

spec 簡易マニュアル (BL-3A)

2007.02.05

spec 関連部分:若林裕助

C-plot 関連部分:若林裕助, 小山一郎

1 困った時は

パルスがずれたと言ってきた

パルスモーターコントローラが信用できる時は No, 信用できない時は Yes を入れて下さい。

ビームモニターに計数されなくても測定を続けたい

chk_thresh=-1 して下さい。monitor の計数が chk_thresh より大きくなると、ビームが来ていないと判断して測定を止めます。

何故か波長を 1.54Å と思っているみたいだ

getE して下さい。

getE 等のモノクロ制御コマンドが見つからないといわれた

一度 quit して、fourc に入り直して下さい。

ct や scan など、様々なものが変だ

reset_stars をしてみてください。

2 起動

```
huber4:~$ three_a
```

実は中身は

```
perl /usr/local/lib/spec.d/specstars/specstars &  
fourc
```

です。一行目がモノクロ制御 PC との通信開始で、二行目が spec の起動です。面倒なのでコマンド一発で起動するようにしました。spec が起動したら、“u ps” とタイプして、perl のプロセス番号をノートにメモして置いてください。実験終了後に“kill プロセス番号” とタイプして、通信デーモンを終了してください。

一度 three_a で起動した後、quit して入りなおすときには

```
fourc
```

としてください。また、意図的に通信デーモンと fourc をばらばらに起動したい場合などは、上に書いた perl の二行を手で入力してももちろん構いません。spec に何らかのオプションをつけて起動したい場合などはそうせざるを得ないでしょう。

spec は前回終了時の情報を記憶しています。読み込んだマクロや格子定数などの記憶を保持しています。前回の記憶を無くさせたい時には、fourc に-f オプションをつけて起動して下さい。

fourc を起動すると、

Welcome to "fourc" Release 4.03.11

:

Opening "SPEC/CD/site.mac" at input nest level 1.
Opening "SPEC/CD/stage.mac" at input nest level 2.
Opening "SPEC/CD/monochro.mac" at input nest level 2.
Opening "SPEC/CD/slit.mac" at input nest level 2.
Opening "SPEC/CD/atten.mac" at input nest level 1.

1.FOURC>

と出て、コマンドを受け付けるようになります。
最初に起動したときは、startup というマクロを実行する必要があります。

1.FOURC> startup

で、startup マクロを実行してください。大部分の項目はただ return を押すだけで問題ありません。入れる必要があるところは、データファイルの名前程度です。

データファイル名入力例：

(newsample)

Title for scan headers (fourc)?

(newfile)

Data file (/dev/null)? LSM0327.dat <---ここにデータファイル名

Last scan number (0)? ~~~~~

Using "LSM0327.dat". Next scan is number 1.

格子定数などは、後からでも足りしますのでさほど重要ではありません。後から入れるときは、setlat コマンドを使います。もちろん、ここで入れてもいいです。

ここまでの、最初の準備は終わりました。

3 ビームライン調整

3.1 このセクションで使うコマンド、キーワード

gap_control	アンジュレータのギャップの状況を表示
_flg_gap_	ギャップ制御法のスイッチ。0 では gap 固定, 1 では特定の次数の光を使う, 2 では何次光を使うかも自動設定。 1,2 のときの gap 制御は moveE の瞬間に行われる。
harmonics	_flg_gap_ が 1 のとき, 何次光を使うかを保持する変数。
wh_gap	現在の gap の値を出力
mv_gap gap	挿入光源のギャップを gap(mm) に変更します。
reset_stars	モノクロとの通信を初期化します。
check detune_pls	モノクロの $\delta\theta$ をチューン/デチューンする。 引数を入れないとチューンし, 引数を入れるとその数のパルス数だけデチューンする。

3.2 手順

2007年2月現在、入射のたびにSGU #3は全開にされ、そのままパーミットされます。ですので、入射後にはmv_gapか、しかるべき_flg_gapの場合にはmoveEでgapを閉める必要があります。specからgapを変更できない場合、モノクロ制御PCのデスクトップにあるIDgap_cntrl.exeを利用します。

スキャンなどを途中で止めた後、

Socket error on "localhost:7388": Interrupted system call.

