

IRAFを用いたスペクトル解析 [付録]

ロングスリット分光

すばる春の学校2010 HDS解析講習資料

2010年5月31日(月)－6月2日(水)

A はじめに

本文¹ではHDSの観測モードのうち、クロスディスペルザを使って幅広い波長域を含んだデータを取得する場合について、解析手順を説明しました。この付録では、クロスディスペルザを使わずに、ロングスリットを使って狭い波長域のみを含むデータの解析について説明します。

A.1 クロスディスペルザモードとロングスリットモード

HDSのようなエシェル分光器では、高い干渉次数を用いる際、異なる干渉次数に対するスペクトルが重なって得られます(本文 p4)。データの記録の仕方には、クロスディスペルザモードとロングスリットモードの二種類があります。本文でとりあげたクロスディスペルザモードは、エシェルとは別の分散素子(クロスディスペルザ)を使ってエシェルによる波長分散と垂直方向に光を分散するために、2次元検出器上にスペクトルが折り畳まれて記録されます(図1左)。そのため、検出器上に広い波長域にわたるデータが記録できます。一方、ロングスリットモードの場合は、クロスディスペルザの代わりに平面鏡を用い、さらに限られた波長範囲の光だけが透過するフィルターを装着します。その結果、特定の波長範囲の光だけが記録されます(図1右)。このロングスリットモードでは、狭い波長域しか記録できないかわりに、検出器上で天体の空間方向の情報を広く読み取ることができるので、銀河や星間雲などひろがった天体の観測に適しています。

A.2 作業の流れ

クロスディスペルザモードの場合と基本的には同じで、オーバースキャン領域の処理(本文 p9)、バイアス・ダーク補正(本文 p13)、フラットフィールドイング(本文 p19)を行います。スペクトルを抽出する段階からは、ロングスリットモードの場合、空間方向を積分した一次元スペクトルではなく、空間方向の情報を保ったままの2次元スペクトルを導出します。次に波長較正をして、横軸をピクセルから波長に直します。その際、スリット像のゆがみも補正します。

A.3 ターゲット天体

解析講習では若い星から吹き出ているジェットである、HH46-47という天体について、解析を行います。天体のデータは図2のようにスリットをあてて取得されています(スリットが天体のどの位置にあっているかは、スリットビューワで確認されます)。

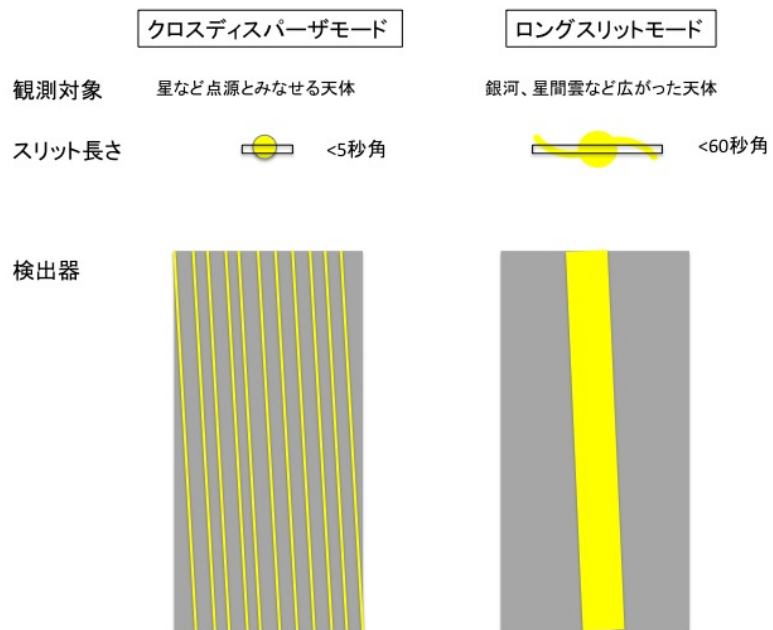


図 1: クロスディスペルザモードとロングスリットモードの概念図。

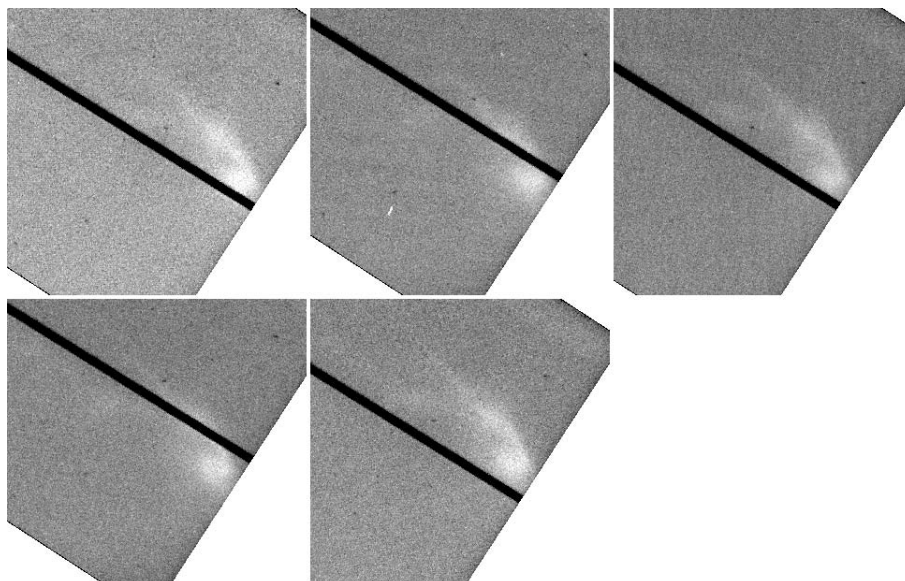


図 2: スリットビューワの画像。白くぼやけて見えるのが天体で黒い帯状に写っているのがスリット。

A.4 解析で使用するファイル

作業に移る前に、解析で使う画像ファイルを確認しておきましょう。本文 p7-8 にしたがって IRAF を立ち上げたら、データのあるディレクトリに移動し、次のようにコマンドを入力してみてください。

```
ecl> imhead *.fits
```

