

情報処理 MATLAB プログラミング

平井 慎一

立命館大学 ロボティクス学科

講義の流れ

- 1 行列とベクトル
- 2 グラフ
- 3 常微分方程式
- 4 パラメータの引き渡し
- 5 まとめ

サンプルプログラム

ウェブサイトの「サンプルプログラム」をクリック。
ダウンロードした zip ファイルを解凍。

`draw_graph.m` グラフを描く

`van_der_Pol.m` ファンデルポール方程式の標準形

`van_der_Pol_solve.m` ファンデルポール方程式を数値的に解く

⋮

ベクトルと行列

列ベクトル

```
x = [ 2; 3; -1 ];
```

行ベクトル

```
y = [ 2, 3, -1 ];
```

行列

```
A = [ 4, -2, 1; ...  
      -2, 5, 2; ...  
      -2, 3, 2 ];
```

ベクトルと行列

記号... は文が続くことを表す.

列ベクトル

```
x = [ 2; ...  
      3; ...  
      -1 ];
```

列ベクトル

```
x = [ 2; 3; -1 ];
```

ベクトルと行列

乗算

```
p = A*x;
```

```
q = y*A;
```

```
>> p
```

```
p =
```

```
1
```

```
9
```

```
3
```

```
>>
```

ベクトルと行列

乗算

```
p = A*x;
```

```
q = y*A;
```

```
>> q
```

```
q =
```

```
     4     8     6
```

```
>>
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1  
    -2     5     2  
    -2     3     2
```

```
>> A(3,2)
```

```
ans =
```

```
     3
```


行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1
    -2     5     2
    -2     3     2
```

```
>> A(3,2) = 6;
```

```
>> A
```

```
A =
```

```
     4     -2     1
    -2     5     2
    -2     6     2
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1  
    -2     5     2  
    -2     3     2
```

```
>> A(3, :)
```

```
ans =
```

```
    -2     3     2
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1  
    -2     5     2  
    -2     3     2
```

```
>> A(:,2)
```

```
ans =
```

```
    -2  
     5  
     3
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1
    -2     5     2
    -2     3     2
```

```
>> A(:,2) = [ 0; 2; 1 ];
```

```
>> A
```

```
A =
```

```
     4     0     1
    -2     2     2
    -2     1     2
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1  
    -2     5     2  
    -2     3     2
```

```
>> A(3,:) = [ 3, -5, -1 ];
```

```
>> A
```

```
A =
```

```
     4     -2     1  
    -2     5     2  
     3    -5    -1
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1  
    -2     5     2  
    -2     3     2
```

```
>> B = A([1,3],:);
```

```
>> B
```

```
B =
```

```
     4     -2     1  
    -2     3     2
```

行列の操作

```
>> A
```

```
A =
```

```
     4     -2     1
    -2     5     2
    -2     3     2
```

```
>> C = A(:, [2,1]);
```

```
>> C
```

```
C =
```

```
    -2     4
     5    -2
     3    -2
```

基本行操作

$A(3,:) = 5*A(3,:);$

3行目を5倍する

$A(1,:) = A(1,:) + 4*A(2,:);$

1行目に2行目の4倍を加える

$A([3,1],:) = A([1,3],:);$

1行目と3行目を交換する

基本行操作

スクリプトファイル `matrix.m`

```
A = [ 4, -2, 1; ...  
      -2, 5, 2; ...  
      -2, 3, 2 ];
```

```
A(3,:) = 5*A(3,:);
```

```
A
```

を作成し、実行せよ。他の基本行操作も確認せよ。

連立一次方程式を解く

```
A = [ 4, -2, 1; ...  
      -2, 5, 2; ...  
      -2, 3, 2 ];
```

```
p = [ 1; 9; 3 ];
```

連立一次方程式 $Ax = p$ を解く

```
>> x = A\p;
```

```
>> x
```

```
x =
```

```
2
```

```
3
```

```
-1
```

```
>> A*x
```

連立一次方程式を解く

スクリプトファイル linear.m

```
A = [ 4, -2, 1; ...  
      -2, 5, 2; ...  
      -2, 3, 2 ];  
p = [ 1; 9; 3 ];  
x = A \ p;  
x
```

を作成し、実行せよ。

