数式処理による数学研究とプレゼンテーション
Study of Mathematics and Presentation by Computer Algebra

城西大学 理学部 大島利雄
Toshio Oshima, Faculty of Science, Josai University

1 Introduction

パーソナル・コンピュータが普及し始めたのは 1980 年初めであり，実用面ではまず文書や論文の作成に使っていた．私が数学の最新の研究に使ったのは，半単純対称空間（実半単純 Lie 群も含む）の離散系列表現の主系列表現への埋め込みの研究 [MO] で，成果を古典型実半単純 Lie 群の離散系列表現の場合に `C` のプログラムで実現した．数学の計算は少し複雑にすると計算時間はいくらでも増大する．1980 年代に作ったこの `C` プログラムは組み合わせ論的なもので，現在のコンピュータでも有効であり，それを用いて当時の 16bit パソコンから現在までのコンピュータの速度比較を行っており，その結果は [O1] にある．当時から比べると，速度は 1 万倍以上になっている．

証明にパソコンを使った論文には [OO] があり，数式処理言語 `Reduce` を使った計算を 1993 年頃に行った（定理は 1994 年に発表の結果 [OOS] に含まれるが，証明の出版は後年になった）．2 変数の関数微分方程式の解を求める問題となるが，関数を Laurent 展開することにより，係数の関係式で生成される多項式環の両側イデアルの準素分解の問題に帰着させた．計算機の処理能力を明らかに越える問題であったが，`Reduce` の因数分解の機能などを電卓のように使って証明をすることができた．変数や生成元の数が大きく，手計算ではとても無理な多項式イデアルの計算である．6 桁またはそれ以下の整数係数の 24 変数の 2 次多項式 26 個が生成元であったが，場合分けをして，9 桁またはそれ以下の整数係数の 8 変数（単項式 40 項以上の和で表せる 5 次式などを含む）多項式 6 個が生成するイデアルの問題がその 1 つになり，それが最も複雑であった．

2007 年頃に多変数の特殊関数（球関数の一般化）の接続問題を，関数を 1 次元特異集合に制限することによって求めることを試み（結果は [OS] にある．示野が `Maple` を使って微分方程式系の制限の計算を行った），それがきっかけで Fuchs 型の線型常微分方程式の研究を始めた（cf. [O5] のあとがき）．結果を予測するには，微分作用素の計算が必要であった．組み合わせ論的部分は `C` でプログラム [O3] を書いたが，微分作用素環の計算をするには数式処理言語が必要であった．いくつか候補を考えたが，私自身は `C` のプログラムを書くのが慣れていてたこと，また中身が公開されていて必要なら調べ得ること，などから `Risa/Asir` で微分作用素環のライブラリ `muldif.rr` [O4] を作って研究することにした．2007 年の 11 月の初めに微分作用素の計算を行う関数を作成して計算を行い，4 階の even family と呼ばれる常微分方程式の解の接続問題を解いた．[OS] はその結果を使っている．

even/odd family という微分方程式は，今まで研究されていなかった最も簡単な rigid な Fuchs 型線型常微分方程式の系列であり，得られた接続公式は一般の rigid な場合にも拡張

されるのではないかと考えて，`Risa/Asir` を使って実験し，特異点が 3 個の場合に一般公式の予想を立てた．rigid な方程式とは，各特異点での解の局所モノドロミー群から大域的モノドロミー群が決まってしまう方程式のことである．なお，決まらないときのモジュライのパラメータはアクセサリー・パラメータと呼ばれる．

この予想は組み合わせ論的問題にも帰着でき，`C` のプログラム [O3] によって 40 階以下の 400 万以上の rigid な場合に正しいことを確かめたが，1998 年の 4 月に証明が得られた．その後, 特異点が 4 点以上の場合にも通用する別の証明が得られ，さらに翌年 5 月には rigid でない場合の見かけの特異点を持たない単独高階 Fuchs 型方程式の存在の必要十分条件とその構成の問題（Deligne-Katz-Simpson 問題）が解決できた．

このような方程式は，スペクトル型，すなわち各特異点での局所モノドロミー行列の Jordan 標準形の型 — 対角化可能なときは特異点での固有値の重複度情報 — で分類される．n 階で特異点が $p+1$ 個の場合は，スペクトル型は，n の分割の $p+1$ 個の組という組み合わせ論的データである．

このような流れで，一般の Fuchs 型線型常微分方程式に対して，接続問題，解の積分表示やべき級数表示，モノドロミー群の既約条件，隣接関係式とそれを与える微分作用素などの研究を行い，Kac-Moody ルート系との関係などを含めての結果が [O6] にまとめられた．現在は `Risa/Asir` のライブラリ `muldif.rr` の関数を必要に応じて増やしながら，同様の視点で不確定特異点をもつ場合や多変数の超幾何関数の研究を続けている．

一方，[O6] の結果はコンピュータ上で実現できるものがほとんどであるので，それを `muldif.rr` の関数として作成した．また結果が誰にでも分かるように，TEX のプレビューア `dviout` [O2] を通して綺麗に表示できるようにした．任意の数式を, 扱い易い $\mathcal{AMS}$TEX のソースに変換し，またそれを TEX のプレビューアで表示する，という関数も `muldif.rr` の中にある．数式処理で数式を作成して論文原稿に張り込む，という用途に便利に使えることも目指している．なお，これは数式を TEX のソースに変更するという元々備わっていた関数を元にしている．

2 有理関数と分数階微分

[O6] では，多項式または有理関数係数の微分作用素

$$P = a_n(x)\,\partial^n + a_{n-1}(x)\,\partial^{n-1} + \cdots + a_0(x) \quad (\partial = \tfrac{d}{dx}) \tag{1}$$

に対し，Gauge 変換（または addition）

$$\mathrm{Ad}\big(f(x)^\lambda\big)P = f(x)^\lambda \circ P \circ f(x)^{-\lambda} \quad (f(x) \text{ または } \log f(x) \text{ は有理関数}) \tag{2}$$