ファイル名はすべて、'HDSA' にフレーム ID がついたものになっています。FLAT, BIAS, COMPARISON と記されたファイルは較正用、" HH46-47" などと括弧書きで記されたファイルは天体のデータです。HDS は 2 つの CCD カメラがあり、通常は 1 回の露出でフレーム ID が奇数と偶数のふたつの fits ファイルが作られますが (本文 p9)、今回解析するロングスリットモードでは、ひとつの CCD に対応する奇数のフレーム ID のデータのみを使用します。

以下の章では、これらのファイルを使った実際の作業について詳しく説明します。

B オーバースキャン領域処理、バイアス補正、フラットフィールドイング

まずはじめに、CCD の画像のなかで不要なオーバースキャン領域の処理を行います。それには、本文 p9-p11 を参考に IRAF のスクリプト `overscan.cl` を使います。処理が終わったら、一度画像を表示してみましょう (本文 p11)。うまくいっていれば、図 3 のような画像が得られるはずですが、この画像ではスリットの長さに対応する幅のスペクトルが検出器の右半分に写っています。スペクトル上の明るく写っている部分が天体からの光です。

次に天体のスペクトルから CCD 自身からのシグナルを差し引く、バイアス補正を行います。バイアス補正の方法は本文 p13-14 を参照してください。

バイアス補正が終わったら、フラットフィールドイングを行い、CCD の感度のムラを補正します。感度ムラの補正には、天体ではなく連続光を出すランプの光をスリットにあててとったフラットデータを用います (本文 p19)。この感度ムラだけを反映するフラットデータで天体のデータを割れば、補正されたデータを得ることができます。しかし、このフラットデータのカウン트가大きすぎたり小さすぎたりすると、天体を割った結果得られるデータの数値が大きくばらついてその後の解析が難しくなります。そこでフラットデータは 1 に近い値をとるように規格化しておきます。ロングスリットモードの場合、次のようにして行います。

まずは、本文 p19 にしたがって複数のフラットデータを `combine` で合わせて作ったフラットフレームの断面図を見てみましょう。

```
ecl> implot flat
```

光があたっていてカウンタ数の高い部分の平均値をタスク `imstat` を使って計算し、この数でデータを割ります。例えば次のようにします。

```
ecl> imstat flat[1300:1600,*]
#          IMAGE      NPIX      MEAN      STDDEV      MIN      MAX
flat[1300:1600,*] 1234100 28167.  11293.  72.08  78559.
ecl> imarith flat / 28167 flatn
```

ここでは `flat.fits` というファイルのうち、光があたっている領域、1300-1600 列を `imstat` の入力に指定しています。すると、その領域のピクセル数 (`NPIX`)、平均値 (`MEAN`)、標準偏差 (`STDDEV`)、最大・最小値 (`MIN`・`MAX`) が出力されています。

結果得られたフレームの断面図を見て、光のあたっている部分が規格化されているかチェックしましょう。

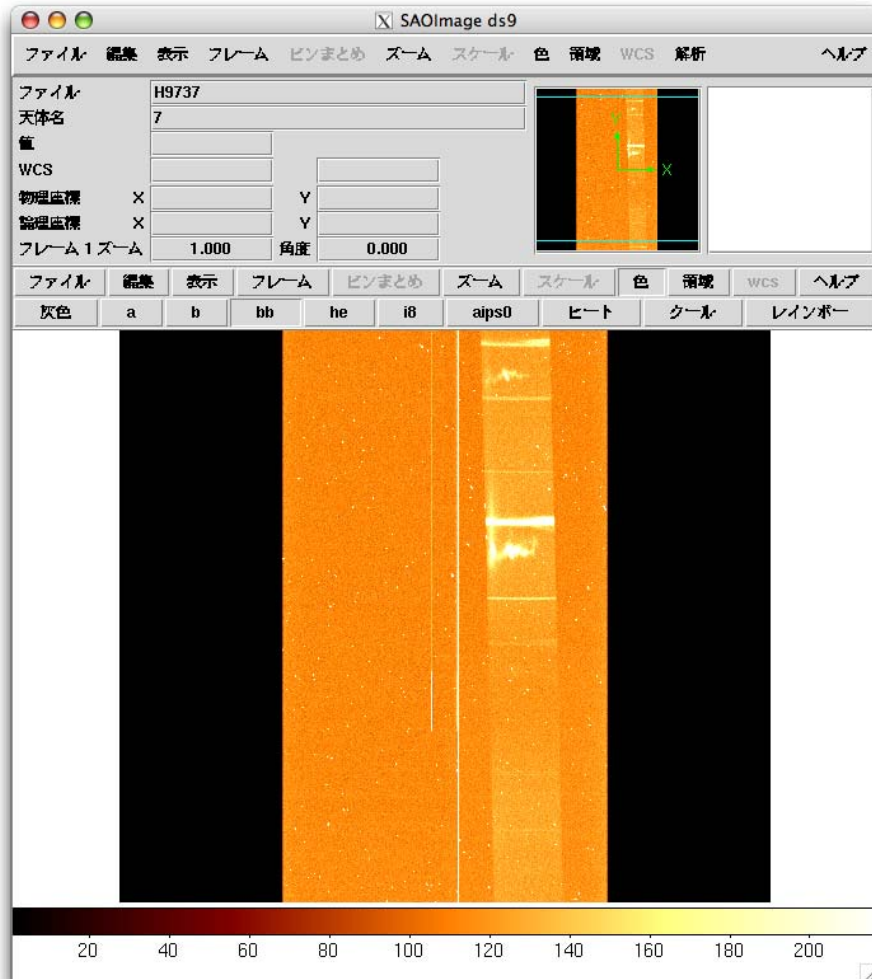


図 3: ds9 によるデータ表示例。ロングスリットモードによる天体のデータの 2 次元画像を表示している。

```
ecl> implot flat
```

この段階では、光があたっていない部分は 0 に近いカウント数になっています。このままで天体を割ると、その部分だけ数値が大きくなって解析がしづらくなってしまうため、`imreplace` を使って 1 に置き換えます。

```
ecl> imreplace flatn 1 upper=0.1
```

