中級者向けの IRAF チュートリアル

光学赤外線天文連絡会

2010 年 8 月 1 日

この文章は、IRAF exercise パッケージに含まれているガイドブックを和訳したものである。1997 年頃に、国立天文台データ解析センターが主体となって和訳を完成させた。その後 2010 年に光学赤外線天文連絡会が改訂を行った。

第 I 部
準備

エクササイズパッケージをインストールしましょう。http://iraf.nao.ac.jp の画面左側に”tutorial” というリンクがあります。そこをクリックし、”iraf/misc” というリンクをさらにクリックします。そして “exer211.tar.Z” というファイルをダウンロードしましょう。

次に、ダウンロードしたファイルを login.cl ファイルのあるディレクトリに移動させます。そして、以下のコマンドを打ちましょう。

```
% tar -zxvf exer211.tar.Z
```

このテキストを通じて、% や cl> のプロンプトの直後に示されているコマンドはユーザーによって入力されるものとします。#記号はコメントを意味します。

第 II 部

Exercise 1. intro

この章では、IRAF の基本操作について演習を行ないます。このチュートリアルでは、皆さんが xgterm と ds9 が使える環境にあることを想定しています。それでは、xgterm から IRAF を起動しましょう。intro のディレクトリには演習に使用する画像が数枚あります。そこに移りましょう。

```
% ds9                  # ds9 を起動します。
% cl                   # IRAF を起動します。あなたは、IRAF login ディレクトリにいます。
cl> cd exercises/intro
# ここでは、exercises ディレクトリは IRAF login ディレクトリの下にあることを想定しています。
```

IRAF を動作させるために、正しい端末タイプ[*1] であることを確認します。これは簡単にチェックできますし、間違っている場合は再設定することが可能です。

```
cl> stty                  # 現在の端末タイプが表示される。
cl> stty xgtermc nlines=40 ncols=80
                          # 端末タイプは xgterm、文字数 80, 行数 40 に設定
                          [あなたが開いたウインドウの行数は設定値と異なるかも知れません]
```

最初に、この演習を通して使用する画像を解凍します。この操作には、タスク RFITS を使用します。

```
cl> unlearn rfits
cl> rfits fintro* "" junk old+
```

つぎに、 IRAF の基本操作を幾つかやってみましょう。

```
cl> diskspace         # 利用できるディスク容量の表示
cl> path              # カレントディレクトリの表示
cl> dir               # カレントディレクトリに含まれるファイルの表示
cl> dir *.fits        # 画像ファイルのみの表示
cl> dir uparm         # uparm ディレクトリのファイルの表示
cl> dir l+            # 属性も含めたディレクトリ情報の表示
cl> package           # ロードされているパッケージの表示
cl> help images       # パッケージ"images"のヘルプをみる。
cl> phelp imheader    # "imheader"のヘルプを表示。コマンドを実行するために
                        正しいパッケージがロードされていますか？
    q                 # ヘルプの終了
cl> lpar imh          # お、へましたかな?
```

[*1] 端末タイプは以下が使えます。xgterm,xterm,gterm,vt640,vt100

```
cl> lpar imhead        # 隠しパラメータがあることに注意して下さい
cl> unlearn imhead     # パラメータの初期化
cl> imhead im010       # ヘッダーの短い表示 - 何の情報が得られますか？
cl> imhead             # 画像の名前を指定しないと、全ての画像ファイルの情報を出力します。
cl> imhead im010 l+
cl> ^ | page           # いま、なにをやったかな?
cl> lpar imhead
cl> imhead im010 l+ u-
cl> epar imhead        # ヘッダー情報を全て表示するように修正して下さい。
    :q                 # 更新したパラメータを保存し eparam を終了
                         [crtl-C で終了するとパラメータは更新され
                          ません]
cl> lpar imhead        # パラメータは本当に更新されているかな？
cl> imhead im011 | page
cl> dir uparm          # uparm はどんなディレクトリでしょう？
                         "imheader"のパラメータファイルが見つけられますか？
cl> unlearn imhead     # デフォルトのパラメータに初期化
cl> dir uparm          # 何か変化はありますか？
cl> imhead im010 l+ | page
                       # キーワード "exptime" を探してみましょう。
cl> unlearn hselect
cl> lpar hselect
cl> hselect im*.fits $I,exptime yes
                       # hselect のヘルプの表示 - 特定のキーワード
                         を表示できます
cl> unlearn hedit
cl> lpar hedit
cl> hedit im*.fits notice "test data" add+
                       # ヘッダに、新しいキーワード (NOTICE) を加え
                         値 (test data) を設定
cl> imhead im010 l+    # キーワードが書き込まれていますか？
cl> hselect im*.fits $I,notice yes
```

パラメータについてどんなことを学びましたか？隠しパラメータと必ず入力しなければならないパラメータがありました。同じ結果を得るのにさまざまな方法で実現できることを理解しましたか？

カレントディレクトリ (exercises/intro) には、同じ天域を撮影した２枚の画像 (im010,im011) があります。しかし、両者の間には僅かな位置のずれがあります。一方の画像をシフトして２枚の画像の平均を求めましょう。

```
cl> dir im*.fits       # 2 枚画像があります
cl> imhead im*.fits    # 同じ天域か確認
```

```
cl> unlearn display
cl> epar display      # fill=yes にしましょう。
cl> lpar display      # タスク display のパラメータを表示
cl> display im010 1   # １枚目の画像をイメージディスプレイに表示
```

ちょっと、脇道にそれましょう。いままでに、ds9 を使用した経験がなければ、いい機会ですので勉強しましょう。まず、frame と zoom の使い方を憶えましょう。これらのオプションはマウスで選択できます。ちなみに、プルダウンメニューは、ウィンドゥの上部にあります。

各プルダウンメニュー開いて、項目に目を通して下さい。マウスの左ボタンをクリックすれば、プルダウンメニューを開くことができます。

ウィンドゥの左上には、カーソルのおおよその座標と、画素の推測値が表示されています (この情報が不用の場合は、プルダウンメニュー View の information panel をクリックすることで消去できます)。

右上の水色の線で囲まれたボックスは、”panner” box です。現在、画像のどの部分を見ているのかを知ることが出来ます。いまは、画像全体を見ています。

マウスの右ボタンを押したまま、画面を上下に動かすとコントラストが調整できます。また左右に動かすと明るさの調整が出来ます。お試しください。

画面の部分拡大（パンとズーム）はマウスの中央ボタンで制御できます。カーソルを適当な位置に移動して、中央ボタンをクリックして下さい。カーソルを移動してクリックすると、カーソルのある位置が画面の中央に移動（パン）します。使い方がマスターできるまで遊んで下さい。終えたら、最初の拡大率に戻しましょう。

この画像と im011 をブリンクさせてみましょう。2 番目の画像バッファに読み込みましょう。画像バッファには全部で 16 枚の画像を蓄えることが出来ます。

```
cl> display im011 2
```

それでは、frame メニューの中の”frame blink” ボタンを押して 2 枚の画像のブリンクをしてみましょう。”Blink interval” の項目で、好みに応じてブリンクの周期を変えることもできます。僅かなずれに気がつきましたか？それでは、”single” ボタンを押してブリンクを終了して下さい。

ds9 の使い勝手が良いと感じるまで練習して下さい。以上では、紹介していない機能もあります。

では、画像のずれを修正する問題に戻りましょう。

```
cl> display im010 1   # im010 の再表示
cl> unlearn imexamine
cl> lpar imexamine    # このタスクを利用して対話的にズレを求めます。
cl> imexamine im010 1 # カーソルを ds9 に移します - カーソルは
                      円形に変わり点滅しています。imexamine を起
                      動すると "interactive image cursor mode"に
                      なります。
```

a. **カーソルを明るめの星の上に移動し** "a" **を押して下さい。**
   xgterm **に以下の情報が表示されます。**
   x **座標** y **座標 等級　フラックス**

スカイの平均カウント ピークカウント 楕円率　楕円の方位角 FWHM
- 詳細はヘルプを御覧下さい。

b. "?" でカーソルヘルプが表示されます - xgterm 上で "q" を押すと終了します。

c. 両方のフレームから、3 つ程度明るい星を選んで位置を測定し、ズレの量を求めます。ズレの平均値を修正量とします。まずは im010 の画像から 星を 3 つ選んで "a" を押します - 偏りを生じないように選びます。

d. ds9 上で "d" を入力して下さい。すると、xgterm で次の画像に入力を促すメッセージが表示されます。画像 "im011"を入力して下さい。

この画像でも同じ星の位置を測定します。

e. ds9 上で "q" を入力すると "imexamine" が終了します。

2 枚目の画像をずらして 1 枚目と重ね合わせるために、移動量の平均値を求めましょう。求められた移動量は x=-0.53 と y=-1.68 です。皆さんの値は同じ位になりましたか？

```
cl> lpar help
cl> help imshift sec=description  # ヘルプの特定のセクション、
                                    description のみを表示する。
cl> help imshift sec=example
cl> unlearn imshift
cl> lpar imshift
cl> imshift im011 s011 -0.53 -1.68 # im011 を先ほど求めた量だけシフト
                                   させる
```

では、im010 と s011 をブリンクして、画像のシフトが正しく行なわれたか確認しましょう。ブリンクのやり方は憶えていますね？うまくできましたか？

では、簡単な画像演算をやってみましょう。im010 と s011 の平均を 2 つの方法で求めます。

```
cl> unlearn imsum imarith
```

a.
```
cl> lpar imsum
cl> imsum im010,s011 aver1 pixt=r calct=r option=average v+
    <"epar imsum"でパラメータを編集して":go"でも同じ結果が得られます。 >
```
[演算と出力される画像のタイプは REAL(実数型) であることに注意して下さい。]

b.
```
cl> lpar imarith
```

```
cl> imarith im010 + s011 aver2 pixt=r calct=r v+
    <"epar imarith"でパラメータを編集し ":go"でも結果は変わりません。>
cl> imarith aver2 / 2 aver2        # 演算結果は元のファイルに上書き
                                     されます。
```

c. [両方の演算結果は同じはずですが、実際はどうでしょう。]

```
cl> unlearn imstatistics
cl> lpar imstat
cl> imstat aver*.fits
```

