Extracción y procesamiento de datos de lluvia

CHIRPS Daily via Google Earth Engine

Isaac Arroyo

6 de mayo de 2024

Tabla de contenidos

Capítulo 1

Introducción

En este documento se encuentran documentados los pasos y el código para la extracción mensual[1] de variables derivadas de la precipitación, tales como: precipitación mensual promedio, anomalía de la precipitación en porcentaje con respecto de la normal y anomalía de la precipitación en milímetros con respecto de la normal.

Cada aspecto del código se documenta en diferentes capítulos.

[1]Con posibilidad de que adaptar el código para que el periodo sea semanal

Capítulo 2

Sobre los datos

Los fuente de los datos se llama **CHIRPS (Climate Hazards Group InfraRed Precipitation With Station Data) Daily**, se puede encontrar en diferentes lugares, uno de estos la plataforma **Google Earth Engine**.

De acuerdo con la descripción:

> *Climate Hazards Group InfraRed Precipitation with Station data (CHIRPS) is a 30+ year quasi-global rainfall dataset. CHIRPS incorporates 0.05° resolution satellite imagery with in-situ station data to create gridded rainfall time series for trend analysis and seasonal drought monitoring.*

Estos datos cuentan con la **precipitación diaria** medida en milímetros (mm) desde Enero 01, de 1981 hasta el mes inmediato anterior a la fecha actual[1]

El procesamiento de texto es similar al realizado en el proyecto "Desplazamiento climático: La migración que no vemos".

```
import ee                                                    ①
import geemap                                                ②
import time

try:
    ee.Initialize()                                          ③
    print("Se ha inicializado correctamente")
except:
    print("Error en la inicialización")
```

① Importar API de (Google) Earth Engine
② Importar `geemap` para la creación de mapas interactivos tipo folium
③ Inicializar API

[1]Esto quiere decir, que si la fecha *actual* es Abril 2024, entonces los datos cubren hasta Marzo 2024

Capítulo 3

Varibles y constantes

La extracción de datos se planea que sea periódica a niveles estatales y municipales, por lo que se dejan declarados variables que se mantendrán constantes (como el rango del promedio *normal* o histórico) o (valga la redundancia) cambiarán dependiendo de los datos que se quieran. La Tabla 3.1 entra a mayor detalle de lo que se esta haciendo

Tabla 3.1: Variables y constantes

Variable	Tipo	Notas
`geom_mex`	Constante	Geometría del perímetro de México, usada para delimitar espacialmente la información
`fc`	Cambiante	`ee.FeatureCollection` de las geomtrías e información de los Estados o Municipios de México
`chirps`	Constante	`ee.ImageCollection` de CHIRPS Daily

```
select_fc = "ent"                                                              ①
dict_fc = dict(                                                                ②
    ent = "projects/ee-unisaacarroyov/assets/GEOM-MX/MX_ENT_2022",
    mun = "projects/ee-unisaacarroyov/assets/GEOM-MX/MX_MUN_2022")

fc = ee.FeatureCollection(dict_fc[select_fc])                                  ③

geom_mex = (ee.FeatureCollection("USDOS/LSIB/2017")                            ④
            .filter(ee.Filter.eq("COUNTRY_NA", "Mexico"))                      ⑤
            .first()                                                           ⑥
```

```
            .geometry())                                                    ⑦


chirps = (ee.ImageCollection('UCSB-CHG/CHIRPS/DAILY')                       ⑧
          .select("precipitation")
          .filter(ee.Filter.bounds(geom_mex)))                              ⑨
```

① Seleccion de `ee.FeatureCollection`, sea de Entidades (`ent`), Municipios (`mun`) o Cuencas Hidrológicas (`ch`)
② Diccionario con los *paths* hacia la `ee.FeatureCollection` de elección
③ Carga de `ee.FeatureCollection` de interés
④ `ee.FeatureCollection` de división politica de los países del mundo
⑤ Filtro donde la propiedad `COUNTRY_NA` sea igual a **Mexico**
⑥ Selección de la primera `ee.Feature`
⑦ Extracción de únicamente la geometría
⑧ `ee.ImageCollection` de CHIRPS Daily
⑨ Limitar el raster a la geometría de México

Capítulo 4

Modificiación de la `ee.ImageCollection`

`chirps` es una `ee.ImageCollection` de más de 15 mil imágenes (`ee.Image`). Lo que se busca hacer es tener una colección de poco más de 40 imágenes con 12 bandas, estas bandas son los meses del año y la información de cada banda será la misma: la precipitación mensual.