上のコマンドでは入力ファイル `flatn.fits` にたいして、0.1 より小さいカウント数を 1 に置き換えるように指示しています。

結果を `ds9` で表示し、図 4 のように規格化されているかチェックしましょう。規格化が終わったら、このフレームで天体のデータを割ります。その際、あらかじめ入力ファイル名と出力ファイル名をリストにしておいて、それぞれ好きなファイル名で保存し、次のようにファイル名の先頭に `@` をつけて指定すると、リストにしたファイルを一気に処理できるので簡単です (本文 p41 参照)。

```
echelle> imarith @obj.lst / flatn @objf.lst
```

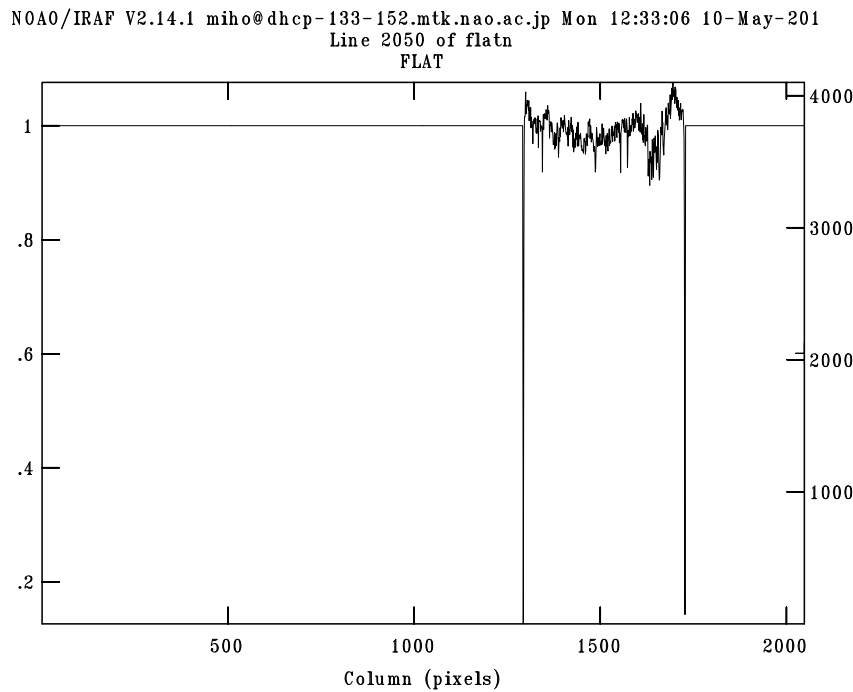


図 4: 平均値で規格化したフラットフレーム。スリット以外の光があたらない部分はカウントが小さく、天体を割ったときに大きな値を出してしまうことを避けるために、1 に置き換えてある。

C オーダートレース

次に、ここまで処理した画像データから、スペクトルの写っている部分を取り出すために、`echelle` パッケージの `apall` というタスク (`noao.imred.echelle`) を用いてオーダートレースを行います。オーダートレースは、

まずスペクトルが写っている位置の参照データを明るい星(標準星)の画像データを使って作ります(図5)。
そのために、まず IRAF 上で echelle パッケージに移動して apall のパラメータを下記のように設定します。

```
ecl> noao
    artdata.    astutil.    imred.      nproto.    onedspec.   twodspec.
    astcat.     digiphot.  mtlocal.    observatory rv.
    astrometry. focas.     nobsolete.  obsutil.   surfphot.
```

```
noao> imred
    argus.     crutil.    echelle.    iids.      kpnocoude.  specred.
    bias.      ctioslit.  generic.    irred.     kpnoslit.   vtel.
    ccdred.    dtoi.     hydra.      irs.       quadred.
```

```
imred> echelle
    apall      aprecenter  demos       refspectra  sflip
    apdefault@ apresize   deredden    sapertures  slist
    apedit     apscatter  dispcor     sarith       specplot
    apfind     apsum      doecslit    scombine    specshift
    apfit      aptrace    dofoe       scopy        splot
    apflatten  bplot      dopcor      sensfunc    standard
    apmask     calibrate  ecentify    setairmass
    apnormalize continuum  ecreidentify setjd
```

```
echelle> epar apall
```

I R A F

Image Reduction and Analysis Facility

```
PACKAGE = echelle
```

```
TASK = apall
```

```
input      =          H9791bf List of input images
(output    =          H9791bfa) List of output spectra
(apertur=  ) Apertures
(format    =          echelle) Extracted spectra format
(referen=  ) List of aperture reference images
(profile=  ) List of aperture profile images

(interac=  yes) Run task interactively?
(find     =          yes) Find apertures?
(recente=  yes) Recenter apertures?
(resize   =          yes) Resize apertures?
(edit     =          yes) Edit apertures?
(trace    =          yes) Trace apertures?
(fittrac=  yes) Fit the traced points interactively?
(extract=  yes) Extract spectra?
```

```

(extras =          no) Extract sky, sigma, etc.?
(review =          yes) Review extractions?

(line  =          INDEF) Dispersion line
(nsum  =          50) Number of dispersion lines to sum or median

                                # DEFAULT APERTURE PARAMETERS

(lower =          -5.) Lower aperture limit relative to center
(upper =          5.) Upper aperture limit relative to center
(apidtab=         ) Aperture ID table (optional)

                                # DEFAULT BACKGROUND PARAMETERS

(b_funct=        chebyshev) Background function
(b_order=         1) Background function order
(b_sampl=         *) Background sample regions
(b_naver=        -3) Background average or median
(b_niter=         3) Background rejection iterations
(b_low_r=         3.) Background lower rejection sigma
(b_high_=         3.) Background upper rejection sigma
(b_grow =         0.) Background rejection growing radius

                                # APERTURE CENTERING PARAMETERS

(width  =         5.) Profile centering width
(radius =         10.) Profile centering radius
(thresho=        0.) Detection threshold for profile centering

                                # AUTOMATIC FINDING AND ORDERING PARAMETERS

nfind  =          1  Number of apertures to be found automatically
(minsep =          2.) Minimum separation between spectra
(maxsep =         150.) Maximum separation between spectra
(order  =          increasing) Order of apertures

                                # RECENTERING PARAMETERS

(aprecen=         ) Apertures for recentering calculation
(npeaks =         INDEF) Select brightest peaks
(shift  =          yes) Use average shift instead of recentering?

                                # RESIZING PARAMETERS

