Maple 14 における新規パッケージおよび改良されたパッケージ

Maple 14 に含まれる新規パッケージおよび改良された既存パッケージは以下のとおりです。

新規パッケージ

CUDA パッケージ

DifferentialAlgebra パッケージ

改良されたパッケージ

DEtools パッケージ

DynamicSystems パッケージ

Groebner パッケージ

IntegerRelations パッケージ

LinearAlgebra パッケージ

PDEtools パッケージ

RegularChains パッケージ

RootFinding パッケージ

StringTools パッケージ

Threads[Task] パッケージ

CUDA

- Maple では、NVIDIA の CUDA 技術を使用して LinearAlgebra の行列の乗算を加速する GPU のサポートが追加されました。新規の CUDA パッケージを使用して GPU 加速を有効にすることができます。詳細および例題については、 Efficiency Improvements in Maple 14 を参照してください。

DifferentialAlgebra

- 新規の DifferentialAlgebra パッケージは旧 Maple バージョンにおける diffalg に代わるパッケージで、多項式型の連立微分方程式 (ODE、PDE、DAE、PDAE) の単純化および操作と、主要計算をより高速にするために直接 C 言語で実行する改良された一連のコマンドおよび、より使いやすいインターフェイスを備えています。
- パッケージの主要機能は RosenfeldGroebner コマンドによって提供されます。このコマンドは微分方程式システムを 3 角形分割し、変数を 1 つずつ除去していくことで可積分条件に対してシステムを簡単化するか、特異解を求めて微分方程式システムを解きます。
- 微分イデアルに属するかを判定する BelongsTo、別の連立偏微分方程式に連立偏微分方程式を約する ReducedForm、偏微分方程式の形式的ベキ級数を計算する PowerSeriesSolution など、パッケージには関連問題に対応するためのコマンドも提供されています。微分システムの数学的特性を解析したり、代数的操作や関連プログラミングを実行するためなどに使用する他のコマンドについては、DifferentialAlgebra に記載されている一覧表を参照してください。
- DifferentialAlgebra パッケージの新機能を使用すると、より高速でより詳細な DE システムの解析が可能になります。新しい機能は次のような特徴を持ち合わせています：
  - 従属変数 (関数) 間で従属性が異なるシステムや従属性が全くないシステムを処理するための柔軟性を持ち合わせています。

- システムには任意の変数および関数を含めることができます。
- 一般解のみ、特異解のみ、またはすべての解 (デフォルトの動作) を計算するよう選択することができます。
- (任意関数の新しい扱い方に関連して) 非自明な微分体上で多項式型連立微分方程式を単純化することができます。
- 従属変数と (または) 独立変数の等級序列変更後に (イデアルで表現されている) 等価微分方程式システムを高速に再計算できる最新のアルゴリズムがパッケージに含まれています。

• 計算をより簡単、より速く、そしてより柔軟に実行できるように再設計されたインターフェイスについては、次のような特徴を持ち合わせています：

- より簡単に問題を入力でき、解を読みやすく、短い表記で返せるようにさまざま数学的表記が使用できるようになりました。入力に使用されている表記法を示す必要はありません。出力に使用する表記法は任意で指定可能です。
- 計算の基礎体を指定する必要がなくなりました。基礎体は、その場で、一貫して入力から自動的に導き出されます。
- 微分環を再定義する必要なく、微分環に含まれている従属変数や関数、独立変数や関数、任意変数や関数に変更または追加を行うことができます。
- PDE システムの任意変数や任意関数間の代数的関係を relations = {..equations..} の形式で RosenfeldGroebner in a natural way, as in relations = {..equations..}に直接渡す (指定する) ことができます。

パラメトリックな微分方程式システム

```
> with(DifferentialAlgebra);
```

[*BelongsTo*, *ChangeRanking*, *DifferentialRing*, *Equations*, *Get*, *Inequations*, *Is*, *NormalForm*, *PowerSeriesSolution*, *ReducedForm*, *RosenfeldGroebner*, *Tools*] (2.1.1)

パラメータ p にも依存する次の $u(x)$ についての非線形 ODE を検討します。

```
> ODE := (p-1)*diff(u(x),x)-u(x)^2;
```

$$ODE := (p-1)\left(\frac{d}{dx}u(x)\right) - u(x)^2 \tag{2.1.2}$$

次の微分環を使用すると、p' が (従属性がなく、すべての導関数がゼロの) 従属変数であるかのように ODE" を操作することができます。これにより、パラメータ化した ODE の概念を実現します。

```
> ring := DifferentialRing(derivations = [x], blocks = [u,p]);
```

ring := *differential_ring* (2.1.3)

問題を解析し、特異解を求めると、特異解は臨界値 $p=1$ に関連していることが分かります。

```
> ideal := RosenfeldGroebner(ODE, ring);
```

ideal := [*regular_differential_chain*, *regular_differential_chain*] (2.1.4)

