インタラクティブシステム論
第10回

梶本裕之
Twitter ID kajimoto
ハッシュタグ #ninshiki

日程

4/12　イントロダクション
4/19　Scilabの紹介（西6号館3階PCルーム）
4/26　フーリエ変換
5/03　休日
5/10　フーリエ変換と線形システム
5/17　信号処理の基礎
5/24　信号処理応用1（相関）
5/31　信号処理応用2（画像処理)
6/07　～中間チェック ～
6/14　出張により休講
6/21　ラプラス変換
6/28　古典制御の基礎
7/05　行列
7/12　行列と最小二乗法
7/19　ロボティクス
7/26　～期末チェック～

行列

行列．．．1，2年でやったはず

今日の内容

●固有値とは，固有ベクトルとは
●行列の対角化：どうやるか，なぜうれしいのか
●制御における行列：さわりだけ

キーワードは，
固有値，固有ベクトル，対角化

行列：データ列を変換するもの

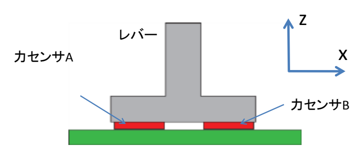
$$\mathbf{y} = \mathbf{A}\mathbf{x} \qquad \mathbf{x} = \mathbf{A}^{-1}\mathbf{y}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

（例1）
y:観測データ，A:システムの性質，x:媒介変数

（例2）
y:フーリエ空間での周波数成分，A:フーリエ変換行列，
x:実空間でのデータ系列

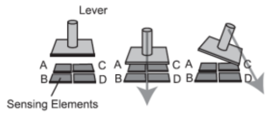
（例）2軸力センサ



$$F_Z = x_A + x_B$$
$$F_X = k(x_A - x_B)$$

$$\underbrace{\begin{bmatrix} F_Z \\ F_X \end{bmatrix}}_{\text{2x1ベクトル}} = \underbrace{\begin{bmatrix} 1 & 1 \\ k & -k \end{bmatrix}}_{\text{2x2行列}} \underbrace{\begin{bmatrix} x_A \\ x_B \end{bmatrix}}_{\text{2x1ベクトル}}$$

（例）多軸力センサ



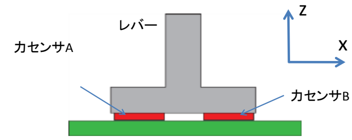
$$F_Z = k_1(x_A + x_B + x_C + x_D)$$
$$F_X = k_2((x_A + x_B) - (x_C + x_D))$$
$$F_Y = k_3((x_A + x_C) - (x_B + x_D))$$

$$\underbrace{\begin{bmatrix} F_z \\ F_x \\ F_y \end{bmatrix}}_{3} = \underbrace{\begin{bmatrix} \quad & \quad & \quad & \quad \\ & & & \\ & & & \end{bmatrix}}_{3\text{x}4\text{行列}} \underbrace{\begin{bmatrix} x_A \\ x_B \\ x_C \\ x_D \end{bmatrix}}_{4}$$

一般には正方行列ではない！！

(例)6軸力センサには8つ以上のセンサエレメント内蔵

力センサのキャリブレーション（較正）

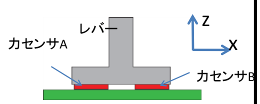


$$F_Z = k_1 x_A + k_2 x_B$$
$$F_X = k_3 x_A + k_4 x_B$$

$$\begin{bmatrix} F_Z \\ F_X \end{bmatrix} = \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix} \begin{bmatrix} x_A \\ x_B \end{bmatrix}$$

k1～k4のパラメータは元々未知．
これを求めなければ使えない！！

逆行列



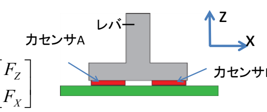
$$\begin{bmatrix} F_Z \\ F_X \end{bmatrix} = \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix} \begin{bmatrix} x_A \\ x_B \end{bmatrix}$$

