

MATLABではじめる プログラミング教室

奥野 貴俊・中島 弘史

【共著】

コロナ社

まえがき

私が MATLAB を最初に使ったのは、1995 年の春、大学の 4 年生になり卒論研究を始めたときにさかのぼる。恥ずかしながら、学部 3 年生までの授業で習得しておくべきプログラミング言語 (Pascal, C) などは苦手でなんとか単位をもらったことを記憶している。とにかく単位を取ることに必死で、なんのためにプログラミングをやっているのかも定かではない、そんな時間を過ごしていた。1995 年と言えば、この本を手にとった読者はご存知ないかもしれないが、爆発的にパソコンが普及し始めるキッカケとなった Windows95 が発売され、それと同時にインターネットが各家庭に設置され始めた年として、インターネット元年として記憶されている。当時の大学のコンピュータ環境と言えば、インターネットへの接続は当たり前ではあったが、OS としては UNIX が基本であった。MATLAB も配属先の研究室では UNIX 環境にインストールされて利用されていたため、MATLAB を使うために UNIX も最低限のレベルで同時に扱えるようになる必要があった。担当教授が世界の研究者と交流を持ちながら、世界の研究現場の現状をよく把握されていたので、MATLAB には当たり前のようにな研究室にて触れることができたが、当時は日本ではまだまだ知られていないプログラミング言語であったように思う。

修士課程のころ、海外の国際会議にて何度か発表する機会に恵まれ、海外の大学を見て回ったとき、必ずやっていたことは、その大学の書店を巡ることであった。自分の専門分野に近い書棚にいくと、必ず置いてあったのが MATLAB の関連書籍だった。自分にとっては専門分野の書籍よりも MATLAB の関連書籍はずっと身近で「この使い方は知っている、これは知らない」と頷きながら立ち読みをしたことをよく覚えている。あれから 20 年近く経ち、なんの因果かわからないが、現在母校の大学で新規に開講された MATLAB を学ぶ「計測制御プログラミング」という授業の非常勤講師を担当している。これは大学が 2015 年に大規模な MATLAB のライセンス契約を結んだことに端を発しているが、いまでも MATLAB が世界中で幅広く利用されていることが単純に嬉しく、またその実益が (教育的にも研究的にも) かなり大きいということを改めて感じている。

MATLAB の最大の長所は、私が学生だったときから変わらないが、専門分野の詳しいことがわからなくても、なんとなく使ってみることができるという点にある。例えば、FFT という音響信号処理や画像処理の分野では当たり前を使う数値計算がたったの 1 行で「使えてしまう」のである。当時から例えば FFT の中身がわからないまま利用できてしまうことが問

題視されていた（いまもそのキライはある）が、私はこの点はむしろ非常にポジティブに捉えている。なぜなら、コンピュータを使うことに対してハードルが高いと、それ以上先にはなかなか踏み出せないからである。つまり「自分で考える」という最も大切で楽しい思考の時間にたどり着く前に挫折しがちであるからである。理工系では課題や問題に対して「やってみる」という、実践しようとする基本的態度がいつの時代でも大切に重要であると私は信じている。

賛否両論はあるかもしれないが、私はこのとにかくどうなるかわからないが「やってみる」という態度に価値を置き、後の「自分で考える」ことの大切さと楽しさが多くの方に理解されるように、その価値をこの本の中心的な柱に据えようと思う。「とにかくやってみる」という境地から、「自分で考える」境地に至るまで、MATLAB は、頭の中のアイデアを具現化する際に非常に有益なツールとして皆さんをサポートしてくれることと思う。その最初のステップとしてこの本を選んで頂けたら、とても嬉しく思う。

2017年8月

著者を代表して 奥野貴俊

本書の特長

- ・理工系大学に入って MATLAB を初めて使う方が慣れ親しむためのテキスト
- ・大学の授業（半期）でも使用しやすい 13 テーマを収録
- ・体験的な自学自習を可能とした解説と演習問題
- ・音響信号処理の専門家が音を材料に楽しめるテーマを多数収録
- ・高校数学と大学数学との橋渡しの教材としても有用
- ・プログラミングに興味がある高校生でも自習可能

