Supplementary Material 3 – Scripts for the three automated biodiversity offsetting scenarios

We used R version 3.6.1

A – Script for the scenarios *Old way (S) and Local river basin plan (SF)*

```
rm(list=ls())

#Choosing the work directory
wd<-"/Simulations/"
setwd(wd)

# #Importing the packages
# library(RPostgreSQL)
# library(gdata)
#
# drv <- dbDriver("PostgreSQL")
# con <- dbConnect(drv, dbname = "acv",
#                host = "localhost", port = 5432,
#                user = "postgres", password = "lerat")
#
# #Importing the offsetting need synthesis (map 1)
# query<-"select * from offsetting_need"
# offsetting_need<-dbGetQuery(con,query)
# str(offsetting_need)
#
# #Importing the response capacity table (map 2)
# query<-"select
unique_id,sws_name,surf_ha,ecorestopot,ecorestopot_polygon,in_eco_corridor from
response_capacity_shp"
# response_capacity_shp<-dbGetQuery(con,query)
# str(response_capacity_shp)
#
# #Importing the response capacity synthesis (map 2)
# query<-"select * from response_capacity"
# response_capacity<-dbGetQuery(con,query)
# str(response_capacity)
#
# #cleaning
# rm("con","drv","query","wd")
# save.image("E:/Analysis/ACV_esenet/REV_2020/Simulations/Data/Data needed to
simulate.RData")

load(paste(wd,"Data/","Data needed to simulate.RData",sep=""))

#Choosing the settings for the basic scenarios of the paper or the supplementary analysis
ss_scenario<-"S" # S (Old way), SF (Local river basin plan), please choose
```

```
prio_eco_corridor<-"no_prio_eco_corridor" # prio_eco_corridor, no_prio_eco_corridor,
please choose
ecorestopot6<-"with6" # without6 (excluding polygons of highest ecological restoration
potential per ha), with6 (including polygons of highest ecological restoration potential per
ha), please choose
ecorestopot0<-"with0" # without0 (excluding polygons of lowest ecological restoration
potential per ha), with0 (including polygons of lowest ecological restoration potential per
ha), please choose

#Setting a output folder for this scenario
dir.create(paste(wd,"Outputs/",sep=""))
setwd(paste(wd,"/Outputs/",sep=""))
scen<-paste(ss_scenario,prio_eco_corridor,ecorestopot6,ecorestopot0,sep="_") #Creating
the name of folder variable (scenario)
dir.create(scen) #Creating the folder
setwd(paste(getwd(),scen,sep="/")) #Locating in the created folder

#Applying the settings regarding the polygons of highest and lowest ecological restoration
potential per ha
if(ecorestopot6=="without6") # Polygons of highest ecological restoration potential per ha
  {response_capacity_shp<-
subset(response_capacity_shp,response_capacity_shp$ecorestopot!=6)}
if(ecorestopot0=="without0") # Polygons of highest ecological restoration potential per ha
  {response_capacity_shp<-
subset(response_capacity_shp,response_capacity_shp$ecorestopot!=0)}

#Indicating that sws_name and id are factors
response_capacity_shp$sws_name<-as.factor(response_capacity_shp$sws_name)
response_capacity_shp$unique_id<-as.factor(response_capacity_shp$unique_id)

#Ignoring the sub-watershed (sws) where the offsetting need is below 0.5ha
subset(offsetting_need,(surf_ha*2)<0.5)
offsetting_need<-subset(offsetting_need,(surf_ha*2)>=0.5)

#Verifying that the sub-watersheds are independent (for each sub-watershed, the objective
area (twice the offsetting need in area) must fit into the response capacity in area)
(offsetting_need$surf_ha*2)<response_capacity[response_capacity$sws_name%in%offsetti
ng_need$sws_name,]$surf_ha

#Calculating the number of polygons intersecting ecological corridors per sub-watershed
table(response_capacity_shp$sws_name,response_capacity_shp$in_eco_corridor)



#Setting the number of simulation required
nsimus<-5000
```

```
#Creating a table of the synthethic results
res<-NULL


#Launching loops for each sub-watershed, whatever the order because the sub-watersheds
are independent
for(i in 1:nsimus)
{

for (SWS in unique(offsetting_need$sws_name)) #loop on sub-watershed
{

#Setting the objectives and indicators to follow
obj_surf<-offsetting_need[which(offsetting_need$sws_name==SWS),]$surf_ha*2 # Area of
the objective need for the sub-watershed (twice the impacted area)
response_capacity_SWS<-subset(response_capacity_shp,sws_name==SWS)  # Creating a
response capacity pool for selecting the offsetting measures in the sub-watershed
cur_surf<-0 #Creating the object of the sum of the area of selected wetlands
delta_surf<-0 #Creating the object of the difference between the area of the objective need
and the current area of selected wetlands (during the simulations)
selection_WET<-NULL #Creating the object to fill with the selected wetlands (WET)

#Selecting the wetlands (offsetting measures)
while (cur_surf<obj_surf) #Selecting wetlands as long as we are below the area of the
objective need
{

if(prio_eco_corridor=="no_prio_eco_corridor") #If there is no priority for selecting wetlands
in ecological corridors
{
   if(cur_surf<=(obj_surf/2)&ss_scenario=="SF") #And if we are below 100% of the impacted
area and in the SF scenario
   {
   chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$ecorestopot==max(respons
e_capacity_SWS$ecorestopot)),]$unique_id,1) #Selecting the wetlands with the highest
ecological restoration potential per ha in decreasing order
   } else
   {
   chosen_id<-sample(response_capacity_SWS$unique_id,1) #Otherwise (beyond 100% of
the impacted area in the SF scenario and in the S scenario in general), randomly selecting
wetlands
   }
}
```

```
if(prio_eco_corridor=="prio_eco_corridor") #If there is a priority for selecting wetlands in
ecological corridors
{

if(TRUE%in%response_capacity_SWS$in_eco_corridor&max(response_capacity_SWS$ecores
topot)%in%response_capacity_SWS$ecorestopot[which(response_capacity_SWS$in_eco_co
rridor==T)]) #As long as there are wetlands intersecting ecological corridors and being of the
highest ecological restoration potential per ha
 {
  if(cur_surf<=(obj_surf/2)&ss_scenario=="SF") #And if we are below 100% of the impacted
area and in the SF scenario
  {
  chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$ecorestopot==max(respons
e_capacity_SWS$ecorestopot)&response_capacity_SWS$in_eco_corridor==T),]$unique_id,1
) #Selecting the wetlands intersecting ecological corridors and with the highest ecological
restoration potential per ha (in decreasing order but note when the last wetland of highest
potential has been selected, if the new wetland of highest potential is out of ecological
corridors, the loop jumps into the next if, but should come back here if, again, the new
wetland of highest potential will be intersecting ecological corridors)
  } else
  {
  chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$in_eco_corridor==T),]$uniq
