Cartografía Lunar con R

Francisco J. Rodríguez Aragón
Responsable de Datos en AFI
Miembro de la Agrupación Astronómica de Madrid
https://www.aam.org.es/
Enero 2022

Data Analytics

Índice

1 Introducción

Introducción

Se presenta en esta ocasión el desarrollo de una aplicación que en principio para nada tiene que ver para lo que fue concebido R que es el tratamiento estadístico de datos.

En el desarrollo que se explica a continuación se explica cómo dado un mapa, en este caso el de la superficie lunar, seríamos capaces de señalar accidentes topográficos en él.

Por tanto más que la importancia en sí de la aplicación que se va a explicar al máximo detalle, estaría el promover el desarrollo de estas iniciativas incidiendo por tanto en el carácter totalmente general de R.

La astronomía en general es un área generadora de una gran cantidad de información, gran parte de ella centralizada en instituciones públicas – privada como la NASA, gran parte de la información de las misiones espaciales se publica para uso público

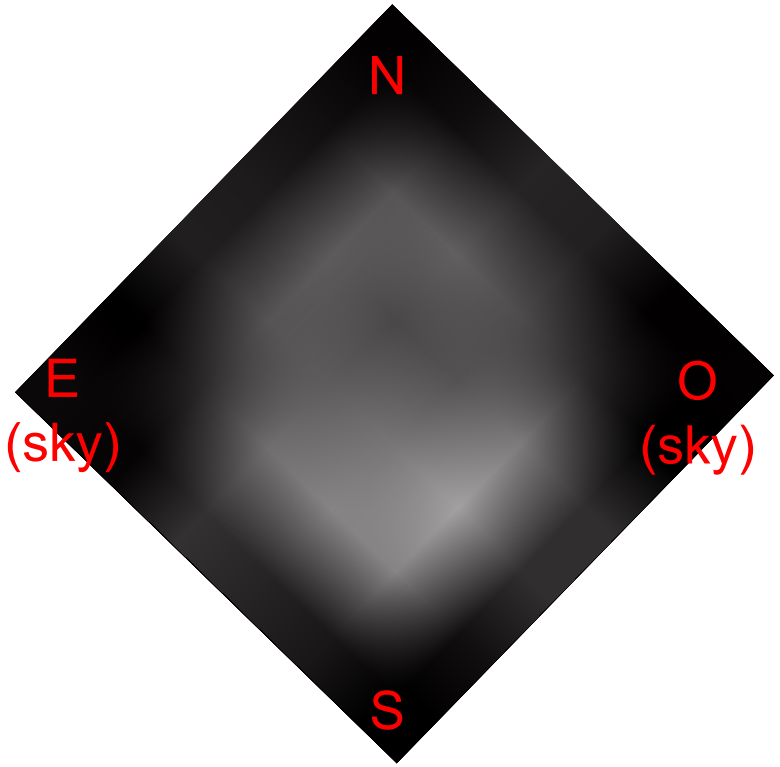
Actualmente practico la astronomía como hobby a través de la ***Agrupación Astronómica de Madrid*** donde he de admitir que mi conocimiento en la disciplina, sobre todo en la parte de astrofotografía ha aumentado considerablemente

Con la aplicación a desarrollar, trato de tener algo en el móvil que me permita conocer exactamente, dónde está lo que quiero observar

Introducción

Planteo por tanto el siguiente problema práctico:

¿Dada la siguiente foto orientada Norte – Sur ? ¿Dónde tendría que enfocar el telescopio para encontrar el Cráter Copérnico?



Introducción

Actualmente uno de los mejores softwares para localización de accidentes geográficos en la superficie lunar y que admite descarga gratuita es el denominado *Atlas Virtual Lunar* y que se puede descargar en:

https://virtual-moon-atlas.uptodown.com/windows

Esta versión es muy completa y permite aprender mucho sobre la superficie lunar, la única pega es que no hay versión en móvil, pero si es cierto que para una observación lunar, donde por lo general se puede hacer desde casi cualquier sitio, ya que no le afecta en gran grado la contaminación lumínica, si llevamos el portátil cerca, sin duda hace las delicias por sus múltiples funciones, claro está, mucho más avanzadas que las que se presentan aquí.

