

# Conservacion Oso Andino

Edwin Uribe

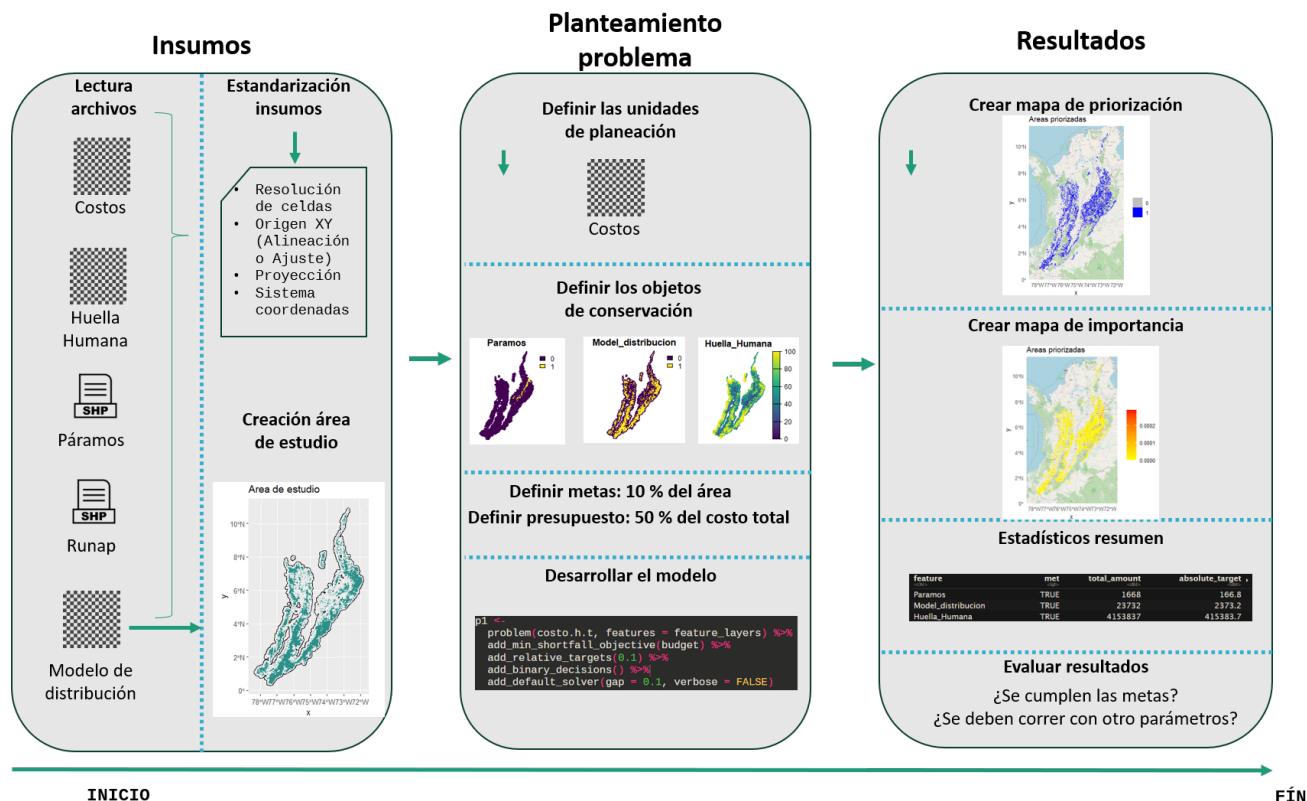
2024-09-19

## Objetivo

Elaborar una rutina documentada en R para identificar áreas prioritarias de conservación del oso andino *Tremarctos ornatus*.

Objeto de conservación: el oso andino y sus áreas de distribución, que incluyen ecosistemas de páramo y zonas con baja intervención humana.

## Diagrama de flujo



## Primeros pasos:

### 1. Configuración de entorno de trabajo

```
# Al ejecutar por primera se debe activar la instalacion de paquetes
#packages <- c("prioritizr", "prioritizrdata", "terra", "sf", "raster", ##"fasterize", "ggplot2", "viridis")

# Instalar los paquetes que no estan ya instalados
#install.packages(setdiff(packages, rownames(installed.packages())))

# install.packages("highs", repos = "https://cran.rstudio.com/")

# Cargar librerias necesarias
library(prioritizr)
library(prioritizrdata)
library(terra)
library(sf)
library(raster)
library(fasterize)
library(ggplot2)
library(viridis)
library(ggspatial)

# Opcionales
# Quitar notacion cientifica
options(scipen=999)
```