目 次

1. まずは試してみる

— 解の公式をプログラムしてみよう —

1.1	MATLAB を試してみよう	1
1.1.1	早速 MATLAB を立ち上げてみる	1
1.1.2	MATLAB の超基本コマンドのまとめ	5
1.2	MATLAB エディタを試してみよう	5
1.2.1	エディタに解の公式を書いてみる	5
1.2.2	さまざまな値の表現方法	7
1.2.3	ベクトル作成で重要な役割を持つコロン	9
1.2.4	MATLAB の MAT は MATRIX の MAT	12
	演習問題	14

2. ループと条件分岐ってなに？

— 電卓を越えたプログラム —

2.1	電卓の域を越える MATLAB の使い方をマスターしよう	15
2.1.1	for (ループ) について知ろう	15
2.1.2	if (条件分岐) について知ろう	17
2.1.3	else を if と一緒に使う	17
2.2	まだあるループと条件分岐	21
2.2.1	もう一つのループ while	21
2.2.2	もう一つの条件分岐 switch	23
	演習問題	24

3. サイン・コサインも思いのまま

— 自分だけのコマンド作成 —

3.1	まずは MATLAB に備わっているコマンドを試してみよう	25
3.1.1	その前に 1, 2 章を簡単に復習する	25

3.1.2 よく使うコマンドを使ってみる	25
3.1.3 数値計算でよく使うコマンドを使ってみる	27
3.2 オリジナルのコマンドをつくってみよう	30
演習問題	32

4. レポートや論文でも使えるグラフ表示 — plot のワザを習得! —

4.1 これまでの plot コマンドの使い方とより正確な表示方法	33
4.1.1 plot(x,y) に慣れよう	33
4.1.2 グラフの表示範囲を意識しよう	34
4.2 グラフをもっとデコレーションしよう	36
4.2.1 線の種類とマーカー	36
4.2.2 横軸・縦軸のラベル, グラフのタイトル	38
4.2.3 片対数グラフと両対数グラフ	39
4.3 フィギュアウィンドウの扱い方	40
4.3.1 フィギュアウィンドウの作成と消去	40
4.3.2 一つのフィギュアウィンドウにいくつものグラフを描く	41
4.4 ほかにもあるグラフプロット	43
4.4.1 離散信号を示すなら stem プロット	43
4.4.2 ヒストグラムも簡単に表示できる histogram コマンド	44
演習問題	45

5. 2D から 3D へ — おしゃれな 3D 曲面も描ける —

5.1 3D は 2D の延長?	46
5.1.1 まずは 2D を復習しながら 3D にしてみる	46
5.1.2 3D グラフ描画の plot3 を使ってみる	48
5.2 線 から 面 へ	49
5.2.1 曲面表示の mesh と surf	50
5.2.2 グラフの色合いの変更 colormap	53
5.3 2D のほうが見やすい場合もある	54
演習問題	55

6. MATLAB へ入れたり出したり — 地味だけど大切なデータのやり取り —

6.1	MATLAB で最も手っ取り早いデータの保存と読み込み方法	56
6.1.1	保存・読み込みの最強コマンド <code>save</code> と <code>load</code>	56
6.1.2	<code>save</code> と <code>load</code> でテキストファイルも扱える	58
6.2	多少煩わしいけれど、より正確なファイルの入出力方法	60
6.2.1	テキストファイルへの保存 <code>fprintf</code>	61
6.2.2	テキストファイルの読み込み <code>fscanf</code>	63
	演習問題	64

