at 国立天文台

Python+Jupyter notebookによる光赤外撮像データ処理入門

2017-08-24, 25 中島 康

Jupyter notebookの使い方

講習0 : Pythonとnotebook

講習1 : IRAFを使ってみる

講習2 : IRAFで一次処理　　PyRAF(IRAF)

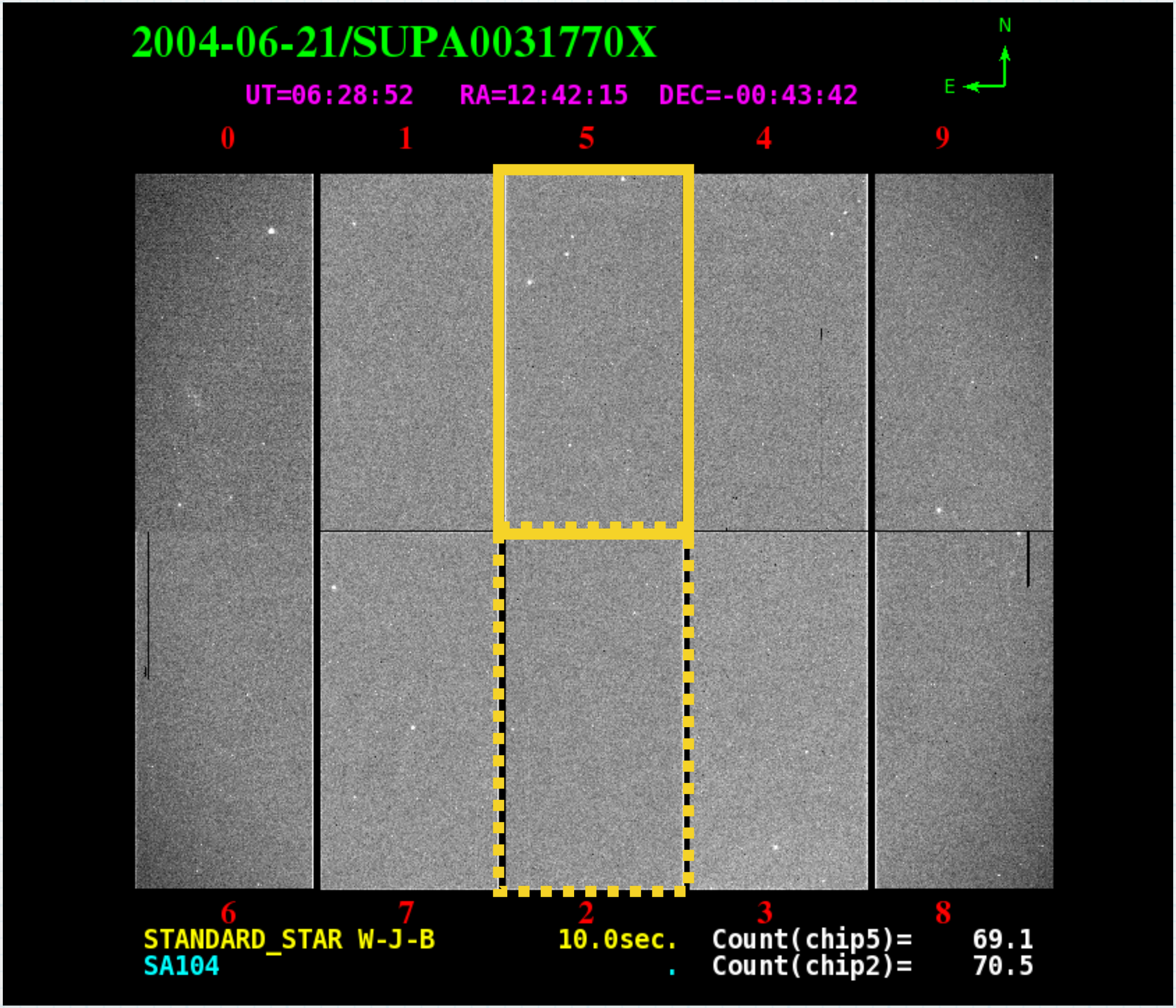
講習3 : 星の測光

講習4 : Numpyの基本

講習5 : astropy.io.fitsの基本

講習6 : matplotlibの基本

講習7 : スクリプト作成等


2004-06-21/SUPA0031770X
UT=06:28:52 RA=12:42:15 DEC=-00:43:42
N
E
0
1
5
4
9
6
7
2
3
8
STANDARD_STAR W-J-B
SA104
10.0sec.
Count(chip5)= 69.1
Count(chip2)= 70.5

- Python interactive + script + ノートブックで記録に

- サンプルデータ(Suprime Cam)のデータ処理、測光を行いながら、基本的なスキルを身につける。

- 今回の資料を見ながら、それを思い出して、後に自分のデータ処理、解析に応用することができるようになる。

- 講習で作成したnotebookファイルを持ち帰って、自分のマシンで復習あるいは今後の参考に (USB?クラウド?)

