平成25年8月15日　作成

　　　　　　　　　　　　　　　　　　　　　　　平成28年4月20日　修正

1. **Matrab で利用できるデータ**
2. Matrab でしか使えないデータ……………….. \*.mat ファイル (binary data)
3. Matrab で使えるその他のデータで、Matrab 以外の環境でも使えるもの
4. \*.dat ファイル (Ascii data)
5. \*.txt ファイル
6. \*.bmp, \*.gif ファイルなどの画像データ
7. \*.wav ファイルなどの音楽データ
8. その他も？
9. **ワークスペース上で入力、あるいは外部ファイルから読み込んだデータの保存方法**
10. \*.mat ファイル（**binary data**, 単一・複数変数可能）

 save(filename)

 save(filename, variables)

 save(filename, ‘-struct’, structName, fieldNames)

 ……..

　save filename

save filename variables

……..

（註１）variables を省略すると、すべての複数変数を保存。ただし、matrab での各変数は、原則的に行列（ベクトルを含む）なので、１つ１つの変数は原則的には多次元データであることに注意。

（註２）save filename 等の ( ) を用いないコマンドは、**コマンド構文**　(command syntax) と呼ばれ、この場合は、入力文字列を単一引用符で囲む必要はない。コマンド構文の使用は、コマンドラインからの保存用に便利。

（註３）これに対して、save(filename) 等の ( ) 付のコマンドは、**関数構文** (function syntax) と呼ばれ、この場合は filename はリテラル文字列なので、関数構文を用いる時は単一引用符でこれを囲まないとエラーになる。すなわち、

　（誤）save(pqfile.mat)

　（正）save(‘pqfile.mat’)

　（正）save pqfile.mat

例１

　p=rand(1,10);

 q=ones(10);

 save pqfile.mat p q

　これにより、pqfile.mat なる mat ファイル に 1×10 ベクトル p と 10×10 行列 q が別々の binary変数として保存される。

　ただし、この場合、working directory がもしMATLAB であるとすれば、このファイルはその直下に保存される。そこで、もしこのファイルを MATLAB の下の data の下の multivar の下に保存したければ、

 save data/multivar/pqfile.mat p q

と指定しなくてはならない。

例２

　p=rand(1,10);

 q=ones(10);

 save pqfile.mat q

　つまり、ワーキングスペースに作成された２つの変数のうち１つだけを保存したければ、このように指定すればよい。

1. \*.dat ファイル（**Ascii data**, 単一変数のみ）

　save \*.dat filename /ascii

　ここで、dat ファイルは Ascii data なので、mkeditor などのエディタを用いて開いたり編集したりすることが可能。

1. \*.txt ファイル
2. save コマンドによる方法

save filename variables

例２

p=rand(1,10);

q=ones(10);

save pqfile.txt p q

これにより、pqfile.txt なる txt ファイル に 1×10 ベクトル p と 10×10 行列 q なる２つの行列（ベクトルを含む）が、その順に縦方向に並ぶ 11×10行列として、pqfile.txt の名前で単一ファイル（単一行列）として保存される。なお、その書式はmatrab が自動的にその標準形式で決定する。

　これに対して、もしつぎのように列数が異なる複数の変数を text ファイルとして保存すると、保存時にはエラーは表示されないが、load コマンドでコマンド入力しようとすると「列数が前の行とおなじでなければならない」旨のエラーが表示され、うまく保存されていないことがわかる。

うえの点については、以降の、load コマンドの項も参照、すなわちload コマンドで複数の変数を入力できるのは、mat ファイルのみ）。ちなみに、これをMkeditor などのテキストエディタで入力しようとしても入力できない。Mkeditor では、これはmatファイルとして保存されていることが表示される：

 誤例

p=rand(1,10);

q=ones(11);

save pqfile.txt p q

1. fprintf コマンドによる方法

　fprintf fid format A ・・・

このコマンドを用いると、ワークスペース上で入力したデータ A 等を特定の書式（format）で保存。

1. \*.bmp, \*.gif ファイル

**３**．**ワークスペース上への外部ファイルの読み込み方法**

1. **キーボード入力**

例３

　x=input(‘Enter x:’)

 disp(x)