隠しパラメータの値は、コマンドラインから入力できますが、パラメータファイルの中身は変更されません。どうすれば変更できますか？ IRAF は端末出力をファイルにリダイレクトする機能を持つほか、 あるタスクの出力を別のタスクの入力とするパイプ機能を持っています。さらにヒストリー機能があり、リコールすることも出来ます。

```
cl> history              # ヒストリーの表示
cl> ^                    # 直前に実行したコマンドを呼び出して実行。
                           数字を沿えると、その番号のコマンド実行できる。
cl> e lpar               # 直前の lpar コマンドを呼びだします。実行する前
                           にコマンドラインの編集をすることが出来ます。
                           カーソルの移動には、矢印の記号がついたキーを
                           押して下さい。文字は、カーソルの左に挿入され
                           ます。また、DEL キーで文字の消去が出来ます。
cl> e                    # 直前に実行したコマンドの呼びだし
                           上/下の矢印ついたキーでヒストリーを遡ること
                           が出来ます。
cl> history 100          # 直前に実行した 100 のコマンドの表示
cl> history 100 > hfile  # それをファイルに出力
cl> page hfile           # ヒストリーを書き込んだファイルの表示
cl> history 100 | page   # ファイル出力をしないで、ヒストリーを表示
```

”>” と ”|” の違いは何でしょう？

では、IRAF のプロットタスクを使ってみましょう。プロットタスクには対話的なものと、そうでないものの２つが存在します。

```
cl> help plot            # "plot" パッケージのヘルプを表示
cl> contour s011         # シフトした画像のコントアを表示
                           同画像のイメージを ds9 に表示して
                           プロットとイメージを比べて下さい。
```

```
cl> =gcur                  # 直前のプロットを呼び出して対話的に見ましょう。現在"interactive graphics cursor mode"になっています。カーソルがプロットウィンドゥ内にないと、コマンドは受け付けられません。
    :.help                 # カーソルオプションは IRAF の全プロットタスクにあります。q でヘルプを終了します。
```

Z - 明るい星のコントアにカーソルを当てて "Z" を入力

A - 拡大されたプロットに座標を表示

C - カーソル直下の座標値を表示

0 - 全体の再表示

caps 以外を叩けば終了します。

```
cl> surface s011           # 同じ画像の鳥瞰図の表示
cl> surface s011 >G meta   # 鳥瞰図をファイルに保存
```

```
cl> implot s011            # これは、対話的タスクで２次元画像の検査に大変便利です。"?"を入力して下さい。
                           カーソルコマンドとは、 : command の形式でで入力する小文字のコマンドです。カーソルコマンドはタスク毎に意味が異なります。
```

c - column plot, カーソル直下の列に沿ったプロファイルが表示されます。

```
    :l 100  - 100 行目をプロット
    :c 150 200  - 150-200 列の平均をプロット
    :.write meta           # ファイルに保存
```

< "implot"に慣れるまで練習して下さい。これから使う機会は多いと思いますよ。>

"Z"以外のコマンドでプロットを拡大することは出来ますか？

```
    q                      # 終了
```

```
cl> implot dev$wpix        # IRAF とともに配布されている画像 (M51) の表示
    :w world               # この画像のヘッダには RA と DEC の情報が書いてあります。
    :f %H                  # hh:mm:ss の書式に変換
    :.write meta
    c                      # 銀河の中心にカーソルを移動して"c" を入力すると列方向のプロットが得られる。
    :f %h                  # dd:mm:ss に変換
    :.write meta
```

```
    q                       # 終了
```

今までに幾つかのプロットをファイル meta に保存してきました。それらを見てみましょう。

```
cl> unlearn gkimosaic gkidir
cl> gkidir meta             # プロット出来るファイルのリスト
cl> lpar gkimosaic
cl> gkimos meta             # メタコードファィルのプロット
    q                       # プロットモードの終了
cl> gkimos dev$vdm.gki      # IRAF と共に配布したメタコードファイル
    <space bar>             # 次のプロットの進む
    q                       # プロットモードの終了
```

次の数分間は、IMEXAMINE の練習に使いましょう。強力なツールなので憶えておくと大変に有益です。

```
cl> display dev$wpix 1          # これは、IRAF と共に配布されている
                                  M51 の画像です。wpix と pix の画像は
                                  同じものですが、wpix のヘッダには
                                  を world coordinate の情報が書き込
                                  まれています 。


cl> imexamine

[カーソルを座標 224, 131 の星に当てる]
    ?                           # カーソルオプションの表示。"q"で終了。
    z                           # カーソル周辺の値の表示。
    m                           # ボックス内部の統計量を算出。

[銀河腕の中にある塊のあたりにカーソルを移動しましょう]
    s                           # 鳥瞰図の表示
    :epar                       # 直前に実行したコマンドのパラメータ
                                  を編集。コマンドは ds9 で入力し
                                  ますが、文字列はグラフィック画面に
                                  現れます。xgterm でのパラメータ編集
                                  は:q で終了します。
    l                           # 行プロット
    g                           # "interactive graphics cursor mode"
                                  の起動。カーソルがグラフィックウィン
                                  ドゥに移ります。
    :naverage 10                # 10 行平均をプロット。
    i                           # "interactive image cursor mode"へ復
```

```
                                    帰。カーソルが ds9 に移ります。
    [他のオプションもお試し下さい。]
    q                               # 終了。
```

ds9 ウインドウに表示した画像の上に座標グリッドを重ねてみましょう。

```
cl> display dev$wpix 1 xmag=0.8 ymag=0.8
cl> wcslab dev$wpix 1               # 何も変化がない場合は"gflush"を入力
                                      して下さい。
```

xmag, ymag のパラメータは、どんな働きをしたかお分かりですね？
さて、チュートリアル演習 1 を終了するにあたって、いらないファイルを消しておきましょう。

```
cl> dir
cl> delete hfile,meta               # 通常のファイルの消し方
cl> imdelete *.fits ver+            # 画像ファイルの消し方（確認付き）
                                      im*.fits は残そうとお思いでしょうが、
                                      このチュートリアルでは、以後使いま
                                      せん。

cl> logout
```

ウインドウをクローズする前に、まず IRAF を終了するように心がけて下さい。

参考文献

A Beginner's Guide to Using IRAF, by Jeannette Barnes, August 1993.

----------end of exercise----------

第 III 部

Exercise 2. ccd1

この章では IRAF による CCD データの一次処理—すなわちバイアスやダークの差し引き、及びフラット化などを行います。この章に出てくるデータは Kitt Peak 国立天文台で George Jacoby が撮った撮像データです。スペクトルの一次データ処理も撮像データの処理とほぼ同様ですが、フラット化は異なる方法の時もあります。このことは第 4 部で詳しく述べます。この章では、既に皆さんが第 2 章を学び IRAF の基礎的な動作ができることを前提とします。

この章の進め方には 2 通りの方法があります。まず第一には、同時に 2 つ以上のタスクを実行しない、丁寧なやり方、第二には初心者には実際の動作が分かりにくいかも知れませんが、一次処理にはお勧めのやり方です。

さあ xgterm(又はそれと同等のウィンドウ) から IRAF にログインしましょう。そしてあなたの IRAF のホームディレクトリーに行き、その次に exercises の phot というディレクトリーに行きましょう。

```
cl> cd
cl> cd exercises/phot
```

次に rfits でイメージを解凍しましょう。

```
cl> unlearn rfits
cl> rfits fm92* ‘‘’’ junk old+
```

すると今いるディレクトリーに m92*.fits というファイルがいくつかできましたね。これらはバイアスフレーム一つ、V および B バンドのフラットフレーム 2 つと 4 つの天体画像データになっています。

```
cl> dir
cl> imhead m92*.fits
```

バイアスフレームはノイズを下げるため 25 フレームの平均になっています。フラットも S/N を上げるため、それぞれ 5 フレームの平均になっています。ちなみにピクセルタイプは short つまり 16 ビットです。

方法 1

まず IMAGES パッケージの中の imcombine を使ってバイアスフレームを平均しましょう。フラットについても同様です。バッドピクセルや宇宙線を除去するために、このタスク中ではある種のピクセルの除去ができます。これらの手続きは m92*.fits では既に処理済ですので、次に進みましょう。

ここで 2 つのことを決めなくてはいけません。引き算のためのオーバースキャンの領域を決めることと、出力の画像サイズを決めることです。この CCD では、32 行がオーバースキャンになっていますが、それを全て使う必要はありません。オーバースキャン領域と、フレームの端にあるバッドピクセルのならんだ行や列は出力画像を作るためには取り除かねばなりません (この作業をトリミングといいます)。フラットフィールドの画像を例に

して implot を使ってパラメーターを決めましょう。

```
cl> display m92006 1
cl> implot m92006
```

オーバースキャンの行 (横行) を決めましょう。:l m n, :c m n, C (m,n は適用な数値) を使うといいでしょう。オーバースキャンの中でも、カウントが傾いている所は避けましょう。もしオーバースキャンの領域が決まったら、その行の平均をプロットしてみましょう。どの行にしましたか? 数値はメモしておきましょう。私は 335∼350 にしました。

さて次に、出力すべき行と列を決めましょう。いくつかの行や列をプロットして見て、どのような数値になっているのか見てみましょう。その次にプロットする領域を広げる必要があります。まず列を見て見ましょう。左端に何かありますか?右端はどうですか? オーバースキャンの領域やバッドピクセルの並んだ行や列は取り除きましょう。同じことを行についてもしましょう。

私は 1∼318 行と 2∼510 列を選びました。いいですか?

