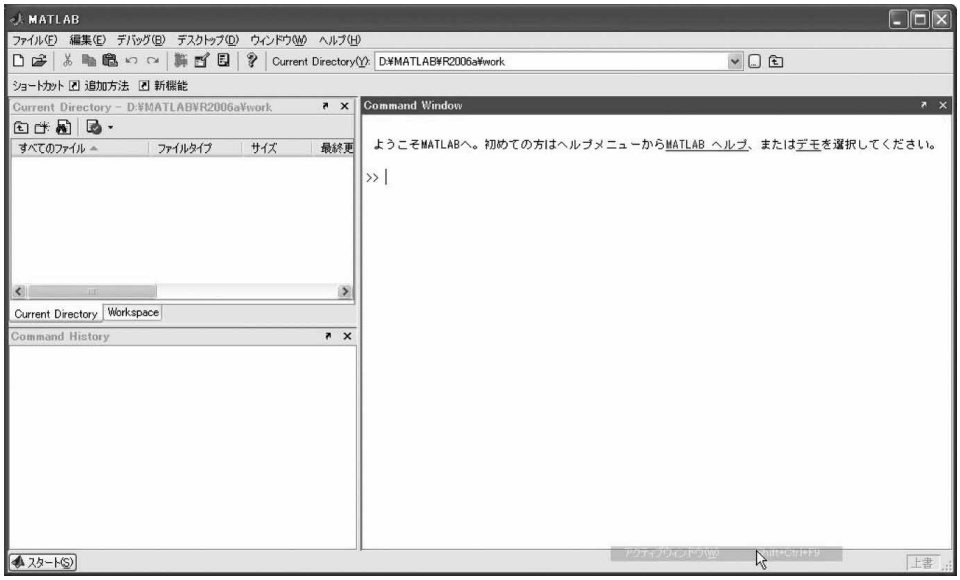

図2.2　起動直後の画面

(2) 終わり方

MATLABのCommand Window上(**図2.2**では右側)のプロンプト記号>>に続いて

```
>> exit
```

と入力することで，MATLABを終了することができる．

● オンライン・ヘルプ

MATLABには豊富な数の関数があらかじめ用意されている．そこで，各関数の利用法の理解にはオンライン・ヘルプ機能は欠かせない．オンライン・ヘルプ機能の使い方について解説を進めよう．

MATLABを終了していたら，立ち上げ直そう．まず，fft関数が何を行う関数なのかを調べてみよう．

Command Window上で，

```
>> help fft
```

を実行する．すると，**図2.3**のような表示が現れる．

ヘルプ文が長く，画面をはみ出してしまう場合には，あらかじめmore　onと設定しておくと便利である．すると，ヘルプ文をはじめとしたCommand Window上の出力結果が1画面分の長さに区切られて表示される．

```
>> more on
>> help fft
```

