```
[289] F = [-3*x^3+4*y^2+(-2*z-3)*y+3*z^2,(-8*y-4)*x^2+(2*z+3)*y,
          -2*x^2-3*x-2*y^2+2*z*y-z^2]$
8.7e-05sec(7.892e-05sec)
[290] buchberger(F)$
49.02sec + gc : 3.573sec(53.4sec)
[291]  gr(F,V,2)$
163.1sec + gc : 4.694sec(168.2sec)
[292]  hgr(F,V,2)$
0.02296sec(0.02374sec)
```

これでわかるように, gr に実装されている改良も, あらゆる入力に対して有効なわけではなく, かえって悪影響を及ぼす場合もある. hgr というのは, gr にある前処理, 後処理を付け加えたもので, おおむね安定だが, やはり遅くなる場合もある. 計算効率の点では, グレブナ基底計算はまだ発展途上であり, 改良や新しいアルゴリズムも発表され続けている. 200310 以降の Asir には nd_ から始まる新しいグレブナ基底計算関数がテスト実装されている;
nd_gr(多項式リスト，変数リスト，標数，順序). 有限体上の計算の場合 gr より 数倍から数十倍高速である.

```
[1039] F = [-3*x^3+4*y^2+(-2*z-3)*y+3*z^2,(-8*y-4)*x^2+(2*z+3)*y,
          -2*x^2-3*x-2*y^2+2*z*y-z^2]$
[1040] nd_gr(F,[x,y,z],13,2);
ndv_alloc=4896
[z^14+4*z^13+2*z^12+7*z^11+ ....  ]
```

なお標数 0, 辞書式順序での計算は係数膨張を招きやすいので, 直接計算をする場合には nd_gr_trace(多項式リスト, 変数リスト,1,1,2) (斉次化つきトレースアルゴリズム) を使うほうが安全である.」

4.5　グレブナ基底と多変数連立代数方程式系の解法

多変数の連立方程式系

$$f_1(X) = \cdots = f_m(X) = 0 \tag{4.1}$$

$(X = (x_1, \ldots, x_n),\ f_i(X) \in \mathbf{Q}[X])$ を解く場合にも, イデアル

$$I = \langle f_1, \ldots, f_m \rangle = \{g_1(X)f_1(X) + \cdots + g_m(X)f_m(X) \mid g_i(X) \in \mathbf{Q}[X]\}$$

を考えることが有効である. 1 変数の場合とちがい, I は一般には単項イデアルではないが, グレブナ基底を計算することにより, 解を求めやすい形に変形することができる.

$\mathbf{Q}[X]/I$ の $\mathbf{Q}$ 上のベクトル空間としての次元 m が有限である場合, I を 0 次元イデアル とよぶ. 根の多重度を適切に定義してやると m は連立方程式系 (4.1) の複素解の個数に一致することが知られている ([1, 5 章 2 節 命題 3, 3 節 命題 8] に多重度のない場合の証明がある).

次の定理は 0 次元イデアルの定義とメンバシップアルゴリズムアルゴリズムを与えている定理 4.1 より容易に証明できる ([1, 5 章 3 節 定理 6] も参照).

定理 4.3 次は同値である.

1. I は 0 次元イデアルである.

2. I の, ある項順序に関する I のグレブナ基底が, 各変数 x_i に対して, initial term が x_i のべきであるような元を含む.

3. I の, 任意の項順序に関する I のグレブナ基底が, 各変数 x_i に対して, initial term が x_i のべきであるような元を含む.

この定理より, 解が有限個かどうかの判定が (任意の項順序の) グレブナ基底により判定できる.

例 4.11 $I = \langle x - yz, y - zx, z - xy \rangle$ の $x > y > z$ なる辞書式順序に関するグレブナ基底は $\{\underline{z^3} - z, \underline{-z^2y} + y, \underline{-y^2} + z^2, \underline{x} - zy\}$ である (冗長な要素は省いてある). よって定理より I は 0 次元イデアルである.

問題 4.7 与えられたイデアルの 0 次元性を判定する関数を書け.

また, この定理を, 辞書式順序の場合に適用すると, 次を得る.

定理 4.4 I を 0 次元イデアルとし, G を $x_1 > x_2 > \cdots > x_n$ なる辞書式順序に関する I のグレブナ基底とすると, G は

$$\begin{array}{rcl} g_1(x_1, x_2, \ldots, x_n) & = & x_1^{d_1} + g_{1,d_1-1}(x_2, \ldots, x_n)x_1^{d_1-1} + \cdots + g_{1,0}(x_2, \ldots, x_n) \\ g_2(x_2, \ldots, x_n) & = & x_2^{d_2} + g_{2,d_2-1}(x_3, \ldots, x_n)x_2^{d_2-1} + \cdots + g_{2,0}(x_3, \ldots, x_n) \\ & \cdots & \\ g_{n-1}(x_{n-1}, x_n) & = & x_{n-1}^{d_{n-1}} + g_{n-1,d_{n-1}-1}(x_n)x_{n-1}^{d_{n-1}-1} + \cdots + g_{n-1,0}(x_n) \\ g_n(x_n) & = & x_n^{d_n} + g_{n,d_n-1}x_n^{d_n-1} + \cdots + g_{n,0} \end{array}$$

という形の元を含む.

