計算機を使う観点からの導入

野呂 正行 [1]　　**濱田 龍義** [2]

[1] **神戸大学**/JST CREST

[2] **福岡大学**/JST CREST

VMware Player/Fusion のインストール

- VMware Player (Windows 用)
  1. Google で VMware Player を検索して, ダウンロードページに行く.
     最新版は 9/3 時点で 2.5.3 である.
  2. いくつかの質問に答える
     email アドレスを含む質問があるが, 正しく入力しても, いままで実際に email が届いたことは一度もない.
  3. ダウンロードしたらインストール
- VMware Fusion (Mac 用)
  VMware Fusion (残念ながら有料) を入手, インストールする. こちらは売り物なので, 説明書その他に従って下さい.

仮想マシンファイル

1. **KNOPPIX/Math DVD 版の ISO イメージ**
   knxm2008n-kobe.iso; **4GB**
2. **仮想マシン本体**
   次のいずれかを使う.
   - math2008n-crest-0912.exe
     Windows 用自己解凍ファイル, NTFS 用 (通常はこちら)
   - math2008n-crest-0912.zip
     Mac の内蔵ディスク用 (通常はこちら)
   - math2008n-crest-2G-0912.exe
     Windows 用自己解凍ファイル, FAT32 用
   - math2008n-crest-2G-0912.zip
     Mac の 外付け FAT32 ディスク用

仮想マシンのインストール

1. インストール先の選択
   十分な空き領域 (余裕をみて最低 6GB) のあるパーティションを選ぶ.
2. インストール先ファイルシステムの確認
   Windows なら
   (マイ) コンピュータ->当該ディスク右クリック-> プロパティ
3. 仮想マシンファイルを選択したパーティションにコピー
   USB メモリ, SD カード, USB ポータブル HDD などから仮想マシンファイル (iso と仮想マシン本体) をコピーする.
4. 仮想マシン本体の展開
   仮想マシン本体ファイルを実行して仮想マシンを展開
   $\Rightarrow$ フォルダ `math2008n-crest` ができる.
5. iso ファイルを仮想マシンフォルダ内に移動する.

仮想マシンの起動, 終了

- 仮想マシンフォルダ内の `math2008n.vmx` ファイルをダブルクリックする.
  これはメモリを 512MB 使用する設定.
  ダイアログがでたら Enter を押せばよい.
- 実メモリが少ない場合には, `math2008n-256M.vmx` を使う (256MB 使用).
- 起動後にメモリ量を変更できる.
  `VMware Player`->トラブルシューティング (Mac の場合仮想マシン->設定) から変更できる.
  メモリがふんだんにある場合には, 適宜増やすのもよい.
- 終了は, K メニュー -> ログアウト
  終了直前に出るダイアログには単に Enter を入力
  ⇒ 「はい」を選ぶと, 再起動は失敗する
  ⇒ VMware Player -> トラブルシューティング -> パワーオフして終了. その後改めて起動すれば OK.

共有フォルダ

1. 共有フォルダに指定するフォルダを作成
   ホスト側で, 共有フォルダに指定するディレクトリを作成
2. 共有フォルダを有効にする.
   仮想マシン起動後 VMware Player->共有フォルダ (Mac の場合 仮想マシン->共有フォルダ) で共有フォルダを有効にする.
3. 作成したディレクトリを共有フォルダに指定する.
   Mac の場合, 一度デフォルトの設定を削除してから, 新規に共有フォルダを追加する.
   フォルダを共有フォルダに指定してから, 名前を shared_folder に変更する.
4. デスクトップアイコンの変更
   下部のペンギンから Mount Shared Folder を実行
   実体は /mnt/hgfs/shared_folder である. シェルからアクセスする場合はこのパス名を用いる.

suspend, resume

- suspend
  仮想マシンウィンドウを×で閉じると, 現在の状態をセーブして suspend 状態となる.
  この状態で, ホストマシンをシャットダウンすることができる.
  仮想マシンフォルダ内に, 一時停止のマークのついたアイコンが見える.
- resume
  suspend 状態で, で, vmx ファイルをダブルクリックすると, resume する.
- 仮想マシンの移動, コピー
  仮想マシンフォルダごと移動, コピーできる.

