動的システム入門

宇都宮大学 工学研究科

吉田 勝俊

http://edu.katzlab.jp/lec/dsys/

2012.7.23 版

はじめに

時間変動する要素の集まりを，**動的システム** (dynamical system) という．例えば，運動するロボットは、時間変動する多数の関節角の集まりなので，動的システムの 1 つである．本書では，このような動的システムの動き方を，うまく調整するための方法を概説する．内容的には，**システム制御**と**最適制御**のつまみ食いである．

まず 1〜4 章において，調整すべき動的システムの表現と**安定論**について学ぶ．そのための関門が，**行列指数関数** e^A である (A は行列)．「自然対数の底 e に行列乗せて何それ？」というのが，初心者の素直な反応だと思う．ともかく，これをやっつけてしまえば，動的システムの学習は半分以上済んだも同然である．

以上の安定論を基礎に，5〜6 章では，動的システムの安定性を人為的に変更する方法を構成する．その集大成を**システム制御理論**というが，学習のポイントは，**斜交成分**である．これを使うと，動的システムの表現をシンプルにできる．その延長線上に，本書第 1 の設計論である**固有値設定問題**が導ける．

続く 7〜9 章は，**最適制御理論**への入門である．最適制御とは，人為的な評価指標 (乗り心地とか省エネとか) を最大または最小とするような，特殊な制御入力を作り出すための方法論である．そのための大きな関門が，**変分法**だ．これをマスターすると，本書第 2 の設計論である**最適レギュレータ**なるものが自然に導ける．

10 章は，以上のおさらいである．自律型ロボットに「棒立て遊び」をさせられたら目標達成だ．そのための一連の作業に違和感なく取り組めたなら，1〜9 章の学習はひとまず成功したと見てよかろう．読者の奮闘に期待したい．

著者しるす

ダウンロード サイト のご案内

本書では，読者がプログラム例 **Code 1〜10**，**Code 13〜18** を実行していくことを前提に，話を進めます[1)]．これらは下記のサイトからダウンロードできます．

`http://edu.katzlab.jp/lec/dsys/prog/`

実行環境によるファイル形式の違い (文字や改行のコード) に対応するため，ファイルは置かず，サイトの本文に記載しました．手持ちのテキスト・エディタに，コピー&ペーストしてご利用ください．プログラムを実行すると理解が深まるので，必ず実行してみて下さい．

[1)] **Code 11, 12, 19**〜は，別の講義 `http://edu.katzlab.jp/lec/dsys/u` 用です．

目次

1

動的システム

時間変化する要素の集りを，**動的システム** (dynamical system) という．**力学系**ともいう．単振り子を例に，必要な用語や算法を導入していこう．

1.1 状態方程式

1.1.1 状態量とその微分

動的システムでは，多数の要素が時間変化するので，その状態は，**時変** (time-varying)[1)]の数ベクトル，

$$\boldsymbol{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} \quad \underset{\text{表記}}{\overset{\text{短縮}}{\Longleftrightarrow}} \quad \boldsymbol{x}(t) = [x_i(t)] \tag{1.1}$$

で表すのが普通である．これを**状態量**または**状態変数** (state variable) という．

動的システムの支配方程式は，状態量 $\boldsymbol{x}(t)$ という時間の関数を解に持つ必要がある．そうなる方程式として，微分方程式がよく使われる[2)]，状態量 $\boldsymbol{x}(t)$ の微分方程式を書き下すには，状態量 $\boldsymbol{x}(t)$ の時間微分が必要だ．そこで，全成分を，同じ時間微分にさらしたものを，ベクトルの時間微分と定める． 以下，$\dot{x} \equiv dx/dt$ と書く．

算法 1.1 (ベクトルの時間微分)

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}^{\cdot} \equiv \begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \vdots \\ \dot{x}_n(t) \end{bmatrix} \quad \underset{\text{表記}}{\overset{\text{短縮}}{\Longleftrightarrow}} \quad \dot{\boldsymbol{x}}(t) \equiv \left[\dot{x}_i(t)\right] \tag{1.2}$$

[1)] 時変の… $\Longleftrightarrow$ 時間変化する….

[2)] 積分方程式を使うケースもある．

例題 1.1　状態量 $\boldsymbol{x}(t) = \begin{bmatrix} \cos 2t \\ \sin 2t \\ e^{-3t} \end{bmatrix}$ の時間微分 $\dot{\boldsymbol{x}}(t)$ を求めよ.

1.1.2　状態方程式

次のような，ベクトル形式で書かれた連立微分方程式を，**状態方程式** (equation of state) という．これが，動的システムの支配方程式の一般形である.

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) \quad \equiv \boldsymbol{f}(x_1, \cdots, x_n) \quad \equiv \begin{bmatrix} f_1(x_1, \cdots, x_n) \\ f_2(x_1, \cdots, x_n) \\ \vdots \\ f_n(x_1, \cdots, x_n) \end{bmatrix} \tag{1.3}$$

►►　n 変数関数 $f(x_1, \cdots, x_n)$ の独立変数 $x_1, \cdots, x_n$ を，ベクトル $\boldsymbol{x} = [x_i]$ にまとめて，$f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$ と書いた.

►► **(余談)**　Octave/Scilab 等の数値解析ソフトでいえば，3 変数の代入を，

```
x=1; y=2; z=3;
f( x, y, z );
```

と書くか，

```
xx=[1,2,3];
f( xx );
```

と書くかの違いだ.

特に，右辺が行列 A で書けてしまうような状態方程式,

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} \tag{1.4}$$

を，**線形** (linear) であるという．そうでない一般の状態方程式 (1.3) は，**非線形** (nonlinear) であるといわれる.

1.2　1 階化

図 1.1 の単振り子を例にとろう．長さ l，質量 m の単振り子の鉛直下向きからの角度を θ とすると，次の運動方程式[3)]が得られる.

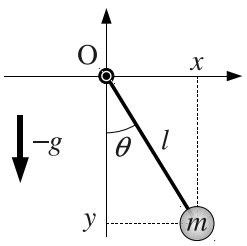


図 1.1　単振り子

[3)] ニュートンの運動法則に従う支配方程式のこと.

$$\ddot{\theta} + c\dot{\theta} + k\sin\theta = 0, \quad c = \gamma/(ml^2),\ k = g/l \tag{1.5}$$

γ は粘性抵抗係数，g は重力加速度である．角度を与えて手を離せば，振り子運動が起こる．すなわち，θ が時間変化するので，単振り子は θ を状態量とする動的システムである．

おいおい，単振り子 (1.5) には 2 階微分 $\ddot{\theta}$ があって，動的システムの一般形 (1.3) に当てはまらないじゃないか．というわけで，次のように **1 階化**する．

目標は，2 階 1 変数を，1 階 2 変数に変形することだ．そのために，角度を第 1 変数 $x_1 = \theta$，角速度を第 2 変数 $x_2 = \dot{\theta}$ においてやる．両者の関係から，まず，

$$\dot{x}_1 = \dot{\theta} = x_2 \quad \therefore \dot{x}_1 = x_2 \tag{1.6}$$

という 1 階の微分方程式が得られる．さらに，$\ddot{\theta} = (\dot{\theta})^{\cdot} = \dot{x}_2$ を使うと，(1.5) は，

$$\dot{x}_2 + c\,x_2 + k\,\sin x_1 = 0 \quad \therefore \dot{x}_2 = -c\,x_2 - k\,\sin x_1 \tag{1.7}$$

のように書き直せる．(1.6), (1.7) を連立すると，

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -c\,x_2 - k\,\sin x_1 \end{cases} \tag{1.8}$$

となるが，これが 1 階化された単振り子の運動方程式である．1 階化された (1.8) は，元の運動方程式 (1.5) と等価である．すなわち，変数変換 $x_1 = \theta$, $x_2 = \dot{\theta}$ を介して，互いに他方を導ける．

最後に，(1.8) をベクトル表記すると，**単振り子の状態方程式**，

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{\cdot} = \begin{bmatrix} f_1(x_1, x_2) \\ f_2(x_1, x_2) \end{bmatrix} = \begin{bmatrix} x_2 \\ -c\,x_2 - k\,\sin x_1 \end{bmatrix} \tag{1.9}$$

が得られる．状態量については，改めて，

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} \tag{1.10}$$

を単振り子の状態量と定めればよい．

その他，一般の n 階の m 連立方程式についても，同様に 1 階化できる．例えば，3 階のシステム，

$$a\dddot{x} + b\ddot{x} + c\dot{x} + dx = 0 \tag{1.11}$$

であれば，変数を増やして $x_1 = x$, $x_2 = \dot{x}$, $x_3 = \ddot{x}$ と置けばよい．あるいは，2 階の 2 連立システム，

$$\ddot{x} + g(\dot{x}, x, \dot{y}, y) = 0, \quad \ddot{y} + h(\dot{x}, x, \dot{y}, y) = 0, \tag{1.12}$$

であれば，通し番号で $x_1 = x$, $x_2 = \dot{x}$, $x_3 = y$, $x_4 = \dot{y}$ などと置けば，問題なく 1 階化できる．

1.3　平衡点と安定性

図 1.1 p2 の単振り子は，下死点[4] $\theta = 0$ で釣合う．釣合うとは，そこで動かなくなることである．そう考えると，振り子は上死点 $\theta = \pi$ でも釣合う．

上下の違いは，**安定性** (stability) である．下死点の釣合いは，ちょっとずらしても重力で元に戻る．このような安定な釣合い点を，**安定平衡点** (stable equilibrium) という．他方，上死点では，重力が逆効果となり，ちょっとでもずらすと振り子は落ちる．このような不安定な釣合い点を，**不安定平衡点** (unstable equilibrium) という．

こうした「釣合い」を数式表現してみよう．まず，釣合った状態では，振り子は停止している．これは，

$$\dot{x}_1 = \dot{\theta} = 0 \tag{1.13}$$

と書ける．さらに，停止し続けるには，加速度も，

$$\dot{x}_2 = \ddot{\theta} = 0 \tag{1.14}$$

でなければならない．なぜなら，ある瞬間の速度が 0 でも，加速度が残っていると，速度が発生する．以上をまとめると，単振り子が「釣合う」ための条件は，

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \mathbb{O} \tag{1.15}$$

である．一般に，次の算法が成立する．

算法 1.2　状態量の時間微分が，

$$\dot{\boldsymbol{x}}(t) = \mathbb{O} \tag{1.16}$$

のとき，その動的システムは**平衡状態**にあるという．この条件を**平衡条件**という． また，平衡条件を状態方程式に代入した，

$$\mathbb{O} = \boldsymbol{f}(\boldsymbol{x}) \tag{1.17}$$

を**平衡方程式** (equilibrium equation) という．その解を**平衡点**という．

単振り子で試すと，平衡方程式は，

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} = \begin{bmatrix} x_2 \\ -cx_2 - k\sin x_1 \end{bmatrix} \tag{1.18}$$

となり，これを解くと確かに 2 つの平衡点

---

[4] 下死点とは，振り子が真下を向いた姿勢のこと．真上を上死点という．

$$\bar{\boldsymbol{x}} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \pi \\ 0 \end{bmatrix} \tag{1.19}$$

が得られる[5)].

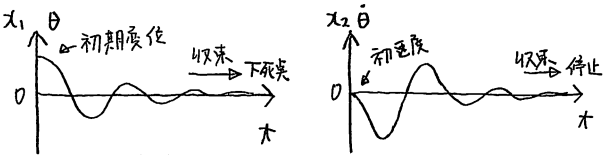
実習 1.1 **Code 1** を実行し，単振り子の安定平衡点と不安定平衡点を観察せよ．

1.4 時間応答と相軌道

Code 1 で観察した単振り子の，安定平衡点 (下死点) まわりの挙動で説明する．

1.4.1 時間応答

角変位 $x_1 = \theta$ と角速度 $x_2 = \dot{\theta}$ の動きを，時間軸上にスケッチすると，次のようになるはずだ．



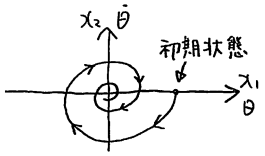
まず，角変位 $x_1(t) = \theta(t)$ については，正の初期変位 $x_1(0) > 0$ から，下死点 $x_1 = \theta = 0$ を何度も行き過ぎながら，最終的に下死点に収束する．また，角速度 $x_2(t) = \dot{\theta}(t)$ については，停止状態 $x_2(0) = 0$ から，負の方向 (変位の逆方向) にいったん加速し，減速，加速を繰り返しながら，停止状態 $x_2 = 0$ に収束している．

このような，時間軸上に描いた状態量 (の成分) を，**時間応答** (time response) という．振動解析の分野では，振動波形ともいう．

1.4.2 相軌道

単振り子の角速度 $\dot{\theta}$ の時間変化は，角度 θ と連動している．変数 θ と同じ変数の微分 $\dot{\theta}$ なのだから当然だ．ちなみに，上で描いた手書きの時間応答は，連動のさせ方が不正確である．だからといって，2 つのグラフを見比べるのは面倒だ．

そこで，状態量 $\boldsymbol{x}(t) = \big(\theta(t), \dot{\theta}(t)\big)$ の時間変化を，そのまま (x_1, x_2) 平面上に描いてみる．



[5)] 正確には，n を整数として，$(x_1, x_2) = (n\pi, 0)$ が単振り子の平衡点．ぐるぐる回れるので．

得られた曲線を，**相軌道** (phase portrait) という．その土台となる (x_1, x_2) 平面を，**相平面**，3 次元以上なら，**相空間** (phase plane/space) という．制御工学の分野では，相空間を**状態空間** (state space) ともいう[6)]．

実習 1.2　**Code 1** の単振り子の運動について，角変位 $x_1(t)$ と角速度 $x_2(t)$ の時間応答と，相軌道 $\boldsymbol{x}(t) = \big(x_1(t), x_2(t)\big)$ を観察せよ．

問題 1.1　次の非線形動的システムについて，

$$\ddot{x} + \dot{x} - x + x^3 = 0$$

(1)1 階化せよ．(2) 平衡点を求めよ．(3) 時間応答と相軌道を描け．

♣ 1 章の補足

● 例題 1.1 p2 の解答例

算法 1.1 より，

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} \cos 2t \\ \sin 2t \\ e^{-3t} \end{bmatrix}^{\cdot} \equiv \begin{bmatrix} (\cos 2t)^{\cdot} \\ (\sin 2t)^{\cdot} \\ (e^{-3t})^{\cdot} \end{bmatrix} = \begin{bmatrix} -2\sin 2t \\ 2\cos 2t \\ -3e^{-3t} \end{bmatrix} \quad /\!/$$

● 実習 1.1 p5 の解答例

初期値 `x0` を `[0;0]` の付近にすれば安定平衡点 (下死点) まわりの挙動が観察できる．`[π;0]` の付近にすれば不安定平衡点 (上死点) まわりの挙動となる．

● 実習 1.2 p6 の解答例

Code 1 の `for` から `endfor` までを，

```
subplot(2,2,1); plot(tt,xx(:,1));
xlabel("t"); ylabel("x_1(t)");
subplot(2,2,2); plot(tt,xx(:,2));
xlabel("t"); ylabel("x_2(t)");
subplot(2,2,3); plot(xx(:,1),xx(:,2));
xlabel("x_1(t)"); ylabel("x_2(t)");
```

に書き換えると，$x_1(t)$ と $x_2(t)$ の時間応答と，相軌道 $\big(x_1(t), x_2(t)\big)$ が観察できる．

● 問題 1.1 p6 の略解

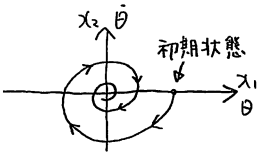
$x_1 = x$, $x_2 = \dot{x}$ とおいて 1 階化できる．平衡点は $\bar{x}_2 = 0$, $\bar{x}_1 = -1, 0, 1$.

[6)] 同様の言い換えとして，**状態軌道**はよく聞くが，「状態平面」は聞かないかなあ …

2

ベクトル場と線形化

単振り子について，実習 1.2 で見た相軌道はこんなだった．



すなわち，下死点 $x_1 = \theta = 0$ での停止状態 $x_2 = \dot{\theta} = 0$ に向う単振り子の運動は，相軌道として見ると，相空間の原点 $(x_1, x_2) = (0, 0)$ に吸い込まれる渦巻になる．この渦巻の中心に，単振り子の安定平衡点 $(0, 0)$ がある．

2.1　ベクトル場

このような平衡点の配置を見るのに，便利な作図法がある．**ベクトル場** (vector field) という．ベクトル場とは，相軌道 $\boldsymbol{x}(t)$ の速度 $\dot{\boldsymbol{x}}(t)$ を作図したものだ．ベクトル場を描くには，状態方程式を微分方程式と見ないで，速度 $\dot{\boldsymbol{x}}(t)$ の定義式と見る．

単振り子の状態方程式を例にとろう．

$$\dot{\boldsymbol{x}} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{\cdot} = \begin{bmatrix} x_2 \\ -k \sin x_1 - c x_2 \end{bmatrix}, \quad c = \frac{\gamma}{ml^2},\ k = \frac{g}{l} \tag{2.1}$$

これを速度の定義式とみる．例えば，$k = 1$, $c = 1/2$ のとき，軌道が $\boldsymbol{x} = (2, 3)$ に差し掛かったときの速度成分は，速度の定義式 (2.1) より，

$$\dot{\boldsymbol{x}} = \begin{bmatrix} x_2 \\ -k \sin x_1 - c x_2 \end{bmatrix} = \begin{bmatrix} 3 \\ -1 \cdot \sin(2) - (1/2) \cdot (3) \end{bmatrix} \approx \begin{bmatrix} 3 \\ -2.41 \end{bmatrix}$$

となる．この速度 $\dot{\boldsymbol{x}} = (3, -2.41)$ を矢印で表し，$\boldsymbol{x} = (2, 3)$ を始点として，相空間に描いたものが，ベクトル場である．

実習 2.1　ノートに描け．

一般に，場所に応じて決まる量を**場** (field) というが，特に，場所に応じてベクトルが決まるものをベクトル場という．

単振り子のベクトル場を，図 2.1 に示す．作図には **Code 3** を使用した．図中の矢印が，各点 $\boldsymbol{x}$ における速度ベクトル $\dot{\boldsymbol{x}}$ である．左の渦巻の中心が安定平衡点 $\bar{\boldsymbol{x}} = (0, 0)$，中央の丸印が不安定平衡点 $\bar{\boldsymbol{x}} = (\pi, 0)$，右の渦巻の中心は 1 回転後の安定平衡点 $\bar{\boldsymbol{x}} = (2\pi, 0)$ である．こうしたベクトル場の使い道を述べておく．

2.1.1　相軌道の予測

図 2.1 の曲線は，相軌道である．これから，

- 相軌道は，ベクトル場の矢印をたどるように動く．

ということが分かる．したがって，ベクトル場をたどれば，相軌道の概形が予測できる．

実習 2.2　図 2.1 の適当な場所にペン先を置き，矢印をたどりながら，そこを初期値とする軌道の予想を描け．

2.1.2　平衡点の予測

図 2.1 からもう 1 つ分かることは，

- 内向きの矢印の中心には，安定平衡点がある．

ということだ．その付近の軌道は，安定平衡点に向って減衰する．また，中央の不安定平衡点の付近では，中央に向う軌道が，左右に逃げている．このように，

- 外向きの矢印の中心には，不安定平衡点がある．

特に，この例では，外向きと内向きの流れが複合しているが，このような複合型の不安定平衡点を，**鞍点** (あんてん，saddle point) という．鞍点が見つかったら，**初期値敏感性** (sensitivity to initial conditions) に注意しよう．鞍点の近くでは，図 2.1 p9 のように，初期値の僅かな違いで，軌道が逆方向に飛ばされる．こうした初期値に敏感な性質を，初期値敏感性という．

2.2　線形化

これまでずっと，右辺が 1 次式 (すなわち行列) で書けないような，非線形状態方程式を扱ってきた．数値計算する分には，平衡点も，軌道も，ベクトル場も簡単に求まり，何の不自由もない．しかし，1 つだけ難点がある．非線形のままだと，次章で学ぶような，安定性の一般論が展開できないのだ[1)]．

[1)] 非線形用にも**リアプノフの方法**というのがあるが，汎用性は期待できない．リアプノフ関数というのを試行錯誤的に見つける必要があり，見つからない場合も多い．

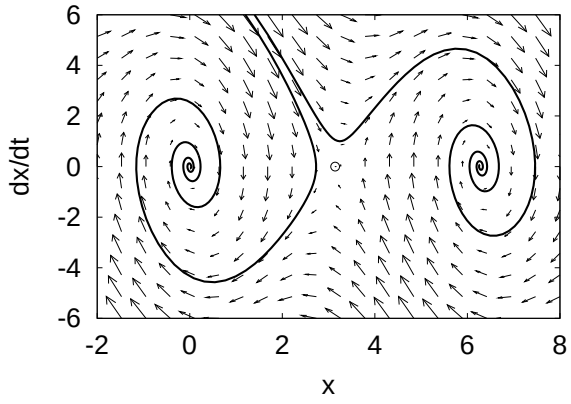
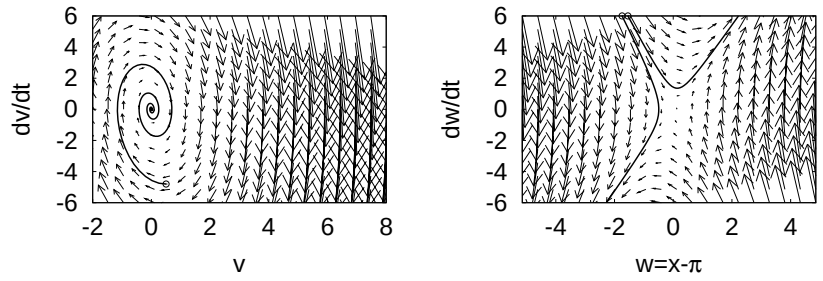


図 2.1 単振り子 (2.1) のベクトル場



(a) $\bar{\boldsymbol{x}} = (0, 0)$ での線形化　　(b) $\bar{\boldsymbol{x}} = (\pi, 0)$ での線形化

図 2.2 線形化方程式 (2.2), (2.4) のベクトル場 (線形化ベクトル場)

ベクトル場を観察すれば，おおよその安定性は分かるが，それは状態量が 2 次元までの話だ．状態量が n 次元になると，ベクトル場の矢印も n 次元ベクトルとなるので，もはや簡明な図示は得られず，見た目で判断する方法は使えない．

そこで，非線形な状態方程式にそっくりな，線形の状態方程式を作る．この操作を，**線形化** (linearization) という．

2.2.1 単振り子の線形化

(a) 安定平衡点 $\bar{\boldsymbol{x}} = (0, 0)$ まわりの線形化　単振り子の状態方程式 (2.1) p7 に，よく知られた近似 $\sin x \approx x$ を使うと，

$$\dot{\boldsymbol{x}} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{\cdot} = \begin{bmatrix} x_2 \\ -k \sin x_1 - c x_2 \end{bmatrix} \approx \begin{bmatrix} x_2 \\ -k x_1 - c x_2 \end{bmatrix}$$

となる．小さい x_i を改めて v_i と書くと，

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}^{\cdot} = \begin{bmatrix} v_2 \\ -k v_1 - c v_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{2.2}$$

が得られる．違う変数 $\boldsymbol{v} = (v_1, v_2)$ で書くのは，元の状態方程式とは式の形が違うからである．これを，**線形化方程式** (linearized equation) という．この 1 次式しか含まない線形化方程式を使って，他人の空似を実現する．

線形化方程式 (2.2) のベクトル場を，図 2.2 の (a) に示す．これを，**線形化ベクトル場**という．曲線は相軌道 $\boldsymbol{v}(t)$，白丸は初期値を表す．元の図 2.1 と比較すると，安定平衡点 $\bar{\boldsymbol{x}} = (0, 0)$ の近くでは，ベクトル場も，相軌道も，そっくりだ．他人の空似は大成功といってよい．しかし，安定平衡点 $\bar{\boldsymbol{x}} = (0, 0)$ から離れると，両者のベクトル場は大きく違ってくる．相軌道もまたしかりだ．例えば，図 2.1 にあった中央の鞍点や，右の渦巻は，図 2.2 の (a) には存在しない．このように，

- 線形化の近似精度は，平衡点から離れると悪化する．

(b)　不安定平衡点 $\bar{\boldsymbol{x}} = (\pi, 0)$ まわりの線形化　平衡点から離れた部分をカバーするには，別の線形化方程式にバトンタッチすればよい．そこで今度は，不安定平衡点 $\bar{\boldsymbol{x}} = (\pi, 0)$ を中心に線形化してみる．

そのために，$\bar{\boldsymbol{x}} = (\pi, 0)$ を中心とする状態量 $\boldsymbol{x}' = \boldsymbol{x} - \bar{\boldsymbol{x}} = (x_1 - \pi, x_2)$ をとり，状態方程式を，

$$\begin{bmatrix} x_1' \\ x_2' \end{bmatrix}^{\cdot} = \begin{bmatrix} x_2' \\ -k\sin(x_1' + \pi) - cx_2' \end{bmatrix} = \begin{bmatrix} x_2' \\ k\sin x_1' - cx_2' \end{bmatrix} \tag{2.3}$$

のように書き直す．sin の符号が反転した．これより，2 つ目の線形化方程式は，

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}^{\cdot} = \begin{bmatrix} w_2 \\ kw_1 - cw_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ k & -c \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \tag{2.4}$$

となる．この系の状態量 $\boldsymbol{w}$ は，$\bar{\boldsymbol{x}} = (\pi, 0)$ からの相対変位なので，元の状態量 $\boldsymbol{x}$ と同じ軸にプロットするときは，$\boldsymbol{w}' = \boldsymbol{w} + (\pi, 0) = (w_1 + \pi, w_2)$ とする必要がある．

得られた線形化方程式 (2.4) のベクトル場 (線形化ベクトル場) を，図 2.2 の (b) に示す．今度は，中央の鞍点付近のベクトル場や軌道が良く再現できている．

それと引き換えに，左右の渦巻が消失した．左右に飛された後，渦巻を描くはずの軌道は，まっすぐのままである．ここでもやはり，平衡点 $\bar{\boldsymbol{x}} = (\pi, 0)$ から離れると，誤差が拡大している．

教訓

- 線形化方程式は，平衡点の数だけ存在する．
- 線形化方程式は，対応する平衡点の近くでしか信用できない．

2.2.2　システマティックな線形化

以上の線形化では，状態方程式を書き変える必要があった．このような面倒を解消するために，ここで述べるヤコビ行列という道具を使う．

n 変数関数 $y = f(x_1, \cdots, x_n)$ の独立変数 x_i を，微少量 v_i だけずらす．このとき生じる，従属変数 y のずれ，

$$\delta y \equiv f(x_1 + v_1, \cdots, x_n + v_n) - f(x_1, \cdots, x_n) \tag{2.5}$$

は，**全微分**の公式で，

$$\delta y = \frac{\partial f}{\partial x_1} v_1 + \cdots + \frac{\partial f}{\partial x_n} v_n \tag{2.6}$$

と書ける．(2.5) と (2.6) から，δy を消去すると，次の公式が得られる．

算法 2.1 n 変数関数 $y = f(x_1, \cdots, x_N)$ は，微小量 v_i に対して，

$$f(x_1 + v_1, \cdots, x_n + v_n) = f(x_1, \cdots, x_n) \quad + \quad \frac{\partial f}{\partial x_1} v_1 + \cdots + \frac{\partial f}{\partial x_n} v_n$$

と展開される．

►► (偏微分) 多変数関数 $f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$ の x_i 以外を定数とみて，x_i で微分したものを $\frac{\partial f}{\partial x_i}$ と書き，x_i に関する f の**偏微分係数**という．特に，特定の点 $\boldsymbol{x} = \bar{\boldsymbol{x}} = (\bar{x}_1, \cdots, \bar{x}_n)$ を代入したものを，次のように書く．

$$\frac{\partial f}{\partial x_i}\Big|_{\boldsymbol{x}=\bar{\boldsymbol{x}}}, \quad \frac{\partial f(\bar{\boldsymbol{x}})}{\partial x_i}, \quad \frac{\partial f(\bar{x}_1, \cdots, \bar{x}_n)}{\partial x_i}$$

状態方程式 (1.3) p2 の各成分 $\dot{x}_i = f_i(x_1, \cdots, x_n)$ の x_i を，平衡点 $\bar{x}_i$ からの微小なずれ v_i で，$x_i = \bar{x}_i + v_i$ と書き直すと，

$$\begin{cases} \text{左辺: } \dot{x}_i = (\bar{x}_i + v_i)^{\cdot} = \dot{v}_i, \qquad \because \text{平衡点 } \bar{x}_i \text{ は定数} \\ \text{右辺: } f_i(x_1, \cdots, x_n) = f_i(\bar{x}_1 + v_1, \cdots, \bar{x}_n + v_n) \\ \qquad = f_i(\bar{x}_1, \cdots, \bar{x}_n) \quad + \quad \dfrac{\partial f_i}{\partial x_1} v_1 + \cdots + \dfrac{\partial f_i}{\partial x_n} v_n \quad \because \text{算法 2.1} \\ \qquad = \dfrac{\partial f_i}{\partial x_1} v_1 + \cdots + \dfrac{\partial f_i}{\partial x_n} v_n \quad \because \text{平衡条件 } f_i(\bar{x}_1, \cdots, \bar{x}_n) = 0 \end{cases} \tag{2.7}$$

となる．これらを縦に並べて，

$$\begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}^{\cdot} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} v_1 + \cdots + \frac{\partial f_1}{\partial x_n} v_n \\ \vdots \\ \frac{\partial f_n}{\partial x_1} v_1 + \cdots + \frac{\partial f_n}{\partial x_n} v_n \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{\boldsymbol{x}=\bar{\boldsymbol{x}}} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix}$$