```

```

(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 width
(peak   =         yes) Is ylevel a fraction of the peak?
(bkg    =         no) Subtract background in automatic width?
(r_grow =         0.) Grow limits by this factor
(avglimi=        yes) Average limits over all apertures?

          # TRACING PARAMETERS

(t_nsum =          10) Number of dispersion lines to sum
(t_step =          10) Tracing step
(t_nlost=          3) Number of consecutive times profile is lost before qui
(t_funct= legendre) Trace fitting function
(t_order=          3) Trace fitting function order
(t_sampl=          *) Trace sample regions
(t_naver=          1) Trace average or median
(t_niter=          3) Trace rejection iterations
(t_low_r=         3.) Trace lower rejection sigma
(t_high_=         3.) Trace upper rejection sigma
(t_grow =          0.) Trace rejection growing radius

          # EXTRACTION PARAMETERS

(backgro=         none) 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?
(saturat=        INDEF) Saturation level
(readnoi=         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)

```

input には標準星のデータを、output には好きな名前をいれます。このタスクは、データの断面を表示して、スペクトルがどの位置に写っているかを探します (find)、そして、スペクトルの中心位置を調べ (recente)、データを切り出す幅 (アパーチャと呼びます) を決めます (resize)、自動で決めたあと、必要なら手動で修正を加えます (edit)、これらを全て行う場合には、yes にしておきます。アパーチャのサイズと位置が決まっ

たら、スペクトルのトレースを行い、スペクトルの位置を関数でフィットしたうえで、2次元画像からアパーチャの幅を持ったスペクトルを導出します。これらを行うために、trace、fittrac、extractをyesにしておきます。

他に重要なパラメータは、

- nfind: スペクトルを何本検出するか。ロングスリットモードではスペクトルが一本しか写っていないので、1にします。
- peak, ylevel: peakをyesにすると、ylevelで指定した箇所で自動的にアパーチャのサイズを決めます。例えばpeakをyes、ylevelを0.1にするとスペクトルのピークの高さの10%のところがアパーチャの両端になります。

:goと入力して実行すると、図6左のような画面が表れます。これは自動的に標準星の位置を見つけだし、アパーチャを設定したところです。星の位置にアパーチャがきていることを確認したら、qを押してオーダートレースを行います(図6右)。このフィットがよくない場合には、関数の次数をあげたり、フィットをかける範囲を変更したりします。関数の次数の変更には、画面上で、:order 3のように入力します。また、フィットをかける範囲は、両端でsを入力することで指定します。変更後あらためてフィットを行うには、fを入力します。フィットに満足したらqを入力し、終了します。

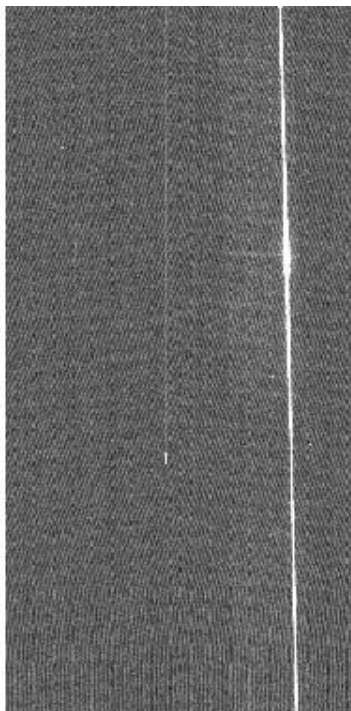


図 5: 標準星の画像

標準星の画像で作った参照データを使って、天体データと comparison データからスペクトルを抽出します。タスクはさきほどと同じ apall です。ただし今回は、パラメータの format を strip とします。これにより、スリット長方向の空間情報を保存したままオーダーごとにスペクトルを抽出することができます。さらに、reference では、さきほどオーダートレースをおこなった標準星の画像を参照データとしてファイル名を

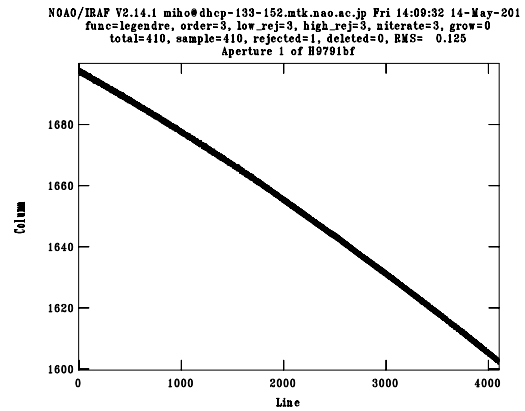
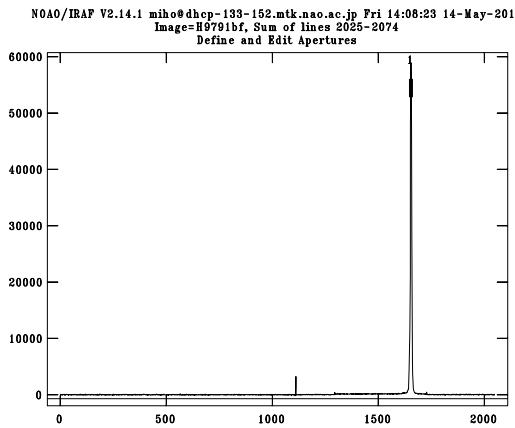


図 6: 左: apall 実行時に表示される標準星データの断面図。自動的にアパーチャを検出したところ。右: 標準星のスペクトルの画像上での位置を、関数でフィットしたところ。

指定します。天体と標準星は、同じ分光器の設定でとっているため、スペクトルが写る位置は基本的に同じです。したがって、先ほどの標準星画像を参照することで、作業を簡略化することができます。スペクトルの位置探し (find) とトレース (trace および fittrac) で参照データを利用するので、これらを no にしておきます。アパーチャのサイズ (resize) は、天体とオーダートレースをおこなった標準星では異なり、天体自身から決めるべきですから、yes にしておきます。実行時の様子を図 7 に示します。この apall での作業の結果、分散方向が縦から横に変化します。ds9 で確認してみてください。