これを $\mathbf{f} = \mathbf{A}\mathbf{x}$ と書く．

ここで，
「ある力を加えたときの各センサ出力は？」
という逆の関係を考える．

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{f} = \mathbf{G}\mathbf{f} \qquad \begin{bmatrix} x_A \\ x_B \end{bmatrix} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} F_Z \\ F_X \end{bmatrix}$$

逆行列の「測定」



$$\mathbf{x} = \mathbf{G}\mathbf{f} \qquad \begin{bmatrix} x_A \\ x_B \end{bmatrix} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} F_Z \\ F_X \end{bmatrix}$$

(1) Fz=1，Fx=0の力を加え，各センサの出力を記録

$$\begin{bmatrix} x_A \\ x_B \end{bmatrix} = \qquad = \qquad$$

各センサの出力に，逆行列の成分g1，g3が現れる！

(2) Fz=0,Fx=1の力を加え，各センサの出力を記録

$$\begin{bmatrix} x_A \\ x_B \end{bmatrix} = \qquad = \qquad$$

各センサの出力に，逆行列の成分g2，g4が現れる！

逆行列の「測定」

$$\mathbf{x} = \mathbf{G}\mathbf{f} = \mathbf{A}^{-1}\mathbf{f} \qquad \begin{bmatrix} x_A \\ x_B \end{bmatrix} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} F_Z \\ F_X \end{bmatrix}$$

G=A⁻¹の成分，g1～g4が得られたので，
その逆行列を計算すればAが得られる．

$$\mathbf{f} = \mathbf{A}\mathbf{x}$$

まとめると，

$$\left.\begin{aligned} \begin{bmatrix} x_A \\ x_B \end{bmatrix} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} g_1 \\ g_3 \end{bmatrix} \\ \begin{bmatrix} x_A \\ x_B \end{bmatrix} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} g_2 \\ g_4 \end{bmatrix} \end{aligned}\right\} \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix}$$

行列に単位行列をかけたことに相当

単位力でなくて良い

$$\begin{bmatrix} g_1 & g_2 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} \overset{\text{1回目の入力}}{f_{z1}} & f_{z2} \\ f_{x1} & \underset{\text{2回目の入力}}{f_{x2}} \end{bmatrix} = \begin{bmatrix} \overset{\text{1回目の出力}}{m_1} & m_2 \\ m_1 & \underset{\text{2回目の出力}}{m_2} \end{bmatrix}$$

$$\mathbf{G}\mathbf{F} = \mathbf{M}$$

$$\mathbf{G} = \mathbf{M}\mathbf{F}^{-1}$$

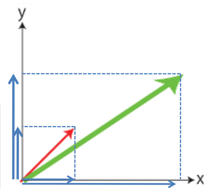
1. 2回**既知の**力ベクトルを加えて，各センサの出力を得る
2. 力ベクトルを並べたものを力行列**F**，
   センサ出力を並べたものを行列**M**とする
3. 力行列の逆行列**F**⁻¹を**M**にかければ，行列**G**が得られる．
4. **Gの逆行列が望んだ「較正行列」A**

ほとんどすべての行列は，ベクトルを「引き延ばす」ものである（1）

$$\mathbf{v} = \mathbf{A}\mathbf{u} \qquad \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

$\mathbf{A} = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$ の時，

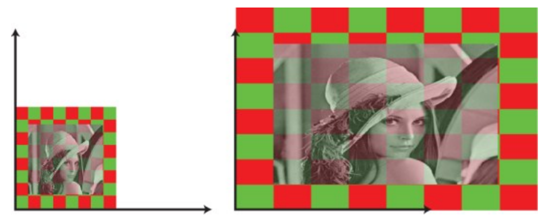
$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} =$$