ue_id,1)  #Otherwise (below 100% of the impacted area in the SF scenario and in the S
scenario in general), randomly selecting wetlands intersecting ecological corridors
  }
 }


if(TRUE%in%response_capacity_SWS$in_eco_corridor==FALSE|!(max(response_capacity_S
WS$ecorestopot)%in%response_capacity_SWS$ecorestopot[which(response_capacity_SWS
$in_eco_corridor==T)])) #When there are no more wetlands intersecting ecological corridors
or when the wetlands of highest ecological restoration potential per ha are not in ecological
corridors
 {
  if(cur_surf<=(obj_surf/2)&ss_scenario=="SF") #And if we are below 100% of the impacted
area and in the SF scenario
  {
   chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$ecorestopot==max(respons
e_capacity_SWS$ecorestopot)&response_capacity_SWS$in_eco_corridor==F),]$unique_id,1
) #Selecting the wetlands out of ecological corridors and with the highest ecological
restoration potential per ha (in decreasing order but note when the last wetland of highest
potential has been selected, if the new wetland of highest potential is intersecting ecological
corridors, the loop jumps back into the previous if, but should come back here if, again, the
```

```
new wetland of highest potential will be out of ecological corridors or when there will be no
more wetlands intersecting ecological corridors)
   } else  #Otherwise (beyond 100% of the impacted area in the SF scenario and in the S
scenario in general)
   {
     if(TRUE%in%response_capacity_SWS$in_eco_corridor) #If there are still wetlands
intersecting ecological corridors, randomly selecting wetlands intersecting ecological
corridors
     {
     chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$in_eco_corridor==T),]$uniq
ue_id,1)
     } else #If there are no more wetlands intersecting ecological corridors, randomly selecting
wetlands out of ecological corridors
     {
     chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$in_eco_corridor==F),]$uniq
ue_id,1)
     }
   }
  }
}

selection_WET<-
rbind(selection_WET,response_capacity_SWS[which(response_capacity_SWS$unique_id==c
hosen_id),]) #Adding the selected wetland to the object selection_WET
response_capacity_SWS<-response_capacity_SWS[-
which(response_capacity_SWS$unique_id==chosen_id),] #Removing the selected wetland
from the response capacity pool
cur_surf<-sum(selection_WET$surf_ha) #Calculating the current area of the group of
wetlands choosen as offsetting measures (i.e. of object selection_WET)

}

#When the area of the objective need is reached or exceeded
delta_surf<-cur_surf-obj_surf #Looking at the delta

selection_WET$trunc<-F #By default, the wetlands are not cropped

if(delta_surf>0.5) #If the area of the objective need is exceeded by more than 0.5 ha
{
#Looking at the difference between the area of the last selected wetland and the delta
 if((selection_WET[nrow(selection_WET),]$surf_ha-delta_surf)>=0.5) #If the remaining area
is greater than or equal to 0.5 ha, we crop the last selected wetland
 {
```

```
 selection_WET[nrow(selection_WET),]$surf_ha<-
selection_WET[nrow(selection_WET),]$surf_ha-delta_surf #Cropping the last selected
wetland (i.e. reducing its area)
 selection_WET[nrow(selection_WET),]$ecorestopot_polygon<-
selection_WET[nrow(selection_WET),]$surf_ha*selection_WET[nrow(selection_WET),]$ecor
estopot #Updating the area of the last selected wetland
 selection_WET[nrow(selection_WET),]$trunc<-T #Flagging the last selected wetland as
altered
  }  else #If the remaining area is below 0.5 ha, we drop the last selected wetland
  {
    selection_WET<-selection_WET[-nrow(selection_WET),]
  }
cur_surf<-sum(selection_WET$surf_ha) #Updating the current area of the group of wetlands
choosen as offsetting measures (i.e. of object selection_WET)
}

#Storing the results during the simulations
res$sim_id<-c(res$sim_id,i) #The ID of the simulation
res$SWS<-c(res$SWS,SWS) #The studied sub-watershed
res$offsetting_need_surf<-c(res$offsetting_need_surf,obj_surf) #The area of the objective
need of the sub-watershed
res$offsetting_need_ecovalue<-
c(res$offsetting_need_ecovalue,offsetting_need$ecovalue[which(offsetting_need$sws_nam
e==SWS)]) #The absolute ecological score of the offsetting need of the sub-watershed
res$response_capacity_surf<-
c(res$response_capacity_surf,response_capacity$surf_ha[which(response_capacity$sws_na
me==SWS)]) #The area of the ecological restoration potential of the sub-watershed
res$response_capacity_ecorestopot<-
c(res$response_capacity_ecorestopot,response_capacity$ecorestopot[which(response_cap
acity$sws_name==SWS)]) #The absolute ecological restoration potential of the sub-
watershed

##Calculating the areas of the selected wetlands
res$cur_surf_in_eco_corridor<-
c(res$cur_surf_in_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corridor=
=T),]$surf_ha)) #The areas of the selected wetlands intersecting ecological corridors
res$cur_surf_out_eco_corridor<-
c(res$cur_surf_out_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corridor
==F),]$surf_ha)) #The areas of the selected wetlands out of ecological corridors
res$cur_surf<-c(res$cur_surf, sum(selection_WET$surf_ha)) #The total area of the selected
wetlands

##Calculating the ecological score that would result from the restoration of the selected
wetlands
res$ecorestopot_in_eco_corridor<-
c(res$ecorestopot_in_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corri
```

```
dor==T),]$ecorestopot_polygon)) #The ecological score that would result from the
restoration of the selected wetlands intersecting ecological corridors
res$ecorestopot_out_eco_corridor<-
c(res$ecorestopot_out_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corr
idor==F),]$ecorestopot_polygon)) #The ecological score that would result from the
restoration of the selected out of ecological corridors
res$ecorestopot<-c(res$ecorestopot, sum(selection_WET$ecorestopot_polygon)) #The total
ecological score that would result from the restoration of the selected wetlands

res$nb_WET<-c(res$nb_WET,nrow(selection_WET)) #Number of selected wetlands that may
be a proxy for the number of landowners

} #End of the loop for the sub-watersheds

print(paste(i,"sur", nsimus))
} #End of the loop for the N simulations

#Calculating the net ecological score after the implementation of the offsetting measures for
all the sub-watersheds
res$net_ecovalue<-res$ecorestopot-res$offsetting_need_ecovalue

res_table<-data.frame(res) #Creating a table of the results
res_table

#Creating figures (at the study area scale + at each sub-watershed scale), beware of
changing the scales and increments if necessary: ylim=c(..,..), seq(..,..,..)
