Astronomía
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS UNIVERSIDAD DE CHILE

# INFORME INSTRUMENTACIÓN ASTRONÓMICA

Tarea 1


| | |
|---|---|
| Estudiante: | Camilo Núñez Barra |
| RUT: | 20.533.326-6 |
| Profesor: | Leonardo Bronfman |
| Auxiliar: | Paulina Palma |
| Curso: | AS3201 Astronomía Experimental |
| Fecha: | 4 de mayo de 2021 |


## 1. Introducción

Se utiliza el radiotelescopio MINI.

## 2. Hot–Cold Test

### 2.1. Marco teórico

La señal de un objeto celeste recibida por un radiotelescopio debe pasar por la atmósfera antes de llegar al receptor de la antena. El receptor tiene corrientes internas y fenómenos de transporte de electrones y fotones con ondas estacionarias que aumentan la entropía y generan también un error sistemático correspondiente a ruido blanco.

El ruido blanco del receptor se puede disminuir al aumentar el tiempo de integración pues la señal del cielo permanece constante y la señal del receptor disminuye relativamente su tamaño, por lo que disminuir el ruido permite disminuir el tiempo invertido para detectar una señal astronómica y a la vez permite no empeorar la señal.

Una manera común de eliminar errores sistemáticos es mediante la calibración del instrumento de medición.

El método Hot–Cold Test permite calibrar el telescopio al caracterizar la temperatura de ruido del receptor mediante dos cargas cuyas temperaturas son conocidas y diferentes.

Una carga es un material absorbente adherido a un pedazo de madera con un mango. El material es un absorbente electromagnético que absorbe la radiación con muy poca reflexión por lo que se supone como cuerpo negro.

Sea $T_{\text{rec}}$ la temperatura de ruido del receptor y $G_{\text{rec}}$ la ganancia del receptor. Una carga a temperatura ambiente $T_{\text{hot}}$ se pone enfrente de la bocina de la antena, permitiendo medir una potencia espectral $W_{\text{hot}}$ dada por,

$$W_{\text{hot}} = G_{\text{rec}} k T_{\text{rec}} + G_{\text{rec}} k T_{\text{hot}}, \tag{2.1}$$

donde $k$ es la constante de Boltzmann. Análogamente, para una carga fría a temperatura $T_{\text{cold}}$, la potencia espectral medida es,

$$W_{\text{cold}} = G_{\text{rec}} k T_{\text{rec}} + G_{\text{rec}} k T_{\text{cold}}. \tag{2.2}$$

Se define el factor $Y$ como el cuociente entre la medición de la potencia espectral para la carga caliente y para la fría,

$$Y = \frac{W_{\text{hot}}}{W_{\text{cold}}}, \tag{2.3}$$

que permite determinar la temperatura de ruido del receptor mediante variables medidas,

$$T_{\text{rec}} = \frac{T_{\text{hot}} - Y T_{\text{cold}}}{Y - 1}. \tag{2.4}$$

En términos de operación del telescopio, la medición de $T_{\text{rec}}$ se hace cada día que es observa, por lo que en una campaña de meses se tiene que hacer todos los días la medición. Esta constante medición corresponde a un chequeo del estado de la electrónica del telescopio pues si repentinamente difiere mucho la temperatura de ruido del receptor con respecto a la medición del día anterior es porque está funcionando mal y se debe arreglar. En realidad se usa la potencia por canales de frecuencia y no la potencia espectral de todo el ancho de banda, tal como se describe en la sección 2.4, por lo que la temperatura de ruido del receptor cambia con la frecuencia.

### 2.2. Datos y metodología

Esta calibración utiliza una carga caliente a temperatura ambiente en la cúpula del radiotelescopio MINI y una carga fría a temperatura de nitrógeno líquido, cuyas mediciones están en la tabla 1.

El MINI es pequeño, permitiendo acceder a la bocina por una escalera. Se sube y se pone la carga caliente enfrente de la bocina procurando apuntar el material absorbente a ella. Mediante un *powermeter* se mide la potencia integrada de la señal para todo el ancho de banda que tiene el receptor y se anota la lectura en la tabla 1.