IRAF Image Reduction and Analysis Facility

- アメリカのNOAO (国立光学天文台)が作成 (1986~)
- 光赤外データ処理・解析のデファクトスタンダード
- 画像演算、測光などのタスク(プログラム)の集合
- 長期間、多くの人によって使われてきた「枯れたシステム」--- バグが淘汰されている
- IRAFのコマンドラインから対話的データ処理
- 独自のプログラム言語 IRAF-CL

Python

- 今、最も広く使われているコンピュータ言語のひとつ
- PyRAFが「仲介」することでPythonからIRAFのタスクを利用できる！
- 数多くのライブラリ ― 数値計算、統計、可視化、機械学習、、、、、、
- 多くの教科書、ウェブ上の情報

Python

- 今、最も広く使われているコンピュータ言語のひとつ
- PyRAFが「仲介」することでPythonからIRAFのタスクを利用できる！
- 数多くのライブラリ ― 数値計算、統計、可視化、機械学習、、、、、、
- 多くの教科書、ウェブ上の情報

実験ノート(研究ノート)をとろう！

Jupyter notebookがひとつのソリューション

Jupyter notebookの使い方

- Jupyter notebookの起動
- 新規ノートブックファイルの作成
- Pythonとの対話
- 対話を保存
- Jupyter notebookの終了

メモの書き込み

Markdownで書き込み

講習0 : Pythonとnotebook

- notebook形式の資料の使い方
- Pythonのforループ ~ 字下げが意味をもつ
- Pythonのモジュール
- notebookでの変数の読み込み

ipython の補完機能　~ [tab]キー

最初の何文字か　+ [tab]

候補のリスト

- 定義済み変数
- 使用可能な関数

など

変数.何文字か　+ [tab]

