数式処理 *J.JSSAC (2004)*
Vol. 11, No. 1, pp. 17 - 28

特集「KNOPPIX/Math」

Singular と Surf

横田博史*

概　要

Many CAS are included in Knoppix/Math. In this article, I mention to some introductory examples for Singular and how to draw algebraic curves and surfaces on Singular with surf.lib.

1 概要

Singular は可換代数、代数幾何と特異点理論に特化した計算機代数システムである。Singular は C 言語風の処理言語を持っており、この処理言語で記述したライブラリによって機能を拡張して行く事が可能である。Singular 自体は GUI、エディタやグラフ表示機能を持っていないが、Emacs や Surf といった外部プログラムで代用する事が可能である。

次に、Surf は二次元平面内の代数曲線、三次元ユークリッド空間に埋込まれた代数曲面を描写するものである。この Surf は、機能的に制約のある C 言語風の処理言語を持っている。但し、Surf は基本的に与えられた函数の表示を行うもので、与えられた式の簡約化といった複雑な数式処理は出来ない。

2 Singular 初歩

Singular での入力では、末尾にセミコロン; を必ず付ける。セミコロンが無い場合には、Singular は入力が継続していると判断して入力行の評価を行わず、プロンプトを">"から"."にしている。

Singular の対象 (Object) には全て型 (Type) がある。整数の和、差、積のみは、そのまま入力して、結果がちゃんと帰って来るが、それ以外は最初に基礎環 (Base Ring) を定義しなければならない。

ここで基礎環の定義は次の様にして行う：

```
ring 環の名前=係数体,(変数 1,... ,変数 n),順序;
```

*ponpoko@cap.bekkoame.ne.jp

ここで、係数体は実数 $\mathbb{R}$、複素数 $\mathbb{C}$、有理数 $\mathbb{Q}$、整数 $\mathbb{Z}$ 等が選べ、標数 p は 2147483629 以下の整数が指定可能である。更に、指定可能な順序の代表的なものには、辞書式順序 (lp)、斉次辞書式順序 (Dp)、斉次逆辞書式順序 (dp) と重み付き逆辞書式順序 (wp) と重み付き辞書式順序 (Wp)、負辞書式順序 (ls)、負斉次逆辞書順序 (ds)、負斉次辞書順序 (Ds)、負重み付き逆辞書式順序 (ws) と負重み付き辞書式順序 (Ws) 等が選べる。順序の詳細に関しては Singular のマニュアルや参考文献 [1, pp.13-14] を参照されたい。

基礎環を定義すれば、その基礎環上の多項式 (poly) やイデアル (ideal)、更に商環 (qring) が扱える。以下に多項式、イデアルと商環の定義方法を示す：

```
poly   多項式名=多項式;
ideal  イデアル名=多項式_{1},...,多項式_{n};
ideal  商環名=イデアル;
```

次に、商環 R/I の定義で指定するイデアル I は単純に `ideal` で定義したものも使えるが、その場合はイデアルが標準基底で無いと警告する。イデアルの標準基底は `std(イデアル);` で得られるので、そちらを使う方が警告が出なくて良い。尚、標準基底を計算する命令に `groebner` もあるが、`std` 命令と違い `groebner` は環の順序に制約がある。

Singular は入力された大文字と小文字を判別する事に注意する。又、一度定義した対象は、名前を Singular に入力すればその内容が表示される。

次に各対象の定義例を示す：

```
> ring r=0,(x,y,z),dp;
> poly unit_circle_eq=x^2+y^2-1;
> ideal unit_circle=x^2+y^2-1;
> unit_circle_eq;
x2+y2-1
> unit_circle;
unit_circle[1]=x2+y2-1
> ideal points=unit_circle_eq,y-x;
> points;
points[1]=x2+y2-1
points[2]=-x+y
> points[1];
x2+y2-1
>
```

