Appendix 2

R code to simulate natural resource management in the presence of conflict, lobbying and non-compliance.

Below we provide an annotated R script that can be used to replicate the simulations presented in this study.

The only R package required to run a simulation is *GMSE* (Duthie et al. 2018).

```
require('GMSE')
```

The following values can be varied to test the effect of target, budget, decision-making bias and user compliance on management outcome:

```
M.TGT = 1000                # Manager target
U.TGT = 0                   # User target
C.TGT = 2000                # Conservationist target

MB = 10000                  # Manager budget

decision.bias = 0           # Decision bias level
user.compliance = 0         # User compliance level

nsteps = 10                 # Number of time steps
minc = 10                   # Minimum cost of an action
```

The following functions define the population growth model, the observation model, lobbying pressure by users and conservationists, and illegal harvesting pressure by users.

```
### Population growth model

pop_model <- function(X,
```

```
                      K,
                      ig){
  Xn <-
round((X*K*exp(rnorm(1,ig,ig/10)))/(K+X*(exp(rnorm(1,ig,ig/10))-1)))
  return(Xn)
}

### Observation model

obs_model <- function(resource_vector){
      X_obs <- resource_vector
      return(X_obs)
}

### Conservation lobbying function

ClobbyingF <- function(bias_level,
                       conservation_target,
                       population_size){
  if (population_size > conservation_target){
    res <- 0
  }
  else{
    res <- ((((1-
bias_level)+1)^(1/conservation_target))^(conservation_target-
population_size))-1
  }
  return(res)
}

### User lobbying function

UlobbyingF <- function(bias_level,
                       user_target,
                       population_size,
                       carrying_capacity){
  if (population_size < user_target){
    res <- 0
  }
  else{
    res <- ((((1-bias_level)+1)^(1/(carrying_capacity-
user_target)))^(population_size-user_target))-1
  }
  return(res)
}

### Illegal offtake function

IllegalHarvestF <- function(compliance_level,
                            user_target,
                            population_size,
                            carrying_capacity){
  if (population_size < user_target){
    res <- 0
  }
  else{
```

```
    res <- ((((1-compliance_level)+1)^(1/(carrying_capacity-
user_target)))^(population_size-user_target))-1
  }
  return(res)
}
```

Population parameters are set to:

```
Kk = 2000                    # Wildlife population carrying capacity
rmax = 0.2                   # Wildlife population intrinsic growth
rate
```