候補のリスト

- 使用可能なメソッド

ドット

0文字なら全候補

講習1 : IRAFを使ってみる

IRAFとPythonをつなぐPyRAF

IRAFの基本タスク(コマンド)
display, imexam, imstat

IRAFタスクのパラメータ設定

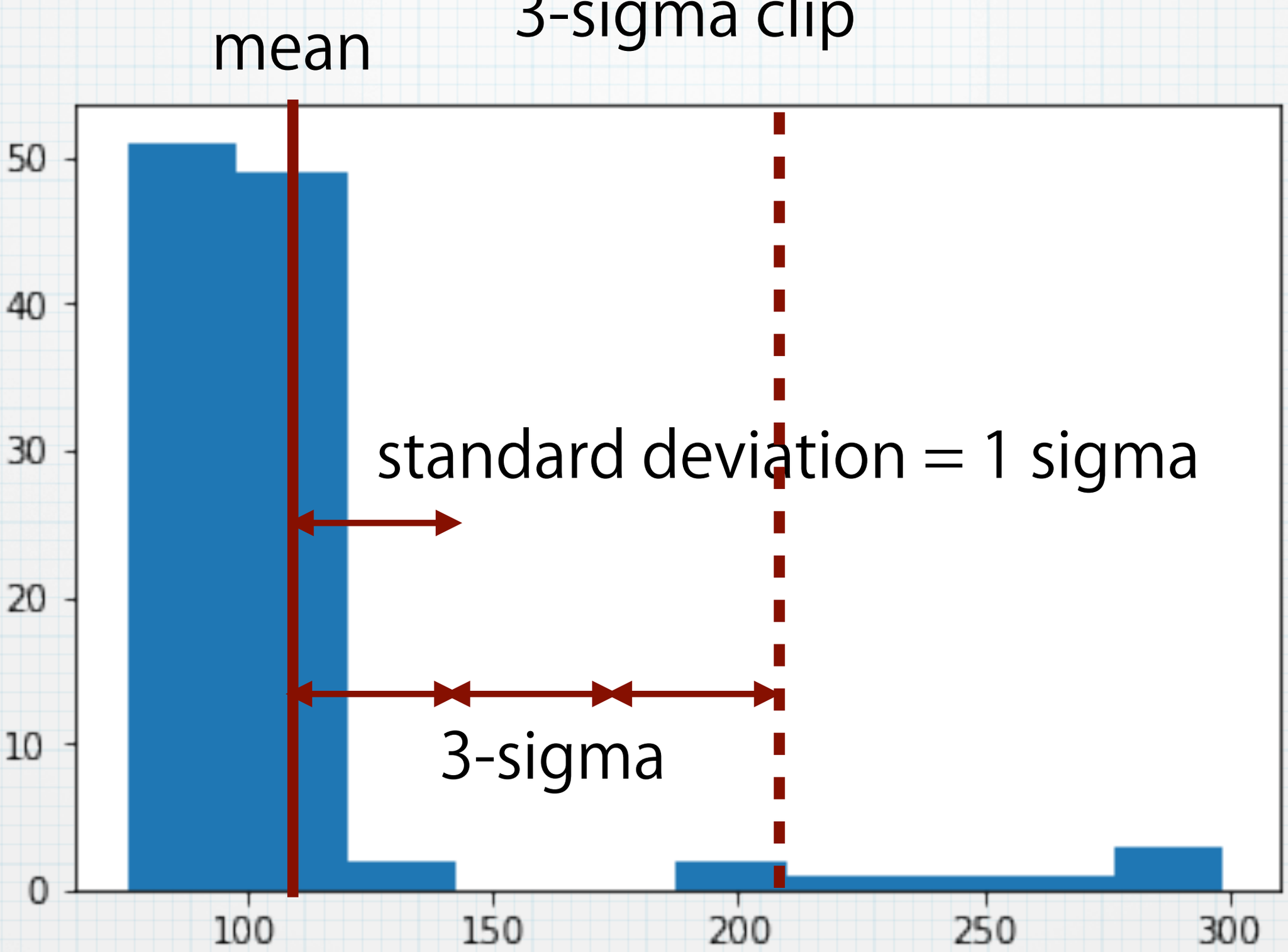
演習1

演習1

data1/SUPA00317885.fitsはtarget2の生データです。

新しいノートブックファイルを作成し、

1. imexamでバックグラウンドの値とばらつき、星の特徴量、を調べる。
2. imstatでカウント値のmedian, mean, standard deviationを求める。
3. imstatのnclipを2以上にしてみてください。どうなりましたか？


mean
3-sigma clip
standard deviation = 1 sigma
3-sigma
50
40
30
20
10
0
100
150
200
250
300

講習2 : IRAFで1次処理

生データ取得　＋ 一次処理に必要なデータ取得

天体以外の信号の除去

感度と光学系のムラの補正

一次処理

一次処理済みデータ

例) 点光源の測光 (＋ 標準星データ取得)

測光値の較正

可視光

raw(x, y) = flat(x, y) x object(x, y) + dark(x, y) + bias(x, y)

- flat(x,y) : ピクセル間の**感度むら**, 光学系の**透過むら**
  - object(x,y)=const. の光がやってきたとしても flat(x,y)×const. が検出される。
- 近年のCCDではdarkは無視できるレベル
- biasは、raw(x, y)のオーバースキャン領域から推測 (x方向への依存は無しと仮定し)

raw(x, y) = flat(x, y) x object(x, y) + dark(x, y) + bias(x, y)

flat(x, y)

1.1	0.9	0.9
1.0	1.2	0.8
1.1	0.7	1.1

×

object(x, y)