### 2. Lectura de insumos

Se utilizaron un total de 5 insumos, los cuales se agruparon en un directorio de trabajo específico.

```
# Establecer directorio de trabajo
setwd('D:/Convocatorias/Humboldt/Prueba_Tecnica_Asistente1')
# Insumos
costo <- raster('Capa_costos/RASTER/Beneficio_Neto_Total.tif')
model.distribucion = raster('Tremarctos ornatus/Tremarctos ornatus.tif')
hh <- raster('Huella Humana/IHEH_2018.tif')
paramos <- st_read('Paramos/Complejos de Paramos_Escala100k.shp')
runap <- st_read("D:/TNC/Biofisica_INPUTS/RUNAP.shp")
```

### 3. Configuración área de estudio

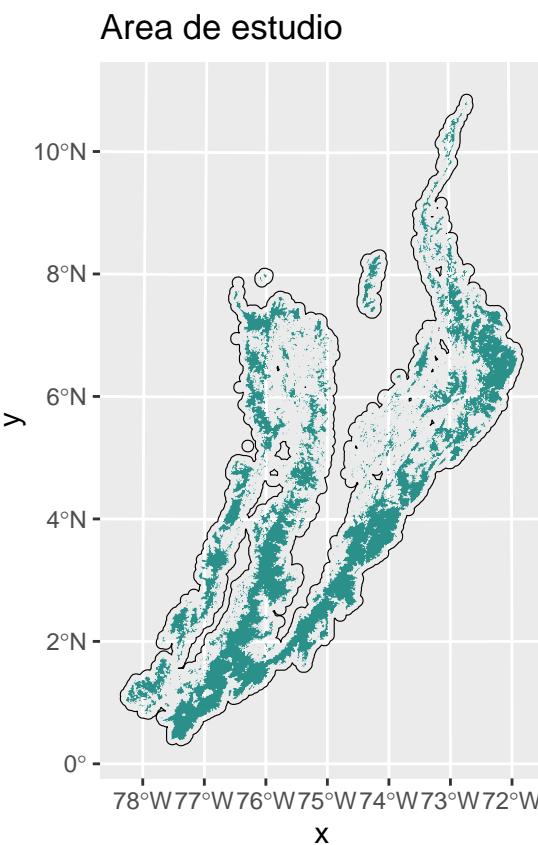
Es importante establecer la extensión y los límites del área de estudio para configurar el número exacto de píxeles en los que se ejecutará el algoritmo. Todo lo que esté clasificado fuera de esta máscara se marcará como NA. El área de estudio se definió a partir de un buffer de 10 km aplicado al modelo de distribución de la especie.

El tamaño del buffer es arbitrario, pero tiene como objetivo considerar las zonas aledañas al modelo de distribución, que también son importantes para la conservación de la especie.

```

# Establecer area de estudio (ae) ----
ae <- model.distribucion
ae <- projectRaster(ae, crs = 32618, method = 'ngb')
ae.poly <- as.polygons(rast(ae), dissolve=TRUE)
ae.poly <- buffer(ae.poly, width=10000)
ae <- crop(ae, extent(st_as_sf(ae.poly)))
# Crear plantilla
template <- raster(resolution = 4000, crs= crs(ae),ext = extent(ae))

```



#### 4. Homogenizar insumos

El paquete *prioritizr* requiere que todos los insumos tengan la misma resolución, extensión, origen y sistema de coordenadas. En esta sección del código, la función `resample()` nos permite alinear las capas para que sean procesados correctamente.

```

# Homogenizar insumos ----

# Rasterizar insumos SHP
paramos <- fasterize(paramos, template)
runap <- st_crop(st_transform(runap, crs = st_crs(ae)), ae)
runap <- fasterize(runap, template)

# Iteraracion sobre insumos
#Lista

```

```

tiffs = list(costo = costo, model.distribucion = model.distribucion, hh = hh, paramos = paramos, runap = runap)
# Loop
for (i in 1:length(tiffs)){
  r <- tiffs[[i]]
  r1 <- projectRaster(r, crs = crs(ae), method = "ngb")
  r1 <- crop(r1, extent(ae))
  r1 <- resample(rast(r1), rast(template), method = "near")
  # Las capas resultantes se guardan con el sufijo ".h"
  assign(paste0(names(tiffs)[i], '.h'), r1)
  message(paste('Insumo preparado:', names(tiffs)[i]))
}

## Insumo preparado: costo

## Insumo preparado: model.distribucion

## Insumo preparado: hh

## Insumo preparado: paramos

## Insumo preparado: runap