（作用）x軸成分を3倍，y軸成分を2倍に引き延ばす

ほとんどすべての行列は，ベクトルを「引き延ばす」ものである（1）

$$\mathbf{A} = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$$



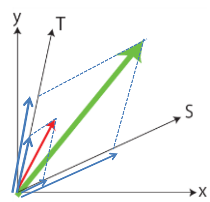
（作用）x軸成分を3倍，y軸成分を2倍に引き延ばす

ほとんどすべての行列は，ベクトルを「引き延ばす」ものである（2）

$\mathbf{A} = \begin{bmatrix} 8 & -2 \\ 3 & 1 \end{bmatrix}$ の時は？．．．．ちょっと分からない．

作用が，
- 謎のS軸成分をs倍，
- 謎のT軸成分をt倍

に引き延ばすことだと仮定してみる．

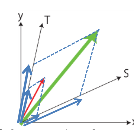


ただしもはや，謎のS,T軸は直交していなくて良い．

固有ベクトルと固有値

$$\mathbf{A} = \begin{bmatrix} 8 & -2 \\ 3 & 1 \end{bmatrix}$$

固有ベクトル，固有値とは，
謎のS，T軸，およびs,t倍
のことである．



（求める手続き）
(1) λ倍されるだけで方向不変のベクトルがあると仮定

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{u}$$

(2) 式変形

$$\mathbf{A}\mathbf{u} = \lambda\mathbf{u} = \lambda\mathbf{I}\mathbf{u}$$

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{u} = \mathbf{0}$$

$$\begin{bmatrix} 8-\lambda & -2 \\ 3 & 1-\lambda \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = 0$$

固有ベクトルと固有値

$$\begin{bmatrix} 8-\lambda & -2 \\ 3 & 1-\lambda \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = 0$$

$$\begin{bmatrix} a-\lambda & b \\ c & d-\lambda \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = 0$$

$$(a-\lambda)u_x + bu_y = 0 \longrightarrow u_y = -(a-\lambda)u_x / b$$

$$cu_x + (d-\lambda)u_y = 0$$

$$cu_x - (d-\lambda)(a-\lambda)u_x / b = 0$$

$$\left((a-\lambda)(d-\lambda) - bc\right)u_x = 0$$

$$\therefore (a-\lambda)(d-\lambda) - bc = 0$$

この解λ1，λ2を固有値と呼び，
対応するベクトルを固有ベクトルと呼ぶ．

固有ベクトルと固有値

$$\begin{bmatrix} 8-\lambda & -2 \\ 3 & 1-\lambda \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = 0$$

$\lambda_1 = 2$ に対応する固有ベクトルは

$$\begin{bmatrix} 8-2 & -2 \\ 3 & 1-2 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = 0$$

大きさを1とすれば，$k = 1/\sqrt{10}$

$\lambda_2 = 7$ に対応する固有ベクトルは

$$\begin{bmatrix} 8-7 & -2 \\ 3 & 1-7 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix} = 0$$

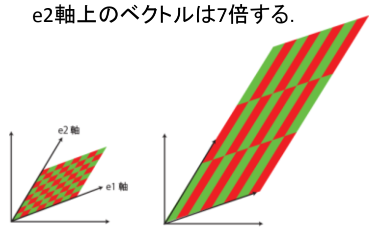
大きさを1とすれば，$k = 1/\sqrt{5}$

作用：e1軸上のベクトルは2倍，
e2軸上のベクトルは7倍する．

固有ベクトルと固有値

$$\mathbf{A}=\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}$$

作用：e1軸上のベクトルは2倍，
e2軸上のベクトルは7倍する.

