微分積分学続論II ・ Matlabノート

（更新：2020年4月8日）

Karel Švadlenka・京都大学
*2020*年 前期

Contents

1　はじめに

このノートは Matlab ソフトウェアを用いて微分方程式を解いたり，解を可視化したりする方法について基礎的な解説をしている．Matlab をお持ちの方はこのまま利用できるが，お持ちでなければ，次のやり方がある：

- 現在のコロナウイルスの感染拡大の状況を支援するため，MathWorks Japan 社から京都大学に対してすべてのツールボックスを利用できる MATLAB キャンパスライセンスを提供していただきました．京都大学に在籍する全ての教職員と学生（kyoto-u.ac.jp ドメインのメールアドレスを保有する全員，st.kyoto-u.ac.jp などのサブドメインも含む）が対象です．インストール台数も無制限です．期間は 4 月 1 日から 8 月 31 日です．ダウンロード方法や使用方法は添付の PDF や京都大学 MATLAB ポータルサイト
https://jp.mathworks.com/academia/tah-portal/kyoto-university-31485310.html
を見てください．ぜひ活用していただければと思います．

- Matlab の無料版である Octave を使う．ただし，Octave のコマンドなどの形式が Matlab と異なる場合がある．

- Matlab Mobile を使う．MathWorks 社でアカウントが作れば，無料で Matlab の機能が利用できる現時点でのサービスである．アカウントを作成するには，https://jp.mathworks.com/index.html でログインのボタンを押して,「アカウントを作成する」リンクをクリックする．そのあと，Matlab Mobile アプリをスマートフォンなどにインストールして，ログインすれば，利用できる．

2　1 階微分方程式

微分方程式

$$y'(x) = f(x,y)$$

の解き方・可視化について説明する．例として，

$$y'(x) = (1-x)y + \frac{x}{2}$$

を用いる．

2.1　関数の定義

右辺の関数 f を定義：

- 1 変数関数：例えば，$g(x) = (1-x)x$，

```
g = @(x) (1-x).*x;
```

- 2 変数関数：上の例では，

```
f = @(x,y) (1-x).*y+x/2;
```

注意： 微分方程式の右辺として定義するとき，自励系などのように x の変数が右辺に陽に現れない場合でも，$@(x,y)$ の部分ですべての変数を記す必要がある．

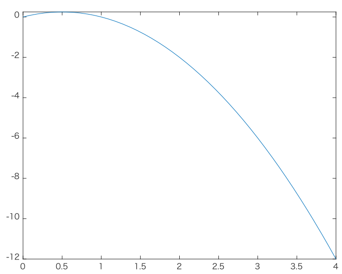
　このように定義すれば，関数のある点での値を求めたり：

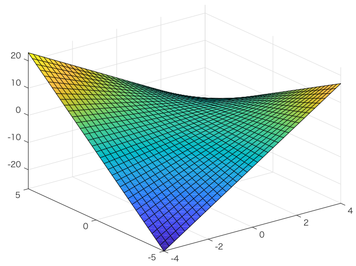
```
g(0.45),  f(1,2)
```

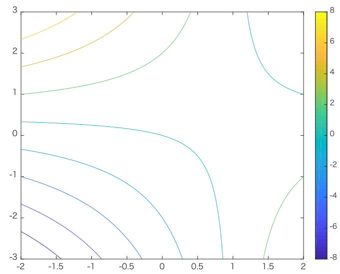
グラフを表示したりできる：

```
fplot(g,[0,4])
fsurf(f,[-4 4 -5 5])
fcontour(f,[-2,2,-3,3]); colorbar
```

上のコマンドはそれぞれ，関数 g の区間 $[0,4]$ でのグラフ，関数 f の長方形領域 $[-4,4]\times[-5,5]$ におけるグラフ，関数 f の長方形領域 $[-2,2]\times[-3,3]$ における等高線を出力する：







