IRAF: Riduzione dati Spettroscopici longslit

A.A. 2013/14 Versione 2.0

A. Pizzella

Contents

1.1 Introduzione 2
1.2 Bias 3
1.3 Flatfield 4
1.4 Ccdproc 6
1.5 Calibrazione in lunghezza d'onda 6
1.6 Rimozione raggi cosmici 9
1.7 Sottrazione del cielo 10
1.8 Calibrazione in flusso 11

Questa breve dispensa ha un duplice scopo. Da un lato si descrive la procedura di riduzione di dati spettroscopici a fentidura lunga (longslit) utilizzando il programma di riduzione IRAF. Contemporeamente, prendendo spunto dalla riduzione stessa, vengono spiegate alcuni concetti importanti per l'utilizzo dei dati. Queste informaizoni appaiono come note aggiuntive man mano che si procede nella riduzione dei dati.

1.1 Introduzione

Il segnale che leggiamo sul CCD si può rappresentare come una matrice di conteggi $ADU(x, y)$ (ADU=Analog Digital Units oppure DN=Data Numbers). Ovvero abbiamo una indicazione dei conteggi presenti nelle varie posizioni del piano focale dove è posto il CCD. $ADU(x, y)$ non coincide però con $I(x, y)$ che rappresenta il segnale intrinseco nel piano del cielo. Le ADU che noi vediamo sono

$$ADU(x, y) = I(x, y) \times FF(x, y) + BIAS(x, y,) + DARK(x, y).$$

DARK rappresenta le correnti parassite che si generano nel CCD a causa principalmente dell'agitazione termica, BIAS è un livello di punto zero introdotto dall'elettronica che legge il CCD, FF indica la risposta dei singoli pixel del CCD che può variare da pixel a pixel anche a causa delle ottiche, polvere, etc.

La prima fase della riduzione dati consiste nel determinare $I(x, y)$ a partire da $ADU(x, y)$. In questa dispensa trascuriamo l'effetto del DARK in quanto nel CCD che abbiamo utilizzato è trascurabile. Abbiamo quindi che

$$I(x, y) = (ADU(x, y,) - BIAS(x, y,))/FF(x, y)$$

IRAF applica le correzioni di BIAS e FF con il comando `ccdproc`. Per poter utilizzare questo comando è però necessario aver ottenuto le immagini BIAS e FF da utilizzare.

1.2 Bias

Le immagini di bias sono ottenute con 0 secondi di integrazione a shutter chiuso e luci spente in modo che non vi sia alcuna luce che raggiunge il CCD. Serve per misurare il livello di zero del CCD. Questo livello andrà sottratto a tutti i frame come primo passo della riduzione dati. Per evitare di aumentare il rumore del nostro frame scientifico, l'immagine di bias che utilizzeremo dovrà essere il meno rumorosa possibile. Per fare questo combiniamo assieme più bias in modo da ottenere un *masterbias*. Il rumore si somma quadraticamente

$$Noise_{1+2} = \sqrt{Noise_1^2 + Noise_2^2}$$

Quindi se $Noise_1 >> Noise_2$ allora $Noise_{1+2} \sim Noise_1$. Nel caso dei Bias, se calcoliamo la media di 10 bias, il masterbias avrà un rumore

$$Noise_{mbias} = Noise_{bias}/\sqrt{10}$$

e quando andro' a sottrarre il masterbias al mio frame scientifico avrò

$$Noise_{s-mb} = \sqrt{Noise_s^2 + Noise_{bias}^2/10}$$

Dato che $Noise_{bias}$ non è altro che il R.O.N. (read out noise) del CCD e che $Noise_S$ è almeno pari al RON (sarà in realtà più alto perché si aggiunge il rumore poissoniano dei fotoni) allora trovo che $Noise_{s-mb} \sim Noise_s$. In definitiva, già 10 bias sono sufficienti (il che non vieta di prenderne di più).

Si utilizza il comando `zerocombine`. Per utilizzarlo bisogna entrare nei package `noao imred ccdred`. Il comando calcola la media o la mediana delle immagini di input pixel per pixel. Il risultato sarà quindi una immagine delle stesse dimensioni di quelle di partenza. In ogni pixel conterrà la media (o mediana) dei valori di quello stesso pixel nelle immagini di input.

I parametri più importanti da impostare (con `epar zerocombine`) sono

iinput = " filelist"	List of zero level images to combine
(output = "masterbias")	Output zero level name
(combine = "average")	Type of combine operation
(reject = "minmax")	Type of rejection
(ccdtype = " ")	CCD image type to combine
(nlow = 1)	minmax: Number of low pixels to reject
(nhigh = 1)	minmax: Number of high pixels to reject

In questo modo

- @filelist . *filelist* è un file di testo che contiene riga per riga le immagini di bias da combinare assieme
- stiamo calcolando la media (in alternativa potrei calcolare la mediana) dei frames
- inibisco il controllo sul descrittore *ccdtype*
- utilizzo il metodo *minmax* per eliminare i pixel affetti da raggi cosmici
- in combinazione con *minmax* elimino 1 pixel *high* (cioè il vlaore più alto tra quelli dello stesso pixel) e 1 pixel *low* (cioè il valore più basso tra quelli dello stesso pixel).

Per esplorare le altre possibilità/opzioni del comando zerocombine conviene utilizzare l'*help* in linea di IRAF: `help zerocombine`.