両方を考慮すると次のようになります。

```
> Equations(ideal, solved);
```

$$\left[\left[\frac{d}{dx}u(x) = \frac{u(x)^2}{p-1}\right], [u(x)=0, p=1]\right] \tag{2.1.5}$$

p は任意変数とみなし、値に関しては検討しません。

```
> ideal := RosenfeldGroebner(ODE, ring, arbitrary = p);
```

ideal := [*regular_differential_chain*] (2.1.6)

```
> Equations(ideal, solved);
```

$$\left[\left[\frac{d}{dx}u(x) = \frac{u(x)^2}{p-1}\right]\right] \tag{2.1.7}$$

べき級数解

より単純な jet notation を使用して直接入力された熱伝導方程式を検討します (下付き文字は導関数を表しています)。

```
> PDE := [u[t, t]-u[x]];
```

$$PDE := [u_{t,\,t} - u_x] \tag{2.2.1}$$

これは 1 従属変数 u(t, x) についての問題です。微分環を設定します：

```
> ring := DifferentialRing(derivations = [t, x], blocks = [u]);
```

$$ring := differential\_ring \tag{2.2.2}$$

PDE のイデアルを計算します。

```
> ideal := RosenfeldGroebner(PDE, ring);
```

$$ideal := [regular\_differential\_chain] \tag{2.2.3}$$

jet 表記による 2 つの任意関数 u(0, x) = f(x) および D[1](u)(0, x) = g(x) に依存する初期値問題の級数解は次のようになります。

```
> PowerSeriesSolution(ideal, 2, [u = f(x), u[t] = g(x)]);
```

$$\left[\left[u(t,x) = f(0) + \mathrm{D}(f)(0)\,x + g(0)\,t + \frac{1}{2}\,\mathrm{D}^{(2)}(f)(0)\,x^2 + \mathrm{D}(g)(0)\,t\,x + \frac{1}{2}\,\mathrm{D}(f)(0)\,t^2\right]\right] \tag{2.2.4}$$

正規形

次の PDE システムを検討します。

```
> sys := [diff(g(x,y),x) = 2*g(x,y)^2+2, diff(g(x,y),y)^2 =
  -1/4*(-g(x,y)^4-2*g(x,y)^2-1)/y];
```

$$sys := \left[\frac{\partial}{\partial x}\,g(x,y) = 2\,g(x,y)^2 + 2,\ \left(\frac{\partial}{\partial y}\,g(x,y)\right)^2 = -\frac{1}{4}\,\frac{-g(x,y)^4 - 2\,g(x,y)^2 - 1}{y}\right] \tag{2.3.1}$$

これは 1 未知数 $g(x,y)$ に関する問題です。微分環を設定します。

```
> ring := DifferentialAlgebra[DifferentialRing](derivations =
  [x,y], blocks = [g]):
```

特異イデアルを無視し、一般イデアルのみを計算します。

```
> ideal := RosenfeldGroebner(sys, ring, singsol = none);
```

$$ideal := [regular\_differential\_chain] \tag{2.3.2}$$

計算されたイデアルで一般解を表す方程式を確認します。

```
> eqs := Equations(ideal[1], solved);
```

$$eqs := \left[\frac{\partial}{\partial x}\,g(x,y) = 2\,g(x,y)^2 + 2,\ \left(\frac{\partial}{\partial y}\,g(x,y)\right)^2 = -\frac{1}{4}\,\frac{-g(x,y)^4 - 2\,g(x,y)^2 - 1}{y}\right] \tag{2.3.3}$$

解の 1 つは次のようになります (pdetest参照)

```
> sol := tan(2*x-y^(1/2));
```

$$sol := \tan\left(2\,x - \sqrt{y}\right) \tag{2.3.4}$$

```
> pdetest(g(x,y) = sol, eqs);
```

$$[0, 0] \tag{2.3.5}$$

下記の 2 つの式、expr1 および expr2 を検討します。

```
> expr1 := 4*diff(g(x,y), x,y)*y/(g(x,y)^4+2*g(x,y)^2+1) - 4*
  diff(g(x,y), y)*y*(4*g(x,y)^3*diff(g(x,y), x) + 4*g(x,y)*diff
  (g(x,y),x))/(g(x,y)^4+2*g(x,y)^2+1)^2;
```

$$expr1 := \frac{4\left(\frac{\partial^2}{\partial y\,\partial x}\,g(x,y)\right)y}{g(x,y)^4+2\,g(x,y)^2+1} - \frac{4\left(\frac{\partial}{\partial y}\,g(x,y)\right)y\left(4\,g(x,y)^3\left(\frac{\partial}{\partial x}\,g(x,y)\right)+4\,g(x,y)\left(\frac{\partial}{\partial x}\,g(x,y)\right)\right)}{\left(g(x,y)^4+2\,g(x,y)^2+1\right)^2} \tag{2.3.6}$$

```
> expr2 := - diff(g(x, y), x, y)/diff(g(x, y), y)^2;
```

$$expr2 := -\frac{\frac{\partial^2}{\partial y\,\partial x}\,g(x,y)}{\left(\frac{\partial}{\partial y}\,g(x,y)\right)^2} \tag{2.3.7}$$