プリンタの設定 (PS プリンタの場合)

1. ペンギン->Confugure->Configure Printer を実行
2. 追加->プリンタ/クラスを追加
3. バックエンド選択でリモート LPD キューを選択
4. LPD キュー情報でプリンタホスト, キュー名を入力
   ホスト：p-418.math.kobe-u.ac.jp　キュー：PS_DUP
5. プリンタ機種選択で Postscript プリンタを選択
6. プリンタテスト->設定
   Page Size：A4　Double-Sided Printing：Long Edge
   Miscellaneous->GhostScript pre-filtering：Convert to PS level2
7. 一般情報で名前をつける
   p-418 としておく.

以上により, lpr -Pp-418 ... でファイルが印刷できる.

数学ソフトウェアに関する文書の検索

数学ソフトウェアのマニュアル, 参考書はいくつかの場所に分散している.

- `/usr/share/doc`
  種々の文書がおかれるディレクトリ
- `/usr/local/Math-ja`
  日本語文書がある.
  `knoppix-math` **からリンクされている.**
- **デスクトップの Math-Doc-Search**
  Math-Doc-Search を起動し, Query にキーワード (日本語 OK) を並べてサーチする.
  $\Rightarrow$ **大抵のものを簡単に探し当てることができる.**

その他

- 背景 (壁紙) の変更
  背景上で右クリック -> デスクトップを設定
  からできる. キャラクターなしの壁紙は
  `/cdrom/KNOPPIX/background.jpg`
- USB メモリの使用
  1. USB メモリを挿す
     ダイアログが出たら,「何もしない」を選んで OK
  2. USB メモリのアイコンがデスクトップに現れる
     クリックすれば開く
  3. 書き込み可にする
     アイコン右クリックで Change read/write mode
  4. 取り外す前にアンマウント

  Windows 側に渡す場合には, 上部のデバイスメニューから切断すればよい

Macaulay2 の起動

- $\sqrt{x}$ メニュー, または KNOPPIX-Math-Start から起動する.
  Konsole が起動し, その中で Macaulay2 が起動する. (emacs) を選ぶと emacs のバッファ内で Macaulay2 が起動する. getting started で推奨されている使い方である.
- 端末エミュレータ (Konsole など) から起動する.
  自分で立ち上げた端末エミュレータのシェルからコマンド `M2` を実行すると, その端末エミュレータ内で Macaulay2 が起動する.

Macaulay2：ヘルプその他

- **ヘルプ, マニュアル**
  コマンド `viewHelp` **を実行すると, ブラウザが起動する. 最初は,**

  ```
  Macaulay 2 -> getting started -> a first Macaulay
  ```

  などをざっと眺めてみることをお勧めする.
  個々のコマンドは, `index` **から調べることができる.**
- **ファイルのロード**
  ファイルのロードは `load`**, パッケージのロードは** `loadPackage` **で行う.**

Macaulay2：基礎環の宣言と多項式の入力

- Macaulay2 では, 基礎環を明確に宣言する必要がある.
- 係数体として, 有理数体は QQ, 有限体 $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ は ZZ/p により入力できる.
- ある基礎環が設定されている場合, そこに含まれない不定元を使用しようとしても拒否される.
- 各多項式はそれが生成された時の環に属する.
  異なる環に属する多項式は, 含まれる不定元が一致していても, 一方を他方の環に写す必要がある.

Macaulay2：基礎環および項順序

基礎環の宣言と多項式の入力

```
i1 : R=QQ[x,y,z];
i2 : f=(x+y+z)^2
      2           2                   2
o2 = x  + 2x*y + y  + 2x*z + 2y*z + z
o2 : R
i3 : g=y+u
stdio:3:4:(1):[0]: error: no method for binary operator + applied to
objects:
--            y (of class R)
--      +     u (of class Symbol)
i4 : S=QQ[x,y,z,u]
i5 : f+u
stdio:5:2:(1):[0]: error: expected pair to have a method for '+'
i6 : h=(map(S,R))(f);
i7 : h+u
      2           2                   2
o7 = x  + 2x*y + y  + 2x*z + 2y*z + z  + u
o7 : S
```

Macaulay2：項順序

項順序は基礎環を定義するときに指定する.
デフォルトは全次数逆辞書式順序であり, それ以外の順序を指定する場合には `MonomialOrder` により指定する. 代表的なものを挙げる.

