クォータニオン入門加筆

金谷一朗

2014 年 11 月 1 日

目次

記号一覧

実数　実数は小文字のローマ文字を使う．例: a, b, c.

複素数　複素数は小文字のギリシャ文字を使う．例: α, β, γ．ただし虚数単位は i で表す．

行列　行列は大文字を使う．例: A, B, C．ただし単位行列は 1 で表し，零行列は 0 で表す．

クォータニオン　クォータニオンは大文字のギリシャ文字を使う．例: Φ, Ψ, Σ．ただしクォータニオン単位は I, J, K で表す．

ベクトル　ベクトルは太字を使う．ベクトルの表現として複素数，行列，クォータニオンを使うことがあるが，全て小文字のローマ文字を使う．例: $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}$．基底ベクトルは $\boldsymbol{e}$ で表す．

スピノール　スピノールは太字を使い，全て小文字のギリシャ文字を使う．例: $\boldsymbol{\phi}, \boldsymbol{\chi}, \boldsymbol{\psi}$.

作用素　ベクトルに対する作用素は太字を使う．作用素の表現として複素数, 行列, クォータニオンを使うことがあるが，全て大文字のローマ文字を使う．例: $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$.

添字　添字は i, j, k を用いる．

集合　集合は太字の非イタリックのローマ文字を用いる．例: $\mathbf{A}, \mathbf{B}, \mathbf{C}$.

関数　関数はローマ字の小文字で表す．例： f, g, h.

微小量と無限小量　微小な量には接頭辞として Δ を使う．例: Δt．無限小量には接頭辞として δ を使う．例: δt.

その他　ネイピア数は e で表す．クロネッカーのデルタは δ で表す．円周率は π で表す．パウリ行列は σ で表す．

第 0 章

オリジナル版の内容のまとめ

0.1 実数・複素数・クォータニオン — 数

0.1.1 実数

C++ 言語では **double** 型に単項プラス，単項マイナス，和，差，積，商の 6 個の演算子が定義されている．これを「**double** 型は**数としてのインタフェース**を持つ」と言う．

数としてのインタフェースは実際には次のリストに集約される．

和の演算子 $a+b$ の $+$ 演算子．C++ 言語の和演算子．

零元（ゼロ，和の単位元） $0+a=a+0=a$ であるような 0．C++ 言語のリテラル `0`.

負元（和の逆元） a に対して $-a+a=0$ となるような $-a$．C++ 言語の単項マイナス．

積の演算子 $a\cdot b$ の $\cdot$ 演算子．普通は省略される．C++ 言語の積演算子．

単位元（イチ） $1a=a1=a$ であるような 1．C++ 言語のリテラル `1.0`.

逆元 a に対して $a^{-1}a=1$ であるような a^{-1}．C++ 言語ではデフォルトで用意されていないがラムダ式 `[](`**double** x`) { `**return** `1.0`$/x$; } を用いて容易に実装可能である．

C++ 言語の double 型の元になっている**実数**は上述のインタフェースを持つ．上述の 6 個のインタフェースは

和　演算子，単位元，逆元

積　演算子，単位元，逆元

という 3 個ずつのインタフェースに分類できる．

和と積にはそれぞれ次の関係がある．

$$\begin{aligned} abc &= (ab)c && (1)\\ &= a(bc) && (2)\\ a+b+c &= (a+b)+c && (3)\\ &= a+(b+c) && (4) \end{aligned}$$

これを**結合律（結合則）**と呼ぶ．

和と積が混在した場合は常に積が優先される．

$$ab + c = (ab) + c \tag{5}$$

和と積の間には次の関係が成り立つ．

$$a(b + c) = ab + ac \tag{6}$$
$$(a + b)c = ac + bc \tag{7}$$

これを**分配律（分配則）**と呼ぶ．

零元（ゼロ，和の単位元）と任意の元との積は常に零元である．

$$0a = a0 = 0 \tag{8}$$

0.1.2　複素数

実数に限らず，**複素数**も上述の 6 個のインタフェース，結合律，分配律に従う．複素数とは実数単位 1 の時数倍と虚数単位 i の実数倍との和である．a, b を実数とすると，$\alpha = 1a + ib$ が複素数の一般形である．

虚数単位は次の性質を持つ．

$$i^2 = -1 \tag{9}$$

複素数は数としてのインタフェースに加えて次のインタフェースを持つ．

共役複素数　ある複素数 α が $\alpha = 1a + ib$ であるとき $\alpha^* \equiv 1a - ib$ なる α^* を α の共役複素数と呼ぶ．

複素数のノルム　ある複素数 α について，

$$\|\alpha\| \equiv \sqrt{\alpha^* \alpha} \tag{10}$$

を α のノルムと呼ぶ．ノルムは「大きさ」という概念に近い．

複素数 α の逆数（逆複素数）α^{-1} は次のように求めることが出来る．

$$\alpha^{-1} = \frac{\alpha^*}{\|\alpha\|^2} \tag{11}$$

0.1.3　クォータニオン

$\Phi = 1a + Ib + Jc + Kd$ なる数 Φ を**クォータニオン**（四元数）と呼ぶ．ただし I, J, K はそれぞれクォータニオン単位であって，

$$I^2 = J^2 = K^2 = IJK = -1,\ IJ = -JI = K,\ JK = -KJ = I,\ KI = -IK = J \tag{12}$$

であるとする．

クォータニオンは数としてのインタフェースに加えて次のインタフェースを持つ．

共役クォータニオン　あるクォータニオン Φ が $\Phi = 1a + Ib + Jc + Kd$ であるとき $\Phi^* \equiv 1a - Ib - Jc - Kd$ なる Φ^* を Φ の共役クォータニオンと呼ぶ．

クォータニオンのノルム あるクォータニオン Φ について，

$$\|\Phi\| \equiv \sqrt{\Phi^* \Phi} \tag{13}$$

を Φ のノルムと呼ぶ．ノルムは「大きさ」という概念に近い．

クォータニオン Φ の逆数（逆クォータニオン）Φ^{-1} は次のように求めることが出来る．

$$\Phi^{-1} = \frac{\Phi^*}{\|\Phi\|^2} \tag{14}$$

0.2 行列 — もうひとつの数

0.2.1 連立線形方程式と行列

未知数 x に関する線形方程式

$$ax + b = 0 \tag{15}$$

の解は $x = -a^{-1}b$ である．

未知数 x_1, x_2 に関する連立線形方程式

$$a_{1,1}x_1 + a_{1,2}x_2 + b_1 = 0 \tag{16}$$
$$a_{2,1}x_1 + a_{2,2}x_2 + b_2 = 0 \tag{17}$$

の解について，新たな記号を発明して

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{18}$$

と書き直し，

$$A \equiv \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}, X \equiv \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, B \equiv \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, 0 \equiv \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{19}$$

とすると，未知数 x_1, x_2 に関する連立線形方程式は

$$AX + B = 0 \tag{20}$$

と書け，シンプルで美しく見える．演算規則をうまく調整すると，上述の連立線形方程式の解は $X = -A^{-1}B$ と書ける．このようにして作った $A, B, X, 0$ を**行列**と呼ぶ．

行列 A の逆行列 A^{-1} が存在するか否かの判定 (determinant) に**行列式**という演算子が使われる．行列 A の行列式は $\det A$ または $|A|$ と書く．

0.2.2 正方行列

各要素が実数からなり，行と列の大きさが等しい行列を**実正方行列**と呼ぶ．実正方行列を A とすると次のように書ける．

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1n} \\ a_{21} & a_{22} & & & & \\ \vdots & & \ddots & & & \\ a_{i1} & & & a_{ij} & & \\ \vdots & & & & \ddots & \\ a_{n1} & & & & & a_{nn} \end{bmatrix} \tag{21}$$

そこで実正方行列 A は，その要素と添字を使って $[a_{ij}]$ と書くこともある．

実正方行列には

- 和
- 零元
- 負元

が定義されている．行列 $[a_{ij}]$ と行列 $[b_{ij}]$ の和は

$$[a_{ij}] + [b_{ij}] \equiv [a_{ij} + b_{ij}] \tag{22}$$

であり，行列の零元（ゼロ行列）0 はすべての要素が 0 であるような行列である．

行列 $[a_{ij}]$ と行列 $[b_{ij}]$ の積も定義されており

$$[a_{ij}][b_{ij}] \equiv \sum_{k=1}^{n} [a_{ik} b_{kj}] \tag{23}$$

である．この定義から，積の単位元（単位行列）1 は

$$1 \equiv \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{bmatrix} \tag{24}$$

