

Sevilla R - Sevilla, 8 de Enero de 2019

# Depuración geométrica-topológica de datos geográficos con R.

*Jerónimo Carranza Carranza*

[jeronimo.carranza@asterionat.com](mailto:jeronimo.carranza@asterionat.com)

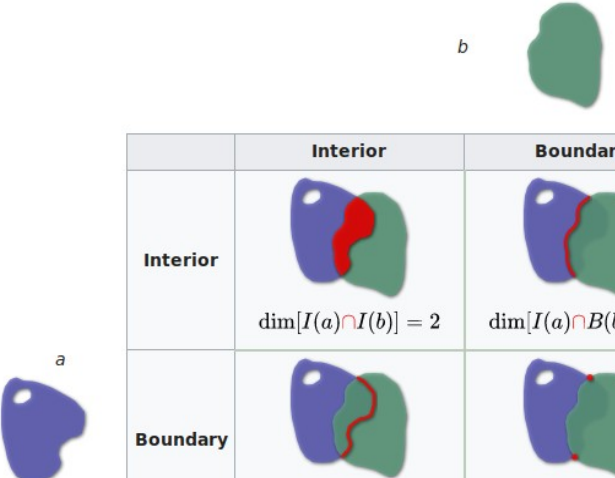





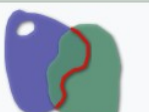
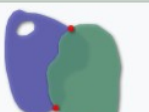


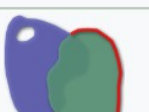

[www.asterionat.com](http://www.asterionat.com)

# Depuración geométrica-topológica

La **topología geoespacial** hace referencia a las relaciones espaciales entre los diferentes elementos gráficos (puntos, líneas, polígonos) que representan las entidades geográficas.

La relación espacial entre dos geometrías  $a$  y  $b$  (punto, línea o polígono) se basa **formalmente** en la matriz resultante de la intersección entre sus Interiores, sus Bordes y sus Exteriores y la dimensionalidad de dichas intersecciones (**DE-9IM**):



	Interior	Boundary	Exterior
Interior	 $\dim[I(a) \cap I(b)] = 2$	 $\dim[I(a) \cap B(b)] = 1$	 $\dim[I(a) \cap E(b)] = 2$
Boundary	 $\dim[B(a) \cap I(b)] = 1$	 $\dim[B(a) \cap B(b)] = 0$	 $\dim[B(a) \cap E(b)] = 1$
Exterior	 $\dim[E(a) \cap I(b)] = 2$	 $\dim[E(a) \cap B(b)] = 1$	 $\dim[E(a) \cap E(b)] = 2$





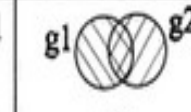
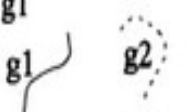
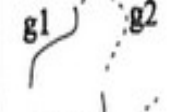
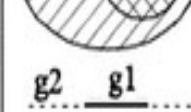



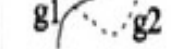
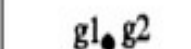


## Predicados topológicos:

- **Disjunto**: FF\*FF\*\*\*\*
- **Intersecta**: No(Disjunto)
- **Toca**: FT\*\*\*\*\* o F\*\*T\*\*\*\*\* o F\*\*\*T\*\*\*\*
- **Contiene**: T\*\*\*\*\*FF\*
- **Dentro**: T\*F\*\*F\*\*\*
- **Cubre**: T\*\*\*\*\*FF\* o \*T\*\*\*\*\*FF\* o \*\*\*T\*\*FF\* o \*\*\*\*T\*FF\*
- **Cruza**: T\*T\*\*\*\*\* o T\*\*\*\*\*T\*\* o 0\*\*\*\*\*  
 $\dim(a) < \dim(b)$     $\dim(a) < \dim(b)$     $\dim(\text{any}) = 1$
- **Solapa**: T\*T\*\*\*T\*\* o T\*T\*\*\*T\*\*  
 $\dim = 0$  o  $2$     $\dim = 1$

Ejemplo DE-9IM de dos polígonos

StringCode = '212101212' o

T\*\*\*\*\* - \*T\*\*\*\*\* - \*\*\*T\*\*\*\*\* - \*\*\*\*T\*\*\*\*\*

disjunto	toca	dentro	cruza	solapa
				
				
				

# Depuración geométrica-topológica

Actualmente en general, se almacenan geometrías simples de las que se derivan sus relaciones topológicas que se contrastan (y mantienen) con las **reglas** topológicas establecidas en un modelo de datos determinado.

