RegularChains パッケージを使った
より賢い方程式解法と動的システムの安定性解析

Maplesoft / サイバネットシステム（株）

はじめに

連立方程式系を解くことは、あらゆる数学的計算の中でもっとも頻繁に行なう手段と言えます。これは工学分野でも同様で、線形的なものであれば消去法等で、非線形的なものであれば多変数のためのニュートン法などを用いて、与えられた連立方程式系の解を計算していきます。

一方、数式処理システムである Maple では、連立方程式系に対して数値的な解を求めるだけでなく、より数学的洞察が可能な手段も用意されています。

この資料では、新バージョン毎に強化されている RegularChains パッケージを紹介し、工学をはじめとした応用問題について Maple 独自の連立方程式系に対する手法を解説します。

なお、本資料を理解するためには、多少の代数学の知識（群や環、剰余類、多項式環）が必要となりますが、不明な部分はそのまま読み飛ばして頂いて構いません。

RegularChains とは？

詳細な数学的定義は割愛しますが、ここでは簡単に RegularChains の定義について紹介します。
RegularChains とは、与えられた連立方程式系（多項式の集合）に対して、より簡易な多項式系の集合のことを意味します。また、RegularChains を求めることを、「多項式系を三角化する（Triangularization または Triangular Decomposition）」と定義します。

少しだけ厳密に表現するならば、上記は次のように表現できます。

p : 多項式系 S の解 ⇔ ∃ $i : p$はより簡易な多項式集合$\{S_1, S_2, \ldots, S_N\}$の中の S_i の解

これは、連立方程式の解法アルゴリズムとして有名なグレブナー基底とも似ている概念です。

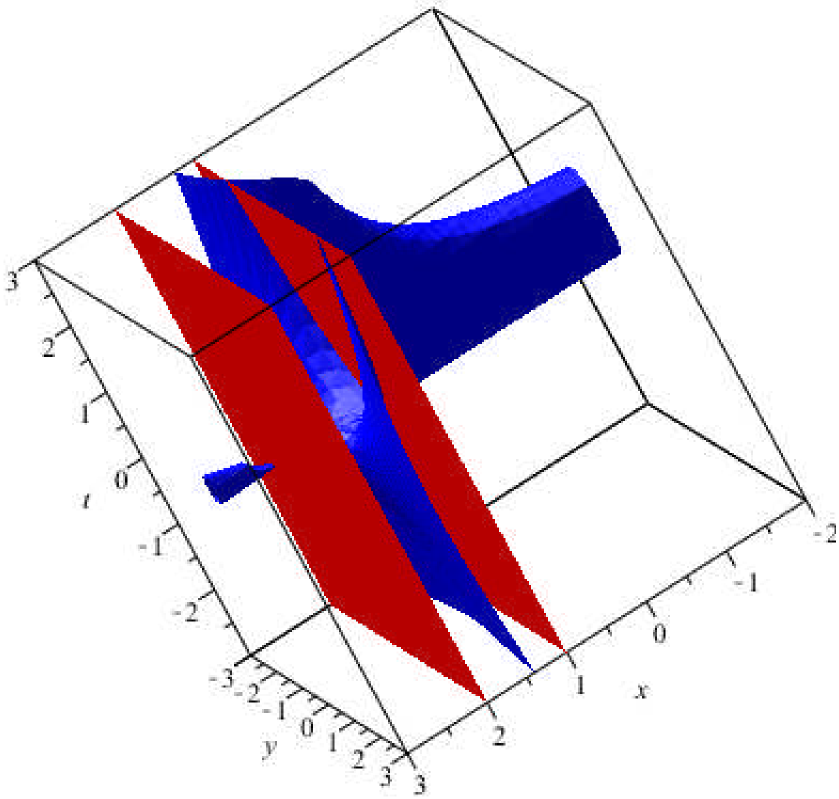
なお、各 $S_1, S_2, \ldots, S_N$ を代数系（algebraic system）と言います。特にその係数及び解が実数であることを想定する場合は、半代数系（semi-algebraic system）と言います。一つ一つの代数系は、方程式や不等式・非等式などからなる集合で、解を表現しているものです。

ここでは、半代数系を求めるとはどういうことかを、実例で理解してみましょう。
変数 x, y, t からなる次の連立多項式系を考えます。

```
> restart;
> F := [(x-1)*(x-2) = 0,
        (x-1)*(y^2+t^2)+(x-2)*(y^2-t)=0];
```

$$F := [(x-1)(x-2)=0, (x-1)(y^2+t^2)+(x-2)(y^2-t)=0] \quad \textbf{(2.1)}$$

```
> plots[implicitplot3d](
    F, x=-2..3, y=-3..3, t=-3..3,
    color=[red,blue],
    grid=[30,30,30],
    axes=boxed,
    style=surface
  );
```



この系はグラフにしてみると上記のようになります。この系の解を求めるというのは、二つの多項式で定義されている曲面の交点（もしくは交線）を求めるということになります。

この多項式系に対して、RegularChains パッケージを用いるには次のような手順のコマンドを用います。

```
> with(RegularChains):
> # 多項式環を定義
  P := PolynomialRing([y,x,t]);
```

$$P := polynomial\_ring \tag{2.2}$$

```
> # 定義した多項式環上で方程式系を実三角化
  rc := RealTriangularize(F, P, output=record);
```

$$rc := \left\{ \begin{array}{c} y = 0 \\ x - 2 = 0 \\ t = 0 \end{array} \right., \left\{ \begin{array}{c} y^2 - t = 0 \\ x - 1 = 0 \\ t > 0 \end{array} \right., \left\{ \begin{array}{c} y = 0 \\ x - 1 = 0 \\ t = 0 \end{array} \right. \tag{2.3}$$

ここで変数 P は、与えられた多項式の集合が属する多項式環を定義しています。（変数には順序があることに注意してください）
定義した多項式環 P は、RealTriangularize コマンドの2番目のパラメータとして指定されています。

上記のコマンド RealTriangularize は、多項式系 F を多項式環 P 上で実三角化する、という意味になります。また、RealTriangularize コマンドの結果（戻り値）は、基本的に半代数系となります。

上記のコマンドで得られた結果を考察してみます。
RealTriangularize コマンドによる結果の1番目と3番目の結果の半代数系を見てみましょう。

```
> rc[1],rc[3];
```

$$\left\{\begin{array}{c} y=0 \\ x-2=0 \\ t=0 \end{array}\right., \left\{\begin{array}{c} y=0 \\ x-1=0 \\ t=0 \end{array}\right. \quad (2.4)$$

これは、共に $t=0$ の場合における解を構成しています。つまり、 $t=0$ の場合、

$\{x-2=0, y=0\}, \{x-1=0, y=0\}$

が解となること、すなわち $\{x=2, y=0, t=0\}, \{x=1, y=0, t=0\}$ という空間上の点がこの系の解（交点）であることがわかります。実際、先に描画したグラフ（曲面）で、描画点数をより細かくすることで、この点の存在を見ることができます。

念のため、得られた解をそれぞれ元の方程式系に代入すると、確かに成立していることがわかります。

```
> eval(F, [x=2, y=0, t=0]);
  eval(F, [x=1, y=0, t=0]);
```

$$[0=0, 0=0]$$
$$[0=0, 0=0] \quad (2.5)$$

一方、2番目の解（半代数系）を見てみます。

```
> rc[2];
```

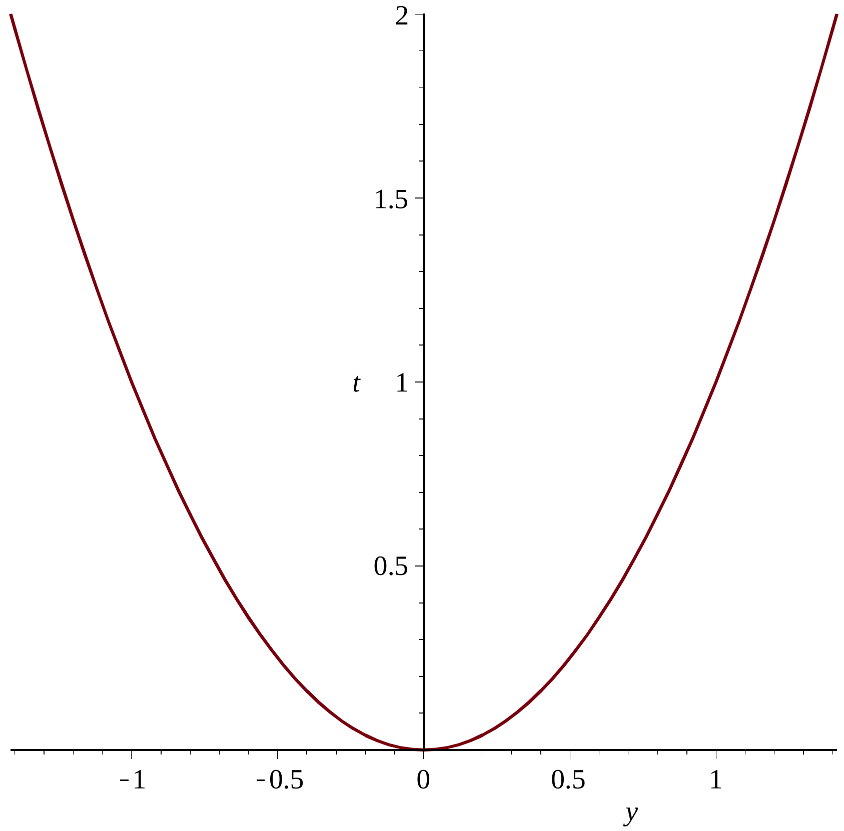
$$\left\{\begin{array}{c} y^2-t=0 \\ x-1=0 \\ t>0 \end{array}\right. \quad (2.6)$$

これは、$t>0$ の場合、つまり t の値が正である場合の解は、$x=1, y^2=t$ によるパラメトリックな形で解が構成されていることを意味しています。
実際、上記のグラフを見ると明らかなように、$t>0$ の場合は $x=1$ という平面と曲面による交線、すなわち $y^2=t$ という放物線が存在していることがわかります。これを具体的に確認するために、$x=1$ を方程式系に代入して、y, t のグラフを見てみます。

```
> Fyt := eval(F,x=1)[2];
```

$$Fyt := -y^2+t=0 \quad (2.7)$$

```
> plots[implicitplot](Fyt,y=-2..2,t=-2..2,gridrefine=2);
```



このように、RegularChains パッケージを用いた多項式系の分解（三角化）は、与えられた多項式系に対して単にある種の解（値）を求めるだけでなく、解の特性も含めて理解することが可能になります。

RegularChains パッケージを用いた多項式系の解法は、実係数を含む場合だけでなく、パラメータを含む場合にも機能します。
次の単純な多項式を考えてみます：

```
> f := x^2 + x - c;
```

$$f := x^2 + x - c \tag{2.8}$$

いま、この多項式 f が常に正となる場合（つまり $f > 0$）の x は、パラメータ c に関してどのように表現できるかを考えます。

通常、この種の方程式や不等式を解く場合、一般的には solve コマンドを用います。しかし、残念ながらこの種の問題に対して solve コマンドは有用な情報を提供しません。

```
> solve(f>0, [x,c]);
```

$$[[c < x^2 + x]] \tag{2.9}$$

そこで、RegularChains パッケージを用いて、解 x がパラメータ c を用いてどのように表現されるかを見てみます。

```
> # 多項式環の定義
R := PolynomialRing([x,c]);
```

$$R := polynomial\_ring \tag{2.10}$$

```
> # SemiAlgebraicSetTools サブパッケージの円筒的代数分解
# (CylindricalAlgebraicDecomposition) コマンドを適用
cad := SemiAlgebraicSetTools:-CylindricalAlgebraicDecompose(
   [[f>0]], R, output=cadcell
);
```

$$cad := [cad\_cell, cad\_cell, cad\_cell, cad\_cell, cad\_cell] \tag{2.11}$$

```
> # 計算結果を表示
Display(cad,R);
```

$$\left[\begin{cases} x = x \\ c < -\frac{1}{4} \end{cases}, \begin{cases} x < -\frac{1}{2} \\ c = -\frac{1}{4} \end{cases}, \begin{cases} -\frac{1}{2} < x \\ c = -\frac{1}{4} \end{cases}, \begin{cases} x < -\frac{1}{2} - \frac{\sqrt{4c+1}}{2} \\ -\frac{1}{4} < c \end{cases}, \begin{cases} -\frac{1}{2} + \frac{\sqrt{4c+1}}{2} < x \\ -\frac{1}{4} < c \end{cases}\right] \tag{2.12}$$

```
> # SamplePoint コマンドを上記結果に適用し、個々のサンプル値
# を計算
sp := map(SamplePoints, cad, R):
Display(sp, R);
```

$$\left[\begin{cases} x = 0 \\ c = -3 \end{cases}, \begin{cases} x = -2 \\ c = -\frac{1}{4} \end{cases}, \begin{cases} x = 1 \\ c = -\frac{1}{4} \end{cases}, \begin{cases} x = -3 \\ c = 1 \end{cases}, \begin{cases} x = 2 \\ c = 1 \end{cases}\right] \tag{2.13}$$

このように、RegularChains パッケージの個々のコマンド群を用いると、c の値（または範囲）によって $f > 0$ を満たす x の範囲を厳密に得ることが出来るようになります。

動的システムの平衡点に対する安定性解析

この章では、RegularChains パッケージを使って微分方程式による動的システムを、よりパラメトリックに解析する手段の一例を紹介します。

動的システム $\frac{d\,x(t)}{dt} = F(u, x)$ の平衡点とは、$F(u, x)$ の実零点を意味しています。
なお、ここで u はパラメータです。

いま、次の連立微分方程式で表現される動的システムを考えます：

```
> restart;
> with(RegularChains):
> sys :=
  [ diff(x(t),t) = -x(t)+s/(1+y(t)^2),
    diff(y(t),t) = -y(t)+s/(1+x(t)^2)];
```

(3.1)

$$sys := \left[\dot{x}(t) = -x(t) + \frac{s}{1+y(t)^2}, \dot{y}(t) = -y(t) + \frac{s}{1+x(t)^2}\right] \quad (3.1)$$

簡便のため、$x(t) = x, y(t) = y$ として上記の系を書き換えたものを sys2 とします。（右辺のみを取り出します）

```
> sys2 := map(rhs, eval(sys, [x(t)=x, y(t)=y]));
```

$$sys2 := \left[-x + \frac{s}{1+y^2}, -y + \frac{s}{1+x^2}\right] \quad (3.2)$$

次に、sys2 に対するフルビッツの安定性行列を計算します。

```
> H1 := -(diff(sys2[1],x) + diff(sys2[2],y));
  H2 := diff(sys2[1],x)*diff(sys2[2],y) - diff(sys2[1],y)*diff(sys2
  [2],x);
```

$$H1 := 2$$

$$H2 := 1 - \frac{4 s^2 y x}{(1+y^2)^2 (1+x^2)^2} \quad (3.3)$$

ここで得られたフルビッツの安定性行列から、漸近的に安定な平衡点を求めるために、次の各式を計算します。

```
> P := [numer(normal(sys2[1]))=0,
        numer(normal(sys2[2]))=0,
        numer(H2)>0,
        x>0,
        y>0,
        s>0];
```

$$P := [-x - x y^2 + s = 0, -y - y x^2 + s = 0, 0 < 1 + 2 x^2 + x^4 + 2 y^2 + 4 y^2 x^2 + 2 y^2 x^4 + y^4 + 2 y^4 x^2 + y^4 x^4 - 4 s^2 y x, 0 < x, 0 < y, 0 < s] \quad (3.4)$$

この方程式系に対して、RegularChains パッケージの ParametricSystemTools サブパッケージ内にある、 RealComprehensiveTriangularize コマンドを使って三角化を行ないます。このコマンドは、系にパラメータを含む場合の三角化を行ないます。

まず、計算を行なう多項式環を定義します。

```
> R := PolynomialRing([y, x, s]);
```

$$R := polynomial\_ring \quad (3.5)$$

続いて、パラメータ s に関する方程式系 P に対して三角化を行ないます。
RealComprehensiveTriangularize コマンドの2番目の引数で与えている整数値は、パラメータの個数、すなわちこの場合はパラメータ s ひとつであるため、 1 を与えています。

```
> with(ParametricSystemTools):
> rctd := RealComprehensiveTriangularize(P, 1, R);
```

$$rctd := [[[1, squarefree\_semi\_algebraic\_system], [2, squarefree\_semi\_algebraic\_system]], [[semi\_algebraic\_set, [\,]], [semi\_algebraic\_set, [1]], [semi\_algebraic\_set, [2]]]] \quad (3.6)$$

RealComprehensiveTriangularize コマンドの戻り値は、2つのデータから構成されます：

最初の要素は、無平方化された半代数系、2番目はパラメータ空間の分割に関する情報です。

得られた半代数系から、さらに2つの解を持つパラメータ空間、つまり双安定（bistable）なパラメータの領域を求めます。

```
> rctd2 := RealComprehensiveTriangularize(rctd, R, 2);
```

$$rctd2 := [[[1, squarefree\_semi\_algebraic\_system]], [[semi\_algebraic\_set, [1]]]] \tag{3.7}$$

戻り値 rctd2 の中で計算された半代数系には、パラメータ s が満たすべき値の範囲の情報が用意されています。

```
> Display(rctd2[2][1][1],R);
```

$$[2 < s] \tag{3.8}$$

さらに、rctd2 の最初の要素には、上記 s の範囲の中で解を2つ持つ（すなわち双安定な）領域の分割に関する情報が含まれています：

```
> Display(rctd2[1][1][2],R);
```

$$\begin{cases} x\,y - 1 = 0 \\ x^2 - s\,x + 1 = 0 \\ y > 0 \\ 8\,x\,s^3 - 6\,x\,s^5 - 4\,s^2 + 5\,s^4 - s^6 + x\,s^7 > 0 \\ x > 0 \end{cases} \tag{3.9}$$

なお、ここでいずれの等式や不等式において、x, y は共に s の関数として考えることが出来るという点について注意して下さい。式 **(3.8)** で得られたパラメータ範囲において、上記の領域 **(3.8)** でシステムはフルビッツ安定になります。

このように、RegularChains パッケージのコマンド群を用いることで、単に方程式系を（数式処理を用いて）解くだけではなく、解の存在領域（可解領域）を明確にすることが可能になります。得られた領域に関する情報は、その後の具体的な値探索（最適値探索）を適用する際にも利用できると共に、最適化の過程で生じる不適切な値の検定にも利用することが可能です。