これらの式は、実際は同値関係にあり、ideal で表示した微分イデアルの法になります。すなわち、g(x, y) = sol で評価した場合、そして一般的には、微分イデアルのどの解で評価してもこれらの 2 つの式は同じになります：

```
> simplify(subs(g(x,y) = sol, expr1 - expr2));
```

$$0 \tag{2.3.8}$$

expr1 と expr2 の間の同値性はこれらの式の正規形を計算し、構文的に同等であることから確認できます。

```
> NF[1] := NormalForm(expr1, ideal);
```

$$NF_1 := \left[-\frac{16\left(\frac{\partial}{\partial y}\,g(x,y)\right)g(x,y)\,y}{g(x,y)^4+2\,g(x,y)^2+1}\right] \tag{2.3.9}$$

```
> NF[2] := NormalForm(expr2, ideal);
```

$$NF_2 := \left[-\frac{16\left(\frac{\partial}{\partial y}\,g(x,y)\right)g(x,y)\,y}{g(x,y)^4+2\,g(x,y)^2+1}\right] \tag{2.3.10}$$

```
> NF[1] - NF[2];
```

$$[0] \tag{2.3.11}$$

DEtools パッケージ

積分因子を使用する DEtools[particularsol]

particularsol は、ODE が持つことができる (元が n 未満の) 積分因子の不完全集合からなる n 階 ODE の特解が計算できるようになりました。これは、ODE を一般的に解くことはできないが、特解でも差し支えない場合に便利です。たとえば、次の 2 階非線形 ODE を検討します。

```
> ode := diff(y(x), x,x) = (-p*y(x)^p*(p- 1)*diff(y(x), x)^2 +
  x*diff(y(x), x)*y(x)^2 + y(x)^3)/(y(x)^(p+1))/p;
```

$$ode := \frac{d^2}{dx^2}\,y(x) = \frac{-p\,y(x)^p\,(p-1)\left(\frac{d}{dx}\,y(x)\right)^2+x\left(\frac{d}{dx}\,y(x)\right)y(x)^2+y(x)^3}{y(x)^{p+1}\,p} \tag{3.1.1}$$

上記 2 階非線形 ODE の一般解は未知です。Maple 13 では、対称式を検討することにより、任意ではない定数に依存した 2 つの特解が計算できました。また、この *ode* の 1 つの積分因子も計算できました。Maple 14 では、これを基に、一連において 3 つ目の解となる 1 つの任意変数 *_C1* に依存する特解が計算できます。

```
> particular_sol := DEtools[particularsol](ode);
```

(3.1.2)

$$particular\_sol := y(x) = 0, y(x) = e^{\frac{\ln\left(\frac{1}{2}\frac{x^2(p-1)}{p}\right)}{p-1}}, y(x) = \frac{\left(\frac{p\,x^2 - x^2 + 2\,\_C1\,p}{p}\right)^{\frac{1}{p-1}}}{2^{\frac{1}{p-1}}} \quad (3.1.2)$$

この新しい特解の利用例を説明するために、初期値でこの *ode* を解くことを意味する初期値問題を検討します。

```
> iv := y(0) = 1, D(y)(0) = 0;
```

$$iv := y(0) = 1, D(y)(0) = 0 \quad (3.1.3)$$

標準的なアプローチでは、まず *ode* の一般解を計算しますが、これは未知であるため、このアプローチは使用できません。そのため、まず particularsol を使用してその 1 つのパラメータ (_C1) を計算し、さまざまな方法で、初期条件に合うようにパラメータの調整を試みます。たとえば、このような目的のために Maple 13 で導入された DEtools[IVPsol] に使用する特解と初期値を直接渡します。ODE と初期値問題全体を解くには、たとえば、次のように指定します。

```
> DEtools[IVPsol]([iv], particular_sol[3]);
```

$$y(x) = \frac{\left(\frac{p\,x^2 - x^2 + 2\,p}{p}\right)^{\frac{1}{p-1}}}{2^{\frac{1}{p-1}}} \quad (3.1.4)$$

または、この特解を dsolve に渡し、この与えられた解から問題を解く方法もあります。

```
> dsolve([ode, iv], solution = particular_sol[3]);
```

$$y(x) = \frac{\left(\frac{p\,x^2 - x^2 + 2\,p}{p}\right)^{\frac{1}{p-1}}}{2^{\frac{1}{p-1}}} \quad (3.1.5)$$

このアプローチは、ユーザーによる指定がなくとも Maple 14 の dsolve によって自動的に試みられることにご留意ください (その場合は、まず一般解が求められ、これが失敗してからの試みとなるため計算には非常に時間がかかります)。

```
> sol := dsolve([ode, iv]);
```

$$sol := y(x) = \frac{\left(\frac{p\,x^2 - x^2 + 2\,p}{p}\right)^{\frac{1}{p-1}}}{2^{\frac{1}{p-1}}} \quad (3.1.6)$$