続きを表示する際には，スペース・キーなどを押せばよい．moreの機能を停止したければ，

```
>> more off
```

とすればよい．

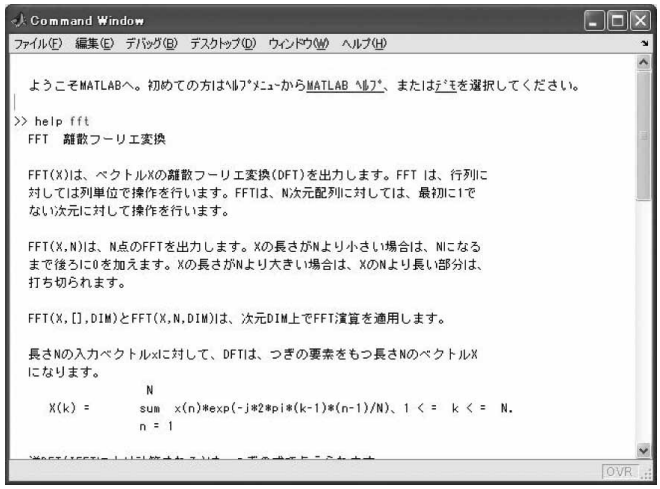

図2.3
オンライン・ヘルプの使用例

例題2.1 **conv関数**

conv関数は何を行う関数か？

解

conv関数は，畳み込み演算や多項式の積演算を行う関数である．

例題2.2 **help .**

help .(ヘルプ・ドット)というコマンドを実行すると，何が表示されるか？

解

help . を実行すると，演算子や特殊記号の説明が表示される．

例題2.3 **help i**

help i(ヘルプ・アイ)というコマンドを実行すると，何が表示されるか？

解

help iを実行すると，複素数の説明が表示される．help j(ヘルプ・ジェイ)，help pi(ヘルプ・ピーアイ)も確かめてみよう．

関数をはじめとするMATLABの操作方法がわからなくなったら，このオンライン・ヘルプ機能を利用すると便利である．より詳細な説明が必要になったら，docコマンドを利用すればよい．例えば，Command Window上で，

```
>> doc fft
```

を実行すると，**図2.4**に示すようなオンライン・ドキュメントを閲覧することができる．

● 行列やベクトルの扱い

MATLABの大きな利点の一つに，行列操作が簡便であることが挙げられる．MATLABにおいて

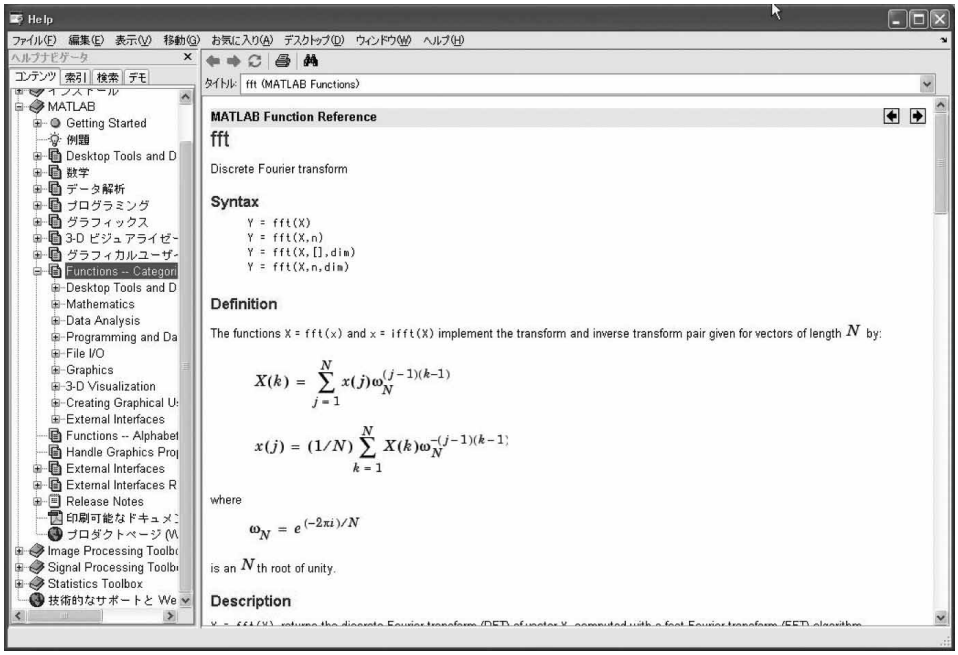

図2.4　オンライン・ドキュメントの使用例

行列やベクトルを扱う方法について解説しよう．

まず，

$$\mathbf{A}=\begin{pmatrix}1 & 2\\ 3 & 4\\ 5 & 6\end{pmatrix} \quad \cdots\cdots (2.1)$$

$$\mathbf{B}=\begin{pmatrix}1 & 2 & 3\\ 4 & 5 & 6\end{pmatrix} \quad \cdots\cdots (2.2)$$