7. オーディオ&画像データもお手のもの？ — .wav や.jpg は特別扱い？ —

7.1	やはり煩わしいけれど、バイナリファイルの保存と読み込み	66
7.1.1	バイナリファイルへの保存 <code>fwrite</code>	66
7.1.2	バイナリファイルの読み込み <code>fread</code>	68
7.2	特別扱いその1：オーディオデータの保存と読み込み	68
7.2.1	オーディオデータの保存 <code>audiowrite</code>	69
7.2.2	オーディオデータの読み込み <code>audioread</code>	71
7.3	特別扱いその2：画像データの保存と読み込み	71
7.3.1	画像データの保存 <code>imwrite</code>	71
7.3.2	MATLAB で扱う画像データの種類	73
7.3.3	画像ファイルの読み込み <code>imread</code>	74
	演習問題	75

8. 理工系なら絶対に知っておきたいこと — 最小二乗法を考える！ —

8.1	最小二乗法ってなにをするためのモノなのか	76
8.1.1	まずはゴールの設定	76
8.1.2	2点を通る直線	77
8.1.3	3点を通る直線？	78
8.2	最小二乗法をできるだけシンプルに解説	79

8.2.1	最小二乗法の本質について	79
8.2.2	違う角度から最小二乗法をしてみる	82
8.2.3	最小二乗解を求め、近似直線を引いてみる	84
	演習問題	86

9. サイン波を音として聴く

— 周波数って？ シンセサイザの基本の音 —

9.1	A4 のサイン波をつくる	87
9.1.1	時間に依存したサイン波をつくる準備	87
9.1.2	もう一つの準備, サンプリング周波数と時間軸	90
9.1.3	いよいよサイン波を MATLAB でつくる	91
9.2	サイン波でドレミをつくる	93
9.2.1	ドレミの周波数とは？	93
9.2.2	いよいよドレミをつくる	94
9.3	シンセサイザの基本の音をつくってみる	95
9.3.1	矩形波をつくってみる	95
9.3.2	三角波をつくってみる	97
9.3.3	ステレオの音のつくり方	100
	演習問題	101

10. 時間と周波数の関係

— よく知らなくても使える FFT —

10.1	自分の声を MATLAB で録音してみよう	102
10.1.1	イヤホンをマイク代わりに？	102
10.1.2	オーディオ録音のコマンド <code>audiorecorder</code>	102
10.2	サイン波を分析してみる	105
10.2.1	振幅周波数スペクトルって？	105
10.2.2	時間データを周波数データにするには高速フーリエ変換 <code>fft</code>	107
	演習問題	112

11. 超簡単なノイズ低減&リバーブ!

— じつは音響信号処理のキホン —

11.1 信号を処理する線形システム	113
11.1.1 世の中に数多くある線形システム	113
11.1.2 線形とは?	114
11.2 移動平均による雑音抑制処理	115
11.2.1 まずは使用する雑音付加音声信号の準備	115
11.2.2 雑音を減らす方法を考えてみましょう	115
11.3 音に響きを付加する畳込み演算	119
11.3.1 音の響きを付加するための問題設定	119
11.3.2 畳込み演算を一步ずつ	120
演習問題	121

12. GUIってなに?

— 日常にあふれているアプリの中身を知る —

12.1 とりあえず GUI 開発環境を動かしてみる	122
12.1.1 GUI 開発は <code>guide</code> で始まる	122
12.1.2 GUI プログラミングは <code>Callback</code> 関数へ	126
12.2 音の分析アプリを GUI 開発環境を使ってつくる	128
12.2.1 <code>guide</code> を入力する前にしておくこと	129
12.2.2 まずはオーディオファイルを読み込んで上半分に <code>plot</code> してみる	129
演習問題	134

13. アプリをつくる側になってみる

— 結局 MATLAB って簡単だったね —

13.1 <code>guide</code> コマンドで GUI 開発環境を再起動	135
13.1.1 12 章のおさらいと本章でやることの確認	135
13.1.2 どんなオーディオデータでも対応できることが重要	136
13.2 複数の <code>Callback</code> 関数間を行ったり来たり	138
13.2.1 ポップアップメニューを使うための準備	138
13.2.2 ポップアップメニューには約束事がある	140