そうしたらオーバースキャンの部分を引いて、トリミングをしましょう。いくつかのパッケージをロードして、今決めたパラメーターを colbias というタスクに書き入れましょう。

```
cl> noao
cl> imred
cl> bias
cl> help colbias
cl> unlearn colbias       # unlearn は何をするタスクでしょうか？
cl> epar colbias          # 下記のように編集します
```

タスク lpar を colbias についてやれば以下のようなパラメーターがあらわれるでしょう。

```
cl> lpar colbias
        input = "m92*.fits"      Input images
       output = "%m%tr%92*.fits" Output images
        (bias = "[335:350,2:510]") Bias section
        (trim = "[1:318,2:510]") Trim section
      (median = no)              Use median instead of average in column bias?
 (interactive = yes)             Interactive?
    (function = "chebyshev")     Fitting function
       (order = 1)               Order of fitting function
  (low_reject = 3.)              Low sigma rejection factor
 (high_reject = 3.)              High sigma rejection factor
    (niterate = 1)               Number of rejection iterations
```

```
    (logfiles = "")             Log files
    (graphics = "stdgraph")     Graphics output device
      (cursor = "")             Graphics cursor input
        (mode = "ql")
```

オーバースキャンとトリミングの値は「画像の領域」として入っています。出力の画像の名前は分かりますか? 出力される実際の名前は以下のようにして見れます。タスク section が画像のテンプレートをテストします。

```
cl> sections %m%tr%92*.fits
```

さて colbias を実行する準備ができました。私達が入力したパラメーター通りにオーバースキャン部分を引いて、トリミングをしてくれることでしょう。タスクはインタラクティブに行われますので、まずはオーバースキャンの平均のプロットを見ることになります。時にはフィッティングパラメーターを変えることもありますが、今はこのままにしておきましょう。

```
cl> colbias
cl> dir
cl> imhead tr*.fits          # 新しい画像の大きさはいくつですか？
cl> implot tr92007           # フラットをチェックしましょう
```

次にそれぞれの画像からバイアスを引くことをしましょう。最も良い方法は imarith を使うことです。まず、処理を行う画像のリストを作りましょう。そしてこれを imarith の入力、出力としましょう。つまり入力画像は重ね書きされます。

```
cl> files tr*.fits > zlist
cl> edit zlist               # バイアスフレームは除きましょう
cl> imhead @zlist
cl> unlearn imarith
cl> epar imarith             # 下記のように編集します
```

lpar をするとこうなります。

```
cl> lpar imarith
     operand1 = "@zlist"        Operand image or numerical constant
           op = "-"             Operator
     operand2 = "tr92001"       Operand image or numerical constant
       result = "@zlist"        Resultant image
       (title = "")             Title for resultant image
     (divzero = 0.)             Replacement value for division by zero
     (hparams = "")             List of header parameters
```

```
   (pixtype = "")            Pixel type for resultant image
  (calctype = "")            Calculation data type
   (verbose = yes)           Print operations?
     (noact = no)            Print operations without performing them?
      (mode = "ql")
```

imarith を実行しましょう。

```
cl> imarith
```

次にダークの差し引きをせねばなりません。noao.imred.genric パッケージの darksub を使いましょう。この引き算をする前にフレームは露出時間を会わせておかねばなりません。したがって、露出時間はヘッダーの中に書くことが必要です。私達はダークフレームがありませんのでこのステップは飛ばしましょう。

さあ、最後にフラット化です。私達はフラットを 2 枚持っていますので、まずこれらをノーマライズ (画像の平均値を 1 にすること) する必要があります。imstatistics を使って 2 枚のフラットをノーマライズするための数値を決め、imarith によってノーマライズしたフラットを作りましょう。

```
cl> phelp imstatistics
cl> imstat tr92006,tr92007 fields=''image,mode''
cl> imarith tr92006 / 1313 Bflat
cl> imarith tr92007 / 1468 Vflat
cl> implot Bflat                  # Vflat もチェックしましょう
cl> display Bflat 1               # Vflat も見ましょう
```

各天体画像を適合するフラットで割りましょう。???はあなた自身で考えてください。なぜ imarith を 2 回する必要があるのでしょうか。

```
cl> imarith tr92010,tr92011 / ??? n92010,n92011
cl> imarith tr92014,tr92015 / ??? ?????????????
cl> imhead n92*.fits
```

こうしてできた最終画像は display や implot で確認しましょう。スカイは画像全体でフラットですか? ドームフラットは時としてフラット画像には適当でないこともあります。複数のスカイフラットが使われるべきです。ccdred パッケージの mkskyflat を参照してください。

さて、最終画像は消してしまいましょう。そして別の方法をトライしてみましょう。

```
cl> imdelete tr*.fits,n92*.fits,Bflat,Vflat ver+
cl> del zlist
cl> imhead m92*.fits
```

方法 2

まず、ディレクトリーにあるファイルをチェックしましょう。

```
cl> imhead m92*.fits
```

データを解析するため ccdred パッケージを使うことにしましょう。これはより能率的な方法です。パッケージがロードしてあるか確かめましょう。必要なパッケージは noao.imred の ccdred です。

```
cl> package
cl> ????                        # 打つべきコマンドは自分で考えましょう
```

ccdred は先程やった手順を同じ方法でおって行きます。ただし手順はたった一つのタスクにまとめられています。タスクを実行するには画像のヘッダーのある情報を使います。

ccdred はヘッダーにあるキーワードと数値を探します。もしヘッダーのキーワードや数値が適当でない名前ならば translation ファイルが使われます。必要なキーワードは 3 つ—IMAGETYP(object,flat,zero 等)EXPTIME(ダークを引くため)SUBSET(フィルターを決める)—です。

ccdlist を使えばパッケージがヘッダーを正しく認識しているかどうかを確かめられます。

```
cl> imheader m92015 l+      # imagettyp,extime,subset を探しましょう
cl> unlearn ccdred
cl> lpar ccdlist
cl> ccdlist m92*.fits
cl> ccdlist m92*.fits l+
```

画像ファイルは KPNO のものですから translation ファイルは正しくセットアップされているはずです。

```
cl> setinstrument           # translation ファイルを特定します
    ?
    direct
```

[こうすると自動的に ccdred のパラメーターをセットする epar に入ります。
verbose というパラメーターを yes にし、:q と入力しましょう。]

[すると今度は ccdproc の epar に入ります。
方法 1 と同様のパラメーターをセットし、:q を入力しましょう。]

```
cl> ccdlist m92*.fits
cl> type subsets
```

```
cl> dir ccddb$kpno          # kpno の translation files
cl> type ccddb$kpno/direct.dat
cl> lpar ccdred
```

ccdred パッケージはヘッダーの情報を認識したようです。ピルセルタイプも認識されていますね。入力画像のピクセルタイプは short ですが pixeltype パラメーターは計算途中と出力のピクセルタイプを表示しています。今は出力画像を入力画像に重ね書きしていますから、もし元のファイルを残しておきたいのなら backup パラメーターをセットしておきましょう。バイアスとフラットは zerocombine と flatcombine で作ることができます。いまはもうこれらの画像は作ってありますので飛ばしましょう。さて、ccdproc のパラメーターをセットしましょう。ここで biassec と trimsec パラメーターに注意しましょう。これらは image にセットされていますが、もしヘッダーの中の正しい値を認識しているのなら何もする必要はありません。しかし、注意深く見ると前計算した値はヘッダーのものとは違いますね。epar と打ってパラメーターをセットしましょう。

```
cl> imhead m92015 l+
cl> epar ccdproc
cl> lpar ccdproc
```

例えばこんな風です。

```
cl> lpar ccdproc
       images = "m92*.fits"     List of CCD images to correct
     (ccdtype = "object")       CCD image type to correct
   (max_cache = 0)              Maximum image caching memory (in Mbytes)
      (noproc = no)             List processing steps only?\n
      (fixpix = no)             Fix bad CCD lines and columns?
    (overscan = yes)            Apply overscan strip correction?
        (trim = yes)            Trim the image?
     (zerocor = yes)            Apply zero level correction?
     (darkcor = no)             Apply dark count correction?
     (flatcor = yes)            Apply flat field correction?
    (illumcor = no)             Apply illumination correction?
   (fringecor = no)             Apply fringe correction?
     (readcor = no)             Convert zero level image to readout correction?
     (scancor = no)             Convert flat field image to scan correction?\n
    (readaxis = "line")         Read out axis (column|line)
     (fixfile = "")             File describing the bad lines and columns
     (biassec = "[335:350,2:510]") Overscan strip image section
     (trimsec = "[1:318,2:510]") Trim data section
        (zero = "")             Zero level calibration image
```

```
        (dark = "")             Dark count calibration image
        (flat = "")             Flat field images
       (illum = "")             Illumination correction images
      (fringe = "")             Fringe correction images
  (minreplace = 1.)             Minimum flat field value
    (scantype = "shortscan")    Scan type (shortscan|longscan)
       (nscan = 1)              Number of short scan lines\n
 (interactive = yes)            Fit overscan interactively?
    (function = "chebyshev")    Fitting function
       (order = 1)              Number of polynomial terms or spline pieces
      (sample = "*")            Sample points to fit
    (naverage = 1)              Number of sample points to combine
    (niterate = 1)              Number of rejection iterations
  (low_reject = 3.)             Low sigma rejection factor
 (high_reject = 3.)             High sigma rejection factor

        (grow = 0.)             Rejection growing radius
        (mode = "ql")
```

zerocor と flatcor の画像は入力リストに入っていますから、特定する必要はありません。では、走らせてみましょう。

```
cl> ccdproc
cl> page logfile
cl> imhead m92*.fits
cl> imhead m92014 l+     # ヘッダーに計算手順が書かれていますね
```

