Informe sobre las tareas de la auxiliatura de Pykinemetry

Laura Natalia Martínez Ramírez

1. Tarea 1

1.1. Mapa de velocidad ideal

La primera parte consistió en crear una función para obtener un mapa de velocidad de una galaxia circular a partir del perfil de velocidad más simple:

$$v_{cir}(r) = \begin{cases} v_t \cdot \frac{r}{r_t} & r \leq r_t \\ v_t & r > r_t \end{cases},$$

donde $\mathbf{r_t}$ es el radio en el que la galaxia alcanza su máxima velocidad y $\mathbf{v_t}$ el valor de la máxima velocidad. Esta función llamada **ideal_vel_map** tiene como parámetros de entrada: el número de filas y columnas de la matriz cuadrada (**npix**), el radio máximo de la galaxia ($\mathbf{r_{max}}$), $\mathbf{r_t}$ y $\mathbf{v_t}$. El *output* es una matriz con el mapa de velocidad de la galaxia y valores **nan** por fuera de esta. Es importante destacar que a esta función se le agregó un pequeño borde del 3 % para que los límites de la galaxia no quedaran justo en el borde de la matriz (estética).

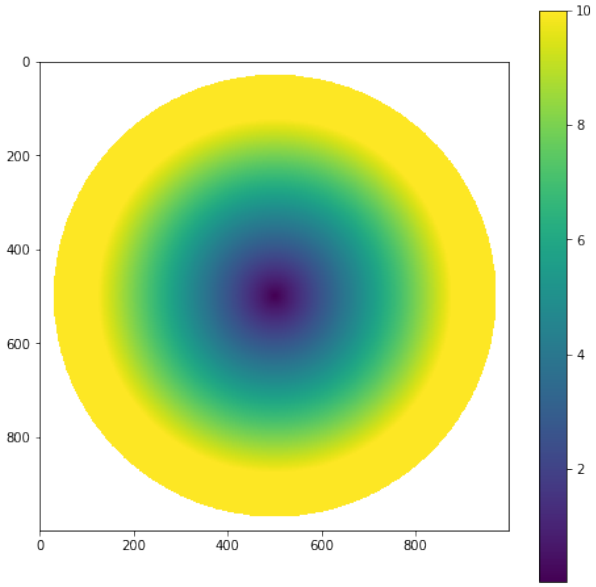

Figura 1: Resultado de la función para: $npix = 1000$, $r_{max} = 50$, $r_t = 40$ y $v_t = 10$.

1.2. Mapa de velocidad observado

En la segunda parte de esta tarea, se requeria construir una función que retorne el mapa de velocidad para una galaxia inclinada y rotada. Para tal fin, es necesario encontrar la velocidad a lo largo de la línea de visión, definida como:

$$V_{LOS}(R) = V_{sys} + V_{cir}(r) \times \frac{sin(i)cos(\psi - \psi_0)}{\alpha} \tag{1}$$

donde $V_{cir}(r)$ corresponde al perfil radial de velocidad circular de la galaxia, i es el ángulo de inclinación respecto a la línea de visión, Ψ_0 el ángulo de posición o ángulo entre el semieje mayor de la galaxia y la horizontal,Ψ el ángulo de cada pixel respecto a la horizontal y V_{sys} la velocidad sistémica o de traslación de la galaxia. Además α está definido como:

$$\alpha = \sqrt{cos^2(\psi - \psi_0) + \frac{sin^2(\psi - \psi_0)}{cos^2(i)}}, \tag{2}$$

y r como:

$$r = R\alpha, \tag{3}$$

donde R es la distancia observada de cada punto de la galaxia al centro (distancias reales) y r es cómo lucen estas distancias teniendo en cuenta la inclinación y la rotación. Esta velocidad (ecuación 1) corresponde a la velocidad medida de la galaxia por medio del efecto Doppler, por lo que solo tiene en cuenta los movimientos a lo largo de la línea de visión. Para obtener V_{LOS}, primero se definieron las matrices principales que posteriormente se operan de acuerdo con la ecuación 1. Estas matrices son: Ψ *(figura 2 arriba izquierda)*, α *(figura 2 arriba derecha)*, R *(figura 2 abajo izquierda)* y r *(figura 2 abajo derecha)*.