- 辞書式順序
  例：`QQ[x,y,z,MonomialOrder=>Lex]`
  この例は, $x > y > z$ なる辞書式順序を持つ多項式環を宣言している.
- ブロック項順序
  例：`ZZ/37[x,y,z,u,v,MonomialOrder=>{2,3}]`
  この例は, $\{x, y\} >> \{z, u, v\}$ で, 各ブロックに全次数逆辞書式を適用するブロック順序を持つ多項式環を宣言している.

多項式の先頭項は Macaulay2 では lead monomial と呼ばれ, `leadMonomial` で取り出せる. その係数は `leadCoefficient`, 係数つきの先頭項は `leadTerm` で取り出せる.

Macaulay2 : グレブナー基底の計算

- **Macaulay2 でのイデアルの生成は** $\texttt{ideal}(p_1, \ldots, p_l)$ **による.**
- **グレブナー基底は** gb **で行う.**
  項順序は環が知っているので引数はイデアル 1 つのみである. 結果はグレブナー基底というオブジェクトで返される.
- **生成系は** gens **により行列 (行ベクトル) として取り出せる.**
  生成系 g **の** i **番目の要素は,** g_i **により取り出せる.** i **は** 0 **から始まることに注意する.**

例：cyclic-7

次の例は, cyclic-7 の全次数逆辞書式順序によるグレブナー基底計算である.

Macaulay2 によるグレブナー基底計算

```
i1 : R=QQ[c0,c1,c2,c3,c4,c5,c6];
i2 : I=ideal(c6*c5*c4*c3*c2*c1*c0-1,...
o2 : Ideal of R
i3 : G=gb I;
i4 : g=gens G;
                1        209
o4 : Matrix R  <--- R
i5 : g_0
o5 = | c0+c1+c2+c3+c4+c5+c6 |
       1
o5 : R
```

Macaulay2：イニシャルイデアルの計算

- イデアル $I \subset R$ $(R = K[x_1, \ldots, x_n])$ のグレブナー基底 G の先頭項から I のイニシャルイデアル $\mathrm{in}(I)$ が得られる.
- R/I は G の先頭項で割れない単項式全体 M で K 上張られる.
  M の元を標準単項式 (standard monomial) と呼ぶ.
- I の次元が 0 なら標準単項式は有限個である.
  これは $R/\mathrm{in}(I)$ の $\mathbb{Q}-$ 基底を与える `basis` により得られる.
- $\dim_K R/\mathrm{in}(I) = \dim_K R/I$ であり, この値は, $\overline{K}^n$ における I の零点の重複度込みの個数に等しい.

例：イニシャルイデアルの計算

Macaulay2 によるイニシャルイデアルの計算

```
i1 :  R=QQ[x,y,z];
i2 : I=ideal(x^2*y^2-z^2,x^3-y*z^2,x^2*z^4-y^2);
o2 : Ideal of R
i3 : J=ideal leadTerm I
              3   2 2   3 2   5   6   2 4
o3 = ideal (x , x y , y z , y , z , x z )
o3 : Ideal of R
i4 : dim I
o4 = 0
i5 : S=R/J
o5 = S
o5 : QuotientRing
i6 : basis S
o6 = | 1 x x2 x2y x2yz x2yz2 x2yz3 x2z x2z2 x2z3 xy xy2 xy3 xy4    ...
     ------------------------------------------------------------ ...
     xy2z2 xy2z3 xy2z4 xy2z5 xyz xyz2 xyz3 xyz4 xyz5 xz xz2 xz3 xz4  ...
     ------------------------------------------------------------ ...
     y4 y4z y3z y2z y2z2 y2z3 y2z4 y2z5 yz yz2 yz3 yz4 yz5 z z2 z3 z4 z5 |
              1       52
o5 : Matrix S  <--- S
```

Macaulay2：商および剰余の計算

Macaulay2 では, 多項式をグレブナー基底または行列で割った商および剰余が計算できる.

- `remainder`(f, g) : f を g で割った剰余 r を返す.
- `quotient`(f, g) : f を g で割った商 q を返す.
- `quotientRemainder`(f, g) : f を g で割った商 q, 剰余 r に対し sequence (q, r) を返す.