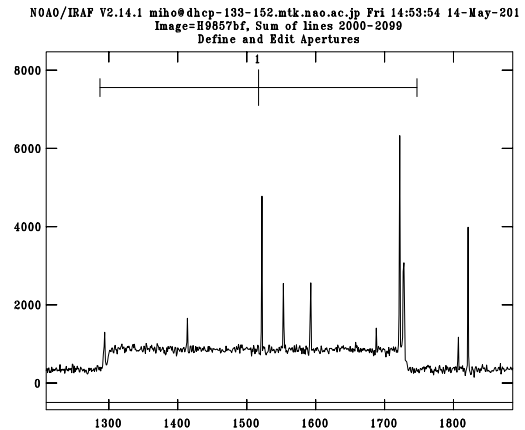
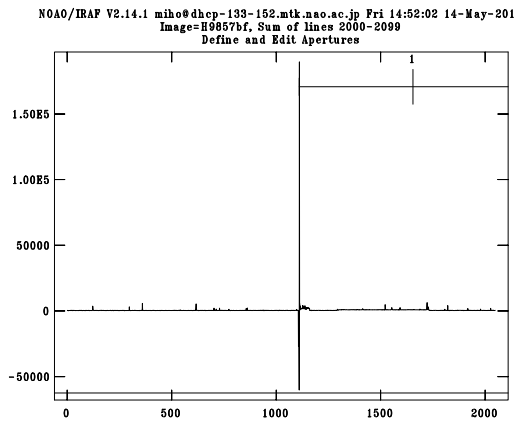


図 7: 左: 天体画像で、アパーチャを検出したところ。天体からの光が弱いために、スペクトルが写っている場所が見えにくい。右: 左の図を拡大したところで盛り上がっている部分が天体からの光。天体からの光が入るように、アパーチャを設定しなおす。(s コマンドでアパーチャ中心をシフトさせ、:lower, :upper コマンドでアパーチャの幅を変える。)

D トリミング

抽出したスペクトルをみると、光があたっていない部分に端に写りこんでいる場合があります。残しておくとのちの解析が難しくなります。また、エシェル分光器の特徴で、波長が折り畳まれて記録されるために、ひとつのオーダーからはみ出た長波長側のスペクトルが、短波長側に重なっている場合があります。そのま

までは波長較正ができないため、不要な部分を切り落とします。ds9 を使ってどの部分を切り落とすか決め、imcopy というタスクで実行します。

```
ec1> imcopy H9857bfa.0001[1800:4099,15:440] H9857bfat
```

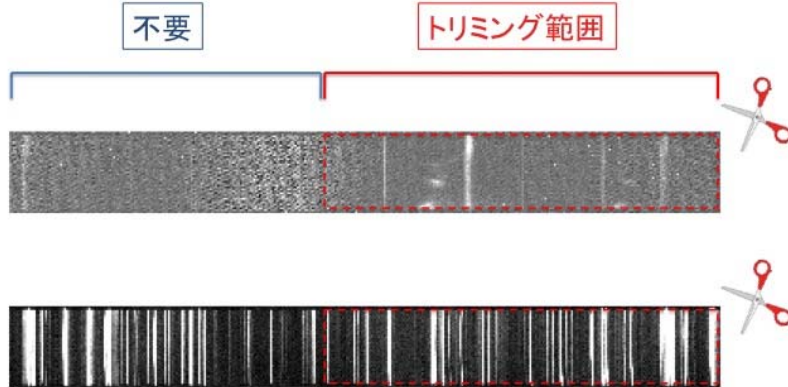


図 8: 2次元スペクトルの抽出を終えたあとの天体フレームと COMPARISON フレーム。上下の光があたっていない部分と、スペクトルの左側 1/3 は imcopy で切り落とす

E 波長較正

以上の解析で得られた画像は横軸が CCD のピクセル数になっており、天体の情報を取り出すには、横軸を波長の単位に直す必要があります。それには、波長があらかじめ分かっている比較光源 (Th-Ar ランプ) のスペクトルを使って、CCD ピクセル数から波長へ変換する関数を求め、その関数を天体の画像に適用します。また、比較光源のデータを見ると分かるように、スリット像は CCD の並びに沿っていません (図 9 上)。つまり、横軸方向のピクセル数と波長の対応関係は、縦軸方向の場所によって異なるということです。これを補正します。最終的には横軸を波長 (\AA)、縦軸が空間方向になるように、2次元座標変換を行います。そのためには、タスク identify, reidentify, fitcoords, transform を使います。

E.1 COMPARISON フレームの波長同定: identify, reidentify

まずは、IRAF の longslit パッケージにあるタスク identify を使って横軸を波長 (\AA) に直します。それにはまず、天体と同じ設定でとった比較光源 (Th-Ar ランプ) データ COMPARISON について、どの輝線がどの波長に相当するか、資料と見比べながら同定します。詳しいやりかたは本文 p26 を見てください。ロングスリットモードの場合、まず longslit パッケージに移動して、identify のパラメータを下記のように設定します。

```
echelle> bye
```