この例では基礎環として $r = \mathbb{Z}[x,y,z]$ を定義し、それから多項式の unit_circle_eq とイデアルの points を定義している。この様に、Singular では代入は=を用いる。対象の内容は単純に対象名を入力すれば多項式 unit_circle_eq の様に返される。イデアルは番号付で返されるので、イデアルを生成する多項式を取り出したい時は、この例の points[1] の様にすれば一番目の生成元が取り出せる。ここで、取り出した生成元の型は多項式 (poly) 型である。

次に、商環の定義例を示す：

```
> qring ri=std(x^2+y^2-1);
> ri;
//   characteristic : 0
//   number of vars : 3
//        block   1 : ordering dp
//                  : names    x y z
//        block   2 : ordering C
// quotient ring from ideal
_[1]=x2+y2-1
> r;
//   characteristic : 0
//   number of vars : 3
//        block   1 : ordering dp
//                  : names    x y z
//        block   2 : ordering C
```

この例では前の基礎環 $\mathbb{Z}[x,y,z]$ 上で、商環の定義で必要なイデアルを std(x^2+y^2-1) で与え、商環 ri を $\mathbb{Z}[x,y,z]/\left\langle x^2+y^2-1\right\rangle$ で定義している。

Singular では環を複数定義出来るが、定義する毎にポインタが新規に定義された環に移動して行く。多項式、イデアル、写像等はポインタがある環上で定義されるので、必要に応じてポインタを戻す必要がある。ポインタをある環に移す場合には、setring 命令を用いる。使い方は setring 環の名前; と環の名前を直接指定すれば良い。

Singular では写像が扱える。写像の定義では map 函数を用いるが、この場合、対象が含まれる環を基礎環にした状態で写像の定義を行う。以下に写像の定義例を示しておく：

```
> ring r1=0,(x,y,z),dp;
> ring r2=0,(a,b),dp;
> map f=r1,a,b,0;
> f;
f[1]=a
f[2]=b
f[3]=0
>
```

この例では、環 r1 と環 r2 を各々 $\mathbb{Z}[x,y,z]$、$\mathbb{Z}[a,b]$ とし、r1 から r2 への写像 $f(x,y,z) \to (a,b,0)$ を値域となる基礎環の変数に対して一つ一つ定めれば良い。

更に、写像によるイデアルの逆像も計算可能である。例えば、上の例で r2 のイデアル i2 の f による逆像は `preimage(r1,f,i2);` で計算可能である。次の例ではイデアル i2 を零イデアルとして、f の核を計算している：

```
> ring r1=0,(x,y,z),dp;
> ring r2=0,(a,b),dp;
> map f=r1,a,b,0;
> ideal i2=0;
> setring r1;
> preimage(r2,f,i2);
_[1]=z
>
```

尚、この例では零イデアルを `ideal i2=0;` で定義しているが、単純に、`ideal i2` とすると、`i2=0` として定義される。

Singular はその処理言語を用いてライブラリが構築出来る。又、ライブラリの読込は LIB 命令 (大文字) を用いる。Singular には様々なライブラリが附属している。この標準で附属しているライブラリの中に、Singular で定義した函数を Surf を用いて可視化する `surf.lib` が含まれている。次の節では Singular で求めた多項式函数をこの `surf.lib` を用いて描く方法について簡単に解説する。

3 Singular から surf を使う

ここでは Singular から Surf を用いて図形表示する方法について簡単に述べる。Surf の詳細についてはマニュアル [2] を参照されたい。

Singular から Surf を使って曲線や曲面を描く為には予めライブラリ surf.lib を読込んでおかなければならない。Singular 言語で記述されたライブラリの読込は LIB 命令で、LIB "surf.lib"; の様に実行する。尚、参考文献 [1] に附属の CD-ROM に収録されている Singular-2.0.3 の surf.lib では曲線を描けない問題 (曲面を描いてしまう) があるが、KNOPPIX/Math に収録されている Singular-2.0.4 以降では修正されている。