この定理により, まず 1 変数多項式 $g_n(x_n)$ の根を求め, それを $g_n(x_{n-1}, x_n)$ に代入すれば, x_{n-1} に関する 1 変数多項式を得る. その根を求め $g_{n-2}(x_{n-2}, x_{n-1}, x_n)$ に代入して, と, 次々に 1 変数多項式の根を求めることで, もとの方程式系の解の候補を求めることができる. (本当の解かどうかは, 他のすべて多項式の零点になっているかどうかをチェックしなければならない. 根を近似で求めてあると, このチェックは容易ではない.)

例 4.12 例 4.11 のイデアル I の零点を求めよう. まず, $z^3 - z = 0$ より $z = -1, 0, 1$.

1. $z = -1$

   $-y^2 + z^2 = 0$ より $y = -1, 1$. このとき $(x, y, z) = (-1, 1, -1), (1, -1, -1)$.

2. $z = 0$

   $-y^2 + z^2 = 0$ より $y = 0$. このとき $(x, y, z) = (0, 0, 0)$.

3. $z = 1$

   $-y^2 + z^2 = 0$ より $y = -1, 1$. このとき $(x, y, z) = (1, 1, 1), (-1, -1, 1)$.

例 4.13 n 変数の多項式を n 本ランダム生成し, それらで生成されるイデアルの辞書式順序グレブナ基底を求めて, どういう形をしているか観察せよ.

この例題の実験を行なう場合, n をあまり大きくすると計算できない. せいぜい 4, 5 程度にする. また, 次数も 2, 3 次程度にしておくこと. 辞書式順序グレブナ基底は, いきなり `gr` などで計算しても大抵ムリなので, Asir マニュアルをよく読んで, 上手な計算方法を学ぶこと. 実際にやってみればわかるように, 辞書式順序グレブナ基底は大変特徴的な形をしている場合が多い:

$$\begin{array}{rcl} g_1(x_1, x_n) & = & x_1 - h_1(x_n) \\ g_2(x_2, x_n) & = & x_2 - h_2(x_n) \\ & \cdots & \\ g_{n-1}(x_{n-1}, x_n) & = & x_{n-1} - h_{n-1}(x_n) \\ g_n(x_n) & = & h_n(x_n) \end{array}$$

これを shape base と呼ぶ. この形の場合には, $g_n(x_n) = 0$ の根を求めれば, 他の変数の値は代入により求まる.

例 4.14 与えられたイデアルが shape base を持つかどうか判定し, shape base を持つ場合に零点の近似値を計算する関数を書け.

1 変数多項式の根は, `pari(roots,Poly)` で計算できる. たとえば, `pari(roots,x^3-1)` で $x^3 - 1 = 0$ の (複素) 近似根を計算できる. ただし,

```
 ***   the PARI stack overflows !
  current stack size: 65536 (0.062 Mbytes)
  [hint] you can increase GP stack with allocatemem()
```

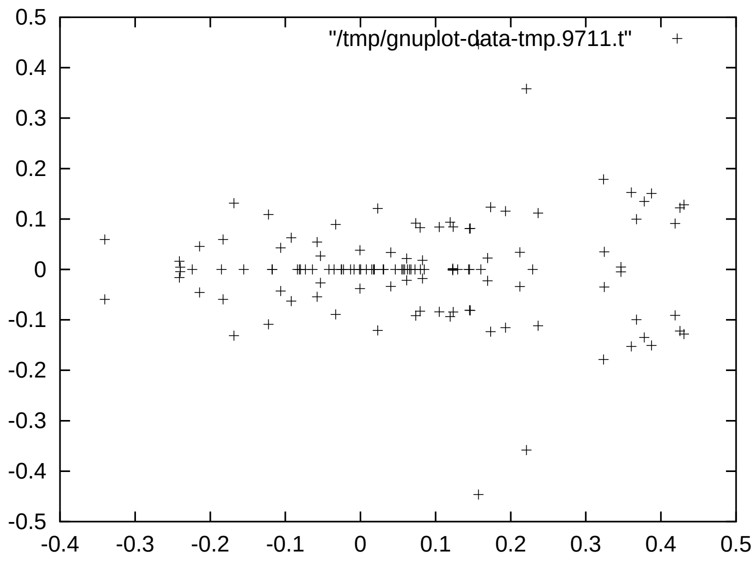
というようなエラーが出た場合には,

```
[295] pari(allocatemem,10^6)$
```