A continuación y análogamente, se pone la carga fría enfrente de la bocina y se mide la potencia espectral con el *powermeter* pero esperando a que la lectura correspondiente converja tras disminuir la temperatura. Esta medición está en la tabla 1.

| Carga | Temperatura K | Potencia dBm |
|---|---|---|
| Hot | 300 | −44.50 |
| Cold | 77 | −47.94 |

Cuadro 1: Temperatura y potencia medidas para las cargas del Hot–Cold Test

El *powermeter* entrega potencias en decibeliomilivatio (dBm), que es una escala logarítmica acorde a las eventuales amplificaciones y disminuciones de las señales. Una potencia $W$ en escala logarítmica de dBm se convierte en una potencia $P$ en escala lineal de W como se muestra a continuación,

$$P = 10^{\frac{W-3}{10}}. \tag{2.5}$$

### 2.3. Cálculo de $T_{\text{rec}}$

Se convierten las potencias de la tabla 1 a vatios según la ecuación 2.5 y se calcula $Y$ según la ecuación 2.3, obteniendo $Y = 2.2$. Esto permite usar la ecuación 2.4 para obtener $T_{\text{rec}} = 107.6\,\text{K}$. Este cálculo se desarrolla en el código 1.

### 2.4. Comparación con calibración del MINI

El software del MINI tiene el comando `%hct` para ingresar todo el sistema a una subrutina de Hot–Cold Test. Se usan cargas las mismas temperaturas de la tabla 1.

El sistema espera a que se ponga la carga caliente en la bocina del receptor y se marca la medición al presionar una botonera, permitiendo tener una potencia por cada canal de frecuencia de la señal, mostrando una variación en todo el espectro. Ahora se pone la carga fría y se apreta la botonera, midiendo una potencia por cada canal que también varía en todo el espectro pero que es menor.

Se usan los dos vectores de potencia para calcular el factor Y según la ecuación 2.3 y luego la temperatura de ruido del receptor por canal según la ecuación 2.4. Finalmente, se promedian las temperaturas de todos los canales, resultando una temperatura de ruido del receptor de $T'_{\text{rec}} = (150.9 \pm 4.6)\,\text{K}$.

Se aprecia que $|T'_{\text{rec}} - T_{\text{rec}}| = 43.3\,\text{K}$.

## 3. Antenna Tipping

### 3.1. Marco teórico

La turbulencias atmosféricas y el vapor de agua presente en la atmósfera distorsionan la señal de un objeto celeste dando lugar a un error sistemático que no depende del radiotelescopio.

La atmósfera en sus distintas capas varía la temperatura y la densidad pero en este experimento se hace una aproximación a primer orden para establecer que contribuye una temperatura de ruido $T_{\text{atm}}$ fija. Además, se hace también una aproximación al igualar la temperatura ambiente del domo $T_{\text{amb}}$ con $T_{\text{atm}}$.

La opacidad cenital $\tau_{\text{w}}$ debido al contenido de agua en la atmósfera y la opacidad cenital total $\tau_0$ de la atmósfera, que incluye al oxígeno, se aproximan también a primer orden para decir que son iguales.

El método Antenna Tipping permite calibrar el receptor del telescopio al determinar el estado y opacidad cenital de la atmósfera.

Se mide la potencia del cielo $W_{\text{sky}}$ detectada por el telescopio a un determinado intervalo de frecuencias y elevación $\varphi$, que está dada por,

$$W_{\text{sky}} = c\left(T_{\text{rec}} + T_{\text{atm}}\left(1 - \exp\left(-\tau_0/\sin\varphi\right)\right)\right), \tag{3.1}$$

donde $T_{\text{rec}}$ es la temperatura de ruido del receptor y $c$ es una constante que, por ejemplo, puede depender de la ganancia del receptor.