Singular から Surf を使って描けるものは、基本的に一つの多項式、又は一つの単項生成イデアルに限定される。描く対象が曲線か曲面であるかどうかは、基礎環の変数の個数から自動的に決定している。又、曲面の場合、Singular からインタラクティブに視点を変更するといった機能も無いが、Surf の命令を引き渡す事が可能なので、多少の融通なら効く。但し、surf.lib では引き渡された命令を単純に Surf のスクリプトの所定の位置に挿入するだけなので、引き渡した命令が結果に反映されるとは限らない事に注意する。

次に Singular を立ち上げてから曲線を描くまでの処理を示す：

```
yokota@kahn:~> Singular-2-0-5
                     SINGULAR                                 /
 A Computer Algebra System for Polynomial Computations   /   version 2-0-5
                                                        0<
     by: G.-M. Greuel, G. Pfister, H. Schoenemann         \   March 2004
FB Mathematik der Universitaet, D-67653 Kaiserslautern    \
> ring r=0,(x,y),dp;
> LIB "surf.lib";
// ** loaded /opt/Singular/2-0-5/LIB/surf.lib (1.19.2.6,2002/07/17)
> plot(x^3+x^2-y^2);
```

この例で示す様に、surf.lib を用いて図形を描く為には、何よりも、多項式かイデアルが定義可能な状態になっている必要があり、その為、最初に基礎環を定義しなければならない。ここでは基礎環として、$\mathbb{Z}[x,y]$ を定義している。尚、surf.lib の読込は LIB "surf.lib"; で行うが、これは対象を描く直前に読込まれていれば良く、立ち上げて直ぐに読込んでいても構わない。このライブラリの読込後に plot(x^3+x^2-y^2) を実行すれば、Singular は Surf 向けのスクリプトを生成して呼出した Surf に引き渡し、Surf は図 1 に示す曲線を描く。

この surf.lib では、Surf に命令を引き渡す事も可能である。この場合、セミコロンで Surf の命令を区切ったものを plot 命令に引き渡せば良い：

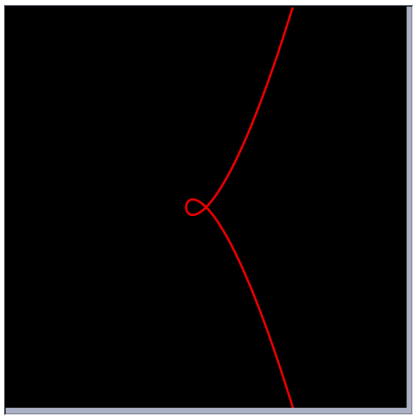

図 1: plot($x^3 + x^2 - y^2$) で描いた曲線

```
plot(多項式,"命令 1; 命令 2;... 命令 n;");
```

但し、Surf に引き渡されても効果の無い場合もある。これは引き渡した命令が挿入される場所によって結果として無意味な事がある為である。一番確実なのは、直接 Surf 側で操作する事である。その為には (図 1 に示す) グラフが表示されているウインドウを右クリックして、図 2 に示す Surf の画面を呼出せば良い。ここで、Singular で生成したスクリプトは左側の入力欄にある。

この入力欄に命令の追加や修正を行った後、右側の `execute script` ボタンを押せば直ちに描写を開始する。又、途中で処理を止めたければ、右下の `Stop` ボタンを押せば良い。最後に Surf を終えたければ、グラフ上で `q` を入力するか、制御ウインドウの File メニューから Quit を選べば良い。尚、`plot` 命令で引き渡した Surf の命令に誤りがある場合、このウインドウが呼出され、問題の個所にカーソルが置かれている。

この様に `surf.lib` を用いた場合、Singular から会話的に操作が行えず、詳細は Surf のスクリプトを調整して描き直す必要がある。この `surf.lib` を改良したライブラリも幾つか存在しているが、Surf を用いて図形を描くもので現在の Singular に同梱されているライブラリは `surf.lib` のみである。

`surf.lib` を用いて描けるイデアルは基本的に単項イデアルに限定される。次に、前述の `preimage` を用いて、イデアルが定める曲面を描く例を示す：