- 引数 f は行列, g はグレブナー基底または行列
- g がグレブナー基底の場合, 商は 0
- g が行列の場合, $gq + r = f$ を満たす q, r が計算される. g がイデアルの生成系を並べた行ベクトルの場合, $q_0 g_0 + \cdots + q_l g_l = f$ を満たす $q_0, \ldots, q_l$ が列ベクトルとして返される.
- 剰余計算の最も簡単な応用=メンバーシップ, 包含関係
  $f \in I \Leftrightarrow f$ を I のグレブナー基底による剰余が 0
  $I \subset J \Leftrightarrow I$ の生成系の各元の, J のグレブナー基底による剰余

例：メンバーシップテスト

商および剰余の計算

```
i1 : R=QQ[x,y,z];
i2 : I=ideal(x^4*y^2+z^2-4*x*y^3*z-2*y^5*z,x^2+2*x*y^2+y^4);
i3 : G=gb I;
i4 : g=gens G;
i5 : f=y*z-x^3;
i6 : remainder(matrix{{f}},G)
o6 = | -x3+yz |
i7 : remainder(matrix{{f^2}},G)
o7 = | 2x2y3z+2x3yz+2y2z2+2xz2 |
i8 : remainder(matrix{{f^3}},G)
o8 = 0
i9 : qr=quotientRemainder(matrix{{f^3}},g);
o9 : Sequence
i10 : q=qr_0;
i11 : g*q
o11 = | -x9+3x6yz-3x3y2z2+y3z3 |
i12 : g*q-f^3
o12 = 0
```

Macaulay2：消去法

- **消去イデアルの計算**
  I **を多項式環** $K[Z]$ $(Z = X \cup Y, X \cap Y = \emptyset)$ **のイデアルとするとき,** $I_Y = I \cap K[Y]$ **の生成系は,** $X >> Y$ **なる任意の消去順序** $<$ **に関する** I **のグレブナー基底** G **に対し** $G_Y = G \cap K[Y]$ **により与えられる.**
- G_Y **は既にグレブナー基底**
  G_Y **は** I_Y **の** $<_Y = <|_{K[Y]}$ **に関するグレブナー基底になっている.**
- **消去順序としては, 計算効率の問題から, 通常はブロック順序を使うのが望ましい.**
- `selectInSubring`
  G **から** G_Y **を求める.**
- `selectInSubring`(i, m)
  行列 m **から** i **番目 (この場合は** $i \geq 1$**) までのブロックに属する変数を含まない列のみを取り出した行列を返す.**

例：消去イデアル

次の例では $I \cap \mathbb{Q}[z]$ の生成系を計算している. $\{x, y\} >> \{z\}$ で, 各ブロックで全次数逆辞書式順序を適用するブロック順序を設定してグレブナー基底 G を計算したあと, 変数が z のみからなる多項式を G から取り出して Gz としている.

消去イデアルのグレブナー基底の計算

```
i1 : R=QQ[x,y,z,MonomialOrder=>{2,1}];


i2 : I=ideal(x^2-z,x*y-1,x^3-x^2*y-x^2-1);
o2 : Ideal of R
i3 : G=gens gb I;
              1        3
o3 : Matrix R  <--- R
i4 : Gz=selectInSubring(1,G)
o4 = | z3-3z2-z-1 |
```

Macaulay2：イデアル演算 1

$R = K[x_1, \ldots, x_n]$ とする.

- **イデアルの共通部分**
  R のイデアル $I = \langle f_1, \ldots, f_k \rangle$, $J = \langle g_1, \ldots, g_l \rangle$ に対し, t を新しい変数とすれば

$$I \cap J = \langle tf_1, \ldots, tf_k, (1-t)g_1, \ldots, (1-t)g_l \rangle \cap R$$

  ただし, 右辺のイデアルは $K[x_1, \ldots, x_n, t]$ で考える. よって, 消去イデアル計算により共通部分が計算できる.
- **イデアル商**
  R のイデアル I, J に対し $I : J = \{f \mid fJ \subset I\}$ である.
  $J = \langle g_1, \ldots, g_l \rangle$ なら $I : J = \bigcap_{i=1}^{l} I : \langle g_i \rangle$ である. $I : \langle g \rangle$ を $I : g$ と書く. $I : g = (I \cap \langle g \rangle)/g$ である. 右辺は, $I \cap \langle g \rangle$ の生成系の各元を g で割ったもので生成されるイデアルである. よって, $I : J$ は共通部分計算により計算できる.