を得る．これが，線形化方程式の一般形だ．出てきた行列を，**ヤコビ行列**という．下付きの $\boldsymbol{x} = \bar{\boldsymbol{x}}$ は，偏微分してから代入する操作を表す．以上，次の算法が得られた．

算法 2.2　状態方程式 $\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x})$ の平衡点 $\bar{\boldsymbol{x}}$ まわりの**線形化方程式**は，

$$\dot{\boldsymbol{v}} = \left(\frac{\partial \boldsymbol{f}\,(\bar{\boldsymbol{x}})}{\partial \boldsymbol{x}} \right) \boldsymbol{v} \tag{2.8}$$

で得られる．状態量 $\boldsymbol{v}(t)$ は平衡点 $\bar{\boldsymbol{x}}$ からの微小な「ずれ」を表す．行列，

$$\frac{\partial \boldsymbol{f}\,(\bar{\boldsymbol{x}})}{\partial \boldsymbol{x}} \equiv \left[\begin{array}{ccc} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_N}{\partial x_1} & \cdots & \frac{\partial f_N}{\partial x_N} \end{array} \right]_{\boldsymbol{x}=\bar{\boldsymbol{x}}} \qquad \text{(偏微分後 } \boldsymbol{x} = \bar{\boldsymbol{x}} \text{ を代入)} \tag{2.9}$$

を，$\bar{\boldsymbol{x}}$ まわりの**ヤコビ行列** (Jacobian matrix) という．

例題 2.1　(2.1) p7 の単振り子を，算法 2.2 で線形化せよ．基準点は $\bar{\boldsymbol{x}} = (0, 0)$ および $(\pi, 0)$ とせよ．

問題 2.1　問題 1.1 p6 の非線形システムを，各平衡点まわりで線形化せよ．

2.3　制御系の線形化

5.3 節 p35 以降では，状態量 $\boldsymbol{x}$ の他に，制御入力といわれるベクトル $\boldsymbol{u}$ を付加した，

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{2.10}$$

という状態方程式を考える．$\boldsymbol{x}$ は n 次元，$\boldsymbol{u}$ は m 次元とする．

$\boldsymbol{x}, \boldsymbol{u}$ の各成分を，基準点 $\bar{x}_i, \bar{u}_j$ からの微小なずれ y_i, v_j で，$x_i = \bar{x}_i + y_i$，$u_j = \bar{u}_j + v_j$ と書くと，(2.7) p11 と同様の計算により，

$$(\bar{x}_i + y_i)^{\cdot} = \dot{y}_i = \left(\frac{\partial f_i}{\partial x_1} y_1 + \cdots + \frac{\partial f_i}{\partial x_n} y_n \right) + \left(\frac{\partial f_i}{\partial u_1} v_1 + \cdots + \frac{\partial f_i}{\partial u_m} v_m \right) \tag{2.11}$$

となる．これを縦に並べて，ヤコビ行列で表記すると，

$$\dot{\boldsymbol{y}} = \left[\begin{array}{ccc} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{array} \right] \left[\begin{array}{c} y_1 \\ \vdots \\ y_n \end{array} \right] + \left[\begin{array}{ccc} \frac{\partial f_1}{\partial u_1} & \cdots & \frac{\partial f_1}{\partial u_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial u_1} & \cdots & \frac{\partial f_n}{\partial u_m} \end{array} \right] \left[\begin{array}{c} v_1 \\ \vdots \\ v_m \end{array} \right]$$

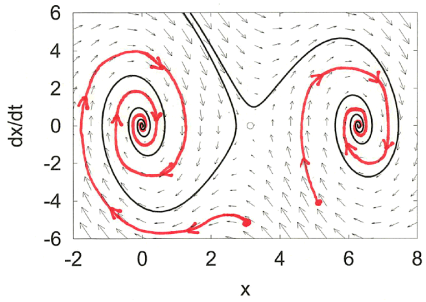
より，

$$\dot{\boldsymbol{y}} = \left(\frac{\partial \boldsymbol{f}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})}{\partial \boldsymbol{x}} \right) \boldsymbol{y} + \left(\frac{\partial \boldsymbol{f}(\bar{\boldsymbol{x}}, \bar{\boldsymbol{u}})}{\partial \boldsymbol{u}} \right) \boldsymbol{v} \tag{2.12}$$

という線形化が得られる．制御入力の基準点はゼロ $\bar{\boldsymbol{u}} = \mathbb{O}$ に取ることが多い．

♣ 2 章の補足

● 実習 2.2 p8 の解答例



● 例題 2.1 p12 の解答例

$\boldsymbol{f} = (f_1, f_2)^T = (x_2, -k\sin x_1 - cx_2)^T$ を順に偏微分すると，

$$\frac{\partial f_1}{\partial x_1} = 0, \quad \frac{\partial f_1}{\partial x_2} = 1, \quad \frac{\partial f_2}{\partial x_1} = -k\cos x_1, \quad \frac{\partial f_2}{\partial x_2} = -c$$

となり，ヤコビ行列は $\frac{\partial \boldsymbol{f}(\bar{\boldsymbol{x}})}{\partial \boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ -k\cos x_1 & -c \end{bmatrix}$．これに平衡点を代入すると，$\frac{\partial \boldsymbol{f}(\begin{bmatrix} 0 \\ 0 \end{bmatrix})}{\partial \boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ -k\cos 0 & -c \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix}$，$\frac{\partial \boldsymbol{f}(\begin{bmatrix} \pi \\ 0 \end{bmatrix})}{\partial \boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ -k\cos \pi & -c \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ k & -c \end{bmatrix}$ となる．確かに，(2.2) p9 と (2.4) p10 と同じ行列が得られる．

● 問題 2.1 p12 の略解

平衡点 $\bar{\boldsymbol{x}} = (\bar{x}_1, \bar{x}_2)^T$ まわりのヤコビ行列は $\dfrac{\partial \boldsymbol{f}\,(\bar{\boldsymbol{x}})}{\partial \boldsymbol{x}} = \begin{bmatrix} 0 & 1 \\ 1 - 3\bar{x}_1^2 & -1 \end{bmatrix}$ //

3

行列指数関数

結論からいうと，線形の状態方程式 $\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t)$ の解は，$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0)$ と書ける．問題は e^{At} の部分で，指数関数の肩に行列が乗っている．この違和感が解消できれば目標達成.

▶▶ 実は大したことはなくて，Octave/Scilab 等では，ごく手軽に，

```
octave:1> A=[1,2;1,1]          octave:2> expm(A)
A =                            ans =
   1   2                          5.9209   7.4388
   1   1                          3.7194   5.9209
```

などと計算できる普通の関数である．通常の指数関数でも，例えば e^2 の値は電卓に聞くしかないので，使い勝手は似たようなものだ.

3.1 指数関数の作り方

まず，通常の指数関数を，1 次元の状態方程式から導く方法を述べる．ここで述べる方法を n 次元化すると行列指数関数が得られる．本節はその前哨戦である.

3.1.1 1 次元の初期値問題

1 次元の状態方程式に，状態量 $x(t)$ の初期値を付加する.

$$\dot{x} = ax, \quad x(0) = c \quad (\text{定数}) \tag{3.1}$$

これを**初期値問題** (initial value problem) という．常微分方程式論によれば，この問題の解は，各 x_0 に対して 1 通りである．というわけで，どんな方法で見つけても，同じ解が得られる.

3.1.2 逐次近似法による接近

ここでは，(3.1) の解を，**ピカールの逐次近似法** (Picard's iteration method)[1] という方法で見つける．この方法は，公式,

$$x_{n+1}(t) = x(0) + \int_0^t a x_n(\tau) d\tau \quad (n = 0, 1, 2, \cdots) \tag{3.2}$$

に関数 $x_n(t)$ を繰り返し代入していく方法である．その結果，数列ならぬ関数列 $x_0(t), x_1(t), x_2(t), \cdots$ が得られるが，ピカールによれば，この関数列は，代入回数の極限 $n \to \infty$ において，初期値問題 (3.1) の解に収束する．

$$x_0(t), x_1(t), x_2(t), \cdots \quad \to x(t) \quad (n \to \infty) \tag{3.3}$$

こうした繰り返し代入 (3.2) によって真の解に接近していく方法を，ピカールの逐次近似法という．

初期値問題の (3.1) の解に接近するには，初項を初期値 $x_0(t) = c$ に選べばよい．(3.2) に代入すると，第 1 項は，

$$x_1(t) = x(0) + \int_0^t a x_0(\tau) d\tau = c + \int_0^t ac\, d\tau = c + act$$

となる．さらに代入すると，第 2 項は，

$$x_2(t) = c + \int_0^t a x_1(\tau) d\tau = c + \int_0^t \left(ac + a^2 c\tau \right) d\tau = c + act + a^2 c \frac{t^2}{2}$$

同じく，第 3 項は，

$$\begin{aligned} x_3(t) = c + \int_0^t a x_2(\tau) d\tau &= c + \int_0^t \left(ac + a^2 c\tau + a^3 c \frac{\tau^2}{2} \right) d\tau \\ &= c + act + a^2 c \frac{t^2}{2} + a^3 c \frac{t^3}{3 \cdot 2} \end{aligned}$$

となる．

実習 3.1 $x_3(t)$ を (3.2) に代入して，$x_4(t)$ を求めよ．

同様に続けていくと，(階乗 $n! = n(n-1)(n-2)\cdots 1$)

$$x_\infty(t) = \left(1 + at + \frac{(at)^2}{2!} + \frac{(at)^3}{3!} + \frac{(at)^4}{4!} + \cdots \right) c \tag{3.4}$$

という無限級数が得られる．ピカールによれば，これが初期値問題 (3.1) の解である．

実際，$t = 0$ を代入すると，$x_\infty(0) = c$ となり初期値を満足する．さらに，(3.4) を t で微分すると，

$$\begin{aligned} \dot{x}_\infty(t) &= \left(0 + a + a^2 t + \frac{a^3 t^2}{2!} + \frac{a^4 t^3}{3!} + \frac{a^5 t^4}{4!} + \cdots \right) c \\ &= a \underbrace{\left(1 + at + \frac{(at)^2}{2!} + \frac{(at)^3}{3!} + \frac{(at)^4}{4!} + \cdots \right) c}_{\bigstar} \end{aligned}$$

となり項が 1 つ減るが，$\bigstar$ には終りがないので，元の (3.4) と区別できない．ゆえに，

$$\dot{x}_\infty(t) = a\bigstar = a x_\infty(t) \tag{3.5}$$

となる．以上，$x_\infty(t)$ は初期値 $x(0) = c$ を満足する微分方程式 (3.1) の解である．

3.1.3　指数関数のお出まし

初期値問題の解 (3.4) のカッコ内の無限級数を，

$$e^{at} = \exp(at) \quad \equiv \quad 1 + at + \frac{(at)^2}{2!} + \frac{(at)^3}{3!} + \frac{(at)^4}{4!} + \cdots \tag{3.6}$$

と書き，**指数関数** (exponential function) という．いわゆる普通の指数関数のことである．指数関数を (3.4) に代入すると，初期値問題 (3.1) の解は，

$$x(t) = e^{at}x(0) \tag{3.7}$$

と書ける．

3.2　ベクトルの積分

次節では，ベクトル $\boldsymbol{x}(t)$ を積分する．これを次のように定義する．

算法 3.1 (ベクトルの積分)

$$\int \boldsymbol{x}(\tau)d\tau \equiv \begin{bmatrix} \int x_1(\tau)d\tau \\ \vdots \\ \int x_n(\tau)d\tau \end{bmatrix} \overset{\text{短縮}}{\underset{\text{表記}}{\Longleftrightarrow}} \int [x_i(\tau)]d\tau \equiv \left[\int x_i(\tau)d\tau\right] \tag{3.8}$$

ようするに，全成分を，同じ積分にさらすことを，ベクトルの積分と定義する．

例題 3.1　ベクトル $\boldsymbol{x}(t) = \begin{bmatrix} t^2 \\ e^t \end{bmatrix}$ の積分 $\int_0^t \boldsymbol{x}(\tau)d\tau$ を求めよ．

次節で使うが，定ベクトル $\boldsymbol{a}$ に対して，(短縮表記 $\boldsymbol{a} = [a_i]$ を用いる)

$$\int_0^t \boldsymbol{a}\, d\tau = \int_0^t [a_i]d\tau \overset{(3.8)}{\underset{\text{さらす}}{=}} \left[\int_0^t a_i\, d\tau\right] \overset{\text{普通の}}{\underset{\text{積分}}{=}} [a_i t] \overset{\text{スカラ倍}}{=} t\,[a_i] = t\boldsymbol{a} \tag{3.9}$$

となる．これをもう一度積分すると，$\boldsymbol{a}$ は定ベクトルより，

$$\int_0^t \tau\boldsymbol{a}\, d\tau = \left(\int_0^t \tau d\tau\right)\boldsymbol{a} = \frac{t^2}{2}\boldsymbol{a} \quad = \frac{t^2}{2!}\boldsymbol{a} \tag{3.10}$$

となる．さらに積分すると，

$$\int_0^t \frac{\tau^2}{2}\boldsymbol{a}\, d\tau = \left(\int_0^t \frac{\tau^2}{2}d\tau\right)\boldsymbol{a} = \frac{t^3}{3\cdot 2}\boldsymbol{a} \quad = \frac{t^3}{3!}\boldsymbol{a} \tag{3.11}$$

となる．(階乗 $n! = n(n-1)(n-2)\cdots 1$)

3.3 行列指数関数の作り方

指数関数の作り方をベクトルで行うと，行列指数関数 e^{At} が出てくる．

3.3.1 n 次元の初期値問題

n 次元の初期値問題を考える．A は行列である．

$$\dot{\boldsymbol{x}} = A\boldsymbol{x}, \quad \boldsymbol{x}(0) = \boldsymbol{c} \quad (\text{定ベクトル}) \tag{3.12}$$

この方程式の解も，各 $\boldsymbol{x}_0$ に対して 1 通りである．

3.3.2 逐次近似法による接近

1 次元と同様に，**ピカールの逐次近似法** [1] を適用する．ベクトル版の公式は，

$$\boldsymbol{x}_{n+1}(t) = \boldsymbol{x}(0) + \int_0^t A\boldsymbol{x}_n(\tau)d\tau \quad (n = 0, 1, 2, \cdots) \tag{3.13}$$

である．ここから出てくる，ベクトル値の関数列 $\boldsymbol{x}_0(t), \boldsymbol{x}_1(t), \boldsymbol{x}_2(t), \cdots$ の極限が，初期値問題 (3.12) の解である．

まず，初項を初期値 $\boldsymbol{x}_0(t) = \boldsymbol{c}$ に選ぶ．(3.13) に代入すると，第 1 項は，

$$\boldsymbol{x}_1(t) = \boldsymbol{x}(0) + \int_0^t A\boldsymbol{x}_0(\tau)d\tau = \boldsymbol{c} + \int_0^t \underbrace{A\boldsymbol{c}}_{\text{定}}\, d\tau = \boldsymbol{c} + tA\boldsymbol{c} \quad \because (3.9)\ \text{p16}$$

となる．さらに代入すると，第 2 項は，

$$\boldsymbol{x}_2(t) = \boldsymbol{c} + \int_0^t A\boldsymbol{x}_1(\tau)d\tau = \boldsymbol{c} + \int_0^t \left(A\boldsymbol{c} + \tau A^2\boldsymbol{c}\right) d\tau = \boldsymbol{c} + tA\boldsymbol{c} + \frac{t^2}{2!}A^2\boldsymbol{c} \ \because (3.10)\ \text{p16}$$

同じく，第 3 項は，

$$\begin{aligned}\boldsymbol{x}_3(t) = \boldsymbol{c} + \int_0^t A\boldsymbol{x}_2(\tau)d\tau &= \boldsymbol{c} + \int_0^t \left(A\boldsymbol{c} + \tau A^2\boldsymbol{c} + \frac{\tau^2}{2!}A^3\boldsymbol{c}\right) d\tau \\ &= \boldsymbol{c} + tA\boldsymbol{c} + \frac{t^2}{2!}A^2\boldsymbol{c} + \frac{t^3}{3!}A^3\boldsymbol{c} \quad \because (3.11)\ \text{p16}\end{aligned}$$

となる．

実習 3.2 $\boldsymbol{x}_3(t)$ を (3.13) に代入し，$\boldsymbol{x}_4(t)$ を求めよ．

同様に続けていくと，

$$\boldsymbol{x}_\infty(t) = \left(E + tA + \frac{t^2}{2!}A^2 + \frac{t^3}{3!}A^3 + \frac{t^4}{4!}A^4 + \cdots\right)\boldsymbol{c} \tag{3.14}$$

という無限級数が得られる (E は単位行列)．これが初期値問題 (3.12) の解である．

実際，$t = 0$ を代入すると，$\boldsymbol{x}_\infty(0) = \boldsymbol{c}$ となり初期値を満足する．さらに，(3.14) を t で微分すると，

$$\dot{\boldsymbol{x}}_\infty(t) = \left(O + A + tA^2 + \frac{t^2}{2!}A^3 + \frac{t^3}{3!}A^4 + \frac{t^4}{4!}A^5 + \cdots \right) \boldsymbol{c} \quad (O \text{ は零行列})$$

$$= A \underbrace{\left(E + tA + \frac{t^2}{2!}A^2 + \frac{t^3}{3!}A^3 + \frac{t^4}{4!}A^4 + \cdots \right) \boldsymbol{c}}_{\bigstar} \quad (3.15)$$

となるが，★ には終りがないので，元の (3.14) と区別できない．ゆえに，

$$\dot{\boldsymbol{x}}_\infty(t) = A\bigstar = A\boldsymbol{x}_\infty(t) \tag{3.16}$$

となる．以上，$\boldsymbol{x}_\infty(t)$ は初期値 $\boldsymbol{x}(0) = \boldsymbol{c}$ を満足する微分方程式 (3.12) の解である．

3.3.3　行列指数関数のお出まし

初期値問題の解 (3.14) のカッコ内の無限級数を，

$$e^{At} = \exp(At) \quad \equiv \quad E + tA + \frac{t^2}{2!}A^2 + \frac{t^3}{3!}A^3 + \frac{t^4}{4!}A^4 + \cdots \tag{3.17}$$

と書き，**行列指数関数** (matrix exponential function) という [1]．行列 A のスカラ倍 tA を，逆順 At に書くのは制御工学の慣習である．

- 制御工学の本 e^{At}　vs　数学の本 e^{tA}

という 2 つの流儀がある．本書は，制御工学の慣習に従う．いずれにしても，行列指数関数を (3.14) に代入すると，初期値問題 (3.12) の解は，

$$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0) \qquad \big(\text{数学では } e^{tA}\boldsymbol{x}(0)\big) \tag{3.18}$$

と書ける．以上，(3.12) p17，(3.17)，(3.18) をまとめると，次の算法が得られる．

算法 3.2 (線形状態方程式の解)　線形状態方程式の初期値問題，(A は行列)

$$\dot{\boldsymbol{x}} = A\boldsymbol{x}, \quad \boldsymbol{x}(0) = \boldsymbol{c} \quad (\text{定ベクトル})$$

の解は，

$$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0)$$

と書ける．e^{At} は，行列指数関数 (3.17) である[1)]．

行列指数関数 e^{At} を手計算するときに，無限級数は使わない．普通の指数関数 e^x を計算するときに，無限級数まで戻らないのと同じだ．行列指数関数は，次の公式で式変形できる．(3) 以外は普通の指数関数と同じである．

[1)] 実習 2.2 p8 でいうと，$\boldsymbol{x}(0)$ は最初にペン先を置く座標である．

算法 3.3 (行列指数関数の性質) A, B は行列，E は単位行列，t は数とする．

(1) $\frac{d}{dt}e^{At} = Ae^{At}$.

(2) 零行列 O に対して，$e^O = E$.

(3) $AB = BA$ のときに限り，$e^{A+B} = e^A e^B$ とできる．

(4) e^A の逆行列は，$(e^A)^{-1} = e^{-A}$.

▶ **証明　A3 節** p20

実習 3.3　状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の軌道 $\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0)$ を，次の行列について求めよ．

$$A_1 = \begin{bmatrix} 0 & 1 \\ -9.8 & -1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 & 1 \\ 9.8 & -1 \end{bmatrix}$$

図 2.2 p9 と同じ初期値を使うと，同じ軌道が得られる．

3.4　推移行列と e^A の数値解法

状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の解に，数の足し引きと，算法 3.3 p19 の指数法則を使うと，

$$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0) = e^{A(t-t_0+t_0)}\boldsymbol{x}(0) = e^{A(t-t_0)}e^{At_0}\boldsymbol{x}(0) = e^{A(t-t_0)}\boldsymbol{x}(t_0) \quad (3.19)$$

のように，初期値の時刻をシフトできる．このシフトをつかさどる $e^{A(t-t_0)}$ を改めて，

$$\Phi(t, t_0) \equiv e^{A(t-t_0)} \quad (3.20)$$

として取り分け，**推移行列** (transition matrix) と呼ぶ．初期時刻 t_0 を定めた後の議論では，略して $\Phi(t)$ とも書く．推移行列 $\Phi(t, t_0)$ は，行列指数関数 e^{At} を忘れて，その先の計算に没頭するための道具である．式変形には次の算法を使う．

算法 3.4 (推移行列の性質)　推移行列 $\Phi(t, t_0) \equiv e^{A(t-t_0)}$ は次の性質を持つ．

(1) 推移法則 …… $\Phi(t_2, t_0) = \Phi(t_2, t_1)\Phi(t_1, t_0)$.

(2) 逆行列 …… $\Phi(t, s)^{-1} = \Phi(s, t)$.　$\Phi(t, t) = E$.　　(E は単位行列)

(3) 行列微分方程式 …… $\dot{\Phi}(t, t_0) = A\Phi(t, t_0)$.　　($t_0$ は定数)

すなわち，推移行列は状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ と同じ形の方程式を満足する．

▶ **証明　B3 節** p20

算法 3.4 (3) を使うと，行列指数関数 e^A の数値解法が作れる．まず，$t_0 = 0$ とおいて，初期値問題,

$$\dot{\Phi}(t) = A\Phi(t), \qquad \Phi(0) \equiv e^{A0} = E \text{ (単位行列)} \quad (3.21)$$

を解くと，$\Phi(t) = e^{A(t-t_0)} = e^{At}$ が得られる．この解 $\Phi(t)$ を，**基本解**という．基本解の時刻 $t = 1$ の値が $\Phi(1) = e^A$ である．

実習 3.4　**Code 6** を実行し，(3.21) の数値解と，組み込みの行列指数関数を比較せよ．

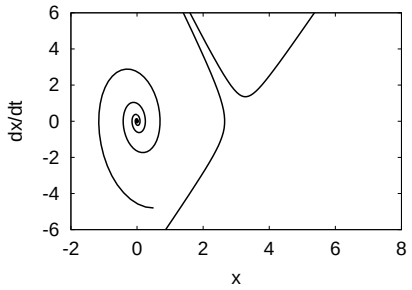
♣ 3 章の補足

● 例題 3.1 p16 の解答例

算法 3.1 より，$\displaystyle\int_0^t \boldsymbol{x}(\tau)d\tau = \int_0^t \begin{bmatrix}\tau^2\\ e^\tau\end{bmatrix} d\tau \equiv \begin{bmatrix}\int_0^t \tau^2 d\tau\\ \int_0^t e^\tau d\tau\end{bmatrix} = \begin{bmatrix}t^3/3\\ e^t-1\end{bmatrix}$ //

● 実習 3.3 p19 の解答例

Code 5 を実行すると，図 2.2 p9 と同じ軌道が得られる．



● 実習 3.4 p19 の解答例

次のような結果が得られる．確かに一致している．

```
octave:1> source "expAt2.m"
Xmat =
  -0.600702   0.010061
  -0.098601  -0.610764
ans =
  -0.600702   0.010061
  -0.098601  -0.610764
```

A3　算法 3.3 p19 の証明

証明のヒントだけ述べる．これらの公式を証明する段階では無限級数を使う．(1) は (3.15) p18 で示した．(2) は無限級数に代入すれば明らか．(3) は，指数関数の指数法則 $e^{x+y}=e^x e^y$ の証明[2)]と同じである．(2), (3) を組み合せると $E=e^O=e^{A-A}=e^A e^{-A}$ より，e^A には逆 e^{-A} がある．これが (4) である．

B3　算法 3.4 p19 の証明

(1) は (3.19) p19 と同じ．(2) は算法 3.3 p19 より $\Phi(t,t)=e^O=E$．これと (1) より前半も出る．(3) は $\dot{\Phi}(t,t_0)=\left(e^{A(t-t_0)}\right)^{\cdot}=Ae^{A(t-t_0)}=A\Phi(t,t_0)$ より明らか．

[2)] 複素関数論や複素解析の教科書に大抵載っている．

4

安定判別

実習 3.3 p19 の行列 A_1 は安定な軌道を作り，A_2 は不安定な軌道を作った．このように，状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の安定性は，A の成分によって変化する．こうした安定性の変化は，A の固有値から簡単に判別できる．

4.1 線形代数の復習

4.1.1 固有値と固有ベクトル

行列 A の作用が，適当なベクトル $\boldsymbol{v} \neq \mathbb{O}$ に対して[1)]，

$$A\boldsymbol{v} = s\boldsymbol{v} \tag{4.1}$$

のように書けるとき，係数 s を，A の**固有値** (eigenvalue) という．そうなるベクトル $\boldsymbol{v} \neq \mathbb{O}$ を，s に属する**固有ベクトル** (eigenvector) という．

►► (固有ベクトルの長さ) 固有ベクトル $\boldsymbol{v}$ の長さは任意である．実際，(4.1) の両辺を定数 a 倍すると，

$$a(A\boldsymbol{v}) = a(s\boldsymbol{v}) \implies A(a\boldsymbol{v}) = s(a\boldsymbol{v})$$

となるので，$\boldsymbol{v}$ の定数倍もまた固有ベクトルである．

求める手順は次の通り．

(1) **固有方程式** (eigen-equation) $|A - sE| = 0$ を解き[2)]，固有値 $s_1, s_2, \cdots$ を求める．

(2) $s = s_i$ を $A\boldsymbol{v}_i = s\boldsymbol{v}_i$ に代入して，s_i に属する固有ベクトル $\boldsymbol{v}_i$ の方向を定める (長さは定まらないので各自で勝手に決める)．

例題 4.1 行列 $A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$ の固有値と固有ベクトルを求めよ．

[1)] $\boldsymbol{v} = \mathbb{O}$ だと，常に等式が成立して意味がないので，$\boldsymbol{v} \neq \mathbb{O}$ とする．

[2)] $|X|$ は行列 X の行列式，E は単位行列を表す．

4.1.2　固有ベクトルが作る基底

これから学ぶ安定判別の計算は，次の算法が発端になる．

算法 4.1　n 次行列 A の固有値 $s_1, \cdots, s_n$ が重複しないとき[3)]，対応する固有ベクトルの組 $\mathcal{V} = \langle \boldsymbol{v}_1, \cdots, \boldsymbol{v}_n \rangle$ は線形独立となる．ゆえに n 次元空間の**基底**となる．

n 次元空間の**基底** (basis) とは，次の 2 条件を満足するベクトルの組 $\mathcal{V} = \langle \boldsymbol{v}_1, \cdots, \boldsymbol{v}_n \rangle$ のことである．

(1) どんな n 次元ベクトル $\boldsymbol{x}$ を持ってきても，適当な係数 $a_1, \cdots, a_n$ で，

$$\boldsymbol{x} = a_1 \boldsymbol{v}_1 + \cdots + a_n \boldsymbol{v}_n \tag{4.2}$$

と書ける．

(2) その係数 $a_1, \cdots, a_n$ の定まり方は，各 $\boldsymbol{x}$ に対して一通りである．

本書の仮定

一般に，固有値が重複すると固有ベクトルの個数が減るので，基底が構成しにくくなる．その場合は「ジョルダン標準形」の理論で基底を構成するが，本書では，重複しないと仮定して省く．これは工業的にはありうる仮定で，実測値からなる行列の成分は，必ず確率的にゆらぐ．ゆえに固有値も確率的にゆらぐので，実測される固有値が，重複条件に確定するとは考えにくい．

4.1.3　行列指数関数の固有値

以上の線形代数をベースに，安定判別のための，強力なアイテムを追加する．次の算法である．証明は章末 (**A4 節** p29) に示す．

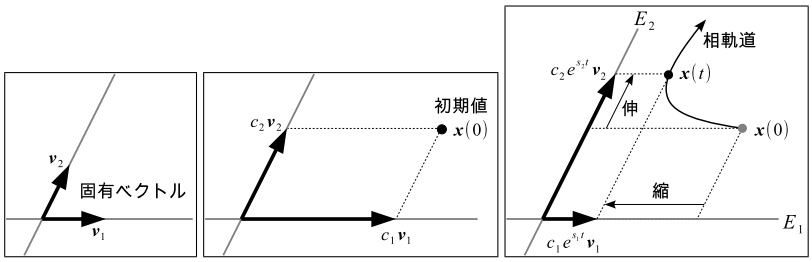
算法 4.2　行列 A の固有値が s であるとき，行列 e^A の固有値は e^s となる．固有ベクトル $\boldsymbol{v}$ は共通である．すなわち，

$$\underset{\text{行列}}{A}\,\boldsymbol{v} = \underset{\text{固有値}}{s}\,\boldsymbol{v} \quad \Longrightarrow \quad \underset{\text{行列}}{e^A}\,\boldsymbol{v} = \underset{\text{固有値}}{e^s}\,\boldsymbol{v}$$

となる．A の定数倍 At についても同様に，$e^{At}\boldsymbol{v} = e^{st}\boldsymbol{v}$ となる．

状態方程式の安定判別は，この算法で完成したも同然だ．あとは，細かな計算を淡々と処理するだけである．

[3)]「固有値が重複しないとき」=「全ての固有値が相異なるとき」



(a) 固有ベクトル (b) 初期値 (c) E_1, E_2 方向の伸縮

図 4.1 固有空間 E_1, E_2 方向の伸縮 (鞍点の場合)