でなければならないことがわかる．単位行列 1 は $[\delta_{ij}]$ とも書く．デルタ記号を使うのは歴史的理由である．

0.2.3　直交行列とユニタリ行列

行列 $[a_{ij}]$ に対して，行と列を入れ替えた $[a_{ji}]$ は元の行列の**転置行列**と呼ばれる．転置行列は

$$[a_{ij}]^t \equiv [a_{ji}] \tag{25}$$

のような記号を使って表す．もし

$$[a_{ij}]^t = [a_{ij}] \tag{26}$$

であるならば，行列 $[a_{ij}]$ は**対称行列**である．もし

$$[a_{ij}]^t = -[a_{ij}] \tag{27}$$

であるならば，行列 $[a_{ij}]$ は**反対称行列**である．

実数の代わりに複素数を用いた正方行列を**複素正方行列**と呼ぶ．いま複素正方行列を $[\alpha_{ij}]$ で表すとき，その共役と転置を行った $[\alpha^*_{ji}]$ を**共役転置行列**と呼ぶ．共役転置行列を作る操作には特別な記号が割り当てられており，次のように表す．

$$[\alpha_{ij}]^\dagger \equiv [\alpha^*_{ji}] \tag{28}$$

もし

$$[a_{ij}]^\dagger = [a_{ij}] \tag{29}$$

であるならば，行列 $[\alpha_{ij}]$ は**エルミート行列**である．もし

$$[a_{ij}]^\dagger = -[a_{ij}] \tag{30}$$

であるならば，行列 $[\alpha_{ij}]$ は**反エルミート行列**である．

実正方行列 A について，もし

$$A^t A = 1 \tag{31}$$

であるならば，行列 A は**直交行列**である．複素正方行列 A について，もし

$$A^\dagger A = 1 \tag{32}$$

であるならば，行列 A は**ユニタリ行列**である．

0.3 行列による 2 次元の回転と内積

0.3.1 ベクトル

ベクトルには

- 和
- 零元（ゼロベクトル）
- 負元（逆ベクトル）

がある．またベクトルは実数倍が出来る．

ベクトル $\boldsymbol{p}$ のノルム $\|\boldsymbol{p}\|$ という量を定義できる．ノルムの定義は複数あるが，最もよく用いられているものは，ベクトルをユークリッド空間における位置とみなし，その位置の原点からの距離とする定義である．

0.3.2 内積

二つのベクトル $\boldsymbol{p}, \boldsymbol{q}$ の間に**内積**という演算が定義できる．内積は $\langle \boldsymbol{p}, \boldsymbol{q} \rangle$ で表す．ベクトルをユークリッド空間における位置 $\overrightarrow{OP}, \overrightarrow{OQ}$ とみなしたとき，二つのベクトルのなす角度を t として，

$$\langle \boldsymbol{p}, \boldsymbol{q} \rangle \equiv \|\boldsymbol{p}\| \|\boldsymbol{q}\| \cos t \tag{33}$$

と定義するのが，最も一般的な内積の定義である．この定義に従えば，ベクトル $\boldsymbol{p}$ のノルム $\|\boldsymbol{p}\|$ は

$$\|\boldsymbol{p}\| = \sqrt{\langle \boldsymbol{p}, \boldsymbol{p} \rangle} \tag{34}$$

である．

幾何学的な座標系を導入すると便利なことが多々ある．座標系を表すベクトルを**基底ベクトル**と呼ぶ．基底ベクトルとしていま $\boldsymbol{e}_1, \boldsymbol{e}_2$ があるとする．

ベクトル $\boldsymbol{p}$ の**成分**を p_1, p_2 で表すと，

$$p_i = \langle \boldsymbol{p}, \boldsymbol{e}_i \rangle \tag{35}$$

である．ただし i は $1,2$ である．ベクトルは成分と基底ベクトルから次のように合成できる．

$$\boldsymbol{p} = \sum_{i=1}^{2} p_i \boldsymbol{e}_i \tag{36}$$

基底ベクトルの組として**正規直交系**を選ぶとは

$$\|\boldsymbol{e}_1\| = \|\boldsymbol{e}_2\| = 1 \tag{37}$$
$$\langle \boldsymbol{e}_1, \boldsymbol{e}_2 \rangle = 0 \tag{38}$$

を満たすような $\boldsymbol{e}_1, \boldsymbol{e}_2$ を選ぶということである．一般には

$$\langle \boldsymbol{e}_i, \boldsymbol{e}_j \rangle = \delta_{ij} \tag{39}$$

と書くことが多い．

0.3.3　ベクトルの回転

ベクトル $\boldsymbol{p}$ の正規直交系での成分 p_1, p_2 を行列風に

$$\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \tag{40}$$

と書くと便利なことがある．ベクトル $\boldsymbol{p}$ で表される位置（これを今後 $\overrightarrow{OP}$ としよう）を原点まわりに t 回転させた位置（これは $\overrightarrow{OP'}$ とする）のベクトル $\boldsymbol{p}'$ の成分は次のように計算出来る．

$$\begin{bmatrix} p_1' \\ p_2' \end{bmatrix} = \begin{bmatrix} \cos t & -\sin t \\ \sin t & \cos t \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} \tag{41}$$

証明はオリジナル版を参照.

ここで行列

$$\boldsymbol{T}(t) \equiv \begin{bmatrix} \cos t & -\sin t \\ \sin t & \cos t \end{bmatrix} \tag{42}$$

を導入し，ベクトルと行列を意図的に混同すると

$$\boldsymbol{p}' = \boldsymbol{T}(t)\boldsymbol{p} \tag{43}$$

という簡潔な式が得られる．ここで行列だとか成分だとかを一切忘れて，ベクトル $\boldsymbol{p}$ に作用するものとして $\boldsymbol{T}(t)$ を捉える．この $\boldsymbol{T}(t)$ は**作用素**と呼ばれる．

0.4　複素数による 2 次元の回転

0.4.1　複素数で表す２次元ベクトル

正規直交系の基底ベクトルとは

$$\langle \boldsymbol{e}_i, \boldsymbol{e}_j \rangle = \delta_{ij} \tag{44}$$

を満たしてさえいればよい．もし内積の定義を都合よく選べば

$$\boldsymbol{e}_1 = 1,\ \boldsymbol{e}_2 = i \tag{45}$$

なる座標系を作ることが出来る．実際この座標系は**複素座標系**またはガウス座標系と呼ばれる．ここに内積の定義として

$$\langle \alpha, \beta \rangle \equiv \alpha^* \beta \tag{46}$$

を採用した．

0.4.2 回転

複素座標系における回転の作用素 $\boldsymbol{U}(t)$ は次の形を取る．

$$\boldsymbol{U}(t) = \cos t + i \sin t \tag{47}$$

ベクトル $\boldsymbol{p}$ を回転させるとは，作用素 $\boldsymbol{U}(t)$ は次のように左から掛けることで表現される．

$$\boldsymbol{p}' = \boldsymbol{U}(t)\boldsymbol{p} \tag{48}$$

オイラーの公式

$$\exp it = \cos t + i \sin t \tag{49}$$

を用いると，回転 $\boldsymbol{U}(t)$ は

$$\boldsymbol{U}(t) = \exp it \tag{50}$$

とさらに簡潔に書ける．

0.4.3 ベクトルと行列と複素数の関係

２次元ベクトルが行列でも複素数でも書けるのは，基底ベクトルの取り方次第だからである．基底ベクトルに正規直交系を選ぶと便利であった．正規直交系とは基底ベクトル $\boldsymbol{p}_i$ が

$$\langle \boldsymbol{e}_i, \boldsymbol{e}_j \rangle = \delta_{ij} \tag{51}$$

でありさえすればよく，内積をうまく定義してやれば自由に基底ベクトルを選べる．

行列スタイルを採用して

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{52}$$

としても良かったし，複素数スタイルを採用して

$$\boldsymbol{e}_1 = 1, \boldsymbol{e}_2 = i \tag{53}$$

としても良かった．どちらかと言えば複素数スタイルのほうが数としてのインタフェースを使えるので優れているとは言える．そこで数としてのインタフェースを保ちつつ行列も使えないかと考えると

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{54}$$

という基底ベクトルも良いことに気づくだろう．この場合 $\boldsymbol{e}_1$ のほうは単位行列 1 と同じであるので，もうひとつの $\boldsymbol{e}_2$ のほうを虚数単位 i に対応させて

$$i' \equiv \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{55}$$

と名づけても構わない．

0.5　行列による 3 次元の回転と外積

0.5.1　外積

2 次元のユークリッド空間を 3 次元に拡張するのはわけないことだ．とりわけ行列スタイルであればほとんど自動的に

$$\boldsymbol{e}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}, \boldsymbol{e}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}, \boldsymbol{e}_3 = \begin{bmatrix}0\\0\\1\end{bmatrix}, \tag{56}$$