200	204	201
198	400	202
205	199	200

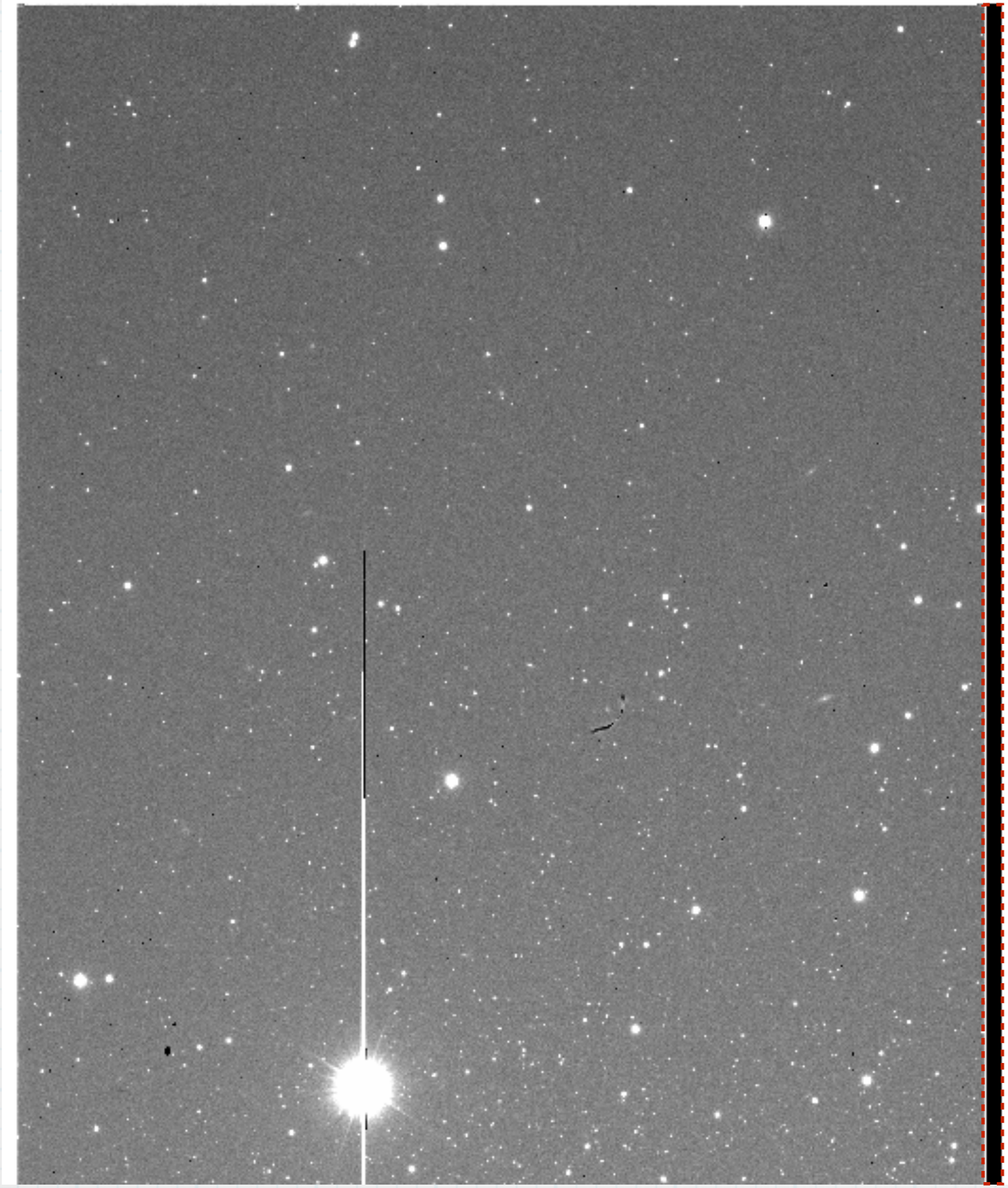
=

220	183	181
198	480	162
226	139	220

可視光

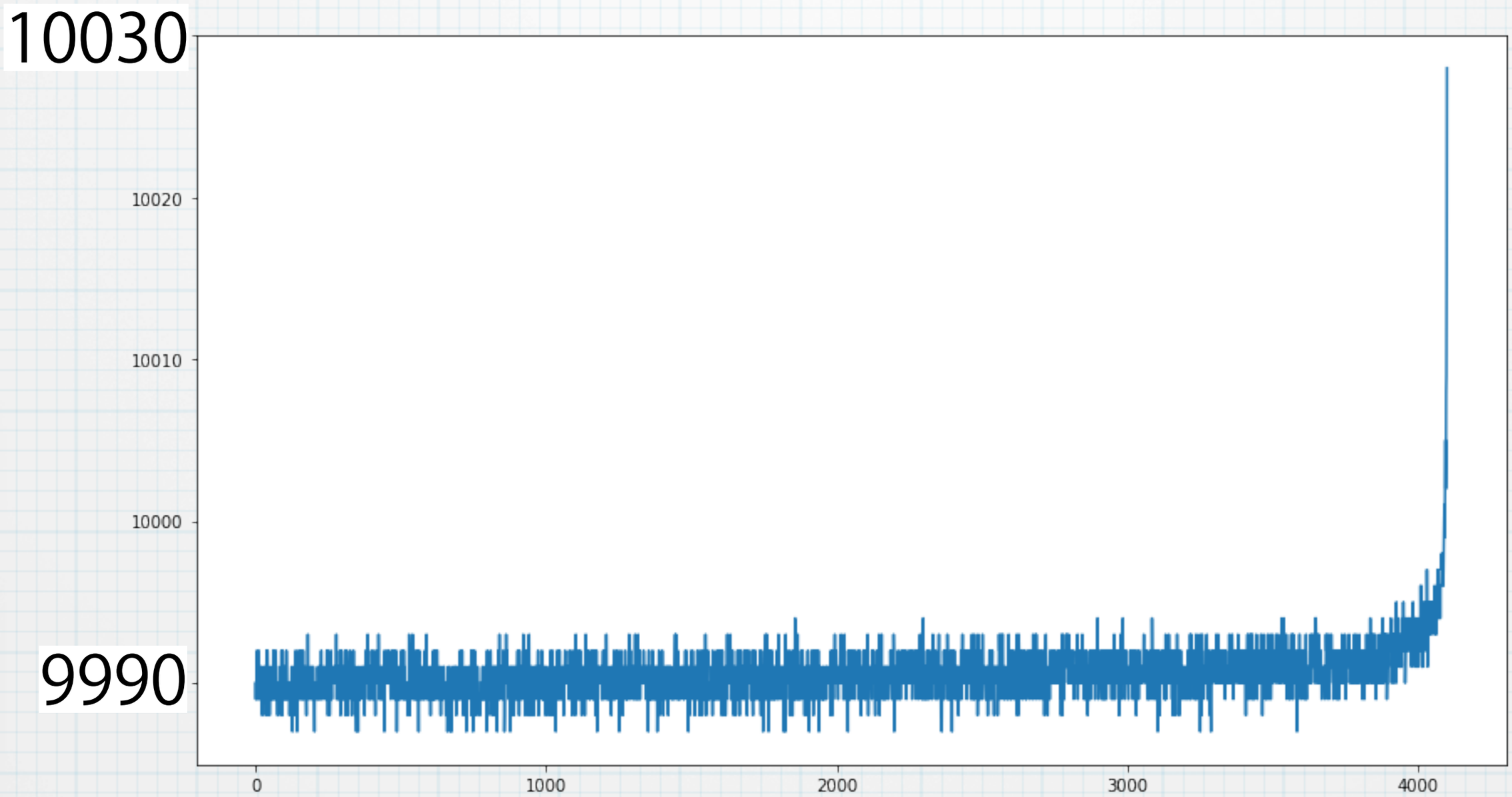
raw(x, y) = flat(x, y) x object(x, y) + dark(x, y) + bias(x, y)

- flat(x,y) : ピクセル間の**感度むら**, 光学系の**透過むら**
  - object(x,y)=const. の光がやってきたとしても flat(x,y)×const. が検出される。
- 近年のCCDではdarkは無視できるレベル
- biasは、raw(x, y)のオーバースキャン領域から推測 (x方向への依存は無しと仮定し)



ここがオーバー
スキャン領域

Y方向にも大部分で一様で、端のほうで **0.数%** 増加



フラット

$$raw(x, y) = flat(x, y) \times object(x, y) + dark(x, y) + bias(x, y)$$

object(x, y) ＝ constant (一様光)を観測してやれば、

$$flat(x, y) = \frac{raw(x, y) - bias(x, y)}{constant}$$

$$constant = median(raw(x, y) - bias(x, y))$$

規格化

1次処理

ドームフラットデータから flat(x, y)

天体生データから bias(x, y)

$$\text{object}(x, y) = \frac{\text{raw}(x, y) - \text{bias}(x, y)}{\text{flat}(x, y)}$$

iraf.imcombine()

im01.fits

101	104	102
99	100	101
103	97	100

im02.fits

105	101	99
100	99	102