という行列の積$\mathbf{C}=\mathbf{AB}$を求めることを例として，解説を進める．

行列**A**，**B**の定義は，Command Window上で

```
>> A = [ 1 2 ;
         3 4 ;
         5 6 ];
>> B = [ 1 2 3 ;
         4 5 6 ];
```

と入力する．すなわち，行方向は要素間をスペースで区切り，列方向は各行ベクトル間を;(セミコロン)で区切って並べる．

また，行列積$\mathbf{C}=\mathbf{AB}$は，

```
>> C = A * B;
```

表2.1　行列(あるいはベクトル)に対する基本操作

コマンド	内　容
X+Y, X-Y, X*Y	加算，減算，乗算
X.*Y, X./Y	要素ごとの乗算，要素ごとの除算
kron(X,Y)	クロネッカー・テンソル積
X.', X'	転置，エルミート転置
fliplr(X), flipud(X), rot90(X)	行反転，列反転，90度回転
X(:)	列ベクトル化
X(:,iCol), X(iRow,:)	iCol番目列ベクトル，iRow番目行ベクトル
X(:,iCol:jCol)	[X(:,iCol), X(:,iCol+1), ..., X(:,jCol)]
X(iRow:jRow)	[X(iRow,:); X(iRow+1,:); ...; X(jRow,:)]

と実行する．すると，Cに演算結果が代入される．Command Window上で

```
>> C
```

のように末尾に;(セミコロン)を付けずに変数名Cを入力することで，

```
C =
     9 12 15
    19 26 33
    29 40 51
```

のように表示され，行列**C** の内容を確かめることができる．

ベクトルは列数が1，あるいは行数が1の行列と考えられるので，同じように操作できる．

行列(あるいはベクトル)X，Yに対する基本的な演算操作を**表2.1** にまとめる．

例題2.4　行列転置

行列**B**の転置と行列**A**の転置の積**D**を計算してみよう．

解

Command Window上で

```
>> D = B.' * A.';
>> D
```

と入力すると，

```
D =
     9 19 29
    12 26 40
    15 33 51
```

のように表示される．

例題2.5　行列要素の選択

行列**A**の上の2本の行ベクトルからなる正方行列と，行列**B**の左にある2本の列ベクトルからなる正方行列との和**E**を計算してみよう．

解

Command Window 上で

```
>> E = A(1:2,:) + B(:,1:2);
>> E
```

と入力すると，

```
E =
     2  4
     7  9
```

と表示される．

なお，MATLABでは行列要素のインデックスは1から始まることに注意されたい．

2.2 ループ処理と条件分岐処理

MATLABでは，ループ処理のためにfor文とwhile文が利用できる．また，条件分岐のためにif文とswitch文が利用できる．for文とif文の使用法を，簡単な例を用いて解説しよう．

● ループ処理

まず，for文を用いたループ処理の例を見てみよう．例えば，

$$s = \sum_{n=0}^{10} n \quad \cdots\cdots (2.3)$$

という演算は，Command Window 上で

```
>> s = 0; % s の初期値設定
>> for n = 0:10 % n を0 から10 まで繰り返す
     s = s + n; % s にn の値を累積加算
   end
```

と実行できる．

```
>> s
```

と入力すると

```
s =
    55
```

と表示される．for文は，forの直後に記述された変数の仕様に従い，forとendの間の処理を繰り返す．for文，while文の詳細については，help for，help whileを参照されたい．なお，%から行末まではコメント文となり，処理には影響しない．

例題2.6　for文

k行n列目の要素が，

$$[\mathbf{C}]_{k,n} = \cos\left(\frac{k(2n+1)\pi}{8}\right), \quad k, n = 0, 1, 2, 3 \quad \cdots\cdots (2.A)$$

と与えられる4×4行列$\mathbf{C}$を生成しよう．cos関数，定数piを用いてもよい．ただし，MATLABでは行列のインデックスが1から始まることに注意されたい．

