Como colofón del curso en línea de “Enfoques, Métodos y Herramientas para la Conectividad Ecológica” se desarrolla un proyecto en donde se utilizan las herramientas del programa Makurhini escrito en lenguaje de programación “R”.

En este trabajo final se seleccionaron datos GIS (SIG) del repositorio de la *CONABIO*.

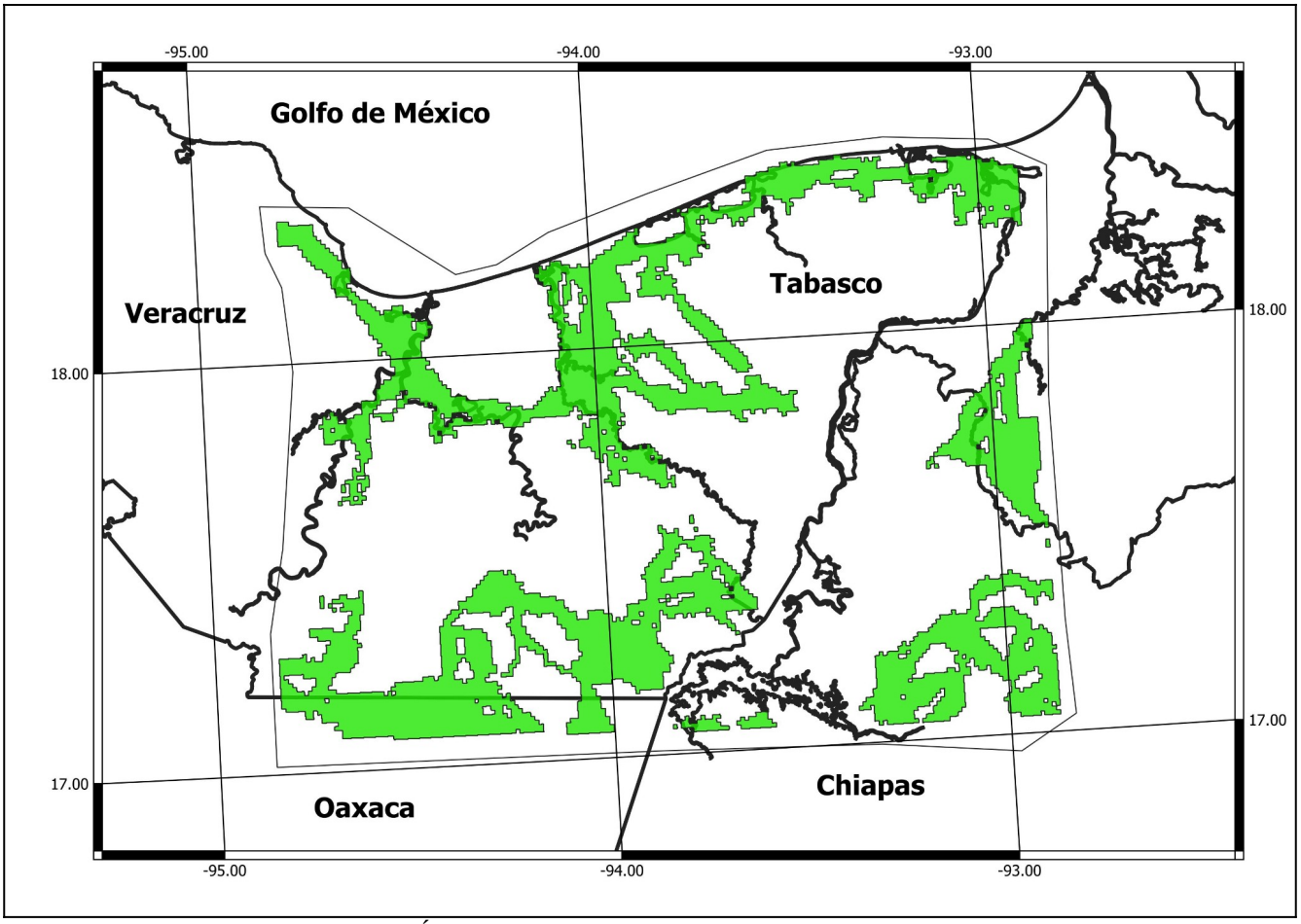



Figura 1: Área de estudio. En color verde: *Corredor Bioclimático*.

Los datos del corredor bioclimático se consideran los nodos o parches ecológicos, se incluye la presencia de cuerpos hidrográficos como son los ríos, los cuales pueden ser fronteras para ciertas especies en su dispersión a través del paisaje seleccionado, la región no presenta cuerpos orográficos de importancia como montañas, volcanes, desfiladeros o sumideros.

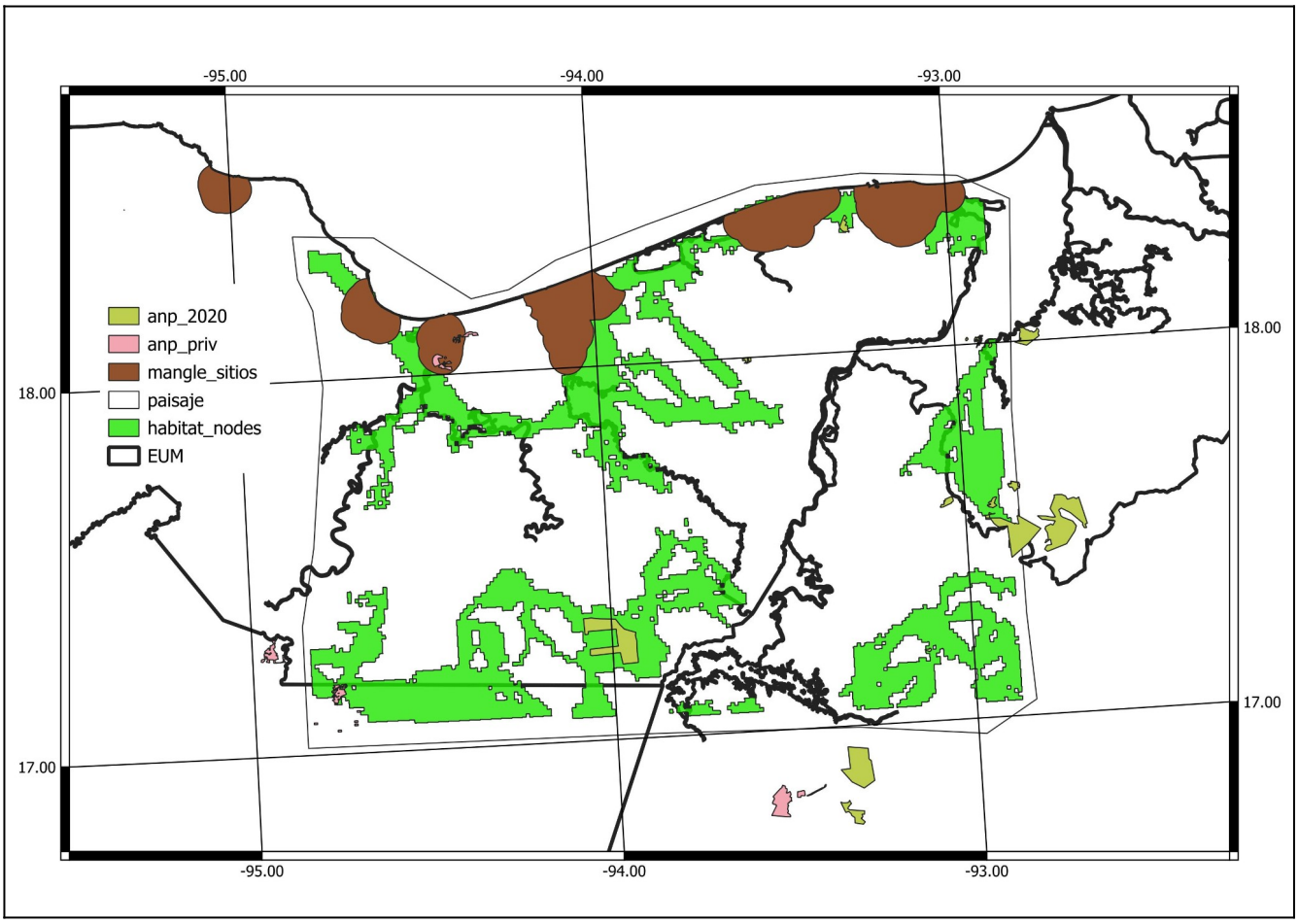



Figura 2: ANP´s privadas, *estatales*, ejidales, privadas o voluntarias, *sitios de manglar* con importancia ecológica.

La información de los datos SIG fue re-proyectada del esferoide EPSG 4326 al EPSG 6362, los archivos del repositorio en formato raster se proyectaron a ESRI Shapefile en el esferoide EPSG 6362. El polígono irregular al que llamaremos paisaje tiene una área de 3,176,476 ha. Los parches ecológicos suman 31 nodos dentro del paisaje.

**Análisis de fragmentación resultados:**

| | Summary.landscape.metrics | Viewer.Panel..Value |
|---|---|---|
| 1 | Patch area (ha) | 747 226.54 |
| 2 | Number of patches | 31 |
| 3 | Size (mean) | 24 104.08 |
| 4 | Patches < minimum patch area (100km²) | 1 |
| 5 | Patches < minimum patch area (%) | 0.01 |
| 6 | Total edge | 3 994.49 |
| 7 | Edge density | 0.01 |
| 8 | Patch density | 0 |
| 9 | Total Core Area (ha) | 558 855.24 |
| 10 | Cority | 0.81 |
| 11 | Shape Index (mean) | 0.19 |
| 12 | FRAC (mean) | 0.74 |
| 13 | MESH (ha) | 56 760.58 |

Tabla 1: Resultados de la Fragmentación a nivel de paisaje (edge = 500m).

De los resultados de la tabla 1: Se contabilizaron 31 parches ecológicos, solamente un (1) parche ecológico se encuentra por debajo de los 100 km² de área, la densidad del borde es de 0.01, el parch density es cero (0) lo que nos dice que es bajo el número o cantidad de parches en el paisaje, el área núcleo de los parches es de 74.79% respecto al área total de los parches, lo vemos reflejado en cority con un valor de 0.81 indicando la cantidad de área de zona núcleo (el valor va de 0 a 1); del área total de los parches ecológicos solo 56,760.58 hectáreas se encuentran fuera de una fragmentación potencial. El porcentaje de fragmentación en el paisaje es de 98.21% usando el MESH (Effective Mesh Size). La distancia de borde utilizada fue de 500m.

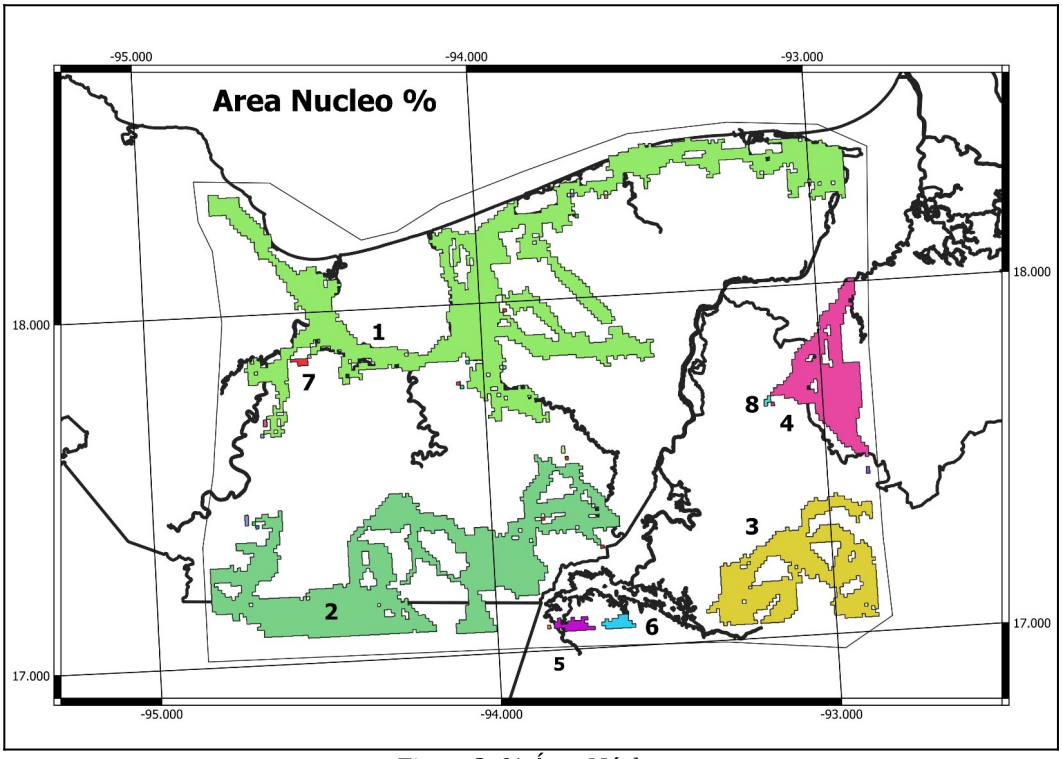



Figura 3: % Área Núcleo:
1 = 72.81, 2 = 78.02, 3 = 74.93, 4 = 80.10, 5 = 40.78, 6 = 52.46, 7 = 29.03, 8 = 1.89

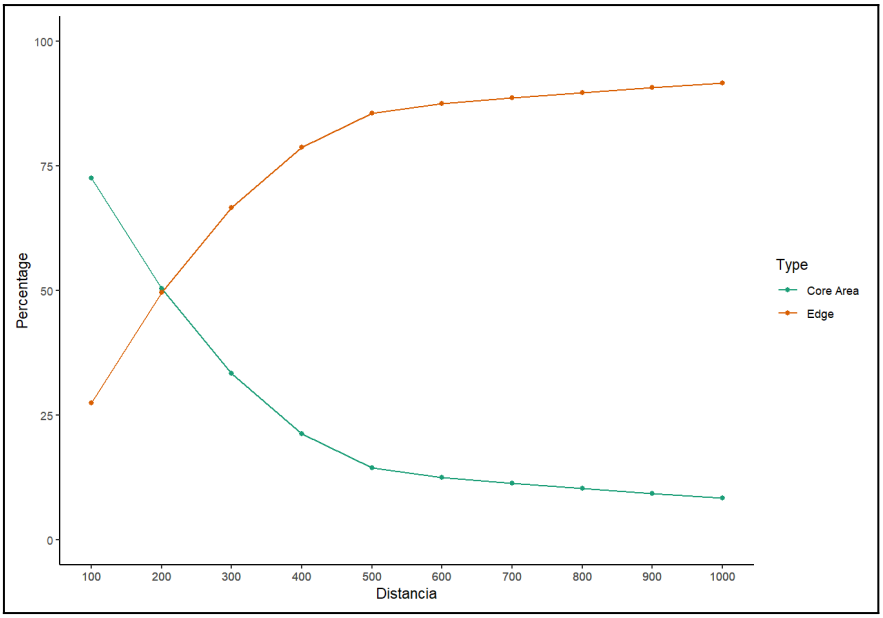



Figura 4: Efecto de borde vs área núcleo.

De los resultados que observamos en la figura 4 podemos decir que un efecto de borde de 500m repercute en que la zona núcleo decrezca a un 20%, en otras palabras si definimos una zona de amortiguamiento para los parches ecológicos de este ejemplo, usaríamos en el mejor de los casos una distancia de borde de 100m a un máximo de 300m.

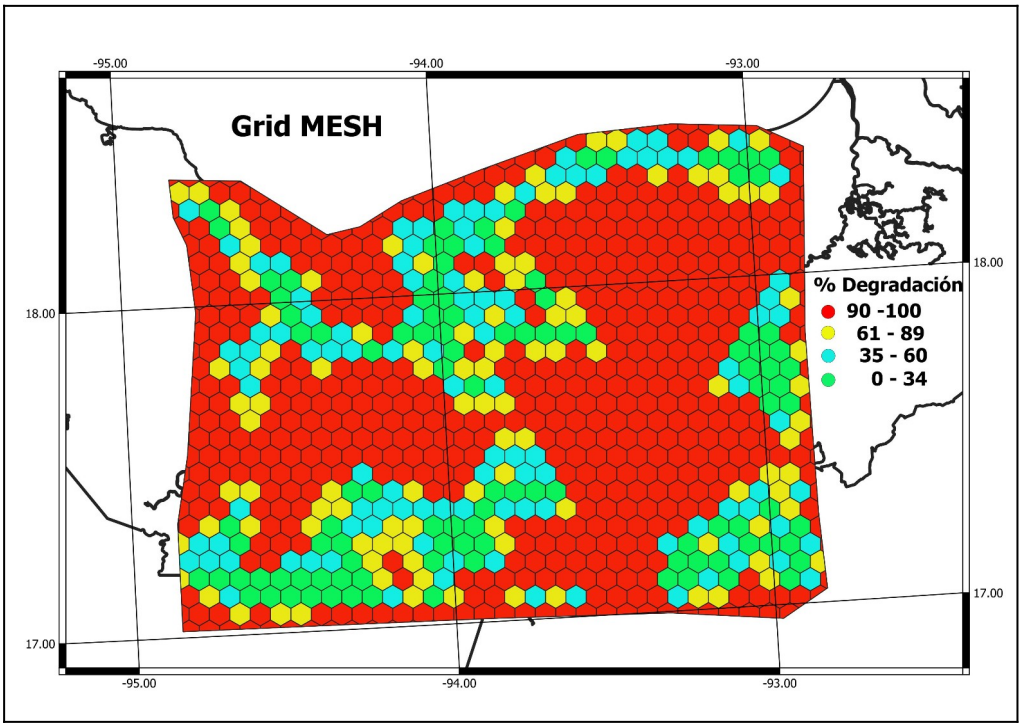



Figura 5: Gráfica MESH (% de Degradación) celdas de 40km

**Índices de Centralidad:**

Con la función Centrality de makurhini se calcularon los índices de centralidad en los parches ecológicos en el paisaje, considerando una distancia de 10,000m (10km) a partir del centroide del nodo y con una probabilidad de dispersión de 0.5.

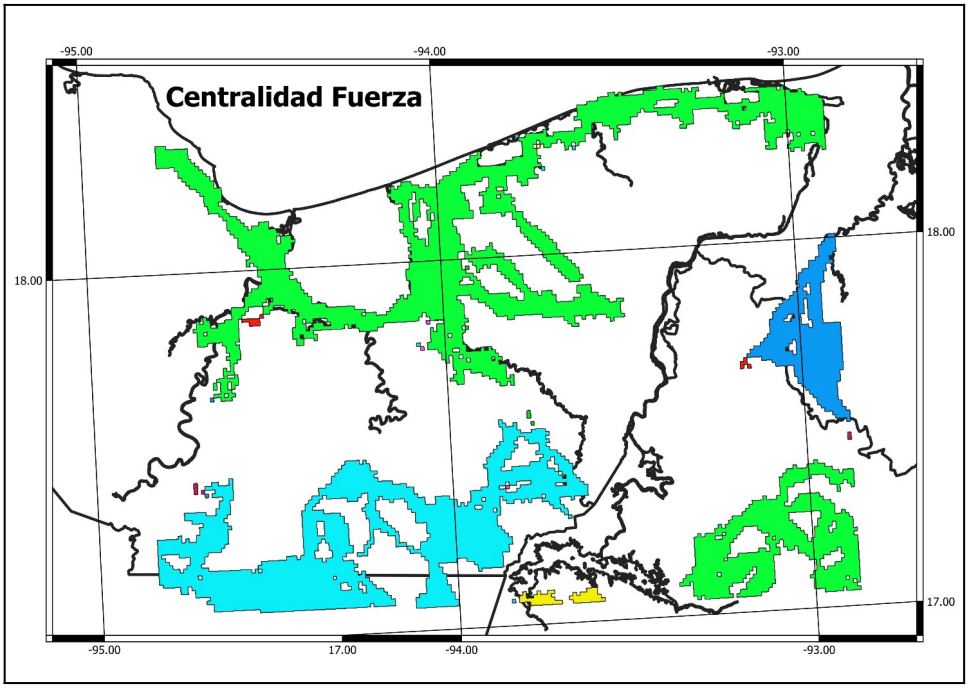



Figura 6: Indice de Centralidad Fuerza (Centrality Strength)

Los resultados del índice de centralidad fuerza en los nodos ecológicos indican que los nodos en color azul ($1.56 \times 10^6 – 3.01 \times 10^6$) tienen mayor peso para la probabilidad de dispersión, seguidos de lo nodos de color verde ($7.24 \times 10^5 – 1.55 \times 10^6$), los nodos de color rojo ($4.08 \times 10^5 – 7.24 \times 10^5$), los nodos de color amarillo ($1.04 \times 10^5 – 1.53 \times 10^5$) y por último los nodos en color cían (27587 – $1.04 \times 10^5$).

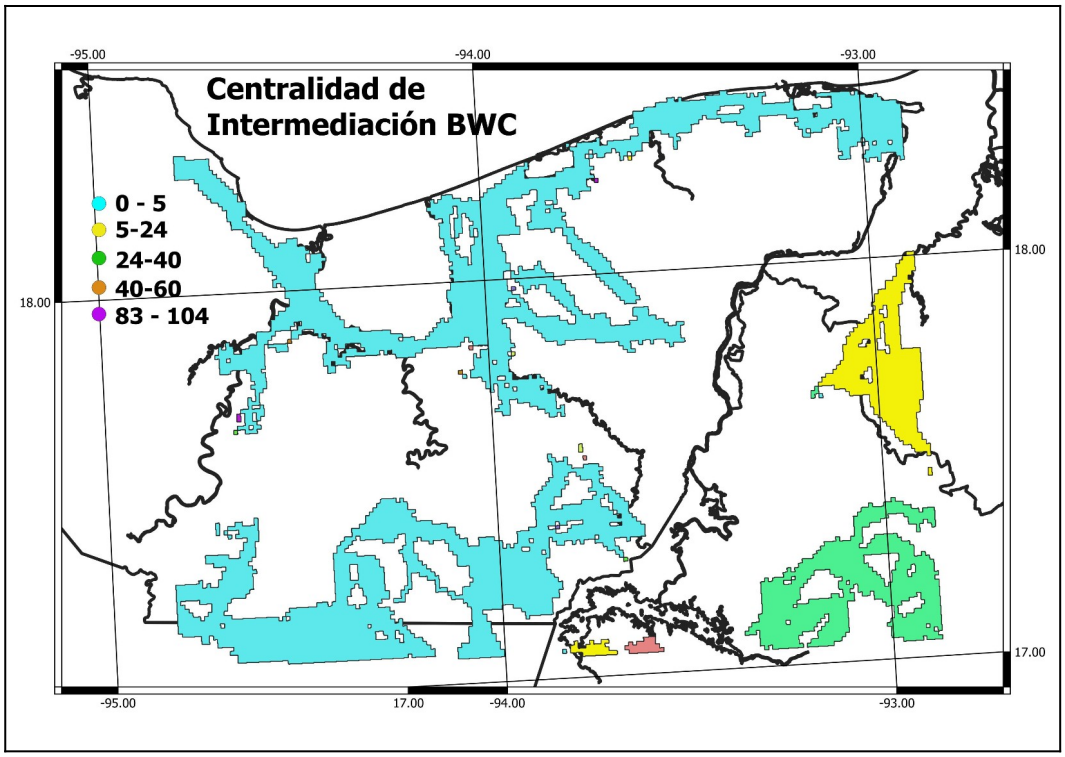



Figura 7: Indice de Centralidad de Intermediación.