2.2　方向の場の表示

　次のコマンドをうてば，微分方程式 $y'=f(x,y)$ の方向の場を表示できる．毎回コピーしなくてもよいように，一連のコマンドを M-file として保存しておくとよい．ここでは，変数 x を表す記号として t を使っている．

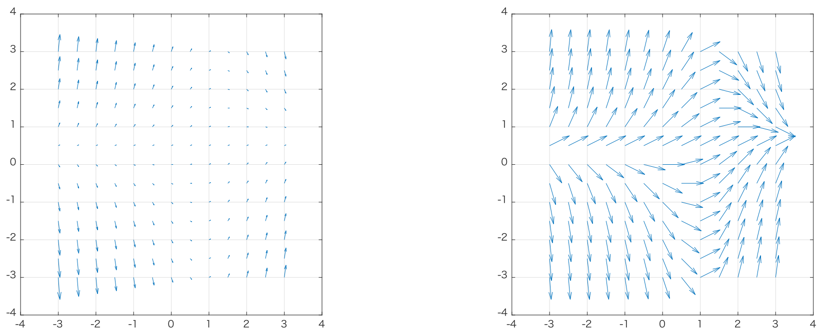
`t=-3:0.5:3; y=-3:0.5:3;`	ベクトル場が表示される格子点の定義
`[T,Y]=meshgrid(t,y);`	格子点の行列の作成
`dT=ones(size(T));`	すべての格子点で $dx=1$ とする
`dY=f(T,Y);`	各格子点で dy を微分方程式の右辺で決める
`quiver(T,Y,dT,dY)`	(x,y) から $(x+dx,y+dy)$ へ向かう矢印を描く
`axis equal; axis([-4 4 -4 4]);`	軸の範囲などを指定
`grid on;`	見やすくするために格子線をひく

最初のコマンドは $x=t$ 変数の範囲（ここでは $[-3,3]$），y 変数の範囲（ここでは $[-3,3]$）と方向場のベクトルを描く間隔（ここでは 0.5 ごとに）を指定している．

これで y' の大きさによって矢印の長さが変わるが，長さの変化が大きいときは見づらいので，次のやり方ですべての矢印の長さを揃える：

`N=sqrt(dT.^2+dY.^2);`	ベクトルの長さを求める
`dT=dT./N; dY=dY./N;`	長さ 1 に正規化
`quiver(T,Y,dT,dY)`	矢印を描く
`axis equal; axis([-4 4 -4 4]);`	軸の範囲などを指定
`grid on;`	見やすくするために格子線をひく

前半と後半のコマンドの結果はそれぞれ以下のような図を出力させる：



矢印は本質的な意味がないので，それを非表示にしたり，長さや色を調節できる：

```
quiver(T,Y,dT,dY,'ShowArrowHead','off','AutoscaleFactor',0.5,'color',[0
0 1])
```

2.3　数値計算で解を求める

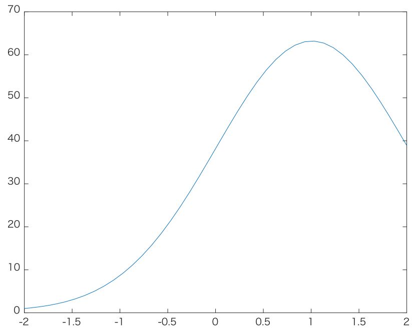
近似的に微分方程式を解くために，Matlab の基本的な ODE ソルバー `ode45` を用いる．初期値問題