101	98	96

im03.fits

98	102	105
103	102	99
105	101	103

averageでcombine

101.3	102.3	102
100.7	100.3	100.7
103	98.7	99.7

medianでcombine

101	102	102
100	100	101
103	98	100

iraf.imcombine()

im01.fits

101	104	102
99	100	101
103	97	100

im02.fits

105	101	99
100	99	102
101	98	96

im03.fits

98	102	105
103	102	99
105	101	103

```
iraf.imcombine(input='im01.fits, im02.fits, im03.fits',
               output='test.fits', combine='median')
```

combine='average'

演習2

2-1

target2を観測した、'./data1/SUPA00317885.fits' について、バイアス引き+フラット割りの処理をしましょう。これは5番フレームです。フィルターも同じBバンドなので、フラット割りには、'bflatn5.fits'が使えます。

この結果のフレームを'btarget2n5.fits'と呼ぶことにします。(後の演習で利用します)

2-2

2番フレームの生データ'./data2/SUPA00317702.fits' について、
バイアス引き+フラット割りの処理をしましょう。

先ほどの5番フレームとは違い、これは2番フレームなので、2番フレームのためのフラットを作成する必要があります。

(1) './data2/SUPA00317502.fits' を規格化したものをフラットとして作成する。
(2) './data2/SUPA003175[0-6]2.fits' から平均のフラットを作成する。
(3) 上のどちらかのフラットを使って、バイアス引き後のフラット割りを行う。