DEtools[firint] と線形 ODE の第 1 積分の完全集合の計算

DEtools[firint] コマンドは線形 ODE の第 1 積分の完全集合が計算できるようになりました。すなわち、互いに独立した、ODE の階数と同じ数だけの第 1 積分が計算できるようになりました。次は、典型的な例です：

```
> PDEtools[declare](g(x), y(x), prime = x);
```

g(x) will now be displayed as g

y(x) will now be displayed as y

derivatives with respect to x of functions of one variable will now be displayed with ' (3.2.1)

```
> ode := diff(y(x), x,x) = ((diff(g[1](x), x,x))*g[2](x)-g[1](x)
*(diff(g[2](x), x,x)))*(diff(y(x), x))/(g[2](x)*(diff(g[1](x),
x))-g[1](x)*(diff(g[2](x), x)))+((diff(g[1](x), x))*(diff(g[2]
(x), x,x))-(diff(g[1](x), x,x))*(diff(g[2](x), x)))*y(x)/(g[2]
(x)*(diff(g[1](x), x))-g[1](x)*(diff(g[2](x), x)))+((diff(g[0]
(x), x,x))*(-g[2](x)*(diff(g[1](x), x))+g[1](x)*(diff(g[2](x),
x)))+((diff(g[1](x), x))*g[0](x)-g[1](x)*(diff(g[0](x), x)))*
(diff(g[2](x), x,x))-((diff(g[2](x), x))*g[0](x)-g[2](x)*(diff
(g[0](x), x)))*(diff(g[1](x), x,x)))/(-g[2](x)*(diff(g[1](x),
x))+g[1](x)*(diff(g[2](x), x)));
```

$$ode := y'' = \frac{\left(g_1 g_2'' - g_1'' g_2\right) y'}{g_2 g_1'' - g_1 g_2''} + \frac{\left(g_1'' g_2' - g_1' g_2''\right) y}{g_2 g_1'' - g_1 g_2''} + \frac{g_0''\left(-g_2 g_1' + g_1 g_2'\right) + \left(g_1' g_0 - g_1 g_0'\right) g_2'' - \left(g_2' g_0 - g_2 g_0'\right) g_1''}{-g_2 g_1'' + g_1 g_2''} \tag{3.2.2}$$

この *ode* は、斉次部分については解として基底 $g_1(x)$ および $g_2(x)$ を持ち、非斉次方程式全体の特殊解として $g_0(x)$ を持つ最も一般的な 2 階非斉次線形 ODE です。解の基底は以下のように表現されます。

```
> basis_sol := [[g[1](x), g[2](x)], g[0](x)];
```

$$basis\_sol := [[g_1, g_2], g_0] \tag{3.2.3}$$

basis_sol が *ode* の完全基底であるか確認するには、*basis_sol* で一般解を構築し、それを odetest で検証します。

```
> sol := y(x) = _C1*g[1](x) + _C1*g[2](x) + g[0](x);
```

$$sol := y = \_C1\, g_1 + \_C1\, g_2 + g_0 \tag{3.2.4}$$

```
> odetest(sol, ode);
```

$$0 \tag{3.2.5}$$

次は、*ode* の第 1 積分の完全集合を計算します。

```
> first_integrals := DEtools[firint](ode, y(x), basis =
basis_sol);
```

$$first\_integrals := \frac{\left(y - g_0\right) g_2' - g_2\left(y' - g_0'\right)}{-g_2 g_1' + g_1 g_2'}, \frac{\left(-y + g_0\right) g_1' + g_1\left(y' - g_0'\right)}{-g_2 g_1' + g_1 g_2'} \tag{3.2.6}$$

結果を検証します。

```
> map(DEtools[firtest], [first_integrals], ode, y(x));
```

$$[0, 0] \tag{3.2.7}$$

動的システム

- 新規コマンド NyquistPlot が DynamicSystems パッケージに追加されました。このコマンドは線形システムのナイキストプロットを表示します。
- 新規コマンド StepProperties がこのパッケージに追加されました。このコマンドはステップ応答の特性を計算します。
- 2 つの新規コマンド、EquilibriumPoint および Linearize がこのパッケージに追加されました。EquilibriumPoint は非線形システム DAE の局所的平衡点を求めます。Linearize は非線形微分方程式の集合を線形化して状態空間形式に変換することができます。これらのコマンドを使用すると、トリム (trim) 条件を解析したり、トリム (平衡) 点で線形化して得られた線形モデルの状態空表現で作業ができるため、制御システムの設計に有用です。
- 新規コマンド SystemConnect がこのパッケージに追加されました。これを使用する

と、さまざまな DynamicSystems オブジェクトを相互接続することができます。

- 新規オプション adaptive が BodePlot、FrequencyResponse、MagnitudePlot、NyquistPlot および PhasePlot コマンドに追加されました。このオプションを使用すると、適応的に間隔を空ける周波数点を指定してプロットの見栄えを向上することができます。