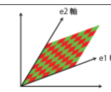


- やはり引き延ばす作用である
- ただし引き延ばす軸（固有ベクトル）は非直交

行列と座標変換

- 引き延ばす作用である
- ただし引き延ばす軸（固有ベクトル）は非直交（斜交）

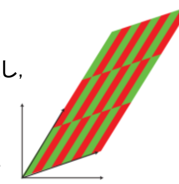
いまいちわかりにくい．．．



行列の作用を，

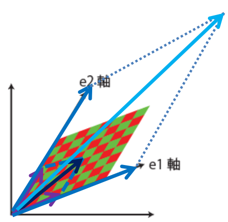
（1）引き延ばし軸での成分表示に変換し，
（2）各成分を引き延ばし，
（3）合成して元に戻す

ように分解すればわかりやすい．．．はず？？



まとめると

行列Tの作用は次の3段階に分解できる.
（1）引き延ばし軸での成分表示に変換し，
（2）各成分を引き延ばし，
（3）合成して元に戻す

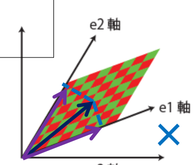


引き延ばし軸での成分表示（誤）

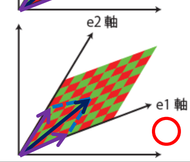
行列の作用を，
（1）引き延ばし軸での成分表示に変換し，
（2）各成分を引き延ばし，
（3）合成して元に戻す

内積で成分を取れる？．．．誤り

斜交座標では
「成分を取り出す操作」≠「内積」



先に（3）「合成」を考えよう.
（1）はその逆のはず



合成して元に戻す操作

行列の作用を，
（1）引き延ばし軸での成分表示に変換し，
（2）各成分を引き延ばし，
（3）合成して元に戻す

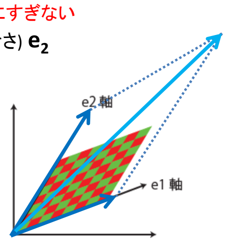
この操作は単なるベクトルの足し算にすぎない

($\mathbf{e}_1$成分の大きさ) $\mathbf{e_1}$＋($\mathbf{e}_2$成分の大きさ) $\mathbf{e_2}$

$$=\begin{bmatrix}\mathbf{e}_1 & \mathbf{e}_2\end{bmatrix}\begin{bmatrix}\mathbf{e}_1\text{軸成分}\\ \mathbf{e}_2\text{軸成分}\end{bmatrix}$$

$\mathbf{P}=\begin{bmatrix}\mathbf{e}_1 & \mathbf{e}_2\end{bmatrix}$ とおいて

$$=\mathbf{P}\begin{bmatrix}\mathbf{e}_1\text{軸成分}\\ \mathbf{e}_2\text{軸成分}\end{bmatrix}$$



引き延ばし軸での成分表示（改）

栄養成分表示　1本(200 ml)あたり

エネルギー	133	kcal
たんぱく質	6.8	g
脂　質	0.9	g
炭水化物	10.0	g
ナトリウム	85	mg
カルシウム	227	mg

行列の作用を，
（1）引き延ばし軸での成分表示に変換し，
（2）各成分を引き延ばし，
（3）合成して元に戻す

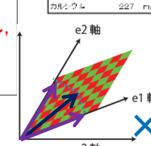
（3）「合成」が，

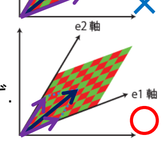
$$\mathbf{P}\begin{bmatrix}\mathbf{e}_1\text{軸成分}\\ \mathbf{e}_2\text{軸成分}\end{bmatrix}$$

で出来るのだから，（1）はその逆のはず.
すなわち

$$\mathbf{P}^{-1}\begin{bmatrix}x\text{軸成分}\\ y\text{軸成分}\end{bmatrix}$$

により**引き延ばし軸での成分表示**ができる



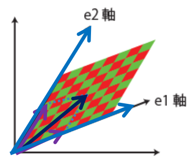


引き延ばし軸での引き延ばし

行列の作用を,
（1）引き延ばし軸での成分表示に変換し,
（2）各成分を引き延ばし,
（3）合成して元に戻す

各成分を
固有ベクトル$\mathbf{e}_1$軸に沿ってλ_1倍,
固有ベクトル$\mathbf{e}_2$軸に沿ってλ_2倍する.