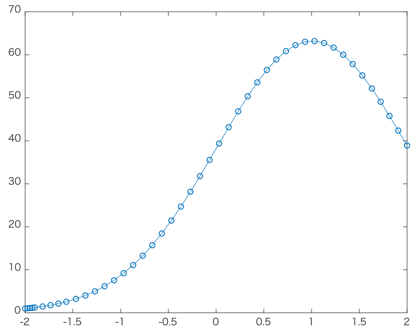
$$y'(x) = f(x,y), \qquad y(x_0) = y_0$$

を区間 $[x_0, x_1]$ で解くには `ode45(f,[x0,x1],y0)` と入力する．例えば，例の微分方程式を初期条件 $y(-2) = 1$ の下，区間 $[-2,2]$ で解く場合，

```
[xs,ys] = ode45(f,[-2,2],1);plot(xs,ys)
```

として，次の図を得る：



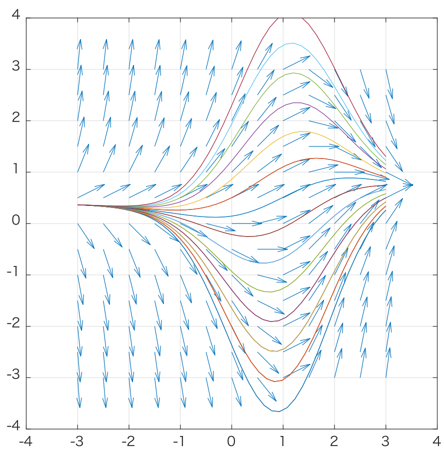


右の図は数値計算に実際に使われた節点をプロットしているが，`plot(xs,ys,'o-')` を使う．

方向の場と解曲線を重ねて表示するときは，最初に描画した図を維持させるコマンド `hold on` をタイプすればよい．すなわち，上で説明した方法で方向の場を表示させた上で，次のコマンドをうつ：

```
hold on
for y0=0.3634:0.0002:0.366
      [xs,ys] = ode45(f,[-3,3],y0); plot(xs,ys)
end
hold off
```

上記のコマンドは初期値 $y_0 = 0.3634$ からはじめて 0.0002 刻みで増やして $y_0 = 0.366$ までの初期値に対する解をすべて出力して，次の結果となる：



`[xs,ys] = ode45(f,[-3,3],y0);` のコマンドは Matlab が使用しているアルゴリズムで決まる点 x での解しか求まらないので，ベクトル v で指定される点 $v = (x_0, \ldots, x_n)$ での解の値が必要な場合は，`[xs,ys] = ode45(f,v,y0);` という形で用いる．ベクトル v の最初の成分は初期の x_0 で成分の数が 3 以上でなければならない．例えば，`[xs,ys] = ode45(f,-1:0.5:1,y0);` は $x_0 = -1, x_1 = -0.5, x_2 = 0, x_3 = 0.5, x_4 = 1$ での解の値をベクトル `ys` に保存する．それらの値を `[xs,ys]` で出力させることができる．
また，解が指定した区間全体で存在しない場合，`Failure at x=...` のように，解が発散する点についてのメッセージが出る．

Matlab には複数の ODE ソルバーがある．例えば，

- ode45 : 陽的 Runge-Kutta (4,5) 公式に基いた単一ステップ ソルバー．基本的なもので，微分方程式を解くときはまずこれを試すとよい．
- ode23 : 陽的 Runge-Kutta (2,3) 公式による単一ステップ ソルバー．方程式が中程度にスティッフなときやそれほど正確な結果を要求しないときに ode45 より効率がよい場合がある．
- ode15s : 1 次ー 5 次の数値微分式 (NDF) に基づく可変ステップ、可変次数ソルバー．スティッフな問題の場合や微分代数方程式を解く場合におすすめ．

などである．詳しくは [1] を参照して下さい．

最も単純な数値解法はオイラー法である．$y'(x) = f(x, y)$, $y(x_0) = y_0$ をオイラー法で解くとき，方程式を解く区間 $[x_0, x_f]$ を n 等分して，初期値 (x_0, y_0) から始めて，短いステップ $h = \frac{x_f - x_0}{n}$ の間，解をその接線で近似する：

$$x_{i+1} = x_i + h, \qquad y_{i+1} = y_i + f(x_i, y_i)h, \qquad i = 0, 1, \ldots, n.$$