注意 : 2番フレームはオーバースキャン領域が5番とは異なる。

講習3 : 星の測光

視野の星を検出させ、 iraf.daofind()

それらをアパーチャ測光 iraf.phot()

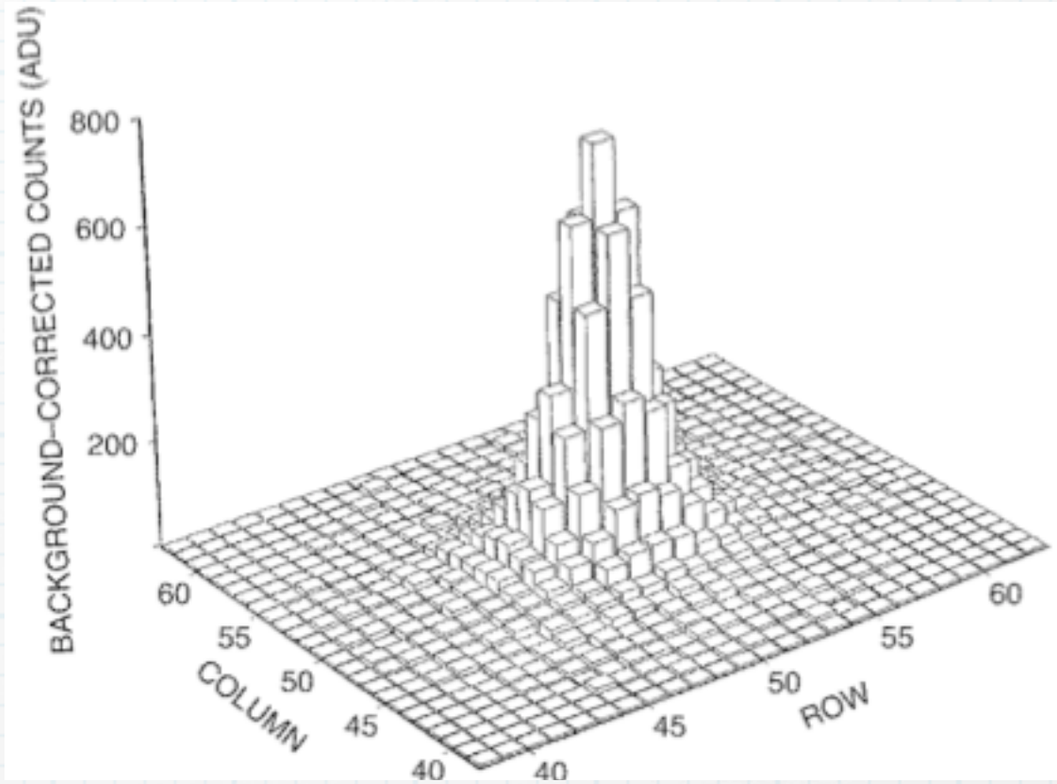
測光値の較正 ~ 標準星

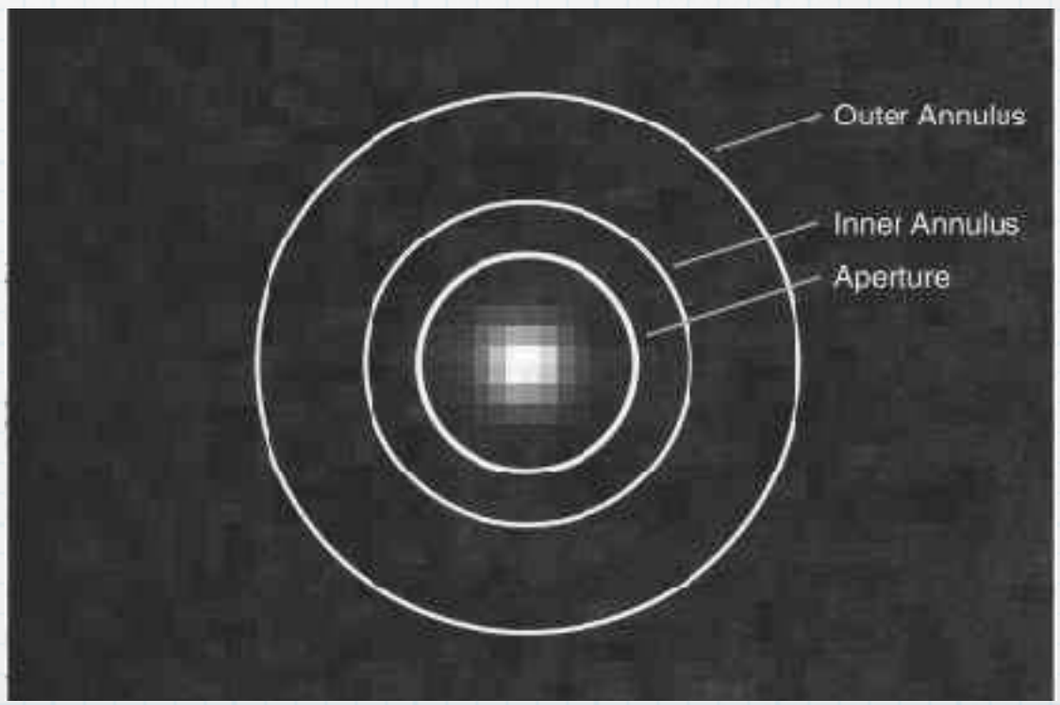
演習3

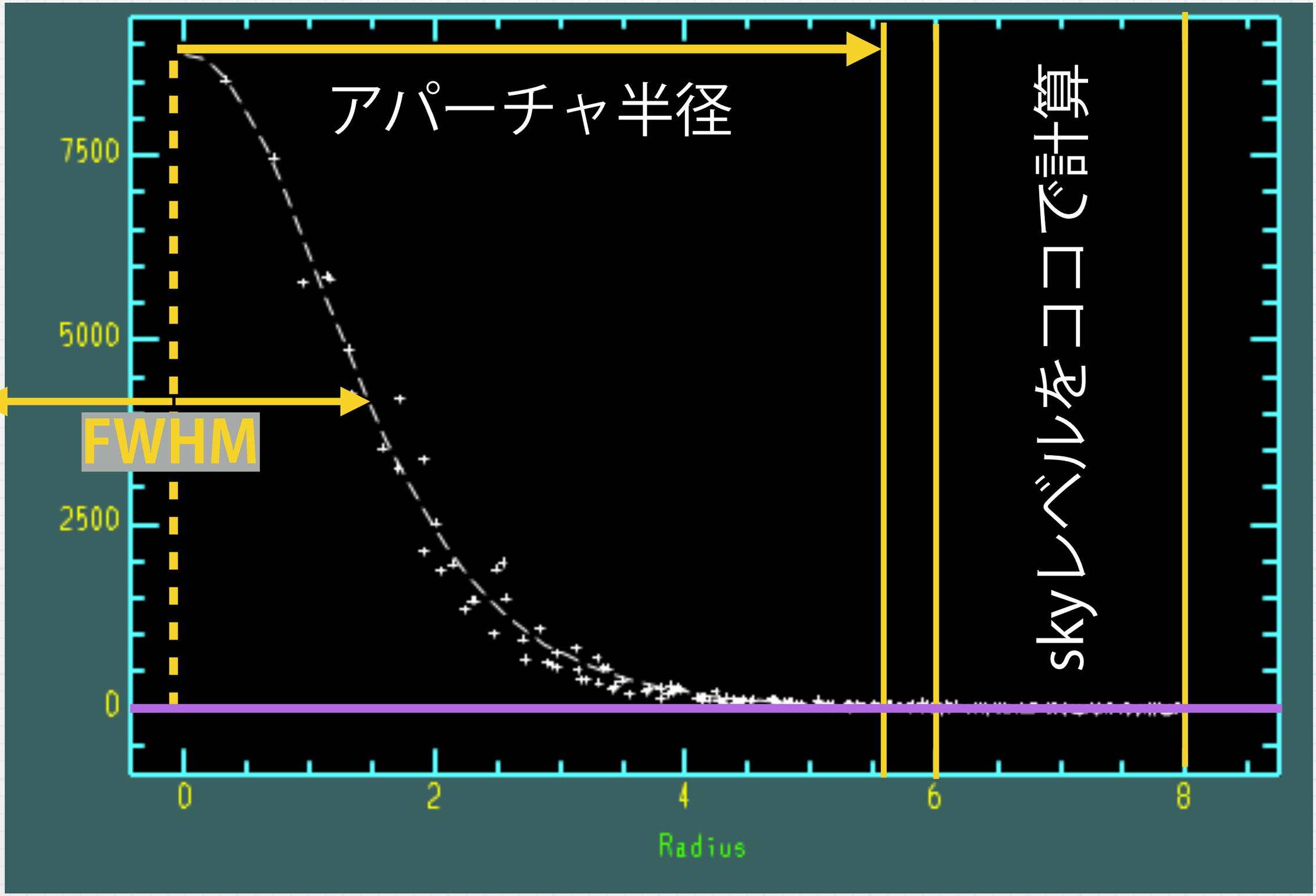
星の測光　アパーチャ測光

- 星を含む円内のカウントを総計する。
- 明るさの分かっている星のカウントと比べて、明るさを求める。

標準星の観測が必要






アパーチャ半径
FWHM
skyレベルをココで計算
7500
5000
2500
0
0
2
4
6
8
Radius

iraf.phot()のパラメータ設定

```
med = 69.  # 背景レベルとばらつき、fwhm
std = 7.4
fwhm = 7.0

iraf.apphot.unlearn()  # デフォルト値に戻しておく

iraf.apphot.datapars.datamax = 50000  # サチった星を数えない
iraf.apphot.datapars.readnoise = 10 # 検出器に特有な値
iraf.apphot.datapars.epadu = 2.5 # 検出器に特有な値
iraf.apphot.datapars.itime = 10  # 積分時間

iraf.apphot.findpars.threshold = 7  # 7シグマ以上のものを検出せよ
iraf.apphot.findpars.sharphi = 0.8  # 星っぽくないものを除くため

# fwhmで決まるパラメータ
iraf.apphot.datapars.fwhmpsf = fwhm
iraf.apphot.centerpars.cbox = max(5.0, fwhm)
