Extracción y procesamiento de datos de lluvia

CHIRPS Daily via Google Earth Engine

Isaac Arroyo

7 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 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, donde el último capítulo estará la función final, con la que se resume y concluye el proceso de extracción.

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 time

try:
    ee.Initialize()                                          ②
    print("Se ha inicializado correctamente")
except:
    print("Error en la inicialización")
```

① Importar API de (Google) Earth Engine
② 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
`date_year_interes`	Cambiante	Año del que se van a extraer las métricas de precipitación
`fc`	Cambiante	`ee.FeatureCollection` de las geomtrías e información de los Estados o Municipios de México
`chirps`	Constante	`ee.ImageCollection` de CHIRPS Daily
`year_base_inicio`	Constante	Inicio del periodo historico para el cálculo de la normal, Enero 01 de 1981
`year_base_fin`	Constante	Fin del periodo historico para el cálculo de la normal, Diciembre 31 de 2010
`geom_mex`	Constante	Geometría del perímetro de México, usada para delimitar espacialmente la información

```python
date_year_interes = 2023 ①
select_fc = "mun" ②
dict_fc = dict( ③
    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]) ④

year_base_inicio = 1981 ⑤
year_base_fin = 2010
geom_mex = (ee.FeatureCollection("USDOS/LSIB/2017") ⑥
            .filter(ee.Filter.eq("COUNTRY_NA", "Mexico")) ⑦
            .first() ⑧
            .geometry()) ⑨

chirps = (ee.ImageCollection('UCSB-CHG/CHIRPS/DAILY') ⑩
          .select("precipitation")
          .filter(ee.Filter.bounds(geom_mex))) ⑪
```

① Selección del año de interés
② Seleccion de `ee.FeatureCollection`, sea de Entidades (`ent`), Municipios (`mun`) o Cuencas Hidrológicas (`ch`[1])
③ Diccionario con los *paths* hacia la `ee.FeatureCollection` de elección
④ Carga de `ee.FeatureCollection` de interés
⑤ Fechas de inicio y fin del periodo historico para el cálculo de la normal (30 años)
⑥ `ee.FeatureCollection` de división politica de los países del mundo
⑦ Filtro donde la propiedad `COUNTRY_NA` sea igual a **Mexico**
⑧ Selección de la primera `ee.Feature`
⑨ Extracción de únicamente la geometría
⑩ `ee.ImageCollection` de CHIRPS Daily
⑪ Limitar el raster a la geometría de México

[1] Pendiente subir a Google Earth Engine

Capítulo 4

Reducción a valores mensuales

`chirps` es una `ee.ImageCollection` de más de 15 mil imágenes (`ee.Image`). El procesamiento y tratado de las imagenes es *pesado*, y aunque la computadora no se encargue de hacer el trabajo, esto puede hacer que el servidor de Google Earth demore en hacer los cálculos y la transformación de los datos raster.

Es por eso que en el procesamiento se incluye un primer filtro: la creación de una `ee.ImageCollection` de 365 imágenes[1].

```
chirps_year_interes = (chirps
    .filter(ee.Filter.calendarRange(start = date_year_interes,
                                    field = "year")))
```

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 mes para poder agruparlas y sumar la precipitación. Para ello se crea una función que haga ese etiquetado en cada una de las imágenes

```
def func_tag_month(img):                                                    ①
    full_date = ee.Date(ee.Number(img.get("system:time_start")))            ②
    n_month = ee.Number(full_date.get("month"))                             ③
    return img.set({"n_month": n_month})                                    ④

chirps_year_interes_tagged = chirps_year_interes.map(func_tag_month)        ⑤
```

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`

[1] 366 si es año bisiesto

③ De la fecha se obtiene el valor numérico de la semana del año
④ Asignación de año y semana del año como propiedades de la `ee.Image`
⑤ Crear nueva `ee.ImageCollection` con el etiquetado

4.2 Precipitación mensual del año de interés

Para una fácil extracción de los valores del año es necesario tener la información de la precipitación (o la métrica de interés) como una `ee.Image` de 12 bandas, donde cada banda es el valor mensual de la región.

Para crear una imagen de 12 bandas se necesita primero una `ee.ImageColletion` de 12 imágenes.

Para ello, se va a crear una lista de 12 imágenes.

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

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_reduce2months(n_month):                                    ①
    return (chirps_year_interes_tagged                              ②
            .filter(ee.Filter.eq("n_month", n_month))
            .sum()                                                  ③
            .set({"n_month": n_month}))                             ④
```

