Reducción de espectros

Previo a cualquier proceso de reducción, con la tarea *imheader* buscamos, en cualquier imagen, el tamaño del CCD y los valores de la ganancia y del ruido de lectura.

En este caso la ganancia del CCD es 3.9, la relación señal ruido es de 4.6 y el tamaño de la imagen es [394,576].

1. PRERREDUCCIÓN

1.1 CORREGIMOS POR *OVERSCAN* Y *TRIMMING*

PASO 1: Encontrar la región de *overscan* de la imagen.

La región del *overscan* corresponde a pixeles virtuales que resultan de hacer leer al CCD valores adicionales a los que tiene físicamente. En estos valores solo hay ruido y un valor sistemático que agrega la electrónica (ese valor es el valor del *overscan*).

Dicha región se ubica desplegando una imagen (con el ds9 por ejemplo) y teniendo en cuenta el tamaño de la imagen. Es recomendable utilizar un *flat*.

PASO 2: Definir la región de los bordes que se desean cortar en la dirección de dispersión.

En general es suficiente cortar entre 5 y 10 pixeles de cada lado.

Para corregir por *overscan* y *trimming* vamos a utilizar al tarea CCDPRO.

■■■■■

Cargamos los paquetes: *noao, imred* y *ccdred.* Si se desea definir los parámetros que vienen por defecto en el paquete *ccdred* se puede utilizar la tarea *unlearn:*

```
ecl> unlearn ccdred
```

Primero generamos la lista de entrada a la tarea. Es decir un archivo que contenga una lista de todas las imágenes (bias, flats, objetos y comparaciones). Para ello se puede utilizar el comando:

```
home> ls *.fit > todos.lst
```

Ahora generamos la lista de salida de la tarea. Es decir un archivo que contenga una lista de todas las imágenes que estarán corregidas con el comando:

```
home> awk '$0="OT"$0' todos.lst > OTtodos.lst
```

Para hacer los dos pasos anteriores en una sola línea:

```
home> ls *.fit | awk '{print $0="OT"$0}' > OTtodos.lst
```

Ahora con la tarea *CCDPRO* corregimos por *overscan* y *trimming* todas las imágenes de manera interactiva:

```
ecl> ccdpro @todos.lst output=@OTtodos.lst ccdtype=" " fixpix- overscan+ trim+
zerocor-     darkcor-     flatcor-     readaxis=line     biassec=[385:394,1:576]
trimsec=[4:367,4:572] interactive+ function="legendre" order=3
```

Una vez que corremos la tarea, por pantalla pregunta si queremos ajustar el *overscan* de modo interactivo. A lo que debemos contestar que si. Luego se despliega una terminal gráfica de iraf, la irafterm. Sobre esta terminal vamos a ajustar el polinomio al *overscan*. Lo mas recomendable es utilizar una función de **legendre** de orden 3. El RMS del ajuste debe ser ~0.5 o menor.

Los parámetros de ajuste se pueden modificar desde la irafterm. Por ejemplo: para cambiar el orden del polinomio hay que escribir lo siguiente sobre la irafterm:

```
irafterm> :o 4
```

donde el 4 representa el orden nuevo que le queremos asignar al polinomio de ajuste.

Para ver todas las opciones se tipea ? sobre la irafterm

1.2. CORRECCION POR BIAS

PASO 1: Combinar todos los *bias* (corregidos por *overscan*).

Para esto se utiliza la tarea *ZEROCOMBINE*.

PASO 2: Restar el Bias promedio de todas las imágenes.

Para esto se utiliza la tarea *CCDPRO*.

■■■■■

Primero generamos el archivo de entrada a la tarea *zerocombine*. Es decir un archivo que contenga una lista de todos los *bias* con el comando

```
ecl> ls OTbias*.fits > OTbias.lst
```

Ahora promediamos los *bias*:

```
ecl> zerocombine @OTbias.lst output=Zero combine=average reject=minmax process-
delete- clobber- scale=none lsigma=3. hsigma=3. rdnoise=4.6 gain=3.9
```

