数値演算ソフトウェア GNU Octave Ver.3 による信号処理

Signal processing using GNU Octave Ver.3

苣木　禎史[*1]
Yoshifumi CHISAKI

[*1]熊本大学
Kumamoto University

This paper introduces a high-level interactive language for numerical computations. The software, GNU Octave Ver.3, is mostly compatible MATLAB. Installation and how to use the Octave are mentioned. Moreover, examples for signal processing is explained using some toolboxes.

1. はじめに

GNU Octave(以下，Octave) は，数値計算のための高級言語である．様々な分野の関数群があり，アルゴリズム開発におけるの前段階での効率的な作業が期待できるソフトウェアである．信号処理分野では，自己相関や高速フーリエ変換に関する関数群，窓関数群などがある．ほかに線型代数，多項式，統計，画像処理に関する関数群がある．可視化は，Octave から gnuplot を呼び出すことによりシームレスに可能である．

人工知能学会誌のシリーズ特集「研究のツールボックス」では，Octave Ver.2 について述べた [Chisaki 05]．Version 3 になり，より一層商用ソフトウェア MATLAB のコマンド群への対応が広がっており，商用ソフトウェアとのソースコードの共有化により，効率的な研究の道具となりつつある．主な変更点は，MATLAB に近いグラフィックス関数，関数群のサーチパスの設定，Warning 制御，system 関数の取り扱い，ビルトイン変数の関数化などである．

Octave は，GPL(GNU General Public License) に基づき無償で利用できるソフトウェアであり，Linux と組み合わせることで，自宅でも無償で利用できるメリットがある．また，1988 年から開発が始まっており，1994 年の Ver.1 のリリースから長期に渡って着々と開発が進むに伴い，開発利用者が拡大したため web での情報検索によっても多くの情報が得られる．さらに，いくつかの研究分野での活用を解説した書籍もあり，自学できる環境が整っている．

本論文では，研究での活用への第一歩となる，インストール，基本的な操作方法，信号処理への応用例を取りあげる．

2. インストール

配布はソースコードと実行形式のバイナリの両者が配布されており，実行形式をダウンロードを行い，ダブルクリックすることによりインストールが可能である．

2.1 Linux へのインストール

Linux へのインストール方法について述べる．ここでは，Ubuntu 9.10 へのインストールを示すが，昨今は多くのディストリビューションでソフトウェアパッケージ管理のアプリケーションが導入されており，それらを用いて下記と同様に簡単にインストールできる．

ソースコードをダウンロードし，コンパイルを行ってインストールも可能であるが，複数のソフトウェアパッケージを手順良くインストールする必要がある．ここでは，APT (Advanced Package Tool) を用いたインストール方法を以下に示す．

```
chisaki@ubuntu910: $ sudo apt-get install octave3.2
パッケージリストを読み込んでいます... 完了
依存関係ツリーを作成しています
状態情報を読み取っています... 完了
以下の特別パッケージがインストールされます:
  gnuplot gnuplot-nox~gnuplot-x11 groff
   imagemagick libamd2.2.0 libarpack2
(中略)
アップグレード: 0 個、新規インストール: 37 個、削除: 0 個、保留: 0 個.
38.5MB 中 0B のアーカイブを取得する必要があります.
この操作後に追加で 113MB のディスク容量が消費されます.
続行しますか [Y/n]? Y
パッケージからテンプレートを展開しています: 100%
パッケージを事前設定しています ...
(中略)
libc-bin のトリガを処理しています ...
ldconfig deferred processing now taking place
chisaki@ubuntu910:~$
```

ここに示されているように，Octave で必要とされるアプリケーション群も自動的にインストールされる．

MATLAB のような GUI 画面で操作を行うために，Qt を用いたフロントエンド qtoctave もある．ただし，Qtoctave を利用するは，Ubuntu 9.10 では Ver.3.0.5 がインストールされる．Qtoctave のインストールは，

```
> sudo apt-get install qtoctave
```

でインストールを行い，

```
> qtoctave
```

と入力して起動する．

オンラインドキュメントを利用するには，

```
> sudo apt-get install octave3.2-info
```

を実行しておく必要がある．

連絡先: 苣木 禎史，熊本大学 大学院自然科学研究科 情報電気電子工学専攻，〒 860-8555 熊本市黒髪 2-39-1，chisaki@cs.kumamoto-u.ac.jp