① Función para sumar todas las imágenes que pertenezcan a un mes en específico. Esta función itera sobre elementos de una `ee.List`, estos elementos son los los meses que ocupa la `ee.ImageCollection`
② Se filtran aquellas imágenes que sean del mes de interés
③ Se reduce la colección a una imagen a través de la suma
④ Se le asigna la propiedad del mes que representa.

```
list_monthly_pr = (list_months.map(func_reduce2months))             ①

img_coll_monthly_pr = (ee.ImageCollection                           ②
                           .fromImages(list_monthly_pr))
```

① Se aplica la función para reducir el número de imagenes en la colección, como resultado da una lista de 12 imágenes
② Se crea una colección de imágenes a partir de una lista.

4.3 Meses como bandas de una `ee.ImageCollection`

Ya que se logró tener una colección de 12 imágenes, entonces se crea la imagen de 12 bandas

```
img_monthly_pr = (img_coll_monthly_pr
  .toBands()                                                    ①
  .rename([f"0{i}" if i < 10 else str(i) for i in range(1,13)]))  ②
```

① Pasar `ee.ImageCollection` de n imágenes a una `ee.Image` de n bandas
② Renombrar el nombre de las bandas a los números de los meses

Capítulo 5

Métricas a extraer

El objetivo es poder extraer información mensual sobre las lluvias de cada año, a las que se les prestará atención son:

- **Precipitación**
- **Anomalía de precipitación en mm**
- **Anomalía de precipitación en porcentaje**

Para las últimas dos hace falta tener el valor de la **acumulación normal**.

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).

Esta tarea se tiene que hacer con ayuda de dos funciones. La primera etiquetará por año y mes la colección *base*. La segunda función tendrá que reducir a 12 imagenes cada año de esa base.

```
def func_tag_year_month_base_period(img):                                    ①
    full_date = ee.Date(ee.Number(img.get("system:time_start")))
    n_year = ee.Number(full_date.get("year"))
    n_month = ee.Number(full_date.get("month"))
    return img.set({"n_month": n_month, "n_year": n_year})


def func_reduce2yearmonths_base_period(n_year):                              ②
    imgcoll_year_interes = (base_period_tagged
```

```
                              .filter(ee.Filter.eq("n_year", n_year)))
    def func_reduce2months_base_period(n_month):
        return (imgcoll_year_interes
                .filter(ee.Filter.eq("n_month", n_month))
                .sum()
                .set({"n_year": n_year, "n_month": n_month}))
    list_monthly_pr_per_year = (list_months
                                .map(func_reduce2months_base_period))

    return list_monthly_pr_per_year
```

① Función para etiquetar año y mes
② Función para crear una lista de listas de precipitaciones mensuales para cada año del periodo base

Con estas dos funciones se creará una colección de imagenes de ≈ 360 imágenes

```
base_period = (chirps                                                          ①
  .filter(ee.Filter.calendarRange(start = year_base_inicio,
                                  end = year_base_fin, field = "year")))

base_period_tagged = base_period.map(func_tag_year_month_base_period)          ②

base_period_tagged_reduced_year_month = (ee.ImageCollection.fromImages(        ③
    (ee.List                                                                   ④
    .sequence(year_base_inicio, year_base_fin)
    .map(func_reduce2yearmonths_base_period)                                   ⑤
    .flatten())))                                                              ⑥

base_pr_monthly_accumulation = (ee.ImageCollection.fromImages(                 ⑦
    list_months.map(lambda n_month: (                                          ⑧
                      base_period_tagged_reduced_year_month                    ⑨
                      .filter(ee.Filter.eq("n_month", n_month))
                      .mean()                                                  ⑩
                      .set({"n_month": n_month}))))                            ⑪
  .toBands()                                                                   ⑫
  .rename([f"0{i}" if i < 10 else str(i) for i in range(1,13)]))               ⑬
```

① Limitar la colección los años del periodo de referencia
② Etiquetar el año y mes al que pertenece cada imagen
③ Crear una `ee.ImageCollection` a partir de una lista
④ La es una secuencia de números que representan los años del periodo base