できたファイルはこの後の exercise で使いますので、残しておきましょう。

参考文献

```
Type "help ccdred" to see a list of the tasks in this package.  Each task
has an online help page.  Also see the list of "Additional Help Topics".

A User's Guide to CCD Reductions with IRAF, by Phil Massey, June 1992.

There is a "demo" task in the IMRED.CCDRED.CCDTEST package that may be
instructive to run as well.
```

----------------------------------end of exercise---------------------------------

第 IV 部

Exercise 3. ccd2

ここでは、スペクトルデータを扱います。ccdred、twodspec.longslit というパッケージのなかにあるタスクを使用します。サンプルデータは Kitt Peak にある Coude Feed telescope で観測されたものです。

スペクトルデータの存在するディレクトリにいき、rfits でデータを IRAF フォーマットにします。ヘッダーをみれば、バイアス画像が 10 枚、スカイフラットが 3 枚、フラットフィールドが 4 枚、コンパリソン画像が 2 枚、そしてオブジェクト画像が数枚見つかるでしょう。

```
cl> cd                          # IRAF のホームディレクトリに移動
cl> cd exercises/spec           # spectral data のあるディレクトリに移動
cl> unlearn rfits               # rfits をパラメータファイルから切り離す
cl> help unlearn
cl> rfits fsp* "" junk old+     # rfits を実行
cl> dir                         # ファイルリストを表示
cl> imhead sp*.fits             # ヘッダーを表示
```

ccdred パッケージをロードしましょう (phelp ccdred で ccdred の場所がわかるので、下記の????に入る言葉を考えましょう)。

```
cl> package
cl> ????
```

続いて

```
cl> unlearn ccdred              # 説明省略
cl> imhead sp0020 l+            # sp0020 というデータの long header を表示
cl> ccdlist sp*.fits            # CCD データの解析処理情報を表示
```

setinstrument で観測装置のパラメータをセットします。今回の場合はイメージヘッダーのキーワードとその値はパッケージが認識できるようになっています。また、setinstrument は ccdproc というタスクのパラメータを設定します。

```
cl> dir ccddb$kpno              # 複数の CL ファイルが見つかります
cl> type ccddb$kpno/coude.cl    # ccdproc のパラメータ設定をするファイ
                                  ルが見れます
cl> setinstrument
    ?                           # Instrument ID を聞いてきますが、?で
                                  一覧が表示されます
    coude                       # クーデを選択
```

すると、ccdred のパラメータの一覧が表示されます。変更する箇所はないので :q を入力してください。次に ccdproc のパラメータの一覧が表示されます。パラメータ flatcor が no になっているのを確認してから :q を入力してください。

10 枚あったバイアス画像を平均して 1 枚にします。使用するタスクは zerocombine です。このタスクのパラメータは変更する必要はありません。各ピクセルの最大値と最小値は除いて平均操作を行います。

```
cl> lpar zerocombine        # zerocombine のパラメータをリストアップ
```

以下のようなパラメータが表示されるでしょう。

```
cl> lpar zerocombine
        input =                 List of zero level images to combine
     (output = "Zero")          Output zero level name
    (combine = "average")       Type of combine operation
     (reject = "minmax")        Type of rejection
    (ccdtype = "zero")          CCD image type to combine
    (process = no)              Process images before combining?
     (delete = no)              Delete input images after combining?
    (clobber = no)              Clobber existing output image?
      (scale = "none")          Image scaling
    (statsec = "")              Image section for computing statistics
       (nlow = 0)               minmax: Number of low pixels to reject
      (nhigh = 1)               minmax: Number of high pixels to reject
      (nkeep = 1)               Minimum to keep (pos) or maximum to reject (neg
      (mclip = yes)             Use median in sigma clipping algorithms?
     (lsigma = 3.)              Lower sigma clipping factor
     (hsigma = 3.)              Upper sigma clipping factor
    (rdnoise = "0.")            ccdclip: CCD readout noise (electrons)
       (gain = "1.")            ccdclip: CCD gain (electrons/DN)
     (snoise = "0.")            ccdclip: Sensitivity noise (fraction)
      (pclip = -0.5)            pclip: Percentile clipping parameter
      (blank = 0.)              Value if there are no pixels
       (mode = "ql")
```

zerocombine というタスクは ccdtype が zero の画像のみを処理しますので、入力ファイルは sp*.fits としても差し支えありません。

```
cl> zerocombine sp*.fits
```

結果を imstatistics で確認すると、処理後の平均バイアス画像のほうが未処理のバイアス画像よりノイズのばらつきが小さくなっていることがわかるでしょう。

```
cl> imstat sp000?.fits,sp0010,Zero # 各画像の統計量を表示
cl> display Zero 1 fill+           # 平均バイアス画像を画像表示
cl> implot Zero                    # 平均バイアス画像のプロファ
                                     イルを表示
```

次に、あるフラット画像を見ながら、overscan region と trim parameters を決定します。

```
cl> implot sp0014
```

overscan region は後ろの 32 列であることがわかります。ここでは、105 列から 130 列までとします。また、実際のデータが記録されている部分は 34 列から 74 列まで、1 行から 1022 行までとします。あなたの判断はいかがでしょう。

現在、イメージヘッダーの biassec(overscan region) と trimsec(trim parameters) の両パラメータは、上できめた値とは違っているはずです。imhead で確認してみましょう。

```
cl> imhead sp0020 l+
```

ccdproc のパラメータを epar で編集しましょう。変更するところは、biassec と trimsec を上記の設定値に、zerocor を yes に、すべてのイメージ画像が処理されるように ccdtype を空白にします。

```
cl> epar ccdproc
```

ccdproc のパラメータを表示すると以下のようになります。

```
cc> lpar ccdproc
       images = ""              List of CCD images to correct
     (ccdtype = "")             CCD image type to correct
   (max_cache = 0)              Maximum image caching memory (in Mbytes)
      (noproc = no)             List processing steps only?\n
      (fixpix = no)             Fix bad CCD lines and columns?
    (overscan = yes)            Apply overscan strip correction?
         (trim = yes)           Trim the image?
     (zerocor = yes)            Apply zero level correction?
     (darkcor = no)             Apply dark count correction?
     (flatcor = no)             Apply flat field correction?
    (illumcor = no)             Apply illumination correction?
   (fringecor = no)             Apply fringe correction?
     (readcor = no)             Convert zero level image to readout correction?
     (scancor = no)             Convert flat field image to scan correction?\n
    (readaxis = "line")         Read out axis (column|line)
     (fixfile = "")             File describing the bad lines and columns
     (biassec = "[105:130,1:1022]") Overscan strip image section
     (trimsec = "[34:74,1:1022]") Trim data section
```

```
        (zero = "Zero")         Zero level calibration image
        (dark = "")             Dark count calibration image
        (flat = "")             Flat field images
       (illum = "")             Illumination correction images
      (fringe = "")             Fringe correction images
  (minreplace = 1.)             Minimum flat field value
    (scantype = "shortscan")    Scan type (shortscan|longscan)
       (nscan = 1)              Number of short scan lines\n
 (interactive = yes)            Fit overscan interactively?
    (function = "chebyshev")    Fitting function
       (order = 1)              Number of polynomial terms or spline pieces
      (sample = "*")            Sample points to fit
    (naverage = 1)              Number of sample points to combine
    (niterate = 1)              Number of rejection iterations
  (low_reject = 3.)             Low sigma rejection factor
 (high_reject = 3.)             High sigma rejection factor
        (grow = 0.)             Rejection growing radius
        (mode = "ql")
```

次は、フラット画像を平均します。全部で 4 枚ありました。使うタスクは flatcombine です。フラット画像の様子をプロットしてみましょう。

```
cl> implot sp0014
    :i sp0015
    o
    :l 512
    :i sp0016
    o
    :l 512
    :i sp0017
    o
    :l 512
```

また、flatcombine のパラメータを以下のように設定します。

```
cl> epar flatcombine
```

```
> lpar flatcombine
        input =                 List of flat field images to combine
     (output = "Flat")          Output flat field root name
    (combine = "average")       Type of combine operation
```

```
 (reject = "crreject")     Type of rejection
(ccdtype = "flat")         CCD image type to combine
(process = yes)            Process images before combining?
(subsets = no)             Combine images by subset parameter?
 (delete = no)             Delete input images after combining?
(clobber = no)             Clobber existing output image?
  (scale = "mode")         Image scaling
(statsec = "")             Image section for computing statistics
   (nlow = 1)              minmax: Number of low pixels to reject
  (nhigh = 1)              minmax: Number of high pixels to reject
  (nkeep = 1)              Minimum to keep (pos) or maximum to reject (neg
  (mclip = yes)            Use median in sigma clipping algorithms?
 (lsigma = 3.)             Lower sigma clipping factor
 (hsigma = 3.)             Upper sigma clipping factor
(rdnoise = "rdnoise")      ccdclip: CCD readout noise (electrons)
   (gain = "gain")         ccdclip: CCD gain (electrons/DN)
 (snoise = "0.")           ccdclip: Sensitivity noise (fraction)
  (pclip = -0.5)           pclip: Percentile clipping parameter
  (blank = 1.)             Value if there are no pixels
   (mode = "ql")
```

flatcombine を行います。

```
cl> flatcombine sp*.fits
cl> imhead Flat
cl> implot Flat
```

