Centro Nacional de Desarrollo e Investigación en Tecnologías Libres (CENDITEL)
Revista Electrónica Conocimiento Libre y Licenciamiento (CLIC), Mérida – Venezuela
ISSN: 2244-7423

Python como herramienta para el análisis de las imágenes astronómicas del Telescopio Reflector

Python as a tool for the analysis of astronomical images from the Reflector Telescope

Jaidary Mejía [1]
Giuliat Navas [2]
Universidad Nacional Abierta, Trujillo, Venezuela[1]
Fundación Centro de Investigaciones de Astronomía Francisco J. Duarte, Mérida, Venezuela[1,2]
jaidarymejia30@gmail.com[1]
giuliatnavas@gmail.com[2]

Fecha de recepción: 26/04/2024
Fecha de aceptación: 13/06/2024
Pág: 23 – 49
DOI: 10.5281/zenodo.18186139

Resumen

Las imágenes astronómicas observadas a través de telescopios terrestres juegan un rol importante para el conocimiento del universo, ya que a través de la observación y el análisis de los cuerpos celestes, se contribuye significativamente con informaciones valiosas para la comprensión de la formación y la evolución del universo. Este trabajo emplea la programación con Python como herramienta para el análisis de las imágenes astronómicas, utilizando Astropy como biblioteca principal, permitiéndonos crear diversos códigos destinados al análisis y a la caracterización de las imágenes astronómicas, las cuales fueron aplicadas a observaciones obtenidas a través del telescopio Reflector, de 1m de diámetro, ubicado en el Observatorio Astronómico Nacional (OAN) Llano del Hato, en el estado Mérida. En el presente trabajo, los códigos creados en Python pueden ser empleados por estudiantes, aficionados o investigadores para el procesamiento, la calibración y el análisis de las imágenes astronómicas, todo ello con el fin de extraer información relevante para los diferentes proyectos de investigación. Por razones de acceso, Pyhton es un lenguaje de alto nivel con un eficiente tiempo de ejecución para procesar y analizar datos, lo cual es ideal para procesar un volumen importante de imágenes astronómicas de manera eficiente.

Es conocido que las imágenes astronómicas deben ser tratadas previamente para corregir efectos no deseados introducidos por el telescopio durante la observación, es por ello que, en este trabajo se plantea entre otros, un código para la calibración en volumen de las imágenes a través de la creación de los bias, dark y flat maestros. Adicionalmente se presentan códigos como herramientas para hacer estudios estadísticos y cálculos de los perfiles en x y en y, de las imágenes astronómicas, con el fin de comprender y visualizar la distribución de la intensidad luminosa presente en la imagen. La aplicación de estas herramientas computacionales y actualizadas son fundamentales para obtener una comprensión más profunda de los objetos celestes observados y sirven para evaluar la calidad de las imágenes obtenidas a través de los telescopios, cumpliéndose con el tratamiento previo de las imágenes astronómicas y contribuyendo así, al avance de la investigación astronómica y al entendimiento del universo.

Palabras clave: imágenes astronómicas, astronomía, Astropy, calibración, Python.

Abstract

Astronomical images observed through ground-based telescopes play an important role in the understanding of the universe, as through the observation and analysis of celestial bodies, it significantly contributes valuable information for the comprehension of the formation and evolution of the universe. This work employs programming with Python as a tool for the analysis of astronomical images, using Astropy as the main library, allowing us to create various codes aimed at the analysis and characterization of astronomical images, which were applied to observations obtained through the 1m diameter Reflector telescope, located at the National Astronomical Observatory (OAN) Llano del Hato, in the State of Mérida. In the present work, the codes created in Python can be used by students, amateurs, or researchers for the processing, calibration, and analysis of astronomical images, all with the aim of extracting relevant information for various research projects. For accessibility reasons, Python is a high-level language with efficient run-time for processing and analyzing data, which is ideal for efficiently processing a significant volume of astronomical images. It is known that astronomical images must be prepossessed to correct unwanted effects introduced by the telescope during observation. Therefore, in this work, among other things, a code is proposed for the calibration of images through the creation of master bias, dark, and flat frames. Additionally, codes are presented as tools for conducting statistical studies and calculating the profiles in x and y of astronomical images, in order to understand and visualize the distribution of the luminous intensity present in the image. The application of these computational and updated tools are fundamental to gain a deeper understanding of the observed celestial objects and serve to assess the quality of the images obtained through telescopes, fulfilling the prepossessing of astronomical images and thus contributing to the advancement of astronomical

research and the understanding of the universe

Keywords: astronomical images, astronomy, Astropy, calibration, Python.

Introducción

La astronomía contemporánea se desenvuelve en un entorno donde la proliferación de datos juega un papel fundamental en la investigación científica. La adopción de telescopios modernos equipados con dispositivos de carga acoplada *CCD* (*Charge Coupled Device*) ha transformado la captura de imágenes astronómicas, generando grandes volúmenes de datos que varían según las técnicas y tipos de telescopios empleados. Estas cámaras CCD han ampliado significativamente las posibilidades de recopilación de datos tanto desde telescopios terrestres como desde satélites espaciales.

Las imágenes astronómicas constituyen una fuente primordial de datos. Estas imágenes se representan como matrices bidimensionales de valores de píxeles, donde cada valor corresponde a una medida de brillo (Craig, 2023). Normalmente, estas imágenes vienen acompañadas de metadatos que proporcionan información contextual sobre las condiciones de adquisición, como el telescopio utilizado, la duración de la exposición y la ubicación en el cielo del objeto observado (Pössel, 2019).

Aunque el telescopio se considera el instrumento más emblemático de los astrónomos, en términos de tiempo de uso, las computadoras podrían asumir ese título al procesar y analizar todos los datos recopilados. Los científicos dependen en gran medida del software, el cual crean, utilizan y modifican continuamente. El lenguaje más citado en las publicaciones revisadas por pares en la década de 1990 era *Fortran*. Una década más tarde, *IDL* tomó la delantera. En la actualidad, Python ha ocupado esa posición preeminente, como se observa en la Figura 1, que muestra la evolución del uso de diferentes lenguajes de programación según las citas en revistas indexadas en la base de datos *Astrophysics Data System* (ADS) (Astronomical Society of the Pacific, 2024).

En la actualidad, el incremento en el volumen de datos astronómicos generados por telescopios modernos ha generado una creciente demanda de herramientas eficientes para su análisis. Por ejemplo, el telescopio de Sondeo ESO VISTA produce aproximadamente 150 TB de datos anualmente (Pössel, 2019). Esta situación ha propiciado el desarrollo y la adopción de herramientas en línea, como los *Jupyter Notebooks*, que se han convertido en elementos esenciales para realizar operaciones y análisis de datos de manera remota y colaborativa (Strasbourg Astronomical Data Center, 2020). Una de las ventajas clave de usar Jupyter Notebooks es su flexibilidad. Los usuarios podemos emplear una variedad de lenguajes de programación, incluyendo Python, que se destaca en este proyecto debido a su versatilidad. Además, los *Jupyter*

Notebooks ofrecen una plataforma interactiva para la visualización de datos astronómicos. Con la integración de bibliotecas de visualización como *Matplotlib* se crean gráficos, diagramas y visualizaciones interactivas para explorar y comunicar los resultados de manera efectiva.

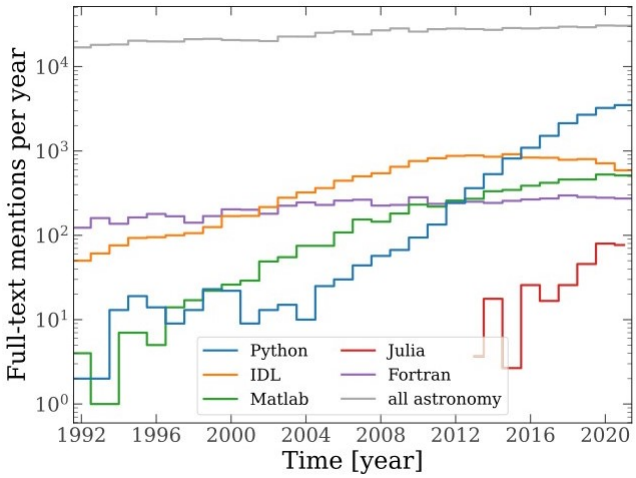

Figura 1: Lenguajes de programación citados anualmente (indicados en la leyenda de la figura), en referencia a las publicaciones de la base de datos de *Astrophysics Data System* (ADS). El eje x representa, el paso de los años, el eje y representa, el número de citaciones de los siguientes lenguajes de programación: Python, IDL, Matlab, Julia, Fortran, y cualquier otro empleado en la astronomía.

Fuente: Astronomical Society of the Pacific (2024).

La elección de Python como lenguaje de programación para este propósito se basa en su versatilidad y en la amplia gama de bibliotecas especializadas disponibles, debido a que Python ofrece una sintaxis clara y legible que facilita tanto la escritura como la comprensión del código. Una de las ventajas de utilizar Python es que es un lenguaje multipropósito, lo que significa que se puede emplear para una amplia variedad de tareas. Esto permite utilizar las mismas herramientas y habilidades para diferentes proyectos, lo que aumenta la eficiencia y productividad (Aldcroft et al., 2013).

En el año 2013, una red de colaboradores introdujo una biblioteca diseñada para facilitar tareas rutinarias en la investigación astronómica, como la manipulación y transformación de coordenadas, denominada Astropy. Actualmente, Python cuenta con muchas bibliotecas especializadas para el análisis de datos astronómicos, además de Astropy, se ha encontrado otras bibliotecas populares como NumPy y SciPy para operaciones numéricas y científicas, y Matplotlib para la visualización de datos. Estas bibliotecas proporcionan una herramienta clave para el análisis y la visualización de las imágenes astronómicas, lo que nos ha permitido llevar a cabo las investigaciones de manera eficiente y precisa (Pérez, 2019).

En este contexto, y reconociendo la necesidad de herramientas computacionales para entender, estudiar y analizar eficientemente las imágenes astronómicas, el objetivo de este trabajo es desarrollar diversos códigos eficientes para la manipulación de un importante volumen de imágenes astronómicas utilizando Python, como lenguaje de programación. Estas herramientas se han diseñado para el procesamiento, visualización y el análisis de las imágenes astronómicas obtenidas del telescopio Reflector de 1 m de diámetro, ubicado en el Observatorio Astronómico Nacional (OAN) y las cuales pueden ser empleadas con cualquier imagen astronómica en formato FITS (*Flexible Image Transport System*).

Anteriormente, el intercambio de imágenes astronómicas entre institutos y observatorios involucraba una diversidad de equipos informáticos y métodos para la grabación de datos digitales. Cada instituto desarrollaba su propio software para manejar sus formatos de imagen específicos, lo que requería la creación de programas adicionales para convertir entre el formato estándar y el local. En 1977, durante la "Workshop on Standards for Image Pattern Recognition", después de revisar múltiples formatos existentes, se concluyó que ninguno era lo suficientemente simple, flexible y universal. Como respuesta a esta necesidad, se acordó diseñar un nuevo formato (Wells et al., 1981). Así nació el formato FITS. FITS es el formato de archivo más utilizado actualmente en el mundo de la astronomía, y fue desarrollado especialmente para almacenar una gran cantidad de informaciones astronómicas en un solo archivo. La ventaja fundamental es que las cabeceras (*headers* o metadatos) se encuentran en ASCII, donde ponemos información importante sobre la imagen astronómica observada, como por ejemplo: fecha, hora de observación, ubicación del telescopio, características técnicas del instrumento, campo de la observación, etc.

Las imágenes astronómicas son representaciones visuales de objetos celestes y fenómenos cósmicos capturados por instrumentos de observación astronómica (Astronomical Society of the Pacific, 2013). Estas imágenes se componen de matrices bidimensionales de valores, donde cada píxel corresponde a la cantidad de luz que incidió en ese punto durante el tiempo de exposición. La información capturada puede variar según la longitud de onda de la radiación electromagnética detectada, y el tipo de instrumento utilizado (Astronomical Society of the Pacific, 2024). La información dentro de la imagen viene en forma de metadatos, por lo que, deben guardarse en formato de archivo FITS para mantener toda la información.

En el marco de este trabajo, la observaciones empleadas fueron tomadas del Telescopio Reflector de 1 m de diámetro, ubicado en el Observatorio Astronómico Nacional en el sector de Llano del Hato de la ciudad de Mérida, Venezuela (8 °47'11"N, 70 °52'18,8"O, 3600 $msnm$). Para estas observaciones, se le acopló una cámara CCD, modelo FLI PL4240 256 al telescopio en su configuración F/5,2 en donde la luz recorre una distancia focal de 5.150 mm, con una anchura del pixel de 13,5 μm, cubriendo un campo de visión de 19 arcmin x 19 arcmin de arco (Downes, 2006).

Desarrollo

Visualización de una imagen astronómica en formato FITS con Python

Antes de comenzar a desarrollar los diversos códigos para la manipulación de imágenes astronómicas, es importante destacar que, se quiere trabajar en un entorno virtual para mayor interacción y facilidad a la hora de almacenar datos, de manera que se establecen algunas premisas:

- El código se desarrolla en un notebook en Google Colab
- Las imágenes están almacenadas y clasificadas en carpetas en Google Drive
- El código completo esta en el siguiente enlace: jupyter notebook en google colab

Como se describió anteriormente las imágenes astronómicas son representaciones visuales de objetos celestes. En el proceso de manipulación de dichas imágenes, primero vamos aprender a desplegarlas y Visualizarlas. Dentro del *notebook* y usando Python como lenguaje. El primer paso consiste en importar las bibliotecas necesarias, en este caso 'Astropy' que nos permite abrir y trabajar con los datos de archivos FITSs y 'matplotlib.pyplot' para visualizar los datos en forma de imagen, para ello se desarrollo el código de la Tabla 1.

Tabla 1: Código para visualizar una imagen astronómica en formato FITS

```
# Función para procesar y mostrar una imagen FITS
def mostrar_imagen(path):
  with fits.open(path) as hdul:
    data = hdul[0].data
    header = hdul[0].header
    # Configura la escala logarítmica ZScale
    zscale = ZScaleInterval()
    zmin, zmax = zscale.get_limits(data)
    # Visualización con escala logarítmica
    plt.figure(figsize=(8, 8))
    plt.imshow(data, cmap='gray', norm=LogNorm(vmin=zmin, vmax=zmax), origin='lower')
    plt.colorbar(label='Intensidad (log)')
    plt.title(f"Imagen: {header.get('OBJECT')}")
    plt.show()
for path in calibration_image_paths:
  mostrar_imagen(path)
# Muestra la imagen astronómica
mostrar_imagen(astronomical_image_path)
```

Fuente: Elaboración propia (2024).

Este código toma la ruta de un archivo FITSs como argumento desde google drive, abre el archivo utilizando 'fits.open(path)', lo que devuelve un objeto HDUList que contiene los datos de la imagen y el encabezado asociado. Extrae los datos de la imagen y el encabezado del primer

HDU (unidad de datos principal) usando 'hdul[0].data' y hdul[0].header' respectivamente. Utiliza ZScaleInterval de la biblioteca astropy.visualization para calcular los límites mínimos y máximos de la escala de la imagen. Esto asegura que la visualización tenga un buen contraste y no pierda detalles importantes. Se normaliza los datos utilizando 'LogNorm' para una escala logarítmica de los valores de intensidad. Muestra una barra de color con la intensidad en escala logarítmica utilizando 'plt.colorbar()'.

Para la visualización de la imagen de la Figura 2, se aplica una escala logarítmica para mejorar la variación de intensidad de la luz en la imagen, en la barra de intensidad que esta a la derecha podemos observar la variación de intensidad, donde los valores más bajos representan niveles de intensidad más oscuros y los valores más altos representan niveles de intensidad más brillantes.

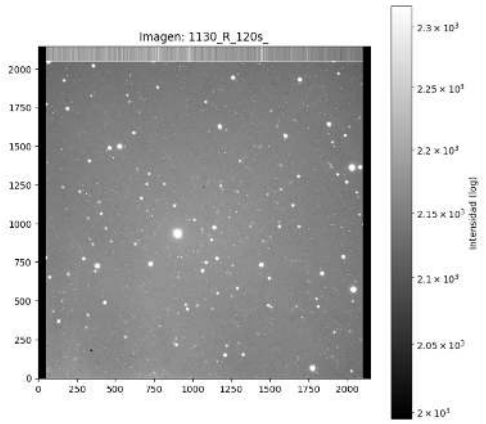

Figura 2: Imagen astronómica obtenida para estudiar el asteroide 1130, tomada por el Telescopio Reflector a 120 segundos de exposición en el filtro R, resultante de la compilación del código de la Tabla 1.

Fuente: OAN (2024).

Observación de la cabecera de imagen astronómica en formato FITS con Python

Como se explicó anteriormente, estas imágenes astronómicas al estar en formato FITS tienen una cabecera que aporta información importante para los astrónomos, de manera que se desarrolló un código en Pyhton en donde se puede visualizar la cabecera de cada imagen astronómica, además, los astrónomos no solo desean visualizarlo, sino también necesitan tener acceso a ellos, para modificarlos, usarlos o incorporar nuevas variables en la cabecera.

Si se desea extraer alguna información en específico de la cabecera, como por ejemplo: telescopio empleado, fecha de observación, hora local de la observación, tiempo de exposición, tamaño del detector CCD, tamaño de las imágenes, equinoccio, filtro, RA (ascensión recta) y DEC (declinación) del centro de la imagen, etc., luego de conocer como están escritas en

la cabecera, la podemos imprimir por pantalla, asignarle un nombre para que el investigador pueda emplearlas en diversos cálculos durante su investigación. Las informaciones de la cabecera de una imagen son de gran importancia para los astrónomos, ya que gracias a ella, podrán referenciar sus observaciones con los catálogos estelares, de manera que el código para imprimir los valores de ciertos parámetros en la cabecera se puede observar en la Tabla 2.

Tabla 2: Código para imprimir en pantalla algunas variables de la cabecera de una imagen astronómica

```
# Lectura de la cabecera de una imagen FITS
for path in calibration_image_paths:
  with fits.open(path) as hdul:
    header = hdul[0].header
  print(fÇabecera de la imagen de calibración: {header.get('OBJECT')}")
  print(header)
  print("\n- "*50 + "\n")
  print(f"Telescopio empleado: {header.get('TELESCOP', 'No disponible')}")
  print(f"Fecha de observación: {header.get('DATE-OBS', 'No disponible')}")
  print(f"Tiempo de exposición: {header.get('EXPTIME', 'No disponible')} segundos")
  print(f"Tamaño del Detector CCD: {header.get('DETSIZE ', 'No disponible')}")
  print(f"Tamaño de las imágenes: {header.get('BITPIX', 'No disponible')} números de bits por píxel")
  print(f.ªquinoccio: {header.get('EQUINOX', 'No disponible')}")
  print(f"Filtro: {header.get('FILTER', 'No disponible')}")
  print(fRA del centro de la imágen: {header.get('OBJCTRA', 'No disponible')}")
  print(f"DEC del centro de la imágen: {header.get('OBJCTDEC', 'No disponible')}")
  print(f"Hora local de la observación: {header.get('LTOBS', 'No disponible')}")
  print("\n- "*50 + "\n")
  print('\n')
```

Fuente: Elaboración propia (2024).

Al ejecutar el código de la Tabla 2 obtenemos los siguientes resultados: Telescopio empleado: Reflector 1m f20.9, fecha de observación: 2024-01-21, tiempo de exposición: 120,06 segundos, tamaño del detector CCD: 1:2048,1:2048, tamaño de las imágenes: 16 números de bits por píxel, equinoccio: 2000,0, filtro: 3- Bessell R, RA del centro de la imagen: 08 34 53,546, DEC del centro de la imagen: +15 04 33,784 y hora local de la observación: 01:06:25,127.

Estos son algunos de los datos más elementales almacenados en la cabecera de las imágenes astronómicas, aunque pueden variar de acuerdo al investigador y al estudio a realizar. Por ejemplo, la fecha de observación '2024-01-21' indica que la observación se realizó el 21 de enero de 2024, la hora local de la observación, indica que la imagen fue tomada a 1 hora 6 minutos 25,127 segundos, el tipo de filtro empleado fue R, y están las coordenadas Ra y Dec del centro de la imagen.

Cálculo del TUC y su incorporación en la cabecera de las imágenes astronómicas con Python

El Tiempo Universal Coordinado (TUC) es el tiempo difundido por las señales horarias con una precisión de ±0s,00002 y es tomado como base para definir la hora oficial de cada país o zona. El Tiempo universal coordinado nos da un tiempo medio común, pero referido al meridiano de Greenwich. Un sistema de estándares para todo el globo terrestre y está basado en las zonas o husos horarios, basados en incrementos de 15°(1 hora) de longitud, aunque, en la práctica, son los gobiernos de los distintos países quienes decretan el llamado tiempo de zona (TZ, ZT), tomando generalmente como base un número entero de horas que represente la longitud media λ_m de una zona o país determinado, de modo que $TZ = TUC + \lambda_m$ (Abad et al., 2002).

El TUC es fundamental para la astrometría de asteroides y cometas, puesto que es necesario para reportar las posiciones (Ra y Dec) de los cuerpos en el tiempo exacto, en el que fueron capturado por el CCD del telescopio. Estos reportes se envían al Minor Planer Center (MPC) y con ello se contribuye con la observación de las órbitas de los cuerpos menores (Minor Planet Center Staff, 2024). De manera que para nuestra investigación es fundamental calcular el TUC de cada imagen astronómica observada en el OAN e incorporarla en su respectiva cabecera.

De acuerdo a la ubicación geográfica del OAN, el TUC de cada imagen astronómica debe ser calculada de la siguiente manera: $TUC = LTOBS + 4 + (EXPTIME)/2$ de manera que, el TUC de cada imagen astronómica tomada del Reflector del OAN, se calculará como la suma del tiempo local de observación (registrado en la cabecera de cada imagen como LTOBS), más 4 horas (por la ubicación geográfica de Venezuela de acuerdo al Meridiano de Greenwich), más la mitad del tiempo de exposición de cada imagen (registrado en las cabecera como EXPTIME), de esta manera tenemos el tiempo universal coordinado, en el cual el CCD del telescopio estaba recibiendo la luz de los cuerpos celestes de un campo determinado.

Si las condiciones atmosféricas son las idóneas en el OAN, en una noche de observación con el telescopio Reflector, y un tiempo de exposición de 120 s, se obtendrían aproximadamente 276 imágenes astronómicas, pero si el proyecto amerita reducir el tiempo de exposición a 30 s, se obtendrían 900 imágenes astronómicas por noche, siendo ya un volumen significativo de datos a trabajar, lo que nos obliga buscar alternativas para automatizar todos los procesos de corrección y de análisis, de manera que en este trabajo se propone el cálculo del TUC en cada imagen astronómica e incorporarlo a su cabecera, a través del código de la Tabla 3.

Tabla 3: Código para calcular el TUC de cada imagen astronómica e incorporarla en su cabecera

```
#iterar sobre las carpetas especificadas en la lista
for directory in asteroides: # Nombre del archivo de texto para el asteroide actual
txt_filename = f'{directory.split(/")[-1]}_resultados.txt'
nombre_asteroide = directory.split(/")[-1]
  print(f'\nNOMBRE DEL ASTEROIDE: {nombre_asteroide}")
  print(————————————————\n")
  # Abrir el archivo de texto en modo escritura
  with open(txt_filename, 'w') as txt_file:
    # Iterar sobre los archivos FITS en el directorio actual
    for filename in os.listdir(directory):
      if filename.endswith('.fit'):
        fits_file_path = os.path.join(directory, filename)
        try:
          with fits.open(fits_file_path) as hdul:
            header = hdul[0].header # Obtener el header
            # Extraer la fecha de observación del header
            fecha_str = header['DATE-OBS'] # Fecha en formato string
            # Extraer el tiempo de exposición del header
            exposicion_segundos = int(header['EXPTIME']) # Tiempo de exposición en s
            # Extraer la hora local de observación del header
            tiempo_str = header['LT-OBS'] # Hora local en formato string (HH:MM:SS)
            # Convertir cada parte a su formato correspondiente
            fecha = definir_fecha(fecha_str)
            tiempo = definir_tiempo(tiempo_str)
            exposicion = definir_exposicion(exposicion_segundos)
            # Convertir el tiempo actual a un objeto datetime completo
            fecha_tiempo = datetime.combine(fecha, tiempo)
            # Sumar la exposición dividida entre 2 al tiempo
            fecha_tiempo += exposicion / 2
            # Sumar 4 horas al tiempo resultante
            fecha_tiempo += timedelta(hours=4)
            # Verificar si la hora inicial es menor o igual a 12
            if tiempo.hour <= 12:
              if fecha_tiempo.hour >= 12:
                # Sumar un día y restar 12 horas
                fecha_tiempo += timedelta(days=1)
                fecha_tiempo -= timedelta(hours=12)
              else:
              if fecha_tiempo.hour &gt;= 24:
                # Sumar un día y restar 24 horas
                fecha_tiempo += timedelta(days=1)
                fecha_tiempo -= timedelta(hours=24)
              # Extraer la nueva fecha y el nuevo tiempo por separado
              nueva_fecha = fecha_tiempo.strftime(" %Y-%m-%dT%H:%M:%S")    # ISO 8601
              # Actualizar el valor DATE-OBS en el header con la nueva fecha
              hdul[0].header['DATE-OBS'] = nueva_fecha
              # Guardar el nuevo archivo FITS con el header actualizado
              new_fits_filename = f'{filename.split(".")[0]}_UTC.fit'
              hdul.writeto(new_fits_filename, overwrite=True)
```

Fuente: Elaboración propia (2024).

Calculado e incorporado el TUC en la cabecera de cada imagen, se procede a sus respectivas

calibraciones, para retirarle a las imágenes astronómicas el ruido electrónico incorporado por el CCD, la corriente oscura del sensor y el viñeteo.

Cálculo de las imágenes de calibración con Python

Según el articulo (AstroBasics, 2024) las imágenes de calibración son imágenes que se utilizan para corregir o eliminar los efectos no deseados introducidos por la cámara y el instrumento, durante la adquisición de imágenes científicas. Las imágenes de calibración no muestran objetos astronómicos reales, sino muestran información sobre el rendimiento y las características del equipo de observación. Hay tres tipos principales de imágenes de calibración: bias, dark y flat. Cada uno de ellos corrige un aspecto específico en la imagen de luz.

Es importante destacar que para aplicar efectivamente estas imágenes de calibración durante el proceso de corrección de las imágenes científicas, es necesario crear "imágenes maestras". Las imágenes maestras son el resultado de promediar una serie de imágenes de calibración del mismo tipo, proporcionando así una representación más precisa de las características de las imágenes de calibración.

Para crear las imágenes maestras de calibración a través de Python, se emplea 'np.average()' de la biblioteca 'Numphy', para así calcular el promedio de ellas (Lutz, 2009). El código que se propone para calcular el bias maestro, dark maestro y flat maestro se visualiza en la Tabla 4.

Tabla 4: Código para calcular los bias, dark y flat maestros.

```
# Calcular los masters
master_bias = np.average(bias_data, axis=0)
master_dark = np.average(dark_data, axis=0)
master_flat = np.average(flat_data, axis=0)
```

Fuente: Elaboración propia (2024).

Las imágenes bias son tomadas con el tiempo de exposición más corto posible (suele estar en el rango de microsegundos) y con la tapa del objetivo en el telescopio puesta. Se utilizan para medir el ruido electrónico que genera el sensor al leer los valores de cada píxel. Este ruido es constante y estará presente en todas las demás imágenes de calibración y en las observaciones astronómicas, puesto que es el ruido electrónico producido por el CCD (AstroBasics, 2024). El bias maestro que calculamos con el código de la Tabla 4, es el resultado de promediar diez imágenes bias y se puede observar en la Figura 3 (izquierda).

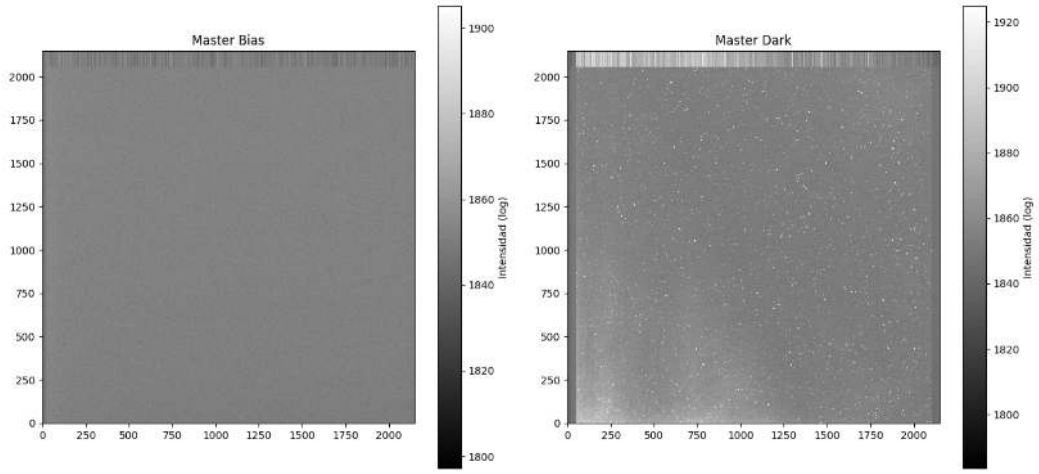

Figura 3: (Izquierda) imagen bias maestro, (derecha) imagen dark maestro calculadas a partir de 10 imágenes bias, dark respectivamente, obtenidas la noche de observación del asteroide 1130, resultante del código aplicado de la Tabla 4.
Fuente: OAN (2024).

Las imágenes darks son tomada con el obturador de la cámara cerrado, y con la misma configuración de la cámara para la adquisición de las imágenes astronómicas. Se utilizan para medir el ruido térmico que produce el sensor al calentarse durante una exposición larga. Al restar la imagen dark de las imágenes astronómicas, se elimina la señal causada por la corriente oscura del sensor, reduciendo el ruido del fondo y mejorando la calidad de la imagen final (AstroBasics, 2024). El dark maestro que calculamos en el código de la Tabla 3, es el resultado de promediar diez imágenes darks a 120 s, de exposición y se puede observar en la Figura 4 (derecha).

Las imágenes flats son observaciones capturadas en frente de una pantalla blanca iluminada uniformemente, o éstas pueden hacerse de cielo, posicionando el telescopio a una zona del cielo nublada uniformemente. Éstas imágenes corrigen la iluminación no uniforme y el viñeteo existente causado por el instrumento. Las imágenes flats ayudan a identificar y a corregir las diferencias de sensibilidad de los píxeles, producto del sensor de la cámara, garantizando que las imágenes calibradas finales tengan una iluminación uniforme en todos los píxeles (AstroBasics, 2024). El flat maestro se calcula a través del código que se propone en la Tabla 4, y es el resultado de promediar diez imágenes flats en el filtro R y el cual se puede observar en la Figura 4.

En la Figura 4 se puede observar manchas circulares producto de la humedad y el polvo recibido en la parte óptica del instrumento; también se puede observar que la imagen es mas clara en el centro de la imagen y se oscurecen hacia los bordes, producto del viñeteo. El viñeteo es un fenómeno óptico que causa una disminución en el brillo de la imagen hacia los bordes.

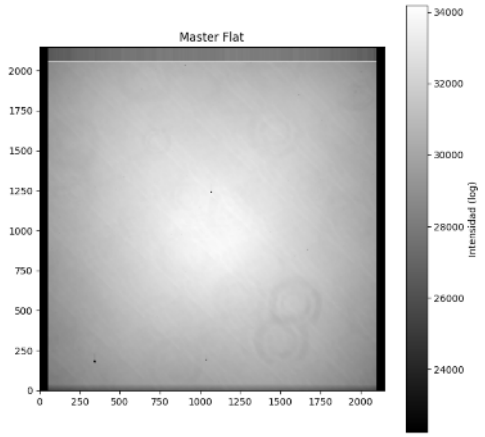

Figura 4: Imagen Flat maestro es calculada a partir de 10 imágenes flats obtenidas la noche de observación del asteroide 1130, resultante de la compilación del código de la Tabla 4.
Fuente: OAN (2024).

Aplicación de las imágenes de calibración a las imágenes astronómicas con Python

Calculadas ya las imágenes maestras de calibración: bias maestro, dark maestro y flat maestro, procedemos a aplicarlas a cada una de las imágenes astronómicas. La Figura 5, representa un esquema visual de como se aplicarían matemáticamente las imágenes de calibración a la imagen astronómica.

El primer paso consiste en promediar las imágenes bias, darks y flats para generar las respectivas imágenes maestras. Luego, a la imagen astronómica le restamos el dark maestro, con el fin de eliminar la corriente oscura y el ruido electrónico generado por el CCD (Craig, 2023). Seguidamente, al resultado de esa resta, se le divide por la diferencia normalizada entre el flat maestro y el bias maestro, matemáticamente hablando se aplica la fórmula 1.

$$\text{Imagen astronómica corregida} = \frac{\text{imagen astronómica} - \text{dark maestro}}{\textit{normalizado}(\text{flat maestro} - \text{bias maestro})} \tag{1}$$

Ahora bien, el código que se propone en Python para corregir cada imagen astronómica del ruido electrónico, la corriente oscura y el viñeteo, entre otros efectos, se puede observar en la Tabla 5.

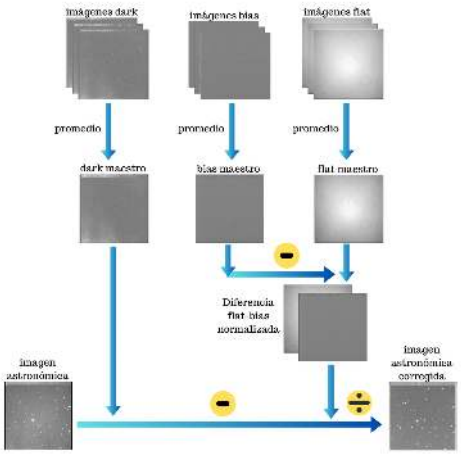

Figura 5: Esquema matemático del proceso de calibración de una imagen astronómica con bias, dark y flats.

Fuente: Elaboración propia (2024).

Tabla 5: Código para corregir las imágenes astronómicas a través de los bias, darks y flats maestros

```
#CALIBRACION DE LA IMAGEN ----------------------------------
folder_path = '/content/drive/MyDrive/Practicas_CIDA/Practica2_fits'
fits_files = [file for file in os.listdir(folder_path) if file.startswith('1130_R_120s_') and file.endswith('.fit')]
for fits_file in fits_files:
    # Ruta completa del archivo FITS
    astronomical_image_path = os.path.join(folder_path, fits_file)
    # Cargar los datos de los archivos FITS
    with fits.open(astronomical_image_path) as hdul:
    astronomical_image_data = hdul[0].data
    astronomical_image_header = hdul[0].header
    # Calcular la diferencia entre el master flat y el master bias
    difference = master_flat - master_bias
    # Calcular el valor mínimo y máximo de la diferencia
    min_difference = np.min(difference)
    max_difference = np.max(difference)
    # Normalizar la diferencia utilizando el valor mínimo y máximo
    normalized_difference = (difference - min_difference) / (max_difference - min_difference)
   epsilon = 1e-6
    # Aplicar la corrección de la imagen astronómica
    corrected_image_data = (astronomical_image_data - master_dark) / (normalized_difference + epsilon)
    # Guardar la imagen corregida como un nuevo archivo FITS manteniendo el header original
    output_path = os.path.join(folder_path, f'{fits_file.split(".")[0]}_calibrada.fits')
    fits.writeto(output_path, corrected_image_data.astype(np.float32),     header=astronomical_image_header, overwrite=True)
    # Mostrar la imagen corregida
    mostrar_imagen(corrected_image_data, f"{fits_file.split('.')[0]}_calibrada"))
```

Fuente: Elaboración propia (2024).

La imagen resultante de la aplicación del código de la Tabla 5 a la imagen astronómica '1130_R_120s01212024020627001.fit' puede verse en la Figura 6. Visualmente se ve mayor uniformidad en la imagen, menor ruido de fondo y mayor definición en los círculos blancos, los cuales representan las estrellas, los asteroides, cometas, etc., que en comparación a la Figura 2 de la imagen sin corregir.

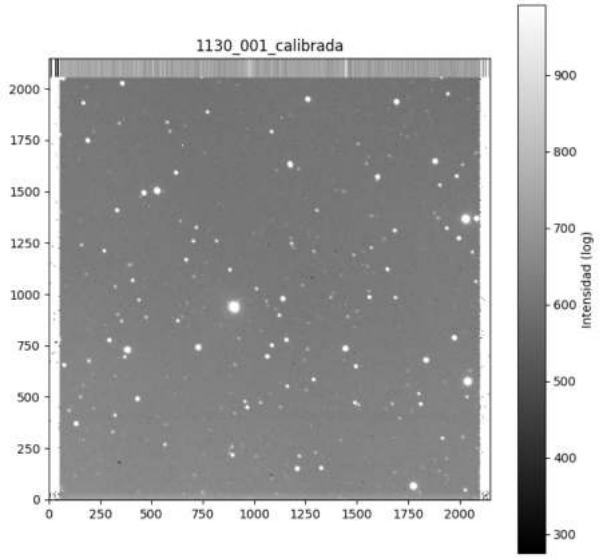

Figura 6: Imagen astronómica luego de ser calibradas con los *bias*, *dark* y *flats* maestros, resultante de la compilación del código de la Tabla 5.
Fuente: Elaboración propia (2024).

No es suficiente mapear una imagen astronómica para poder medir la calidad de la observación, o determinar las características de los cuerpos celestes que allí intervienen, de manera que, nosotros como investigadores, estudiantes y asistentes científicos, nos vemos en la necesidad natural de buscar o desarrollar herramientas que nos permitan mejor cuantificar las observaciones y los resultados, de allí surge la necesidad de estudiar los píxeles individualmente.

Cálculo y visualización de los valores de un píxel en las imágenes imágenes astronómicas con Python

El valor de un píxel es también conocido como intensidad luminosa o brillo en ese punto. El valor de un píxel en una imagen astronómica representa la cantidad de luz o radiación electromagnética que incide en ese punto específico (Astronomical Society of the Pacific, 2024). Esta medida se visualiza comúnmente como una escala de grises en una imagen en blanco y negro, donde los píxeles más brillantes tienen valores más altos y los más oscuros, valores más bajos. La determinación precisa de este valor se logra mediante la calibración de las cuentas del píxel, teniendo en cuenta factores como la ganancia y otras correcciones necesarias en la

imagen para representar de manera precisa la intensidad de la luz.

Las cuentas son los valores numéricos almacenados en una imagen astronómica en bruto, obtenidos directamente del detector del telescopio y son expresadas en ADU (Unidad Digital Analógica). Estos valores representan la cantidad de luz o radiación electromagnética detectada por cada píxel en la imagen (Neira, 2024).

Continuando con las herramientas, ahora se incorporan nuevas funciones basadas en la definición de las coordenadas x,y de los píxeles a los que queremos conocer sus cuentas. Se define una función llamada 'mostrar_imagen(path)' para procesar y mostrar la imagen en general y los valores del píxel. El código que se propone en Python para conocer las cuentas en un píxel dado y visualizarlo en la imagen astronómica se puede observar en la Tabla 6, y como resultado de éste código, se obtienen la Figura 7, en la cual, se pueden diferenciar los valores de los pixeles en rojo y azul.

Tabla 6: Código para conocer las cuentas en un píxel dado y visualizarlo en la imagen astronómica

```
# Coordenadas de los píxeles para mostrar el valor (x,y)
pixel1 = (901, 936) # Píxel 1
pixel2 = (300, 300) # Píxel 2
# Función para procesar y mostrar una imagen FITS
def mostrar_imagen(path):
  with fits.open(path) as hdul:
    data = hdul[0].data
    header = hdul[0].header
    # Configura la escala logarítmica ZScale
    zscale = ZScaleInterval()
    zmin, zmax = zscale.get_limits(data)
    # Muestra los valores de los píxeles
    valor_pixel1 = data[pixel1[1], pixel1[0]]                    # Coordenadas invertidas (y, x)
    valor_pixel2 = data[pixel2[1], pixel2[0]]                    # Coordenadas invertidas (y, x)
    plt.text(pixel1[0], pixel1[1], f"{valor_pixel1:.2f}", color='red')          # píxel 1
    plt.text(pixel2[0], pixel2[1], f"{valor_pixel2:.2f}", color='blue')         # píxel 2
    # Ajusta los límites del eje
    plt.xlim(0, data.shape[1])
    plt.ylim(0, data.shape[0])
    plt.title(fÏmagen: {header.get('OBJECT')}")
    plt.show()
```

Fuente: Elaboración propia (2024).

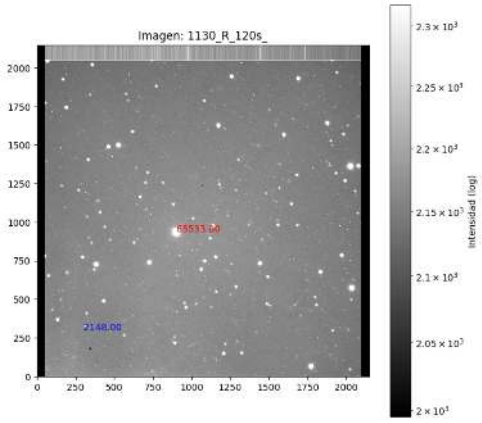

Figura 7: Imagen astronómica resultante de la compilación del código de la Tabla 6, en donde se muestran las cuentas en dos píxeles diferentes en la imagen. En color rojo, las cuentas en el pixel (901, 936), y en color azul, las cuentas en el píxel (300, 300).
Fuente: Elaboración propia (2024).

En la Figura 7, se observa el valor de 65.535 en color rojo, el cual corresponde al número de cuentas que tiene el píxel (901, 936). Ese píxel está prácticamente en el centro de esa estrella, y ese valor en las cuentas indica que este píxel llegó a saturación, debido a que es una estrella bastante brillante y el CCD en ese píxel recibió mucha luz durante la exposición. Por otro lado, las cuentas de 2.148 (en color azul), le corresponde a la estrella débil ubicada en el píxel (300, 300). Se observa que la luz recibida en ese píxel es mayor que el fondo del cielo, pero es menos representativa que la que recibió el píxel (901, 936). En decir, las cuentas más altas indican una mayor recogida de luz (cuerpos celestes brillantes, valores de magnitudes bajas), mientras que los valores más bajos en las cuentas indican una menor recogida de luz en los píxeles de la imagen (objetos débiles, valores de magnitudes altas). Este tipo de herramienta permite estudiar y valorar cada píxel en la imagen, permitiéndole al investigador o al asistente hacer ajustes en los protocolos de observación, bajando o subiendo el tiempo de exposición en la observación, ya sea subiendo el tiempo de exposición para observar mejor a los objetos débiles, o bajando el tiempo de exposición para no saturar al objeto celeste brillante en estudio.

Cálculo y visualización de los perfiles de intensidad en x, y en y de las imágenes imágenes astronómicas con Python

Los perfiles son representaciones gráficas de la variación de la intensidad de la luz en una imagen a lo largo de una línea específica, ya sea en dirección horizontal (eje x), o vertical (eje y). Estos perfiles se calculan y se aplican a las imágenes astronómicas, con el propósito de analizar, entender y visualizar la distribución de la luz de las observaciones, y así extraer información sobre las características, tales como, la forma, el brillo, etc., de una estrella, un asteroide, un cometa, un satélite, de estructuras astronómicas, del fondo del cielo, etc., (Astronomical Society of the Pacific, 2024). Cabe destacar que dependiendo de la orientación,

un perfil de intensidad en un eje muestra la distribución general de los píxeles de la imagen, mientras que en el otro eje, muestra los valores de la intensidad.

Características generales para la interpretación del perfil de intensidad de una imagen astronómica:

1. **Localización de objetos astronómicos:** Las estrellas, satélites, cometas, asteroides y otros objetos astronómicos suelen aparecer como picos agudos en el perfil de intensidad. Estos picos representan regiones donde la intensidad de la luz es significativamente mayor que su entorno, lo que indica la presencia de objetos luminosos.

2. **Detección de gradientes de fondo:** El perfil de intensidad puede revelar la presencia de gradientes de fondo en la imagen. Un gradiente de fondo se manifiesta como una variación gradual en la intensidad de la luz a lo largo de una fila o columna. Estos gradientes pueden deberse a la presencia de la Luna, contaminación lumínica, el brillo del cielo nocturno, o la presencia de nebulosas o galaxias difusas.

3. **Identificación de artefactos o defectos:** Los picos anómalos o las discontinuidades inusuales en el perfil de intensidad pueden ser indicativos de artefactos de imagen, como píxeles calientes, píxeles muertos, o reflejos de luz no deseados. Estos artefactos pueden distorsionar la imagen y afectar su calidad.

En el marco de este trabajo, para calcular y graficar el perfil de intensidad de una imagen astronómica a través de Python, se utilizan las bibliotecas 'numpy' y 'matplotlib.pyplot' (Lutz, 2009). El código propuesto se encarga de calcular el perfil de intensidad en un píxel específico de la imagen FITS. Primero se procede a abrir la imagen fit, se accede a sus valores de intensidad, luego, conociendo el píxel a estudiar, se obtiene los valores de las cuentas lo largo del eje horizontal (eje x) y del eje vertical (eje y) del píxel indicado.

Estos perfiles se calculan tomando los valores de intensidad a lo largo de la columna y de la fila especificadas por las coordenadas del píxel que se desea estudiar. Finalmente, la función devuelve los perfiles de intensidad vertical y horizontal, junto con la intensidad en el píxel de interés. El código se presenta en la Tabla 7 aplicado al píxel (901,936) de la imagen Astronómica corregida. Los perfiles son mostrados en la Figura 8. En ambos perfiles se ve un importante pico de aproximadamente 65.535 cuentas, lo que es de esperarse, puesto que los perfiles son calculados en el centro de una estrella. Por otra parte, se observa a largo del eje x, y del eje y, que sobresalen otras señales mas tenues, las cuales pueden representar estrellas, asteroides, cometas u otro cuerpo celeste, pero de menor brillo. Adicionalmente, el perfil de intensidad me permite determinar que el fondo del cielo está alrededor de 1800 cuentas. Este tipo de herramientas se puede aplicar a cualquier píxel y a cualquier imagen astronómica o de calibración, con el objetivo de caracterizarlas, cuantificarlas y evaluar su calidad.

Tabla 7: Código de la función para calcular los perfiles x, y en las imágenes astronómicas

```
# Función para calcular el perfil de intensidad en un píxel
def calcular_perfil_intensidad(imagen, x, y):
  with fits.open(imagen) as hdul:
    data = hdul[0].data
    intensidad = data[y, x] # Coordenadas invertidas (y, x)
    perfil_vertical = data[:, x] # Perfil vertical
    perfil_horizontal = data[y, :] # Perfil horizontal
    return perfil_vertical, perfil_horizontal, intensidad
# Mostrar los resultados en gráficas separadas
    plt.figure(figsize=(20, 6))
    # Gráfico para el perfil de intensidad vertical
    plt.subplot(1, 2, 1)
    plt.semilogy(perfiles_x, color='black', linewidth=0.5) # Ajusta el grosor de la línea
    plt.title(f'Perfil de Intensidad Vertical - {imagen_name} - {pixel_name}')
    plt.xlabel('Posición en el Eje Y')
    plt.ylabel('Intensidad (escala logarítmica)')
    plt.text(0, np.max(perfiles_x), f'Value: {intensidad}', color='blue')
    # Gráfico para el perfil de intensidad horizontal
    plt.subplot(1, 2, 2)
    plt.semilogy(perfiles_y, color='black', linewidth=0.5) # Ajusta el grosor de la línea
    plt.title(f'Perfil de Intensidad Horizontal - {imagen_name} - {pixel_name}')
    plt.xlabel('Posición en el Eje X')
    plt.ylabel('Intensidad (escala logarítmica)')
# Rutas de las imágenes FITS astronómicas
astronomical_image_paths = [('/content/drive/MyDrive/Practicas_CIDA/Resultados/1130_001_calibrada_F1.fits', 'Imagen
Astronómica Corregida') ]
# Coordenadas del píxel para evaluar
pixel = (901, 936)
# Mostrar los perfiles de intensidad para cada imagen astronómica y el píxel especificado
for imagen_path, imagen_name in astronomical_image_paths:
  mostrar_perfil_intensidad(imagen_path, pixel, 'Pixel (901, 936)', imagen_name)
    plt.tight_layout()
    plt.show()
```

Fuente: Elaboración propia (2024).

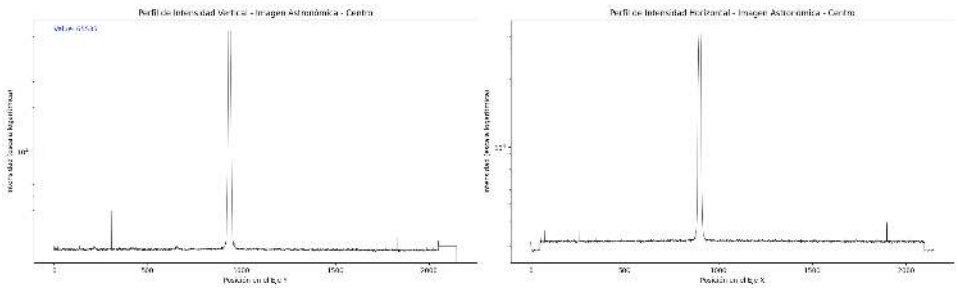

Figura 8: Perfil del píxel (901, 936) de la imagen astronómica calibrada en el eje vertical, y en el eje horizontal, resultante de la aplicación del código propuesto en la Tabla 7.

Fuente: Elaboración propia (2024).

Cálculo estadístico como herramientas para el análisis de las imágenes imágenes astronómicas con Python

Las estadísticas de una imagen astronómica proporcionan información cuantitativa sobre las características clave de la distribución de intensidades de luz en la imagen. Entre ellas se pueden nombrar (Academia Balderix, 2024):

La Media (Promedio): Representa el promedio de todos los valores de intensidad de los píxeles en la imagen. Una media más alta indica que la imagen en general es más brillante, mientras que una media más baja indica que la imagen es más oscura. La media puede ser útil para evaluar el nivel general de iluminación de la imagen.

La Mediana: Es el valor que ocupa la posición central cuando los valores de intensidad se ordenan de menor a mayor. Es menos sensible a los valores extremos que la media y proporciona una medida de la intensidad típica de la imagen. Una mediana más alta indica que la mitad de los píxeles tienen intensidades más altas, mientras que una mediana más baja indica lo contrario.

La Desviación Estándar: Es una medida de la dispersión o variabilidad de los valores de intensidad alrededor de la media. Representa cuánto se alejan los valores individuales de la media. Una desviación estándar más alta indica una mayor variabilidad en la intensidad de la imagen, mientras que una desviación estándar más baja indica una menor variabilidad y una distribución más uniforme de intensidades.

Al interpretar las estadísticas de una imagen astronómica, es importante considerar cómo estas medidas se relacionan entre sí, y cómo pueden influir en la calidad y la interpretación de la imagen en general. Por ejemplo, una alta desviación estándar junto con una mediana baja puede indicar una distribución de intensidades sesgada o una presencia significativa de ruido en la imagen. Comprender estas estadísticas puede ayudar a ajustar el tiempo de exposición en la observación, y mejorar así la calidad de la imagen astronómica.

Para calcular las estadísticas de una imagen astronómica con Python, primero se lee la imagen .fits con 'fits.open()', luego se calcula el promedio, la mediana y la desviación estándar utilizando las funciones 'np.mean()', 'np.median()' y 'np.std()' de Numphy (Lutz, 2009). El código que se propone para ello, se puede visualizar en la Tabla 8.

Tabla 8: Código para el cálculo estadístico como herramientas para el análisis de las imágenes astronómicas

```
# Función para calcular el promedio, mediana y desviación estándar de una imagen FITS
def calcular_estadisticas(imagen_path):
  with fits.open(imagen_path) as hdul:
    data = hdul[0].data
    promedio = np.mean(data)
    mediana = np.median(data)
    desviacion_estandar = np.std(data)
  return promedio, mediana, desviacion_estandar
# Calcular estadísticas para ambas imágenes
mean_sin_corregir, median_sin_corregir, std_dev_sin_corregir = calcular_estadisticas(hdul[0].data)
mean_corregida, median_corregida, std_dev_corregida = calcular_estadisticas(corrected_image_data)
# Mostrar resultados
print()
print(."Estadísticas de la imagen sin corregir:")
print("Media:", mean_sin_corregir)
print("Mediana:", median_sin_corregir)
print("Desviación estándar:", std_dev_sin_corregir)
print("\nEstadísticas de la imagen corregida:")
print("Media:", mean_corregida)
print("Mediana:", median_corregida)
print("Desviación estándar:", std_dev_corregida)
```

Fuente: Elaboración propia (2024).

Luego de aplicar el código de la Tabla 8 a la imagen astronómica, a una imagen bias, dark y a una imagen flat, se presentan los resultados en la Tabla 9.

Tabla 9: Resultados de la aplicación del código de la Tabla 8, en donde se calcula la media, la mediana y la desviación estándar a la imagen astronómica corregida, a una imagen bias, dark y a una imagen flat.

	Media	**Mediana**	**Moda**
Imagen astronómica	2.204,8187	2.213,0	350,950
Imagen bias	1.849,6799	1.851,0	58,4269
Imagen dark	1.855,6399	1.855,0	89,6589
Imagen flat	29.641,2388	30.944,0	6.399,5588

Fuente: Elaboración propia (2024).

Las estadísticas de la imagen astronómica indican que, el promedio de intensidad de luz en la imagen está alrededor de 2.152,48 cuentas, con una mediana de 2.157,0 y una desviación estándar de aproximadamente 433,51. Esto sugiere que la distribución de intensidades de los

píxeles varía alrededor del promedio, mostrando una tendencia central alrededor de la mediana. Es decir, el 34,1 % de los píxeles están a ±433,51 de la mediana, el 47, % de los píxeles están a ±867,02 de la mediana. La desviación estándar señala que la luminosidad en la imagen está muy dispersa, indicando la presencia de ciertos objetos celestes brillantes.

Este tipo de herramientas puede ser aplicado a cualquier imagen astronómica o de calibración, lo que resulta interesante para poder analizar sus cuentas de manera cuantificables. Las estadísticas de las imágenes bias indican que el promedio de intensidades es de aproximadamente 1.849,68, la mediana, es de 1.851 muy cercana al promedio y la desviación estándar es de 58,43. Estos resultados nos permite concluir que la imagen bias no tiene presencia de cuerpos celestes brillantes. El ruido electrónico producido por el CCD es recibido por cada píxel casi de manera uniforme, permitiéndonos concluir que los píxeles están en buen funcionamiento y tienen una eficiencia similar. Los valores resultante del dark, nos permite concluir que la corriente oscura es uniforme en cada píxel de la imagen, sin embargo, en la imagen flat, observamos fuertes valores en la desviación estándar, permitiéndonos concluir que la distribución de la luz no es uniforme en cada píxel, esto puede ser producto del viñeteo, puesto que tenemos valores altos de luminosidad en el centro del campo, y valores bajos en los bordes.

Cálculo de histogramas como herramientas para el análisis de las imágenes astronómicas con Python

El histograma es una herramienta que resulta de gran utilidad cuando se pretende encontrar la tendencia que presentan datos colectados durante algun tiempo, los cuales una vez tratados, se presentan en forma de gráfica, lo que facilita identificar si estos datos siguen algun patrón o se comportan conforme a alguna distribución de probabilidad (Brassard y Ritter, 1994).

En este trabajo, se busca calcular el histograma a una imagen astronómica, con ello se busca conocer la distribución de los píxeles de la imagen en función de la intensidad de luz recibida. El histograma se distribuye mediante tamaños de intervalos definidos llamado “Binning”, donde cada bin representa un rango de valores de intensidad de los píxeles, y la altura del bin indica la frecuencia con la que aparecen los valores de los píxeles en ese rango (Izar y González, 2004).

El eje horizontal (eje x) equivale a la intensidad luminosa, partiendo de cero, o negro absoluto, pasando por tonos medios, hasta alcanzar el blanco (de izquierda a derecha). El eje vertical (eje y) representa el número de píxeles de la imagen que tiene tal luminosidad. De este modo, la superficie de cada una de las barras que forman el histograma refleja la mayor o menor frecuencia de píxeles de la imagen, que tiene cada valor de intensidad en la imagen que tienen cada valor de luminosidad.

Es importante acotar que, un histograma con una distribución sesgada hacia la izquierda indica que la imagen tiene muchos píxeles oscuros, mientras que un histograma con una distribución sesgada hacia la derecha indica que, la imagen tiene muchos píxeles brillantes.

De manera que, para crear histogramas a partir de una imagen astronómica usando Python, primero se carga una imagen FITS en específico, por la ruta "imagen_path", luego, se abre el archivo y se aplanan los datos para obtener una sola dimensión. Seguidamente, se calcula el histograma de los datos utilizando un número de *bins* adecuado (el cual es ajustable) para representar la distribución de intensidades en la imagen. Para evitar errores al calcular el logaritmo de cero, se eliminan los valores del histograma que son iguales a cero. Finalmente, se gráfica el histograma con el método ZScale (escala logarítmica), ver Figura 9. El código que se propone para esta tarea está visible en la Tabla 10.

Tabla 10: Código para el cálculo del histograma en una imagen astronómica

```
import numpy as np
import matplotlib.pyplot as plt
from astropy.io import fits
from google.colab import drive
calibration_image_paths = [
  ('/content/drive/MyDrive/Practicas_CIDA/Practica1_fits/bias01202024164210001.fit', 'Bias'),
  ('/content/drive/MyDrive/Practicas_CIDA/Practica1_fits/Flat_R-1s_01202024193232001.fit', 'Flat'),
  ('/content/drive/MyDrive/Practicas_CIDA/Practica1_fits/Darks_120s_R_01202024165041001.fit', 'Dark')
]
astronomical_image_path = [('/content/drive/MyDrive/Practicas_CIDA/Practica1_fits/ 1130_R_120s_01212024020627001.fit',
'Imagen Astronómica')]
# Función para cargar y mostrar el histograma en escala logarítmica zscale de una imagen
def mostrar_histograma(imagen_path, imagen_name):
  with fits.open(imagen_path) as hdul:
    data = hdul[0].data.flatten()
    # Calcular el histograma
    hist, bins = np.histogram(data, bins=np.linspace(np.min(data), np.max(data), 850))
    nonzero_hist = hist[hist &gt; 0] #Eliminar los valores iguales a cero para evitar log(0)
    # Graficar el histograma en escala logarítmica zscale
    plt.figure(figsize=(18, 6))
    plt.hist(data, bins=bins, color='black', log=True) #Usar log=True para escala logarítmica zscale
    plt.title(f'Histograma en Escala Logarítmica ZScale - {imagen_name}')
    plt.xlabel('Valor del Pixel')
    plt.ylabel('Numero de Pixels')
    plt.grid(True)
    plt.show()
# Mostrar el histograma en escala logarítmica zscale para cada imagen
for imagen_path, imagen_name in [astronomical_image_path] + calibration_image_paths:
  mostrar_histograma(imagen_path, imagen_name)
```

Fuente: Elaboración propia (2024).

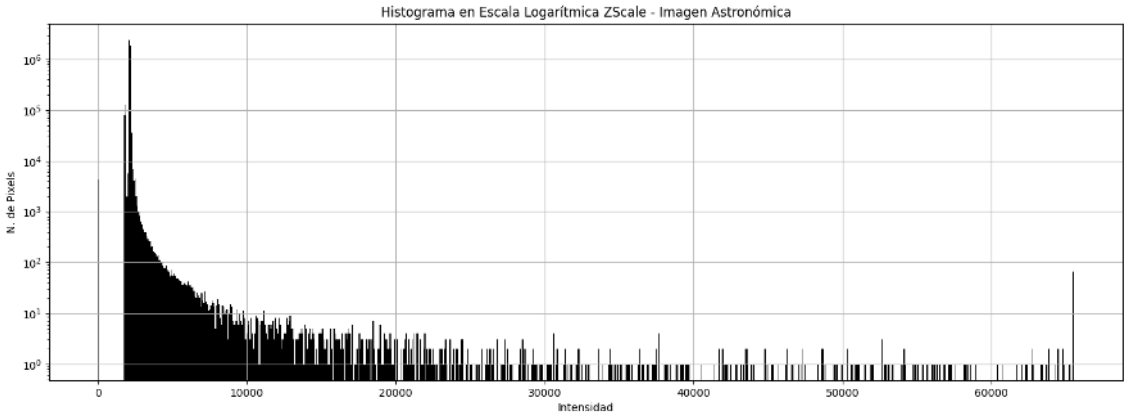

Figura 9: Histograma resultante luego de ejecutar el código de la Tabla 10 para la imagen astronómica. Los valores en el eje x, varían entre 0 y 65.535. Estos valores corresponden a la intensidad de la luz (cuentas) en la imagen astronómica. Los valores del eje y, varían entre 0 y 10^7 (escala logarítmica). Estos valores corresponde al número de píxeles respecto a un determinado rango luminoso.

Fuente: Elaboración propia (2024).

En la Figura 9 se puede apreciar que la mayor acumulación de píxeles se encuentra a la izquierda de la distribución de intensidades, esto quiere decir que, la imagen analizada esta compuesta principalmente por píxeles oscuros, lo cual corresponde en efecto al fondo del cielo, los picos separados a partir de 10.000 son píxeles en donde tenemos presencia de objetos celestes brillantes. La mayor cantidad de píxeles se encuentra en una distribución de intensidades entre 0 a 3.000 cuentas, allí tenemos el fondo del cielo, y algunas estrellas o cuerpos celestes mas débiles. Esta gráfica complementa la información ya aportada anteriormente por el análisis estadístico de la imagen astronómica. A pesar que los rangos de las cuentas de la imagen varían entre 0 y 65.536, la media es de 2.204,818, la mediana es de 2.213,0 y la desviación estándar es de 350,950. Efectivamente el histograma está volcado a la izquierda, siendo los píxeles luminosos poco representativos: Este resultado corresponde con lo observado en la Figura 7.

Es importante remarcar que, este tipo de distribución es la esperada en las observaciones astronómicas, ya que los pocos puntos luminosos representarían a las estrellas, los asteroides, cometas, satélites, etc., en el campo. Por otro lado, si el histograma estuviese volcado a la derecha, entonces tendríamos una saturación en la imagen, por lo que se descartaría la observación para su estudio. En ese caso, si todas las imágenes salen de esa manera, el investigador, el aficionado, o el asistente científico (el cual es el operador de los telescopios) debe evaluar ¿el por qué de ese resultado? Puesto que debió ser producto de un tiempo de exposición muy largo, presencia de la Luna en el campo, o contaminación lumínica. De manera que, se debe ajustar el protocolo de observación, ya sea reduciendo el tiempo de exposición, cambiar las coordenadas, o suspender las observaciones, puesto que ese tipo de imágenes saturadas no son interesantes para la investigación en astronomía.

De manera que, este tipo de herramientas son muy interesantes a emplear en las imágenes FITS, ya sea para cuantificar la distribución de la luz en las observaciones, determinar la calidad de ellas, y/o corregirlas para las diversas investigaciones.

Conclusiones

Los programas desarrollados en este trabajo como herramientas para el procesamiento y la caracterización de imágenes astronómicas, representan una contribución significativa al análisis de datos astronómicos, debido a su capacidad para facilitar y optimizar el tratamiento y la corrección en volumen de datos provenientes del observatorio Astronómico Nacional Llano del Hato. Estas herramientas proporcionan una forma mas eficiente y precisa de trabajar las imágenes FITS en volumen, y no individualmente como se venían trabajando a través de programas comerciales pagos, en donde solo puedes cambiar ciertas variable, o lo que el programa te permite, en cambio, a través de las rutinas realizadas en Python, los investigadores pueden obtener información detallada sobre cualquier píxel en la imagen, y/o controlar toda la información de la cabecera de la imagen. Con estas rutinas, ofrecemos independencia y autonomía al investigador, para que pueda controlar y manipular cada píxel y valor existente, pudiendo estudiar detalladamente y caracterizar cualquier estrella, asteroide, cometa, cúmulo, nebulosas, galaxia, o cualquier otro cuerpo celeste presente en la observación.

Las rutinas propuestas en este trabajo enseñan desde el inicio, a como leer a través de la programación con Python una imagen FITS, muestra como visualizarlos directamente sin la necesidad de depender de otros programas, y progresivamente va ofreciendo rutinas cada vez mas completas para entender la observación, extraer metadatos esenciales del encabezado de la imagen, corregir los efectos no deseados introducidos durante la observación, permitiéndole al investigador controlar la eliminación de ruido electrónico, térmico y la iluminación no uniforme del sensor CCD, contribuyendo a la investigación del universo y a la comprensión de los fenómenos celestes.

Un aspecto clave del proceso, es la calibración de las imágenes astronómicas, la cual consiste en la creación de imágenes como el bias maestro, dark maestro y el flat maestro. Estas imágenes maestras son fundamentales para la corrección precisa de las imágenes científicas. En este trabajo, el investigador puede controlar a partir de los códigos propuestos los detalles del proceso, o ajustar la ecuación de calibración de acuerdo al recurso de datos existentes en la noche de observación.

Las herramientas de análisis propuestas aquí, tal como el de determinar el perfil de intensidad en cualquier píxel de la imagen astronómica, es una herramienta rápida, fácil de usar, y muy importante a conocer, puesto que, muchos programas comerciales pagos no lo ofrecen, ademas a través de estas rutinas se pueden controlar, manipular y operar directamente

cada píxel de manera independiente.

Las herramientas de los histogramas y el estudio estadístico de las imágenes astronómicas nos permiten inmediatamente caracterizar la distribución de la luz en la observación, para así entender e identificar objetos luminosos, conocer el gradiente del fondo de la imagen, determinar los valores de saturación, entre otras características importantes que se pueden identificar. Los códigos que se proponen en este trabajo proporcionan una herramienta gratuita y muy valiosa para la comprensión profunda y detallada de las imágenes astronómicas, contribuyendo significativamente a la investigación astronómica.

Python es un lenguaje versátil que cuenta con una amplia gama de bibliotecas y herramientas específicamente diseñadas para el análisis de datos astronómicos. Estas bibliotecas proporcionan funciones y métodos optimizados facilitando las tareas en astronomía, lo que simplifica y agiliza el proceso de análisis de datos. Python es una comunidad activa y colaborativa, que desarrolla constantemente nuevas herramientas y módulos para el análisis de datos. Esto garantiza que los científicos tengan acceso a las últimas técnicas y métodos de análisis, así como a un amplio soporte y recursos en línea para resolver problemas y compartir conocimiento. En comparación con otros lenguajes como C++, C, Fortran, Iraf, IDL, R, y otros programas comerciales pagos, Python ofrece una sintaxis más clara y legible, facilitando el desarrollo libre y gratuito de códigos, lo que lo convierte actualmente en la opción ideal para el análisis de imágenes astronómicas, ya que no existen limitaciones para operarlas y/o manipularlas.

Referencias

Abad, A., Docobo, J., y Elipe, A. (2002). *Curso de Astronomía. Colección textos docentes*. Prensas Universitarias de Zaragoza.

Academia Balderix. (2024). *Probabilidad y Estadística*. Academia Balderix. https://www.probabilidadyestadistica.net/media-y-desviacion-estandar/

Aldcroft, T., Robitaille, T., Refsdal, B., y Muench, G. (2013). *Python for Astronomers*. Smithsonian Astrophysical Observatory. https://python4astronomers.github.io/index.html

AstroBasics. (2024). *Bias, Flats, Darks, Flats*. AstroBasics. https://astrobasics.de/en/basics/bias-flats-darks-darkflats/

Astronomical Society of the Pacific. (2013). *What is an Astronomical Image?* Astronomical Society of the Pacific. https://astrosociety.org

Astronomical Society of the Pacific. (2024). *Astropy: A Community Python Library for Astronomy*. Astropy v6.0.1. https://astrosociety.org

Brassard, M., y Ritter, D. (1994). *Capital Services Memory Jogger II*. Massachusetts, USA.

Craig, M. (2023). *Astropy Community. CCD Data Reduction Guide*. https://www.astropy.org/ccd-reduction-and-photometry-guide/v/dev/notebooks/01-01-astronomical-CCD-image-components.html

Downes, J. (2006). Caracterización de la cámara FLI. Publicación interna. *RevMexAA*.

Izar, J., y González, J. (2004). *Las 7 Herramientas Bàsicas de la Calidad. Capitulo V El Histograma*. Edition:1. Editorial Universitaria Potosina.

Lutz, M. (2009). *Learning Python. Fourth Edition*. O'reilly.

Minor Planet Center Staff. (2024). *MPEC 2024-G193: Observations and Orbits of Comets and a/ Objects* (Minor Planet Electronic Circular) (April 2024). Minor Planet Center. https://doi.org/10.48377/MPEC/2024-G193

Neira, F. (2024). *Cómo elegir cámara para Astrofotografía*. Mas allá del azul pálido. https://www.fernandoneirapaz.com/como-elegir-camara-para-astrofotografia/

Pérez, J. (2019). *Python en Astrofísica*. Instituto de Astrofísica de Canarias. https://research.iac.es/sieinvens/python-course/astropy.html

Pössel, M. (2019). A Beginner's Guide to Working with Astronomical Data. *The Open Journal Astrophys*, *3*(2). https://doi.org/10.21105/astro.1905.13189arXiv:1905.13189[astro-ph.IM]

Strasbourg Astronomical Data Center. (2020). *Astropy Community. CCD Data Reduction Guide*. https://cds-astro.github.io/tutorials/intro.html

Wells, D., Greisen, E., y Harten, R. (1981). A Flexible Image Transport System. *Astronomy & Astrophysics Supplement Series*, 363.