1. **各種テキストファイルのインポートの方法**

　MATLAB では、対話モードでもプログラムによる方法ででも、幾つかのインポートツールにより、MATLABに各種のテキストファイル（.txt, .dat, .csv, .asc, .tab, 及び .dlm）をインポートできる。これらのテキストファイルは、四角形でなくてもよく、また行および列のヘッダーを持つこともできる。さらに、これらのファイルには、数値と非数値のテキストを組み合わせて使用でき、１つまたは複数の文字で区切ることもできる：

1. インポートツール

　これを用いるには、対話モードでMATLAB 画面上にある「インポートツール」タブを用いる。この方法は、数値以外のデータもサポートしている。

1. grades.txt

 John 　Ann Mark Rob

88.4 　91.5 　89.2 　77.3

83.2 　88.0 　67.8 　91.0

77.8 　76.3 　　92.5

92.1 　96.4 　81.2 　84.6

 手順

1. MATLAB画面上の「インポートツール」タブをクリックする。
2. これにより現れる「データのインポート」画面（「ファイルを開く」画面と同じ）で、インポートすべきデータのあるフォルダとファイルの拡張子を選択し、選択すべきファイル名をクリックし、「開く」ボタンをクリックする。
3. これにより、MATLAB用の特別な画面が現れ、その左端にインポートすべきファイルの内容が表示される。そこでは、既に上記データの中のヘッダー部と４行４列の数値部の区別がなされ、数値部には枠がはめられる。
4. この段階で、画面中央部やや左側にある縦に並んだタブ「列ベクトル、行列、セル配列」の中から適切なものを選択できる。デフォールトは列ベクトルである。

　例えば、その中から「行列」を選択し、画面右上の「選択のインポート」をクリックし、「データのインポート」をクリックすると、ワークスペースには４行４列の行列が入る。この時のMATLABでのファイル名は、gradesとなり、拡張子は無視される。一方、「列ベクトル」を選択すると、ヘッダー部のラベルを変数とする４つの列ベクトルがワークスペース上に生成され、それらはラベルのアルファベット順に表示される。共に、grades.txt ファイルの中の欠測値（空白）に対応する箇所にはMATLABの非数値の表示 NaN が入る。

1. load コマンド

　このコマンドでは任意の拡張子を持つファイルを読み込むが、数値以外のデータはサポートしない。

S = load(filename)

S = load(filename, variables)

S = load(filename, '-mat', variables)

S = load(filename, '-ascii')

load(filename, \_\_\_)

load filename \_\_\_

（註１）S = load(filename) は、MAT ファイルから構造体配列に変数を、ASCII ファイルから倍精度配列にデータを読み込む。

（註２）この方法では、任意の拡張子を持つファイルの読み込みが可能。

1. data/multivar/occup1955part.txt の 読み込み

ここで、working directory が MATLAB なるディレクトリーにあるとして、データがその下の data フォルダの下のさらに下の multivar なるディレクトリの下に occup1955part.txt

として保存されているとする。この場合にこのファイルをワークスペースに入力するには、つぎのようにすればよい：

 load data/multivar/occup1955part.txt

また、このファイルをコマンドウインドウに表示するには、拡張子をつけたままではエラーになるので、拡張子を除き、つぎのようにすればよい：

　occup1955part

（註３）ただし、variables として（複数の）変数名を指定できるのは、.mat ファイルの場合のみ。それ以外のファイルにこの variables を指定してもエラーとなり、読み込めない。

（註４）１人分のデータが複数行に亘っており、かつ各行で書式が異なるデータ（例えば p79103uf.txt ）は、行により書式が異なり、matrab では入力エラーとなり入力できない。たぶん、そのようなデータは一旦 sas の put コマンドで個々のデータ間を空白あるいはコンマで区切ったものに変換し、かつ各人のデータを複数行ではなく一行データとして出力させ、そのうえで matrab で読み込むようにすればよいだろう。