```
>> A*x
```

を実行し、解を確認せよ。

グラフ

```
>> x = [0:10]'
```

```
x =
```

```
0
```

```
1
```

```
2
```

```
3
```

```
...
```

```
>> f = x.*x
```

```
f =
```

```
0
```

```
1
```

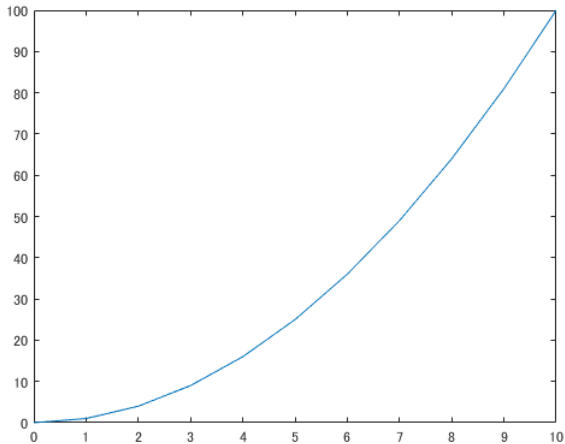
```
4
```

```
9
```

```
...
```

グラフ

```
>> plot(x,f)
```



要素単位の演算

演算子 `.*` や `./` は、要素単位で乗算や除算を実行

$$\begin{bmatrix} 2 \\ 5 \\ -3 \end{bmatrix} .* \begin{bmatrix} 3 \\ -1 \\ -3 \end{bmatrix} = \begin{bmatrix} 6 \\ -5 \\ 9 \end{bmatrix}$$

$$\begin{bmatrix} 6 \\ -5 \\ 1 \end{bmatrix} ./ \begin{bmatrix} 3 \\ -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 1/2 \end{bmatrix}$$

グラフ

```
>> t = [0:0.1:10]'
```

```
t =
```

```
    0
```

```
  0.1000
```

```
  0.2000
```

```
  0.3000
```

```
  ...
```

```
>> x = sin(t)
```

```
x =
```

```
    0
```

```
  0.0998
```

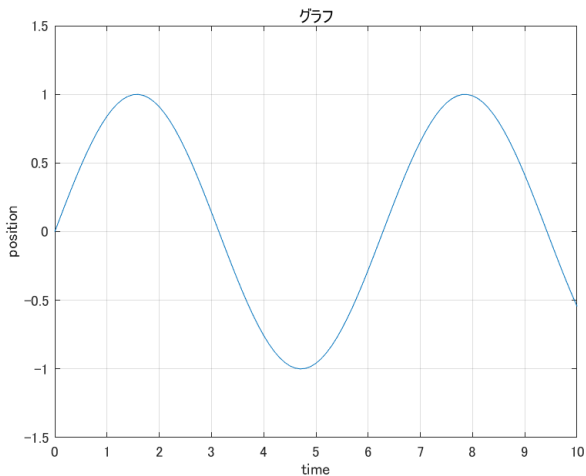
```
  0.1987
```

```
  0.2955
```

```
  ...
```

グラフ

```
>> plot(t,x)
```



ベクトル化関数

関数 **cos**, **sin**, **exp**, **log** 等は、ベクトルを引数とすることができる。

$$\sin \begin{bmatrix} 0 \\ \pi/6 \\ \pi/3 \end{bmatrix} = \begin{bmatrix} \sin(0) \\ \sin(\pi/6) \\ \sin(\pi/3) \end{bmatrix} = \begin{bmatrix} 0 \\ 1/2 \\ \sqrt{3}/2 \end{bmatrix}$$

$$\exp \begin{bmatrix} 0 \\ \log 2 \\ \log 3 \end{bmatrix} = \begin{bmatrix} \exp(0) \\ \exp(\log 2) \\ \exp(\log 3) \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

グラフ

ファイル draw_graph.m