この操作は,

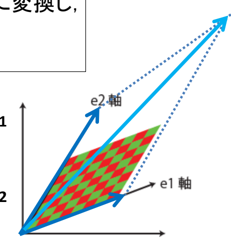


$$\begin{bmatrix} \mathbf{e}_1\text{軸成分を}\lambda_1\text{倍} \\ \mathbf{e}_2\text{軸成分を}\lambda_1\text{倍} \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \mathbf{e}_1\text{軸成分} \\ \mathbf{e}_2\text{軸成分} \end{bmatrix}$$

合成して元に戻す

行列の作用を,
（1）引き延ばし軸での成分表示に変換し,
（2）各成分を引き延ばし,
（3）合成して元に戻す

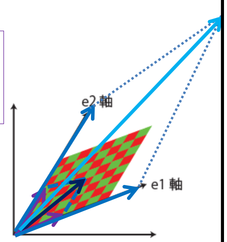
この操作は,
（これまでに求めた$\mathbf{e}_1$成分の大きさ） $\mathbf{e}_1$
＋
（これまでに求めた$\mathbf{e}_2$成分の大きさ） $\mathbf{e}_2$



$$= \mathbf{P} \begin{bmatrix} \text{これまでに求めた}\mathbf{e}_1\text{軸成分} \\ \text{これまでに求めた}\mathbf{e}_2\text{軸成分} \end{bmatrix} \qquad \mathbf{P} = [\mathbf{e}_1 \quad \mathbf{e}_2]$$

まとめると

行列Tの作用は次の3段階に分解できる.
（1）引き延ばし軸での成分表示に変換し,
（2）各成分を引き延ばし,
（3）合成して元に戻す



$$\mathbf{Ax} = \mathbf{P} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_0 \end{bmatrix} \mathbf{P}^{-1}\mathbf{x}$$

固有値を対角成分に並べた行列をTと置く. $\mathbf{T} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_0 \end{bmatrix}$

$$\mathbf{Ax} =$$

行列の対角化を数式で導出

行列の対角化を数式で導出する。
まず、2つの固有値をλ1、λ2、固有ベクトルをe1,e2とする。

2つの式を「まとめて」書くと次のようになる。

[e1,e2]をP、固有値を対角成分に持つ行列をTと書き、左辺のPを右辺に移項すると

（こちらのほうが簡単！ただし
この式の「意味」は前述のとおり）

重要な応用：$\mathbf{A}^n$

$$\mathbf{A}^n\mathbf{x} = \left(\mathbf{PTP}^{-1}\right)^n \mathbf{x}$$
$$= \mathbf{PTP}^{-1}\mathbf{PTP}^{-1}\mathbf{PTP}^{-1} \cdots \mathbf{PTP}^{-1}\mathbf{x}$$
$$= \mathbf{PT}^n\mathbf{P}^{-1}\mathbf{x}$$
$$= \mathbf{P} \begin{bmatrix} \lambda_1^{\ n} & 0 \\ 0 & \lambda_2^{\ n} \end{bmatrix} \mathbf{P}^{-1}\mathbf{x}$$

$$\because \mathbf{T}^n = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}\cdots\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} \lambda_1^{\ n} & 0 \\ 0 & \lambda_2^{\ n} \end{bmatrix}$$

行列のn乗を簡単に計算することができる

重要な結論：nが非常に大きくなった時の$\mathbf{A}^n$

$$\mathbf{A}^n\mathbf{x} = \mathbf{P} \begin{bmatrix} \lambda_1^{\ n} & 0 \\ 0 & \lambda_2^{\ n} \end{bmatrix} \mathbf{P}^{-1}\mathbf{x}$$

行列の固有値λの絶対値が引き延ばしの倍率だから,

固有値が

●一つでも1より大きければ, A^nは**発散**する
●全て1より小さければ, A^nは0に収束する

例：$\mathbf{A}^n$

$$\mathbf{A}=\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}$$

$$\mathbf{A}^2=\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}=\begin{bmatrix}58 & -18\\ 27 & -5\end{bmatrix}$$

$$\mathbf{A}^3=\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}=\begin{bmatrix}58 & -18\\ 27 & -5\end{bmatrix}\begin{bmatrix}8 & -2\\ 3 & 1\end{bmatrix}=\begin{bmatrix}410 & -134\\ 201 & -59\end{bmatrix}$$