Análogamente, la potencia para una carga enfrente de la bocina a temperatura ambiente $T_{\text{amb}}$ (la misma carga Hot del Hot–Cold Test), la potencia es,

$$W_{\text{amb}} = c\left(T_{\text{rec}} + T_{\text{amb}}\right). \tag{3.2}$$

Aplicando las aproximaciones, se define,

$$\Delta W \equiv W_{\text{amb}} - W_{\text{sky}} = cT_{\text{amb}} \exp(-\tau_{\text{w}}/\sin\varphi), \tag{3.3}$$

y $z = \pi/2 - \varphi$, permitiendo obtener la relación,

$$\ln(\Delta W) = -\sec(z)\,\tau_{\text{w}} + \ln(cT_{\text{amb}}), \tag{3.4}$$

que es una relación lineal de $\ln(\Delta W)$ con respecto a $-\sec(z)$, donde $\tau_{\text{w}}$ es la pendiente de la recta.

## 3.2. Datos y metodología

Se utiliza una carga caliente a temperatura ambiente en la cúpula del radiotelescopio MINI, midiendo la potencia que muestra Domo en la tabla 2.

Se mide con un *powermeter* la potencia espectral de diez puntos a distintas elevaciones y azimut fijo, entregando los resultados de la tabla 2.

| Punto | Elevación ° | Potencia dBm |
|---|---|---|
| 1 | 23.50 | −45.56 |
| 2 | 16.60 | −45.22 |
| 3 | 12.84 | −45.01 |
| 4 | 10.48 | −44.88 |
| 5 | 8.85 | −44.80 |
| 6 | 7.66 | −44.73 |
| 7 | 6.76 | −44.71 |
| 8 | 6.04 | −44.67 |
| 9 | 5.47 | −44.65 |
| 10 | 4.99 | −44.63 |
| Domo | | −44.54 |

Cuadro 2: Elevación y potencia para los distintos puntos a azimut fijo. Se incluye domo con carga caliente

## 3.3. Cálculo de $\tau_{\text{w}}$

El siguiente procedimiento corresponde al código 2. Se lee el archivo `antdip_AE2021A` provisto por el equipo docente con los datos necesarios para realizar el ajuste lineal según la ecuación 3.4. El gráfico de este se muestra en la figura 1. La pendiente de la recta establece un valor de 0.257 350 1 para la opacidad cenital debido al contenido de agua en la atmósfera.

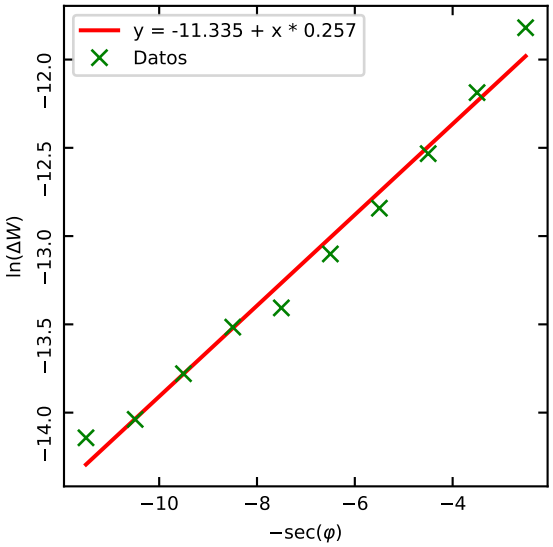


Figura 1: Equis roja es medición del telescopio. Línea azul es ajuste lineal. La pendiente es $\tau_{\text{w}}$

## 3.4. Comparación con calibración del MINI

El software del MINI tiene el comando `%antdip` para ingresar todo el sistema a una subrutina de Antenna Dipping.

Primeramente se debe ingresar la cantidad de masas de aire por punto y se usa 1, que es el valor típico.

Luego el sistema recolecta automáticamente información para diez puntos separados por una masa de aire a través de la elevación a azimut fijo, midiendo la mayor elevación primero y luego disminuye progresivamente. El telescopio constantemente evalúa su posición y solo toma datos cuando está apuntando con cierta tolerancia a la coordenada determinada por el sistema de control.

La información recolectada corresponde a la diferencia de potencia de la ecuación 3.3, donde la potencia para el cielo es la que apunta según la elevación correspondiente y la potencia para la carga caliente es según el *chopper*, una carga absorbente electromagnética, que está dentro de la bocina de la antena y automática y periódicamente obstruye la visión del telescopio gracias a un motor.