Groebner

- グレブナ基底の多項式に含まれる単項式を調査する場合は、Groebner[Support] コマンドを使用します。

```
> Groebner[Support]([x^2+23*2+12, y-x/2-1/3]);
```

$$[[1, x^2], [1, x, y]] \tag{5.1}$$

IntegerRelations パッケージ

- IntegerRelations[PSLQ] コマンドは複素数体上で計算できるように拡張されました。これはたとえば、複素代数的数の最小多項式を計算するために使用できます：

```
> alpha := sqrt(2)+(-1)^(1/3);
```

$$\alpha := \sqrt{2} + (-1)^{1/3} \tag{6.1}$$

```
> powers := map(evalf, [seq(expand(alpha^i), i=0..4)]);
```

$$powers := [1., 1.914213562 + 0.8660254037\,I, 2.914213562 + 3.315515146\,I, 2.707106782 + 8.870387035\,I, -2.500000000 + 19.32423841\,I] \tag{6.2}$$

```
> u := IntegerRelations[PSLQ](powers);
```

$$u := [7, 2, -1, -2, 1] \tag{6.3}$$

```
> minpoly := PolynomialTools[FromCoefficientList](u,x);
```

$$minpoly := 7 + 2x - x^2 - 2x^3 + x^4 \tag{6.4}$$

```
> radnormal(eval(minpoly, x=alpha));
```

$$0 \tag{6.5}$$

```
> unassign('u');
```

LinearAlgebra

- LinearAlgebra パッケージのコマンドへの出力オプションの与え方に改良が加えられました。LinearAlgebra パッケージに含まれているコマンドの多くは 1 つ以上の Matrices (行列) または Vectors (ベクトル) を返します。今までは、行列またはベクトルコンストラクタにオプションを渡す方法として、オプション outputoptions=[...] を指定するしかありませんでした。この方法は安全で確実ですが、少々不便でした。このリリースから、これらのコンストラクタに渡すオプションは LinearAlgebra コマンドに直接指定することが可能になりました。次に例を示します：

```
> LinearAlgebra[RandomMatrix](5, datatype=float);
```

$$\begin{bmatrix} 26. & -82. & -20. & 33. & -95. \\ 6. & 12. & -7. & -88. & -65. \\ 85. & 20. & 92. & -15. & -51. \\ 2. & 0. & 91. & -74. & -31. \\ -16. & -9. & -93. & -54. & 43. \end{bmatrix} \tag{7.1}$$

- オプション outputoptions=[...] を使用する今までの方法は、引き続き有効です。特に LinearAlgebra コマンドが 1 つ以上のオブジェクトを返すような場合は、この方法を使用する必要があります。次に例を示します：

```
> LinearAlgebra[LUDecomposition]( <1,2,3;4,5,6;7,8,9>, output=
```

```
['P','L','U'], outputoptions['L']=[datatype=float] );
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1. & 0. & 0. \\ 4. & 1. & 0. \\ 7. & 2. & 1. \end{bmatrix}, \begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -6 \\ 0 & 0 & 0 \end{bmatrix} \tag{7.2}$$

- コンストラクタのオプションがコマンドの呼出手順 (直接指定) と outputoptions サブオプションの両方に指定されている場合、指定順に関わらず後者が優先されます。これにより、すべての出力に適用するオプションを指定しておき、特定の出力に対してのみ別のコンストラクタオプションを指定することが可能になります。

```
> LinearAlgebra[LUDecomposition]( <1,2,3;4,5,6;7,8,9>, output=
['P','L','U'], datatype=float, outputoptions['L']=[datatype=
integer] );
```

$$\begin{bmatrix} 1. & 0. & 0. \\ 0. & 1. & 0. \\ 0. & 0. & 1. \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 7 & 2 & 1 \end{bmatrix}, \begin{bmatrix} 1. & 2. & 3. \\ 0. & -3. & -6. \\ 0. & 0. & 0. \end{bmatrix} \tag{7.3}$$

- 新規コマンド CARE と DARE が LinearAlgebra パッケージに追加されました。これらのコマンドはそれぞれ、連続時間代数リッカチ方程式 (Continuous Algebraic Ricatti Equation) および離散時間代数リッカチ方程式 (Discrete Algebraic Ricatti Equation) と呼ばれる実数型の陰的線形システムの数値解を求めるために使用できます。これらの方程式は最適レギュレータ (LQR; Linear-Quadratic Regulator) の計算など、最適制御の分野でよく使用される方程式です。

PDEtools パッケージ

- PDEtools には 2 つの新しいコマンドが追加されました。
  - Solve；このコマンドの目的は 2 つです：1 つ目の目的はsolve、dsolve および pdsolve の機能を 1 つのコマンドで提供し、ユーザーの入力から実際に呼び出すコマンドを判断することで、2 つ目の目的は非多項式型の代数方程式システムと微分方程式システムおよび、任意数の identity 変数向けとして solve[identity] で提供されている機能を一般化し、指定されている変数に依存しない解を計算する新規の関連機能を提供することです。
  - InvariantEquation；与えられた対称群を持つことができる不変式を計算します。
- PDE システムの操作と PDEtools ライブラリのコマンドを使用したプログラム作成に役立つ 5 つの新しい PDEtools:-Library コマンドが追加されました。これらは次のとおりです：JetNumbersToJetVariables、JetVariablesToJetNumbers、NormalizeBoundaryConditions、SolveCas

アプリケーションと例題