2.2 MacOSX へのインストール

MacOSX 用の実行形式が [Octave-sourceforge] の Octave.app for Mac OS X から入手できる． octave-3.2.3-i386.dmg が原稿執筆時の最新版である．この DMG 形式をダブルクリックすると実行形式が現れるので，それをアプリケーションフォルダにコピーするとインストール完了である．また，MacPorts で配布されているパッケージも port コマンドでインストールできる．

2.3 Microsoft Windows へのインストール

Microsoft Windows(以下，Windows) 用のインストーラーが [Octave-sourceforge] の Windows installer から入手できる．原稿執筆時の最新版は Octave-3.2.4_i686-pc-mingw32_gcc-4.4.0_setup.exe である．また，Cygwin ユーザーはパッケージとしてインストール可能である．

3. 基本的な利用方法

3.1 PATH の設定

Octave は，ビルトイン（内部）コマンドと外部コマンドとして動作する関数ファイルから成る．Octave のインストールと同時に Octave に同梱されている関数群が導入される．これらの関数群は既に PATH が通っており直ぐに利用できる．

ユーザーが導入した関数は，関数のファイルがあるディレクトリに PATH を通すことにより利用できる．

具体的には次の通りである．ユーザーのホームディレクトリに .octaverc をエディタで作成し，

```
addpath(genpath ("\$TARGET\_DIR1;\$TARGET\_DIR2"));
```

と記述すると，Octave 起動時に .octaverc が読み込まれ，PATH に$TARGET_DIR1 と$TARGET_DIR2 の順番に設定され，その後に Octave に同梱されている関数群があるディレクトリが PATH に加えられる．具体的には，ホームディレクトリ以下に， /octave/m/func1，/octave/m/func2，/octave/m/func3, のディレクトリを作成し，そのディレクトリ以下に関数を導入したら，

```
addpath(genpath ("~/octave/m/func1;~/octave/m/func2;
                          ~/octave/m/func3"));
```

とすればよい．PATH が通っている確認は，Octave を起動後に path コマンドで行う．

3.2 起動と終了

Octave を起動するには，shell のコマンドプロンプトが表示されている状態で，

```
> octave
```

と入力すると，下記のように Octave が起動する．

```
> octave
GNU Octave, version 3.2.2
Copyright (C) 2009 John W. Eaton and others.
This is free software; see the source code for copying
conditions.
There is ABSOLUTELY NO WARRANTY; not even for
MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE.  For details,
type 'warranty'.

Octave was configured for "i486-pc-linux-gnu".

Additional information about Octave is available
at http://www.octave.org.

Please contribute if you find this software useful.
For more information,
visit http://www.octave.org/help-wanted.html

Report bugs to <bug@octave.org>
(but first, please read
http://www.octave.org/bugs.html to learn
how to write a helpful report).

For information about changes from
previous versions, type 'news'.

octave:1>
```

上記で示されているように，Octave 自身の情報はホームページ [Octave] で入手できる．

また，

```
octave > news
```

と入力すると，前バージョンからの改変情報が表示される．

Octave の終了は，exit, quit または，Ctrl+d を入力する．

3.3 オンラインヘルプ

コマンドの引数の数やそれらの意味などは，help 関数名で得られる．例えば plot コマンドの場合，

```
octave > help plot
```

で確認できる．

また，doc コマンドでマニュアルを読むことができる．その際には octave3.2-info をインストールしておく必要がある．

3.4 四則演算などの基本的な演算

Octave は行列演算が可能であり，それの要素ごとの演算も可能である．利用できる演算子の一部を表 1 に示す．

表 1: 演算子の例.

! or ~	Not
+	Addition
++	Increment
*	Multiplication
.*	Element by element – Multiplication
=	Assignment
==	Equality test

これらの演算子を利用した例を下記に示す．

例)

行列

$$a = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

を定義する．

```
octave > a = [ 1 2 ; 3 4 ]
```

括弧内のセミコロンは行を区切るセパレータである．

```
octave > a [return]
```

とすると，

```
octave > a = [ 1 2 ; 3 4 ]
ans =
 1 2
 3 4
```

と表示される．ここで演算子 ++ を使ってみると，

```
octave > ++a
ans =

 2 3
 4 5

octave >
```

となる．入力の最後にセミコロンを加えると，演算の結果をコンソールに表示しない．具体的には，

```
octave > b = [ 5 6 ];
octave >
```

となる．行列の足し算，掛け算は次のようになる．

```
octave > a + a
ans =

 4  6
 8 10

octave > a * a
ans =

 16 21
 28 37

octave >
```

スカラー倍にするには， .* の演算子を用いる．

```
octave > a .* a
ans =

   4  9
  16 25

octave >
```