# basic : S_prio_eco_corridor_with6_with0, SF_prio_eco_corridor_with6_without0, a
F+100S in the other script
# test_prio : S_no_prio_eco_corridor_with6_with0,
SF_no_prio_eco_corridor_with6_without0, a F+100S in the other script
# test_without6 : S_prio_eco_corridor_without6_with0,
SF_prio_eco_corridor_without6_without0, a F+100S in the other script

# Figure 6: the 3 “basic”
# Suppl mat. 4 = 1: the 3 “basic”, 2: the 3 “test_prio”, 3: the 3 “test_without6”

# rm(list=ls())
wd<-"Simulations/Outputs/SF_prio_eco_corridor_without6_without0/" #please choose
setwd(wd)
load(file=paste(getwd(),"/res.Rdata",sep=""))
pdf("SF_prio_eco_corridor_without6_without0_12_2020.pdf") #please choose

## Figure for all the SWS together
ecovalue<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
```

```
surface<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#windows()
boxplot(ecovalue, surface, main="ALL", col = c("grey70","white"), ylab="% relative to
impacted ecological score and
area",xaxt="n",ylim=c(100,220),yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1,2),tick = T,labels = c("Ecological score","Area"),cex.lab=4)
axis(side=2,at = seq(100,220,20),tick = T,labels = seq(100,220,20), cex.axis=0.95,las=2)
abline(h=100,lty=2)

# Figure per SWS
name_SWS<-NULL
ecovalue_SWS<-matrix(NA,nrow = 5000,length(unique(res_table$SWS)))
surface_SWS<-matrix(NA,nrow = 5000,length(unique(res_table$SWS)))
i<-1

for(SWS in unique(res_table$SWS))
{
  name_SWS[i]<-SWS
  ecovalue_SWS[,i]<-
100*tapply(res_table[which(res_table$SWS==SWS),]$ecorestopot,res_table[which(res_tabl
e$SWS==SWS),]$sim_id,sum)/tapply(res_table[which(res_table$SWS==SWS),]$offsetting_ne
ed_ecovalue,res_table[which(res_table$SWS==SWS),]$sim_id,sum)
  surface_SWS[,i]<-
2*100*tapply(res_table[which(res_table$SWS==SWS),]$cur_surf,res_table[which(res_table$
SWS==SWS),]$sim_id,sum)/tapply(res_table[which(res_table$SWS==SWS),]$offsetting_need
_surf,res_table[which(res_table$SWS==SWS),]$sim_id,sum)
  i<-i+1
}

#windows()
boxplot(ecovalue_SWS[,1],surface_SWS[,1],ecovalue_SWS[,2],surface_SWS[,2],ecovalue_S
WS[,3],surface_SWS[,3],ecovalue_SWS[,4],surface_SWS[,4],ecovalue_SWS[,5],surface_SWS[,
5],ecovalue_SWS[,6],surface_SWS[,6],ecovalue_SWS[,7],surface_SWS[,7],ecovalue_SWS[,8],
surface_SWS[,8],ecovalue_SWS[,9],surface_SWS[,9],main=NA, at =
c(1,2,3.5,4.5,6,7,8.5,9.5,11,12,13.5,14.5,16,17,18.5,19.5,21,22), names = NA, las = 1, col =
c("grey70","white"), ylab="% relative to impacted ecological score and area",xlab="Sub-
watersheds", xaxt="n",ylim=c(50,500),yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1.5,4,6.5,9,11.5,14,16.5,19,21.5),tick = F,labels = c(name_SWS),
cex.axis=0.95)
axis(side=2,at = seq(50,500,50),tick = T,labels = seq(50,500,50), cex.axis=0.95,las=2)
legend("topright",c("Ecological score","Area"),fill=c("grey55","black"),bty="n",border=NA)
abline(h=100,lty=3)
abline(v=2.75,lty=1, col="grey90")
abline(v=5.25,lty=1, col="grey90")
```

```
abline(v=7.75,lty=1, col="grey90")
abline(v=10.25,lty=1, col="grey90")
abline(v=12.75,lty=1, col="grey90")
abline(v=15.25,lty=1, col="grey90")
abline(v=17.75,lty=1, col="grey90")
abline(v=20.25,lty=1, col="grey90")

dev.off()

#Filling the synthesis table
res_synth<-
data.frame(cbind(tapply(res$offsetting_need_surf,res$SWS,mean),tapply(res$offsetting_ne
ed_ecovalue,res$SWS,mean),tapply(res$response_capacity_surf,res$SWS,mean),tapply(res$
response_capacity_ecorestopot,res$SWS,mean),tapply(res$cur_surf,res$SWS,mean),tapply(
res$cur_surf_in_eco_corridor,res$SWS,mean),tapply(res$cur_surf_out_eco_corridor,res$S
WS,mean),tapply(res$ecorestopot,res$SWS,mean),tapply(res$ecorestopot_in_eco_corridor,
res$SWS,mean),tapply(res$ecorestopot_out_eco_corridor,res$SWS,mean),tapply(res$net_e
covalue,res$SWS,mean),tapply(res$nb_WET,res$SWS,mean)))
names(res_synth)<-c("Offsetting need in area (200%)","Offsetting need in ecological
score","Response capacity in area","Response capacity in ecological restoration
potential","Total area of offsets","Area of offsets in ecological corridors","Area of offsets out
of ecological corridors","Total restored ecological score","Restored ecological score in
ecological corridors","Restored ecological score out of ecological corridors","Net ecological
score","Nb WET")
res_synth<-cbind(row.names(res_synth),res_synth)
names(res_synth)[1]<-"Sub-watershed"
row.names(res_synth)<-NULL
write.csv2(res_synth,paste(scen," Synthesis table.csv"),row.names =F)

#Saving the workspace
rm(list=ls()[-
which(ls()%in%c("res_table","res_synth","ecorestopot0","ecorestopot6","ss_scenario","prio
_eco_corridor","scen"))])
save.image("res.Rdata")
```

<u>B – Script for the scenario *Prospective*</u>

```
rm(list=ls())

#Choosing the work directory
wd<-"/Simulations/"
setwd(wd)

# #Importing the packages
# library(RPostgreSQL)
# library(gdata)
#
# drv <- dbDriver("PostgreSQL")
```

```
# con <- dbConnect(drv, dbname = "acv",
#               host = "localhost", port = 5432,
#               user = "postgres", password = "lerat")
#
# #Importing the offsetting need synthesis (map 1)
# query<-"select * from offsetting_need"
# offsetting_need<-dbGetQuery(con,query)
# str(offsetting_need)
#
# #Importing the response capacity table (map 2)
# query<-"select
unique_id,sws_name,surf_ha,ecorestopot,ecorestopot_polygon,in_eco_corridor from
response_capacity_shp"
# response_capacity_shp<-dbGetQuery(con,query)
# str(response_capacity_shp)
#
# #Importing the response capacity synthesis (map 2)
# query<-"select * from response_capacity"
# response_capacity<-dbGetQuery(con,query)
# str(response_capacity)
#
# #cleaning
# rm("con","drv","query","wd")
# save.image("E:/Analysis/ACV_esenet/REV_2020/Simulations/Data/Data needed to
simulate.RData")

load(paste(wd,"Data/","Data needed to simulate.RData",sep=""))

#Choosing the settings for the basic scenarios of the paper or the supplementary analysis
ss_scenario<-"F+100S" # F+100S (Prospective)
prio_eco_corridor<-"prio_eco_corridor" # prio_eco_corridor, no_prio_eco_corridor, please
choose
ecorestopot6<-"with6" # without6 (excluding polygons of highest ecological restoration
potential per ha), with6 (including polygons of highest ecological restoration potential per
ha), please choose
ecorestopot0<-"with0" # without0 (excluding polygons of lowest ecological restoration
potential per ha), with0 (including polygons of lowest ecological restoration potential per
ha), please choose