```
> with(PDEtools):
```

次は、微分方程式ではない式の例です。

```
> eq[1] := a*x^2 + b*x + c;
```

$$eq_1 := a\,x^2 + b\,x + c \tag{8.1}$$

```
> Solve(eq[1], x);
```

$$x = \frac{1}{2}\,\frac{-b + \sqrt{b^2 - 4\,a\,c}}{a}, x = -\frac{1}{2}\,\frac{b + \sqrt{b^2 - 4\,a\,c}}{a} \tag{8.2}$$

次は、ODE 問題とその級数解の例です。

```
> eq[2] := diff(y(x),x) = y(x);
```

$$eq_2 := y' = y \tag{8.3}$$

```
> Solve(eq[2], series);
```

$$y = y(0) + y(0)\,x + \frac{1}{2}\,y(0)\,x^2 + \frac{1}{6}\,y(0)\,x^3 + \frac{1}{24}\,y(0)\,x^4 + \frac{1}{120}\,y(0)\,x^5 + O(x^6) \tag{8.4}$$

次は、境界条件付きの PDE 問題の例です。

```
> eq[3] := [diff(u(x, t), t)+c*(diff(u(x, t), x)) = -lambda*u(x,
  t), u(x, 0) = phi(x)];
```

$$eq_3 := \left[u_t + c\,u_x = -\lambda\,u(x,t),\ u(x,0) = \phi(x)\right] \tag{8.5}$$

```
> Solve(eq[3]);
```

$$u(x,t) = \phi(x - t\,c)\,e^{-\lambda t} \tag{8.6}$$

上記例は solve、dsolve または pdsolve を 1 つのコマンドで呼び出し、入力および出力に対して統一された形式を持つ利点を示しています。Solve は追加機能も提供します：与えられた変数と独立した解を計算できます。independentof が指定されていると、不等式が含まれていてもシステムを解くことができます。

```
> eq[4] := [k*a*c*(a+b)*exp(k*d*t)-2*a*exp(k*t)*k+Q*(-c+a)*x, a <>
  0];
```

$$eq_4 := \left[k\,a\,c\,(a+b)\,e^{k\,d\,t} - 2\,a\,e^{k\,t}\,k + Q\,(-c+a)\,x,\ a \neq 0\right] \tag{8.7}$$

```
> Solve(eq[4], {a, b, c, d}, independentof = {t, x});
```

$$\left\{a = a,\ b = -\frac{a^2-2}{a},\ c = a,\ d = 1\right\} \tag{8.8}$$

微分方程式または微分方程式システムについても与えられた変数と独立した解を計算できます。

```
> eq[5] := diff(f(x,y),x)*diff(g(x,y),x) + diff(f(x,y),y)*diff(g
  (x,y),y) + g(x,y)*(diff(f(x,y), x,x) + diff(f(x,y), y,y)) = -1;
```

$$eq_5 := f_x\,g_x + f_y\,g_y + g(x,y)\,\left(f_{x,x} + f_{y,y}\right) = -1 \tag{8.9}$$

次は、x にも、y にも依存しない PDE の解の例です。

```
> Solve(eq[5], independentof = x);
```

$$\left\{f(x,y) = \_F1(y),\ g(x,y) = \frac{-y + \_C1}{\_F1_y}\right\} \tag{8.10}$$

```
> Solve(eq[5], independentof = y);
```

$$\left\{f(x,y) = \_F1(x),\ g(x,y) = \frac{-x + \_C1}{\_F1'}\right\} \tag{8.11}$$

- 新しい InvariantEquations コマンドは与えられた対称式内の不変式を計算します。以下に示す 1 次対称群の無限小のリストについて検討します。

```
> S := [x, 1, u];
```

$$S := [x, 1, u] \tag{8.12}$$

これらの無限小の対称変換の基礎となっている 1 階偏微分の不変式は次のようになります。

```
> PDE := InvariantEquation(S, u(x, t), name = Lambda);
```

```
* Partial match of 'name' against keyword 'arbitraryfunctionname'
```

$$PDE := \Lambda\left(-\ln(x) + t,\ \frac{u(x,t)}{x},\ u_x,\ \frac{u_t}{x}\right) \tag{8.13}$$

S における 3 階 PDE の不変式は次のようになります。

```
> PDE := InvariantEquation(S, u(x, t), order = 3);
```

$$PDE := \_F1\left(-\ln(x) + t,\ \frac{u(x,t)}{x},\ u_x,\ \frac{u_t}{x},\ u_{t,x},\ u_{x,x}\,x,\ \frac{u_{t,t}}{x},\ u_{t,t,x},\ u_{x,x,x}\,x^2,\ u_{t,x,x}\,x,\right. \tag{8.14}$$

$$\left(\frac{u_{t,t,t}}{x}\right)$$

上記入力例のように、arbitraryfunctionname が示されていない場合は _Fn (n は整数) の形式の名前が使用されます。この PDE の不変性はいくつかの方法で検証できます。最も単純な方法は SymmetryTest を使用して S と PDE が対称であると検証する方法だと思われます。