iraf.apphot.fitskypars.annulus = 3 * fwhm
iraf.apphot.photpars.apertures = 2 * fwhm

iraf.apphot.fitskypars.dannulus = 10.

# 背景のレベルとばらつきで決まるパラメータ
iraf.apphot.datapars.sigma = std
iraf.apphot.datapars.datamin = med - 5 * std

iraf.apphot.photpars.zmag = 27  # 等級のゼロ点
```

測光対象ファイル

iraf.daofind(‘xxxxxx.fits’, output=‘out1.coo’)

検出結果のテキストファイル

測光対象ファイル

daofindの結果
のファイル

iraf.phot('xxxx.fits', coords='out1.coo'

output='out1.mag')

測光結果を書き出す
(テキスト)ファイル

iraf.photの
測光結果ファイル

iraf.txdump(‘out1.mag’,

fields=‘xc, yc, mag, merr’, Stdout=‘result1.txt’)

抽出する項目

この結果の
書き出し先

演習3

演習2-1で処理をしたbtarget2n5.fitsで測光をしてみましょう。このときも、オーバースキャン領域などの不要な部分を削除して行いましょう。

‘btarget2n5.fits’の視野の中には測光標準星は写っていません。ただし、上のtarget1と近い時間に観測したデータですので、等級ゼロ点は同じだと仮定し、上と同じ較正値(1.672)を使ってください。

target1とは積分時間が異なりますので注意してください。

iraf.obsutil.psfmeasure

講習4 : Numpyの基本

- 配列のベクトル演算 ~ ndarray
- numpyの関数をいくつか