Matlab にはこの解法は組み込まれていないが，以下の M-file を作ることで試すことができる：

```
function [x, y] = euler(f, x0, y0, xf, n)
h = (xf - x0)/n;
x = zeros(n+1, 1); y = zeros(n+1, 1);
x(1) = x0; y(1) = y0;
for i = 1:n
     x(i + 1) = x(i) + h;
     y(i + 1) = y(i) + h*f(x(i), y(i));
end
plot(x,y)
```

2.4　シンボリックの形で解を求める

微分方程式をシンボリックに（数式の形で）解くために，シンボリック変数と方程式を定義して，`dsolve` のコマンドを用いる．例えば，パラメータ $a \in \mathbb{R}$ に対して $y' = ay$ を解く場合，

```
syms a y(x)
eqn = diff(y,x) == a*y;
sol(x) = dsolve(eqn)
```

と入力する．このとき，

```
sol(x) = C1*exp(a*x)
```

と，積分定数 C_1 を含む形で出力される．以前に定義した関数 f を用いて方程式を指定することもできる：`eqn = diff(y,x) == f(x,y)`.

変数（ここでは x）や積分定数などの定数に具体的な値を代入することができる．例えば，上の

C_1 を 2 にするために，`subs(sol,'C1',2)` と打つ．

Matlab がシンボリック解を見つけることができなかったら，空シンボルが返される（それは解が存在しないという意味ではない）．このとき，前節で説明した数値解法を適用するとよい．また，複数の解が見つかった場合，それを成分とするベクトルが返される．

方程式 $y' = ay$ に初期条件 $y(0) = 3$ をつけて，$a = 0.7$ を代入したときの解をプロットするためのコマンドは次のようになる：

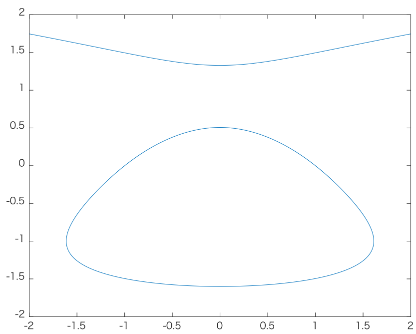
```
syms y(x) a
eqn = diff(y,x) == a*y;
cond = y(0) == 3;
sol(x) = dsolve(eqn,cond)
sol(x) = subs(sol,'a',0.7)
fplot(sol,[0,2])
```

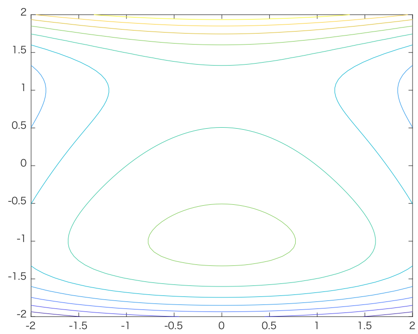
微分方程式の解が陰的な形でしか求まらない場合がある．例えば，$y' = \frac{x}{y^4-1}$ という方程式を初期条件 $y(1) = 0$ の下，Matlab で解いてみると，

```
syms y(x)
eqn = diff(y,x) == x/(y^4-1);
cond = y(1) == 0;
sol(x) = dsolve(eqn,cond)
```

次の結果を返す：`sol(x) = root(z^5 -5*z-(5*x^2)/2+5/2, z, 2)`．これは，解 $y(x)$ が $y^5 - 5y - \frac{5}{2}x^2 + \frac{5}{2} = C,\ C \in \mathbb{R}$ の根であることを意味する．可視化は次のように行う：

```
syms z x
sol1 = z^5 - 5*z - (5*x^2)/2 + 5/2 ==0;
fimplicit(sol1,[-2 2 -2 2])
```