```
> SymmetryTest(S, PDE);
```

$$\{0\} \tag{8.15}$$

InvariantEquations を使用して n 次対称群による式の不変量を計算することもできます。たとえば、次の、2 つの従属変数 $u(x,t), v(x,t)$ を含む問題の 5 つの対称式のリストを検討します。

```
> G := [[_xi[x] = 0, _xi[t] = 1, _eta[u] = 0, _eta[v] = 0], [_xi
  [x] = 0, _xi[t] = 0, _eta[u] = 1, _eta[v] = 0], [_xi[x] = 1, _xi
  [t] = 0, _eta[u] = 2*t, _eta[v] = 0], [_xi[x] = t, _xi[t] = 0,
  _eta[u] = t^2+x, _eta[v] = 0], [_xi[x] = 1/2*x+3/2*t^2, _xi[t] =
  t, _eta[u] = t^3+3*t*x, _eta[v] = -v]];
```

$$G := \left[[\_\xi_x = 0, \_\xi_t = 1, \_\eta_u = 0, \_\eta_v = 0], [\_\xi_x = 0, \_\xi_t = 0, \_\eta_u = 1, \_\eta_v = 0], [\_\xi_x = 1, \_\xi_t = 0, \_\eta_u = 2\,t, \_\eta_v = 0], \left[\_\xi_x = t, \_\xi_t = 0, \_\eta_u = t^2 + x, \_\eta_v = 0\right], \left[\_\xi_x = \frac{1}{2}\,x + \frac{3}{2}\,t^2, \_\xi_t = t, \_\eta_u = t^3 + 3\,t\,x, \_\eta_v = -v\right]\right] \tag{8.16}$$

関連の 1 階不変 PDE (1st order invariant PDE) 族は次のようになります：

```
> PDE := InvariantEquation(G, [u, v](x, t), arbitraryfunctionname
  = Lambda);
```

$$PDE := \Lambda\left(\frac{v_x}{v(x,t)^{3/2}}, \frac{-4\,x + u_x^2 + 2\,u_t}{v(x,t)}, \frac{u_x\,v_x + v_t}{v(x,t)^2}\right) \tag{8.17}$$