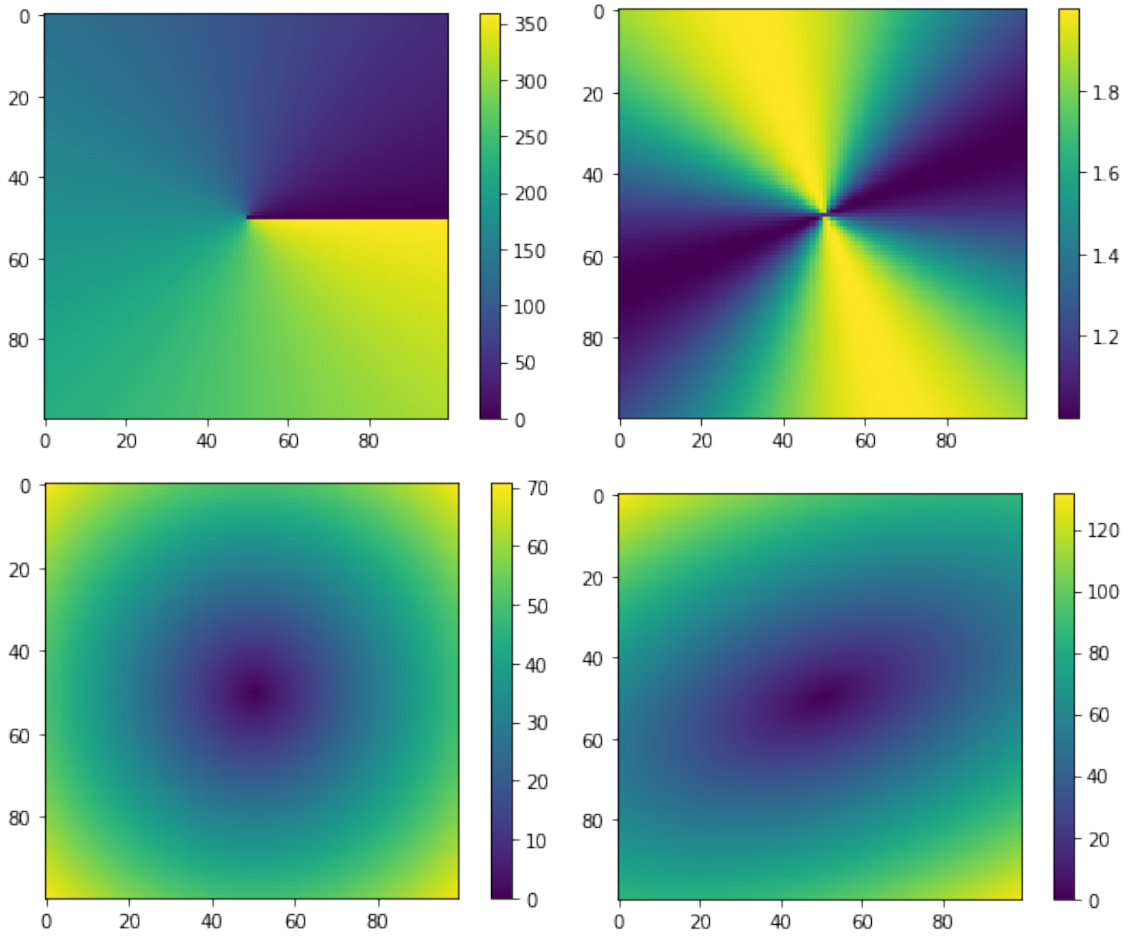

Figura 2: Matrices necesarias para construir el mapa de velocidad a lo largo de la línea de visión: Ψ *(arriba izquierda)*, α *(arriba derecha)*, R *(abajo izquierda)* y r *(abajo derecha)*

Una vez definidas estas matrices, se agrega el perfil radial de velocidad circular. En este caso, se decidió cambiar el perfil usado anteriormente por uno exponencial con derivada continua (figura 3 izquierda). Aplicando las respectivas operaciones a estas matrices según la ecuación (1) y reemplazando por *nan* los valores de la matriz ubicados fuera de la galaxia, se obtiene el mapa de velocidad (figura 3 derecha).