#Setting an output folder for this scenario
dir.create(paste(wd,"Outputs/",sep=""))
setwd(paste(wd,"/Outputs/",sep=""))
scen<-paste(ss_scenario,prio_eco_corridor,ecorestopot6,ecorestopot0,sep="_") #Creating
the name of folder variable (scenario)
dir.create(scen) #Creating the folder
setwd(paste(getwd(),scen,sep="/")) #Locating in the created folder
```

```
#Applying the settings regarding the polygons of highest and lowest ecological restoration potential per ha
if(ecorestopot6=="without6") # Polygons of highest ecological restoration potential per ha
{response_capacity_shp<-
subset(response_capacity_shp,response_capacity_shp$ecorestopot!=6)}
if(ecorestopot0=="without0") # Polygons of highest ecological restoration potential per ha
{response_capacity_shp<-
subset(response_capacity_shp,response_capacity_shp$ecorestopot!=0)}

#Indicating that sws_name and id are factors
response_capacity_shp$sws_name<-as.factor(response_capacity_shp$sws_name)
response_capacity_shp$unique_id<-as.factor(response_capacity_shp$unique_id)

#Ignoring the sub-watershed (sws) where the offsetting need is below 0.5ha
subset(offsetting_need,(surf_ha)<0.5)
offsetting_need<-subset(offsetting_need,(surf_ha)>=0.5)

#Verifying that the sub-watersheds are independent (for each sub-watershed, the objective area (twice the offsetting need in area) must fit into the response capacity in area)
(offsetting_need$surf_ha)<response_capacity[response_capacity$sws_name%in%offsetting_need$sws_name,]$surf_ha

#Calculating the number of polygons intersecting ecological corridors per sub-watershed
table(response_capacity_shp$sws_name,response_capacity_shp$in_eco_corridor)

#Creating a table of the results
res<-NULL

#Setting the number of simulation required
nsimus<-5000

#Launching loops for each sub-watershed, whatever the order because the sub-watersheds are independent
for(i in 1:nsimus)
{

  for (SWS in unique(offsetting_need$sws_name)) #loop on sub-watershed
  {

#Setting the objectives, intermediary calculation objects and indicators to follow
obj_ecovalue<-offsetting_need[which(offsetting_need$sws_name==SWS),]$ecovalue # Absolute ecological score of the objective need for the sub-watershed
obj_surf<-offsetting_need[which(offsetting_need$sws_name==SWS),]$surf_ha # Area of the objective need for the sub-watershed (100%)
response_capacity_SWS<-subset(response_capacity_shp,sws_name==SWS)  # Creating a response capacity pool for selecting the offsetting measures in the sub-watershed
```

```
cur_ecovalue<-0 #Creating the object of the sum of the absolute ecological score of the restored selected wetlands
delta_ecovalue<-0 #Creating the object of the difference between the absolute ecological score of the objective need and the current absolute ecological score of selected restored wetlands (during the simulations)
cur_surf<-0 #Creating the object of the sum of the area of selected wetlands
delta_surf<-0 #Creating the object of the difference between the area of the objective need and the current area of selected wetlands (during the simulations)
selection_WET<-NULL #Creating the object to fill with the selected wetlands (WET)
surf_full_ecovalue<-NA #Creating the object to measure the area of the selected wetlands when the ecological score of the objective need is reached or exceeded for the first time
surf_full_ecovalue_reached<-F #Creating the object verifying if the surf_full_ecovalue been reached yet or not (false by default)

#Selecting the wetlands (offsetting measures)
while (cur_ecovalue<obj_ecovalue | cur_surf<obj_surf) #Selecting wetlands as long as we are below the ecological score of the objective need OR the area of the objective need (if it has not been reached when the ecological score is reached)
{

if(prio_eco_corridor=="no_prio_eco_corridor") #If there is no priority for selecting wetlands in ecological corridors
{
   if(cur_ecovalue<obj_ecovalue) #If we are below the ecological score of the objective need
   {
     chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$ecorestopot==max(response_capacity_SWS$ecorestopot)),]$unique_id,1) #Selecting the wetlands with the highest ecological restoration potential per ha in decreasing order
   } else
   {
     chosen_id<-sample(response_capacity_SWS$unique_id,1) #Otherwise, beyond the ecological score of the objective need, randomly selecting wetlands
   }
}

  if(prio_eco_corridor=="prio_eco_corridor") #If there is a priority for selecting wetlands in ecological corridors
{

if(TRUE%in%response_capacity_SWS$in_eco_corridor&max(response_capacity_SWS$ecorestopot)%in%response_capacity_SWS$ecorestopot[which(response_capacity_SWS$in_eco_corridor==T)]) #As long as there are wetlands intersecting ecological corridors and being of the highest ecological restoration potential per ha
   {
     if(cur_ecovalue<obj_ecovalue) #If we are below the ecological score of the objective need
```

```
    {
      chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$ecorestopot==max(respons
e_capacity_SWS$ecorestopot)&response_capacity_SWS$in_eco_corridor==T),]$unique_id,1
) #Selecting the wetlands intersecting ecological corridors and with the highest ecological
restoration potential per ha (in decreasing order but note when the last wetland of highest
potential has been selected, if the new wetland of highest potential is out of ecological
corridors, the loop jumps into the next if, but should come back here if, again, the new
wetland of highest potential will be intersecting ecological corridors)
    } else
    {
      chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$in_eco_corridor==T),]$uniq
ue_id,1)  #Otherwise, if we are beyond the ecological score of the objective need, randomly
selecting wetlands intersecting ecological corridors
    }
  }


if(TRUE%in%response_capacity_SWS$in_eco_corridor==FALSE|!(max(response_capacity_S
WS$ecorestopot)%in%response_capacity_SWS$ecorestopot[which(response_capacity_SWS
$in_eco_corridor==T)])) #When there are no more wetlands intersecting ecological corridors
or when the wetlands of highest ecological restoration potential per ha are not in ecological
corridors
  {
    if(cur_ecovalue<obj_ecovalue) #If we are below the ecological score of the objective
need
    {
      chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$ecorestopot==max(respons
e_capacity_SWS$ecorestopot)&response_capacity_SWS$in_eco_corridor==F),]$unique_id,1