と表示された際には、reset_starsを実行してください。上のエラーメッセージはモノクロ制御PCとの通信が途中で遮断された場合の物で、通信を初期化するだけで問題は何もなくなります。一方、これをしてないと非常に悲惨な目にあります。例えば、getEを二度実行すると、一度目は変な数字が、二度目は正しい数字が出る、などが代表的な症状です。reset_starsは余分に実行して悪いことは起こりません。変かな？と思ったら試してみてください。

moveEで使いたいエネルギーを出し、checkでデチューンを必要に応じて行ってください。なお、 $\delta\theta$ は若干バックラッシュが大きいようです。問題があるときには、

mv_stars dth1 pls

で直接 $\delta\theta$ を動かすこともできます。相対値では動かさせませんので注意！

4 架台調整

4.1 このセクションで使うコマンド、キーワード

moveE energy	energykeVのX線にあわせてモノクロを設定する。
dscan motor_name start stop interval time	motor_nameのモーターをstartからstopまで、interval+1点測定します。一点あたりの露光時間はtime秒です。start, stopを、現在地からの相対値で表します。
chk_thresh	ビームモニターの値がこれより小さくなったら測定を中止するという閾値を表す。
CEN	直前のスキャンの半値幅の中心を表す変数。

4.2 手順

架台は4軸動くようになっています。y, z, ry, rzです。それぞれ、水平方向、鉛直方向、y軸を中心とした回転、z軸を中心とした回転、を表します。単位はmm及び度です。四軸側では架台のy,z,ry,rzいずれも0近辺でどのようなエネルギーでも光が通るはずですが。

ピークの形がきちんとしている時、便利な変数CENを使う事ができます。CENという変数には、直前のスキャンの半値幅の中心が代入されています。ピーク位置はplxMAXという変数に代入されています。これを使って、次のようにしてピークをリファインしていくことができます。

```
57.FOURC> dscan y -2 2 30 1
      :
      :
58.FOURC> mv y CEN
59.FOURC> dscan rz -2 2 30 1
      :
      :
```

ただし、CEN はピークトップの半分のカウント数までピークの両側で落ちていないと正しい値を返しません。ここは人間が判断してやる必要があります。

重要! getE 等のモノクロ制御コマンドが動かない場合の対処法

コマンドが見つからないといわれた場合：一度 quit して、fourc に入り直すと復帰します。

まだだめな場合：担当者を呼んで下さい。

5 スリット調整

自動化されましたので、架台調整が終わった状態で slit1;slit2[改行] と入力してください。重要なのは、下流イオンチェンバーでスキンのグラフ表示をするようになっていて、スリットのソフトリミットが充分広い可動範囲を許す事、及びピンホールを通った光が全部下流イオンチェンバーに入っていることです。現在の状態よりスリットを広げると言う操作は(一番最後以外)しませんので、途中で止まる事はありません。slit1;slit2 が終わった後は、スリットは開いた状態になっています。

以下は、手動で調整をやる時のやり方です。

5.1 このセクションで使うコマンド、キーワード

<code>ascan motor_name start stop interval time</code>	<code>motor_name</code> のモーターを <code>start</code> から <code>stop</code> まで、 <code>interval+1</code> 点測定します。一点あたりの露光時間は <code>time</code> 秒です。
<code>slt, slb, sll, slr, s2t, s2b, s2l, s2r</code>	それぞれ前スリット上, 下, 左, 右, 後スリット上, 下, 左, 右。左右は下流から見てのもの。

5.2 手順

流儀はいくつか有り得ますが、私が PF に来た時に習った手法を示します。

回折計にピンホールを入れたままの状態、後ろのイオンチェンバーをカウンターとして使います。後ろのイオンチェンバーの計数が、スリット開放時の半分になるようにスリットのブレードを閉じ、そこをそのブレードの 0 と定義します。

```
98.FOURC> ascan slt -1 1 40 1
      :
      :
99.FOURC> mv slt ***          <--ここに半分になった slt の座標を書く
100.FOURC> set slt 0
101.FOURC> mv slt -5
102.FOURC> ascan slb 1 -1 40 1
      :
      :
```