Los parches ecológicos en color magenta, anaranjado y verde presentan ser de importancia como parches de interconexión o stepping stones que permiten la conexión entre los parches existentes en el paisaje. Los parches en color cían no se deben discriminar, deben considerarse de prioridad de preservación y / o cuidado para no fracturar la conectividad en la dispersión de las especies en el paisaje.

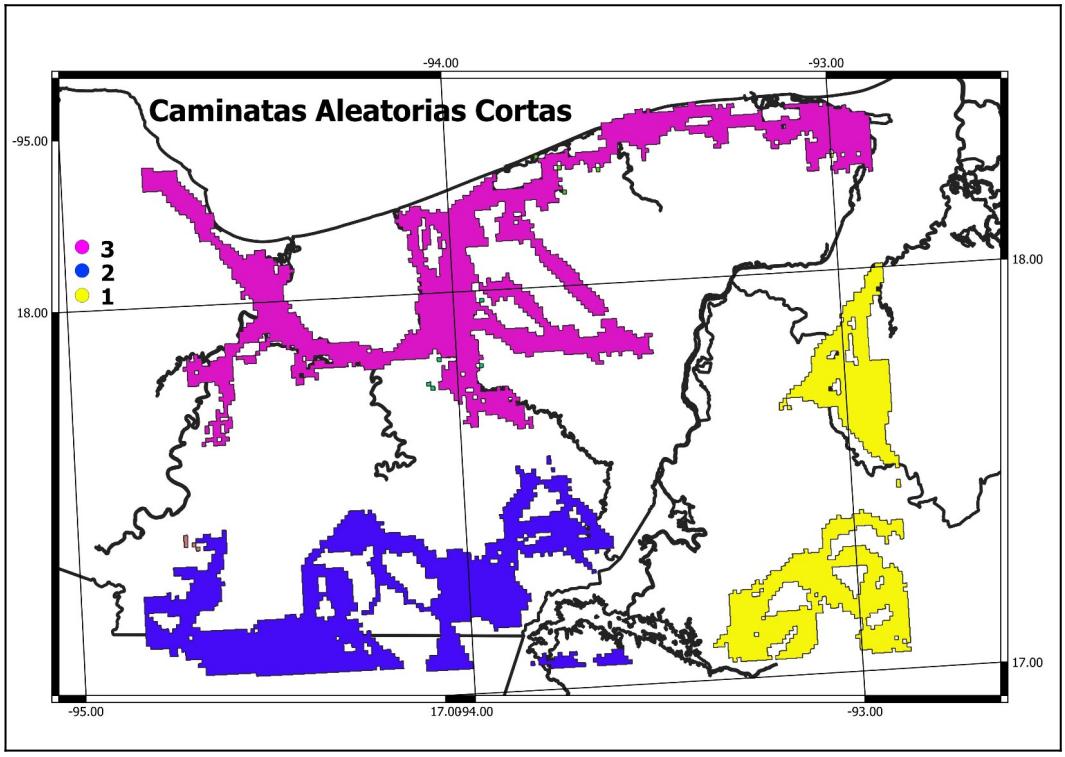



Figura 8: Caminatas aleatorias cortas.

La figura 8 muestra tres grupos de parches en el paisaje, el grupo en color magenta es el de mayor extensión permitiendo a un animal o semoviente tener una caminata aleatoria por un espacio o distancia mayor, no se debe considerar que los grupos en azul y amarillo no son de importancia, debemos entender que también son de importancia por lo que deben considerarse para su preservación o restauración, buscando que los tres grupos de parches ecológicos se interconecten con más facilidad para el desplazamiento de la especie y la preservación del entorno ecológico en los parches y entre los parches.

**Índice Integral de Conectividad (IIC) y Probabilidad de Conectividad (PC)**

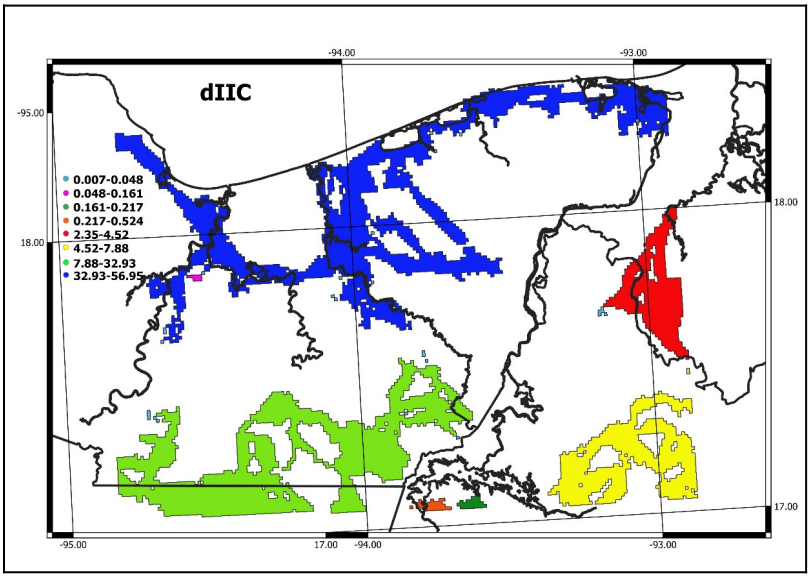



Figura 9: dIIC.

El IIC se calculó con la función de Makurhini MK_dPCIIC usando distancia euclidiana a partir del borde del parche ecológico, un keep = 0.1, distancia de dispersión de 10000 (10km), métrica IIC, usando el paisaje como atributo máximo (LA en la función) y opción onlyoverall = FALSE.

El PC se calculó con la función de Makurhini MK_dPCIIC usando distancia euclidiana a partir del borde del parche ecológico, un keep = 0.1, distancia de dispersión de 10000  (10km), métrica PC, usando el paisaje como atributo máximo (LA en la función), opción onlyoverall = FALSE y una probabilidad de dispersión de la especie de 0.5.

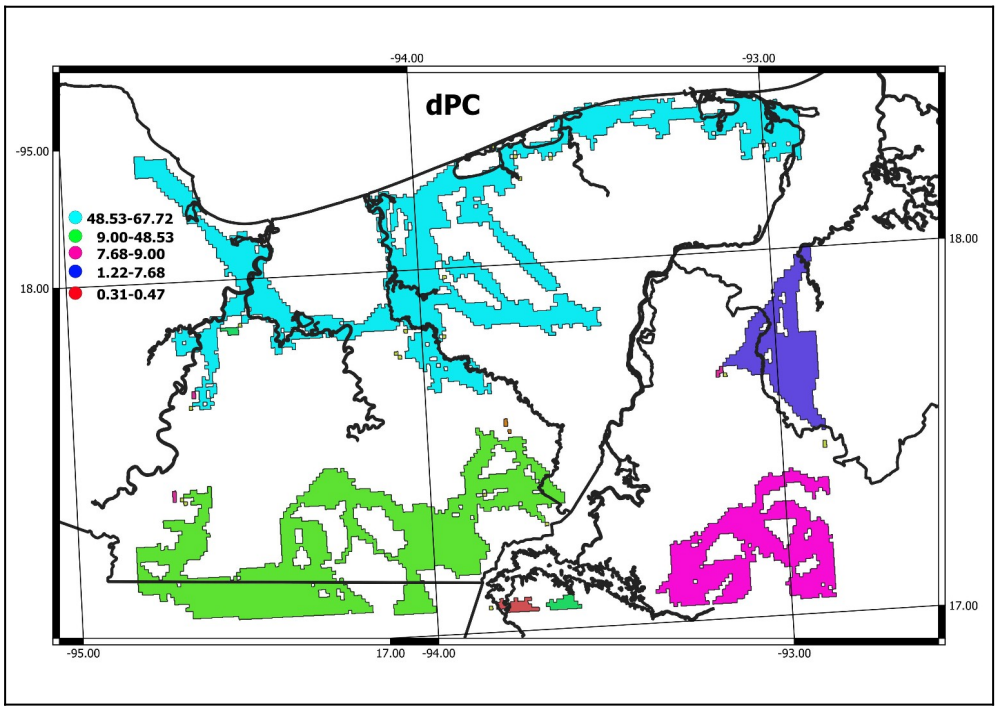



Figura 10: dPC.

**Índice dIICIntra y dPCIntra:**

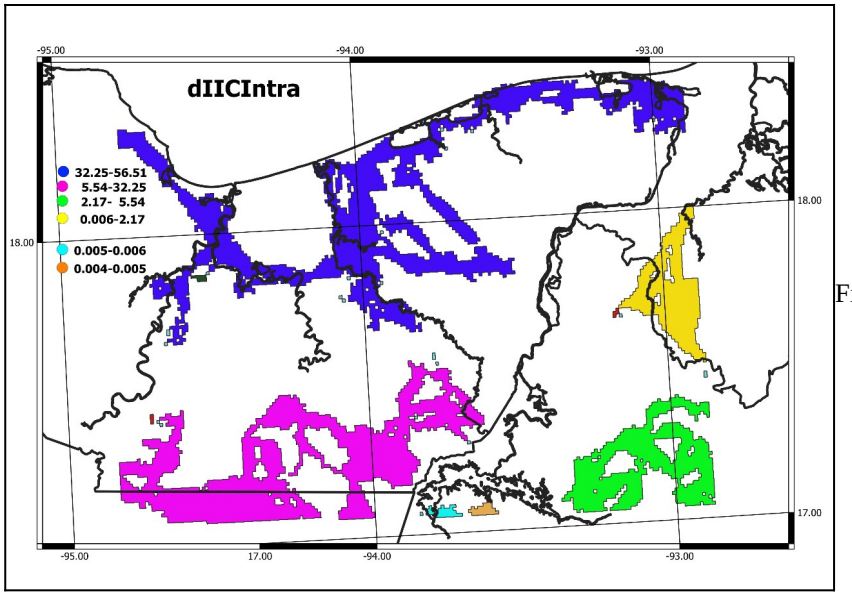



Figura 11: dIICIntra

El índice dIICIntra e dPCIntra se usa para calcular la interconectividad en el interior del parche ecológico (figuras 11 y 12) lo que permite tener una idea de la dispersión o caminata aleatoria de la especie en el nodo, se observan los nodos de mayor peso en ambos índices, en algunos nodos los valores obtenidos son muy similares.

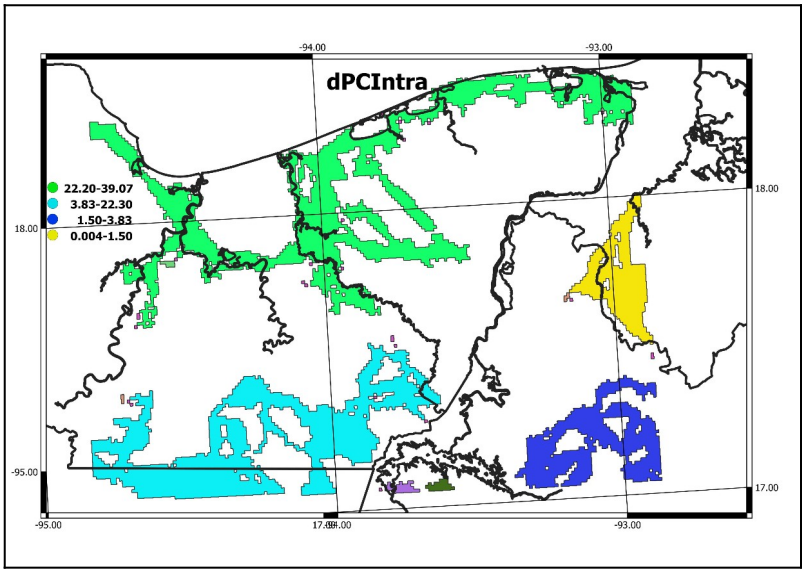



Figura 12: dPCIntra

**Índice dIICFlux y dPCFlux:**

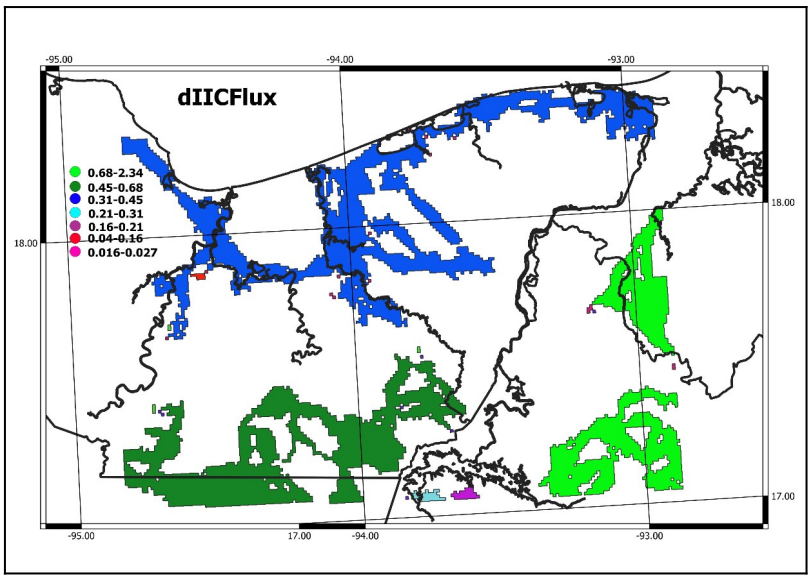



Figura 13: dIICFlux

Los índices dIICFlux y dPCFlux nos muestran los resultados del cálculo del flujo de un parche hacia los parches vecinos y reciprocamente, los valores obtenidos son representativos a manera de mostrar que parches son los de mayor peso para un flujo de especies entre parches vecinos, estos parches reciben flujo y emiten flujo para la interconexión y tránsito de especies en el paisaje.

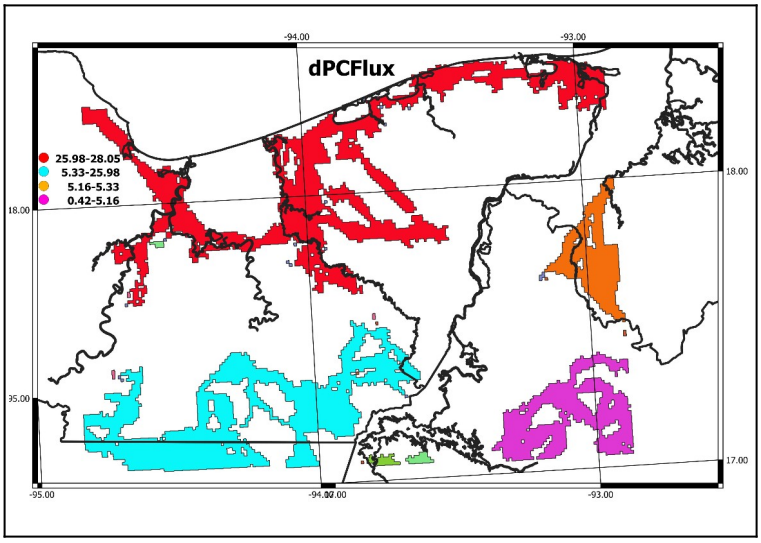



Figura 14: dPCFlux

**Índice dIICConn y dPCConn:**

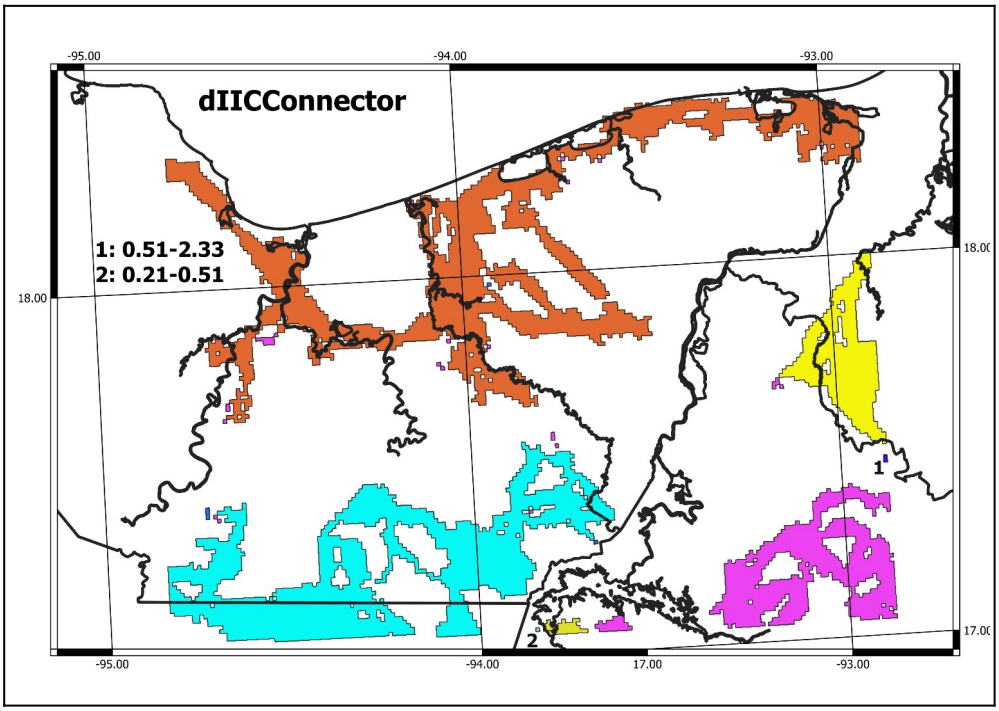



Figura 15: dIICConn

En las figuras 15 y 16 se muestra el dIICCon y el dPCConn respectivamente, en este índice se calculan los pesos de parches ecológicos que pueden funcionar como stepping stones o parches de paso de la especie de un nodo a otro nodo. En la figura 15 se obtiene que los parches marcados con los números 1 y 2 según a modelación fungen como parches de paso. En la figura 16 solo se obtiene del proceso un solo parche de paso (número 1).

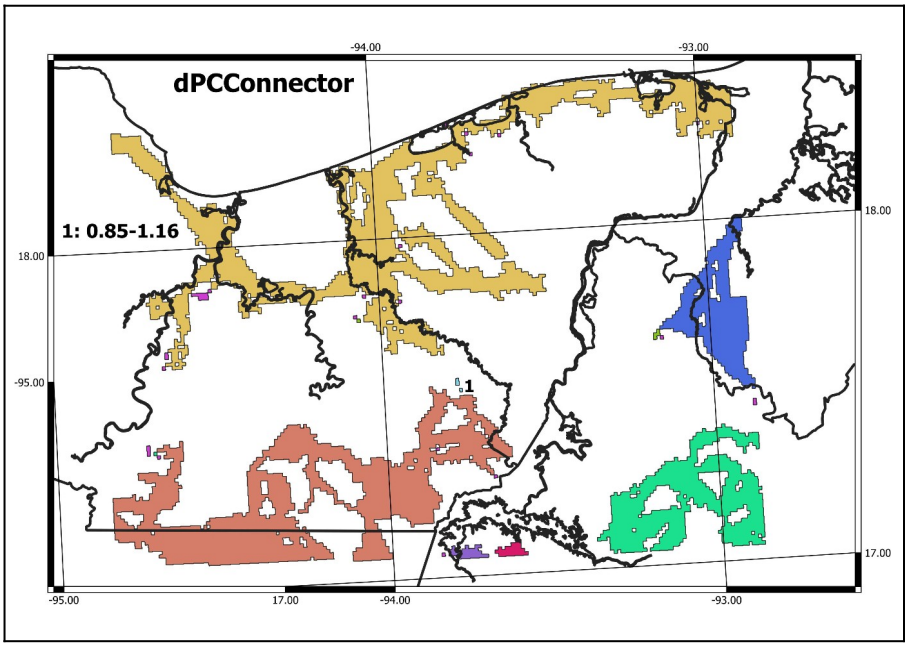



Figura 16: dPCConn

**Prioridad de Restauración:**

Del total de los parches ecológicos en el paisaje se seleccionaron aleatoriamente doce (12) parches en condición de no existentes en el paisaje y como candidatos a su integración / restauración en el paisaje. Se consideró distancia euclidiana a partir de borde de cada parche con un factor keep=0.1, una distancia de dispersión de 10000 (10km), una probabilidad de dispersión de 0.5 y onlyrestore = FALSE.