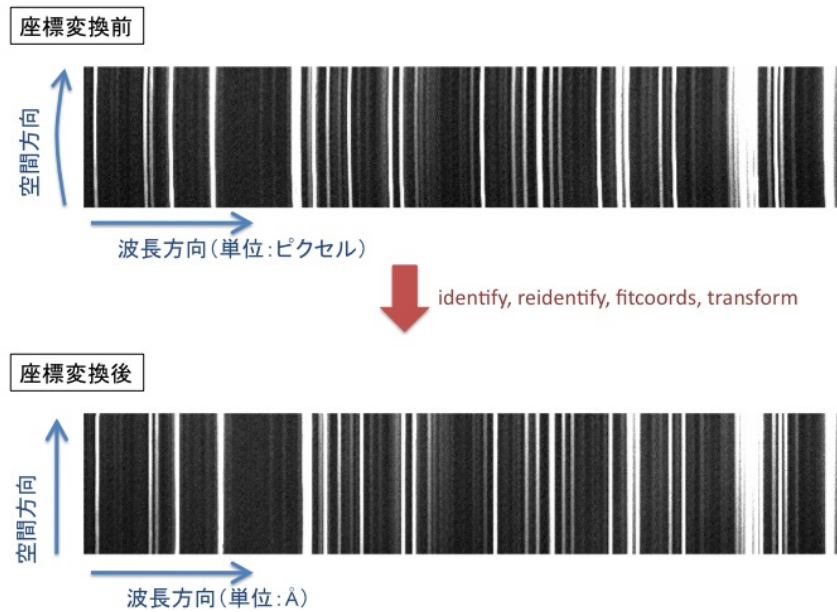


図 9: 座標変換の概念図

```

argus.      crutil.      echelle.    iids.      kpnocoude.  specred.
bias.       ctioslit.    generic.   irred.     kpnoslit.   vtel.
ccdred.     dtoi.       hydra.     irs.       quadred.

imred> bye
  artdata.  astutil.    imred.     nproto.    onedspec.   twodspec.
  astcat.   digiphot.  mtlocal.   observatory rv.
  astrometry. focas.     nobsolete. obsutil.   surfphot.

noao> twodspec
  apextract.  longslit.

twodspec> longslit
  aidpars@   deredden   identify   sarith     specplot
  autoidentify dopcor    illumination scopy      specshift
  background extinction lcalib     sensfunc   splot
  bplot      fceval    lscombine  setairmass standard
  calibrate  fitcoords reidentify setjd      transform
  demos      fluxcalib response   sflip

longslit> epar identify

```

I R A F

Image Reduction and Analysis Facility

```

PACKAGE = longslit
    TASK = identify

images =          H9855bat  Images containing features to be identified
(section=        middle line) Section to apply to two dimensional images
(databas=        database) Database in which to record feature data
(coordli=        linelists$thar.dat) User coordinate list
(units =         ) Coordinate units
(nsum =          20) Number of lines/columns/bands to sum in 2D images
(match =         20.) Coordinate list matching limit
(maxfeat=        50) Maximum number of features for automatic identificatio
(zwidth =        100.) Zoom graph width in user units
(ftype =         emission) Feature type
(fwidth =        7.) Feature width in pixels
(cradius=        10.) Centering radius in pixels
(thresho=        50.) Feature threshold for centering
(minsep =        2.) Minimum pixel separation
(funcutio=        chebyshev) Coordinate function
(order =         3) Order of coordinate function
(sample =        *) Coordinate sample regions
(niterat=        5) Rejection iterations
(low_rej=        3.) Lower rejection sigma
(high_re=        3.) Upper rejection sigma
(grow =          0.) Rejection growing radius
(autowri=        yes) Automatically write to database
(graphic=        stdgraph) Graphics output device
(cursor =        ) Graphics cursor input
crval =          Approximate coordinate (at reference pixel)
cdelt =          Approximate dispersion
(aidpars=        ) Automatic identification algorithm parameters
(mode =          ql)

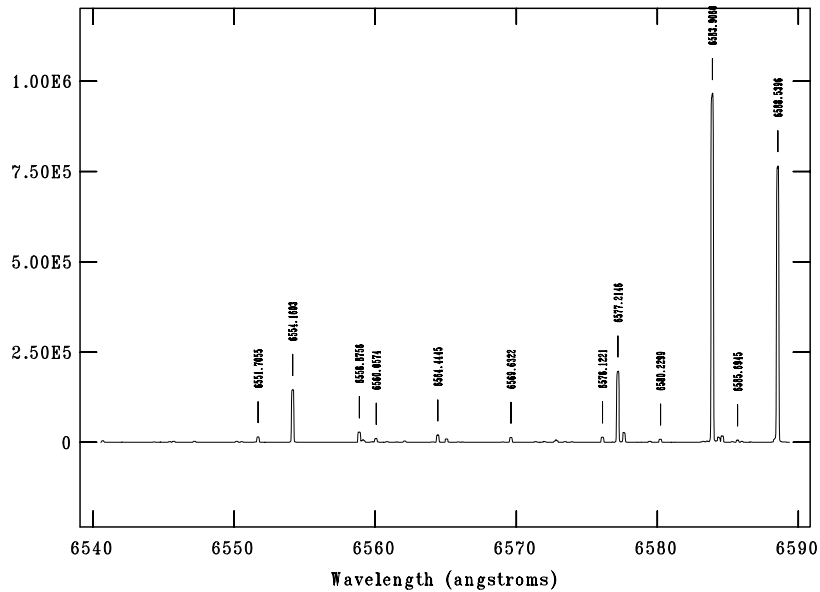
```

images には、波長同定をしたい COMPARISON フレームの名前をいれます。上から 2 番目の section には、スリット像のどの行について波長同定をするかを指定します。ここでは中心部の行 (middle line) を指定します。coordli は、参照するラインリストを指定しています。波長同定のときには、すべて手動で波長を入力するのは大変なので、このラインリスト (Tr-Ar ランプの場合は、'linelists\$thar.dat') を参照して、途中から自動で入力できるようにします。

:go で実行すると、図 10 のように、比較光源のスペクトルが表れます。identify を使った波長同定については本文 26p を参照してください。

identify の結果、スリットの中央部の行について、CCD のピクセルと波長の対応が得られました。しかしそれ以外の場所 (行) でも同じピクセル数-波長の対応関係にあてはまるとは限りません。そこでタスク reidentify によって、他の行についても正しい波長をいれる作業を reidentify を使って行います。

```
longslit> epar reidentify
```



NOAO/IRAF V2.14.1 miho@dhcp-133-152.mtk.nao.ac.jp Thu 14:40:04 13-May-201

図 10: タスク identify で波長を同定したスペクトル

I R A F

Image Reduction and Analysis Facility

PACKAGE = longslit

TASK = reidentify

```

referenc=      H9855bat Reference image
images  =      H9855bat Images to be reidentified
(interac=      no) Interactive fitting?
(section=      middle line) Section to apply to two dimensional images
(newaps =      no) Reidentify apertures in images not in reference?
(overrid=      yes) Override previous solutions?
(refit  =      yes) Refit coordinate function?

(trace  =      yes) Trace reference image?
(step   =      2) Step in lines/columns/bands for tracing an image
(nsum   =      2) Number of lines/columns/bands to sum
(shift  =      0.) Shift to add to reference features (INDEF to search)
(search =      0.) Search radius
(nlost  =      0) Maximum number of features which may be lost

(cradius=      10.) Centering radius
(thresho=      0.) Feature threshold for centering
  