The code below runs one management iteration of `nsteps` time steps under user lobbying for given values of `M.TGT`, `U.TGT`, `C.TGT`, `MB`, `decision.bias`, `user.compliance` and `minc` specified above. All other values appearing in the calls to gmse_apply not defined above are default values as described in [1].

```
# Sample user budget for the set of time steps

UB <- sample(5000:10000,
             size=nsteps,
             replace=T)

# Run initial gmse_apply (the manager calls the genetic algorithm)

sim1 <- gmse_apply(res_mod = pop_model,
                   obs_mod = obs_model,
                   K = Kk,
                   ig = rmax,
                   X = 1000,
                   user_budget = UB[1],
                   minimum_cost = minc,
                   manager_budget = MB,
                   manage_target = M.TGT,
                   scaring = F,
                   culling = T,
                   castration = F,
                   feeding = F,
                   stakeholders = 1,
                   manage_freq = 1,
                   manager_sense = 1,
                   public_land = 0,
                   land_ownership = F,
                   group_think = F,
                   ga_mingen = 200,
                   get_res = "Full")
```

```
# Extract resource abundance

Nt <- sim1$resource_vector

# Extract cost

c.t <- sim1$manager_vector

# Derive quota

q.t <- floor(UB[1]/c.t)

# Maximum possible quota given user budget

max.q <- floor(UB[1]/minc)

# Slight correction to Kk in the case that stochasticity overshoots
the
# carrying capacity

CC <- ifelse(Nt>Kk,Nt+1,Kk)

# Derive probability of unregulated harvest

phi.t <- UlobbyingF(bias_level = decision.bias,
                    user_target = U.TGT,
                    population_size = Nt,
                    carrying_capacity = CC)

# Is harvest unregulated?

P.t <- rbinom(n=1,size=1,prob=phi.t)

# If P(t)=0

if (P.t == 0){
  q.prime.t <- q.t
}

# Otherwise...

if (P.t==1){

  # The new quota is then derived as

  q.prime.t <- max.q

}

# Derive maximum possible harvest at minimum cost

max.h.t <- floor(UB[1]/minc)

# If the maximum possible harvest is smaller or equal to the lobbied
quota
# No need to poach as harvesting is unregulated
```

```
if (max.h.t <= q.prime.t){
  h.t <- max.h.t
}

else{

  # Slight correction to Kk in the case that stochasticity overshoots
the
  # carrying capacity

  CC <- ifelse(Nt>Kk,Nt+1,Kk)

  # Derive probability of illegal harvest

  psi.t <- IllegalHarvestF(compliance_level = user.compliance,
                           user_target = U.TGT,
                           population_size = Nt,
                           carrying_capacity = CC)

  # Derive illegal harvest based on max harvest and probability

  Y.t <- sum(rbinom(n=max.h.t,size=1,prob=psi.t))

  if (Y.t==q.prime.t){
    h.t <- q.prime.t
  }

  else{
    h.t <- max(c(q.prime.t,Y.t))
  }

}

# Create results data.frame

results   <- matrix(dat = NA, nrow = nsteps, ncol = 9)

results[1,1] <- sim1$resource_vector;                 # Number of
resources
results[1,2] <- sim1$observation_vector;     # Observed number of
resources
results[1,3] <- q.t                       # Harvesting quota before
lobbying
results[1,4] <- q.prime.t                  # Harvesting quota after
lobbying
results[1,5] <- h.t                          # Harvest after illegal
offtake
results[1,6] <- decision.bias                              # Manager
bias level
results[1,7] <- user.compliance                       # User compliance
level
results[1,8] <- 1                                                  #
Time step
results[1,9] <- UB[1]                                           # User
budget

# Apply harvest
```

```
sim1$X <- sim1$resource_vector-h.t

#####################################
### Run through rest of time steps ##
#####################################

for (time_step in 2:nsteps){

  # Run gmse_apply

  sim_new <- tryCatch(gmse_apply(old_list = sim1,
                                 res_mod = pop_model,
                                 obs_mod = obs_model,
                                 get_res = "Full",
                                 user_budget = UB[time_step]),
                      error=function(err) NA)

  if (is.na(sim_new)==T){     # If the resource goes extinct

    # Add results to data.frame

    results[time_step:nsteps,1] <- 0
    results[time_step:nsteps,2] <- 0
    results[time_step:nsteps,3] <- NA
    results[time_step:nsteps,4] <- NA
    results[time_step:nsteps,5] <- NA
    results[time_step:nsteps,6] <- decision.bias
    results[time_step:nsteps,7] <- user.compliance
    results[time_step:nsteps,8] <- time_step:nsteps
    results[time_step:nsteps,9] <- NA

    print("Extinction")

    break

  }

  if (class(sim_new)=="list"){     # If the resource does not go
extinct

    # Extract resource abundance

    Nt <- sim_new$resource_vector

    # Extract cost

    c.t <- sim_new$manager_vector

    # Derive quota

    q.t <- floor(UB[time_step]/c.t)

    # Maximum possible quota given user budget

    max.q <- floor(UB[time_step]/minc)
```

```
    # Slight correction to Kk in the case that stochasticity
overshoots the
    # carrying capacity

    CC <- ifelse(Nt>Kk,Nt+1,Kk)

    # Derive probability of unregulated harvest

    phi.t <- UlobbyingF(bias_level = decision.bias,
                        user_target = U.TGT,
                        population_size = Nt,
                        carrying_capacity = CC)

    # Is harvest unregulated?

    P.t <- rbinom(n=1,size=1,prob=phi.t)

    # If P(t)=0

    if (P.t == 0){
      q.prime.t <- q.t
    }

    # Otherwise...

    if (P.t==1){

      # The new quota is then derived as

      q.prime.t <- max.q

    }

    # Derive maximum possible harvest at minimum cost

    max.h.t <- floor(UB[time_step]/minc)

    # If the maximum possible harvest is smaller or equal to lobbied
quota
    # No need to poach as harvesting is unregulated

    if (max.h.t <= q.prime.t){
      h.t <- max.h.t
    }

    else{

      # Slight correction to Kk in the case that stochasticity
overshoots
      # the carrying capacity

      CC <- ifelse(Nt>Kk,Nt+1,Kk)

      # Derive probability of illegal harvest

      psi.t <- IllegalHarvestF(compliance_level = user.compliance,
                               user_target = U.TGT,
```

```
                                population_size = Nt,
                                carrying_capacity = CC)

      # Derive illegal harvest based on max harvest and probability

      Y.t <- sum(rbinom(n=max.h.t,size=1,prob=psi.t))

      if (Y.t==q.prime.t){
        h.t <- q.prime.t
      }

      else{
        h.t <- max(c(q.prime.t,Y.t))
      }

    }

    # Add results to data frame
    results[time_step,1] <- sim_new$resource_vector;
    results[time_step,2] <- sim_new$observation_vector;
    results[time_step,3] <- q.t
    results[time_step,4] <- q.prime.t
    results[time_step,5] <- h.t
    results[time_step,6] <- decision.bias
    results[time_step,7] <- user.compliance
    results[time_step,8] <- time_step
    results[time_step,9] <- UB[time_step]

    # Apply harvest

    sim_new$X <- sim_new$resource_vector-h.t;

    sim1 <- sim_new

  }

}
```