を採用すれば良いことがわかる．

ここで，3 次元空間で非常にうまくいくトリックを導入する．次に述べる**外積**という演算を 3 次元ベクトル同士に定義する．

$$\boldsymbol{r} = \boldsymbol{p} \times \boldsymbol{q} \tag{57}$$

ここにベクトル $\boldsymbol{r}$ はベクトル $\boldsymbol{p}$ および $\boldsymbol{q}$ に直交し，そのノルムがベクトル $\boldsymbol{p}$ とベクトル $\boldsymbol{q}$ の張る平行四辺形に等しいとする．ベクトル $\boldsymbol{r}$ の向きは，右手で直交座標系を作り，ベクトル $\boldsymbol{p}$ を右手親指，ベクトル $\boldsymbol{q}$ を右手人差し指とした場合，右手中指の方向である．定義から，ベクトル $\boldsymbol{p}$ とベクトル $\boldsymbol{q}$ の角度を t としたときに

$$\|\boldsymbol{r}\| = \|\boldsymbol{p}\|\|\boldsymbol{q}\| \sin t \tag{58}$$

である．

外積は成分ごとに計算すると手っ取り早い．

$$\boldsymbol{p} \times \boldsymbol{q} = \begin{bmatrix}p_2q_3 - p_3q_2\\p_3q_1 - p_1q_3\\p_1q_2 - p_2q_1\end{bmatrix} \tag{59}$$

少しでもスタイリッシュにしたければ行列式を使うことは出来る．

$$\boldsymbol{p} \times \boldsymbol{q} = \det \begin{bmatrix}\boldsymbol{e}_1 & p_1 & q_1\\\boldsymbol{e}_2 & p_2 & q_2\\\boldsymbol{e}_3 & p_3 & q_3\end{bmatrix} \tag{60}$$

三重積

$$\boldsymbol{p} \times \boldsymbol{q} \times \boldsymbol{r} = \boldsymbol{q}\langle \boldsymbol{p}, \boldsymbol{r}\rangle - \boldsymbol{r}\langle \boldsymbol{p}, \boldsymbol{q}\rangle \tag{61}$$

は大切な関係である．

0.5.2　回転

3 次元ユークリッド空間の回転を考える．いま 3 軸まわりの回転だけを考えると，それは 2 次元の回転と変わらない．3 軸まわりの t 回転を $\boldsymbol{T}_3(t)$ とすると

$$\boldsymbol{T}_3(t) = \begin{bmatrix}\cos t & -\sin t & 0\\\sin t & \cos t & 0\\0 & 0 & 1\end{bmatrix} \tag{62}$$

である．同じく 2 軸まわりは

$$\boldsymbol{T}_2(t) = \begin{bmatrix} \cos t & 0 & \sin t \\ 0 & 1 & 0 \\ -\sin t & 0 & \cos t \end{bmatrix} \tag{63}$$

であり，1 軸まわりは

$$\boldsymbol{T}_1(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & -\sin t \\ 0 & \sin t & \cos t \end{bmatrix} \tag{64}$$

である．

これらの回転行列のうち，ふたつを組み合わせれば 3 次元の回転は全て表現できる．

0.5.3　もう一つの回転

回転の計算に外積を使うことも出来る．ベクトル $\boldsymbol{p}$ をベクトル $\boldsymbol{r}$ まわりに t 回転させたベクトル $\boldsymbol{p}'$ は

$$\boldsymbol{p}' = \boldsymbol{p}\cos t + \boldsymbol{r} \times \boldsymbol{p}\sin t + \boldsymbol{r}\langle \boldsymbol{r}, \boldsymbol{p}\rangle(1 - \cos t) \tag{65}$$

である．ただし $\|\boldsymbol{r}\| = 1$ を仮定した．証明はオリジナル版を参照．[*1]

0.6　クォータニオンによる 3 次元の回転

0.6.1　パウリ行列

2 次元の場合，正規直交系の基底ベクトルとして行列と複素数のどちらも選べた．3 次元の場合の複素数に相当する基底ベクトルはあるだろうか．次の複素行列は 3 次元の正規直交基底であることが知られている．

$$\sigma_1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \sigma_2 = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \sigma_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{66}$$

これらの行列は**パウリ行列**と呼ばれている．

パウリ行列は様々な良い性質を持つ．各々の行列の自乗は単位行列になる．

$$\sigma_1{}^2 = 1, \sigma_2{}^2 = 1, \sigma_3{}^2 = 1 \tag{67}$$

各々の行列の積は，残りの行列になる．

$$\sigma_1\sigma_2 = \sigma_3, \sigma_2\sigma_3 = \sigma_1, \sigma_3\sigma_1 = \sigma_2 \tag{68}$$

この性質は，すなわち通常の行列積がベクトルの外積として使えることを示す．

各々の行列の内積が 1 になるように，内積を定義することができる．内積の定義を

$$\langle A, B\rangle \equiv \frac{1}{2}\mathrm{tr}\left(A^t B\right) \tag{69}$$

ここに tr は対角成分の総和をとる演算子で，トレースと呼ばれる．この定義を用いると，パウリ行列の各々の内積は 0 になる．

$$\langle \sigma_1, \sigma_2\rangle = \langle \sigma_2, \sigma_3\rangle = \langle \sigma_3, \sigma_1\rangle = 0 \tag{70}$$

[*1] この式はロドリゲスの式と呼ばれている．

一方で，同じ行列同士の内積は 1 になる．

$$\langle \sigma_1, \sigma_1 \rangle = \langle \sigma_2, \sigma_2 \rangle = \langle \sigma_3, \sigma_3 \rangle = 1 \tag{71}$$

パウリ行列を 3 次元ベクトルの基底にすることができる．

$$\boldsymbol{p} = \sum_{i=0}^{3} p_i \sigma_i \tag{72}$$

パウリ行列を使った回転も可能であるが，その応用であるクォータニオンについて先に述べる．パウリ行列に関してはリー代数の章で再び述べる．

0.6.2　クォータニオン

パウリ行列に一工夫を加えると，クォータニオンが得られる．

$$\Phi = 1a + i\sigma_3 b + i\sigma_2 c + i\sigma_1 d \tag{73}$$

なる量 Φ はクォータニオンとしての性質をすべて持つ．また $i\sigma_3, i\sigma_2, i\sigma_1$ はクォータニオン単位の性質を持つ．そこで

$$\boldsymbol{e}_1 = i\sigma_3,\ \boldsymbol{e}_2 = i\sigma_2,\ \boldsymbol{e}_3 = i\sigma_1 \tag{74}$$

を基底ベクトルとして採用しよう．

ベクトル $\boldsymbol{p}$ をベクトル $\boldsymbol{r}$ まわりに t 回転させる演算子を $U(\boldsymbol{r}, t)$ とする．回転後のベクトル $\boldsymbol{p}'$ は演算子 $\boldsymbol{U}(\boldsymbol{r}, t)$ を用いて

$$\boldsymbol{p}' = \boldsymbol{U}^*(\boldsymbol{r}, t)\boldsymbol{p}\boldsymbol{U}(\boldsymbol{r}, t) \tag{75}$$

のように計算できる．ここに

$$\boldsymbol{U}(\boldsymbol{r}, t) = 1 \cos\frac{t}{2} + \boldsymbol{r} \sin\frac{t}{2} \tag{76}$$

である．証明はオリジナル版にある．

0.6.3　球面線形補間

省略.