4.1 Etiquetado de año y mes del año

Para ir agrupando y sumando la precipitación mensual, hay que tener las imágenes etiquetadas con el año y mes para poder agruparlas y sumar la precipitación. Para ello se crea una función que haga ese etiquetado en cada una de las imágenes

```
1 def func_tag_year_month(img):                                          ①
2     full_date = ee.Date(ee.Number(img.get("system:time_start")))       ②
3     n_year = ee.Number(full_date.get("year"))                          ③
4     n_month = ee.Number(full_date.get("month"))                        ④
5     return img.set({"n_month": n_month, "n_year": n_year})             ⑤
6
7 chirps_tagged = chirps.map(func_tag_year_month)                        ⑥
```

① La función toma una sola `ee.Image`
② Obtener la fecha de la imagen, como esta en formato UNIX, se tiene que transformar a fecha con `ee.Date`
③ De la fecha se obtiene el valor numérico del año
④ De la fecha se obtiene el valor numérico de la semana del año
⑤ Asignación de año y semana del año como propiedades de la `ee.Image`
⑥ Crear nueva `ee.ImageCollection` con el etiquetado

4.2 Precipitación mensual (1981-2023)

Antes de crear la colección de ≈ 43 imágenes con 12 bandas cada una, hay que reducir la colección de 365 imagenes por año a 12 imágenes por año, donde cada imagen tenga la precipitación del mes.

```
list_months = ee.List.sequence(1, 12)
list_years = ee.List.sequence(1981, 2023)
```

La transformación conlleva múltiples iteraciones, y mientras que en JavaScript se puedan declarar funciones dentro de `map`, para el caso se Python se tendran que crear las funciones a parte, y después serán llamadas a su respectivo `map`.

```
def func_iter_years(n_year):                                                    ①
    img_coll_year_interes = (chirps_tagged
                             .filter(ee.Filter.eq("n_year", n_year)))           ②
    def func_iter_years_iter_months(n_month):                                   ③
        return (img_coll_year_interes                                           ④
                .filter(ee.Filter.eq("n_month", n_month))
                .sum()
                .set({"n_year": n_year, "n_month": n_month}))                   ⑤

    list_monthly_pr_per_year = (list_months                                     ⑥
                                .map(func_iter_years_iter_months))

    return list_monthly_pr_per_year                                             ⑦
```

① Función que itera sobre elementos de una `ee.List`, estos elementos son los los años que ocupa la `ee.ImageCollection`
② Se filtra el año de interes
③ Función para iterar sobre elementos de una `ee.List`, estos elementos son los los meses que ocupa la `ee.ImageCollection`
④ Por cada año y cada mes (de ese año), se va a retornar (de función anidada) se va a regresar una `ee.Image`, que es resultado de reducir la `ee.ImageCollection` que cumple con las condiciones del año-mes. El reductor principal es la suma.
⑤ La nueva imagen tiene como propiedades el año y el mes que representa.
⑥ Se aplica la función que itera sobre meses, a la lista de meses del año.
⑦ El resultado final de la función *general*, es que por cada elemento (año) hay una lista de 12 imagenes, que representan la precipitación mensual de ese elemento.

```
list_year_monthly_pr = (list_years                                              ①
                        .map(func_iter_years)
```

```
                          .flatten())                                          ②

img_coll_year_monthly_pr = (ee.ImageCollection                                 ③
                            .fromImages(list_year_monthly_pr))
```