今度はスカイフラット、オブジェクト画像、コンパリソン画像を処理します。まず、 overscan、trim、zero correct を行います。

```
cl> files sp*.fits > plist   # リストファイルの作成
cl> edit plist               # bias 画像と flat 画像を削除
cl> ccdlist @plist           # CCD データ処理の歴史を表示
cl> ccdproc @plist           # CCD データ処理
cl> ccdlist @plist           # notice reduction flags
cl> imhead sp0025 l+         # notice reduction history
```

フラット処理を行います。よく使われるのが、twodspec.longslit パッケージの中の response というタスクです。スリット方向 (行方向) にデータを平均して関数フィットをします。このフィットした関数で各列の値を除算します。その結果、応答 (response) 関数が求められます。response タスクは Interactive Curve Fitting (ICFIT) ルーチンを使用しています。

```
cl> twodspec
cl> longslit
```
波長分散は列方向です。`lpar longslit` **で** `dispaxis` **が** 2 **になっていることを確かめましょう。**
```
cl> unlearn response
cl> phelp response
cl> lpar response
cl> response Flat Flat Resp

     ?                      # look at cursor options
    :function cheby        # chebyshef 関数でフィット
    :order 5               #  5次関数
    f
    k
    h
    q
cl> implot Resp
```

次のステップはスカイフラット画像にスリット方向に依存性がないかを調べます。平均したスカイ画像を、先ほど求めた応答関数で除算します。

```
cl> lpar combine
cl> combine sp0011,sp0012,sp0013 Sky reject=crreject \
    scale=mode weight=!exptime rdnoise=rdnoise gain=gain
cl> implot Sky
cl> unlearn imarith
cl> imarith Sky / Resp SkyFlat
cl> display SkyFlat 1 fill+
cl> implot SkyFlat
```

スカイフラット画像に星が重なっているので、これを取り除く必要があります。そこで、illumination タスクで、スリット方向の illumination function を求めて、補正を行います。具体的には、波長方向 (列方向) にいくつかの部分に分解し (default は 5)、それぞれの部分で波長方向に足しあわせます。そして、スリット方向のプロファイルから直線フィットしてフラットによる成分を取り除き、illumination function を求めます。

```
cl> phelp illumination
cl> unlearn illumination
cl> lpar illum
```
デフォルトでは `nbin=5` **となっています。たぶんこのままで良いでしょう。**
```
cl> illum SkyFlat Illum
```

まず最初のビン (5 **ピクセル) を補正します。**

```
?                       # display cursor options
:function cheby
:order 2                # straight line
:high 1                 # set sigma rejection to 2
:grow 2                 # radius of pixel rejection
:niterate 2             # iterate the solution
f                       # fit new parameters
q                       # accept and go to next bin
```

f と q を繰り返し、全てのビンを補正します。

```
cl> implot Illum
cl> imarith SkyFlat / Illum test    # Illum タスクがうまく働いたか確かめましょう
cl> implot test                     # いろいろな行を見てみましょう
```

最終的に応答関数と illumination function の補正は以下のようになります。

```
cl> imarith Resp * Illum SuperFlat
cl> implot SuperFlat
    :l 200 300
    :i Resp
    o
    :l 200 300              # Resp ファイルと SuperFlat ファイルを比べましょう
cl> epar ccdproc             # flatcor を yes に、flat を SuperFlat にしましょう

cl> ccdproc @plist
cl> imhead sp0020 l+        #今まで解析したことが記録されています
```

以上で、一次リダクションは終了です。この後の解析は Exercise 5 に書かれています。

参考文献

Type "help ccdred" to see a list of the tasks in this package. Each task has an online help page. Also see the list of "Additional Help Topics".

There is online help for both RESPONSE and ILLUMINATION.

A User's Guide to CCD Reductions with IRAF, by Phil Massey, June 1992.

A User's Guide to Reducing Slit Spectra with IRAF, by Phil Massey, Frank Valdes, Jeannette Barnes, April 1992.

------------------------------end of exercise-----------------------------------

第 V 部

Exercise 4. photom

ここでは、器械等級の測定から、標準システムへ変換するまでの過程を、順を追って勉強します。使用するタスクは APPHOT と PHOTCAL パッケージをに含まれています。測光するデータは、exercise 2 で処理した M92 の画像です。これらの画像は、CCDPROC タスク を用いて処理されており、測光をする準備が整っています。

あなたは、xgterm から IRAF に login していると想定しています。xgterm がない場合は、それに代わるグラフィック端末エミュレータ (xterm, kterm など）でも構いません。さらに、aperture photometry の実行に際してはイメージディスプレイが必要になります。

まず、IRAF のホーム・ディレクトリに移り、さらに exercises/phot に移動しましょう。

```
cl> cd
cl> cd exercises/phot
```

ここのディレクトリには m92*.fits という名の画像が合計 4 つ（B,V フィルターで撮った画像が 2 枚づつ）存在します。これらの画像の測光を行ないましょう。まず、画像が既に処理済みであるか確認しましょう。確認の方法は憶えていますか？

```
cl> dir
cl> imhead m92*.fits
cl> imhead m92010 l+
```

APERTURE PHOTOMETRY

まず最初にすることは、イメージヘッダを整えることです。ヘッダ情報は、整約の測光の過程で幾つか使用されるため、それらが適切に準備されているのかチェックしなければなりません。ここでは、露出時間、フィルター、エアマスが必要になります。しかし、注意深くヘッダをみると、EXPTIME と FILTERS はありますが、エアマスに関するキーワードは見当たりません。

では、キーワード AIRMASS をヘッダに登録しましょう。その作業には、ASTUTIL パケージに含まれているタスク SETAIRMASS を使います。このタスクにより、観測時のエアマス[*2] 計算されます。エアマスの計算に必要な幾つかの情報は、すでにヘッダに書き込まれています。

```
cl> astutil
cl> phelp setairmass
cl> unlearn setairmass
cl> lpar setairmass
cl> setairmass m92*.fits update-  # 正しく計算できているか確認。ちな
                                    みに、画像は KPNO で撮像されました。
cl> setairmass m92*.fits
cl> imhead m92014 l+              # キーワードが付加されているか確認
```

[*2] ここで計算されるエアマスは effective airmass とよばれるもので、定義は help で知ることが出来ます。

```
cl> bye                                 # astutil パッケージの開放
```

これで、aperture photometry をする準備が整いました。DIGIPHOT をロードし、さらに APPHOT をロードしてください。

```
cl> digiphot
cl> apphot
```

測光をするためには、アパーチャ径を決める必要があります。アパーチャ径は星像の全半値幅 (FWHM) に依存します。FWHM は IMEXAMINE で測ることが出来ます。

```
cl> display m92010 1
cl> imexamine                           # カーソルを明るい星に移す
    r                                   # 表示される FWHM はフィットした
                                          Gaussian の値です。 他の幾つかの星
                                          についても求めて下さい
```

星の光を全て測るためには、アパーチャ径を FWHM の 4 から 5 倍に設定します。この場合、FWHM は約 3.3 ピクセルなので、適当なアパーチャ径は 14 ピクセルです。しかし、対象の星々は比較的暗いため、アパーチャ径を 10 ピクセルにしましょう。全てのフレームの測光を同じアパーチャで行なうために、残りのフレームについても FWHM が大体同じであることを確認しましょう。IMEXAMINE を継続して下さい。

```
    d                                   # m92011 をディスプレイに表示
    r                                   # 数個の星を測定
    d                                   # m92014 を表示
    r                                   # 数個の星を測定
    d                                   # m92015 を表示
    r                                   # 数個の星を測定
    q                                   # 終了 (quit)
```

タスク QPHOT[*3] を対話モードで起動し、1 枚目の画像に写っている幾つかの星の明るさを測りましょう。EPAR でパラメータを以下のように編集して下さい。

```
cl> unlearn apphot
cl> epar qphot
```

```
ap> lpar qphot
        image = ""              Input image
         cbox = 5.              Centering box width in pixels
      annulus =                 Inner radius of sky annulus in pixels
```

[*3] Quick aperture PHOTometry

```
     dannulus =                Width of the sky annulus in pixels
    apertures = "10"           List of photometry apertures
      (coords = "")            Coordinate list
      (output = "default")     Results file
    (plotfile = "")            Plot metacode file
        (zmag = 26.)           Zero point of magnitude scale
    (exposure = "exptime")     Exposure time image header keyword
     (airmass = "airmass")     Airmass image header keyword
      (filter = "filters")     Filter image header keyword
     (obstime = "ut")          Time of observation image header keyword
       (epadu = 14.)           Instrument gain
 (interactive = yes)           Interactive mode
    (radplots = no)            Plot the radial profiles in interactive mode
     (verbose = no)            Print messages
    (graphics = "stdgraph")    Graphics device
     (display = "stdimage")    Display device
   (icommands = "")            Image cursor: [x y wcs] key [cmd]
   (gcommands = "")            Graphics cursor: [x y wcs] key [cmd]
        (mode = "ql")
```

それでは QPHOT を起動します。最初にパラメータの入力を要求してきますが、値は後で対話的に設定し直すことが出来るので、ここで入力する値はさほど重要ではありません。とりあえず、 cbox=5, annulus=15, dannulus=10, apertures=10 に設定します。annulus は sky annulus の内径で dannulus はその幅 (外径ー内径) です。

撮影された領域のチャートは、docs/m92.ps に PostScript 形式で保存されています。

```
cl> display m92010 1
cl> qphot m92010
    ?                          # カーソルオプションの表示
    q
    i                          # 星 6 をポイント
```

[extraction box を 25 に設定（星の中心から半径 25 ピクセルまで表示することが出来ます）]

["v"を入力して対話的設定モードを選択]

[centering box を指定します。2.5 で良いでしょう。"return"を押すと 2.5 に設定されます。5 くらいに設定してもいいです]

[inner sky radius を決めます。15 で良いでしょう。"return"を押すと
15 に設定されます]