0.7　テンソルとスピノール

ベクトル $\boldsymbol{p}$ を成分で p_i と書いてみる．回転の演算子 $\boldsymbol{T}$ も成分で T_{ij} と書いてみる．ベクトルの回転は

$$p'_j = \sum_{i=1}^{N} T_{ij} p_i \tag{77}$$

である．行列の書き方を用いると次のように書き直せる．

$$[p'_i] = [T_{ij}][p_i] \tag{78}$$

または

$$\boldsymbol{p}' = \boldsymbol{T}\boldsymbol{p} \tag{79}$$

このように変換される p_i を 1 階の**テンソル**と呼ぶ．次のように変換されるテンソルもあり，これを 2 階テンソルと呼ぶ．

$$P'_{kl} = \sum_{i=1}^{N}\sum_{j=1}^{N} T_{ik}T_{jl}P_{ik} \tag{80}$$

この式を行列を用いて書くと，行列の演算の非対称性から若干の工夫が必要になる．結局

$$[P'_{ij}] = [T_{ij}]^t[P_{ij}][T_{ij}] \tag{81}$$

または

$$\boldsymbol{P}' = \boldsymbol{T}^t\boldsymbol{P}\boldsymbol{T} \tag{82}$$

となる．

繰り返すと 1 階テンソルとは

$$\boldsymbol{p}' = \boldsymbol{T}\boldsymbol{p} \tag{83}$$

と変換される量である．2 階テンソルとは

$$\boldsymbol{P}' = \boldsymbol{T}^t\boldsymbol{P}\boldsymbol{T} \tag{84}$$

と変換される量である．

ここで 1 階テンソルはクォータニオンを使えば

$$\boldsymbol{p}' = \boldsymbol{U}^*\boldsymbol{p}\boldsymbol{U} \tag{85}$$

と書けたことを思い出そう．では

$$\boldsymbol{\phi}' = \boldsymbol{U}\boldsymbol{\phi} \tag{86}$$

なる量 $\boldsymbol{\phi}$ はあるだろうか．この $\boldsymbol{\phi}$ こそが**スピノール**である．

スピノールは行列で表示できる．

$$\boldsymbol{\phi} = \begin{bmatrix}\phi_1 \\ \phi_2\end{bmatrix} \tag{87}$$

共役スピノールを定義しておくと，スピノールの内積が計算しやすい．

$$\boldsymbol{\phi}^* = \begin{bmatrix}-\phi_2 & \phi_1\end{bmatrix} \tag{88}$$

こうしておけば，スピノールの内積は

$$\langle\boldsymbol{\phi},\boldsymbol{\chi}\rangle = \boldsymbol{\phi}^*\boldsymbol{\chi} \tag{89}$$

と演算できて都合が良い．展開すると

$$\begin{aligned}\langle\boldsymbol{\phi},\boldsymbol{\chi}\rangle &= \boldsymbol{\phi}^*\boldsymbol{\chi} && (90)\\ &= \begin{bmatrix}-\phi_2 & \phi_1\end{bmatrix}\begin{bmatrix}\chi_1 \\ \chi_2\end{bmatrix} && (91)\\ &= \phi_1\chi_2 - \phi_2\chi_1 && (92)\end{aligned}$$

であり，この量は回転に対して不変である．

スピノールは 2π 回転で符号が入れ替わる．

$$\boldsymbol{\phi} = -U(2\pi)\boldsymbol{\phi} \tag{93}$$

スピノールの掛け算の間にパウリ行列を挟むと楽しい．

$$p_i = \boldsymbol{\phi}^* \sigma_i \boldsymbol{\chi} \tag{94}$$

このようにして出来た p_i は 1 階 3 元テンソルとしての変換性を示す．いま

$$\sigma_0 \equiv 1 \tag{95}$$

を導入すると，

$$\langle \boldsymbol{\phi}, \boldsymbol{\chi} \rangle = \boldsymbol{\phi}^* \sigma_0 \boldsymbol{\chi} \tag{96}$$

$$p_i = \boldsymbol{\phi}^* \sigma_i \boldsymbol{\chi} \tag{97}$$

であるから $\langle \boldsymbol{\phi}, \boldsymbol{\chi} \rangle$ と p_i をひとつにして，4 元のテンソル p_i ただし $i = \{0, 1, 2, 3\}$ を考えても良い．

第 1 章

群・環・体とクォータニオン

この賞では「代数的構造」にスポットライトをあてる．まず回転が「群」という構造を作っていることを示し，群から「環」「体」とより複雑な構造を見ていく．最後に，実数，複素数，そしてクォータニオンが同じ「体」のメンバであることを確認する．

1.1 回転群

2 次元の回転の合成を考えてみよう．原点わまりの t 回転を $\boldsymbol{T}(t)$ とする．また回転 $\boldsymbol{T}(t)$ ともう一つの回転 $\boldsymbol{T}(u)$ の合成（連続させた回転）を $\boldsymbol{T}(u) \bullet \boldsymbol{T}(t)$ とする．

まず t 回転させて続いて u 回転させるのと，一気に $t+u$ 回転させるのは同じことなので，

$$\boldsymbol{T}(u) \bullet \boldsymbol{T}(t) = \boldsymbol{T}(t+u) \tag{1.1}$$

が成り立つ．この式の左辺をインタフェース，右辺を実装とみなしてみよう．

2 項演算子のままだと記述が不便なので，関数ぽく書いてみることにする．

$$\bullet(\boldsymbol{T}(u), \boldsymbol{T}(t)) = \boldsymbol{T}(t+u) \tag{1.2}$$

これは $\boldsymbol{A} \bullet \boldsymbol{B}$ を $\bullet(\boldsymbol{A}, \boldsymbol{B})$ と書いただけで，意味は同じである．

さて，回転を表す C++ クラスを `rot2d` としてみよう．クラス `rot2d` の二つのインスタンスから合成された新しい `rot2d` インスタンスを返すコンストラクタは次のようになるだろう．

```
class rot2d {
  private:
    double t;
  public:
    rot2d(double _t): t(_t) { }
    rot2d(const rot2d &a, const rot2d &b) {
      t = a.t + b.t;
    }
  // ...
};
```

このコンストラクタ `rot2d(const rot2d &, const rot2d &)` こそが合成の関数 $\bullet$ なのである．

さて，この合成の演算子 $\bullet$ には次のような結合律が成り立つ．

$$\boldsymbol{T}(t) \bullet (\boldsymbol{T}(u) \bullet \boldsymbol{T}(v)) = (\boldsymbol{T}(t) \bullet (\boldsymbol{T}(u)) \bullet \boldsymbol{T}(v) \tag{1.3}$$
$$= \boldsymbol{T}(t + u + v) \tag{1.4}$$

これは `rot2d(rot2d(a, b), rot2d(c))` と `rot2d(rot2d(a), rot2d(b, c))` が同じ意味のインスタンスを生成するのと同じことである．

ところで，回転には「何もしない」回転もある．原点周り 0 の回転だ．回転 $\boldsymbol{T}(0)$ は何もしないから，次式が成り立つ．

$$\boldsymbol{T}(0) \bullet \boldsymbol{T}(t) = \boldsymbol{T}(t) \bullet \boldsymbol{T}(0) = \boldsymbol{T}(t) \tag{1.5}$$

回転 $\boldsymbol{T}(0)$ は C++ コードでは `rot2d(0)` に相当する．次のようなデフォルトコンストラクタにしても良いかもしれない．

```
class rot2d {
  // ...
  public:
    rot2d(): t(0) { } // 何もしない回転を作る
  // ...
};
```

回転 $\boldsymbol{T}(t)$ には逆回転 $\boldsymbol{T}(-t)$ がある．

$$\boldsymbol{T}(-t) \bullet \boldsymbol{T}(t) = \boldsymbol{T}(0) \tag{1.6}$$

逆回転を表す特別な記号を発明しておこう．

$$\boldsymbol{T}^{-}(t) \equiv \boldsymbol{T}(-t) \tag{1.7}$$

左辺をインタフェース，右辺を実装と考えると，回転角度 t の符号を入れ替えるというディテールからインタフェースを隠すことができる．C++ での実装は次のようになるだろう．

```
class rot2d {
  // ...
  public:
    rot2d inverse() const { return rot2d(-t); }
  // ...
};
```

メンバ関数 `inverse` は自身の逆回転を新たに生成して返す．

ここで，回転 $\boldsymbol{T}$ 全体の集合 $\mathbf{T}$ を考えて見る．集合 $\mathbf{T}$ の元（要素）は無限にあり，

$$\mathbf{T} = \{\boldsymbol{T}(0), \boldsymbol{T}(t_1), \boldsymbol{T}(t_2), \dots\} \tag{1.8}$$

となる．このような集合 $\mathbf{T}$ と，その集合の元に対する合成の規則（または関係）である演算子 $\bullet$ について，結合律が成り立ち，「何もしない」元（単位元）があり，逆元がある場合，組み合わせ $(\mathbf{T}, \bullet)$ を**群**と呼ぶ．

この節で見たような回転を表す群を特別に**回転群**と呼ぶ．

1.2 群

改めて群について考える．

集合 $\mathbf{A}$ の元 $a_1, a_2 \in \mathbf{A}$ を考える．元 a_1, a_2 を引数に取り，集合 $\mathbf{A}$ の元を返す関数 $\bullet$ を考えよう．どういうわけか，人々は単に

$$\bullet(a_1, a_2) \tag{1.9}$$

と書くよりも

$$a_1 \bullet a_2 \tag{1.10}$$

と書くことを好む．また $\bullet$ のことを関数ではなくて**中置演算子**と呼ぶ．いずれにせよ，2引数関数を中置演算子として書くのは単なる**シンタックスシュガー**である．