Por tanto, con la explicación de lo que se da aquí en cierto modo se va a entender parte del funcionamiento de la anterior aplicación

Introducción

Obviamente no llego al nivel de virtual moon pero ya veremos más adelante claro está y las ideas se agradecen …

2 | Datos de la Luna ¿Dónde buscarlos?

Datos de la Luna. ¿Dónde buscarlos?

- Ahora que estoy empezando con incorporar elementos de Astronomía a mis desarrollos en R, ya que estoy re-descubriendo mi *vocación frustrada*, lo primero que veo es que hay un mar de datos que impone y aún me abruma cuando quiero hacer cualquier cosa

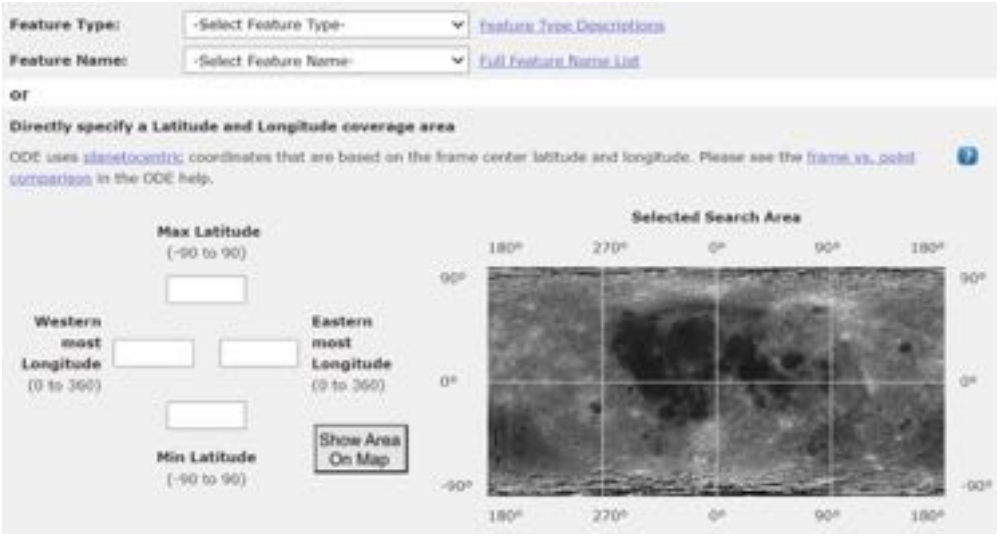
- Uno de esos repositorios de información está en la NASA, y si no se va con ideas muy en concreto de lo que se quiere, es muy posible perderse en búsquedas de información ya que hay muchísimas cosas, muchas para ciudadano de a pié (como es el caso), pero también otras de cierto nivel científico que requiere un conocimiento profundo para su tratamiento

- Ni que decir que hay información en otras muchísimas agencias, pero ya la que existe aquí solamente, resulta ser abrumadora

¿Cómo preparé la información? Parte I: Localización

- La información la preparar a partir del contenido de la siguiente url: https://ode.rsl.wustl.edu/moon/indextools.aspx?displaypage=lolardr

- Aquí se llega a la siguiente web de la NASA:



- Lo que interesaba era pues el contenido de la lista *Full Feature Name List*

- El problema de la anterior lista es que no era un fichero, sino que de los más de 9000 accidentes registrados, iba ofreciéndolos de 100 en 100

¿Cómo preparé la información? Parte II: Extracción

- Pues nada, había que hacer una cosa que me gusta poco …..

Sí eso … Trabajar !!!!!!!