解

Command Window上で

```
>> for iCol=1:4
       n = iCol - 1;
       for iRow=1:4
           k = iRow - 1;
           C(iRow,iCol) = ...
               cos(k*(2*n+1)*pi/8);
       end
   end
```

と実行すればよい．

```
>> C
```

と入力すると，

```
C =
 1.0000  1.0000  1.0000  1.0000
 0.9239  0.3827 -0.3827 -0.9239
 0.7071 -0.7071 -0.7071  0.7071
 0.3827 -0.9239  0.9239 -0.3827
```

と表示される．

なお，複数行にまたがるコマンドは，...の後に改行しなければならない．

● 条件分岐処理

次に，if文を用いた条件分岐処理について解説しよう．例えば，$n = 0$ならば$d = 1$，$n \neq 0$ならば$d = 0$，それ以外ではdは値をとらないという処理

$$d = \begin{cases} 1 & n = 0 \\ 2 & n \neq 0 \\ \phi & \text{その他} \end{cases} \quad \cdots\cdots (2.4)$$

は，あらかじめ変数nに値を代入した上で，

```
>> if n==0 % n が0 ならば
       d = 1; % d は1.
   elseif n~=0 % n が0 ではなければ,
       d = 0; % d は0.
```

```
else % それ以外では，
  d = []; % d は空．
end
```

のように実行できる（例えばあらかじめn=[]が設定されていれば，dは空となる）．if文はifの直後にある条件式に従い，条件分岐を行う．条件が満たされた場合，ifに続く処理が実行され，条件が満たされなかった場合，else以下の処理が実行される．elseifは，条件付きのelse文である．if文の詳細についてはhelp ifを参照されたい．また，条件式として使用される比較演算子，論理演算子については，help relopを参照されたい．

例題2.7　条件分岐

例題2.6で与えられた4×4行列**C**の1行目の行ベクトルに係数1/2を乗じ，残りの行ベクトルに係数$1/\sqrt{2}$を乗じて，4×4行列**D**を生成してみよう．sqrt関数を用いてよい．

解

例題2.6の解答に示したコマンドの実行直後に

```
>> for iRow=1:4
     k = iRow - 1;
     if k==0
       D(1,:) = C(1,:)/2;
     else
       D(iRow,:) = ...
         C(iRow,:)/sqrt(2);
    end
  end
```

と実行すればよい．

```
>> D
```

と入力すると，

```
D =
   0.5000   0.5000   0.5000   0.5000
   0.6533   0.2706  -0.2706  -0.6533
   0.5000  -0.5000  -0.5000   0.5000
   0.2706  -0.6533   0.6533  -0.2706
```

と表示される．なお，この行列**D**は，第7章で詳しく解説する離散コサイン変換（DCT：Discrete Cosine Transform）行列と呼ばれる直交行列である．

if文のほかにもswitch文によって処理の分岐を実現できる．詳細についてはhelp switchを参照されたい．

2.3 グラフ表示

MATLABには，便利なグラフ表示機能が多数用意されている．以下では，信号の周波数特性表示を例として，グラフ表示に使うstem関数とplot関数の使い方を解説しよう．なお，周波数特性については次章においてあらためて解説を行うので，ここではグラフ表示の一例として読み進めてもらいたい．

● stem関数

まず準備として，rand関数を使って適当な8×1ベクトル**x**を生成しよう．Command Window上で

```
>> x = rand(8,1);
```

と入力すればよい．直後に

```
>> x
```

と入力すると，例えば

```
x =
    0.9501
    0.2311
    0.6068
    0.4860
    0.8913
    0.7621
    0.4565
    0.0185
```

のように表示される．rand関数は一様分布の疑似乱数を生成する関数なので，実際には上記のものと異なることがある．

では，ベクトル**x**のstem表示を行ってみよう．Command Window上で

```
>> stem(x)
```

と入力すると，**図2.5**のような表示が得られる．stem表示は，離散時間信号(数列)を表すのに便利な機能である．

● plot 関数

では次に，plot関数を利用してベクトル**x**の周波数振幅特性を見てみよう．Command Window上で

```
>> A = 20*log10(abs(fft(x,512)));
>> plot(A)
```