3.5 定数

ビルトイン定数は，例えば，pi が 3.1415.... として定義されている．他には，i, j, I, J(虚数)，e(自然対数の底)，NaN(Not a number)，Inf(無限大) などがある．ここで示した定数を定義する文字を変数とすると定数が上書きされるため用いてはいけない．

3.6 予約語

予約語として，if - else or elseif - endif, for - endfor, while - endwhile など，他の多くの言語で利用されているものが利用できる．例えば，for の使い方は help コマンドを使用すると，

```
octave> help for

 -- Keyword: for I = RANGE
     Begin a for loop.
          for i = 1:10
           i
          endfor

     See also: do, while

Additional help for built-in functions and operators is
available in the on-line version of the manual.  Use the command
‘doc <topic>’ to search the manual index.

Help and information about Octave is also available on the WWW
at http://www.octave.org and via the help@octave.org
mailing list.
octave>
```

と標準出力され，具体例とともに確認できる．

3.7 付属の関数群

Octave のインストールと同時にインストールされる関数群は，ビルトイン関数である三角関数なども含め様々なものが用意されている．MATLAB で利用されている関数名と全く同じものもあるが，関数によっては引数の値の行と列が入れ替わっているものなどがあり，完全に互換性があるとはいえない．インストール時に導入される関数群以外に，無償で配布されているパッケージも追加できる．また，関数は自分で作成可能である．

同時にインストールされる関数群の一部を表 2 に示す．信号処理に関する関数群 signal には，自己相関，窓関数，高速フーリエ変換などがある．また，スペクトログラムなどを表示や画像処理のための画像関数群 image も充実している．さらには，線形代数，統計処理などに関する関数群もあり，Octave は様々な分野のデータ処理をはじめて学ぶにためのツールとして十分に機能する．

表 2: 関数群の例.

audio (音響)	音ファイル入出力, 録音再生，線形圧縮変換 他
image (画像)	画像ファイル入出力, イメージ変換及び表示 他
io (入出力)	標準出力への書式付き出力 他
linear-algebra (線形代数)	ランク，トレースの計算 他
signal (信号処理)	窓関数，高速フーリエ変換, 畳み込み 他
special-matrix (特殊行列)	ヒルベルト行列 他
statistics (確率)	累積分布，確率密度, ロジスティックモデル 他
strings (文字列)	文字列操作，進数変換 他

3.8 ユーザ定義関数

ユーザ定義関数は以下の手順で作成できる．

1. ファイルのパスを ~/.octaverc で記述する．
2. function〜endfunction の間に記述する．
3. %% を用いて online help を記述する．(省略可)

なお，Ver.2 で利用できたファイルのパスを指定する LOADPATH は使えないため，Ver.3 では先に説明した addpath 関数を利用する．

例として円の面積を表示する関数ファイルを以下に示す．この関数は引数で円の半径を指定し，面積を計算する．引数の個数が 1 以外のときは使用法を表示し，終了する．

最初の %% は，コメントに相当する．空行で挟んでいる%% header part 2 の領域は，octave の help コマンドで利用されるオンラインヘルプである．例えば，help fftconv と入力して表示されるメッセージと，/usr/share/octave/3.2.2/m/signal/fftconv.m に記述されているコメントを比較すればわかる．

circle_area.m

```
%% header part 1
%% (ライセンスなど)

%% header part 2
%% (ヘルプメッセージ)

%% header part 3
%% (作者・作成日など)

function y = circle_area (r)

if (nargin != 1)
usage
        ("circle_area (radius)");
else
y = pi * r * r;
endif

endfunction
```

4. バッチ処理

シミュレーションなどでは，アルゴリズムを固定し，パラメータを変更することがある．その様な場合，幾度も同じことをコマンドラインで入力するのではなく，バッチファイルを作成し，それをコマンドラインで利用することで効率化を図れる．

エディタで作成し保存した，バッチファイルの例を bat1.m に示す．

```
for k = 1 : 10 : 50 %初期値 1，終了値 50，間隔 10
   kk = k * k;
endfor
```

これを実行するには，以下のように Octave のコマンドラインで bat1 と入力する．このバッチにより繰り返し作業が効率的に行える．

```
octave > bat1   % 1 回目
kk = 1
kk = 121
kk = 441
kk = 961
kk = 1681
octave > bat1   % 2 回目
kk = 1
kk = 121
kk = 441
kk = 961
kk = 1681
octave >
```