13.2.3 handles の再登場	141
13.2.4 アプリケーションの完成度を高めるエラー回避	143
演習問題	145
引用・参考文献	146
演習問題解答	147
索引	156

9

サイン関数を音として聴く — 周波数って？ シンセサイザの基本の音 —

本章では、これまで描いてきたサイン関数を、高さを持った音としてのサイン波とするために基礎から始めます。特に周波数や角周波数のような物理量、またデジタルで音を扱う際に理解が必要なサンプリング周波数のような用語も簡単に解説します。その後、MATLABで440 Hzのサイン波を作成します。サイン波の周波数を変化させれば音の高さが変わるので、ドレミの周波数を計算で求め、ドレミの音をつくっていきます。さらにシンセサイザの基本の音となる矩形波・三角波も作成し、最後に、MATLABでステレオの音の扱い方について解説します。音に興味がある方は本章を利用して、さまざまなアイデアを発展させてほしいと思います。

ゴール 周波数の関数としてのサイン波を作成して、ドレミの音階が作れるようになる！

9.1 A4のサイン波をつくる

9.1.1 時間に依存したサイン波をつくる準備

ピアノやキーボードの鍵盤を見たことがあると思いますが、「A4」と言われてすぐになんのことかわかりますか？ Aというのはドレミで言うところの「ラ」、4は図9.1に示すように鍵盤の低い音から数えて4番目を意味します（国際式MIDI規格）。つまり「A4」とは、鍵盤の低いほうから数えて「4番目のラ」を指します。この「A4」はオーケストラのコンサートが始まる前に各楽器間の調律を合わせるために使われる音の高さで、周波数で言うとおおよそ「440 Hz」です。ちなみにこの音の高さは昔から時報でも使われている音の高さで、さまざまな場面で基準となる音の高さです。

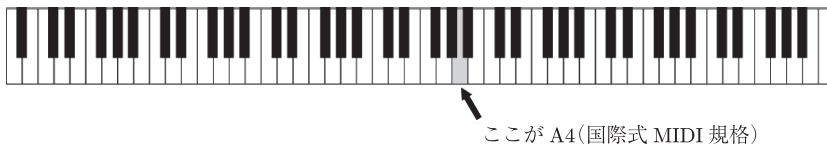


図9.1 鍵盤のA4

では、まずこの高さの音をMATLABで作成してみましょう。ここでは高校の数学で学ぶサイン関数を用いて実現してみます。その理由は、サイン関数は音響の世界では「純音」と

呼ばれ、単一の周波数を持つ波なので、440 Hz を再現するには1番単純だからです。しかし、これまでと同じようにはいきません。それはなぜでしょう。 $\sin(\theta)$ の θ は $0 \sim 2\pi$ まで変化するとは言え、ただの角度だからです。サイン関数を音として感じるためには、普段から聴いている音楽と同様に時間という概念が必要で、 θ のままではその時間を含んだ形になっていません。そもそも、周波数とは“1秒あたり”の振動数、つまり1秒という「時間」に対して振動する回数なのです。

ここで θ という角度の変数を、 ω という「角周波数」と「時間」 t を使って、ある意味無理やり時間を使って表現していきます。つまり

$$\theta = \omega t \quad (9.1)$$

とします。角周波数 ω は、1秒あたりに変化する角度を意味し、単位は [rad/s] です。理解を深めるために、**図9.2**を見てください。図(a)に描かれている円は「単位円」と呼ばれる原点を中心に半径1で描かれる円です。このグラフでは、注目する座標が角度ゼロの位置から反時計回りに円の上を周回すると考え、現状は θ となっています。また図(b)のグラフは図(a)の単位円に対応した $\sin(\theta)$ を表しています。このグラフの横軸は θ に対応し、縦軸は図(a)の単位円上を周回している点の高さ、つまり半径が1なので $\sin(\theta)$ に対応しています。一旦話を戻しますが、角周波数 ω の単位は [rad/s] ですと説明しました。角周波数はこの単位からもわかるように、1秒あたりに変化する角度、つまり物理量としては「速度」のようなものです。 ω を速度とみなすと、 t は時間であり単位は [s] ですので、 ωt は「速度 × 時間」と考えることができ、 θ は距離とみなすことができます。繰り返しになりますが、 θ は角度ではあるのですが、 ωt とすることで、時間に依存した角度とすることができます。