[outer sky radius を決めます。25 で良いでしょう。"return"を押すと
幅は 10 に設定されます]

[アパーチャ径を決定します。10 に設定して"return"を押して下さい]

```
q                              # 対話型設定モードの終了
```

[測光された情報 - center, sky, magnitude, error はスクリーンの上に
表示されます]

[では、全ての標準星の明るさをを 個々の星の radial plot を表示しなが
ら測定していきましょう]

```
w                              # 今、計算したパラメータを保存
:radplot yes                   # radial profile option を指定
```

[以下の星を順番に測定 6, 9, 5, 8, 7, 23, 13, 20, 18 - 星にカーソル
をあててスペースバーを押す]

```
q
```

測定した結果はファイルに保存されます。ファイルの名前は、測光した画像の名前に.mag.1 が付加された形になっています。ファイルを見てみましょう。TXDUMP コマンドを使うことで、ファイル中の項目を選択的に抜き出すことが出来ます。星の等級と等級エラーを抜きだしてグラフにプロットし、どのような傾向にあるか確認することも出来ます。

```
cl> dir
cl> page m92010.mag.1
cl> lpar txdump
cl> txdump m92010.mag.1 image,xcenter,ycenter,mag,msky,stdev yes
cl> txdump m92010.mag.1 mag,merr yes | graph point+
```

他の 3 フレームについては、今測定した星々の座標リストを入力することで、非対話的に QPHOT を実行することが出来ます。しかし、今測定した V バンドフレームと B バンドフレームで、撮像領域のずれが小さいことを確認しなければなりません。

```
cl> display m92014 1
```

[星 6 の座標値を読みとって、同星のリスト中の座標値と比較して下

さい。 - x 方向におおよそ 4.5 **ピクセルのずれがありますが、許容範囲内でしょう。**]

TXDUMP を使って座標リストを作りましょう。必要とあらば、LISTS パッケージの LINTRAN タスクで座標値をずらすことも出来ます。画像の上に座標リストをプロットして、測定する星を間違えていないか確認しましょう。

```
cl> txdump m92010.mag.1 xcenter,ycenter yes > coords
cl> type coords
cl> display m92010 1
cl> tvmark 1 coords mark=circle radii=10 color=205
```

QPHOT のパラメータを以下のように編集して下さい。B,V 画像間における座標のずれを補うため cbox を若干大きめに設定しました。

```
cl> epar qphot

ap> lpar qphot
        image = "m92011,m92014,m92015" Input image
         cbox = 7.              Centering box width in pixels
      annulus = 15.             Inner radius of sky annulus in pixels
     dannulus = 10.             Width of the sky annulus in pixels
    apertures = "10"            List of photometry apertures
      (coords = "coords")       Coordinate list
      (output = "default")      Results file
    (plotfile = "")             Plot metacode file
        (zmag = 26.)            Zero point of magnitude scale
    (exposure = "exptime")      Exposure time image header keyword
     (airmass = "airmass")      Airmass image header keyword
      (filter = "filters")      Filter image header keyword
     (obstime = "ut")           Time of observation image header keyword
       (epadu = 14.)            Instrument gain
 (interactive = no)             Interactive mode
    (radplots = no)             Plot the radial profiles in interactive mode
     (verbose = no)             Print messages
    (graphics = "stdgraph")     Graphics device
     (display = "stdimage")     Display device
   (icommands = "")             Image cursor: [x y wcs] key [cmd]
   (gcommands = "")             Graphics cursor: [x y wcs] key [cmd]
        (mode = "ql")
```

```
cl> qphot
cl> dir *.mag*                          # 画像ごとに mag ファイルがありますね?
cl> txdump m92*.mag.1 mag,merr yes | graph point+
cl> txdump *.mag* xcenter,ycenter,mag,merr,ifilter yes
```

測定結果を見てみましょう。Merr は等級測定のエラーです。幾つか大きな値が見られますが - 星が暗いことに起因しているのでしょうか？とりあえず、これらの値を使って先に進みましょう。気に入らない場合はもう一度やり直すことも出来ますから。フィルター番号と実際の対応は V=60、B=50 です。

PHOTOMETRIC CALIBRATIONS

では、PHOTOCAL パッケージのタスクで標準システムへの変換を行ないましょう。

```
cl> photcal
cl> unlearn photcal
```

まず最初に MKCATALOG の要求にしたがって標準星の情報を含むファイルをつくります。が、そのファイルは stds というファイルで存在します。もう一つ関連するファイル fstds.dat がありますが、これは stds の書式を記述したものです。時間が許せば MKCATALOG を使ってみて下さい。

```
cl> page stds
cl> page fstds.dat
```

QPHOT で測定した画像リストをファイルに記述します。V 等級や B-V を計算するため、バンド毎にグルーピングする必要があります。ファイルの中身は次のようにします。

```
cl> edit imsets
cl> type imsets
    M92 : m92010 m92014
    M92 : m92011 m92015
```

それでは、実際に標準星の観測情報を含むファイルを作製します。このタスクは。*.mag*ファイルから器械等級を抜きだし、imsets ファイルの記述に従ってグループ化します。フィルターは V=60 と B=50 で表されています。

```
cl> phelp mknobsfile
cl> epar mknobsfile
```

```
ph> lpar mknobsfile
    photfiles = "*.mag*"        The input list of APPHOT/DAOPHOT databases
    idfilters = "60,50"         The list of filter ids
       imsets = "imsets"        The input image set file
 observations = "sobs"          The output observations file
```

```
   (obsparams = "")            The input observing parameters file
  (obscolumns = "2,3,4")       The format of obsparams
   (minmagerr = 0.001)         The minimum error magnitude
      (shifts = "")            The input x and y coordinate shifts file
    (apercors = "")            The input aperture corrections file
    (aperture = 1)             The aperture number of the extracted magnitude
   (tolerance = 8.)            The tolerance in pixels for position matching
  (allfilters = yes)           Output only objects matched in all filters
     (verify = no)             Verify interactive user input ?
    (verbose = yes)            Print status, warning and error messages ?
       (mode = "ql")
```

```
cl> mknobsfile
```

sobs を見てみましょう。stds ファイルの星の名前と対応できるように、名前を書き換えてやる必要があります。我々は 6, 9, 5, 8, 7, 23, 13, 20, 18 の順番 9 つの星を測光しましたね。

```
cl> edit sobs
```

MKCONFIG で comfiguration file を作ります。このファイルには、器械等級を較正するための式と、stds, sobs に含まれるデータの記述があります。実は、このファイルもすでにディレクトリに存在します。名前は m92fig です。時間の許す限り MKCONFIG を試して下さい。

```
cl> phelp mkconfig
cl> type m92fig
```

ここまでで、 V, B-V の較正式を計算する準備が整いました。では、FITPARM で較正式を求めましょう。このタスクは対話形式なので多様なフィットが可能です。パラメータを以下のように編集して、実行して下さい。

```
cl> phelp fitparams
cl> epar fitparams
```

```
ph> lpar fitparams
 observations = "sobs"          List of observations files
     catalogs = "stds"          List of standard catalog files
       config = "m92fig"        Configuration file
   parameters = "calib"         Output parameters file
   (weighting = "uniform")      Weighting type (uniform,photometric,equations)
  (addscatter = yes)            Add a scatter term to the weights ?
   (tolerance = 3.0000000000000E-5) Fit convergence tolerance
     (maxiter = 15)             Maximum number of fit iterations
     (nreject = 0)              Number of rejection iterations
```

```
  (low_reject = 3.)             Low sigma rejection factor
 (high_reject = 3.)             High sigma rejection factor
        (grow = 0.)             Rejection growing radius
 (interactive = yes)            Solve fit interactively ?
     (logfile = "STDOUT")       Output log file
(log_unmatche = yes)            Log any unmatched stars ?
     (log_fit = no)             Log the fit parameters and statistics ?
 (log_results = no)             Log the results ?
      (catdir = "")             The standard star catalog directory
    (graphics = "stdgraph")     Output graphics device
      (cursor = "")             Graphics cursor input
        (mode = "ql")
```

```
cl> fitparams
    ?
```

[最初に V filter のフィットが表示されます。不適切なデータを除去し、再びフィットを試みることが出来ます。現在、カーソルモードで動作させています。残差が小さくなるようにしてください]

```
    :vshow                  # エラーとフィットの結果を表示
    :results                # フィットの結果をリストする

    q                       # 次のフィット（B フィルター）に移る
```

[V と同様、残差を小さくして下さい]

```
    q                       # 気に入らなければもう一度フィットをやり直すことも出来ます。終了する前には最後のフィット結果を保存することを忘れずに。
cl> page calib
```

計算で求められた係数値と初期値を比較してみて下さい。タスク FITPARAMS はフィッティングの過程で係数の値を操作することが出来ます。標準星をたくさん測定していれば、全ての係数を一度に解くことも可能ですが、我々の扱っているデータは、エアマスのレンジが狭いため減光係数を定数にする必要があります。では、最後に sobs に登録された標準星の器械等級を較正しましょう。これは、タスク INVERTFIT で行ないます。パラメータを以下のように変更して下さい。

```
cl> epar invertfit
```

```
cl> lpar invertfit
 observations = "sobs"          List of observations files
       config = "m92fig"        Configuration file
   parameters = "calib"         Fitted parameters file
        calib = "results"       Output calibrated standard indices file
    (catalogs = "stds")         List of standard catalog files
      (errors = "obserrors")    Error computation type (undefined,obserrors,equ
     (objects = "all")          Objects to be fit (all,program,standards)
       (print = "")             Optional list of variables to print
      (format = "")             Optional output format string

      (append = no)             Append output to an existing file ?
      (catdir = "")             The standard star catalog directory
        (mode = "ql")

          cl> invertfit
          cl> page results
```

測光の最初に戻り、大きな aperture で測光を試み、今求めた結果と差がでるか確認する演習もやってみて下さい。QPHOT は一度の実行で、複数の aperture サイズによる測光が可能です。また、MKOBSFILE は aperture サイズの指定が出来ます。

参考文献

A User's Guide to Stellar CCD Photometry with IRAF, by Philip Massey and
Lindsey E. Davis, April 1992.