や Laplace 変換 L，すなわち (1) の P が多項式係数のとき $(x,\partial) \mapsto (-\partial, x)$ で与えられる変換, および P の左辺から有理関数をかけて係数が x の多項式でその共通因子が自明になるように正規化する変換 R が基礎で，特に Katz [K] の導入した middle convolution に相当する変換 mc_μ は

$$\mathrm{mc}_\mu = \mathrm{L}^{-1} \circ \mathrm{R} \circ \mathrm{Ad}(x^{-\mu}) \circ \mathrm{L} \circ \mathrm{R} \tag{3}$$

で与えられる．なお，mc_μ は $-\mu$ 階の（分数階）微分に対応する作用素で，$\mathrm{Ad}(\partial^{-\mu})\circ \mathrm{R}$ とも書かれ，ϑ と ∂ のみで表される作用素は $(\vartheta,\partial)\mapsto(\vartheta-\mu,\partial)$ の変換を受ける（$\vartheta := x\,\partial$）．

rigid な Fuchs 型高階微分方程式 $Pu=0$ は自明な方程式 $\partial\, u=0$ から mc_μ や $\mathrm{Ad}\big((x-c)^\lambda\big)$ の複数回の合成によって構成される（cf. [O6]）．たとえば，Gauss の超幾何関数 $F(a,b,c;x)$ の満たす微分方程式は

$$\begin{aligned}
P &= \mathrm{mc}_{1-a}\circ \mathrm{Ad}\big(x^{a-c}(x-1)^{c-b-1}\big)\,\partial\\
&= \mathrm{mc}_{1-a}(\partial-\tfrac{a-c}{x}-\tfrac{c-b-1}{x-1})\\
&= \mathrm{mc}_{1-a}\,\partial\big(x(1-x)\,\partial-((-a+b+1)x+a-c)\big)\\
&= \mathrm{mc}_{1-a}\big((\vartheta+1-a+c)\,\partial-(\vartheta+1)(\vartheta-a+b+1)\big)\big)\\
&= (\vartheta+c)\,\partial-(\vartheta+a)(\vartheta+b)\\
&= x(1-x)\,\partial^2+\big(c-(a+b+1)x\big)\,\partial-ab
\end{aligned}$$

で与えられ，そのことから $Pu=0$ の解の積分表示

$$\frac{1}{\Gamma(1-a)}\int_0^x t^{a-c}(1-t)^{c-b-1}(x-t)^{-a}dt \tag{4}$$

が得られる．

一般の Fuchs 型常微分方程式に対しても，$\mathrm{Ad}\big((x-c)^\lambda\big)$ や mc_μ によって接続係数や解の積分/べき級数表示，既約性などがどのように変わるかを調べることによって [O6] の多くの結果が得られている．

3 Risa/Asir

有理関数係数の微分作用素環の計算を `Risa/Asir` で行うには，注意が必要で，特に有理関数係数の多項式や，成分が有理関数の行列の演算が，そのままでは思い通りにならないことがある．いくつかの例を挙げてみよう．

```
[0] 2/x-1/x+1/x-1/x;
(x^3)/(x^4)
[1] x/(x+y)+y/(x+y);
(x^2+2*y*x+y^2)/(x^2+2*y*x+y^2)
[2] x/y*y/x;
(y*x)/(y*x)
[3] 1/(1/x);
(x)/(1)
[4] deg((a/b)*x^2,x);
0
[5] diff((1/a)*x+1/b,x);
(b^2*a)/(b^2*a^2)
[6] diff((x+1)^(-3),x);
(-3*x^2-6*x-3)/(x^6+6*x^5+15*x^4+20*x^3+15*x^2+6*x+1)
[7] A = newmat(2,2,[[a,0],[0,1/a]]);
[ a 0 ]
[ 0 (1)/(a) ]
[8] det(A);
```

```
internal error (SEGV)
return to toplevel
[9] coef(x+1/a,1,x);
0
```

このような場合も、より望まれる形で結果が得られるように作ったライブラリが `muldif.rr` である．それに含まれるいくつかの関数をあげてみる．

- `muldo()`：有理関数係数の（偏）微分作用素の（行列の）積を計算する．
- `expat()`：常微分方程式の確定特異点における特性指数を求める．
- `dform()`：有理関数係数の 1 次と 2 次の微分形式の計算．
- `mygcd()`, `mylcm()`：整数環，有理関数係数 1 変数多項式環，有理関数係数常微分作用素環において，（左または右）最大公約元や最小公倍元を求める．
- `mdivisor()`： 上のユークリッド環を成分とする行列に対して行と列の基本変形を行って標準化する（変換行列も得られる．可換なときは，単因子を求めることになる．常微分作用素環では [O6, Lemma 1.10] の実現）．
- `stoe()`：線型常微分方程式の 1 階のシステムを単独高階に変換する．
- `solpokubo()`：大久保型常微分作用素の固有多項式と固有値を求める．
- `spgen()`：rigid なスペクトル型，あるいは与えられた rigid 指数をもつスペクトル型のリストを求める．
- `sproot()`：スペクトル型と Kac-Moody ルート系や Weyl 群との関係を得る（cf. [O6, (7.30), (7.35), (7.40)] の計算）
- `getbygrs()`：スペクトル型あるいは一般化 Riemann scheme を与えて，方程式（[O6, Theorem 6.14]），解の積分/べき級数表示（[O6, Theorem 8.1]），隣接関係式（[O6, Theorem 11.3]），接続公式（[O6, Theorem 12.6]），既約条件（[O6, Corollary 10.12]）などを得る．また結果を TEX のソースにしたり, `dviout` などで表示する．
- `shiftop()`：rigid な Fuchs 型方程式の特性指数の任意の整数のずらしを与える shift operator を構成し（[O6, Theorem 11.2]），逆の作用素との合成で得られるスカラー（1 次式の積に分解）を求める（[O6, Theorem 11.8]）．
- `conf1sp()`：Fuchs 型常微分方程式の Poincaré rank 1 の合流を示す．
- `m2mc()`：4 点の特異点を持つ rigid な Fuchs 型常微分方程式のスペクトル型に対応する 2 変数の超幾何微分方程式を Pfaff 型で求め, それの一般化 Riemann scheme や既約条件などを得る（結果を TEX のプレビューアで表示できる）．Appell の 4 種の超幾何やその一般化を含む未開拓（現在研究中）の超幾何系となる．