これで $\sin(\theta)$ という角度に依存したものから、時間に依存した $\sin(\omega t)$ というサイン波をつくる準備ができました。しかし、話の冒頭で説明した A4 の周波数にあたる 440 Hz を $\sin(\omega t)$

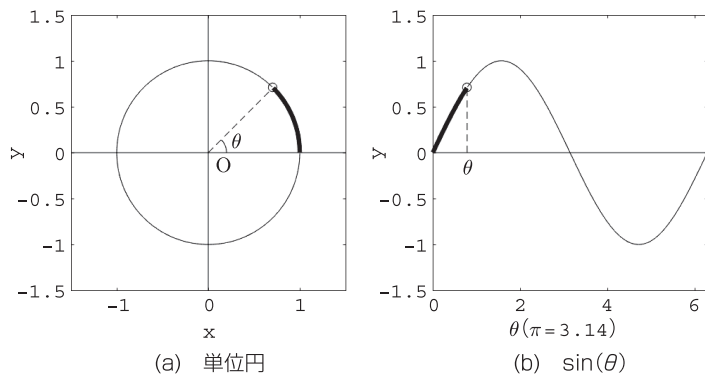


図9.2 単位円と $\sin(\theta)$ の関係

に代入することはできません。先ほど「周波数とは“1秒当たり”の振動数である」と言いました。これは言い換えると図(a)の単位円周上を「1秒間に何回まわったか」という「回転数」を意味します。これを紐解くために、つぎの例を考えてみましょう。

速度に対応する角周波数 ω が $\pi/2$ のとき、1秒間に変化する角度は

$$\theta = \omega t = \frac{\pi}{2} \cdot 1 = \frac{\pi}{2} [\text{rad}] \tag{9.2}$$

となり、**図9.3**(a)のグラフとなります。このときの角度 θ は $\pi/2$ であり、単位円周上を1/4周していることがわかります。

同様に、角周波数 ω が π のとき、1秒間に変化する角度は

$$\theta = \omega t = \pi \cdot 1 = \pi [\text{rad}] \tag{9.3}$$

となり、**図**(b)のグラフとなります。このときの角度 θ は π であり、単位円周上を1/2周していることがわかります。

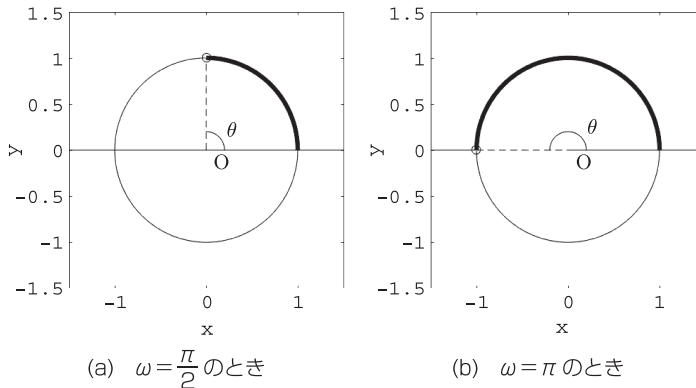


図9.3 時間 $t=1$ で角周波数 $\omega = \frac{\pi}{2}, \pi$ としたときの角度 θ

つまり、周波数（回転数）を f とすると

$$\omega = \frac{\pi}{2} \text{ のとき, } f = \frac{1}{4}$$

$$\omega = \pi \text{ のとき, } f = \frac{1}{2}$$

という関係が得られます。この二つの関係性を眺めるとわかりますが、 f に 2π を掛けると ω と等しくなります。一般的に書くと

$$\omega = 2\pi f [\text{rad/s}] \tag{9.4}$$

とすることができるでしょう。ちなみに、 f は1秒当たりの回転数であるので単位は $1/\text{s}$ となり、これを Hz（ヘルツ）と呼びます。つまり、440 Hz とは、単位円周上を1秒間に440