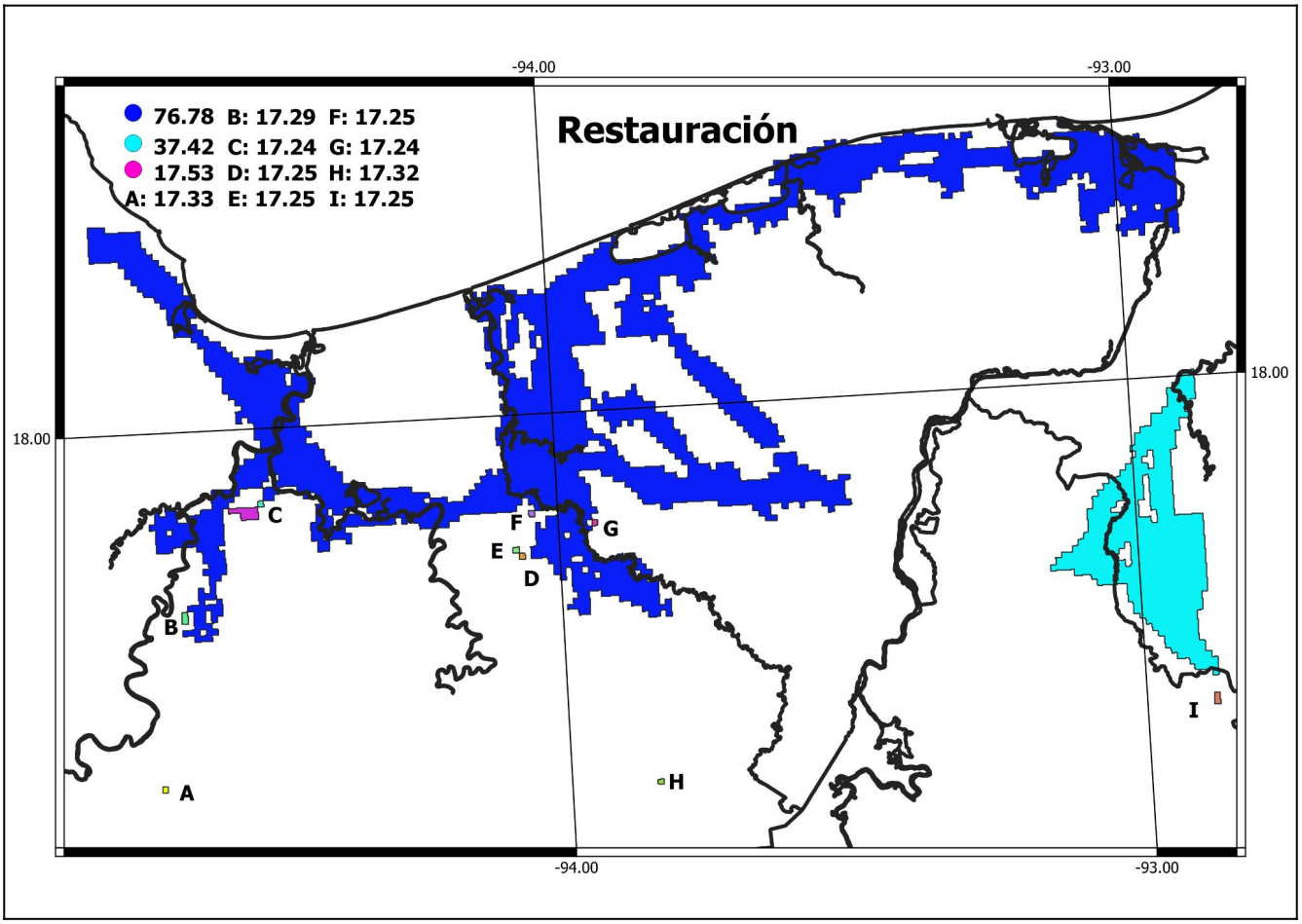


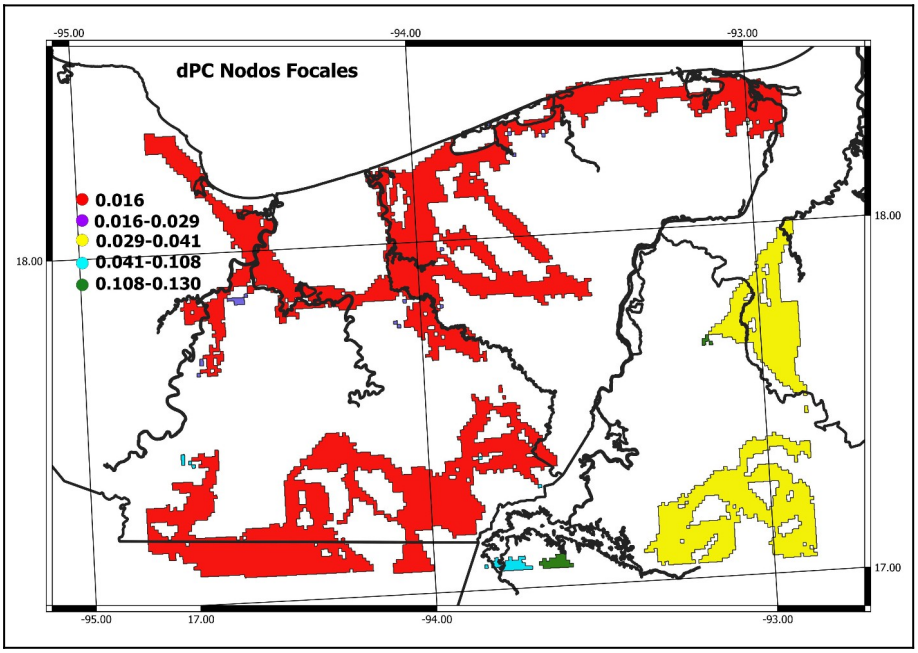

Figura 17: Restauración.

Los valores que se muestran en la figura 17 indican el peso o importancia de los parches propuestos a ser restaurados, no se deben ignorar o desechar lo pequeños parches dado que pueden contribuir a la conectividad del paisaje como steeping stones, mismo que se ha observado en figuras 13, 15 y 16.

**Prioridades de Conservación y Restauración por Nodos Focales.**

Los parches son sometidos a un análisis de conectividad PC considerando a partir del parche se realiza una búsqueda de parches contiguos o cercanos a distancia buffer de 20000m (20km) con una distancia de dispersión de la especie de 10000m (10km), una probabilidad de 0.5, una distancia euclidiana a partir del borde del parche con un factor keep = 0.1, sin fragmentación.

Figura 18: dPC Nodos focales.



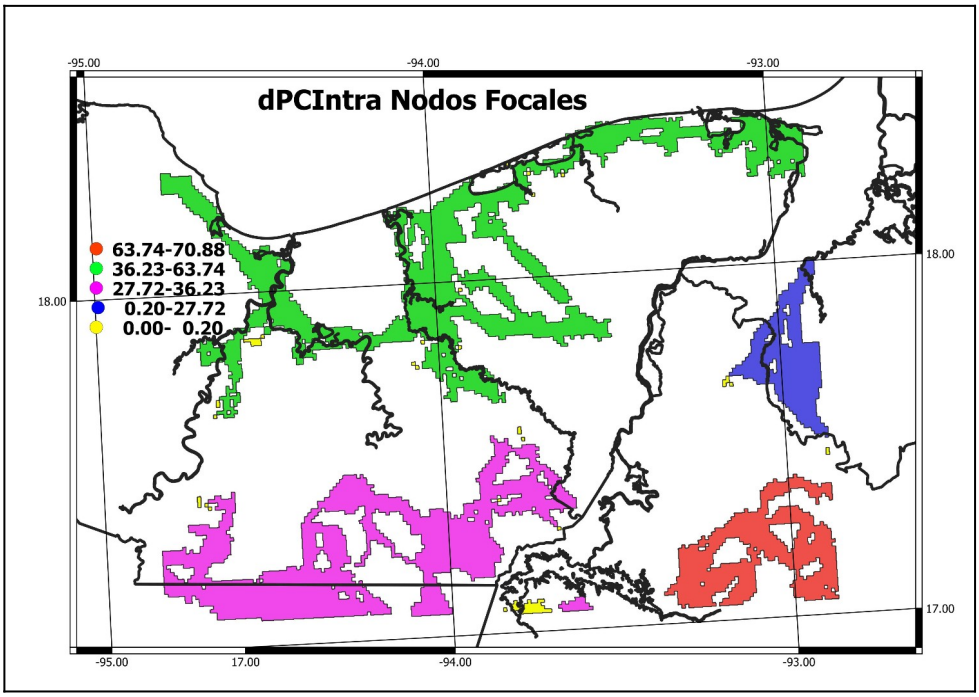




Figura 19: dPCIntra Nodos Focales.

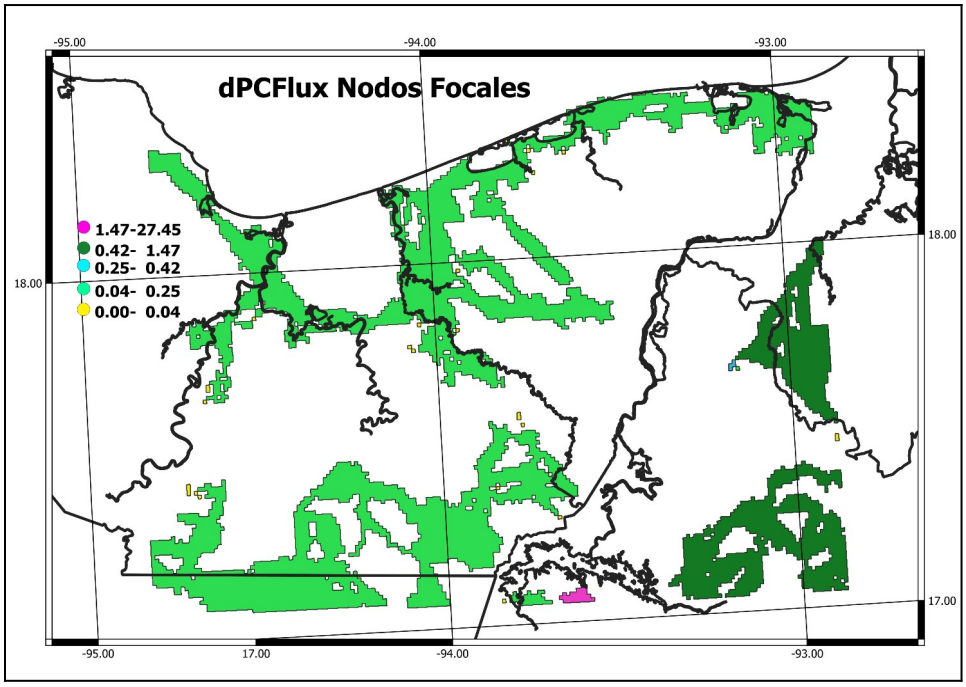



Figura 20: dPCFlux Nodos Focales.

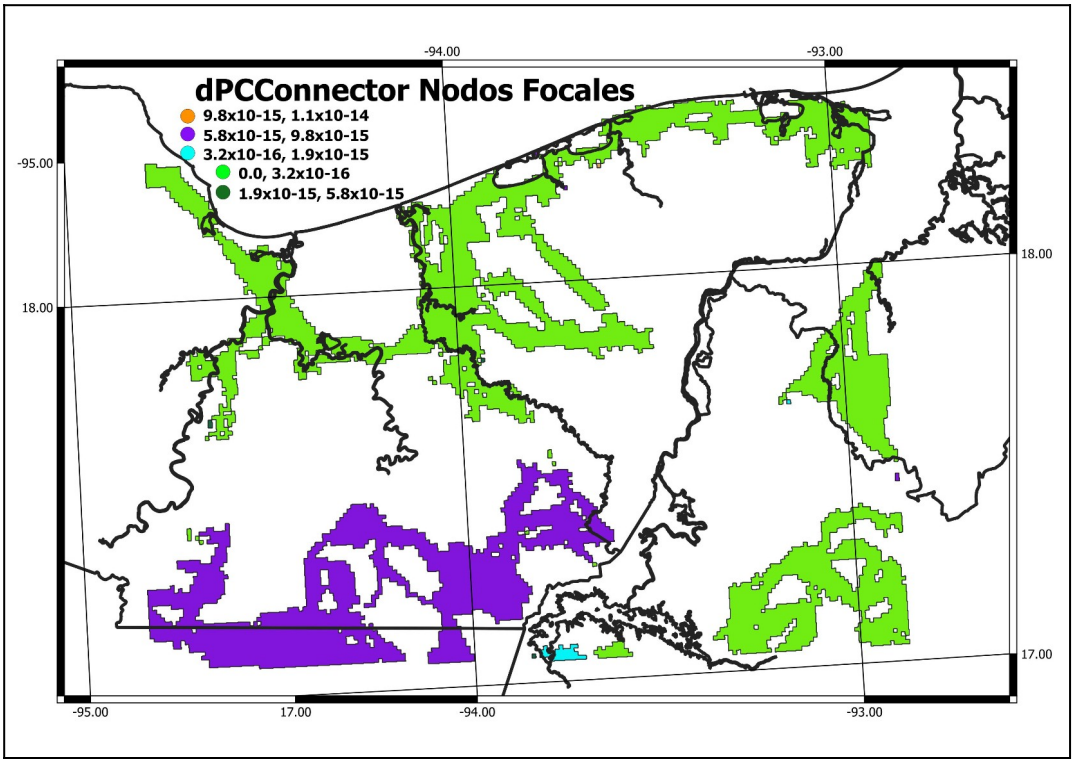



Figura 21: dPCConnector Nodos Focales.

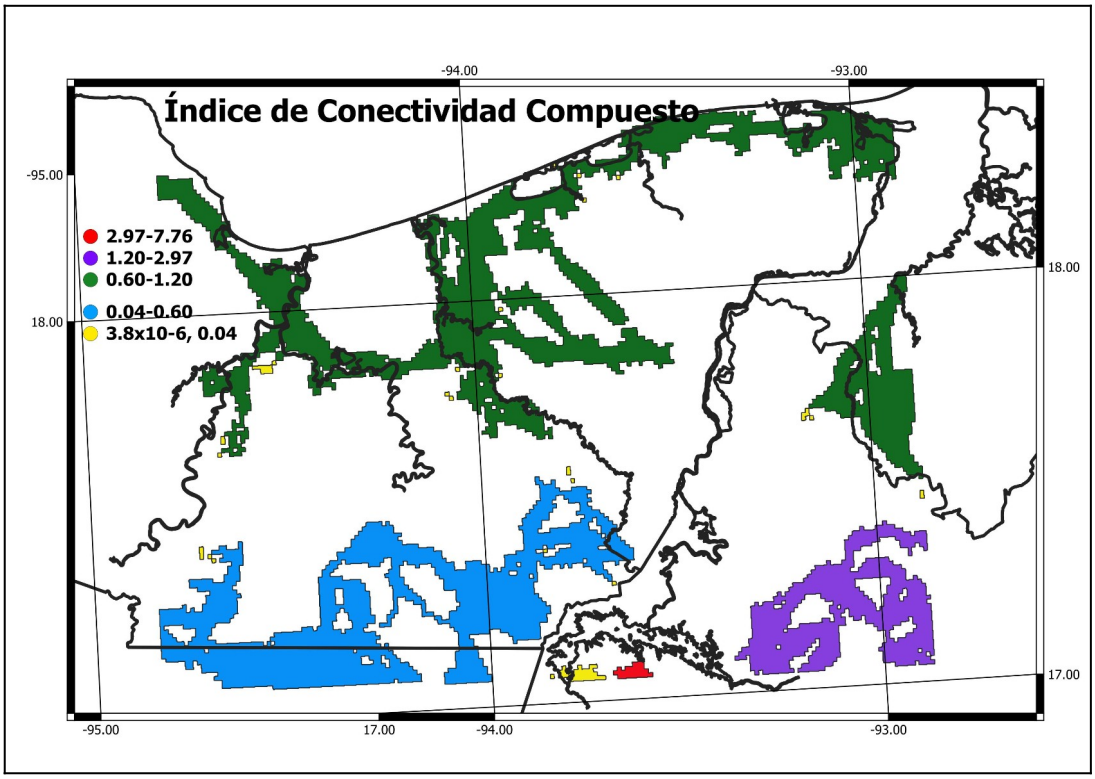



Figura 22: ICCf o CCIf.

Los resultados del análisis focal de los nodos ecológicos arrojan como resultado que el parche ubicado al norte del paisaje (el más grande), los dos parches centrales ubicados al sur del paisaje y el parche localizado al extremos sur-este son parches que presentan según el modelado un buen flujo intra-parche y buen flujo entre parches, por lo que aptos para su conservación, de la figura 22 observamos que dos grupos de parches (color amarillo y color azul celeste) sus valores son menores en comparación con el resto de los parches, estos parches deben ser considerados para su restauración e integración al paisaje para evitar evitar o aminorar la fragmentación en el paisaje.

**Prioridad de Enlaces:**

En el proceso de la generación de enlaces de conexión entre parches ecológicos se utilizó una matriz de resistencia del *impacto humano* en el paisaje, la información se obtuvo de la *CONABIO*, de los 31 parches se seleccionaron aleatoriamente 13 parches, la matriz de resistencia se re-proyecto del esferoide EPSG 4326 al EPSG 6362 y sus valores se ajustaron de 0 a 100.

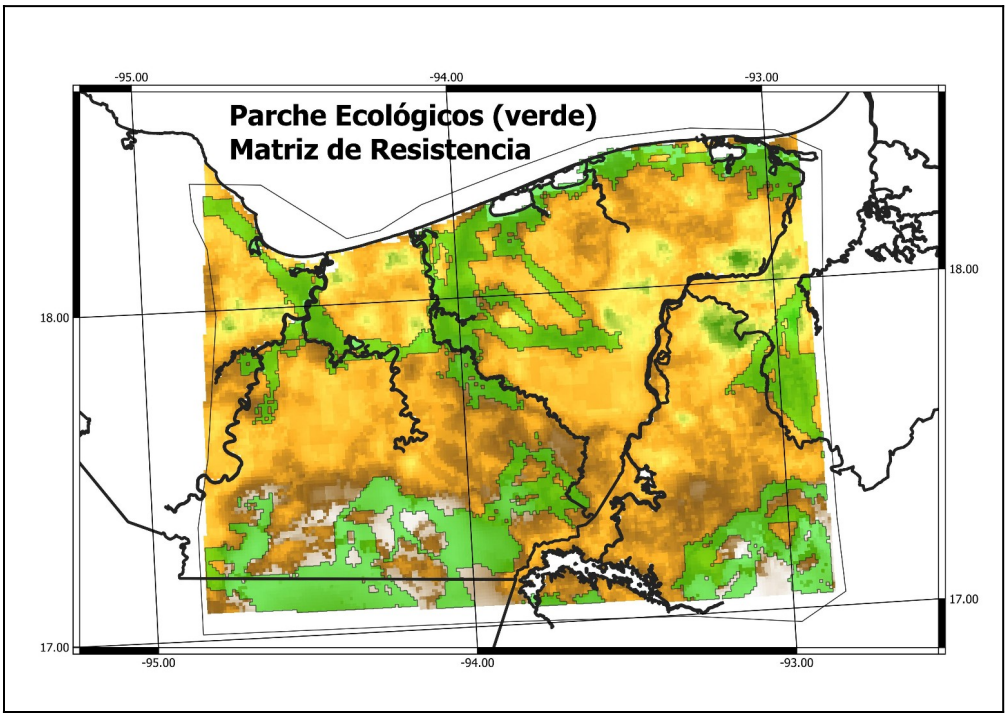



Figura 23: Parches Ecológicos y la Matriz de Resistencia (Impacto Humano).

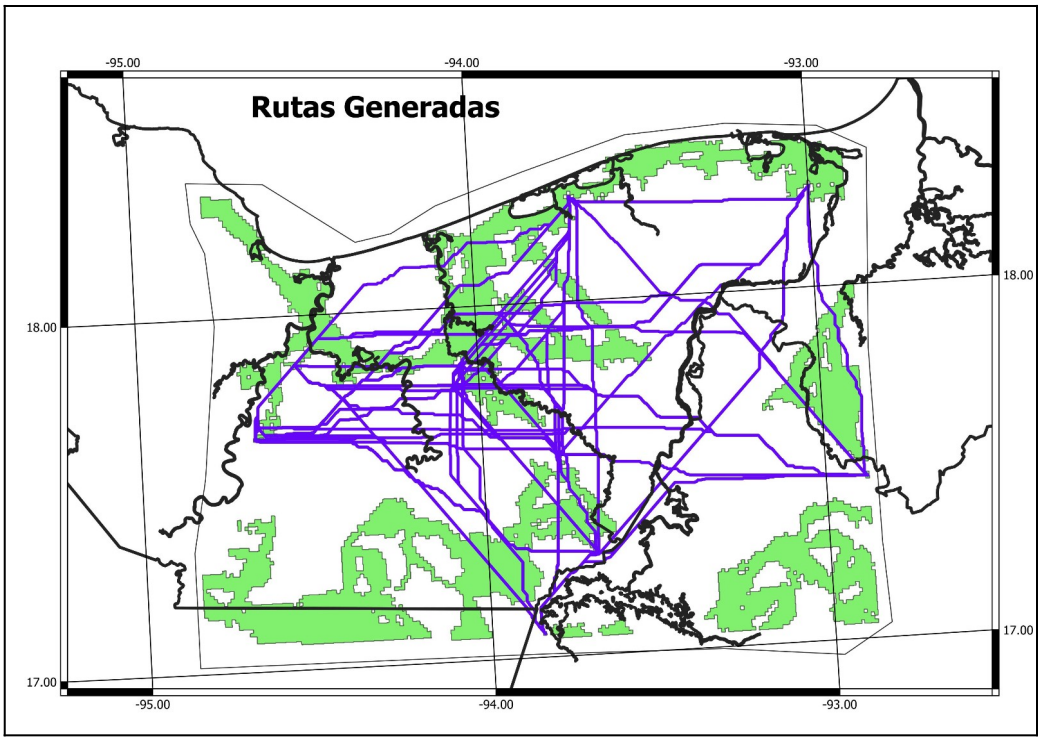



Figura 24: Rutas o enlaces generados.

De los 31 parches originales, 13 se seleccionaron aleatoriamente, generándose 78 rutas de enlace entre los parches.

**Prioridad de enlaces link removal:**

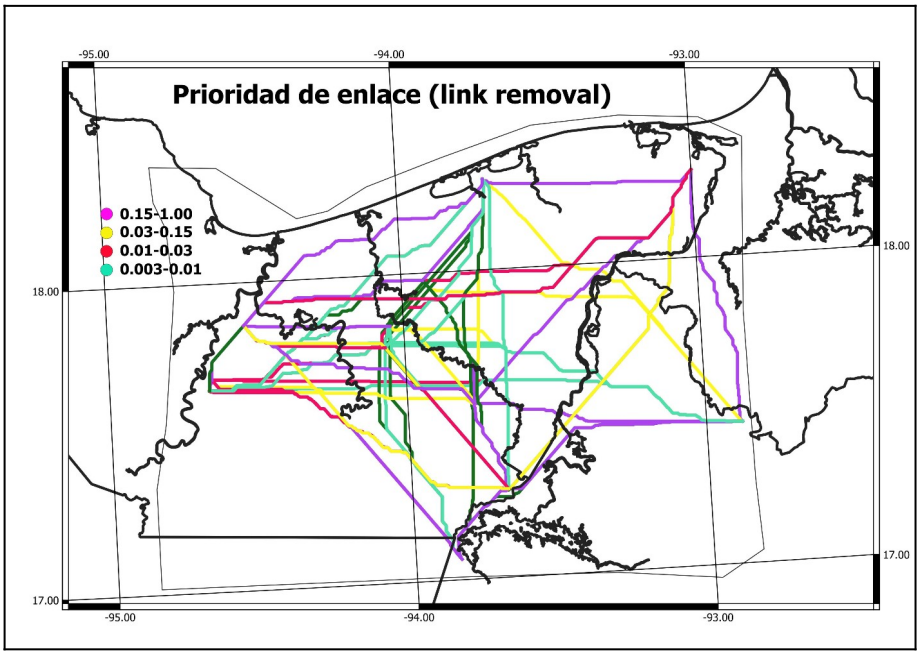



Figura 25: Link Removal

Se calculó usando la opción: “least cost”.

