Maple 15 の計算アルゴリズムの改良

はじめに

Maple 15 は、記号計算と数値計算の両方について、新規のまたは改良された計算アルゴリズムを含みます。

最適化

- Optimization[LPSolve] に新しい反復内点法が追加されました。この計算法は、ウォータールー大学の H. Wolkowicz 博士等の開発によるものです。この方法は、大型の疎線形プログラム問題において、従来の有効制約法に比べ効率が大幅に改善されています。

多項式の演算

- インプリメンテーションの改善により、高次かつ密な多項式の乗算、除算、累乗で処理速度が 4 倍以上になっています。メモリ使用は Maple 14 よりも 3/4 以上少なくなっています。
- 以下、効率のよい乗算、累乗、除算、法演算の例を示します。

```
> f,g := seq(randpoly(x,degree=10^4,dense),i=1..2):
> p := CodeTools[Usage](expand(f*g)):
memory used=1.70MiB, alloc change=1.87MiB, cpu time=68.00ms, real time=
69.00ms
> p := CodeTools[Usage](expand((5*x-3*y)^10000)):
memory used=34.72MiB, alloc change=34.62MiB, cpu time=251.00ms, real
time=251.00ms
> n := prevprime(2^512):
> f := Expand((1+x+y+z+t)^30) mod n:
> CodeTools[Usage](Divide(f,1+x+y+z+t,'q') mod n);
memory used=4.79MiB, alloc change=4.75MiB, cpu time=169.00ms, real time=
169.00ms
```

$$true \tag{3.1}$$

- divide は、二番目のコールで見られるように、多項式が割り切れない場合すぐに判定します。

```
> f,g := seq(randpoly([x,y,z],degree=30,terms=3000),i=1..2):
> p := expand(f*g):
> CodeTools[Usage](divide(p,f,'q'));    # computes quotient
memory used=227.52KiB, alloc change=0 bytes, cpu time=901.00ms, real
```

time=901.00ms

true (3.2)

```
> CodeTools[Usage](divide(p+1,f,'q'));  # fails instantly
```

memory used=0.61MiB, alloc change=0 bytes, cpu time=83.00ms, real time=83.00ms

false (3.3)

- 名前 f、g、n、p をアサイン取り消し。

```
> f:='f': g:='g': n:='n': p:='p':
```

疎行列

- 疎行列の様々な演算のサポートが大幅に改善されました。ハードウェアの浮動小数点乗算、転置、ブロック・コピー、結合、部分行列の抽出が、ネーティブサポートにより瞬時に計算できるようになりました。

```
> M := LinearAlgebra[RandomMatrix](1000,storage=rectangular,density=.003,datatype=float):
> time(Matrix(M,storage=sparse));
 # 今： .012s、Maple 14では .200s
> M1 := LinearAlgebra[RandomMatrix](1000,storage=sparse,density=.003,datatype=float):
> M2 := LinearAlgebra[RandomMatrix](1000,storage=sparse,density=.003,datatype=float):
> time(M1.M2);
 # 今： .004s、Maple 14 では .032s
> time(Matrix(M1,transpose=true));
 # 今： .000 前： 5.104
> time(Matrix(M1,storage=rectangular));
 # 今： .008 前： 0.052
> time(<M1|M2>);
 # 今： .000 前： 10.096
> time(<M1,M2>);
 # 今： .000 前： 10.148
> MD := LinearAlgebra[RandomMatrix](1000,shape=diagonal,datatype=float):
> time(MD.M1);
 # 今： .000 前： 204
> time(M1[..,1..500]);
 # 今： .000 前： 2.528
> V1 := Vector([seq(2*i,i=1..500)]);
```

(4.1)

$$V1 := \begin{bmatrix} 1 \,..\, 500\ Vector_{column} \\ Data\ Type: anything \\ Storage: rectangular \\ Order: Fortran\_order \end{bmatrix} \tag{4.1}$$

```
> time(M1(..,V1));
```