以下は，[O6] とは直接関係がないが有益な関数．

- `myhelp()`：`muldif.rr` の関数のマニュアルを表示する．関数名で参照することもできる（関数名による参照は Microsoft Windows の環境のみ）．
- `mtransbys()`： スカラーに対する変換の関数をリスト，ベクトルや行列に拡張する（`map()` と異なり再帰的）．

- `fmult()`：変換を定義する関数に対し，それを（パラメータを変えながら）複数回合成した変換を行う．
- `simplify()`：（複数個の）線型関係などをもつパラメータを含む式で，そのパラメータを適当に選び直して，式を簡単化する．
- `polybyvalue()`：1 変数の $n-1$ 次多項式を，n 個の点での値で与える．
- `getroot()`：1 変数多項式の根を有理式の範囲で求める．
- `polinsym()`, `polinvsym()`：与えられた変数についての対称多項式と，基本対称式を変数とする多項式との相互変換．
- `pfrac()`：有理式をある変数について部分分数展開する．
- `bernuille()`：ベルヌーイ多項式を得る．
- `pcoef()`：多項式の正べきの与えられた項の係数を求める（多項式の正べきが大きすぎて展開できないときも有効）．
- `lsol()`：有理関数係数の連立 1 次方程式を，有理関数体の中で解く．
- `lnsol()`：有理関数係数の連立 1 次方程式の有理数解を求める．
- `lsort()`：リストに対し合併，共通部分，共通部分の削除，同じ元の削除などの操作を行う．
- `vnext()`：ベクトルの成分（同じ成分があってもよい）を並べ替えて辞書式順序で次に続くものを得る．
- `myimage()`, `mykernel()`：有理式成分の行列に対し，核と像を求める．
- `mmod()`：行列で与えた線形変換の，商空間への射影を求める．
- `mgen()`：様々な一般行列を容易に作る（成分が `a_ij` の一般行列，対称行列，対角行列など）．
- `s2m()`：数値が成分の行列を文字列で簡単に作る．例えば対角成分が $1, 2, 3$ の 3 次対角行列は，`"1,02,003"` と表せる．
- `mydet()`, `mydet2()`：有理式成分の行列の行列式を求める．
- `myinv()`：有理式成分の行列の逆行列を求める．
- `str_str()`：文字列から部分文字列を探す．
- `str_subst()`：文字列から部分文字列を探して別の文字列に置き換える．
- `sord()`：置換群の元を Bruhat order で比較する．
- `my_tex_form()`：数式を TEX のソースに変換する．
- `dviout()`：数式を TEX のプレビューア（デフォルトは `dviout`）で表示する．source special を使っているので，プレビューアの画面をマウスでクリックすると，TEX のソースファイルが開かれ，該当部分にジャンプする．ソースの変更やコピーが可能．ソースは順次追加される．
- `dviout0()`：`muldif.rr` で生成した TEX のソースの削除や編集, プレビューアでの再表示などができる．

4 muldif.rr

以下は，現時点（2014 年 2 月）において muldif.rr で定義されている関数と簡単な説明のリストである（作成途中で最終形でない関数も含まれる）．関数のより詳しい説明は，[O4] にある muldif.pdf などを参照してください．muldif.rr 本体も同じ場所にあります．

muldif.rr では，たとえば x, y の多項式とは，それ以外の変数の有理式を係数とする変数 x と y の多項式を意味する．また，有理式の係数は有理数とする．同様に微分作用素とは，（パラメータや変数の）有理式を係数とする線型微分作用素を意味する．

1. Functions related to differential operators

1.1. Fundamental functions

1. muldo(p_1,p_2,[x,∂_x]) または muldo(p_1,p_2,x)
   muldo(p_1,p_2,[[x_1,∂_{x_1}],[x_2,∂_{x_2}],...])
   :: 有理関数係数の常（または偏）微分作用素（の行列）の積 ($\Leftarrow [\partial_x, x] = 1$)
2. muledo(p_1,p_2,[x,∂_x]) または muledo(p_1,p_2,x)
   :: Euler 型常微分作用素（の行列）の積 ($\Leftarrow [\partial_x, x] = x$)
3. transdo(p,[[x_1,∂_{x_1}],[x_2,∂_{x_2}],...],[[y_1,∂_{y_1}],[y_2,∂_{y_2}],...])
   :: 微分作用素の変換（$x_i \mapsto y_i = y_i(x)$, $\partial_{x_j} \mapsto \partial_{y_j} = c_j(x) + \sum a_{j\nu}(x)\,\partial_{x_\nu}$）
4. translpdo (p,[[x_1,∂_{x_1}],[x_2,∂_{x_2}],...],mat)
   :: 微分作用素の線形座標変換（$x_i \mapsto \sum_j (mat)_{ij} x_j$）
5. appldo(p,r,[x,∂_x]) または appldo(p,r,[[x_1,∂_{x_1}],[x_2,∂_{x_2}],...])
   :: 微分作用素（の行列）の有理式（の行列）への作用の計算
6. adj(p,[x,∂_x]) または adj(p,[[x_1,∂_{x_1}],[x_2,∂_{x_2}],...])
   :: 微分作用素（の行列） p の formal adjoint
7. sftpexp(p,[x,∂_x],q,r) または sftpexp(p,[[x_1,∂_{x_1}],...],q,r])
   :: 微分作用素 p を $q^{-r} \circ p \circ q^r$ と変換する
8. appledo(p,r,[x,∂_x])
   :: Euler 型常微分作用素の有理式への作用の計算
9. divdo(p_1,p_2,[x,∂_x]|rev=1)
   :: 常微分作用素の割り算
10. mygcd(p_1,p_2,[x,∂_x]|rev=1) または mygcd(p_1,p_2,[x]|rev=1)
    mygcd(p_1,p_2,x), mygcd(p_1,p_2,0)
    :: 常微分作用素（または x の多項式，または正整数）p_1 と p_2 の GCD
11. mylcm(p_1,p_2,[x,∂_x]|rev=1) または mylcm(p_1,p_2,[x]|rev=1)
    mylcm(p_1,p_2,x), mylcm(p_1,p_2,0)
    :: 常微分作用素（または x の多項式，または正整数）p_1 と p_2 の LCM
12. m1div(m,n,[x,∂_x]) または m1div(m,n,[x])
    m1div(m,n,x)
    :: 常微分作用素（or x の多項式）の正方行列 m と有理式（or x を含まない有理式）の正方行列 n に対し，$m = R[1](\partial_x - n) + R[0]$ (or $m = R[1](x - n) + R[0]$ となるリスト $R = [R[0], R[1]]$ を返す.
13. qdo(p_1,p_2,[x,∂_x])
    :: 常微分方程式 $p_1 u = 0$ に対し $q_1 p_2 u = 0$ となる微分作用素 q_1 と $q_2 p_2 u = u$ となる微分作用素 q_2 のリスト $[q_1, q_2]$ を返す
14. mdivisor(m,[x,∂]|trans=1,step=1)
    mdivisor(m,x|trans=1,step=1), mdivisor(m,0|trans=1,step=1)
    :: 有理関数係数 1 変数多項式/常微分作用素や整数の行列の単因子を得る
15. sqrtdo(p,[x,∂_x])
    :: $x \mapsto 1/x$ で（x のべき倍を除いて）不変な微分作用素 p に対する変数変換 $x \mapsto y = x + \sqrt{x^2 - 1}$
16. toeul(p,[x,∂_x],n)
    :: 確定特異点型常微分作用素を $x = n$ で Euler 型に変換
17. fromeul(p,[x,p_x],n)
    :: Euler 型常微分作用素を元に戻す（toeul($p, [x, \partial_x], n$) の逆変換）