回、回転することを意味しています。さらに言えば、 f が大きくなれば ω が大きくなり、回転速度（角周波数）が速くなることを意味します。

以上を踏まえると、時間に依存したサイン波は、元の角度のみに依存したサイン関数から、以下のように変形できることがわかります。

$$\sin(\theta) \rightarrow \sin(\omega t) \rightarrow \sin(2\pi f t) \quad (9.5)$$

9.1.2 もう一つの準備, サンプルング周波数と時間軸

式(9.5)で一応の準備ができたように思えますが、もう一つだけ理解が必要となることがあります。それは、音のデータだけでなく、画像データでもなんでも、コンピュータに保存されているデータは数値であるということです。数値であるということはなんらかの方法で連続的なデータを「数字に置き換え、離散化している」ということです。例えば、音の情報で言えば、耳の鼓膜にて受け取って聞いている音は連続的な情報ですが、コンピュータやCDに入っている音の情報は数字に置き換えられた離散データということです。もっと身近な言葉で言えば、離散データとはデジタルデータであると言っても差し支えないでしょう。ここでは詳細まで知る必要はありませんが、離散化するための方法、サンプルングについて、簡単に理解することが必要です。

図9.4にあるサイン波のサンプルングの様子を示します。

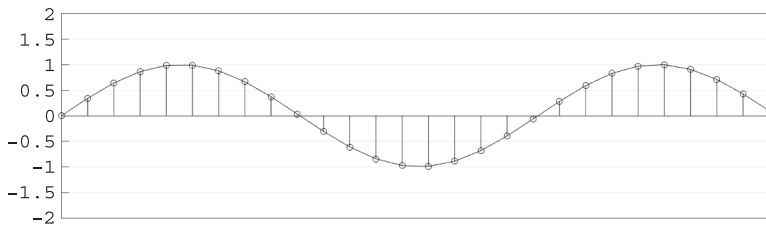


図9.4 あるサイン波のサンプルングの様子

連続的に変化するサイン波を離散データにするために、サンプルングと呼ばれる処理が行われます。サンプルングは連続的なデータを数字に置き換える処理であり、その数字に置き換える際、グラフ中の丸印のようにある一定の時間間隔で連続データの値を抽出（標本化）します。この例の場合、サンプルングの結果として得られる最初の10点の数値列は

$$0, 0.3387, 0.6374, 0.8607, 0.9823, 0.9877, 0.8763, 0.6613, 0.3681, 0.0314, \dots$$

です。この一定の時間間隔は「サンプルング周期」と呼ばれ、ここでは T_s [s]とします。CDではこのサンプルング周期は $1/44100$ [s]と規格で決められており、これは1秒間を44100点で表す、つまりサンプルングするということを意味しています。このサンプルング周期の

逆数を**サンプリング周波数**と呼び、音データのデジタル化においては重要なパラメータとなります。改めてこの関係を式で表すと

$$F_s = \frac{1}{T_s} \text{ [1/s (= Hz)]} \quad (9.6)$$

となります。したがって、CDのサンプリング周波数は44100 Hzということになります。このサンプリング周波数は、現在作成しようとしているサイン波のA4の周波数440 Hzとはまったく別のものですので注意をしてください。

ここで式(9.5)で表された $\sin(2\pi ft)$ の変数 t に対応する「サンプリング周波数が44100 Hzの場合の1秒分の時間軸」を作成します。サンプリング周波数が44100 Hzの音データを1秒分用意するということは、 $1/44100$ 秒の時間間隔でサンプリングされた離散データが44100点分必要ということです。これをMATLABのプログラムで表すと

```
>> t = 0:1/44100:1-1/44100;
```