Macaulay2：イデアル演算 2

- **saturation**

  R **のイデアル** I, J **に対し** $I : J^{\infty} = \bigcup_{m=1}^{\infty} (I : J^m)$ **である.**

  $J = \langle g_1, \ldots, g_l \rangle$ **なら** $I : J = \bigcap_{i=1}^{l} (I : \langle g_i \rangle^{\infty})$ **である.** $I : \langle g \rangle^{\infty}$ **を** $I : g^{\infty}$ **と書く.** $I : g^{\infty} = (I + \langle tg - 1 \rangle) \cap R$ **である. ただし右辺のイデアルは** $K[x_1, \ldots, x_n, t]$ **で考える. よって saturation は共通部分計算により計算できる.**

- $f \in \sqrt{I}$ **の判定 (radical メンバーシップ判定)**

  R **のイデアル** I, $f \in R$ **に対し,**

  $f \in \sqrt{I} \Leftrightarrow I + \langle tf - 1 \rangle = K[x_1, \ldots, x_n, t]$ **である.**

  $I + \langle tf - 1 \rangle = K[x_1, \ldots, x_n, t]$ **は** $I + \langle tf - 1 \rangle$ **の (任意項順序に関する) 簡約グレブナー基底が** $\{1\}$ **であることと同値だから,** $f \in \sqrt{I}$ **か否かはグレブナー基底を計算することで判定できる.**

例：イデアル演算

イデアルの演算

```
i1 : R=QQ[x,y];
i2 : I=ideal(x^4-y^5,x^3-y^7);
o2 : Ideal of R
i3 : I1=quotient(I,x)
              5    4   3 2    2   7    2 3
o3 = ideal (y  - x , x y  - x , x  - x y )
o3 : Ideal of R
i4 : I2=quotient(I,x^2)
              2 2       5    4   6      3
o4 = ideal (x y  - x, y  - x , x  - x*y )
i5 : I3=quotient(I,x^3)
                2       5    4   5    3
o5 = ideal (x*y  - 1, y  - x , x  - y )
i6 : I4=quotient(I,x^4)
                2       5    4
o6 = ideal (x*y  - 1, y  - x )
i7 : J=saturate(I,x)
                2       5    4   5    3
o7 = ideal (x*y  - 1, y  - x , x  - y )
i8 : I3==I4
o8 = true
i9 : I2==I3
o9 = false
```

例 : radical メンバーシップ

radical メンバーシップ判定

```
i1 : R=QQ[t,x,y,z];
i2 : I=ideal(x^4*y^2+z^2-4*x*y^3*z-2*y^5*z,
     x^2+2*x*y^2+y^4);
o2 : Ideal of R
i3 : f=y*z-x^3;
i4 : gens gb (I+ideal(t*f-1))
o4 = | 1 |
```

グレブナー基底が $\{1\}$ **なので,** $f \in \sqrt{I}$ **と判定できる.**

Asir の起動方法

- $\sqrt{x}$ メニュー, または KNOPPIX-Math-Start から起動する.
  (openxm) の方を起動すれば, 種々のライブラリファイルを自動的に読みこんで起動する.
- 端末エミュレータから起動する.
  Asir 単体ではコマンドライン編集機能を持たないので, `openxm fep asir` を実行する.

Asir：ヘルプその他

- **ヘルプ, マニュアル**
  ヘルプは `help("function")` **で引ける. マニュアルはデスクトップの Math-Doc-Search で引くか,** `helph()` **コマンドでブラウザを立ち上げて HTML 形式のマニュアルを見るのが便利である.**
- **ファイルのロード**
  ファイルは `load` **により行う. 環境変数** `ASIRLOADPATH` **で指定されたディレクトリを順に探す. この値は, シェルから** `openxm env` **を実行すると見ることができる.**

Asir：多項式の入力

- Asir ではアルファベット小文字で始まり, アルファベット, 数字, _ (アンダースコア) からなる文字列が不定元である.
- 入力された多項式は再帰表現により保持されている. 再帰表現とは, 多項式を, 主変数に関する一変数多項式として表現するもので, 係数は, 主変数を含まない多項式である.

多項式の入力

```
[1518] F=(x+y+z)^2;
x^2+(2*y+2*z)*x+y^2+2*z*y+z^2
[1519] G=F+u;
x^2+(2*y+2*z)*x+y^2+2*z*y+z^2+u
```

Asir：分散表現多項式と項順序

- グレブナー基底に関連する計算は, 分散表現で行われる.
  グレブナー基底関連計算を行う場合, 暗黙あるいは明示的に分散表現への変換を行う.
- グレブナー基底関連計算など, 項順序が必要が計算の都度, 項順序を指定する必要がある.
- Asir においては, 項順序は変数順序と項順序型により指定される.
  変数順序は不定元を並べたリストで表現する.
  この順序は単項式を指数ベクトルで表示する場合の各指数のインデックスを決める.
  例えば, 変数順序が $[x, y, z, u, v, w]$ で与えられた場合, $x^a y^b z^c u^d v^e w^f$ は (a, b, c, d, e, f) で表示される.