① Se aplica la función para reducir el número de imagenes en la colección
② Como el resultado es una lista de (poco más de 40) listas (de 12 imágenes), entonces se tiene que *aplanar* es decir, sacar los elementos de cada sublista y que sean parte de la lista completa/general
③ Se crea una colección de imágenes a partir de una lista de imágenes

4.3 Meses como bandas de imágenes

Ya que se logró pasar de más de 15 mil imágenes a poco más de 500, ya es tiempo de reducir a ≈ 40 imágenes de a 12 bandas cada una.

```
def imgcoll2bands(n_year):                                                     ①
    img_12bands = (img_coll_year_monthly_pr                                    ②
        .filter(ee.Filter.eq("n_year", n_year))
        .toBands()                                                             ③
        .rename([f"0{i}" if i < 10 else str(i) for i in range(1,13)])          ④
        .set({"n_year": n_year}))                                              ⑤
    return img_12bands                                                         ⑥

img_coll_year_monthly_pr_bands = (ee.ImageCollection                           ⑦
    .fromImages(list_years.map(imgcoll2bands)))
```

① Función para transformar una `ee.ImageCollection` de $\boldsymbol{n}$ imágenes a una `ee.Image` de $\boldsymbol{n}$ bandas
② Filtrar por año de interés
③ Transformar a una imagen con el número de bandas igual al número de imagenes que tenía la colección.
④ Agregar a la imagen, la propiedad del año al que pertenece.
⑤ Se regresa una imagen de 12 bandas (si es el año esta completo)
⑥ Aplicar la función a una lista de años.

Capítulo 5

Cálculo de anomalías de precipitación

5.1 Acumulación normal

De acuerdo a con el glosario de la NOAA una anomalía es la desviación de una unidad dentro de un periodo en una región en específico con respecto a su promedio histórico o normal. Este promedio es usualmente de 30 años.

Para el caso del CHIRPS, es de 1981 hasta el 2010 (incluyendo a diciembre)

```
base_period = (img_coll_year_monthly_pr
               .filter(ee.Filter.lte("n_year", 2010)))                    ①
base_pr_monthly_accumulation = (ee.ImageCollection
    .fromImages(
        (list_months                                                      ②
         .map(lambda n_month: (base_period                                ③
                               .filter(ee.Filter.eq("n_month", n_month))
                               .mean()                                    ④
                               .set("n_month", n_month)))))               ⑤
    .toBands()                                                            ⑥
    .rename([f"0{i}" if i < 10 else str(i) for i in range(1,13)]))        ⑦
```

① De la colección cuyas imagenes son de 12 bandas, se filtran aquellas que sean del 2010 *para abajo*
② Ir por meses
③ Filtrar al mes del interes
④ Calcular el promedio de la precipitación de esos 30 años
⑤ A esa imagen darle la propiedad del valor del mes
⑥ Como el resultado es una colección de 12 imágenes, se cambia a una imagen de 12 bandas
⑦ Renombrar las bandas (meses) con el número del mes del año

5.2 Anomalía en milimetros

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc dictum turpis ullamcorper pharetra pretium. Vivamus eu pellentesque nibh. Mauris ac massa faucibus, condimentum eros at, vehicula justo. Cras ultrices gravida risus, quis tempor tortor hendrerit quis. Aliquam erat volutpat. Nullam tincidunt iaculis varius. Donec tristique leo non sapien sagittis, in tincidunt lorem bibendum. Integer commodo sem vel risus hendrerit efficitur. Pellentesque ut tincidunt ante, finibus sodales tellus.

$$\text{anom}_{\text{mm}} = \overline{x}_i - \mu_{\text{normal}}$$

```
#TODO: COMENTAR 04
img_coll_year_monthly_anomaly_mm = (img_coll_year_monthly_pr_bands
    .map(lambda img: (img
                      .subtract(base_pr_monthly_accumulation)
                      .copyProperties(img, img.propertyNames()))))
```

5.3 Anomalía en porcentaje

Mauris porta lorem nisi, et mollis ligula eleifend sed. Donec tristique sed orci quis cursus. Pellentesque vulputate vel turpis eget maximus. Cras et rutrum neque, et accumsan felis. Nam vel leo scelerisque, pharetra quam feugiat, fermentum leo. Nullam consequat turpis non eros fermentum suscipit. Suspendisse sed dui nec tellus vulputate volutpat at nec tortor. Etiam tempus ut sapien non condimentum.