4.2 実固有値の安定性

状態方程式 $\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t)$ の解は，算法 3.2 p18 より，

$$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0), \quad \boldsymbol{x}(0) = \boldsymbol{c} \tag{4.3}$$

と書けた．手始めに 2 次元で考えると，2 次行列 A から固有値・固有ベクトルが 2 組得られる．

$$A\boldsymbol{v}_1 = s_1\boldsymbol{v}_1, \quad A\boldsymbol{v}_2 = s_2\boldsymbol{v}_2 \tag{4.4}$$

ここでは，固有値 s_i は実数と仮定する．このとき，$\boldsymbol{v}_i$ は実ベクトルとなる．

4.2.1 安定性のカラクリ

ここで，算法 4.1 p22 を使うと，どんな初期値でも，

$$\boldsymbol{x}(0) = \boldsymbol{c} \quad = c_1\boldsymbol{v}_1 + c_2\boldsymbol{v}_2 \tag{4.5}$$

と書ける．これを，(4.3) に代入すると，

$$\boldsymbol{x}(t) = e^{At}\boldsymbol{x}(0) = e^{At}\left(c_1\boldsymbol{v}_1 + c_2\boldsymbol{v}_2\right) = c_1\left(e^{At}\boldsymbol{v}_1\right) + c_2\left(e^{At}\boldsymbol{v}_2\right)$$

となる．ここで，算法 4.2 によれば，括弧内の項は，$e^{At}\boldsymbol{v}_i = e^{s_it}\boldsymbol{v}_i$ であるから，

$$\boldsymbol{x}(t) = c_1e^{s_1t}\boldsymbol{v}_1 + c_2e^{s_2t}\boldsymbol{v}_2 \tag{4.6}$$

となって，行列指数関数が消える．この式で動けるのは，指数関数 e^{s_it} の部分だけだ．なぜなら，固有ベクトル $\boldsymbol{v}_i$ は定ベクトル，初期値の展開係数 c_i は定数である．

(4.5) と (4.6) の意味するところを図示すると，図 4.1 のようになる．(a) まず行列 A から固有ベクトル $\boldsymbol{v}_1, \boldsymbol{v}_2$ が確定する．(b) 初期値 $\boldsymbol{x}(0)$ を頂点とする平行四辺形から展開係数 c_1, c_2 が決まる．(c) これに e^{At} を作用させると，固有ベクトルの延長線 E_1, E_2 上で，e^{s_1t}, e^{s_2t} 倍の伸縮が起り，時刻 t の $\boldsymbol{x}(t)$ が決まる．(図は鞍点の場合)

ちなみに，固有ベクトルの延長線 E_1, E_2 を，**固有空間** (eigenspace) または**不変部分空間** (invariant subspace) という．固有空間の重要な性質として，

- 固有空間上の初期値 $\boldsymbol{x}(0)$ から始まる解 $\boldsymbol{x}(t)$ は，そこから出れない．

例えば，E_1 上の初期値は $\boldsymbol{x}(0) = c_1\boldsymbol{v}_1 + 0\boldsymbol{v}_2$ なので，解は $\boldsymbol{x}(t) = c_1 e^{s_1 t}\boldsymbol{v}_1$ となり，E_1 を抜けるための成分を持たないので，未来永劫，E_1 上で伸縮することになる．

以上，解軌道 $\boldsymbol{x}(t)$ の増減は，2 つの指数関数 $e^{s_1 t}$，$e^{s_2 t}$ の増減に帰着する．増減の方向は，固有ベクトル $\boldsymbol{v}_1, \boldsymbol{v}_2$ が与える．以上が安定性のカラクリだ．

4.2.2　安定性の分類

指数関数 $e^{s_1 t}$，$e^{s_2 t}$ が，具体的にどう増減するかは，指数である固有値 s_1, s_2 の値で決まる．単独の指数関数 e^{st} の増減は，

- $s < 0 \implies e^{st} \to 0\ (t \to \infty)$
- $s > 0 \implies e^{st} \to \infty\ (t \to \infty)$

の 2 パターンに分類できるので，s_1, s_2 の正負の組合せによって，表 4.1 のような分類が得られる．

表 4.1　線形状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の安定判別 (実固有値 s_1,s_2，2 次元)

固有値の符号	解挙動 $(t \to \infty)$	分類名
$s_1,\ s_2 < 0$	$\boldsymbol{x}(t) \to 0\boldsymbol{v}_1 + 0\boldsymbol{v}_2 = \boldsymbol{0}$	安定結節点 (stable node)
$0 < s_1,\ s_2$	$\boldsymbol{x}(t) \to \pm(\infty\boldsymbol{v}_1 + \infty\boldsymbol{v}_2) = \pm\boldsymbol{\infty}$	不安定結節点 (unstable node)
$s_1 < 0 < s_2$	$\boldsymbol{x}(t) \to 0\boldsymbol{v}_1 \pm \infty\boldsymbol{v}_2 = \pm\boldsymbol{\infty}$	鞍点 (saddle)

►► (中立安定)　指数が $s = 0$ のとき，指数関数は一定値 $e^{st} = 1$ になるが，これを**中立安定** (neutrally stable) という．人間の平衡機能が中立安定的であるとする報告もあり興味深いが，これを加味すると分類が込み入るので，表 4.1 では省いた．

実習 4.1　実習 3.3 p19 の行列 A_2 の固有値を求め，表 4.1 で判別せよ．

このように，固有値の実部に 1 つでも正のものがあると，軌道は発散する．

4.3　複素固有値の安定性

2 次元の線形状態方程式の解の表示，

$$\boldsymbol{x}(t) = c_1 e^{s_1 t}\boldsymbol{v}_1 + c_2 e^{s_2 t}\boldsymbol{v}_2 \qquad (4.6)\ \text{再掲}$$

において，固有値 s_1, s_2 が複素数の場合を考える．

4.3.1 複素数の復習

必要な算法を挙げておく．(詳細は数学の教科書に譲る)

(1) 複素数 $z = a + ib$ と共役 $\overline{z} = a - ib$．$(i = \sqrt{-1})$

(2) 絶対値 $|a + ib| = \sqrt{a^2 + b^2}$，偏角 $\angle(a + ib) = \tan^{-1}(b/a)$．

※ $\tan^{-1}$ は tan の逆関数．$s = \tan\theta \iff \tan^{-1} s = \theta$．

(3) **オイラーの公式**: $e^{i\theta} = \cos\theta + i\sin\theta$．

(4) **極形式への変形**: $a + ib = |a + ib|e^{i\tan^{-1}\angle(a+ib)} = \sqrt{a^2 + b^2}\, e^{i\tan^{-1}(b/a)}$．

(5) オイラーの公式の共役 $\overline{e^{i\theta}} = e^{-i\theta}$，指数関数の共役 $\overline{e^{z}} = e^{\overline{z}}$．

(6) 複素ベクトル $\boldsymbol{z} = \boldsymbol{a} + i\boldsymbol{b}$ と共役 $\overline{\boldsymbol{z}} = \boldsymbol{a} - i\boldsymbol{b}$．

(7) 積の共役 $\overline{ab} = \overline{a}\,\overline{b}$，スカラ倍の共役 $\overline{a\boldsymbol{b}} = \overline{a}\,\overline{\boldsymbol{b}}$．

(8) 実部 $\mathrm{Re}\{\boldsymbol{a} + i\boldsymbol{b}\} = \boldsymbol{a}$．　虚部 $\mathrm{Im}\{\boldsymbol{a} + i\boldsymbol{b}\} = \boldsymbol{b}$．

(9) 共役の和は実数 (ベクトル) $\boldsymbol{z} + \overline{\boldsymbol{z}} = 2\,\mathrm{Re}\{\boldsymbol{z}\}$．

4.3.2 実数解

実行列の複素固有値は必ず，共役なペア $s, \overline{s}$ として得られる．このとき，固有ベクトルも，初期値の係数も共役なペア $\boldsymbol{v}, \overline{\boldsymbol{v}}$，$c, \overline{c}$ となる．以上を (4.6) に代入すると，

$$\boldsymbol{x}(t) = ce^{st}\boldsymbol{v} + \overline{c}e^{\overline{s}t}\overline{\boldsymbol{v}} = ce^{st}\boldsymbol{v} + \overline{ce^{st}}\,\overline{\boldsymbol{v}} = ce^{st}\boldsymbol{v} + \overline{ce^{st}\boldsymbol{v}} = 2\,\mathrm{Re}\{ce^{st}\boldsymbol{v}\} \qquad (4.7)$$

という実数解が得られる．

4.3.3 歪んだ楕円軌道

$c = a + ib$，$s = \gamma + i\omega$，$\boldsymbol{v} = \boldsymbol{u} + i\boldsymbol{w}$ とおいて，(4.7) の {} 内に代入すると，

$$\begin{aligned} ce^{st}\boldsymbol{v} &= (a + ib)e^{(\gamma+i\omega)t}(\boldsymbol{u} + i\boldsymbol{w}) = e^{\gamma t}\Big\{(a + ib)(\cos\omega t + i\sin\omega t)(\boldsymbol{u} + i\boldsymbol{w})\Big\} \\ &= e^{\gamma t}\Big\{a\cos\omega t\,\boldsymbol{u} - a\sin\omega t\,\boldsymbol{w} - b\cos\omega t\,\boldsymbol{w} - b\sin\omega t\,\boldsymbol{u} + i(\cdots)\Big\} \\ &= e^{\gamma t}\Big\{\cos\omega t(a\boldsymbol{u} - b\boldsymbol{w}) - \sin\omega t(a\boldsymbol{w} + b\boldsymbol{u}) + i(\cdots)\Big\} \end{aligned}$$

となるので，定ベクトル $\boldsymbol{U}_1 = 2(a\boldsymbol{u} - b\boldsymbol{w}), \boldsymbol{U}_2 = -2(a\boldsymbol{w} + b\boldsymbol{u})$ を用いると，(4.7) は，

$$\boldsymbol{x}(t) = 2\,\mathrm{Re}\{ce^{st}\boldsymbol{v}\} = e^{\gamma t}\Big(\cos\omega t\,\boldsymbol{U}_1 + \sin\omega t\,\boldsymbol{U}_2\Big) \quad \equiv e^{\gamma t}\boldsymbol{U}(\omega t) \qquad (4.8)$$

のように整理できる．$\boldsymbol{U}(\omega t)$ は，仮に $\boldsymbol{U}_1 \perp \boldsymbol{U}_2$ なら，きれいな楕円軌道 (を傾けたもの) を表す．しかし一般には，楕円の影を斜めに投影したような形の軌道になる．この楕円軌道に $e^{\gamma t}$ を乗じると，軌道振幅が指数関数的に拡大したり $(\gamma > 0)$，縮小したり $(\gamma < 0)$ する．これが (4.8) が表す解軌道 $\boldsymbol{x}(t)$ である．

このように，複素固有値の場合，

- 実部 γ の符号で，安定 $(\gamma < 0)$ か，不安定 $(\gamma > 0)$ かが決まる．
- 特に，固有値が純虚数 $(\gamma = 0)$ のときは単振動 (楕円軌道) が起こる．
- 虚部 ω は，楕円軌道の角振動数を与える．

ということが分かる．分類名とともに表 4.2 にまとめておく．

実習 4.2　実習 3.3 p19 の行列 A_1 の固有値を求め，表 4.2 で判別せよ．

表 4.2　線形状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の安定判別 (複素固有値 $\gamma \pm i\omega$，2 次元)

固有値実部の符号	分類名
$\gamma < 0$	安定渦状点 (stable focus)
$\gamma > 0$	不安定渦状点 (unstable focus)
$\gamma = 0$	渦心点 (center)　　※単振動
固有値虚部 ω は，楕円軌道 $\boldsymbol{U}(\omega t)$ の角振動数	

4.4　多次元の場合

状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ が n 次元の場合は，固有値・固有ベクトルが n 組，

$$A\boldsymbol{v}_1 = s_1\boldsymbol{v}_1,\ A\boldsymbol{v}_2 = s_2\boldsymbol{v}_2,\ \cdots,\ A\boldsymbol{v}_n = s_n\boldsymbol{v}_n \tag{4.9}$$

となり，解の表示が，

$$\boldsymbol{x}(t) = c_1 e^{s_1 t}\boldsymbol{v}_1 + c_2 e^{s_2 t}\boldsymbol{v}_2 + \cdots + c_n e^{s_n t}\boldsymbol{v}_n \tag{4.10}$$

となる．このなかに複素固有値があれば，共役のペアごとに，(4.8) p25 による実数化，

$$\begin{aligned}\boldsymbol{x}(t) &= \cdots + c_k e^{s_k t}\boldsymbol{v}_k + c_{k+1} e^{s_{k+1} t}\boldsymbol{v}_{k+1} + \cdots \\ &= \cdots + \underbrace{c_k e^{s_k t}\boldsymbol{v}_k + \overline{c_k} e^{\overline{s_k} t}\overline{\boldsymbol{v}_k}}_{e^{\gamma_k t}\boldsymbol{U}_k(\omega_k t)} + \cdots\end{aligned}$$

が起こる．それだけだ．

4.4.1　安定性の判別

多次元の場合は，n 個の固有値 $s_1, \cdots, s_n$ の組合せが多岐にわたるため，表 4.1 p24 や表 4.2 p26 のような分類名は，特に用意されないようだ[4)]．

こうした事情から，多変量を扱うのが仕事の制御工学などでは，解軌道が安定か否かだけを問題にする場合も多い．これは簡単で，n 次元の解，

$$\boldsymbol{x}(t) = c_1 e^{s_1 t}\boldsymbol{v}_1 + c_2 e^{s_2 t}\boldsymbol{v}_2 + \cdots + c_n e^{s_n t}\boldsymbol{v}_n$$

のなかに，1 つでも，固有値実部が正の項があると，他の項が全て負でも，この正の項が無限大まで発達する．

$$\boldsymbol{x}(t) = \cdots + c_k e^{(\text{正})t}\boldsymbol{v}_k + \cdots \quad \to \quad \infty\boldsymbol{v}_k$$

$$\text{または} \quad \boldsymbol{x}(t) = \cdots + e^{(\text{正})t}\boldsymbol{U}_k(\omega_k t) + \cdots \quad \to \quad \infty\boldsymbol{U}_k(\omega_k t)$$

逆にいうと，固有値実部が全て負ならば，軌道 $\boldsymbol{x}(t)$ は安定である．

[4)] 状況に応じて，2 次元の分類名を流用することはよくある．例えば，固有値 $s = -1, -2, 1$ の平衡点を鞍点と呼ぶのは現象的に自然かな．

4.4.2 オーバーシュート (振動) の判別

振動とは，基準点を行ったり来たりするタイプの運動である．固有値に虚部があると，楕円軌道の発生によって，振動が起こる．

例えば，クレーンによる荷物運びを思い浮べよう．クレーンの動きを相当にうまく制御しないと，目標点に到達後も，荷物の往復運動 (振動) はなかなか止まらない．この現象を目標点で観察すると，荷物は目標点を超えて行き過ぎていく．この行き過ぎ現象を，制御工学では，**オーバーシュート** (overshoot) という．オーバーシュートに続いて起こる振動を，**残留振動** (residual vibration) という．

ここで，固有値に虚部が存在すると，対応する解は楕円軌道，すなわち振動する状態量を含むので，必ずオーバーシュートが発生する．

本章のまとめとして，次の教訓を挙げておこう．動的システムを扱う技術者に必須の教訓である．

教訓

(1) 固有値に正の実部が 1 つでもある $\Longrightarrow$ 軌道は不安定になる．

(2) 固有値に虚部がある $\Longrightarrow$ オーバーシュートが起こる．

4.4.3 実固有値のオーバーシュート

一般に，**教訓** (2) の逆は成立しないので注意が必要だ．すなわち，固有値に虚部がなくても，オーバーシュートは起こる可能性がある．実際に起こるかどうかは，初期値 $\boldsymbol{x}(0)$ による．実例を見てみよう．例えば，実固有値の解，

$$x(t) = 2e^{-2t} - e^{-t}$$

について，微分 $\dot{x}(t) = -4e^{-2t} + e^{-t}$ も見ながら増減表を作ると，

t	0	$\cdots$	$\ln 2$	$\cdots$	$\ln 4$	$\cdots$	∞
$x(t)$	1	$\searrow$	0	$\searrow$	$-1/8$	$\nearrow$	-0
$\dot{x}(t)$	$-$	$-$	$-$	$-$	0	$+$	$+0$

のようになり，$x = 0$ を横切ってから 0 に収束する動き (オーバーシュート) が出てくる．ところが同じ e^{-2t}, e^{-1} でも係数が

$$x(t) = 2e^{-2t} + e^{-t}$$

の場合は，恒等的に $x(t) > 0$ なので $x = 0$ は横切らない．こうした係数を決めているのは，初期値 $\boldsymbol{x}(0) = \big(x(0), \dot{x}(0)\big)^T$ である．このように，実固有値でも初期値 $\boldsymbol{x}(0)$ の範囲によっては，オーバーシュートが起こる．

これに対して，固有値に虚部があると，ほとんど全ての初期値に対して，必ずオーバーシュートが発生する[5)]．

[5)] 正確にいうと，初期値 $\boldsymbol{x}(0)$ が，複素固有ベクトル $\boldsymbol{v} \pm i\boldsymbol{w}$ の $\boldsymbol{v}, \boldsymbol{w}$ で張られる成分を持てばオーバーシュートが発生する．これは，ほとんどの初期値でそうなることを意味する．

♣ 4 章の補足

● 例題 4.1 p21 の解答例

$$\left|\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} - s\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right| = \begin{vmatrix} 0-s & 1 \\ -2 & -3-s \end{vmatrix} = s^2+3s+2=0$$ より，固有値は $s=-1,-2$．1 つ目の固有値 $s=s_1=-1$ を $A\boldsymbol{v}=s\boldsymbol{v}$ に代入すると，

$$\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} v_2 \\ -2v_1-3v_2 \end{bmatrix} = -\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

より，$v_1+v_2=0$ の関係が得られる．ゆえに固有ベクトルの方向は $\boldsymbol{v}_1=\left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right]$．2 つ目 $s=s_2=-2$ に対しては，

$$\begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} v_2 \\ -2v_1-3v_2 \end{bmatrix} = -2\begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

より，$2v_1+v_2=0$ が得られ，固有ベクトルの方向は $\boldsymbol{v}_2=\begin{bmatrix} 1 \\ -2 \end{bmatrix}$ //

● 実習 4.1 p24 の解答例

Octave で固有値を求めると，

```
octave:1> A2=[0,1;9.8,-1]
A2 =
   0.00000   1.00000
   9.80000  -1.00000
octave:2> eig(A2)
ans =
   2.6702
  -3.6702
```

より，$-3.6702<0<2.6702$ なので，鞍点に判別される．実習 3.3 の実行例でも，中央に向いながら左右に飛される鞍点特有の軌道 (**2.1.2 節** p8) が見てとれる．

● 実習 4.2 p25 の解答例

Octave で固有値を求めると，

```
octave:1> A1=[0,1;-9.8,-1]
A1 =
   0.00000   1.00000
```

```
  -9.80000  -1.00000
octave:2> eig(A1)
ans =
  -0.5000 + 3.0903i
  -0.5000 - 3.0903i
```

より，複素固有値で実部が負なので，安定渦状点に判別される．実習 3.3 の実行例でも，原点に収束する渦巻が見てとれる．この渦巻運動の角振動数は ≈ 3.0903 である．

A4 算法 4.2 p22 の証明

証明の核となるアイデアは，足し引きゼロの変形 $A = sE + (A - sE)$ を使って，行列指数関数 e^A を，

$$e^A = e^{sE+(A-sE)} \overset{\text{算法 3.3}}{\underset{(3)}{=}} e^{sE}e^{(A-sE)} \overset{(*2)}{=} e^s e^{(A-sE)} \tag{*1}$$

のように変形することである．実際，e^{sE} は，指数関数の定義 (3.6) p16 より，

$$e^{sE} = \sum_{k=0}^{\infty} \frac{1}{k!}(sE)^k = \sum_{k=0}^{\infty} \frac{1}{k!}(s^k)E = \underbrace{\left(\sum_{k=0}^{\infty} \frac{s^k}{k!}\right)}_{(3.6)} E = e^s E \tag{*2}$$

となるので，$e^{sE}e^{(A-sE)} = e^s E e^{(A-sE)} = e^s e^{(A-sE)}$ を得る．

次に，(*1) で簡略化した e^A を，固有ベクトルに作用させる．

$$e^A \boldsymbol{v} = e^s \underbrace{e^{(A-sE)}\boldsymbol{v}}_{\bigstar} \tag{*3}$$

e^s は通常の指数関数でスカラだから，問題は ★ である．ここで，第 2 のアイデアとして，固有値の定義式 (4.1) p21 を移項したもの，

$$A\boldsymbol{v} = s\boldsymbol{v} \iff (A - sE)\boldsymbol{v} = \mathbb{O}$$

を用意しておく．この両辺に $(A - sE)$ を乗じても，右辺は $\mathbb{O}$ のままだから，

$$(A - sE)^k \boldsymbol{v} = \mathbb{O} \quad (k = 1, 2, \cdots) \tag{*4}$$

という公式が得られる．この (*4) を使うと，なんと，

$$\begin{aligned} \bigstar = e^{(A-sE)}\boldsymbol{v} &= \left(E + (A - sE) + \frac{1}{2}(A - sE)^2 + \cdots\right)\boldsymbol{v} \\ &= \left(\boldsymbol{v} + \underbrace{(A - sE)\boldsymbol{v}}_{\mathbb{O}} + \frac{1}{2}\underbrace{(A - sE)^2\boldsymbol{v}}_{\mathbb{O}} + \cdots\right) = \boldsymbol{v} \end{aligned}$$

になってしまう．これを (*3) に戻して，算法 4.2 の式，

$$e^A \boldsymbol{v} = e^s \boldsymbol{v}$$

を得る．$A\boldsymbol{v} = s\boldsymbol{v} \implies (At)\boldsymbol{v} = (st)\boldsymbol{v}$ より，$e^{At}\boldsymbol{v} = e^{st}\boldsymbol{v}$ も示せる．証明終り．

5

対角正準形

状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ を簡略化する１つめの方法を述べる．**対角化**という．行列 A を対角化すると，固有値を並べただけの対角行列に書き変わる．

5.1　斜交成分入門

斜めに測ったベクトルの成分を**斜交成分**という．線形変換を，丁度よい斜交成分で表すと，固有値を並べた対角行列が得られる．これが対角化の原理である．

5.1.1　列ベクトル

その前に，行列の便利な表記法を１つ．行列の各列に着目して，

$$A = \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \end{bmatrix} = \begin{bmatrix} \overset{\boldsymbol{u}}{\boxed{\begin{matrix} u_1 \\ u_2 \end{matrix}}} & \overset{\boldsymbol{v}}{\boxed{\begin{matrix} v_1 \\ v_2 \end{matrix}}} \end{bmatrix} \equiv [\boldsymbol{u}, \boldsymbol{v}] \tag{5.1}$$

のような表記を考える．各列を表すベクトル $\boldsymbol{u}, \boldsymbol{v}$ を**列ベクトル**という．

列ベクトルを使った，次の変形をよく使う．

$$x_1\boldsymbol{u} + x_2\boldsymbol{v} = \begin{bmatrix} x_1u_1 + x_2v_1 \\ x_1u_2 + x_2v_2 \end{bmatrix} = \begin{bmatrix} u_1 & v_1 \\ u_2 & v_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = [\boldsymbol{u}, \boldsymbol{v}] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{5.2}$$

5.1.2　直交成分と斜交成分

図 5.1 のように，ベクトル $\boldsymbol{x} = \left[\begin{smallmatrix} 2 \\ 1 \end{smallmatrix}\right]$ の縦横の寸法を考える．当然ながら，横の寸法は $x_1 = 2$，縦の寸法は $x_2 = 1$ である．このように直角に測った $\boldsymbol{x}$ の寸法 $\widetilde{\boldsymbol{x}} = \left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix}\right]$ を，$\boldsymbol{x}$ の**直交成分**という．しかし，寸法の測り方は縦横だけではない．図 5.2 のように斜めに測ると，同じベクトル $\boldsymbol{x}$ でも寸法の値は変化する．このような，$\boldsymbol{x}$ を対角線とする平行四辺形の 2 辺の寸法 $\widetilde{\boldsymbol{x}}' = \left[\begin{smallmatrix} x_1' \\ x_2' \end{smallmatrix}\right]$ を，$\boldsymbol{x}$ の**斜交成分**という．

直交成分を，斜交成分の特殊なケース (平行四辺形が直角) とみると，次のような言い方ができる．すなわち，特殊な成分である直交成分 $\widetilde{\boldsymbol{x}}$ ではたまたま $= \boldsymbol{x}$ となる．しかし，普通の斜交成分 $\widetilde{\boldsymbol{x}}'$ では $\neq \boldsymbol{x}$ が当たり前！

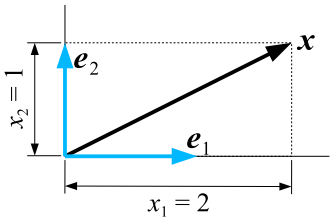


図 5.1 $\boldsymbol{x}$ の直交成分

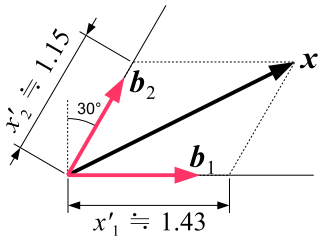


図 5.2 同じ $\boldsymbol{x}$ の斜交成分 (図の寸法は実測値で誤差を含む)

5.1.3 基底変換行列

直交成分 $\boldsymbol{x}$ と斜交成分 $\widetilde{\boldsymbol{x}}'$ の関係を数式表現する．直交成分の特殊性 $\widetilde{\boldsymbol{x}} = \boldsymbol{x}$ を思い出しておく．

さて，図 5.2 の斜交成分の平行四辺形に沿った単位ベクトル $\boldsymbol{b}_1 = \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix}, \boldsymbol{b}_2 = \begin{bmatrix} b_{12} \\ b_{22} \end{bmatrix}$ をとると，ベクトル $\boldsymbol{x}$ は，

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = x'_1\boldsymbol{b}_1 + x'_2\boldsymbol{b}_2 = x'_1 \begin{bmatrix} b_{11} \\ b_{21} \end{bmatrix} + x'_2 \begin{bmatrix} b_{12} \\ b_{22} \end{bmatrix} \tag{5.3}$$

と書ける．この単位ベクトルの組 $\mathcal{B} = \langle \boldsymbol{b}_1, \boldsymbol{b}_2 \rangle$ を，**斜交基底** (oblique basis) という．各 $\boldsymbol{b}_i$ を，**斜交基底ベクトル**という．

ここで，(5.3) の**うまい書き直し方**があって，列ベクトル (5.2) p30 を使うと，

$$\widetilde{\boldsymbol{x}} = \boldsymbol{x} = x'_1\boldsymbol{b}_1 + x'_2\boldsymbol{b}_2 = [\boldsymbol{b}_1, \boldsymbol{b}_2] \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} = [\boldsymbol{b}_1, \boldsymbol{b}_2]\widetilde{\boldsymbol{x}}' \quad \equiv T\widetilde{\boldsymbol{x}}' \tag{5.4}$$

のように，斜交成分 $\widetilde{\boldsymbol{x}}'$ をくくり出せる．以上，次の算法が得られた．

$$\widetilde{\boldsymbol{x}} = T\widetilde{\boldsymbol{x}}' \quad \text{または} \quad \widetilde{\boldsymbol{x}}' = T^{-1}\widetilde{\boldsymbol{x}}, \quad T \equiv [\boldsymbol{b}_1, \boldsymbol{b}_2] \tag{5.5}$$

このように，直交座標 $\widetilde{\boldsymbol{x}}$ と斜交座標 $\boldsymbol{x}'$ の関係は，斜交基底ベクトル $\boldsymbol{b}_1, \boldsymbol{b}_2$ を列とする行列 $T = [\boldsymbol{b}_1, \boldsymbol{b}_2]$ で表せる．行列 T を，**基底変換行列** (change of basis matrix) という．

例題 5.1 図 5.1 の直交成分 $\widetilde{\boldsymbol{x}}$ と，図 5.2 の斜交成分 $\widetilde{\boldsymbol{x}}'$ の関係を，(5.6) で計算し，

図の実測値と比較せよ．図の斜交基底ベクトルは $\boldsymbol{b}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \boldsymbol{b}_2 = \begin{bmatrix} 1/2 \\ \sqrt{3}/2 \end{bmatrix}$ である．

ベクトル $\boldsymbol{x}$ が n 次元の場合は，基底ベクトルを n 個 $\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n$，基底変換行列を n 次の $T = [\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n]$ とすればよい．同様に次の算法が成立する．

算法 5.1　ベクトル $\boldsymbol{x}$ の直交成分を $\widetilde{\boldsymbol{x}}$ とし，斜交基底 $\mathcal{B} = \langle \boldsymbol{b}_1, \cdots, \boldsymbol{b}_n \rangle$ による斜交成分を $\widetilde{\boldsymbol{x}}'$ とする．このとき両者の関係は，次の変換に従う．

$$\widetilde{\boldsymbol{x}} = T\widetilde{\boldsymbol{x}}' \quad \text{または} \quad \widetilde{\boldsymbol{x}}' = T^{-1}\widetilde{\boldsymbol{x}} \qquad (\widetilde{\boldsymbol{x}} = \boldsymbol{x}) \tag{5.6}$$

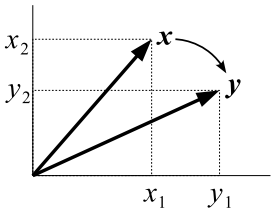
ただし，$T \equiv [\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n]$ は基底変換行列である．