⑤ A cada elemento (número) se le aplica una función. Esta función regresa una lista de 12 imágenes por año, es decir, el resultado es una lista de 30 elementos, donde cada elemento es una lista de 12 imágenes.
⑥ Se cambia la lista de sublistas a una lista, es decir, se *desempacan* los elementos de las sublistas.
⑦ A partir de una lista se crea una colección
⑧ Esta lista es de 12 elementos, la lista del promedio historico de precipitación por cada mes del periodo base.
⑨ Se filtra por el mes indicado
⑩ Obtener la media de los 30 años
⑪ Marcar como propiedad el año del mes
⑫ De una coleccción de 12 imágenes, se crea una imagen de 12 bandas
⑬ Renombramiento de las bandas (número del mes)

5.2 Anomalía en milimetros

Es la diferencia en milimetros, de la precipitación de un determinado mes $(\overline{x}_i)$ y el promedio histórico o la normal (μ_{normal}) de ese mes

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

```
img_monthly_anomaly_mm = ee.Image((img_monthly_pr
  .subtract(base_pr_monthly_accumulation)                          ①
  .copyProperties(img_monthly_pr, img_monthly_pr.propertyNames())))  ②
```

① Restar el promedio histórico
② Copiar todas las propiedades en la nueva imagen

5.3 Anomalía en porcentaje

Es el resultado de dividir la diferencia de la precipitación de un determinado mes $(\overline{x}_i)$ y el promedio histórico o la normal (μ_{normal}) entre la normal de ese mismo mes.

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

```
img_monthly_anomaly_prop = ee.Image((img_monthly_pr
  .subtract(base_pr_monthly_accumulation)                          ①
  .divide(base_pr_monthly_accumulation)                            ②
  .copyProperties(img_monthly_pr, img_monthly_pr.propertyNames())))  ③
```

① Restar el promedio histórico
② Dividir entre el promedio histórico
③ Copiar todas las propiedades en la nueva imagen

Capítulo 6

De raster a CSV

6.1 Información de `ee.Image` a `ee.FeatureCollection`

> Este apartado se hará la demostración con **`img_monthly_pr`** pero puede ser aplicado a cualquiera de las imágenes que se crearon

Para poder exportar la información como una tabla de CSV, primero se tiene que almacenar o reducir la información a las geometrias de las regiones del país (sean entidades, municipios o cualquier otro tipo de división).

```
img2fc_monthly_pr = (img_monthly_pr
  .reduceRegions(                                                    ①
      collection = fc,                                               ②
      reducer = ee.Reducer.mean(),                                   ③
      scale = 5566)                                                  ④
  .map(lambda feature: (ee.Feature(feature)                          ⑤
                        .set({'n_year': date_year_interes})
                        .setGeometry(None))))

fc_monthly_pr = ee.FeatureCollection(                                ⑥
  (img2fc_monthly_pr
   .toList(3000)                                                     ⑦
   .flatten()))                                                      ⑧
```

① Se crea una `ee.FeatureCollection` a partir de la información de la imagen de 12 bandas
② La información que se extraerá vendrá de las geometrías de México (sean entidades, municipios o cualquier otro tipo de división de interés)
③ Se extraerá el promedio de la región
④ La escala a la que se hará la reducción, debe ser la misma a la que se encuentra la imagen. Esta puede encontrarse en la página de información de la imagen o colección

⑤ Cuando se crea la nueva `ee.FeatureCollection`, se itera por cada `ee.Feature` para poder asignar la propiedad (columna) del año de la información. Las 12 bandas se transforman en tambien en columnas, entonces se tiene la información mensual. Finalmente se elimina la geometría asignada porque de esta manera la exporación es más fácil y no demora mucho.
⑥ Se crea una `ee.FeatureCollection` a partir de una lista de *features*
⑦ Transformar la `ee.Feature` a una lista de máximo 3000 elementos
⑧ Se eliminan sublistas (de existir).

6.2 Exportar `ee.FeatureCollection` a CSV

Dentro del editor de código de Earth Engine existe la función para exportar una tabla, pero para el caso de la API de Python se usa la librería de **`geemap`** a través de la función **`ee_export_vector_to_drive`**.

```
from geemap import ee_export_vector_to_drive

description_task = f"{select_fc}_monthly_pr_{date_year_interes}"

ee_export_vector_to_drive(
  collection= fc_monthly_pr,
  description= description_task,
  fileFormat= "CSV",
  folder= "pruebas_ee")
```

Capítulo 7

Código final

Tras explicar cada aspecto del procesamiento y extracción de los datos se concluye el documento con la función para pasar los datos raster de CHIRPS a un archivo CSV.