今： .060 前： 2.624

高精度代数リカッチ方程式

- 線形代数 パッケージの高精度代数リカッチ方程式解法
- 連続・離散の代数リカッチ方程式を解くための CARE と DARE の各コマンドは、ハードウェア倍精度以上で動作するように強化されました。

```
> with(LinearAlgebra):
> A := Matrix([[1.0, .2, 1.0], [0., .1, .45], [2.0, 2.3, 1.3]]);
```

$$A := \begin{bmatrix} 1.0 & 0.2 & 1.0 \\ 0. & 0.1 & 0.45 \\ 2.0 & 2.3 & 1.3 \end{bmatrix} \tag{5.1}$$

```
> B := Matrix([[0, 1], [0, 0], [1, 0]]);
```

$$B := \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \tag{5.2}$$

```
> Q := Matrix([[1, 0, 0], [0, 1, 0], [0, 0, 1]]);
```

$$Q := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

```
> R := Matrix([[1.0, 0.], [0., 1.0]]);
```

$$R := \begin{bmatrix} 1.0 & 0. \\ 0. & 1.0 \end{bmatrix} \tag{5.4}$$

```
> X := CARE(A, B, Q, R);
```

$$X := \begin{bmatrix} 3.41350912431039 & 1.85810170520121 & 2.41829858578592 \\ 1.85810170520121 & 9.76324094274857 & 4.65230672399637 \\ 2.41829858578592 & 4.65230672399637 & 3.72188050348369 \end{bmatrix} \tag{5.5}$$

```
> Digits:=20:
> X := CARE(A, B, Q, R);
```

(5.6)

$$X := \begin{bmatrix} 3.4135091243103892062 & 1.8581017052012091792 & 2.4182985857859171196 \\ 1.8581017052012091792 & 9.7632409427485836328 & 4.6523067239963752516 \\ 2.4182985857859171196 & 4.6523067239963752516 & 3.7218805034836886482 \end{bmatrix} \quad (5.6)$$

パラメトリック解

場合分けのある多項方程式解

- 多項方程式解の場合分けを作成するためのユーザーレベルにおける新機能として、新コマンド SolveTools[Parametric] が加わり、solve コマンドに parametric オプションが追加されました。

```
> SolveTools[Parametric]({a*x}, {x}, {a});
```

$$\begin{cases} [\{x = x\}] & a = 0 \\ [\{x = 0\}] & a \neq 0 \end{cases} \quad (6.1.1)$$

- デフォルトとして、これらの新しいコマンドは区分的な解を戻しますが、それは最も一般的な場合の解と、不活性な 関数コールを含むその他の場合の解を含みます。

```
> result := SolveTools[Parametric]({a*x^2-(b+a)*x+b}, {x});
```

$$result := \begin{cases} \%SolveTools_{Parametric}(\{-x\,b + b\}, \{x\}, \{b\}) & a = 0 \\ \left[\{x = 1\}, \left\{x = \frac{b}{a}\right\}\right] & a \neq 0 \end{cases} \quad (6.1.2)$$

```
> result := eval(result, a=0);
```

$$result := \%SolveTools_{Parametric}(\{-x\,b + b\}, \{x\}, \{b\}) \quad (6.1.3)$$

```
> value(result);
```

$$\begin{cases} [\{x = x\}] & b = 0 \\ [\{x = 1\}] & b \neq 0 \end{cases} \quad (6.1.4)$$

- 新しい解法オプションは、入力に新たな処理を加えますが、SolveTools[Parametric] と同じバックエンドを用いており、得られる解も同じです。詳細は 解/パラメトリック のヘルプページをご覧ください。

```
> solve( a*x^2-(b+a)*x+b, x, 'parametric');
```

$$\begin{cases} \%SolveTools_{Parametric}(\{-x\,b + b\}, \{x\}, \{b\}) & a = 0 \\ \left[\{x = 1\}, \left\{x = \frac{b}{a}\right\}\right] & a \neq 0 \end{cases} \quad (6.1.5)$$

定和分

- sum コマンドは、パラメトリックな境界を持つ定和分が扱えるように改良されました。任意キーワード parametric で動作を制御できます。

```
> sum(1/k, k=a..b);
Warning, unable to determine if the summand is singular in the
interval of summation; try to use assumptions or use the parametric
option
```

$$\sum_{k=a}^{b} \frac{1}{k} \tag{6.2.1}$$

```
> sum(1/k, k=a..b, parametric);
```

$$\begin{cases} 0 & a = b+1 \\ \Psi(b+1) - \Psi(a) & 1 \le a \text{ and } 0 \le b \\ \Psi(-b) - \Psi(1-a) & a \le 0 \text{ and } b \le -1 \\ FAIL & otherwise \end{cases} \tag{6.2.2}$$