18. `expat(`p`,[`x`,`∂_x`],`n`)`
:: 確定特異点型常微分作用素の $x=n$ での特性指数を求める
19. `sftexp(`p`,[`x`,`∂_x`],`n`,`r`)`
:: 常微分作用素 p を $(x-n)^{-r}\circ p\circ(x-n)^r$ に変換する
20. `fractrans(`p`,[`x`,`∂_x`],`n_0`,`n_1`,`n_2`)`
:: 常微分作用素 p に $(n_0,n_1,n_2)\mapsto(0,1,\infty)$ という一次分数変換を行う
21. `chkexp(`p`,[`x,∂_x`],`n`,`r`,`m`)`
:: $j=0,\dots,m-1$ の全てに対し，確定特異点型常微分作用素 p の解 u_j で $(x-n)^{-(r+j)}u_j$ が $x=n$ で正則でそこでの値が 1 となるものの存在条件
22. `soldif(`p`,[`x,∂_x`],`n`,`q`,`m`)`
:: 常微分作用素 p の $x=n$ の近傍での $z^q(1+\sum_{j=1}^{\infty}c_jz^j)$ の形の形式解に対し，長さ $m+1$ のベクトル `[1` c_1 c_2 $\cdots$ c_m`]` を返す（$z=x-n$）.
23. `okuboetos(`p`, [`x,∂_x`] | `$diag=[c_1,c_2,\dots]$`)`
:: 単独 m 階 Okubo 型方程式 $pu=0$（n 階の項の係数が n 次以下の多項式）を Okubo 型の 1 階のシステムに変換する
24. `stoe(`p`,[`x,dx`],`m`)`
:: 1 階の常微分方程式系を単独高階に直す
25. `dform(`ℓ,x`|dif=1)`
:: 変数 $x[0],x[1],\dots$ の 1 次微分形式 $\sum\ell[i][0]d(\ell[i][1])$, または 2 次微分形式 $\sum\ell[i][0]d(\ell[i][1])\wedge d(\ell[i][2])$ の計算. `dif=1` は 1 次微分形式の外微分の計算.
26. `solpokubo(`p`,[`x,∂_x`],`n`)`
:: 単独 Okubo 型常微分作用素 p の n 次の固有多項式と固有値を求める

1.2. Fractional calculus

この項は，[O6] の主要結果（基本的部分は [O5] で解説）を Risa/Asir 上で実現したものとなっている.

27. `laplace(`p`,[`x,∂_x`])` または `laplace(`p`,[[`x_1,∂_{x_1}`],[`x_2,∂_{x_2}`],...])`
:: 微分作用素 p の（部分）Laplace 変換
28. `laplace1(`p`,[`x,∂_x`])` または `laplace1(`p`,[[`x_1,∂_{x_1}`],[`x_2,∂_{x_2}`],...])`
:: 微分作用素 p の（部分）逆 Laplace 変換
29. `mc(`p`,[`x,∂_x`],`r`)`
:: 常微分作用素 p の middle convolution $mc_r(p)$
30. `mce(`p`,[`x,∂_x`],`n`,`r`)`
:: 常微分作用素 p を $(\partial_x-n)^{-r}\circ p\circ(\partial_x-n)^r$ と変換
31. `rede(`p`,[`x,∂_x`])` または `rede(`p`,[[`x_1,∂_{x_1}`],[`x_2,∂_{x_2}`],...])`
:: 微分作用素 p の reduced representative を返す
32. `ad(`p`,[`x,∂_x`],`f`)`
:: 常微分作用素 p の ∂_x を ∂_x-f に置き換える変換
33. `add(`p`,[`x,∂_x`],`f`)`
:: 常微分作用素 p の ∂_x を ∂_x-f に置き換える addition，すなわち `red(ad())`
34. `vadd(`p`,[`x,∂_x`],[[`c_0,r_0`],[`c_1,r_1`],...])`
:: versal addition `add(`p`,[`x,∂_x`],`$\sum_{j\geq 0}\frac{r_jx^j}{\prod_{\nu=0}^{j}(1-c_\nu x)}$`)`
35. `addl(`p`,[`x,∂_x`],`f`)`
:: 常微分作用素の addition の Laplace 変換 `laplace1(add(laplace()))`
36. `cotr(`p`,[`x,∂_x`],`f`)`
:: 常微分作用素 p の $x\mapsto f(x)$ による座標変換
37. `rcotr(`p`,[`x,∂_x`],`f`)`
:: 常微分作用素 p の $x\mapsto f(x)$ による座標変換の reduced representative
38. `s2sp(`p`|num=1)`
:: スペクトル型を表す文字列と “数のリストのリスト” との変換
39. `chkspt(`m`|mat=1)` または `chkspt(`m`|opt=`t`)` または `fspt(`m`,`t`)`
:: 分割の組 m（スペクトルタイプ）または generalized Riemann scheme（GRS）をチェックして $[pts, ord, idx, fuchs, rod, redsp, fspt]$ を返す
`opt="sp", "basic", "construct", "strip", "short", "long", "sort"`
40. `spgen(`n`|eq=1,str=1,pt=[`k,ℓ`],sp=`m`,basic=1)`
:: 階数 n 以下の rigid な分割の組（あるいは与えられたものの軌道）を得る