- 二次元配列の注意点
- 演習4

演習4

(1) 下の二次元配列の演算をnumpyを用いて行ってください。

10	11	12
20	21	22
30	31	32

＋

100	110	120
200	210	220
300	310	320

(2) 下の3つの二次元配列のメジアンを、numpy.stack()とnumpy.median()を用いて求めてください。

10	11	12
20	21	22
30	31	32

100	110	120
200	210	220
300	310	320

20	22	24
40	42	44
60	62	64

講習5 : astropy.io.fitsの基本

- IRAFなしでFITSの処理
- getdata()でFITSの読み込み
- numpyを使って処理
- writeto()でFITSの書き出し
- 演習5

演習5

5-1. 上のmy10x10.fitsを作成し、DS9で見てxとyが反転していることを確かめてください。

5-2. astropy.io.fitsとnumpyを使って以下の処理をしてください。

(1) ‘./data2/SUPA00317502.fits'から2番フレーム用のフラットを作成。

(2) './data2/SUPA00317882.fits' について、バイアス引き+フラット割りの処理

(3) trimmingして(2)の結果からオーバスキャン部をとりのぞく。

講習6 : matplotlibの基本

- ヒストグラム (光度関数)
- X-Y プロット (等級 vs 等級エラー)
- FITS画像の表示
- FITS画像にオーバープロット
- 演習6

演習6

演習3で行った'btarget2n5.fits'の測光結果を用いて、

(1) 「光度関数のヒストグラム」と 「等級vs等級エラーのプロット」を作成してください。

(2) FITS画像をnotebookに表示して、そこに測光した星をプロットしてください。