Asir：項順序型の設定

変数リストに対し, 次のような項順序型が設定できる.

- 単純な項順序型

  0 ：全次数逆辞書式順序
  1 ：全次数辞書式順序
  2 ：辞書式順序

- ブロック項順序型

  $[[O_1, n_1], [O_2, n_2], \ldots, [O_l, n_l]]$ なるリストのリスト
  典型例：$[[0, n_1], [0, n_2]]$：先頭の n_1 変数を消去するための消去順序

  1. 変数リストを左から $n_1, n_2, \ldots, n_l$ $(n_1 + \cdots + n_l = n)$ ずつのブロックに分ける.
  2. i 番目のブロックに単純項順序型 O_i を適用する
  3. 1 番目のブロックから, 大小が決まるまで順に行う.

Asir：分散表現多項式の操作

- dp_ord(Ord)
  項順序型の設定. 項順序型は関数の引数として与える場合もある.
- dp_ptod(F,V)
  設定されている項順序型, 変数順序 V で定まる項順序で, 多項式 F を分散表現に変換する.
- dp_ht
  先頭項 (係数 1) の取り出し
- dp_hc
  先頭係数の取り出し
- dp_hm
  係数つきの先頭項の取り出し

用語は初期に使われていたものを採用しており, 最近の用法と異なることに注意されたい.

例：分散表現多項式の演算

分散表現への変換, 演算

```
[1532] F=x^2*y+y^3*z+x*z+x+1;
y*x^2+(z+1)*x+z*y^3+1
[1533] dp_ord(0)$
[1534] DF0=dp_ptod(F,[x,y,z]);
(1)*<<0,3,1>>+(1)*<<2,1,0>>+(1)*<<1,0,1>>+(1)*<<1,0,0>>
+(1)*<<0,0,0>>
[1535] dp_ord(2)$
[1536] DF2=dp_ptod(F,[x,y,z]);
(1)*<<2,1,0>>+(1)*<<1,0,1>>+(1)*<<1,0,0>>+(1)*<<0,3,1>>
+(1)*<<0,0,0>>
[1537] G=F+u;
y*x^2+(z+1)*x+z*y^3+u+1
[1538] DG=dp_ptod(G,[u,x,y,z]);
(1)*<<1,0,0,0>>+(1)*<<0,2,1,0>>+(1)*<<0,1,0,1>>
+(1)*<<0,1,0,0>>+(1)*<<0,0,3,1>>+(1)*<<0,0,0,0>>
[1539] dp_ht(DG);
(1)*<<1,0,0,0>>
```

Asir : グレブナー基底の計算

gr をロードしておく. (KNOPPIX/Math では不要)

$Plist$:イデアルを表す多項式リストである.
$Vlist$: 変数リスト
Ord : 項順序型

- nd_gr($Plist, Vlist, Char, Ord$)
  $\langle Plist \rangle$ の簡約グレブナー基底を計算する.
  $Char = 0$ のとき有理数体係数,
  $Char$ が素数のとき有限体 $\mathbb{F}_{Char}$ 上で計算する.
  結果は多項式のリストである. リスト G の i 番目の要素は $G[i]$ (i は 0 から始まる) で取り出せる.
- nd_gr_trace($Plist, Vlist, Homo, Prime, Ord$)
  $\langle Plist \rangle \subset \mathbb{Q}[Vlist]$ の簡約グレブナー基底を計算する.
  $Prime$ は 1 を指定しておく.
  $Homo$ が 1 のとき, 斉次化を経由して計算する.
  $Homo$ が 0 の とき斉次化を経由しないで計算する.
  ほとんどの場合 $Homo = 1$ が安全

例：グレブナー基底計算

Asir によるグレブナー基底計算

```
[1517] load("cyclic")$
[1527] C=cyclic(7);
[c6*c5*c4*c3*c2*c1*c0-1,...]
[1528] V=vars(C);
[c0,c1,c2,c3,c4,c5,c6]
[1529] nd_gr(C,V,31991,0)$
...
2.016sec + gc : 0.072sec(2.089sec)
[1530] nd_gr(C,V,0,0)$
```

(5 分待って中断)

```
[1530] G=nd_gr_trace(C,V,1,1,0)$
...
19.54sec + gc : 5.428sec(25.02sec)
[1531] G[0];
(((23853922665902000713066 2*c6*c4-...
[1532] length(G);
209
```

nd_gr：係数膨張のため計算が進まなくなる.

nd_gr_trace + $Homo = 1$：25 秒で計算が終了する.