スリットの座標は、下流側から見て上から下がプラス、左から右がプラスです。スリットは閉じる側にバックラッシュを取る設定になっていますので、スキンの方向によって時間がかかる方、かからない方が存在します。8枚のブレード全部で数分の違いになると思いますが、これを気にしないなら適当にスキンをかければいいです。

この作業が終わったら、コリメータを外します。

6 センター出し, サンプルマウント

6.1 このセクションで使うコマンド, キーワード

config

6.2 手順

センター出しは省略します。サンプルマウントも好きなようにやるといいでしょう。chi と phi は皆さんネジをゆるめて適当にぐるぐる回しているでしょうから、設定を間違える可能性があります。chi=0にした時、th と phi が回転方向も含めて一致する様にセットして下さい。chi は下流側から見て反時計回りが正方向です。

phi の回転方向を逆にしたい場合があると思います。その場合、config で phi の Sign of user * dial を 1, あるいは-1(今の設定と逆) にして下さい。また、どちらの場合も chi の設定をいいかげんにしないで下さい。chi 軸が鉛直な状態が $\pm 90^\circ$, 水平な状態が $0^\circ, 180^\circ$ です。回折計の目盛りと spec の思っている角度を完璧に合わせて下さい。これがずれていると、phi 軸の回転軸が曲がってついている事になって、UB がうまく立たない事になります。

config はいいかげんな書き換えをしないで下さい。場合によっては修復に数日を要す可能性もありますし、嘘のデータを吐き出している事に気づかない可能性も 0 ではありません！わからなかったらスタッフを呼んで下さい。

7 軸立て

7.1 このセクションで使うコマンド, キーワード

setlat	格子定数入力
or0	UB の計算に使う第一のブラッグ位置設定
or1	UB の計算に使う第二のブラッグ位置設定
reflex_beg	reflex モード開始
reflex	reflex ファイルに書き込み
reflex_end	reflex モード終了
qdofilename	マクロファイル <i>filename</i> 実行
calcL	reflex 後の格子定数の計算
pa	or0,1, 格子定数の表示
set_lm <i>motor_name lower upper</i>	リミッタ設定

7.2 手順

setlat で格子定数を入れます。

Bragg を見つけます。

or0 h k l (h, k, lはそのブラッグの指数) と入力します。

もう一つ、独立な Bragg を見つけます。

or1 h k l (h, k, lはそのブラッグの指数) と入力します。

これで、一応は UB が計算されます。この場合、or0 と格子定数を信用して、or1 は参考につかわれるだけです。また、or0, or1 は方向だけを見て、長さを見ません。つまり、例えば or0 0 2 0 とした場合、その位置は $(0\eta 0)$ と設定されますが、 η が何になるかは格子定数と波長から計算され、or0 に合わせて格子定数を計算し直したりはしません。次に or1 1 2 0 とした場合について考えます。これによって a 軸の方向が決められるだけで、or1 1 2 0 を実行した場所が $(\xi 2\xi 0)$ になる保証はありません。こうしようとしても、格子定数を書き換えないと一般にはあいませんので、or0, or1 は格子定数を優先するという都合上、普通はどうしてもずれません。

いくつかの反射から格子定数, UB を最小二乗で計算するには reflex_beg reflex reflex_end という 3 つの一連のコマンドを使用します。まず reflex_beg コマンドを使用して各反射の四軸角を一時的なファイルに記録します。

```
20.FOURC> reflex_beg
Reflections file (reflex)?
```

デフォルトのファイル名は reflex です。次にこのファイルに四軸角を順に記録していきます。

```
21.FOURC> ここで 2 2 0 反射を見つける
```

```
22.FOURC> reflex 2 2 0
```

```
23.FOURC> ここで 2 0 2 反射を見つける
```

```
24.FOURC> reflex 2 0 2
```

```
25.FOURC> .... これらの手順を繰り返す
```

```
26.FOURC> reflex_end
Type "qdo reflex" to recalculate orientation matrix
```

reflex_end コマンドは入力された反射から reflex ファイルを実際に作成させるために用います。最後に reflex ファイルを実行します。

```
27.FOURC> qdo reflex
Opened command file 'reflex' at level 1.
```