5.1.4　線形変換の斜交成分表示

次のような行列で書ける変換を，**線形変換** (linear transformation) という．

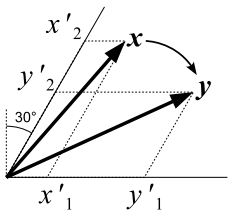
$$\boldsymbol{y} = A\boldsymbol{x} \tag{5.7}$$

変換前後のベクトル $\boldsymbol{x}, \boldsymbol{y}$ の直交成分 $\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}}$ は，直交成分の特殊性より，同じ行列 A で，



$$\widetilde{\boldsymbol{y}} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = A\widetilde{\boldsymbol{x}} \qquad (\widetilde{\boldsymbol{y}} = \boldsymbol{y}, \widetilde{\boldsymbol{x}} = \boldsymbol{x}) \tag{5.8}$$

と動く．ところが，同じ動き $\boldsymbol{x} \to \boldsymbol{y}$ を，斜交成分で表そうとすると，



$$\widetilde{\boldsymbol{y}}' = \begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} a_{11}' & a_{12}' \\ a_{21}' & a_{22}' \end{bmatrix} \begin{bmatrix} x_1' \\ x_2' \end{bmatrix} = A'\widetilde{\boldsymbol{x}}' \qquad (\widetilde{\boldsymbol{y}}' \neq \boldsymbol{y}, \widetilde{\boldsymbol{x}}' \neq \boldsymbol{x}) \tag{5.9}$$

のように，違う行列 A' が必要になる．とはいえ，A' が表すベクトルの動き $\boldsymbol{x} \to \boldsymbol{y}$ は同じである．このような行列 A', A を，互いに**相似** (similar) であるという．

互いに相似な行列 A と A' の関係を求める．算法 5.1 によれば，ベクトル $\boldsymbol{x}, \boldsymbol{y}$ の直交成分と斜交成分の関係は，

$$\boldsymbol{x} = \widetilde{\boldsymbol{x}} = T\widetilde{\boldsymbol{x}}', \quad \boldsymbol{y} = \widetilde{\boldsymbol{y}} = T\widetilde{\boldsymbol{y}}', \quad T = [\boldsymbol{b}_1, \boldsymbol{b}_2] \tag{5.10}$$

と書けた．これを (5.7) に代入すると，

$$T\widetilde{\boldsymbol{y}}' = \boldsymbol{y} = A\boldsymbol{x} = AT\widetilde{\boldsymbol{x}}' \quad \Longrightarrow \quad \widetilde{\boldsymbol{y}}' = T^{-1}AT\widetilde{\boldsymbol{x}}' \equiv A'\widetilde{\boldsymbol{x}}' \tag{5.11}$$

という，斜交成分用の変換則が判明する．これより，A と A' の関係は，$A' = T^{-1}AT$ であることが分かる．

以上の式変形は，行列の次数によらず成立するから，次の算法が得られる．

算法 5.2 n 次元ベクトルの動き $\boldsymbol{x} \to \boldsymbol{y}$ が，n 次行列で，

$$\boldsymbol{y} = A\boldsymbol{x} \qquad (\widetilde{\boldsymbol{y}} = \boldsymbol{y}, \widetilde{\boldsymbol{x}} = \boldsymbol{x})$$

のように表されているとき，同じ動きは，斜交成分では，

$$\widetilde{\boldsymbol{y}}' = A'\widetilde{\boldsymbol{x}}', \qquad A' \equiv T^{-1}AT$$

と書ける．T は算法 5.1 の基底変換行列である．

5.1.5 線形変換の対角化表示

ここで，斜交基底 $\mathcal{B} = \langle \boldsymbol{b}_1, \boldsymbol{b}_2 \rangle$ を，行列 A の固有ベクトル，

$$A\boldsymbol{b}_i = s_i\boldsymbol{b}_i \quad (n = 1, \cdots, n) \tag{5.12}$$

で構成すると，面白いことが起こる．A の斜交成分表示が，

$$\begin{aligned} A' = T^{-1}AT &= [\boldsymbol{b}_1, \boldsymbol{b}_2]^{-1}A[\boldsymbol{b}_1, \boldsymbol{b}_2] = [\boldsymbol{b}_1, \boldsymbol{b}_2]^{-1}[A\boldsymbol{b}_1, A\boldsymbol{b}_2] \\ &= [\boldsymbol{b}_1, \boldsymbol{b}_2]^{-1}[s_1\boldsymbol{b}_1, s_2\boldsymbol{b}_2] \quad \because \text{固有ベクトル} \\ &= \underbrace{[\boldsymbol{b}_1, \boldsymbol{b}_2]^{-1}[\boldsymbol{b}_1, \boldsymbol{b}_2]}_{\text{単位行列 } I} \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \overset{\text{短縮}}{=\!=}_{\text{表記}} \operatorname{diag}\{s_1, s_2\} \end{aligned} \tag{5.13}$$

のように，固有値 s_1, s_2 を並べた対角行列になってしまう．このような行列の変形法を，**対角化** (diagonalization) という．n 次元でも同様に，次の算法が成立する．

算法 5.3 算法 5.2 の斜交基底を，行列 A の固有ベクトルに選ぶと，A の斜交成分表示は，固有値を並べた対角行列になる[1)]．

$$A' = T^{-1}AT = \begin{bmatrix} s_1 & & 0 \\ & \ddots & \\ 0 & & s_n \end{bmatrix} \overset{\text{短縮}}{=\!=}_{\text{表記}} \operatorname{diag}\{s_1, \cdots, s_n\} \tag{5.14}$$

▶▶　固有ベクトル $\boldsymbol{b}_i$ を，わざわざ単位ベクトルに直して計算する必要はない．$\boldsymbol{b}_i$ の大きさは，左からの T^{-1} と右からの T で丁度キャンセルするので，$\boldsymbol{b}_i$ の大きさを変えても計算結果は同じになる．

例題 5.2　例題 4.1 p21 の行列 $A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$ を対角化せよ．

5.2　状態方程式の対角化

5.2.1　対角正準形

算法 5.3 を使うと，n 次元の状態方程式，

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} \tag{5.15}$$

が簡略化できる．まず，行列 A の固有値・固有ベクトル，

$$A\boldsymbol{v}_k = s_k\boldsymbol{v}_k \qquad (k = 1, \cdots, n) \tag{5.16}$$

をとり，固有ベクトルで基底 $\mathcal{V} = \langle \boldsymbol{v}_1, \cdots, \boldsymbol{v}_n \rangle$ を構成する．$\mathcal{V}$ は一般に斜交基底である．この $\mathcal{V}$ による斜交成分 $\boldsymbol{y}$ を使うと，ベクトル $\boldsymbol{x}$ は，算法 5.1 p32 より，

$$\boldsymbol{x} = T\boldsymbol{y}, \qquad T \equiv [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n] \tag{5.17}$$

と書ける．この両辺を時間微分すると，T は定数行列だから，

$$\dot{\boldsymbol{x}} = T\dot{\boldsymbol{y}} \tag{5.18}$$

も得られる．(5.17), (5.18) を (5.15) に代入すると，$T\dot{\boldsymbol{y}} = \dot{\boldsymbol{x}} = A\boldsymbol{x} = AT\boldsymbol{y}$ より $\dot{\boldsymbol{y}} = T^{-1}AT\boldsymbol{y}$ となるが，算法 5.3 より，$T^{-1}AT = \operatorname{diag}\{s_1, \cdots, s_n\}$ となるので，

$$\dot{\boldsymbol{y}} = \begin{bmatrix} s_1 & & 0 \\ & \ddots & \\ 0 & & s_n \end{bmatrix} \boldsymbol{y} \tag{5.19}$$

のような簡略化が得られる．これを，**対角正準形** (diagonal canonical form) という．

5.2.2　モード展開

4.4 節 p26 によれば，状態方程式 (5.15) の解は，(5.16) の固有値・固有ベクトル $A\boldsymbol{v}_i = s_i\boldsymbol{v}_i$ を使って，

$$\boldsymbol{x}(t) = c_1 e^{s_1 t}\boldsymbol{v}_1 + c_2 e^{s_2 t}\boldsymbol{v}_2 + \cdots + c_n e^{s_n t}\boldsymbol{v}_n \tag{5.20}$$

[1)] 固有値に重複がある場合は，算法 5.3 は一般には成立しない．**本書の仮定** p22 参照．

と書けた．実はこれを，**モード展開** (modal expansion) という [2]．固有ベクトル $\boldsymbol{v}_i$ を，**固有モード** (eigenmode)[2] といい，時間変動する係数部分，

$$y_i(t) \equiv c_i e^{s_i t} \qquad (i = 1, \cdots, n) \tag{5.21}$$

を，**モード座標** (modal coordinate) または単に**モード** (mode) という．この $y_i(t)$ を解とする方程式が，実は，対角化した状態方程式 (5.19) なのである．(5.19) を解くと，モード座標 $y_i(t)$ の時間変化が直接求まる．初期値は $\boldsymbol{y}(0) = T^{-1}\boldsymbol{x}(0)$ である．

実際，対角化した状態方程式 (5.19) の成分は，

$$\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}^{\cdot} = \begin{bmatrix} s_1 & & 0 \\ & \ddots & \\ 0 & & s_n \end{bmatrix} \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} s_1 y_1 \\ \vdots \\ s_n y_n \end{bmatrix} \tag{5.22}$$

なので，これは n 個の独立な 1 次元状態方程式，

$$\dot{y}_i = s_i y_i \qquad (i = 1, \cdots, n) \tag{5.23}$$

を表す．その解は (3.7) p16 より，

$$y_i(t) = c_i e^{s_i t}, \quad c_i = y_i(0) \qquad (i = 1, \cdots, n)$$

であり，確かにこれはモード座標 (5.21) である．

実習 5.1　例題 5.2 の行列 A を用いて，状態方程式 $\dot{\boldsymbol{x}} = A\boldsymbol{x}$ の軌道 $\boldsymbol{x}(t)$，モード座標の軌道 $\boldsymbol{y}(t)$，およびモード座標から復元した軌道 $\boldsymbol{x}(t) = y_1(t)\boldsymbol{v}_1 + y_2(t)\boldsymbol{v}_2$ を比較せよ．

5.3　可制御性行列

線形の状態方程式 (5.15) に，人為的な入力 $u(t)$ を追加したもの，

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t) \tag{5.24}$$

を，**線形制御系** (linear control systems) という．入力 $u(t)$ はスカラと仮定する[3]．$\boldsymbol{b}$ はベクトルで，入力 $u(t)$ が伝わる状態方程式の行と，伝わる強度を表す．

►► 制御工学の慣習で，ベクトル $\boldsymbol{b}$ のスカラ $u(t)$ 倍を，数学とは逆順 $\boldsymbol{b}\,u(t)$ に書く．

---

[2] 長さを 1 にしたものを固有モードという場合もある．

[3] これを **1 入力**という．入力がベクトル $\boldsymbol{u}(t)$ のときは**多入力**という．

5.3.1　可制御性の定義

簡単のため 2 次元で考える．(5.24) を対角化すると，$T\dot{\boldsymbol{y}} = AT\boldsymbol{y} + \boldsymbol{b}\,u(t)$ より，

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}^{\cdot} = \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} + \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} u(t), \qquad \boldsymbol{\beta} \equiv T^{-1}\boldsymbol{b} \tag{5.25}$$

のような対角正準形が得られる．成分で書けば，

$$\begin{cases} \dot{y}_1 = s_1 y_1 + \beta_1 u(t) \\ \dot{y}_2 = s_2 y_2 + \beta_2 u(t) \end{cases} \tag{5.26}$$

である．ここで仮に，$\beta_1 = 0$ とすると，

$$\begin{cases} \dot{y}_1 = s_1 y_1 \\ \dot{y}_2 = s_2 y_2 + \beta_2 u(t) \end{cases} \tag{5.27}$$

となって，制御入力 $u(t)$ が，1 番目のモード $y_1(t)$ に伝わらなくなる．同様に $\beta_2 = 0$ とすれば，今度は，2 番目のモード $y_2(t)$ が操作できない．このような操作不可能なモードが存在する状況を，**不可制御** (uncontrollable) という．

逆に，制御入力 $u(t)$ が全てのモード $y_i(t)$ に伝わる状況，

$$\beta_i \neq 0 \qquad (\text{全ての } i) \tag{5.28}$$

を，**可制御** (controllable) という．

5.3.2　可制御性の必要十分条件

与えられた制御系 $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t)$ が，可制御 (5.28) かどうかを判定するには，制御系を対角化しなければならない．こうした面倒を避けて，制御系の構造 $A, \boldsymbol{b}$ から直接判定する方法を考える．

そのためには，$\boldsymbol{\beta}$ に関する条件式を，$A, \boldsymbol{b}$ だけで書き下せればよい．手始めに，$\boldsymbol{b}$ と $\boldsymbol{\beta}$ の関係を探すと，

$$\boldsymbol{b} = T\boldsymbol{\beta} = [\boldsymbol{v}_1, \boldsymbol{v}_2] \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} = \beta_1 \boldsymbol{v}_1 + \beta_2 \boldsymbol{v}_2 \tag{5.29}$$

がある．A との関係も欲しいので，両辺に A を乗じると，$\boldsymbol{v}_i$ は固有ベクトルだから，

$$A\boldsymbol{b} = \beta_1 A\boldsymbol{v}_1 + \beta_2 A\boldsymbol{v}_2 = \beta_1 s_1 \boldsymbol{v}_1 + \beta_2 s_2 \boldsymbol{v}_2 = [\boldsymbol{v}_1, \boldsymbol{v}_2] \begin{bmatrix} \beta_1 s_1 \\ \beta_2 s_2 \end{bmatrix} \tag{5.30}$$

となる．ここで，画期的なアイデアとして，$\boldsymbol{b}, A\boldsymbol{b}$ を列とする行列を作ると，

$$U_c = [\boldsymbol{b}, A\boldsymbol{b}] = \left[[\boldsymbol{v}_1, \boldsymbol{v}_2] \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix},\ [\boldsymbol{v}_1, \boldsymbol{v}_2] \begin{bmatrix} \beta_1 s_1 \\ \beta_2 s_2 \end{bmatrix} \right] = [\boldsymbol{v}_1, \boldsymbol{v}_2] \left[\begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix},\ \begin{bmatrix} \beta_1 s_1 \\ \beta_2 s_2 \end{bmatrix} \right]$$

$$= T \begin{bmatrix} \beta_1 & \beta_1 s_1 \\ \beta_2 & \beta_2 s_2 \end{bmatrix} = T \underbrace{\begin{bmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{bmatrix}}_{B} \underbrace{\begin{bmatrix} 1 & s_1 \\ 1 & s_2 \end{bmatrix}}_{V} \tag{5.31}$$

のような変形が得られる[4]．とどめに，両辺の行列式をとると，

$$|U_c| = |T\,B\,V| = |T| \cdot |B| \cdot |V| \tag{5.32}$$

という積が得られる．まず，T には逆行列があるから，$|T| \neq 0$ がいえる．さらに，$|V| = s_2 - s_1 \neq 0$ も分かる (固有値は重複しない)．ゆえに，$|B| = \beta_1\beta_2$ を介して，

$$\beta_i \neq 0 \text{ (全ての } i) \quad \underset{\text{十分}}{\overset{\text{必要}}{\Longleftrightarrow}} \quad |B| = \beta_1\beta_2 \neq 0 \quad \underset{\text{十分}}{\overset{\text{必要}}{\Longleftrightarrow}} \quad |U_c| = |[\boldsymbol{b}, A\boldsymbol{b}]| \neq 0 \tag{5.33}$$

という，必要十分条件の連鎖が作れる．すなわち，可制御性 (5.28) は，行列 $U_c = [\boldsymbol{b}, A\boldsymbol{b}]$ の行列式が $\neq 0$ であることと等価である．

同様の計算により，状態方程式が 3 次元なら $U_c = [\boldsymbol{b}, A\boldsymbol{b}, A^2\boldsymbol{b}]$，4 次元なら $U_c = [\boldsymbol{b}, A\boldsymbol{b}, A^2\boldsymbol{b}, A^3\boldsymbol{b}]$ という行列を構成して，$|U_c| \neq 0$ ならば可制御と判定される．この行列 U_c を，**可制御性行列** (controllability matrix) という．

なお，線形代数によれば，n 次行列 U_c に関して，次の 2 条件は等価である．

$$|U_c| \neq 0 \quad \overset{\text{等価}}{\Longleftrightarrow} \quad \operatorname{rank} U_c = n \text{ (rank は行列の階数)}$$

標準的な制御理論では，$\operatorname{rank} U_c = n$ のほうが使われる[5]．以上，次の算法が得られた．

算法 5.4 (可制御性の判定則) n 次元線形制御系 $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t)$ が，**可制御** (全てのモードに入力が伝わる) であるための必要十分条件は，**可制御性行列**，

$$U_c \equiv [\boldsymbol{b}, A\boldsymbol{b}, \cdots, A^{n-1}\boldsymbol{b}] \tag{5.34}$$

の階数が n となることである．(すなわち $|U_c| \neq 0$ となること)

►► (可制御性の等価な条件) 詳細は省くが，算法 5.4 の判定則は，次の各条件とそれぞれ等価であることが数学的に証明できる [3].

(1) 任意の初期状態 $\boldsymbol{x}(0)$ を，有限時間内に任意の状態 $\boldsymbol{x}(t)$ に移す入力 $u(t)$ が作れる．

(2) **状態フィードバック** p41: $u(t) = -K\boldsymbol{x}$ により，**閉ループ系** p46: $A - \boldsymbol{b}K$ の固有値を任意に設定可能.

(3) 任意の時刻 t で，$\displaystyle\int_0^t e^{A\tau}\boldsymbol{b}\left(e^{A\tau}\boldsymbol{b}\right)^T d\tau$ が**正定値行列** p66 となる．

---

[4] 小郷 [2] の 97〜101 頁など．

[5] 多入力 $\boldsymbol{u}(t)$ の理論でも同様に $\operatorname{rank} U_c = n$ と書けることによる.

♣ 5 章の補足

● 例題 5.1 p31 の解答例

基底変換行列 $T=[\boldsymbol{b}_1,\boldsymbol{b}_2]=\begin{bmatrix}1 & 1/2\\ 0 & \sqrt{3}/2\end{bmatrix}$ より，$\boldsymbol{x}'=T^{-1}\boldsymbol{x}=$

$\frac{2}{\sqrt{3}}\begin{bmatrix}\sqrt{3}/2 & -1/2\\ 0 & 1\end{bmatrix}\begin{bmatrix}2\\ 1\end{bmatrix}=\begin{bmatrix}2-1/\sqrt{3}\\ 2/\sqrt{3}\end{bmatrix}\approx\begin{bmatrix}1.42\\ 1.16\end{bmatrix}\approx$ 実測値 $\begin{bmatrix}1.43\\ 1.15\end{bmatrix}$ となる．実測値のほうに誤差がある．(手動で測ったので大目にみてね)

● 例題 5.2 p34 の解答例

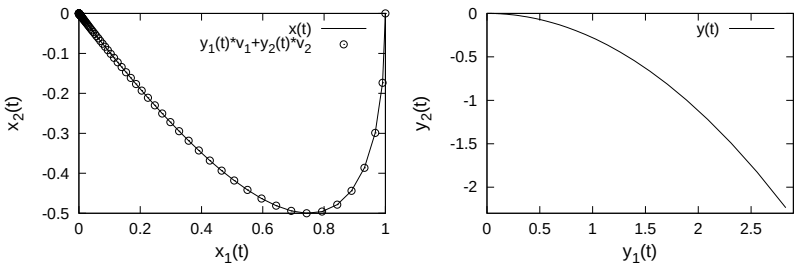
例題 4.1 で求めた固有ベクトルより，基底変換行列を $T=[\boldsymbol{v}_1,\boldsymbol{v}_2]=\begin{bmatrix}1 & 1\\ -1 & -2\end{bmatrix}$ とする．このとき，

$$A'=T^{-1}AT=\begin{bmatrix}2 & 1\\ -1 & -1\end{bmatrix}\begin{bmatrix}0 & 1\\ -2 & -3\end{bmatrix}\begin{bmatrix}1 & 1\\ -1 & -2\end{bmatrix}$$
$$=\begin{bmatrix}-2 & -1\\ 2 & 2\end{bmatrix}\begin{bmatrix}1 & 1\\ -1 & -2\end{bmatrix}=\begin{bmatrix}-1 & 0\\ 0 & -2\end{bmatrix}$$

となり，確かに，例題 4.1 の固有値 $s=-1,-2$ を並べた対角行列が得られる．

● 実習 5.1 p35 の解答例

Code 7 を実行すると，次のような比較結果が得られる．元の状態量 $\boldsymbol{x}(t)$ と，モード座標 $\boldsymbol{y}(t)$ の軌道は異なるが (左右実線)，復元すると元に戻る (左○).



6

可制御正準形

状態方程式を簡略化する 2 つめの方法を述べる．内容的には，**1.2 節** p2 の **1 階化** の焼き直しである．特殊な斜交基底を使うと，1 階化したシステムと同形の行列 A' が得られる．これを応用すると，制御系の固有値を人為的に変更できるようになる．

6.1 状態フィードバック

6.1.1 1 階化の逆操作 — “1 本化”？

単振り子を上死点で線形化した状態方程式 (2.4) p10，※記号を x とした．

$$\dot{\boldsymbol{x}} = A\boldsymbol{x}: \quad \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ k & -c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad = \begin{bmatrix} x_2 \\ kx_1 - cx_2 \end{bmatrix} \tag{6.1}$$

を考える．これは，そもそも 2 階常微分方程式 (1.5) p3 だったものを **1 階化**し，さらにそれを線形化したものだ．

ここで，1 階化の逆操作を試みるが，これが本章で学ぶ「可制御正準形」の中核となる操作である．とりあえず，$x_1 = x$ とおこう．これに方程式の第 1 成分 $\dot{x}_1 = x_2$ を使うと，$x_2 = \dot{x}_1 = \dot{x}$ となるから，2 次元の状態量が単独の変数 x で表せる．

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

このとき，方程式の第 2 成分 $\dot{x}_2 = kx_1 - cx_2$ は，

$$\dot{x}_2 = \ddot{x} \quad = kx - c\dot{x}$$

となり，2 階の常微分方程式，

$$\ddot{x} + c\dot{x} - kx = 0 \tag{6.2}$$

が 1 本得られる．以上，1 階 2 連立の状態方程式が，2 階 1 連立に書き変わり，1 階化の逆操作が完了した．便宜上，この操作を “1 本化” と呼ぼう[1)]．

[1)] 本書でしか通用しない造語なので，ご注意ください．

6.1.2　固有方程式

“1 本化” した状態方程式 $\ddot{x} + c\dot{x} - kx = 0$ の安定性は，$x(t) = e^{st}$ を代入して調べる．$\dot{x}(t) = se^{st}$, $\ddot{x}(t) = s^2e^{st}$ とともに代入すると，

$$0 = \ddot{x} + c\dot{x} - kx = s^2e^{st} + cse^{st} - ke^{st} = (s^2 + cs - k)e^{st} \tag{6.3}$$

となるが，$e^{st} > 0$ より (指数関数のグラフを考えよ)，

$$s^2 + cs - k = 0 \tag{6.4}$$

という 2 次方程式が得られる．これを，**固有方程式**という．同じ固有方程式を 4 章の計算では，状態方程式 (6.1) p39 の行列 A から求めた．

$$0 = |A - sE| = \begin{vmatrix} -s & 0 \\ k & -c - s \end{vmatrix} = s^2 + cs - k \tag{6.5}$$

どちらの方法も，同じ固有方程式に帰着するから，求まる固有値は同じである．

お気付きの読者もいると思うが，状態方程式の “1 本化” と，固有方程式の間には，美しい対応関係がある．

$$\ddot{x} + c\dot{x} - kx = 0 \qquad \text{(6.2) 再掲}$$

$$s^2 + cs - k = 0 \qquad \text{(6.4) 再掲}$$

つまり，“1 本化” した状態方程式の固有方程式は，**同じ係数**の代数方程式になる．一般に，n 階線形常微分方程式の固有方程式は，同じ係数の n 次方程式になる．

6.1.3　状態フィードバック

(6.2) の減衰を $c = 0$ としたものを考える．力学的には，減衰の無い逆立ちした振り子となる．

$$\ddot{x} - kx = 0, \qquad k > 0 \tag{6.6}$$

固有方程式は同じ係数の $s^2 - k = 0$ であるから，固有値は実数で，符号は，

$$-\sqrt{k} < 0 < \sqrt{k} \tag{6.7}$$

である．ゆえに上死点は，**鞍点**であり，不安定と判定される．振り子を支えずに逆さにしたのだから当然だ．さて，この不安定状態を**安定化**する技術を考えよう．そのために，(6.6) の右辺に外力 $u(t)$ を付加する．

$$\ddot{x} - kx = u(t) \tag{6.8}$$

人間が設定できる $u(t)$ を，**制御入力** (control input) という．できないものは，**外乱** (disturbance) と呼ばれる．

上死点を安定化するには，$u(t)$ をうまく設定して，振り子を落そうとする重力を打

ち消せばよい．そうなる制御入力に，

$$u(t) = -K_1 x \tag{6.9}$$

がある．状態量 x に比例する制御入力である．この制御方式を**比例制御** (proportional control) または短く **P 制御**という．(6.8) p40 に代入して移項すると，

$$\ddot{x} - kx = u(t) = -K_1 x \qquad \therefore \ddot{x} - (k - K_1)x = 0 \tag{6.10}$$

となる．このように，P 制御は，仮想的なバネ $K_1 x$ (フックの法則) を付加する効果を持つ．同じ係数の固有方程式 $s^2 - (k - K_1) = 0$ より，固有値は，

$$s = \pm\sqrt{k - K_1} \tag{6.11}$$

へと変化する．ここで，K_1 を十分大きくして $k - K_1 < 0$ としてやれば，固有値は純虚数 $s = \pm i\sqrt{|k - K_1|}$ となるので，振り子は強まった復元力によって**単振動**を始め，落ちなくなる．落ちないが止まらない．

そこでさらに，状態量の微分 $\dot{x}$ に比例する力を追加してやる．

$$u(t) = -K_1 x - K_2 \dot{x} \tag{6.12}$$

追加した $-K_2\dot{x}$ を，**微分制御** (derivative control) または **D 制御**という．**P** と **D** を合せた (6.12) を **PD 制御**という．これを (6.8) p40 に代入して移項すると，

$$\ddot{x} - kx = u(t) = -K_1 x - K_2 \dot{x} \qquad \therefore \ddot{x} + K_2\dot{x} - (k - K_1)x = 0 \tag{6.13}$$

となる．このように，D 制御は，仮想的な減衰器[2)] $K_2\dot{x}$ として働く．同じ係数の固有方程式 $s^2 + K_2 s - (k - K_1) = 0$ より，固有値は，

$$s = \frac{-K_2 \pm \sqrt{K_2^2 + 4(k - K_1)}}{2} \tag{6.14}$$

となる．ここで，K_1, K_2 を上手く調整して，$s = \text{負} \pm \sqrt{\text{負}}$ としてやれば**減衰振動**が得られる．あるいは，$\text{負} - \sqrt{\text{正}} < \text{負} + \sqrt{\text{正}} < 0$ に設定すれば，**振動のない**減衰が得られる．いずれにしても，振り子の倒れ角 x は 0 に収束し，振り子は上死点での静止状態に向う．以上，逆立ちした振り子は PD 制御で安定化できる．

PD 制御の x と $\dot{x}$ は，そもそもの状態方程式 (6.1) p39 の状態量 $\boldsymbol{x}$ の成分なので，次のように一般化できる．

$$u(t) = -K_1 x - K_2 \dot{x} \quad = -\begin{bmatrix} K_1, K_2 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \equiv -K\boldsymbol{x} \quad (\boldsymbol{x} \text{ は状態量}) \tag{6.15}$$

この形の制御方式を**状態フィードバック** (state feedback) という．比例定数を並べた横ベクトル K は**フィードバックゲイン** (feedback gain) または単に**ゲイン**と呼ばれる．フィードバックとは，入力 $u(t)$ の結果である出力 $\boldsymbol{x}(t)$ の情報を，再び入力 $u(t)$ に戻す操作のことである．PD 制御は状態フィードバックの特別な場合であり，状態量 $\boldsymbol{x}$ の成分が比例と微分に区分できるものが PD 制御である．

[2)] 減衰器 (ダンパー) とは，速度 $\dot{x}$ に比例した抵抗力を発生する装置のこと．自動車のサスペンションを覗くと，工業製品としての減衰器が見える．

6.1.4　固有値設定問題

PD 制御を受ける単振り子 (6.13),

$$\ddot{x} + K_2\dot{x} - (k - K_1)x = 0 \tag{6.16}$$

を考える．このシステムがとるべき固有値 s_1, s_2 を先に定めておいて，そうなるゲイン K_1, K_2 を逆算する操作を，**固有値設定** (eigenvalue placement) という[3)]．

原理は簡単で，固有値 s_1, s_2 を持つ固有方程式,

$$(s - s_1)(s - s_2) = s^2 - (s_1 + s_2)s + s_1 s_2 = 0 \tag{6.17}$$

を用意する．これと，固有値を設定したいシステムの固有方程式,

$$s^2 + K_2 s - (k - K_1) = 0 \tag{6.18}$$

を見比べればよい．係数の比較により,

$$K_1 = k + s_1 s_2, \quad K_2 = -(s_1 + s_2) \tag{6.19}$$

という公式が得られる．これで固有値の設定は完了である．

例題 6.1　固有値が $s_1, s_2 = -1 \pm 2i$ (減衰振動) となるような，ゲイン K_1, K_2 を求めよ．$k = 2$ とせよ．

▶ **解答例**　$K_1 = 2+(-1+2i)(-1-2i) = 2+1+4 = 7$, $K_2 = -\{(-1+2i)+(-1-2i)\} = 2$. `K1=7; K2=2; k=2; roots([1,K2,-(k-K1)])` とすれば，Octave/Scilab で検算できる．