) #Selecting the wetlands out of ecological corridors and with the highest ecological
restoration potential per ha (in decreasing order but note when the last wetland of highest
potential has been selected, if the new wetland of highest potential is intersecting ecological
corridors, the loop jumps back into the previous if, but should come back here if, again, the
new wetland of highest potential will be out of ecological corridors or when there will be no
more wetlands intersecting ecological corridors)
    }else  #Otherwise, if we are beyond the ecological score of the objective need
    {
      if(TRUE%in%response_capacity_SWS$in_eco_corridor) #If there are still wetlands
intersecting ecological corridors, randomly selecting wetlands intersecting ecological
corridors
      {
        chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$in_eco_corridor==T),]$uniq
ue_id,1)
```

```
      } else #If there are no more wetlands intersecting ecological corridors, randomly
selecting wetlands out of ecological corridors
      {
        chosen_id<-
sample(response_capacity_SWS[which(response_capacity_SWS$in_eco_corridor==F),]$uniq
ue_id,1)
      }
     }
   }
  }

selection_WET<-
rbind(selection_WET,response_capacity_SWS[which(response_capacity_SWS$unique_id==c
hosen_id),]) #Adding the selected wetland to the object selection_WET
response_capacity_SWS<-response_capacity_SWS[-
which(response_capacity_SWS$unique_id==chosen_id),] #Removing the selected wetland
from the response capacity pool
cur_ecovalue<-sum(selection_WET$ecorestopot_polygon) #Calculating the current absolute
ecological score of the group of wetlands choosen as offsetting measures (i.e. of object
selection_WET) /// pas sûre du ecorestopot_polygon qui remplace uc_gains
cur_surf<-sum(selection_WET$surf_ha) #Calculating the current area of the group of
wetlands choosen as offsetting measures (i.e. of object selection_WET)

if (cur_ecovalue>obj_ecovalue & surf_full_ecovalue_reached==F)
{
  surf_full_ecovalue<-cur_surf
  surf_full_ecovalue_reached<-T
}

}

#Dealing with the last pick
selection_WET$trunc<-F #By default, the wetlands are not cropped
if (cur_surf>obj_surf & cur_ecovalue>obj_ecovalue) # The wetland is cropped only when the
two objectives are exceeded (overwhelming majority of cases)
{

# Case "A" where the last pick allowed to reach the ecological score objective while the area
objective was reached before the last polygon

if (sum(selection_WET$surf_ha[-nrow(selection_WET)])>=obj_surf)
  {
    delta_ecovalue<-cur_ecovalue-obj_ecovalue
    remaining_ecovalue<-selection_WET$ecorestopot_polygon[nrow(selection_WET)]-
delta_ecovalue # Ecological score that was the target of the last polygon
    remaining_surf<-remaining_ecovalue/selection_WET$ecorestopot[nrow(selection_WET)]
# Area that was the target of the last polygon
```

```
   if(remaining_surf>=0.5) # We crop according to the exact target
   {
   selection_WET$ecorestopot_polygon[nrow(selection_WET)]<-remaining_ecovalue
   selection_WET$surf_ha[nrow(selection_WET)]<-remaining_surf
   } else # We crop by keeping exactly 0.5 ha, we accept to exceed the area objective
(actually already exceeded) of 0.5 ha and to exceed a bit the ecological score
   {
   selection_WET$surf_ha[nrow(selection_WET)]<-0.5
   selection_WET$ecorestopot_polygon[nrow(selection_WET)]<-
0.5*selection_WET$ecorestopot[nrow(selection_WET)]
   }
   }

#Cases "B", "D" and "E" where the last pick allowed to reach the 2 objectives
if (sum(selection_WET$ecorestopot_polygon[-
nrow(selection_WET)])<obj_ecovalue&sum(selection_WET$surf_ha[-
nrow(selection_WET)])<obj_surf)  #True for "B" and "D"
{

# We test the impact of cropping the last polygon according to the area objective, and then
we look the positioning according to the ecological score objective
surf_polys_precedents<-sum(selection_WET$surf_ha[1:(nrow(selection_WET)-1)])
delta_surf<-cur_surf-obj_surf
remaining_surf<-selection_WET$surf_ha[nrow(selection_WET)]-delta_surf
surf_tot_with_remaining<-surf_polys_precedents+remaining_surf
ecovalue_tot_with_remaining_surf<-
sum(selection_WET$ecorestopot_polygon[1:(nrow(selection_WET)-
1)])+remaining_surf*selection_WET$ecorestopot_polygon[nrow(selection_WET)]


  if(ecovalue_tot_with_remaining_surf<obj_ecovalue) #Then we are in the case "B", the
ecological score is not yet reached, it is the same process than on the case "A"
  {
   delta_ecovalue<-cur_ecovalue-obj_ecovalue
   remaining_ecovalue<-selection_WET$ecorestopot_polygon[nrow(selection_WET)]-
delta_ecovalue # Ecological score that was the target of the last polygon
   remaining_surf<-remaining_ecovalue/selection_WET$ecorestopot[nrow(selection_WET)]
# Area that was the target of the last polygon

   if(remaining_surf>=0.5) # We crop according to the exact target
   {
     selection_WET$ecorestopot_polygon[nrow(selection_WET)]<-remaining_ecovalue
     selection_WET$surf_ha[nrow(selection_WET)]<-remaining_surf
   } else # We crop by keeping exactly 0.5 ha, we accept to exceed the area objective
(actually already exceeded) of 0.5 ha and to exceed a bit the ecological score
   {
```

```
    selection_WET$surf_ha[nrow(selection_WET)]<-0.5
    selection_WET$ecorestopot_polygon[nrow(selection_WET)]<-
0.5*selection_WET$ecorestopot[nrow(selection_WET)]
   }
   selection_WET[nrow(selection_WET),]$trunc<-T #Flagging the last selected wetland as
altered
  }

  if(ecovalue_tot_with_remaining_surf>=obj_ecovalue) # Then we are in the case "D" (>) or
"E" (=), we already exceed the ecological score objective or we reached it exactly, it is the
same process than on the case "C"
  {
  delta_surf<-cur_surf-obj_surf
   if(delta_surf>0.5) # If the area of the objective need is exceeded by more than 0.5 ha
   {
    # Looking at the difference between the area of the last selected wetland and the delta
    if((selection_WET[nrow(selection_WET),]$surf_ha-delta_surf)>=0.5) #If the remaining
area is greater than or equal to 0.5 ha, we crop the last selected wetland
    {
     selection_WET[nrow(selection_WET),]$surf_ha<-
selection_WET[nrow(selection_WET),]$surf_ha-delta_surf # Cropping the last selected
wetland (i.e. reducing its area)
     selection_WET[nrow(selection_WET),]$ecorestopot_polygon<-
selection_WET[nrow(selection_WET),]$surf_ha*selection_WET[nrow(selection_WET),]$ecor
estopot #Updating the ecological score of the last selected wetland