これで UB が最小二乗 fit されます。格子定数は、calcL を実行後、pa を実行すると見ることができます。or0, or1 などを実行すると、UB の値を上書きしてしまうので、その場合には qdo reflex を再度実行して下さい。

8 コマンドリスト

使用頻度が高い、もしくは必ず使わなくてはならないコマンドとその説明をならべました。

8.1 モーター駆動

```
mv motor_name angle
```

motor_name の名前のモーターを *angle* に回す。角度は絶対値。スリットなどのモーターの場合、*angle* は角度ではなく位置を表す。

例 : mv tth 30

mvr *motor_name angle*

mv の角度設定を現在地からの相対値で表したものの。

例 : mvr th -1

umv *motor_name angle*

mv の移動中、現在地を表示するもの。

例 : umv phi 20

umvr *motor_name angle*

mvr の移動中、現在地を表示するもの。

例 : umvr chi -2

an *tth th*

tth と th を一度に動かす。角度指定は絶対値。

atten *attenuator*

アッテネーターを変える。atten *n* で、*n* 番のアッテネーターに設定する。1,2,3,4 番はアルミの 0.1,0.2,0.3,0.4mm が入っている。10 番は開放である。5 から 9 は保証されていない。

set *motor_name value*

motor_name のモーターの値の原点を変え、現在の値を *value* にする。

例 : set th 0

config

リミッタの設定などを行います。カーソルキーで変更したいものにカーソルを持っていき、書き換えます。書き換えを決定するには、w を押します。メニューから抜け出るには ctrl+c で抜けます。リミッタ以外の部分の書き換えは、通常は行わないでください。

wa

モーターの位置を表示します。二段に表示されますが、上段の数字が set で 0 点を設定した後の値で、通常使うのは上段です。

wh

wa の縮小版

8.2 モノクロメータ

`moveE energy`

energy keV の X 線にあわせてモノクロを設定する。

例 : `moveE 6.655`

`getE`

現在の X 線のエネルギー, 波長を表示する。

例 :

`getE`

E = 12.3985 keV, 1.00001 Angstroms

`check`

モノクロの $\delta\theta$ を最強位置に調整する。

`check pulse`

モノクロの $\delta\theta$ を最強位置から *pulse* パルスだけならず。普通は 50 パルス程度デチューンすれば充分。

8.3 四軸関連

`or0 h k l`

現在地を (*h k l*) の逆格子点であると設定し, UB の計算に用いる。

`or1 h k l`

`or0` と同じ。格子定数がわかっている場合で, UB を立てるのに最低二つは点がいるため。

`ca h k l`

逆格子の *h k l* の位置の角度を計算する。

例 :

```
7.FOURC> ca 1 0 0
```

```
Calculated Positions:
```

```
H K L = 1 0 0
```

```
Alpha = 0 Beta = 0 Azimuth = 90
```

```
Omega = 0 Lambda = 1.54
```

```
Two Theta      Theta      Chi      Phi
    60.0000     30.0000     0.0000     0.0000
```

`br h k l`

$h k l$ に回折計をあわせる。

ubr $h k l$

br の u 版。

setlat

startup の所に出てきた、格子定数を入力するコマンドです。これが既に入力してあることが四軸関連のコマンドが動作する前提条件となります。

reflex

3 つ以上の Bragg から UB と格子定数を決定する。詳細は section 軸立て を参照。

onsim offsim

時には四軸回折計を実際に動作はさせたくないが、特定の四軸角を反射として入力したい、ということがあります。この場合、spec をシミュレーションモードに切り替え、回折計を反射位置に移動させ、シミュレーションモードを解除する、という操作を行います。これを行うのが onsim offsim です。onsim で spec はシミュレーションモードになり、offsim でそれが解除されます。