Tras medir los diez puntos, automáticamente el telescopio apunta al domo y toma una medición de referencia con la carga caliente. A continuación, se debe ingresar la temperatura actual y la

humedad relativa, además de un parámetro cualitativo de 0 a 3 que indica el grado de cobertura del cielo debido a las nubes y sirve para el registro histórico.

Finalmente, el software realiza el ajuste lineal y entrega el valor $\tau'_{\mathrm{w}} = 0.257\,350\,1$, además de otros parámetros como la estimación de la eficiencia del telescopio y la temperatura de brillo del agua en el cielo.

Se aprecia que $\tau_{\mathrm{w}} = \tau'_{\mathrm{w}}$.

## 4. Observaciones

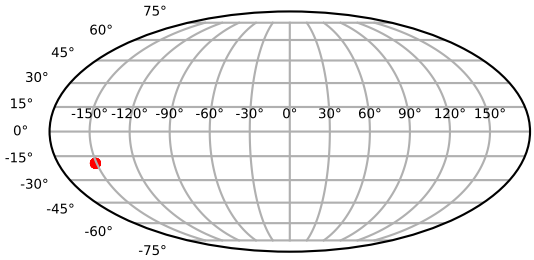


Figura 2: Coordenadas galácticas

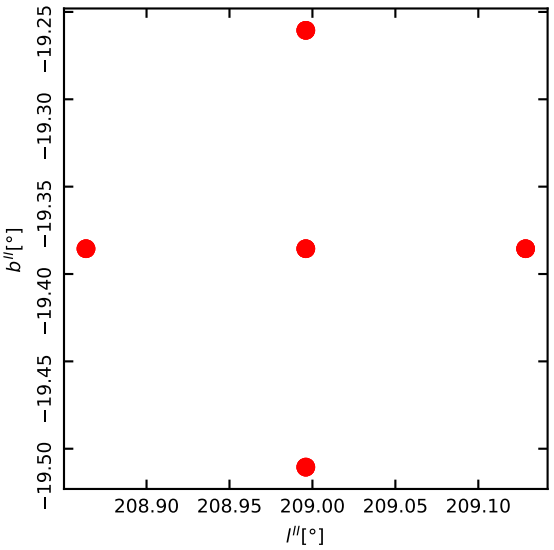


Figura 3: Cruz de observación

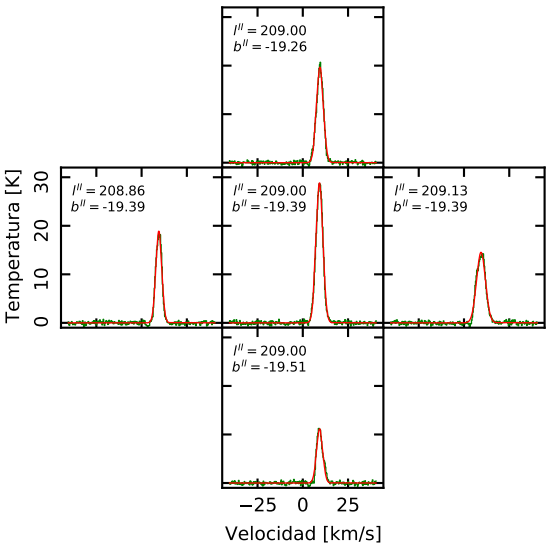


Figura 4: Ajuste gaussiano 1

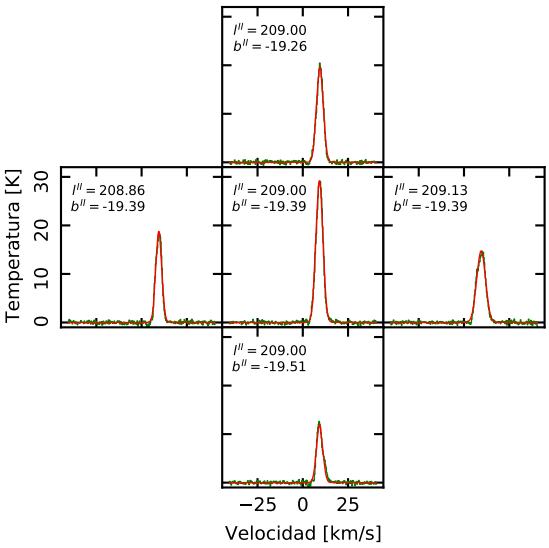


Figura 5: Ajuste gaussiano 2

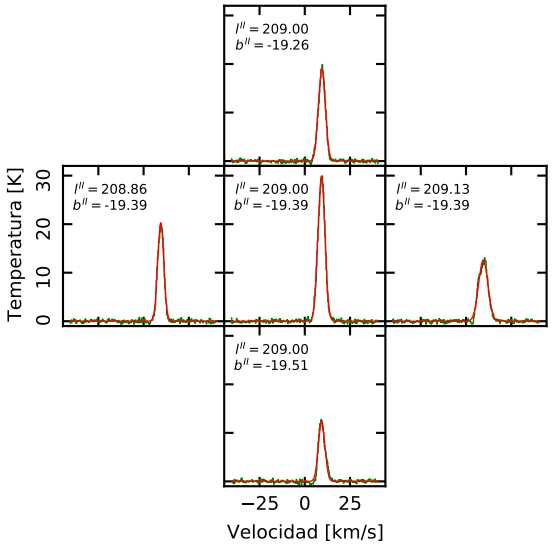


Figura 6: Ajuste gaussiano 3

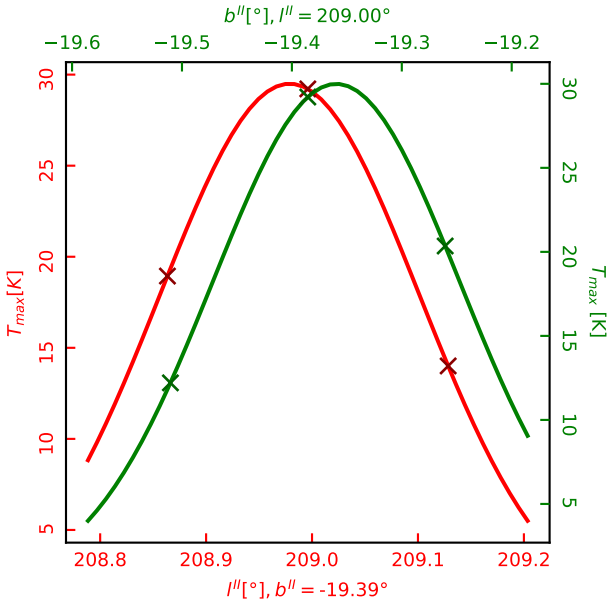


Figura 7: Ajusta gaussiano de temperatura máxima

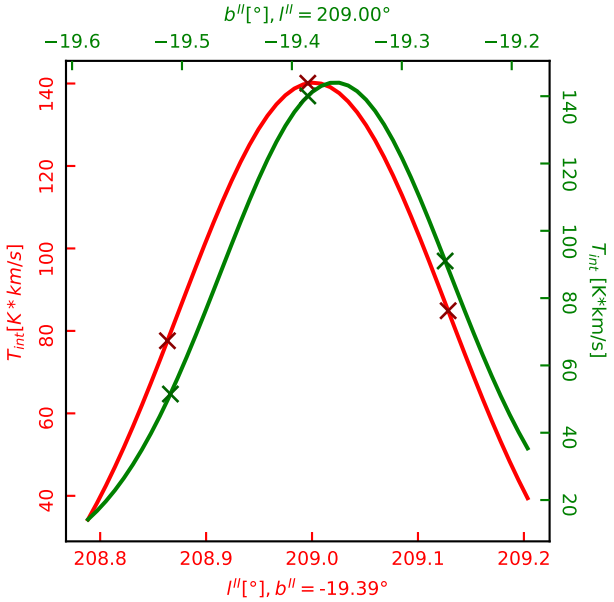


Figura 8: Ajuste gaussiano de temperatura integrada

## 5. Conclusiones

Se concluye gracias a tabla 2 que, a menor elevación, mayor es la potencia, pues la cantidad de atmósfera que se observa y, por lo tanto, su cantidad de emisión observada, es también mayor.