```

## Carpinteria específica por capas

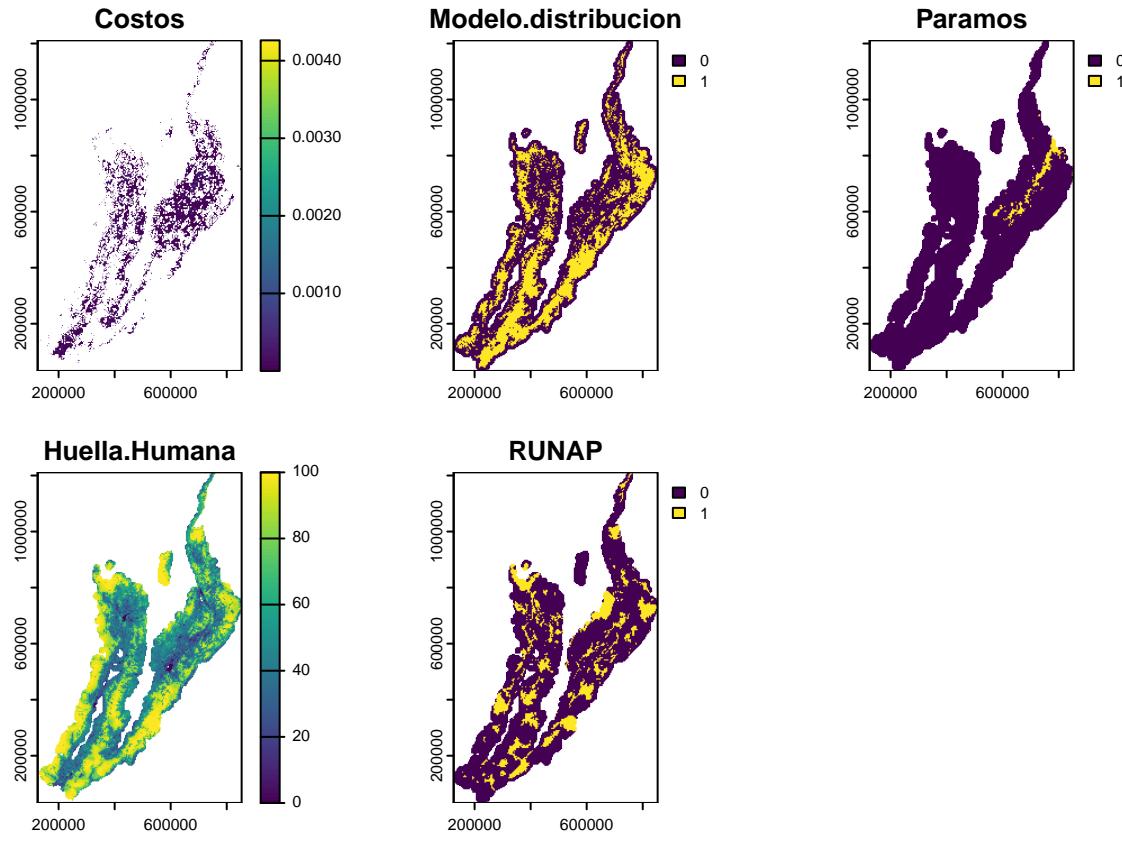
En algunos casos, es necesario realizar procesos específicos que modifiquen las capas para que se ingresen correctamente en el modelo de priorización.

Es importante destacar que la capa de costos se dividió por 1 billon para reducir el número de dígitos en los píxeles, permitiendo que el algoritmo procese los valores más rápidamente. Similarmente, la capa de huella humana se invirtió para que se seleccionaran las zonas de menor intervención.

```

# Carpinteria específica por capas
# Costos
costo.h.t = costo.h/10000000000000
costo.h.t = mask(costo.h.t, ae.poly)
# Modelo de distribucion
# remplazar con ceros donde la especie no este presente, dejar NA todo lo que esta por fuera del area de
model.distribucion.h[is.na(model.distribucion.h)] <- 0
model.distribucion.h = mask(model.distribucion.h, ae.poly)
# Paramos
paramos.h[is.na(paramos.h)] <- 0
paramos.h= mask(paramos.h, ae.poly)
# Huella Humana
# Invertir datos para priorizar los que tengan menor huella humana
hh.h.i <- 100 - hh.h
hh.h.i= mask(hh.h.i, ae.poly)
# runap
runap.h[is.na(runap.h)] <- 0
runap.h= mask(runap.h, ae.poly)