La función toma como argumentos:

1. El año de interés
2. La métrica de interes:

   - Precipitación → `pr`
   - Anomalía de precipitación en mm → `anomaly_pr_mm`
   - Anomalía de precipitación en porcentaje → `anomaly_pr_prop`

3. Tipo de `ee.FeatureCollection`:

   - Entidades → `ent`
   - Municipios → `mun`
   - Cuencas Hidrológicas → `ch`

```
import ee                                                                    ①
from geemap import ee_export_vector_to_drive

try:
    ee.Initialize()                                                          ②
    print("Se ha inicializado correctamente")
except:
    print("Error en la inicialización")
```

① Cargar librerías y funciones necesarias
② Inicializar sesion de Earth Engine

7.1 Funciones escenciales

```
def func_tag_month(img):                                                    ①
    full_date = ee.Date(ee.Number(img.get("system:time_start")))
    n_month = ee.Number(full_date.get("month"))
    return img.set({"n_month": n_month})

def func_tag_year_month_hist_pr(img):                                       ②
    full_date = ee.Date(ee.Number(img.get("system:time_start")))
    n_year = ee.Number(full_date.get("year"))
    n_month = ee.Number(full_date.get("month"))
    return img.set({"n_month": n_month, "n_year": n_year})
```

① Función para *taggear* únicamente el mes
② Función para *taggear* año y mes. Usada únicamente para la `ee.ImageCollection` que cubre la normal de 30 años (1981-2010)

7.2 Función de extracción, procesamiento y exportación de datos

```
def extract_from_chirps_daily(                                              ①
        year = 2024,
        metrica_interes = "pr",
        tipo_fc = 'ent'):

    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[tipo_fc])

    geom_mex = (ee.FeatureCollection("USDOS/LSIB/2017")
                .filter(ee.Filter.eq("COUNTRY_NA", "Mexico"))
                .first()
                .geometry())

    chirps = (ee.ImageCollection('UCSB-CHG/CHIRPS/DAILY')                   ③
              .select("precipitation")
              .filter(ee.Filter.bounds(geom_mex)))

```

```
    chirps_year = (chirps.filter( # 4>
        ee.Filter.calendarRange(start = year, field = "year")))                    ④

    chirps_year_tagged = chirps_year.map(func_tag_month)                           ⑤

    list_months = ee.List.sequence(1, 12)

    def func_reduce2months(n_month):
        return (chirps_year_tagged
                .filter(ee.Filter.eq("n_month", n_month))
                .sum()
                .set({"n_month": n_month}))

    list_month_pr = (list_months.map(func_reduce2months))                          ⑥

    imgcoll_month_pr = ee.ImageCollection.fromImages(list_month_pr)

    if year > 2023:                                                                ⑦
        img_bands = ["01", "02", "03"]
    else:
        img_bands = [f"0{i}" if i < 10 else str(i) for i in range(1,13)]

    img_month_pr = imgcoll_month_pr.toBands().rename(img_bands)                    ⑧

    if metrica_interes != "pr":                                                    ⑨
        hist_pr = (chirps
            .filter(ee.Filter.calendarRange(1981, 2010, field = "year")))

        hist_pr_tagged = hist_pr.map(func_tag_year_month_hist_pr)

        def func_reduce2yearmonths_hist_pr(n_year):
            imgcoll_interes = (hist_pr_tagged
                               .filter(ee.Filter.eq("n_year", n_year)))
            def func_reduce2months_hist_pr(n_month):
                return (imgcoll_interes
                        .filter(ee.Filter.eq("n_month", n_month))
                        .sum()
                        .set({"n_year": n_year, "n_month": n_month}))
            list_month_pr_per_year = (list_months
                                      .map(func_reduce2months_hist_pr))
            return list_month_pr_per_year

        hist_pr_tagged_reduced_year_month = (ee.ImageCollection
            .fromImages((ee.List.sequence(1981, 2010)
```

```
                        .map(func_reduce2yearmonths_hist_pr)
                        .flatten())))

    img_hist_pr = (ee.ImageCollection.fromImages(
        list_months.map(lambda n_month: (
                        hist_pr_tagged_reduced_year_month
                        .filter(ee.Filter.eq("n_month", n_month))
                        .mean()
                        .set({"n_month": n_month}))))
        .toBands()
        .rename([f"0{i}" if i < 10 else str(i) for i in range(1,13)]))

    if metrica_interes == "anomaly_pr_mm":                                  (10)
        img_metrica_interes = ee.Image(
            (img_month_pr
            .subtract(img_hist_pr.select(img_bands))
            .copyProperties(img_hist_pr, img_hist_pr.propertyNames())))
    else:
        img_metrica_interes = ee.Image(                                     (11)
            (img_month_pr
            .subtract(img_hist_pr.select(img_bands))
            .divide(img_hist_pr.select(img_bands))
            .copyProperties(img_hist_pr, img_hist_pr.propertyNames())))
else:
    img_metrica_interes = img_month_pr                                      (12)

img2fc_metrica_interes = (img_metrica_interes                               (13)
    .reduceRegions(collection = fc,
                   reducer = ee.Reducer.mean(),
                   scale = 5566)
    .map(lambda feature: (ee.Feature(feature)
                          .set({'n_year': year})
                          .setGeometry(None))))

fc_metrica_interes = ee.FeatureCollection(
    img2fc_metrica_interes.toList(3000).flatten())

# Guardar este pedo
descr_task = f"chirps_daily_{metrica_interes}_{tipo_fc}_{year}"             (14)
folder_name = f"gee_chirps_daily_{metrica_interes}"

print(f"Va al servidor: '{descr_task}' y se gurda en {folder_name}")        (15)
ee_export_vector_to_drive(
    collection = fc_metrica_interes,
```

```
        description= descr_task,
        fileFormat= "CSV",
        folder= folder_name)
    return None
```