Por último, se construyó una función llamada **obs_vel_map** encargada de realizar el procedimiento previamente explicado para calcular V_{LOS} y cuyos parámetros de entrada son: el número de filas y columnas de la matriz cuadrada (**npix**), el radio máximo de la galaxia ($\mathbf{r_{max}}$), el radio del decaimiento exponencial ($\mathbf{r_t}$), la velocidad máxima a la que tiende el perfil de velocidad ($\mathbf{v_t}$), el ángulo de inclinación (**incl**), el ángulo de posición ($\mathbf{\Psi_0}$) y la velocidad sistémica ($\mathbf{V_{sys}}$). El *output* es una matriz con el mapa de velocidad observada (V_{LOS}) de la galaxia y valores **nan** por fuera de esta.

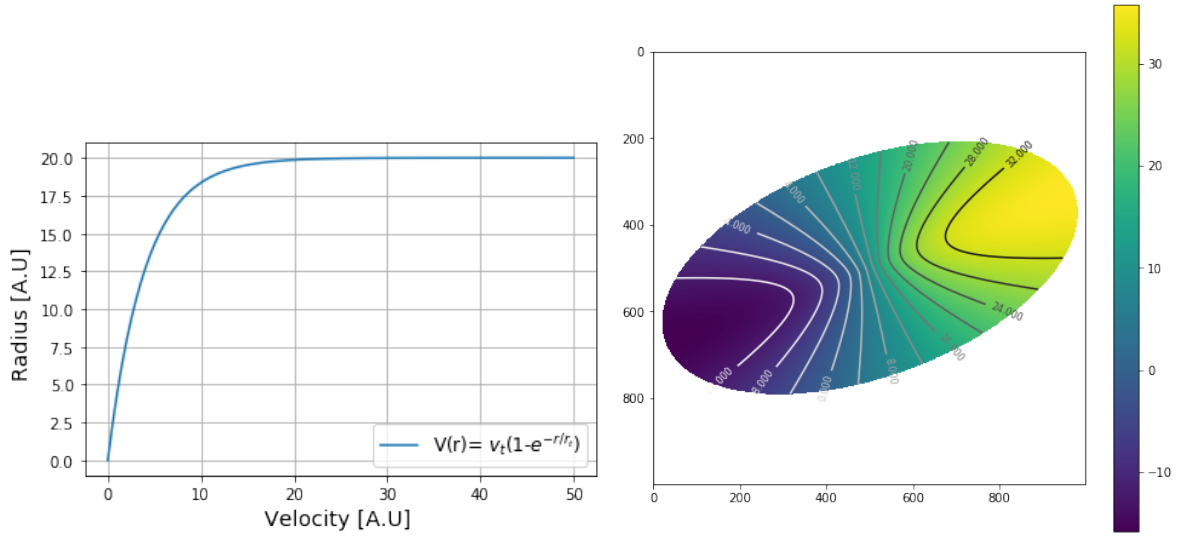

Figura 3: Perfil de velocidad circular usado para obtener V_{LOS} (izquierda) y mapa de velocidad V_{LOS} de la galaxia inclinada y rotada (derecha).