# 6. Anexo

Código 1: Hot–Cold Test

```python
#!/usr/bin/python3

def dBm2W(W):
    """Converts an arbitrary power ‘W‘ in dBm to W."""
    return 10 ** ((W - 3) / 10)

def T_rec(T_hot, T_cold, y_factor):
    """Calculates receiver noise temperature via hot & cold temperatures and y factor."""
    return (T_hot - y_factor * T_cold) / (y_factor - 1)

def Y_factor(P_hot, P_cold):
    """Y factor via hot & cold power in W."""
    return P_hot / P_cold

T_hot = 300 # K
T_cold = 77 # K

W_hot = -44.5 # dBm
W_cold = -47.94 # dBm

Y = Y_factor(dBm2W(W_hot), dBm2W(W_cold))

print("Y = {}".format(Y))
print("T_rec = {} K".format(T_rec(T_hot, T_cold, Y)))
```

Código 2: Antenna Dipping

```python
#!/usr/bin/python3

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_excel("../antdip_AE2021A.xlsx", engine='openpyxl')

_secz = df.iloc[:-1, 3]
lndP = df.iloc[:-1, 7]

p = np.poly1d(np.polyfit(_secz, lndP, 1))
print("tau_w = {}".format(p.coefficients[0]))
print("ln(deltaW) = -sec(z) * {} + {}". format(*p.coefficients))

plt.rcParams.update({'font.size': 7})
fig, ax = plt.subplots(figsize=(3.25, 3.25))

ax.plot(_secz, p(_secz), label="y = {1:.3f} + x * {0:.3f}".format(*p.c), color="red")
ax.plot(_secz, lndP, label="Datos", marker='x', fillstyle="none", linestyle="", color="green")
ax.legend()

ax.set_xlabel(r"$-\sec(\varphi)$")
ax.set_ylabel(r"$\ln(\Delta W)$")
ax.yaxis.set_tick_params(rotation=90)
ax.tick_params(direction="in", top=True, right=True)

fig.savefig("../informe/taufit.pdf")
plt.show()
```