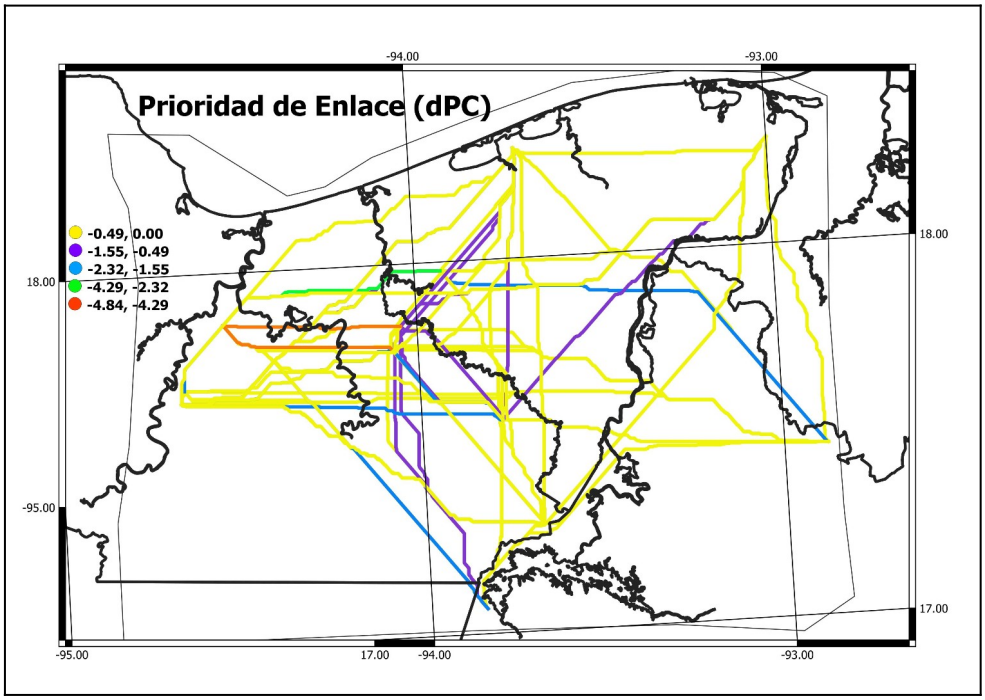



Figura 26: Prioridad de enlace.

En la prioridad de enlace (dPC) se consideraron 50 enlaces y una reducción de la resistencia en el paisaje del 40%.

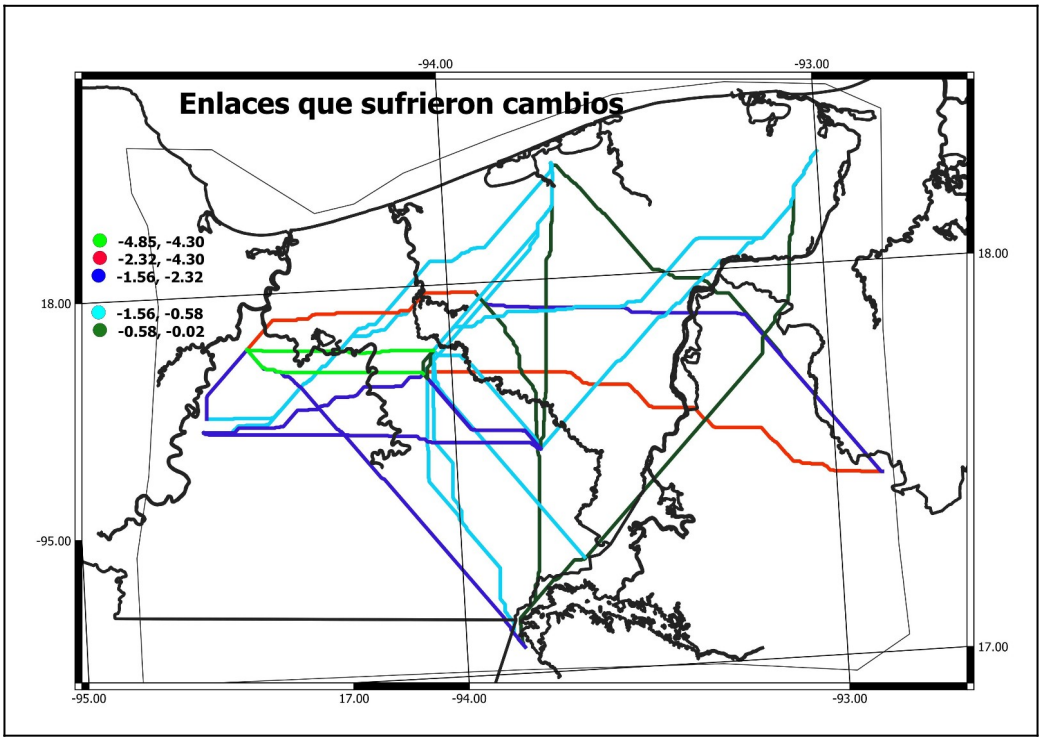



Figura 27: Enlaces que sufrieron cambios.

Los valores positivos se consideran como pérdidas o degradación de enlaces, enlaces prioritarios para conservar son aquellos con los valores positivos más altos. Los valores negativos corresponden a mejoras en los enlaces, y los enlaces prioritarios para restaurar son aquellos con los valores negativos más pequeños.

**ECA:**

El índice ECA (Área de Conectividad Equivalente) se expresa en unidades de área, se puede entender como el área de un parche ecológico que facilita la migración de una especie de un punto A a un punto B por corredores biológicos o rutas de menor costo. Su valor en área no puede ser mayor al área del paisaje ni al área del parche más grande en el paisaje.

Para el cálculo del ECA se continuo utilizando el área del paisaje usada con anterioridad, los parches ecológicos y se implementaron archivos vectoriales SIG del cambio de cobertura arbórea para el tipo de vegetación "Tropical or sub-tropical broadleaf evergreen forest" para los años *2005*, *2008* y *2011* del repositorio de datos de la *CONABIO*. Estos archivos en formato raster se vectorizaron a ESRI Shape File esferoide EPSG 6362.

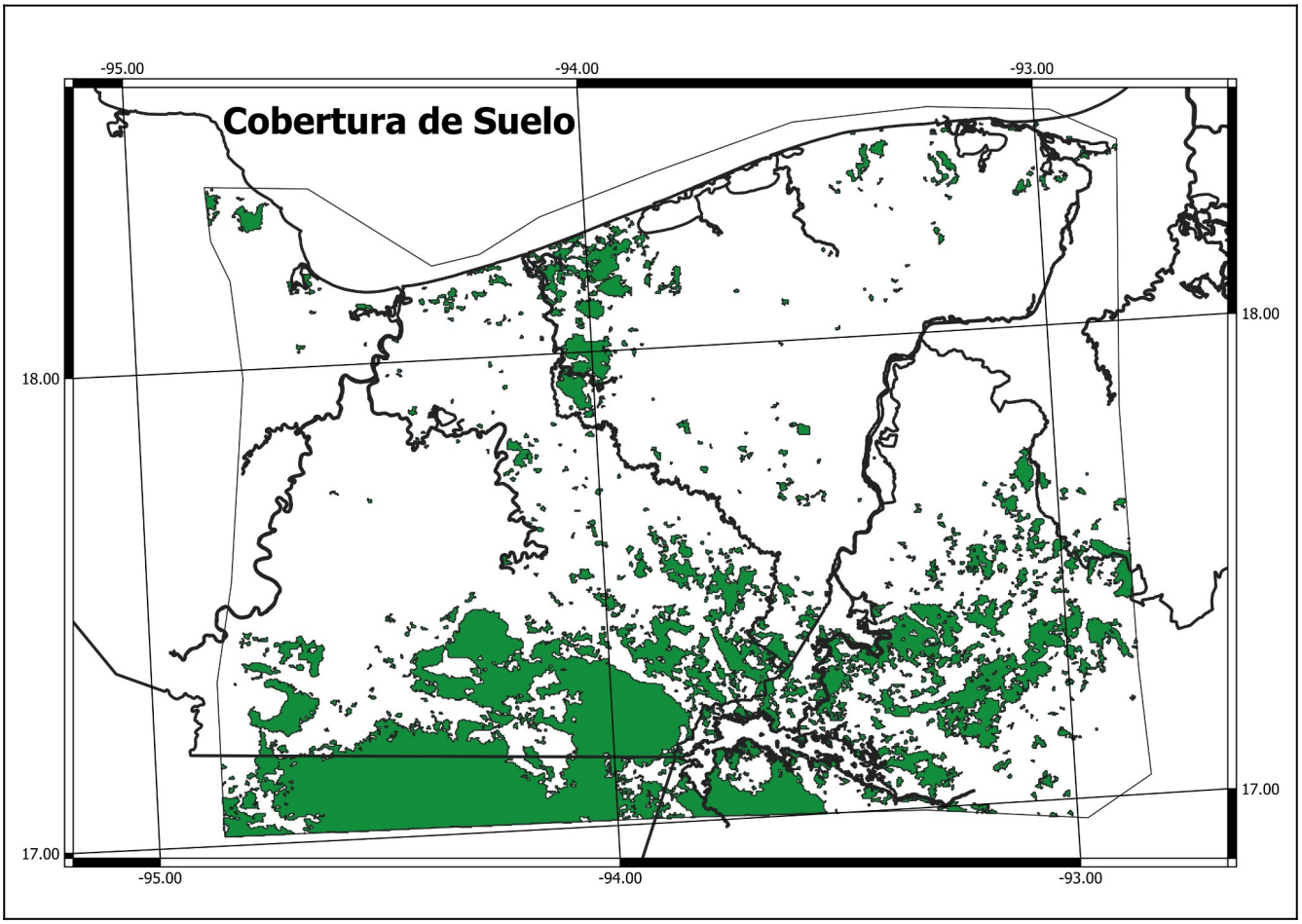



Figura 28: Cobertura arbórea Tropical or sub-tropical broadleaf evergreen forest.

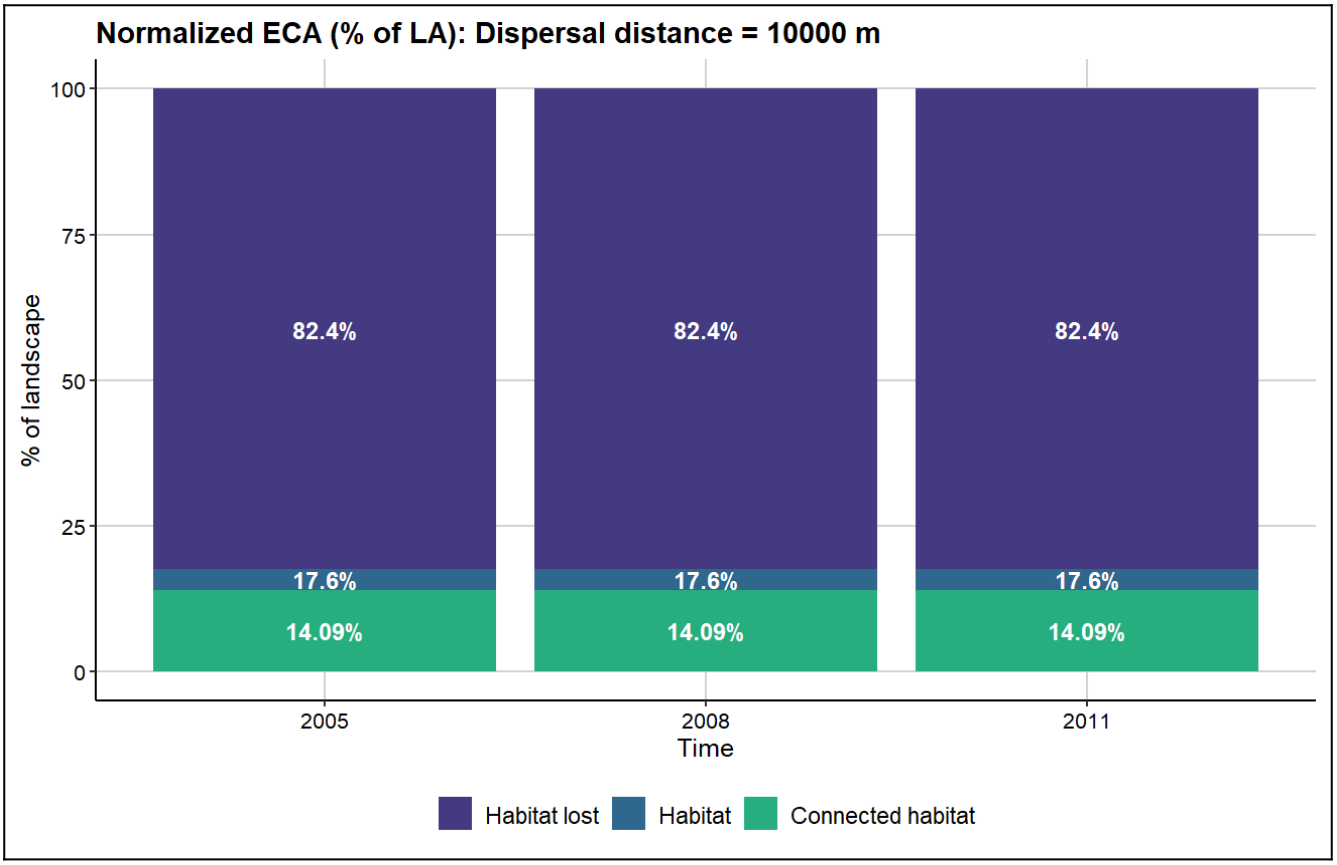



Figura 29: Resultados del ECA en los periodos de 2005, 2008 y 2011.

El ECA normalizado con respecto al área del paisaje fue de 16.37, ECA normalizado con respecto al área de los parches ecológicos 69.56. Se observa en la gráfica 29 que los porcentajes de Habitat y Connected Habitad no han cambiado en los tres periodos de tiempo.

| Time | Max. Landscape attribute (ha) | Habitat area (ha) | Distance threshold | ECA (ha) | Normalized ECA (% of LA) | Normalized ECA (% of habitat area) | dA | dECA | rECA | dA/dECA comparisons | Type of change |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2005 | 3176476 | 559065 | 10000 | 447447 | 14.09 | 80.03 | -82.4 | -85.91 | 1.04 | dECA < dA < 0 | + Connectivity loss |
| 2008 | 3176476 | 559022 | 10000 | 447432 | 14.09 | 80.04 | -0.01 | 0 | 0.43 | dA < dECA < 0 | + Habitat loss |
| 2011 | 3176476 | 559003 | 10000 | 447427 | 14.09 | 80.04 | 0 | 0 | 0.32 | dA < dECA < 0 | + Habitat loss |

Tabla 2: Resultados del ECA.

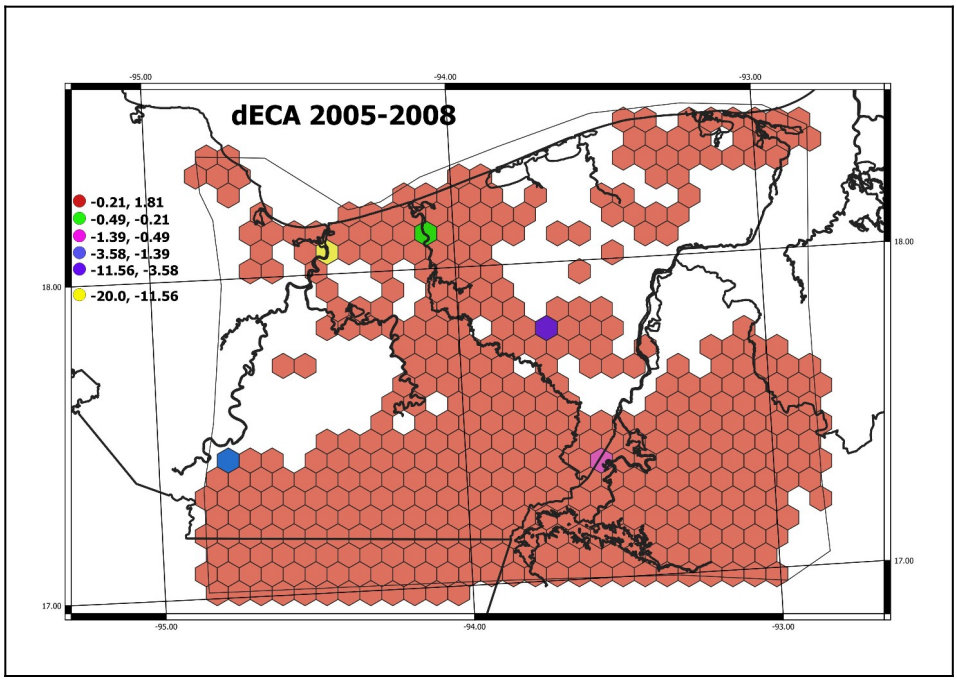



Figura 30: dECA 2005-2008.

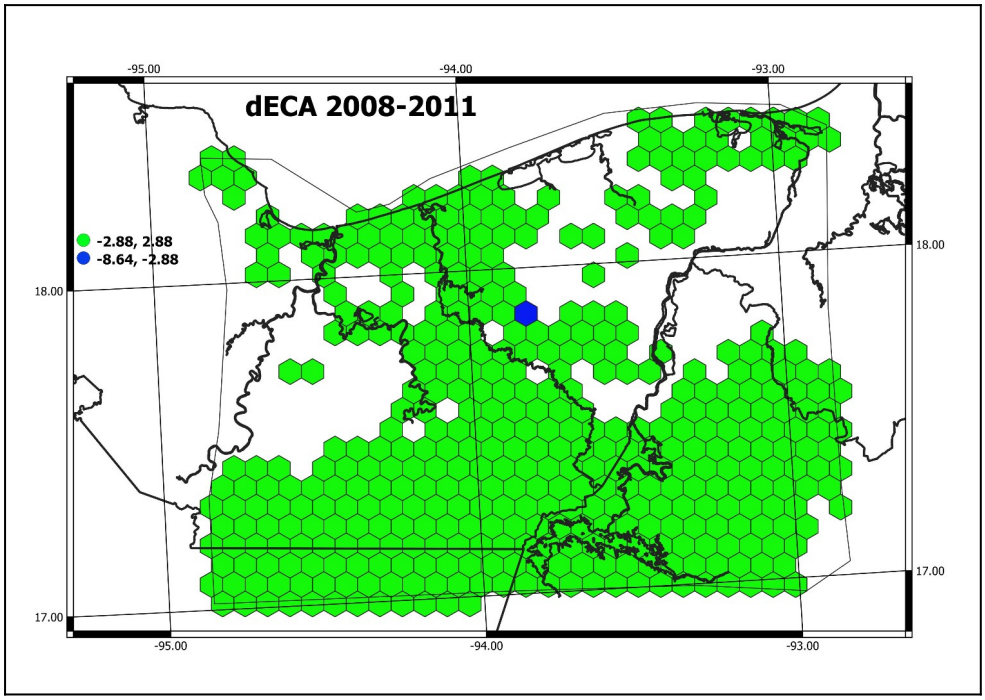



Figura 31 dECA 2008-2011.

El dECA es el cambio porcentual en conectividad entre periodos de tiempo.

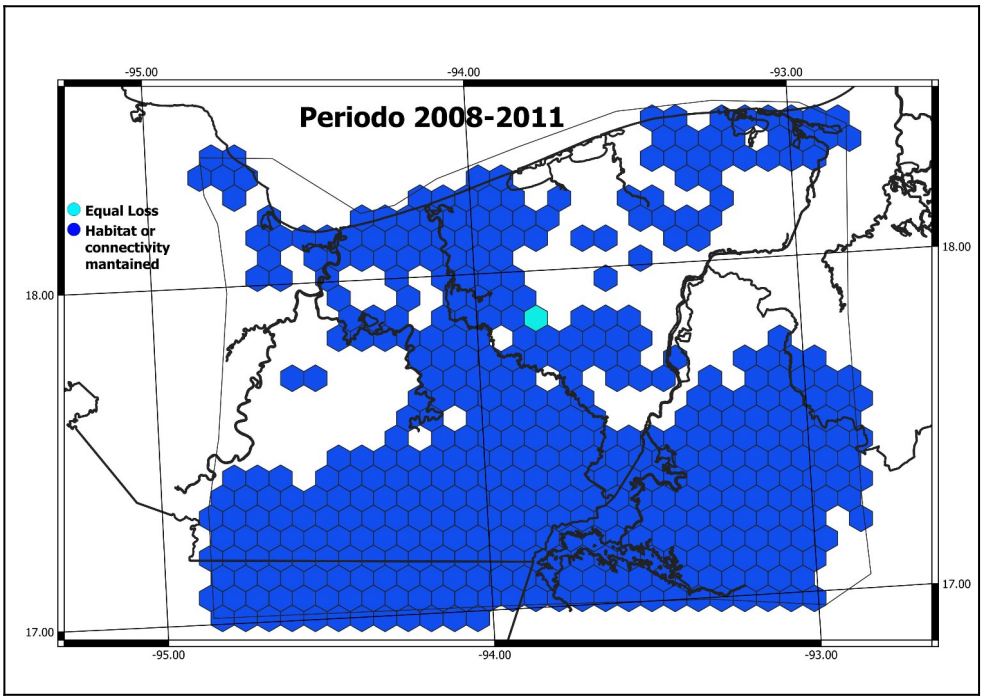



Figura 32: Cambios 2008-2011

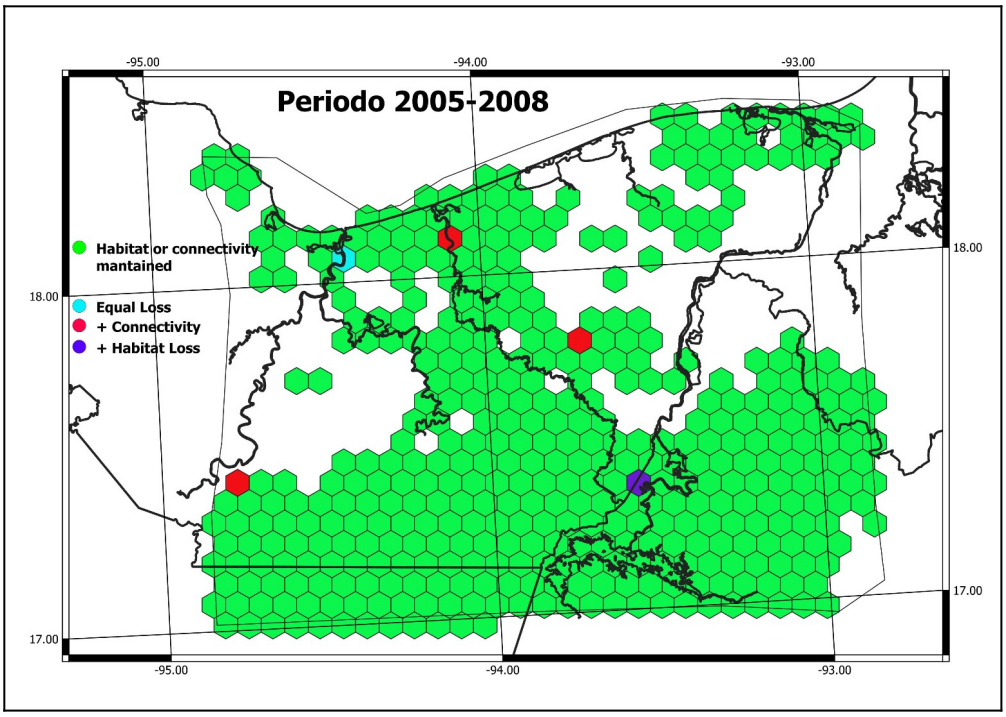



Figura 33: Cambios 2005-2008.