Ahora generamos el archivo de entrada a la tarea *ccdpro*. Es decir una lista que contenga todas las imágenes excepto los *bias*. También, generamos el archivo de salida de la tarea *ccdpro*, es decir, un archivo que contenga una lista de todas las imágenes que estarán corregidas por *bias*.

```
home> awk '$0!~/OTbias/' OTtodos.lst > Ottodos-bias.lst
home> awk '$0="B"$0' OTtodos-bias.lst > BOTtodos.lst
```

Ahora restamos el *bias* promedio:

```
ecl> ccdpro @OTtodos-bias.lst output=@BOTtodos.lst ccdtype=" " fixpix- overscan- trim- zerocor+ darkcor- flatcor- illumcor- fringecor- readcor- scancor- readaxis=line zero=Zero.fits
```

1.3 CORRECCION POR FLAT

PASO 1: Combinar los *flats* (corregidos por *overscan, trimming* y *bias*).

Para esto se utiliza la tarea *FLATCOMBINE*.

PASO 2: Normalizar el *flat* promedio.

Esto es necesario para poder corregir por las irregularidades propias del CCD. Para esto se utiliza la tarea RESPONSE.

PASO 3: Dividir todas las imágenes por el *flat* promedio normalizado.

Para esto se utiliza la tarea *CCDPRO.*

■■■■■

Primero generamos un archivo que contenga una lista de todos los *flats* con el comando:

```
ecl> ls BOTffl*.fits > BOTflat.lst
```

Ahora promediamos los *flats:*

```
ecl> flatcombine @BOTflat.lst output=Flat combine=average reject=avsigclip process- subsets- delete- clobber- scale=mode rdnoise=4.6 gain=3.9
```

Ahora para normalizar el *flat* promedio cargamos los paquetes: *twodspec* y *longslit* para poder utilizar la tarea *RESPONSE:*

```
ecl> response calibrat=Flat.fits normaliz=Flat.fits response=NFlat.fits interactive+ order=6 low_rej=1. niterat=3
```

Una vez que corremos la tarea se despliega la *irafterm*, en la cual debemos ajustar un polinomio a la respuesta del CCD para poder normalizar el *flat*.

Al dividir el *flat* por el polinomio, logramos quedarnos con las irregularidades del CCD debidas a cuestiones externas al mismo. Para lograr un buen ajuste se pueden modificar ciertos parámetros como ser: el orden del polinomio, los puntos reyectados, la cantidad de iteraciones.

Como lo que estamos tratando de ajustar es la respuesta del CCD, el orden del polinomio no debe ser demasiado alto. Es decir que el “ruido” no debe ser ajustado.

Ahora, generamos los archivos de entrada y de salida de la tarea *ccdpro.* Esto es un archivo con las imágenes de ciencia y lámparas de comparación y otro contenga una lista de todas las imágenes que estarán corregidas:

```
home> awk '$0!~/BOTffl/' BOTtodos.lst > BOTtodos-flat.lst
home> awk '$0="F"$0' BOTtodos-flat.lst > FBOTtodos.lst
```

Finalmente dividimos por *flat* promedio normalizado:

```
ecl> ccdpro @BOTtodos-flat.lst output=@FBOTtodos.lst ccdtype=" " fixpix-
overscan- trim- zerocor- darkcor- flatcor+ illumcor- fringecor- readcor-
scancor- readaxis=line flat=NFlat.fits
```

2. EXTRACCIÓN DE LOS ESPECTROS

PASO 1: Encontrar el espectro, es decir la apertura.

Esto se puede hacer manualmente examinando un corte eje espacial e indicando el pico apropiado con un cursor, o puede hacerse automáticamente si el espectro apropiado es el pico mas fuerte presente.

PASO 2: Definir las ventanas de extracción y del fondo del cielo.

En la practica, esto se realiza especificando el tamaño de la ventana de extracción en términos del numero de pixeles a la izquierda del centro del perfil de la apertura, y el numero de pixeles a la derecha del mismo.

De forma similar, la región de fondo del cielo se define en términos de una región a la izquierda y a la derecha del centro de perfil. Uno puede entonces examinar estas regiones sobre un corte a lo largo del eje espacial, y redefinirlas si es necesario.

PASO 3: Trazar el centro del perfil espacial en función del eje de dispersión.