Código 3: Observaciones

```python
#!/usr/bin/python3

import glob
import itertools
import numpy as np
import scipy.optimize
```

```
import scipy.integrate
import astropy.stats
import matplotlib.pyplot as plt

paths = sorted(glob.glob("../sec_mierc_sem1_2021/sdf*"))

lii = np.zeros(len(paths))
bii = np.zeros(len(paths))
v = np.zeros((len(paths), 364-109+1))
T = np.zeros((len(paths), 364-109+1))
for i, path in enumerate(paths):
    with open(path) as f:
        [_, lii[i]], [_, bii[i]] = np.genfromtxt(itertools.islice(f, 22, 24, None))
        v[i], T[i] = np.genfromtxt(path, unpack=True, skip_header=108)

liirad = np.radians((lii - 180) % 360 - (360 - 180))
biirad = np.radians((bii - 180) % 360 - (360 - 180))

plt.rcParams.update({'font.size': 5})
fig = plt.figure(figsize=(3.25, 3.25/2))
ax = fig.add_subplot(111, projection="mollweide")
ax.grid(True)
ax.scatter(liirad, biirad, marker='.', color="red")
fig.savefig("../informe/lb.pdf")
plt.show()

plt.rcParams.update({'font.size': 7})
fig, ax = plt.subplots(figsize=(3.25, 3.25))
ax.scatter(lii, bii, color="red")
ax.set_xlabel(r"$l^{II} [\degree]$")
ax.set_ylabel(r"$b^{II} [\degree]$")
ax.yaxis.set_tick_params(rotation=90)
ax.tick_params(direction="in", top=True, right=True)
fig.savefig("../informe/cruz.pdf")
plt.show()

def gaussian(x, height, center, width):
    return height * np.exp(-0.5 * ((x - center) / width) ** 2)

Tmax = np.zeros((5, 3))
for j in range(3):
    fig, axs = plt.subplots(3, 3, sharex=True, sharey=True, figsize=(3.25, 3.25))
    axs = axs.flatten()
    for i in range(9):
        axs[i].set_axis_off()
    for i, n in enumerate([2, 4, 5, 6, 8]):
        axs[n-1].set_axis_on()
        axs[n-1].set_ylim((-1,32))
        axs[n-1].yaxis.set_tick_params(rotation=90)
        axs[n-1].tick_params(direction="in", top=True, right=True)
        popt, pcov = scipy.optimize.curve_fit(gaussian, v[5*j+i], T[5*j+i], p0=[20, 10, 1])
        axs[n-1].plot(v[5*j+i], T[5*j+i], color="green", drawstyle="steps-mid", linestyle="-",
    linewidth=0.5)
        axs[n-1].plot(v[5*j+i], gaussian(v[5*j+i], *popt), color="red", drawstyle="default",
    linestyle="-", linewidth=0.5)
        Tmax[i, j] = T[5*j+i].max()
        plt.text(0.3, 0.85, r"$l^{II}=$"+f"{lii[5*j+i]:.2f}", fontsize=5, ha='center', va='center',
     transform=axs[n-1].transAxes)
        plt.text(0.3, 0.75, r"$b^{II}=$"+f"{bii[5*j+i]:.2f}", fontsize=5, ha='center', va='center',
     transform=axs[n-1].transAxes)
    axs[8-1].set_xlabel(r"Velocidad [km/s]")
    axs[4-1].set_ylabel(r"Temperatura [K]")
    plt.subplots_adjust(wspace=0, hspace=0)
    fig.savefig("../informe/specfit{}.pdf".format(j+1))
    plt.show()

Tmax = Tmax.mean(axis=1)
fig = plt.figure(figsize=(3.25, 3.25))

# https://doi.org/10.1109/MSP.2011.941846
p = np.poly1d(np.polyfit(lii[0:5][[1, 2, 3]], np.log(Tmax[[1, 2, 3]]), 2))
heightlii = np.exp(p.c[2]-p.c[1]*p.c[1]/(4*p.c[0]))
```

```
centerlii = -p.c[1]/(2*p.c[0])
widthlii = np.sqrt(-1/(2*p.c[0]))

xlii = np.linspace(lii.min()-0.075, lii.max()+0.075)
ylii = gaussian(xlii, heightlii, centerlii, widthlii)

axlii = fig.add_subplot(111, label=r"$l^{II}$")
axlii.plot(xlii, ylii, color="red")
axlii.plot(lii[0:5][[1, 2, 3]], Tmax[[1, 2, 3]], marker='x', color="darkred", linestyle="")
axlii.set_xlabel(r"$l^{II} [\degree], b^{II}=$"+f"{bii[2]:.2f}"+r"$\degree$", c="red")
axlii.set_ylabel(r"$T_{max} [K]$", color="red")
axlii.tick_params(axis='x', colors="red")