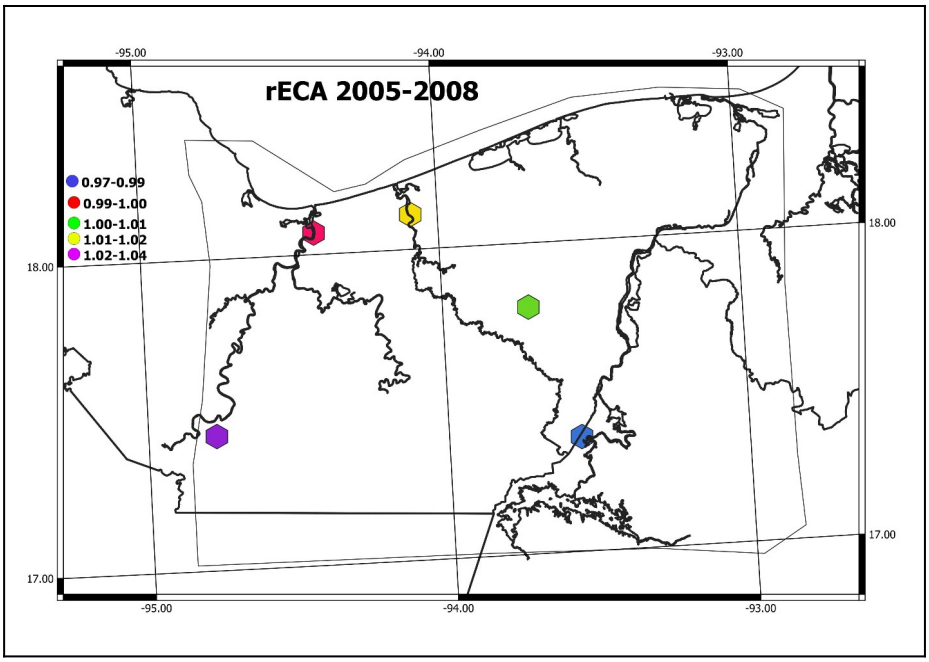



Figura 34:
rECA 2005-2008

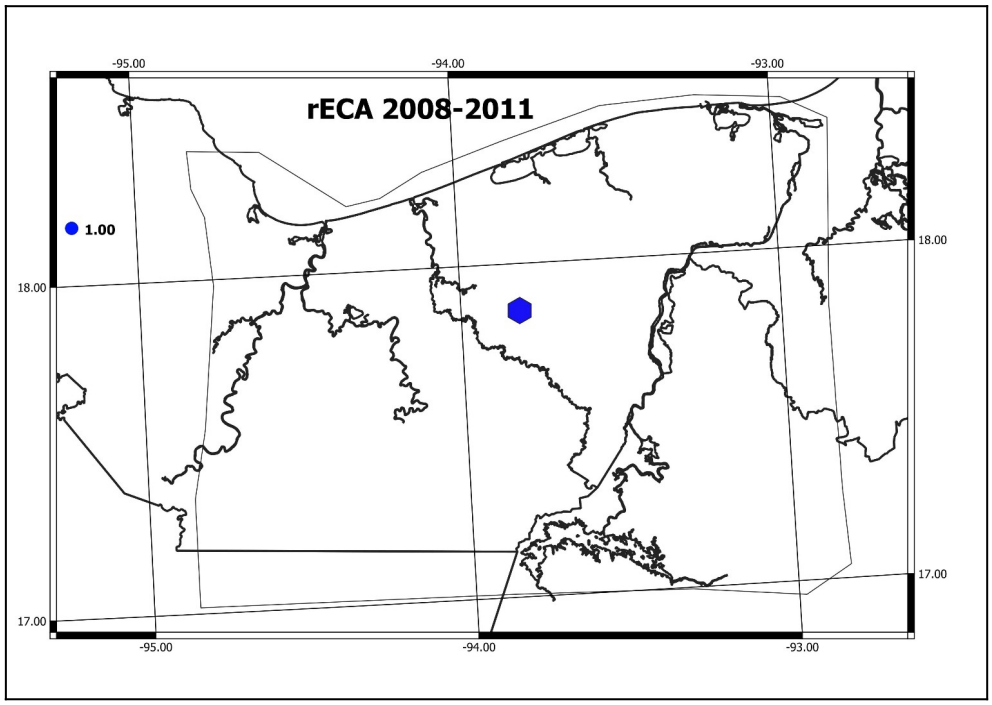



Figura 35: rECA 2008-2011.

Los resultados de la figura 29 y los de la tabla 2 dan como resultado que no se cumple con la meta *Aichi 11* de la Biodiversidad Biológica 2011-2020. Las figuras 32 y 33 muestran sitios con cambios en conectividad y hábitat en el paisaje. La rECA > 1 indica un cambio desproporcionado en la conectividad del hábitat, si es < 1 el cambio de la conectividad se debe a factores aleatorios. Las áreas en blanco en los resultados del dECA y rECA son debidas a que solo se consideró como cobertura arbórea a la vegetación “Tropical or sub-tropical broadleaf evergreen forest”.

**Protected Connected Indicator.**

Para los cálculos se utilizaron las áreas protegidas estatales data 2020 del repositorio de la *CONABIO* y el arreglo de parches ecológicos utilizado en previos cálculos de conectividad.

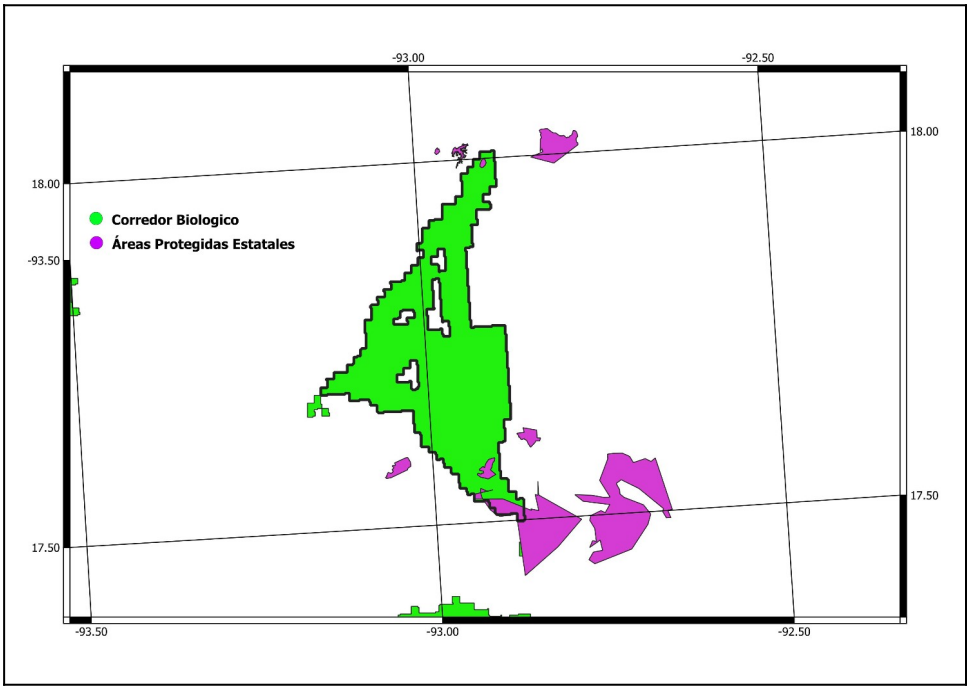



Figura 36: Ecorregión 19.

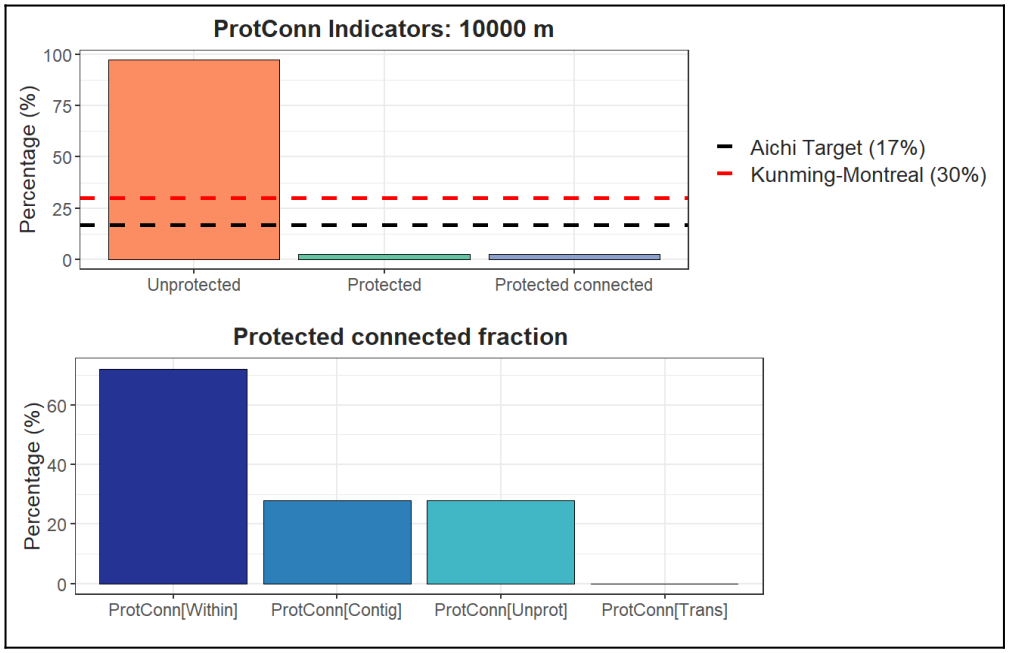



Figura 37: Estimación de ProtConn.

| | Index | Value | ProtConn indicator | Percentage |
|---|---|---|---|---|
| 1 | EC(PC) | 1606.7 | Prot | 2.7 |
| 2 | PC | 0 | Unprotected | 97.3 |
| 3 | Maximum landscape attribute | 63711.77 | ProtConn | 2.52 |
| 4 | Protected surface | 1722.58 | ProtUnconn | 0.18 |
| 5 | | | RelConn | 93.27 |
| 6 | | | ProtConn_Prot | 72.16 |
| 7 | | | ProtConn_Trans | 0.01 |
| 8 | | | ProtConn_Unprot | 27.83 |
| 9 | | | ProtConn_Within | 72.16 |
| 10 | | | ProtConn_Contig | 27.84 |
| 11 | | | ProtConn_Within_land | 1.82 |
| 12 | | | ProtConn_Contig_land | 0.7 |
| 13 | | | ProtConn_Unprot_land | 0.7 |
| 14 | | | ProtConn_Trans_land | 0 |

Tabla 3: Resultados de ProtConn en la ecoregión 19.

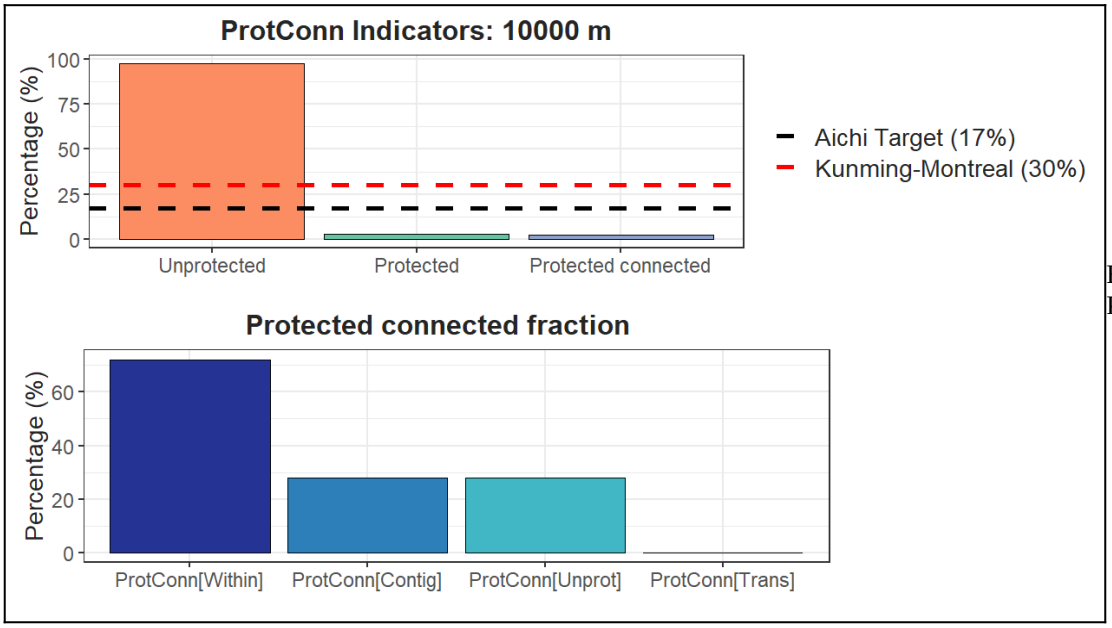



Figura 38: Estimación de ProtConn.

| | Index | Value | ProtConn indicator | Percentage |
|---|---|---|---|---|
| 1 | EC(PC) | 1606.52 | Prot | 2.7 |
| 2 | PC | 0 | Unprotected | 97.3 |
| 3 | Maximum landscape attribute | 63711.77 | ProtConn | 2.52 |
| 4 | Protected surface | 1722.58 | ProtUnconn | 0.18 |
| 5 | | | ProtUnconn_Design | 0.18 |
| 6 | | | ProtConn_Bound | 2.52 |
| 7 | | | RelConn | 93.26 |
| 8 | | | ProtConn_Prot | 72.17 |
| 9 | | | ProtConn_Trans | 0 |
| 10 | | | ProtConn_Unprot | 27.83 |
| 11 | | | ProtConn_Within | 72.17 |
| 12 | | | ProtConn_Contig | 27.83 |
| 13 | | | ProtConn_Within_land | 1.82 |
| 14 | | | ProtConn_Contig_land | 0.7 |
| 15 | | | ProtConn_Unprot_land | 0.7 |
| 16 | | | ProtConn_Trans_land | 0 |

Tabla 4: Resultados de ProtConn en la ecorregión 19, protconnbound = TRUE.

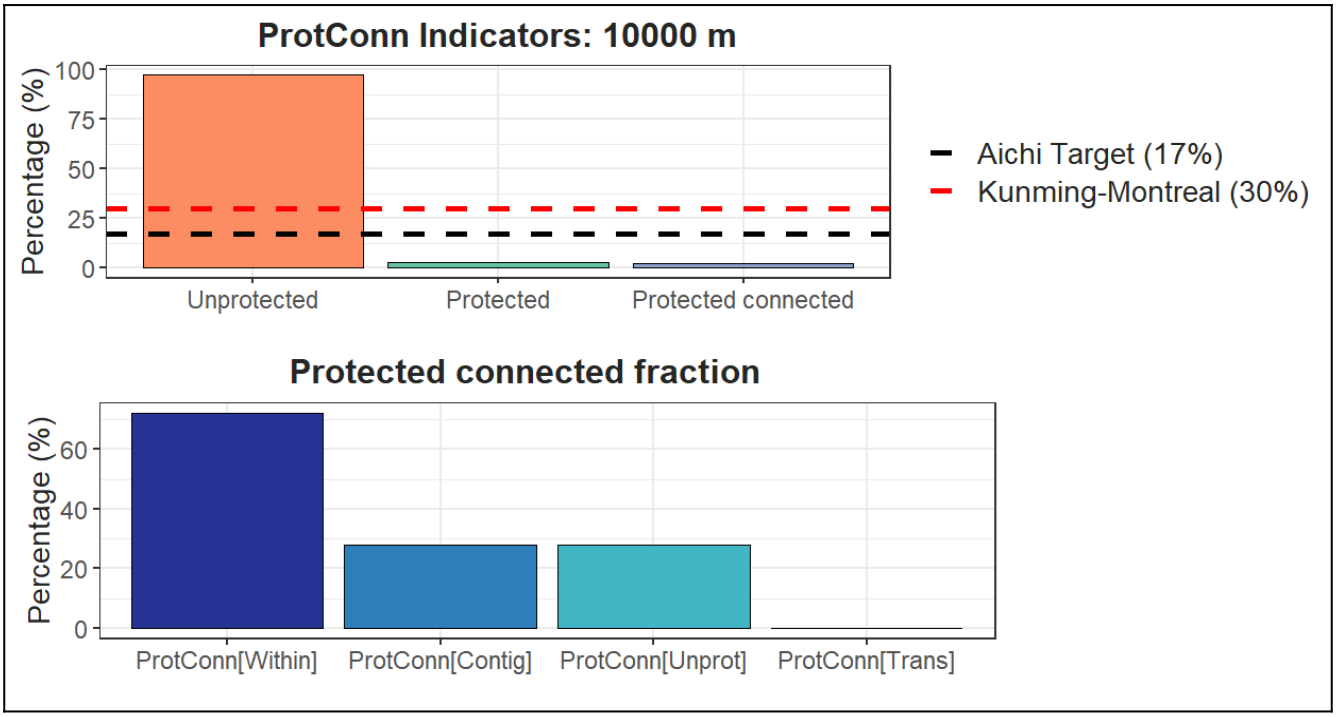



Figura 39: Estimación de ProtConn con la contribución de cada área protegida.

| | Index | Value | ProtConn indicator | Percentage |
|---|---|---|---|---|
| 1 | EC(PC) | 1606.7 | Prot | 2.7 |
| 2 | PC | 0 | Unprotected | 97.3 |
| 3 | Maximum landscape attribute | 63711.77 | ProtConn | 2.52 |
| 4 | Protected surface | 1722.58 | ProtUnconn | 0.18 |
| 5 | | | RelConn | 93.27 |
| 6 | | | ProtConn_Prot | 72.16 |
| 7 | | | ProtConn_Trans | 0.01 |
| 8 | | | ProtConn_Unprot | 27.83 |
| 9 | | | ProtConn_Within | 72.16 |
| 10 | | | ProtConn_Contig | 27.84 |
| 11 | | | ProtConn_Within_land | 1.82 |
| 12 | | | ProtConn_Contig_land | 0.7 |
| 13 | | | ProtConn_Unprot_land | 0.7 |
| 14 | | | ProtConn_Trans_land | 0 |

Tabla 5: Resultados de ProtConn con la contribución de cada área protegida.

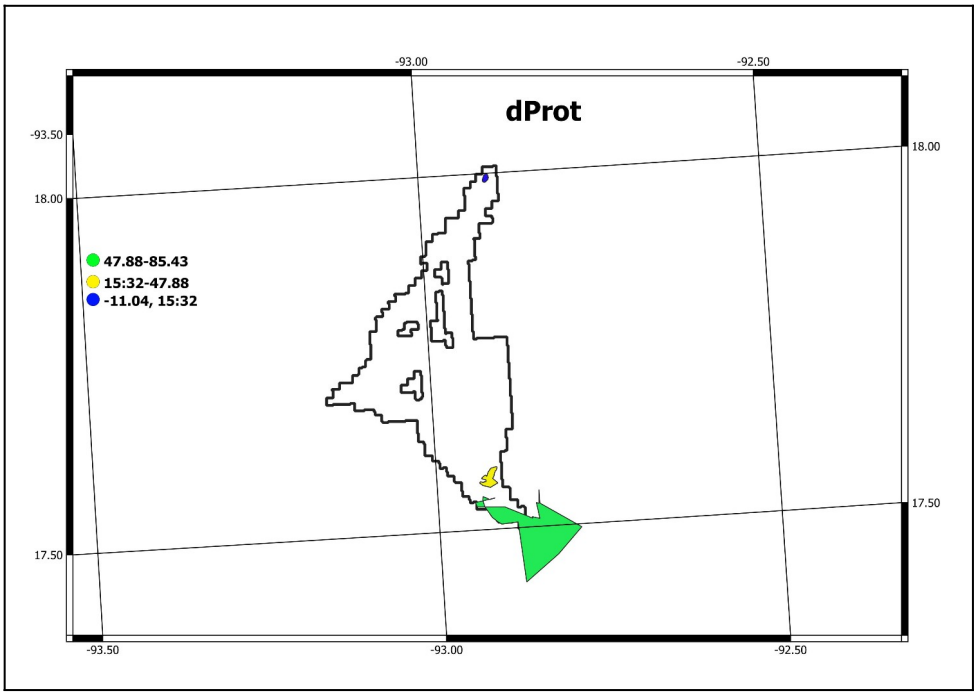



Figura 40: delta protegido.

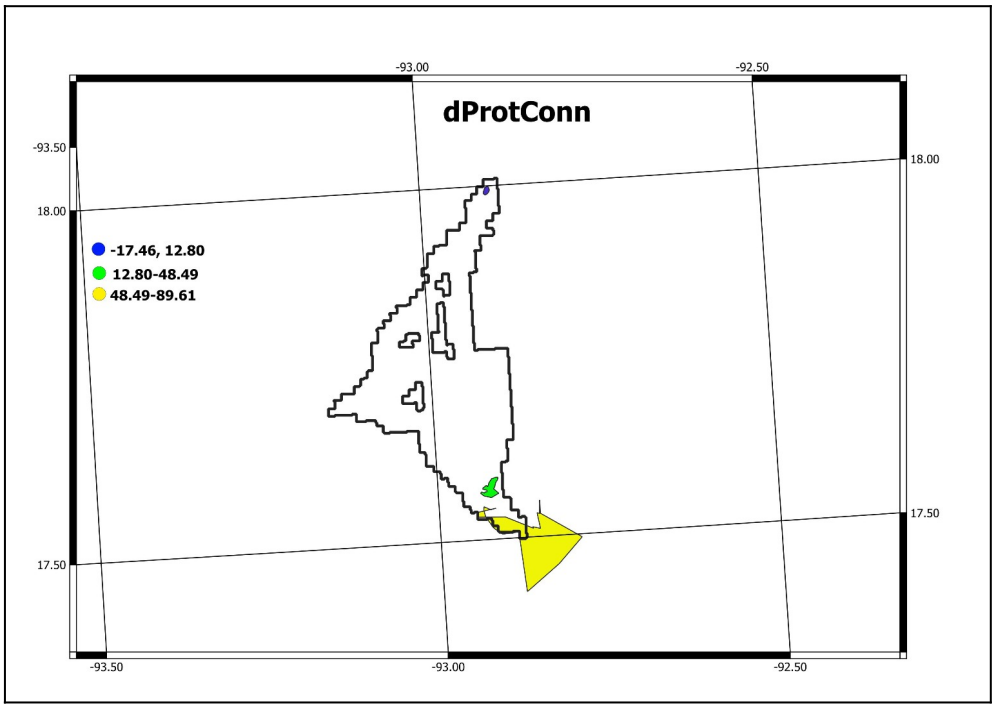



Figura 41: delta ProtConn.