$P=\begin{bmatrix}1/\sqrt{10} & 2/\sqrt{5}\\ 3/\sqrt{10} & 1/\sqrt{5}\end{bmatrix}$ $\lambda_1=2\ \lambda_2=7$ を代入して，

$$\mathbf{A}^3=\mathbf{PT}^3\mathbf{P}^{-1}=\begin{bmatrix}1/\sqrt{10} & 2/\sqrt{5}\\ 3/\sqrt{10} & 1/\sqrt{5}\end{bmatrix}\begin{bmatrix}2^3 & 0\\ 0 & 7^3\end{bmatrix}\begin{bmatrix}1/\sqrt{10} & 2/\sqrt{5}\\ 3/\sqrt{10} & 1/\sqrt{5}\end{bmatrix}^{-1}$$

$$=\cdots=\begin{bmatrix}410 & -134\\ 201 & -59\end{bmatrix}$$

固有値が大きいのでどんどん大きくなる

ほとんどすべての行列は，ベクトルを「引き延ばす」ものである（3）

$\mathbf{A}=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$ の時は？．．．．回転と習ったはず

以前と同様に固有値と固有ベクトルを求めてみる．

$$\begin{bmatrix}\cos\theta-\lambda & -\sin\theta\\ \sin\theta & \cos\theta-\lambda\end{bmatrix}\begin{bmatrix}u_x\\ u_y\end{bmatrix}=0$$

$$(a-\lambda)(d-\lambda)-bc=0$$

$$(\cos\theta-\lambda)(\cos\theta-\lambda)+\sin^2\theta=0$$

回転行列の固有値＝exp(jθ)

$$(\cos\theta-\lambda)(\cos\theta-\lambda)+\sin^2\theta=0$$

$$\cos^2\theta-2\lambda\cos\theta+\lambda^2+\sin^2\theta=0$$

$$\lambda^2-2\lambda\cos\theta+1=0$$

$$\lambda=\frac{2\cos\theta\pm\sqrt{4\cos^2\theta-4}}{2}$$

$$=\frac{2\cos\theta\pm j\sqrt{4\sin^2\theta}}{2}$$

$$=\cos\theta\pm j\sin\theta$$

$$=\exp(\pm j\theta)$$

回転行列の固有ベクトル

$\cos\theta+j\sin\theta$ に対応する固有ベクトルは

$$\begin{bmatrix}\cos\theta-\lambda & -\sin\theta\\ \sin\theta & \cos\theta-\lambda\end{bmatrix}\begin{bmatrix}u_x\\ u_y\end{bmatrix}=\begin{bmatrix}\cos\theta-\cos\theta-j\sin\theta & -\sin\theta\\ \sin\theta & \cos\theta-\cos\theta-j\sin\theta\end{bmatrix}\begin{bmatrix}u_x\\ u_y\end{bmatrix}$$

$$=\sin\theta\begin{bmatrix}-j & -1\\ 1 & -j\end{bmatrix}\begin{bmatrix}u_x\\ u_y\end{bmatrix}=0$$

$\therefore u_x=ju_y$ $\quad\mathbf{e}_1=k\begin{bmatrix}j\\ 1\end{bmatrix}$ 大きさを1とすれば例えば，$k=1/\sqrt{2}$

$\cos\theta-j\sin\theta$ に対応する固有ベクトルは

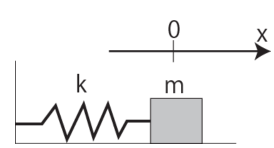
$\mathbf{e}_2=k\begin{bmatrix}1\\ j\end{bmatrix}$ 大きさを1とすれば例えば，$k=1/\sqrt{2}$