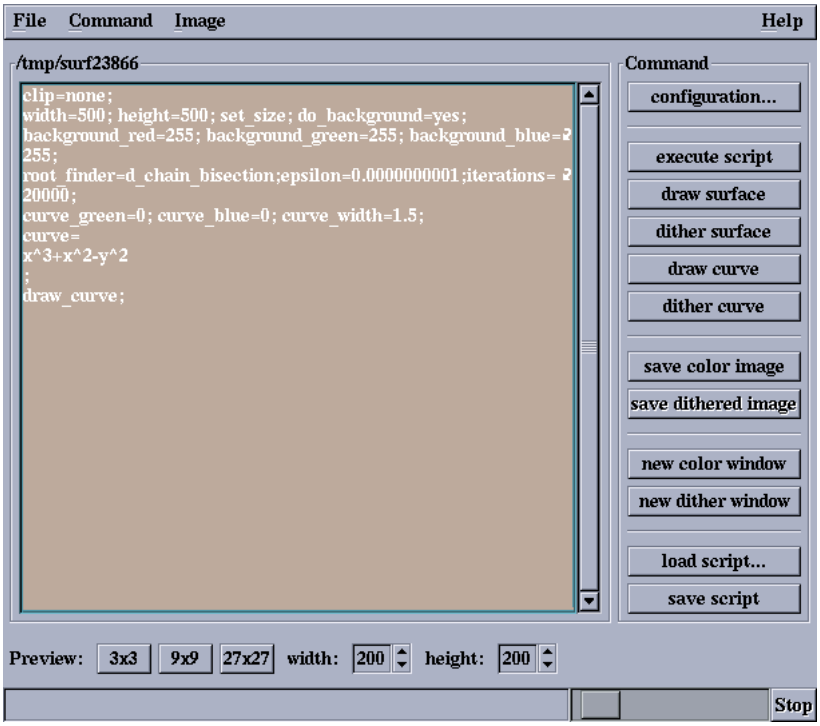

図 2: Surf の制御ウインドウ

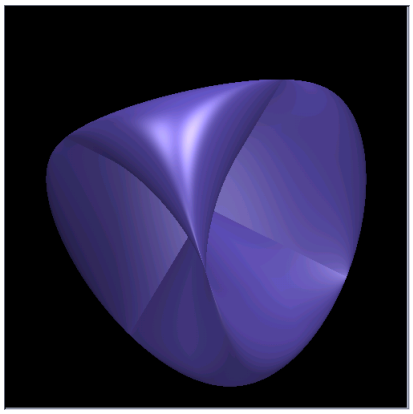

図 3: シュタイナーのローマ曲面

```
> ring r=0,(x,y,z),dp;
> poly sp4=x2+y2+z2-16;
> ideal i1=sp4;
> map f=r,xy,yz,zx;
> ideal steiner=preimage(r,f,i1);
> steiner;
steiner[1]=x2y2+x2z2+y2z2-16xyz
> plot(steiner,"background_red=0;background_green=0;
. background_blue=0;rot_x=2;rot_y=0.5;");
```

この例では、写像 $f : (x, y, z) \to (xy, yz, zx)$ による半径 4 の三次元球面の逆像を preimage で求め、plot 命令に Surf の背景色を設定する命令と一緒に引き渡している。尚、この背景色の設定が長い為に途中で改行を入れているが、Singular は行末のセミコロンが未入力の為に、行が継続中であると判断して、ピリオド . を出している事に注意されたい。

尚、曲線や曲面が、助変数表示されている場合、そのまま plot に助変数が定義するイデアルを引き渡しても、plot で図形を描けない。しかし、eliminate 命令を併用して描く事が可能な場合がある。この eliminate 命令は不要なイデアルの変数を削除する命令である。以下に、eliminate 命令を用いて図 4 に示すデカルトの葉状曲線を描く例を示す：