6.2　可制御正準形

前節と同じことを，一般の n 次元線形制御系,

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t) \tag{6.20}$$

で行いたい．それにはまず，“1 本化” しなければならない．

6.2.1　可制御正準形

n 次元線形制御系を “1 本化” するには，次のような A_c と $\boldsymbol{b}_c$ だと都合がよい．

$$\dot{\boldsymbol{y}} = A_c\boldsymbol{y} + \boldsymbol{b}_c\,u(t)\,:$$

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \vdots \\ \dot{y}_{n-1} \\ \dot{y}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_1 & -a_2 & -a_3 & \cdots & -a_n \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(t) \tag{6.21}$$

[3)] **極設定**，**極配置** とも呼ばれる．

この形式の状態方程式を，**可制御正準形** (controllable canonical form) という．行列 A_c を，**コンパニオン行列** (companion matrix) という[4]．

なぜ都合がよいかだが，$y_1 = y$ とおくと，状態方程式の各行により，順次,

$$y_2 = \dot{y}_1 = \dot{y}, \quad y_3 = \dot{y}_2 = \ddot{y}, \quad \cdots, \quad y_n = \dot{y}_{n-1} = y^{(n-1)} \quad \left(y^{(n-1)} \equiv \frac{d^{n-1}y}{dt^{n-1}} \right)$$

となってくれる．これらを最終行に代入すると，

$$\dot{y}_n = y^{(n)} = -a_1 y - a_2 \dot{y} - \cdots - a_n y^{(n-1)} + u(t)$$

となり，制御入力付きの “1 本化”，

$$y^{(n)} + a_n y^{(n-1)} + \cdots + a_2 \dot{y} + a_1 y = u(t) \tag{6.22}$$

が簡単に求まる．これが可制御正準形の大きな特徴である．

6.2.2 可制御正準形の求め方

実は，一般の線形制御系 (6.20) についても，もしそれが**可制御** p37 ならば，適当な斜交成分 $\boldsymbol{y} = T_c^{-1}\boldsymbol{x}$ が存在して，可制御正準形に変換できる．問題は，そのための基底変換行列 T_c^{-1} の求め方だ．

まず，元の A と $\boldsymbol{b}$ の成分を使って，**可制御性行列** (5.34) p37,

$$U_c = [\boldsymbol{b}, A\boldsymbol{b}, \cdots, A^{n-1}\boldsymbol{b}] \tag{6.23}$$

を作る．同じく，A の固有方程式,

$$|sE - A| = s^n + a_n s^{n-1} + \cdots + a_2 s + a_1 \tag{6.24}$$

の係数 $a_1, \cdots, a_n$ を求めて，表 6.1 のルールで次の行列を作る[5]．

$$W = [w_{ij}] = \begin{bmatrix} a_2 & a_3 & \cdots & a_n & 1 \\ a_3 & \iddots & a_n & 1 & \\ \vdots & a_n & 1 & & \\ a_n & 1 & & & \\ 1 & & & \mathbf{0} & \end{bmatrix} \tag{6.25}$$

以上の U_c，W から基底変換行列,

表 6.1 行列 W の成分 w_{ij} の決め方

$i+j$	2	3	$\cdots$	n	$n+1$	それ以上
w_{ij}	a_2	a_3	$\cdots$	a_n	$a_{n+1} = 1$	0

[4] 小郷 [2] の 101〜105 頁など．

[5] 名前が見つからない．

$$T_c = U_c W \tag{6.26}$$

を作って，斜交成分 $\boldsymbol{y} = T_c^{-1}\boldsymbol{x}$ をとり，これで制御系 (6.20) p42 を表示すると，

$$\dot{\boldsymbol{y}} = A_c\boldsymbol{y} + \boldsymbol{b}_c\, u(t), \quad A_c = T_c^{-1}AT_c, \quad \boldsymbol{b}_c = T_c^{-1}\boldsymbol{b} \tag{6.27}$$

となるが，これがなんと，可制御正準形 (6.21) p42 の形になるのだ．すなわち，次の算法が得られる．

算法 6.1 (可制御正準形)　可制御な n 次元線形制御系 $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t)$ を，斜交成分，

$$\boldsymbol{y} = T_c^{-1}\boldsymbol{x}, \quad T_c \equiv U_c W$$

で表示したもの，

$$\dot{\boldsymbol{y}} = A_c\boldsymbol{y} + \boldsymbol{b}_c\, u(t), \quad A_c = T_c^{-1}AT_c, \quad \boldsymbol{b}_c = T_c^{-1}\boldsymbol{b}$$

は，可制御正準形 (6.21) p42 となる．U_c は**可制御性行列** (5.34) p37，W は (6.25) p43 で定まる行列である．

実習 6.1　**Code 9** を実行せよ．乱数で無作為に作った $A, \boldsymbol{b}$ について，対応する可制御正準形の $A_c, \boldsymbol{b}_c$ が求まる．

▶▶ (可制御正準形のための斜交基底)　ちなみに，(6.26) の変換行列 T_c の各列は，$\boldsymbol{v}_k = \left(\sum_{i=1}^{n-k+1} a_{k+i}A^{i-1}\right)\boldsymbol{b}$ となる．それらからなる斜交基底 $\mathcal{V} = \langle \boldsymbol{v}_1, \cdots, \boldsymbol{v}_n \rangle$ で，元の状態量をモード展開 $\boldsymbol{x}(t) = y_1(t)\boldsymbol{v}_1 + \cdots + y_n(t)\boldsymbol{v}_n$ したときの係数 $y_i(t)$ が，可制御正準形の解になっている．

6.3　固有値設定問題

可制御正準形を応用して，固有値設定の一般論を構成する．n 次元の線形制御系，

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t) \tag{6.28}$$

を考える．$\boldsymbol{x}, \boldsymbol{b}$ は n 次元ベクトル，A は n 次正方行列，$u(t)$ はスカラである．

6.3.1　固有値設定の一般論

図 6.1 の流れに沿って固有値設定を進める．まず，算法 6.1 で (6.28) を可制御正準形 $\dot{\boldsymbol{y}} = A_c\boldsymbol{y} + \boldsymbol{b}_c\,u(t)$ に書き直し，これに状態フィードバック，

$$u(t) = -K\boldsymbol{y} = -K_1y_1 - K_2y_2 - \cdots - K_ny_n \tag{6.29}$$

を加える．$y_1 = y$, $y_2 = \dot{y}$, $\cdots$, $y_n = y^{(n-1)}$ として “1 本化” すると，

$$y^{(n)} + a_n y^{(n-1)} + \cdots + a_2\dot{y} + a_1 y = u(t) = -\left(K_1 y + K_2\dot{y} + \cdots + K_n y^{(n-1)}\right)$$

となり，移項して整理すると，

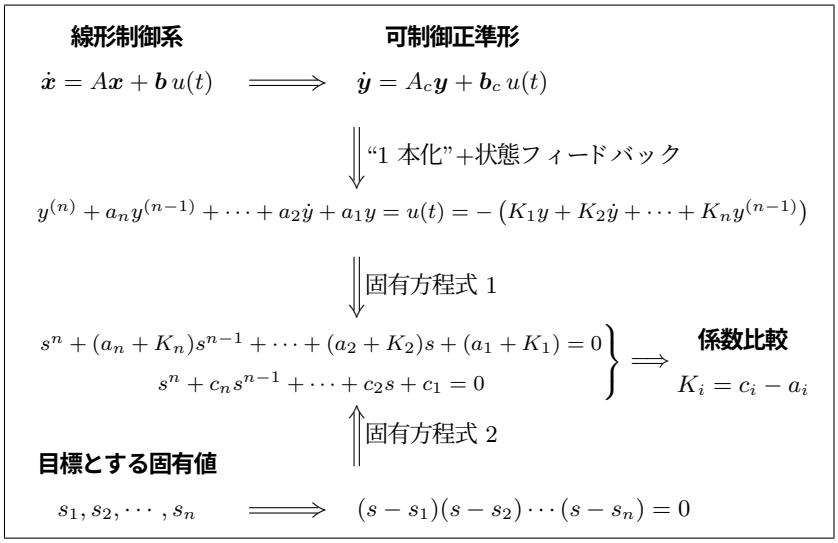


図 6.1 固有値設定の流れ

$$y^{(n)} + (a_n + K_n)y^{(n-1)} + \cdots + (a_2 + K_2)\dot{y} + (a_1 + K_1)y = 0 \tag{6.30}$$

を得る．固有方程式は同じ係数の n 次方程式,

$$s^n + (a_n + K_n)s^{n-1} + \cdots + (a_2 + K_2)s + (a_1 + K_1) = 0 \tag{6.31}$$

である．このように，可制御正準形に状態フィードバックを加えると，固有方程式の係数 $(a_i + K_i)$ を，ゲイン K_i の調整で操作できる．

他方，望みの固有値 $s_1, \cdots, s_n$ を根とする n 次方程式,

$$(s - s_1)(s - s_2)\cdots(s - s_n) = 0$$

を作り，これを展開して，目標の固有方程式,

$$s^n + c_n s^{n-1} + \cdots + c_2 s + c_1 = 0 \tag{6.32}$$

を用意する．

ここで，固有方程式 (6.31) の係数 $(a_i + K_i)$ が，目標の係数 c_i となるように，ゲイン K_i を調整する．

$$a_i + K_i = c_i \qquad \therefore K_i = c_i - a_i \tag{6.33}$$

このとき，$u(t) = -K\boldsymbol{y}$ を受ける可制御正準形の固有値は，望みの値 $s_1, \cdots, s_n$ に調整される．以上で**固有値設定**は完了である．

最後に，斜交成分 $\boldsymbol{y} = T_c^{-1}\boldsymbol{x}$ に対するフィードバック則を，元の状態量 $\boldsymbol{x}$ で表すと，$u(t) = -K\boldsymbol{y} = -KT_c^{-1}\boldsymbol{x}$ より,

$$u(t) = -K_p\boldsymbol{x}, \quad K_p \equiv (c_1 - a_1, \cdots, c_n - a_n)T_c^{-1} \tag{6.34}$$

というフィードバック則が判明する．以上，次の算法が得られた.

算法 6.2 (固有値設定)　線形制御系 $\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t)$ が可制御ならば，その固有値は状態フィードバック，

$$u(t) = -K_p\boldsymbol{x}, \quad K_p \equiv (c_1 - a_1, \cdots, c_n - a_n)T_c^{-1}$$

によって自由に設定できる．a_i は A の固有方程式，

$$|A - sE| = s^n + a_n s^{n-1} + a_{n-1}s^{n-2} + \cdots + a_2 s + a_1 = 0$$

の係数，c_i は目標とする固有値 $s_1, \cdots, s_n$ を解とする固有方程式，

$$(s - s_1)(s - s_2)\cdots(s - s_n) = s^n + c_n s^{n-1} + c_{n-1}s^{n-2} + \cdots + c_2 s + c_1 = 0$$

の係数であり，T_c は算法 6.1 p44 の基底変換行列である．

6.3.2　閉ループ系の固有値

算法 6.2 で設定した固有値を検算するには，元の制御系に状態フィードバックを代入して整理した，

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t) = A\boldsymbol{x} - \boldsymbol{b}\,K_p\boldsymbol{x} = (A - \boldsymbol{b}\,K_p)\boldsymbol{x} \tag{6.35}$$

を考えればよい．得られた $\dot{\boldsymbol{x}} = (A - \boldsymbol{b}\,K_p)\boldsymbol{x}$ を，**閉ループ系** (closed-loop system) という．算法 6.2 による計算に間違えがなければ，行列 $(A - \boldsymbol{b}\,K_p)$ の固有値は，目標の固有値に一致する．ちなみに，縦ベクトル $\boldsymbol{b}$ と横ベクトル K_p の積は，次のように計算するとつじつまが合う．

$$\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} (K_1, K_2, \cdots, K_n) \equiv \begin{bmatrix} b_1K_1 & b_1K_2 & \cdots & b_1K_n \\ b_2K_1 & b_2K_2 & \cdots & b_2K_n \\ \vdots & \vdots & \ddots & \vdots \\ b_nK_1 & b_nK_2 & \cdots & b_nK_n \end{bmatrix} \tag{6.36}$$

この演算を，(ベクトルの) **直積** (outer product) という[6)]．

閉ループとは，閉曲線のような閉じた構造を表す用語である．状態フィードバックを代入した線形制御系は，

$$\dot{\boldsymbol{x}} = (A - \boldsymbol{b}\,K_p)\boldsymbol{x}$$

となって，外的な変数 $u(t)$ が無くなる．これは，制御系が「自律化」したことを意味する．この構造が安定なら，外からの追加情報「$u(t)$」無しに，自分 $\boldsymbol{x}$ だけで安定が保てる．このような自分自身に「閉じた」構造を，**閉ループ**構造という．状態フィードバックは，こうした閉ループ構造を実現するための代表的な技術である．

実習 6.2　**Code 10** を実行せよ．乱数で無作為に作った $A, \boldsymbol{b}$ について，目標の固有値を実現するゲイン K_p が求まり，そのときの時間応答が表示される．

---

6)「テンソル積」ともいう．

♣ 6 章の補足

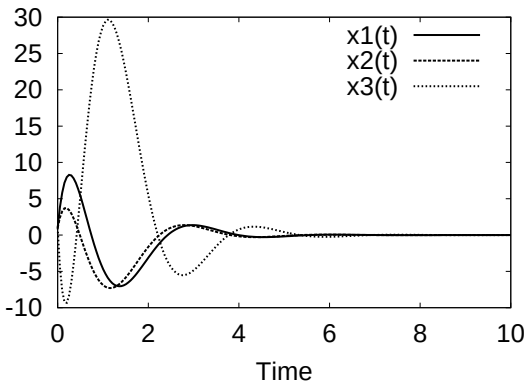
● 実習 6.1 p44 の解答例

例えば，次のような出力が得られる．乱数で $A, \boldsymbol{b}$ を生成するため，数値は実行のたびに変化する．-0.00000 は計算機誤差によるもので 0 を意味する．

```
A =
   0.418497   1.287161  -0.328318
   0.415787  -0.977187   0.892458
  -0.056767  -0.272135  -1.011646
bb =
   0.14321
   2.01271
   1.15817
aa =
  -1.04692  -0.15471   1.57034
W =
  -0.15471   1.57034   1.00000
   1.57034   1.00000   0.00000
   1.00000   0.00000   0.00000
Tc =
   3.935885   2.495248   0.143207
  -1.427317   2.286994   2.012710
  -1.035453   0.091197   1.158165
Ac =
   0.00000   1.00000   0.00000
   0.00000  -0.00000   1.00000
   1.04692   0.15471  -1.57034
bbc =
   0
   0
   1
```

● 実習 6.2 p46 の解答例

次のような出力が得られる．乱数で状態方程式を生成するため，時間応答は実行のたびに変化する．



7

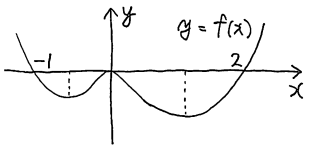
ラグランジュの未定乗数法

最適制御という制御法がある．人為的な評価指標 (省エネとか乗り心地とか) を最小または最大とするような，制御入力 $u(t)$ を作る方法である．数学的には，拘束条件付き変分問題というものに帰着するのだが，その前段階の，関数の最小化から入門する．

7.1 関数の最小化 — 微分法

7.1.1 最小化の必要条件

4 次曲線 $y = f(x) = x^2(x+1)(x-2)$ を考える．



この曲線の高さが最小となる点 $x = a$ を探すには，微分 0 の点を求めればよい．

$$f'(x) = 2x(x+1)(x-2) + x^2(x-2) + x^2(x+1) = 4x^3 - 3x^2 - 4x = 0 \quad (7.1)$$

Octave/Scilab 等で解くと[1]，$a \approx -0.693,\ 0,\ 1.443$ のような候補が判明する．正解は 1.443 である．真ん中の 0 については，追加の条件「最小値のまわりで，グラフは下に凸」で退場とできる．しかし，両端 $-0.693,\ 1.443$ のどちらが最小値を与えるかは，$f(a)$ の値を見ないと判らない．

こうした最小値と微分 0 の因果関係は，

- $f(x)$ が $x = a$ で最小値をとる $\overset{\text{ならば}}{\Longrightarrow}$ 微分 $f'(a) = 0$

のように整理できる．一般に，因果関係 $P \implies Q$ が成立するとき，

- P を「Q であるための**十分条件**」．Q を「P であるための**必要条件**」．

---

[1] いずれもコマンドは `roots([4,-3,-4,0])`

という[2]．したがって，以上の議論は，次のような文章で表現できる．

算法 7.1 関数 $f(x)$ が $x=a$ で最小となるための**必要条件**は，$f'(a)=0$ である．

必要条件「$f'(a)=0$」の泣き所として，必要条件を満す $x=a$ が必ずしも最小点とはならない．しかしながら，最小化の対象が複雑になると，必要条件だけでも求まれば御の字で，必要条件が唯一の手掛かりという状況も増えてくる．

7.1.2 多変数関数の場合

独立変数を増やして $\boldsymbol{x}=(x,y)$ とし，2 次曲面の高さ $z=f(\boldsymbol{x})=f(x,y)=x^2+y^2$ を最小化する問題を考える．これを，本書では，

$$\min_{(x,y)} : f(x,y) \quad \text{または} \quad \min_{\boldsymbol{x}} : f(\boldsymbol{x}) \tag{7.2}$$

と表記する．最小値を与える $\boldsymbol{x}$ を $\boldsymbol{a}=(a,b)$ と書いておく．これを**最小点**という．

結論からいうと，曲面 $f(x,y)$ が (a,b) で最小となるための必要条件は，

$$\frac{\partial f}{\partial x}(a,b)=\frac{\partial f}{\partial y}(a,b)=0 \tag{7.3}$$

である．

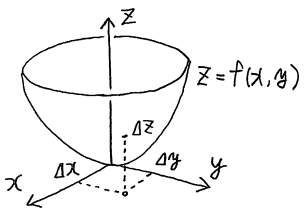
►► **(偏微分)** 多変数関数 $f(\boldsymbol{x})=f(x_1,\cdots,x_n)$ の x_i 以外を定数とみて，x_i で微分したものを $\frac{\partial f}{\partial x_i}$ と書き，x_i に関する f の**偏微分係数**という．これに特定の点 $\boldsymbol{x}=\boldsymbol{a}=(a_1,\cdots,a_n)$ を代入したものを，

$$\frac{\partial f}{\partial x_i}\Big|_{\boldsymbol{x}=\boldsymbol{a}}, \quad \frac{\partial f}{\partial x_i}(\boldsymbol{a}), \quad \frac{\partial f}{\partial x_i}(a_1,\cdots,a_n)$$

などと書く．

必要条件 (7.3) の原理は簡単だ．独立変数を最小点 (a,b) から微少量 $\Delta x,\Delta y$ だけずらしたときの高さの増分，

$$\Delta z \equiv f(a+\Delta x, b+\Delta y)-f(a,b) \tag{7.4}$$



[2]「物事は十分なほうから必要なほうに流れる」と覚えるとよいらしい．

を 0 にすればよい．このとき，曲面は平ら (傾斜 0) となり，1 変数関数の微分 $f'(a) = 0$ と同等な状況が表せる．

冒頭の 2 次曲面で計算すると，

$$\begin{aligned}\Delta z &= (a+\Delta x)^2 + (b+\Delta y)^2 - a^2 - b^2 \\ &= (2a)\Delta x + (2b)\Delta y + \underline{(\Delta x^2 + \Delta y^2)} \\ &= \frac{\partial f}{\partial x}(a,b)\,\Delta x + \frac{\partial f}{\partial y}(a,b)\,\Delta y + \underline{(\Delta x^2 + \Delta y^2)}\end{aligned}$$

だが，$\Delta x, \Delta y$ が十分小さければ，下線部は 0 とみなせるので，高さの増分は，

$$\Delta z = \frac{\partial f}{\partial x}(a,b)\,\Delta x + \frac{\partial f}{\partial y}(a,b)\,\Delta y \tag{7.5}$$

と書ける．この形式の増分を，$f(x,y)$ の**全微分** (total differential) という．

ここで，$\Delta x, \Delta y$ は微小ながら 0 ではない．ぴったり 0 だと，ずらしたことにならない．したがって，(7.5) の増分 Δz が 0 となるために，

$$\frac{\partial f}{\partial x}(a,b) = \frac{\partial f}{\partial y}(a,b) = 0 \tag{7.3 再掲}$$

が要請される．n 変数関数で同様に考えると，次の算法が得られる．

算法 7.2　関数 $f(x_1, \cdots, x_n)$ が，$\boldsymbol{a} = (a_1, \cdots, a_n)$ で最小となるための**必要条件**は，

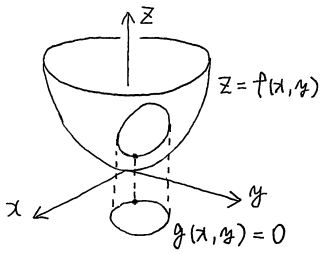
$$\frac{\partial f}{\partial x_1}(\boldsymbol{a}) = \frac{\partial f}{\partial x_2}(\boldsymbol{a}) = \cdots = \frac{\partial f}{\partial x_n}(\boldsymbol{a}) = 0 \tag{7.6}$$

である．

7.2　ラグランジュの未定乗数法

7.2.1　拘束条件付きの最小化

独立変数 (x,y) に制約があり，xy 平面を自由に動けない場合を考える．



制約を表す条件式 $g(x,y) = 0$ を，**拘束条件** (constraint condition) という[3]．こうした制約下で，関数 $f(x,y)$ を最小化する問題を考えよう (上図でいうと，曲面に描かれ

[3] **付帯条件**ともいう．

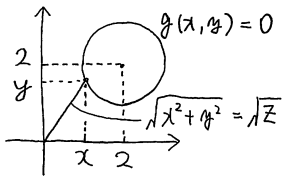
た曲線上で最小点を探す問題)．このような問題を，本書では，次のように表記する[4)]．

$$\min_{(x,y)} : f(x,y), \quad \text{sub.to} : g(x,y) = 0 \tag{7.7}$$

7.2.2 代入による解法

具体例として，2 次曲面 $f(x,y) = x^2 + y^2$ の最小化問題に，拘束条件，

$$g(x,y) = (x-2)^2 + (y-2)^2 - 1 = 0 \quad \text{※中心 } (2,2) \text{ の単位円} \tag{7.8}$$



を課してみる．図のように，この最小化問題は，単位円周上の (x,y) の組合せの中で，原点からの距離の 2 乗 $z = f(x,y) = x^2 + y^2$ を最小化する問題，すなわち原点からの距離 $\sqrt{z}$ を最小化する問題と等価である．

拘束条件 $g(x,y) = 0$ は，y について解けるので，

$$y = 2 \pm \sqrt{1 - (x-2)^2} \tag{7.9}$$

これを，最小化したい $f(x,y)$ に代入すると，

$$f(x,y) = x^2 + \left\{ 2 \pm \sqrt{1-(x-2)^2} \right\}^2 \quad \equiv f(x) \tag{7.10}$$

となり，1 変数関数 $f(x)$ の最小化に帰着するので，必要条件 $f'(a) = 0$ によって，最小点の候補が出せる．

しかし，平方根 $\sqrt{\cdots}$ を含む代数方程式 (7.10) を解くのは，いかにも面倒だ．このようなときに次の方法が役にたつ．

7.2.3 ラグランジュの未定乗数法

実は，拘束条件付きの最小化問題，

$$\min_{(x,y)} : f(x,y), \quad \text{sub.to} : g(x,y) = 0 \tag{7.11}$$

は，新たな独立変数 λ を導入することで，拘束条件無しの最小化問題，

$$\min_{(x,y,\lambda)} : h(x,y,\lambda), \quad h(x,y,\lambda) \equiv f(x,y) + \lambda g(x,y) \tag{7.12}$$

に書き直せる．この書き直し法を，**ラグランジュの未定乗数法**という．λ を**ラグランジュ乗数** (Lagrange multiplier) という．

2 次曲面の例題に適用すると，新たに最小化すべき関数は，

[4)] sub.to は「… subjected to ～」 $\Longleftrightarrow$ ～が課された…の略.

$$h(x,y,\lambda) \equiv f(x,y) + \lambda g(x,y) = x^2 + y^2 + \lambda\left\{(x-2)^2 + (y-2)^2 - 1\right\} \quad (7.13)$$

である．この 3 変数関数が最小化する必要条件は，算法 7.2 p50 より，

$$0 = \frac{\partial h}{\partial x} = 2x + 2\lambda(x-2) \quad \therefore x = \frac{2\lambda}{1+\lambda}$$
$$0 = \frac{\partial h}{\partial y} = 2y + 2\lambda(y-2) \quad \therefore y = \frac{2\lambda}{1+\lambda}$$
$$0 = \frac{\partial h}{\partial \lambda} = (x-2)^2 + (y-2)^2 - 1$$

となる．第 1〜2 式の x, y を第 3 式に代入すると，$x - 2 = \frac{2\lambda}{1+\lambda} - 2 = \frac{-2}{1+\lambda}$ 等より，

$$0 = (x-2)^2 + (y-2)^2 - 1 = \frac{4}{(1+\lambda)^2} + \frac{4}{(1+\lambda)^2} - 1$$
$$\Longrightarrow (1+\lambda)^2 = 8 \quad \therefore \lambda = -1 \pm 2\sqrt{2} \quad (7.14)$$

が早々に判明する．これを x, y に代入すると，

$$x = y = \frac{2\lambda}{1+\lambda} = \frac{\not{2}\lambda}{\pm\ \not{2}\sqrt{2}} = \frac{-1 \pm 2\sqrt{2}}{\pm\sqrt{2}} = 2 \mp \frac{1}{\sqrt{2}} \quad (7.15)$$

が得られる．したがって，この問題の最小点の候補は，

$$(x,y,\lambda) = \left(2 \mp \frac{1}{\sqrt{2}},\ 2 \mp \frac{1}{\sqrt{2}},\ -1 \pm 2\sqrt{2}\right) \quad (7.16)$$

である．便宜上の変数 λ を除外して考えると，最小点の候補は，

$$(x,y) = \underline{\left(2 - \frac{1}{\sqrt{2}},\ 2 - \frac{1}{\sqrt{2}}\right)},\quad \left(2 + \frac{1}{\sqrt{2}},\ 2 + \frac{1}{\sqrt{2}}\right) \quad (7.17)$$

である．冒頭の図でいうと，これらの点は，原点 $(0,0)$ と単位円の中心 $(2,2)$ を結ぶ直線と，単位円周との交点である．このうちの近い方 (下線部) が，最小点となる．

このように，ラグランジュの未定乗数法は，独立変数について拘束条件 $g(x,y) = 0$ が簡単に解けないときに，威力を発揮する．

7.3　多次元の場合

以上の算法は，n 変数関数 $f(\boldsymbol{x}) = f(x_1, \cdots, x_n)$ と，m 本の拘束条件式，

$$\boldsymbol{g}(\boldsymbol{x}) = \mathbb{O}: \quad \begin{bmatrix} g_1(x_1, \cdots, x_n) \\ \vdots \\ g_m(x_1, \cdots, x_n) \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$$

に対して，次のように一般化される．

算法 7.3 拘束条件付きの最小化問題,

$$\min_{\boldsymbol{x}} : f(\boldsymbol{x}), \quad \text{sub.to} : \boldsymbol{g}(\boldsymbol{x}) = \mathbb{O} \tag{7.18}$$

を考える．$\boldsymbol{x}$ は n 次元，$\boldsymbol{g}$ は m 次元とする．この問題は，新たな $n+m$ 変数関数

$$h(\boldsymbol{x}, \boldsymbol{\lambda}) \equiv f(\boldsymbol{x}) + \lambda_1 g_1(\boldsymbol{x}) + \cdots + \lambda_m g_m(\boldsymbol{x}) \ = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$$

の最小化問題 (拘束条件無し)，

$$\min_{(\boldsymbol{x}, \boldsymbol{\lambda})} : h(\boldsymbol{x}, \boldsymbol{\lambda}) \tag{7.19}$$

と等価である．その最小点 $\boldsymbol{x} = \boldsymbol{a}$, $\boldsymbol{\lambda} = \boldsymbol{b}$ の必要条件は，算法 7.2 p50 より，

$$\frac{\partial h}{\partial x_i}(\boldsymbol{a}, \boldsymbol{b}) = 0 \quad (i = 1, \cdots, n), \qquad \frac{\partial h}{\partial \lambda_j}(\boldsymbol{a}, \boldsymbol{b}) = 0 \quad (j = 1, \cdots, m) \tag{7.20}$$

で与えられる．m 次元ベクトル $\boldsymbol{\lambda}$ を，ラグランジュ乗数 (ベクトル) という．

►► $\boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x}) = (\lambda_1, \cdots, \lambda_m) \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix} = \lambda_1 g_1(\boldsymbol{x}) + \cdots + \lambda_m g_m(\boldsymbol{x})$

8

最適制御入門

人為的な評価指標 (省エネとか乗り心地) を最小化 (または最大化) する制御方法を，**最適制御** (optimal control) という．ここでは，そうなる制御入力 $u(t)$ の見付け方を考える．

8.1 最適制御問題

最適制御問題 (optimal control problem) とは，与えられた**評価関数** (cost function)，

$$\text{例えば} \quad J = \int_{t_0}^{t_1} qx(\tau)^2 + ru(\tau)^2 d\tau \tag{8.1}$$

を最小化するように，制御系，

$$\text{例えば} \quad \dot{x}(t) = ax(t) + bu(t) \tag{8.2}$$

の制御入力 $u(t)$ を定めよ，という問題である．q, r は評価関数の性質を決める係数で，**重み** (weight) と呼ばれる．前章の表記で，この問題を記述すれば，

$$\min_{(x,u)} : J, \quad \text{sub.to} : \dot{x}(t) = ax(t) + bu(t) \tag{8.3}$$

となる．J の最小化を**最適化**と呼び，最小化を実現する制御入力 $u(t) = u^*(t)$ を**最適入力**という．前章との大きな違いは，

- 最小化すべき関数 J の値が，時間波形 $x(t), u(t)$ で決まること．
- 拘束条件が微分方程式であること．

の 2 点である．これらを解決すれば，最適制御問題は解ける．

最適制御問題は，境界条件 $x(t_0)$, $x(t_1)$ の指定方法によって，様々なバリエーションがとれる．本書では，代表的な 2 種類の境界条件を考える．

(a) 両端固定　積分区間 $\int_{t_0}^{t_1}$ の両端で，状態量が指定される境界条件：

$$x(t_0) = x_0, \quad x(t_1) = x_1 \tag{8.4}$$

(b) 終端自由 初期値は指定されるが，終端値は指定されない境界条件：

$$x(t_0) = x_0, \quad x(t_1) = \text{任意} \tag{8.5}$$

8.2 汎関数の最小化 — 変分法

関数形 $x(t)$ を代入すると値が定まるような，関数の関数，

$$J[\text{関数 } x(t)] = \text{値} \tag{8.6}$$

を，**汎関数** (functional) という．最適制御の評価関数 (8.1) は，汎関数の一例である．その歴史的な原型として，ここでは，次のような積分型の汎関数を考える．

$$J[x(t)] = \int_{t_0}^{t_1} L\big(x(\tau), \dot{x}(\tau)\big) d\tau \tag{8.7}$$

L は適当な 2 変数関数である．以下，こうした汎関数を最小化する方法を述べるが，これを**変分法** (variational method) という [4].