```



## Desarrollo del modelo

### 1. Calcular el presupuesto

Dado que la capa de costos se transformó, el resultado del presupuesto está en Billones de pesos. Este presupuesto indica el dinero que se destinará al proyecto de conservación. En este caso, se definió arbitrariamente un presupuesto del 20% del total de los costos de oportunidad del área de estudio

```
budget <- terra:::global(costo.h.t, "sum", na.rm = TRUE)[[1]] * 0.2
print(paste(budget, 'Billones de pesos'))
```

## [1] "0.122491767802882 Billones de pesos"

### 2. Plantear el problema

El paquete *prioritizr* funciona a partir de la ejecución de un problema por medio de la función `problem()`. En esta función hay una capa de unidades de planeación que está representada por los costos de oportunidad (`costo.h.t`). Adicionalmente, los *features* representan los objetos de conservación a tener en cuenta en el problema. Finalmente, también se define una meta de conservación (*target*) del 10 %, que significa el porcentaje del área de estudio que se busca conservar.

```

#Reunir objetos de conservacion
feature_layers = c(paramos.h, model.distribucion.h, hh.h.i)
# Cambiar nombre de capas
names(feature_layers) <- c("Paramos", "Model_distribucion", "Huella_Humana")
# plot(feature_layers,col = viridis(100), axes = FALSE)

#Crear problema
p1 <-
  problem(costo.h.t, features = feature_layers) %>%
  add_min_shortfall_objective(budget) %>%
  add_relative_targets(0.10) %>%
  add_binary_decisions() %>%
  add_default_solver(gap = 0.1, verbose = FALSE)

# Calcular el número de unidades de planeacion
number_of_planning_units(p1)

```

## [1] 6197

### 3. Solucionar primera aproximación

Se ejecuta la función `solve()` para realizar la priorización.

```

s1 <- solve(p1)
# Visualizar resultados
s_transformed <- projectRaster(raster(s1), crs = 3857, method = 'ngb')
raster_df <- as.data.frame(s_transformed, xy = TRUE)

ggplot() + annotation_map_tile(zoom = 8, type = "osm") +
  geom_raster(data = na.omit(raster_df), aes(x = x, y = y,
                                              fill = factor(Beneficio_Neto_Total))) +
  scale_fill_manual(values = c("grey", 'blue'), na.value = NA) +
  theme_minimal() +
  labs(title = "Areas priorizadas S1", fill = "")

```

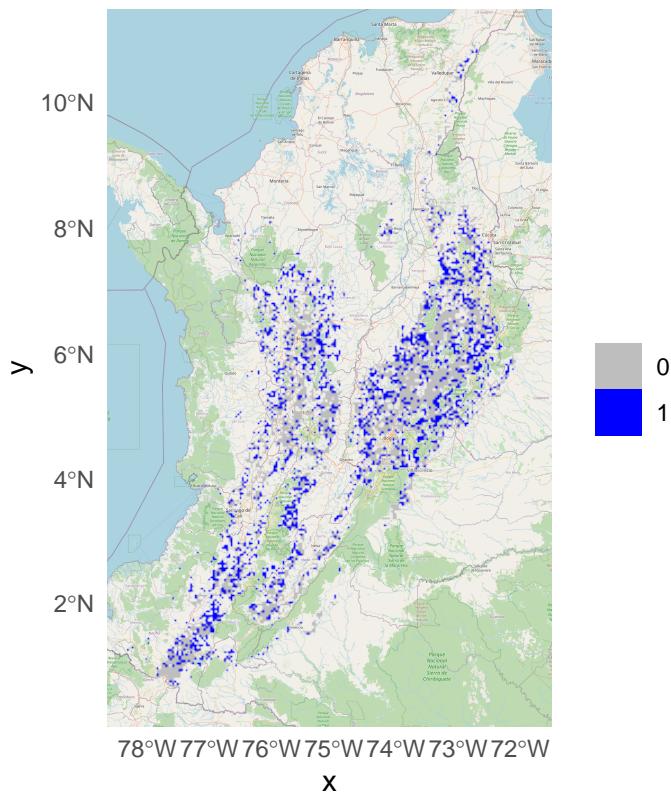
```

## Warning: Raster pixels are placed at uneven horizontal intervals and will be shifted
## i Consider using 'geom_tile()' instead.
## Raster pixels are placed at uneven horizontal intervals and will be shifted
## i Consider using 'geom_tile()' instead.

## Zoom: 8