① La **precipitación** del **2024** en las **entidades** de México es lo que por *default* se extraerá
② Carga de geometrías y `ee.FeatureCollections`
③ Carga de CHIRPS Daily
④ Selección del año del cual se obtendrán las metricas
⑤ Etiquetado de los meses a la `ee.ImageCollection` de interés
⑥ Reducción de una `ee.ImageCollection` de $\approx$ 365 imágenes a una de (máximo) 12 imágenes.
⑦ Si el año de interés es menor o igual que el 2023, entonces se tiene información de todos los meses (12 bandas), de lo contrario son menos
⑧ Renombramiento de las bandas a el número de los meses del año
⑨ Si la métrica **no es la precipitación (`'pr'`)**, es decir es anomalía de la precipitación en porcentaje (`'anomaly_pr_prop'`) o en milimetros (`'anomaly_pr_mm`'), entonces se hace el cálculo del promedio histórico de la precipitación de cada uno de los meses (`img_hist_pr`)
⑩ Identificar si es anomalía de la precipitación en milimetros (`'anomaly_pr_mm`')
⑪ Si no es `'anomaly_pr_mm`' entonces se hace la división del promedio histórico para el cálculo de la anomalía de la precipitación en porcentaje (`'anomaly_pr_prop'`)
⑫ Si la métrica de interés **es la precipitación (`'pr'`)**, entonces no se hace el cálculo del promedio histórico.
⑬ Se crea `ee.FeatureCollection` de la `ee.Image`
⑭ Se crean las variables para exportar los datos
⑮ Se exportan los resultados

7.3 Extracción

Con el codigo creado en esta Sección lo único que queda por hacer es iterar o seleccionar el año, división política y tipo de métrica a extraer

```

for anio in range(1981, 2025):                                          ①
    extract_from_chirps_daily(year = anio,                              ②
                              metrica_interes= "pr",
                              tipo_fc= "ent")

    extract_from_chirps_daily(year = anio,                              ③
                              metrica_interes= "pr",
```

```
                              tipo_fc= "mun")

    extract_from_chirps_daily(year = anio,                                    ④
                              metrica_interes= "anomaly_pr_prop",
                              tipo_fc= "ent")

    extract_from_chirps_daily(year = anio,                                    ⑤
                              metrica_interes= "anomaly_pr_prop",
                              tipo_fc= "mun")
```

① Extraer información de todos los años disponibles
② Precipitación en los estados de México
③ Precipitación en los municipios de México
④ Anomalía de precipitación en porcentaje con respecto a la normal en los estados de México
⑤ Anomalía de precipitación en porcentaje con respecto a la normal en los municipios de México