などを実行して, `pari` の使用できるメモリを増やすこと.

```
[826] load("katsura");
1
[831] katsura(7);
[u0+2*u7+2*u6+2*u5+2*u4+2*u3+2*u2+2*u1-1,
2*u6*u0+2*u1*u7-u6+2*u1*u5+2*u2*u4+u3^2,
2*u5*u0+2*u2*u7+2*u1*u6-u5+2*u1*u4+2*u2*u3,
2*u4*u0+2*u3*u7+2*u2*u6+2*u1*u5-u4+2*u1*u3+u2^2,
2*u3*u0+2*u4*u7+2*u3*u6+2*u2*u5+2*u1*u4-u3+2*u1*u2,
2*u2*u0+2*u5*u7+2*u4*u6+2*u3*u5+2*u2*u4+2*u1*u3-u2+u1^2,
2*u1*u0+2*u6*u7+2*u5*u6+2*u4*u5+2*u3*u4+2*u2*u3+2*u1*u2-u1,
u0^2-u0+2*u7^2+2*u6^2+2*u5^2+2*u4^2+2*u3^2+2*u2^2+2*u1^2]
```

Knoppix/Math で解く方程式: ギャラリ: $n = 7$ での katsura の方程式系. u0 から u7 が未知変数.



Knoppix/Math で解く方程式: ギャラリ: $n = 7$ での katsura の方程式系の解の第一成分 u0

課題: Buchberger の原論文の内容をまとめなさい.
参考: 記号の対称表.

この本での記号	Buchberger の原論文
$\mathrm{in}_{\prec}(f)$	$LPP(f)$ (leading power product)
monomial ideal M についての標準 monomial	MPP (monomial power product) relative M

Knoppix/math には, 中山君による visual Buchberger が用意されている. (icms2006-non-free の folder から Computer Algebra Animation へ.)

4.6 微分方程式の差分化より得る連立代数方程式の解法例

次のような手順で調べていく.

1. 解が有限個か?
2. 複素数解は何個か?
3. 準素 ideal 分解 (因数分解みたいなもの) できるか? 方程式がより簡単になればよい.
4. lexicographic order の gb は直接計算できるか?

5. 上の直接計算がつらいときは, 計算が楽になる weight を探し, asir なら `minipoly` とか, 基底を変換する `tolex`, `tolex_gsl` を試す.
6. ideal に含まれる 1 変数代数方程式を解く.
7. 解のたしかめ.
8. phc などの numerical homotopy による solver を試す.

Todo: くわしい説明.

4.6.1 Lorentz モデル

Lorentz モデルの平衡点の座標を計算してみよう. Lorentz 方程式の場合, 未知変数 p_1, p_2, p_3 に対する連立代数方程式 (3.4)

$$
\begin{aligned}
0 &= -ap_1 + ap_2, \\
0 &= -p_1p_3 + bp_1 - p_2, \\
0 &= p_1p_2 - cp_3
\end{aligned}
$$

の解が 定常状態 である. これを $a = 10, b = 20, c = 2.66$ の場合に計算してみる.
asir 版. `Egr/a-lorentz.txt`

```
A=10; B=20; C=266/100;
F=[-A*p1+A*p2, -p1*p3+B*p1-p2, p1*p2-C*p3];
V=[p1,p2,p3];
G=hgr(F,[p1,p2,p3],0);
print(["zero dim?", zero_dim(G,V,0)]);
M=dp_mbase(map(dp_ptod,G,V));
print(M);
P3=minipoly(G,V,0,p3,t);
print(pari(roots,P3));
P2=minipoly(G,V,0,p2,t);
print(pari(roots,P2));
P3=minipoly(G,V,0,p3,t);
print(pari(roots,P3));
P1=minipoly(G,V,0,p1,t);
print(pari(roots,P1));  /* Use subst */

print(primadec(F,V));

L=tolex(G,V,0,V);  /* tolex does not accept coeff like 2/3 */
LG = tolex_gsl(G,V,0,V);
end$
```

M2 で上の計算の一部をやる場合. `Egr/a-m2.txt`

```
R=QQ[p1,p2,p3];
F= ideal {-10*p1+10*p2, -p1*p3+20*p1-p2, p1*p2-(266/100)*p3};
G= gens gb F;
print G;
print dim F;
```

PHC (homotopy 法による solver を呼ぶ場合) `Egr/a-phc.txt`

```
F=[-10*p1+10*p2, -p1*p3+20*p1-p2, 100*p1*p2-266*p3];
Ans=phc.phc(F);
print("phc interface is now buggy (not fixed). See tmp.output.0");
end$
```