```

```

(addfeat=          yes) Add features from a line list?
(coordli=  linelists$thar.dat) User coordinate list
(match  =          20.) Coordinate list matching limit
(maxfeat=          25) Maximum number of features for automatic identificatio
(minsep =          2.) Minimum pixel separation

(databas=          database) Database
(logfile=          logfile) List of log files
(plotfil=          ) Plot file for residuals
(verbose=          yes) Verbose output?
(graphic=          stdgraph) Graphics output device
(cursor =          ) Graphics cursor input

answer =          no  Fit dispersion function interactively?
crval  =          Approximate coordinate (at reference pixel)
cdelt  =          Approximate dispersion
(aidpars=          ) Automatic identification algorithm parameters
(mode  =          ql)

```

referenc には、基準として参照するデータとして、identify で波長同定が済んでいる画像のファイル名をいれます。images は、reidentify で波長を同定したい画像のファイル名を指定します。ここでは identify が済んでいる COMPARISON フレーム自身について波長同定をしたいので、referenc と同じファイル名を指定します。:go で実行すると、各行での波長同定の結果が次のように表示されます。

```
REIDENTIFY: NOAO/IRAF V2.14.1 miho@dhcp-133-152.mtk.nao.ac.jp Wed 09:56:18 19-May-2010
```

```
Reference image = H9855bat, New image = H9855bat, Refit = yes
```

Image Data	Found	Fit Pix	Shift	User Shift	Z Shift	RMS
H9855bat[* ,211]	10/10	16/16	0.0578	0.00116	1.77E-7	0.018
H9855bat[* ,209]	16/16	16/16	0.0804	0.00141	2.14E-7	0.0143
H9855bat[* ,207]	15/16	15/15	0.15	0.00288	4.39E-7	0.0186
H9855bat[* ,205]	15/15	16/16	-0.168	-0.00334	-5.1E-7	0.0156
H9855bat[* ,203]	15/16	15/15	0.0573	0.00132	2.02E-7	0.0156
H9855bat[* ,201]	14/15	16/16	-0.144	-0.00282	-4.3E-7	0.0187
H9855bat[* ,199]	15/16	16/16	0.106	0.00196	2.98E-7	0.0164

```

:
:

```

ここではパラメータ step で指定した行間隔で、同定の結果を示しています。” ** Too many features lost ** ” というメッセージが出て同定が途中で止まる場合は、パラメータの step (波長同定を行う行間隔) や、nsum (何行分足し合わせてから同定するか) を変えて、もういちど実行してみてください。

ラインを同定した結果は'database/id[filename]' というファイルに保存されます。

E.2 座標変換関数の決定 : fitcoords

longslit パッケージのタスク fitcoords を使って、先ほど reidentify を適用した COMPARISON フレームについて、座標変換のためのフィッティングを行います。座標変換には、reidentify で記録した 2 次元パラメータを使います。fitcoords のパラメータの設定例は以下のとおりです。

```
longslit> epar fitcoords
```

```

                                I R A F
                                Image Reduction and Analysis Facility

PACKAGE = longslit
TASK = fitcoords

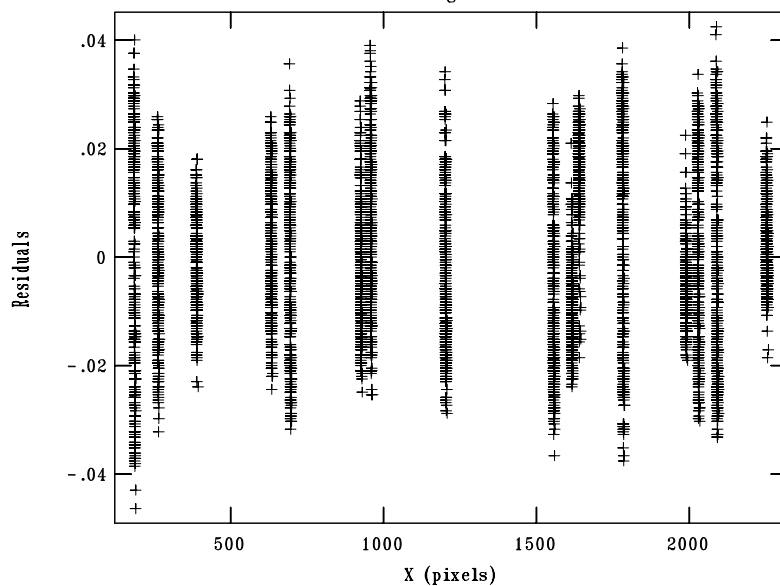
images =          H9855bat Images whose coordinates are to be fit
(fitname=        H9855bat) Name for coordinate fit in the database
(interac=        yes) Fit coordinates interactively?
(combine=        yes) Combine input coordinates for a single fit?
(databas=       database) Database
(deletio=        deletions.db) Deletion list file (not used if null)
(funcutio=       chebyshev) Type of fitting function
(xorder =        6) X order of fitting function
(yorder =        6) Y order of fitting function
(logfile=        STDOUT,logfile) Log files
(plotfil=        plotfile) Plot log file
(graphic=        stdgraph) Graphics output device
(cursor =        ) Graphics cursor input
(mode =          ql)
```

images はフィッティングをしたい画像のファイル名で、先ほど identify, reidentify を適用して座標変換のための情報を記録した COMPARISON フレームを指定します。実行すると、フィッティングのための x 方向のピクセル数に対して、フィットした関数からの残差が、図 11 のように表示されます。飛び抜けて残差が大きい点は、identify や reidentify での輝線の同定が間違っている可能性が高いので、点の近くにカーソルをもっていって、d→p と入力するとフィットから外すことができます。不要な点を消して改めて f を入力してフィットします rms が 0.02 より小さくなれば、おおむねフィットがもっともらしいと言えます。フィットに満足したら、q を入力して終了します。最後に座標変換パラメータを記録するかどうか聞かれるので yes で答えます。結果の座標変換パラメータは 'database/fc[filename]' に保存されます。