Aunque supongamos que el eje espacial esta exactamente a lo largo de una fila o columna, el espectro no sera exactamente perpendicular al eje espacial (es decir, el espectro estelar no es exactamente paralelo a lo que estamos tomando como la teoría). En su lugar, el centro exacto del perfil espacial se desplazara ligeramente con la ubicación a lo largo del eje de dispersión. Hay por lo menos tres razones para esto:

a) las ópticas de cámara introducen distorsiones que serán peores a lo largo del eje mas largo (dispersión),
b) las redes no se sitúan exactamente en sus celdas, y
c) la refracción atmosférica diferencial hará que el extremo azul del espectro sea desplazado a lo largo de la ranura mas cerca del cenit que el del extremo rojo del espectro. Este ultimo efecto sugiere que podemos esperar que el angulo formado por el espectro y el eje de dispersión difieran, a menudo de manera significativa, de una exposición a otra.

PASO 4: Sumar el espectro dentro de la ventana de extracción, restando el cielo.

En cada punto a lo largo del eje de dispersión, los datos dentro de la apertura de extracción (centrada espacialmente en base al valor que la traza esta en ese punto) se suma, y el fondo del cielo se resta. Para esto se utiliza la tarea *APALL*.

■■■■■

2.1 EXTRACCIÓN DE LOS ESPECTROS DE CIENCIA

Para comenzar cargamos el paquete: *apextract.* Primero generamos las listas de entrada y salida de la tarea *apall*, es decir, un archivo que contenga una lista de todos los espectros de ciencia ya prerreducidos y otra lista con los que ya estarán extraídos.

```
home> ls FBOTobj*.fits > FBOTobj.lst
home> awk '$0="E"$0' FBOTobj.lst > EFBOTobj.lst
```

De ser necesario, agregamos en el *header* de la imagen el eje de dispersión a través del parámetro *dispaxis*. Si el eje de dispersión es horizontal tendrá valor 1 y si es vertical valdrá 2:

```
ecl> hedit FBOT*.fits dispaxis valor addonly+
```

Luego corremos la tarea *apall:*

```
ecl> apall @FBOTobj.lst nfind=1 output=@EFBOTobj.lst interact+ find+ recente+
resize+ edit+ trace+ fittrac+ extract+ extras+ review+ b_funct=chebyshev
b_niter=2  b_order=2  b_sample=[-10:-6,6:10]  bkg+  t_funct=legendre  t_order=3
t_niter=2  backgro=fit weights=variance saturat=16000 readnoi=4.6 gain=3.9
```

Si la tarea *apall* se corre con los parámetros ***extras=yes, backgro=fit*** y ***weights=variance***, en el archivo fits que sale de la tarea quedan guardadas 4 extensiones:

Primera extensión: se guarda la salida de la tarea *apall*
Segunda extensión: se guarda el espectro crudo (si *weights=variance*)
Tercera extensión: se guarda el espectro de cielo (si *backgro=fit*)
Cuarta extensión: se guarda el espectro de dispersión (si *weights=variance*)

Cuando corremos la tarea, por pantalla nos pregunta: Si queremos buscar la apertura, si queremos redimensionar la apertura, y si queremos editar la apertura. A todas estas preguntas contestamos que sí. Luego se despliega el *irafterm* donde nos muestra un corte espacial de la apertura. En esta terminal con las presionando las letras "l" (lower, izquierda) y "u" (upper, derecha) podemos redimensionar la apertura.

Recordar que presionando ? sobre el *irafterm* se accede al menú de ayuda.

Una vez que terminamos de definir la posición y el tamaño de la ranura debemos definir los parámetros del fondo del cielo. Para ello presionamos (sobre el *irafterm*) la letra "b" (de *background*). Sobre este nuevo despliegue debemos ajustar las posiciones y los tamaños de las ventanas de fondo del cielo y ajustarles un polinomio. En general se utiliza un polinomio de **chevyshev** de orden 2. Los

parámetros de ajuste se pueden modificar interactivamente en el *irafterm*. Cuando terminamos salimos del ajuste de fondo del cielo con la letra "q" (de *quit*).

Una vez definida la apertura y el fondo del cielo hay que pasar al ajuste de la traza. Para ello salimos del ajuste de la apertura con la letra "q". Luego, sobre la *irafterm* nos pregunta: Si queremos trazar la apertura, si queremos ajustar la posición de la traza, y si queremos hacer el ajuste de manera interactiva. A todas estas preguntas contestamos que sí.

Generalmente el ajuste de la traza se realiza con un polinomio de **legendre** de grado 3. Lo ideal es que RSM < 0.02.

Una vez ajustada la traza salimos con la letra "q". Luego, sobre la *irafterm* nos pregunta: Si queremos escribir la apertura en el directorio *DATABASE*, si extraemos el espectro de la apertura, si queremos ver el espectro extraído, y si queremos ver el espectro extraído de la apertura. A todas estas preguntas contestamos que si.

En el directorio *database* se guardará un archivo con el mismo nombre que el archivo que extrajimos en el cual se guardará información sobre los ajustes que realizamos durante el proceso de extracción.

2.2 EXTRACCIÓN DE LOS ESPECTROS DE COMPARACIÓN

Ahora generamos nuevamente los archivos de entrada y salida de la tarea *apall* pero ahora los archivos deben contener las listas de todas las comparaciones, con el mismo criterio que lo hicimos para las imágenes de ciencia.

```
home> ls FBOTcomp*.fits > FBOTcomp.lst
homa> awk '$0="E"$0' FBOTcomp.lst > EFBOTcomp.lst
```

Es importante ver que entre el archivo FBOTobj.lst y el FBOTcomp.lst cada comparación se corresponda con su objeto (es decir que estén listados en el mismo orden). En el caso en el que una comparación se corresponda con 2 objetos distintos, debe estar repetido en la lista en su correspondiente orden.

Para extraer los espectros de compración no es necesario corregirlos por el fondo del cielo. Vamos a extraerlos con los mismos parámetros que definimos en los espectros de ciencia.

```
ecl> apall @FBOTcomp.lst nfind=1 output=@EFBOTcomp.lst reference=@FBOTobj.lst
interact- find- recente- resize- edit- trace- fittrac- extract+ extras- review-
bkg- backgro=none weights=none saturat=16000 readnoi=4.6 gain=3.9
```

3. CALIBRACION EN LONGITUD DE ONDA

Como nuestro patrón para las longitudes de onda son las lámparas de comparación, primero vamos a calibrar los espectros de lámpara con el objetivo de obtener una transformación entre la posición de los pixeles y la longitud de onda correspondiente. Luego aplicaremos esa transformación a todos los espectros de ciencia.

3.1. CALIBRACIÓN DE LAS COMPARACIONES

PASO 1: Determinar la solución de dispersión.

Hay que encontrar una trasformación para relacionar los pixeles en la dirección de dispersión con la longitud de onda. Para hacerlo hay que asignarle a las líneas de la lámpara de comparación la longitud de onda correspondiente. Una vez identificadas las líneas hay que ajustar un polinomio que relacione los pixeles con las longitudes de onda. Para esto se utiliza las tareas *IDENTIFY* y *REIDENTIFY.*

■■■■■

Vamos a utilizar una lámpara de comparación como prueba para la calibración, para ello hacemos una copia de un espectro de comparación cualquier. (Este paso no es estrictamente necesario, yo lo hago sólo por una cuestión organizativa).

```
ecl> imcopy EFBOTcomp01.fits comp_ref.fits
```

Hacemos la identificación de las líneas en una lámpara de comparación. Para este paso es preciso tener una imagen de la lámpara de comparación (en la misma región espectral) con las líneas identificadas. Así podremos asignarle la longitud de onda correspondiente a cada línea de nuestro espectro de referencia.

```
ecl> identify comp_ref.fits function="legendre"
```

Cuando corremos la tarea se despliega el espectro de la lámpara. Con la letra "m" (de *mark*) asignamos las longitudes de onda a las líneas identificadas. Conviene hacer esto con cuatro o cinco líneas, asegurándonos de tomar al menos una en cada borde. Luego, con la letra "f" (de *fit*) ajustamos un polinomio de bajo orden (2 ó 3 dependiendo de la cantidad de puntos) y salimos con la "q". Ahora, con la "l" (de *lines*) traemos todas las líneas que están guardadas en la base de datos del iraf y volvemos a ajustar el polinomio con la letra "f". El polinomio de ajuste generalmente es mayor o igual a 3 y el RMS debe ser, en lo posible, menor a 0.2 y en el gráfico de errores los puntos deben verse distribuidos al azar.

Una vez ajustado el polinomio salimos del ajuste con la letra "q". Luego, sobre la *irafterm* nos pregunta: si queremos guardar las identificaciones en el *DATABASE*. A lo que contestamos que sí.

Ahora identificamos las líneas de todas las demás lámparas tomando como referencia la primera

```
ecl>  reidentify  reference=comp_ref.fits  images=@EFBOTcomp.lst  interact+
override+ newap-
```

3.2. CALIBRACIÓN DE LA CIENCIA

PASO 1: Primero debemos asignarle a cada espectro de ciencia su lámpara de comparación. Esto lo hacemos con la tarea *refspec.*

PASO 2: Aplicamos la transformación definida con la comparación al espectro de ciencia para que quede calibrado en longitud de onda. Esto lo hacemos con la tarea *dispcor.*

■■■■■

Primero cargamos el paquete: *onedspec*. Luego, a cada objeto le asignamos su espectro de referencia para la calibración.

```
ecl> refspec @EFBOTobj.lst referen=@EFBOTcomp.lst select=match override+
```

Generamos un archivo que contenga una lista de todas los objetos que estarán calibrados en longitud de onda:

```
ecl> awk '{sub(/EFBOT/,"W")}1' EFBOTobj.lst > Wobj.lst
```

Aplicamos la calibración.

```
ecl> dispcor @EFBOTobj.lst output=@Wobj.lst linearize+
```

4. CALIBRACIÓN EN FLUJO

Las que vamos a utilizar en este ejemplo están en el directorio /home/iraf/noao/lib/onedstds/spec16cal/. Para ver los distintos tipos de estrellas estándares hay utilizar la sentencia:

```
ecl> page onedstds$README
```

PASO 1: Estimar la cantidad de cuentas por longitud de onda y asignarle el valor de flujo correspondiente. Esto lo hacemos con la tarea *STAND*.

PASO 2: Ajustar la función de sensitividad como función de la longitud de onda.

Para realizar el ajuste es necesario corregir por extinción atmosférica. Para ello se puede utilizar una tabla estándar de extinción, o se puede tratar de determinar la extinción empírica de nuestros datos. También se pueden realizar "cambios grises" de una observación en particular, eliminar puntos u observaciones, e interactuar generalmente hasta que se tenga un ajuste satisfactorio a los puntos. Para esto utilizamos la tarea *SENSFUNC*.

PASO 3: Aplicar la función de sensitividad a la ciencia. Esto lo hacemos con la tarea *CALIB.*

■■■■■

4.1. CALIBRACIÓN DE LA ESTRELLA ESTÁNDAR

Para este ejemplo vamos a trabajar con la estrella estándar HR 4468 calibrada en longitud de onda (será el archivo Wstd.fits) y utilizaremos el archivo *casleoext* para la corrección por extinción atmosférica. Entonces lo primero que hay que hacer es muestrear el continuo:

```
ecl> stand Wstd.fits output=std
extinct=/home/yael/Todo/Observaciones/CASLEO/casleoext
```

```
caldir=/opt/anaconda/pkgs/iraf.gmisc-2010_11_18-1/iraf_extern/gmisc/lib/
onedstds/spec16cal/ interact+ star_nam=hr4468blue mag=4.68 magband=V
```

Cuando corremos la tarea nos pregunta: si queremos editar las bandas, a lo que contestamos que sí.

Es importante que las bandas caigan sobre el continuo, por lo que las que caigan sobre las líneas debemos borrarlas con la letra "d". Salimos con "q".

Ahora hay que determinar las funciones de sensibilidad del detector y de extinción. El archivo de entrada a la tarea *sens* es el archivo de salida de la tarea *stand*:

```
ecl> sens standard=std sensitiv=sens
extinct=/home/yael/Todo/Observaciones/CASLEO/casleoext interact+ graphs="irs"
function=legendre order=10
```

Cuando corremos la tarea nos pregunta: si queremos ajustar la apertura interactivamente, a lo que contestamos que sí. A continuación, sobre el *irafterm* hay que ajustar el polinomio a la función de sensibilidad.

4.2. CALIBRACION DE LA CIENCIA

Generamos dos archivos que contengan la lista de todos los objetos que queremos calibrar y la lista de los que estarán calibrados en flujo:

```
home> awk '$0=$0"[0]"' Wobj.lst > Wobj_ext.lst

home> awk '{sub(/fits\[0\]/,"fits")}1' Wobj_ext.lst > zz
home> awk '$0="F"$0' zz > FWobj.lst
home> rm zz
```

Y los calibramos:

```
ecl> calib @Wobj_ext.lst output=@FWobj.lst sensiti=sens
extinction=/home/yael/Todo/Observaciones/CASLEO/casleoext
```

5. NORMALIZAR UN ESPECTRO

La normalización de un espectro se puede hacer de dos maneras: una es dividiendo a todo el espectro por el flujo correspondiente a una longitud de onda determinada, la otra es ajustar una función al continuo y dividir al espectro por dicha función.

5.1 Dividir al espectro por el flujo correspondiente a una longitud de onda determinada

Para saber cual es el flujo que le corresponde a una longitud de onda determinada, λ, podemos:

- Desplegar el espectro con la tarea *SPLOT* del paquete *onedspec*. Luego sobre el *irafterm* se posiciona el cursor en el punto deseado y se presiona la barra espaciadora. Sobre el borde inferior se mostrarán los valores correspondientes a este punto.
- Se puede convertir el archivo *fits* a un archivo *ascii* de dos columnas: λ y flujo. Esto se hace con la tarea *LISTPIX*.

```
ecl> listpix FWobj.fits[*,1,1] wcs=world > FWobj.dat
```

Luego, del archivo ascii se puede obtener el valor del flujo correspondiente a la longitud de onda λ.

Finalmente, para normalizar el espectro hay que dividirlo por el valor del flujo. Esto se hace con la tarea SARIT.

```
ecl> sarit  FWobj.fits / flujo output=NFWobj.fits
```

5.2. Ajustar una función al continuo y dividir al espectro por dicha función

Para hacer estas dos cosas al mismo tiempo se puede utilizar la tarea *splot* del paquete *onedspec*. Para ello, primero desplegamos el espectro (que puede o no estar calibrado en flujo):

```
ecl> splot FWobj.fits
```

Sobre el irafterm presionamos la letra "t". Al pie de la terminal nos lista una serie de opciones, en este caso elegimos la opcion NORMALIZE. Luego, se va modificando el orden del polinomio hasta conseguir el mejor ajuste. Salimos con la letra “q”.

Si queremos guardar el espectro normalizado como una nueva imagen se presiona la letra "i" y al pie de la irafterm se ingresa el nombre del archivo. Finalmente salimos con la letra "q".

6. CÁLCULO DE LA RESOLUCIÓN Y DISPERSIÓN ESPECTRAL.

Lo que usualmente llamamos dispersión, formalmente es la **Dispersión Lineal Inversa** que se mide en angstroms por milímetro o por pixel y se calcula como:

Dispersión= $\Delta_\lambda / \Delta_x$

donde Δ_x corresponde a un elemento del detector. Usualmente, en los CCDs, se toma como elemento 2 pixeles ($\Delta_x = 2$). Δ_λ es la cantidad de angstroms que entran en ese elemento de detector.

El valor de *dw* de la salida que se muestra por pantalla al ejecutar la tarea *DISPCOR* corresponde a la Dispersión Lineal Inversa en unidades de angstrom por pixel.

El **Poder Resolvente** es igual a:

$\Delta_\lambda = w_{ef} *$ dispersión $= w_{ef} * \Delta_\lambda / \Delta_x$

Este w_{ef} es el máximo entre otros dos valores: w_0 y w_d. El w_0 es el valor mínimo entre el ancho de la imagen de la ranura sobre el detector y el *seeing*; y el w_d es la capacidad del detector para resolver detalles que, de acuerdo al teorema del muestreo, es de 2-3 pixeles.