     selection_WET[nrow(selection_WET),]$trunc<-T #Flagging the last selected wetland as
altered
    } else #If the remaining area is below 0.5 ha, we drop the last selected wetland
    {
    selection_WET<-selection_WET[-nrow(selection_WET),]
    }
   }
  }
}

#Case "C" where the last pick allowed to reach the area objective while the ecological score
was reached before the last polygon

if (sum(selection_WET$ecorestopot_polygon[-nrow(selection_WET)])>=obj_ecovalue)
{
  delta_surf<-cur_surf-obj_surf
  if(delta_surf>0.5) # If the area of the objective need is exceeded by more than 0.5 ha
  {
   # Looking at the difference between the area of the last selected wetland and the delta
   if((selection_WET[nrow(selection_WET),]$surf_ha-delta_surf)>=0.5) #If the remaining
area is greater than or equal to 0.5 ha, we crop the last selected wetland
   {
```

```
    selection_WET[nrow(selection_WET),]$surf_ha<-
selection_WET[nrow(selection_WET),]$surf_ha-delta_surf # Cropping the last selected
wetland (i.e. reducing its area)
    selection_WET[nrow(selection_WET),]$ecorestopot_polygon<-
selection_WET[nrow(selection_WET),]$surf_ha*selection_WET[nrow(selection_WET),]$ecor
estopot # Updating the area of the last selected wetland
    selection_WET[nrow(selection_WET),]$trunc<-T # Flagging the last selected wetland as
altered
   } else # If the remaining area is below 0.5 ha, we drop the last selected wetland
   {
    selection_WET<-selection_WET[-nrow(selection_WET),]
   }
   }
}
cur_ecovalue<-sum(selection_WET$ecorestopot_polygon) #Updating the current absolute
ecological score of the group of wetlands choosen as offsetting measures (i.e. of object
selection_WET)
cur_surf<-sum(selection_WET$surf_ha) #Updating the current area of the group of wetlands
choosen as offsetting measures (i.e. of object selection_WET)

} #End of the block Dealing with last pick

#Storing the results during the simulations
res$sim_id<-c(res$sim_id,i) #The ID of the simulation
res$SWS<-c(res$SWS,SWS) #The studied sub-watershed
res$offsetting_need_surf<-c(res$offsetting_need_surf,obj_surf) #The area of the objective
need of the sub-watershed
res$offsetting_need_ecovalue<-
c(res$offsetting_need_ecovalue,offsetting_need$ecovalue[which(offsetting_need$sws_nam
e==SWS)]) #The absolute ecological score of the offsetting need of the sub-watershed
res$response_capacity_surf<-
c(res$response_capacity_surf,response_capacity$surf_ha[which(response_capacity$sws_na
me==SWS)]) #The area of the ecological restoration potential of the sub-watershed
res$response_capacity_ecorestopot<-
c(res$response_capacity_ecorestopot,response_capacity$ecorestopot[which(response_cap
acity$sws_name==SWS)]) #The absolute ecological restoration potential of the sub-
watershed
res$surf_full_ecovalue<-c(res$surf_full_ecovalue,surf_full_ecovalue) #The area of the
selected wetlands when the ecological score of the objective need is reached or exceeded
for the first time

##Calculating the areas of the selected wetlands
res$cur_surf_in_eco_corridor<-
c(res$cur_surf_in_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corridor=
=T),]$surf_ha)) #The areas of the selected wetlands intersecting ecological corridors
```

```
res$cur_surf_out_eco_corridor<-
c(res$cur_surf_out_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corridor
==F),]$surf_ha)) #The areas of the selected wetlands out of ecological corridors
res$cur_surf<-c(res$cur_surf, sum(selection_WET$surf_ha)) #The total area of the selected
wetlands

##Calculating the ecological score that would result from the restoration of the selected
wetlands
res$ecorestopot_in_eco_corridor<-
c(res$ecorestopot_in_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corri
dor==T),]$ecorestopot_polygon)) #The ecological score that would result from the
restoration of the selected wetlands intersecting ecological corridors
res$ecorestopot_out_eco_corridor<-
c(res$ecorestopot_out_eco_corridor,sum(selection_WET[which(selection_WET$in_eco_corr
idor==F),]$ecorestopot_polygon)) #The ecological score that would result from the
restoration of the selected out of ecological corridors
res$ecorestopot<-c(res$ecorestopot, sum(selection_WET$ecorestopot_polygon)) #The total
ecological score that would result from the restoration of the selected wetlands

res$nb_WET<-c(res$nb_WET,nrow(selection_WET)) #Number of selected wetlands that may
be a proxy for the number of landowners

} #End of the loop for the sub-watersheds

print(paste(i,"sur", nsimus))
} #End of the loop for the N simulations

#Calculating the net ecological score after the implementation of the offsetting measures for
all the sub-watersheds
res$net_ecovalue<-res$ecorestopot-res$offsetting_need_ecovalue

res_table<-data.frame(res) #Creating a table of the results

#Creating figures (at the study area scale + at each sub-watershed scale), beware of
changing the scales and increments if necessary: ylim=c(..,..), seq(..,..,..)
# basic : F+100S_prio_eco_corridor_with6_without0, a S and SF in the other script
# basic_test_prio : F+100S_no_prio_eco_corridor_with6_without0, a S and SF in the other
script
# basic_without6 :  F+100S_prio_eco_corridor_without6_without0, a S and SF in the other
script

# Figure 6 : the 3 "basic"
# Suppl mat. 4 = 1: the 3 "basic", 2: the 3 "test_prio", 3: the 3 "test_without6"

#rm(list=ls())
wd<-"/Users/acv/MyCore/1 Projets/ESNET/DATA 2020/2
Simulations/Outputs/F+100S_prio_eco_corridor_without6_without0/" #please choose
```

```
setwd(wd)
load(file=paste(getwd(),"/res.Rdata",sep=""))
pdf("F+100S_prio_eco_corridor_without6_without0_11_2020.pdf") #please choose

## Figure for all the SWS together
ecovalue<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface<-
(100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need_su
rf,res_table$sim_id,sum))

#windows()
boxplot(ecovalue, surface, main="ALL", col = c("grey70","white"), ylab="% relative to
impacted ecological score and
area",xaxt="n",ylim=c(100,170),yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1,2),tick = T,labels = c("Ecological score","Area"),cex.lab=4)
axis(side=2,at = seq(100,170,10),tick = T,labels = seq(100,170,10), cex.axis=0.95,las=2)
abline(h=100,lty=2)

## Figure per SWS
name_SWS<-NULL
ecovalue_SWS<-matrix(NA,nrow = 5000,length(unique(res_table$SWS)))
surface_SWS<-matrix(NA,nrow = 5000,length(unique(res_table$SWS)))
i<-1

for(SWS in unique(res_table$SWS))
{
  name_SWS[i]<-SWS
  ecovalue_SWS[,i]<-