8.2.1 変分

汎関数 (8.7) が最小値をとるような関数 $x(t)$ を求めたい．そのための手順は，前章の関数の最小値と同じで，汎関数の独立変数に相当する $x(t)$ の関数形を微少量 $\delta x(t)$ だけずらしたときの，汎関数の増分を 0 とおけばよい．

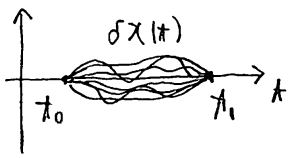
$$\delta J[x(t)] \equiv J[x(t) + \delta x(t)] - J[x(t)] \quad = 0 \tag{8.8}$$

ここで，関数形の微少量 $\delta x(t)$ とは，微小振幅の関数を表し，これを，$x(t)$ の**変分** (variation) という．通常の微少量 Δx の値を定めないのと同じで，具体的な関数形は定めない．ただし，境界条件は，次のように明確に定める．

(a) 両端固定 求める関数 $x(t)$ の両端を固定，

$$x(t_0) = x_0, \quad x(t_1) = x_1 \tag{8.9}$$

したときの最小化問題では，次のような，積分区間 $\int_{t_0}^{t_1}$ の両端で 0 になる変分を使う．



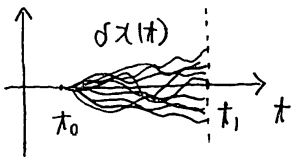
$$\delta x(t_0) = 0, \quad \delta x(t_1) = 0 \tag{8.10}$$

$\delta x(t)$ の様々な候補を束にして図示した．微少量 Δx の実体は様々な数値 $0.01, -0.002, \cdots$ だが，変分 $\delta x(t)$ は様々なグラフである．

(b) 終端自由　求める関数 $x(t)$ の終端を自由,

$$x(t_0) = x_0, \quad x(t_1) = \text{任意} \tag{8.11}$$

にしたときの最小化問題では，初期値 $x(t_0)$ だけを 0 に固定した変分を使う.



$$\delta x(t_0) = 0, \quad \delta x(t_1) = \text{任意} \tag{8.12}$$

8.2.2　汎関数の増分

汎関数 (8.7) の増分 (8.8) は，関数の全微分 (7.5) p50 と同様の展開,

$$L\big(x + \delta x, \dot{x} + \delta\dot{x}\big) - L\big(x, \dot{x}\big) = \frac{\partial L}{\partial x}(x, \dot{x})\delta x + \frac{\partial L}{\partial \dot{x}}(x, \dot{x})\delta\dot{x}$$

と部分積分を使って，次のように整理できる．証明は章末 (**A8 節** p62) に示す.

算法 8.1

$$\delta J[x(t)] = \frac{\partial L}{\partial \dot{x}}\Big|_{t=t_1}\delta x(t_1) - \frac{\partial L}{\partial \dot{x}}\Big|_{t=t_0}\delta x(t_0) + \int_{t_0}^{t_1}\left\{\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)\right\}\delta x d\tau \tag{8.13}$$

►► ($\dot{x}$ による偏微分)　時間微分 $\dot{x}$ による偏微分に違和感を覚えるなら，$b = \dot{x}$ などと置いて，b で偏微分してから，$b = \dot{x}$ に戻せばよい．ようするに，$\dot{x}$ による偏微分とは，2 変数関数 $L(x, \dot{x})$ の第 2 変数 $\dot{x}$ による偏微分にすぎない.

8.2.3　最小化の必要条件 (両端固定)

関数 $x(t)$ の両端が指定された最小化問題を考える.

$$x(t_0) = x_0, \quad x(t_1) = x_1 \tag{8.9 再掲}$$

これに対応する変分,

$$\delta x(t_0) = 0, \quad \delta x(t_1) = 0 \tag{8.10 再掲}$$

を汎関数の増分 (8.13) に代入すると，第 1〜2 項が消えて,

$$\delta J[x(t)] = \int_{t_0}^{t_1}\left\{\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)\right\}\delta x d\tau \quad = 0$$

となる．ここで，変分の関数形 $\delta x(t)$ は，$t_0 < t < t_1$ で任意と仮定したので，上式が 0 となるには，被積分関数が,

$$\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = 0 \tag{8.14}$$

となる以外にない．これを，**オイラーの方程式** (Euler's equation) という．

以上，最小化の必要条件の汎関数バージョンが得られた．すなわち，(8.7) p55 の汎関数 $J[x(t)]$ を最小化する関数 $x(t)$ は，オイラーの方程式 (8.14) の解となる．

8.2.4 最小化の必要条件 (終端自由)

次に，関数 $x(t)$ の終端を指定しない最小化問題を考える．

$$x(t_0) = x_0, \quad x(t_1) = \text{任意} \tag{8.11 再掲}$$

これに対応する変分，

$$\delta x(t_0) = 0, \quad \delta x(t_1) = \text{任意} \tag{8.12 再掲}$$

を汎関数の増分 (8.13) に代入すると，今度は，第 1 項が残る．

$$\delta J[x(t)] = \frac{\partial L}{\partial \dot{x}}\Big|_{t=t_1} \delta x(t_1) + \int_{t_0}^{t_1} \left\{ \frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) \right\} \delta x d\tau \quad = 0$$

今考えている変分 δx は，$\delta x(t)$ $(t_0 < t < t_1)$ が任意であることに加えて，終端値 $\delta x(t_1)$ も任意である．ゆえに，オイラーの方程式に加えて，$\frac{\partial L}{\partial \dot{x}}\big|_{t=t_1} = 0$ も要請される．したがって，終端自由の場合の最小化の必要条件は，2 連立，

$$\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = 0, \quad \frac{\partial L}{\partial \dot{x}}\Big|_{t=t_1} = 0 \tag{8.15}$$

となる．

8.2.5 多次元の場合

n 次元 $\boldsymbol{x}(t) = [x_i(t)]$ のときは，各成分 $x_i(t)$ に対して，同様な算法が成立する．すなわち，両端固定「$\boldsymbol{x}(t_0) = \boldsymbol{x}_0$, $\boldsymbol{x}(t_1) = \boldsymbol{x}_1$」に対して，

$$\frac{\partial L}{\partial x_i} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}_i}\right) = 0 \qquad (i = 1, \cdots, n) \tag{8.16}$$

終端自由「$\boldsymbol{x}(t_0) = \boldsymbol{x}_0$, $\boldsymbol{x}(t_1) =$ 任意」に対して，

$$\frac{\partial L}{\partial x_i} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}_i}\right) = 0, \quad \frac{\partial L}{\partial \dot{x}_i}\Big|_{t=t_1} = 0 \qquad (i = 1, \cdots, n) \tag{8.17}$$

である．これらをベクトルにまとめて，次のように書く．

算法 8.2 (オイラーの方程式)　積分型の汎関数,

$$J[\boldsymbol{x}(t)] = \int_{t_0}^{t_1} L\big(\boldsymbol{x}(\tau), \dot{\boldsymbol{x}}(\tau)\big) d\tau, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0,\ \boldsymbol{x}(t_1) = \boldsymbol{x}_1 \tag{8.18}$$

を最小化するベクトル値関数 $\boldsymbol{x}(t)$ は，**オイラーの方程式**,

$$\frac{\partial L}{\partial \boldsymbol{x}} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\boldsymbol{x}}}\right) \equiv \left[\frac{\partial L}{\partial x_i} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}_i}\right)\right] = \mathbb{O} \tag{8.19}$$

を満す．終端自由 ($\boldsymbol{x}(t_1) =$ 任意) では，$\boldsymbol{x}(t_1)$ の代りに次の終端条件が課される．

$$\frac{\partial L}{\partial \dot{\boldsymbol{x}}}\Big|_{t=t_1} \equiv \left[\frac{\partial L}{\partial \dot{x}_i}\Big|_{t=t_1}\right] = \mathbb{O} \tag{8.20}$$

8.3　最適制御の解法

8.1 節 p54 の例を一般化した (具体的な式を隠した)，次の最適制御問題を考える．

$$\begin{cases} \min\limits_{x,u} : J = \displaystyle\int_{t_0}^{t_1} L\big(x(\tau), u(\tau)\big) d\tau \\ \text{sub.to} : \dot{x}(t) = f\big(x(t), u(t)\big) \end{cases} \quad \text{with } x(t_0) = x_0,\ x(t_1) = x_1 \tag{8.21}$$

境界条件 $x(t_0) = x_0,\ x(t_1) = x_1$ を明示した．

8.3.1　ラグランジュの未定乗数法

関数の最小化と同様に，ラグランジュ乗数 λ を使って，新たな被積分関数,

$$L'\big(x(\tau), u(\tau), \lambda(\tau)\big) \equiv L(x(\tau), u(\tau)) + \lambda\{f\big(x(\tau), u(\tau)\big) - \dot{x}(\tau)\} \tag{8.22}$$

を導入すると，拘束条件付きの問題 (8.21) は，拘束条件無しの問題,

$$\min_{x,u,\lambda} : J = \int_{t_0}^{t_1} L'\big(x(\tau), u(\tau), \lambda(\tau)\big) d\tau, \quad x(t_0) = x_0,\ x(t_1) = x_1 \tag{8.23}$$

に書き直せる．算法 8.2 p58 によれば，J の最小値を与える各関数 $\boldsymbol{x}(t) \equiv \big(x(t), u(t), \lambda(t)\big)$ は，次のオイラー方程式の解である．

$$0 = \frac{\partial L'}{\partial x} - \frac{d}{dt}\left(\frac{\partial L'}{\partial \dot{x}}\right) = \frac{\partial L}{\partial x} + \lambda\frac{\partial f}{\partial x} - \frac{d}{dt}(-\lambda) \quad \therefore \dot{\lambda} = -\frac{\partial}{\partial x}(L + \lambda f) \tag{8.24}$$

$$0 = \frac{\partial L'}{\partial u} - \frac{d}{dt}\left(\frac{\partial L'}{\partial \dot{u}}\right) = \frac{\partial L}{\partial u} + \lambda\frac{\partial f}{\partial u} - \frac{d}{dt}(0) \quad \therefore 0 = \frac{\partial}{\partial u}(L + \lambda f) \tag{8.25}$$

$$0 = \frac{\partial L'}{\partial \lambda} - \frac{d}{dt}\left(\frac{\partial L'}{\partial \dot{\lambda}}\right) = f\big(x, u\big) - \dot{x} - \frac{d}{dt}(0) \quad \therefore \dot{x} = f\big(x, u\big) \tag{8.26}$$

この 3 連立方程式 (微分方程式 2 本 (8.24), (8.26) と条件式 1 本 (8.25)) の成立が，**最適化の必要条件** (J の最小化の必要条件) である．ただし，終端自由 ($x_1 =$ 任意) の

問題では，$x(t)$ の終端条件の代りに，$\lambda(t)$ の終端条件，

$$0 = \frac{\partial L'}{\partial \dot{x}}\Big|_{t=t_1} = -\lambda(t_1) \quad \therefore \lambda(t_1) = 0 \tag{8.27}$$

が課される．

8.3.2 例題 — 最適制御の解法

試しに，**8.1 節**の例で計算してみよう．この問題の L と f は，

$$L = q\,x(t)^2 + r\,u(t)^2, \quad f = a\,x(t) + b\,u(t) \tag{8.28}$$

である．これを最適化の必要条件 (8.24)〜(8.26) に代入すると，

$$\dot{\lambda}(t) = -2q\,x(t) - a\,\lambda(t), \quad 0 = 2r\,u(t) + b\,\lambda(t), \quad \dot{x}(t) = a\,x(t) + b\,u(t) \tag{8.29}$$

となる．まず，第 2 式より，最適入力 $u(t) = -\frac{b}{2r}\lambda(t)$ が判明する．これを第 3 式に代入すると，2 連立の微分方程式，

$$\begin{cases} \dot{x}(t) = a\,x(t) - \frac{b^2}{2r}\,\lambda(t) & \text{(状態方程式)} \\ \dot{\lambda}(t) = -2q\,x(t) - a\,\lambda(t) & \text{(随伴方程式)} \end{cases} \tag{8.30}$$

が得られる．第 1 式は，状態変数 $x(t)$ に関する状態方程式である．第 2 式を，**随伴方程式** (adjoint equation) という．このとき，ラグランジュ乗数 $\lambda(t)$ は，**随伴変数** (adjoint variable) と呼ばれる．

(8.30) を解くと，評価関数 $J = \int L d\tau$ を最小化する $x(t)$ と $\lambda(t)$ の候補が求まる．これらから最適入力の候補 $u(t) = -\frac{b}{2r}\lambda(t)$ が定まる．境界条件は，それぞれ，

- **両端固定** $\cdots$ $x(t_0) = x_0,\, x(t_1) = x_1$
- **終端自由** $\cdots$ $x(t_0) = x_0,\, \lambda(t_1) = 0$ $\quad \because$ (8.24) p58

として与える．以上が，最適制御問題の解法のあらすじである[1)]．

実習 8.1 (8.30) の数値解を求め，状態 $x(t)$ と最適入力 $u(t)$ の時間応答を観察せよ．境界条件は，両端固定 $x(0) = 1,\, x(1) = 2$ と，終端自由 $x(0) = 1,\, \lambda(1) = 0$ の 2 種類とし，その他のパラメータは，$a = 1,\, b = 1,\, q = 100,\, r = 1$ とせよ．次に，重み q, r を変化させて，重みの効果を考察せよ．

[1)] ここで紹介したのは「変分法による解法」である．もう1つ「動的計画法による解法」というのがある．前者の解は最適化 (評価関数の最小化) の必要条件，後者の解は十分条件を与える．前者は適用範囲が広いが，最適入力の候補しか得られない．後者は最適入力そのものが直接が求まるが，適用範囲が狭い．線形制御系では両者の答は一致する．

8.3.3　ハミルトニアン

(8.24)〜(8.26) には共通項 $L+\lambda f$ がある．これを表す関数,

$$H(x,u,\lambda) \equiv L(x,u) + \lambda\, f(x,u) \tag{8.31}$$

を導入し，**ハミルトニアン** (Hamiltonian) と呼ぶ．(8.24)〜(8.26) に代入すると,

$$\dot{\lambda} = -\frac{\partial H}{\partial x} \qquad \text{(随伴方程式)} \tag{8.32}$$

$$\frac{\partial H}{\partial u} = 0 \qquad \text{(入力の条件)} \tag{8.33}$$

$$\dot{x} = \frac{\partial H}{\partial \lambda} \quad \big(= f(x,u)\big) \qquad \text{(状態方程式)} \tag{8.34}$$

という簡明な表現が得られる．ただし，$\lambda(t)$ の終端条件だけは仲間外れで,

$$0 = \frac{\partial L'}{\partial \dot{x}}\Big|_{t=t_1} = -\lambda(t_1) \quad \therefore \lambda(t_1) = 0 \qquad \text{(8.24) 再掲}$$

のままである．

8.3.4　多次元の場合

状態変数が n 次元ベクトル $\boldsymbol{x}(t) = [x_i(t)]$ のときは，随伴変数も同次元のベクトル $\boldsymbol{\lambda}(t) = [\lambda_i(t)]$ となる．その他も含めて,

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix},\ \boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 \\ \vdots \\ \lambda_n \end{bmatrix},\ \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) = \begin{bmatrix} f_1(\boldsymbol{x},\boldsymbol{u}) \\ \vdots \\ f_n(\boldsymbol{x},\boldsymbol{u}) \end{bmatrix},\ \boldsymbol{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_m \end{bmatrix} \tag{8.35}$$

とする．制御入力 $\boldsymbol{u}$ は m 次元とした．$L(\boldsymbol{x},\boldsymbol{u})$ はそもそもスカラである．

こうして多次元化しても，n 次元用のハミルトニン (スカラ),

$$H(\boldsymbol{x},\boldsymbol{u},\boldsymbol{\lambda}) \equiv L(\boldsymbol{x},\boldsymbol{u}) + \sum_{k=1}^{n} \lambda_k f_k(\boldsymbol{x},\boldsymbol{u}) \tag{8.36}$$

と，各成分,

$$x = x_i, \quad \lambda = \lambda_i \quad (i = 1,\cdots,n), \qquad u = u_j \quad (j = 1,\cdots,m) \tag{8.37}$$

を，順次 (8.31)〜(8.34) に代入していけば，最適化の必要条件が得られる．

その結果を，(2.9) p12 の**ヤコビ行列**と，その各行に相当する**勾配**,

$$\frac{\partial L}{\partial \boldsymbol{x}} \equiv \left(\frac{\partial L}{\partial x_1}, \cdots, \frac{\partial L}{\partial x_n} \right) \quad \text{※横ベクトル} \tag{8.38}$$

で整理すると，次のような n 次元の公式が得られる[2)]．証明を **B8 節** p63 に示す．

[2)] ここまで徹底的に整理しなくても問題は解けるが，市販の教科書の多くがこの形式で書かれている．

算法 8.3 (最適制御問題の解法) n 次元の最適制御問題,

$$\begin{cases} \min\limits_{(\boldsymbol{x},\boldsymbol{u})} : J = \int_{t_0}^{t_1} L\big(\boldsymbol{x}(\tau),\boldsymbol{u}(\tau)\big)d\tau \\ \text{sub.to} : \boldsymbol{x}(t) = \boldsymbol{f}\big(\boldsymbol{x}(t),\boldsymbol{u}(t)\big) \end{cases} \quad \text{with } \boldsymbol{x}(t_0)=\boldsymbol{x}_0,\ \boldsymbol{x}(t_1)=\boldsymbol{x}_1 \tag{8.39}$$

を考える．J を最小化する $\boldsymbol{u}(t)$ は，次の方程式の解となる．(^{T} は転置)

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{\partial H}{\partial \boldsymbol{x}}\right)^T = -\left(\frac{\partial L}{\partial \boldsymbol{x}}\right)^T - \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{\lambda} \quad \text{(随伴方程式)} \tag{8.40}$$

$$\mathbb{O} = \left(\frac{\partial H}{\partial \boldsymbol{u}}\right)^T = \left(\frac{\partial L}{\partial \boldsymbol{u}}\right)^T + \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)^T \boldsymbol{\lambda} \quad \text{(入力の条件)} \tag{8.41}$$

$$\dot{\boldsymbol{x}} = \left(\frac{\partial H}{\partial \boldsymbol{\lambda}}\right)^T = \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) \quad \text{(状態方程式)} \tag{8.42}$$

終端自由の場合は，状態変数の終端条件 ($\boldsymbol{x}_1$ = 任意) の代りに，随伴変数の終端条件,

$$\boldsymbol{\lambda}(t_1) = \mathbb{O} \tag{8.43}$$

が課される．

♣ 8 章の補足

● 実習 8.1 p59 の解答例

(8.30) の境界値問題を自動的に解くアルゴリズムもあるが，他書に譲る．ここでは，手計算で等価な初期値問題を導出し，これを解いて数値解を求める．まず，(8.30) を,

$$\dot{\boldsymbol{q}}(t) = \begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix}^{\cdot} = \begin{bmatrix} a & -b^2/(2r) \\ -2q & -a \end{bmatrix} \begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix} = A\boldsymbol{x} \tag{*}$$

の形式で書く．すると，解は行列指数関数を使って $\boldsymbol{q}(t) = \exp(At)\boldsymbol{q}(0)$ と書けるので，1 秒後の解は,

$$\boldsymbol{q}(1) = \exp(A\cdot 1)\boldsymbol{q}(0) = \exp(A)\boldsymbol{q}(0)$$

と表せる．ここで，$\exp(A)$ の列ベクトル $\exp(A) = [\boldsymbol{a}_1, \boldsymbol{a}_2]$ をとると，上式は,

$$\dot{\boldsymbol{q}}(1) = \begin{bmatrix} x(1) \\ \lambda(1) \end{bmatrix} = x(1)\boldsymbol{e}_1 + \lambda(1)\boldsymbol{e}_2 = \exp(A)\boldsymbol{q}(0) = [\boldsymbol{a}_1,\boldsymbol{a}_2]\begin{bmatrix} x(0) \\ \lambda(0) \end{bmatrix} = x(0)\boldsymbol{a}_1 + \lambda(0)\boldsymbol{a}_2$$

のように表記できる．$\boldsymbol{e}_1, \boldsymbol{e}_2$ は標準基底 $\left[\begin{smallmatrix}1\\0\end{smallmatrix}\right], \left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$ である．準備完了． まず，両端固定の場合は $x(0) = x_0$, $x(1) = x_1$ が既知である．既知量を右辺に集めると,

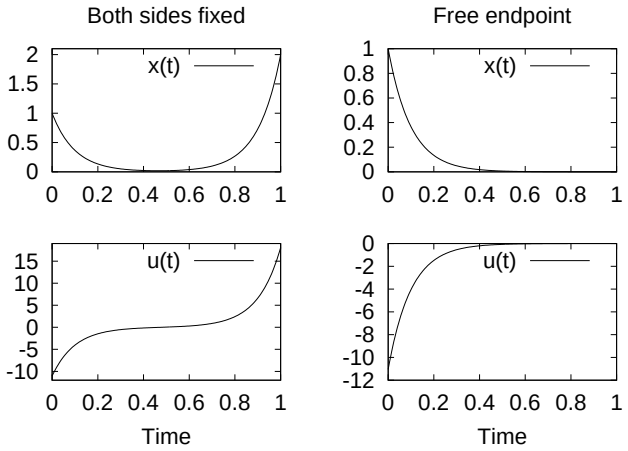
$$-\lambda(0)\boldsymbol{a}_2 + \lambda(1)\boldsymbol{e}_2 = x_0\boldsymbol{a}_1 - x_1\boldsymbol{e}_1 \implies [-\boldsymbol{a}_2, \boldsymbol{e}_2]\begin{bmatrix} \lambda(0) \\ \lambda(1) \end{bmatrix} = x_0\boldsymbol{a}_1 - x_1\boldsymbol{e}_1$$

より，未知だった初期値 $\lambda(0)$ が割り出せる．得られた初期条件 $\boldsymbol{q}(0) = \left[\begin{smallmatrix} x_0 \\ \lambda(0) \end{smallmatrix}\right]$ で与式 (*) を解けば，両端固定の最適軌道が求まる．他方，終端自由の場合は $x(0) = x_0$,

$\lambda(1)=\lambda_1=0$ が既知なので，

$$-\lambda(0)\boldsymbol{a}_2+x(1)\boldsymbol{e}_1=x_0\boldsymbol{a}_1-\lambda_1\boldsymbol{e}_2 \implies [-\boldsymbol{a}_2,\boldsymbol{e}_1]\begin{bmatrix}\lambda(0)\\ x(1)\end{bmatrix}=x_0\boldsymbol{a}_1-\lambda_1\boldsymbol{e}_2$$

より，未知の初期値 $\lambda(0)$ が定まる．以上，境界値問題が初期値問題に変換できた．これらを解いて $x(t),u(t)$ を表示する **Code 13** を実行せよ．次のような結果が得られる．



重みの効果だが，出力 $x(t)$ の重み q を大きくすると，$x(t)$ の絶対値が小さい区間が増え，$x(t)$ の変化が急になる．また，出力 $u(t)$ の重み r を大きくすると，$x(t)$ の絶対値が小さくなる．

A8　算法 8.1 p56 の証明

増分の定義 (8.8) p55 に，汎関数 (8.7) p55 を代入すると，

$$\begin{aligned}\delta J[x(t)] &= J[x(t)+\delta x(t)]-J[x(t)]\\ &=\int_{t_0}^{t_1} L\big(x(\tau)+\delta x(\tau),\dot{x}(\tau)+\delta\dot{x}(\tau)\big)d\tau-\int_{t_0}^{t_1} L\big(x(\tau),\dot{x}(\tau)\big)d\tau\\ &=\int_{t_0}^{t_1}\underbrace{L\big(x(\tau)+\delta x(\tau),\dot{x}(\tau)+\delta\dot{x}(\tau)\big)-L\big(x(\tau),\dot{x}(\tau)\big)}_{\bigstar}\,d\tau \qquad (8.44)\end{aligned}$$

となる．被積分項 ★ を，(7.5) と同じ要領で展開すると，※ (τ) は省略

$$\begin{aligned}\bigstar &= L\big(x+\delta x,\dot{x}+\delta\dot{x}\big)-L\big(x,\dot{x}\big)\\ &=\frac{\partial L}{\partial x}(x,\dot{x})\delta x+\frac{\partial L}{\partial\dot{x}}(x,\dot{x})\delta\dot{x}+\underline{O(\delta x,\delta\dot{x})^2}\end{aligned}$$

となるが，$\delta x,\delta\dot{x}$ の振幅が十分小さければ，下線部は 0 とみなせるので，被積分項は，

$$\bigstar=\frac{\partial L}{\partial x}(x,\dot{x})\delta x+\frac{\partial L}{\partial\dot{x}}(x,\dot{x})\delta\dot{x} \qquad (8.45)$$

となる．

この ★ を (8.44) に代入すると，増分は，$\big((x,\dot{x})$ は省略$\big)$

$$\delta J[x(t)] = \int_{t_0}^{t_1} \bigstar d\tau = \int_{t_0}^{t_1} \frac{\partial L}{\partial x}\delta x d\tau + \underline{\int_{t_0}^{t_1} \frac{\partial L}{\partial \dot{x}}\delta \dot{x} d\tau} \tag{8.46}$$

となる．仕上げに，第 2 項 (下線) を部分積分すると，

$$\int_{t_0}^{t_1} \frac{\partial L}{\partial \dot{x}}\delta \dot{x} d\tau = \left[\frac{\partial L}{\partial \dot{x}}\delta x(\tau)\right]_{t_0}^{t_1} - \int_{t_0}^{t_1} \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)\delta x d\tau \tag{8.47}$$

を得る．これを (8.46) に戻すと，算法 8.1 p56 の，

$$\begin{aligned}\delta J[x(t)] &= \int_{t_0}^{t_1} \frac{\partial L}{\partial x}\delta x d\tau + \left[\frac{\partial L}{\partial \dot{x}}\delta x(\tau)\right]_{t_0}^{t_1} - \int_{t_0}^{t_1} \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)\delta x d\tau \\ &= \left[\frac{\partial L}{\partial \dot{x}}\delta x(t)\right]_{t=t_1} - \left[\frac{\partial L}{\partial \dot{x}}\delta x(t)\right]_{t=t_0} \\ &\quad + \int_{t_0}^{t_1} \left\{\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)\right\}\delta x d\tau \quad (8.13)\ \text{p56}\end{aligned}$$

が得られる．証明終り．

B8 算法 8.3 p61 の証明

ハミルトニアン (8.36) p60 より，まず，随伴方程式の i 成分は，

$$\dot{\lambda}_i = -\frac{\partial H}{\partial x_i} \implies \dot{\lambda}_i = -\frac{\partial L}{\partial x_i} - \sum_{k=1}^{n} \lambda_k \frac{\partial f_k}{\partial x_i}$$

である．横ベクトルとして並べて，勾配 (8.38) p60 とヤコビ行列 (2.9) p12 で整理すると，

$$\begin{aligned}\dot{\boldsymbol{\lambda}}^T = (\lambda_1,\cdots,\lambda_n)^{\cdot} &= -\left(\frac{\partial L}{\partial x_1},\cdots,\frac{\partial L}{\partial x_n}\right) \\ &\quad -\left(\lambda_1\frac{\partial f_1}{\partial x_1}+\cdots+\lambda_n\frac{\partial f_n}{\partial x_1},\cdots,\lambda_1\frac{\partial f_1}{\partial x_n}+\cdots+\lambda_n\frac{\partial f_n}{\partial x_n}\right) \\ &= -\frac{\partial L}{\partial \boldsymbol{x}} - (\lambda_1,\cdots,\lambda_n)\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & & \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \\ &= -\frac{\partial L}{\partial \boldsymbol{x}} - \boldsymbol{\lambda}^T\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} = -\frac{\partial}{\partial \boldsymbol{x}}(L - \boldsymbol{\lambda}^T\boldsymbol{f}) = -\frac{\partial H}{\partial \boldsymbol{x}} \end{aligned} \tag{8.48}$$

となる．両辺の転置をとれば，(8.40) p61 を得る．

同様の計算で，入力の条件 (8.41) も，状態方程式 (8.42) も示せる．

また，終端自由のときの，随伴変数の終端条件は $\lambda_i(t_1) = 0$ より，$\boldsymbol{\lambda}(t_1) = [\lambda_i(t_1)] = \mathbb{O}$ となる．証明終り．