Finalmente, la **resolución, R**, se calcula como:

$R= \lambda / \Delta_\lambda$

Así, la relación entre la resolución y la dispersión sería:

$R = \lambda / (w_{ef} * \text{dispersión})$

Entonces, para calcular la dispersión espectral hay que calcular la cantidad de angstroms que entran en 2 pixeles (se toma dos pixeles porque en teoría el ancho de las líneas de la lampara debería ser de dos pixeles). Entonces si N_p es la cantidad de pixeles y λ_i y λ_f son las longitudes de onda inicial y final del espectro, la dispersión por pixel será:

$\text{dispersion} = (\lambda_f - \lambda_i) / N_p$

Ahora, la resolución, R, se puede estimar de distintas maneras. Si λ_c es la longitud de onda central del espectro:

$R = \lambda_c / (2 * \text{dispersion}) = \lambda_c * N_p / [\ 2 * (\ \lambda_f - \lambda_i)\ ]$

Ahora, en vez de tomar como unidad de medida los pixeles (dos en este caso) lo mas adecuado sería tomar como unidad de medida el FWHM de las líneas. Para ello se puede usar el espectro de la lámpara de comparación o el el espectro de cielo. Este ultimo es el que tiene la resolución real ya que va a estar afectado por el *seeing*. Es importante buscar líneas que no sean un blend. Así el poder resolvente es:

$R = \lambda_{cl} / \text{FWHM}$

donde λ_{cl} es la longitud de onda central de la línea que estamos midiendo. Se toman las medidas de varias líneas y luego se promedian.

Para medir las líneas se puede ejecutar la tarea *splot* para desplegar el espectro y ajustar una gaussiana a una línea con presionando la letra 'k' sobre el continua a ambos lados de la línea. Los valores medidos quedan guardados en el archivo *splot.log*.

Es de esperar que en un espectro de baja resolución el R no sea constante a lo largo de todo el espectro. Si bien, el FWHM en pixeles de las líneas de la lámpara de comparación es constante a lo largo de todo el espectro (ya que es una imagen de la ranura) al calibrar en longitud de onda se pierde la homogeneidad del FWHM debido a la ecuación de la red. Dando lugar a una resolución diferente en distintas regiones del espectro. En cambio en un echell el R es constante debido a como esta construido el espectrográfo.

Según el teorema del muestreo de Niquyst-Shannon, un pulso gaussiano va a estar bien muestreado si su FWHM es de 2 ó 3 pixeles. Por lo tanto, en la práctica el ancho mínimo de ranura que se debe utilizar es aquel para el cual las lineas de la lámpara de comparación tengan un FWHM de 2 - 3 pixeles. Esta configuración instrumental dará la mayor resolución. Para una ranura menor empieza haber problemas de submuestreo (por lo que se dificulta medir la longitud de onda central y el FWHM de las líneas) y de refracción. Para ranuras más anchas, se pierde en resolución pero se gana en flujo. Obviamente que la resolución límite también estará afectada por el *seeing*. Tener en cuenta que 273 μm equivalen en el cielo a 3 arcsec (sacado de la página de CASLEO).

En las publicaciones es importante aclarar en a que longitud de onda corresponde el R que estamos midiendo.

7. CÁLCULO DE LA RELACIÓN SEÑAL RUIDO (SNR)

Una posibilidad es calcularlo con la tarea *splot*. Se despliega el espectro con la tarea *splot* y sobre una región de continuo se selecciona una muestra presionando la tecla “m” en los extremos de la muestra.

Otra posibilidad es calcularla construyendo un espectro de señal-ruido. Para ello vamos a llamar S al espectro de ciencia extraído con la tarea *APALL* (corregido por el fondo de cielo), llamamos B al espectro de cielo que también se extrae con la tarea APALL y se guarda en la tercera extensión del espectro extraído. Luego, el espectro de señal-ruido será:

SNR= S / sqtr(S + B)

En las publicaciones es importante aclarar en a qué longitud de onda corresponde la SNR que estamos midiendo.

Para generar el espectro SNR vamos a utilizar la tarea *SARIT*. Primero generamos el espectro S + B:

```
ecl> sarit input1=Wobj.fits[*,1,1] op=+ input2=Wobj.fits[*,1,3] output=SB.fits
```

Luego calculamos sqtr(S + B):

```
ecl> sarit input1=SB.fits op=sqrt output=sqrtSB.fits
```

Finalmente calculamos SNR:

```
ecl> sarit input1=Wobj.fits[*,1,1] op=/ input2=sqrtSB.fits output=SNR.fits
```