Full List of Features Available in the Lunar Orbital Data Explorer

1 2 3 4 5 6 7 8 9 10 ... >>

Feature Name	Feature Type	Origin	Center Point (Lat/Lon)
8 Homeward	Crater	Crater visible in the iconic Â□□Earthriseâ□□ colour photograph, taken aboard Apollo 8 by W. A. Anders on December 24, 1968, symbolizes the safe return to Earth of Apollo 8. The crater was previously designated Ganskiy (Hanskiy) M.	-12.0249,97.0932
Abbe	Crater	Ernst Karl; German optician, physicist, astronomer (1840-1905).	-57.5755,174.775
Abbe H	Satellite Feature	Ernst Karl; German optician, physicist, astronomer (1840-1905).	-58.4449,177.584
Abbe K	Satellite Feature	Ernst Karl; German optician, physicist, astronomer (1840-1905).	-59.8159,176.858
Abbe M	Satellite Feature	Ernst Karl; German optician, physicist, astronomer (1840-1905).	-61.7462,175.239
Abbot	Crater	Charles Greeley; American astrophysicist (1872-1973).	5.5562,54.7431
Abel	Crater	Niels Henrik; Norwegian mathematician (1802-1829).	-34.6257,85.7762

- En definitiva se podía hacer 2 cosas: (1) pensar y hacer un scrapping maravilloso que trajese la información. (2) bajarlos a lo besita con el famoso *copy – paste* a un Excel para posteriormente tratar la información debidamente

Adivina adivinanza: ¿Cómo lo hice?

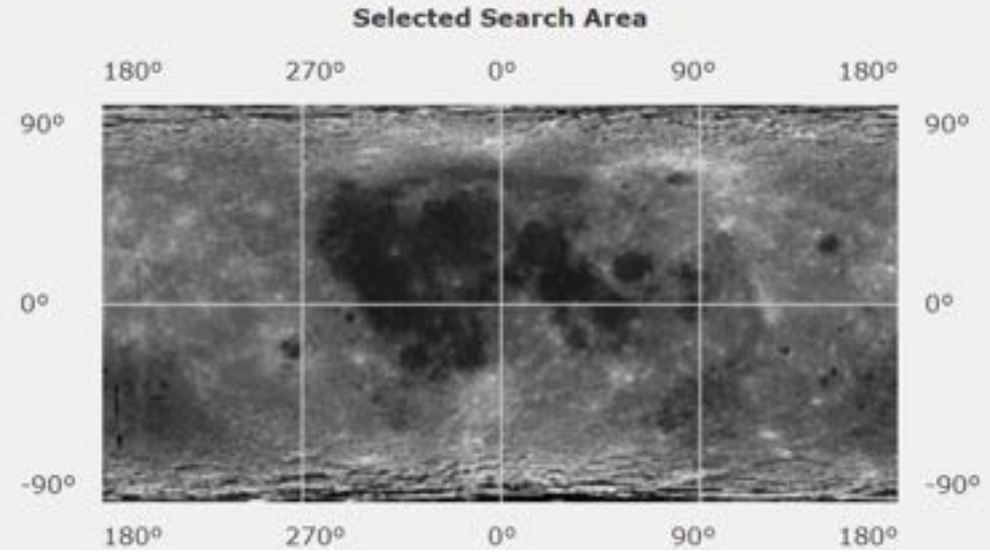
- Bueno, tras haberlo llevado a un Excel, se pudo separar la información por columnas perfectamente con algún que otro truquillo y después empleé el separador |, lo cual fue suficiente para que la información no se mezclase

```
"ch_name"|"ch_type"|"ch_data"|"num_lat"|"num_long"
"8 Homeward"|"Crater"|"Crater visible in the iconic Earthrisea colour photograph, taken aboard Apollo 8 by W A Ande
"Abbe"|"Crater"|"Ernst Karl; German optician, physicist, astronomer (1840-1905)"|-57.5755|174.775
"Abbe H"|"Satellite Feature"|"Ernst Karl; German optician, physicist, astronomer (1840-1905)"|-58.4449|177.584
"Abbe K"|"Satellite Feature"|"Ernst Karl; German optician, physicist, astronomer (1840-1905)"|-59.8159|176.858
"Abbe M"|"Satellite Feature"|"Ernst Karl; German optician, physicist, astronomer (1840-1905)"|-61.7462|175.239
"Abbot"|"Crater"|"Charles Greeley; American astrophysicist (1872-1973)"|5.5562|54.7431
"Abel"|"Crater"|"Niels Henrik; Norwegian mathematician (1802-1829)"|-34.6257|85.7762
```