```

## Areas priorizadas S1

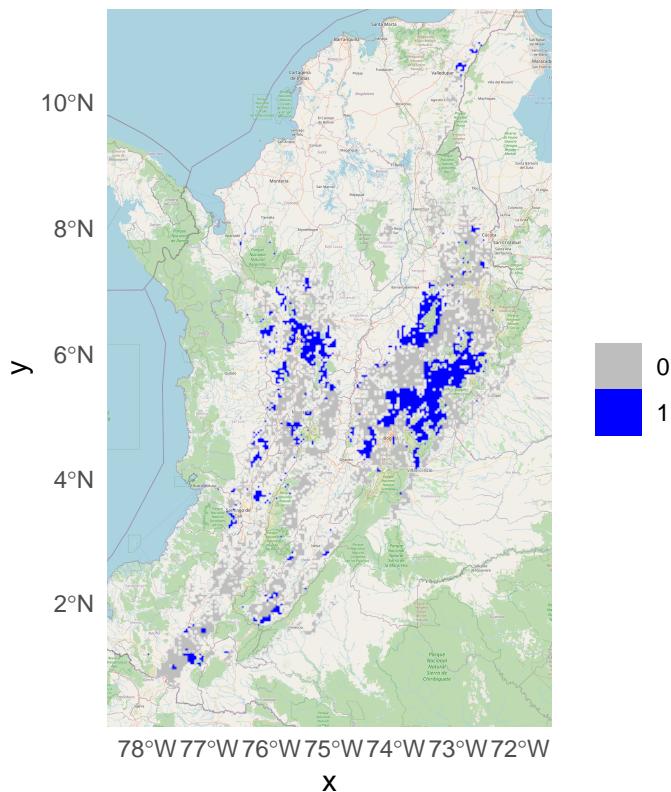


## 4. Possible optimización

No obstante, los resultados se encuentran altamente fragmentados por lo que se puede optimizar la solución. En este caso se agregó penalidades de borde `add_boundary_penalties()` para que agrupara más los resultados. Similarmente, se agregó el mapa de área protegidas para que funcionaran como restricciones `add_locked_in_constraints()` que permitieran atraer y optimizar las áreas prioritarias a los núcleos de las áreas protegidas.

```
p2 <-  
  problem(costo.h.t, features = feature_layers) %>%  
  add_min_shortfall_objective(budget) %>%  
  add_relative_targets(0.1) %>%  
  add_binary_decisions() %>%  
  add_default_solver(gap = 0.1, verbose = FALSE) %>%  
  add_locked_in_constraints(runap.h) %>%  
  add_boundary_penalties(penalty = 0.001, edge_factor = 0.05)  
s2 <- solve(p2)  
  
## Warning: Raster pixels are placed at uneven horizontal intervals and will be shifted  
## i Consider using 'geom_tile()' instead.  
## Raster pixels are placed at uneven horizontal intervals and will be shifted  
## i Consider using 'geom_tile()' instead.  
  
## Zoom: 8
```

## Areas priorizadas S2



## 5. Extracción de resultados

Se pueden extraer la importancia de las áreas priorizadas, para ver que pixeles o unidades de planeación tienen los puntajes más altos con la función eval\_ferrier\_importance(s2)