Asir：イニシャルイデアルの計算

1. グレブナー基底を計算する.
2. 基底の各元を dp_ptod で分散表現に変換
3. dp_ht で先頭項を取り出す
4. 必要があれば, dp_dtop で再帰表現に戻す

0 次元 ⇒ dp_mbase により, 標準単項式全体を計算できる.

Asir によるイニシャルイデアルの計算

```
[1517] B=[x^2*y^2-z^2,x^3-y*z^2,x^2*z^4-y^2];
[y^2*x^2-z^2,x^3-z^2*y,z^4*x^2-y^2]
[1518] V=[x,y,z]$
[1519]  G=nd_gr(B,V,0,0);
[z^4*x^2-y^2,-y^4+z^6,-y^2*x+y^5,-z^2*x+z^2*y^3,y^2*x^2-z^2,x^3-z^2*y]
[1520] D=map(dp_ptod,G,V)$ H=map(dp_ht,D)$
[1521] [1522] map(dp_dtop,H,V);
[z^4*x^2,z^6,y^5,z^2*y^3,y^2*x^2,x^3]
[1523] map(dp_dtop,dp_mbase(H),V);
[z^5*y^2*x,z^4*y^2*x,z^5*y*x,z^5*y^2,z*y^4*x,z^3*y*x^2,...]
[1524] length(@@);
52
```

Asir：剰余計算

p_nf：剰余の分母を払って整数係数で返す
剰余が 0 か否かの判定に用いる.
p_true_nf：$[num, den]$ なるリストを返す
num/den が真の剰余となる.

剰余計算

```
[1517] B=[u2*u0-2*u2+3,(2*u1-1)*u0^2-u0-2*u2,
2*u1^3+u2+4]$
[1518] V=[u0,u1,u2]$
[1519] G=nd_gr(B,V,0,0);
[10*u2^4+126*u2^3+637*u2^2+(586*u1-907)*u2-...]
[1520] Q=p_nf(u0^5+u1^5+u2^5,G,V,0);
2851262910*u2^3+30078832770*u2^2+(22194374760*u1-...
[1521] QR=p_true_nf(u0^5+u1^5+u2^5,G,V,0);
[2851262910*u2^3+30078832770*u2^2+...,35373600]
```

Asir : 消去法: $I_Y = I \cap K[Y]$ の生成系の計算

- ブロック順序によるグレブナー基底を使う.
  有理数体上で計算する場合には nd_gr_trace を
  $Home = 1$ で使う.
- の消去順序グレブナー基底 G から I_Y のグレブナー基底
  G_Y を取り出す.
  elimination (ライブラリ primdec_mod に定義されて
  いるがマニュアルにはない) を使う.

消去イデアルの計算

```
[1518] load("primdec_mod")$
[1664] B=[u2*u0-2*u2+3,(2*u1-1)*u0^2-u0-2*u2,2*u1^3+u2+4]$
[1665] V=[u0,u1,u2]$
[1666] G1=nd_gr_trace(B,V,1,1,[[0,2],[0,1]])$
[1667] elimination(G1,[u2]);
[8*u2^9+72*u2^8+292*u2^7-2036*u2^6-198*u2^5+20682*u2^4-...]
```

Asir：最小多項式の計算

`minipoly`(G, V, Ord, F, T) **(in** `gr`**)**

- **有理数体係数多項式環の** 0 **次元イデアル** $I = \langle G \rangle$ **および多項式** F **に対し,** $m(f) \in I$ **を満たすような 0 でない最小次数の多項式** $m(T)$ **を計算する.**
- G **は項順序** (V, Ord) **でのグレブナー基底,** F **は多項式,** T **は** V **に含まれない変数.**

katsura-7 での $u7$ の最小多項式の計算

```
[1518] load("katsura")$
[1522] B=katsura(7)$
[1523] V=[u0,u1,u2,u3,u4,u5,u6,u7]$
[1524] G=nd_gr_trace(B,V,1,1,0)$
[1525] minipoly(G,V,0,u7,t)$
[1526] deg(@@,t);
128
```

消去順序グレブナー基底計算と比較してみるとおもしろい.