5. 可視化

処理結果を容易に可視化できることも Octave の特徴である．これは gnuplot をバックエンドで動作させることで可能としている．Octave Ver.3 では，可視化の関数が大きく変更されている．gnuplot に関する説明は，[Kawano] や [Yabuki 96] にあるので詳細はそちらに譲る．gnuplot では set コマンドを利用して様々な制御を行っている．一方，Octave Ver.2 では set に相当する gset を用いていたが，Ver.3 から大きく変更され，set 関数は図を制御するための別の意味をもつ関数となった．図の制御は，図をオブジェクトとして扱う概念が導入され，図はハンドル番号により管理され，ハンドル番号を指定することで表示先の位置，ラベル，メモリなどが制御可能となる．

例題として，正規分布に従う乱数を発生する関数 randn, ヒストグラムを表示する hist 関数，グラフにタイトルをつける title 関数を用いてグラフを描いてみる．以下を実行すると図 1 のように表示される．各関数の引数の説明や利用の仕方は，help 関数名 (例えば help randn) と入力することにより確認できる．.

histogram.m

```
title( "Histgram" );
      % タイトルを設定.
      %  gnuplot の set title で同様のことができる.
randn( "seed", 0 );
      % 乱数初期化
hist(randn( 1, 2 ^ 21 ), ( -5 : 0.01 : 5 ), 1 );
      % ヒストグラムの表示
```

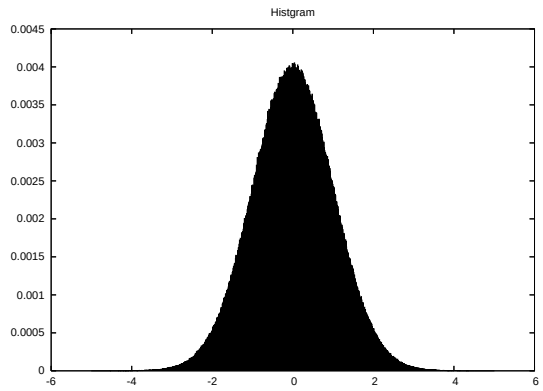

図 1: グラフ (ヒストグラム) の表示.

次に，ビルトイン関数 sin, cos を用いて，正弦波，余弦波を表示させる．一行目で変数 t に初期値 0，終了値 0，間隔 0.1 の行列を作成する．二行目で，引数 1 は x 軸のデータ列で t を与えている．引数 2 には y 軸のデータであり，t のそれぞれの要素値に対応する cos 関数の計算値を与えている．また，引数 3 はレジェンドのラベルである．同様に，引数 4〜6 も x 軸,y 軸のデータ，そしてラベルを与えている．

plot_sincos.m

```
t = 0 : 0.1 : 6.3;
      % 時間データ列の作成
plot (t, cos(t), "-;cos(t);", t, sin(t), "+3;sin(t);");
      % ビルトイン関数 sin,cos による表示
```

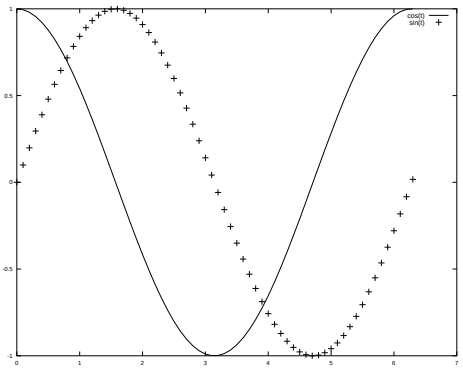

図 2: ビルトイン関数 sin,cos によるグラフ.

6. 音響信号処理での実践

ここでは，実践例題として，音声の時系列信号を周波数領域に変換し，時系列信号ではわかりにくい周波数成分を可視化によって確認する流れを，正弦波を用いてその手順を示す．

6.1 窓掛け処理

高速フーリエ変換の前処理である窓処理に必要な窓関数について述べる．音響信号処理では離散時間でのデータをフレーム単位で処理することがある．その第一歩として，1 フレームの信号を作成して，信号と窓関数をたたみ込む窓掛けを行い，その様子をグラフで確認する例を以下に示す．1 フレームの幅であるタップ長は 1024，窓は Hanning を用いる．サンプリング周波数は 10 kHz である