100*tapply(res_table[which(res_table$SWS==SWS),]$ecorestopot,res_table[which(res_tabl
e$SWS==SWS),]$sim_id,sum)/tapply(res_table[which(res_table$SWS==SWS),]$offsetting_ne
ed_ecovalue,res_table[which(res_table$SWS==SWS),]$sim_id,sum)
  surface_SWS[,i]<-
100*tapply(res_table[which(res_table$SWS==SWS),]$cur_surf,res_table[which(res_table$S
WS==SWS),]$sim_id,sum)/tapply(res_table[which(res_table$SWS==SWS),]$offsetting_need
_surf,res_table[which(res_table$SWS==SWS),]$sim_id,sum)
  i<-i+1
}

#windows()
boxplot(ecovalue_SWS[,1],surface_SWS[,1],ecovalue_SWS[,2],surface_SWS[,2],ecovalue_S
WS[,3],surface_SWS[,3],ecovalue_SWS[,4],surface_SWS[,4],ecovalue_SWS[,5],surface_SWS[,
5],ecovalue_SWS[,6],surface_SWS[,6],ecovalue_SWS[,7],surface_SWS[,7],ecovalue_SWS[,8],
surface_SWS[,8],ecovalue_SWS[,9],surface_SWS[,9],main=NA, at =
c(1,2,3.5,4.5,6,7,8.5,9.5,11,12,13.5,14.5,16,17,18.5,19.5,21,22), names = NA, las = 1, col =
```

```
c("grey70","white"), ylab="% relative to impacted ecological score and area",xlab="Sub-
watersheds", xaxt="n",ylim=c(80,260),yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1.5,4,6.5,9,11.5,14,16.5,19,21.5),tick = F,labels = c(name_SWS),
cex.axis=0.95)
axis(side=2,at = seq(80,260,20),tick = T,labels = seq(80,260,20), cex.axis=0.95,las=2)
legend("topright",c("Ecological score","Area"),fill=c("grey55","black"),bty="n",border=NA)
abline(h=100,lty=3)
abline(v=2.75,lty=1, col="grey90")
abline(v=5.25,lty=1, col="grey90")
abline(v=7.75,lty=1, col="grey90")
abline(v=10.25,lty=1, col="grey90")
abline(v=12.75,lty=1, col="grey90")
abline(v=15.25,lty=1, col="grey90")
abline(v=17.75,lty=1, col="grey90")
abline(v=20.25,lty=1, col="grey90")

dev.off()

#Filling the synthesis table
res_synth<-
data.frame(cbind(tapply(res$offsetting_need_surf,res$SWS,mean),tapply(res$offsetting_ne
ed_ecovalue,res$SWS,mean),tapply(res$response_capacity_surf,res$SWS,mean),tapply(res$
response_capacity_ecorestopot,res$SWS,mean),tapply(res$cur_surf,res$SWS,mean),tapply(
res$cur_surf_in_eco_corridor,res$SWS,mean),tapply(res$cur_surf_out_eco_corridor,res$S
WS,mean),tapply(res$ecorestopot,res$SWS,mean),tapply(res$ecorestopot_in_eco_corridor,
res$SWS,mean),tapply(res$ecorestopot_out_eco_corridor,res$SWS,mean),tapply(res$net_e
covalue,res$SWS,mean),tapply(res$surf_full_ecovalue,res$SWS,mean,na.rm=T),tapply(res$
nb_WET,res$SWS,mean))) ## // check que j'ai bien fait pour le surf_full_ecovalue na.rm =T
je ne sais pas ce que ça veut dire
names(res_synth)<-c("Offsetting need in area (200%)","Offsetting need in ecological
score","Response capacity in area","Response capacity in ecological restoration
potential","Total area of offsets","Area of offsets in ecological corridors","Area of offsets out
of ecological corridors","Total restored ecological score","Restored ecological score in
ecological corridors","Restored ecological score out of ecological corridors","Net ecological
score","Area when the ecological score of the offsetting need is reached for the first
time","Nb WET")
res_synth<-cbind(row.names(res_synth),res_synth)
names(res_synth)[1]<-"Sub-watershed"
row.names(res_synth)<-NULL
write.csv2(res_synth,paste(scen," Synthesis table.csv"),row.names =F)

#Saving the workspace
rm(list=ls()[-
which(ls()%in%c("res_table","res_synth","ecorestopot0","ecorestopot6","ss_scenario","prio
_eco_corridor","scen"))])
save.image("res.Rdata")
```

C – Script for the supplementary analysis (Suppl. Mat. 4)

```
#Creating comparison figures, please choose and change the names in pdf creation and load
(beware of changing the scales and increments if necessary: ylim=c(..,..), seq(..,..,..))
# comparison_basic : S_prio_eco_corridor_with6_with0,
SF_prio_eco_corridor_with6_without0, F+100S_prio_eco_corridor_with6_without0
# comparison_basic_test_prio : S_no_prio_eco_corridor_with6_with0,
SF_no_prio_eco_corridor_with6_without0, F+100S_no_prio_eco_corridor_with6_without0
# comparison_basic_without6 : S_prio_eco_corridor_without6_with0,
SF_prio_eco_corridor_without6_without0, F+100S_prio_eco_corridor_without6_without0

###### Comparison of the basic scenarios - Figure 5
# comparison_basic : S_prio_eco_corridor_with6_with0,
SF_prio_eco_corridor_with6_without0, F+100S_prio_eco_corridor_with6_without0

pdf("comparison_basic_11_2020.pdf") #please choose

#1 - Scenario S
load(file=paste(getwd(),"/Outputs/S_prio_eco_corridor_with6_with0/res.Rdata",sep=""))
#please choose

name_1<-"Old way"
ecovalue_1<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_1<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#2 - Scenario SF
load(file=paste(getwd(),"/Outputs/SF_prio_eco_corridor_with6_without0/res.Rdata",sep=""
)) #please choose

name_2<-"Local river basin plan"
ecovalue_2<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_2<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#3 - Scenario F+100S
load(file=paste(getwd(),"/Outputs/F+100S_prio_eco_corridor_with6_without0/res.Rdata",s
ep="")) #please choose

name_3<-"Prospective"
```

```
ecovalue_3<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_3<-
(100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need_su
rf,res_table$sim_id,sum))

#4 - Scenario landscape scale
name_4<-"Landscape scale"
ecovalue_4<-117 # manually enter the value for the landscape scale scenario
surface_4<-203 # manually enter the value for the landscape scale scenario

#windows()

par(mar=c(3,4,1,1))
boxplot(ecovalue_1, surface_1,ecovalue_2, surface_2,ecovalue_3, surface_3,ecovalue_4,
surface_4, main=NA, at = c(1,2,3.5,4.5,6,7,8.5,9.5), names = NA, las = 1, col =
c("grey70","white"), ylab="% relative to impacted ecological score and
area",xaxt="n",ylim=c(100,220), yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1.5,4,6.5,9),tick = F,labels =
c(name_1,name_2,name_3,name_4),cex.lab=4)
axis(side=2,at = seq(100,220,20),tick = T,labels = seq(100,220,20), cex.axis=0.95,las=2)
legend("topright",c("Ecological score","Area"),fill=c("grey55","black"),bty="n",border=NA)
abline(h=100,lty=2)
abline(v=2.75,lty=1, col="grey90")
abline(v=5.25,lty=1, col="grey90")
abline(v=7.75,lty=1, col="grey90")

dev.off()

###### Analysis of the effect of prioritizing the location of offsetting measures in ecological
corridors - Figure 8
# comparison_basic_test_prio : S_no_prio_eco_corridor_with6_with0,
SF_no_prio_eco_corridor_with6_without0, F+100S_no_prio_eco_corridor_with6_without0

pdf("comparison_basic_test_prio_11_2020.pdf") #please choose