E.3 座標変換の実行 : transform

これまでの作業で、座標変換を行うためのパラメータが出揃いました。ここではそれらのパラメータを使って、座標変換を実行します。それには transform を使います。まずは先ほどから座標変換の基準として使っている COMPARISON フレームについて実行します。パラメータの設定例は以下のとおりです。

NOAO/IRAF V2.14.1 miho@dhcp-133-152.mtk.nao.ac.jp Mon 11:41:45 17-May-201
 Function = chebyshev, xorder = 6, yorder = 6, rms = 0.01572
 Fit User Coordinates to Image Coordinates for H9855bat



NOAO/IRAF V2.14.1 miho@dhcp-133-152.mtk.nao.ac.jp Mon 11:41:45 17-May-201

図 11: fitcoords の実行画面

```
longslit> epar transform
```

I R A F

Image Reduction and Analysis Facility

PACKAGE = longslit

TASK = transform

```
input      =      H9855bat  Input images
output     =      H9855batw Output images
(mininput  =      ) Input masks
(moutput   =      ) Output masks
fitnames=      H9855bat  Names of coordinate fits in the database
(databas=      database) Identify database
(interpt=      spline3) Interpolation type
(x1       =      INDEF) Output starting x coordinate
(x2       =      INDEF) Output ending x coordinate
(dx       =      INDEF) Output X pixel interval
(nx       =      INDEF) Number of output x pixels
(xlog    =      no) Logarithmic x coordinate?
(y1       =      INDEF) Output starting y coordinate
(y2       =      INDEF) Output ending y coordinate
(dy       =      INDEF) Output Y pixel interval
(ny       =      INDEF) Number of output y pixels
```

```
(ylog   =          no) Logarithmic y coordinate?
(flux   =          yes) Conserve flux per pixel?
(blank  =          INDEF) Value for out of range pixels
(logfile=          STDOUT,logfile) List of log files
(mode   =          ql)
```

終わったら ds9 で出力した画像 ('H9855batw.fits') を確認してみましょう。図 9 の下ののように、輝線がまっすぐになっているでしょうか。

変換が正常にできたことを確認したら、天体の画像にも transform を実行します。その際、input は天体の画像ファイル名、output は好きな名前、fitnames は fitcoords を適用した COMPARISON フレームのファイル名を指定します。

F スペクトルの表示

解析が終わった画像をプロットして、天体のスペクトルを見てみましょう。そのためにはタスク `splot` が便利です。

```
longslit> splot H9863bfatw
```

上のようにファイル名を指定して実行すると、どのラインをプロットするか聞かれます。まずはデフォルト 1 (スリット長の端) のままりターンをしてみましょう。図 12 のようにスペクトルが表示されます。他のラインに移動してプロットするには、'(または)' を入力します。拡大するには、w を入力したあと、拡大したい領域の左下にカーソルをおいて e、続けて右上にカーソルをおいて e と入力します。

練習問題

- どの波長に何のスペクトル線が検出されていますか。
- ひとつのスペクトル線に注目して、どの場所 (ライン) で一番強いでしょうか (ヒント: スペクトル線の強度をはかるには `splot` 実行時に、スペクトル線の両脇で k を入力します。)
- ひとつのスペクトル線に注目して、スペクトル線の中心波長や幅が場所ごとに変化するかどうか調べましょう。

以上で基本的な解析作業は終了です。ここからはデータを解釈し、ターゲットの天体についてさまざまな物理量を決定する段階に入ります。例えばロングスリットモードのデータを使うと、天体の空間構造や、天体に付随するガスの運動状態などを知ることができます²。今後の解析については本文 p37-40 を参照してください。

参考文献

1. 青木和光、すばる/HDS データ解析講習会資料「IRAF を用いたスペクトル解析」
2. Nishikawa et al. 2008, ApJ, 684, 1260

NOAO/IRAF V2.14.1 miho@dhcp-133-152.mtk.nao.ac.jp Mon 13:58:12 17-May-201
[H9863bfatw[*],297]: HH46-47 - Aperture 1 1800. ap:297 beam:0

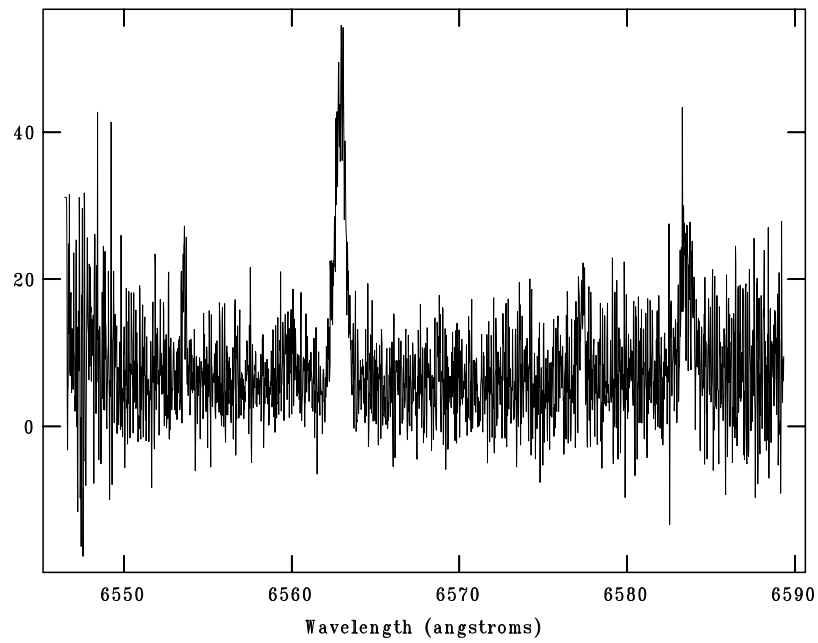


図 12: plot によるスペクトルの表示