```
t = [0:0.1:10]';  
x = sin(t);  
plot(t,x);  
title(' グラフ');      % グラフの表題  
xlabel('time');        % 横軸のラベル  
ylabel('position');   % 縦軸のラベル  
ylim([-1.5,1.5]);    % 縦軸の範囲  
saveas(gcf,'draw_sine_graph.png'); % グラフの保存
```

ファイル draw_graph.m を実行すると、グラフを描き、描いたグラフをファイルに保存する。

常微分方程式を数值的に解く

ファンデルポール (van der Pol) 方程式

$$\ddot{x} - 2(1 - x^2)\dot{x} + x = 0$$

⇓

$\dot{x} = v$ とおくと $\ddot{x} = \dot{v}$ なので

$$\begin{cases} \dot{x} = v \\ \dot{v} = 2(1 - x^2)v - x \end{cases}$$

⇓

標準形

$$\mathbf{q} = \begin{bmatrix} x \\ v \end{bmatrix}, \quad \dot{\mathbf{q}} = \begin{bmatrix} v \\ 2(1 - x^2)v - x \end{bmatrix}$$

常微分方程式を数值的に解く

標準形を定義するファイル `van_der_Pol.m`

ファイルの名前 "`van_der_Pol`" と関数の名前 "`van_der_Pol`" を一致させる

```
function dotq = van_der_Pol (t, q)
    x = q(1);
    v = q(2);
    dotx = v;
    dotv = 2*(1-x^2)*v - x;
    dotq = [dotx; dotv];
end
```

常微分方程式を数值的に解く

The screenshot shows the MATLAB R2020a interface. The title bar reads 'MATLAB R2020a - academic use'. The top ribbon includes tabs for 'ホーム', 'プロット', and 'アプリ'. The 'New' button (represented by a plus sign) is active, and its dropdown menu is open, showing options: 'スクリプト' (Script), 'ライブスクリプト' (Live Script), '関数' (Function), 'ライブ関数' (Live Function), 'クラス' (Class), 'System object', 'プロジェクト' (Project), 'Figure', 'アプリ' (App), and 'Simulink モデル' (Simulink Model). The 'Script' option is highlighted. The background shows the MATLAB workspace with a folder named 'MATLAB' containing various files like 'graph.tex', 'MATLAB_programn', 'matrix.tex', 'ode.tex', and 'parameter.tex'. The Command Window at the bottom shows the path '2 > edu > 情報処理 > No_2_3_MATLAB_programming'.

常微分方程式を数值的に解く

The screenshot displays the MATLAB R2020a software interface. The title bar reads "MATLAB R2020a - academic use". The top ribbon contains tabs for "ホーム", "プロット", "アプリ", "エディター", "パブリッシュ", and "表示". The "エディター" tab is active, showing a code editor window titled "エディター - Untitled*".

The code editor contains the following MATLAB code:

```
1 function [outputArg1,outputArg2] = untitled(inputArg1,inputArg2)
2 %UNTITLED この関数の概要をここに記述
3 % 詳細説明をここに記述
4     outputArg1 = inputArg1;
5     outputArg2 = inputArg2;
6 end
7
8
```

The left sidebar shows the "現在のフォルダー" (Current Folder) with a tree view of files and folders, including "MATLAB", "graph.tex", "MATLAB_programming.aux", "MATLAB_programming.log", "MATLAB_programming.nav", "MATLAB_programming.out", "MATLAB_programming.pdf", "MATLAB_programming.snm", "MATLAB_programming.synctex.gz", "MATLAB_programming.synctex.gz", "MATLAB_programming.tex", "MATLAB_programming.toc", "MATLAB_programming_page24.png", "matrix.tex", "ode.tex", and "parameter.tex".

At the bottom, the "コマンドウィンドウ" (Command Window) is visible, showing the MATLAB prompt "f>>" and a cursor.

常微分方程式を数值的に解く

関数ファイル `van_der_Pol.m` を作成せよ.

```
>> q = [ 2; 0]
```

```
>> van_der_Pol(0,q)
```

を実行せよ. 作成した関数を用いることができる.

常微分方程式を数值的に解く

スクリプトプログラム `van_der_Pol_solve.m`

```
interval = 0.00:0.10:10.00;  
qinit = [ 2.00; 0.00 ];  
[time, q] = ode45(@van_der_Pol, interval, qinit);  
plot(time, q(:,1), '-');
```

ファイル `van_der_Pol_solve.m` を作成し、実行せよ。

常微分方程式を数値的に解く

時刻 t と変数 x の関係をグラフで表す