さて，この演算子 $\bullet$ について，$a_1, a_2, a_3 \in \mathbf{A}$ のとき

$$a_1 \bullet a_2 \in \mathbf{A} \tag{1.11}$$

であり，結合律

$$a_1 \bullet (a_2 \bullet a_3) = (a_1 \bullet a_2) \bullet a_3 \tag{1.12}$$

が成り立ち，次のような単位元 ϵ があり，

$$\epsilon \bullet a_1 = a_1 \bullet \epsilon = a_1 \tag{1.13}$$

かつ任意の元 $a \in \mathbf{A}$ の逆元 a^- ただし

$$a^- \bullet a = \epsilon \tag{1.14}$$

が存在するとき，集合 $\mathbf{A}$ と演算子 $\bullet$ の組み $(\mathbf{A}, \bullet)$ を群と呼ぶのであった．

例えば，整数全体からなる集合 $\mathbb{Z}$ は加算演算子 $+$ との組み $(\mathbb{Z}, +)$ は群を作っている．確かめてみると，

1. $a, b \in \mathbb{Z}$ のとき $a + b \in \mathbb{Z}$ である
2. $a + (b + c) = (a + b) + c$ である
3. $0 + a = a + 0 = a$ なる単位元 0 がある
4. $a^- + a = 0$ なる逆元 a^- がある

である．ちなみに加算に関して言うと，a^- は通常の数学では $-a$ と書く．

加算に関して言えば $a + b = b + a$ と，二項演算子の前後を入れ替えても結果は等しい．これを**可換律**が成り立っていると呼び，可換律の成り立つ群を**可換群**または**加法群**または**アーベル群**と呼ぶ．

1.3　環と代数

群の条件をおさらいしておこう．群とは，ある集合 $\mathbf{A}$ とその集合の元に対して定義されている演算子（2 引数関数，関係）$\bullet$ について，

1. $a, b \in \mathbf{A}$ について $a \bullet b \in \boldsymbol{A}$　である
2. 演算子 $\bullet$ について結合律が成り立つ
3. 演算子 $\bullet$ に関する単位元がある
4. 演算子 $\bullet$ に関する逆元がある

の条件が揃っているものを指す．このうち，条件 1 から 3 までだけを満たすものを**モノイド**または**単位的半群**，条件 1 から 2 までだけを満たすものを**半群**と呼ぶ．

いま，集合 $\mathbf{A}$ にもうひとつの演算子 $\maltese$ が定義されていたとする．そして，もし組み合わせ $(\mathbf{A}, \bullet)$ が可換群であり，$(\mathbf{A}, \maltese)$ がモノイドであり，二つの演算子 $\bullet$ と $\maltese$ の間に次の関係（これを**分配律**と呼ぶ）

$$a \maltese (b \bullet c) = (a \maltese b) \bullet (a \maltese c) \tag{1.15}$$
$$(a \bullet b) \maltese c = (a \maltese c) \bullet (b \maltese c) \tag{1.16}$$

がある場合，組み合わせ $(\mathbf{A}, \bullet, \maltese)$ を**環**と呼ぶ．環はしばしば**代数**とも呼ばれる．（環以外にも代数と呼ばれる構造はあり，それが本書の最後に述べる束である．）

抽象的な話が続いたが，これまでの集合 $\mathbf{A}$ を例えば整数全体の集合 $\mathbb{Z}$ とし，演算子 $\maltese$ を乗算，演算子 $\bullet$ を加算とすると，実数全体の集合は加算，乗算に関して環になっていることがわかる．このとき，加算の単位元 ϵ は 0 であり，乗算の単位元 ε は 1 である．

1.4　体

ある代数的構造すなわち集合 $\mathbf{A}$ と演算子 $\bullet, \maltese$ の組み合わせ $(\mathbf{A}, \bullet, \maltese)$ について，

- 演算子 $\bullet$ に関して可換群をなす
- 演算子 $\bullet$ と $\maltese$ の間に分配律が成り立つ
- 演算子 $\maltese$ に関してモノイドをなす

であるとき，組み合わせ $(\mathbf{A}, \bullet, \maltese)$ を環と呼ぶのであった．ここで最後の条件を少し厳しくして，

- 演算子 $\bullet$ に関して可換群をなす
- 演算子 $\bullet$ と $\maltese$ の間に分配律が成り立つ
- 演算子 $\maltese$ に関して群をなす（ただし演算 $\bullet$ の単位元に関しては逆元が無くても良い）

とした場合，組み合わせ $(\mathbf{A}, \bullet, \maltese)$ を**体**と呼ぶ．

整数とその加算，乗算の組み合わせ $(\mathbb{Z}, +, \cdot)$ は環ではあるが，体ではない．これは乗法 $\cdot$ に関する逆元がないからである．例えば 2 は整数であり，その乗法単位元は 1 だが，$\frac{1}{2}$

は整数ではないので，$(\mathbb{Z}, +, \cdot)$ は体としての性質を満たしていない．

一方で，実数とその加算，乗算の組み合わせ $(\mathbb{R}, +, \cdot)$ は体である．むしろ，実数の性質を抽象化したものが体と言っても良いだろう．

複素数 $\mathbb{Z}$ とその加算，乗算の組み合わせ $(\mathbb{Z}, +, \cdot)$ もまた体である．そして，我らがクォータニオンもまた体である．

1.4.1 四元数体

クォータニオンは体のひとつである．クォータニオンのことを四元数体と言う．体の性質とは，代数的構造 $(\mathbf{A}, \bullet, \maltese)$ について

- 演算子 $\bullet$ に関して可換群をなす
- 演算子 $\bullet$ と $\maltese$ の間に分配律が成り立つ
- 演算子 $\maltese$ に関して群をなす（ただし演算 $\bullet$ の単位元に関しては逆元が無くても良い）

が成り立つことであった．

ここで，クォータニオン全体からなる集合 $\mathbb{Q}$ を考え，これの加算 $+$ と乗算 $\cdot$ からなる組み合わせ $(\mathbb{Q}, +, \cdot)$ は，

- 加算 $(+)$ に関して可換群をなす
- 加算 $(+)$ と乗算 $(\cdot)$ の間に分配律が成り立つ
- 乗算 $(\cdot)$ に関して群をなす（ただし 0 に関しては逆元が無い）

を満たしている．

実数，複素数が乗算に関して可換群を作っているのに対し，クォータニオンの乗算は可換ではない．これをもって，クォータニオンの作る体を**斜体**と呼ぶこともある．

これが，クォータニオンがちゃんと「数」の仲間であることの，理論的な裏付け（後付け）である．

第 2 章

リー代数

この章では，回転の「舞台裏」であるリー代数について紹介する．リー代数は無限小の回転を決定している構造のことである．2 次元の場合，2×2 行列でも複素数でも回転を表現できた．3 次元の場合，3×3 行列でもクォータニオンでも回転を表現できた．その背後を統一的に説明するのがリー代数なのである．

2.1 テイラー展開

複素関数 f があるとする．関数 f は無限回微分できるとし，関数 f を n 回微分したものを $f^{(n)}$ と書く．関数 f は次のように分解（展開）できることが知られている．

$$f(t) = \sum_{i=0}^{\infty} \frac{f^{(i)}(a)}{i!}(t-a)^i \tag{2.1}$$

$$= f(a) + f^{(1)}(a)t + \frac{1}{2}f^{(2)}(a)t^2 + \dots \tag{2.2}$$

これをテイラー展開または冪級数展開と呼ぶ．特に $a = 0$ の場合をマクローリン展開と呼ぶ．

三角関数 $\sin t$ や $\cos t$ は無限回の微分が可能なので，テイラー展開できる．結果だけ書くと次のようになる．

$$\sin t = t - \frac{1}{6}t^3 + \frac{1}{120}t^5 - \dots \tag{2.3}$$

$$\cos t = 1 - \frac{1}{2}t^2 + \frac{1}{24}t^4 - \dots \tag{2.4}$$

テイラー展開のご利益は，関数を加算乗算に分解できるだけでなく，パラメタ t が非常に小さい時に顕著になる．パラメタ t が非常に小さい事を強調するために Δt と書くことにしよう．パラメタ Δt が $|\Delta t| \ll 1$ とすると，$\Delta t^2 \simeq 0$ とみなせるから，先ほどの三角関数