音響信号処理に関しては，MATLAB を用いて解説を進める [Adachi 02] などを参考にしていただきたい．

conv_window.m

```
WL=1024
      % 代入．セミコロンがないので代入値が表示される．
wn=hanning(WL);
      % hanning 窓の係数を計算して wn に代入．
subplot(3,1,1);
      % 縦 3 分割，横 1 分割のグラフエリアを確保し，
        1 番目の領域を指定．
title("Hanning window");
      % グラフのタイトルを設定．
plot(wn);
      % グラフの表示．
axis([1,1024,-1,1]);
      % グラフの x，y 軸の表示範囲を設定．
subplot(3,1,2);
      % グラフエリア内の 2 番目の領域を指定．
title("Sinusoidal wave");
      % グラフのタイトルを設定．
dt = 0 : 1 / 10000 : 1 / 10000 * (WL -1 );
      % 離散時間の作成．(1024 サンプル分)
      %  dt = linspace(0,1/10000 * (WL-1) , WL);
      % でも可能．
y = 1.0 * sin ( 2 * pi * 100 .* dt + 0 );
      % 振幅 1，周波数 100 Hz，初期位相 0 rad．の正弦波．
      % y = sinetone(100, 10000,
      % 1/10000 * ( WL - 1), 1.0); でも可能．
      %式中の .* は行列の要素毎の積演算．
plot(y);
axis([1,1024,-1,1]);
      % グラフの x，y 軸の表示範囲を設定．
subplot(3,1,3);
title("Weighted signal");
plot( y .* wn' );
axis([1,1024,-1,1]);
      % グラフの x，y 軸の表示範囲を設定．
```

図 3 に窓掛け処理の結果を示す．一番上から順に，窓関数，観測信号，窓掛けを施した信号であり，意図したとおりに処理できていることが確認できる．Ver.2 以前では，x 軸, y 軸の範囲は，

```
gset xrange[1, 1];
gset yrange[-1, 1];
```

としていたが，Ver.3 では MATLAB と同じく，

```
axis([1,1024,-1,1]);
```

で設定している．

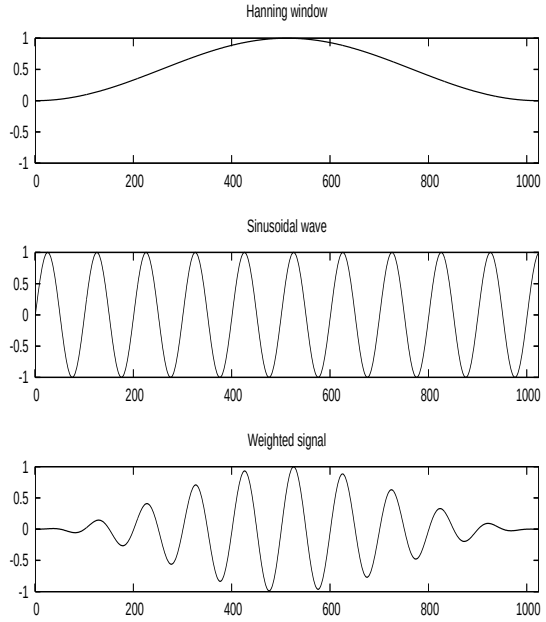

図 3: 窓掛け処理の例 (multiplot を利用).

6.2 高速フーリエ変換

窓掛け処理が施されたフレーム信号を高速フーリエ変換するには，以下の手順を踏む．図 4 にその結果を示す．

conv_window_fft.m

```
WL=1024;
wn=hanning(WL);
dt = 0 : 1 / 10000 : 1 / 10000 * (WL -1 );
y = 1.0 * sin ( 2 * pi * 100 .* dt + 0 );
fftrst = fft( y .* wn' );
subplot(3,1,1);
title("Real part");
plot(real(fftrst));
subplot(3,1,2);
title("Imaginary part");
plot(imag(fftrst));
fftrst_pow = abs(fftrst) .^ 2;
subplot(3,1,3);
title("Power");
plot( 10 * log10( fftrst_pow / max( fftrst_pow )));
      % 最大値で割り， 0 dB を設定して表示．
```

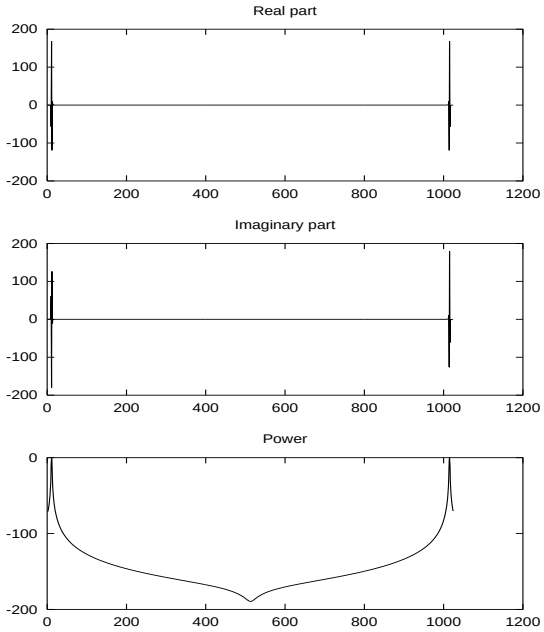

図 4: STFT 処理の例.