```
> map(SymmetryTest, G, PDE, [u, v](x, t));
```

$$[\{0\}, \{0\}, \{0\}, \{0\}, \{0\}] \tag{8.18}$$

RegularChains

RegularChains パッケージには、Maple 14 で以下の新規コマンドが追加されました：

- Intersect：増分法を使用して多項式システムを解きます。
- RealTriangularize：多項式システムの実数根を 3 角分解します。
- CylindricalAlgebraicDecompose：多項式型の不変集合 F の円筒代数分解を計算します。
- SubresultantChain：1 変数多項式とされる 2 つの多項式の部分終結式チェインを暗号化するデータ構造を作成します。値による表現、単項式を基底とした表現、ベズー (Bezout) 行列を用いた表現など、さまざまな部分終結式チェインの表現がサポートされています。

また、既存のコマンドに関しては、以下の改良がなされました：

- RegularGcd：regular chain を法とする GCD (最大公約数) をより高速に計算する新規の呼出手順が提供されています。
- ComprehensiveTriangularize：可読な出力を表示する新規オプション piecewise が提供されています。
- RealRootIsolate：代替アルゴリズムとして Discoverer が提供されています。有限の複素解を持つ半代数的システムの実数根も分離します。
- Triangularize：新しく、より高速なアルゴリズムを実装しています。このアルゴリズムは特に有限個の解を持つシステムに対して有効です。

- 新しく、より高速なアルゴリズムは RegularGcd および Regularize に対しても実装されています。

RootFinding

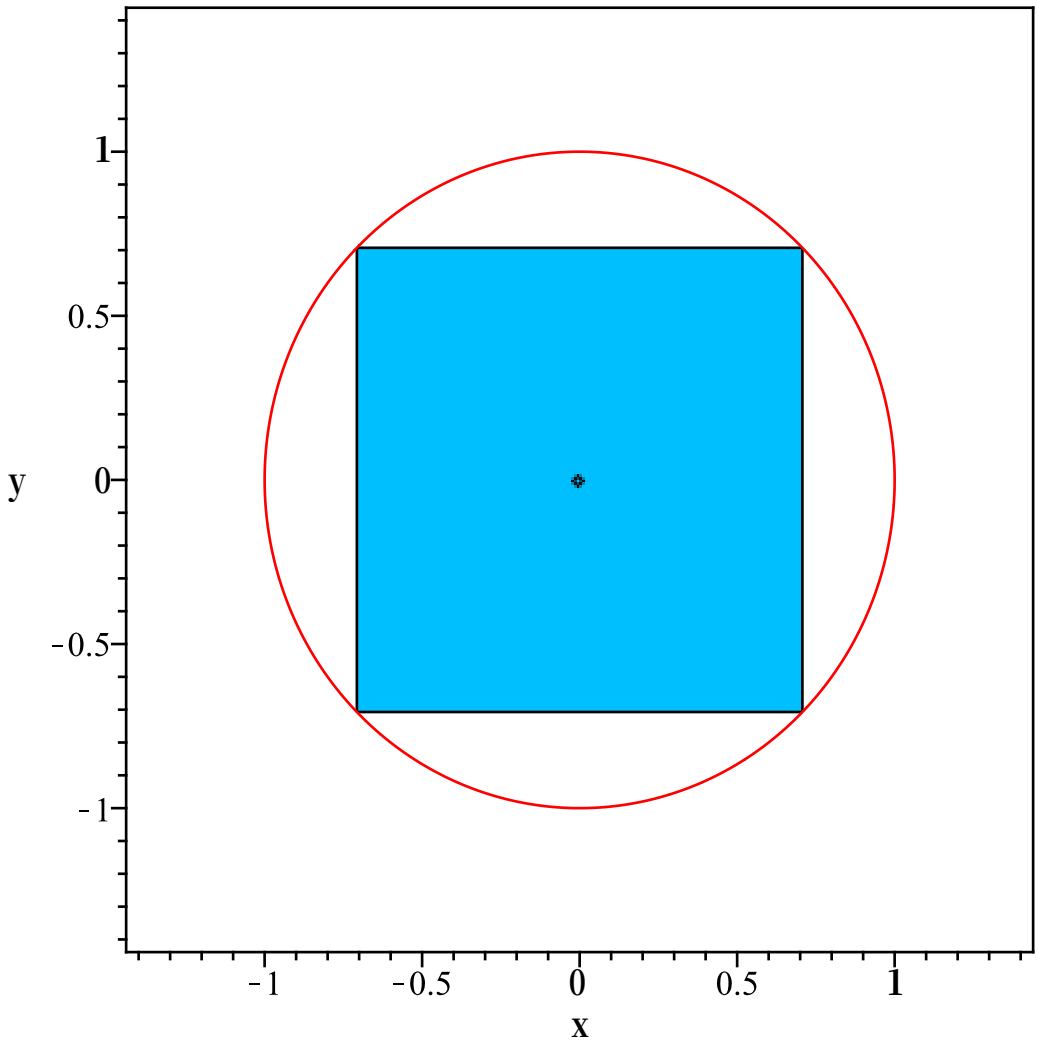
- RootFinding パッケージには 3 つの新規コマンドが追加されました：
  - 与えられた実数型の多様体上に位置しない点に対して、EnclosingBox はその点を囲み、多様体と交差しない矩形を計算します。
  - HasRealRoots コマンドは与えられた多項式の集合が実数体上で解けるかどうかを調べます。
  - WitnessPoints コマンドは実数多様体または実数多様体の補集合に接続されている各コンポーネント上の点を求めることができます。
- アプリケーションと例題

次は、原点を含む単位円の中に納まる矩形を計算し、表示する例です：

```
> with(RootFinding):
> circle := x^2+y^2-1;
```

$$circle := x^2 + y^2 - 1 \tag{10.1}$$

```
> EnclosingBox(circle, [x=0, y=0], 'output'='plot');
```



次は、円が双曲線と交差するかを調査する例です：

```
> HasRealRoots([circle, x*y-1]);
```

$$false \tag{10.2}$$

```
> HasRealRoots([circle, 2*x*y-1]);
```

$$true \tag{10.3}$$

```
> Isolate([circle, 2*x*y-1], [x,y]);
```

$$[[x = -0.7071067812, y = -0.7071067812], [x = 0.7071067812, y = 0.7071067812]] \tag{10.4}$$

```
> HasRealRoots([circle, 4*x*y-1]);
```

$$true \tag{10.5}$$

```
> Isolate([circle, 4*x*y-1], [x,y]);
```

$$[[x = -0.2588190451, y = -0.9659258263], [x = -0.9659258263, y = -0.2588190451], [x = 0.9659258263, y = 0.2588190451], [x = 0.2588190451, y = 0.9659258263]] \tag{10.6}$$

- RootFinding[Parametric] サブパッケージが改良されました：
  - DiscriminantVariety および CellDecomposition コマンドは変数より多くの方程式からなるシステムを処理することができるようになりました。
  - CellDecomposition コマンドは $g \neq 0$) 形式の不等式も受け付けるように拡張されました。また、新規オプション output が追加されました。

StringTools

- StringTools パッケージには新規コマンド IsEodermdrome が追加されました。
- StringTools パッケージにはまた、データ圧縮関連の新規コマンドが 2 つ (Compress および Uncompress) が追加されました。これらのコマンドは文字列やバイト配列を含むデータの集合に対して zlib 圧縮または解凍を実行します。

Threads[Task]

- Task Programming Model には新しい関数が追加されました。タスク内で Return 関数を使用すると、タスクモデルの実行を中断し、Start から特定の値を返すことができます。

関連項目

Index of New Maple 14 Features, Updates to Differential Equation (DE) Solvers in Maple 14