となり、 t の長さを確認すると

```
>> length(t)
```

```
ans =
```

```
44100
```

```
>>
```

となっています。確認ですが、まず時間軸 t は0秒から1秒までのデータであり、その間隔が $1/44100$ となるように書かれています。そして時間軸の t を $1/44100$ からではなく、0から始めるようにするため、0を含めた分、 $1/44100$ ($1/44100$ 間隔においては1点分に相当)を差し引くことで全体のデータ数を44100点としています。

ここでつくった時間軸を、サンプリング周波数が F_s の場合の n 秒分の時間軸として一般化すると

```
>> t = 0:1/F_s:n-1/F_s;
```

とすることができます。これでようやくサイン波を描く準備ができたので、次項で440 Hzのサイン波を作成します。

9.1.3 いよいよサイン波をMATLABでつくる

下記にサンプリング周波数が44100 Hzの場合の、440 Hzのサイン波を1秒分作成するMATLABのプログラムを示します。

索引

<p>【あ行】</p> <p>移動平均 115</p> <p>インスペクタウィンドウ 124</p> <p>インタラクティブ 2</p> <p>オブジェクト 124</p> <p>【か行】</p> <p>片対数グラフ 39</p> <p>カレントディレクトリ 57</p> <p>関数形式 58</p> <p>逆行列 56</p> <p>強度画像 74</p> <p>クォーテーションマーク 22</p> <p>矩形波 95</p> <p>グラフのタイトル 38</p> <p>構造体 132</p> <p>高速フーリエ変換 107</p> <p>コマンドウィンドウ 1</p> <p>コマンド形式 58</p> <p>コマンドプロンプト 2</p> <p>コロソ 9, 12</p> <p>【さ行】</p> <p>最小二乗 78</p> <p>三角波 97</p> <p>サンプリング周波数 69, 91</p>	<p>周波数軸 109</p> <p>出力引数 3</p> <p>条件分岐 17</p> <p>振幅周波数スペクトル 106</p> <p>正規方程式 83</p> <p>制御文字 62</p> <p>セミコロン 3, 4, 12</p> <p>線形システム 113</p> <p>線種のオプション 36</p> <p>線の色 37</p> <p>線のスタイル 37</p> <p>【た行】</p> <p>畳込み演算 119</p> <p>縦軸のラベル 38</p> <p>転置行列 56</p> <p>ドット 14</p> <p>【は行】</p> <p>ファイル ID 61</p> <p>フィールドの幅 62</p> <p>フォーマット 62</p> <p>複素数 29</p> <p>プロパティ 124</p> <p>平均移動 116</p> <p>ベクトル (配列) 7</p> <p>変換文字 62</p>	<p>【ま行】</p> <p>マーカー 37</p> <p>無限ループ 22</p> <p>【や行】</p> <p>横軸のラベル 38</p> <p>【ら行】</p> <p>両対数グラフ 39</p> <p>ループ 15</p> <p>レイアウトウィンドウ 123</p> <p>【わ行】</p> <p>ワークスペース 4</p> <p>【その他】</p> <p>arcsin 98</p> <p>Callback 関数 127</p> <p>LaTeX フォーマット 39</p> <p>MATLAB エディタ 5</p> <p>permission 61</p> <p>precision 67</p> <p>“fig” ファイル 126</p> <p>“m” ファイル 126</p> <p>“mat” ファイル 57</p> <p>12 平均律 93</p>
--	--	--

コマンド		
<p>【A】</p> <p>abs 29, 109</p> <p>-ascii 59</p> <p>asin 99</p> <p>audioread 69, 71</p> <p>audiorecorder 103</p> <p>audiowrite 69, 104</p> <p>axis 35</p> <p>axis image 73</p>	<p>【B】</p> <p>break 23</p> <p>【C】</p> <p>case 23</p> <p>ceil 27</p> <p>clc 4</p> <p>clear 4</p> <p>close 41</p> <p>close all 41</p> <p>close(n) 41</p>	<p>colormap 53, 72</p> <p>conv 120</p> <p>【D】</p> <p>disp 16</p> <p>【E】</p> <p>edit 5</p> <p>else 17</p> <p>elseif 18</p> <p>exp 28</p>