n が 0 や負のときは，rigidity index が n の basic なものを得る

41. `sproot(`p`,`t`|dviout=1,only=`k`,null=1)`
:: スペクトル型を与えて構成やルートの情報を示す．t=`"base"`, `"length"`, `"type"`, `"part"`, `"pair"`, `"pairs"`, sp
42. `sp2grs(`m`,`a`,`ℓ`|mat=1)`
:: spectral type から generalized Riemann scheme を生成する
43. `ssubgrs (`m, ℓ`)`
:: Generalized Riemann scheme m の ℓ に対する特性指数和
44. `mcgrs(`m`,[`$r_1, r_2, \ldots, r_n$`]|mat=1)`
:: middle convolution と addition を generalized Riemann scheme に施す
45. `redgrs(`m`|mat=1)`
:: 常微分作用素の generalized Riemann scheme の 1-step reduction
46. `getbygrs(`m`,`t`|perm=`l`,var=`v`,pt=[`$p_1, \ldots$`],mat=1)` または
`getbygrs(`m`,[`$t, s_1, s_2, \ldots$`]|perm=`l`,ver=`v`,pt=[`$p_1, \ldots$`],mat=1)`
:: generalized Riemann scheme (GRS) で定義される Fuchs 型常微分方程式の解析（GRS は短縮形またはスペクトルタイプでもよい）
t =`"reduction", "construct", "connection", "operator", "series"`
`"TeX", "Fuchs", "basic", "", "All", "irreducible", "recurrence"`
s =`"TeX", "dviout", "keep", "simplify", "short", "general"`
`"operator", "irreducible", "sft", "top0", "x1", "x2"`
l は特異点の置換または互換，`var` は exponents の変数, $p_1, \ldots$ は特異点の位置（∞ は除く）．
47. `shiftop(`ℓ, s`|zero=1,raw=`k`,all=`t`,dviout=1)`
:: rigid なスペクトル型 ℓ と shift s から shift 作用素を求める
48. `conf1sp(`m`|x2=`± 1`,conf=0)`
:: スペクトル型 m の微分作用素の Poincare rank 1 の合流過程を示す
49. `m2mc(`l`,[`a_0, a_y, a_1, c`]|swap=1,small=1,simplify=0)`
`m2mc(`l, c`|small=1,simplify=0,int=0,swap=`t`)`
:: Pfaff 形式 $du = (A_0 \frac{dx}{x} + A_y \frac{d(x-y)}{x-y} + A_1 \frac{d(x-1)}{x-1} + B_0 \frac{dy}{y} + B_1 \frac{d(y-1)}{y-1})u$ の x 変数での addition+middle convolution を求める（$\ell = [A_0, A_y, A_1, B_0, B_1]$）．
ℓ がスペクトル型や Riemann scheme のとき，c =`"GRC"`, `"GRSC"`, `"Pfaff"`, `"sp"`, `"pairs"`, `"irreducible"`, `"All"`, `"swap"`
50. `mmc(`l`,[`a_0`,...]|mult=1)`
51. `linfrac01(`ℓ`|over=1)`
:: $x = 0, 1, \infty, y, z, \ldots$ の一次分数変換のリスト（$\ell = [x, y]$ etc.）．
52. `lft01(`t, ℓ`)`
:: $\ell = (x_1, x_2, x_3, \ldots)$ に対する特殊一次分数変換

1.3. Some operators

53. `okubo3e([`$p_{0,1}, \ldots, p_{0,m}$`],[`$p_{1,1}, \ldots, p_{1,n}$`],[`$p_{2,1}, \ldots, p_{2,m+n}$`])`
:: 0, 1, ∞ に確定特異点を持つ $m+n$ 階の単独 Okubo 型微分作用素を求める
54. `fuchs3e([`$p_{0,1}, \ldots, p_{0,n}$`],[`$p_{1,1}, \ldots, p_{1,n}$`],[`$p_{2,1}, \ldots, p_{2,n}$`])`
:: 0, 1, ∞ に確定特異点を持つ n 階の Fuchs 型微分作用素を求める
55. `ghg([`$p_{1,1}, p_{1,2}, ..., p_{1,m}$`],[`$p_{2,1}, p_{2,2}, ..., p_{2,n}$`])`
:: 一般超幾何関数 ${}_mF_n(p_1, p_2; x)$ の満たす微分作用素
56. `even4e([`$p_{1,1}, p_{1,2}, p_{1,3}, p_{1,4}$`],[`$p_{2,1}, p_{2,2}$`])`
:: 4 階 even family (Rigid)
57. `odd5e([`$p_{1,1}, p_{1,2}, p_{1,3}, p_{1,4}, p_{1,5}$`],[`$p_{2,1}, p_{2,2}$`])`
:: 5 階 odd family (Rigid)
58. `rigid211([`$p_{0,1}, p_{0,2}$`],[`$p_{1,1}, p_{1,2}$`],[`q_0, q_1`])`
:: Type 211,211,211
59. `extra6e([`$p_{1,1}, p_{1,2}, p_{1,3}, p_{1,4}, p_{1,5}, p_{1,6}$`],[`$p_{2,1}, p_{2,2}$`])`
:: Extra case (Rigid)
60. `eofamily([`$p_{0,1}, p_{0,2}$`],[`$p_{1,1}$`],[`$p_{2,1}, \ldots, p_{2,n}$`])`
:: even/odd family