9

最適レギュレータ

終端自由の最適制御問題に制限を設けて，状態方程式を**線形**，評価関数を **2 次形式** (以下で学ぶ) に限定し，終端時間を無限大にすると，制御入力 $\boldsymbol{u}(t)$ を事前に決める必要がなくなる．具体的には，状態フィードバック $\boldsymbol{u}(t) = -K\boldsymbol{x}(t)$ で最適制御が実現できてしまう．問題は，そうなるゲイン K の見付け方である．

9.1 斜交成分の内積と 2 次形式

「2 次形式」なるものを理解する鍵は「内積」である．

9.1.1 直交成分による内積と長さ

簡単のため 2 次元ベクトル $\boldsymbol{x}, \boldsymbol{y}$ を考える．これらの成分どうしの積の和,

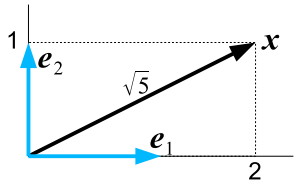
$$\boldsymbol{x} \cdot \boldsymbol{y} \equiv x_1 y_1 + x_2 y_2 = (x_1, x_2) \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \boldsymbol{x}^T \boldsymbol{y} \tag{9.1}$$

を**内積** (inner product) と呼んだ．内積の値はスカラであることに注意せよ．

この内積を使うと，ベクトル $\boldsymbol{x}$ の**長さ** $|\boldsymbol{x}|$ は，内積による自乗の平方根,

$$|\boldsymbol{x}| = \sqrt{\boldsymbol{x} \cdot \boldsymbol{x}} = \sqrt{x_1^2 + x_2^2} \tag{9.2}$$

で書ける[1)]．例えば，次のベクトル $\boldsymbol{x}$ の長さを計算してみよう．($\boldsymbol{e}_1, \boldsymbol{e}_2$ は標準基底)



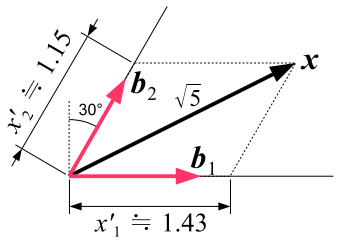
図より，直交成分は $\boldsymbol{x} = \left[\begin{smallmatrix} 2 \\ 1 \end{smallmatrix}\right]$ だから，その長さは次のように計算できる．

$$|\boldsymbol{x}| = \sqrt{\boldsymbol{x} \cdot \boldsymbol{x}} = \sqrt{2^2 + 1^2} = \sqrt{5} \tag{9.3}$$

[1)] これを称して数学では「内積から誘導された長さ (ノルム)」などという．

9.1.2 斜交成分による内積と長さ

それでは，同じベクトル $\boldsymbol{x}$ の寸法として，次のような斜交成分 $\boldsymbol{x}' = \left[\begin{smallmatrix} x'_1 \\ x'_2 \end{smallmatrix}\right]$ が与えられた場合はどうだろうか．



実は，こうした斜交成分 $\boldsymbol{x}', \boldsymbol{y}'$ に対しても，それ用の内積 $*$ を作ることができる．

$$\boldsymbol{x}' * \boldsymbol{y}' \equiv (x'_1, x'_2) \begin{bmatrix} \boldsymbol{b}_1 \cdot \boldsymbol{b}_1 & \boldsymbol{b}_1 \cdot \boldsymbol{b}_2 \\ \boldsymbol{b}_2 \cdot \boldsymbol{b}_1 & \boldsymbol{b}_2 \cdot \boldsymbol{b}_2 \end{bmatrix} \begin{bmatrix} y'_1 \\ y'_2 \end{bmatrix} = \boldsymbol{x}'^T G \boldsymbol{y}' \tag{9.4}$$

行列 G を，**計量テンソル**という．G の成分は，斜交基底ベクトル $\boldsymbol{b}_1, \boldsymbol{b}_2$ どうしの普通の内積である．例えば，図の斜交基底の計量テンソルは，$\boldsymbol{b}_1 = \left[\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right]$, $\boldsymbol{b}_2 = \left[\begin{smallmatrix} 1/2 \\ \sqrt{3}/2 \end{smallmatrix}\right]$ より，

$$G = \begin{bmatrix} \boldsymbol{b}_1 \cdot \boldsymbol{b}_1 & \boldsymbol{b}_1 \cdot \boldsymbol{b}_2 \\ \boldsymbol{b}_2 \cdot \boldsymbol{b}_1 & \boldsymbol{b}_2 \cdot \boldsymbol{b}_2 \end{bmatrix} = \begin{bmatrix} 1 & 1/2 \\ 1/2 & 1 \end{bmatrix} \tag{9.5}$$

のようになる．ちなみに，標準基底の計量テンソルは単位行列となる．

さて，斜交成分用の内積 (9.4) は，非常にうまく作られていて，普通の内積と同様に，ベクトル $\boldsymbol{x}$ の長さが，自乗の平方根で計算できる．

$$|\boldsymbol{x}| = \sqrt{\boldsymbol{x}' * \boldsymbol{x}'} = \sqrt{\boldsymbol{x}'^T G \boldsymbol{x}'} \tag{9.6}$$

試しに，手動で測った図の斜交成分 $\boldsymbol{x}' = \left[\begin{smallmatrix} 1.43 \\ 1.15 \end{smallmatrix}\right]$ から，(9.6) で長さを計算すると，

```
--> G=[1,1/2;1/2,1];
--> xx=[1.43;1.15];
--> sqrt(xx'*G*xx)
ans =  2.2387
--> sqrt(5)
ans =  2.2361
```

のように，正解 $\sqrt{5} \approx 2.24$ が得られる[2)]．

9.1.3 正定値 2 次形式

一般に，ベクトル $\boldsymbol{x}$ の斜交成分 $\boldsymbol{y}$ (前節では $\boldsymbol{x}'$ と書いた) から計算した「ベクトル $\boldsymbol{x}$ の長さ」の 2 乗，

$$|\boldsymbol{x}|^2 = \boldsymbol{y}^T G \boldsymbol{y} \tag{9.7}$$

を，**正定値 2 次形式** (positive-definite quadratic form) という．正定値とは，

[2)] 手動で寸法を測ったので，誤差は大目に見てね．

- 原則 $\boldsymbol{y}^T G\boldsymbol{y} > 0$ であり，$\boldsymbol{y}^T G\boldsymbol{y} = 0$ となるのは $\boldsymbol{y} = \mathbb{O}$ のときに限る．

という性質のことだ．$\boldsymbol{y}^T G\boldsymbol{y}$ はベクトルの長さの 2 乗だから，あたり前の性質である．

こうした背景を忘れて，一般の行列 A に対する演算法則，

$$\boldsymbol{y}^T A\boldsymbol{y} = (\text{ベクトルの転置})(\text{行列})(\text{ベクトル}) \tag{9.8}$$

だけに着目したものを，行列 A の **2 次形式**という．A を勝手に選ぶと，上式の値はマイナスにもなる．このようなマイナスの 2 次形式は，ベクトルの長さの 2 乗とは，全くの別物となる．

そこで「長さ」が欲しいときは，2 次形式に「正定値」なる条件を課す．2 次形式の符号は，結局のところ，行列 A の成分の選び方に左右される．ゆえに「正定値」を基準とした，行列の分類法がある．

- 2 次形式 $\boldsymbol{y}^T A\boldsymbol{y}$ が正定値 (ベクトルの長さの 2 乗) となるような行列 A を，**正定値行列** (positive-definite matrix) という．これを「$A > 0$」と書く．

以上，正定値行列 A で作った 2 次形式 $\boldsymbol{y}^T A\boldsymbol{y}$ は，適当な斜交成分 $\boldsymbol{y}$ で測られた，何らかのベクトルの長さの 2 乗を表す．

もう 1 つ，**非負定値**または**半正定値**行列 (non-negative definite/positive semidefinite) というのがあり，これを「$A \geq 0$」と書く．例えば，

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \implies \boldsymbol{y}^T A\boldsymbol{y} = y_1^2 \tag{9.9}$$

が非負定値の一例である．これは，$\boldsymbol{y} = \left[\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right] \neq \mathbb{O}$ に対して $\boldsymbol{y}^T A\boldsymbol{y} = 0$ となるので，正定値ではない．しかし絶対にマイナスにはならないので「非負」である．

2 次形式 $\boldsymbol{y}^T A\boldsymbol{y}$ を，零ベクトル $\boldsymbol{y} = \mathbb{O}$ の検出器とみなすと，$A > 0$ (正定値) なら，取りこぼしはない．$\boldsymbol{y}^T A\boldsymbol{y} = 0$ を検出したら必ず $\boldsymbol{y} = \mathbb{O}$ である．しかし，$A \geq 0$ (非負定値) だと，取りこぼしが起こる．$\boldsymbol{y}^T A\boldsymbol{y} = 0$ となる $\boldsymbol{y} \neq \mathbb{O}$ が存在するからだ．

次の判定則が知られている[3)]．

算法 9.1 (正定値行列の判定則 1)　実行列 A が正定値 (非負定値) であるための必要十分条件は，A が対称行列かつ A の全ての固有値が正 (非負) であることである．

算法 9.2 (正定値行列の判定則 2)　実行列 A が正定値 (非負定値) であるための必要十分条件は，A が対称行列で，なおかつ小行列式，

$$D_k \equiv \begin{vmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kk} \end{vmatrix} \quad (1 \leq k \leq n) \tag{9.10}$$

が，全ての k で $D_k > 0$ (≥ 0) となることである．

[3)] 大関 [5] の 5.4 節など．

9.1.4 対称 2 次形式の勾配

対称行列 $A = A^T$ で作った 2 次形式,

$$Q = \boldsymbol{y}^T A \boldsymbol{y} = y_1^2 a_{11} + 2y_1 y_2 a_{12} + y_2^2 a_{22} \quad (a_{12} = a_{21})$$

について，$\boldsymbol{y}$ で勾配 (8.38) p60 をとると，

$$\begin{aligned}\frac{\partial Q}{\partial \boldsymbol{y}} = \left(\frac{\partial Q}{\partial y_1},\ \frac{\partial Q}{\partial y_2}\right) &= (2y_1 a_{11} + 2y_2 a_{12},\ 2y_1 a_{21} + 2y_2 a_{22}) \\ &= 2(y_1, y_2)\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = 2\boldsymbol{y}^T A\end{aligned}$$

となる．すなわち，次の公式が得られる．

$$\frac{\partial}{\partial \boldsymbol{y}}\left(\boldsymbol{y}^T A \boldsymbol{y}\right) = 2\boldsymbol{y}^T A \quad (A = A^T) \tag{9.11}$$

この公式は，$\boldsymbol{y}$ や A の次元によらず成立する．

9.2 最適レギュレータ (LQR) 問題

状態方程式が**線形**，評価関数が **2 次形式**，境界条件が**終端自由**であるような最適制御問題を考える．これを，(線形の) **最適レギュレータ問題** (optimal regulator problem) または **LQR 問題** (linear quadratic regulator problem) という[4)]．

$$\begin{aligned}&\min_{(x,u)} : J = \int_{t_0}^{t_1} \left\{\boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{u}^T R \boldsymbol{u}\right\} d\tau, \quad Q \geq 0, \quad R > 0, \\ &\text{sub.to} : \dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u}, \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \quad \boldsymbol{x}(t_1) = \text{終端自由}\end{aligned} \tag{9.12}$$

Q, R はともに対称行列で，**重み**と呼ばれる．状態の重みは $Q \geq 0$ (非負定値)，入力の重みは $R > 0$ (正定値) とされる．

9.2.1 最適化の必要条件

この問題のハミルトニアンは,

$$H = L + \boldsymbol{\lambda}^T \boldsymbol{f}, \quad L = \boldsymbol{x}^T Q \boldsymbol{x} + \boldsymbol{u}^T R \boldsymbol{u}, \quad \boldsymbol{f} = A\boldsymbol{x} + B\boldsymbol{u} \tag{9.13}$$

となるが，これを算法 8.3 p61 に代入すると，ヤコビ行列 (2.9) p12 と 2 次形式の勾配 (9.11) p67 を用いて,

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{\partial L}{\partial \boldsymbol{x}}\right)^T - \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)^T \boldsymbol{\lambda} = -2Q\boldsymbol{x} - A^T \boldsymbol{\lambda} \qquad \because Q^T = Q \tag{9.14}$$

$$\mathbb{O} = \left(\frac{\partial L}{\partial \boldsymbol{u}}\right)^T + \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)^T \boldsymbol{\lambda} = 2R\boldsymbol{u} + B^T \boldsymbol{\lambda} \qquad \because R^T = R \tag{9.15}$$

[4)] 加藤 [6] の p.41，吉川 [7] の p.145 など．

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = A\boldsymbol{x} + B\boldsymbol{u} \tag{9.16}$$

という必要条件が得られる．まず，(9.15) の両辺に R^{-1} を掛けて，

$$\boldsymbol{u} = -\frac{1}{2}R^{-1}B^T\boldsymbol{\lambda} \tag{9.17}$$

が判明する．これを，(9.16) に代入すると，

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u} = A\boldsymbol{x} - \frac{1}{2}BR^{-1}B^T\boldsymbol{\lambda} \tag{9.18}$$

となる．ここで，$\boldsymbol{x}, \boldsymbol{\lambda}$ の全成分を並べたベクトル $\boldsymbol{q}$ をとると，必要条件は，

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{bmatrix}^{\cdot} = \begin{bmatrix} A & -\frac{1}{2}BR^{-1}B^T \\ -2Q & -A^T \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{bmatrix} = M\boldsymbol{q} \tag{9.19}$$

のように短くまとまる．これが，最適レギュレータ問題 (9.12) の必要条件である．

ちなみに，実習 8.1 p59 で観察した終端自由の時間応答は，(9.19) の解の一例となっている．

9.2.2　最適制御の状態フィードバック表示

さて，(9.19) の解は，一般に，行列指数関数によって，

$$\boldsymbol{q}(t) = \exp\{M(t-s)\}\,\boldsymbol{q}(s) \tag{9.20}$$

と書けた．ここで，思いつきにくいテクニックとして，終端 $t = t_1$ を基準に解を表示してやる[5)]．

$$\boldsymbol{q}(t) = \Phi(t)\,\boldsymbol{q}(t_1), \quad \Phi(t) \equiv \exp\{M(t-t_1)\} \tag{9.21}$$

こうすると，うまいことに，随伴変数の終端条件 $\boldsymbol{\lambda}(t_1) = \mathbb{O}$ が代入できて，

$$\boldsymbol{q}(t_1) = \begin{bmatrix} \boldsymbol{x}(t_1) \\ \boldsymbol{\lambda}(t_1) \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}(t_1) \\ \mathbb{O} \end{bmatrix} \tag{9.22}$$

となる．これを (9.21) に代入し，推移行列 $\Phi(t)$ を 4 等分して整理すると，

$$\underline{\begin{bmatrix} \boldsymbol{x}(t) \\ \boldsymbol{\lambda}(t) \end{bmatrix}} \overset{(9.19)}{\equiv} \boldsymbol{q}(t) \overset{(9.21)}{=} \Phi(t)\,\boldsymbol{q}(t_1) \overset{(9.22)}{=} \begin{bmatrix} \Phi_{11}(t) & \Phi_{12}(t) \\ \Phi_{21}(t) & \Phi_{22}(t) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}(t_1) \\ \mathbb{O} \end{bmatrix} = \underline{\begin{bmatrix} \Phi_{11}(t)\boldsymbol{x}(t_1) \\ \Phi_{21}(t)\boldsymbol{x}(t_1) \end{bmatrix}} \tag{9.23}$$

が得られる．下線部から $\boldsymbol{x}(t_1)$ を消去すると，

$$\boldsymbol{\lambda}(t) = \Phi_{21}(t)\boldsymbol{x}(t_1) = \Phi_{21}(t)\Phi_{11}(t)^{-1}\boldsymbol{x}(t) \quad \equiv 2P(t)\,\boldsymbol{x}(t) \tag{9.24}$$

という関係が判明する．後の計算で係数を見やすくするため $\Phi_{21}(t)\Phi_{11}(t)^{-1}$ を $2P(t)$ と書いた．これを，制御入力 (9.17) p68 に代入すると，めでたく，

[5)] このテクニックにおいて，時間のパラメータ $t - t_1$ は積分区間の全域 (終端だけ 0) でマイナスとなる．これを称して「逆時間で解く」という言い方をしばしば耳にする．

$$\boldsymbol{u}(t) = -\frac{1}{2}R^{-1}B^T\boldsymbol{\lambda}(t) = -R^{-1}B^T\,P(t)\,\boldsymbol{x}(t) \tag{9.25}$$

という，状態フィードバック則が出てくる．残された問題は，謎の行列，

$$P(t) \equiv \frac{1}{2}\Phi_{21}(t)\Phi_{11}(t)^{-1} \tag{9.26}$$

の求め方だ．

9.2.3 リカッチ方程式

絶妙なことに，$P(t)$ を時間微分すると，推移行列 $\Phi_{ij}(t)$ が消去できる．そのために，次の微分公式を使う．証明は章末 (**A9 節** p73) に示す．

算法 9.3 (行列の時間微分) 時間に依存する行列 $A(t), B(t)$ に対して，

(1) $\{A(t)\,B(t)\}^{\cdot} = \dot{A}(t)\,B(t) + A(t)\,\dot{B}(t)$

(2) $\{A(t)^{-1}\}^{\cdot} = -A(t)^{-1}\,\dot{A}(t)\,A(t)^{-1}$

早速，(9.26) の $P(t)$ を微分すると，(以下，(t) は省く)

$$\dot{P} = \frac{1}{2}\dot{\Phi}_{21}\Phi_{11}^{-1} + \frac{1}{2}\Phi_{21}(\Phi_{11}^{-1})^{\cdot} = \frac{1}{2}\dot{\Phi}_{21}\Phi_{11}^{-1} - \underbrace{\frac{1}{2}\Phi_{21}\Big(\Phi_{11}^{-1}}_{P}\,\dot{\Phi}_{11}\Phi_{11}^{-1}\Big)^{\text{算法 9.3 (2)}} \tag{9.27}$$

となる．次に，Φ は，$\dot{\boldsymbol{q}} = M\boldsymbol{q}$ (9.19) の**推移行列**だから，算法 3.4 p19 (3) より，同じ形の微分方程式 $\dot{\Phi} = M\Phi$ を満足する．ゆえに，

$$\begin{bmatrix}\dot{\Phi}_{11} & \dot{\Phi}_{12}\\ \dot{\Phi}_{21} & \dot{\Phi}_{22}\end{bmatrix} = \dot{\Phi} = M\Phi = \begin{bmatrix}M_{11} & M_{12}\\ M_{21} & M_{22}\end{bmatrix}\begin{bmatrix}\Phi_{11} & \Phi_{12}\\ \Phi_{21} & \Phi_{22}\end{bmatrix} \tag{9.28}$$

が成立し，$\dot{\Phi}_{ij}$ は Φ_{ij} で書き下せる．

$$\dot{\Phi}_{11} = M_{11}\Phi_{11} + M_{12}\Phi_{21}, \quad \dot{\Phi}_{21} = M_{21}\Phi_{11} + M_{22}\Phi_{21} \tag{9.29}$$

これらを，(9.27) に代入すると，

$$\dot{\Phi}_{21}\Phi_{11}^{-1} = (M_{21}\Phi_{11} + M_{22}\Phi_{21})\Phi_{11}^{-1} = M_{21} + M_{22}(2P)$$

$$\dot{\Phi}_{11}\Phi_{11}^{-1} = (M_{11}\Phi_{11} + M_{12}\Phi_{21})\Phi_{11}^{-1} = M_{11} + M_{12}(2P)$$

より，

$$\dot{P} = \frac{1}{2}M_{21} + M_{22}P - P(M_{11} + 2M_{12}P) = \frac{1}{2}M_{21} + M_{22}P - PM_{11} - P(2M_{12})P$$

が得られる．(9.19) p68 を見ながら，M_{ij} を代入していくと，最終的に，

$$\dot{P} = -Q - A^TP - PA + PBR^{-1}B^TP \tag{9.30}$$

という P に関する微分方程式が得られる．これを，**リカッチ方程式** (Riccati's equation)

という．この方程式の境界条件は，終端条件,

$$P(t_1) = O \quad (\text{零行列}) \tag{9.31}$$

である．なぜなら，(9.21) p68 より，

$$\Phi(t_1) = \exp\{M(t_1 - t_1)\} = E \quad (\text{単位行列}) \tag{9.32}$$

だから，

$$\Phi(t_1) = \begin{bmatrix} 1 & & O \\ & \ddots & \\ O & & 1 \end{bmatrix} = \begin{bmatrix} \Phi_{11}(t_1) & \Phi_{12}(t_1) \\ \Phi_{21}(t_1) & \Phi_{22}(t_1) \end{bmatrix}$$

より，$\Phi_{11}(t_1) = E$，$\Phi_{21}(t_1) = O$ (零行列) となる．これを (9.26) p69 に代入して，

$$P(t_1) = \frac{1}{2}\Phi_{21}(t_1)\Phi_{11}(t_1)^{-1} = O\,E = O \quad (\text{零行列})$$

を得る．以上の結果 (9.25) p69, (9.30), (9.31) をまとめると，次の算法が得られる．

算法 9.4 (有限時間 LQR)　最適レギュレータ (LQR) 問題 (9.12) p67 の最適入力は,

$$\boldsymbol{u}(t) = -K(t)\,\boldsymbol{x}(t), \quad K(t) \equiv R^{-1}B^T\,P(t) \tag{9.33}$$

で与えられる．ただし，行列 $P(t)$ はリカッチ方程式,

$$\dot{P} = -Q - A^T P - PA + PBR^{-1}B^T P, \quad P(t_1) = O \tag{9.34}$$

の解である．

►► (9.33) のフィードバック制御則は，ゲイン $K(t)$ が時間変化するが，これを**可変ゲイン制御** (variable gain control) という．

算法 8.3 p61 において随伴変数 $\boldsymbol{\lambda}(t)$ が果してきた役割を，わざわざ，リカッチ方程式の $P(t)$ で置き換えたことのメリットは,

- リカッチ方程式の解 $P(t)$ は，状態量 $\boldsymbol{x}(t)$ とは無関係に (9.34) だけから求まる.

ということに尽きる．その結果，システムの構造 A, B, Q, R さえ決まれば，どんな状態量 $\boldsymbol{x}(t)$ にも通用する制御則 (9.33) が得られる．

実習 9.1　実習 8.1 p59 で観察した終端自由の最適制御問題,

$$\begin{cases} \min : J = \int_{t_0}^{t_1} q\,x(\tau)^2 + r\,u(\tau)^2 d\tau, \\ \text{sub.to} : \dot{x}(t) = a\,x(t) + b\,u(t), \quad x(t_0) = x_0, \quad x(t_1) = \text{任意} \end{cases} \tag{9.35}$$

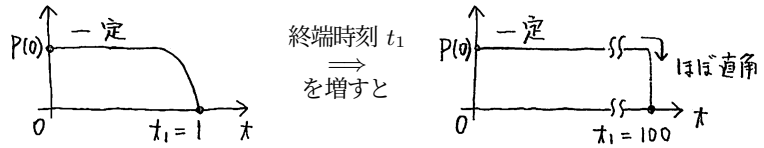
は最適レギュレータ問題となる．対応するリカッチ方程式,

$$\dot{P} = -P - aP - Pa - Pb\frac{1}{r}bP, \quad P(t_1) = 0 \tag{9.36}$$

の数値解 $P(t)$ を求め，制御入力 $u(t) = -\frac{1}{r}bP(t)x(t)$ と状態量 $x(t)$ の時間応答を観察せよ．次に，終端時刻 t_1 を増加させ，$P(t)$ のグラフがどう変化するか観察せよ．

9.3 無限時間最適レギュレータ

実習 9.1 で見たリカッチ方程式の解 $P(t)$ はこんなだった．$P(t)$ はほとんど一定値で，終端時刻 t_1 付近でいきなり急減少した．



ここで，終端時刻 t_1 を大きくすると，急減少する場所が，どんどん右に追いやられる．これを進めて，$t_1 = \infty$ の極限を考えると，減少部分は無限遠点に追いやられ，$P(t)$ は，無限遠点を除く t の全域で，定数関数 $P(t) = P(0)$ になるだろう．

実は，この $P(t)$ の性質は，実習 9.1 に特有のものではなく，リカッチ方程式 (9.34) が示す普遍的な性質である．一般に，次の性質が数学的に証明されている．

$$t_1 = \infty \quad \Longrightarrow \quad P(t) = P(0) \quad \text{※定数行列} \tag{9.37}$$

証明は長いので他書 [6] に譲る．このとき，時間微分 $\dot{P}(t)$ は零行列だから，リカッチ方程式 (9.34) の左辺が $\dot{P}(t) = O$ となり，

$$O = -Q - A^T P - PA + PBR^{-1}B^T P \tag{9.38}$$

という代数方程式が得られる．これを，**リカッチ代数方程式** (Riccati's algebraic equation) という．その結果，可変ゲインだった最適入力 (9.33) p70 が，

$$\boldsymbol{u}(t) = -K\,\boldsymbol{x}(t), \quad K = R^{-1}B^T\,P(0) \quad \text{※定数行列} \tag{9.39}$$

のように固定ゲインのものに変更される．可変ゲイン $K(t)$ のような時間の関数をプログラムで保持するのは面倒だが[6)]，$t_1 = \infty$ の条件では固定ゲイン，すなわち定数行列 K を 1 つ保持するだけで済む．また，数値解法も確立していて，Octave/Scilab 等で簡単に K が求まる．

以上の結果 (9.38), (9.39) をまとめたのが，次の算法である (若干式を整理した)．

算法 9.5 (無限時間 LQR) 終端時刻を $t_1 = \infty$ とすると，最適レギュレータ (LQR) 問題 (9.12) p67 の最適入力は，

$$\boldsymbol{u}(t) = -K\,\boldsymbol{x}(t), \quad K \equiv R^{-1}B^T\,P(0) \tag{9.40}$$

で与えられる．ただし，行列 $P(0)$ はリカッチ代数方程式，

$$A^T P + PA - PBR^{-1}B^T P + Q = O \quad \text{(零行列)} \tag{9.41}$$

の解である．

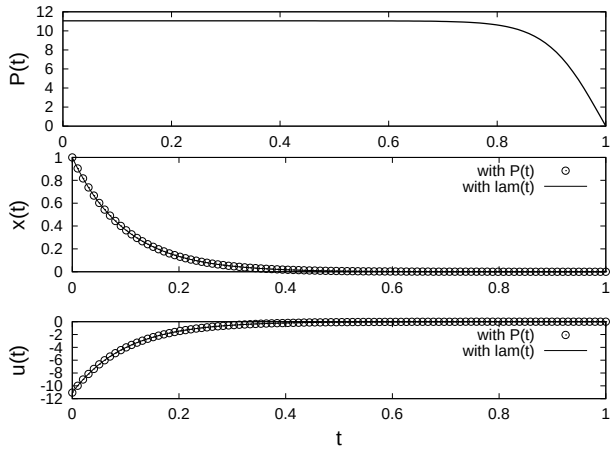
[6)] **Code 14** では，$K(t)$ を時間刻み Δt の数列 $K_i \equiv K(t_i + i\Delta t)$ として求め，中途半端な時刻の値は，線形補間という方法で補った．Δt を小さくすれば精度が上るが，データ量が増えてしまう．

実習 9.2　**Code 15** を実行せよ．乱数で無作為に生成した $A, \boldsymbol{b}$ に対して，無限時間最適レギュレータ K が求まり，状態量の時間応答がプロットされる．Octave/Scilab 等には，最適レギュレータを求める専用の関数があり，ユーザーがリカッチ代数方程式を解く必要はない．

♣ 9 章の補足

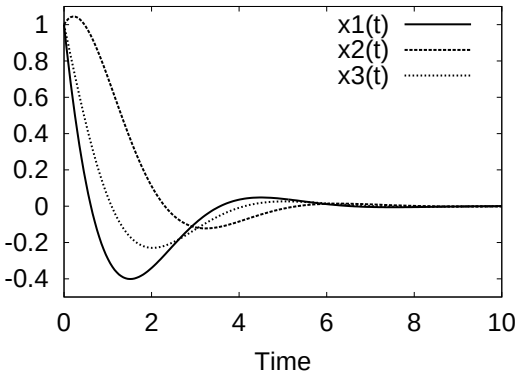
● 実習 9.1 p70 の解答例

Code 13 を実行した後に，**Code 14** を実行せよ．次のような結果が得られる．中段・下段の実線は実習 8.1 p59 の時間応答，○印は今回求めた時間応答である．同じ問題の解なので実線と○は重なる．



● 実習 9.2 p72 の解答例

次のような結果が得られる．乱数で状態方程式を生成するため，時間応答は実行のたびに変化する．



A9 算法 9.3 p69 の証明

積の微分 なるべく楽をするために順にいく．前提として「変数の積」の微分，

$$\{xy\}^{\cdot} = \dot{x}y + x\dot{y} \tag{*}$$

を認めておこう．また，行列の微分 $\dot{A}$ は，ベクトルの微分 (1.2) と同様に，各成分 a_{ij} に対する微分であると定めておく．これを列ベクトルで表現すると，

$$\dot{A} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n]^{\cdot} = [\dot{\boldsymbol{a}}_1, \cdots, \dot{\boldsymbol{a}}_n] \tag{**}$$

である．まず，ベクトル $\boldsymbol{x} = [x_i]$ のスカラー倍の微分は，

$$\{a\boldsymbol{x}\}^{\cdot} = \{[ax_i]\}^{\cdot} \overset{(1.2)}{=} [(ax_i)^{\cdot}] \overset{(*)}{=} [\dot{a}x_i + a\dot{x}_i] = \dot{a}[x_i] + a[\dot{x}_i] = \dot{a}\boldsymbol{x} + a\dot{\boldsymbol{x}} \tag{*1}$$

次に，行列を列ベクトルで $A = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n]$ と表記すると，**5.1.1 節** p30 より，

$$A\boldsymbol{x} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n][x_i] = x_1\boldsymbol{a}_1 + \cdots + x_n\boldsymbol{a}_n$$