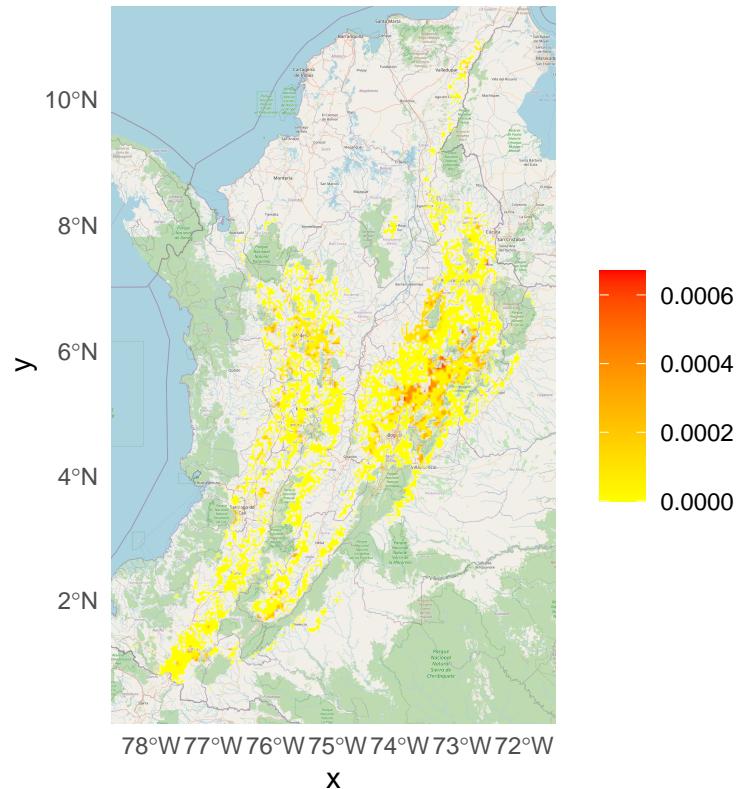
```
# Calcular puntajes de importancia
rc <- p2 %>% eval_ferrier_importance(s2)

# Preparar para plot
s_transformed <- projectRaster(raster(rc[["total"]]), crs = 3857, method = 'ngb')
raster_df <- as.data.frame(s_transformed, xy = TRUE)

# Graficar los resultados
ggplot() + annotation_map_tile(zoom = 8, type = "osm") +
  geom_raster(data = na.omit(raster_df), aes(x = x, y = y,
                                              fill = total)) +
  scale_fill_gradient(low = "yellow", high = "red", na.value = NA) +
  theme_minimal() +
  labs(title = "Areas priorizadas", fill = "")

## Zoom: 8
```

## Areas priorizadas



Tambien se puede extraer el costo total de la solución, y el porcentaje del meta de conservación que se puede alcanzar a partir de la solución establecida. De esta forma se puede evaluar el desempeño de las dos soluciones.

### Desempeño solución 2

```
# costo de la solucion 2
costo.total = eval_cost_summary(p2, s2)
# Resultado en Billones de pesos
print(costo.total$cost)
```

```
## [1] 0.1224917
```

```
# Resumen del cumplimiento de las metas por cada objeto de conservacion
p2_target_coverage <- eval_target_coverage_summary(p2, s2)
print(p2_target_coverage)
```

```
## # A tibble: 3 x 9
##   feature    met  total_amount absolute_target absolute_held absolute_shortfall
##   <chr>     <lgl>      <dbl>            <dbl>        <dbl>            <dbl>
## 1 Paramos    TRUE       419             41.9         77              0
## 2 Model_dis~ FALSE      5984            598.          396            202.
## 3 Huella_Hu~ FALSE     1064409        106441.       71152          35289.
```

```

## # i 3 more variables: relative_target <dbl>, relative_held <dbl>,
## #   relative_shortfall <dbl>

# Extraer el porcentaje promedio del cumplimiento de las metas de conservacion
print(mean(p2_target_coverage$met) * 100)

```

```
## [1] 33.33333
```

## Desempeño solución 1

```

# costo de la solucion 1
costo.total = eval_cost_summary(p1, s1)
# Resultado en Billones de pesos
print(costo.total$cost)

```

```
## [1] 0.03506292
```

```

# Resumen del cumplimiento de las metas por cada objeto de conservacion
p1_target_coverage <- eval_target_coverage_summary(p1, s1)
print(p1_target_coverage)

```

```

## # A tibble: 3 x 9
##   feature    met  total_amount absolute_target absolute_held absolute_shortfall
##   <chr>     <lgl>      <dbl>        <dbl>       <dbl>            <dbl>
## 1 Paramos    TRUE       419         41.9        56             0
## 2 Model_dis~ TRUE      5984        598.        599            0
## 3 Huella_Hu~ TRUE     1064409     106441.     106467          0
## # i 3 more variables: relative_target <dbl>, relative_held <dbl>,
## #   relative_shortfall <dbl>

```

```

# Extraer el porcentaje promedio del cumplimiento de las metas de conservacion
print(mean(p1_target_coverage$met) * 100)

```

```
## [1] 100
```

## Conclusiones

Este resultado indica que se necesita más presupuesto en la solución 2 para cumplir la meta del 10 %,aúnque esta solución nos presenta una configuración menos fragmentada de los objetos de conservación, se necesita de un presupuesto mayor para poder cumplir las metas de conservación. En este sentido, se opta por la solución 1.

Este tipo de decisiones dependen de los tomadores de decisiones y de todo el soporte teórico que respalde al proyecto de conservación.