基本対称式の作成と
グレブナ基底を用いた対称式の基本対称式による計算例

```
> restart;
```

基本対称式の生成

次で定義する elemsympol 関数は、変数のリスト vars を与え、基本対称式を生成します。基本対称式の左辺側変数のシンボルを2番目の引数で指定します。

```
> elemsympol := proc(vars::list, s::symbol)
    local p,k,eqns;

    eqns :=
    seq(`+`(
     {map(p->`*`(p[]),combinat[permute](vars,k))[]}[]),
     k=1..nops(vars));

    seq(s[k]=eqns[k], k=1..nops(vars));
  end proc;
```

$$elemsympol := \mathbf{proc}(vars::list, s::symbol) \tag{1.1}$$

local *p*, *k*, *eqns*;

$eqns := seq(`+`(\{map(p \rightarrow `*`(p[]), combinat[permute](vars, k))[]\}[]), k = 1 ..nops(vars));$

$seq(s[k] = eqns[k], k = 1 ..nops(vars))$

end proc

使用例：
まず、変数のリストを用意します。

```
> v := [x[1],x[2]];
```

$$v := [x_1, x_2] \tag{1.2}$$

割り当てた変数リストを使って基本対称式を生成します。左辺側の変数は 2 番目の引数で指定します。

```
> elemsympol(v,s);
```

$$s_1 = x_1 + x_2,\ s_2 = x_1 x_2 \tag{1.3}$$

より高次な場合も同様の使用方法で基本対称式を生成します。

```
> elemsympol([x[1],x[2],x[3],x[4]], sigma);
```

$$\sigma_1 = x_1 + x_2 + x_3 + x_4,\ \sigma_2 = x_1 x_2 + x_1 x_3 + x_1 x_4 + x_2 x_3 + x_2 x_4 + x_3 x_4,\ \sigma_3 = x_1 x_2 x_3 + x_1 x_2 x_4 + x_1 x_3 x_4 + x_2 x_3 x_4,\ \sigma_4 = x_1 x_2 x_3 x_4 \tag{1.4}$$

グレブナ基底を用いた対称式の基本対称式による計算

ある対称式を、基本対称式を用いて表現するには、グレブナ基底パッケージで多項式 F の正規形を計算するコマンド NormalForm を用いて実現することができます。いくつかの例題を通じてその処理方法を紹介します。
まず、グレブナ基底パッケージを読み込みます。

```
> with(Groebner);
```

[*Basis, FGLM, HilbertDimension, HilbertPolynomial, HilbertSeries, Homogenize, InitialForm, InterReduce, IsProper, IsZeroDimensional, LeadingCoefficient, LeadingMonomial, LeadingTerm, MatrixOrder, MaximalIndependentSet, MonomialOrder, MultiplicationMatrix, MultivariateCyclicVector, NormalForm, NormalSet, RationalUnivariateRepresentation, Reduce, RememberBasis, SPolynomial, Solve, SuggestVariableOrder, TestOrder, ToricIdealBasis, TrailingTerm, UnivariatePolynomial, Walk, WeightedDegree*] **(2.1)**

例題 1

次の対称式 p を考えます。

```
> p := t[1]^2+t[2]^2;
```

$$p := t_1^2 + t_2^2 \quad \textbf{(2.1.1)}$$

p の変数は、indets コマンドを使って求めることができます。

```
> vars := indets(p)[];
```

$$vars := t_1, t_2 \quad \textbf{(2.1.2)}$$

前章で定義した elemsympol 関数で基本対称式を用意します。

```
> es := elemsympol([vars], s);
```

$$es := s_1 = t_1 + t_2, s_2 = t_1 t_2 \quad \textbf{(2.1.3)}$$

生成された基本対称式は等式となっています。グレブナ基底を計算するには「左辺 — 右辺」の形でなければならないので、map コマンドと式の左辺(lhs)及び右辺(rhs)を取り出すコマンドを用いて式の形を変形します。

```
> F := map(p->lhs(p)-rhs(p), [es]);
```

$$F := \left[s_1 - t_1 - t_2, s_2 - t_1 t_2\right] \quad \textbf{(2.1.4)}$$

F のグレブナ基底を計算します。変数順序は、変数のリスト vars で用意されている変数の全次数(total degree)とします。

```
> gb := Basis(F, tdeg(vars, s[1], s[2]));
```

$$gb := \left[t_1 + t_2 - s_1, t_2^2 - t_2 s_1 + s_2\right] \quad \textbf{(2.1.5)}$$

グレブナ基底 gb を用いて、F の正規形を計算します。この結果が p の基本対称式による表現となります。

```
> NormalForm(p, gb, tdeg(vars, s[1], s[2]), 'q');
```

$$s_1^2 - 2 s_2 \quad \textbf{(2.1.6)}$$

得られた式が元の対称式と同一であることを確認するために、変数 es で定義した基本対称式のリストを上記の結果に代入してみます。

```
> eval((2.1.6), [es]);
```

$$\left(t_1 + t_2\right)^2 - 2 t_1 t_2 \quad \textbf{(2.1.7)}$$

この結果を展開すると、確かに元の対称式が得られます。

```
> p = expand((2.1.7));
```

$$t_1^2 + t_2^2 = t_1^2 + t_2^2 \quad \textbf{(2.1.8)}$$

例題 2

例題 1 同様に、対称式を変数 p に定義します。

```
> p := x[1]^3*x[2]+x[1]*x[2]^3+x[1]^2+x[1]*x[2]+x[2]^2;
```