8.4 スリット系

cvslit1 *center width*

入射側スリットの縦を、中心を *center* に取って、幅 *width* に設定する。

例：

```
13.FOURC> set slit 0
      :
      :
39.FOURC> cvslit1 0 1
```

スリットで半割を出して、半割の位置を set で 0 に設定しておく、このコマンドは第一引き数が 0 で使えて便利です。

cvslit2, chslit1, chslit2

最後の 1,2 が上流、下流のスリットを表し、二文字目の v,h が垂直、水平を表すとして、それぞれ対応した方向の cvslit1 と考えたもの。

8.5 測定

ct *time*

カウントして、結果を画面に表示する。*time* は省略可能。

uct

ct の u 版

ascan motor_name start stop interval time

motor_name のモーターを *start* から *stop* まで、*interval*+1 点測定します。一点あたりの露光時間は *time* 秒です。例：ascan th 30 32 30 1

dscan motor_name start stop interval time

ascan と同じですが、*start*, *stop* を、現在地からの相対値で表します。

th2th start stop interval time

$\theta - 2\theta$ スキャンを行います。角度は相対値で、 2θ で表します。

center width_th2th width_th width_chi width_phi interval time

自動ピークリファインマクロです。*width_** の二倍の幅にわたって測定し、 $\theta-2\theta$, ω , χ , ϕ スキャンを行います。3 周イテレーションを行って、センターを決定します。

hklscan Hstart Hstop Kstart Kstop Lstart Lstop interval time

逆格子空間でのスキャンを行います。意味は分かるかと思えます。*start*, *stop* は絶対値であり、相対値ではありません。

hscan Hstart Hstop interval time

現在 ($\xi\eta\zeta$) にいるとした場合、($\Xi\eta\zeta$) の Ξ をスキャンします。類似品に *kscan*, *lscan* があります。*Hstart* *Hstop* は絶対値です。

Escan start stop interval time (fixQ)

エネルギースキャンを行います。*fixQ* オプションをつけると、エネルギー変化に伴って 2θ と ω を動かして同じ逆空間の点のエネルギースキャンを行います。エネルギーの単位は keV です。

8.6 温度調整

温調を始める時には、`_LTC_is_on_ = 1` として下さい。温調装置の電源を切った時には、必ず `_LTC_is_on_ = 0` として下さい。`_LTC_is_on_ = 1` の状態だと、GPIB の応答が無いと `scan` ができなくなります。

ramp モードは CRYO-CON 専用モードです。ほかの温調ではそのようなコマンドは作っていません。

te *set_temp*

温度コントローラの `set temp` を *set_temp* にセットする。

showtemp

温度を読み取る。

rampON

ramp モードにする。温度を変えている途中でも、変化率の制御をすぐに始める (はず)。ramp モードに入ったら、後は `te` するだけで常に ramp モードのまま (のはず)。

rampRATE (*rate*)

ramp モードの場合, *rateK/min.* で温度を変化させる。引数が無い場合, 現在の *rate* を表示する。

rampQ

現在 ramp モードであるかどうか表示する

rampOFF

ramp モードを切る。

8.7 マクロ実行

qdo *file_name*

file_name のファイルの中身を実行する。

マクロファイルの作り方

マクロファイルは通常のテキストファイルであり、なんらかのエディタでこれを書く必要があります。テキストエディタはいくつかありますが、これらの使用法は省略します。UNIX の参考書を見て下さい。1999 年 4 月 26 日現在, BL-16A2 で普通に使われているエディタは vi と mule です。

マクロファイルはユーザーが必要なコマンドを自分で作ったり、一連のコマンドを順に自動で実行させるために使われます。一連のコマンドを順に自動で実行させるためには、ただ、そのコマンドを一行ずつ書き並べるだけで OK です。例えば θ 軸を $\pm 1\text{deg}$ の範囲でスキャンし、その半値幅の中心に移動させるためには次のようなマクロを作ります。

```
dscan th -1 1 40 1
mv th CEN
```

これを test.mac という名前で保存したとすると qdo test.mac と入力することによって実行させることができます。ここで CEN という変数は (正確にはマクロですが) 直前のスキャンの半値幅の中心が代入されています。ピーク位置は pl_xMAX という変数に代入されています。