61. ev4s(p_1,p_2,p_3,p_4,p_5)
:: Heckman-Opdam 超幾何の (BC_2, BC_1) 型制限常微分 (Rigid)
62. b2e(p_1,p_2,p_3,p_4,p_5)
:: Heckman-Opdam 超幾何の (BC_2, A_1) 型制限常微分 (Non Rigid)
63. heun([a,b,c,d,e],p,r|)
:: Heun の微分方程式を与える．r はアクセサリパラメータ

2. Useful functions

2.1. Extended function

64. myhelp(h)
:: muldif.rr のマニュアルを表示する
65. chkfun(f,s)
:: 関数 f（= 文字列）が定義済みかどうか調べ，未定義なら load(s) を実行
66. makev([ℓ_1,ℓ_2,...]|num=1)
:: $\ell_1, \ell_2 \ldots$ を合わせて一つの変数名を作る
67. mysubst(r,[v_1,r_1]), mysubst(r,[[v_1,r_1],...])
:: subst(r,v_1,r_1,...) と同等．r が複雑で r_1 が有理式のときに特に有効.
68. fmult(f,m,ℓ,n)
:: $m_i \mapsto m_{i+1} = f(m_i, l[i], n[0], n[1], \ldots)$ という変換（$m_0 = m$）
69. mtransbys(f,m,ℓ)
:: スカラーに関する変換 $f()$ をリスト，ベクトルまたは行列 m に拡張する
70. mmulbys(f,m,n,ℓ)
:: 和が定義された objects の 2 つに対して 1 つの object を与える演算 f を，objects を成分とするベクトルまたは行列 m と n の演算に拡張する
71. cmpsimple(p,q|comp=t)
:: 式 p と q の簡単さを比較
72. simplify(p,ℓ,t|var=[$x_1, x_2, \ldots$])
:: $\ell = [\ell_0, \ell_1]$ のときは，p（の各要素毎）に subst(*,ℓ_0,ℓ_1) を調べてより簡単なら置き換える ($t = 1 \sim 7$)．$\ell = [\ell_1]$ で ℓ_1 が多項式のときは，ℓ_1 に一次に含まれる含まれる変数の線形関係式とみて簡単化する．複数調べるときは ℓ をリストや多項式のリストとする．
73. getel(m,i)
:: m がリスト，ベクトル，行列で i が非負整数なら $m[i]$ を返す

2.2. Numbers, polynomials and rational functions

74. abs(p)
:: 整数または実数 p の絶対値を返す
75. calc(p,[s,q]), calc(p,s)
:: 数や有理式に対して演算を施す
76. isint(p)
:: p が整数かどうか調べる
77. isvar(p)
:: p が変数かどうか調べる
78. radd(p,q)
:: 有理式（の行列）p と q の和を既約有理式（の行列）の形で計算する
79. rmul(p,q)
:: 有理式（の行列）p と q の積を既約有理式（の行列）の形で計算する
80. polbyroot([$p_1, p_2, ..., p_n$],x)
:: 多項式を根で与える
81. polbyvalue([[a_1, b_1],...,[a_n, b_n]],x)
:: x の $n-1$ 次多項式を n 個の点 $x = a_i$ での値 b_i で与える
82. pgen([[x_1, n_1],[x_2, n_2]...],a|sum=n,shift=m,sep=1,num=1)
:: 係数が a_* で x_i が n_i 次，全体で n 次以下の $x_1, \ldots$ の一般多項式を作る
83. rpdiv(p,q,x)
:: x の多項式の割り算
84. easierpol(p,x) または easierpol(p,[$x_1, x_2, \ldots$])
:: 有理式係数の x の多項式の係数に x を含まない有理式をかけて，係数の最大公約元が 1 の整数係数

の多項式に変換

85. `getroot(`p`,`x`|mult=1)`
:: 多項式の根を有理式の範囲で求める
86. `polinsym(`p`,[`$x_1,\ldots,x_n$`],`s`)`
:: $(x_1,\ldots,x_n)$ の対称有理式を基本対称式で表す
87. `polinvsym(`p`,[`$x_1,\ldots,x_n$`],`s`)`
:: `polinsym(`p`,[`$x_1,\ldots,x_n$`],`s`)` の逆関数
88. `pol2sft(`p`,`x`|sft=`t`)`
:: shifted power 多項式を与える
89. `polinsft(`p`,`x`)`
:: shifted power 多項式に直す（`pol2sft()` の逆変換）
90. `fctrtos(`r`)` または `fctrtos(`r`|TeX=t,var=`x`)`
:: 有理式を因数分解した形の文字列に変換する
91. `mulsubst(`r`,[[`$p_{1,0},p_{1,1}$`],[`$p_{2,0},p_{2,1}$`],...])`
:: 有理式またはそのリスト，ベクトル，行列 r に複数の代入 $p_{j,0}\mapsto p_{j,1}\quad(j=1,2,\ldots)$ を同時に行う（`[[x,y],[y,x]]` で `x` と `y` を交換）
92. `tohomog(`r`,[`$x_1,x_2,\ldots$`],`y`)`
:: $(x_1,x_2,\ldots)$ の有理式に変数 y を導入して $(y,x_1,x_2,\ldots)$ の斉次式にする
93. `substblock(`p`,`x`,`q`,`y`)`
:: x の多項式 p,q に対し，$y=q$ とおいて p を x の次数が `mydeg(`q`,`x`)` 未満の (x,y) の多項式に直す．
94. `invf([`$p_1,\ldots,p_n$`],[`$x_1,\ldots,x_n$`],[`$y_1,\ldots,y_n$`])`
:: $y_j=p_j(x)\quad(j=1,\ldots,n)$ を $x_j=q_j(y)\quad(j=1,\ldots,n)$ と解く
95. `mydeg(`p`,`x`|opt=1)`
:: `deg(`p`,`x`)` と同じ．p は行列や配列で係数は有理式でよい．
96. `mymindeg(`p`,`x`|opt=1)`
:: p がスカラーのときは `mindeg(`p`,`x`)` と同じ．係数は有理式でよいが，p が行列などのスカラーでないときは 0 以外の成分の最小次数を返す．
97. `mycoef(`p`,`n`,`x`)`
:: `coef(`p`,`n`,`x`)` と同じ．p は行列や配列で係数は有理式でよい．
98. `pcoef(`p`,`m`,`q`)`
`pcoef(`p`,`m`,[[`$x_1,..,x_n$`],[`$m_1,\ldots,m_n$`]])`
:: 多項式 p^m を展開したときの単項式 q に対する係数を返す
99. `cterm(`p`|var=[`$x,y,\ldots$`])`
:: 多項式の定数項を返す．変数を指定可能．
100. `mydiff(`p`,`x`)`
:: `diff(`p`,`x`)` と同じ．p は行列や配列で係数は有理式でよい．
101. `myediff(`p`,`x`)`
:: `ediff(`p`,`x`)` と同じ．p は行列や配列で係数は有理式でよい．
102. `ptol(`p`,`x`|opt=0)`
:: x の多項式 p の係数のリストを返す
103. `pfrac(`p`,`x`)`
:: x の有理式 p を部分分数展開し，分子，分母（多項式とべき）の組のリストを返す
104. `lpgcd([`$p_1,p_2,\ldots$`])`
:: 多項式 $p_1,p_2,\ldots$ の共通因子を返す
105. `prehombf(`p`,`q`|mem=`±1`)`
:: 概均質ベクトル空間の相対不変式 p の b 函数を得る．q は双対多項式．
106. `intpoly(`p`,`x`)`
:: 変数 x の多項式 p の原始関数で，定数項が 0 のものを返す．
107. `powsum(`n`)`
:: $1^n+2^n+\cdots+m^n$ の m を x で置き換えた $n+1$ 次多項式を返す
108. `bernoulli(`n`)`
:: n 次の Bernoulli 多項式 $B_n(x)$ を返す

