

# Análisis de Geomorfometría Fluvial, Aplicado a la Cuenca Ozama, República Dominicana, Utilizando Tecnología Geoespacial de Código Abierto (Open Source)

**Edel Tejeda Nova** *Agrimensor y Estudiante de Lic. en Geografía Mención Representación Espacial, Universidad Autónoma de Santo Domingo (UASD)*

La geomorfología a través de una de sus ramas llamada geomorfología fluvial se encarga especialmente del estudio de la red y la cuenca con múltiples fines, desatando la prevención de inundaciones y el uso sustentable de la misma. La cuenca Ozama situada al centro-sur de la República Dominicana es idónea para la realización de distintos estudios morfométricos por tener zonas de vida, áreas protegidas y zonas de uso urbano. Los estudios de este tipo aplicados a esta cuenca son escasos o inexistentes, este trabajo aporta nuevos conocimientos sobre los parámetros morfométricos de la cuenca y su red de drenaje. Todo el flujo de trabajo se efectuó en un ambiente de software de código abierto lo que permitirá replicar este estudio sin ningún costo.

*Keywords:* geomorfología de cuenca, cuenca Ozama, cuenca hidrográfica

## 1 Introducción

La geomorfología como topografía analítica, se fundamenta en la deducción de los antecedentes de la superficie terrestre y trata de predecir posibles cambios en el futuro en su configuración (Pedraza Gilsanz, 1996). Una de las ramas de la geomorfología con mayor desarrollo en las últimas décadas, es la geomorfología fluvial, que se encarga del estudio de los sistemas fluviales, especialmente la red y la cuenca.

Una cuenca hidrográfica es una unidad morfológica que está delimitada de forma natural en donde sus aguas superficiales convergen hacia un lecho fluvial, y por medio de una red de cauces principales fluyen al mar (Gaspari, Rodríguez Vagaría, Senisterra, Delgado, & Besteiro, 2013).

Los estudios morfométricos de cuencas y redes de drenaje, aportan conocimiento sobre el comportamiento morfodinámico e hidrológico, contribuyendo así al diseño informado de medidas de prevención en escenarios de abundantes precipitaciones, así como a la planificación del uso sustentable de la misma (Domínguez et al., 2003). Por otra parte Arriaga-Cabrera et al. (2009) resalta que el estudio de una cuenca es un avance para las políticas de administración sustentable porque en la actualidad no existen estudios que explique el comportamiento hídrico y morfométrico que permitan determinar y predecir su dinámica.

La cuenca del río Ozama, situada al sur-centro del República Dominicana, es idónea para distintos estudios, dado que es una cuenca modelo por múltiples razones: 1) Es una de las principales en términos ecológicos, porque existen en ella tres zonas de vida y dos zonas de transición en donde interactúan diferentes tipos de flora y fauna; 2) Se han declarado varias áreas protegidas dentro de la cuenca; 3) Una parte importante de su superficie incluye terrenos destinados para uso urbano, pero con preencia también destacada de la agricultura y la ganadería (Medio-Ambiente, 2012). Si bien se trata de una cuenca importante para el país y para el entorno urbano, su escasa instrumentación impone retos añadidos al conocimiento de sus procesos específicos. Sin embargo, con las técnicas y la información de elevación disponibles actualmente, es posible profundizar

en el conocimiento de esta cuenca mediante estudios morfométricos aplicados a geomorfología fluvial utilizando herramientas de bajo costo.

En la República dominicana los estudios de geomorfología fluvial son escasos y dispersos. La cuenca del río Ozama no escapa a esta realidad, y sólo cuenta con estudios generales y, aunque existe información geológica, no existen análisis morfométricos de la misma. Este trabajo aporta nuevo conocimiento sobre la red de drenaje del Ozama (e.g. orden de red, análisis hortoniano, análisis de cursos más largos para determinar perfiles longitudinales e índices de concavidad), e igualmente estudia la delimitación de la cuenca y subcuenca a partir de algoritmos de dirección. En todo el flujo de trabajo, se empleó software de código abierto (*open source*), y se generaron productos mediante *script* reproducible, lo que permitirá replicar este estudio sin ningún costo.

Es una cuenca importante pero poco instrumentada, la exploración visual de la misma revela patrones que despiertan la curiosidad. Concretamente, la cuenca del río Ozama tiene una forma peculiar (e.g. de perímetro muy rectilíneo), y su red de drenaje presenta unas características bastante llamativas (e.g. cursos de largo desarrollo, perfiles de elevación muy variables).

En este sentido, interesa conocer cuáles son los controles que explican su morfología, por qué determinados sectores de la misma presentan evidencia preliminar de procesos de reorganización del drenaje y, en su caso, formular una o varias hipótesis que expliquen estos fenómenos. Comúnmente, la morfometría de cuenca deja señales inequívocas (e.g. hipsometría) con las cuales interpretar posibles controles litológicos y estructurales. La caracterización de la red de drenaje definida por el método de Horton resalta la existencia (si los hubiere) de patrones en el orden de red y en la forma de las redes de drenaje según sus órdenes. Si la cuenca Ozama llegara a presentar algún patrón, conoceremos de qué factores o elementos dependen.

Según Horton (1945) comúnmente la razón de bifurcación es constante para todos los órdenes de red de una cuenca. Uno de los objetivos es determinar si este fenómeno ocurre en nuestra cuenca de estudio y si difiere la razón de bifurcación calculada por medio de coeficientes de regresión, de la creada por el promedio de las razones de bifurcación de cada par de órdenes de red, tratando de explicar las grandes diferencias que puedan surgir.

La configuración espacial resultado de los análisis aplicados a los perfiles longitudinales y los índices de concavidad de los cursos más largos de una cuenca, muy a menudo arrojan que se relacionan con la litología, por lo que, el análisis aplicado a la cuenca Ozama persigue comprobar si su curso más largo se rige bajo este criterio y si existe alguna evidencia en la cuenca Ozama de uno o varios fenómenos de reorganización del drenaje.

## 2 Metodología

Para responder las preguntas de investigación, en este estudio se recopilaron datos de elevación, conocimiento de terreno y productos cartográficos, todos referidos a la cuenca del río Ozama y su entorno. Dichas fuentes fueron procesadas con algoritmos de morfometría fluvial dentro de un ambiente de programación estadística, para obtener las principales tendencias.

### 2.1 Área de Estudio

La cuenca del río Ozama abarca una superficie de 2,847.15 kilómetros cuadrados, se encuentra ubicada entre las coordenadas geográficas de 18°58'30.393" N y 18°23'40.846" N latitud norte y 70°16'5.369" W y 69°24'27.891" W longitud oeste (Medio-Ambiente, 2012). Por su tamaño, es la cuarta cuenca mas grande de la República Dominicana, y en cuanto a longitud de curso más largo

es la TERCERA (Gutiérrez, 2014). Según Medio-Ambiente (2012), el río nace en la Loma Palo Bonito (loma Siete Cabezas), la cual forma parte de la sierra de Yamasa. (Ver figura 1)

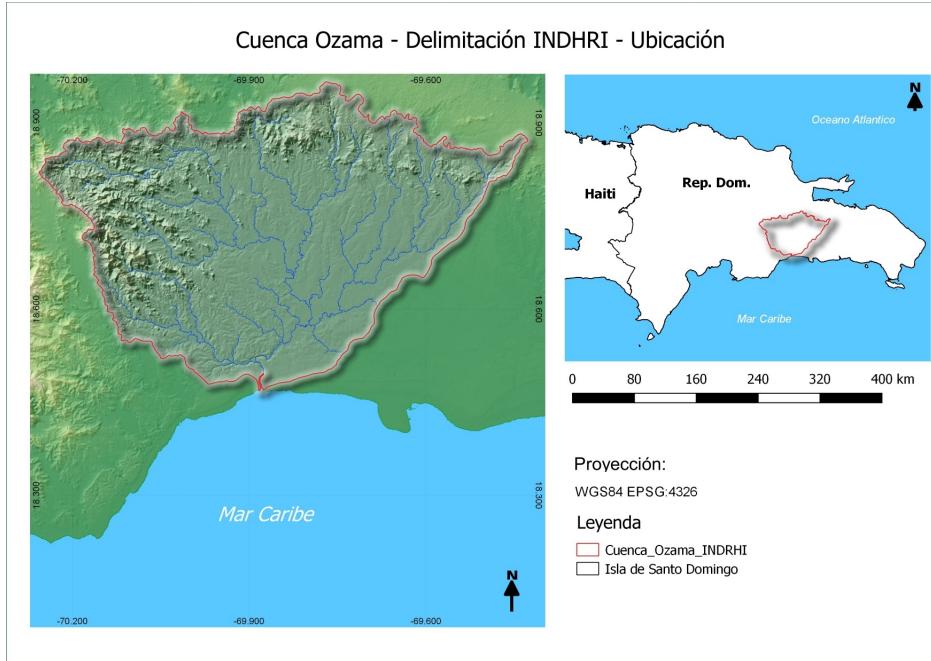


Figure 1: Ubicacion cuenca Ozama

Cubre el área geomórfica de la Llanura Costera del Caribe, incluyendo áreas de roca de tipo caliza de arrecifes costeros y depósitos aluviales y de origen lacustre marino, con una superficie de 1,719.65,km<sup>2</sup> (61.60%). La zona montañosa comprende la cordillera oriental, la región de la Sierra de Yamasa y la llanura montañosa de Los Haitises, con una superficie de 1,072.40,km<sup>2</sup> (38.40 %). El tipo geológico está compuesto por materiales sedimentarios, diversos arrecifes, grava, conglomerado (tipo Santo Domingo en La Romana), calizas grises tipo Hatillo, sedimentos aluviales de lagos oceánicos en cauces fluviales, terrazas planicies aluviales y valles (Medio-Ambiente, 2012).

## 2.2 Fuentes de datos y métodos

Para el estudio y análisis de la cuenca Ozama, se calcularon sus parámetros morfométricos, abarcando aspectos de los atributos de la red y de la cuenca. Todos los cálculos se efectuaron utilizando el flujo de trabajo de semiprocesamiento automático usando GRASS GIS en R (José Ramón Martínez Batlle, 2020), a través del paquete rgrass7 (Bivand, 2019; Martínez Batlle, 2020).

Se utilizó un modelo digital de elevación (DEM) espacial existente (SRTM3 v2.1 y AW3D-30m v1), mejorado con el algoritmo *Multi-Error Removed Improved Terrain* (MERIT), el cual eliminó y corrigió los errores de orientación absoluta, ruido de rayas, ruido de moteado y orientación de altura de árbol (Yamazaki, 2018). Este MERIT DEM fue descargado del sitio web de Dai YAMAZAKI's (<http://hydro.iis.u-tokyo.ac.jp/~yamadai>). También se utilizaron multiples complementos de GRASS GIS para calcular los parámetros morfométricos, entre los que destacan *r.watershed*, *r.stream\** and *r.basin* (Martínez Batlle, 2020).

El flujo de trabajo consistió en crear región y localización de GRASS tomando como referencia la extensión y el sistema de coordenadas del DEM. Posteriormente, se generaron el límite de la

cuenca utilizando el complemento de GRASS `r.water.outlet`.

Para obtener los cursos más largos de la cuenca Ozama y los cursos más largo de sus cuencas tributarias así como sus perfiles de longitudinales, sus índices de concavidad y sus curvas hipsométricas se ejecutaron una serie de scripts, que forman parte de la guía creada por Martínez Batlle (2020).

### 3 Resultados

Una vez finalizada la ejecución de los algoritmos de medición de morfología fluvial, se obtuvieron una serie de informaciones que ayudarán a dar respuesta a las preguntas de investigaciones y a comprender los detalles de los parámetros geomorfológicos de la Cuenca de Ozama.

#### 3.1 Delimitación y Forma

La cuenca extraída en este estudio alcanzó ca.  $3,400 \text{ km}^2$  y un perímetro de 420,km. Estas cifras contrastan con las calculadas a partir del polígono de delimitación del INDRHI, cuyos valores de área y perímetro fueron de  $2,740 \text{ km}^2$  y 300,km, respectivamente. En cuanto a forma, la cuenca extraída en este estudio se extiende hacia el este sobrepasando, por mucho, los límites de la cuenca del INDRHI (Ver figura 3 y 2).



Figure 2: Delimitacion Cuenca Rio Ozama (INDRHI)

#### 3.2 Datos de Elevación y Pendiente

La cuenca del río Ozama alcanza como elevación maxima aprox. 900 metros sobre el nivel del mar, con promedio de 110 metros, mediana en 60 metros y un valor mínimo de 0 metros con fuerte sesgo a la derecha (Ver figura 4). Esto implica que la mayor parte de las elevaciones de la cuenca del río ozama son bajas.

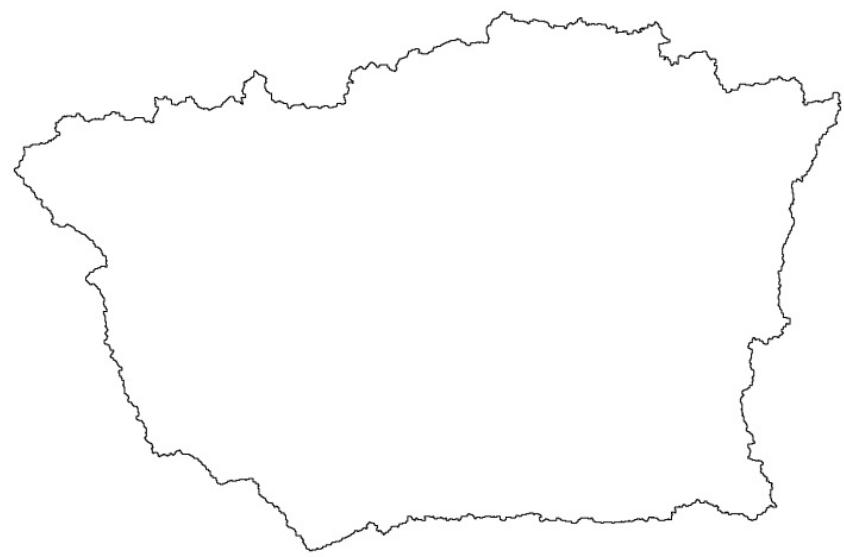


Figure 3: Delimitacion Cuenca Rio Ozama

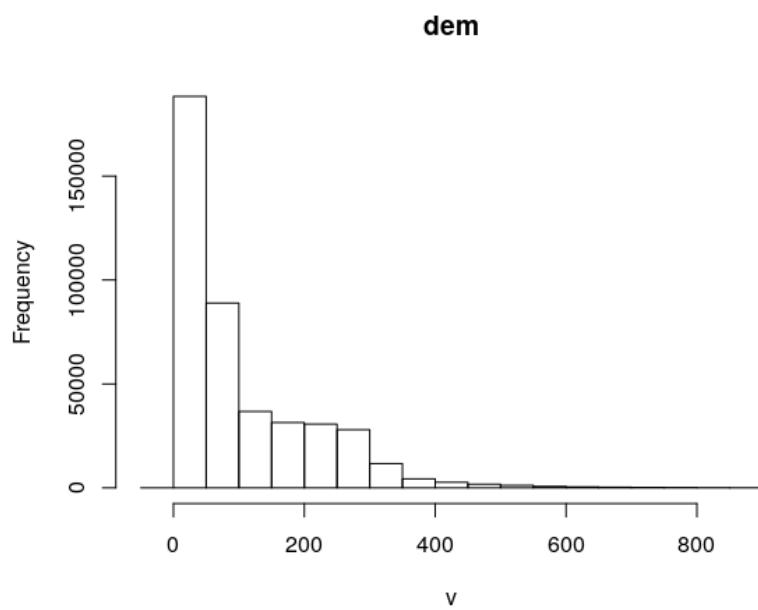


Figure 4: Histograma de elevaciones

La pendiente máxima de la cuenca del río Ozama es de aproximadamente 45 grados, la pendiente promedio es de 37 grados, la mediana es de 20 grados y la mínima es de 0 grados (Ver figura 5). La cuenca ozama cuenta con pendientes que desde el punto de vista agropecuario se corresponden a los mejores suelos para su aprovechamiento mientras que los procesos erosivos y los movimientos de masas son menores.

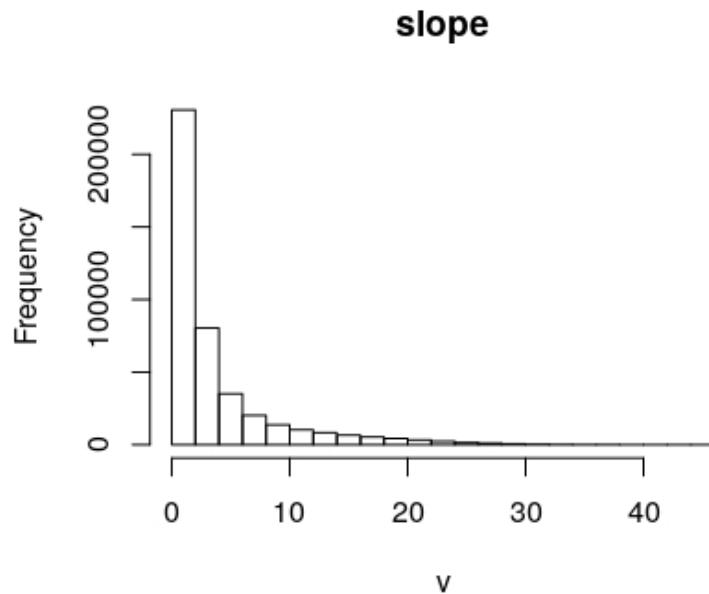


Figure 5: Histograma de pendiente

### 3.3 Red de Drenaje, Orden de red y análisis hortoniano

Partiendo de su cauce principal se generó la red de drenaje de la cuenca del río Ozama (Ver figura 6), la cual tiene un orden de red mínimo de 1 y un máximo de 7, según el método de Horton aplicado para determinar su grado de ramificación (Horton, 1945) (Ver figura 7).

#### 3.3.1 Razón de bifurcación promedio para cada par de ordenes de red

Aplicando el add-on de GRASS GIS r.stream.stats se obtuvieron las estadísticas por orden de red, esta muestra que la cuenca del río Ozama tiene 1,328 cursos fluviales de orden 1, 274 de orden 2, 63 de orden 3, 13 de orden 4, 5 de orden 5, 2 de orden 6 y 1 de orden 7. La razón de bifurcación para el par de ordenes 1-2 es  $1,328/274=4.846$ , para el par 2-3 es  $274/63=4.349$ , para el par 3-4 es  $63/13=4.846$ , para el par 4-5 es  $13/5=2.600$ , para el par 5-6 es  $5/2=2.500$ , para el par 6-7 es  $2/1=1$ . El valor promedio sería  $R_b=3.523679$  (Ver Tabla 1).

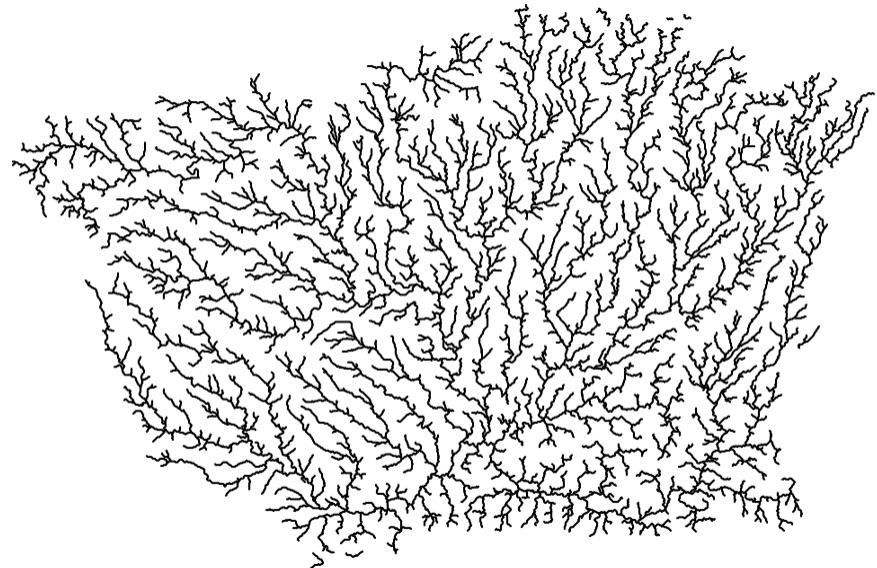


Figure 6: Red de drenaje de la cuenca Ozama

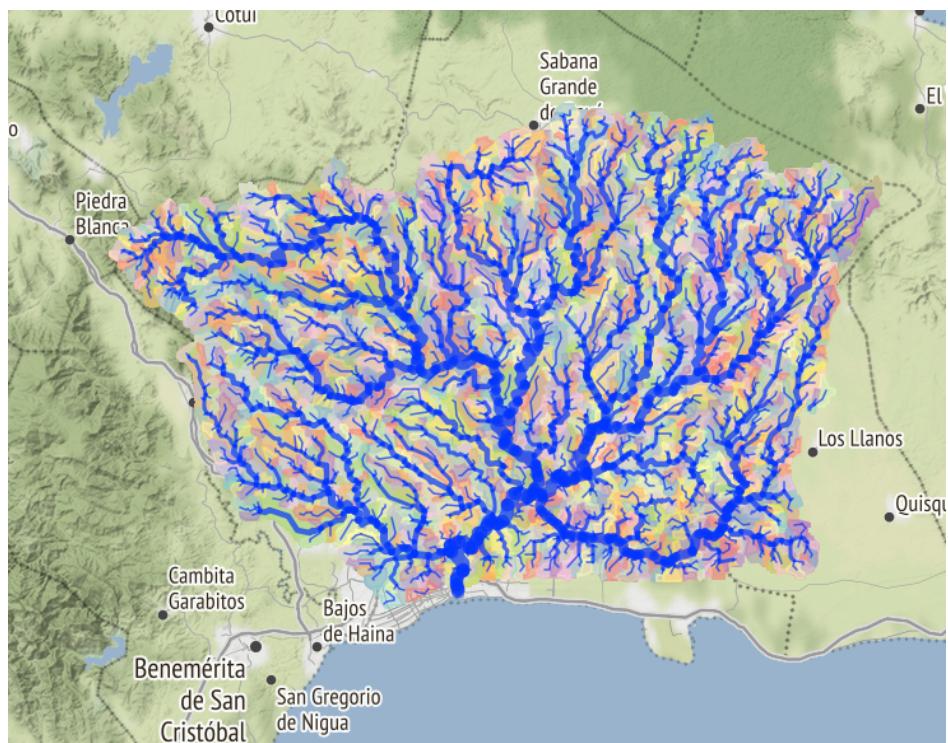


Figure 7: Cuenca Ozama y su red de drenaje

Table 1: Ordenes de red con sus respectivos cursos fluviales

Orden	No. de Cursos
1	1,328
2	274
3	63
4	13
5	5
6	2
7	1

### 3.3.2 Razón de bifurcación calculada por medio de coeficientes de regresión

Ya obtenido el numero de cursos fluviales y el modelo generado con dichos datos se creo la pendiente de la recta para calcular la razon de bifurcación, obteniendo como resultado  $R_b=3.361632$  (Ver figura 8).

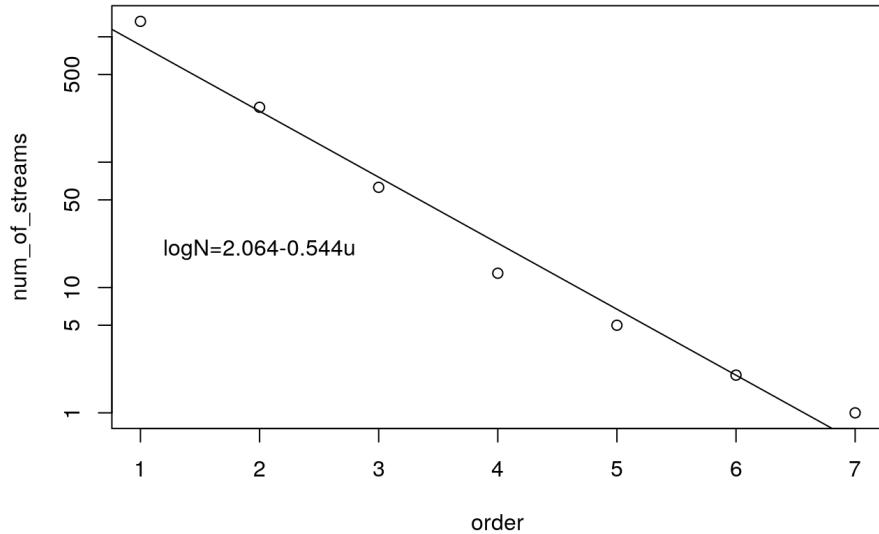


Figure 8: Recta de regresión para el modelo “numero de curso fluviales en función del orden de red” y ecuación correspondiente

### 3.4 Perfiles Longitudinales e Índices de Concavidad

La complejidad de la Cuenca Ozama obligó a que este análisis se dividiera en seis partes, estas se corresponden a subcuentas que se definieron para abarcar todos los perfiles longitudinales con sus valores de concavidad.

### 3.4.1 Subcuenta Principal Rio Ozama

Los perfiles longitudinales de los cursos fluviales que drenan la subcuenca principal Rio Ozama, presentan características similares, tales como en el rango de valores de concavidad. Un detallado análisis permite observar 5 grupos de tendencias, en donde el grupo uno (1) presenta errores en desembocadura y cabecera que pudieran ser ocasionados por DEM, el grupo dos (2) muestra meseta en cabecera, provocando índices de concavidad negativos (convexos), el tercer grupo (3) es formado por los que tienen concavidad perfecta, el cuarto (4) grupo se caracteriza por cambios que pudieran ocurrir en la litología, formando los llamados sifones, por último se observan perfiles con índices de concavidad cercanos a cero, estos forman el grupo número cinco (5) y abarcan los perfiles rectilíneos (Ver figura 9, 10 y 11). Los perfiles longitudinales cóncavos son muy pronunciados y los más frecuentes en la subcuenca principal Rio Ozama.



Figure 9: Subcuenta principal Rio Ozama

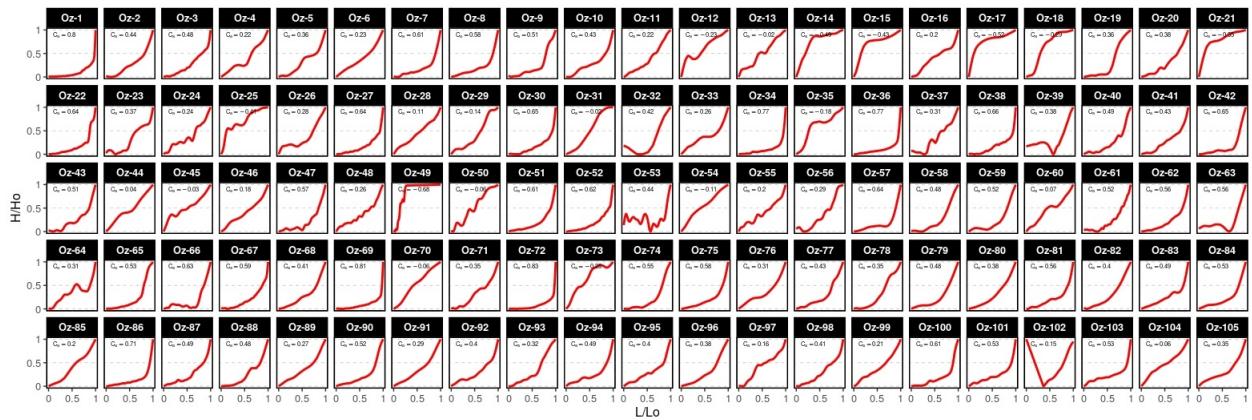


Figure 10: Perfiles Longitudinales e Indices de Concavidad de la subcuenta principal Rio Ozama

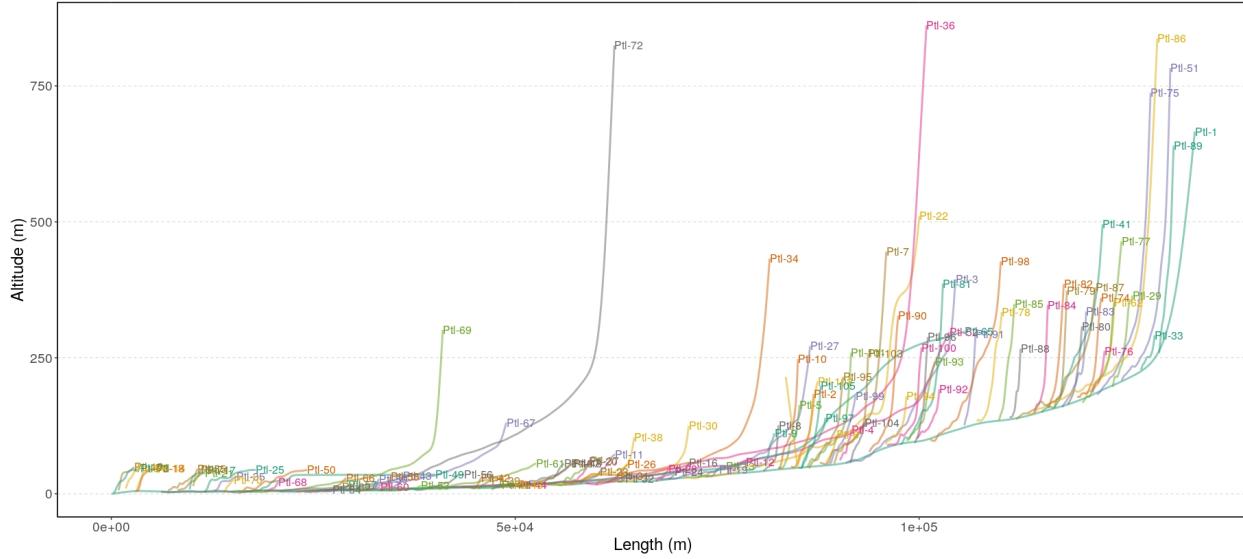


Figure 11: Todos los Perfiles de elevación de la subcuenta principal Rio Ozama

### 3.4.2 Subcuenca Rio Yabacao

En la subcuenca Rio Yacao al igual que en la subcuenca principal del Rio Ozama, los patrones de perfiles rectilíneos, concavidad perfecta, perfiles convexos y los perfiles de sifones están presentes, predominado de entre ellos los perfiles cóncavos.(Ver figura 12, 13 y 14).



Figure 12: Subcuenta Rio Yabacao

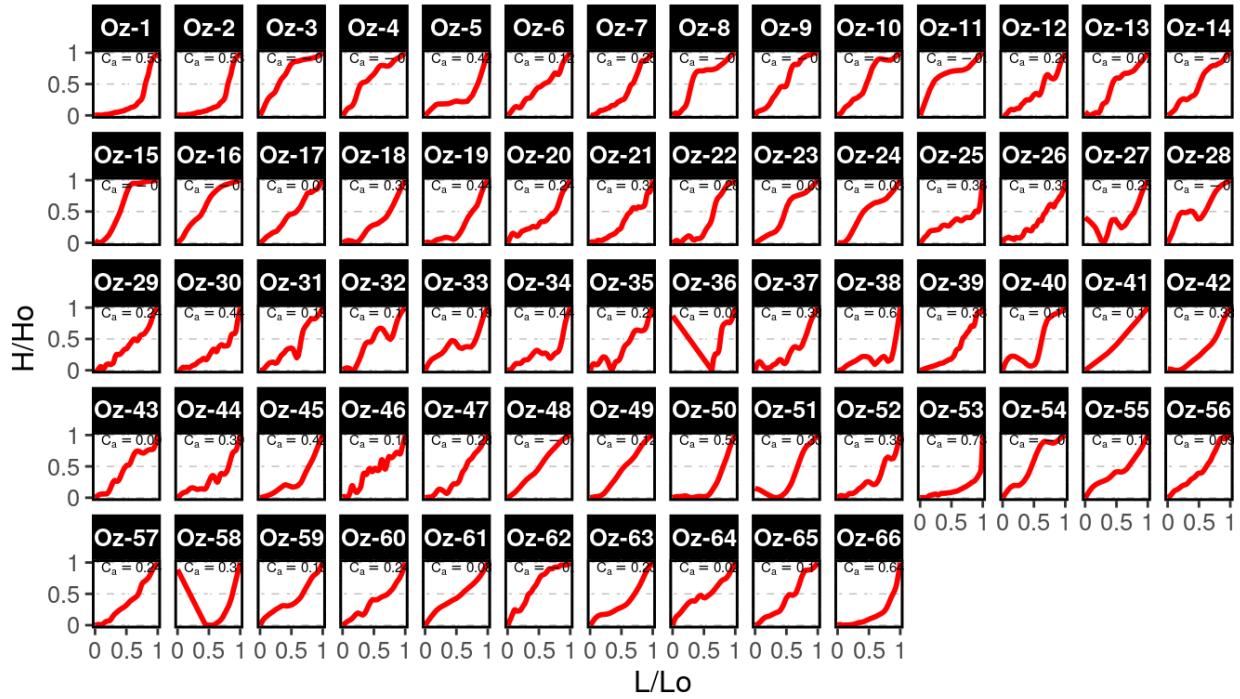


Figure 13: Perfiles Longitudinales e Indices de Concavidad de la subcuenta Rio Yabacao

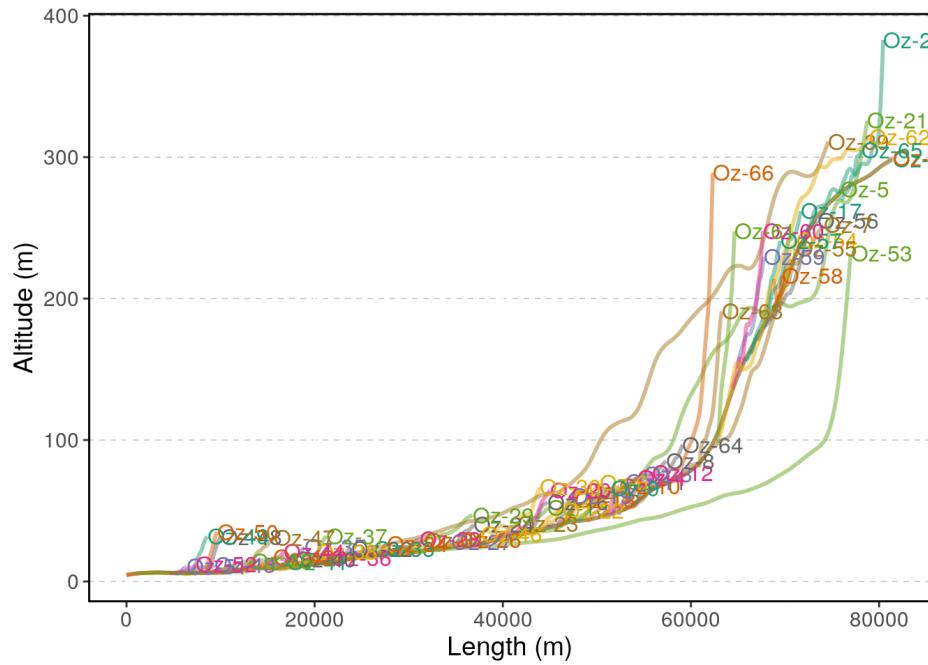


Figure 14: Todos los Perfiles de elevación de la subcuenta principal Rio Yabacao

### 3.4.3 Subcuenca Arroyo Ocoa

Los perfiles LFP que drenan el Arroyo Ocoa muestran un patrón rectilíneo predominante. Presentan claramente tres patrones, uno formado por perfiles convexos con meseta en cabecera, perfiles rectilíneos y una gran cantidad de sifones.(Ver figura 15, 16 y 17).

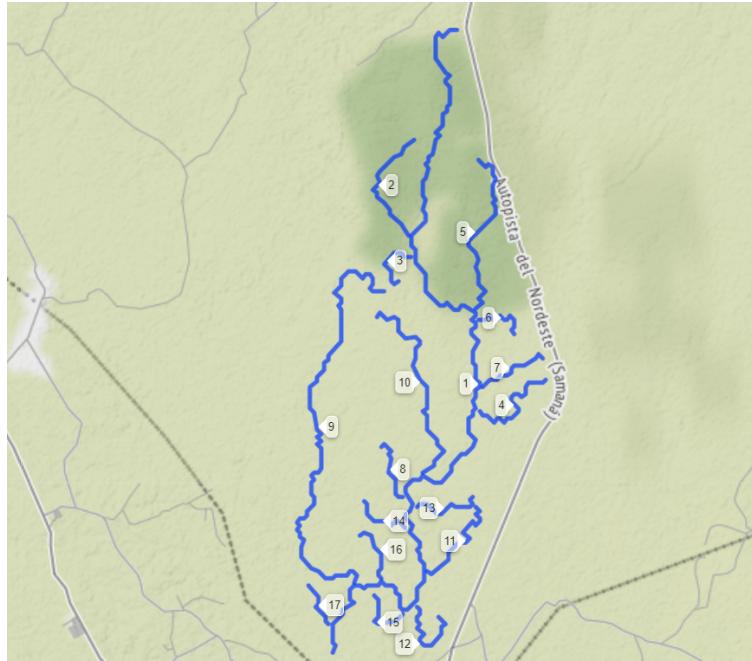


Figure 15: Subcuenta Arroyo Ocoa

### 3.4.4 Subcuenca Rio La Savita

Con altura máxima de aproximadamente 500 metros, la subcuenca Rio La Savita, posee perfiles longitudinales cóncavos predominantes, así como perfiles de concavidad perfecta y en cantidades muy mínimas presenta sifones y mesetas en cabeceras. (Ver figura 18, 19 y 20).

### 3.4.5 Subcuenca Arroyo Yuca

Los perfiles longitudinales que drenan el Arroyo Yuca presentan dos grupos de patrones, el cóncavo y el rectilíneo. Destacándose más los perfiles cóncavos.(Ver figura 21, 22 y 23).

### 3.4.6 Subcuenca Rio Isabela

Los perfiles cóncavos están muy presentes en la subcuenca del Rio Isabela, evidencia de esto, es la presencia de una gran cantidad de perfiles cóncavos perfectos, sin embargo también presentan un grupo de valores de índices de concavidad negativos dando origen a mesetas en cabeceras poco pronunciadas.(Ver figura 24, 25 y 26).

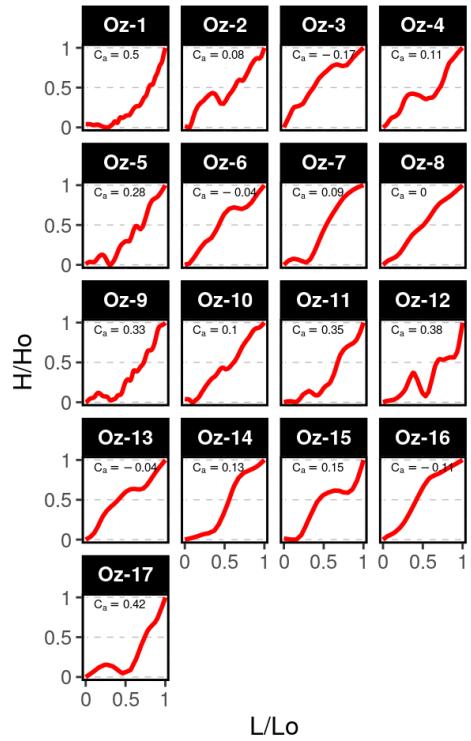


Figure 16: Perfiles Longitudinales e Indices de Concavidad de la subcuenta Arroyo Ocoa

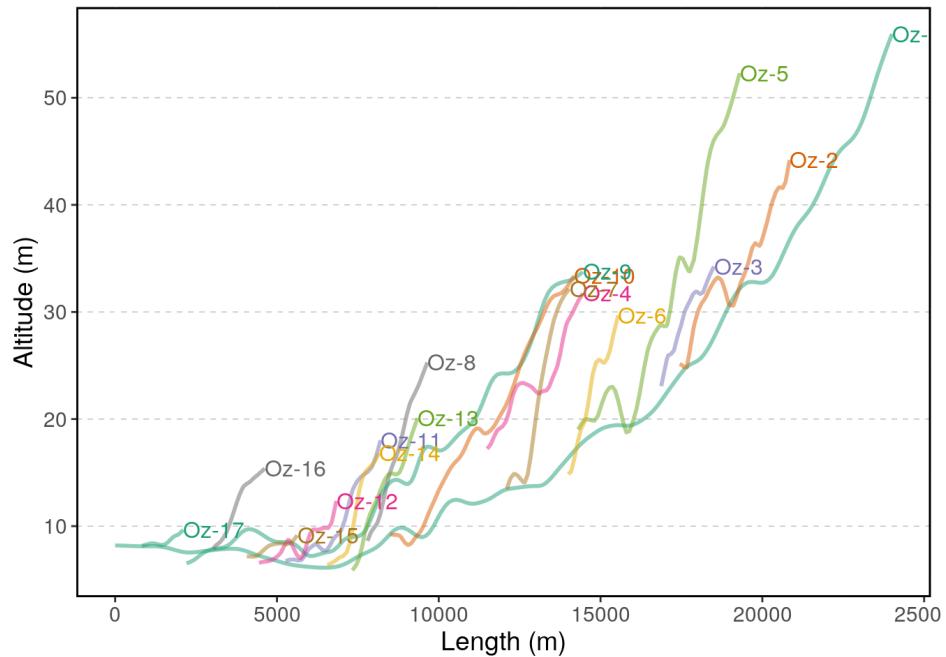


Figure 17: Todos los Perfiles de elevación de la subcuenca Arroyo Ocoa

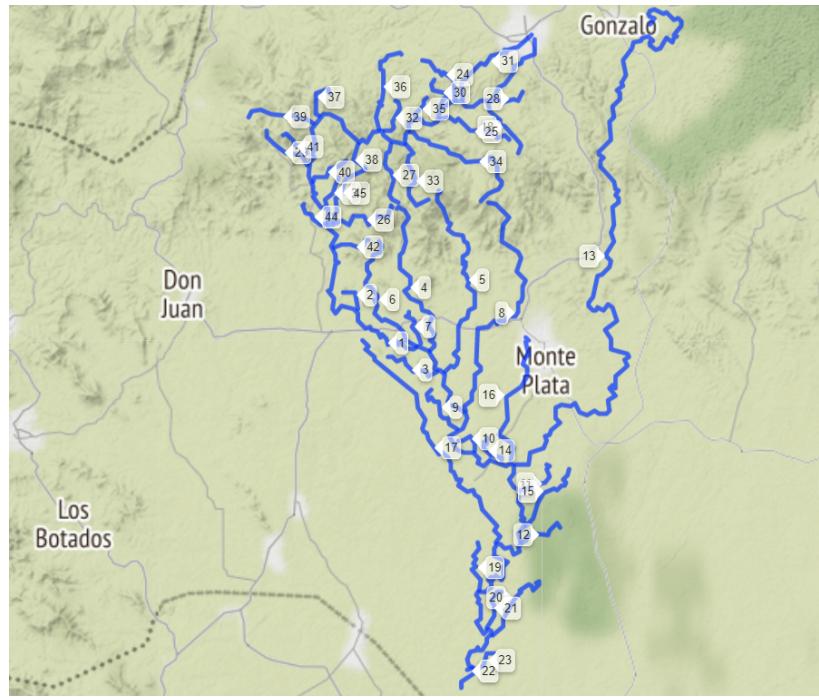


Figure 18: Subcuenta Rio La Savita

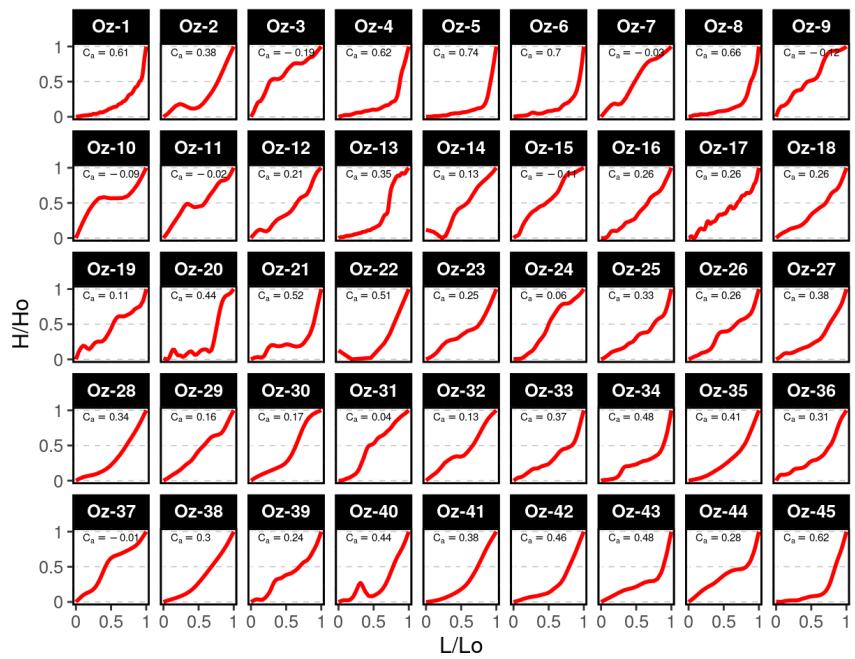


Figure 19: Perfiles Longitudinales e Indices de Concavidad de la subcuenta Rio La Savita

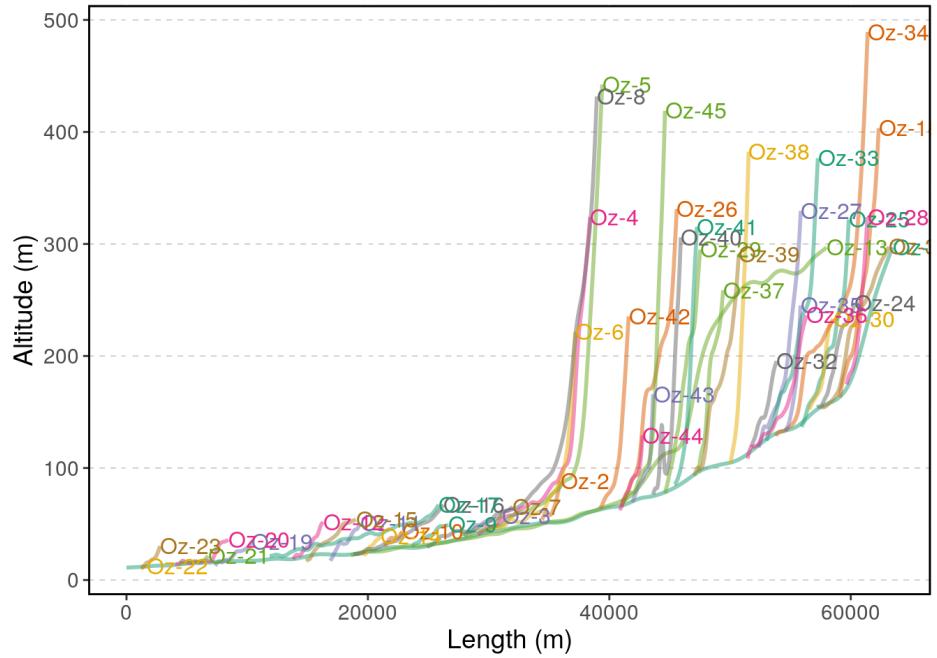


Figure 20: Todos los Perfiles de elevación de la subcuenta principal Rio Savita

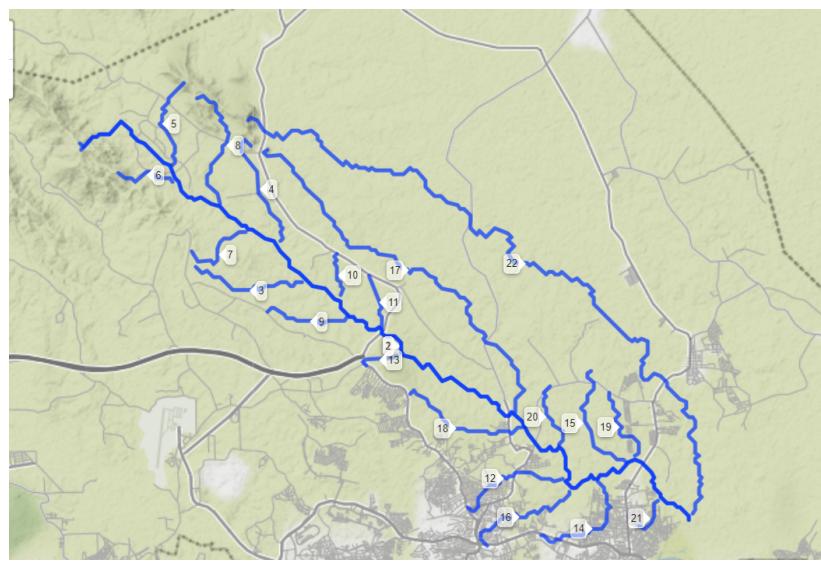


Figure 21: Subcuenta Arroyo Yuca

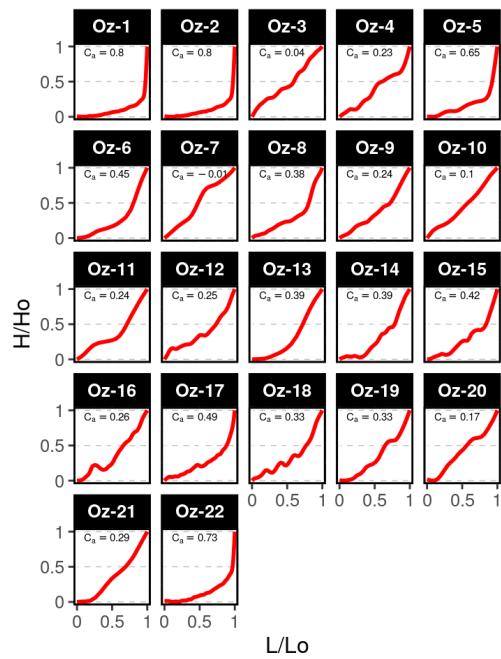


Figure 22: Perfiles Longitudinales e Indices de Concavidad de la subcuenta Arroyo Yuca

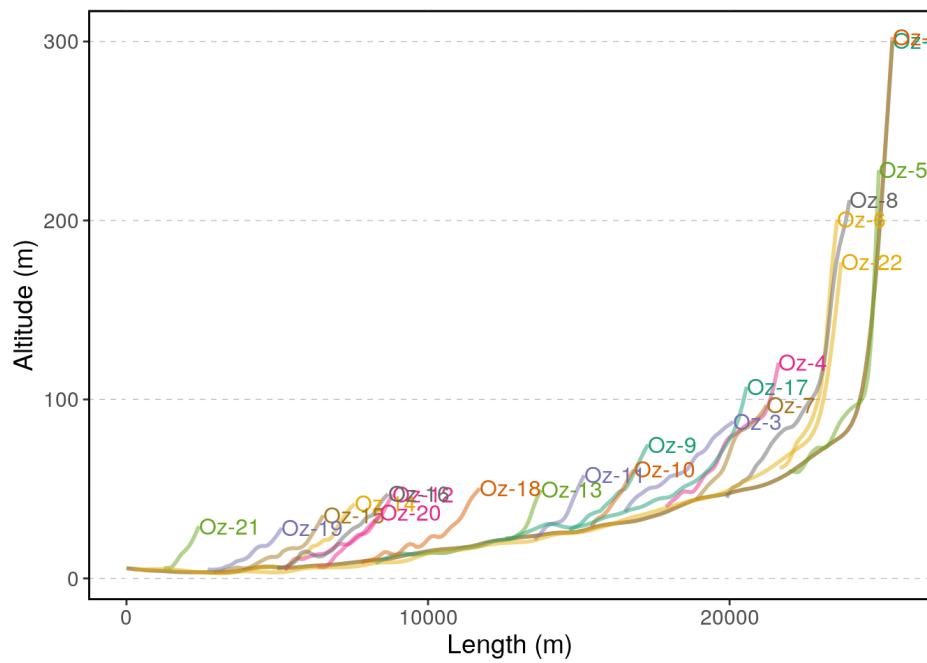


Figure 23: Todos los Perfiles de elevación de la subcuenta Arroyo Yuca



Figure 24: Subcuenta Rio Isabela

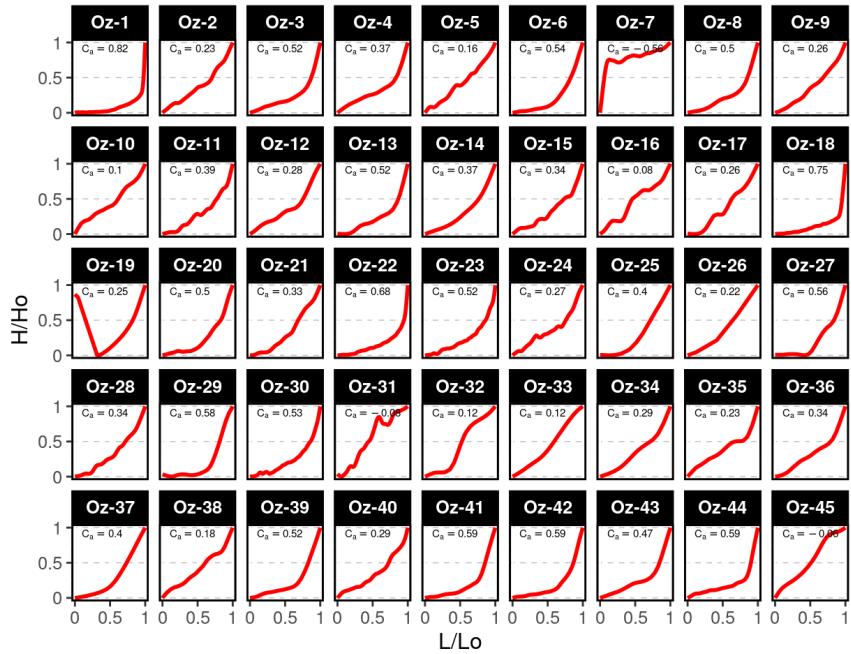


Figure 25: Perfiles Longitudinales e Indices de Concavidad de la subcuenta Rio Isabela

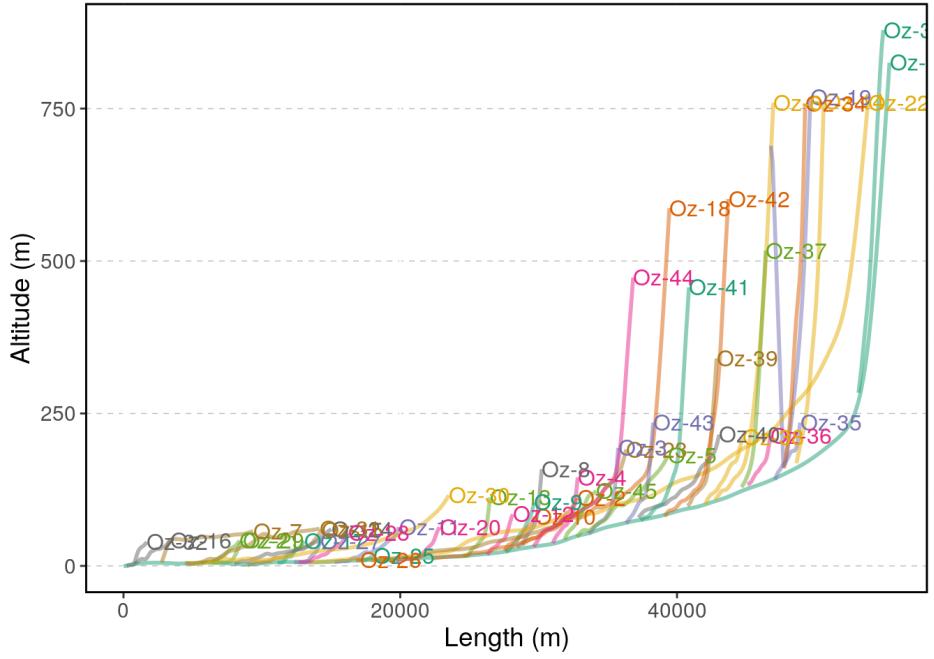


Figure 26: Todos los Perfiles de elevación de la subcuenta Rio Isabela

### 3.5 *Curso mas largo de la cuenca*

Loma Rancho de Yagua-Loma Palo Bonito, allí nace el Ozama, o al menos allí se enraíza su curso más largo, desembocando en el Mar Caribe (Ver figura 27).

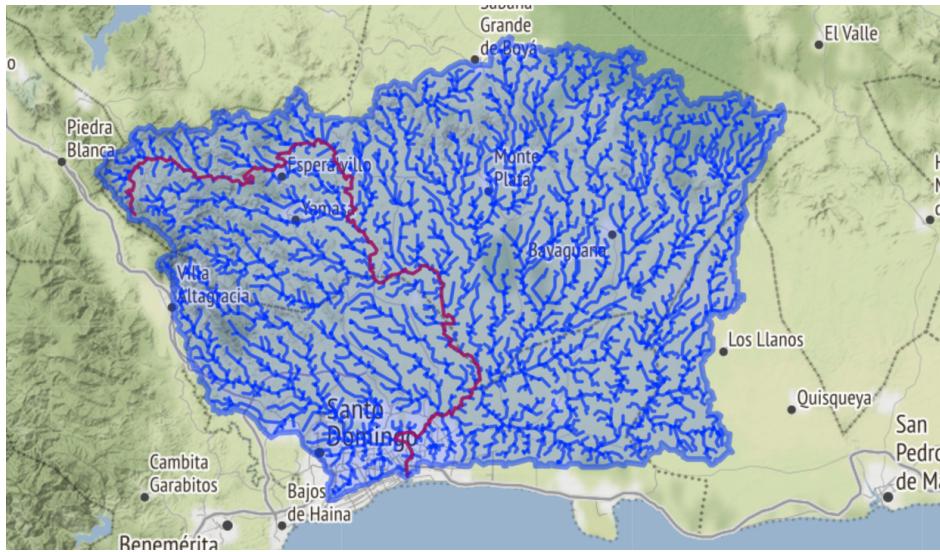


Figure 27: Curso mas largo de la cuenca Ozama

#### 4 Discusión

A partir de los hallazgos encontrados, se observa de manera visual dos posibles codos de captura, a) Uno ubicada al este de Piedra y al norte de Esperalvillo, existiendo probabilidad de que el curso fluvial más largo del Ozama drenara sus aguas hacia los Haitises; b) el segundo es el En curso fluvial que sale de Sabana Grande de Boya hacia el Rio Ozama, pudiendo este en el pasado drenar sus aguas en el Rio Camú, o en algún otro rio del noreste de la Isla de Santo Domingo.

Al superponer la Cuenca Ozama generada con el Mapa Geológico Nacional, en las zonas donde están estos posibles codos de capturas, predominan las rocas clásticas y las rocas calizas, por lo que, se plantea la posible existencia de piratería kárstica en estas zonas.

Esta hipótesis se relaciona con lo planteado por Jose Ramon Martinez Batlle (2019), quien señala que la piratería kárstica es un motor potencial del reordenamiento del drenaje, y que ocurre en zonas de afloramiento de piedras calizas. De igual manera también se relaciona con lo establecido por Gutiérrez Elorza (2008) , quien explica que si la roca se erosiona fácilmente, como margas o arcilla, se produce una rápida erosión que puede provocar la división de un río, para finalmente producir una captura por otro río.

La caracterización morfológica de la red de drenaje de la Cuenca Ozama, presenta cuatro patrones de redes de drenaje bien definidos (Paralela, Rectangular, Dendrítica, Enrejada), esto según Gutiérrez Elorza (2008) es debido a la interacción fluvial con los materiales erosionables, argumento que pudimos comprobar, al observar (en el mapa Geológico Nacional) que la cuenca drena sus aguas a través de una diversidad de rocas.

Si el clima y la litología permanecen sin cambios, la razón de bifurcación ( $R_b$ ) explicada por el resultado de la relación entre el número de cauces de cualquier orden y el número de cauces en el orden inferior, dicho resultado siempre será constante entre un orden y otro (Horton, 1945). Las razones de bifurcación calculada para los pares de órdenes de red 4-5, y 5-6, de la Cuenca Ozama no se ajustan al criterio planteado por Horton, debido a que estas varían con valores que van desde 1 a 2. La guía elaborada por Martínez Batlle (2020) explica que la razón de bifurcación calculada por los métodos de coeficientes de regresión y por el promedio de las razones de bifurcación de cada par de órdenes de red, difieren en sus resultados, esto fue comprobado para la razón de

bifurcación calculada a la Cuenca Ozama por ambos métodos, sin embargo el error resultante es un valor muy pequeño.

## 5 Script reproducible

### 5.1 Paquetes

```
library(rgrass7)
```

### 5.2 Región de GRASS

```
gisdbase <- 'grass-data-test' #Base de datos de GRASS GIS
wd <- getwd() #Directorio de trabajo
wd
loc <- initGRASS(gisBase = "/usr/lib/grass78/",
                  home = wd,
                  gisDbase = paste(wd, gisdbase, sep = '/'),
                  location = 'rdom',
                  mapset = "PERMANENT",
                  override = TRUE)
```

```
knitr::opts_chunk$set(
  echo = TRUE,
  collapse=TRUE,
  eval = T
)
```

```
source(
  knitr:::purl(
    'intro-rgrass.Rmd',
    output=tempfile()
  )
)
```

### 5.3 Proceso 0

```
#Quité el paquete rgrass7, porque ya se carga al ejecutar el script intro-rgrass.Rmd
library(sf)
library(raster)
library(sp)
```

#### 5.4 Definir proyección basado en una fuente externa, en este caso, el DEM MERIT

```
gmeta()
dem <- 'data/dem.tif'
#Definir la proyección de la región basada en DEM
execGRASS(
  cmd = 'g.proj',
  flags = c('t','c'),
  georef=dem)
gmeta()
```

#### 5.5 Importar mapa raster

```
execGRASS(
  cmd = 'r.in.gdal',
  flags=c('overwrite','quiet'),
  parameters=list(
    input=dem,
    output='dem'
  )
)
```

#### 5.6 Actualizar la extensión de la región al DEM, sólo por precaución

```
execGRASS(
  cmd = 'g.region',
  parameters=list(
    raster = 'dem',
    align = 'dem'
  )
)
```

#### 5.7 Mostrar la definición de la región

```
gmeta()
```

#### 5.8 Para completar, importar un mapa vectorial también

```

demext <- 'data/dem-extension.geojson'
execGRASS(
  cmd = 'v.in.ogr',
  flags=c('overwrite', 'quiet'),
  parameters=list(
    input=demext,
    output='dem_extent'
  )
)

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

source('borrar_mascara_si_la_hubiere.R')
unlink_.gislock()

```

## 6 title: “Calcular parámetros hidrográficos con r.watershed. Visualizar con leaflet”

```

knitr:::opts_chunk$set(
  echo = TRUE,
  collapse=TRUE,
  eval = T
)
options(knitr.duplicate.label = "allow")

```

### 6.1 ejecutar Scrit anteriores

```

source(
  knitr:::purl(
    'proyeccion-importar-fuente-extension_C_Propia.Rmd',
    output=tempfile()
  )
)
knitr:::opts_chunk$set(fig.path = "img/calcwshed/")

```

## 6.2 *g.list*

```
execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)
```

## 6.3 Calcular parámetros hidrográficos de interés usando *r.watershed*

```
execGRASS(
  "r.watershed",
  flags = c('overwrite', 'quiet'),
  parameters = list(
    elevation = "dem",
    accumulation = "accum-de-rwshed",
    stream = "stream-de-rwshed",
    drainage = "drainage-dir-de-rwshed",
    basin = 'basins',
    half_basin = 'half-basins',
    threshold = 80
  )
)
```

## 6.4 Traer capas a R

```
# Usar Spatial
library(sp)
use_sp()
#Paquete manejo de los raster
library(raster)
#DEM
dem <- raster(readRAST('dem'))
#Basins
basins <- raster(readRAST('basins'))
#Stream network
stream <- raster(readRAST('stream-de-rwshed'))
stream3857 <- projectRaster(stream, crs = CRS("+init=epsg:3857"), method = 'ngb')
#Generar un vectorial de extensión de capa en EPSG:4326
e <- extent(stream)
e <- as(e, 'SpatialPolygons')
```

```
proj4string(e) <- CRS("+init=epsg:32619")
e <- spTransform(e, CRSobj = CRS("+init=epsg:4326"))
```

## 6.5 Visualizar capas con leaflet

```
library(leaflet)
library(leafem)
r_wshed_salida <- leaflet() %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addRasterImage(dem, group='DEM', opacity = 0.5) %>%
  addRasterImage(
    ratify(basins),
    group='basins', opacity = 0.7,
    colors = sample(rep(RColorBrewer::brewer.pal(12, 'Set3'), 1000))) %>%
  addRasterImage(stream3857, project = F, group='str', opacity = 0.7, method = 'ngb', colors =
  addLayersControl(
    overlayGroups = c('terrain','DEM','basins','str'),
    options = layersControlOptions(collapsed=FALSE)) %>%
  addHomeButton(extent(e), 'Ver todo')
r_wshed_salida
#La siguiente linea toma una captura de este mapa (toma tiempo, paciencia)
r_wshed_salida %>% mapview::mapshot(file = 'r_wshed_salida.png')
```

## 6.6 Limpiar archivo de bloqueo del conjunto de mapas de GRASS. Quitar máscara, si la hubiere

```
source('borrar_mascara_si_la_hubiere.R')
unlink_.gislock()
```

### 6.6.1 Extraer una cuenca de drenaje con r.water.outlet. Visualizar con mapview y leaflet

```
knitr:::opts_chunk$set(
  echo = TRUE,
  collapse=TRUE,
  eval = T
)
options(knitr.duplicate.label = "allow")

source(
  knitr:::purl(
    'Calcular parámetros hidrográficos con r watershed Visualizar con leaflet cuen propia.Rmd',
    output=tempfile())
```

```

    )
)
knitr:::opts_chunk$set(fig.path = "img/basinoutlet/")

```

## 6.7 Obtener las coordenadas de la desembocadura de la cuenca de interés

```

library(mapview)
red_de_r_watershed <- mapview(
  stream3857, method='ngb', col.regions = 'blue',
  legend = FALSE, label = FALSE, maxpixels = 1801674
)#Los cursos fluviales no aparecen continuos, porque no imprime los rásters completamente
red_de_r_watershed
red_de_r_watershed %>% mapview::mapshot(file = 'red_de_r_wshed_salida.png')

```

## 6.8 Convertir las coordenadas lat/lon a EPSG:32619

```

my_trans <- function(coords = NULL) {
  require(sp)
  pt <- SpatialPoints(matrix(coords, ncol = 2), CRS("+init=epsg:4326"))
  foo <- spTransform(pt, CRSobj = CRS("+init=epsg:32619"))
  bar <- as.vector(coordinates(foo))
  return(bar)
}
ozama_out <- my_trans(coords = c(-69.88087, 18.47427))
ozama_out

```

## 6.9 Extraer la cuenca de interés

```

execGRASS(
  "r.water.outlet",
  flags = c('overwrite', 'quiet'),
  parameters = list(
    input = 'drainage-dir-de-rwshed',
    output = 'ozama-basin',
    coordinates = ozama_out
  )
)

```

## 6.10 Convertir la cuenca a vectorial en GRASS

```
execGRASS(  
  "r.to.vect",  
  flags = c('overwrite', 'quiet'),  
  parameters = list(  
    input = 'ozama-basin',  
    output = 'ozama_basin',  
    type = 'area'  
)  
)
```

## 6.11 Traer a R la cuenca del Cuenca Ozama

```
ozama_bas <- readVECT('ozama_basin')  
ozama_bas  
plot(ozama_bas)  
ozama_bas4326 <- spTransform(ozama_bas, CRSobj = CRS("+init=epsg:4326"))  
ozama_bas4326_leaf <- leaflet() %>%  
  addProviderTiles(providers$Stamen.Terrain) %>%  
  addRasterImage(stream, opacity = 0.7, method = 'ngb', colors = 'blue') %>%  
  addPolygons(data = ozama_bas4326) %>%  
  leafem::addHomeButton(extent(ozama_bas4326), 'Ver cuenca')  
ozama_bas4326_leaf  
ozama_bas4326_leaf %>% mapview::mapshot(file = 'ozama_bas4326_salida.png')
```

## 6.12 Limpiar archivo de bloqueo del conjunto de mapas de GRASS. Quitar máscara, si la hubiere

```
source('borrar_mascara_si_la_hubiere.R')  
unlink_.gislock()
```

### 6.12.1 “Extraer una red drenaje con r.stream.extract. Visualizar con leaflet”

```
knitr::opts_chunk$set(  
  echo = TRUE,  
  collapse=TRUE,  
  eval = T  
)  
options(knitr.duplicate.label = "allow")
```

```

source(
  knitr::purl(
    'crear-una-cuenca-con-r-water-outlet.Rmd',
    output=tempfile()
  )
)
knitr::opts_chunk$set(fig.path = "img/extractnet/")

```

### 6.13 Mostrar lista nuevamente

```

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

### 6.14 Usar la cuenca del Ozama como máscara

```

execGRASS(
  "r.mask",
  flags = c('verbose', 'overwrite', 'quiet'),
  parameters = list(
    vector = 'ozama_basin'
  )
)

```

### 6.15 Extraer la red de drenaje de la cuenca de interés

```

execGRASS(
  "r.stream.extract",
  flags = c('overwrite', 'quiet'),
  parameters = list(
    elevation = 'dem',
    threshold = 80,
    stream_raster = 'ozama-stream-de-rstr',
    stream_vector = 'ozama_stream_de_rstr'
  )
)

```

## 6.16 Traer a R la red de drenaje de la Cuenca Ozama

```
ozama_net <- readVECT('ozama_stream_de_rstr', ignore.stderr = T)
ozama_net
plot(ozama_net)
ozama_net4326 <- spTransform(ozama_net, CRSobj = CRS("+init=epsg:4326"))
ozama_net4326
ozama_centroid <- coordinates(rgeos::gCentroid(ozama_bas4326))
ozama_centroid
ozama_net_r <- raster(readRAST('ozama-stream-de-rstr'))
ozama_net_r
ozama_net_r3857 <- projectRaster(ozama_net_r, crs = CRS("+init=epsg:3857"), method = 'ngb')
ozama_net_r3857
red_de_r_stream <- leaflet() %>%
  setView(lng = ozama_centroid[1], lat = ozama_centroid[2], zoom = 10) %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addRasterImage(ozama_net_r3857, opacity = 0.7, method = 'ngb', colors = 'grey20', group = 'str_raster')
  addPolylines(data = ozama_net4326, weight = 3, opacity = 0.7, group = 'str_vect') %>%
  leafem::addHomeButton(extent(ozama_net4326), 'Ver todo') %>%
  addLayersControl(
    overlayGroups = c('terrain','str_vect','str_raster'),
    options = layersControlOptions(collapsed=FALSE))
red_de_r_stream
red_de_r_stream %>% mapview::mapshot(file = 'red_de_r_stream_salida.png')
```

## 6.17 Limpiar archivo de bloqueo del conjunto de mapas de GRASS

```
unlink_.gislock()
```

### 6.17.1 “Orden de red, morfometría y análisis hortoniano usando r.stream”

```
knitr::opts_chunk$set(
  echo = TRUE,
  collapse=TRUE,
  eval = T
)
options(knitr.duplicate.label = "allow")
```

```
6.18 #
```

```

source(
  knitr::purl(
    'extraer-red-de-drenaje-con-r-stream.Rmd',
    output=tempfile()
  )
)
knitr::opts_chunk$set(fig.path = "img/streamorder/")

```

### 6.19 Imprimir lista de mapas ráster y vectoriales dentro en la región/localización activa

- Nótese que los paquetes requeridos en esta sesión (`rgrass7`, `raster`, `leaflet`, `leafem`), fueron en el bloque anterior al ejecutarse el código contenido en el archivo `extraer-red-de-drenaje-con-r-stream.Rmd`. Igualmente, dicho bloque de código creó todos los objetos necesarios para realizar este tutorial.

```

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

### 6.20 Crear mapa de dirección de flujo a partir de `r.stream`

```

execGRASS(
  "r.stream.extract",
  flags = c('overwrite', 'quiet'),
  parameters = list(
    elevation = 'dem',
    threshold = 80,
    direction = 'drainage-dir-de-rstr'
  )
)

```

### 6.21 Crear mapas de órdenes de red

```

execGRASS(
  "r.stream.order",
  flags = c('overwrite', 'quiet'),
  parameters = list(
    stream_rast = 'ozama-stream-de-rstr',
    ...
  )
)

```

```

direction = 'drainage-dir-de-rstr',
elevation = 'dem',
accumulation = 'accum-de-rwshed',
stream_vect = 'order_all',
strahler = 'order-strahler',
horton = 'order-horton',
shreve = 'order-shreve',
hack = 'order-hack-gravelius',
topo = 'order-topology'
)
)
)

```

## 6.22 Mostrar lista nuevamente

```

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

## 6.23 Visualizar la red con leaflet

### 6.23.1 Simbología única

```

{r, results='hide', warning=FALSE, message=FALSE}
order <- readVECT('order_all')

order4326 <- spTransform(order, CRSobj = CRS("+init=epsg:4326"))
leaflet() %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addPolylines(
    data = order4326, weight = 3, opacity = 0.7, group = 'order',
    label = ~as.character(strahler),
    highlightOptions = highlightOptions(color = "white",
                                         weight = 5, bringToFront = F, opacity = 1),
    labelOptions = labelOptions(noHide = F,
                                style = list(
                                  "font-size" = "8px",
                                  "background" = "rgba(255, 255, 255, 0.5)",
                                  "background-clip" = "padding-box",
                                  "padding" = "1px")))) %>%
  leafem::addHomeButton(extent(order4326), 'Ver todo') %>

```

```

addLayersControl(
  overlayGroups = c('terrain', 'order'),
  options = layersControlOptions(collapsed=FALSE))

```

### 6.23.2 Simbología aplicando grosor según orden de red

```

orden_de_red <- leaflet() %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addPolylines(
    data = order4326, weight = order4326$strahler*1.5, opacity = 0.7, group = 'order',
    label = ~as.character(strahler),
    highlightOptions = highlightOptions(color = "white",
                                         weight = 5, bringToFront = F, opacity = 1),
    labelOptions = labelOptions(noHide = F)) %>%
  leafem::addHomeButton(extent(order4326), 'Ver todo') %>%
  addLayersControl(
    overlayGroups = c('terrain', 'order'),
    options = layersControlOptions(collapsed=FALSE))
orden_de_red
orden_de_red %>% mapview::mapshot(file = 'orden_de_red_salida.png')

```

## 6.24 Delimitar cuencas según orden de red de Strahler

### 6.24.1 Obtener órdenes de red mínimo y máximo

```

#Estadísticas para obtener los valores mínimo y máximo del orden de red de Strahler
rinfo.ordstra <- execGRASS(
  'r.info',
  flags = 'r',
  parameters = list(
    map = 'order-strahler'
  )
)
#Órdenes de red mínimo y máximo
minmaxord <- as.numeric(
  stringr::str_extract_all(
    attributes(rinfo.ordstra)$resOut,
    "[0-9]+"
  )
)
minmaxord

```

#### 6.24.2 Delimitar cuencas, convertirlas de ráster a vectorial

```
sapply(
  min(minmaxord):max(minmaxord),
  function(x){
    execGRASS(
      "r.stream.basins",
      flags = c('overwrite','c','quiet'),
      parameters = list(
        direction = 'drainage-dir-de-rstr',
        stream_rast = 'order-strahler',
        cats = as.character(x),
        basins = paste0('r-stream-basins-',x)
      )
    )
    execGRASS(
      "r.to.vect",
      flags=c('overwrite','quiet'),
      parameters = list(
        input = paste0('r-stream-basins-',x),
        output = paste0('r_stream_basins_',x),
        type = 'area'
      )
    )
  }
)
```

#### 6.24.3 Representar las cuencas con leaflet

```
{r, results='hide', warning=FALSE, message=FALSE}
sapply(
  min(minmaxord):max(minmaxord),
  function(x){
    assign(
      paste0('orden', x),
      spTransform(readVECT(paste0('r_stream_basins_',x),driver = 'SQLite'), CRSobj = CRS("+init=
      envir = .GlobalEnv)
    )
  }
)
```

```
paleta <- RColorBrewer::brewer.pal(12, 'Set3')
cuencas_y_orden_de_red <- leaflet() %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addPolygons(data = orden7, stroke = T, weight = 2,
              color = ~paleta, fillOpacity = 0.4, group = '07') %>%
  addPolygons(data = orden6, stroke = T, weight = 2,
```

```

        color = ~paleta, fillOpacity = 0.4, group = '06') %>%
addPolygons(data = orden5, stroke = T, weight = 2,
            color = ~paleta, fillOpacity = 0.4, group = '05') %>%
addPolygons(data = orden4, stroke = T, weight = 2,
            color = ~paleta, fillOpacity = 0.4, group = '04') %>%
addPolygons(data = orden3, stroke = T, weight = 2,
            color = ~paleta, fillOpacity = 0.4, group = '03') %>%
addPolygons(data = orden2, stroke = T, weight = 2,
            color = ~paleta, fillOpacity = 0.4, group = '02') %>%
addPolygons(data = orden1, stroke = T, weight = 2,
            color = ~paleta, fillOpacity = 0.4, group = '01') %>%
addPolylines(
  data = order4326, weight = order4326$strahler*1.5,
  opacity = 0.7, group = 'str_order') %>%
leafem::addHomeButton(extent(order4326), 'Ver todo') %>%
addLayersControl(
  overlayGroups = c('terrain','01','02','03','04','05','06','07','str_order'),
  options = layersControlOptions(collapsed=FALSE))
cuencas_y_orden_de_red
cuencas_y_orden_de_red %>% mapview::mapshot(file = 'cuencas_y_orden_de_red_salida.png')

```

## 6.25 Estadísticas de red resumidas por orden de red.

```

execGRASS(
  "r.stream.stats",
  flags = c('overwrite','quiet','o'),
  parameters = list(
    stream_rast = 'order-strahler',
    direction = 'drainage-dir-de-rstr',
    elevation = 'dem',
    output = 'ozama_stats.txt'
  )
)
file.show('ozama_stats.txt')
d <- read.csv("ozama_stats.txt", skip=1, header=TRUE)
d
plot(num_of_streams~order, data=d, log="y")
mod <- lm(log10(num_of_streams)~order, data=d)
abline(mod)
text(2, 20, 'logN=2.064-0.544u')
rb <- 1/10^mod$coefficients[[2]]
rb

```

## 6.26 Estadísticas de red ampliadas

```
execGRASS(  
  "r.stream.stats",  
  flags = c('overwrite', 'quiet'),  
  parameters = list(  
    stream_rast = 'order-strahler',  
    direction = 'drainage-dir-de-rstr',  
    elevation = 'dem',  
    output = 'ozama_stats_expanded.txt'  
  )  
)  
file.show('ozama_stats_expanded.txt')
```

## 6.27 Limpiar archivo de bloqueo del conjunto de mapas de GRASS

```
rbp <- mean(d$num_of_streams[-length(d$num_of_streams)]/d$num_of_streams[-1])  
rbp  
  
unlink_.gislock()
```

### 6.27.1 Calcular índices de concavidad y perfiles longitudinales de cursos fluviales

```
knitr::opts_chunk$set(  
  echo = TRUE,  
  collapse=TRUE,  
  eval = T  
)  
options(knitr.duplicate.label = "allow")  
  
source(  
  knitr::purl(  
    'orden-de-red.Rmd',  
    output=tempfile()  
  )  
)  
knitr::opts_chunk$set(fig.path = "img/profilesconcav/")
```

## 6.28 Imprimir lista de mapas ráster y vectoriales dentro en la región/localización activa

```

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

### 6.29 Obtener coordenada

```
mapview(order, col.regions = 'blue', legend = FALSE)
```

### 6.30 Obtener cursos más largos (cargar función propia)

```

{r, results='hide', warning=FALSE, message=FALSE}
devtools::source_url('https://raw.githubusercontent.com/geofis/rgrass/master/lfp_network.R') #Cargamos la función propia
LfpNetwork(
  xycoords = my_trans(c(-69.88117, 18.47503)),
  suffix = 'ozm',
  stream_vect = 'order_all',
  direction = 'drainage-dir-de-rstr'
)

```

### 6.31 Imprimir lista de mapas ráster y vectoriales

```

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

### 6.32 Representar con leaflet

```

{r, results='hide', warning=FALSE, message=FALSE}
lfp <- readVECT('LfpNetwork_lfp_all_final_ozm')

```

```

lfp4326 <- spTransform(lfp, CRSobj = CRS("+init=epsg:4326"))
lfp_con_id <- leaflet() %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addPolylines(
    data = lfp4326, weight = 3, opacity = 0.7, group = 'order',
    label = ~as.character(cat),
    highlightOptions = highlightOptions(color = "white",
                                         weight = 5, bringToFront = F, opacity = 1),
    labelOptions = labelOptions(noHide = T,
                                style = list(
                                  "font-size" = "8px",
                                  "background" = "rgba(255, 255, 255, 0.5)",
                                  "background-clip" = "padding-box",
                                  "padding" = "1px")))) %>%
  leafem::addHomeButton(extent(lfp4326), 'Ver todo') %>%
  leafem::addMouseCoordinates() ## Esto es para agregar coordenadas
lfp_con_id
lfp_con_id %>% mapview::mapshot(file = 'lfp_con_id_salida.png')

```

### 6.33 Exportar a KML

```

{r, results='hide', warning=FALSE, message=FALSE}
execGRASS(
  'v.out.ogr',
  flags = c('overwrite','quiet'),
  parameters = list(
    input = 'LfpNetwork_lfp_all_final_ozm',
    output = 'lfp_ozm_kml.kml',
    format = 'KML',
    dsco = 'NameField=cat'
  )
)

```

### 6.34 Obtención de perfiles longitudinales e índices de concavidad

```

{r, results='hide', warning=FALSE, message=FALSE}
source('lfp_profiles_concavity.R') #Cargado como función "LfpProfilesConcavity"
ozama_conv_prof <- LfpProfilesConcavity(
  xycoords = my_trans(c(-69.88117, 18.47503)),
  network = 'LfpNetwork_lfp_all_final_ozm',
  prefix = 'Oz',
  dem = 'dem',
  direction = 'drainage-dir-de-rstr',

```

```
crs = '+init=epsg:32619',
smns = 0.5,
nrow = 5)
```

### 6.35 Mostrar resultados

```
ozama_conv_prof$profiles
ozama_conv_prof$concavityindex
ozama_conv_prof$dimensionlessprofiles$layers[[2]]$aes_params$size <- 2
ozama_conv_prof$dimensionlessprofiles$theme$text$size <- 12
ozama_conv_prof$dimensionlessprofiles
```

### 6.36 Tabla $dx/dy$ , tanto en metros como adimensional. Útiles para construir profiles por cuenta propia

```
ozama_conv_prof$lengthzdata %>% tibble::as.tibble()
ozama_conv_prof$lengthzdatadmnls %>% tibble::as.tibble()
```

### 6.37 Limpiar archivo de bloqueo del conjunto de mapas de GRASS

```
unlink_.gislock()
```

#### 6.37.1 Parámetros de cuenca con *r.basin*

```
knitr:::opts_chunk$set(
  echo = TRUE,
  collapse=TRUE,
  fig.path = "img/rbasin/",
  eval = T
)
options(knitr.duplicate.label = "allow")
```

```
library(rgrass7)
gisdbase <- 'grass-data-test' #Base de datos de GRASS GIS
wd <- getwd() #Directorio de trabajo
wd
loc <- initGRASS(gisBase = "/usr/lib/grass78/",
                  home = wd,
                  gisDbase = paste(wd, gisdbase, sep = '/'),
                  location = 'c_ozama',
```

```

        mapset = "PERMANENT",
        override = TRUE)
gmeta()
execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

### 6.38 Convertir a números enteros la extensión y la resolución del DEM

```

library(raster)
rutadem <- 'data/dem.tif'
rawextent <- extent(raster(rutadem))
rawextent
devtools::source_url('https://raw.githubusercontent.com/geofis/rgrass/master/integerextent.R')
devtools::source_url('https://raw.githubusercontent.com/geofis/rgrass/master/xyvector.R')
newextent <- intext(e = rawextent, r = 90, type = 'inner')
newextent
gdalUtils::gdalwarp(
  srcfile = 'data/dem.tif',
  dstfile = 'data/demint.tif',
  te = xyvector(newextent),
  tr = c(90,90),
  r = 'bilinear',
  overwrite = T
)

```

### 6.39 Importar a sesión de GRASS

```

  {r, results='hide', warning=FALSE, message=FALSE}
rutademint <- 'data/demint.tif'
execGRASS(
  "g.proj",
  flags = c('t','c'),
  georef=rutademint)
gmeta()
execGRASS(
  "r.in.gdal",
  flags='overwrite',
  parameters=list(
    input=rutademint,

```

```

        output="demint"
    )
)
execGRASS(
  "g.region",
  parameters=list(
    raster = "demint",
    align = "demint"
  )
)

```

```

gmeta()
execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

#### 6.40 Generar red de drenaje para obtener coordenada posteriormente

```

execGRASS(
  "r.stream.extract",
  flags = c('overwrite','quiet'),
  parameters = list(
    elevation = 'demint',
    threshold = 80,
    stream_raster = 'stream-de-rstr',
    stream_vector = 'stream_de_rstr'
  )
)
execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

#### 6.41 Obtener coordenada

```

{r, results='hide', warning=FALSE, message=FALSE}
library(sp)
use_sp()
library(mapview)
netw <- spTransform(
  readVECT('stream_de_rstr'),
  CRSobj = CRS("+init=epsg:4326"))

mapview(netw, col.regions = 'blue', legend = FALSE)

```

#### 6.42 Transformar coordenada a EPSG:32619 como número entero

```

source('my-trans.R')
outlet <- as.integer(my_trans(c(-69.88117,18.47503)))

```

#### 6.43 Ejecutar r.basin

```

{r, results='hide', warning=FALSE, message=FALSE}
pref <- 'rbasin_ozama'
execGRASS(
  "r.basin",
  flags = 'overwrite',
  parameters = list(
    map = 'demint',
    prefix = pref,
    coordinates = outlet,
    threshold = 80,
    dir = 'salidas-rbasin/ozama'
  )
)

execGRASS(
  'g.list',
  flags = 't',
  parameters = list(
    type = c('raster', 'vector')
  )
)

```

Si r.basin arrojara error (sólo en el caso de error, no en caso de advertencia), ejecutar este bloque para borrar las salidas anteriores y reejecutar el r.basin:

```

execGRASS(
  "g.remove",
  flags = 'f',
  parameters = list(
    type = c('raster', 'vector'),
    pattern = paste0(pref, '*')
  )
)

```

#### 6.44 Cargar los vectoriales transformados a EPSG:4326 para visualizar en leaflet

```

{r, results='hide', warning=FALSE, message=FALSE}
rbnetw <- spTransform(
  readVECT('rbasin_ozama_demint_network'),
  CRSobj = CRS("+init=epsg:4326"))
rbnetw
rbmain <- spTransform(
  readVECT('rbasin_ozama_demint_mainchannel'),
  CRSobj = CRS("+init=epsg:4326"))
rbmain
rbbasin <- spTransform(
  readVECT('rbasin_ozama_demint_basin'),
  CRSobj = CRS("+init=epsg:4326"))
rbbasin

```

```

library(leaflet)
leaflet() %>%
  addProviderTiles(providers$Stamen.Terrain, group = 'terrain') %>%
  addPolylines(data = rbnetw, weight = 3, opacity = 0.7) %>%
  addPolylines(data = rbmain, weight = 3, opacity = 0.7, color = 'red') %>%
  addPolygons(data = rbbasin) %>%
  leafem::addHomeButton(extent(rbbasin), 'Ver cuenca')

```

#### 6.45 Explorar los parámetros de cuenca

```

library(readr)
rbozamapar1 <- read_csv("salidas-rbasin/ozama/rbasin_ozama_demint_parametersT.csv")
rbozamapar1 %>% tibble::as_tibble()
rbozamapar2 <- read_csv(
  "salidas-rbasin/ozama/rbasin_ozama_demint_parameters.csv",
  skip=2, col_names = c('Parameter', 'Value'))
rbozamapar2 %>% print(n=Inf)

```

## 6.46 Limpiar archivo de bloqueo del conjunto de mapas de GRASS

```
unlink_.gislock()
```

### 6.46.1 Curva e integral hipsométrica

```
knitr::opts_chunk$set(  
  echo = TRUE,  
  collapse=TRUE,  
  eval = T  
)  
options(knitr.duplicate.label = "allow")  
  
source(  
  knitr::purl(  
    'orden-de-red.Rmd',  
    output=tempfile()  
  )  
)  
knitr::opts_chunk$set(fig.path = "img/hypsocurve/")
```

## 6.47 Imprimir lista de mapas ráster y vectoriales dentro en la región/localización activa

```
execGRASS(  
  'g.list',  
  flags = 't',  
  parameters = list(  
    type = c('raster', 'vector')  
  )  
)
```

## 6.48 Representar cuencas

```
{r, results='hide', warning=FALSE, message=FALSE}  
library(sp)  
use_sp()  
library(mapview)  
bas2 <- readVECT('r_stream_basins_2')  
bas3 <- readVECT('r_stream_basins_3')
```

## 6.49 Curva e integral hipsométrica

```
{r, results='hide', warning=FALSE, message=FALSE}
source('integral_hypsometric_curve.R') #Cargada como función "HypsoIntCurve"
HypsoBasinsOrder2 <- HypsoIntCurve(
  basins = 'r_stream_basins_2',
  dem = 'dem',
  labelfield = 'cat',
  nrow = 2,
  labelsize = 4
)
```

```
HypsoBasinsOrder2$HypsoInt
HypsoBasinsOrder2$HypsoCurve
mapview(bas2, zcol='cat', col.regions = 'blue', legend = FALSE) %>%
  addStaticLabels(label = bas2$cat)
```

```
{r, results='hide', warning=FALSE, message=FALSE}
HypsoBasinsOrder3 <- HypsoIntCurve(
  basins = 'r_stream_basins_3',
  dem = 'dem',
  labelfield = 'cat',
  nrow = 1,
  labelsize = 4
)
```

```
HypsoBasinsOrder3$HypsoInt
HypsoBasinsOrder3$HypsoCurve
mapview(bas3, zcol='cat', col.regions = 'blue', legend = FALSE) %>%
  addStaticLabels(label = bas3$cat)
```

## 6.50 Limpiar archivo de bloqueo del conjunto de mapas de GRASS

```
unlink_.gislock()
```

## Referencias

- Arriaga-Cabrera, L., Aguilar, V., Espinoza, J. M., Galindo, C., Herrmann, H., Santana, E., ... Rosenzweig, L. (2009). Regiones prioritarias y planeación para la conservación de la biodiversidad. *Capital Natural de México*, 2, 433–457.
- Batlle, J. R. M. (2019). *Drainage rearrangement as a driver of geomorphological evolution during the upper pleistocene in a small tropical basin*.
- Batlle, J. R. M. (2020, September). Introducción a r. Retrieved from <https://geofis.shinyapps.io/>.

io/tutorial1/

Bivand, R. (2019). *Rgrass7: Interface between grass 7 geographical information system and r*. Retrieved from <https://CRAN.R-project.org/package=rgrass7>

Domínguez, F. M., Ch, A. G.-T., Gómez-Tagle, A. F., Perla, C., Metrópolis, F. R. H., & Tarímbaro, M. (2003). El análisis morfométrico con sistemas de información geográfica, una herramienta para el manejo de cuencas. *Instituto de Investigaciones Sobre Los Recursos Naturales. Morelia, México*.

Gaspari, F., Rodríguez Vagaría, A., Senisterra, G., Delgado, M. I., & Besteiro, S. (2013). *Elementos metodológicos para el manejo de cuencas hidrográficas*.

Gutiérrez Elorza, M. (2008). *Geomorfología*.

Gutiérrez, W. (2014). *Recopilación documental de informaciones relacionadas con la cuenca, calidad de sus aguas, el saneamiento y rehabilitación del río ozama*.

Horton, R. E. (1945). Erosional development of streams and their drainage basins; hydrophysical approach to quantitative morphology. *Geological Society of America Bulletin*, 56(3), 275–370.

Martínez Batlle, J. R. (2020, March). Canal de youtube pelempito1. geomorfología (geo114), licenciatura en geografía, uasd. Retrieved from <https://www.youtube.com/watch?v=9F7BIAUNvRY/&list=PLDcT2n8UzsCSt1-NnUQ8anwHhmouFr0Kv>

Medio-Ambiente. (2012). Ozama. Retrieved from <https://ambiente.gob.do/cuencas-hidrograficas/ozama/#:~:text=La%20cuenca%20del%20r%C3%ADo%20Ozama,'27.891%E2%80%B3W%20longitud%20oeste>.

Pedraza Gilsanz, J. de. (1996). *Geomorfología: Principios, métodos y aplicaciones*.

Yamazaki, D. (2018).