と書けた．ゆえに，

$$\begin{aligned}\{A\boldsymbol{x}\}^{\cdot} &= (x_1\boldsymbol{a}_1)^{\cdot} + \cdots + (x_n\boldsymbol{a}_n)^{\cdot} \\ &\overset{(*1)}{=} (\dot{x}_1\boldsymbol{a}_1 + \cdots + \dot{x}_n\boldsymbol{x}_n) + (x_1\dot{\boldsymbol{a}}_1 + \cdots + x_n\dot{\boldsymbol{x}}_n) = \dot{A}\boldsymbol{x} + A\dot{\boldsymbol{x}}\end{aligned} \tag{*2}$$

また，行列 A, B の積は，後者を列ベクトルで書くと，

$$AB = A[\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n] = [A\boldsymbol{b}_1, \cdots, A\boldsymbol{b}_n]$$

となる．これより，

$$\begin{aligned}\{AB\}^{\cdot} = [A\boldsymbol{b}_1, \cdots, A\boldsymbol{b}_n]^{\cdot} &\overset{(**)}{=} [(A\boldsymbol{b}_1)^{\cdot}, \cdots, (A\boldsymbol{b}_n)^{\cdot}] \\ &\overset{(*2)}{=} [\dot{A}\boldsymbol{b}_1 + A\dot{\boldsymbol{b}}_1, \cdots, \dot{A}\boldsymbol{b}_n + A\dot{\boldsymbol{b}}_n] \\ &= [\dot{A}\boldsymbol{b}_1, \cdots, \dot{A}\boldsymbol{b}_n] + [A\dot{\boldsymbol{b}}_1, \cdots, A\dot{\boldsymbol{b}}_n] = \dot{A}\,B + A\,\dot{B}\end{aligned} \tag{*3}$$

となり，与式が得られる．

逆行列の微分 行列と逆行列の積は単位行列 $A(t)^{-1}A(t) = I$ である．この両辺を微分すると，(*3) より，

$$\{A(t)^{-1}\}^{\cdot}A(t) + A(t)^{-1}\dot{A}(t) = \mathbb{O} \implies \{A(t)^{-1}\}^{\cdot} = -A(t)^{-1}\dot{A}(t)A(t)^{-1} \tag{*4}$$

となって，与式を得る．証明終り．

10

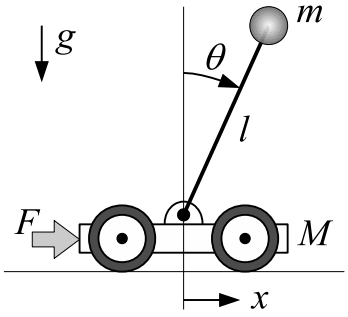
棒立て制御への応用

これまでの学習内容を応用して，自律型ロボットに「棒立て遊び」をさせよう．

10.1　力学モデル

10.1.1　倒立振子

ヒトやロボットの平衡機能を模した，次のカラクリを考える．



摩擦の無い質量 M の台車に，質量 m，長さ l の単振り子を取り付け，台車に推進力 F を発生させる (g は重力加速度)．このカラクリを (台車型の) **倒立振子** (inverted pendulum) という．ここで，F を制御入力として，振り子を倒立状態に安定化する問題を考える．状態量として，台車の水平変位 x と，振り子の倒れ角 θ をとる．

10.1.2　運動方程式 (非線形状態方程式)

倒立振子の運動方程式は次式で与えられる．導出方法は **A10 節** p79 に示す．

$$\begin{bmatrix} M+m & ml\cos\theta \\ \cos\theta & l \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} ml\dot{\theta}^2\sin\theta \\ g\sin\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} F \tag{10.1}$$

1 階化するために，加速度について解くと，

$$\begin{bmatrix}\ddot{x}\\ \ddot{\theta}\end{bmatrix} = \begin{bmatrix}M+m & ml\cos\theta\\ \cos\theta & l\end{bmatrix}^{-1}\left(\begin{bmatrix}ml\dot{\theta}^2\sin\theta\\ g\sin\theta\end{bmatrix} + \begin{bmatrix}1\\0\end{bmatrix}F\right)$$
$$= \frac{1}{\Delta}\begin{bmatrix}ml^2\dot{\theta}^2\sin\theta - mlg\cos\theta\sin\theta\\ -ml\dot{\theta}^2\cos\theta\sin\theta + (M+m)g\sin\theta\end{bmatrix} + \frac{1}{\Delta}\begin{bmatrix}l\\ -\cos\theta\end{bmatrix}F$$

となる．ただし，$\Delta = Ml + ml(1-\cos^2\theta)$ は行列式である．状態量を $\boldsymbol{x} = (x_1, x_2, x_3, x_4)^T \equiv (x, \dot{x}, \theta, \dot{\theta})^T$ とおいて 1 階化すると，次のような非線形制御系が得られる．

$$\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix}^{\cdot} = \begin{bmatrix}x_2\\ \{ml^2x_4^2\sin x_3 - mlg\cos x_3\sin x_3\}/\Delta\\ x_4\\ \{-mlx_4^2\cos x_3\sin x_3 + (M+m)g\sin x_3\}/\Delta\end{bmatrix} + \begin{bmatrix}0\\ l/\Delta\\ 0\\ -\cos x_3/\Delta\end{bmatrix}u(t),$$

$$\Delta = Ml + ml(1-\cos^2 x_3), \quad u(t) \equiv F \tag{10.2}$$

台車の推進力 F を制御入力 $u(t)$ とした．

実習 10.1 **Code 16** を実行し，制御無し $u(t) = 0$ の倒立振子の自由運動を観察せよ．

10.2 制御モデル

10.2.1 状態フィードバック

倒立振子の不安定な自由運動を安定化するために，状態フィードバック，

$$u(t) = -K\boldsymbol{x} \quad = -\begin{bmatrix}K_1, K_2, K_3, K_4\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix} \tag{10.3}$$

を与える．問題はゲイン K を具体的にどのような数値に設定するかだ．やってみると分かるが，この程度の規模の問題でも，無作為に見つけるのは難しい．

10.2.2 制御モデル (線形化方程式)

このようなときに，**6.3 節** p44 の固有値設定や，**9.3 節** p71 の最適レギュレータが役に立つ．これらを使うために，非線形制御系 (10.2) を線形化する．一般には，**2.3 節** p12 の要領で線形化するが，この問題の非線形項は三角関数と 2 次式だけなので，

$$\sin x_i \approx x_i, \quad \cos x_i \approx 1, \quad x_i^2 \approx 0 \tag{10.4}$$

を直接代入して線形化してしまおう．これは，$x, \dot{x}, \theta, \dot{\theta}$ が十分小さいと仮定して，倒立・静止状態 $\boldsymbol{x} = \mathbb{O}$ まわりで線形化することを意味する．実際に (10.2) に代入する

と，$\Delta = Ml$ より，

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}^{\cdot} = \begin{bmatrix} x_2 \\ -mlgx_3/(Ml) \\ x_4 \\ (M+m)gx_3/(Ml) \end{bmatrix} + \begin{bmatrix} 0 \\ l/(Ml) \\ 0 \\ -1/(Ml) \end{bmatrix} u(t),$$

となり，整理すると，

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + \boldsymbol{b}\,u(t): \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}^{\cdot} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{m}{M}g & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{M+m}{Ml}g & 0 \end{bmatrix}}_{A} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ -\frac{1}{Ml} \end{bmatrix}}_{\boldsymbol{b}} u(t) \quad (10.5)$$

という線形化方程式が得られる．この線形制御系 (10.5) は，$x, \dot{x}, \theta, \dot{\theta}$ が十分小さいとき，元の非線形制御系 (10.2) のよい近似になる．

以下では，この線形制御系 (10.5) を使ってゲイン K を求めるが，このような「制御系設計用のモデル」を**制御モデル**という．制御モデルは設計用の単純モデルなので，必ずしも，制御対象の動的システムとは一致しない．今回の設計でも，非線形の動的システム (倒立振子) に対して，線形の制御モデルを用いる．

10.2.3　可制御性の確認

一例として，$M = 2/3$, $m = 1/3$, $l = 1$, $g = 9.8$ のときの可制御性を調べる．可制御性行列 $U_c = [\boldsymbol{b}, A\boldsymbol{b}, A^2\boldsymbol{b}, A^3\boldsymbol{b}]$ のランクは，

```
> M=2/3; m=1/3; l=1; g=9.8;
octave:1> A=[0,1,0,0; 0,0,-m/M*g,0; 0,0,0,1; 0,0,(M+m)/(M*l)*g,0]
A =
    0.00000    1.00000    0.00000    0.00000
    0.00000    0.00000   -4.90000    0.00000
    0.00000    0.00000    0.00000    1.00000
    0.00000    0.00000   14.70000    0.00000
octave:2> bb=[0;1/M;0;-1/(M*l)]
bb =
   0.00000
   1.50000
   0.00000
  -1.50000
octave:3> Uc=[bb,A*bb,A*A*bb,A*A*A*bb]; #組み込み関数を使えば Uc=ctrb(A,bb)
octave:4> rank(Uc)
ans =  4
```

より，$\operatorname{rank} U_c = 4$ である．(10.5) の次元 4 と一致したので，制御モデル (10.5) は可制御である．

10.3 ゲイン調整

10.3.1 固有値設定によるゲイン調整

可制御なので，状態フィードバックによって固有値が設定可能である．

実習 10.2 **Code 17** を実行せよ．固有値 $s = -1, -2, -1 \pm 2i$ を実現するゲイン K が求まり，そのときの制御モデルの挙動が求まる．最後にゲインの値が保存される．

実習 10.3 **Code 16** の `KK=[0,0,0,0];` の行を，

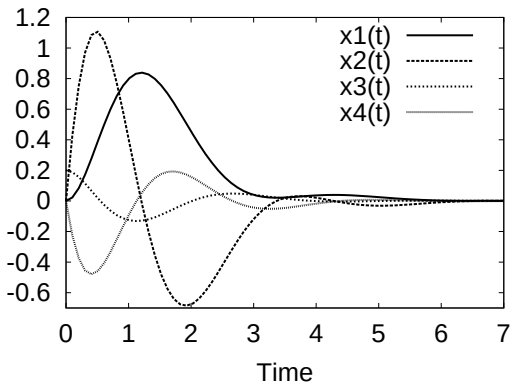
```
load invpGain.dat; KK=KKp;
```

に修正して実行せよ (先に **Code 17** を実行しておかないとエラーとなる)．実習 10.2 で求めたゲインで倒立振子 (10.2) p75 が動作する．

実習 10.4 **Code 16** の `for` から `endfor` までの内容を，

```
plot(tt,xx,[";x1(t);";";x2(t);";";x3(t);";";x4(t);"]);
xlabel("Time");drawnow;
```

に書き換えて実行せよ．倒立振子 (10.2) p75 の時間応答が，次のように求まる．



10.3.2 最適レギュレータによるゲイン調整

倒立振子 (10.2) p75 を線形化した制御モデル (10.5) p76 を使うと，最適レギュレータも計算できる．

実習 10.5 **Code 18** を実行せよ．重み $Q = \operatorname{diag}\{10, 3, 7, 3\}$, $R = 2$ に対する LQR ゲイン K が求まり，そのときの制御モデルの挙動が求まる．最後にゲインの値が保存される．

実習 10.6　**Code 16** の `KK=[0,0,0,0];` の行を，

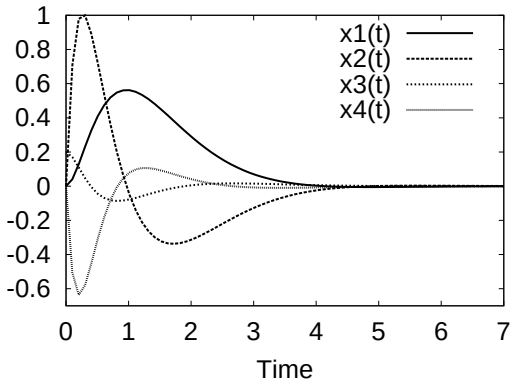
```
load invpLQR.dat; KK=KKopt;
```

に修正して実行せよ (先に **Code 18** を実行しておかないとエラーとなる)．実習 10.5 で求めたゲインで倒立振子 (10.2) p75 が動作する．

実習 10.7　**Code 16** の `for` から `endfor` までの内容を，

```
plot(tt,xx,[";x1(t);";";x2(t);";";x3(t);";";x4(t);"]);
xlabel("Time");drawnow;
```

に書き換えて実行せよ．倒立振子 (10.2) p75 の時間応答が，次のように求まる．



♣ 10 章の補足

A10 (10.1) p74 の導出

ラグランジュ形式の解析力学による導出方法を概要だけ示す．詳細は，例えば拙著 [8] の 12 章を見よ．

機構の位置と姿勢を表すのに最小限必要な変数を一般化座標という．この倒立振子では (x, θ) とする．一般化座標 (x, θ) で，倒立振子の全運動エネルギー T と，全ポテンシャルエネルギー U を書き下すと，それぞれ，

$$\begin{cases} T = \dfrac{M+m}{2}\dot{x}^2 + ml\dot{x}\dot{\theta}\cos\theta + \dfrac{ml^2}{2}\dot{\theta}^2 \\ U = MgG + mg(l\cos\theta + S) \quad (G, S \text{ は台車の重心と支点の高さ}) \end{cases} \tag{10.6}$$

となる．その差 $L = T - U$ を，次の公式 (オイラー・ラグランジュ方程式)，

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = F_q \quad (q = x, \theta) \tag{10.7}$$

に代入すると，(10.1) p74 が導出される．F_x は x 方向の外力，F_θ は θ 方向の外力 (トルク) を表す．

必要な偏微分を計算していくと，

$$\frac{\partial L}{\partial \dot{x}} = (M+m)\dot{x} + ml\dot{\theta}\cos\theta,\ \frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = (M+m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta,$$

$$\frac{\partial L}{\partial x} = 0,\ \frac{\partial L}{\partial \dot{\theta}} = ml\dot{x}\cos\theta + ml^2\dot{\theta},\ \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} = ml^2\ddot{\theta} + ml\ddot{x}\cos\theta - \underline{ml\dot{x}\dot{\theta}\sin\theta},$$

$$\frac{\partial L}{\partial \theta} = -\underline{ml\dot{x}\dot{\theta}\sin\theta} + mgl\sin\theta$$

となる．これらを (10.7) に代入して下線部の相殺を整理すると，

$$\begin{cases} x \text{ 方向：} (M+m)\ddot{x} + (ml\cos\theta)\ddot{\theta} - ml\dot{\theta}^2\sin\theta = F \quad \text{※ } x \text{ 方向の外力} \\ \theta \text{ 方向：} (ml\cos\theta)\ddot{x} + (ml^2)\ddot{\theta} - mlg\sin\theta = 0 \quad \text{※ } \theta \text{ 方向の外力=0} \\ \qquad\qquad \Longrightarrow \cos\theta\ddot{x} + l\ddot{\theta} - g\sin\theta = 0 \end{cases} \tag{10.8}$$

となる．これをベクトルで表したのが (10.1) p74 である．

11

演習 1 — Octave 入門 (60 分)

制御工学の研究者がよく用いる科学技術ソフトを使ってみよう．高機能版の関数電卓と思えば大したことはない．気楽に楽しんでほしい．

なお，本章を終えた人はすぐに次章に進むこと．次章は分量が多い！

11.1 演習の進め方 (グループワーク)

決して単独で進めないこと．2 名以上のグループで取り組み，分からないことがあれば，まずグループで解決せよ．それでも分からなければ，周辺のグループと共同で解決せよ．分かる学生がクラスに 1 人も居なければ，代表者を立てて教員に相談せよ．

11.2 端末を使う

コンピュータが人間の役に立つには，

(1) 人間が，コンピュータに何らかの指示を与える．

(2) その結果を，コンピュータが人間に返答する．

という一往復のキャッチボールが必要だ．これをマウスで行う方式を GUI(graphical user interface)，キーボードで行う方式を CUI (character user interface) という．

端末の起動　CUI を利用するには，まず「端末」と呼ばれるソフトを起動する．

コマンドの実行　次の操作の繰り返しである．

(1) 端末上の**プロンプト**と呼ばれる部分↓にコマンドを書いて，Enterキーを押す．

(2) コンピュータの返答を読む.

以下，プロンプトを略して `[... ~]$`と書く．例えば，端末のプロンプト

```
[... ~]$
```

に続けて，(l)(s)(Enter)とキーボードを打つと，

```
[... ~]$ ls          ←指示
Desktop              ←返答（実行環境によって異なる）
```

という出力を得る．ちなみに，`ls` というコマンドは，現在のフォルダにあるファイル一覧を表示するコマンドで，その返答が得たのが上の実行例である．

代表的なコマンドを，表 11.1 にまとめておく．

表 11.1 端末上の基本コマンド

exit	利用を終了する．
ls	フォルダにあるファイル一覧を表示する．
cd	フォルダを移動する．
	cd 自分のフォルダに戻る．
	cd .. 1つ上のフォルダに移動する．
	cd abc abc という名前のフォルダに移動する．
nautilus .	今いるフォルダーを開く．※ nautilus(スペース)(ピリオド)
gedit	エディターを起動する．
octave	数値解析ソフトを起動する．

コマンド補完機能 長いコマンドを途中まで打って(Tab)を打つと，その後を自動入力してくれる．例えば，(n)(a)(u)(Tab)と打つと，

```
[... ~]$ nautilus
```

までが自動的に入力される．続けて(Space)(.)(Enter)とすれば，GUI のフォルダーが開く．途中までの入力で複数のコマンドが対応するときは，(Tab)を連打すると候補を表示してくれる．

コマンド行編集機能 空のプロンプトで，(↑)(↓)を打つと，以前に打ち込んだコマンドを再利用できる．さらに，(←)(→)を使うと，行内を編集できる．

►► (矢印キーが効かない環境では) 代りに，(Ctrl-p)(Ctrl-n)および(Ctrl-b)(Ctrl-f)を使う．(Ctrl-p)は，(Ctrl)を押した状態で(p)を押すキー操作のこと．

11.3　ファイルを作り編集する

実習 11.1　表 11.1 を見ながら，次の操作を実行せよ．

(1)　フォルダの内容を確認しておく．

```
[... ~]$ ls          ※フォルダの内容
Desktop              ※システム用のフォルダ
```

(2)　新規にファイル aaa.txt を作成するには，まず，

```
[... ~]$ gedit
```

としてテキストエディタを起動する．試しに何か書き込んで，保存しよう．

再度フォルダの内容を確認すると，

```
[... ~]$ ls
Desktop/  aaa.txt
```

となり，確かに aaa.txt が出来てる．

(3)　既存のファイルを閲覧したり，修正するには，

```
[... ~]$ gedit aaa.txt    ←ファイルの閲覧・修正
```

のように，gedit でもう一度開けばよい．

11.4 Octave を使う

Octave というフリーソフト[1)]を使うと，様々な数値計算が簡単に実行できる．

実習 11.2 次の実行例を参考に，端末上で **Octave の起動と終了**を何度か繰り返せ．

```
[... ~]$ octave  ※ Octave 環境に入る
GNU Octave, version 3.0.5 (i686-pc-cygwi
(中略)
octave:1> ← Octave のプロンプト．指示待ち状態．
```

Octave のプロンプトに続けて，例えば (1)(+)(2)(Enter)とタイプする．

```
octave:1> 1+2
ans = 3          ← Octave が出した答え
octave:2> exit   ※ Octave から出る
[... ~]$         ※ 端末に戻った
```

実習 11.3 以下の実行例を，そのまま実行せよ．

11.4.1 四則演算・べき乗

```
octave:1> 1+2-3*4/5              ※四則演算
ans = 0.60000
octave:2> 2^3                      ※べき乗
ans = 8                          ※2の3乗
```

11.4.2 ベクトル・行列

```
octave:1> v=[1,2,3]              ※横ベクトル
v =
  1  2  3
octave:2> x=[1;2;3]              ※縦ベクトル
x =
  1
  2
  3
octave:3> x(2)           ※ベクトルの第 2 成分
ans = 2
octave:4> A=[-1,2,3; 4,5,6; 7,8,9]  ※行列
A =
  -1   2   3
```

[1)] 契約条項を遵守すれば無償で使える．市販の Matlab と互換性を持つ．http://www.octave.org/

```
   4   5   6
   7   8   9
octave:5> A(2,1)          ※行列の 2 行 1 列成分
ans = 4
octave:6> A(2,:)          ※ 2 行目の行ベクトル
ans =
  4  5  6
octave:7> A(:,1)          ※ 1 列目の列ベクトル
ans =
  -1
   4
   7
octave:8> A'                    ※行列の転置
ans =
  -1   4   7
   2   5   8
   3   6   9
octave:9> B=diag([1,2,-3])      ※対角行列
ans =
   1   0   0
   0   2   0
   0   0  -3
```

11.4.3　行列とベクトルの積

```
octave:1> A=[-1,2,3; 4,5,6; 7,8,9];  ※行列．行末の; は出力抑制
octave:2> x=[1;2;3];                 ※縦ベクトル
octave:3> A*x          ※行列×縦ベクトル
ans =
  12
  32
  50
octave:4> x' * x       ※横×縦ベクトル＝内積
ans = 14
octave:5> x * x'       ※縦×横ベクトル＝直積
ans =
   1   2   3     ※ 1*1 1*2 1*3
   2   4   6     ※ 2*1 2*2 2*3
   3   6   9     ※ 3*1 3*2 3*3
octave:6> [1,2,3].*[4,5,6]  ※成分どうしの積
ans =
  4  10  18
```

11.4.4 複素数・n 次方程式

```
octave:1> i        ※虚数単位 i = √−1
i = 0 + 1i
octave:1> s=3+4i              ※複素数
s = 3 + 4i
octave:2> real(s), imag(s)  ※実部，虚部．カンマ , で並べて一括実行
ans = 3
ans = 4
octave:3> abs(s), arg(s)     ※絶対値，偏角
ans = 5
ans = 0.92730
octave:4> roots([1,2,5])  ※ 2 次方程式 s^2 + 2s + 5 = 0
ans =
  -1.0000 + 2.0000i
  -1.0000 - 2.0000i

octave:5> roots([1,2,5,7])  ※ 3 次方程式 s^3 + 2s^2 + 5s + 7 = 0
ans =
  -0.19809 + 2.07975i
  -0.19809 - 2.07975i
  -1.60382 + 0.00000i
```

4 次以上も同様に解けるが，次数を増やすと計算時間が増大し，精度が落ちる．

11.4.5 固有値・固有ベクトル

```
octave:1> A=[-1,2,3; 4,5,6; 7,8,9];  ※行列．行末の ; は出力抑制
octave:2> inv(A)                     ※逆行列
ans =
  -0.50000   1.00000  -0.50000
   1.00000  -5.00000   3.00000
  -0.50000   3.66667  -2.16667
octave:3> eig(A)                     ※行列の固有値のみ
ans =
   15.91419
   -2.77850
   -0.13569
octave:4> [T,ss]=eig(A)              ※固有値・固有ベクトル
T =
  -0.20845  -0.88436   0.15857       ※固有ベクトル v1, v2, v3 を，
  -0.52873   0.10368  -0.79983       ※列ベクトルとする，
```

```
  -0.82280   0.45515   0.57890        ※基底変換行列 T = [v1, v2, v3]
ss =
   15.91419    0.00000    0.00000     ※固有値 s1, s2, s3 を,
    0.00000   -2.77850    0.00000     ※対角要素とする行列.
    0.00000    0.00000   -0.13569     ※すなわち diag([s1,s2,s3])
octave:5> poly(A)  ※ A の固有方程式 s^3 − 13s^2 − 46s − 6 = 0 の係数
ans =
    1.0000  -13.0000  -46.0000   -6.0000
```

11.4.6　ユーザー関数の定義

```
octave:1> function y=f(x) ※ 1 変数関数の定義
> y=x + cos(x);           ※ ; を忘れるな！
> endfunction               ←ここまで
octave:2> plot([0:0.1:5],f([0:0.1:5]));
octave:3> function y=g(m,c,k)  ※ 3 変数関数
> y=roots([m,c,k]);
> endfunction               ←ここまで
octave:4> g(1,0.2,2)
ans =
  -0.1000 + 1.4107i
  -0.1000 - 1.4107i
```

※ 2 行目の ; を忘れると

```
-- less -- (f)orward. (b)ack, (q)uit
```

となってしまうが，端末上で q を打つと復帰する．

11.4.7　関数のグラフ

```
octave:1> tt=linspace(0,2,5)  ※ 0〜2 を 5 分割した等差数列のベクトル
tt =
   0.00000   0.50000   1.00000   1.50000   2.00000
> sin(tt)
ans =
   0.00000   0.47943   0.84147   0.99749   0.90930
octave:3> [sin(0),sin(0.5),sin(1),sin(1.5),sin(2)]
ans =
   0.00000   0.47943   0.84147   0.99749   0.90930
octave:5> plot(tt,sin(tt))   ※ 2 次元プロット
```

11.5 プログラム・ファイルの実行

プログラム・ファイルの作成 試しに，1 章で使う **Code 1** を作る．そのために，**11.3 節** p82 の方法で，テキストエディタ (gedit) を起動し，次の内容を打ち込む．

pend.m

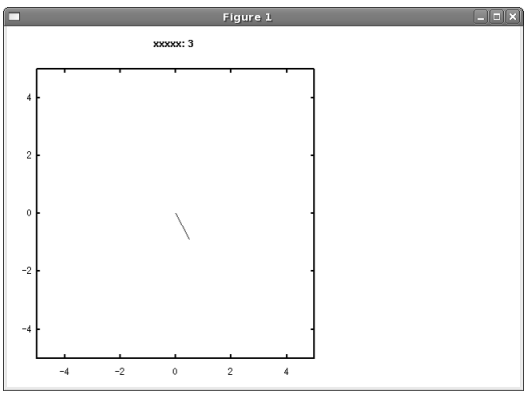
```
# pend.m     ここはコメント行. #〜行末を Octave は無視する
global m l c g;  #大域変数の宣言
m=1; l=1; c=1; g=9.8;
function dx=eqn(x,t)
 global m l c g; #関数の中から大域変数を参照する宣言
 dx(1) = x(2);
 dx(2) = -c/(m*l*l)*x(2)-g/l*sin(x(1));
endfunction
x0=[pi/5;0];              #初期値. pi は円周率
n=100;                    #時間のきざみ数
tt=linspace(0,10,n);      #時間 0〜10 の等比数列
xx=lsode("eqn", x0, tt); #常微分方程式"eqn"を解く
for i=1:n  #簡易アニメーション
 theta=xx(i,1);           #第 1 変数が角度
 plot( [0,sin(theta)], [0,-cos(theta)], 'linewidth', 4); #線分のプロット
 axis([-2,2,-2,2],'square');     #座標軸の設定
 title(sprintf("xxxxx: %d",i)); #タイトルの設定
 grid on; drawnow();
 if ( i==1 ) sleep(1);   #初回は 1 秒待機
 else sleep(0.1); endif  #この数値でアニメーション速度を調整
endfor
```

xxxxx のところを自分の学籍番号に換えて，ファイル名「pend.m」で保存する．

プログラム・ファイルの実行

```
[... ~]$ ls      ※プログラムファイルの所在を確認
Desktop/  pend.m              ※確かにある
[... ~]$ octave               ※ Octave 起動
GNU Octave, ... (略)
octave:1> source "pend.m"     ※実行
```

簡易アニメーションが表示される．



11.6　グラフ（表示画面）を印刷する

まず印刷したいグラフを表示する．例えば，

```
[... ~]$ octave                ※ Octave 起動
GNU Octave, ...（略）
octave:1> source "pend.m"      ※実行．グラフが表示される
```

グラフを表示した状態で，次のコマンドを実行する．

```
octave:2> print -deps graph.eps  ※ EPS ファイルができる
octave:3> print -dpng graph.png  ※ PNG ファイルができる
octave:4> exit                   ※ Octave 終了
[... ~]$ ls
Desktop/  graph.eps  graph.png  pend.m ※できてる
[... ~]$ evince graph.eps        ※ EPS ファイルの表示
[... ~]$ evince graph.png        ※ PNG ファイルの表示
```

終った人は次章に進むこと．次章は分量が多い！

12

演習 2 — 動的システムと固有値 (90 分)

1 章〜6 章の復習

次の実習を順次実行しながら，講義内容を復習せよ.

- **実習 1.1** p5
- **実習 1.2** p6
- **実習 2.2** p8
- **実習 3.3** p19
- **実習 3.4** p19
- **実習 4.1** p24
- **実習 4.2** p25
- **実習 5.1** p35
- **実習 6.1** p44
- **実習 6.2** p46

13

演習 3 — 最適レギュレータ (90 分)

7 章〜9 章の復習

次の実習を順次実行しながら，講義内容を復習せよ．

- **実習 8.1** p59
- **実習 9.1** p70
- **実習 9.2** p72

10 章の自習

次の実習を順次実行しながら，10 章の内容を自習せよ．

- **実習 10.1** p75
- **実習 10.2** p77
- **実習 10.3** p77
- **実習 10.4** p77
- **実習 10.5** p77
- **実習 10.6** p78
- **実習 10.7** p78

関連図書

[1] 長瀬道弘, 微分方程式, 裳華房, (1994).

[2] 小郷寛・美多勉, システム制御理論入門, 実教出版, (1979).

[3] 梶原宏之, 線形システム制御入門, コロナ社, (2003).

[4] 加藤寛一郎, 工学的最適制御, 東京大学出版会, (1991).

[5] 大関清太・遠藤博, 例題と演習でマスターする線形代数, 森北出版, (2008).

[6] 加藤寛一郎, 最適制御入門, 東京大学出版会, (1996).

[7] 吉川恒夫・井村順一, 現代制御論, 昭晃堂, (2009).

[8] 吉田勝俊, 短期集中：振動論と制御理論 (工学系の数学入門), 日本評論社, (2003).

索引