参考. この問題は簡単なので maxima の solve コマンドで一発でとける. `Egr/a-mac.txt`

```
e:solve([-10*p1+10*p2=0,
         -p1*p3+20*p1-p2=0,
         p1*p2 - (266/100)*p3=0],
        [p1,p2,p3]);
float(e);
```

求まった定常状態が安定かどうかは, 次のプログラムで確かめる.
`Egr/a-lo-diff.txt`

```
load("glib3.rr");

def lorentz(P1,P2,P3) {
  glib_window(-25,-25,25,25);
  A=10; B=20; C=2.66;
  Dt = 0.004; T = 0;
  while (T <50) {
      Q1=P1+Dt*(-A*P1+A*P2);
      Q2=P2+Dt*(-P1*P3+B*P1-P2);
      Q3=P3+Dt*(P1*P2-C*P3);
      glib_putpixel(Q1,Q2);
      T=T+Dt;
      P1=Q1; P2=Q2; P3=Q3;
  }
}

print("try to input, for example, lorentz(0,3,0); ");
end$
```

関数が有効に使われている. `lorentz(7.1,7.1,19);` などと初期条件を変えながら実行してみよう.

4.6.2 1次元の反応拡散方程式

次は1次元の反応拡散方程式から導出した連立代数方程式 (3.7) をグレブナ基底で解析するプログラムである.

Todo: プログラムを成形してから 空白をとる. @@code Egr/parter2.rr

4.6.3 1次元, 2 成分の反応拡散方程式

(3.9) に付随する代数方程式系の解析.

Todo: プログラムを成形してから 空白をとる. @@code Egr/pps-alg.rr

Todo: fluid2d.rr (野呂のコメント待ち)

4.6.4 Maple

Maple 11, Noro による例. 確率算法のため, 答が 1 でない. でも計算は早い. `Egr/maple-gb-bug.ml`

```
with(Groebner);
m:=9745285485474621232044564682715507257125628266320184805049402135023604972160190092249693410175276039149793804653137872
B:=[c+m,c]:
g:=Basis(B,tdeg(c));
```

Todo: Singular, Maple 11(SALSA) の例.

関連図書

[1] D.Cox, J.Little, D.O'Shea, *Ideals, Varieties, and Algorithms — An Introduction to Commutative Algebraic Geometry and Commutative Algebra*, 1991, Springer-Verlag.
日本語訳: D. コックス, J. リトル, D. オシー: グレブナ基底と代数多様体入門 (上/下). 落合他訳, シュプリンガー フェアラーク 東京, 2000. ISBN 4-431-70823-5, 4-431-70824-3.
世界的に広く読まれているグレブナ基底の入門書. Buchberger アルゴリズム自体は, 2 章までよめば理解できる. Risa/Asir ドリルの **??** 章 (本章) および **??** 章 (次の章) はコックス達の本をもとにした, グレブナ基底の入門講義等の補足プリントがもとになっている. したがってコックス達の本とともに本章と次の章を読むと理解が深まるであろう. 本章で証明や説明を省略した数学的事実や概念については, コックス達の本の 1 章を参照されたい. 大学理系の教養課程の数学の知識で十分理解可能である.

[2] 野呂: 計算代数入門, Rokko Lectures in Mathematics, 9, 2000. ISBN 4-907719-09-4.
`http://www.math.kobe-u.ac.jp/Asir/ca.pdf` から, PDF ファイルを取得できる.
`http://www.openxm.org` より openxm のソースコードをダウンロードすると, ディレクトリ `OpenXM/doc/compalg` にこの本の TeX ソースがある.

[3] D.E. Knuth: *The Art of Computer Programming*, Vol2. Seminumerical Algorithms, 3rd ed. Addison-Wesley (1998). ISBN 0-201-89684-2.
日本語訳: "準数値算法", サイエンス社.

[4] Risa/Asir ドリル (これは Free Book です)
`http://www.math.kobe-u.ac.jp/HOME/taka/2007/asir-book-2007.pdf` が最新.

[5] 大阿久俊則: D-加群と計算代数, 朝倉書店.
微分方程式とグレブナ基底の入門書. グレブナ基底に関する証明付きの簡潔な解説もある.

[6] 野呂, 横山: グレブナー基底の計算 基礎篇 計算代数入門, 東京大学出版会, 2003. ISBN 4-13-061404-5.
グレブナー基底の基礎から代数方程式系の解法, イデアルの分解まで詳しく解説してある.

[7] 齊藤, 竹島, 平野: グレブナー基底の計算 実践篇 Risa/Asir で解く, 東京大学出版会, 2003. ISBN 4-13-061405-3.
一変数, 多変数の代数方程式 (系) のさまざまな解法を, Risa/Asir 上で実際に計算を行うためのプログラムを示しながら解説している.

[8] Gröbner basis 用のシステムの比較サイト. まだまだ不完全.
`http://waxwing.risc.uni-linz.ac.at/people/ckocsis/GBtest/index.php`