次に，信号の時間の変化に伴う周波数特徴を確認できるスペクトログラムについて述べる．これは信号の一部を切り出して短時間フーリエ変換を行う．そして，時間軸方向に一定量切り出しフレームを移動させて，短時間フーリエ変換を行うという手順を繰り返すことよって得られる．

短時間フーリエ変換は stft 関数で可能であり，時間軸，周波数軸，そしてパワー軸を表示するために image 関数を用いる．ここでは，サンプリング周波数 10 kHz にて，1 kHz の正弦波を短時間フーリエ変換する．縦軸は周波数であり，下から 0 Hz 〜 5 kHz である．横軸は時間であり，パワーの大きさを濃淡によって表すグラフである．ソースコードにある stft(x, 64, 16, 1024) は，窓のサイズを 64 タップ，フレームシフト量を 16 タップ，正の周波数 bin の個数を 1024 で信号を x を解析する．図 5 が，上記の条件で分析した結果であり，約 1 kHz の周波数だけ大きなパワーが持続していることがわかる．次に，1 kHz の正弦波にランダムノイズを加えた信号のスペクトログラムを図 6 に示す．最後に，さらに雑音のエネルギーを増やした場合の結果を図 7 に示す．以上 3 つの結果から，全周波数帯域にエネルギーを持つランダムノイズが増加していく様子がわかる．

stft_addnoise.m

```
mkdir figs
      % カレントディレクトリに figs ディレクトリを作成.
SF = 10000;
      % サンプリング周波数を 10 kHz にする.
SIG_LEN = 2 ^ 16;
      % 信号長を 2^ 16 個にする.
dt = 0 : 1 / SF : 1 / SF * ( SIG_LEN - 1 );
      % 信号長を 2^ 16 個にする.

x = sin ( 2 * pi * 1000 .* dt );
      % 1000 Hz の信号を作成する.
imshow( stft( x, 64, 16, 1024 ));
      % 短時間フーリエ変換を行いスペクトログラムを可視化する.
colormap( 1-gray(256) );
      % 256 階調のグレースケールを反転させる.
print("./figs/stftrst_sin.pdf","-dpdf");
      % PDF 形式でグラフを出力する.

y = sin ( 2 * pi * 1000 .* dt ) + randn ( 1 , WL ) / 20;
      % 雑音を加えた正弦波を生成する.
imshow( stft( y, 64, 16, 1024 ) );
colormap( 1-gray(256) );
print( "./figs/stftrst_addnoise1.pdf","-dpdf" );

z = sin ( 2 * pi * 1000 .* dt ) + randn ( 1 , WL ) / 5;
imshow( stft( z, 64, 16, 1024 ) );
colormap( 1-gray(256) );
print( "./figs/stftrst_addnoise2.pdf","-dpdf" );
```

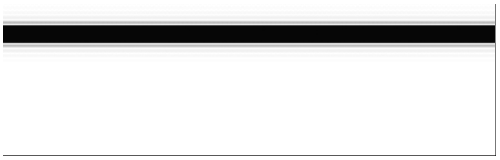

図 5: STFT 処理によるスペクトログラムの例 (正弦波).

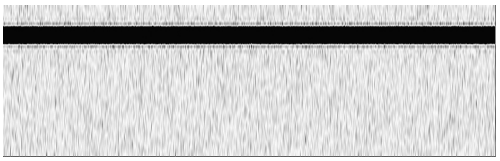

図 6: STFT 処理によるスペクトログラムの例 (雑音付加 1).

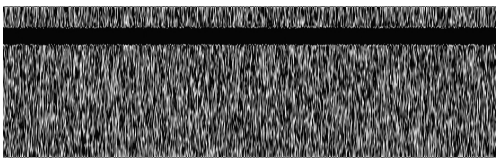

図 7: STFT 処理によるスペクトログラムの例 (雑音付加 2).