（註５）S = load(filename, '-mat', variables) は、ファイルの拡張子に関係なく MAT ファイルとしてファイルを取り扱うように load を強制する。variables の指定はオプションであるが、.txt ファイルでこのコマンドを用いて変数名を付けると、エラーとなる。

（註６）S = load(filename, '-ascii') は、ファイルの拡張子に関係なく ASCII ファイルとしてファイルを取り扱うように load を強制する。

1. load cement

　　　　　　　ここで、データcement には拡張子 .txt がついており、これに

　　　　　　よりワークスペース上では、下記の（註）にあるデータの各列に

自動的にx及びyなる変数名がつき、共に２１行１列の列ベク

トルとして保存される。

　　　　　　　ここで、cement.txt データは数値の区切りが「空白」になって

いる以外は、空白の数は無視されることがわかる。

　また、このデータは .txt ファイルなので、既述の３．の（註３）

にあるように、このコマンドの右側に変数名リストを指定すると

エラーとなり読み込めない。

（**疑問**）この種のデータの各列に対して、ユーザ側から任意の変　　数名をつけて保存することはできないか？

（註）cement.txt　(Martinez & Martinez, 2008)

1 13.0

1 13.3

1 11.8

2 21.9

2 24.5

2 24.7

3 29.8

3 28.0

3 24.1

3 24.2

3 26.2

7 32.4

7 30.4

7 34.5

7 33.1

7 35.7

28 41.8

28 42.6

28 40.3

28 35.7

28 37.3

1. dlmread コマンド

　Ascii デリミタ付ファイルを読み込むコマンド：

M= dlmread(filename)

M = dlmread(filename, delimiter)

M = dlmread(filename, delimiter, R, C)

M = dlmread(filename, delimiter, range)

1. importdata コマンド

　外部のテキストファイル、またはシステムのクリップボードからテキストファイルを入力するためのコマンド：

A = importdata(filename)

　　データを配列A に読み込む時に使用。

A = importdata('-pastespecial')

　　データをクリップボードから読み込む時に使用。

A = importdata(\_\_\_,delimiterIn)

　　データの中のdelimiterIn で指定されるデリミタ（例えば、空白、コンマ、タブ）を、Ascii ファイルの列区切りまたはクリップボードのデータとして解釈する時に使用。

A = importdata(\_\_\_,delimiterIn,headerlinesIn)

　　ヘッダーライン＋１の行から始まる数値データを読み取ること　により、データをAscii ファイルまたはクリップボードから読み込む時に使用。

[A,delimiterOut,headerlinesOut] = importdata(\_\_\_)

1. textscan コマンド

C = textscan(fileID,formatSpec)

C = textscan(fileID,formatSpec,N)

C = textscan(fileID,formatSpec,N,Name,Value)

C = textscan(str,formatSpec,N,Name,Value)

[C,position] = textscan(\_\_\_)

（註１）C= textscan(fileID,formatSpec) は開いたテキストファイル　としてのデータにformatSpecなる書式を指定して、セル配列 C に読み取る。テキストファイルはファイル識別子 fileID で示される。fopen を使用すると、このファイルを開いて fileID 値を取得できる。ファイルから読み取った後、fclose(fileID) を呼び出してファイルを閉じる。

（註２）textscan は、ファイル内のデータを変換指定子の文字列 formatSpec と一致させようとする。

（註３）C= textscan(fileID,formatSpec,N) は、formatSpec を N 回使用して、データを読み取る。ここで N は正の整数である。N 回反復した後でファイルから追加のデータを読み取るには、元の fileID を使用して textscan を再度呼び出す。同じファイル識別子 (fileID) で関数 textscan を呼び出してファイルのテキスト スキャンを再開すると、textscan は、最後の読み取りを終了したポイントから自動的に読み取りを再開する。