複数の C に対する等高線をプロットする場合は，`fcontour(z^5 - 5*z - (5*x^2)/2 + 5/2,[-2 2 -2 2])` を使う（上の右の図を参照）．

3 高階微分方程式

3.1 シンボリックの形で解を求める

閉じた式で解を求めたい場合，dsolve のコマンドを使う．このとき，関数 x の微分を Dx，2 階微分を D2x のように表す．例えば，方程式

$$x'' - 2x' + 2x = \sin t$$

の一般解を出力するには

```
dsolve('D2x-3*Dx+2*x=sin(t)', 't')
ans =C1*exp(t) + (10^(1/2)*cos(t - atan(1/3)))/10 + C2*exp(2*t),
```

同じ方程式と初期条件 $x(0) = 1, x'(0) = -1$ を満たす特殊解を出力するには

```
dsolve('D2x-3*Dx+2*x=sin(t)', 'x(0)=1', 'Dx(0)=-1', 't')
ans =(5*exp(t))/2 - (9*exp(2*t))/5 + (10^(1/2)*cos(t - atan(1/3)))/10
```

と書けばよい．

3.2 数値計算で解を求める

空気抵抗を考慮したバネの方程式

$$x''(t) = -kx(t) - \alpha x'(t)$$

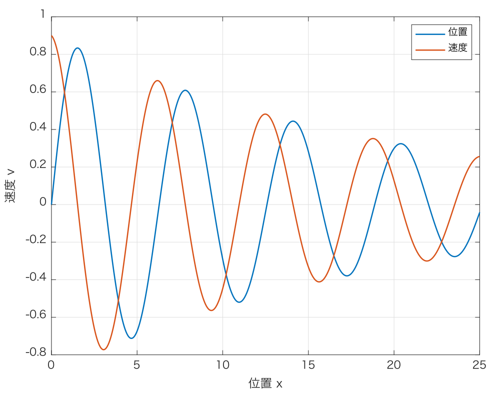
を解くには，まず 1 階化を行う：

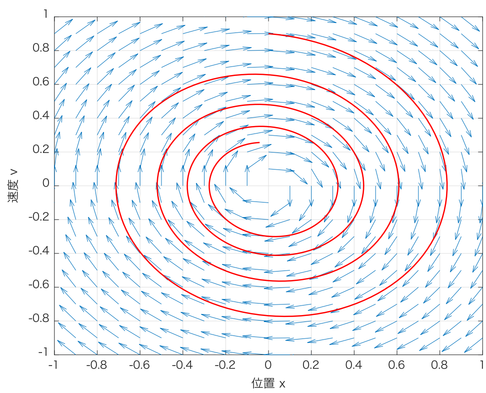
$$\begin{aligned} x'(t) &= v(t) \\ v'(t) &= -kx(t) - \alpha v(t) \end{aligned}$$

以下のコードでは，これを $k = 1, \alpha = 0.1$，そして初期値 $x(0) = 0, v(0) = 0.9$ で解いている．1 階方程式と同じコマンド ode45 を使うが，右辺は ODEbane で定義されるベクトル値関数である．解として出力されるベクトル XV の第 1 成分は x，第 2 成分は v の T で指定された時刻での近似解である．XV の第 1 成分のみを取り出すには，XV(:,1) と打てばよい．これを用いて，左の図では x, v を時間の関数としてプロットした．右の図では，相平面のベクトル場と解曲線を図示した．

```
x = -1:0.1:1;
v = -1:0.1:1;
[x,v] = meshgrid(x,v);
k = 1; alpha = 0.1;
T = 0:0.05:25;
ODEbane = @(t,x)[x(2); -k*x(1)-alpha*x(2)];
[t,XV] = ode45(ODEbane,T,[0;0.9]);
```

```
figure(1);
plot(T,XV(:,1),T,XV(:,2),'LineWidth',1.3);
legend('位置','速度');
grid on;
xlabel('位置 x'); ylabel('速度 v');
figure(2);
V1 = v; V2 = -k*x-alpha*v;
LengthVF = sqrt(V1.^2+V2.^2);
quiver(x,v,V1./LengthVF,V2./LengthVF)
hold on
plot(XV(:,1),XV(:,2),'r','LineWidth',1.3)
grid on; axis ([-1 1 -1 1]);
xlabel('位置 x'); ylabel('速度 v');
hold off
```