$$p := x_1^3 x_2 + x_1 x_2^3 + x_1^2 + x_1 x_2 + x_2^2 \quad (2.2.1)$$

p 内で用いられている変数のリストを抽出します。

```
> vars := indets(p)[];
```

$$vars := x_1, x_2 \quad (2.2.2)$$

得られた変数リストから基本対称式を用意します。

```
> es := elemsympol([vars], s);
```

$$es := s_1 = x_1 + x_2, s_2 = x_1 x_2 \quad (2.2.3)$$

```
> F := map(p->lhs(p)-rhs(p), [es]);
```

$$F := \left[s_1 - x_1 - x_2, s_2 - x_1 x_2\right] \quad (2.2.4)$$

多項式のリスト F からグレブナ基底を計算します。

```
> gb := Basis(F, tdeg(vars, seq(s[k],k=1..3)));
```

$$gb := \left[x_1 + x_2 - s_1, x_2^2 - x_2 s_1 + s_2\right] \quad (2.2.5)$$

計算したグレブナ基底を用いて、p の正規形を計算します。この結果が対称式を基本対称式で表現したものとなります。

```
> NormalForm(p, gb, tdeg(vars, seq(s[k],k=1..3)), 'q');
```

$$-s_2 - 2 s_2^2 + s_1^2 + s_2 s_1^2 \quad (2.2.6)$$

基本対称式を代入し、元の多項式 p と一致するか確認してみましょう。

```
> eval((2.2.6), [es]);
```

$$-x_1 x_2 - 2 x_2^2 x_1^2 + \left(x_1 + x_2\right)^2 + x_1 x_2 \left(x_1 + x_2\right)^2 \quad (2.2.7)$$

```
> p = expand((2.2.7));
```

$$x_1^3 x_2 + x_1 x_2^3 + x_1^2 + x_1 x_2 + x_2^2 = x_1^3 x_2 + x_1 x_2^3 + x_1^2 + x_1 x_2 + x_2^2 \quad (2.2.8)$$

例題 3

3 変数の場合についても確認してみます。

```
> p := expand(((x[3]-x[1])*(x[1]-x[2])*(x[2]-x[3]))^2);
```

$$p := 2 x_3^2 x_1 x_2^3 - 6 x_3^2 x_1^2 x_2^2 + 2 x_3^3 x_1^2 x_2 - 2 x_3^4 x_1 x_2 + 2 x_1^3 x_3 x_2^2 + 2 x_1^3 x_3^2 x_2 + 2 x_1^2 x_3 x_2^3 \quad (2.3.1)$$
$$- 2 x_1 x_3 x_2^4 - 2 x_1^4 x_2 x_3 + 2 x_3^3 x_1 x_2^2 + x_3^4 x_1^2 + x_3^2 x_2^4 - 2 x_3^3 x_2^3 + x_3^4 x_2^2 - 2 x_1^3 x_3^3 + x_1^4$$
$$x_2^2 + x_1^4 x_3^2 - 2 x_1^3 x_2^3 + x_2^4 x_1^2$$

これまでと同様に、変数のリストを用意します。

```
> vars := indets(p)[];
```

$$vars := x_1, x_2, x_3 \quad (2.3.2)$$

基本対称式を求めます。

```
> es := elemsympol([vars],sigma);
```

$$es := \sigma_1 = x_1 + x_2 + x_3, \sigma_2 = x_1 x_2 + x_1 x_3 + x_2 x_3, \sigma_3 = x_1 x_2 x_3 \quad (2.3.3)$$

基本対称式のリストを、「左辺ー右辺」に書き換えます。

```
> F := map(p->lhs(p)-rhs(p), [es]);
```

$$F := \left[\sigma_1 - x_1 - x_2 - x_3, \sigma_2 - x_1 x_2 - x_1 x_3 - x_2 x_3, \sigma_3 - x_1 x_2 x_3\right] \quad (2.3.4)$$

基本対称式の左辺部分の変数リストを svars で定義します。

```
> svars := seq(sigma[k],k=1..nops([es]));
```

$$svars := \sigma_1, \sigma_2, \sigma_3 \tag{2.3.5}$$

多項式 p の変数のリスト vars と基本対称式の左辺の変数のリスト svars の全次数でグレブナ基底を計算します。

```
> gb := Basis(F, tdeg(vars, svars));
```

$$gb := \left[-\sigma_1 + x_1 + x_2 + x_3,\ x_2^2 + x_2 x_3 + x_3^2 - x_2 \sigma_1 - x_3 \sigma_1 + \sigma_2,\ x_3^3 - x_3^2 \sigma_1 + x_3 \sigma_2 - \sigma_3\right] \tag{2.3.6}$$

得られたグレブナ基底から多項式 p の正規形を計算し、対称式 p の基本対称式による表現を得ます。

```
> NormalForm(p, gb, tdeg(vars,svars));
```

$$-27\,\sigma_3^2 + 18\,\sigma_1 \sigma_2 \sigma_3 - 4\,\sigma_2^3 - 4\,\sigma_1^3 \sigma_3 + \sigma_1^2 \sigma_2^2 \tag{2.3.7}$$

```
> eval((2.3.7), [es]);
```

$$-27\,x_3^2 x_1^2 x_2^2 + 18\left(x_1 + x_2 + x_3\right)\left(x_1 x_2 + x_1 x_3 + x_2 x_3\right) x_1 x_2 x_3 - 4\left(x_1 x_2 + x_1 x_3 + x_2 x_3\right)^3 - 4\left(x_1 + x_2 + x_3\right)^3 x_1 x_2 x_3 + \left(x_1 + x_2 + x_3\right)^2 \left(x_1 x_2 + x_1 x_3 + x_2 x_3\right)^2 \tag{2.3.8}$$

対称式 p から上記の結果を引き算することで結果を確認します。

```
> p - expand((2.3.8));
```

$$0 \tag{2.3.9}$$