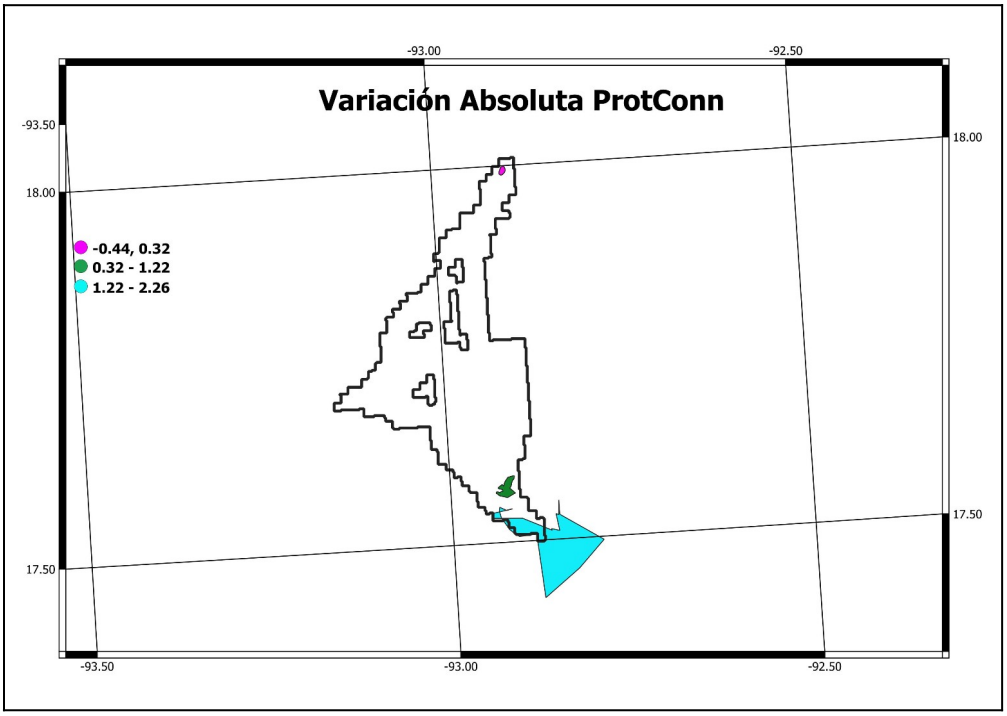



Figura 42: variación absoluta protconn.

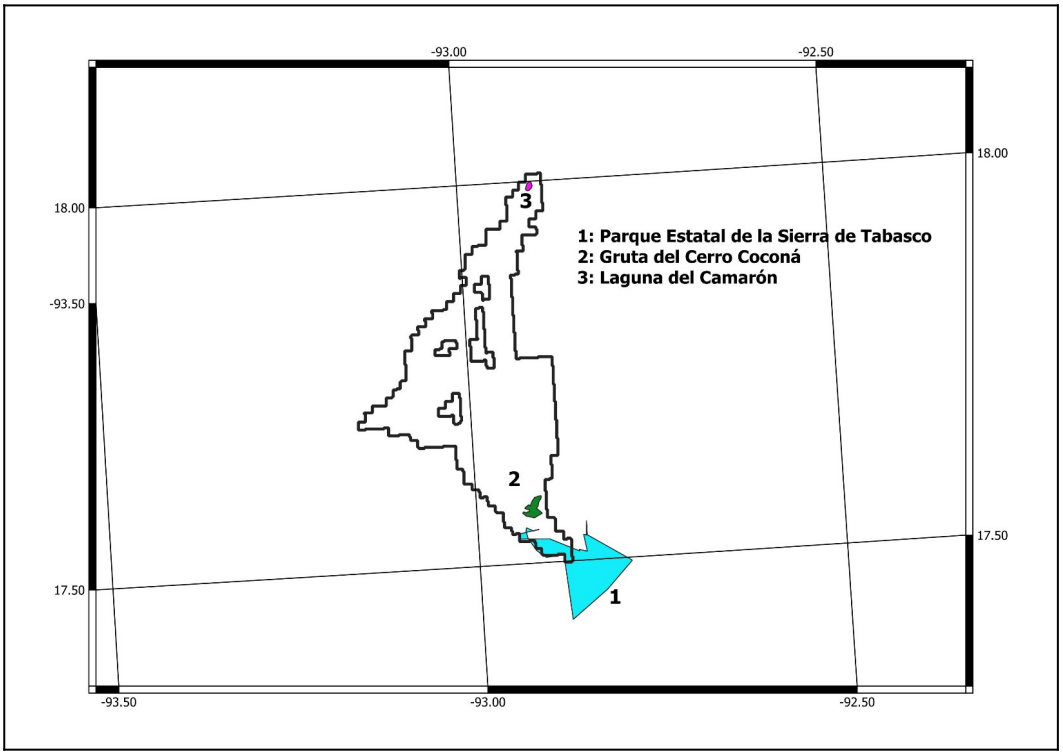



Figura 43: áreas protegidas seleccionadas en la ecorregión 19.

Los resultados en las tablas 3. 4. y 5 muestran que el ECA (EC/PC) tiene un valor de 1606 ha, el máximo atributo del paisaje (ecorregión 19) con 66711.77 ha y el área protegida es de 1722.58 ha, el porcentaje de áreas protegidas con respecto a la ecorregión es del 2.52%, 2.7% del paisaje está protegido por las áreas protegidas, un 0.18 % del área del paisaje o región no se encuentra protegida ni conectada. La conectividad relativa es de 93.26% referida del área protegida y del área protegida con conectividad. Del las área protegidas y conectadas un 72.16% del valor de protegido y conectado corresponde a flujo intra o dentro de la misma área protegida y un 27.84 corresponde a flujo de especies entre áreas protegidas contiguas. El protconn trans es de 0.01% que corresponde a áreas protegidas que se ubican en el buffer alrededor del paisaje o ecorregión.

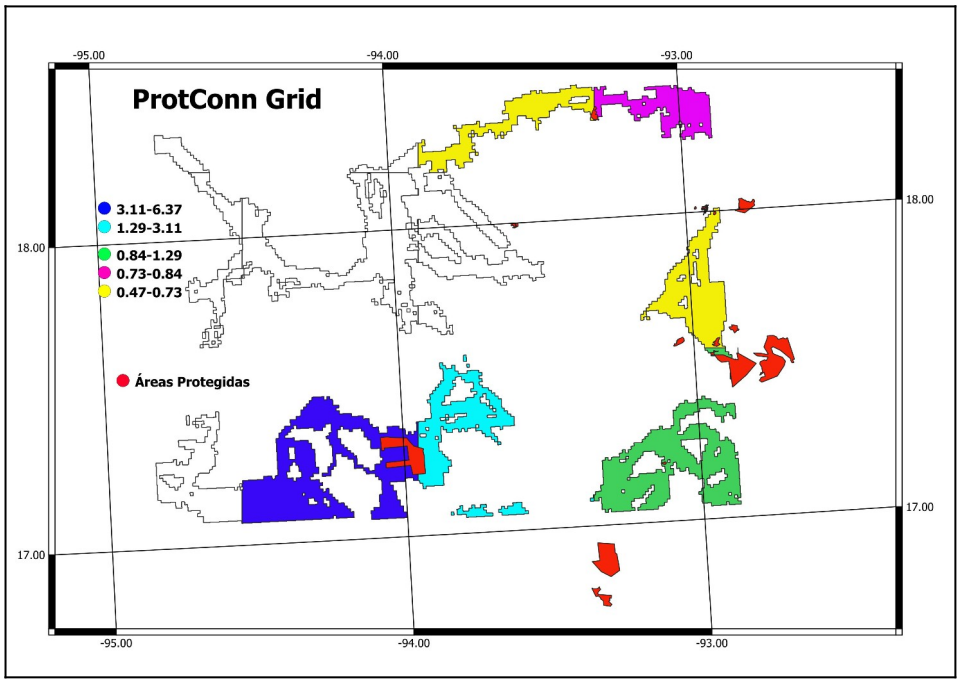



Figura 44: Grid ProtConn.

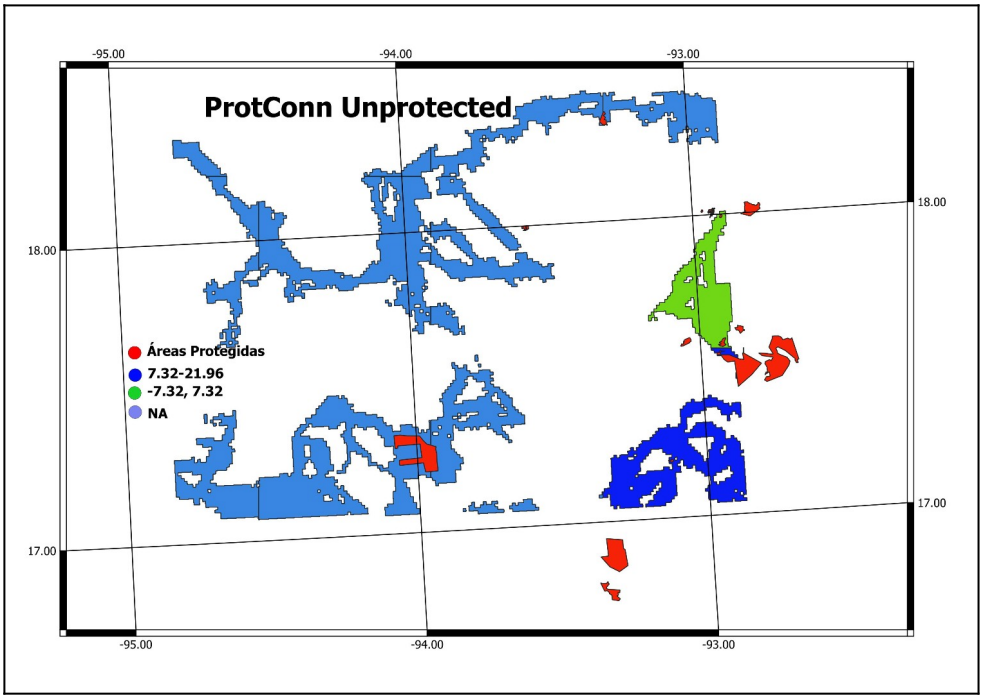



Figura 45: Grid ProtConn Unprotected.

Los resultados en las figuras 44 muestran a nivel de paisaje el aporte de las áreas protegidas en dar conectividad a paisaje y permitan además del flujo intra, el flujo con áreas contiguas y áreas fuera de los límites del paisaje. Las áreas en magenta y en amarillo requieren la atención para su cuidado evitando su fragmentación y afectación en el paisaje. En la figura 45 observamos que existen en el paisaje áreas que no presentan protección por el análisis de el peso de las áreas protegidas existentes en el paisaje, como vemos en la figura 44 en áreas de nodos que no tiene datos disponibles.

Los diferentes procesos de análisis de nodos ecológicos en el paisaje permiten hacer cálculos desde su fragmentación simulada, índice de probabilidad de conexión, nodos focales de dispersión, el DECA y ProtConn. Se analizan varias alternativas de entender que factores en archivos electrónicos pueden influir en la conexión de nodos ecológicos en un entorno o paisaje, permitiendo ver desde diferentes perspectivas de análisis de información SIG puntos de interés a conservar, nodos ecológicos a cuidar, nodos ecológicos a restaurar o insertar en el paisaje, rutas de migración o movimiento de especies, rutas de menor costo al ejemplar, incluyendo los cálculos para estimar que si los nodos ecológicos en el paisaje cumplen con las metas de Aichi o de Kunming-Montreal referentes a la preservación y restauración del ecosistema.

**Referencias:**


CONABIO: http://www.conabio.gob.mx/informacion/gis/

Conicet: Landscape connectivity and the role of small habitat patches as stepping stones, an assessments of the grassland biome in South America. https://ri.conicet.gov.ar/bitstream/handle/11336/72784/CONICET_Digital_Nro.9ea817b4-2ba1-4fe8-8f46-41393cef2d69_A.pdf?sequence=2&isAllowed=y

Convention on Biological Diversity, https://www.cbd.int/sp/targets

Convention on Biological Diversity, https://www.cbd.int/gbf/targets

iies unam, https://www.iies.unam.mx/laboratorios/modelacion-sistemas-socioambientales/pe200623/PE200623_ConectPaisaje2023b.pdf