- sum コマンドの他に、定和分コマンド SumTools[DefiniteSum][Definite] に新オプション parametric が追加されました。

```
> with(SumTools):
> DefiniteSum[Definite](binomial(2*k-3, k)/4^k, k=0..n,
  parametric);
```

$$\begin{cases} 0 & n = -1 \\ \dfrac{2\,(n+1)\,(n+2)\,\mathrm{binomial}(2n-1, n+1)}{(n-1)\,4^{n+1}} & n \le 0 \\ \dfrac{3}{4} & n = 1 \\ \dfrac{2\,(n+1)\,(n+2)\,\mathrm{binomial}(2n-1, n+1)}{(n-1)\,4^{n+1}} + \dfrac{3}{8} & 2 \le n \end{cases} \tag{6.2.3}$$

- ノンパラメトリックな定和分について、除去可能な特異点の扱いが改良されました。デフォルトの挙動としてそのような特異点は除去されますが、_EnvFormal を false に設定することで除去しないようにできます。

```
> sum(1/GAMMA(k), k=-1..5);
```

$$\frac{65}{24} \tag{6.2.4}$$

```
> _EnvFormal := false;
```

$$\_EnvFormal := false \tag{6.2.5}$$

```
> sum(1/GAMMA(k), k=-1..5);
```

エラー　(SumTools:-DefiniteSum:-ClosedForm)　和分範囲で被和数が特異となります

SumTools のその他の改良

- SumTools パッケージとその下位パッケージに BottomSequence と SummableSpace という二つの新しいコマンドが追加されました。
- SumTools[IndefiniteSum][Indefinite]、SumTools[IndefiniteSum][Hypergeometric]、SumTools[IndefiniteSum][Rational] の各無限和コマンドに新しいオプション failpoints が加わりました。
- SumTools[Hypergeometric][Gosper] コマンドに、Gosper のアルゴリズムにおける有理認証関数を返すための新しい任意引数が加わりました。

```
> with(SumTools[Hypergeometric]):
```

```
> Gosper((-1)^k*binomial(n,k), k, 'r');
  r ;
```

$$-\frac{k\,(-1)^{k}\,\mathrm{binomial}(n,k)}{n}$$

$$-\frac{k}{n} \tag{6.3.1}$$

- SumTools[Hypergeometric][DefiniteSumAsymptotic] コマンドは改良され、2 変数有理関数が扱えるようになりました。

```
> DefiniteSum(1/(m^2+k^2)^2, m, k, 0..m);
```

$$FAIL \tag{6.3.2}$$

```
> DefiniteSumAsymptotic(1/(m^2+k^2)^2, m, k, 0..m);
```

$$\frac{1}{8}\,\frac{2+\pi}{m^{3}}+\frac{5}{8\,m^{4}}-\frac{1}{24\,m^{5}}+\mathrm{O}\left(\frac{1}{m^{7}}\right) \tag{6.3.3}$$

- SumTools[Hypergeometric][SumDecomposition] コマンドにおけるキーワードオプション minimize による最小化方法が増えました。

RegularChains

- Maple 15 の RegularChains ライブラリは、有理方程式系の実数解を計算するための様々なツールを提供します。三つの新しいコマンド RealTriangularize、LazyRealTriangularize、SamplePoints は、任意の半代数方程式系を扱うものです。つまり、有理方程式・有理不同式・有理不等式（等号を含むものも含まないものも可）からなる任意の系 S について、その実数解に関する情報を出力します。
- RegularChains[RealTriangularize] は、S 実数解の完全な表現を返します。つまり、幾つかのより簡単な系を返し、ある点がそのうちのひとつの系の解であることが S の解であるための必要十分条件となります。その簡単な系はそれぞれ三角型であり、優れた性質があるため「正則半代数方程式」と呼ばれ、その簡単な系の集合は S の完全三角分解と呼ばれます。
- RegularChains[LazyRealTriangularize] を使うと、ユーザーは S の三角分解を対話的に計算することができます。この機能は、解法が難しい系に適しています。そのような系に対し、LazyRealTriangularize は S の最大次元成分と、その他の成分（一般的に計算がより困難）を生成する漸化式を未計算の状態で返します。
- RegularChains[SamplePoints] は、さらに怠惰（従って経済的な）な解法です。S の解集合の上の連結成分につき少なくとも 1 つのサンプル点を生成します。多くの場合、実用的にはこの方法で十分です。
- RegularChains[Display] は、RegularChains ライブラリのあらゆるオブジェクトタイプに対応した pretty print 用のコマンドです。
- RegularChains[ChainTools][Extend] は、三角形式のセットを通常のチェーンに分解する新しいコマンドです。
- RegularChains[ParametricSystemTools][RealRootClassification] には、formula と samples という二つの新しい output オプションが加わりました。これらのオプションを使うと、計算された集合の形式記述またはサンプル点を得ることができます。

```
> with(RegularChains):
> R := PolynomialRing([z,y,x]):
```