（註４）C= textscan(fileID,formatSpec,N,Name,Value) は、1 つまたは複数の Name,Value のペアの引数によって指定された追加オプションを使用する。この構文グループ内の他の構文のいずれかに Name,Value の引数ペアを追加することができる。

（註５）C= textscan(str,formatSpec,N,Name,Value) は文字列 str からデータを読み取る。前の構文と同様に、この構文で formatSpec、さらにオプションで N および Name,Value 引数ペアを使用する必要がある。ただし、文字列の場合は、textscan を繰り返し呼び出すと、毎回始めからスキャンが再開される。前回の位置からスキャンを再開するには、position 出力を要求すること。

（註６）[C,position] = textscan(\_\_\_) は、上記の構文のいずれかの入力引数を使用して、ファイルまたは文字列の位置を 2 番目の出力引数としてスキャンの最後に返す。ファイルの場合、これは textscan を呼び出した後で ftell(fileID) から返される値である。文字列の場合、position は、関数 textscan により読み込まれた文字数を示す。

　　例３．

　　　fileID = fopen('scan1.dat');

C = textscan(fileID, '%s %s %f32 %d8 %u %f %f %s %f');

fclose(fileID);

celldisp(C)

（註）scan1.dat

　　09/12/2005 Level1 12.34 45 1.23e10 inf Nan Yes 5.1+3i

10/12/2005 Level2 23.54 60 9e19 -inf 0.001 No 2.2-.5i

11/12/2005 Level3 34.90 12 2e5 10 100 No 3.1+.1i

 　うえのコマンドでは、C には３行９列の行列としてではなく、９

　　　　　列のセルからなる内容となり、各セルは３行１列のベクトルとして

　　　　　保存される。そこで、例えば

　　　　　　　C{1}{1} =

　　　　　　　　 09/12/2005

　　　　　　　　は、第１列目のベクトルの第１行要素を表す。

　　　　　　　C{2}{3} =

Level3

 は、第２列目のベクトルの第３行目要素を表す。

　　　　　　　C{3} =

 　　　　　　 12.3400

 　　　　　　 23.5400

 　　　　　　 34.9000

　　　　　　　（**疑問**）ただし、うえのMatlab の celldisp(C) による上のような出力のうち、

　　　　　　　C{5} =

 　　　　　　 4294967295

 　　　　　　 4294967295

 200000

　　　　　と出力されるのはなぜか。

　　　　（註７）formatSpec （書式指定）データフィールドの形式

1. 文字列

データ フィールドの形式は、1 つまたは複数の変換指定子の　文字列として指定する。関数 textscan はファイルまたは文字列を読み取るときに、データを formatSpec 文字列に一致させようとする。関数 textscan がデータ フィールドの一致に失敗した場合は、読み取りを中断し、中断する前に読み取ったすべてのフィールドを返す。ここで、変換指定子の数は、出力配列 C 内のセル数を決めます。

1. 数値フィールド

つぎの表は、数値入力用に使用可能な変換指定子をまとめたものである：

数値入力タイプ　変換指示子　出力クラス

---------------------------------------------------------

整数、符号付き　 %d　 int32

　　　　　　　　　 %d8 int8

 %d16 int16

 %d32 int32

 %d64 int64

整数、符号なし %u uint32

 %u8 uint8

 %u16 uint16

 %u32 int32

 %u64 uint64

浮動小数点数 %f double

 %f32 single

 %f64 double

 %n double

1. 文字フィールド

つぎの表は、文字入力用に使用可能な変換指定子をまとめたものである：

文字列　　 変換指示子　　 詳細

-----------------------------------------------------------------------------------

文字　　　　 %s　　　　　文字列

　　　　　　　　　　 %q　　　　　文字列。二重引用符は、一まとめにするテキストを示す。

　　　　　　　　　　 %c　　　　　区切り記号を含む 1 つの文字

　　　　-----------------------------------------------------------------------------------

 パターンマッ %[...] 一致しない最初の文字に到達す

チング文字列 るまで、かっこ内の文字のみを読み取る。セットに ] を含めるには、それを最初に指定する。

 　　　　例: %[mus] は、'summer ' を 'summ' として読み込む。

%[^...].　　　一致する最初の文字に到達するまで、かっこ内の文字を除外する。] から除外するには、それを最初に指定する。

例: %[^xrg] は、'summer ' 　　を 'summe' として読み込む。

1. オプションの演算子

formatSpec 内の変換指定子は、オプションの演算子を含めることができる。これらの演算子は次の順序で表示される (明確にするためにスペースも含む)。

 例

%\*n5.2f

　　　　　　　　　オプションの演算子としては、次のものが挙げられる：

1. 無視するフィールドと文字

textscan は、特定のフィールドまたはフィールドの一部を無視するように指示していない限り、ファイルのすべての文字を順番に読み取る。

1. フィールドの一部のスキップか読み取り

フィールドの一部をスキップするか読み取るには、次の演

算子を使用する。

1. **グラフィック（画像）ファイルの入力方法**
2. imread コマンド

A=imread (filename, fmt)

[X, map]=imread (・・・)

[・・・]=imread (filename)

[・・・]=imread (URL, ・・・)

[・・・]=imread (・・・, Param1, Val1, Param2, Val2,・・・)

　　　（註１）A=imread (filename, fmt) により、文字列 filename で指定されたファイルから**グレースケールまたはカラーイメージ**を読み取る。ファイルが現在のフォルダまたはMATLABパス上のフォルダにない場合は、絶対パス名を指定する必要がある。

　　　　　　　　また、テキスト文字列 fmtは、その標準ファイル拡張子により、ファイルの形式を指定する。MATLABがサポートしているfmtには、つぎのような各種がある：

　　　　　　　　**bmp** (windows bitmap), **jpeg** (joint photographic experts group), **png** (portable network graph), **cur** (cursor file), **jpeg**

**2000** (joint photographic experts group 2000), **ppm** (portable pixmap), **gif** (graphics interchange format), **pbm** (portable bit

map), **ras** (sun raster), **hdf4** (hierarchical data format), **pcx** (

windows paintbrush), **tiff** (tagged image file format), **ico** (icon file), **xwd** (x window dump), **pgm** (portable graymap)

　　　　　　　　戻り値Aは、イメージデータを含む配列。ファイルにグレースケールイメージが含まれる場合、AはM行N列の配列になる。ファイルにトウルーカラーイメージが含まれる場合は、AはM×N×３の配列になる。また、CMYK色空間を使用するカラーイメージを含んだTIFFファイルの場合、AはM×N×４配列になる。詳細はimreadコマンドを参照のこと。

　　　　　　　　**ここで、filename もfmt もテキストなので両端に引用符を付けないとエラーとなる**。

　　　（註２）[X, map]=imread (・・・) は、filenameのインデックス付イメージをX、その関連するカラーマップをmapに読み取る。イメージカラーマップ値は、自動的に範囲 [0,1] に再スケーリングされる。

　　　（註３）[・・・]=imread (filename) は、その内容からファイルの形式を推定しようとする。

　　　（註４）[・・・]=imread (URL, ・・・) は、インターネット URL から、イメージを読み取る。URL には、プロトコルタイプ（例えば、<http://）を含めなければいけない>。

　　　（註５）[・・・]=imread (・・・, Param1, Val1, Param2, Val2,・・・) は、特定形式のさまざまな特性を制御するパラメータを指定する。詳細は、imread コマンドを参照のこと。

　　　例　C=imread(‘kikyu.bmp’, ‘bmp’);

1. **音声ファイルの入力方法**
2. wavread コマンド

　　　y = wavread(filename)

[y, Fs] = wavread(filename)

[y, Fs, nbits] = wavread(filename)

[y, Fs, nbits, opts] = wavread(filename)

[\_\_\_] = wavread(filename, N)

[\_\_\_] = wavread(filename, [N1 N2])

[\_\_\_] = wavread(\_\_\_, fmt)

siz = wavread(filename,'size')

　　　　　（註１）y = wavread(filename) は、文字列 filename で指定される **WAVE ファイル**を読み込み、サンプルされたデータを y で返す。filename が拡張子を含まない場合、wavread は .wav を付加する。

　　　　　（註２）[y, Fs] = wavread(filename) は、ファイル内のデータをエンコードするために使用されるサンプル レート (Fs) を Hz 単位で返す。

　　　　　（註３）[y, Fs, nbits] = wavread(filename) は、1 サンプルあたりのビット数 (nbits) を返す。

　　　　　（註４）[y, Fs, nbits, opts] = wavread(filename) は、WAV ファイルに格納されている追加情報の構造体 opts を返す。この構造体の内容は、ファイルごとに異なる。一般に使われる構造体のフィールド名は、opts.fmt (オーディオ形式情報) と opts.info (タイトル、著者などを記述するテキスト) である。

　　　　　（註５）[\_\_\_] = wavread(filename, N) は、ファイル内の各チャンネルから最初の N 個のサンプルのみを返す。

　　　　　（註６）[\_\_\_] = wavread(filename, [N1 N2]) は、ファイル内の各チャンネルから N1 から N2 のサンプルのみを返す。

　　　　　（註７）[\_\_\_] = wavread(\_\_\_, fmt) は、ファイルから読み取られるサンプルを表すために使用されるデータ形式 y を指定する。fmt は、つぎのいずれかの値か、部分一致 (大文字と小文字は区別されない) であってもよい：

1. 'double'倍精度の正規化されたサンプル (既定の設定)
2. 'native'ファイルにあるネイティブ データ型のサンプル

　　　　　（註８）siz = wavread(filename,'size') は、実際のオーディオ データではなく filename 内に含まれているオーディオ データのサイズを、ベクトル siz = [samples channels] として返す。

　　　　　（註９）出力スケーリング

　　　　　　　　y の値の範囲は、指定されたデータ形式 fmt に依存する。WAV ファイルで見つけられた一般的なビット幅に基づく出力スケールの例は、'double' と 'native' 書式の両方に対して与えられる：

1. ‘native’ formatsの場合

|  |  |  |
| --- | --- | --- |
| bit 数 | MATLAB data type | Data Range |
|  8 | uint8 (unsigned integer |  0≦y≦255 |
|  16 | int16 (signed integer) | $$-32768\leqq y\leqq 32767$$ |
|  24 | int32 (signed integer) | $$-2^{23}\leqq y\leqq 2^{23}-1$$ |
|  32 | single (floating point) | $$-1.0\leqq y<1.0$$ |

1. ‘double formats の場合

|  |  |  |
| --- | --- | --- |
| bit 数 | MATLAB data type | Data Range |
|  N<32 |  double |  $-1.0\leqq y<1.0$ |
|  N=32 |  double |  $-1.0\leqq y\leqq 1.0$Note: Values in y might exceed -1.0 or +1.0 for the case of N=32 bit data samples stored in the WAV file. |

1. 高谷 (2004, p.183). FT-spcgram1004.m

fname=’crash.wav)’;

[y, fs, nbits]=wavread(fname);

sound(y, fs);

1. auread コマンド

y = auread(aufile)

[y,Fs] = auread(aufile)

[y,Fs,nbits] = auread(aufile)

[\_\_\_] = auread(aufile,N)

[\_\_\_] = auread(aufile,[N1 N2])

siz = auread(aufile,'size')

　　　　　　（註１）y = auread(aufile) は、文字列 aufile で指定されるサウンド ファイルを読み取り、サンプリングされたデータを y に返す。既定の拡張子は .au である。振幅値は、[-1,+1] の範囲。auread は、以下の形式で記述されたマルチチャンネル データをサポートする。