（参考）
回転行列も（拡張された）引き延ばしである

- 一般の行列は，固有値，固有ベクトル共に複素数．
- x,y軸に加えて，複素軸も含めた4次元空間中でこれまでと同様の引き延ばしを行う演算とみなせる．
- 複素固有値の絶対値が引き延ばし倍率，偏角が回転角度を表す．

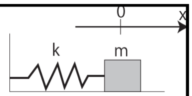
制御における行列

おもりの挙動をシミュレートしたい




```
m=1.0;  //重さ
k=1.0;  //ばね定数
x=1.0;  //初期位置
v=0;    //初期速度
dt=0.1; //時間刻み
record=[];//記録用
for time= 0:dt:10         //時刻
    F=-k*x;               //ばねによって生じる力
    a=F/m;                //生じる加速度
    v= v+a*dt;            //速度
    x= x+v*dt;            //位置
    record = [record,x]; //記録（※テクニック：ベクトルが伸びていく）
end
plot([0:dt:10],record);
```

制御における行列



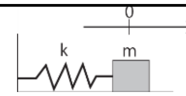

```
for time= 0:dt:10          //時刻
    F=-k*x;                //ばねによって生じる力
    a=F/m;                 //生じる加速度
    v= v+a*dt;             //速度
    x= x+v*dt;             //位置
end
```

位置，速度，加速度を並べた「状態ベクトル」xを定義 $\mathbf{x} = \begin{bmatrix} \mathrm{x} \\ \mathrm{v} \\ \mathrm{a} \end{bmatrix}$

上の関係から，dt時間後の新たな位置，速度，加速度は

$$\begin{bmatrix} \mathrm{x}_{\mathrm{n}} \\ \mathrm{v}_{\mathrm{n}} \\ \mathrm{a}_{\mathrm{n}} \end{bmatrix} = \begin{bmatrix} \quad \\ \quad \\ \quad \end{bmatrix} \begin{bmatrix} \mathrm{x}_{\mathrm{n-1}} \\ \mathrm{v}_{\mathrm{n-1}} \\ \mathrm{a}_{\mathrm{n-1}} \end{bmatrix} = \mathbf{Ax}$$

制御における行列



Scilabコード

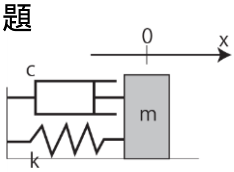
```
m=1.0;   //重さ
k=1.0;   //ばね定数
x=1.0;   //初期位置
v=0;     //初期速度
a=-k/m*x;  //初期加速度
dt=0.01;  //時間刻み
record=[];//記録用

state=[x;v;a];

A=[1,dt,0;0,1,dt;-k/m,0,0];

for time= 0:dt:10  //時刻
  state= A*state;
  record = [record,state(1)];
end

plot([0:dt:10],record);
```

$$\begin{bmatrix} \mathrm{x}_{\mathrm{n}} \\ \mathrm{v}_{\mathrm{n}} \\ \mathrm{a}_{\mathrm{n}} \end{bmatrix} = \begin{bmatrix} 1 & dt & 0 \\ 0 & 1 & dt \\ -k/m & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{x}_{\mathrm{n-1}} \\ \mathrm{v}_{\mathrm{n-1}} \\ \mathrm{a}_{\mathrm{n-1}} \end{bmatrix} = \mathbf{Ax}_{n-1} = \cdots = \mathbf{A}^n \mathbf{x}_0$$

•行列Aのn乗を使えば，
n時刻先の状態をシミュレート可能

•行列Aの固有値を見れば，
システムが将来(n=∞) 収束するか
発散するか予測可能！

レポート課題



●ダンパを加えた際の行列を考え，
同様のシミレーションプログラムを書け

●行列Aの固有値の絶対値が1よりも小さいこと，
すなわち位置が収束することを確認し，コメントに記せ
(Scilabでは固有値はspec()で求められる)

注意：ここで導入した行列はあくまで導入編用で，
シミュレーションとしては不正確です．