- Zeck 代数曲面は次の方程式で定義されます。不等式による制約は曲面の一部を選択するために加えてあります。

```
> F := [x^2+y^2-z^3*(1-z) = 0, 5*y> 1, z<=2];
```

$$F := \left[x^2 + y^2 - z^3\,(1 - z) = 0,\ 1 < 5\,y,\ z \le 2\right] \tag{6.4.1}$$

• 下記は、実数解を公式により完全に記述したものです。

```
> RealTriangularize(F, R,output=record);
```

$$\left\{\begin{array}{c} z^4 - z^3 + y^2 + x^2 = 0 \\ 2 - z > 0 \\ 5\,y - 1 > 0 \\ 256\,y^2 < 27 \\ 256\,y^2 + 256\,x^2 < 27 \end{array}\right\},\ \left\{\begin{array}{c} 4\,z - 3 = 0 \\ 256\,y^2 - 27 = 0 \\ 5\,y - 1 > 0 \\ x = 0 \end{array}\right\},\ \left\{\begin{array}{c} 4\,z - 3 = 0 \\ 256\,y^2 + 256\,x^2 - 27 = 0 \\ 5\,y - 1 > 0 \\ 6400\,x^2 < 419 \end{array}\right\} \tag{6.4.2}$$

• 次に、ウィットネス解を示します。

```
> SamplePoints(F, R,output=record);
```

$$\left\{\begin{array}{c} z = \left[\frac{17}{32}, \frac{9}{16}\right] \\ y = \frac{17}{64} \\ x = 0 \end{array}\right\},\ \left\{\begin{array}{c} z = \left[\frac{57}{64}, \frac{29}{32}\right] \\ y = \frac{17}{64} \\ x = 0 \end{array}\right\},\ \left\{\begin{array}{c} z = \frac{3}{4} \\ y = \left[\frac{41}{128}, \frac{21}{64}\right] \\ x = 0 \end{array}\right\} \tag{6.4.3}$$

• 次の方程式で雪の結晶が表せます。

```
> F2 := [x^3+y^2*z^3-y*z^4 ];
```

$$F2 := \left[x^3 + y^2 z^3 - y\,z^4\right] \tag{6.4.4}$$

• 下記は、実数解の完全な記述です。

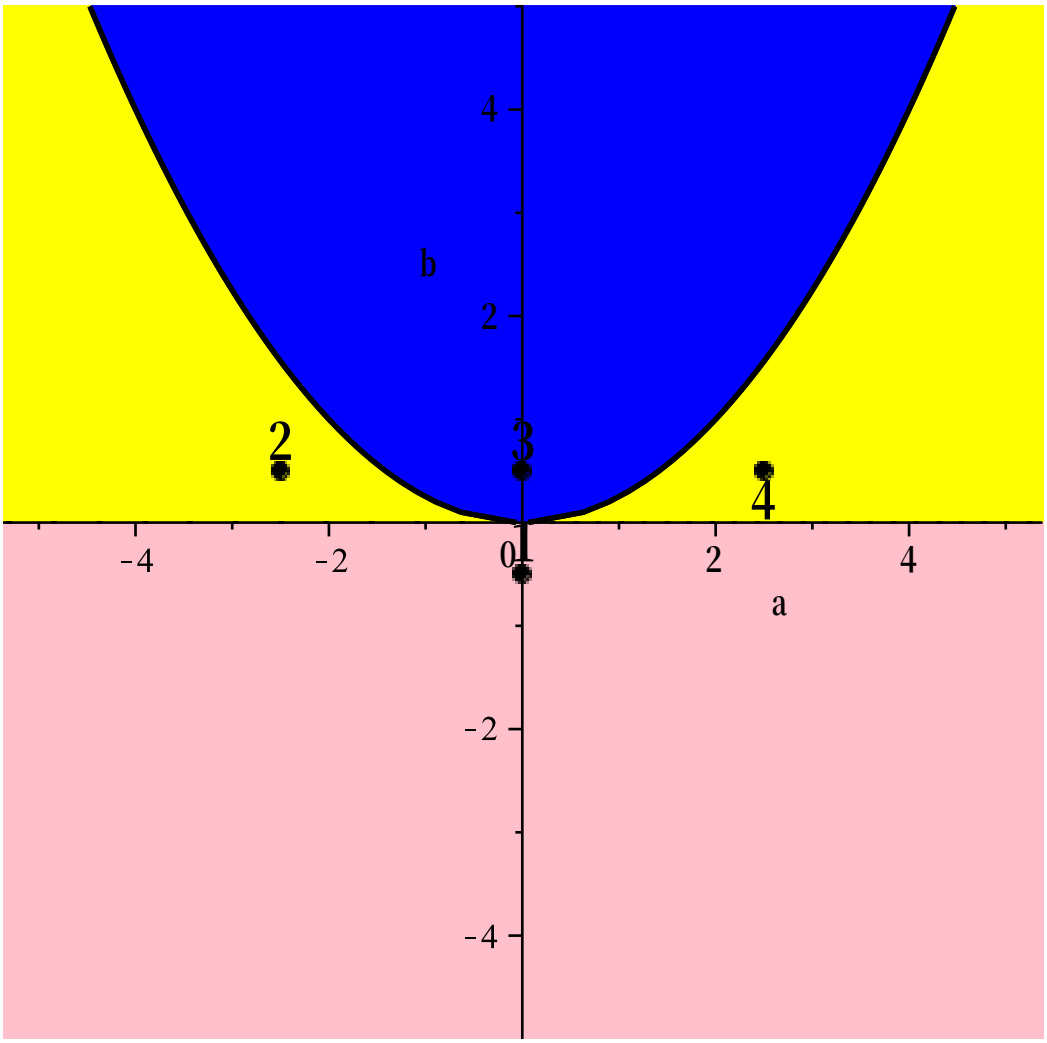
```
> RealTriangularize(F2, R, output=record);
```