2.3. Lists and vectors

109. `findin(`m`,[`$\ell_0,\ell_1,\ldots$`])`

:: m に等しい要素を $\ell_0, \ell_1, \ldots$ から探す

110. `countin(`s`,`m`,[`$\ell_0, \ell_1, \ldots$`])`
:: s 以上 m 以下の $\{\ell_0, \ell_1, \ldots\}$ の要素の個数を返す

111. `mycat([`$\ell_1, \ldots, \ell_m$`]|delim=`s`)`

112. `mycat0 ([`$\ell_1, \ldots, \ell_m$`],`t`|delim=`s`)`
:: $\ell_1, \ldots, \ell_m$ を表示する

113. `vtozv(`v`)`
:: 有理式のベクトルをスカラー倍して単純化する

114. `mulseries(`v_1`,`v_2`)`
:: 2 つのベクトルをべき級数とみなして，その積のベクトルを返す

115. `pluspower(`p`,`x`,`r`,`m`)`
:: $(1+p)^r$ の x に関するべき級数展開を第 m 項まで求める

116. `vprod(`v_1`,`v_2`)`
:: 2 つのベクトルの内積を返す

117. `llbase(`v`,`ℓ`)`
:: 変数 $\ell[0], \ell[1], \ldots$ の一次方程式のベクトル v の標準変換を行う

118. `lsol(`v`,`l`)`
:: 変数 $\ell[0], \ell[1], \ldots$ に関する連立一次方程式を解く

119. `lnsol(`v`,`l`)`
:: 変数 $\ell[0], \ell[1], \ldots$ に関する連立一次方程式の有理数解を求める

120. `lsort(`ℓ_1`,`ℓ_2`,`t`)`
:: リスト ℓ_1 に対し，ℓ_2 との合併，共通部分，または ℓ_2 や共通部分を除く
$t =$ `"cup", "setminus", "cap", "reduce"`

121. `vnext(`v`)`
:: ベクトルの成分を並べ替え，辞書式順序で次のベクトルに変換する

122. `vgen(`v`,`w`,`m`|opt=0)`
:: 成分の和が m の非負成分のベクトル w を順に生成する

2.4. Matrices

123. `dupmat(`m`)`
:: 成分が有理式の行列またはベクトル m の複製を作る

124. `m2v(m)`
:: 行列 m の成分を 1 行目から順に並べてベクトルに変換する

125. `m2l(m)`
:: 有理式，ベクトルあるいは行列 m の成分を順に並べてリストにする

126. `m2lv(m)`
:: 行列 m の行ベクトルを並べてリストにする

127. `lv2m(`l`)`
:: 行ベクトル（行成分のリストでも可）のリスト l から行列を作る

128. `s2m(s)`
:: 有理数成分の行列を文字列で作る．行列サイズは必要最低サイズ

129. `m2diag(m,n)`

130. `mperm(`m`,[`$\sigma_0, \ldots, \sigma_{m-1}$`],[`$\tau_0, \ldots, \tau_{n-1}$`])`
`mperm(`m`,[[`σ_0, σ_1`]],[[`τ_0, τ_1`]])`
:: 行列 m を置換行列（または互換）で変換，または小行列 $\left(m_{\sigma_i \tau_j}\right)$ を作る．

131. `mtranspose(`m`)`
:: 行列 m の転置行列を求める

132. `mpower(`m`,`n`)`
:: 行列 m の n 乗を求める

133. `mtoupper(`m`,`n`)` `mtoupper(`m`,`n`|opt=1)`
:: 行列 m に対し，行の基本変形を行って，行の先頭からの 0 の成分の個数が下の行の方へ狭義単調増加となるようにする．最後の n 列は無視．

134. `mydet(`m`)` または `mydet2(`m`)`
:: 共に `det(`m`)` と同じ．ただし成分は有理式でもよい．

135. myrank(m)
:: 行列 m の rank を求める
136. mykernel(m | opt=1)
:: 行列 m の転置行列の kernel の基底を求める
137. myimage(m | opt=1)
:: 行列 m の転置行列の像の基底を求める
138. mymod(v,[$v_1,\ldots,v_k$] | opt=ℓ)
:: ベクトル $v_1,\ldots,v_k$ で張られる空間の商空間への v の射影を求める
139. mmod(m,[$v_1,\ldots,v_k$] | opt=1)
:: ベクトル $v_1,\ldots,v_k$ で張られる空間の商空間への線形写像 m の射影
140. myinv(m)
:: 正方行列 m の逆行列を求める
141. madj(g,m)
:: 行列 gmg^{-1} を計算する．m は行列のリストか行列のベクトルでもよい.
142. mgen(m,n,a,s|sep=1)
:: size $m \times n$ の一般行列を作成する
143. newbmat(m,n,[[$r_{00},r_{01},\ldots$],[$r_{10},\ldots$],$\ldots$])
:: ブロック行列を作成する
144. meigen(m|mult=1)
:: 行列 m の固有値を返す
145. mdsimplify(m|show=1,type=ℓ)
:: 有理式正方行列 m または，そのリストやベクトルを対角行列で簡単化