新しいコマンドを作成するには次のようにします。

```
def refine_th '
dscan th -1 1 40 1
mv th CEN
'
```

このようなマクロファイルを作成し、これを実行することによって refine_th という新しいコマンドが作成されます。これらのコマンドにパラメータが必要であれば、以下のように \$1 \$2 などの変数を使用します。

```
def refine_th '
if ($# != 3) {
print "Usage: refine_th range interval time"
exit
}
```

```
dscan th -$1 $1 $2 $3
mv th CEN
,
```

\$# という変数はコマンドにいくつパラメータが入力されたかを表し、これが 3 でないときは使い方を表示しています。print や exit はそれぞれご想像通りの動作です。

マクロの中で、その時点での h,k,l の値を使いたい場合があります。その時には、wh を行った後に H,K,L の値を参照する事で、それらの値を使う事ができます。

8.8 その他

newfile

データファイルを新しく作り、次の測定からはその新しいファイルに書き出すようにする。

9 四軸関連の手法

四軸回折計に詳しい人にはあまり必要無いかも知れませんが、一応書いておきます。

四軸で、 h, k, l を決定すると、一つ軸が余分なので、何か条件をつけてやらないと回折計の角度が決まりません。そこで、いくつかの束縛条件の種類があります。spec では、setmode というコマンドで設定する、以下のモードがあります。

- 0.Omega Equals Zero
- 1.Omega Fixed
- 2.Zone or χ and ϕ Fixed
- 3.Phi Fixed or Three Circle
- 4.Azimuth Fixed
- 5.Alpha Fixed
- 6.Beta Fixed

* Fixed というモードの場合、freeze というコマンドで、Fix する角度を設定できます。 ϕ を回しながら同じ逆格子点を測定したい場合、次のようにします。

```
289.FOURC> setmode 3
290.FOURC> freeze 0
291.FOURC> br 1 0 0
292.FOURC> dscan th -.5 .5 30 1
293.FOURC> freeze 30
294.FOURC> br 1 0 0
295.FOURC> dscan th -.5 .5 30 1
296.FOURC> freeze 60
297.FOURC> br 1 0 0
298.FOURC> dscan th -.5 .5 30 1
:
:
```

10 C-PLOT

spec でデータテーキングされたものをプロット (あるいは印刷) するには C-PLOT というプログラムを使用します。起動するにはシェルのコマンドラインから cplot と入力します。測定データを C-PLOT に読みこませるには以下のようにします。

```
PLOT-> f1 scans.4
```

```
(Using public version)
```

```
This is scans.4, version 4.6, 12/11/95.
```

```
For instructions, type "h scans" in cplot.
```

```
Name of file (data)? xray/setup.1 ← ここでファイル名を入力します。
```

```
Using index file which shows 57 scans.
```

```
Print contents (NO)?
```

```
Normalize points (YES)?
```

```
Use #I (re)normalization value (NO)?
```

```
Calculate error bars (YES)?
```

```
Get MCA data if present (NO)?
```

```
Sort and merge points (YES)?
```

```
Rescale axis each time (NO)?
```

```
Retrieve by scan number (YES)?
```

```
Verbose mode (NO)?
```

```
Scans/options (1)
```

```
? 3 ← ここでスキャン番号を入力します。
```

```
Scan number 3
```

```
Thu Oct 1 00:21:26 1998
```

```
Counted to 1 second.
```

```
Two Theta H K L Epoch Seconds Counter 3 Counter 4 Counter 5 Counter 6
```

```
Counter 7 Monitor Detector
```

```
Column for x (1)?
```

```
Column for y (-1)?
```

```
Column for monitor normalization (-2)?
```

```
Found 51 points.
```

```
Returning 51 points to plot.
```

```
(Counts per monitor count.)
```

```
PLOT->
```

データをファイルから読み込んだら、フィルタを設定します。フィルタとはデータを各デバイスに表示させるプログラムで、例えば画面 (X11 デバイス) に表示させるモードにするには

```
PLOT-> zi x11
```

Started graphics filter "x11".