と入力すると，**図2.6**のような表示が得られる．1行目では，**x**の振幅特性([dB])を512点FFT(Fast Fourier Transform)を用いて求めている．2行目が実際にグラフ表示を行っている部分である．plot表示はデータ間を直線で補間してグラフを表示する．オプションの指定によって，点のみの表示も

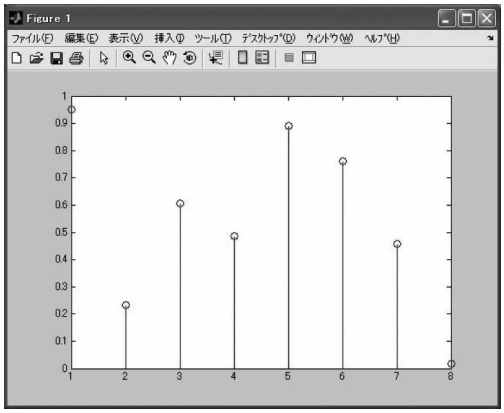

図2.5　stem 表示の例

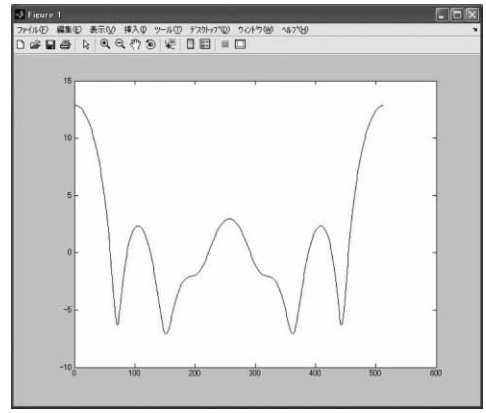

図2.6　plot 表示の例

可能である．詳細については`help plot`を参照されたい．

例題2.8　**plot表示**

同じサイズのベクトル変数x，yを用いて`plot(x,y)`を実行すると，*x*-*y*プロット表示がなされる．変数*t*を0.0から6.3まで0.1刻みで変えながら，関数*f*(*t*) = cos(*t*)を*x*-*y*プロットしてみよう．

解

Command Window上で

```
>> for i = 0:63
     t(i+1) = i/10;
     f(i+1) = cos(t(i+1));
   end
>> plot(t,f)
```

もしくは，

```
>> t = 0.0:0.1:6.3;
>> f = cos(t);
>> plot(t,f)
```

と入力すればよい．

後者の解について解説しよう．

- 1行目は，0.0 ～6.3まで0.1刻みの値を要素とするベクトル変数**t**を生成する．
- 2行目は，ベクトル変数**t**の各要素に対するcos関数の演算を行い，各要素に対応する演算結果をベクトル変数**f**へ代入する．
- 3行目で，ベクトル変数**t**，**f**を用いた*x*-*y*プロット表示を行う．

このように，MATLABの表現をうまく利用することで，for文を省いた簡潔な記述が可能となる．

例題2.9 plot表示

ディジタル信号は周期的な周波数特性を有する．横軸の1周期を2πと正規化して，先のベクトル**x**の周波数振幅特性を表示してみよう．

解

Command Window上で

```
>> f = 0:2/512:2-2/512;
>> A = 20*log10(abs(fft(x,512)));
>> plot(f,A)
>> xlabel(['Normalized Frequency ',...
          '(¥times¥pi rad)'])
>> ylabel('Magnitude (dB)')
```

と実行すればよい〔Unix系OS上では，¥は\(バックスラッシュ)に読み換える〕．

2.4 Mファイルの作成

ここまで読み進めると，MATLABの使い方に慣れると同時に，Command Window上での作業に煩わしさを感じるだろう．この煩わしさは，次に解説するMファイルの作成法と使用法を覚えることで解消される．