$$\sin \Delta t \simeq \Delta t \tag{2.5}$$

$$\cos \Delta t \simeq 1 \tag{2.6}$$

とみなせるのである．

2.2　行列の指数関数

実数 a の指数関数 $\exp a$ は次のようにマクローリン展開できる．

$$\exp a = \sum_{i=0}^{\infty} \frac{a^i}{i!} \tag{2.7}$$

$$= 1 + a + \frac{1}{2}a^2 + \frac{1}{6}a^3 + \dots \tag{2.8}$$

そこで行列 A についても，指数関数 $\exp A$ を次のように定義する．

$$\exp A \equiv \sum_{i=0}^{\infty} \frac{A^i}{i!} \tag{2.9}$$

$$= 1 + A + \frac{1}{2}A^2 + \frac{1}{6}A^3 + \dots \tag{2.10}$$

ただし

$$A^0 = 1 \tag{2.11}$$

とした．

いま，行列 I を

$$I \equiv \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{2.12}$$

と定義しよう．この行列 I の指数関数 $\exp I$ はマクローリン展開できる．

$$\exp I \equiv \sum_{i=0}^{\infty} \frac{I^i}{i!} \tag{2.13}$$

$$= 1 + I + \frac{1}{2}I^2 + \frac{1}{6}I^3 + \dots \tag{2.14}$$

ただし 1 は単位行列のことである．

これで指数関数 $\exp I$ を単なる足し算掛け算に分解できたので，我々が普段知っている算術規則で計算できる．展開すると

$$\exp I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}^2 + \dots \tag{2.15}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} - \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \dots \tag{2.16}$$

のように定数の行列が得られるのだが，パラメタ t を導入して $\exp It$ を計算すると面白いことが起こる．

$$\exp It = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} t + \frac{1}{2}\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}^2 t + \frac{1}{6}\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}^3 t^3 \dots \tag{2.17}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} t - \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} t - \frac{1}{6}\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} t^3 + \dots \tag{2.18}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \left(1 - \frac{1}{2}t^2 + \dots\right) + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \left(t - \frac{1}{6}t^3 + \dots\right) \tag{2.19}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cos t + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \sin t \tag{2.20}$$

最後の変形で次の関係を使った．

$$\cos t = 1 - \frac{1}{2}t^2 + \frac{1}{24}t^4 - \dots \tag{2.21}$$

$$\sin t = t - \frac{1}{6}t^3 + \frac{1}{120}t^5 - \dots \tag{2.22}$$

まとめると，次の関係があることがわかる．

$$\exp It = 1\cos t + I\sin t \tag{2.23}$$

これは行列版オイラーの公式である．

2 次元の回転行列 $\boldsymbol{T}(t)$ を指数関数で書き直してみる．

$$\boldsymbol{T}(t) = \exp It \tag{2.24}$$

$$= 1\cos t + I\sin t \tag{2.25}$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cos t + \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \sin t \tag{2.26}$$

$$= \begin{bmatrix} \cos t & -\sin t \\ \sin t & \cos t \end{bmatrix} \tag{2.27}$$

この式はガウス平面を使った場合の回転と酷似していることがわかろう．ガウス平面では回転演算子 $\boldsymbol{U}(t)$ は

$$\boldsymbol{U}(t) = \exp it \tag{2.28}$$

$$= \cos t + i\sin t \tag{2.29}$$

であった．オイラーの公式は，実数軸からなる正規直交系と，ガウス平面とを結びつけているとも言える．

2.3　3 次元の回転の指数関数表示

行列の指数関数を使うと，回転行列も指数関数で作ることが出来る．

$$\boldsymbol{T}_i = \exp I_i' t \tag{2.30}$$

ここに

$$I_1' = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \tag{2.31}$$

$$I_2' = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{2.32}$$

$$I_3' = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.33}$$

である．行列 I_i' を 3 次元回転の**生成子**と呼ぶ．

ここで三角関数のテイラー展開を思い出そう．

$$\cos t = 1 - \frac{1}{2}t^2 + \frac{1}{24}t^4 - \dots \tag{2.34}$$

$$\sin t = t - \frac{1}{6}t^3 + \frac{1}{120}t^5 - \dots \tag{2.35}$$

もし Δt が十分小さい時，すなわち $|\Delta t| \ll 1$ のときには 2 次以上の項を無視できるから，

$$\cos \Delta t \simeq 1,\ \sin \Delta t \simeq \Delta t \tag{2.36}$$

であり，これを利用すると，

$$\boldsymbol{T}_1(\Delta t_1)\boldsymbol{T}_2(\Delta t_2)\boldsymbol{T}_3(\Delta t_3) = \begin{bmatrix} 1 & \Delta t_3 & -\Delta t_2 \\ -\Delta t_3 & 1 & \Delta t_1 \\ \Delta t_2 & -\Delta t_1 & 1 \end{bmatrix} \tag{2.37}$$

$$= 1 + \sum_{i=1}^{3} I'_i \Delta t_i \tag{2.38}$$

となり，回転演算子を線形に出来る．ここで再び生成子が顔を出すが，これこそが回転の本質を表す重要な行列なのである．それを次に見てみよう．

2.4 リー代数

ある行列 A の指数関数 $\exp At$ が群 $\mathbf{G}$ の元であるとする．すなわち

$$\exp At \in \mathbf{G} \tag{2.39}$$

であるとする．ここに t はパラメタである．

例えば行列 A として I を用いた $\exp It$ は 2 次元の回転行列である．この場合群 $\mathbf{G}$ は 2 次元の回転群である．

行列 A として I'_i ただし $i = \{1, 2, 3\}$ という集合を選んだ $\exp I'_i t$ は 3 次元の回転行列の集合になる．この場合群 $\mathbf{G}$ は 3 次元の回転群である．

行列 A の集合 $\mathbf{A}$ を群 $\mathbf{G}$ の**リー代数**と呼ぶ．

非常に小さな回転を考えてみよう．2 次元でも 3 次元でも同じなので，いまは 2 次元を考える．回転角を Δt として，回転を表す行列を $\boldsymbol{T}(\Delta t)$ とすると，

$$\boldsymbol{T}(\Delta t) = \exp I \Delta t \tag{2.40}$$

であった．回転前のベクトルを $\boldsymbol{p}$，回転後のベクトルを $\boldsymbol{p}'$ とすると

$$\boldsymbol{p}' = \boldsymbol{T}(\Delta t)\boldsymbol{p} \tag{2.41}$$

$$= \exp(I\Delta t)\boldsymbol{p} \tag{2.42}$$

なので，テイラー展開をすると

$$\boldsymbol{p}' = (1 + I\Delta t + (\Delta t \text{ の } 2 \text{ 次以上の項}))\boldsymbol{p} \tag{2.43}$$

$$\simeq \boldsymbol{p} + I\Delta t\boldsymbol{p} \tag{2.44}$$

である．ここで

$$\Delta \boldsymbol{p} \equiv \boldsymbol{p}' - \boldsymbol{p} \tag{2.45}$$

と定義すると，

$$\Delta \boldsymbol{p} = \Delta t I \boldsymbol{p} \tag{2.46}$$

である．だいぶ見慣れた形になってきただろうか．右辺の Δt を左辺の分母へ持っていくと次のようになる．

$$\frac{\Delta \boldsymbol{p}}{\Delta t} = I\boldsymbol{p} \tag{2.47}$$

上式は $\Delta t \to 0$ の極限を考えると理解しやすい．すなわち

$$\frac{d\boldsymbol{p}}{dt} = I\boldsymbol{p} \tag{2.48}$$

であり，形式的に微分演算子 d/dt を引き出すと

$$\frac{d}{dt}\boldsymbol{p} = I\boldsymbol{p} \tag{2.49}$$

となることから，これも全く形式的に

$$\frac{d}{dt} = I \tag{2.50}$$

と言える．これをもって，「行列 I が無限小回転の構造を決定している」と言う．

行列に I を選んだのは全くの任意であったので，上述の議論は 3 次元の生成子 I'_i についても成り立つ．

一般に $\exp At \in \mathbf{G}$ となるような行列 A の集合 $\mathbf{A}$ を集合 $\mathbf{G}$ のリー代数と呼ぶのであった．群 $\mathbf{G}$ に 2 次元の回転群を選べば，集合 $\mathbf{A}$ の元は I だけである．群 $\mathbf{G}$ に 3 次元の回転群を選べば，集合 $\mathbf{A}$ の元は I'_1, I'_2, I'_3 である．すなわち群 $\mathbf{G}$ のリー代数が無限小回転の構造を決定していると言える．無限小回転の構造と言うと長ったらしいが，ようするに回転角による微分と思えば良い．