$$\text{anom}_{\%} = \frac{\overline{x}_i - \mu_{\text{normal}}}{\mu_{\text{normal}}}$$

```
#TODO: COMENTAR 05
img_coll_year_monthly_anomaly_prop = (img_coll_year_monthly_pr_bands
    .map(lambda img: (img
                      .subtract(base_pr_monthly_accumulation)
                      .divide(base_pr_monthly_accumulation)
                      .copyProperties(img, img.propertyNames()))))
```

Capítulo 6

Guardar en tablas por años

Nullam accumsan dolor a justo dapibus, sit amet interdum metus rhoncus. Praesent ac libero hendrerit, dapibus metus ac, dignissim tellus. Nunc ut enim ut ligula posuere eleifend. Vestibulum ac lorem in massa lacinia condimentum sed eget ligula. Maecenas imperdiet felis sit amet arcu viverra tristique. Maecenas suscipit mattis massa, ut malesuada erat consequat tristique. Nulla tincidunt augue vel ante aliquam, in ultricies purus laoreet.

```
#TODO: COMENTAR 06
def func_create_list_of_fc(imgcoll, featurecoll, scale_img_coll = 5566):
    list_fc = list()

    for n_year_interes in range(1981, 2024):
        img_year_month = (imgcoll
                          .filter(ee.Filter.eq("n_year", n_year_interes))
                          .first())

        fc_from_image = (img_year_month
                         .reduceRegions(
                             reducer = ee.Reducer.mean(),
                             collection = featurecoll,
                             scale = scale_img_coll)
                         .map(lambda feature: (ee.Feature(feature)
                                               .set({'n_year': n_year_interes})
                                               .setGeometry(None))))

        fc_final = ee.FeatureCollection((fc_from_image
                                         .toList(3000)
                                         .flatten()))
        list_fc.append(fc_final)

    return list_fc
```

```

def save_all_years_fc(
        list_of_fc,
        descrp,
        filename,
        folder_name,
        fc_type= select_fc):

    iteracion = 0

    for vector in list_of_fc:
        print(f"Mandando a guardar la tabla de {1981 + iteracion}")
        print(f"Bajo el asunto {descrp}")
        final_filename = f"{filename}_{fc_type}_{1981 + iteracion}"
        final_descrp = f"{descrp}_{fc_type}_{1981 + iteracion}"
        geemap.ee_export_vector_to_drive(
            collection= vector,
            description= final_descrp,
            fileNamePrefix= final_filename,
            fileFormat= "CSV",
            folder= folder_name)
        print(f"El nombre del archivo es {final_filename}.csv")
        iteracion += 1

        time.sleep(120)

    return None
```

6.1 Acumulación de lluvias

Vivamus at vestibulum elit. Maecenas in dui at diam aliquet feugiat. In justo nisi, cursus vitae augue a, faucibus consectetur felis. Duis nisi lorem, scelerisque a libero et, posuere vulputate nibh. Nulla dictum enim ac nisi congue egestas. Curabitur volutpat mi nec tristique mattis. Nulla sed dictum ante. Nunc vitae erat neque. Morbi rhoncus ex ac tellus maximus, nec cursus lorem semper. Donec porta congue placerat. Ut finibus est tellus, ut elementum nisl dictum nec. Nulla facilisi. Etiam auctor quam ac nunc condimentum mollis. Pellentesque libero mi, finibus sit amet fermentum non, condimentum eu dolor. Nam hendrerit ullamcorper nunc, tristique facilisis erat sagittis non. Pellentesque fermentum, magna vel feugiat pellentesque, odio diam facilisis erat, et ultricies dui erat at velit.

```
#TODO: COMENTAR 07
list_fc_pr = func_create_list_of_fc(
    imgcoll= img_coll_year_monthly_pr_bands,
    featurecoll= fc)
```