#1 - Scenario S
load(file=paste(getwd(),"/Outputs/S_prio_eco_corridor_with6_with0/res.Rdata",sep=""))
#please choose

name_1<-"Old way \nprio"
ecovalue_1<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
```

```
surface_1<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

load(file=paste(getwd(),"/Outputs/S_no_prio_eco_corridor_with6_with0/res.Rdata",sep="")
)

name_1test<-"Old way \nno prio"
ecovalue_1test<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_1test<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#2 - Scenario SF
load(file=paste(getwd(),"/Outputs/SF_prio_eco_corridor_with6_without0/res.Rdata",sep="""
)) #please choose

name_2<-"LRBP \nprio"
ecovalue_2<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_2<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

load(file=paste(getwd(),"/Outputs/SF_no_prio_eco_corridor_with6_without0/res.Rdata",se
p="")) #please choose

name_2test<-"LRBP \nno prio"
ecovalue_2test<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_2test<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#3 - Scenario F+100S
load(file=paste(getwd(),"/Outputs/F+100S_prio_eco_corridor_with6_without0/res.Rdata",s
ep="")) #please choose

name_3<-"Prospective \nprio"
ecovalue_3<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
```

```
surface_3<-
(100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need_su
rf,res_table$sim_id,sum))

load(file=paste(getwd(),"/Outputs/F+100S_no_prio_eco_corridor_with6_without0/res.Rdat
a",sep="")) #please choose

name_3test<-"Prospective \nno prio"
ecovalue_3test<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_3test<-
(100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need_su
rf,res_table$sim_id,sum))

#4 - Scenario landscape scale
name_4<-"Land. scale \nprio"
ecovalue_4<-117 # manually enter the value for the landscape scale scenario
surface_4<-203 # manually enter the value for the landscape scale scenario

#windows()

par(mar=c(3,4,1,1))
boxplot(ecovalue_1, surface_1, ecovalue_1test, surface_1test,ecovalue_2,
surface_2,ecovalue_2test, surface_2test,ecovalue_3, surface_3,ecovalue_3test,
surface_3test,ecovalue_4, surface_4, main=NA, at =
c(1,2,3.5,4.5,6,7,8.5,9.5,11,12,13.5,14.5,16,17), names = NA, las = 1, col =
c("grey70","white"), ylab="% relative to impacted ecological score and
area",xaxt="n",ylim=c(100,220), yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1.5,4,6.5,9,11.5,14,16.5),tick = F,labels =
c(name_1,name_1test,name_2,name_2test,name_3,name_3test,name_4),cex.axis=0.75)
axis(side=2,at = seq(100,220,20),tick = T,labels = seq(100,220,20), cex.axis=0.95,las=2)
legend("topright",c("Ecological score","Area"),fill=c("grey55","black"),bty="n",border=NA)
abline(h=100,lty=2)
abline(v=2.75,lty=2, col="grey90")
abline(v=5.25,lty=1, col="grey90")
abline(v=7.75,lty=2, col="grey90")
abline(v=10.25,lty=1, col="grey90")
abline(v=12.75,lty=2, col="grey90")
abline(v=15.25,lty=1, col="grey90")

dev.off()

###### Analysis of the effect of removing parcels with no grassland (best agricultural land)
from the possible offsetting parcels - Figure 9
```

```
# comparison_basic_without6 : S_prio_eco_corridor_without6_with0,
SF_prio_eco_corridor_without6_without0, F+100S_prio_eco_corridor_without6_without0

pdf("comparison_basic_without6_11_2020.pdf") #please choose

#1 - Scenario S
load(file=paste(getwd(),"/Outputs/S_prio_eco_corridor_with6_with0/res.Rdata",sep=""))
#please choose

name_1<-"Old way \n"
ecovalue_1<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_1<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

load(file=paste(getwd(),"/Outputs/S_prio_eco_corridor_without6_with0/res.Rdata",sep=""))

name_1test<-"Old way \nparc. excl."
ecovalue_1test<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_1test<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#2 - Scenario SF
load(file=paste(getwd(),"/Outputs/SF_prio_eco_corridor_with6_without0/res.Rdata",sep=""
)) #please choose

name_2<-"LRBP \n"
ecovalue_2<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_2<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

load(file=paste(getwd(),"/Outputs/SF_prio_eco_corridor_without6_without0/res.Rdata",sep
="")) #please choose

name_2test<-"LRBP \nparc. excl."
ecovalue_2test<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
```

```
surface_2test<-
(2*100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need
_surf,res_table$sim_id,sum))

#3 - Scenario F+100S
load(file=paste(getwd(),"/Outputs/F+100S_prio_eco_corridor_with6_without0/res.Rdata",s
ep="")) #please choose

name_3<-"Prospective \n"
ecovalue_3<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_3<-
(100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need_su
rf,res_table$sim_id,sum))

load(file=paste(getwd(),"/Outputs/F+100S_prio_eco_corridor_without6_without0/res.Rdata
",sep="")) #please choose

name_3test<-"Prospective \nparc. excl."
ecovalue_3test<-
(100*tapply(res_table$ecorestopot,res_table$sim_id,sum)/tapply(res_table$offsetting_nee
d_ecovalue,res_table$sim_id,sum))
surface_3test<-
(100*tapply(res_table$cur_surf,res_table$sim_id,sum)/tapply(res_table$offsetting_need_su
rf,res_table$sim_id,sum))

#4 - Scenario landscape scale
name_4<-"Land. scale \n"
ecovalue_4<-117 # manually enter the value for the landscape scale scenario
surface_4<-203 # manually enter the value for the landscape scale scenario

#windows()

par(mar=c(3,4,1,1))
boxplot(ecovalue_1, surface_1, ecovalue_1test, surface_1test,ecovalue_2,
surface_2,ecovalue_2test, surface_2test,ecovalue_3, surface_3,ecovalue_3test,
surface_3test,ecovalue_4, surface_4, main=NA, at =
c(1,2,3.5,4.5,6,7,8.5,9.5,11,12,13.5,14.5,16,17), names = NA, las = 1, col =
c("grey70","white"), ylab="% relative to impacted ecological score and
area",xaxt="n",ylim=c(70,220), yaxt="n",border=c("grey55","black"),outline=F)
axis(side=1,at = c(1.5,4,6.5,9,11.5,14,16.5),tick = F,labels =
c(name_1,name_1test,name_2,name_2test,name_3,name_3test,name_4),cex.axis=0.75)
axis(side=2,at = seq(70,220,20),tick = T,labels = seq(70,220,20), cex.axis=0.95,las=2)
legend("topright",c("Ecological score","Area"),fill=c("grey55","black"),bty="n",border=NA)
abline(h=100,lty=2)
abline(v=2.75,lty=2, col="grey90")
```

```
abline(v=5.25,lty=1, col="grey90")
abline(v=7.75,lty=2, col="grey90")
abline(v=10.25,lty=1, col="grey90")
abline(v=12.75,lty=2, col="grey90")
abline(v=15.25,lty=1, col="grey90")

dev.off()
```