Las relaciones topológicas más **frecuentes** son:

- Toca y no solapa (entre polígonos).
- Contigüidad (entre línea y polígono).
- Pertenencia (arcos a polígonos).
- Conectividad (entre arcos, en redes).
- Inclusión (punto en polígono, línea en polígono, polígono en polígono).

Los datos topológicos o basados en la topología geoespacial sirven para **detectar y corregir errores de digitalización**.

La topología es necesaria para llevar a cabo algunos tipos de análisis espacial, como el análisis de redes.

Dangles



Switchbacks



Knots



Loops



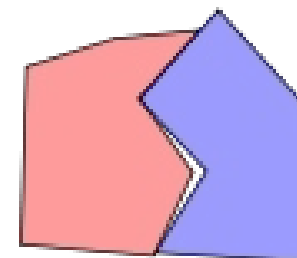
Overshoots



Undershoots



Slivers



Errores topológicos más comunes producidos por una incorrecta digitalización de elementos geográficos.



# El paquete sf

- Implementa el estándar (OGC e ISO) '**simple features**' en R  
*'Simple feature': abstracción de un fenómeno del mundo real que tiene geometría y otros atributos, y en el que la geometría está descrita por segmentos de línea recta o interpolación planar sobre conjuntos de puntos, sin curvas*
- **sf** representa la geometría por puntos, líneas o polígonos, o colecciones de los mismos
- **sf** implementa los **17 tipos 'simple feature'** para todas las dimensiones (XY, XYZ, XYM, XYZM)
- **sf** está soportado por las bibliotecas OSGeo GEOS y GDAL con codificaciones de texto (**WKT**) y binarias (**WKB**)
- En la práctica los objetos sf '**simple features**' en R son **data.frames** (o **tibbles**) con una columna geométrica que **sf** proporciona acceso junto a un conjunto de métricas topológicas, predicados y operaciones que el estándar especifica

# objeto sf

```
> rivers[,c(8,9,19,20,21,22)]
```

Simple feature collection with 138 features and 5 fields

geometry type: LINESTRING

dimension: XY

bbox: xmin: -182235.2 ymin: 1604168 xmax: 103733.7 ymax: 1764059

epsg (SRID): 32629

proj4string: +proj=utm +zone=29 +datum=WGS84 +units=m +no\_defs

First 10 features:

	MAJ_NAME	SUB_NAME	HYNID	HYNOUT	Partes	geometry
1	Senegal	Ferlo	1	0	1	LINESTRING (-68964.3 166351...
2	Senegal	Ferlo	2	0	1	LINESTRING (-133749.2 17390...
3	Senegal	Ferlo	3	0	1	LINESTRING (5161.419 173421...
4	Senegal	Ferlo	4	0	1	LINESTRING (-36094.96 16650...
5	Senegal	Ferlo	5	0	1	LINESTRING (-119340.9 17324...
6	Senegal	Ferlo	6	0	1	LINESTRING (13250.4 1660302...
7	Senegal	Ferlo	7	0	1	LINESTRING (18372.14 161281...
8	Senegal	Ferlo	8	0	1	LINESTRING (15468.31 165887...
9	Senegal	Ferlo	9	0	1	LINESTRING (20544.87 166547...
10	Senegal	Ferlo	10	0	1	LINESTRING (-27903.01 16695...