¿Cómo preparé la información? Parte III: Tratamiento y lectura

- Una vez leídos los datos, el tratamiento con R es muy sencillo, sin embargo había que tener en cuenta las siguientes restricciones:

  - Se tienen datos de la cara visible y de la cara oculta
  - El formato de los datos es lógicamente el de USA
  - Es posible tener que aplicar posteriormente alguna transformación de coordenadas, ya que el sistema de referencia en el que están es de -180º a 180º



ch_name	ch_type	ch_data	num_lat	num_long
Dante	Crater	Alighieri; Italian poet (1265-1321)	25.3552	179.996
Hayford U	Satellite Feature	John Fillmore; American civil engineer, geodesist (1868-1925)	13.9491	179.812
Duner A	Satellite Feature	Nils Christofer; Swedish astronomer (1839-1914)	47.1187	179.784

3 | La librería Magick

Objetivo: Mostrar una foto proyectada de la Luna

- El objetivo del trabajo, era poder ubicar cualquier accidente geográfico de la cara visible de la luna en un mapa circular para ello se necesitaba:

  - Una imagen de la luna proyectada acorde a una determinada proyección
  - Poder escribir sobre la imagen, controlar en todo momento su tamaño, poder contar los píxeles que contiene

- En una primera instancia se optó por trabajar con el siguiente mosaico lunar ubicado en: https://i2.wp.com/cosmicreflections.skythisweek.info/wp-content/uploads/2020/05/HRF8CG-scaled.jpg?ssl=1

Magick: Un descubrimiento

- Hasta descubrir la librería *magick* de R, cuya instalación desde CRAN resulta directa, siempre he preferido para tratamiento de ficheros .jpg la utilización de Python, pero ahora, con esta librería, todo cambia. Agradezco el consejo de Carlos en su utilización

- Esta librería además permite tratamiento avanzado de imágenes que aquí no se tratará y que más adelante sin duda usaré. Para aprender más cabe consultar en: https://cran.r-project.org/web/packages/magick/vignettes/intro.html

- En este estudio sólo fueron necesaria las siguientes funciones de dicha librería:

  - *image_read()*: Permite leer cualquier formato de imagen, png, jpeg, tiff, … fue realmente la función que me convenció de la potencia de esto
  - *image_scale()*: Sirve para re-escalar imágenes a un determinado tamaño. Permite por tanto un encuadre rápido de cualquier imagen a cualquiere " caja" donde se vaya a representar. Esta función la uso bastante en otra aplicación que construí para llevar mis fotos y su historia conmigo en: https://tbfjroar.shinyapps.io/app_fotos/?_ga=2.210265495.362213178.1642851712-570603665.1639696860
  - *Image_draw()*: Permite convertir los elementos *imageck* en objetos de imagen para R que se pueden manipular directamente, en este caso para crearle un cuadrado y dibujar sobre la imagen base como se explica más adelante
  - *image_write()*: Con esta función se cierra el ciclo y permite exportar lo generado por pantalla a la imagen que se quiera en el formato escogido

Cartografía Lunar I: Orthographic proyection

- Como se puede ver aquí proyecciones hay muchas, había que elegir la correcta: https://scitools.org.uk/cartopy/docs/v0.15/crs/projections.html

- Como se observa en los datos. Éstos vienen dado en base a una representación en la esfera considerando por tanto dicha esfera. No obstante, cuando desde una distancia infinita (o suficientemente grande como es el caso), se observan los accidentes geográficos, éstos aparecen como si estuvieran en un plano, no se distingue la curvatura de la esfera

- Así pues si se observa la imagen base, se aprecia que los meridianos no tienen todos el mismo grosor y los paralelos disminuyen de grosor conforme se acercan a los polos

- La solución por tanto para representar los accidentes geográficos se encuentran pro tanto en la conocida proyección orthographica (cuya transcripción al español no me gusta como ortográfica que se confunde con la ortografía …)

- Esta proyección se conoce desde tiempos de Hipparco 200 años antes de Cristo

- Su formulación matemática es bastante sencilla:

$$x = R\cos\varphi\sin(\lambda - \lambda_0)$$
$$y = R\big(\cos\varphi_0\sin\varphi - \sin\varphi_0\cos\varphi\cos(\lambda - \lambda_0)\big)$$