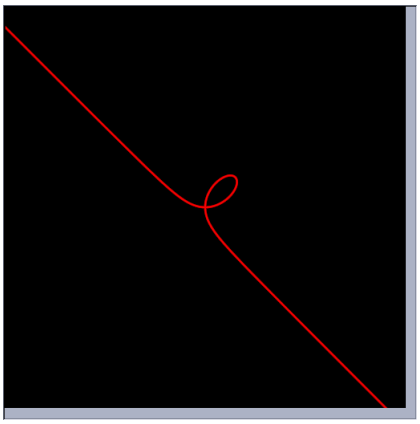

図 4: デカルトの葉状曲線

```
> ring r1=0,(x,y,t),dp;
> ring r2=0,(x,y),dp;
> map f=r1,x,y,0;
> setring r1;
> ideal i1=(1+t^3)*x-3*t,(1+t^3)*y-3*t^2;
> ideal i2=eliminate(i1,t);
> poly c2=i2[1];
> c2;
x3+y3-3xy
> setring r2;
> plot(f(c2));
```

この例では、助変数を含めた環 r1 とそれを除いた環 r2、更に、r1 から r2 への射影 f を定義し、次に、r1 上でイデアル i1 を定義を定義している。この葉状曲線は助変数 t を用いて $x = \frac{3t}{1+t^3}, y = \frac{3t^2}{1+t^3}$ で表現されているが、基礎環 r1 が $\mathbb{Z}[x,y,t]$ の為、イデアル i1 の定義で分子をかけた形にしている。次の eliminate 命令で、助変数 t を削ったイデアル i1 の表現を求めている。この eliminate 函数ではイデアルが返され、それをイデアル i2 と定義している。次に、イデアルは生成元を c1 に示す様に取り出せる。それから基礎環を setring 命令で r2 にして、写像 f による c1 の像を plot 命令で描いた結果が図 4 である。ここで r1 ではなく r2 上で曲線を描いた理由は単純に surf.lib では環の変数の総数で曲線と曲面を判別する為、r1 で (x,y,t) の三変数の為に曲面が描かれる為である。

次に、助変数表示された曲面を描く例として、猿の腰掛 (図 5) の例を以下に示す：

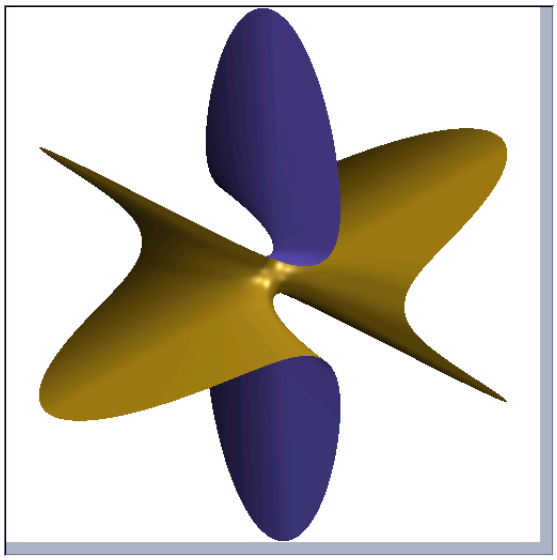

図 5: 猿の腰掛

```
> ring r3=0,(x,y,z,u,v),dp;
> ideal a1=x-u,y-v,z-u^3+3*u*v^2;
> ideal a2=eliminate(a1,uv);
> plot(a2);
```

この猿の腰掛では助変数に u,v を用いている。ここで、eliminate で削除する変数を複数指定する場合は単純に uv の様に並べれば良い。

ここで、Singular で生成した猿の腰掛のスクリプトを以下に示す：

```
root_finder=d_chain_bisection;epsilon=0.0000000001;iterations=20000;
width=500; height=500; set_size; do_background=yes;
background_red=255; background_green=255; background_blue=255;
rot_x=0.14; rot_y=-0.3;
surface=
x^3-3*x*y^2-z
;
draw_surface;
```

