

# **noro\_grcrt**

---

noro\_grcrt User's Manual  
Edition 1.0  
Feb 2019

by Masayuki Noro

---



# 1 CRT パッケージ `norogrcrt.rr`

このマニュアルでは, `asir-contrib` パッケージに収録されている, CRT パッケージ '`norogrcrt.rr`' について解説する. このパッケージを使うには, まず '`norogrcrt.rr`' をロードする.

```
[1831] load("norogrcrt.rr");
```

このパッケージの関数を呼び出すには, 全て `norogrcrt.` を先頭につける. このマニュアルでは, 関連する組み込み関数についても解説する.

## 1.1 CRT によるグレブナー基底計算

### 1.1.1 `f4_chr`

`norogrcrt.f4_chr_dec(b,v,ord[|options])`  
CRT によりグレブナー基底候補を求める.

*return*      多項式リスト

*b*            多項式リスト

*v*            変数リスト

*ord*          項順序型

*options*      下の説明参照.

- 有限体上のグレブナー基底を `nd_f4` で計算し, それらを CRT で貼り合わせることで, 有理数体上のグレブナー基底候補を求める.
- 正当性のチェックは全く行わない.
- オプション `homo=1` を指定すると, 有限体上のグレブナー基底計算を斉次化付きて計算する.
- オプション `weight` を指定すると, `weight` 付きて計算する.
- オプション `proc` を指定すると, 有限体上のグレブナー基底計算を並列行う. `proc` で指定するのは `ox_asir` のストリーム番号のリストで, `norogrcrt.init_pproc` により生成する.

```
[2465] P=norogrcrt.init_pprocs([[0,4]]|nox=1)$
[2466] B=cyclic(8)$
[2467] V=[c0,c1,c2,c3,c4,c5,c6,c7]$
[2468] G=norogrcrt.f4_chr(B,V,0|proc=P)$
...
[Template,0,CRT,5.812,IR,11.388,#P,56]
28.74sec(55.12sec)
```

### 1.1.2 `norogrcrt.init_pprocs`

`norogrcrt.init_pprocs(list[|nox=0|1])`  
並列計算用サーバを起動する.

*return*      ストリーム番号のリスト

*list*            リスト

- `ox_asir` をまとめて起動する.
- 引数 *list* は  $[[host1, n1], [host2, n2], \dots]$  の形のリストで, *hosti* に *ni* 個の `ox_asir` を起動することを意味する. ここで *hosti*=0 の場合, `\codeasir` が実行されているマシンを意味する.
- オプション `nox=1` が指定された場合, サーバ用のメッセージウィンドウは表示されない. デバッグ中はこのオプションを指定しない方が安全である.

```
[2465] P=norogrcrt.init_pprocs([[0,2],["shinohara",2]]|nox=1);
[0,1,2,3]
```

## 1.2 CRT に関する関数

### 1.2.1 `norogrcrt.dp_chrem2`

`norogrcrt.dp_chrem2(g,M,gp,p,stat)`  
CRT による貼り合わせを実行する.

*return*        チェックに失敗した個数

*g*            分散多項式の配列

*M*            正整数

*gp*           分散多項式の配列

*p*            正整数

*stat*        配列

- $g \bmod M$ ,  $gp \bmod p$  を CRT で貼り合わせた  $g1$  を求め,  $g$  に上書きする.  $g1$  は  $g1 = g \bmod M$ ,  $g1 = gp \bmod p$  を満たす.
- $p$  は小さい整数(31bit 未満) でなければならぬ.
- $g[i]$ ,  $gp[i]$  は, サポートが一致した分散多項式でなければならない.  $gp[i]$ , および  $stat[i]=0,1$  の場合の  $g[i]$  はモニックでなければならない.
- $stat$  は長さが  $g$  の長さが等しい整数配列である. 置き換えられ, その後 CRT が実行される. CRT 後  $stat[i]=1$  となる.  $stat[i]=0,1$  のとき, 単に CRT が行われる.  $stat[i]=2$  のとき,  $g[i] \bmod p$  をモニック化したものと  $gp[i]$  が比較され, 等しくない場合に,  $g[i]$  は, 有理数変換前の  $g \bmod M$  に戻され, その後 CRT が実行される.
- 外部関数において, 整数-有理数変換が成功した場合に  $stat[i]=2$  とすることで, `dp_chrem2` を実行したあとに, 不適切な有理数への変換を検出することができる. 返される値は,  $stat[i]=2$  の場合に不適切な有理数変換を検出した個数である.

```
[2465] G=ltov([<<1,2>>+3*<<0,1>>,<<3,4>>+2*<<1,0>>])$
[2466] GM=ltov([<<1,2>>+5*<<0,1>>,<<3,4>>+7*<<1,0>>])$
[2467] Stat=vector(2)$
[2468] Mod=17$
[2469] P=19$
[2470] norogrcrt.dp_chrem2(G,Mod,GM,P,Stat);
0
[2471] G;
[ (1)*<<1,2>>+(309)*<<0,1>> (1)*<<3,4>>+(121)*<<1,0>> ]
[2472] Stat;
[ 1 1 ]
```

### 1.2.2 `norogrcrt.intdptoratdp`

`norogrcrt.intdptoratdp(f,M,B)`

整数-有理数変換を行う.

*return*      多項式

*f*            分散多項式

*M*

*B*            正整数

- *f* を法 *M* で整数-有理数変換を行った結果を返す. 一つでも変換できなかった係数がある場合には 0 を返す.
- *B* は  $M/2$  の平方根を超えない最大の整数で, `norogrcrt.calcb` で計算できる. 複数の多項式の係数を同一の *M*, *B* で変換するので, この値をあらかじめ与えるようにしてある.

[2495] `P=62884891$`

[2496] `A=<<1,2>>+(25632978)*<<0,1>>$`

[2497] `B=norogrcrt.calcb(P);`

5607

[2498] `norogrcrt.intdptoratdp(A,P,B);`

`(1)*<<1,2>>+(-5321/1234)*<<0,1>>`

# Index

(インデックスがありません)

(インデックスがありません)

## 簡単な目次

1	CRT パッケージ <code>norogrcrt.r</code> .....	1
	Index .....	4

## 目次

<b>1</b>	<b>CRT パッケージ <code>norogrcrt.rr</code> .....</b>	<b>1</b>
1.1	CRT によるグレブナー基底計算 .....	1
1.1.1	<code>f4_chr</code> .....	1
1.1.2	<code>norogrcrt.init_pprocs</code> .....	1
1.2	CRT に関する関数 .....	2
1.2.1	<code>norogrcrt.dp_chrem2</code> .....	2
1.2.2	<code>norogrcrt.intdptoratdp</code> .....	3
	<b>Index .....</b>	<b>4</b>