2. Tarea 2

Para esta tarea recibí dos mapas de velocidad de dos galaxias, uno proveniente de una simulación computacional (figura 4 izquierda), y el otro observado en una galaxia real (figura 4 derecha). El objetivo de esta tarea es ajustar el modelo creado en la tarea anterior (**obs_vel_map**) y obtener el conjunto de parámetros ($\mathbf{r_{max}}$, $\mathbf{r_t}$,$\mathbf{v_t}$, **incl**, $\mathbf{\Psi_0}$, $\mathbf{V_{sys}}$) que minimicen el residuo entre el modelo y los datos.

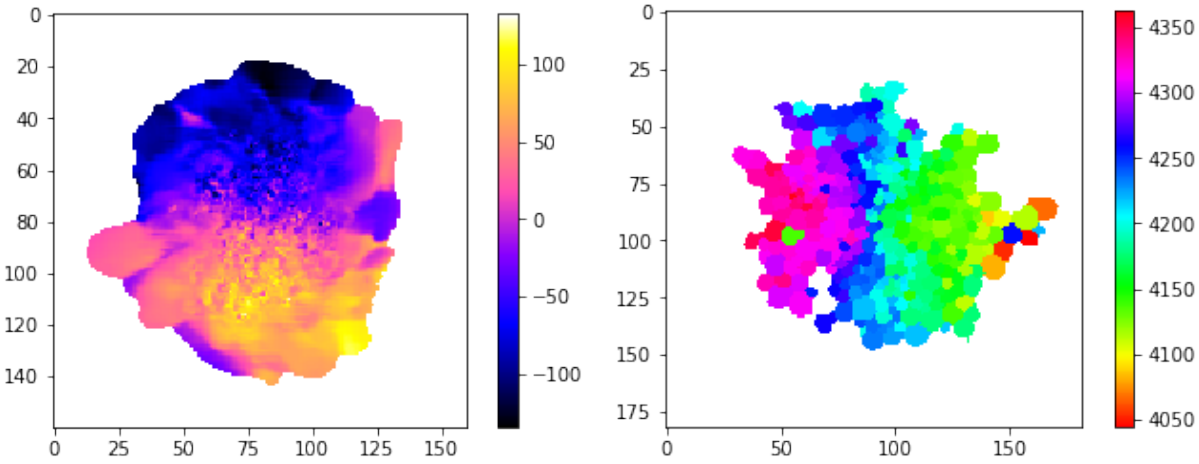

Figura 4: Mapas de velocidad a ajustar: simulado (izquierda) y observado (derecha).

Para realizar el ajuste por medio de la función **optimize.curve_fit** de *scipy* y que este funcione, se deben realizar 4 cambios sustanciales tanto a los datos como al modelo. Estos son:

- Reemplazar los valores *nan* de los datos (ubicados fuera de la galaxia) por el valor promedio de la matriz calculado sin tener en cuenta los valores no finitos (nan).
- De forma consistente con el cambio anterior, en la función se programó que se reemplacen por V_{sys} los valores de la matriz V_{LOS} ubicados fuera de la galaxia.

- El parámetro de entrada *npix* fue cambiado por xx, una matriz cuadrada del mismo tamaño que los datos que contiene las posiciones de las filas. De esta manera, el contenido de la fila i es [i, i, i, ...], donde la cantidad de ies dentro del arreglo, corresponde al número de columnas.
- El *output* de la función ya no es una matriz, sino un arreglo de numpy en una dimensión. Para esto se usó la función **np.ravel()**, de forma que si la matriz tenía dimensiones nxn, ahora será $1xn^2$.

Con estos cambios realizados, se procede a realizar el ajuste de la primera matriz de datos (figura 4 izquierda):

```
v = np.loadtxt("velmap.txt")
v_max = np.nanmax(v)
v_min = np.nanmin(v)
v[np.isnan(v)]= np.nanmean(v)   #nan values are changed for the mean value of the matrix

x = np.arange(160) #number of rows of data
y = np.arange(160) #number of columns of data
yy,xx = np.meshgrid(x,y)

popt,pconv = optimize.curve_fit(obs_vel_map, xx, v.ravel(), bounds = ([40, 5, 100, -90,
200, -10], [80, 80, 500, 90, 360, 5]), method = 'trf')
```