$$\left\{\begin{array}{c} y\,z^4 - y^2 z^3 - x^3 = 0 \\ y > 0 \\ 27\,y^5 + 256\,x^3 > 0 \\ x \neq 0 \end{array}\right\},\ \left\{\begin{array}{c} y\,z^4 - y^2 z^3 - x^3 = 0 \\ y < 0 \\ 27\,y^5 + 256\,x^3 < 0 \\ x \neq 0 \end{array}\right\},\ \left\{\begin{array}{c} z = 0 \\ x = 0 \end{array}\right\},\ \left\{\begin{array}{c} z - y = 0 \\ x = 0 \end{array}\right\},\ \left\{\begin{array}{c} y = 0 \\ x = 0 \end{array}\right\},\ \left\{\begin{array}{c} 64\,z^3 - 16\,y\,z^2 - 12\,y^2 z - 9\,y^3 = 0 \\ 27\,y^5 + 256\,x^3 = 0 \\ x \neq 0 \end{array}\right\},\ \left\{\begin{array}{c} z = 0 \\ y = 0 \\ x = 0 \end{array}\right\} \tag{6.4.5}$$

RootFinding[Parametric]

• 新しいオプション method を利用すれば、RootFinding[Parametric][CellDecomposition] から RegularChains ライブラリを利用することができます。

```
> with(RootFinding[Parametric]):
> m := CellDecomposition([x^2+a*x+b = 0], [x], 'method' = 'RC'):
> CellPlot(m, 'samplepoints');
```




```
> NumberOfSolutions(m);
```

$$[[1, 2], [2, 2], [3, 0], [4, 2]] \tag{6.5.1}$$

```
> unassign('m'):
```

統計

- 統計 パッケージに、新コマンド AutoCorrelation と CrossCorrelation が加わりました。これらのコマンドを用いると、離散時系列の自己相関と二つの離散時系列の相互相関の近似値を効率良く計算できます。

```
> Statistics[CrossCorrelation](<1,2,1,2>, <2,1,2,1>, 2);
```

$$\begin{bmatrix} 0.499999999966518 \\ 1.12500000000563 \\ 1. \\ 0.749999999982922 \\ 0.500000000010712 \end{bmatrix} \tag{7.1}$$

```
> Statistics[AutoCorrelation](<1,-1,1,-1>, 2);
```

(7.2)

$$\begin{bmatrix} 1. \\ -0.749999999942256 \\ 0.499999999875000 \end{bmatrix} \tag{7.2}$$

- Statistics[Sample] コマンドは、既存の rtable に結果を保存することができるようになり、従来のように毎回新しい物を作成する必要がなくなりました。これで、メモリーを効率良く利用できます。

単位

- 単位 パッケージに UseUnit コマンドが加わりました。これは、パッケージ内の計算で使用する単位を選択するのに使えます。このコマンドは、AddSystem と UseSystem の各コマンドよりも簡単で、代わりに使うことができます。

参照

Maple 15 新機能索引, Maple 15 の微分方程式（DE）ソルバの改良, Maple 15 の微分方程式パッケージの改良