6.3 ディジタルフィルタ

ディジタルフィルタは，分子，分母が多項式で成っており，それにより伝達関数が決まる．ディジタルフィルタの詳細は専門書に譲る．ベクトル a と b をそれぞれ，分子，分母の係数ベクトルとし，インパルス応答を求める．図 8 は下記によって得られた結果である．

impulse_response.m

```
a = [ 1 0.5 ];
b = [ 1 1 ];
N=64;
      % インパルス応答の長さ.
imp = zeros(N,1);
      % 全要素が 0 の行列をつくる.
imp(1) = 1;
      % 最初の要素だけ 1 にして，インパルス応答とする.
h = filter ( b , a , imp );
      % インパルス応答の計算.
plot(h);
xlabel("Samples");
ylabel("h(t)");
title("Impulse response");
```

次に，周波数応答ベクトル H と 角周波数ベクトル W を求め，振幅特性と位相特性を求める．その結果を図 9 に示す．なお，周波数は正規化周波数となる．この結果から，振幅の周波数応答から低域通過型フィルタであることがわかる．

amplitude_and_phase_response.m

```
a = [ 1 0.5 ];
b = [ 1 1 ];
[H W] = freqz( b , a , 512 );
plot(W/(2*pi),20*log10(abs(H)));
      % 振幅特性．横軸は正規化周波数.
xlabel("Normalized freqeuncy (Hz)");
ylabel("Gain (dB)");
pause
      % 一時停止.

plot(W/(2*pi),180*angle(H)/pi);
      % 位相特性．横軸は正規化周波数.
xlabel("Normalized freqeuncy (Hz)");
ylabel("Phase (rad)");
```

7. 画像処理での実践

7.1 追加パッケージのインストール

ここでは例としてエッジ検出を行う．edge 関数は Octave をインストールした際には導入されない．そこで，

```
http://octave.sourceforge.net/actuarial/overview.html
```

で，edge 関数が含まれているパッケージを探すと，image package であることがわかる．

次に，image package を，

```
http://octave.sourceforge.net/packages.php
```

からダウンロードする．執筆時点では，image-1.0.12.tar.gz がダウンロードできる．

追加パッケージインストールは Octave 上で行うが，一般ユーザーが書き込みができるディレクトリではないので，インストール時のみ，

```
> sudo octave
```

で，Octave を起動し，

```
octave > pkg install image-1.0.12.tar.gz
```

でパッケージのインストールを行う．

7.2 エッジ検出

サンプルファイル Parrots.bmp のエッジ処理は次の手順でできる．まずは，データを

```
imgdata=imread("Parrots.bmp");
```

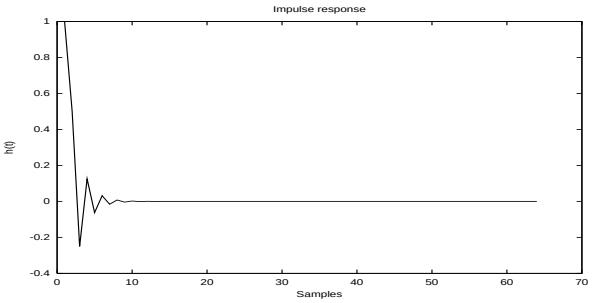

図 8: インパルス応答.

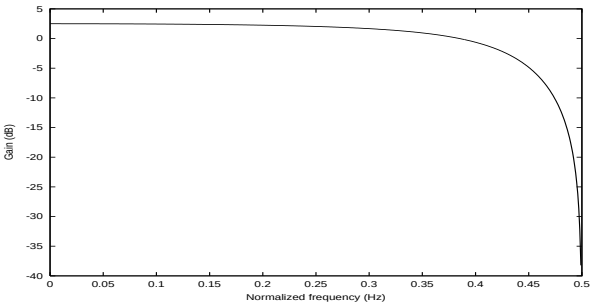

図 9: 周波数応答 (振幅).

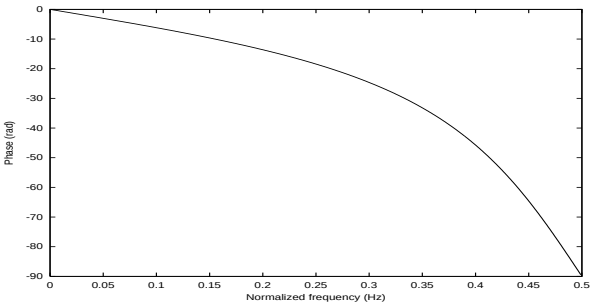

図 10: 周波数応答 (位相).