El siguiente bloque de código es el que se usa para guardar la información del periodo 1981 - 2023. Las solicitudes son enviadas al servidor y el archivo CSV se guardará en la carpeta de Google Drive cuando el servidor haya terminado de ejecutar la solicitud.

```
save_all_years_fc(
    list_of_fc= list_fc_pr,
    descrp= "chirps_daily_precipitation",
    filename= "chirps_daily_pr",
    folder_name= "gee_chirps_daily_pr"
)
```

6.2 Anomalía de lluvias

6.2.1 Anomalía en milimetros

Vivamus at vestibulum elit. Maecenas in dui at diam aliquet feugiat. In justo nisi, cursus vitae augue a, faucibus consectetur felis. Duis nisi lorem, scelerisque a libero et, posuere vulputate nibh. Nulla dictum enim ac nisi congue egestas. Curabitur volutpat mi nec tristique mattis. Nulla sed dictum ante. Nunc vitae erat neque. Morbi rhoncus ex ac tellus maximus, nec cursus lorem semper. Donec porta congue placerat. Ut finibus est tellus, ut elementum nisl dictum nec. Nulla facilisi. Etiam auctor quam ac nunc condimentum mollis. Pellentesque libero mi, finibus sit amet fermentum non, condimentum eu dolor. Nam hendrerit ullamcorper nunc, tristique facilisis erat sagittis non. Pellentesque fermentum, magna vel feugiat pellentesque, odio diam facilisis erat, et ultricies dui erat at velit.

```
#TODO: COMENTAR 07

list_fc_anomaly_pr_mm = func_create_list_of_fc(
    imgcoll= img_coll_year_monthly_anomaly_mm,
    featurecoll= fc)
```

6.2.2 Anomalía en porcentaje

Vivamus at vestibulum elit. Maecenas in dui at diam aliquet feugiat. In justo nisi, cursus vitae augue a, faucibus consectetur felis. Duis nisi lorem, scelerisque a libero et, posuere

vulputate nibh. Nulla dictum enim ac nisi congue egestas. Curabitur volutpat mi nec tristique mattis. Nulla sed dictum ante. Nunc vitae erat neque. Morbi rhoncus ex ac tellus maximus, nec cursus lorem semper. Donec porta congue placerat. Ut finibus est tellus, ut elementum nisl dictum nec. Nulla facilisi. Etiam auctor quam ac nunc condimentum mollis. Pellentesque libero mi, finibus sit amet fermentum non, condimentum eu dolor. Nam hendrerit ullamcorper nunc, tristique facilisis erat sagittis non. Pellentesque fermentum, magna vel feugiat pellentesque, odio diam facilisis erat, et ultricies dui erat at velit.

```
#TODO: COMENTAR 08

list_fc_anomaly_pr_prop = func_create_list_of_fc(
    imgcoll= img_coll_year_monthly_anomaly_prop,
    featurecoll= fc)
```

El siguiente bloque de código es el que se usa para guardar la información del periodo 1981 - 2023. Las solicitudes son enviadas al servidor y el archivo CSV se guardará en la carpeta de Google Drive cuando el servidor haya terminado de ejecutar la solicitud.

```
save_all_years_fc(
    list_of_fc= list_fc_anomaly_pr_prop,
    descrp= "chirps_daily_precipitation_anomaly_prop",
    filename= "chirps_daily_anomaly_pr_prop",
    folder_name= "gee_chirps_daily_anomaly_pr_prop"
)
```