2.5. Strings

146. str_str(s,t)
:: 文字列 s に部分文字列 t が最初に現れる場所を返す
147. str_subst(s,s_0,s_1) または str_subst(s,[$s_{00},s_{01},\ldots$],[$s_{10},s_{11},\ldots$])
str_subst(s,[[s_{00},s_{10}],[s_{01},s_{11}],$\ldots$],0)
:: 文字列 s に含まれる部分文字列 s_0 を s_1 で全て先頭から順に置き換える
148. str_tb([$s_0,s_1,\ldots$],tb) または str_tb(0,tb)
:: テキスト用バッファ tb に文字列 $s_0,\ldots$ を順に追加する，または取り出す

2.6. Permutations

149. ldict(n,m | opt=t)
:: $\{0,1,2,\ldots,m-1\}$ を並べ替えて辞書式順序で $n+1$ 番目のリストを返す
150. ndict(ℓ | opt=t)
:: $\{\ell_0,\ell_1,\ldots,\ell_{m-1}\}$ の並べ替えのリスト ℓ が何番目かを返す（最初は 0 番）
151. nextsub([$a_0,\ldots,a_{m-1}$],n)
:: $\{0,\ldots,n-1\}$ の m 個の部分集合を並べたとき，$\{a_0,\ldots,a_{m-1}\}$ の次の部分集合を返す．最後を与えた時は 0 を返す.
152. nextpart(ℓ)
:: 自然数の分割のリスト ℓ に対し，辞書式順序で次に大きなものを返す
153. transpart(ℓ)
:: 自然数の分割のリスト（Young 図式に対応）ℓ に対し，その双対を返す
154. trpos(a,b,n)
:: 互換 (a,b) にあたる n 次置換群の元を返す
155. sprod(s,t)
:: 置換群の積を返す
156. sinv(s)
:: 置換 s の逆元を返す
157. slen(s)
:: 置換 s の長さを返す
158. sord(s,t)
:: 置換を Bruhat order で比較する

159. my_tex_form(p|subst=[t_0,t_1])
:: print_tex_form() の戻り値から文字列置換や不要部分削除を行い，読みやすいソースに変換
160. dviout(p|clear=1,keep=1,delete=t,fctr=1,mult=1,subst=[$s0,s1$],eq=t,title=s)
:: p を dviout で表示する
161. dviout0(ℓ) または dviout0([$\ell_1,\ell_2,\ldots$])
:: TEX での表示のための内容削除などの基本操作
162. verb_tex_form(p)
:: p を LATEX で表現可能な文字列にする
163. monotos(p)
:: 有理式を文字列に変換．単項式以外では (と) で囲む
164. monototex(p)
:: 有理式を TEX の文字列に変換．単項式以外では (と) で囲む
165. rtotex(p)
:: 数式を TEX の文字列に変換．1 文字を越えるときは { と } で囲む
166. ltotex(l|opt=s,pre="string",small=1)
:: リストまたはベクトルを $s=$ "spt" のとき重複度つきのリストとみて\left\{... または $s=$ "GRS" のとき \begin{Bmatrix} ⋯ の形の Riemann scheme とみて TEX の文字列に変換など
167. mtotex(m|small=1,null=1,2)
:: 行列を TEX の文字列に変換するが，成分が有理式のときは因数分解した形
168. smallmattex(s)
:: TEX のソースで ( ) や { } で囲まれた行列を小サイズに変換する
169. AMSTeX
:: この値が 1 は $\mathcal{AMS}$LATEX を意味する
170. DVIOUT
:: dviout のパス名．myhelp() で指定した関数の解説を示すのに使われる．
171. TeXEq
:: デフォルトの LATEX の数式環境の指定

参考文献

[K] N. M. Katz, *Rigid Local Systems*, Annals of Mathematics Studies **139**, Princeton University Press, 1995.

[MO] T. Matsuki and T. Oshima, Embeddings of discrete series into principal series, The Orbit Method in Representation Theory, Proceeding of a Conference Held in Copenhagen, August to September 1988, 1990, 147–175, Birkhäuser.

[OO] H. Ochiai and T. Oshima, Commuting differential operators of type B_2, Funkcialaj Ekvacioj **46** (2003), 297–336.

[OOS] H. Ochiai, T. Oshima and H. Sekiguchi, Commuting families of symmetric differential operators, Proc. Japan Acad. **70 Ser.A** (1994), 62–67.

[O1] 大島利雄, コンピュータの速度の進化, 1985–2013, http://akagi.ms.u-tokyo.ac.jp/~oshima/computer/speed.htm

[O2] ——, dviout, Ver.3.18.5, 1990–2013, ftp://akagi.ms.u-tokyo.ac.jp/pub/TeX/dviout/3.18/tex318w.zip

[O3] ——, Okubo, a computer program for Katz/Yokoyama/Oshima algorithms for spectral types, 2007-8, ftp://akagi.ms.u-tokyo.ac.jp/pub/math/okubo/okubo.zip

[O4] ——, muldif.rr, a library of the calculation of differential operators for computer algebra Risa/Asir, 2007–2014, ftp://akagi.ms.u-tokyo.ac.jp/pub/math/muldif/

[O5] ——, 特殊関数と代数的線型常微分方程式, 東京大学数理科学レクチャーノート **11**, 2011, 廣惠一希記, http://www.ms.u-tokyo.ac.jp/publication/documents/spfct3.pdf

[O6] T. Oshima, *Fractional calculus of Weyl algebra and Fuchsian differential equations*, MSJ Memoirs **28**, Mathematical Society of Japan, Tokyo, 2012.

[OS] T. Oshima and N. Shimeno, Heckman-Opdam hypergeometric functions and their specializations, RIMS Kôkyûroku Bessatsu **B20** (2010), 129–162.