2.5 回転の舞台裏

リー代数 $\mathbf{A}$ の元 A について，その実数倍 aA もまたリー代数 $\mathbf{A}$ の元である．また証明は面倒臭いのだが，二つの元 A_1, A_2 について，その和 $A_1 + A_2$ もまたリー代数 $\mathbf{A}$ の元である．つまり，リー代数はベクトル空間を張ることになる．

ベクトル空間には，基底ベクトルがつきものである．リー代数においては，生成子が基底ベクトルとして使える．

次に，**交換子積**という記号を導入する．交換子積とは，

$$[a, b] \equiv ab - ba \tag{2.51}$$

という演算子で，非可換の度合いを示すような演算子である．リー代数 $\mathbf{A}$ の元 A_1, A_2 について，その交換子積 $[A_1, A_2]$ もまたリー代数 $\mathbf{A}$ である．

リー代数 $\mathbf{A}$ の生成子を K_n とすると，リー代数はベクトル空間を張っているから

$$[K_i, K_j] = \sum_{k=1}^{n} f_{ijk} K_k \tag{2.52}$$

が言える．ただし f_{ijk} は実数である．この実数 f_{ijk} のことをリー代数の**構造定数**と呼ぶ．

3 次元の回転行列の生成子 I'_n について，その構造定数を調べてみよう．

$$[I'_i, I'_j] = -\sum_{k=1}^{3} \varepsilon_{ijk} I'_k \tag{2.53}$$

であり，この ε_{ijk} は**レビ・チビタ記号**である．レビ・チビタ記号とは，添え字によって $-1, 0, 1$ の異なる値をとる記号であり，直感的に言えば，添え字が $1, 2, 3$ や $2, 3, 1$ のよう

に「自然に」並んでいれば 1 を，添え字が $2,1,3$ や $1,3,2$ のように「不自然に」並んでいれば -1 を，同じ添え字が 2 回以上現れれば 0 を返す．具体的には

$$\varepsilon_{1,2,3} = \varepsilon_{2,3,1} = \varepsilon_{3,1,2} = 1 \tag{2.54}$$
$$\varepsilon_{2,1,3} = \varepsilon_{1,3,2} = \varepsilon_{3,2,1} = -1 \tag{2.55}$$
$$\varepsilon_{1,1,3} = \varepsilon_{1,2,2} = \cdots = 0 \tag{2.56}$$

である．(厳密には「自然な」を偶置換,「不自然な」を奇置換と呼ぶ.)

クォータニオンの場合の生成子はクォータニオン単位に $\frac{1}{2}$ をかけたものになる．$\frac{1}{2}$ をかけるのは生成子のノルムを 1 にするためのもので，本質的なことではない．クォータニオンによる回転の生成子を ς_n とすると

$$\varsigma_1 = \frac{1}{2}I \tag{2.57}$$
$$\varsigma_2 = \frac{1}{2}J \tag{2.58}$$
$$\varsigma_3 = \frac{1}{2}K \tag{2.59}$$

である．ただし I, J, K はクォータニオン単位である．

この生成子 ς_n について構造定数を調べてみると，見事に

$$[\varsigma_i, \varsigma_j] = -\sum_{k=1}^{3} \varepsilon_{ijk}\varsigma_k \tag{2.60}$$

となる．

つまり，行列による回転とクォータニオンによる回転は，同じ構造定数を持っていたのである．これが，2 種類の回転が本質的に同じであることの理由である．

第 3 章

束

プログラミング言語にはたいてい論理演算のための型や演算子がある．C++ 言語で言えば `bool`（ブール）型だ．ディジタルコンピュータは内部では論理演算しか扱えないから，ここまで紹介してきた数学も，実装レベルでは全て論理演算によてシミュレートされている．この論理演算に関しても深い数学構造がある．それが「ブール束」と呼ばれる構造だ．その構造に最後に触れることにしよう．

3.1 ブール代数

集合 $\mathbf{S}$ の元を s_1, s_2, s_3 とする．集合 $\mathbf{S}$ の**部分集合**とは，集合 $\mathbf{S}$ から任意個（0 個でも良い）の元を取り出して作った集合で，次のような集合たちである．

- 空集合 $\emptyset$
- 元が 1 個の集合 $\{s_1\}, \{s_2\}, \{s_3\}$
- 元が 2 個の集合 $\{s_1, s_2\}, \{s_2, s_3\}, \{s_3, s_1\}$
- 元（もと）の集合 $\mathbf{S}$

集合 $\mathbf{S}$ のすべての部分集合を集めた集合を集合 $\mathbf{S}$ の**冪集合**と呼び $\wp(\mathbf{S})$ と書く．今の例で言えば，

$$\wp(\mathbf{S}) = \{\emptyset, \{s_1\}, \{s_2\}, \{s_3\}, \{s_1, s_2\}, \{s_2, s_3\}, \{s_3, s_1\}, \mathbf{S}\} \tag{3.1}$$

である．

次に，集合の「足し算」を決めておく．通常の足し算と異なるので $\uplus$ という記号を使うことにしよう．集合 $\mathbf{A}$ と集合 $\mathbf{B}$ があるとして，

$$\mathbf{A} \uplus \mathbf{B} \equiv \mathbf{A} \cup \mathbf{B} - \mathbf{A} \cap \mathbf{B} \tag{3.2}$$

と定義する．なにやら足し算を定義するのに引き算が出てきて，ごまかしのように見えるかもしれないが，この方が定義が簡単になるというだけなので気にしないでもらいたい．先に演算子 $\cup$ と $\cap$ を説明しておくと，演算子 $\cup$ は二つの集合の和集合（合併，結び）を，演算子 $\cap$ は二つの集合の積集合（共通部分，交わり）を表す．集合の引き算 $(-)$ であるが，これは補集合を表す．補集合は $\mathbf{A} \setminus \mathbf{B}$ とも書く．いずれにせよ，第 1 項から第 2 項を取り除くと考えると良い．（計算機科学の言葉で言えば，演算 $\uplus$ はエクスクルーシブ・オアである．）

さて，集合 $\mathbf{A}, \mathbf{B}$ が冪集合 $\wp(\mathbf{S})$ の元すなわち $\mathbf{A}, \mathbf{B} \in \wp(\mathbf{S})$ であるとき，

- $\mathbf{A} \uplus \mathbf{B} \in \wp(\mathbf{S})$
- $(\mathbf{A} \uplus \mathbf{B}) \uplus \mathbf{C} = \mathbf{A} \uplus (\mathbf{B} \uplus \mathbf{C})$

が成り立つ．また空集合 $\emptyset$ は $\emptyset \uplus \mathbf{A} = \mathbf{A}$ のように単位元として振舞う．つまり，集合 $\wp(\mathbf{S})$ と演算子 $\uplus$ の組み $(\wp(\mathbf{S}), \uplus)$ はモノイドをつくる．

もう一つ演算子を導入しよう．といっても，すでにある積集合演算子 $\cap$ だ．この $\cap$ についても

- $\mathbf{A} \cap \mathbf{B} \in \wp(\mathbf{S})$
- $(\mathbf{A} \cap \mathbf{B}) \cap \mathbf{C} = \mathbf{A} \cap (\mathbf{B} \cap \mathbf{C})$

であり，また元の集合 $\mathbf{S}$ が $\mathbf{S} \cap \mathbf{A} = \mathbf{A}$ のように単位元として振舞う．つまり，集合 $\wp(\mathbf{S})$ と演算子 $\cap$ の組み $(\wp(\mathbf{S}), \cap)$ はモノイドをつくる．

演算子 $\uplus$ と $\cap$ について，次の規則（分配律）を設けることは差し支えない．

$$\mathbf{A} \cap (\mathbf{B} \uplus \mathbf{C}) = (\mathbf{A} \cap \mathbf{B}) \uplus (\mathbf{A} \cap \mathbf{C}) \tag{3.3}$$
$$(\mathbf{A} \uplus \mathbf{B}) \cap \mathbf{C} = (\mathbf{A} \cap \mathbf{C}) \uplus (\mathbf{B} \cap \mathbf{C}) \tag{3.4}$$

あとは逆元がそろえば，集合 $\wp(\mathbf{S})$ と演算子 $\uplus, \cap$ とで環ができるところなのだが，環と近い性質を持つので集合 $\wp(\mathbf{S})$ と演算子 $\uplus, \cap$ の組み $(\wp(\mathbf{S}), \uplus, \cap)$ を**半環**と呼ぶ．