## Script en R:

```
library(Makurhini)
library(sf)

habitat_nodes <- read_sf("cobio_lcc/vector.shp")
nrow(habitat_nodes)
paisaje <- read_sf("paisaje/paisa.shp")
nrow(paisaje)

library(ggplot2)
library(RColorBrewer)
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = habitat_nodes, aes(color = "Parches"), fill = "forestgreen", linewidth = 0.5) +
  scale_color_manual(name = "", values = "black")+
  theme_minimal() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())
area_paisaje <- st_area(paisaje)
area_paisaje <- unit_convert(area_paisaje, "m2", "ha")


# Aplicamos la funcion MK_Fragmentation

Fragmentacion <- MK_Fragmentation(nodes = habitat_nodes,
                                  edge_distance = 500,
                                  #edge_distance = 500
                                  min_node_area = 100,
                                  #min_node_area
                                  landscape_area = area_paisaje,
                                  area_unit = "ha",
                                  perimeter_unit = "km",
                                  #write = "frag01/Fragmenta01/MK_Fragmentacion"
                                  #write =NULL
                                  , plot = TRUE)

#class(Fragmentacion)

list(Fragmentacion)

names(Fragmentacion)

# Estadisticas a Nivel Paisaje
Fragmentacion$`Summary landscape metrics (Viewer Panel)` #  Fragmentacion[1]
write.csv(Fragmentacion[1], "1fragmentacion/frag.csv")

# Estadisticas a Nivel de cada parche
Fragmentacion$`Patch statistics shapefile`
write.csv(Fragmentacion[2], "1fragmentacion/fragparche.csv")

# Grafico % del Area nucleo

ggplot() +
```

```
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Fragmentacion$`Patch statistics shapefile`, aes(fill = CAPercent), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlGn",
    direction = 1,
    name = "% Área Núcleo"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "% Área Núcleo"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)

st_write(Fragmentacion$`Patch statistics shapefile`,
         "1fragmentacion/shp/fragmenta1.shp", driver = "ESRI Shapefile")

# % Del Borde
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Fragmentacion$`Patch statistics shapefile`, aes(fill = EdgePercent), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlGn",
    direction = -1,
    name = "% Borde"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "% Borde"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
# Perimetro
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Fragmentacion$`Patch statistics shapefile`, aes(fill = Perimeter), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlGn",
    direction = -1,
    name = "Perímetro"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "Perímetro"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```
# Razon Perimetro - Area
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Fragmentacion$`Patch statistics shapefile`, aes(fill = PARA), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlGn",
    direction = -1,
    name = "PARA"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "PARA"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)

# Shape Index
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Fragmentacion$`Patch statistics shapefile`, aes(fill = ShapeIndex), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "PiYG",
    direction = -1,
    name = "Shape Index"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "Shape Index"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)

# Fractal Dimension Index
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Fragmentacion$`Patch statistics shapefile`, aes(fill = FRAC), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "PiYG",
    direction = -1,
    name = "FRAC"
  ) +
  theme_minimal() +
  labs(
    title = "Fragmentación a nivel de parche",
    fill = "FRAC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```
mesh <- as.data.frame(Fragmentacion[[1]])
m1 <- mesh
mesh <- mesh[13,2]
# area_paisaje, mesh en ha
area_paisaje
mesh_porcentage <-  (area_paisaje - mesh) * 100 / area_paisaje
mesh_porcentage

# Exploramos distintas distancias de profundidad del efecto borde
# Loop edge distance
# en este ejemplo de 100m a 1000m incrementos 100m

library(purrr)
Fragmentacion.2 <- map_dfr(seq(100, 1000, 100), function(x){
  x.1 <- MK_Fragmentation(nodes = habitat_nodes,
                          edge_distance = x, plot = FALSE)[[2]]
  CA <- mean(x.1$CAPercent)
  Edge <- mean(x.1$EdgePercent)
  x.2 <- rbind(data.frame('Edge distance' = x, Type = "Core Area", Percentage = CA),
               data.frame('Edge distance' = x, Type = "Edge", Percentage = Edge))
  return(x.2)
}, .progress = TRUE)


Fragmentacion.2

# gráficar el promedio del porcentaje de área nucleo de los parches y el porcentaje
# del borde de los parches (%área núcleo + % de borde = 100%)

library(ggplot2)
ggplot(Fragmentacion.2, aes(x=Edge.distance, y=Percentage, group=Type)) +
  geom_line(aes(color=Type))+
  geom_point(aes(color=Type))+ ylim(0,100)+
        scale_x_continuous("Distancia", labels = as.character(Fragmentacion.2$Edge.distance), breaks =
Fragmentacion.2$Edge.distance)+
  scale_color_brewer(palette="Dark2")+
  theme_classic()

# Alcances de la función estimando el índice MESH sobre un grid.
# Grid de 40 km2

Grid_test <- make_grid(x = paisaje, hexagonal = TRUE,
                       cell_area = unit_convert(40, "km2", "m2"),
                       clip = TRUE)
plot(Grid_test)
grid(nx = NULL, ny = NULL,
     lty = 2,      # Grid line type
     col = "gray", # Grid line color
     lwd = 2)      # Grid line width

# Exportar a shp file
write_sf(Grid_test, "frag01/GridMesh/Gridmesh.shp")

# Estimar MESH con un ciclo for i to
#Variable dummy
Grid_test$MESH <- 0
```

```
Grid_test
nrow(Grid_test)

for(i in 1:nrow(Grid_test)){
  cat(paste0(i, " de ", nrow(Grid_test), "\r"))
  grid.i <- Grid_test[i,]
  nodes.i <- suppressWarnings(st_intersection(habitat_nodes, grid.i))

  if(nrow(nodes.i) > 0){
    area_paisaje.i <- st_area(grid.i)
    area_paisaje.i <- unit_convert(area_paisaje.i, "m2", "ha")
    Fragmentacion.i <- MK_Fragmentation(nodes = nodes.i,
                                        edge_distance = 500,
                                        min_node_area = 100,
                                        landscape_area = area_paisaje.i,
                                        area_unit = "ha",
                                        perimeter_unit = "km",
                                        plot = FALSE)
    mesh <- as.data.frame(Fragmentacion.i[[1]])
    mesh <- mesh[13,2]
    mesh_porcentage <-  (area_paisaje.i - mesh)*100/area_paisaje.i
    Grid_test$MESH[i] <- mesh_porcentage
  } else {
    Grid_test$MESH[i] <- 100
  }
}

# Graficamos
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = Grid_test, aes(fill = MESH), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlGn",
    direction = -1,
    name = "% Fragmentación"
  ) +
  theme_minimal() +
  labs(
    title = "GRID fragmentación (MESH)",
    fill = "% Fragmentación"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Exportamos a shp file
write_sf(Grid_test, "1fragmentacion/mesh/mesh.shp")

# ----------------------------------------------------------------------
# Función RMKCentrality
# ----------------------------------------------------------------------
# Librerias a usar

library(ggplot2)
library(sf)
```

```
library(Makurhini)
library(RColorBrewer)

# Importacion de los datos a procesar

habitat_nodes <- read_sf("Datos1/Parches.shp")
paisaje <- read_sf("Datos1/Paisaje.shp")
nrow(habitat_nodes)

# Graficado de los datos

ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = habitat_nodes, aes(color = "Parches"), fill = "forestgreen", linewidth = 0.5) +
  scale_color_manual(name = "", values = "black")+
  theme_minimal() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())

# ------------------------------------------------------------------
# Funcion MK_RMCentrality ---- NO ejecutar
#MK_RMCentrality(
#  nodes,
#  distance = list(type = "centroid"),
#  distance_thresholds = NULL, in m
#  binary = TRUE,
#  probability = NULL, a real value
#  rasterparallel = FALSE,
#  write = NULL,
#  intern = TRUE
#)
# ------------------------------------------------------------------

#Two distance threshold,
centrality_test <- MK_RMCentrality(nodes = habitat_nodes,
                                   distance = list(type = "centroid"),
                                   distance_thresholds = 10000,
                                   probability = 0.5,
                                   #write = "frag01/MKCentrality/mkcentral"
                                   ) #NULL)

centrality_test

# Se aplican otras librerias para graficado

library(classInt)
library(dplyr)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(centrality_test$strength, n = 9,
                                   style = "quantile")

# Crear una nueva variable categórica con los intervalos
# Se escoje la variable strength del archivo resultante de MKCentrality

centrality_test <- centrality_test %>%
```

```
  mutate(strength_q = cut(strength,
                          breaks = breaks$brks,
                          include.lowest = TRUE,
                          dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = centrality_test, aes(fill = strength_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "Fuerza (Q)") +
  theme_minimal() +
  labs(
    title = "Centralidad a nivel de parche (Strength)",
    fill = "Strength\n(Jenks)"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Exportando grafico en driver = geopackage
write_sf(centrality_test,"1fragmentacion/centrality/strength.shp")

# BWC que indica la presencia de steping stones o
# parche que ayudan a al coectividad entre otros parches mayores

breaks <- classInt::classIntervals(centrality_test$BWC, n = 9, style = "jenks")
centrality_test <- centrality_test %>%
  mutate(BWC_jenks = cut(BWC,
                         breaks = breaks$brks,
                         include.lowest = TRUE,
                         dig.lab = 5))
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = centrality_test, aes(fill = BWC_jenks), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "BWC (Jenks)") +
  theme_minimal() +
  labs(
    title = "Centralidad a nivel de parche (BWC)",
    fill = "BWC\n(Jenks)"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Exportando grafico en driver = geopackage
write_sf(centrality_test,"1fragmentacion/bwc/bwc.shp")

# Random Walk agrupa en un paisaje los nodos existentes
# Agrupa nodos según la probabilidad de que una caminata
#       aleatoria permanezca dentro del grupo
# Simula un animal moviéndose aleatoriamente por la red.
# Detecta grupos fuertemente conectados

ggplot() +
```

```
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = centrality_test, aes(fill = as.factor(memb.rw)), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "Set3",
    name = "Membership RW"
  ) +
  theme_minimal() +
  labs(
    title = "Agrupación de parches (Random walks)",
    fill = "Membership RW"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Exportando grafico en driver = geopackage
# write_sf(centrality_test,"frag01/MKCentrality/rnd_walks.gpkg")

# Membresia Louvain
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = centrality_test, aes(fill = as.factor(memb.louvain)), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "Set3",
    name = "Membership LV"
  ) +
  theme_minimal() +
  labs(
    title = "Agrupación de parches (Louvain)",
    fill = "Membership LV"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# Indice Integral de Conectividad y Probabilidad de Conectividad

# Paquetes necesarios
library(ggplot2)
library(sf)
library(Makurhini)
library(RColorBrewer)

# Cargamos los datos de paisaje y nodos, en este caso son del edo de Chiapas
habitat_nodes <- read_sf("cobio_lcc/vector.shp")
nrow(habitat_nodes)
paisaje <- read_sf("paisaje/paisa.shp")

# Graficamos los archivos
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = habitat_nodes, aes(color = "Parches"), fill = "forestgreen", linewidth = 0.5) +
  scale_color_manual(name = "", values = "black")+
```

```
  theme_minimal() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())

# Indice Integral de Conectividad:
# Ejemplo 1 Solo indice para el paisaje
# LA = NULL
# attribute = NULL
# 10 km mediana dispercion
# area_unit = ha
# distance_threshold = 10000 (10km)
# metric = "IIC"

IIC <- MK_dPCIIC(nodes = habitat_nodes,
                 attribute = NULL,
                 area_unit = "ha",
                 distance = list(type = "centroid"),
                 LA = NULL,
                 onlyoverall = TRUE,
                 metric = "IIC",
                 distance_thresholds = 10000,
                 intern = TRUE) #10 km

# Ejemplo 2 de IIC
# Solo indice para el paisaje
# attribute = NULL
# area_unit = "ha"
# distance = (type = centroid)
# LA = area_paisaje
# onlyoverall = TRUE
# metric = IIC
# distance_threshold = 10000 (10km)

area_paisaje <- st_area(paisaje)
area_paisaje <- unit_convert(area_paisaje, "m2", "ha")

IIC <- MK_dPCIIC(nodes = habitat_nodes,
                 attribute = NULL,
                 area_unit = "ha",
                 distance = list(type = "centroid"),
                 LA = area_paisaje,
                 onlyoverall = TRUE,
                 metric = "IIC",
                 distance_thresholds = 10000,
                 intern = TRUE) #10 km

#>     Index        Value
#> 1  IICnum 2.041990e+11
#> 2 EC(IIC) 4.518839e+05  Estimado de Conectividad
#> 3     IIC 2.677956e-02

# Ejemplo 3 con distancias Euclidianas entre bordes delos nodos
# Solo indice para el paisaje
# metric = "IIC"
# attribute = NULL
# distance_threshold = 10000 (10km)
```

```
# area_unit = "ha"

IIC <- MK_dPCIIC(nodes = habitat_nodes,
                 attribute = NULL,
                 area_unit = "ha",
                 distance = list(type = "edge"),
                 LA = area_paisaje,
                 onlyoverall = TRUE,
                 metric = "IIC",
                 distance_thresholds = 10000,
                 intern = TRUE) #10 km

#>     Index        Value
#> 1  IICnum 5.700691e+11
#> 2 EC(IIC) 7.550292e+05 Estimado de Conectividad
#> 3     IIC 7.476136e-02

# Ejemplo 3a se agrega keep 0.1 en distance
# keep = porcentaje de vertices a retener por parche o nodo sin afectar la forma
# metric = "IIC"

IIC <- MK_dPCIIC(nodes = habitat_nodes,
                 attribute = NULL,
                 area_unit = "ha",
                 distance = list(type = "edge", keep = 0.1),
                 LA = area_paisaje,
                 onlyoverall = TRUE,
                 metric = "IIC",
                 distance_thresholds = 10000,
                 intern = TRUE) #10 km

# Ejmplo 4 con distancias Euclidianas en fracciones
# metric = "IIC"
# attribute = NULL
# distance_threshold = 10000 (10km)
# area_unit = "ha"
# Estimar el IIC a nivel paisaje y nivel de parches o nodos

IIC <- MK_dPCIIC(nodes = habitat_nodes,
                 attribute = NULL,
                 area_unit = "ha",
                 distance = list(type = "edge", keep = 0.1),
                 LA = area_paisaje,
                 onlyoverall = FALSE,
                 metric = "IIC",
                 distance_thresholds = 10000,
                 intern = TRUE,
                 # write = "path/filename" sin extension,
                 ) #10 km
IIC

# Graficamos el shape file usando los dados dIIC

library(classInt)
library(dplyr)
```

```
# Calcular los intervalos de Jenks para strength
# style = "jenks" o "quantile"
# dIIC_q Importancia de cada uno de los fragmentos para la conectividad (rango)

breaks <- classInt::classIntervals(IIC$dIIC, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
IIC <- IIC %>%
  mutate(dIIC_q = cut(dIIC,
                      breaks = breaks$brks,
                      include.lowest = TRUE,
                      dig.lab = 5))


# Graficar en usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = IIC, aes(fill = dIIC_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dIIC (jenks)") +
  theme_minimal() +
  labs(
    title = "dIIC",
    fill = "dIIC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# dIICIntra
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(IIC$dIICintra, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
IIC <- IIC %>%
  mutate(dIICintra_q = cut(dIICintra,
                      breaks = breaks$brks,
                      include.lowest = TRUE,
                      dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = IIC, aes(fill = dIICintra_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dIICIntra (jenks)") +
  theme_minimal() +
  labs(
    title = "dIICIntra",
    fill = "dIICIntra"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```
# dIICflux
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(IIC$dIICflux, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
IIC <- IIC %>%
  mutate(dIICflux_q = cut(dIICflux,
                          breaks = breaks$brks,
                          include.lowest = TRUE,
                          dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = IIC, aes(fill = dIICflux_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dIICIntra (jenks)") +
  theme_minimal() +
  labs(
    title = "dIICflux",
    fill = "dIICIflux"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# dIICconector
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(IIC$dIICconnector, n = 8, style = "jenks")

# Crear una nueva variable categórica con los intervalos
IIC <- IIC %>%
  mutate(dIICconnector_q = cut(dIICconnector,
                          breaks = breaks$brks,
                          include.lowest = TRUE,
                          dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = IIC, aes(fill = dIICconnector_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dIICConnector (jenks)") +
  theme_minimal() +
  labs(
    title = "dIICconnector",
    fill = "dIICconnector"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```
# Generamos un achivo SIG
st_write(IIC,"IIC/ICC.shp", driver = "ESRI Shapefile")

# ------------------------------------------------------------------
# Ejemplo 5 con distancias Euclidianas: fracciones y overall
# metric = "IIC"
# attribute = "FALSE"
# distance_thresholds = 10000 (10km)
# unit_area = "ha"

area_paisaje <- st_area(paisaje)
area_paisaje <- unit_convert(area_paisaje, "m2", "ha")

IIC2 <- MK_dPCIIC(nodes = habitat_nodes,
                  attribute = NULL,
                  area_unit = "ha",
                  distance = list(type = "edge", keep = 0.1),
                  LA = area_paisaje,
                  overall = TRUE,
                  onlyoverall = FALSE,
                  metric = "IIC",
                  distance_thresholds = 10000,
                  intern = TRUE) #10 km
IIC2 # Indice a nivel paisaje y a nivel de parche

# ------------------------------------------------------------------
# Ejemplo con fragmentacion

frag = MK_Fragmentation(nodes = habitat_nodes)
frag2 = frag$`Patch statistics shapefile`

IIC3 <- MK_dPCIIC(nodes = frag2,
                   attribute = "CA", # Columna CA (Core Area) del archivo frag2
                   area_unit = "ha",
                   distance = list(type = "edge", keep = 0.1),
                   LA = area_paisaje,
                   overall = TRUE,
                   onlyoverall = FALSE,
                   metric = "IIC",
                   distance_thresholds = 10000,
                   intern = TRUE,
                   write = "plots/IIC_atr/IIC_atr") #10 km

names(IIC3)
list(IIC3)


# Ejemplo 6 varios ubmrales de distancia
# attribute = "FALSE"
# distance_thresholds = 2000, 10000, 50000 (2km, 10km, 50km)
# area_unit = "ha"
# metric = "IIC"

area_paisaje <- st_area(paisaje)
area_paisaje <- unit_convert(area_paisaje, "m2", "ha")
```

```
IIC4 <- MK_dPCIIC(nodes = habitat_nodes,
                  attribute = NULL,
                  area_unit = "ha",
                  distance = list(type = "edge", keep = 0.1),
                  LA = area_paisaje,
                  overall = TRUE,
                  onlyoverall = FALSE,
                  metric = "IIC",
                  distance_thresholds = c(2000, 10000, 50000),
                  intern = TRUE,
                  write = "plots/IIC_6/IIC_6")

list(IIC4)
names(IIC4)

# ------------------------------------------------------------
# Probabilidad de Coenctividad
# metric = "PC"

# Ejemplo 1 Distacia Euclidiana
# metric = "PC"
# attribute = NULL
# distance_threshold = 10000 (10km)
# probability = 0.5
# area_unit = "ha"
# onlyoverall = TRUE
# LA = NULL

habitat_nodes <- read_sf("Data/Parches.shp")
nrow(habitat_nodes)
paisaje <- read_sf("Data/paisaje.shp")

PC <- MK_dPCIIC(nodes = habitat_nodes,
                attribute = NULL,
                area_unit = "ha",
                distance = list(type = "edge", keep = 0.1),
                LA = NULL,
                onlyoverall = TRUE,
                metric = "PC",
                probability = 0.5,
                distance_thresholds = 10000,
                intern = TRUE) #10 km

# Ejemplo 2 Distancia Eculidiana
# attribute = NULL
# area_unit = "ha"
# distance_threshold = 10000
# metric = "PC"
# probability = 0,5
# LA = area_paisaje
# onlyoverall = TRUE

PC <- MK_dPCIIC(nodes = habitat_nodes,
                attribute = NULL,
                area_unit = "ha",
                distance = list(type = "centroid"),
```

```
            LA = area_paisaje,
            onlyoverall = TRUE,
            metric = "PC",
            probability = 0.5,
            distance_thresholds = 10000,
            intern = TRUE) #10 km

# ------------------------------------------------------------------
# Ejemplo 3
# attribute = NULL
# area_unit = "ha"
# distance_threshold = 10000
# metric = "PC"
# probability = 0,5
# LA = area_paisaje
# onlyoverall = FALSE

PC <- MK_dPCIIC(nodes = habitat_nodes,
            attribute = NULL,
            area_unit = "ha",
            distance = list(type = "edge", keep = 0.1),
            LA = area_paisaje,
            onlyoverall = FALSE,
            metric = "PC",
            probability = 0.5,
            distance_thresholds = 10000,
            intern = TRUE) #10 km

# Hacemos graficos con los datos de la función MK_dPCIIC
# metric = "PC"

library(classInt)
library(dplyr)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(PC$dPC, n = 9, style = "jenks")

# dPC
# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_q = cut(dPC,
                breaks = breaks$brks,
                include.lowest = TRUE,
                dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPC (jenks)") +
  theme_minimal() +
  labs(
    title = "dPC",
    fill = "dPC"
  ) +
  theme(
```

```
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
# dPCIntra
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(PC$dPCintra, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPCintra_q = cut(dPCintra,
                          breaks = breaks$brks,
                          include.lowest = TRUE,
                          dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPCintra_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCIntra (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCIntra",
    fill = "dPCIntra"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# ------------------------------------------------------------------
# dPCflux
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(PC$dPCflux, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPCflux_q = cut(dPCflux,
                         breaks = breaks$brks,
                         include.lowest = TRUE,
                         dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPCflux_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCFlux (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCFlux",
    fill = "dPCFlux"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
```

```
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(PC$dPCconnector, n = 9, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPCconnector_q = cut(dPCconnector,
                  breaks = breaks$brks,
                  include.lowest = TRUE,
                  dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPCconnector_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCConnector (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCConnector",
    fill = "dPCConnector"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Generamos un achivo SIG
st_write(PC,"PC/PC.shp", driver = "ESRI Shapefile")

## 5.14 ----------------------------------------------------------------
# Ejmplo
# Distancia euclidiana, fracciones y overall
# attrute = NULL
# area_unit = "ha"

PC <- MK_dPCIIC(nodes = habitat_nodes,
                attribute = NULL,
                area_unit = "ha",
                distance = list(type = "edge", keep = 0.1),
                LA = area_paisaje,
                overall = TRUE,
                onlyoverall = FALSE,
                metric = "PC",
                probability = 0.5,
                distance_thresholds = 10000,
                intern = TRUE) #10 km

# ---------------------------------------------------------------------
# Ejmplo 5
# Distancia Euclidiana, Paralelizar
# El argumento parallel solo se activa si tienes más de 2000 parches.
# Si tienes menos de 4000 parches utiliza parallel_mode = 1

PC <- MK_dPCIIC(nodes = habitat_nodes,
                attribute = NULL,
                area_unit = "ha",
```

```
            distance = list(type = "edge", keep = 0.1),
            LA = area_paisaje,
            onlyoverall = FALSE,
            metric = "PC",
            probability = 0.5,
            distance_thresholds = 10000,
            parallel = 4,
            parallel_mode = 1,
            intern = TRUE) #10 km


set.seed(10)
# De los 31 parches ecológicos seleccionamos 13
parches_restauracion <- sample(1:nrow(habitat_nodes), 13)
parches_restauracion

# Creamos un nuevo campo o columna en parches_restauracion
habitat_nodes$restauracion <- 1

# Se asigna valor de 0 a los parches selecionado a restaurar
# Presumiendo que no existen en el paisaje

habitat_nodes$restauracion[parches_restauracion] <- 0

# Aplicamos la funcion con la opcion "restorarion"

PCrestauracion <- MK_dPCIIC(nodes = habitat_nodes,
                      attribute = NULL,
                      area_unit = "ha",
                      distance = list(type = "edge", keep = 0.1),
                      LA = NULL,
                      restoration = "restauracion",
                      onlyrestor = TRUE,
                      metric = "PC",
                      probability = 0.5,
                      distance_thresholds = 10000,
                      intern = TRUE) #10 km

# Se crea la columna dPCres con la importancia relativa del parche para mejorar
#   la conectividad del paisaje al aparecer cuando es restaurado.
#   Los valores van de -100 a 100, debido a que su contribución puede incluso
#   ser negativa, es decir, disminuye la conectividad global al restaurarlo

ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = PCrestauracion, aes(fill = dPCres), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlBu",
    direction = -1,
    name = "% dPCres"
  ) +
  theme_minimal() +
  labs(
    title = "Restauración",
    fill = "% dPCres"
  ) +
```

```
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Graficamos solo lo parches a reatarurar

PCrestauracion2 <- PCrestauracion[PCrestauracion$restauracion == 0,]

ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = habitat_nodes, aes(color = "Patches"), fill = NA, color = "black") +
  geom_sf(data = PCrestauracion2, aes(fill = dPCres), color = "black", size = 0.1) +
  scale_fill_distiller(
    palette = "RdYlBu",
    direction = -1,
    name = "% dPCres"
  ) +
  theme_minimal() +
  labs(
    title = "Restauración",
    fill = "% dPCres"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

st_write(PCrestauracion2, "Restore/restore.shp", driver = "ESRI Shapefile")
st_write(PCrestauracion,"Restore/restore0.shp", driver = "ESRI Shapefile")

library(ggplot2)
library(sf)
library(terra)
library(raster)
library(Makurhini)
library(RColorBrewer)


habitat_nodes <- read_sf("cobio_lcc/vector.shp")
nrow(habitat_nodes)
paisaje <- read_sf("paisaje/paisa.shp")

ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = habitat_nodes, aes(color = "Parches"), fill = "forestgreen", linewidth = 0.5) +
  scale_color_manual(name = "", values = "black")+
  theme_minimal() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())


test <- MK_Focal_nodes(nodes = habitat_nodes,
                       id = "fid",
                       attribute = NULL,
                       raster_attribute = NULL,
```

```
                fun_attribute = NULL,
                distance = list(type = "edge", keep = 0.1),
                metric = "PC",
                probability = 0.5,
                distance_thresholds = 10000,
                search_buffer = 20000,
                fragmentation = FALSE,
                parallel = 4,
                intern = FALSE)

names(test)

library(classInt)
library(dplyr)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(test$PC, n = 5, style = "jenks")

PC <- test

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_q = cut(PC,
                breaks = breaks$brks,
                include.lowest = TRUE,
                dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPC_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "PC (jenks)") +
  theme_minimal() +
  labs(
    title = "PC en paisajes focales",