Es importante destacar que el ajuste con el **initial guess**, es decir introduciendo el conjunto de parámetros que creemos cercanos a los reales es bastante sensible a los valores ingresados, por lo que no es muy fiable. Por tal razón, se decidió realizar un ajuste con el método **trf** que requiere como ingreso los valores de los límites inferiores y superiores de los parámetros. Adicionalmente, a la matriz de los datos también debe aplicarsele la función $np.ravel()$ para el ajuste. Los parámetros encontrados fueron: $\mathbf{r_{max} = 60{,}0}$, $\mathbf{r_t = 9{,}36626773}$,$\mathbf{v_t = 246{,}61477184}$, $\mathbf{incl = 18{,}31393324}$, $\mathbf{\Psi_0 = 276{,}46496982}$, $\mathbf{V_{sys} = -3{,}3606885}$. Con estos valores de los parámetros se obtuvo el modelo de la figura 5 (izquierda) y al restarse con la matriz original de datos se obtuvo la matriz de la figura 5 (derecha).

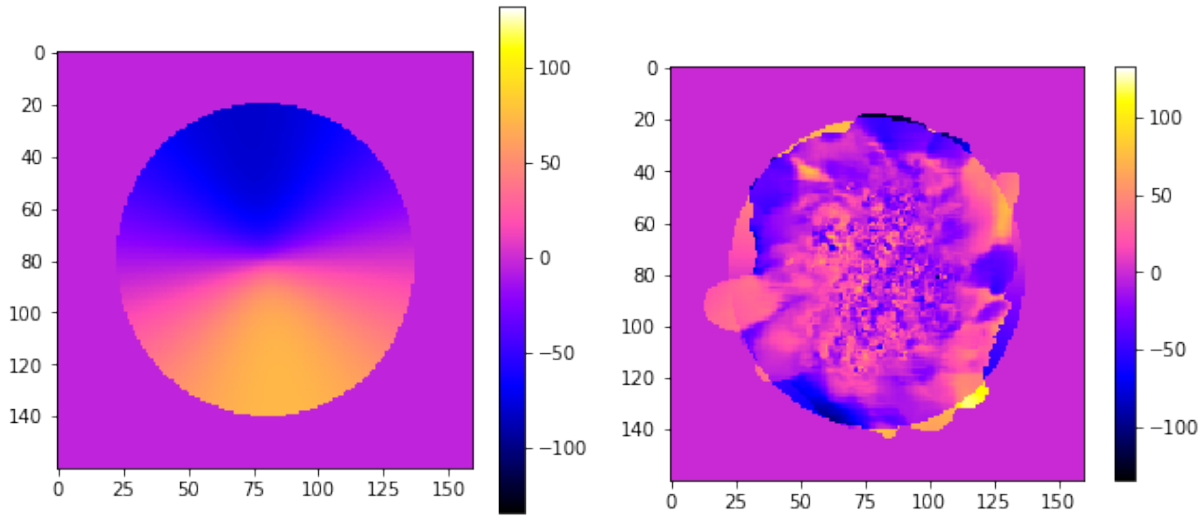

Figura 5: Mapa de velocidad obtenido a partir de los parámetros encontrados en el ajuste y el modelo dado por la función **obs_vel_map** (izquierda) y diferencia entre los mapas de velocidad del modelo ajustado y de los datos (derecha).

Luego se procedió a graficar el histograma de error obtenido de la resta (figura 6)). En esta grafica se observa un histograma de error centrado muy cerca a 0 y con una cintura estrecha. Además, al analizar la figura 5 (derecha) puede verse que muchos de estos errores se generan debido a que el contorno de la galaxia de los datos no corresponde a una elipse, como se esperaría.