A User's Guide to the IRAF Apphot Package, by Lindsey Elspeth Davis, May 1989.

Specifications for the Aperture Photometry Package, Lindsey Davis,
October 1987.

Online help is available for all tasks. Especially see "phelp config"
and "phelp pcintro" in the PHOTCAL package.

----------end of exercise----------

第 VI 部

Exercise 5. spectra

Exersice 3 に引き続き、スペクトルデータの解析を進めます。ここでは、２次元スペクトルデータを１次元にし、波長キャリブレーションをし、星のスペクトル情報を得るまで解析を進めます。

早速、スペクトルデータのあるディレクトリに移動しましょう。

```
cl> cd
cl> cd exercises/spec
```

使用するデータは sp0018 から sp0027 までです。ヘッダーを見てみましょう。

```
cl> dir
cl> imhead sp*.fits
cl> imhead sp0018 l+    # 何に注目しましょうか?
cl> imhead sp*.fits
sp0018.fits[41,1022][real]: comp 6707 start of night
sp0020.fits[41,1022][real]: DHCep 6707
sp0021.fits[41,1022][real]: DHCep 6707
sp0022.fits[41,1022][real]: DHCep 6707
sp0023.fits[41,1022][real]: AHCep 6707
sp0024.fits[41,1022][real]: AHCep 6707
sp0025.fits[41,1022][real]: AHCep 6707
sp0027.fits[41,1022][real]: comp 6707
```

スペクトルの抽出

まず、一次元のスペクトルデータにします。これは、オブジェクトとコンパリゾン画像について行います。使用するタスクは noao.twodspec.longslit のパッケージにあります。すぐにロードしましょう。

```
cl> imred
cl> kpnoslit
```

次に、apall のパラメータを決定します。apall は一次元データの抽出に使うタスクです。星のスペクトルの波長分散方向にどのくらいの幅をとるのかを決めます。大抵、FWFM(Full Width Full Maximum) 程度の幅をとります。implot でスペクトルをスリット方向に表示して、幅を見積もります。

```
cl> implot sp0020       # C コマンドを使って、スペクトルの FWHM を
                          測りましょう。スペクトルのカウントが 50
                          カウントくらいの場所を見つけましょう。
                          カーソルをスペクトルの端に置きます。
                          そして、"C"をタイプします。次に
```

```
                                反対側にカーソルを移動させ"C"を打ち
                                ます。2 度くらい測定したほうがいいでしょう。

        [別の行でも測定して、FWHM の測定精度や、スペクトルの歪みを測りましょう]

        o                       # 次のプロットを上書きします
        :l 50                   # 50 行目
        o
        :l 1000

        c                       # スペクトルの中央にカーソルを持っていき
                                  "c"をタイプします。こうすると、カーソルの
                                  ある列の断面をプロットします。こうして、
                                  星のスペクトルを見ることができます。

        :i sp0025               # 別のスペクトルを見てみましょう
        :l 511

        [納得のいく測定ができたら、implot タスクから抜けます]

        q
```

apall のパラメータを編集します。パラメータの値は以下のリストを参考にしてください。

```
        cl> unlearn apall
        cl> epar apall          # パラメーターを下記のように設定します。
        :q                      # パラメーターを保存します。

cl> lpar apall | page
        input = "@extract"      List of input images
        nfind = 1               Number of apertures to be found automatically
      (output = "")             List of output spectra
      (format = "multispec")    Extracted spectra format
  (references = "")             List of aperture reference images
    (profiles = "")             List of aperture profile images\n
 (interactive = yes)            Run task interactively?
        (find = yes)            Find apertures?
    (recenter = yes)            Recenter apertures?
      (resize = yes)            Resize apertures?
        (edit = yes)            Edit apertures?
```

```
      (trace = yes)           Trace apertures?
    (fittrace = yes)          Fit the traced points interactively?
     (extract = yes)          Extract spectra?
      (extras = no)           Extract sky, sigma, etc.?
      (review = yes)          Review extractions?\n
        (line = INDEF)        Dispersion line
        (nsum = 10)           Number of dispersion lines to sum\n\n# DEFAULT
    (dispaxis = 2)            Dispersion axis (1=along lines, 2=along columns
       (lower = -5.)          Lower aperture limit relative to center
       (upper = 5.)           Upper aperture limit relative to center
   (apidtable = "")           Aperture ID table (optional)\n\n# DEFAULT BACKG
  (b_function = "chebyshev")  Background function
     (b_order = 2)            Background function order
    (b_sample = "-10:-6,6:10") Background sample regions
  (b_naverage = -10)          Background average or median
  (b_niterate = 0)            Background rejection iterations
(b_low_reject = 3.)           Background lower rejection sigma
(b_high_rejec = 3.)           Background upper rejection sigma
      (b_grow = 0.)           Background rejection growing radius\n\n# APERTU
       (width = 6.)           Profile centering width
      (radius = 10.)          Profile centering radius
   (threshold = 0.)           Detection threshold for profile centering\n\n#
      (minsep = 5.)           Minimum separation between spectra
      (maxsep = 1000.)        Maximum separation between spectra
       (order = "increasing") Order of apertures\n\n# RECENTERING PARAMETERS\n
   (apertures = "")           Select apertures
      (npeaks = INDEF)        Select brightest peaks
       (shift = yes)          Use average shift instead of recentering?\n\n#
      (llimit = INDEF)        Lower aperture limit relative to center
      (ulimit = INDEF)        Upper aperture limit relative to center
      (ylevel = 0.1)          Fraction of peak or intensity for automatic wid
        (peak = yes)          Is ylevel a fraction of the peak?
         (bkg = yes)          Subtract background in automatic width?
      (r_grow = 0.)           Grow limits by this factor
   (avglimits = no)           Average limits over all apertures?\n\n# TRACING
      (t_nsum = 10)           Number of dispersion lines to sum
      (t_step = 10)           Tracing step
     (t_nlost = 3)            Number of consecutive times profile is lost bef
  (t_function = "legendre")   Trace fitting function
```

```
     (t_order = 2)              Trace fitting function order
    (t_sample = "*")            Trace sample regions
  (t_naverage = 1)              Trace average or median
  (t_niterate = 0)              Trace rejection iterations
(t_low_reject = 3.)             Trace lower rejection sigma
(t_high_rejec = 3.)             Trace upper rejection sigma
      (t_grow = 0.)             Trace rejection growing radius\n\n# EXTRACTION
  (background = "fit")          Background to subtract
      (skybox = 1)              Box car smoothing length for sky
     (weights = "none")         Extraction weights (none|variance)
        (pfit = "fit1d")        Profile fitting type (fit1d|fit2d)
       (clean = no)             Detect and replace bad pixels?
  (saturation = INDEF)          Saturation level
   (readnoise = "0.")           Read out noise sigma (photons)
        (gain = "1.")           Photon gain (photons/data number)
      (lsigma = 4.)             Lower rejection threshold
      (usigma = 4.)             Upper rejection threshold
     (nsubaps = 1)              Number of subapertures per aperture
        (mode = "ql")
```

ずいぶんと長いパラメータリストですが、変更点は以下の通りです。extras を off に、b_order=2(バックグランドを直線フィット) に、b_naver=-10 に、width=6(星のプロファイルを 6 ピクセルとした) に、nfind=1 に、そして background=fit にするだけです。

さて、apall の入力ファイルのリストを作りましょう。

```
cl> files sp002?.fits > extract
cl> edit extract        # sp0020 から 25 だけを残しましょう。
cl> imhead @extract     # リストを確かめましょう。
```

準備は整いました。

```
cl> apall
```

質問にはすべて yes **と答えましょう**

最初の星 (sp0020.fits) **のプロファイルが表示されます。**
中心とその幅が記されているのに注意してください。

```
?       # オプションを表示します。
b       # バックグランドをフィットさせるモードに入ります。
```

デフォルトのバックグランドが表示されます。
問題はありませんが気になる人は、以下のように変更しましょう。

```
?       # オプションを表示します。
t       # バックグランド領域を初期化します。
s       # 新たに設定するバックグランド領域の左端を決めます。
s       # 右側を決めます。
s       # 二つ目のバックグランド領域の左端を決めます。
s       # 右側を決めます。
f       # 新しいバックグランドをフィットさせます。
:sample # 使用したスカイ領域を表示します。
```

気にならない人は

```
q       # バックグランドモードから抜けます。
```

次は、波長方向にデータをトレースします。

```
q
```

質問にすべて yes と答えます。
すると、スリット方向のプロファイルの中心と
スリット位置ごとのピクセル値がプロットされます。
フィッティングパラメータがプロット図の上に表示されます。
フィットを以下のようにして調整します。

```
:func spline3
:order 3
:niter 1
f
```

満足したら、

```
q       # トレースモードから抜け、抽出モードに移ります。
```

再び、質問にすべて yes と答えます。
すると、一次元のスペクトルが表示されます。しかし、

宇宙線は除去されていません。
次のスペクトルデータに移行します。

```
q       # 次のスペクトルに移ります。
```

そして、同様の処理を続けます。

ここで、ヘッダーを見てみましょう。また、処理がすべて database ディレクトリに一つ一つ保存されています。

```
cl> imhead *.ms*                # 抽出したスペクトルが表示されます。
cl> imhead sp0025.ms 1+ | page  # ヘッダーをすべて表示します。
cl> dir database
cl> page database/apsp0021
```