Asir：0 次元イデアルの項順序変換

辞書式順序グレブナー基底を Buchberger アルゴリズムで直接計算するのは一般に効率が大変悪い.

⇒ 項順序変換が有効

tolex(G, V, Ord, W) **(in** gr**)**

0 次元イデアルの項順序 (V, Ord) **でのグレブナー基底** G **から, 項順序** (W, lex) **のグレブナー基底を計算する.**

次の例は, $katsura - 7$ **の辞書式順序グレブナー基底を項順序変換で計算したものである.**

項順序変換による辞書式順序グレブナー基底の計算

```
[1523] V=[u0,u1,u2,u3,u4,u5,u6,u7]$
[1524] G=nd_gr_trace(katsura(7),V,1,1,0)$
2.676sec + gc : 1.356sec(4.032sec)
[1525] G2=tolex(G,V,0,V)$
279.5sec + gc : 57.68sec(337.5sec)
```

Asir : イデアル演算

イデアルの共通部分, イデアル商, saturation を計算する関数は, ライブラリのあちこちで定義され使われている. (例 : `primdec`)

マニュアルに書かれていないので, 消去イデアル計算を用いてこれらを実装してみると, よい練習になるであろう.

練習問題

1. 以下の各項を行う方法を, Macaulay2, Asir それぞれについて調べよ.
   1. ファイルに結果を書き出す.
   2. 繰り返しを行う.
   3. 多項式の因数分解を行う.
2. $\mathbb{Q}[x, y, z]$ のイデアル
   $I = \langle x^2 + z, xy + y^2 + z, xz - y^3 - 2yz, y^4 + 3y^2z + z^2 \rangle$,
   $J = \langle x^2 + z, xy + y^2 + z, x^3 - yz \rangle$ の包含関係を調べよ.
3. $f_1 = 3x^2yz^2 + 3z + (-2x + 2)y + 2x$,
   $f_2 = 3yz^5 + (-xy^2 + 2)z - 2y^4 + 2y$,
   $f_3 = xy^3z^3 - 2yz^2 - z - 2y + x^2$ に対し,
   $I = \langle f_1, f_2, f_3 \rangle \subset \mathbb{Q}[x, y, z]$ とおく.
   1. $\dim I = 0$ を確かめよ.
   2. $\dim_{\mathbb{Q}} \mathbb{Q}[x, y, z]/I$ を求めよ.
   3. I の $x > y > z$ なる辞書式順序での簡約グレブナー基底が $\{g_0(z), x - g_1(z), y - g_2(z)\}$ という形であることを確かめよ.

練習問題 (つづき)

1. $f_1 = x^2 + y^2 + z^2 - 9$, $f_2 = 3x^2 - y^2 z$, $f_3 = x^2 z - 2y^2 + 2$ に対し, $f_1 = f_2 = f_3 = 0$ を満たす $(x, y, z) \in \mathbb{C}^3$ を全て求めよ.
2. $\alpha = 3^{\frac{1}{5}}, \beta = 5^{\frac{1}{3}}$ とする.
   1. $\alpha + \beta$ の $\mathbb{Q}$ 上の最小多項式を求めよ.
   2. $\frac{1}{\alpha+\beta}$ を α, β の有理数係数多項式で表せ.
3. (Asir でのプログラミング経験がある人向け) イデアルの共通部分, イデアル商, saturation の計算および radical メンバーシップを判定する関数を記述せよ.

参考書

- W. Adams, P. Loustaunau, An Introduction to Gröbner Bases. Graduate Studies in Mathematics, Vol. 3, AMS (1994).
- D. Cox, J. Little, D. O'Shea, Using Algebraic Geometry. GTM Vol. 185, Springer (2005).
- D. Eisenbud, D. Grayson, M. Stillman, B. Sturmfels (Eds.), Computations in Algebraic Geometry with Macaulay 2. Algorithms and Computation in Mathematics 8, Springer-Verlag (2000).
- G.-M. Greuel, G. Pfister, A Singular Introduction to Commutative Algebra. Springer (2007).
- M. Kreuzer, L. Robbiano, Computational Commutative Algebra 1. Springer (2008).