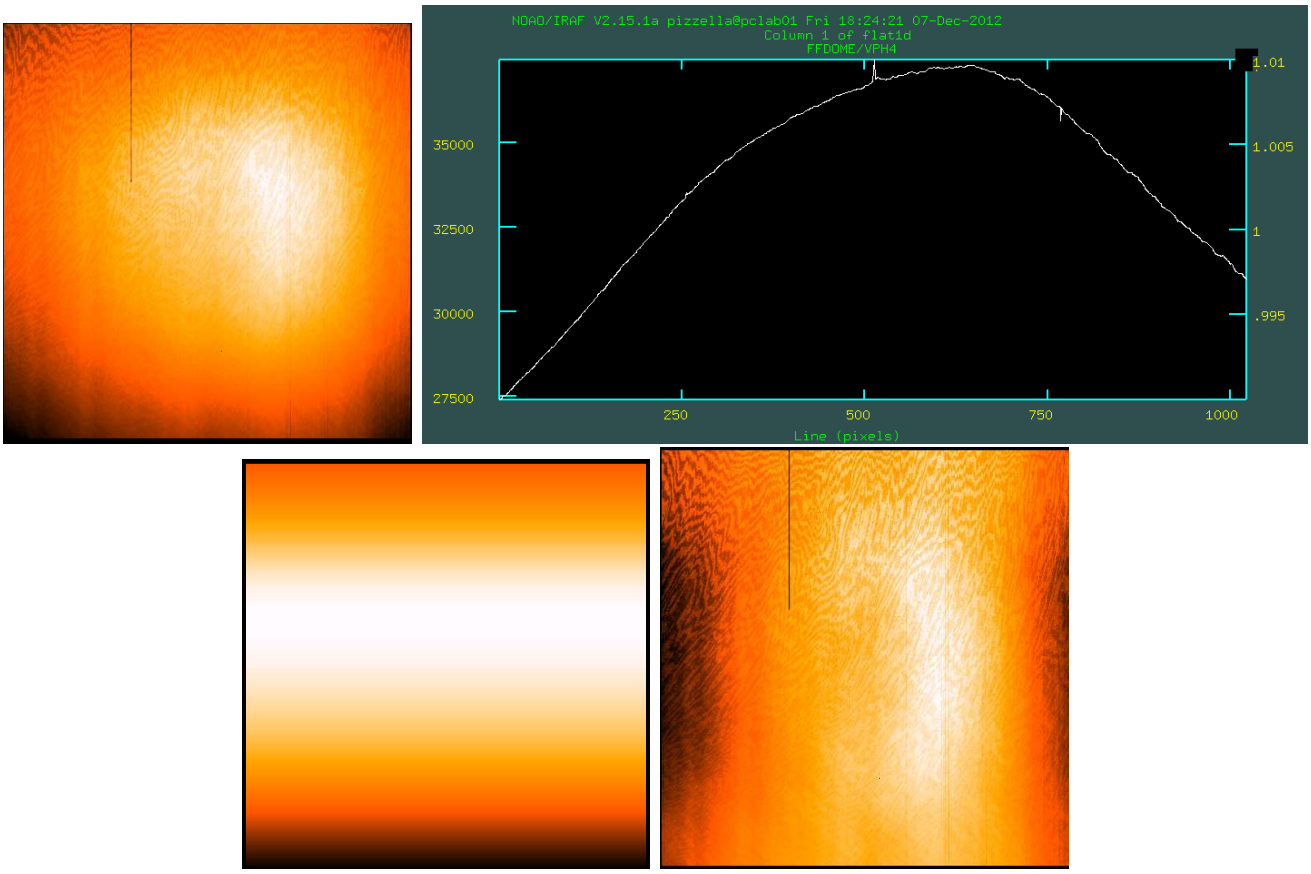

Figure 1.1: Da sinistra in alto verso destra in basso: *flatvph4* ottenuto come media delle singole immagini di flat field; *flat1d* ovvero il profilo del FF ottenuto mediando tutte le colonne di *flatvph4*; *flat2d*, ovvero *flad1d* replicato orizzontalmente in modo da avere le stesse dimensioni di *flatvph4*; *masterflatnorm* ovvero il rapporto tra *flatvph4* e *flad2d*.

1.3 Flatfield

Le immagini di *flatfield* servono per correggere le disomogeneità di sensibilità dovute allo strumento (grism, fenditura, CCD, etc.). Per evidenziare queste disomogeneità si prende l'immagine di un campo completamente uniforme utilizzando lo schermo apposito in cupola ed illuminandolo con una lampada al quarzo che non presenta righe di emissione e risulta quindi il più possibile omogenea in lunghezza d'onda.

Si utilizza il comando `flatcombine`. Per utilizzarlo bisogna entrare nei package noao imred ccdred. Il comando è simile nella filosofia a *zerocombine*. Il risultato sarà quindi una immagine delle stesse dimensioni di quelle di partenza. In ogni pixel conterrà la media (o mediana) dei valori di quello stesso pixel nelle immagini di input. Anche nel caso dei FF si vuole ottenere una immagine dal rumore basso in quando dovremo usarla per dividere l'immagine scientifica di cui non vogliamo aumentare il livello di rumore. Nel caso dei FF però piuttosto che prenderne tanti conviene ottenerli con un livello di segnale più alto possibile in modo da avere il rapporto S/N più alto possibile. Il numero di frames da combinare non sarà quindi dell'ordine delle decine ma più basso.

I parametri più importanti da impostare (con `epar zerocombine`) sono

input = "@filelist "	List of flat field images to combine
(output = "flatvph4.fits")	Output flat field root name
(combine = "median")	Type of combine operation
(reject = "none")	Type of rejection
(ccdtype = " ")	CCD image type to combine
(process = yes)	Process images before combining?
(scale = "mode")	Image scaling
(statsec =)	Image section for computing statistics

In questo modo

- @filelist . *filelist* è un file di testo che contiene riga per riga le immagini di flat da combinare assieme
- stiamo calcolando la mediana
- inibisco il controllo sul descrittore *ccdtype*
- non effettuo alcuna rejection sui raggi cosmici
- processo le immagini prima di combinarle. Dobbiamo infatti sottrarre il bias e tagilare i bordi dove il segnale è assente (trimming)
- avremmo potuto (ma non lo stiamo facendo) riscalare le immagini in intensità prima di combinarle per riportarle, ad esempio, allo stesso valore mediano

anche qui `help flatcombine` permette di vedere tutte le possibili opzioni del comando.
IMPORTANTE:
Dato che abbiamo specificato `process = yes` dobbiamo entrare nei parametri del comando `ccdproc` e settarli opportunamente. Quindi `epar ccdproc` e poi

(trim = yes)	Trim the image?
(zerocor = yes)	Apply zero level correction?

e più avanti

(trimsec = "[40:1040,*]")	Trim data section
(zero = "masterbias.fits")	Zero level calibration image

Di fatto stiamo impostanto il comando `ccdproc` (che vedremo più avanti) in modo che corregga per il bias (o correzione di zero) utilizzando l'immagine *masterbias.fits* e che ritagli le immagini. D'ora in poi verrà usata solo la regione centrale definita [40:1040,*]. Secondo la sintassi IRAF [40:1040,*] significa dal pixel x=40 al pixel x=1040 e dalla y iniziale a quella finale (tutto l'intervallo in y).

Una volta ottenuto il file *flatvph4.fits* (vedi Fig. 1.1) dobbiamo normalizzarlo. La normalizzazione va fatta ad ogni singola lunghezza d'onda e cioè per ogni linea y. Questo perché lungo y abbiamo la direzione della dispersione in lunghezza d'onda del grism. L'illuminazione lungo la direzione y non è omogenea. Dipende infatti dallo spettro della lampada al quarzo che stiamo usando per il FF e dall'efficienza del grism. Non possiamo quindi confrontare linee con diversi valori di y. Dobbiamo allora, per ogni fissata y, dividere il FF per il valore medio dei conteggi a quella stessa y. Sembra complicato ma si può ottenere questo risultato con 3 comandi: `blkavg`, `blkrep` e `imarith`.

blkavg uso il comando `blkavg flatvph4 flat1d 1001 1`. In generale questo comando media il file di input *flatvph4* a blocchi di pixel rettangolari. Nel nostro caso noi vogliamo mediare tutto il file *flatvph4* lungo l'asse x e quindi considero rettangoli larghi come tutta l'immagine[1] e alti 1 pixel. L'output `flat1d` è quindi un file monodimensionale lungo 1023 ottenuto mediando *flatvph4* per ogni y fissata.

[1]con il comando `imhead flatvph4` ottengo l'indicazione delle dimensioni del file flatvph4 e cioè [1001,1023]

blkrep con il comando `blkrep flat1d flat2d 1001 1` creo il file bidimensionale *flat2d* replicando lungo l'asse x 1001 volte e lungo l'asse y una volta (cioè non replico) il file *flatvph4*. *flat2d* ha quindi dimensione [1001,1023] come *flatvph4* e per ogni valore fissato di y ha lo stesso valore.

imarith ora non rimane altro che dividere *flatvph4* per *flat2d* con il comando `imarith flatvph4 / flat2d masterflatnorm`. il file *masterflatnorm* è il risultato. È il mio FF normalizzato. Ha un valore medio pari ad 1. In Fig.1.1 vi sono mostrati i risultati dei vari passi per la creazione di *masterflatnorm*.

1.4 Ccdproc

Ora che ho le immagini del BIAS e del FF posso eseguire la prima fase di data processing utilizzando il comando `ccdproc`. Con e`epar ccdrpoc` imposto i parametri. Devo avere:

(ccdtype = " ")	CCD image type to combine
(trim = yes)	Trim the image?
(zerocor = yes)	Apply zero level correction?
(flatcor = yes	Apply flat field correction?
(trimsec = "[40:1040,*]")	Trim data section
(zero = "masterbias.fits")	Zero level calibration image
(flat = masterflatnorm.fits)	Flat field images

A questo punto posso eseguire il comando `ccdproc @dati.list out=@datip.list` dove *dati.list* e *datip.list* contengono la lista dei file in input e output. I file di input verranno trimmati, sottratti di bias e divisi per FF. L'operazione va fatta sui frame scientifici, sui frame con gli spettri delle lampade di confronto e sui frame delle standard spettrofotometriche.

1.5 Calibrazione in lunghezza d'onda

la filosofia con cui viene seguita la calibrazione in lughezza d'onda è la seguente. Devo avere a disposizione un frame preso nelle stesse condizioni e con lo stesso set-up strumentale dell'immagine scientifica da calibrare. Nel frame devono essere presenti righe spettrali in emissione con un buon segnale e dalla lunghezza d'onda nota con precisione. Generalmente è uno spettro di una lampada tipo Neon, Helio, Argon, Ferro, Torio etc. preso subito prima e/o subito dopo l'immagine scientifica. O anche contemporaneamente come succede se si utilizzano le righe di emissione del cielo notturno. Le righe, una volta identificate, vengono utilizzate per creare una mappa che permette di passare da pixel x,y a lunghezza d'onda (nel nostro caso x, λ). Con IRAF sono previsti 4 comandi da utilizzare in sequenza: `identify`, `reidentify`, `fitcoord` e `transform`.

identify; Il comando va utilizzato nel task `noao twodspec longslit`. I parametri importanti da impostare sono

images = "AF132045p.fits"	Images containing features to be identified
(section = "middle column")	Section to apply to two dimensional images
(coordlist = "neon_lines.dat")	User coordinate list

con cui indichiamo di iniziare dalla colonna centrale, che l'immagine con lo spettro della lampada è AF132045p.fits e che i valori delle lunghezze d'onda della nostra lampada sono nel file *neon_lines.dat*. Eseguiamo il programma in modo interattivo. Per prima cosa ci viene mostrato lo spettro lungo la colonna centrale (Fig. 1.2 in alto). Dobbiamo identificare le righe spettrali aiutandoci con l'atlante delle righe presente nel manuale. Una volta riconosciuta una riga si va con il cursore sulla riga e si preme il tasto `m` (mark). Ci viene chiesta la lunghezza d'onda che

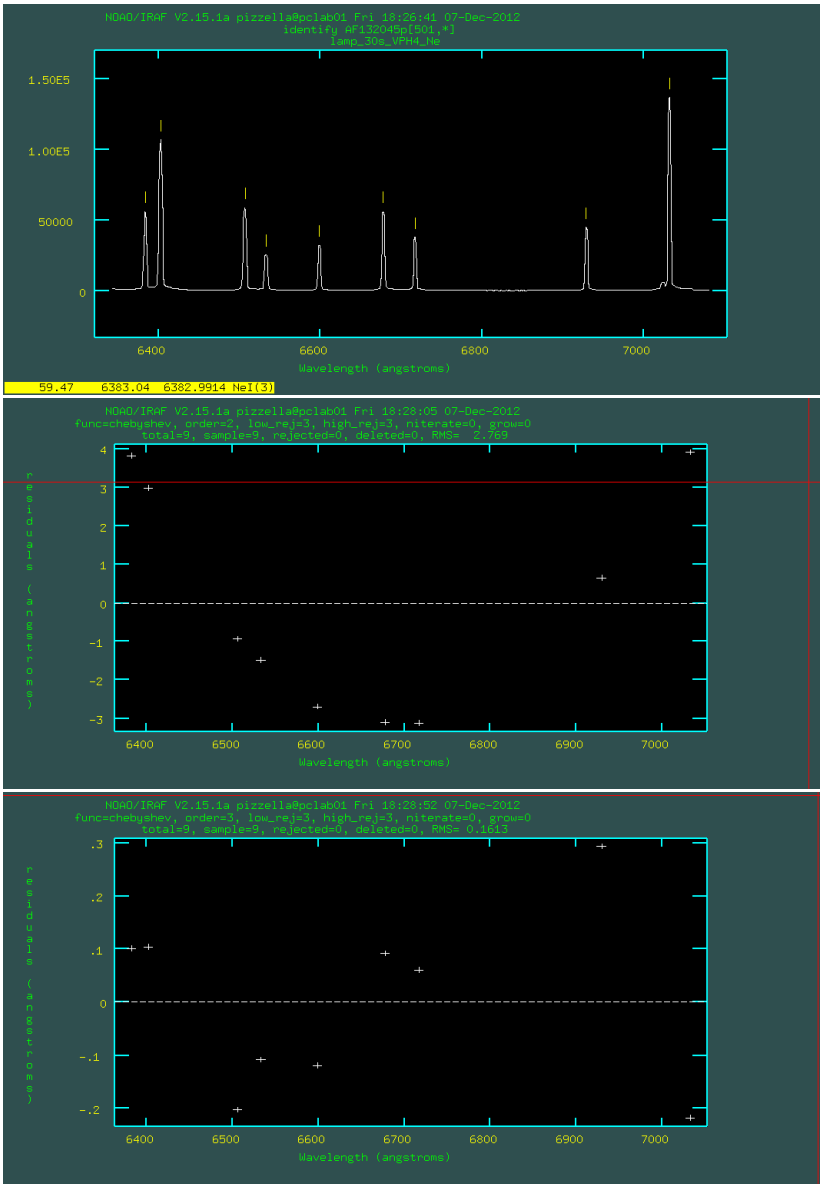

Figure 1.2: comando `identify`. Dall'alto verso il basso: La prima identificazione delle righe; il fit con un polinomio di Chebjshev di grado 2; il fit con un polinomio di Chebjshev di grado 3.

noi scriviamo a mano. L'operazione viene ripetuta 3-4 volte identificando, in genere, una riga al centro e due righe alle due estremità dello spettro. A questo punto, premendo il tasto `l` IRAF, riconosce, a partire dalle righe che abbiamo dato, tutte le altre (Fig. 1.2 nel centro). Se soddisfatti entriamo in modalità *fit* con il comando `f`. Scopo di questa fase è definire una funzione $\lambda = f(x)$ in modo da poter assegnare ad ogni x una valore di lunghezza d'onda (ricordo per chiarezza che nel nostro caso in realtà la lunghezza d'onda è lungo y). `identify` offre diverse possibilità per la funzione f(x). con il comando `:func` si può cambiare funzione. Tipicamente si usano delle funzioni di tipo spline oppure polinomi. Avendo noi poche righe (una decina) preferiamo utilizzare un polinomio di grado abbastanza basso. Tipicamente un polinomio di Chebyshev di ordine 3 (che corrisponde ad un polinomio di secondo grado). Dando appunto il comando `f` la schermata ora ci mostra lungo l'asse x la lunghezza d'onda, lungo y lo scarto dalla funzione lineare $\lambda = a_0 + xa_1$. Utilizzando un grado 3 vediamo che possiamo interpolare i dati abbastanza bene. (Fig. 1.2 in basso). Quando siamo soddisfatti terminiamo il comando con `q` e poi nuovamente `q`. La soluzione trovata viene scritta nel direttorio *database*.

reidentify Dopo aver eseguito l'identify sulla colonna centrale dobbiamo mappare tutto il nostro frame. Essenzialmente dobbiamo ripetere l'operazione per colonne diverse da quella centrale. Il comando *reidentify* lo fa automaticamente a partire dalla soluzione trovata con *identify*. Bisogna impostare i parametri come

reference = "AF132045p"	Reference image
images = "AF132045p"	Images to be reidentified
(override = no)	Override previous solutions?
(refit = yes)	Refit coordinate function?
(trace = yes)	Trace reference image?
(step = "10")	Step in lines/columns/bands for tracing an imag
(nsum = "10")	Number of lines/columns/bands to sum
(nlost = 0)	Maximum number of features which may be lost
(cradius = 5.)	Centering radius
(threshold = 0.)	Feature threshold for centering
(addfeatures = yes)	Add features from a line list?
(coordlist = "neon_lines.dat")	User coordinate list

notare come reidentifichiamo il frame con se stesso. La soluzione trovata viene aggiunta nel direttorio *database*.

fitcoord ora che abbiamo una mappa con la posizione della lunghezza d'onda in funzione di x e anche al variare di y è necessario integrare queste informazioni assieme per ottenere una funzione che per ogni valore in pixel (x,y) ci dia un valore in lunghezza d'onda (x,λ). Il comando *fitcoord* permette di fare questo in maniera interattiva eliminando eventuali righe che non funzionano bene. In particolare si può vedere il risultato dell'interpolazione plottando nel grafico i residui in funzione della lunghezza d'onda (Fig. 1.3 in alto). Il modo con qui si può esaminare e controllare il fit

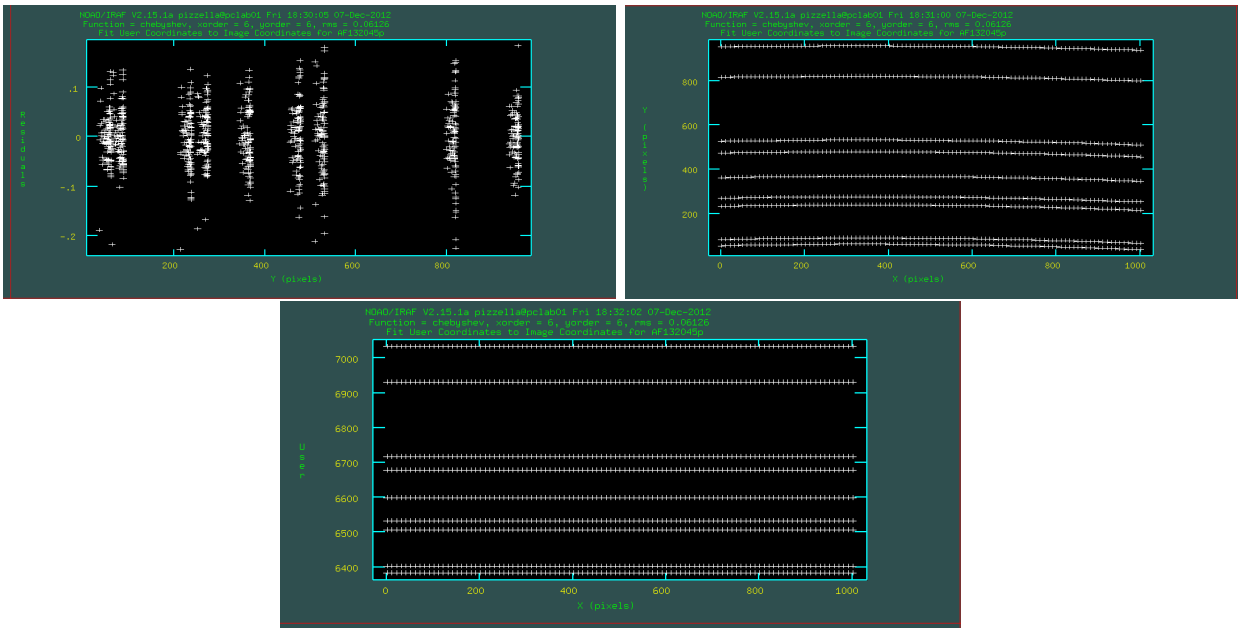

Figure 1.3: comando `fitcoord`. Dall'alto a sinistra verso il basso: Prima schermata del comando. In asse x è mostrata la coordinata y (e cioè quella lungo la lunghezza d'onda) ed in asse y il residuo in Angstrom del fit; stesso fit ma mostrando lungo x la coordinata x del frame e lungo y la posizione delle righe spettrali individuate da `reidentify`; stesso fit mostrando la coordinata x del frame in x e la lunghezza d'onda del fit in y.

è il seguente. Con la finestra grafica selezionata, premere `x` per cambiare la quantità graficata lungo l'asse x del plot. Dopo aver prenuto x viene chiesto cosa si vuole mostrare lungo l'asse x. Le possibili risposte sono il residuo (`r`), la coordinata x (`x`), la coordinata y (`y`), la lunghezza d'onda fittata (`s`). Stessa cosa per selezionare cosa si vuole lungo l'asse y. Premere `y` e poi scegliere tra `x`,`y`,`r` e `s`. Per esempio i grafici in Fig. 1.3 sono stati ottenuti con `yx` e `yr`, `xx` e `yy` ed infine `xx` e `ys`. Una vole cabiate le coordinate premere `r` per riaggiornare il grafico oppure `f` per ripetere il fit e riplottarlo. Volendo, con il comando `d` si possono selezionare i punti e cancellarli. Si va quindi sul punto da cancellare con il cursore e si preme `d` e poi, a scelta `p`, `x`, `y` o `z` a seconda che si voglia cancellare solo quel punto, o i punti con l astessa x, con la stessa y o con la stessa z. Quando si è soddisfatti con `q` si esce dal comando `fitcoord` e si salva il risultato dell'interpolazione come file di testo nel direttorio *database* con il nome *fcAF132045p*.

transform Una volta ottenuto il file *fcAF132045p* che contiene la calibrazione questo viene utilizzato grazie al comando `transform`. Questo comando effettua un *rebinning* del frame. Il frame viene cioè ricampionato. Questa operazione può essere delicata in quanto l'immagine viene interpolata per poter determinare il valore che avranno i pixel nella nuova griglia. Nel caso dei nostri spettri abbiamo visto che è importante utilizzare

(interptype = "linear") Interpolation type

per ottenere un buon risultato. Il comando può essere dato tramite epar

input = AF132045p Input images
output = AF132045pw Output images
fitnames = AF132045p Names of coordinate fits in the database

oppure in linea:

`transform AF132045p AF132045pw AF132045p` Il comando chiede, interattivamente, la direzione dell'asse della dispersione. Se lo spettro appare verticale come nel nostro caso bisogna scrivere "2". Trasformando l'immagine della lampada di confronto posso controllare se la calibrazione è venuta bene ovvero se le righe appaiono perfettamente dritte e perpendicolari alla direzione della dispersione. Se voglio calibrare lo spettro della galassia basta tare il comando transform. Ad esempio: `transform AF132044p AF132044pw AF132045p` e poi battere "2". Un esempio è riportato in Fig.1.4. Nel caso della galassia si può notare come le righe del cielo, che prima presentavano una curvatura, sono ora perfettamente dritte. Se non lo sono la procedura di calibrazione non è andata a buon fine.

1.6 Rimozione raggi cosmici

Una maniera efficacie per eliminare i raggi cosmici è utilizzare il programma *lacos_spec.cl* Una volta scaricato il programma e messo nel proprio direttorio è necessario definire il nuovo *task* in IRAF con il comando

```
task lacos_spec = lacos_spec.cl
```

A questo punto il nuovo comando è diventato un comando come gli altri. Si può editare i parametri con `epar lacos_spec` e settare i parametri in modo appropriato, ed in particolare le informazioni su il gain e il RON. Il programma, oltre all'immagine pulita dai racci cosmici, genera anche una maschera (con estenzione .pl) che contiene la mappa di tutti i pixel dove sono stati trovati raggi cosmici e che sono stati conseguentemente modificati dal programma. Volendo è possibile sottrarre anche le righe di emissione del cielo ma preferiamo farlo successivamente con il comando `background` (Sec.1.7). Per evitare che venga sottratto il cielo è necessario settare i parametri xorder e yorder a 0. Questa routine può impiegare alcuni minuti per completare una immagine. In Fig.1.5 mostriamo un dettaglio dello spettro della galassia prima e dopo la correzione per raggi cosmici.

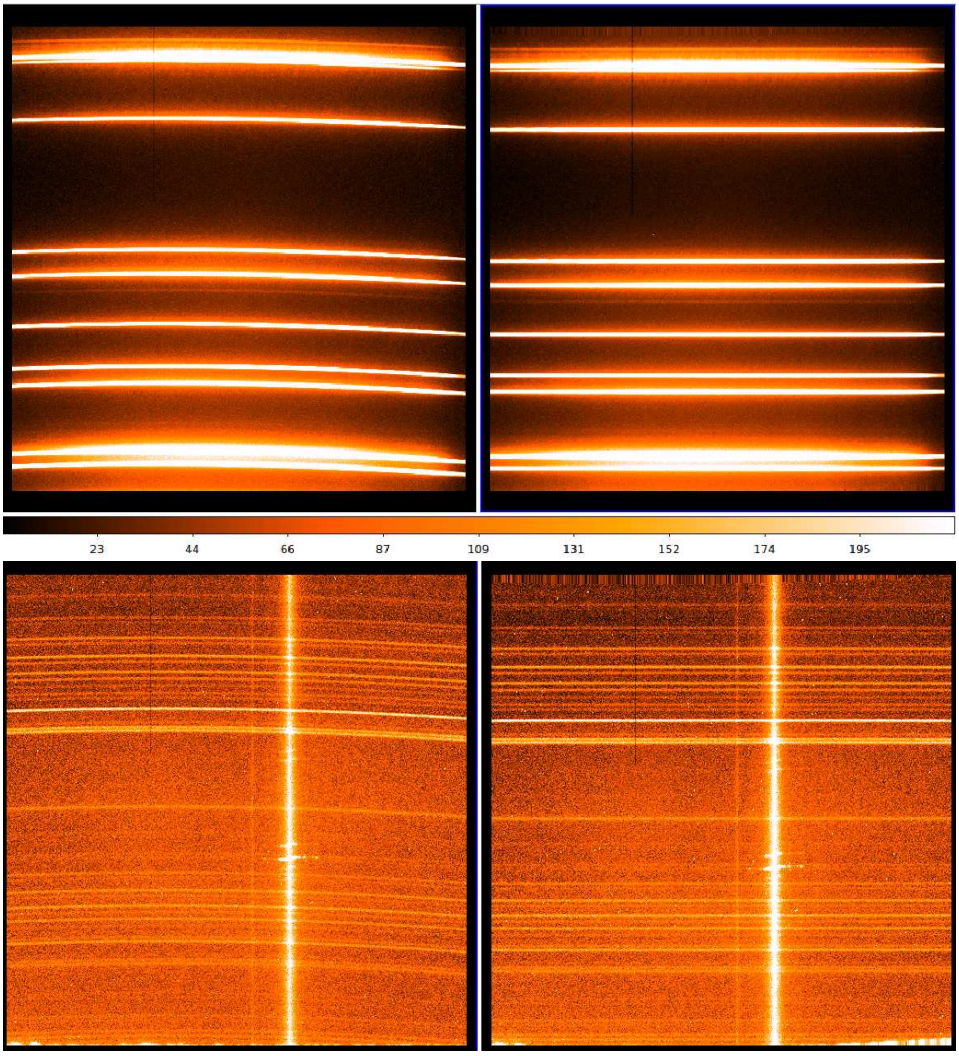

Figure 1.4: In alto: spettro di confonto prima (sinistra) e dopo (destra) la calibrazione in lunghezza d'onda ottenuta con il comando `transform`. Si può notare come le righe di emissione della lampada da curve sono diventate rettilinee. In basso: come sopra ma per lo spettro della galassia.

1.7 Sottrazione del cielo

La sottrazione dello spettro del cielo notturno viene eseguita con il comando `background`. Il comando interpola, per ogni fissata lunghezza d'onda, una funzione (come sempre si può scegliere tra i polinomi di Chebyshev e Legandre oppure una funzione di tipo spline o spline3, e relativo grado). È chiaro che il comando va usato dopo aver rettificato le righe del cielo e cioè dopo aver eseguito la calibrazione in lunghezza d'onda. Un esempio su come utilizzare il comando può essere

input = "AF132044pwc"	Input images to be background subtracted
output = "AF132044pwcs"	Output background subtracted images
(axis = 1)	Axis along which background is fit and subtracted
(interactive = yes)	Set fitting parameters interactively?
(function = "chebyshev")	Fitting function
(order = 3)	Order of fitting function
(low_reject = 2.)	Low rejection in sigma of fit
(high_reject = 3.)	High rejection in sigma of fit

Il primo input che viene chiesto interattivamente è la linea (o colonna) da utilizzare in partenza. Si può utilizzare il display ds9 per scegliere una regione opportuna del frame. Poi viene mostrata la

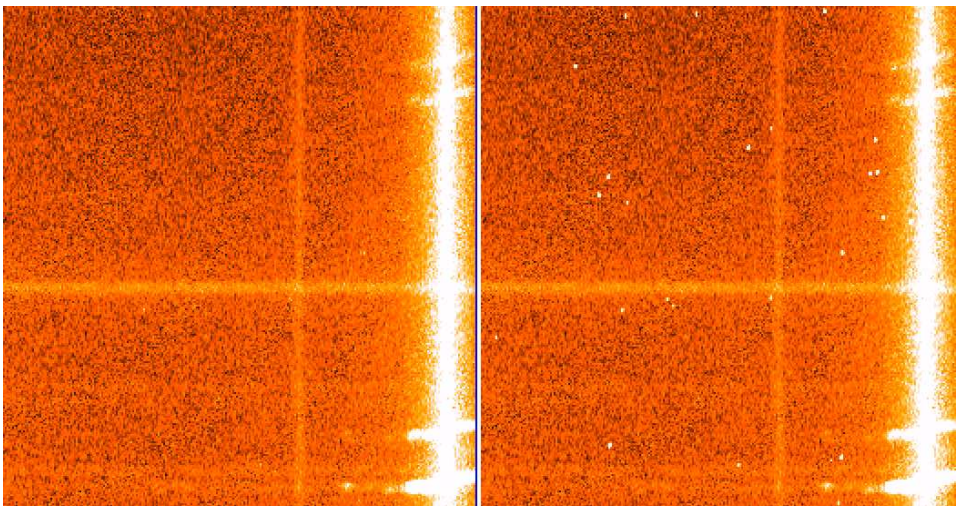

Figure 1.5: Pulizia dai raggi cosmici. A destra il frame prima della pulizia, a sinistra il frame dopo la pulizia dai raggi cosmici.

linea scelta con sopra il fit. È possibile cambiare l'ordine del polinomio (ad esempio `:ord 5`). È poi possibile scegliere una regione da utilizzare per il fit escludendo, ad esempio, le colonne dove si trova la galassia (Fig.1.6 in alto). Bisogna infatti evitare di rimuovere, assieme alle righe di emissione del cielo, il segnale proveniente dal nostro oggetto. Per fare questo si preme il tasto `s` 2 volte per ogni intervallo. Per eseguire il fit come in Fig.1.6 abbiamo premuto il tasto `s` 4 volte, 2 per definire l'intervallo da 10 a 530 e 2 per l'intervallo da 680 a 980. da Quando si è soddisfatti del fit si esce con `q`. Il fit che avviamo fatto per una linea (lunghezza d'onda) specifica viene ripetuto per tutte le linee dell'immagine. Viene creato automaticamente un file fits con il frame sottratto dal cielo. Se si vuole avere il frame che contiene solo il cielo basta fare
`imarith AF132044pwc - AF132044pwcs skyAF132044`.

1.8 Calibrazione in flusso

Per calibrare in flusso uno spettro è necessario aver preso lo spettro di una stella standard spettrofotometrica. Si tratta di stelle delle quali è disponibile una tabella che riporta, a varie lunghezze d'onda, il flusso espresso tipicamente in erg $s^{-1}cm^{-2}Å^{-1}$ e il tipico ordine di grandezza è 10^{-15}. Dal rapporto tra i conteggi ottenuti alle varie lunghezze d'onda e i flussi della stella a quelle stesse lunghezze d'onda si ricava la curva di risposta dello strumento. Dal punto di vista pratico sono necessari alcuni passaggi. È necessario estrarre lo spettro della stella, ricampionare lo spettro da noi osservato per riportarlo allo stesso campionamento della tabella con i flussi, confrontare la tabella e le osservazioni in modo da tenere conto del diverso set-up strumentale che puòrtare ad una curva di risposta non adeguata. I comandi che si usano sono `apall`, `standard` `sensfunc` ed infine `calibrate`.

apall È un comando che si articola in due fasi. Nella prima fase si definiscono le *aperture* da estrarre. Nella seconda fase si determina la posizione dell'apertura al variare della lunghezza d'onda. Nel caso nostro effettivamente abbiamo solo una stella per cui la prima fase può sembrare un po' inutile. Ma non lo sarebbe nel caso di uno spettro di tipo echelle oppure nel caso in cui vi fossero più stelle in fenditura. Come mostrato in Fig.1.7, nella prima fase apall mostra il profilo della luce lungo la fentitura. Qui si può vedere la posizione della stella lungo la fenditura. Si possono usare i tasti `l` e `u` per modificare l'ampiezza dell'apertura. Quando si è soddisfatti si termina la fase 1 con `q`. Nella fase 2 viene mostrato il *trace* dello spettro lungo le colonne. Viene

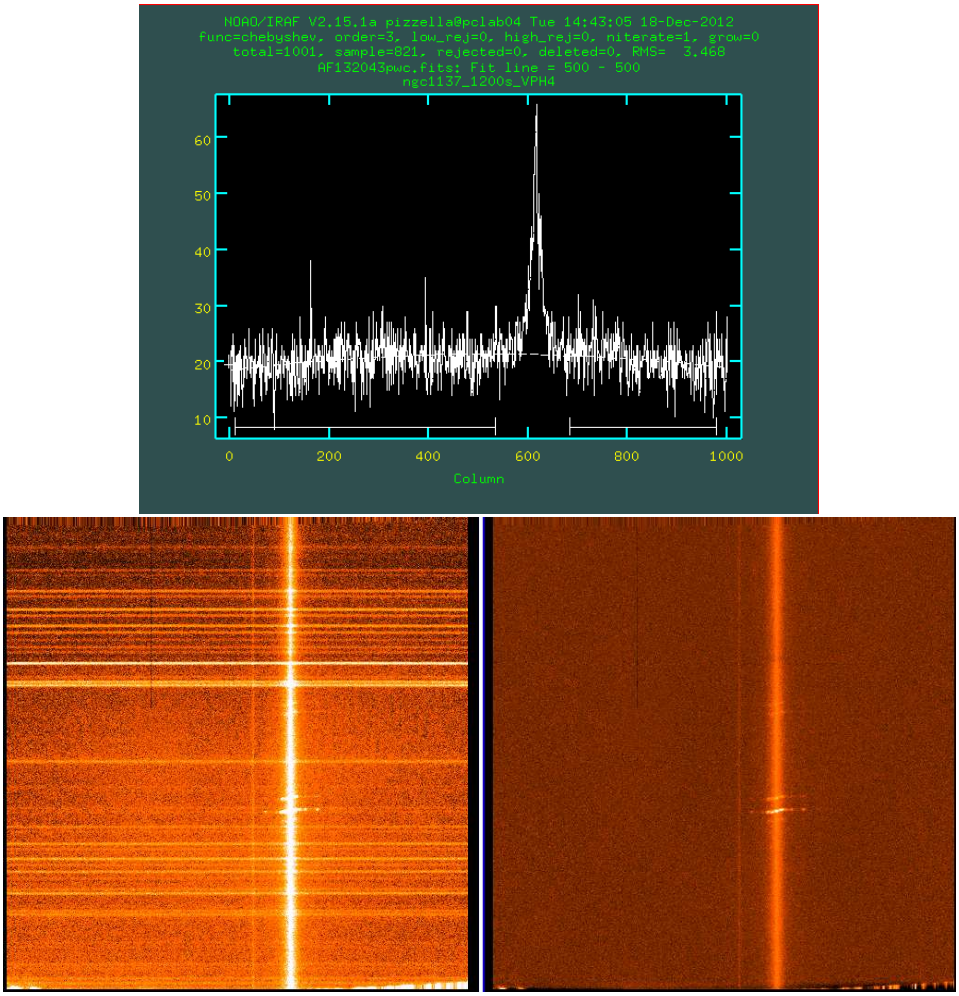

Figure 1.6: comando `background`: In alto plot dell'intensità lungo la fenditura a lunghezza d'onda fissata. Si può riconoscere come il livello del cielo sia pressocchè costante. Il picco di luminosità corrisponde alla posizione della galassia. I due segmenti orizzontali bianchi in basso indicano dove abbiamo scelto di fittare il cielo con il tasto `s`. Si può infine vedere la linea curva bianca che interpola l'andamento del cielo. In basso mostriamo lo spettro della galassia prima (sinistra) e dopo (destra) la sottazione del cielo.

cioè mostrato come cambia la posizione lungo la fenditura della stella al variare della lunghezza d'onda. Lo spettro della stella non è infatti perfettamente rettilineo e allineato con le colonne del CCD. È invege generalmente inclinato per motivi dovuti a un non perfetto allineamento, alle ottiche, o alla riferazione differenziale dell'atmosfera che curva diversamente la luce a seconda della lunghezza d'onda. Nella fase 2 è possibile interpolare l'andamento in modo da estrarre lo spettro in maniera ottimale.

standard . Questo comando serve per adattare il nostro spettro alla tabella di flusso. La stella standard spettrofotometrica è stata osservata con un set-up strumentale differente dal nostro. La tabella con in flussi è data ad intervalli diversi del passo in lunghezza d'onda con cui sono stati calibrati in nostri spettri. Il comando, di cui mostriamo i parametri principali,

input =”AF132033pwsa.fits”	Input image file root name
output =”std”	Output flux file (used by SENSFUNC)
star_name =”mhr718”	Star name in calibration list

L'output del comando è la tabella *std* in cui, oltre ai flussi della stella alle varie lunghezze d'onda è riportato il numero dei conteggi (ADU) da noi misurati.

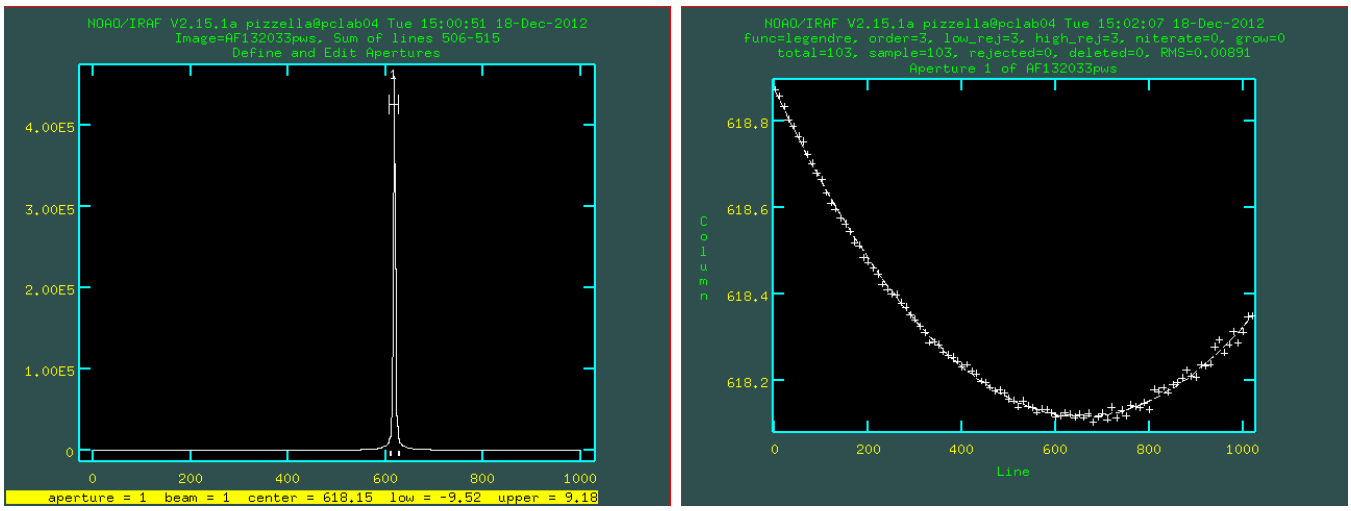

Figure 1.7: comando `apall`: In alto: fase 1, riconoscimento dell'apertura da estrarre. In questo caso abbiamo solo un stella posizionata nella colonna ~ 615. Il segmento bianco orizzontale sopra la stella indica la larghezza entro cui viene sommato il segnale. In basso: fase 2, *tracing* della posizione della stella interpolato con una funzione di Legendre di grado 3. Si può vedere come la posizione della stella, al variare della lunghezza d'onda si sposti di circa 1 pixel, andando dalla colonna 618.1 alla 618.9.

sensfunc Questo comando esegue il rapporto tra il flusso in calibrato della standard e i conteggi da noi misurati. Tale rapporto va oppurtunamente interpolato. Infatti, la diversa risoluzione spettrale tra le nostre osservazioni e la tabella dei flussi fa si che, specie in corrispondenza delle righe di assorbimento stellari, l'andamento del rapporto non è regolare come dovrebbe essere. Nella Fig.1.8 si vede infatti come proprio in corrispondenza della riga Hα 6562Å sia presente una discontinuità. Una cosa simile accade attorno a 6970Å ma in questo caso si tratta di una riga di assorbimiento del cielo. Queste irregolarità vanno eliminate dalla nostra interpolazione. Conviene allora, oltre che a determinare il tipo di funzione e il grado del polinomio come ormai sappiamo fare (comandi `:func` e `:ord`), utilizzare il comando `d` per cancellare i punti di cui non vogliamo tenere conto.

standards = "std"	Input standard star data file (from STANDARD)
sensitivity = "sens"	Output root sensitivity function imagename
answer = "yes"	(no—yes—NO—YES)

calibrate . Il comando serve per applicare la calibrazione in flusso. L'utilizzo è ovvio. Oltre a input ed output è importante specificare la *sensitivity* che deve essere l'output del comando `sensfunc`. Da notare che il comando `calibrate` applica anche una correzione per estinzione atmosferica e tiene automaticamente conto del tempo di posa.

input = "skyAF132043"	Input spectra to calibra
output = "skyAF132043f"	Output calibrated spect
(extinct = yes)	Apply extinction correct
(flux = yes)	Apply flux calibration?
(sensitivity = "sens")	Image root name for ser

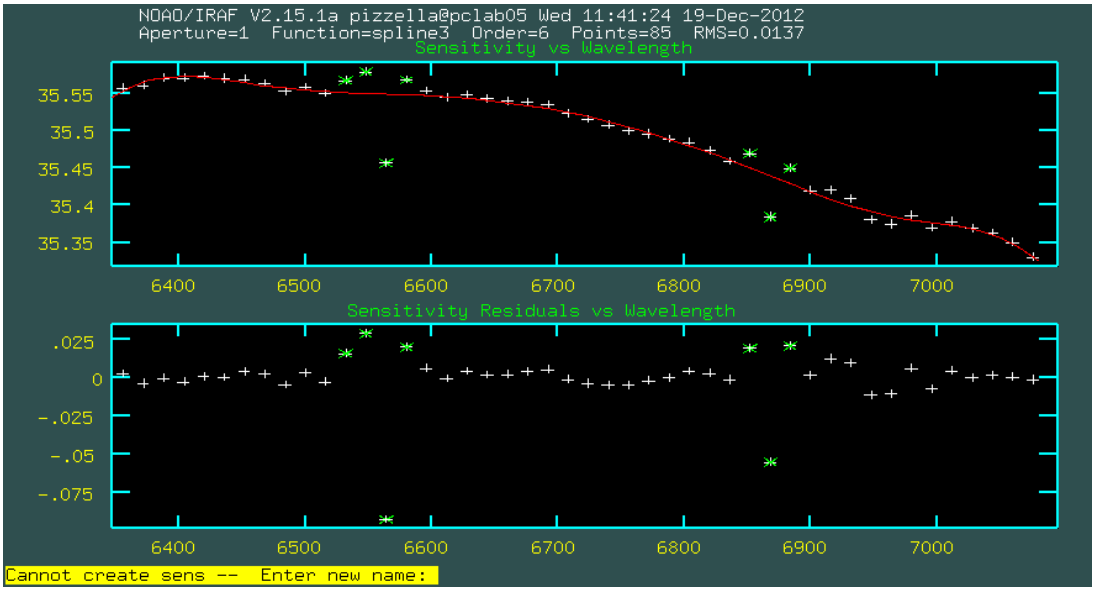

Figure 1.8: comando **sensfunf**, interpolazione della funzione di risposta dello strumento. I punti asterisco-verdi sono i punti eliminati dal fit con il comando `d`.