	[F]	inv	13	round	27
		isfield	144		
fclose	62			[S]	
fft	108			save	57
figure	34, 41	[L]		semilogx	40
figure(n)	41	length	26	shading	52
fix	27	load	58	sign	97
floor	27	log	29	sin	10
fopen	61	log10	29	size	8, 26
for	15	log2	29	soundsc	69
FontSize	38	lookfor	53	sqrt	6
fprintf	62	ls	57	stem	43
fread	68			strcat	132
fscanf	64	[M]		subplot	42
function	30	MarkerSize	49	sum	26
fwrite	67	max	26	surf	52
		mean	26	surfc	53
[G]		mesh	52	switch	23
get	140	meshc	53		
getaudiodata	104	meshgrid	50	[U]	
grid	33			uigetfile	130
guidata	142	[N]			
guide	123	nextpow2	136	[V]	
				view	49
[H]		[O]		[W]	
handles	127, 132	otherwise	23	while	21
histogram	44	[P]		whos	57, 108
		pause	104	[X]	
[I]		pause(n)	104	xlim	35
if	17	plot	10, 33	[Y]	
imag	29	plot3	48	ylim	35
image	54				
imagesc	55	[R]			
imread	74	randn	44, 116		
imwrite	73	real	29		
input	23	recordblocking	104		

— 著者略歴 —

奥野 貴俊 (おくの たかとし)

1996年 工学院大学工学部電子工学科卒業
1998年 工学院大学大学院工学研究科修士課程修了(情報学専攻)
2003年 英国シェフィールド大学自動制御システム工学部博士課程修了
Ph.D.
2005年 リオン株式会社勤務
2010年 英国アルスター大学 Intelligent Systems Research Centre 研究員
2013年 ソラオト経営
2014年 工学院大学非常勤講師
現在に至る

中島 弘史 (なかじま ひろふみ)

1994年 工学院大学工学部電子工学科卒業
1996年 工学院大学大学院工学研究科修士課程修了(情報学専攻)
1996年 工学院大学大学院工学研究科博士課程中退(情報学専攻)
1996年 日東紡音響エンジニアリング株式会社勤務(現、日本音響エンジニアリング株式会社)
2006年 株式会社ホンダ・リサーチ・インスティテュート・ジャパン勤務
2007年 博士(工学)(工学院大学)
2011年 工学院大学准教授
2017年 工学院大学教授
現在に至る

MATLAB ではじめるプログラミング教室

Introductory Programming Course with MATLAB

© Takatoshi Okuno, Hirofumi Nakajima 2017

2017年10月5日 初版第1刷発行



検印省略

著者 奥野 貴俊
中島 弘史
発行者 株式会社 コロナ社
代表者 牛来 真也
印刷所 三美印刷株式会社
製本所 有限会社 愛千製本所

112-0011 東京都文京区千石 4-46-10
発行所 株式会社 コロナ社
CORONA PUBLISHING CO., LTD.
Tokyo Japan

振替 00140-8-14844・電話 (03)3941-3131(代)
ホームページ <http://www.coronasha.co.jp>

ISBN978-4-339-02877-5 C3055 Printed in Japan

(松岡)



<出版者著作権管理機構 委託出版物>

本書の無断複製は著作権法上での例外を除き禁じられています。複製される場合は、そのつど事前に、出版者著作権管理機構(電話 03-3513-6969, FAX 03-3513-6979, e-mail: info@jcopy.or.jp)の許諾を得てください。

本書のコピー、スキャン、デジタル化等の無断複製・転載は著作権法上での例外を除き禁じられています。購入者以外の第三者による本書の電子データ化及び電子書籍化は、いかなる場合も認めていません。落丁・乱丁はお取替えいたします。