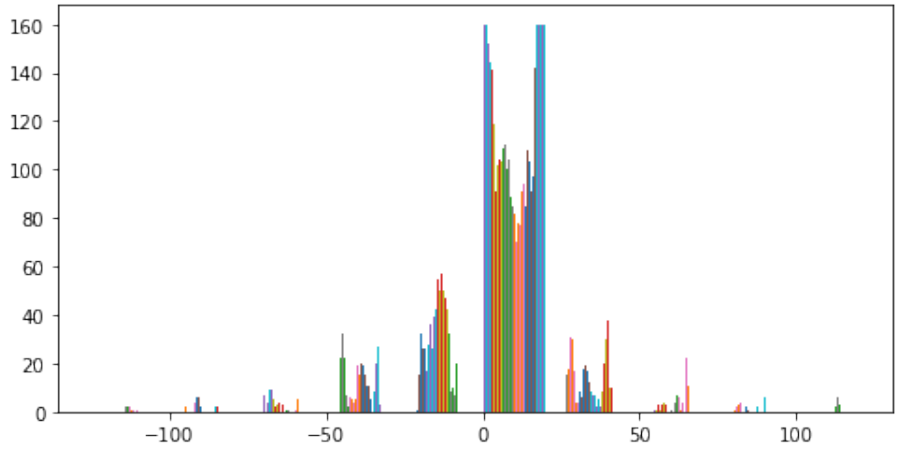

Figura 6: Histograma de error en el ajuste del mapa de velocidad de la galaxia simulada.

A continuación se repitió el mismo procedimiento con los datos de la galaxia observada (figura 4 derecha). En este caso las dimensiones de la matriz eran $182x182$ y los límites de los parámetros fueron:

```
popt2,pconv2 = optimize.curve_fit(obs_vel_map, xx, velmap.ravel(), bounds = ([40, 15, 200,
25, 100, 4050], [85, 60, 350, 60, 200, 4350]), method = 'trf')
```

Los parámetros encontrados por la función **optimize.curve_fit** fueron: $\mathbf{r_{max} = 62{,}5}$, $\mathbf{r_t = 45{,}2469792}$,$\mathbf{v_t = 262{,}85318776}$, $\mathbf{incl = 30{,}75825197}$, $\mathbf{\Psi_0 = 173{,}4111547}$, $\mathbf{V_{sys} = 4220{,}26094556}$. Con estos valores de los parámetros se obtuvo el modelo de la figura 7 (izquierda) y al restarse con la matriz original de datos se obtuvo la matriz de la figura 7 (derecha).