● スクリプト

MATLABでは，Mファイルと呼ばれるスクリプト・ファイルによって一連の処理をまとめることができる．ここでは，MATLAB付属のエディタなどを使用し，ampres.mというファイルを作成してみよう．

MATLABのエディタを利用する場合は，Command Window上で

```
>> edit ampres
```

と実行すればよい．また，使い慣れたテキスト・エディタを利用してもよい．ただし，ファイルの拡張子は .m とすること．

以下の内容をファイルampres.mに記述しよう．

```
x = rand(8,1);
f = 0:2/512:2-2/512;
A = 20*log10(abs(fft(x,512)));
plot(f,A)
xlabel(['Normalized Frequency ',...
        '(¥times¥pi rad)'])
ylabel('Magnitude (dB)')
```

続いてCommand Window上で

```
>> ampres
```
と実行してみよう．すると，横軸を正規化した周波数振幅特性が表示される．

● 関数定義

先のスクリプト・ファイルでは，入力xやFFT点数を変えたいときに，いちいちファイルを編集しなければならない．MATLABでは，スクリプト・ファイルをさらに発展させて，独自の関数を定義することができる．先ほど作成したファイルampres.mを以下のように編集しよう．

```
function A = ampres(x,nPoints)
% AMPRES: Display Amplitude Response
% Copyright (c), 20XX, (Your name),
%             All rights reserved
f = 0:2/nPoints:2-2/nPoints;
A = 20*log10(abs(fft(x,nPoints)));
plot(f,A)
xlabel(['Normalized Frequency ',...
        '(¥times¥pi rad)'])
ylabel('Magnitude (dB)')
```

続いて，Command Window上で

```
>> x = rand(8,1);
>> A = ampres(x,512)
```

と実行してみよう．すると周波数振幅特性が表示される．入力ベクトルxやFFT点数nPointsを変えることができるので，試してみよう．

例題2.10　Mファイル

周波数特性は複素数として与えられ，位相特性も持つ．位相特性を表示するphsres関数も作成してみよう．

解

ファイル名：phsres.m

```
function P = phsres(x,nPoints)
% PHSRES: Display Phase Response
% Copyright (c), 20XX, (Your name),
%             All rights reserved
f = 0:2/nPoints:2-2/nPoints;
X = fft(x,nPoints);
P = 180*angle(X)/pi;
plot(f,P)
xlabel(['Normalized Frequency ',...
```

```
        '(¥times¥pi rad)'])
  ylabel('Phase (degrees)')
```

angle関数についてはオンライン・ヘルプを参照されたい．

例題2.11　Mファイル

subplot関数を用いると，一つのグラフ内に複数のグラフを表示できる(help subplotを参照)．振幅特性と位相特性を同時に表示するfreqres関数を作成してみよう．

解

ファイル名：freqres.m

```
function [A,P] = freqres(x,nPoints)
% FREQRES: Display Frequancy Response
% Copyright (c), 20XX, (Your name),
%             All rights reserved
f = 0:2/nPoints:2-2/nPoints;
X = fft(x,nPoints);
A = 20*log10(abs(X));
P = 180*angle(X)/pi;
subplot(2,1,1);plot(f,A)
xlabel(['Normalized Frequency ',...
        '(¥times¥pi rad)'])
ylabel('Magnitude (dB)')
subplot(2,1,2);plot(f,P)
xlabel(['Normalized Frequency ',...
        '(¥times¥pi rad)'])
ylabel('Phase (degrees)')
```

例題2.12　オンライン・ヘルプ

オンライン・ヘルプ機能を使ってhelp ampres，help phsres，help freqresを調べてみよう．

解

functionに続く，コメントアウトされた行の内容が表示される．

2.5　変数の保存と読み込み

Command Window上で利用した変数は，ワークスペース(Workspace)上に保持されている．しかし，ワークスペース上のデータはMATLABを終了するたびに消えてしまう．以下では，ワークス