集合 $\mathbf{A}$ に対して次のような集合 $\bar{\mathbf{A}}$ が存在する．

$$\bar{\mathbf{A}} \equiv \mathbf{S} - \mathbf{A} \tag{3.5}$$

集合 $\bar{\mathbf{A}}$ を**補元**と呼ぶ．補元は $\complement\mathbf{A}$ とも書く．補元はこれまでの逆元とは異なるが，補元を使うと次の性質が言える．

$$\bar{\mathbf{A}} \uplus \mathbf{A} = \mathbf{S} \tag{3.6}$$
$$\bar{\mathbf{A}} \cap \mathbf{A} = \emptyset \tag{3.7}$$

この組み合わせ（半環） $(\wp(\mathbf{S}), \uplus, \cap)$ は後述するように**ブール束**という環とは別の代数構造を形成している．束のことも慣例上「代数」と呼ぶので，ブール束は**ブール代数**とも呼ぶ．

3.2 ブール型

C++ には bool 型があり，**論理演算（ブール演算）**が定義されている．次のように使うことができる．

```
#include <iostream>
int main() {
  bool t = true;
  bool f = false;
  std::cout << "t && f == " << std::boolalpha << (t && f) << std::endl;
```

```
  return 0;
}
```

論理演算はプログラマにとってなじみ深いものであるが，もう一度おさらいしておこう．まず真を T，偽を F で表すことにする．論理積を $\wedge$ で，論理和を $\vee$ で表すことにしよう．それぞれ C++ では`&&`（または `and`）と`||`（または `or`）に相当する．

論理積は次の規則に従う．

$$T \wedge T = T \tag{3.8}$$
$$T \wedge F = F \tag{3.9}$$
$$F \wedge T = F \tag{3.10}$$
$$F \wedge F = F \tag{3.11}$$

論理和は次の規則に従う．

$$T \vee T = T \tag{3.12}$$
$$T \vee F = T \tag{3.13}$$
$$F \vee T = T \tag{3.14}$$
$$F \vee F = F \tag{3.15}$$

ここで注目して欲しいのは，T と F からなる集合 $\mathbb{B}$ を考えたときに，その元 $a, b \in \mathbb{B}$ の二項演算の結果もまた $\mathbb{B}$ の元である，つまり $a \wedge b \in \mathbb{B}$ かつ $a \vee b \in \mathbb{B}$ である．

次に，演算 $\wedge$ と演算 $\vee$ それぞれに結合律が成り立っているかどうかを見てみる．値が T と F しかないので総当たりすればすぐに結論が出て，$a, b, c \in \mathbb{B}$ のとき

$$a \wedge (b \wedge c) = (a \wedge b) \wedge c \tag{3.16}$$
$$a \vee (b \vee c) = (a \vee b) \vee c \tag{3.17}$$

であることがわかる．

また，論理積の単位元は T であり，論理和の単位元は F であることも読み取れる．

分配律も成り立つ．

$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c) \tag{3.18}$$
$$(a \vee b) \wedge c = (a \wedge c) \vee (b \wedge c) \tag{3.19}$$

ここまでくると，ブール代数と `bool` 型が無関係ではないことに気づくだろう．確かに，ブール代数の演算子 $\uplus$ と論理和 $\vee$ が，演算子 $\cap$ と論理積 $\wedge$ が対応しそうである．そこで，補元があるかどうか考えてみよう．

$$\bar{a} \vee a = T \tag{3.20}$$
$$\bar{a} \wedge a = F \tag{3.21}$$

となるような $\bar{a}$ があれば良く，これは a の論理否定（C++ で言えば`~`演算子または `not` 演算子）である．論理否定は通常 $\neg$ で表現するので，この場合は

$$\bar{a} \equiv \neg a \tag{3.22}$$

と書ける．

おさらいすると，集合 $\{T, F\}$ と，演算子 $\wedge, \vee, \neg$ は代数を作る．演算子が 1 個多いと思われるかもしれないが，実際のところ

$$a \vee b = \neg(\neg a \wedge \neg b) \tag{3.23}$$

という関係（**ド・モルガンの定理**と言う）があるので，演算子は実質 2 個である．

最後に，ブール代数における集合 $\mathbf{S}$ との関係を見ておこう．集合 $\mathbf{S}$ として，元が 1 個の集合を考えればなんでも良く，例えば $\mathbf{S} \equiv \{s_1\}$ としておこう．そうすると $\wp(\mathbf{S}) = \{\emptyset, \mathbf{S}\}$ となる．そこで，

$$T \equiv \mathbf{S} \tag{3.24}$$
$$F \equiv \emptyset \tag{3.25}$$

と決めておけば，`bool` 型の演算とはブール代数そのものと一致することがわかる．

3.3 束

集合 $\mathbf{A}$ と演算子 $\sqcap, \sqcup$ について，$a_1, a_2, a_3 \in \mathbf{A}$ としたときに，結合律

$$a_1 \sqcap (a_2 \sqcap a_3) = (a_1 \sqcap a_2) \sqcap a_3 \tag{3.26}$$
$$a_1 \sqcup (a_2 \sqcup a_3) = (a_1 \sqcup a_2) \sqcup a_3 \tag{3.27}$$

が成り立ち，演算 $\sqcap, \sqcup$ についてそれぞれ可換すなわち

$$a_1 \sqcap a_2 = a_2 \sqcap a_1 \tag{3.28}$$
$$a_1 \sqcup a_2 = a_2 \sqcup a_1 \tag{3.29}$$

であり，次の**吸収律**

$$a_1 \sqcap (a_1 \sqcup a_2) = a_1 \tag{3.30}$$
$$a_1 \sqcup (a_1 \sqcap a_2) = a_1 \tag{3.31}$$

が成り立つとき，$(\mathbf{A}, \sqcap, \sqcup)$ を**束**と呼ぶ．吸収律が成り立たないものを半束と呼ぶ．（ここで束は英語の lattice の訳である．なぜ断るかというと，実は別の数学用語である bandle も日本語では束と呼ぶのだ．）

ある束について，分配律すなわち

$$a_1 \sqcap (a_2 \sqcup a_3) = (a_1 \sqcap a_2) \sqcup (a_1 \sqcap a_3) \tag{3.32}$$
$$(a_1 \sqcap a_2) \sqcup a_3 = (a_1 \sqcap a_3) \sqcup (a_2 \sqcap a_3) \tag{3.33}$$

が成り立つとき，その束を**分配束**と呼ぶ．

またある束について，$\sqcap$ の単位元を 1，$\sqcup$ の単位元を 0 としたときに，元 a に対する補元 $\bar{a}$ すなわち

$$\bar{a} \sqcup a = 1 \tag{3.34}$$
$$\bar{a} \sqcap a = 0 \tag{3.35}$$

なる $\bar{a}$ が存在する場合，その束を**可補則**と呼ぶ．

補元のある分配束のことをブール束またはブール代数と呼ぶ．

`bool` 型の演算子 $\wedge, \vee$ を束の $\sqcap, \sqcup$ にそれぞれ対応させると，`bool` 型は結合律と可換律を満たしていることがわかる．さらに，吸収律をも満たしている．$a_1 \sqcap (a_1 \sqcup a_2) = a_1$ について，a_1, a_2 がそれぞれ T, F の場合について総当たりすると

$$T \wedge (T \vee T) = T \tag{3.36}$$
$$T \wedge (T \vee F) = T \tag{3.37}$$
$$F \wedge (F \vee T) = F \tag{3.38}$$
$$F \wedge (F \vee F) = F \tag{3.39}$$

であり，$a_1 \sqcup (a_1 \sqcap a_2) = a_1$ についても，a_1, a_2 がそれぞれ T, F の場合について総当たりすると

$$T \vee (T \wedge T) = T \tag{3.40}$$
$$T \vee (T \wedge F) = T \tag{3.41}$$
$$F \vee (F \wedge T) = F \tag{3.42}$$
$$F \vee (F \wedge F) = F \tag{3.43}$$

であるから，確かに `bool` 型は束の条件を満たしている．

さらに，`bool` 型は分配律

$$a_1 \wedge (a_2 \vee a_3) = (a_1 \wedge a_2) \vee (a_1 \wedge a_3) \tag{3.44}$$
$$(a_1 \vee a_2) \wedge a_3 = (a_1 \wedge a_3) \vee (a_2 \wedge a_3) \tag{3.45}$$

が成り立っており，否定演算子が補元を与えると考えると分配束かつ可補束である．すなわち，`bool` 型は厳密にブール束なのである．

分配律が成り立たない一般の束は，量子力学の理論的背景を支える強力なバックボーンになっている．この一般の束について多大な貢献をしたのは，ディジタルコンピュータの祖であるフォン・ノイマンである．