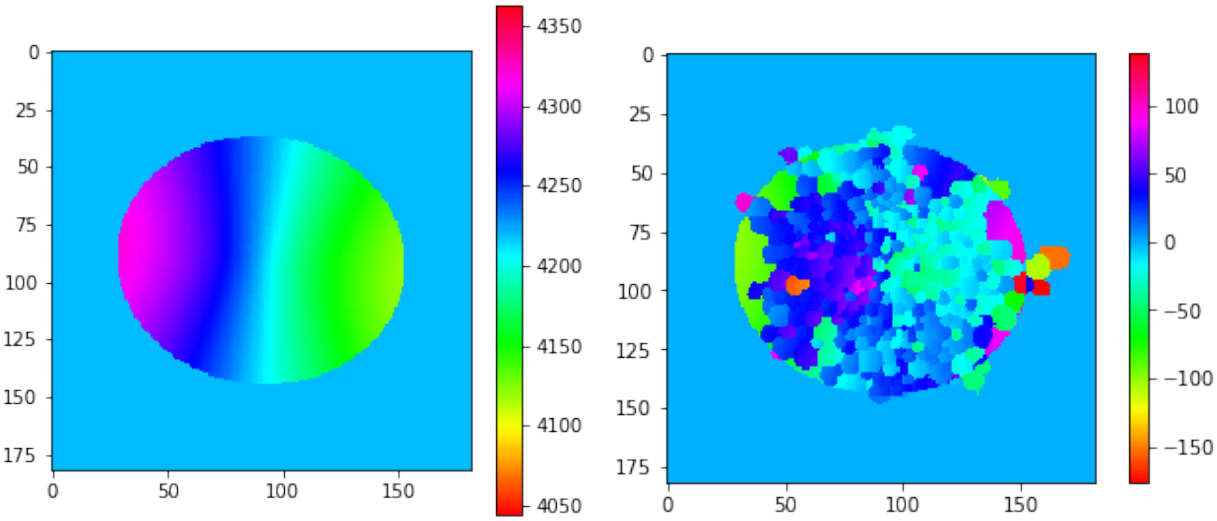

Figura 7: Mapas de velocidad a ajustar: simulado (izquierda) y observado (derecha).

Luego se procedió a graficar el histograma de error obtenido de la resta (figura 8)). En esta grafica se observa un histograma de error centrado muy cerca a 0 pero desviado un poco hacia valores mayores y con una cintura estrecha. Además, al analizar la figura 7 (derecha) puede verse que muchos de estos errores se generan debido la mitad izquierda de la galaxia presenta una velocidad un poco menor que la de los datos. En consecuencia, esa zona con velocidad entre 0 y 50, es la encargada de desplazar el máximo de la distribución de error.

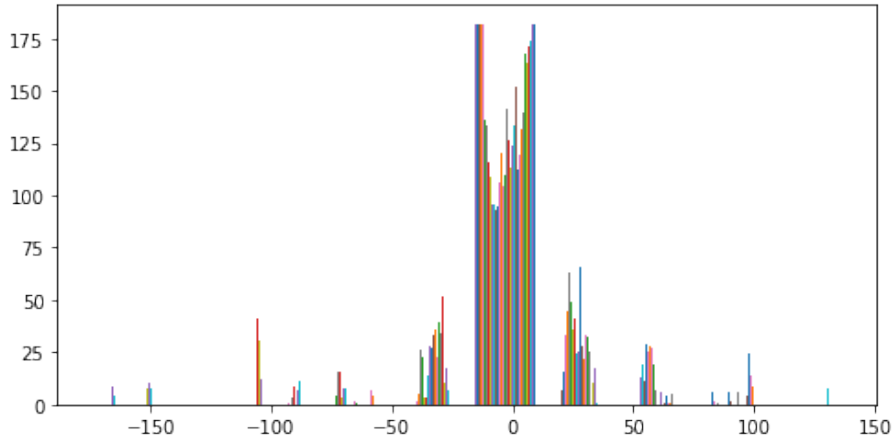

Figura 8: Histograma de error en el ajuste del mapa de velocidad de la galaxia observada.

De forma general puede decirse que los resultados de los ajustes son buenos porque ambos histogramas de error decaen rápido alrededor de 25 aproximadamente, lo cual corresponde aproximadamente al 10 % de V_t.

2.1. Aspectos importantes de la implementación

Es importante destacar que los cambios a la función y a los datos mencionados anteriormente, se realizaron luego de intentar diferentes alternativas. De estas, es importante mencionar dos casos:

- **Ignorar valores nan con bounds:** para esto se usó la función **optimize.curve_fit** con los métodos **trf** y **dogbox**, los bounds de los parámetros y la opción **check_finite = False**. Además, se mantuvo el valor nan fuera de la galaxia en la función **obs_vel_map**. En este caso el resultado obtenido fue: **ValueError: Residuals are not finite in the initial point.**
- **Ignorar valores nan con initial guess:** para esto se usó la función **optimize.curve_fit** con el método **lm** (predeterminado), el initial guess de los parámetros y la opción **check_finite = False**. Además, se mantuvo el valor nan fuera de la galaxia en la función **obs_vel_map**. En este caso el código se ejecutó sin ningún tipo de error. Sin embargo, absolutamente todos los parámetros encontrados correspondian exactamente al **initial guess**, sin importar cúal fuera este.

Creo que el problema de los nan proviene del hecho de que la función también devuelve una matriz con valores nan y no se trata solo de falta de información en la matriz a ajustar.