PLOT->

とします。プリントアウトのためには、`zi psfilter filename` としておいて、その後、UNIX のコマンドラインから `lpr filename` として下さい。

最後に各軸の表示範囲を設定して表示させます。各軸の表示範囲を自動で指定するには `np` コマンドを使います。グラフを表示させるには `zz` というコマンドを使用します。

PLOT-> np

PLOT-> zz

もし手動で各軸の表示範囲を指定したければ `ra` コマンドを用います。以下のようになります。

PLOT-> ra

```
xmin (-2.5) = -2          xmax (0.5) = 0
ymin (0) = -1           ymax (47.8824) = 50
```

PLOT->

グラフの装飾に関して、少しだけ言及します。

グラフのタイトル、軸のタイトルを表示するには `tx` コマンドを使います。`tx[リターン]` の後、聞かれたとおりに入力していきます。その装飾の結果は次に `zz` を行ったときに反映されます。

プロットのシンボルをかえるには、`sy` コマンドを用います。`sy L` は実線、`sy 0` は open circle など、いろいろなシンボルが用意されています。

11 マクロ

C-PLOT は非常に強力なマクロ機能を持っています。要するにコマンドラインに流し込むだけなのですが、旨く使うと、ピークの温度変化を測定したとき、大量のデータを一度にフィッティングして、そのパラメタの温度変化のグラフを出力するようなマクロを作ることができます。あるいは、測定したデータを一枚の紙に8つずつ、コメントつきで出力する、というものも私(若林)はよく使っています。ファイルの中身をコマンドラインに流し込むだけ、という仕様のため、空行に意味がある場合があります。マクロを見る場合には、それも意識する必要があります。

実際のところ、C-PLOT のマクロは汎用性を持たせるのが極めて困難であり、ちょこちょこ書き換えながら使うものです(どのファイルの何番目のスキャンを、何列目のデータを `x`, `y`, モニタの値として使って、軸のタイトルはこれこれ、コメントとしてこれを書き込み、この関数でフィッティング(初期値はこれこれのファイルを参照)しなさい、などという一続きのマクロを作ったら、引き数だらけで何のことだかさっぱりわからないものになり、さらには便利でもないでしょう)。

マクロの例(と、簡単な解説)を以下に添付します。これらのファイルは、`huber2/home/wakabays/` の下の `plotting_macros/`, `fitting_macros/` に置いています。フィッティングのためには、自分のホームディレクトリの下に `functions` というディレクトリを作って、そこに `/home/wakabays/functions` の中身をコピーする必要があります。

```
::::::::::::::::::
alignfig.cpm
```



```

cs
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0

eb 1

u rm par_all.tmp
u cat Par.*>par_all.tmp
f4 fitpar.4 par_all.tmp 5 2

wi -9 -2 6 4
np
ty 0 0 0
tx

Temperature
K
Intensity
arb. units

sy 0
zaltpb

#Width
ft 2
cs
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0

eb 1

f4 fitpar.4 par_all.tmp 5 4

wi -9 -8 6 4
np
ty 0 0 0
tx

Temperature
K
FWHM
rlu

sy 0
#ray 0 .01
zaltpb

#Peak position
ft 2
cs
2.25 2 0
2.25 2 0

```

```

2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0

eb 1

f4 fitpar.4 par_all.tmp 5 3

wi -9 -14 6 4
np
ty 0 0 0
tx

Temperature
K
Peak center
rlu

sy 0
#ray 1.25 1.26
zaltpb

#Back Ground
ft 2
cs
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0

eb 1

f4 fitpar.4 par_all.tmp 5 0

wi -.5 -2 6 4
np
ty 0 0 0
tx

Temperature
K
Back Ground
arb. units

sy 0
zaltpb

#Back Ground(slope)
ft 2
cs
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0
2.25 2 0

```

```
eb 1

f4 fitpar.4 par_all.tmp 5 1

wi -.5 -8 6 4
np
ty 0 0 0
tx

Temperature
K
Back(slope)
arb. units

sy 0
zaltpb

#end
zx
}
```