4　連立微分方程式

基本的には，高階微分方程式の節で紹介したのと同じ方法で解ける（高階方程式を 1 階連立に直して解いた）ので，少し違う方法を用いる例を挙げることにとどめる．

連立微分方程式

$$\begin{aligned} x' &= x + 2y - z \\ y' &= x + z \\ z' &= 4x - 4y + 5z \end{aligned}$$

を初期条件

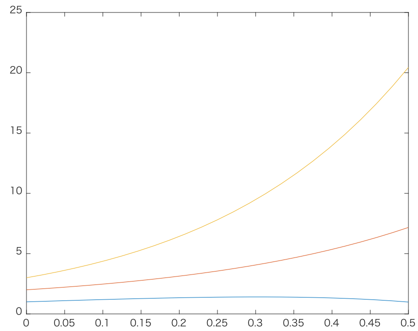
$$x(0) = 1, \qquad y(0) = 2, \qquad z(0) = 3$$

の下でシンボリックに解いて，解をプロットするには，次のようにできる：

```
inits = 'x(0) = 1, y(0) = 2, z(0) = 3';
```

```
[x,y,z] = dsolve('Dx=x+2*y-z','Dy=x+z','Dz=4*x-4*y+5*z',inits)
t = linspace(0,.5,25);
xx=eval(vectorize(x)); yy=eval(vectorize(y)); zz=eval(vectorize(z));
plot(t, xx, t, yy, t, zz)
```

vectorize コマンドは，プロットするために，シンボリックな式を配列の形に変換している．



カオス的な振る舞いを示すことで有名なローレンツの方程式

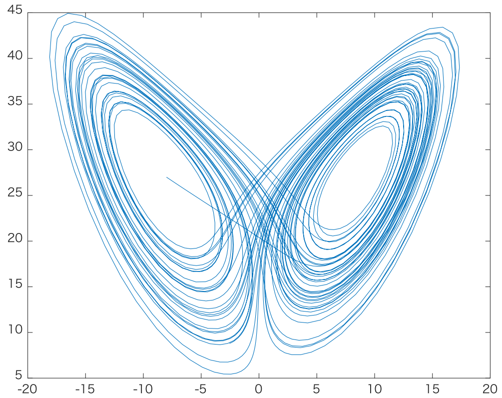
$$
\begin{aligned}
x' &= -\sigma x + \sigma y \\
y' &= -y - xz \\
z' &= -\beta z + xy - \beta\rho
\end{aligned}
$$

をパラメータ $\sigma = 10, \beta = \frac{8}{3}, \rho = 28$ で，初期条件 $x(0) = -8, y(0) = 8, z(0) = 27$ の下で数値的に解いて，x, z-平面で解曲線をプロットするには，次のようにする．まず，右辺の関数を M-file として保存しておく：

```
function xprime = lorenz(t,x);
sig=10; beta=8/3; rho=28;
xprime=[-sig*x(1)+sig*x(2);rho*x(1)-x(2)-x(1)*x(3);-beta*x(3)+x(1)*x(2)];
```

それから，コマンドラインに次のコマンドを打つ：

```
x0 = [-8,8,27];
tspan = [0,50];
[t,x] = ode45('lorenz',tspan,x0);
plot(x(:,1),x(:,3))
```



References

[1] L.F. Shampine, M.W. Reichelt: The Matlab ODE suite,
https://www.mathworks.com/help/pdf_doc/otherdocs/ode_suite.pdf