# sf: operaciones sobre geometrías

## Unarias:

- **cantidades**: length, area
- **dimensión**: 0 = punto, 1 = linea, 2 = area
- **predicados**: is\_valid, is\_simple, is\_empty
- **nuevas geometrías**: buffer, centroid, boundary, convex\_hull, simplify, linemerge, polygonize, node, point\_on\_surface, triangulation

## Binarias / n-arias:

- **cantidades**: distance
- **predicados**: intersects, within, contains, covers, covered\_by, crosses, touches, overlaps, equals, disjoint, all other DE-9IM
- **nuevas geometrías**: intersection, difference, union, sym\_difference

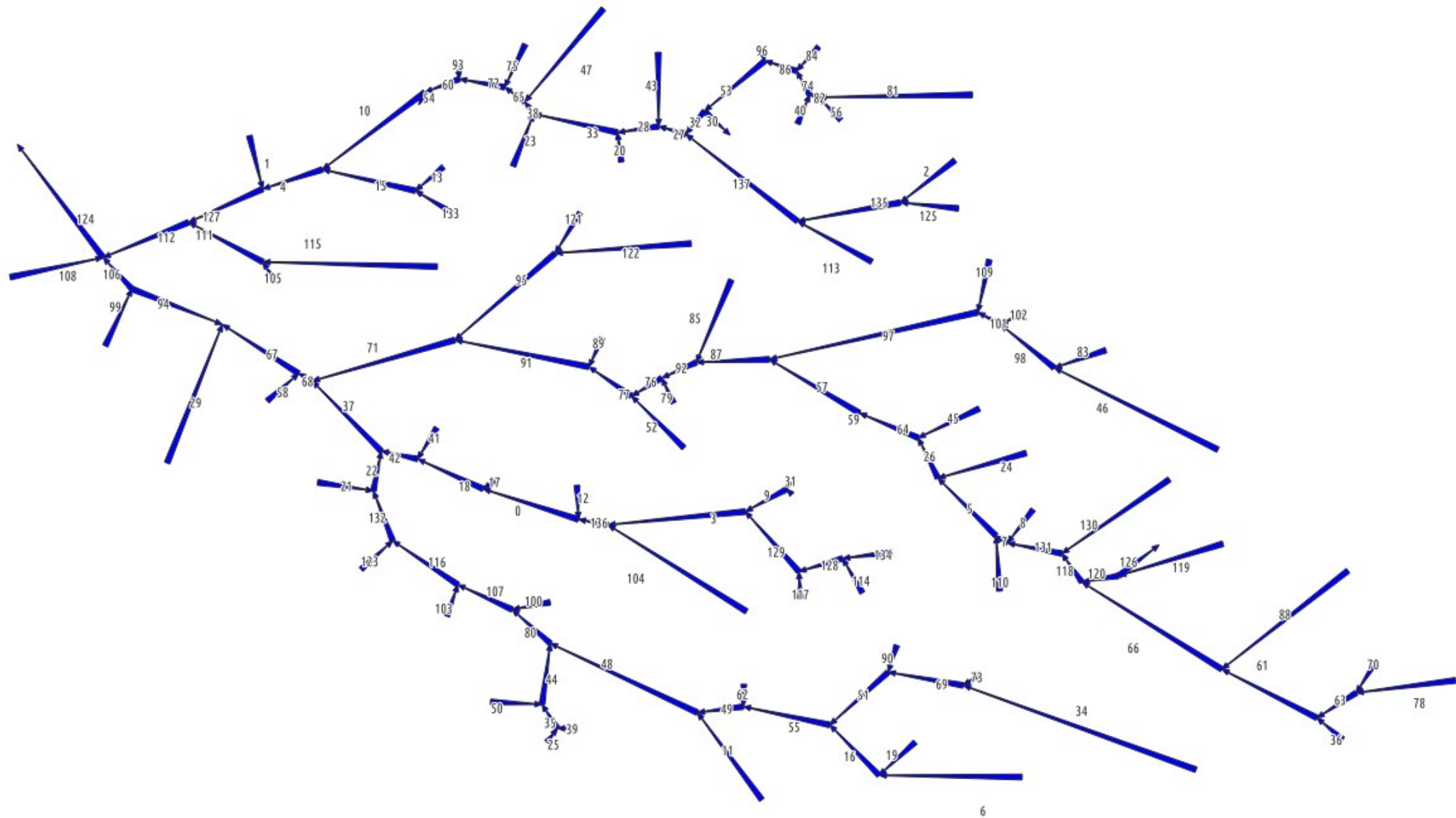
Todas las funciones y métodos empiezan con **st\_**

```
> st_length(rivers[1:8,])
```

Units: [m]

```
[1] 21631.851 13875.216 15783.509 29811.011 13161.745 16289.847 37554.694 2640.066
```

# Caso de uso: red hidrográfica



```
# Devuelve lista con ids de arcos adjacentes arriba o abajo a uno dado
arclist = function(arcid, tree, updw=0){
  arc <- tree[arcid,]
  node = st_line_sample(arc, sample = updw) # Devuelve vértice
  arcs_id = st_intersection(tree, node)$HYNID # Devuelve intersección
  arcs_id = arcs_id[arcs_id != arcid]
  return(arcs_id)
}
```

```
# Chequea si un arco debe invertir su sentido
checkarc = function(arcid, tree){
  uparcs = arclist(arcid, tree, 0)
  dvarcs = arclist(arcid, tree, 1)
  if (any(uparcs %in% checked)) {
    tree[arcid,]$geometry = flip(tree[arcid,])
    tocheck <- c(tocheck, dvarcs)
    flipped <- c(flipped, arcid)
  } else {
    tocheck <- c(tocheck, uparcs)
  }
  return(tocheck)
}
```



# Inverte el sentido de un arco

```
flip = function(arc){  
  crs_arc = st_crs(arc)           # Devuelve coord. ref. sys.  
  prec_arc = st_precision(arc)    # Devuelve precisión  
  coord = st_coordinates(arc)     # Devuelve lista de coordenadas  
  revcoord = apply(coord[,1:2], 2, rev)  
  flipped = st_sfc(st_linestring(revcoord),  
                   crs = crs_arc, precision = prec_arc) # Crea nuevo sf  
  return(flipped)  
}
```

```
# Recorrido y chequeo completo de la red hidrográfica
check = function(hynsf, return_op = flipped){
  checked <- vector()
  tocheck <- vector()
  flipped <- vector()
  outfalls <- hynsf[hynsf$HYNOUT==1,]
  for (i in 1:nrow(outfalls)){
    checked <- c(checked, outfalls[i,]$HYNID)
    tocheck <- checkarc(outfalls[i,]$HYNID, hynsf)
  }
  n = 0
  while(n <= nrow(hynsf) & length(tocheck) > 0) {
    checked <- c(checked, tocheck[1])
    tocheck <- checkarc(tocheck[1], hynsf)
    tocheck <- tocheck[-which(tocheck %in% checked)]
    n = n + 1
  }
  cat('Checked Arcs:', checked, '\n')
  cat('Flipped Arcs:', flipped, '\n')
  return(return_op)
}
```

```
# Reemplazar con su fuente de datos
rivers <- st_read("example_data/rivers02.shp")

# Si HYNID (ID exclusivo entero) y HYNOUT (1 para desembocaduras y 0
# para otros arcos) existen en su fuente de datos y están correctamente
# asignados, puede comentar las líneas siguientes
rivers$HYNID = c(1:nrow(rivers))
rivers$HYNOUT = 0
rivers[c(125),]$HYNOUT = 1 # c(125) is the list of IDs of outfalls
rivers[,]$HYNOUT

# Ejecutar este chunk para probar el script con sus datos
rivers_flipped = check(rivers, flipped)
rivers_flipped

# Ejecutar este chunk para salvar los resultados.
# Reemplazar con su destino.
st_write(rivers, "example_data/rivers02_updated.shp", delete_layer =
TRUE)
```

# Representación geográfica de resultados

library(ggplot2)

ggplot() +

geom\_sf(data = rivers[rivers\$HYNOUT==1,], color = 'green') +

geom\_sf(data = rivers[rivers\$HYNOUT==0,], color = 'blue') +

geom\_sf(data = rivers[rivers\$HYNID %in% rivers\_flipped,],  
color = 'red')