Capítulo 7

Caso específico: 2024

Nullam accumsan dolor a justo dapibus, sit amet interdum metus rhoncus. Praesent ac libero hendrerit, dapibus metus ac, dignissim tellus. Nunc ut enim ut ligula posuere eleifend. Vestibulum ac lorem in massa lacinia condimentum sed eget ligula. Maecenas imperdiet felis sit amet arcu viverra tristique. Maecenas suscipit mattis massa, ut malesuada erat consequat tristique. Nulla tincidunt augue vel ante aliquam, in ultricies purus laoreet.

```
#TODO: explicar por qué es el caso especifo
#TODO: Traducir JavaScript a Python
```

```
/* *  Crear ee.Image de 12 bandas cada una * */
/*
list_year_month = list_months
                .map(function(number){
                return chirps_tagged
                        .filter(ee.Filter.eq("n_year", n_year_interes))
                        .filter(ee.Filter.eq("n_month", number))
                        .sum()
                        .set({"n_month": number});
                });
img_year_month = ee.ImageCollection
                .fromImages(list_year_month)
                .toBands()
                //.rename(["01","02","03"])
                ;
*/

/* * Crear ee.FeatureCollection de 12 columnas y n_estados/municipios * */

// Limitar a un anño en especifico
```

```
n_year_interes = 2020;
img_year_month = img_coll_year_monthly_pr_bands
                 .filter(ee.Filter.eq("n_year", n_year_interes))
                 .first();

fc_from_image = img_year_month
                 .reduceRegions({
                     'reducer': ee.Reducer.mean(),
                     'collection': fc,
                     'scale': scale_img_coll})
                 .map(function(feature){
                     return ee.Feature(feature)
                             .set({'n_year': n_year_interes})
                             .setGeometry(null)});

fc_final = ee.FeatureCollection(fc_from_image.toList(3000).flatten());
```

```
#TODO: explicar por qué es el caso especifo
#TODO: Traducir JavaScript a Python
```

7.1 Acumulación de la precipitación

Nullam accumsan dolor a justo dapibus, sit amet interdum metus rhoncus. Praesent ac libero hendrerit, dapibus metus ac, dignissim tellus. Nunc ut enim ut ligula posuere eleifend. Vestibulum ac lorem in massa lacinia condimentum sed eget ligula. Maecenas imperdiet felis sit amet arcu viverra tristique. Maecenas suscipit mattis massa, ut malesuada erat consequat tristique. Nulla tincidunt augue vel ante aliquam, in ultricies purus laoreet.

```
#TODO: explicar por qué es el caso especifo
#TODO: Traducir JavaScript a Python
```

7.2 Anomalía de la precipitación

7.2.1 Anomalía en milimetros

Nullam accumsan dolor a justo dapibus, sit amet interdum metus rhoncus. Praesent ac libero hendrerit, dapibus metus ac, dignissim tellus. Nunc ut enim ut ligula posuere eleifend. Vestibulum ac lorem in massa lacinia condimentum sed eget ligula. Maecenas imperdiet felis sit amet arcu viverra tristique. Maecenas suscipit mattis massa, ut

malesuada erat consequat tristique. Nulla tincidunt augue vel ante aliquam, in ultricies purus laoreet.

```
#TODO: explicar por qué es el caso especifo
#TODO: Traducir JavaScript a Python
```

7.2.2 Anomalía en porcentaje

Nullam accumsan dolor a justo dapibus, sit amet interdum metus rhoncus. Praesent ac libero hendrerit, dapibus metus ac, dignissim tellus. Nunc ut enim ut ligula posuere eleifend. Vestibulum ac lorem in massa lacinia condimentum sed eget ligula. Maecenas imperdiet felis sit amet arcu viverra tristique. Maecenas suscipit mattis massa, ut malesuada erat consequat tristique. Nulla tincidunt augue vel ante aliquam, in ultricies purus laoreet.

```
#TODO: explicar por qué es el caso especifo
#TODO: Traducir JavaScript a Python
```

7.3 Guardar información actualizada

Nullam accumsan dolor a justo dapibus, sit amet interdum metus rhoncus. Praesent ac libero hendrerit, dapibus metus ac, dignissim tellus. Nunc ut enim ut ligula posuere eleifend. Vestibulum ac lorem in massa lacinia condimentum sed eget ligula. Maecenas imperdiet felis sit amet arcu viverra tristique. Maecenas suscipit mattis massa, ut malesuada erat consequat tristique. Nulla tincidunt augue vel ante aliquam, in ultricies purus laoreet.

```
#TODO: explicar por qué es el caso especifo
#TODO: Traducir JavaScript a Python
```