で読み込む．読み込んだ画像は，

```
imshow(imgdata);
```

で確認できる．読み込んだ画像は図 11 である．グレースケールへの変換は rgb2gray 関数で可能であり

```
monoimgdata=rgb2gray (imgdata);
```

で monoimgdata へ結果を代入する．エッジの検出は，

```
edge_rst=edge(monoimgdata,"Sobel");
```

で edge_rst へ代入される．図 12 および図 13 はエッジ検出を行った結果である．また，画像のヒストグラムは

```
imhist (double(monoimgdata))
```

で得ることができ，その結果を図 14 に示す．

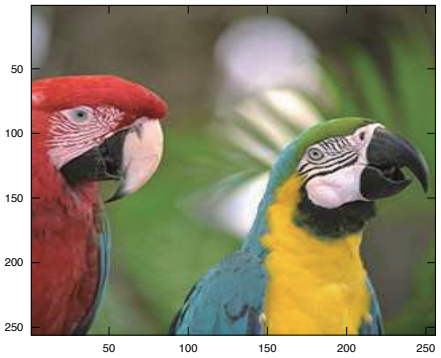

図 11: 原画像.

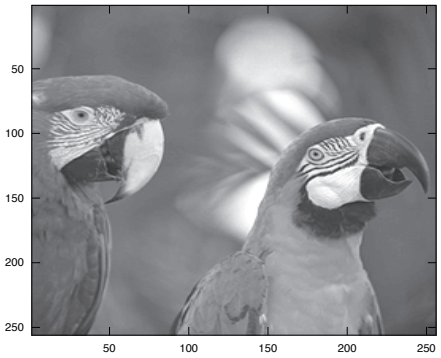

図 12: グレースケール画像.

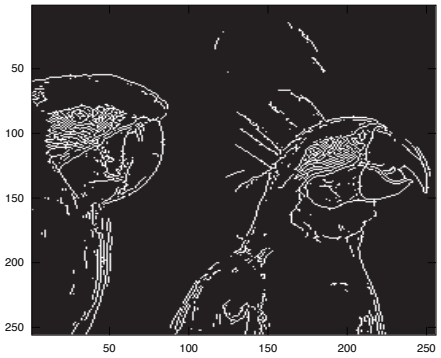

図 13: エッジ検出結果.

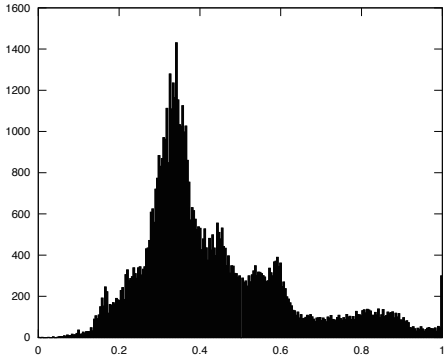

図 14: ヒストグラム.

8. メーリングリスト

octave-jp ML は，下記のサイトで登録できる．初習者のコミュニティとして利用いただけると幸いである．

http://hicc.cs.kumamoto-u.ac.jp/mailman/listinfo/octave-jp

9. まとめ

Octave Ver.3 では，同時に配布されていた関数群が別配布になったものが多い．次期バージョンも同様の方針で開発が行われている．また，可視化に関する関数群も大きく改善され，より一層 MATLAB との互換性が期待できるようにり，さらには次期バージョンでは gnuplot を用いずに独自のグラフィックエンジンを搭載する方針で開発が続けられている．

10. 謝辞

本稿の執筆にあたって，多くの Octave に関する web site を参考にさせていただいた．ここにお礼を申し上げます．また，Octave をはじめ無償で利用できるソフトウェアおよびソースコードを提供してくださっている方々へ深く感謝いたします．

参考文献

[Chisaki 05] 苣木禎史, 針木剛, 宇佐川毅，行列演算ソフトウェア GNU Octave による音響信号処理，人工知能学会誌，20 巻 6 号, pp.699 − 706, 2005.11

[Octave-sourceforge] http://octave.sourceforge.net/

[Octave] http://www.octave.org/

[cygwin] http://cygwin.com/

[SF] http://sourceforge.net/projects/octave

[Kawano] http://t16web.lanl.gov/Kawano/gnuplot/

[Yabuki 96] 矢吹 道郎 (監修)，大竹 敢：使いこなす GNUPLOT，株式会社テクノプレス (1996)

[Adachi 02] 足立 修一：MATLAB によるディジタル信号とシステム，東京電機大学出版局 (2002)