    fill = "PC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Indice dPC focal
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(test$dPC, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_fo = cut(dPC,
                breaks = breaks$brks,
                include.lowest = TRUE,
                dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
```

```
  geom_sf(data = PC, aes(fill = dPC_fo), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPC (jenks)") +
  theme_minimal() +
  labs(
    title = "dPC focal",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Fraccion Intra
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(test$dPCintra, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_intra = cut(dPCintra,
                         breaks = breaks$brks,
                         include.lowest = TRUE,
                         dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPC_intra), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCintra (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCintra",
    fill = "dPCintra"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Fraccion FLUX
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(test$dPCflux, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_flux = cut(dPCflux,
                         breaks = breaks$brks,
                         include.lowest = TRUE,
                         dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPC_flux), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCflux (jenks)") +
  theme_minimal() +
  labs(
```

```
    title = "dPCflux",
    fill = "dPCflux"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# dPC Connector
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(test$dPCconnector, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_conn = cut(dPCconnector,
                        breaks = breaks$brks,
                        include.lowest = TRUE,
                        dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = PC, aes(fill = dPC_conn), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "dPCconnector (jenks)") +
  theme_minimal() +
  labs(
    title = "dPCconnector",
    fill = "dPCconnector"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )


# Índice de Conectividad Compuesto (CCIf)
# No olvidemos que es una herramienta para priorizar cada parche focal
# en función de su contribución individual a la conectividad en la red de
# parches f y thp (dPCf) y la conectividad del paisaje de toda la red (PCf).
# En ese sentido, los parches con valores CCI más altos se encuentran en un
# paisaje bien conectado y su contribución a la conectividad se considera
# importante

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(test$IComp, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
PC <- PC %>%
  mutate(dPC_CCif = cut(IComp,
                        breaks = breaks$brks,
                        include.lowest = TRUE,
                        dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
```

```
  geom_sf(data = PC, aes(fill = dPC_CCif), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlGn", direction = 1, name = "IComp (jenks)") +
  theme_minimal() +
  labs(
    title = "Índice de Conectividad Compuesto",
    fill = "IComp"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# Escribimos un ESRI shapefile con los datos procesados por
# Makurhini_Focal_Nodes
write_sf(PC, "fonodes/IICf/IICf.shp", driver = "ESRI Shapefile")

# ----------------------------------------------------
# Prioridades de enlace entre nodos o parches de vegetacion
# Funcion Makurini:
# MK_dPCIIC_links()
# ----------------------------------------------------

# ------------------------------------------------------------------------
# Rutas de Menor Costo
# Usamos 13 parches o nodos del archivo habitat_nodes
#
# Para seleccionar siempre los mismos de forma aleatoria

set.seed(2)
parches_ejemplo <- habitat_nodes[sample(1:nrow(habitat_nodes), 13),]
parches_ejemplo$Id <- 1:13 #Asignamos un nuevo id

# Graficamos
library(terra)
# data("resistance_matrix", package = "Makurhini")
# ----------------------------------------------------------
# Leemos una matriz de resistencia, en este caso impacto antropogénico
#
resistance_matrix <- raster("matres/impacto.tif")

raster_map <- as(resistance_matrix, "SpatialPixelsDataFrame")
raster_map <- as.data.frame(raster_map)

colnames(raster_map) <- c("value", "x", "y")
ggplot() +
  geom_tile(data = raster_map, aes(x = x, y = y, fill = value), alpha = 0.8) +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = parches_ejemplo, aes(color = "Habitat nodes"), fill = "forestgreen", linewidth = 0.5) +
  scale_fill_gradientn(colors = c("#000004FF", "#1B0C42FF", "#4B0C6BFF", "#781C6DFF",
                                  "#A52C60FF", "#CF4446FF", "#ED6925FF", "#FB9A06FF",
                                  "#F7D03CFF", "#FCFFA4FF"))+
  scale_color_manual(name = "", values = "black")+
  theme_minimal() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())
```

```
library(purrr)
library(gdistance)

# A los valores NA les asignamos un alto valor para evitar que pasen por ahí
resistance_matrix[is.na(resistance_matrix)] <- 1000

#Estimamos la matriz de transición
tr <- transition(resistance_matrix, function(x) 1/mean(x), 8)
#Hacemos una corrección para los movimientos en diagonal
tr <- geoCorrection(tr, type = "c")

#Estimamos el centroide de nuestros parches
centroides <- st_centroid(parches_ejemplo, of_largest_polygon = TRUE)
centroides <- st_coordinates(centroides)

#Loop para estimar corredores entre parches
rutas_list <- list()
counter <- 1
for (i in 1:(nrow(centroides) - 1)) {
  #cat(paste0(i, " de ", nrow(centroides), "\r"))
  counter <- 1
  rutas <- map_dfr((i + 1):nrow(centroides), function(j){
    if(counter <= nrow(centroides)){
      ruta <- shortestPath(tr, centroides[i,], centroides[j,], output = "SpatialLines")
      ruta <- st_as_sf(ruta); st_crs(ruta) <- st_crs(habitat_nodes)
      ruta$from <-i ; ruta$to <- j
      return(ruta)
    }
  })
  rutas_list[[i]] <- rutas
}

rutas_mc <- do.call(rbind, rutas_list)

ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = rutas_mc, aes(color = "corredores"), color = "black", linewidth = 0.5) +
  geom_sf(data = parches_ejemplo, aes(color = "Habitat nodes"),
          fill = "forestgreen", color = NA, linewidth = 0.5) +
  theme_minimal() +
  labs(
    title = "Corredores potenciales"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5)
  )


# ------------------------------------------------------------------------------
# Guardamos las rutas obtenidas como un shapefile
write_sf(rutas_mc, "links/rutas/rutas.shp", driver = "ESRI Shapefile")

# ------------------------------------------------------------------------------
# Aplicamos la función MK_dPCIIC_links(), pero antes exploremos una nueva variante
# de estimar el umbral de distancia cuando usamos una resistencia.
# Estimaremos la distancia efectiva promedio: media de la resistencia x dispersión
```

```
# De esta forma obtenemos una distancia costo.

# Distancia efectiva promedio como umbral de distancia
Effec_mean <- mean(resistance_matrix[], na.rm = TRUE) * 10000 # 10km

#Aplicamos la función
delta <- MK_dPCIIC_links(nodes = parches_ejemplo,
                         attribute = NULL,
                         area_unit = "ha",
                         distance = list(type = "least-cost",
                                         resistance = resistance_matrix),
                         removal = TRUE,
                         metric = "PC",
                         probability = 0.5,
                         distance_thresholds = round(Effec_mean),
                         parallel = NULL,
                         parallel_mode = 0,
                         intern = TRUE)

head(delta)

# Unir los valores de interes en los corredores

#Existen otras formas, ejemplo un nuevo ID
delta$ID_nuevo <- paste0(delta$Destination, "_", delta$Source)

#Guardo las rutas en un objeto nuevo para tener de respaldo mi vector original
rutas_mc2 <- rutas_mc
rutas_mc2$ID_nuevo <- paste0(rutas_mc2$from, "_", rutas_mc$to)

#Aplicar merge
rutas_mc2 <- merge(rutas_mc2, delta, by = "ID_nuevo")

rutas_mc2

library(ggplot2)
library(classInt)
library(dplyr)

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(rutas_mc2$dPC_removal, n = 5, style = "quantile")

# Crear una nueva variable categórica con los intervalos
rutas_mc2 <- rutas_mc2 %>%
  mutate(dPC_q = cut(dPC_removal,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar usando ggplot2 y colores de ColorBrewer
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = rutas_mc2, aes(color = dPC_q), size = 0.5, linewidth = 1) +
  scale_color_brewer(palette = "RdYlBu", direction = -1, name = "dPC remove (Q)") +
  geom_sf(data = parches_ejemplo, aes(color = "Habitat nodes"),
          fill = "forestgreen", color = NA, linewidth = 0.5) +
```

```
  theme_minimal() +
  labs(
    title = "Priorización de enlaces (remove)",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
# Guardamos en shapefile
write_sf(rutas_mc2, "Links/link_remov/rutas_remov.shp", driver = "ESRI Shapefile")

# ------------------------------------------------------------------------------
# Cambios de enlace (Link change)

# Estimamos las distancias de menor costo entre los parches
distancias <- distancefile(parches_ejemplo,
                           id = "Id",
                           type = "least-cost",
                           resistance = resistance_matrix,
                           pairwise = FALSE)

# Escenario donde despues de restaurar disminuyo un 40% la resistencia y aumento
# la permiabilidad en 30 enlaces que tomaremos de forma aleatoria

#No de enlaces
n <- 50
# Numero total de elementos
total_elements <- length(distancias)

# seleccion aleatoria
set.seed(4)
rand_idx <- sample(1:total_elements, n)

# obtener posiciones en la matriz de distancias
rand_positions <- arrayInd(rand_idx, .dim = dim(distancias))

# A esos enlaces les reduciremos un 40% de su valor
distancias_restauracion <- distancias
reduccion <- (40*distancias_restauracion[rand_positions])/100
distancias_restauracion[rand_positions] <- distancias_restauracion[rand_positions]- reduccion

distancias[rand_positions]

distancias_restauracion[rand_positions]

# Aplicamos la funcion MK_dPCIIC_limks
#Distancia efectiva promedio como umbral de distancia

Effec_mean <- mean(resistance_matrix[], na.rm = TRUE) * 10000 # 10km
# [1] 3966740

#Aplicamos la función
delta1 <- MK_dPCIIC_links(nodes = parches_ejemplo,
                          attribute = NULL,
                          area_unit = "ha",
```

```
                distance = distancias,
                removal = TRUE,
                change = distancias_restauracion,
                metric = "PC",
                probability = 0.5,
                distance_thresholds = round(Effec_mean),
                parallel = NULL,
                parallel_mode = 0,
                intern = TRUE)
names(delta1)
# "Link_removal_importances_d3966740" "Link_change_importances_d3966740"

head(delta1$Link_change_importances_d3966740)
# Id Source Destination dPC_change
# 1  1      2           1   0.000000
# 2  2      3           1  -1.685423
# 3  3      4           1   0.000000
# 4  4      5           1   0.000000
# 5  5      6           1   0.000000
# 6  6      7           1   0.000000

# Unimos al vector con los corredores
change_corr <- delta1$Link_change_importances_d3966740
change_corr$ID_nuevo <- paste0(change_corr$Destination, "_", change_corr$Source)

#Por precaución guardamos los corredores en otro objeto y generamos el ID
rutas_mc3 <- rutas_mc
rutas_mc3$ID_nuevo <- paste0(rutas_mc3$from, "_", rutas_mc3$to)
#Unimos
rutas_mc3 <- merge(rutas_mc3, change_corr, by = "ID_nuevo")
rutas_mc3

# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(rutas_mc3$dPC_change, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
rutas_mc3 <- rutas_mc3 %>%
  mutate(dPC_q = cut(dPC_change,
                breaks = breaks$brks,
                include.lowest = TRUE,
                dig.lab = 5))

# Graficar usando ggplot2 y colores de ColorBrewer
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = rutas_mc3, aes(color = dPC_q), size = 0.5, linewidth = 1) +
  scale_color_brewer(palette = "RdYlBu", direction = 1, name = "dPC change (jenks)") +
  geom_sf(data = parches_ejemplo, aes(color = "Habitat nodes"),
          fill = "forestgreen", color = NA, linewidth = 0.5) +
  theme_minimal() +
  labs(
    title = "Priorización de enlaces (change)",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
```

```
    plot.title = element_text(hjust = 0.5)
  )
#
# Guardamos en archivo shapfile
#
write_sf(rutas_mc3, "Links/link_dPC/linksdpc.shp", driver = "ESRI Shapefile")

# Visualizar solo los corredores que sufriedon cambio
# después de mejorar los corredores

rutas_mc4 <- rutas_mc3[rutas_mc3$dPC_change != 0, ]
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(rutas_mc4$dPC_change, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
rutas_mc4 <- rutas_mc4 %>%
  mutate(dPC_q = cut(dPC_change,
                     breaks = breaks$brks,
                     include.lowest = TRUE,
                     dig.lab = 5))

# Graficar usando ggplot2 y colores de ColorBrewer
ggplot() +
  geom_sf(data = paisaje, fill = NA, color = "black") +
  geom_sf(data = rutas_mc4, aes(color = dPC_q), size = 0.5, linewidth = 1) +
  scale_color_brewer(palette = "RdYlBu", direction = 1, name = "dPC change (Jenks)") +
  geom_sf(data = parches_ejemplo, aes(color = "Habitat nodes"),
          fill = "forestgreen", color = NA, linewidth = 0.5) +
  theme_minimal() +
  labs(
    title = "Priorización de enlaces (change)",
    fill = "dPC"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

write_sf(rutas_mc4, "Links/link_change/rutas_change.shp", driver = "ESRI Shapefile")

library(ggplot2)
library(sf)
library(terra)
library(raster)
library(Makurhini)
library(RColorBrewer)


habitat_nodes <- read_sf("cobio_lcc/vector.shp")
nrow(habitat_nodes)
paisaje <- read_sf("paisaje/paisa.shp")

# 9.0 Conectividad dinamica
# ECA = Indice de Área Equivalente ----------------------------------
# Es el área de un unico parche conectado (es decir con probabilidad conexion = 1)
# que tendria el mismo valor de conectividad (PC) que el paisaje real.
```

```
ggplot() +
  geom_sf(data = paisaje, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = habitat_nodes, aes(color = "Parches"), fill = "forestgreen", linewidth = 0.5) +
  scale_color_manual(name = "", values = "black")+
  theme_minimal() +
  theme(axis.title.x = element_blank(),
        axis.title.y = element_blank())

# Estimaremos el índice ECA usando el argumento overall o onlyoverall de la
#             función MK_dPCIIC

ECA1 <- MK_dPCIIC(nodes = habitat_nodes,
                  attribute = NULL,
                  area_unit = "ha",
                  distance = list(type = "edge", keep = 0.1),
                  LA = NULL,
                  metric = "PC",
                  probability = 0.5,
                  onlyoverall = TRUE,
                  distance_thresholds = 10000,
                  intern = FALSE) #10 km
ECA1
#    Index       Value
# 1  PCnum 2.701862e+11
# 2 EC(PC) 5.197943e+05

#Valor del ECA
ECA1[2,2]
# [1] 519794.3

# Lo que obtenemos es el índice ECA o EC el cual tiene unidades de área.
# En nuestro ejemplo estamos trabajando con hectareas,
#             es decir: 519,794.3 ha de bosque potencialmente conectado bajo
#             una mediana de dispersión de 10 km

# Podemos expresarlo en porcentage ya que conocemos el área del paisaje
#         y el área de bosque, a esto se le llama ECA relativo o ECA Normalizado

ECA2 <- ECA1[2,2]

paisaje_area <- st_area(paisaje) # st_area() calcula el area
paisaje_area <- unit_convert(paisaje_area, "m2", "ha")

#ECA normalizado respecto al área del paisaje
(ECA2*100)/paisaje_area
# [1] 16.36387

bosque_area <- st_area(habitat_nodes)
bosque_area <- unit_convert(bosque_area,  "m2", "ha")
bosque_area <- sum(bosque_area)

#ECA normalizado respecto al área del bosque
(ECA2*100)/bosque_area
# [1] 69.56315
```

```
# 9.2 Funcion MK_dECA()
# ------------------ ECA se usa mas para cambios e el paisaje a traves del tiempo

#Sustituir ruta de los .shp
T_2005 <- read_sf("suelo2005/suelo2005T.shp")
T_2008 <- read_sf("suelo2008/suelo2008T.shp")
T_2011 <- read_sf("suelo2011/suelo2011T.shp")


#Creamos una lista, puedes cambiar los nombres, etc.
lista_parches <- list("2005" = T_2005,
                      "2008" = T_2008,
                      "2011" = T_2011)

length(lista_parches)
# [1] 4
names(lista_parches)
# [1] "2005" "2008" "2011"

#



library(ggplot2)
library(patchwork)  # permite agrupaciones de imagenes

p1 <- ggplot() +
  geom_sf(data = T_2005, fill = "forestgreen", color = NA) +
  ggtitle("2005") +
  theme_minimal() +
  theme(axis.text.x = element_blank(),  # Ocultar etiquetas del eje X
        axis.ticks.x = element_blank()) # Ocultar marcas del eje X

p2 <- ggplot() +
  geom_sf(data = T_2008, fill = "forestgreen", color = NA) +
  ggtitle("2008") +
  theme_minimal() +
  theme(axis.text.x = element_blank(),  # Ocultar etiquetas del eje X
        axis.ticks.x = element_blank()) # Ocultar marcas del eje X

p3 <- ggplot() +
  geom_sf(data = T_2011, fill = "forestgreen", color = NA) +
  ggtitle("2011") +
  theme_minimal()

p4 <- ggplot() +
  geom_sf(data = T_2011, fill = "forestgreen", color = NA) +
  ggtitle("2011") +
  theme_minimal()

# Combinar los gráficos en una cuadrícula 2x2 usando patchwork
(p1 + p2) / (p3)

area_estudio <- read_sf("paisaje/paisa.shp")
#Atributo máximo
Max_atributo <- as.numeric(st_area(area_estudio)) * 0.0001 # Hectáreas
```

```
Max_atributo
# [1] 3176476
dECA_test <- MK_dECA(nodes= lista_parches,
                     attribute = NULL,
                     area_unit = "ha",
                     distance = list(type= "edge", keep = 0.1),
                     metric = "PC",
                     probability = 0.5,
                     distance_thresholds = 10000,
                     LA = Max_atributo,
                     plot= c("2005", "2008", "2011"),
                     intern = FALSE)#Puedes cambiar a TRUE para ver el avance

class(dECA_test)
names(dECA_test)

dECA_test$dECA_table
write.csv(dECA_test$dECA_table, "plots/decatest.csv")

hexagons_dECA <- MK_dECA_grid(nodes = lista_parches,
                              nodes_names = c("2005", "2008", "2011"),
                              region = area_estudio,
                              area_unit = "ha",
                              metric = "PC",
                              distance_threshold = 10000,
                              probability = 0.5,
                              distance = list(type = "edge", keep = 0.1),
                              grid = list(hexagonal = TRUE,
                                          cellsize = unit_convert(50, "km2", "m2")),
                              parallel = 6,
                              intern = FALSE)#Puedes cambiar a TRUE para ver el avance
names(hexagons_dECA)
#plot(hexagons_dECA$result_2005.2008)

#st_write(hexagons_dECA$result_1993.2003, "plots/hexa/hex_grid.shp", delete_layer = TRUE)
# ----------------------------------------------------------
library(spatstat)
library(sf)

# ----------------------------------------------------------------
# dECA 2005-2008:

library(classInt)
library(dplyr)

# Calcular los intervalos
dECA_plot <- hexagons_dECA$result_2005.2008
breaks <- classInt::classIntervals(dECA_plot$dECA, n = 9, style = "quantile")

# Crear una nueva variable categórica con los intervalos
dECA_plot <- dECA_plot %>%
  mutate(dECA_q = cut(dECA,
                      breaks = breaks$brks,
                      include.lowest = TRUE,
                      dig.lab = 5))
```

```
ggplot() +
  geom_sf(data = dECA_plot, aes(fill = dECA_q), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "RdYlGn",
    direction = 1,
    name = "% dECA"
  ) +
  theme_minimal() +
  labs(
    title = "GRID dECA 2005-2008",
    fill = "% dECA"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
st_write(dECA_plot, "dECA/2005_2008/dECA05_08.shp", delete_layer = TRUE)

# dECA 2008-2011

# Calcular los intervalos
dECA_plot <- hexagons_dECA$result_2008.2011
breaks <- classInt::classIntervals(dECA_plot$dECA, n = 9, style = "quantile")

# Crear una nueva variable categórica con los intervalos
dECA_plot <- dECA_plot %>%
  mutate(dECA_q = cut(dECA,
                      breaks = breaks$brks,
                      include.lowest = TRUE,
                      dig.lab = 5))
ggplot() +
  geom_sf(data = dECA_plot, aes(fill = dECA_q), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "RdYlGn",
    direction = 1,
    name = "% dECA"
  ) +
  theme_minimal() +
  labs(
    title = "GRID dECA 2003-2007",
    fill = "% dECA"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
st_write(dECA_plot, "dECA/2008_2011/dECA08_11.shp", delete_layer = TRUE)

# dECA 2007-2011

# Calcular los intervalos
dECA_plot <- hexagons_dECA$result_2007.2011
breaks <- classInt::classIntervals(dECA_plot$dECA, n = 9, style = "quantile")

# Crear una nueva variable categórica con los intervalos
dECA_plot <- dECA_plot %>%
```

```
  mutate(dECA_q = cut(dECA,
                      breaks = breaks$brks,
                      include.lowest = TRUE,
                      dig.lab = 5))
st_write(dECA_plot, "plots/dECA07_11/dECA07_11.shp", delete_layer = TRUE)

ggplot() +
  geom_sf(data = dECA_plot, aes(fill = dECA_q), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "RdYlGn",
    direction = 1,
    name = "% dECA"
  ) +
  theme_minimal() +
  labs(
    title = "GRID dECA 2007-2011",
    fill = "% dECA"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

# rECA 2008-2011

# Calcular los intervalos de Jenks
dECA_plot <- hexagons_dECA$result_2005.2008
breaks <- classInt::classIntervals(dECA_plot$rECA, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
dECA_plot <- dECA_plot %>%
  mutate(rECA_q = cut(rECA,
                      breaks = breaks$brks,
                      include.lowest = TRUE,
                      dig.lab = 5))
ggplot() +
  geom_sf(data = dECA_plot, aes(fill = rECA_q), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "RdYlGn",
    direction = 1,
    name = "rECA"
  ) +
  theme_minimal() +
  labs(
    title = "GRID rECA 2005-2008",
    fill = "rECA"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
st_write(dECA_plot, "dECA/rECA_0508/rECA05_08.shp", delete_layer = TRUE)


# Tipo de cambio entre 2005-2008
graf = hexagons_dECA$result_2008.2011
```

```
ggplot() +
  geom_sf(data = area_estudio, aes(color = "Study area"), fill = NA, color = "black") +
  geom_sf(data = hexagons_dECA$result_2008.2011, aes(fill = Type.Change), color = "black", size = 0.1) +
  scale_fill_brewer(
    palette = "Set1",
    name = "Tipo de cambio"
  ) +
  theme_minimal() +
  labs(
    title = "GRID dECA, Tipo de cambio 2005-2008"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
st_write(graf, "dECA/dECA08_11/dECA0811.shp", delete_layer = TRUE)

library(ggplot2)
library(sf)
library(terra)
library(raster)
library(Makurhini)
library(RColorBrewer)


AP <- read_sf("anp_2020/anp_2020.shp")
nrow(AP)
APs <- AP[, c("NOMBRE", "TIPO", "ENTIDAD")]
Ecorreg <- read_sf("cobio_lcc/vector.shp")
Ecorreg_1 <- Ecorreg[19,] #Selecciono la primera fila o primera ecorregion

test <- MK_ProtConn(nodes = APs,
                    region = Ecorreg_1,
                    area_unit = "ha",
                    distance = list(type= "edge", keep = 0.1),
                    distance_thresholds = 10000,
                    probability = 0.5,
                    transboundary = 50000,
                    plot = TRUE,
                    parallel = NULL,
                    protconn_bound = FALSE,
                    delta = FALSE,
                    write = NULL,
                    intern = TRUE)
names(test)
test$`ProtConn Plot`
test$`Protected Connected (Viewer Panel`
write.csv(test$`Protected Connected (Viewer Panel`, "plots/eco19.csv")

# ------------------------------------------------------------------

test2 <- MK_ProtConn(nodes = APs,
                     region = Ecorreg_1,
                     area_unit = "ha",
                     distance = list(type= "edge", keep = 0.1),
```

```
                distance_thresholds = 10000,
                probability = 0.5,
                transboundary = 50000,
                plot = TRUE,
                parallel = NULL,
                protconn_bound = TRUE,
                delta = FALSE,
                write = NULL,
                intern = TRUE)

names(test2)
test2$`ProtConn Plot`
test2$`Protected Connected (Viewer Panel`
write.csv(test2$`Protected Connected (Viewer Panel`, "plots/eco19b.csv")

# ------------------------------------------------------------
# Contribucion de cada AP

test3 <- MK_ProtConn(nodes = APs,
                region = Ecorreg_1,
                area_unit = "ha",
                distance = list(type= "edge", keep = 0.1),
                distance_thresholds = 10000,
                probability = 0.5,
                transboundary = 50000,
                plot = TRUE,
                parallel = NULL,
                protconn_bound = FALSE,
                delta = TRUE,
                write = NULL,
                intern = TRUE)

names(test3)
test3$`ProtConn Plot`
test3$`Protected Connected (Viewer Panel`
write.csv(test3$`Protected Connected (Viewer Panel`, "plots/eco19c.csv")

test3$ProtConn_Delta

# ------------------------------------------------------------------------
# delta Protegido
# ------------------------------------------------------------------------

library(classInt)
library(dplyr)


dProtConn <- test3$ProtConn_Delta
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(dProtConn$dProt, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
dProtConn2 <- dProtConn %>%
  mutate(dProt_q = cut(dProt,
                  breaks = breaks$brks,
                  include.lowest = TRUE,
```

```
                    dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = Ecorreg_1, fill = NA, color = "black") +
  geom_sf(data = dProtConn2, aes(fill = dProt_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlBu", direction = -1, name = "Delta Protegido(jenks)") +
  theme_minimal() +
  labs(
    title = "Delta Prot",
    fill = "Delta Prot"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
write_sf(dProtConn2, "dProt/delta_prot.shp", driver = "ESRI Shapefile")
write_sf(Ecorreg_1, "Ecorreg_19/ecorreg19.shp", driver = "ESRI Shapefile")

# ------------------------------------------------------------------------
# delta ProtConn
# ------------------------------------------------------------------------

dProtConn <- test3$ProtConn_Delta
# Calcular los intervalos de Jenks para strength
breaks <- classInt::classIntervals(dProtConn$dProtConn, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
dProtConn2 <- dProtConn %>%
  mutate(dProtConn_q = cut(dProtConn,
                           breaks = breaks$brks,
                           include.lowest = TRUE,
                           dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = Ecorreg_1, fill = NA, color = "black") +
  geom_sf(data = dProtConn2, aes(fill = dProtConn_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "RdYlBu", direction = 1, name = "Delta ProtConn (jenks)") +
  theme_minimal() +
  labs(
    title = "Delta ProtConn",
    fill = "Delta ProtConn"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )
write_sf(dProtConn2, "dprotconn/dprotconn.shp", driver = "ESRI Shapefile")

# ------------------------------------------------------------------------
# Variacion absoluta de ProtConn
# ------------------------------------------------------------------------

dProtConn <- test3$ProtConn_Delta
# Calcular los intervalos de Jenks para strength
```

```
breaks <- classInt::classIntervals(dProtConn$varProtConn, n = 5, style = "jenks")

# Crear una nueva variable categórica con los intervalos
dProtConn2 <- dProtConn %>%
  mutate(varProtConn_q = cut(varProtConn,
                             breaks = breaks$brks,
                             include.lowest = TRUE,
                             dig.lab = 5))

# Graficar en ggplot2 usando las clases Jenks
ggplot() +
  geom_sf(data = Ecorreg_1, fill = NA, color = "black") +
  geom_sf(data = dProtConn2, aes(fill = varProtConn_q), color = "black", size = 0.1) +
  scale_fill_brewer(palette = "PuOr", direction = -1, name = "varProtConn (jenks)") +
  theme_minimal() +
  labs(
    title = "varProtConn",
    fill = "varProtConn"
  ) +
  theme(
    legend.position = "right",
    plot.title = element_text(hjust = 0.5)
  )

write_sf(dProtConn2, "varprtconn/varptconn.shp", driver = "ESRI Shapefile")
```

```
# ------------------------------------------------------------------------
# ProtConn Multi
# ------------------------------------------------------------------------

test4 <- MK_ProtConnMult(nodes = APs,
                         regions = Ecorreg,
                         area_unit = "ha",
                         distance = list(type= "edge", keep = 0.1),
                         distance_thresholds = 10000,
                         probability = 0.5,
                         transboundary = 50000,
                         plot = TRUE,
                         parallel = 4,
                         protconn_bound = FALSE,
                         delta = FALSE,
                         CI = "all",
                         write = NULL,
                         intern = FALSE)
```

En esta última función se detiene el análisis dado que al correr el algoritmo da la siguiente respuesta: