Algorithm Classifying Roots of Star-shaped Kac-Moody Root Systems

Toshio Oshima

Abstract. There is a correspondence between the imaginary roots of a star-shaped Kac-Moody root system and the spectral types of non-rigid Fuchsian ordinary differential equations. The norm of the root corresponds to the index of rigidity of the equation and the action of the Weyl group corresponds to the middle convolution of the equation. It is known that the Weyl group has finite orbits in the set of imaginary roots with a given norm. The purpose of this paper is to give an algorithm realized in a computer algebra which effectively gives representatives of the orbits. The representatives can be applied to the classification and construction of higher dimensional Painlevé type equations.

1. 星型 Kac-Moody ルート系

星形 Kac-Moody ルート系を復習しておく（cf. [**Kc**]）．添字の集合

$$I := \{0,\, (j,\nu)\,;\, j=0,1,\ldots,\ \nu=1,2,\ldots\}. \tag{1.1}$$

に対し, $\mathfrak{h}$ を基底

$$\Pi = \{\alpha_i\,;\, i\in I\} = \{\alpha_0,\ \alpha_{j,\nu}\,;\, j=0,1,2,\ldots,\ \nu=1,2,\ldots\} \tag{1.2}$$

で張られる無限次元ベクトル空間とし，以下のように定める．

$$I' := I\setminus\{0\},\qquad \Pi' := \Pi\setminus\{\alpha_0\}, \tag{1.3}$$

$$Q := \sum_{\alpha\in\Pi}\mathbb{Z}\alpha\ \supset\ Q_+ := \sum_{\alpha\in\Pi}\mathbb{Z}_{\ge 0}\alpha,\quad Q_- := -Q_+. \tag{1.4}$$

このとき $\alpha\in Q$ は

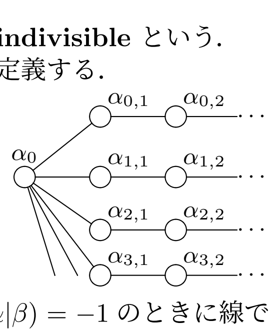
$$\alpha = n\alpha_0 + \sum_{j=0}^{p-1}\sum_{\nu=1}^{n_j-1} n_{j,\nu}\alpha_{j,\nu} \tag{1.5}$$

という表示をもつ（$n_{j,\nu}\neq 0$ となる $n_{j,\nu}$ は有限個）．ここで $n_{j,\nu}\in\mathbb{Z}$ であり，$\alpha\in Q_+$ は $n_{j,\nu}\in\mathbb{Z}_{\ge 0}$ という条件に対応する．

また，$\alpha\in Q$ が $\frac{1}{k}\alpha\notin Q$ $(k=2,3,\ldots)$ を満たすとき **indivisible** という．

さらに，$\mathfrak{h}$ 上の不定値対称双線型形式を以下のように定義する．

$$\begin{aligned}
&(\alpha|\alpha) = 2 \qquad (\alpha\in\Pi),\\
&(\alpha_0|\alpha_{j,\nu}) = -\delta_{\nu,1},\\
&(\alpha_{i,\mu}|\alpha_{j,\nu}) = \begin{cases} 0 & (i\neq j\quad\text{or}\quad |\mu-\nu|>1),\\ -1 & (i=j\quad\text{and}\quad |\mu-\nu|=1).\end{cases}
\end{aligned} \tag{1.6}$$



Π の各元を○で表し，$\alpha,\beta\in\Pi$ に対応する 2 つの○を $(\alpha|\beta)=-1$ のときに線でつないだダイアグラム（Dynkin 図式）で上の関係を表す．

星形 Kac-Moody ルート系とは，Π とその元の間の内積が上のように $(\ |\ )$ で定まっているもので，Π の元を**単純ルート**という．このルート系の **Weyl 群** W は $\alpha \in \Pi$ で定まる**単純鏡映**

$$s_\alpha\ :\ \mathfrak{h} \ni x \mapsto x - (x|\alpha)\alpha \in \mathfrak{h} \tag{1.7}$$

で生成される $\mathfrak{h}$ の線形変換の群と定義される．また

$$\sigma_{i,j}(\alpha_{i,\nu}) = \alpha_{j,\nu},\ \sigma_{i,j}(\alpha_{j,\nu}) = \alpha_{i,\nu},\ \sigma_{i,j}(\alpha_{k,\nu}) = \alpha_{k,\nu}\quad (k \neq i,j),\ \sigma_{i,j}(\alpha_0) = \alpha_0$$

で定まる線形変換 $\sigma_{i,j}$ で生成される群を S_∞ とおき，$\widetilde{W}$ を W と S_∞ で生成される群とする．$\widetilde{W}$ は $(\ |\ )$ を不変に保つことが容易に示される．

注意 1.1 ([**Kc**]). **実ルート**の集合 Δ^{re} は Π の W-軌道 $W\Pi$ となるが，星形 Kac-Moody ルート系のときは $W\alpha_0$ に等しい．一方

$$B := \{\gamma \in Q_+ \setminus \{0\}\,;\ \mathrm{supp}\,\gamma \text{ は連結で } (\gamma, \alpha) \le 0 \quad (\forall \alpha \in \Pi)\} \tag{1.8}$$

とおくと，**正の虚ルート**の集合 Δ_+^{im} は WB となる．また以下の定義を用いる．

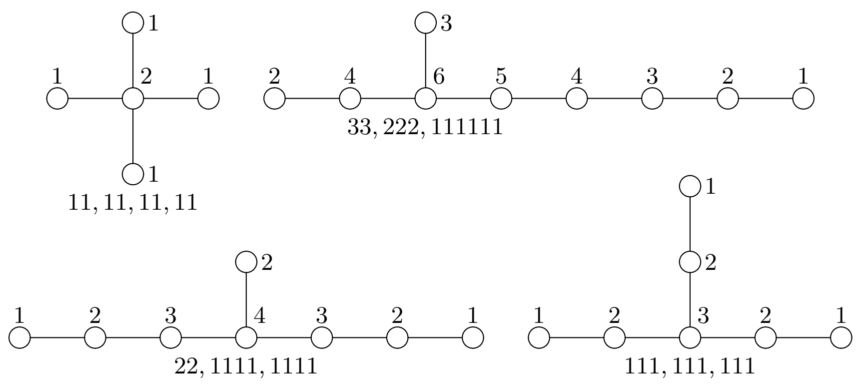
$$\mathrm{supp}\,\gamma := \{\alpha \in \Pi\,;\ n_\alpha \neq 0\} \quad \text{for} \quad \gamma = \sum_{\alpha \in \Pi} n_\alpha \alpha \in \mathfrak{h}. \tag{1.9}$$

このとき $\Delta_\pm^{re} := \Delta^{re} \cap Q_\pm$ とおくと $\Delta^{re} = \Delta_+^{re} \sqcup \Delta_-^{re}$，$WB \subset Q_+$ となる．また $\Delta_-^{im} = -\Delta_+^{im}$，$\Delta^{im} = \Delta_+^{im} \sqcup \Delta_-^{im}$ とおくと虚ルートの集合は $\Delta^{im} = \Delta_+^{im} \sqcup \Delta_-^{im}$，全ルートの集合は $\Delta^{re} \sqcup \Delta^{im}$，正ルートの集合 Δ_+ は $\Delta \cap Q_+ = \Delta_+^{re} \sqcup WB$ に一致する．$L \subset \Pi$ の部分集合 L が**連結**とは，$L_1 \cup L_2 = L$，$L_1 \neq \emptyset$，$L_2 \neq \emptyset$ ならば $v_j \in L_j$ で $(v_1|v_2) \neq 0$ となるものが存在することを意味する．また，$\alpha \in \Delta^{im}$ ならば $\alpha_0 \in \mathrm{supp}\,\alpha$ となる．

α, β を相異なる B の元とすると $W\alpha \cap W\beta = \emptyset$ が成り立つ．従って集合 $\{\alpha_0\} \sqcup B \sqcup (-B)$ は Δ における W-軌道の完全代表系を与える．

“有限型”の星型 Dynkin 図式に対応して次の図式で表される 4 つの星形拡張 Dynkin 図式 $\tilde{D}_4$，$\tilde{E}_8$，$\tilde{E}_7$，$\tilde{E}_6$ がある．なお，“有限型”とは対応する Weyl 群が有限群となることとする．

(1.10)



Q_+ の元 α で上記 Dynkin 図式に台をもつものを，(1.5) のように表したときの係数 $n_{j,\nu}$ を単純ルートを表す○の脇に記して上のように定め，対応する記号 $\tilde{D}_4$ などで示す（下部の数字の列は次節で解説）．これらの 4 つの Q_+ の元 α は，$(\alpha|\alpha) = 0$ を満たす B の元となる．逆に，$(\gamma|\gamma) = 0$ を満たす $\gamma \in B$ はこれら 4 つのいずれかの正の整数倍であることが知られている (cf. 注意 2.6 (i))．

注意 1.2. 整数 N に対し，

$$B_N := \{\alpha \in B \mid (\alpha|\alpha) = N\}. \tag{1.11}$$

とおくと

$$B_N \neq \emptyset \ \Leftrightarrow N \in -2\mathbb{Z}_{\ge 0}.$$

B_0 の元は上で与えた 4 種の indivisible な元の正整数倍となる．さらに，$N<0$ のときは B_N/S_∞ は有限集合となることが [**O1, O2**] で示された (cf. 注意 2.4).

補題 1.3 ([**O1**, Lemma 7.2])**.** $\alpha\in\Delta_+$ が $\operatorname{supp}\alpha\supsetneqq\{\alpha_0\}$ を満たすなら，表示 (1.5) において以下が成り立つ．

$$\begin{aligned} & n\ge n_{j,1}\ge n_{j,2}\ge n_{j,3}\ge\cdots\qquad (j=0,1,\ldots),\\ & n\le\sum n_{j,1}-\max\{n_{j,1},n_{j,2},\ldots\}. \end{aligned}\tag{1.12}$$

さらに $\alpha\in Q_+\setminus\{0\}$ が B に属するための必要十分条件は

$$2n_{j,\nu}\le n_{j,\nu-1}+n_{j,\nu+1}\quad (n_{j,0}:=n,\ j=0,1,\ldots,\ \nu=1,2,\ldots),\tag{1.13}$$

$$2n\le n_{0,1}+n_{1,1}+n_{2,1}+\cdots+n_{p-1,1}.\tag{1.14}$$

この *2* 条件はそれぞれ $(\alpha|\alpha_{j,\nu})\le 0$ と $(\alpha|\alpha_0)\le 0$ に対応する．

与えられた N に対して B_N を効果的に求めるアルゴリズムを与え，それをコンピュータプログラムとして実現することがこの論文の目的である．まず，ルートを自然数の分割の組として表し，数学や数理物理などへの応用に便利な形にする．

$\alpha\in\Delta_+$ は，$\operatorname{supp}\alpha\supset\{\alpha_0\}$ ならば，表示 (1.5) を用いて

$$m_{j,\nu}=n_{j,\nu-1}-n_{j,\nu}\quad (j=1,\ldots,p-1,\ \nu=1,\ldots,n_p)\tag{1.15}$$

とおくと $m_{j,\nu}\in\mathbb{Z}_{\ge0}$ となり

$$n_{j,k}=m_{j,k+1}+\cdots+m_{j,n_j}\quad (k=1,\ldots,n_j-1,\ j=0,\ldots,p-1),\tag{1.16}$$

$$n=m_{j,1}+\cdots+m_{j,n_j}\quad (j=0,\ldots,p-1)\tag{1.17}$$

であるから，自然数 n の分割の p 個の組 (1.17) が対応する．さらに次が成り立つ．

$$(\alpha|\alpha)=2n^2-\sum_{j=0}^{p-1}\Bigl(n^2-\sum_{\nu=1}^{n_j}m_{j,\nu}^2\Bigr).\tag{1.18}$$

この対応で，Q 上の W の作用をが引き起こす自然数の分割の組への作用は：

$$s_{j,\nu}:m_{j,\nu-1}\leftrightarrow m_{j,\nu}\quad (j=0,1,\ldots,\ \nu=1,2,\ldots),\tag{1.19}$$

$$s_0:m_{j,1}\mapsto m_{j,1}+\Bigl(2n-\sum_{j=0}^{p-1}(n-m_{j,1})\Bigr)\quad (j=0,1,\ldots).\tag{1.20}$$

鏡映変換 $s_{j,\nu}$ ($\nu=1,2,\ldots$) は，$m_{j,i}$ の添え字 i の置換を引き起こす．特に，$s_{p,1}$ は p を $p+1$ に変える．

注意 1.4. 次節の記号のもとで，分割の p 個の組 $\mathbf{m}$ が $\alpha\in B$ に対応するための必要十分条件は (2.1), (2.2), (2.8) を満たすこととなる．また，$n_{j,1}=n-m_{j,1}$ であるから，この 3 条件はそれぞれ (1.17), (1.13), (1.14) に対応する．

2. Fundamental tuples

$\mathbf{m}$ を自然数 n の p 個の分割の組とする．すなわち

$$\begin{aligned} \mathbf{m}&=(\mathbf{m}_0,\mathbf{m}_1,\ldots,\mathbf{m}_{p-1}),\\ \mathbf{m}_j&=(m_{j,1},\ldots,m_{j,n_j}) && (j=0,\ldots,p-1),\\ n&=m_{j,1}+\cdots+m_{j,n_j} && (m_{j,\nu}\in\mathbb{Z}_{>0}). \end{aligned}\tag{2.1}$$

簡単のため，$\mathbf{m}_j$ は**単調**，すなわち

$$m_{j,1}\ge m_{j,2}\ge\cdots\ge m_{j,n_j}>0\tag{2.2}$$

とし，さらに**非自明**，すなわち， $j=0,\ldots,p-1$ に対し

$$n_j>1\tag{2.3}$$

とする．また $\mathbf{m}$ を以下のように表す．

$$
\begin{aligned}
&\mathbf{m}_j = m_{j,1}m_{j,2}\cdots m_{p-1,n_{p-1}}, \quad m_{j\nu} = m_{j,\nu},\\
&\mathbf{m} = \mathbf{m}_0;\mathbf{m}_1;\cdots;\mathbf{m}_{p-1,n_{p-1}}\\
&\quad = m_{0,1}m_{0,2}\cdots m_{0,n_0};m_{1,1}\cdots m_{1,n_1};\cdots:m_{p-1,1}\cdots m_{p-1,n_{p-1}},\\
&m^k = \overbrace{m\cdots m}^{k}.
\end{aligned}
\tag{2.4}
$$

定義 2.1. n の分割の p 個の組 $\mathbf{m}$ に対し

$$\operatorname{ord}\mathbf{m} := n = m_{j,1} + \cdots + m_{j,n_j}, \tag{2.5}$$

$$\operatorname{codim}\mathbf{m}_j := n^2 - \sum_{\nu=1}^{n_j} m_{j,\nu}^2, \tag{2.6}$$

$$\operatorname{idx}\mathbf{m} := 2n^2 - \sum_{j=0}^{p-1}\operatorname{codim}\mathbf{m}_j \tag{2.7}$$

とおき，$\operatorname{idx}\mathbf{m}$ を $\mathbf{m}$ のリジッド指数という．$\mathbf{m}$ が **basic** とは

$$2n - \sum_{j=0}^{p-1}\Bigl(n - m_{j,1}\Bigr) \le 0 \tag{2.8}$$

で indivisible なことと定義する．なお，$m_{j,\nu} \in k\mathbb{Z} \quad (j = 0,\dots,p-1,\ \nu = 1,\dots,n_j)$ となる整数 $k > 1$ が存在しないとき **indivisible** という．$\mathbf{m}$ が basic であるか，または $\operatorname{idx}\mathbf{m}' \neq 0$ を満たす basic な $\mathbf{m}'$ と $k \in \mathbb{Z}_{>0}$ によって $\mathbf{m} = k\mathbf{m}'$ となるとき，$\mathbf{m}$ を **fundamental tuple** という．ここで，$\operatorname{idx}k\mathbf{m}' = k^2\operatorname{idx}\mathbf{m}'$ に注意．

$\operatorname{codim}\mathbf{m}_j$ は偶数なので， $\operatorname{idx}\mathbf{m}$ は偶数となることに注意．よって恒等式

$$
\begin{aligned}
&\Bigl(\sum_{j=0}^{p-1}(\operatorname{ord}\mathbf{m} - m_{j,1}) - 2\cdot\operatorname{ord}\mathbf{m}\Bigr)\cdot\operatorname{ord}\mathbf{m} + \sum_{j=0}^{p-1}\Bigl(\sum_{\nu=1}^{n_j}(m_{j,1} - m_{j,\nu})m_{j,\nu}\Bigr)\\
&\quad = -\operatorname{idx}\mathbf{m}
\end{aligned}
\tag{2.9}
$$

から，$\mathbf{m}$ が fundamental なら $p \ge 3$ で $-\operatorname{idx}\mathbf{m}$ は非負整数となることが分かる．

与えられたリジッド指数の fundamental tuple $\mathbf{m}$ を得るには，次の定理が基本となる．

定理 2.2. [**O2**, Proposition 7.13] $\mathbf{m}$ が *fundamental* ならば

$$\operatorname{ord}\mathbf{m} \le 3|\operatorname{idx}|\,\mathbf{m} + 6, \tag{2.10}$$

$$p \ge 3 \Rightarrow \operatorname{ord}\mathbf{m} \le |\operatorname{idx}\mathbf{m}| + 2, \tag{2.11}$$

$$p \le \tfrac{1}{2}|\operatorname{idx}\mathbf{m}| + 4. \tag{2.12}$$

系 2.3. $\mathbf{m}$ が *fundamental tuple* で $\operatorname{ord}\mathbf{m} > |\operatorname{idx}\mathbf{m}| + 2$ ならば $p = 3$ かつ

$$m_{0,1} + m_{1,1} + m_{2,1} = \operatorname{ord}\mathbf{m}. \tag{2.13}$$

fundamental tuple $\mathbf{m}$ を求めるにあたっては，$\mathbf{m}$ が**順序づけられている** (ordered) こと, すなわち以下を $j = 0,\dots,p-2$ に対して仮定してよい．

$$
\begin{aligned}
&\exists N \in \mathbb{Z}_{>0} \text{ with } m_{j,\nu} = m_{j+1,\nu} \quad (0 \le \nu < N) \ \text{ and } \ m_{j,N} \neq m_{j+1,N}\\
&\quad \Rightarrow\ m_{j,\nu} > m_{j,\nu+1}.
\end{aligned}
\tag{2.14}
$$

注意 2.4. 負のリジッド指数をもつ ordered fundamental tuples $\mathbf{m}$ は有限個となることが定理 2.2 から分かる．また上の系も (2.9) から分かる．

例 2.5. (i) [**O2**, Example 7.14] $m \in \mathbb{Z}_{>0}$ に対し

$$\begin{aligned} &D_4^{(m)} : m^2, m^2, m^2, m(m-1)1 && E_6^{(m)} : m^3, m^3, m^2(m-1)1 \\ &E_7^{(m)} : (2m)^2, m^4, m^3(m-1)1 && E_8^{(m)} : (3m)^2, (2m)^3, m^5(m-1)1 \end{aligned} \tag{2.15}$$

はリジッド指数 $2-2m$ をもつ fundamental な $2m$, $3m$, $4m$, $6m$ の分割の組となる．$E_8^{(m)}$, $D_4^{(m)}$ と $11, 11, 11, \cdots$ はそれぞれ (2.10), (2.11), (2.12) において等号を満たす．

(ii) [**Ko**, Corollary 6.3] $\operatorname{idx}\mathbf{m} = 0$ では，4 種の basic な組 $\mathbf{m}$ が存在する：

$$D_4^{(1)} = 1^2, 1^2, 1^2, 1^2 \qquad E_6^{(1)} = 1^3, 1^3, 1^3 \qquad E_7^{(1)} = 2^2, 1^4, 1^4 \qquad E_8^{(1)} = 3^2, 2^3, 1^6$$

(iii) [**O1**, Proposition 8] $\operatorname{idx}\mathbf{m} = -2$ では，13 種の fundamental tuples $\mathbf{m}$ が存在する：

$$\begin{array}{llll} 1^2, 1^2, 1^2, 1^2, 1^2 & 21, 21, 1^3, 1^3 & 2^2, 2^2, 2^2, 21^2 & 3^1, 2^2, 2^2, 1^4 \\ 21^2, 1^4, 1^4 & 32, 1^5, 1^5 & 2^21, 2^21, 1^5 & 3^2, 2^21^2, 1^6 \\ 2^3, 2^3, 2^21^2 & 4^2, 2^4, 2^31^2 & 4^2, 2^4, 2^31^2 & 4^2, 3^22, 1^8 \\ 6^2, 4^3, 2^51^2 & & & \end{array}$$

注意 2.6. (i) $\mathbf{m}$ を $\operatorname{idx}\mathbf{m} = 0$ となる ordered fundamental tuple とする．このとき (2.8) と (2.9) から $m_{j,\nu} = \frac{\operatorname{ord}\mathbf{m}}{n_j}$ および $\frac{\operatorname{ord}\mathbf{m}}{n_0} + \cdots + \frac{\operatorname{ord}\mathbf{m}}{n_{p-1}} = (p-1)\cdot\operatorname{ord}\mathbf{m}$ が分かる．このことから $(n_0, n_2, \ldots, n_{p-1})$ は，$(2,2,2,2)$，$(3,3,3)$，$(2,4,4)$，$(2,3,6)$ のいずれかで，$\operatorname{ord}\mathbf{m}$ は $n_0, \ldots, n_{p-1}$ の最小公倍数であることが示され，例 2.5 (ii) のリストが得られる．

(ii) リジッド指数が -4 及び -6 の fundamental tuples のリストは [**O2**, §13.1.2, §13.1.4] にあるが，-4 のリストでは $3^22, 3^22, 2^4$ が，-6 のリストでは $3^22, 3^22, 2^31^2$ と $3^22, 3^21^2, 2^4$ が抜けている．リジッド指数が -8 以下の fundamental tuples のリストは，数式処理 `Risa/Asir` のライブラリ [**O6**] の関数によって得られる．その関数のアルゴリズムを次節で述べる．

(iii) 自然数の分割において，9 より大きな数はアルファベットを用いた表示を用いることがある．たとえば，$10, 11, 12, \ldots, 16, 17, \ldots$ は $\mathtt{a}, \mathtt{b}, \mathtt{c}, \ldots, \mathtt{f}, \mathtt{g}, \ldots$ などと表す．11 は (11) のようにも記す．よって $\mathtt{bb9} = (11)(11)9$ は自然数 31 の分割の例．

注意 2.7. 自然数の分割の組 $\mathbf{m}$ を与えたとき，それがルートに対応するかどうかの判定は難しくない．実際，[**O6**] の関数 `chkspt( )` は，$\mathbf{m}$ がルートに対応するかどうか判定し，虚ルートならばリジッド指数やそれの W-軌道の代表元を B から返す．

3. Fundamental tuples を求めるアルゴリズム

与えられたリジッド指数をもつ fundamental な分割の組 (2.1) をすべて求めることを考えよう．(2.2) と (2.14) を仮定してよい．(2.10) と (2.12) によって有限個の分割の組に対して条件 (2.7) と (2.8) をチェックすればよい．

$\operatorname{idx}\mathbf{m} = -8$ の場合を考えてみよう．$\operatorname{ord}\mathbf{m} \le 30$ で $p \le 8$ が分かる．$\operatorname{ord}\mathbf{m} = 30$ ならば $\mathbf{m}$ が $E_8^{(8)}$ となることは [**O2**] から分かる．$\operatorname{ord}\mathbf{m} = 29$ とする．(2.12) から $p = 3$ である．自然数 29 の分割数は $p(29) = 4565$ であるから 29 の分割の順序づけられた 3 つの組，約 $4565^3/6 > 10^{10}$ 個についての (2.7) と (2.8) のチェックは，重い計算となる．実際には，両方を満たす $\mathbf{m}$ は存在しない．

この節では，これらの条件の効率的なチェックを行うアルゴリズムを与える．それを用いたコンピュータプログラムを実行すると，$\operatorname{idx}\mathbf{m} = -8$ の場合の 116 種の fundamental な組のリストは約 0.05 秒で得られた（Windows 11 で CPU i7-10700 のコンピュータでの例）．さらに，たとえばリジッド指数が -18 と -50 の場合では，それぞれ 884 種と 40617 種の fundamental な組のリストとなるが，それぞれ計算時間は約 1 秒あるいは 3 分であった．なお，$p(3\cdot 50 + 5) = 66493182097 \fallingdotseq 6.65 \times 10^{10}$ となるので，個々に条件をチェックするのは現実的でない．

数式処理 Risa/Asir [**Risa**] を用いてアルゴリズムの実装を考える．自然数の分割は自然数のリストで，分割の組は分割を表すリストのリストで表す：

$$\begin{aligned}\mathbf{m} &= \big[[m_{0,1},\dots,m_{0,n_0}],[m_{1,1},\dots,m_{1,n_1}],\dots,[m_{p-1,1},\dots,m_{p-1,n_{p-1}}]\big],\\ \mathtt{V}[j]\,[\nu-1] &= V_{j,\nu-1} = m_{j,\nu} \quad (j=0,1,\dots,p-1,\ \nu=1,2,\dots,n_j).\end{aligned} \tag{3.1}$$

ここでは前節の記号を用いている．$\mathbf{m}$ は単調で順序づけられているとする．また，プログラムでは以下の記号を用いる．

$$\begin{aligned}\mathtt{D} &= \operatorname{ord}\mathbf{m} = n,\\ \mathtt{Idx} &= |\mathrm{idx}|,\\ \mathtt{S} &:= \sum_{j=0}^{p}(\operatorname{ord}\mathbf{m} - m_{j,1}) - 2\cdot\operatorname{ord}\mathbf{m},\\ \mathtt{SS} &:= \sum_{j=0}^{p}\Big(\sum_{\nu=1}^{n_j}(m_{j,1}-m_{j,\nu})m_{j,\nu}\Big).\end{aligned} \tag{3.2}$$

このとき $\mathbf{m}$ が fundamental という条件は

$$\mathtt{S} \ge 0 \ \text{ and } \ \mathtt{S}+\mathtt{SS} = \mathtt{Idx} \tag{3.3}$$

であって，さらに $\mathtt{Idx}=0$ のときは $\mathbf{m}$ が indivisible という条件がつく．

ライブラリ os_muldif.rr [**O6**] に含まれる関数 spbasic( ) を解説する．

spbasic(Idx,D|str=1,pt=[k,ℓ])
:: リジッド指数 Idx，ランク D の ordered fundamental tuples $\mathbf{m}$ のリストを返す．
Idx が 0 のときは，indivisible なものを返す
D が 0 のときは，リジッド指数が Idx のものすべてのリストを返す
オプション pt=[k,ℓ] は，分割（特異点）の数が k 以上 ℓ 以下を意味する
オプション pt=[k,k] は pt=k としてもよい
オプション str=1 は，結果の分割の組 $\mathbf{m}$ を例 2.5 にあるような数字を表す文字列のリストとして返す． これらの表示は os_muldif.rr の中の関数 s2sp( ) で相互変換される．

自然数 n の分割は，和が n の非増加の正整数の列として表す．分割は辞書式順序で，分割の組も辞書式順序で大きい順に並べる．

$$\mathtt{aa431} > \mathtt{a99} > \mathtt{a981} \qquad \mathtt{541,3331} > \mathtt{532,433}$$

関数 spbasic(Idx,D) は，上の順序で分割の組 $\mathbf{m}$ が (3.3) を満たすかどうかチェックし，満たす組のリストを返す．定理 2.2 より，$\mathtt{D} > 3\cdot|\mathtt{Idx}|+6$ または $\mathtt{Idx}>0$ または Idx ならば求める組がないこと，さらに $\mathtt{D} > |\mathtt{Idx}|+2$ ならば $p=3$ で，一般には $3\le p\le \frac{1}{2}|\mathtt{Idx}|+4$ の場合をチェックすればよいことが分かる．よって，チェックすべきは有限個の組となるが，$|\mathtt{Idx}|$ や D が大きい場合はその個数は膨大で，不必要なチェックをできるだけ割けることが重要となる．チェックすべき組の大部分は $|\operatorname{idx}\mathbf{m}| > |\mathtt{Idx}|$ となっていることがキーポイントである．

spbasic( ) のコードは以下の 2 つに分けられる．

1. Introduction

1.1, 1.2, 1.3. 関数の引数をチェックし，$\mathtt{D}\ge 3\cdot|\mathtt{Idx}|+6$ のような自明な場合は答えを返す．

1.4. 定理 2.2 に基づいて，$p\le\mathtt{L}$ となる上限 L を求める．

1.5, 1.6. 条件 $\mathtt{SS}\ge 0$ を用いて $m_{0,1}\le\mathtt{T}$ となる上限 T を求め，さらに $m_{0,1}=\mathtt{T}$ となるチェックすべき最初の組を求める．

2. Loop of Main routine

2.1. $\mathtt{D} > |\mathtt{Idx}|+2$ ならば (2.13) をチェックし，満たさない組をスキップする．

2.2. $p=\mathtt{IL}<\mathtt{L}$ であって $\mathtt{S}\ge 0$ となる最小の IL を求める．

そのような IL が存在しないならば，不適な組をスキップしで Loop の最初に戻る．

2.3. $\texttt{K} \le \texttt{IL}$ で $p = \texttt{K}$ のとき $\texttt{SS} > |\texttt{Idx}|$ となる K が存在すれば，不適な組をスキップして Loop の最初に戻る．

2.4. $\texttt{Ix} := \texttt{Idx} - \texttt{S} - \texttt{SS} \le 0$ となる最小の $p = \texttt{J}$ で $\texttt{IL} \le \texttt{J} < \texttt{L}$ となるものを探す．

2.5. J が存在して $\texttt{Ix} = 0$ ならば，求めている組なのでリスト R に入れておく．

2.6. Ix に基づいて不適な組をスキップして Loop の最初へ戻る．

説明のコメントを入れた spbasic( ) のコードを以下に示す．

```
def spbasic(Idx,D)
{
// 1. 初期設定
// Idx を |Idx|に変える
// オプション str（文字列で返す）と pt=を設定する
  Idx=-Idx;
  if((Str=getopt(str))!=1) Str=0;
  Tu0=Tu1=0;
  if(type(Tu=getopt(pt))==4&&length(Tu)==2&&isint(car(Tu))){
    Tu0=Tu[0];Tu1=Tu[1];
  }else if(isint(Tu)) Tu0=Tu1=Tu;
  if(Tu1<3) Tu1=0;
// 1.1. 階数 D が Idx=|idx| に比べて大，すなわち D > 3・Idx + 6 なら解なし
    // 以降 D ≤ 3・Idx + 6 とする
  if(!isint(Idx)||!isint(Idx/2)||Idx<0||!isint(D)||D<0||D==1
    ||D>3*Idx+6) return [];
// 1.2. D = 0 ⇒ 階数が任意の場合の処理
  if(D==0){
    for(R=[],D=3*Idx+6;D>=2;D--)
      R=append(spbasic(-Idx,D|str=Str,pt=[Tu0,Tu1]),R);
    return R;
  }
// 1.3. Idx が 0 の場合，結果を返す
  if(!Idx){
    R=0;
    if(D==2&&Tu0<5&&Tu1>3) R="11,11,11,11";
    if(Tu0<4&&Tu1>2){
      if(D==3) R="111,111,111";
      if(D==4) R="22,1111,1111";
      if(D==6) R="33,222,111111";
    }
    if(!R) return [];
    return [(Str==1)?R:s2sp(R)];
  }
// 1.4. 特異点の個数の最大値 L を設定する
//   D > Idx + 2 のとき特異点は 3 点（L = 3）
//       D = 3・Idx + 6 のときは解を返す
//   D ≤ Idx + 2 のとき，特異点の個数は Idx/2+4 個以下
//   オプション pt= をチェック
  if(D>Idx+2){
    L=3;
```

```
    if(D==3*Idx+6){
      R=[[D/2,D/2],[D/3,D/3,D/3],[D/6,D/6,D/6,D/6,D/6,D/6-1,1]];
      return [(Str==1)?s2sp(R):R];
    }
  }else L=Idx/2+4;
  if(L>Tu1) L=Tu1;
```

// **1.5.** 辞書式順序で求める．$V_{00} \le \mathtt{T}$ となる最大の T を求める ($\Rightarrow V_{j,\nu} \le \mathtt{T}$)
// $V_{00} < \mathtt{D}$, $K \cdot (V_{00} - \mathtt{K}) \le \mathtt{Idx}$ に注意（K=D%T は，$\mathtt{D} + \mathtt{T}\mathbb{Z}$ の中の最小非負整数）

```
  for(T=D-1;T>1;T--){
    K=D%T;
    if((T-K)*K<=Idx) break;
  }
```

// **1.6.** 求めた T で最大の分割を初期値にする
// NP[m]：自然数 D の m 以下の自然数での分割の最初のもの
// NP[m]$=[m,\ldots,m,m']$ （length(NP)$=$ D，$m \ge m' > 0$）
// FS: (2.13) のチェック

```
  V=newvect(L);
  NP=newvect(T+1);
  for(I=1;I<=T;I++) NP[I]=nextpart(D|max=I);
  for(I=0;I<L;I++) V[I]=NP[T];
  S1=NP[1];
  FS=(D>Idx+2 ||(L==3&&D>Idx))?1:0;
```

// **2. Loop of Main routine**
// Idx と D と 分割の組の最大個数 L および V の初期値が与えられている
// 辞書式順序の順に条件を満たす V を探し，見つかった順に R に格納する

```
  for(R=[];;){
```

// **2.1.** $\mathtt{D} > \mathtt{Idx} + 2$ などの場合を考察
// $p = 3$ となり，V_{00}, V_{10}, V_{20} の制限を考察（$V_{00} \ge V_{10} \ge V_{20}$ に注意）
// $V_{00} + V_{10} + V_{20} = \mathtt{D}$ なので $V_{00} + V_{10} + V_{20} \ne \mathtt{D}$ となっていたら調整
// 特に $3 \cdot V_{00} < D$ なら終了

```
    if(FS){
      if(3*car(V[0])<D) break;
```

// **2.1.1.** まず $V_{00} + V_{10} \ge \mathtt{D}$ となっていたら，可能なら V_{10} を $\mathtt{D} - V_{00} - 1$ に変更

```
      if(car(V[0])+car(V[1]) >= D && (T=D-car(V[0])-1) > 0)
        V[1]=V[2]=NP[T];
```

// **2.1.2.** $\mathtt{S} := \mathtt{D} - V_{00} - V_{10} - V_{20}$ を 0 にしたい

```
      S=D-car(V[0])-car(V[1])-car(V[2]);
```

// **2.1.3.** $V_{00} + 2 \cdot V_{10} < \mathtt{D}$ or $(\mathtt{S} \ne 0,\ V_{10} = 1) \Rightarrow V[0]$ 以降を変更し Loop の最初へ

```
      if(car(V[0])+2*car(V[1])<D || (car(V[1])==1 && S) ){
        if(car(V[0])==1) break;
        V[0]=V[1]=V[2]=nextpart(V[0]);
        continue;
      }else if(S<0) V[2]=NP[car(V[2])+S];
```

// **2.1.4.** S が正 または $V_{20} + \mathtt{S} < 1$ のとき V[1], V[2] を変えて Loop の最初へ

```
      if(S>0||var(V[2])+S<1){
        V[1]=V[2]=nextpart(V[1]);
        continue;
```

// **2.1.5.** $V_{00} + V_{10} + V_{20} = \mathtt{D}$ となるよう V_{20} を減らす（常に可能）

```
    }else if(S<0) V[2]=NP[car(V[2])+S];
  }
```

// **2.2.** IL 番目以上で basic となる最小の IL $<$ L を求める

```
  for(S=-2*D,IL=0;IL<L;IL++){
    S+=D-car(V[IL]);
    if(S>=0) break;
  }
```

// **2.2.1.** S $:=$ (D $- V_{00}) + \cdots +$ (D $- V_{L-1,0}) -$ 2D < 0 なら（(2.8) に違反）調整

//　$V_{\mathrm{K}0}$ を調整して (2.8) を満たることのできる最小の K を求める

```
  if(S<0){  /* reducible i.e. IL=L && S<0 */
    for(LL=L-1;LL>=0;LL--){
      if((K=car(V[LL]))+S>0){
        V[LL]=NP[K+S];
        break;
      }else{
        S+=K-1;
        V[LL]=S1;
      }
    }
    if(LL<0) break;
    for(I=LL;I<L;I++) V[I]=V[LL];
    continue;
  }
```

// **2.3.** 以下は S $:=$ (D $- V_{00}) + \cdots +$ (D $- V_{\mathrm{K},0}) -$ 2D ≥ 0, IL $\leq$ K $<$ L を満たす

// IL から L $- 1$ まで basic

// SS $\leq$ Idx をチェックする

```
  for(SS=K=0;K<=IL;K++){
    ST=car(V[K]);SS0=SS;
    for(I=length(V[K])-1;I>0;I--) SS+=(ST-V[K][I])*V[K][I];
    if(SS>Idx) break;
  }
```

// **2.3.1.** K $\leq$ IL で SS $>$ Idx $\Rightarrow$ となったので V[K] を調整 $\Rightarrow$ Loop の最初へ

```
  if(SS>Idx && car(V[K])!=1){
    if((W=nextcod(V[K],SS-Idx))==[]){
      for(T=car(V[K])-1;T>0;T--){
        J=D%T;
        if(SS0+J*(T-J)<=Idx) break;
      W=NP[T];
    }
    for(J=K;J<L;J++) V[J]=W;
    continue;
  }
```

// **2.4.** Idx を実現できているかどうかを点 J まででチェック

//　J が増えると Ix は狭義単調減少なので，最初に Ix ≤ 0 となる J を得る

//　IxF は J $- 1$ のときの Ix

//　Ix ≤ 0 を満たす最小の J を求める

```
  for(Ix=2*D^2+Idx,J=0;J<L;J++){
    IxF=Ix;
    for(Ix-=D^2,TV=V[J];TV!=[];TV=cdr(TV)) Ix+=car(TV)^2;
    if(Ix<=0) break;
```

```
    }
```
// **2.5.** $\mathtt{Ix}=0$ なので求める tuples が得られたので R に格納

```
    if(!Ix&&J>=IL&&J>Tu0-2){
      for(TR=[],K=J;K>=0;K--) TR=cons(V[K],TR);
      R=cons((Str==1)?s2sp(TR):TR,R);
    }
```
// **2.6.** 次のものを探す準備

// **2.6.1.** V_{J0} を変えなければ $\mathtt{Ix}\le 0$ とはなり得ない場合

```
    if(Ix<0 && IxF-mincod(D,car(V[J]))<0) J--;
```
// **2.6.2.** J が見つからなかったそ場合は V[L-1] を次のものの変更に進む

```
    else if(J>=L) J=L-1;
```
// **2.6.3.** V[J] を次のものに変更する

```
    for(I=J;I>=0&&car(V[I])==1;I--);
    if(I<0) break;
    V[I]=nextpart(V[I]);
    for(J=I+1;J<L;J++) V[J]=V[I];
  }
  return R;
}
```

3. spbasic() から呼び出される関数

3.1. 次の分割を返す

nextpart(V|max=N)

:: 自然数の分割において辞書式順序の大きい順で V の次のもの返す.

V が最後の分割なら 0 を返す.

V が正整数なら，V の最初の分割を返す. このとき，max=N が指定されていたならば，V の分割で [N,...,N,M] ($0<\mathtt{M}\le\mathtt{N}$) となるものを返す.

```
[0] S=os_md.nextpart(5);
    [5]
[1] while((S = os_md.nextpart(S)) != 0) print(S);
    [4,1]
    [3,2]
    [3,1,1]
    [2,2,1]
    [2,1,1,1]
    [1,1,1,1,1]
    0
[2] os_md.nextpart(5|max=3);
    [3,2]
```

```
def nextpart(V)
{
  if(isint(V)){
    if(V<1) return [0];
    I=V;
    if(!isint(K=getopt(max))|| K<1) K=I;
    K++;V=[];
  }else{
    if(car(V) <= 1)
      return 0;
    for(I = 0, V = reverse(V); car(V) == 1; V=cdr(V))
      I++;
```

```
    I += (K = car(V));
    V=cdr(V);
  }
  R = irem(I,--K);
  R = (R==0)?[]:[R];
  for(J = idiv(I,K); J > 0; J--)
    R = cons(K,R);
  while(V!=[]){
    R = cons(car(V), R);
    V = cdr(V);
  }
  return R;
}
```

3.2. 条件を満たす次の分割を返す

`nextcod(V,D)`

:: 分割 `V` より後の分割で，成分の平方和が `D` 以上増えるものを返す
存在しないときは `[]` を返す

```
def nextcod(V,D)
{
  if(D<=0){
    if(car(V)==1) return [];
    for(D=0,TV=V;TV!=[];TV=cdr(TV)) D+=car(TV)^2;
    V=nextpart(V);
    for(TV=V;TV!=[];TV=cdr(TV)) D-=car(TV)^2;
  }
  K=length(V)-1;
  for(S=T=0;K>=0; K--){
    T+=(W=V[K]);S+=W^2;
    if(W>2&&T^2-mincod(T,W-1)>=S+D){
      R=nextpart(T|max=W-1);
      for(--K;K>=0;K--) R=cons(V[K],R);
      return R;
    }
  }
  return [];
}
```

3.3. ある条件を満たす行列の共役類の最小余次元を返す

`minicod(D,T)`

:: codim `nextpart(D,T)` (cf. (2.6)) を返す

```
def mincod(D,T)
{
  K=D%T;
  return D^2-T*(D-K)-K^2;
}
```

`spbasic()` における不要なチェックを避ける最適化のコードの効果を，それを外した場合との実行時間の比較で検証してみる．

不要なチェックを避ける最適化のコードの検証

最適化のコード	1.5	2.1	2.31	2.31*	2.61
チェック条件	$\mathtt{SS} > \lvert\mathtt{Idx}\rvert$	$\mathtt{S} \neq 0$	$\mathtt{SS} > \lvert\mathtt{Idx}\rvert$	$\mathtt{SS} > \lvert\mathtt{Idx}\rvert$	$\mathtt{Ix} < 0$
リジッド指数	-16	-26	-8	-16	-26
# fundamental tuples	647	2889	291	647	2889
実行時間 (秒)	0.58	4.92	0.05	0.58	4.92
コードを外した実行時間	26.31	24.47	91.58	32.44	24.67

$\mathtt{Ix} < 0 \Leftrightarrow \mathtt{S} + \mathtt{SS} > \lvert\mathtt{Idx}\rvert$ に注意.

上の "コードを外した実行時間" の 2.31* の項は，**2.31** において `nextcod()` を `nextpart()` で置き換えた時の実行時間

4. Fuchs 型方程式

Riemann 球面 $\mathbb{P}^1$ 上に p 個の特異点を持つランク n の Fuchs 型方程式

$$\mathcal{M} : \frac{du}{dx} = \Big(\sum_{j=1}^{p-1} \frac{A_j}{x - a_j}\Big)u \quad \text{with } A_1 + \cdots + A_{p-1} + A_p = 0 \quad (A_j \in M(n, \mathbb{C}))$$

に対する Deligne-Simpson 問題（$A_1 + \cdots + A_p = 0$ を満たす A_j の共役類の特徴づけ）は星形 Kac-Moody ルート系と関係づけることによって，以下のように [**CB**] によって解決された.

$\mathcal{M}$ のスペクトル型は,（A_j が対角化可能なときは）p 個の特異点における特性指数が定める重複度の組，すなわち，特異点 a_j（$j = p$ のときは無限遠点）における留数行列 A_j の重複度が定める n の分割の p 個の組，として定義される．既約な Fuchs 型方程式 $\mathcal{M}$ のスペクトル型は，それが星型 Kac-Moody ルート系の正ルート α で $\operatorname{supp}\alpha \ni \alpha_0$ かつ，$(\alpha|\alpha) \neq 0$ または α が indivisible なものに対応する，ということで特徴づけられる (cf. [**CB, O2**])．このとき，$\mathcal{M}$ は $2 - (\alpha|\alpha)$ 個のアクセサリー・パラメーターを持つ．方程式 $\mathcal{M}$ はアクセサリー・パラメーターをもたないとき**リジッド**という．すなわち，スペクトル型が実ルートに対応することである．これらは，方程式に対する middle convolutions と addition という変換が，Kac-Moody ルート系における W の作用に対応することから示される.

正の虚ルートの $\widetilde{W}$-軌道は, アクセサリー・パラメータを持つ線型常微分方程式のモノドロミー保存変形によって高次元 Painlevé 方程式に対応している．実際，リジッド指数が -2 の例 2.5 (iii) は新たな 4 次元の Painlevé 方程式の発見を導き，さらに鈴木 [**Su**] は -4 の場合の結果から 6 次元 Painlevé 方程式を構成した．なお，リジッド指数 -10 以上の fundamental tuples の表は [**O4**] にある.

Fuchs 型方程式 $\mathcal{M}$ の場合と同様に，[**O2**] によって $\mathbb{P}^1$ 上の確定特異点型単独高階線型微分方程式にも middle convolution や addition やスペクトル型が定義され，方程式の構成や積分表示，接続問題などの様々な解析がなされた．この場合も星形 Kac-Moody ルート系と Weyl 群の作用とが対応することが基本となった.

Fuchs 型方程式の場合の結果のいくつかは不分岐不確定特異点を持つ場合にも拡張された．対応するスペクトル型は対称 Kac-Moody ルート系のルートで記述でき，middle convolution やその Weyl 群の作用に対応することと軌道の "有限性" が [**HO**] で示され，リジッド指数が -2 の場合の分類表を与えた．これは 4 次元 Painlevé 方程式の分類 [**HKNS**] の根拠となった.

不分岐不確定特異点を許す $\mathbb{P}^1$ 上の線型常微分方程式のスペクトル型は星形 Kac-Moody ルート系に細分構造を入れた分割の組でも表すことができる（cf. [**O3**]）．このスペクトル型を生成する関数やスペクトル型やそれに特性指数を付加した generalized Riemann scheme を用いて微分方程式を解析する関数が `Risa/Asir` のライブラリ [**O6**] にある（`spbasic( )` など 500 個ほどの関数が定義されている）.

5. Acknowledgements

筆者は Fuchs 型方程式のスペクトル型の解析のため，`okubo` [**O5**] というコンピュータプログラムを C で作成して 2008 年に公開した．その機能の 1 つに，指定したリジッド指数の fundamental tuples の出力があった．

リジッド指数が -4 のとき，そのプログラムの結果に抜けている組が 1 つあることが 2019 年に川上氏によって指摘され（cf. 注意 2.6 (ii)），筆者は fundamental tuples を得る関数 `spbasic()` を，数式処理 `Risa/Asir` [**Risa**] のもとで作成してライブラリ [**O6**] に含めた．

その後 2024 年に，L. Yongcha 氏からリジッド指数が -18 のとき，この関数の返す結果に抜けている組があること，および，数理物理の問題 (cf. [**MOTZ**]) との関連を教えていただいた．これを受けて，関数 `spbasic()` のアルゴリズムを大幅に見直して修正し，この論文に述べた形となった．修正前のもので，リジッド指数が -24 までで抜けていたのは指摘を受けた 1 つだけであったが，リジッド指数が -26 でも，2889 種の fundamental tuples のうちの 1 つが抜けていることも分かった．

著者は，プログラムの間違いを指摘していただいた川上氏と L. Yongcha 氏に感謝いたします．

References

[CB] Crawley-Boevey W., On matrices in prescribed conjugacy classes with no common invariant subspaces, and sum zero, Duke Math. J. **118**(2003), 339-352.

[HKNS] Hiroe, H. Kawakami, A. Nakamura and H. Sakai, *4-dimensional Painlevé-type equations*, MSJ Memoirs **37**, Mathematical Society of Japan, 2018.

[HO] Hiroe K. and Oshima T., Classification of roots of symmetric Kac-Moody root systems and its application, Symmetries, Integrable Systems and Representations, Springer Proceedings in Mathematics and Statistics **40** (2012), 195–241.

[Kc] Kac V. C., *Infinite dimensional Lie algebras*, Third Edition, Cambridge Univ. Press 1990.

[Ko] Kostov, V. P., The Deligne-Simpson problem for zero index of rigidity, *Perspective in Complex Analysis*, Differential Geometry and Mathematical Physics, World Scientific 2001, 1–35.

[MOTZ] Mekareeya1 N., Ohmori K., Yuji Tachikawa Y. and Zafrir G., E_8 instantons on type-A ALE spaces and supersymmetric field theories, J. High Energ. Phys. **144** (2017).

[O1] Oshima T., Classification of Fuchsian systems and their connection problem, RIMS Kôkyûroku Bessatsu **B37** (2013), 163–192.

[O2] Oshima T., *Fractional calculus of Weyl algebra and Fuchsian differential equations*, MSJ Memoirs **28**, Mathematical Society of Japan, Tokyo, 2012.

[O3] Oshima, T., Versal unfolding of irregular singularities of a linear differential equation on the Riemann sphere, Publ. RIMS Kyoto Univ. **57**, (2021). 893–920.

[O4] Oshima T., List of fundamental spectral types with index of the rigidity ≥ -10, 2024, `https://www.ms.u-tokyo.ac.jp/~oshima/paper/spect10.pdf`

[O5] Oshima T., `okubo`, a computer program for Katz/Yokoya/Oshima algorithm, 2007–2008, `http://www.ms.u-tokyo.ac.jp/~oshima/`

[O6] Oshima T., `os_muldif.rr`, a library of computer algebra `Risa/Asir`, 2008–2024. `http://www.ms.u-tokyo.ac.jp/~oshima/`

[Risa] Noro M. etc., `Risa/Asir`, Computer algebra, `http://www.math.kobe-u.ac.jp/Asir/asir.html` `http://www.math.kobe-u.ac.jp/Asir/asir.html`

[Su] Suzuki T., Six-dimensional Painlevé systems and their particular solutions in terms of rigid systems, J. Math. Phyics 55 (2014), 57–69.

Email address: `oshima@ms.u-tokyo.ac.jp`