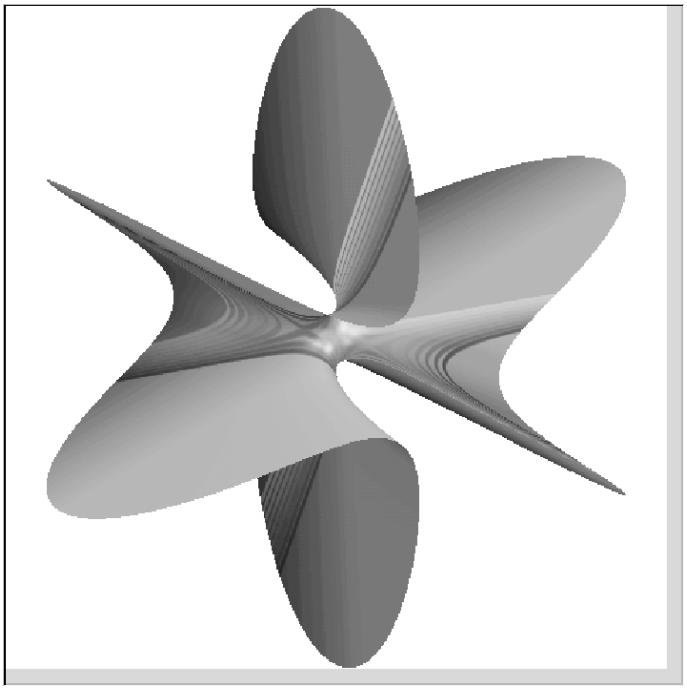

図 6: 猿の腰掛 (2)

このスクリプトの一行目には根を求める為のパラメータがある。次の行で表示画面の大きさ (500 × 500 ピクセル) が設定され、3 行目で背景色を定めている。背景色や曲線の色等は RGB で 0 から 255 迄の整数で指定する。4 行目には X 軸と Y 軸の回転が記述されている。尚、Surf には Z 軸回りの回転の `rot_z` もある。それから、変数 `surface` に引数の多項式が代入される。最後の行の `draw_surface` で実際に図形が描かれる。尚、曲線の場合、多項式は変数 `curve` に代入されて、`draw_curve` で描かれる。因に、`plot` 命令で引き渡された Surf の命令群は描く対象が曲面であれば `surface` の直前に、曲線であれば `curve` の直前に挿入される。

このスクリプトを利用して、背景色を黒、猿の腰掛を半透明にして、平面による断面を Surf を用いて表示してみよう。先ず、背景色を黒にする。その為には、`background_red`、green、blue の値を全て 0 にする。次に曲面の透明度は `illumination` と `transparence` で指定する。平面の方程式は `plane` 変数に代入し、`cut_with_plane` 命令で描く。ここで平面は $2*x+y-z-a$ とし、a を-2.0 から 2.0 の範囲で動かす。Surf でループを使う為には、`if` 文と `goto` 文を用いる必要がある。又、断面の色も a を動かすと変化する様にしたい。この場合、背景色と同様に、`curve_red`、green、blue の RGB$(0\cdots255)$ で設定すれば良い。最後に `cut_with_plane` で `plane` 変数に入れた平面による曲面の断面が描かれる。

```
root_finder=d_chain_bisection;epsilon=0.0000000001;iterations=20000;
width=500; height=500; set_size; do_background=yes;
background_red=0; background_green=0; background_blue=0;
illumination=ambient_light+
diffuse_light+reflected_light
+transmitted_light;
transparence=50;
clear_screen;
rot_x=0.1; rot_y=-0.7;rot_z=0;
surface=
x^3-3*x*y^2-z
;
draw_surface;
double a=-2.0;
int i=0;
loop:
plane=2*x+y-z-a;
curve_red=255-5*i;
curve_green=10*i;
curve_blue=0;
cut_with_plane;
a=a+0.5;
i=i+5;
if (a<=2.0) goto loop;
```

以上、駆け足で Singular から Surf を使って図形を描く事を解説した。この様に、Singular で計算した多項式を単純に表示するだけではなく、Surf の環境下で様々な処理が行える。

参 考 文 献

[1] Greuel, Pfister: A Singular Introduction to Commutative Algebra,Springer,Berlin,2001.

[2] Stephan Endrass: http://surf.sourceforge.net/doc/manual.html.