次は、コンパリゾンランプのスペクトルを一次元にします。この処理は星のスペクトルとは少し違います。なぜなら、コンパリゾンランプには連続光がなく線スペクトルだからです。したがって、星のようにスペクトル波長方向をトレースすることはできません。従って、とりあえず一つの星のトレースの結果を用います。

使用するタスクは apsum です。パラメータは以下のようにしてください。

```
cl> unlearn apsum
cl> epar apsum


cl> lpar apsum


      input = "sp0018,sp0027" List of input images
    (output = "")             List of output spectra
    (format = "multispec")    Extracted spectra format
  (refinput = "sp0018,sp0027" List of input images
    (output = "")             List of output spectra
    (format = "multispec")    Extracted spectra format
 (references = "sp0025")      List of aperture reference images
   (profiles = "")            List of aperture profile images\n
(interactive = no)            Run task interactively?
       (find = no)            Find apertures?
   (recenter = no)            Recenter apertures?
     (resize = no)            Resize apertures?
       (edit = no)            Edit apertures?
      (trace = no)            Trace apertures?
   (fittrace = no)            Fit the traced points interactively?
    (extract = yes)           Extract apertures?
     (extras = no)            Extract sky, sigma, etc.?
```

```
    (review = yes)            Review extractions?\n
      (line = INDEF)          Dispersion line
      (nsum = 10)             Number of dispersion lines to sum\n
(background = "none")         Background to subtract (none|average|fit)
   (weights = "none")         Extraction weights (none|variance)
      (pfit = "fit1d")        Profile fitting type (fit1d|fit2d)
     (clean = no)             Detect and replace bad pixels?
    (skybox = 1)              Box car smoothing length for sky
(saturation = INDEF)          Saturation level
 (readnoise = "0.")           Read out noise sigma (photons)
      (gain = "1.")           Photon gain (photons/data number)
    (lsigma = 4.)             Lower rejection threshold
    (usigma = 4.)             Upper rejection threshold
   (nsubaps = 1)              Number of subapertures per aperture
      (mode = "ql")
```

```
cl> apsum
cl> imhead *.ms.fits
cl> implot sp0018.ms    # 抽出したスペクトルを見てみましょう。
    :i sp0027.ms
    l
    q
```

波長較正

コンパリゾンの波長較正を行います。スペクトル線のチャートは docs というディレクトリの下にあります。ファイル名は spec.eps で

```
cl>!lpr spec.eps
```

とすればプリントできます。使用するタスクは identify と reidentify です。スペクトルの線幅を調べます。

```
cl> unlearn identify reidentify
cl> implot sp0027.ms
```

さて、identify を走らせます。線幅はパラメータはデフォルトの fwidth=4 とします。また、パラメータ coordlist は linelists$thorium.dat にしてください。identify タスクが終了すると、結果が database に保存されます。

```
cl> lpar identify
cl> epar identify
cl> identify sp0027.ms
```

```
?               # オプションを表示します。
w               # ウインドウカーソルモードに移ります。
?               # オプションを表示します。
w
f               # データを反転させ、左側を短波長にします。
```

スペクトル線の強度が強いほうから
カーソルをスペクトル線にあわせて"m"と入力する。
その後、プリントアウトしたスペクトル線のチャートをみて波長を入力する。
"m"と入力したときにビープ音がなったときには、まず"w"と入力、
そしてカーソルを目的のスペクトル線の左側近傍に持ってきて"e"と入力、
それから、今度はカーソルを右側近傍に持ってきて"e"と入力する。
そうすると、表示が拡大される。
そこで、改めて"m"と入力し波長を打ち込む。
拡大表示を解除するには"w"のあとに"a"で回復する。
５つくらいスペクトル線を登録したら、次の手順に進む。

```
f               # 線のフィットを行うため、ICFIT タスクに入ります。
?               # オプションを表示します。
:func cheby     # フィット関数を変更します。
:order 3        # 二次関数にします。
f               # 再フィットします。
q               # IDENTIFY タスクに戻ります。
w               # スペクトルが表示されなければ、再表示します。
a
l               # thorium.dat ファイルから他の線を探します。
f               # 線のフィットを行うため、ICFIT タスクに入ります。
:nite 2         # フィットにあわない線を取り除くため、イテレーションをします。
f               # 再フィットします。
d               # フィットにあわない線にカーソルを持っていき、"d"を押します。
f               # 再フィットします。
q               # IDENTIFY タスクに戻ります。
d               # また線を消します。ICFIT タスクの中で RMS を 0.01 A くらい
                  までに抑えたいですね。
f               # 再フィットします。
l               # 一次関数でフィットしていない部分を見ましょう。
q               # IDENTIFY コマンドに戻ります。
q               # フィットがうまくいったら IDENTIFY から抜けます。
                  結果を保存することを忘れずに。
```

```
cl> dir database        # 結果は idsp0027.ms というファイルに保存されます。
cl> page database/idsp0027.ms
```

以上で、sp0027.ms.fits に対する波長較正が終了します。次に、sp0027.ms をリファレンスとして他のコンパリゾンランプデータの波長較正をします。タスクは reidentify を使います。

```
cl> reidentify sp0027.ms sp0018.ms \
        coord=linelists$thorium.dat v+ inter+


cl> imhead sp0027.ms l+
```

ヘッダーをみると、キーワードの refspec1 が自分自身の名前になっています。次のタスクは refspectra です。これは波長較正したコンパリゾンスペクトルをオブジェクトに割り当てる働きをします。いろいろな方法がありますが、ここでは露出時間の中間の UT(Universal Time) で割り当てることにします。このような UT の値はオブジェクトのヘッダーには記録されていません。そこで、setairmass というタスクで UT を計算します。

```
cl> hselect sp*.ms.fits $I,ut,exptime yes
cl> lpar setairmass
cl> setairmass sp*.ms.fits update-
cl> setairmass sp*.ms.fits
cl> imhead sp0027.ms l+
```

refspectra のパラメータは以下を参考にしてください。

```
cl> epar refspectra


cl> lpar refspectra
      input = "sp*.ms.fits"   List of input spectra
     answer = "no"            Accept assignment?
(references = "sp*.ms.fits")  List of reference spectra
 (apertures = "")             Input aperture selection list
    (refaps = "")             Reference aperture selection list
 (ignoreaps = yes)            Ignore input and reference apertures?
    (select = "interp")       Selection method for reference spectra
      (sort = "utmiddle")     Sort key
     (group = "")             Group key
      (time = yes)            Is sort key a time?
  (timewrap = 17.)            Time wrap point for time sorting
  (override = no)             Override previous assignments?
   (confirm = yes)            Confirm reference spectrum assignments?
    (assign = yes)            Assign the reference spectra to the input spect
```

```
  (logfiles = "STDOUT,logfile") List of logfiles
   (verbose = no)              Verbose log output?
      (mode = "ql")
```

```
      cl> refspectra          # "yes"または"YES"と答えましょう。
                                両者の違いは何ですか
      cl> imhead sp0025.ms l+ # 新しいキーワードがあります。
```

波長の割り当ては dispcor というタスクで行います。イメージヘッダーの refspec というパラメータがどのコンパリゾンデータを用いるかを指定しています。

dispcor のパラメータは以下のようにしてください。

```
      cl> epar dispcor
```

```
      cl> lpar dispcor
      input = "sp*.ms.fits"   List of input spectra
     output = "%sp%l%*.ms.fits" List of output spectra
 (linearize = yes)            Linearize (interpolate) spectra?
  (database = "database")     Dispersion solution database
     (table = "")             Wavelength table for apertures
        (w1 = INDEF)          Starting wavelength
        (w2 = INDEF)          Ending wavelength
        (dw = INDEF)          Wavelength interval per pixel
        (nw = INDEF)          Number of output pixels
       (log = no)             Logarithmic wavelength scale?
      (flux = yes)            Conserve flux?
 (samedisp = no)              Same dispersion in all apertures?
    (global = yes)            Apply global defaults?
(ignoreaps = no)              Ignore apertures?
   (confirm = yes)            Confirm dispersion coordinates?
  (listonly = yes)            List the dispersion coordinates only?
   (verbose = yes)            Print linear dispersion assignments?
   (logfile = "")             Log file
      (mode = "ql")
```

```
      cl> dispcor
```

割り付けを変更しますか (changing assignments) という問いに対しては NO と答えましょう。

```
      cl> dispcor listonly-
```

```
cl> imhead l0025.ms l+
```

これで、データリダクションは終わりです。お疲れさまでした。でも、まだ終わっていません。早速データで遊んでみましょう。

```
cl> splot l0025.ms      # SPLOT コマンドへようこそ。SPLOT は、たくさんの
                                使い道があるコマンドです。
cl> specplot l0020.ms,l0021.ms,l0022.ms
```

同じ天体のスペクトルを 3 枚撮っています。これは、観測者が宇宙線を除去するのを目的としているのかもしれません。

```
cl> scombine l0020.ms,l0021.ms,l0022.ms spec1 \
        combine=median scale=mode
cl> splot spec1
o                       # 元データを重ね書きします。
g
l0020.ms
o
g
l0021.ms
o
g
l0022.ms
g
spec1
q
```

splot には bad pixels をきれいにするためにカーソルキーがあります。また、スペクトル線を測定するためのカーソルキーもあります。詳しくは phelp で勉強してください。

参考文献

A User's Guide to Reducing Slit Spectra with IRAF, by Phil Massey, Frank Valdes, and Jeannette Barnes, April 1992.

Online help is available for all tasks. Especially see "phelp onedspec.package" and "phelp onedspec.specwcs".

See "phelp doslit" for doing it all in one big gulp, and better too, since it extracts the arcs for each stellar spectrum.

```
---------------------------------end of exercise-----------------------------------
```