Cartografía Lunar II: Normalización de la información

- Otro detalle a tener en cuenta es la información proporcionada por la NASA. Los datos están bajo las siguientes restricciones:

  - En latitud: De 0º a +90º sería el hemisferio norte, de 0º a -90ª sería el hemisferio sur
  - En longitud: De -90º a +90º sería la cara visible y el resto la invisible, de lo anterior se tendría que:

    - De -90º a 0º sería lo que corresponde al Oeste del mapa lunar considerado
    - De 0º a 90º sería lo que corresponde al Este lunar considerado

- En el fichero *global.R* se considera en primera instancia la selección de los datos de la cara visible, es decir, las 2 últimas sub – condiciones:

```
#Se selecciona sólo objetos de cara visible
df <- fread('data/geo/moon_geografy.txt', sep = '|')

df_visible <- df[which(df$num_long >= -90 & df$num_long <= 90),]
```

Cartografía Lunar III: Proyección de la información

- Para proyectar los datos según su longitud y latitud, se recurrió a la construcción de la siguiente función en R:

```
ortograph_projection <- function(long, lat){

  long_rad = long * 3.14159 / 180.0
  lat_rad = lat * 3.14159 / 180.0

  coords = c((201) * cos(lat_rad) * sin(long_rad) + (247+252)/2,
             -(201) * sin(lat_rad)  + (216+221)/2)

  return(coords)
}
```

Se pasan los datos a radiantes

Traslación para centrar en el centro de la imagen (x; y)

Radio de la imagen de la Luna en la imagen de longitud 500

- Con esta transformación de datos y por las restricciones de la información, se conseguiría a partir de un punto de la base de datos de partida, generar un punto en la superficie de la esfera

Cartografía Lunar IV: Visualización en el mapa elegido

- Nótese que la función anterior es dependiente del tamaño de la figura a representar y del contenido de ésta (dónde está el objeto y de qué tamaño es)

- La siguiente función genera un recuadro a partir de un punto de longitud – latitud dado y lo representa en el mapa considerado

```
localizador_lunar <- function(long, lat){

    #Se obtienen las coordenadas del punto a representar
    punto <- ortograph_projection(long, lat)

    #Se generan las coordenadas del cuadrado centrado en este punto

    cuadrado <- c(punto[1]-2.5, punto[2]-2.5,
                  punto[1]+2.5, punto[2]+2.5)

    moon2 <- image_scale(moon, "500")

    img <- image_draw(moon2)
    rect(cuadrado[1],
         cuadrado[2],
         cuadrado[3],
         cuadrado[4],
         border = "red",
         lwd = 2)
    dev.off()
    image_write(img,
                'data/moon_mod.png',
                format = 'png')
}
```

- Se observa cómo esta función genera un vector *punto* que es el permite construir un *cuadrado* de tamaño 5, para que pueda ofrecer su localización en el mapa. El código es bastante auto – explicativo y como se observa, se guarda una imagen final en la carpeta de destino denominado *moon_mod.png*

Resultado Final

- Básicamente el resultado final sería el siguiente:

4 | Un shiny interactivo

Una estructura muy básica en el *ui.R*

- A continuación se muestra el shiny desarrollado para tal fin y que se puede visualizar en vuestros móviles en la siguiente url: https://tbfjroar.shinyapps.io/app_moon/?_ga=2.114803589.362213178.1642851712-570603665.1639696860

- En su construcción hay un selector de información que lee los datos directamente filtrados de la cara visible por nombre:

```
column(4, wellPanel(selectizeInput("obj1",
                                   "Selecciona/Busque un accidente geográfico: ",
                                   choices = df_visible$ch_name)
)),
```

- También hay 2 elementos de visualización, una tablita con las características de los accidentes geográficos y el mapa con la ubicación del punto geográfico a representar:

```
  column(6, DTOutput("tbl"))


), #end fluidRow
fluidRow(column(4, imageOutput("image1")
  )
  ) #end fluidRow
```

Algunos detalles del fichero *server.R*

- El fichero *server.R* está escrito de modo reactivo, por lo que cada vez que se elija un objeto de la base de datos a través del *ui.R* se intentará generar tanto la *tabla de contenidos* como el *mapa lunar*

- Respecto de la *tabla de contenidos* se hace uso del habitual *renderDT{}* donde se genera la tabla a observar con un sencillo tratamiento de datos en R

```
accidente <- df_visible %>%
  filter(ch_name == input$obj1)

char <- c("Tipo de accidente",
          "Descripción",
          "Longitud",
          "Latitud")
value <- c(accidente$ch_type,
           accidente$ch_data,
           accidente$num_long,
           accidente$num_lat)

df <- data.frame(char = char,
                 value = value)
```

```
datatable(df,
          colnames = c("Datos del objeto lunar", "Valores"),
          options = list(
            pageLength = 5,
            dom = 't',
            initComplete = JS(
              "function(settings, json) {",
              "$(this.api().table().header()).css({'background-color': '#000',
              'color': '#fff'});",
              "}")),
          rownames = FALSE)
```

Algunos detalles del fichero *server.R*

- Respecto del *mapa* de lo que se hace uso es de *renderImage{}* gracias a esta opción, la integración de *imageck* en shiny resulta total. Bajo esta función de shiny se hace tanto la definición (que resulta más correcto hacerlas en *global.R*) como las llamadas a las funciones anteriormente comentadas para generar la imagen con el marcador final:

```
accidente <- df_visible %>%
  filter(ch_name == input$obj1)

longitud = accidente$num_long
latitud = accidente$num_lat

localiza <- localizador_lunar(longitud, latitud)

# Return a list
list(src = 'data/moon_mod.png',
     contentType = "image/png")

},
deleteFile = FALSE)
```

Se toma el dato que el usuario elije en el ui.R y se genera el vector *accidente*

Se separan las componentes para mejorar su comprensión y simplificar su uso

Se introducen los datos en la función principal para que se generere el dibujo con el puntero

Esta función devuelve la imagen a partir de otra que existe físicamente y ha sido generada (por la función anterior). Hay que indicar que mantenga la representación

Versión 0 de la aplicación Moon Locator

5 | Conclusiones Finales

Conclusiones Finales

- Este trabajo trata de ofrecer la versatilidad de R para problemas más allá del propio cálculo estadístico, aunque como se ha observado, un mínimo de matemáticas, siempre es necesario para cualquier tipo de análisis como el aquí mostrado y para motivarnos a aprender algo nuevo, nada mejor que encontrar grupos como los de la agrupación que nos ayuden a entender este tipo de cosas algo más complejas, pero fascinantes a la vez

https://www.aam.org.es/

- Por supuesto aplicaciones como *Atlas Lunar* están en una liga superior respecto a la aplicación aquí mostrada. Lo único que cabe conseguir es que. al ser una aplicación propia, es factible ir añadiendo modificaciones muy personalizadas que permitan hacer una clara diferenciación de este producto con los ya existentes

- Lo mismo que se ha hecho para la cara visible cabe considerar para la oculta, incluso cabe deducir las antípodas dado un determinado punto visible

- No se han tendido en cuenta muchos otros aspectos de la luna como son, la posición de ésta respecto a la geografía o la fase de ocultación que en posteriores versiones podrían considerarse, eso sí, el nivel matemático empezaría a elevarse en ese momento, pero por otro la haría a la app más interesante

- También otro aspecto a mejorar, es tratar de usar un mapa más nítido (o incluso el mismo a un nivel de calidad superior que también se puede) e incluso modificar los datos para una representación en una pantalla mayor

Conclusiones Finales

- Finalmente, con un poco más de complicación se podrían añadir capas extendidas del área a representar, ampliándolo incluso a un nivel tipo Google, para esto se requeriría un mosaico de alta resolución

- Si alguien se quiere descargar código y datos inclusive, puede acceder a mi git (recuerden que los datos en origen proceden de diversas fuentes, aunque hay una preparación intermedia como se ha comentado, su uso está limitado a las condiciones que desde dichas entidades se refiera, esta presentación a tenido fines exclusivamente didácticos sin ánimo de lucro):

https://github.com/FJROAR/Moon-App

Afi