```
plot(time, q(:,1), '-');
```

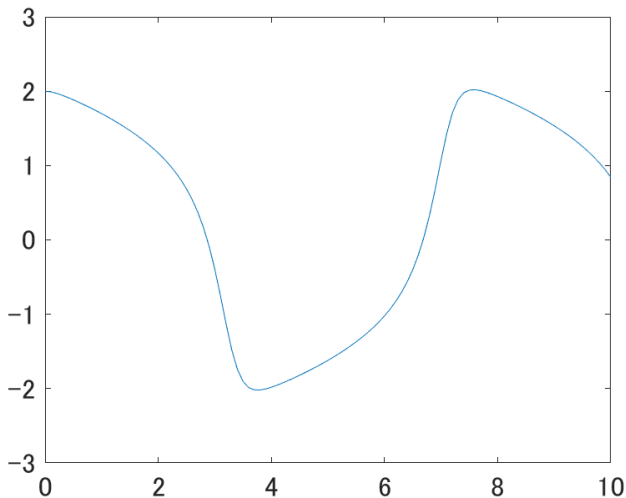
時刻 t と変数 v の関係をグラフで表す

```
plot(time, q(:,2), '-');
```

- '-' 実線
- '--' 破線
- '-.' 一点破線
- ':' 点線

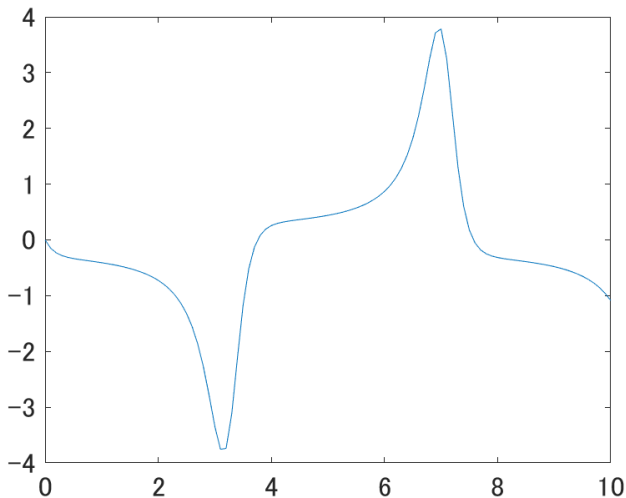
常微分方程式を数值的に解く

時刻 t と変数 x のグラフ



常微分方程式を数値的に解く

時刻 t と変数 v のグラフ



パラメータを有する常微分方程式

微分方程式

$$\ddot{x} + b\dot{x} + 9x = 0$$

b はパラメータ

↓

$$\dot{x} = v$$

$$\dot{v} = -bv - 9x$$

大域変数

関数

```
function dotq = damped_vibration_global (t, q)
    global b;
    x = q(1); v = q(2);
    dotx = v; dotv = -b*v - 9*x;
    dotq = [dotx; dotv];
end
```

プログラム

```
global b;
interval = [0,10];
qinit = [2.00;0.00];
b = 1.00;
[time,q] = ode45(@damped_vibration_global,interval,qinit)
```

入れ子関数

時刻, 状態変数ベクトル, パラメータを引数とする関数

```
function dotq = damped_vibration_param (t, q, b)
    x = q(1); v = q(2);
    dotx = v; dotv = -b*v - 9*x;
    dotq = [dotx; dotv];
end
```

プログラム

```
interval = [0,10];
qinit = [2.00;0.00];
b = 1.00;
damped_vibration = @(t,q) damped_vibration_param (t,q,b);
[time,q] = ode45(damped_vibration,interval,qinit);
```

大域変数 vs 入れ子関数

大域変数

プログラムが単純

大域変数が他の変数と重複する恐れがある

入れ子関数

プログラムがやや複雑

パラメータの値が変わるたびに、関数定義を再実行する必要

他の変数と重複しない

まとめ

MATLAB による数値計算

- ベクトルと行列の計算
- グラフを描く
- 常微分方程式を数値的に解く
- パラメータの引き渡し

付録

違いに注意

- 、 コンマ 要素の区切り
- ； セミコロン 文の最後，列ベクトル (dotq, qinit)
- ⋮ コロン 等間隔の要素列 (interval)
- 。 ピリオド