axlii.tick_params(axis='y', colors="red")
axlii.yaxis.set_tick_params(rotation=90)
axlii.tick_params(direction="in")

p = np.poly1d(np.polyfit(bii[0:5][[0, 2, 4]], np.log(Tmax[[0, 2, 4]]), 2))
heightbii = np.exp(p.c[2]-p.c[1]*p.c[1]/(4*p.c[0]))
centerbii = -p.c[1]/(2*p.c[0])
widthbii = np.sqrt(-1/(2*p.c[0]))
xbii = np.linspace(bii.min()-0.075, bii.max()+0.075)
ybii = gaussian(xbii, heightbii, centerbii, widthbii)
axbii = fig.add_subplot(111, label=r"$b^{II}$", frame_on=False)
axbii.plot(xbii, ybii, color="green")
axbii.plot(bii[0:5][[0, 2, 4]], Tmax[[0, 2, 4]], marker='x', color="darkgreen", linestyle="")
axbii.xaxis.tick_top()
axbii.yaxis.tick_right()
axbii.set_xlabel(r"$b^{II} [\degree], l^{II}=$"+f"{lii[2]:.2f}"+r"$\degree$", c="green")
ylabelaxbii = axbii.set_ylabel(r"$T_{max}$ [K]", color="green")
ylabelaxbii.set_rotation(-90)
axbii.xaxis.set_label_position('top')
axbii.yaxis.set_label_position('right')
axbii.tick_params(axis='x', colors="green")
axbii.tick_params(axis='y', colors="green")
axbii.yaxis.set_tick_params(rotation=-90)
axbii.yaxis.set_label_coords(1.1175, 0.5)
axbii.tick_params(direction="in")

fig.savefig("../informe/tmax.pdf")
plt.show()

Tint = -scipy.integrate.simpson(T, v, axis=1).reshape((3, 5)).T.mean(axis=1)
fig = plt.figure(figsize=(3.25, 3.25))
p = np.poly1d(np.polyfit(lii[0:5][[1, 2, 3]], np.log(Tint[[1, 2, 3]]), 2))
heightlii = np.exp(p.c[2]-p.c[1]*p.c[1]/(4*p.c[0]))
centerlii = -p.c[1]/(2*p.c[0])
widthlii = np.sqrt(-1/(2*p.c[0]))
xlii = np.linspace(lii.min()-0.075, lii.max()+0.075)
ylii = gaussian(xlii, heightlii, centerlii, widthlii)
axlii = fig.add_subplot(111, label=r"$l^{II}$")
axlii.plot(xlii, ylii, color="red")
axlii.plot(lii[0:5][[1, 2, 3]], Tint[[1, 2, 3]], marker='x', color="darkred", linestyle="")
axlii.set_xlabel(r"$l^{II} [\degree], b^{II}=$"+f"{bii[2]:.2f}"+r"$\degree$", c="red")
axlii.set_ylabel(r"$T_{int} [K*km/s]$", color="red")
axlii.tick_params(axis='x', colors="red")
axlii.tick_params(axis='y', colors="red")
axlii.yaxis.set_tick_params(rotation=90)
axlii.tick_params(direction="in")
p = np.poly1d(np.polyfit(bii[0:5][[0, 2, 4]], np.log(Tint[[0, 2, 4]]), 2))
heightbii = np.exp(p.c[2]-p.c[1]*p.c[1]/(4*p.c[0]))
centerbii = -p.c[1]/(2*p.c[0])
widthbii = np.sqrt(-1/(2*p.c[0]))
xbii = np.linspace(bii.min()-0.075, bii.max()+0.075)
ybii = gaussian(xbii, heightbii, centerbii, widthbii)
axbii = fig.add_subplot(111, label=r"$b^{II}$", frame_on=False)
axbii.plot(xbii, ybii, color="green")
axbii.plot(bii[0:5][[0, 2, 4]], Tint[[0, 2, 4]], marker='x', color="darkgreen", linestyle="")
axbii.xaxis.tick_top()
axbii.yaxis.tick_right()
axbii.set_xlabel(r"$b^{II} [\degree], l^{II}=$"+f"{lii[2]:.2f}"+r"$\degree$", c="green")
ylabelaxbii = axbii.set_ylabel(r"$T_{int}$ [K*km/s]", color="green")
ylabelaxbii.set_rotation(-90)
```

```
axbii.xaxis.set_label_position('top')
axbii.yaxis.set_label_position('right')
axbii.tick_params(axis='x', colors="green")
axbii.tick_params(axis='y', colors="green")
axbii.yaxis.set_tick_params(rotation=-90)
axbii.yaxis.set_label_coords(1.1175, 0.5)
axbii.tick_params(direction="in")
fig.savefig("../informe/tint.pdf")
plt.show()

masked = astropy.stats.sigma_clip(T, axis=1, maxiters=None)
rms = np.sqrt(masked.mean(axis=1)**2).reshape((3, 5)).T
meanmasked = astropy.stats.sigma_clip((T[0:5]+T[5:10]+T[10:15])/3, axis=1, maxiters=None)
rmsmean = np.sqrt(meanmasked.mean(axis=1)**2)
print(rms / rmsmean.reshape((5, 1)))
```