The following code runs one management iteration of `nsteps` time steps under conservationist lobbying for given values of `M.TGT`, `U.TGT`, `C.TGT`, `MB`, `decision.bias`, `user.compliance` and `minc` specified above. All other values appearing in the calls to gmse_apply not defined above are default values as described in [1].

```
# Sample user budget for the set of time steps

UB <- sample(5000:10000,
             size=nsteps,
```

```
             replace=T)

# Run initial gmse_apply (the manager calls the genetic algorithm)

sim1 <- gmse_apply(res_mod = pop_model,
                   obs_mod = obs_model,
                   K = Kk,
                   ig = rmax,
                   X = 1000,
                   user_budget = UB[1],
                   minimum_cost = minc,
                   manager_budget = MB,
                   manage_target = M.TGT,
                   scaring = F,
                   culling = T,
                   castration = F,
                   feeding = F,
                   stakeholders = 1,
                   manage_freq = 1,
                   manager_sense = 1,
                   public_land = 0,
                   land_ownership = F,
                   group_think = F,
                   ga_mingen = 200,
                   get_res = "Full")

# Extract resource abundance

Nt <- sim1$resource_vector

# Extract cost

c.t <- sim1$manager_vector

# Derive quota

q.t <- floor(UB[1]/c.t)

# Minimum possible quota

min.q <- 0

# Derive probability of harvest ban

phi.t <- ClobbyingF(bias_level = decision.bias,
                    conservation_target = C.TGT,
                    population_size = Nt)

# Is harvest banned?

P.t <- rbinom(n=1,size=1,prob=phi.t)

# If P(t)=0

if (P.t == 0){
  q.prime.t <- q.t
}
```

```
# Otherwise...

if (P.t==1){

  # The new quota is then derived as

  q.prime.t <- min.q

}

# Derive maximum possible harvest at minimum cost

max.h.t <- floor(UB[1]/minc)

# If the maximum possible harvest is smaller or equal to the lobbied
quota
# No need to poach as harvesting is unregulated

if (max.h.t <= q.prime.t){
  h.t <- max.h.t
}

else{

  # Slight correction to Kk in the case that stochasticity overshoots
the
  # carrying capacity

  CC <- ifelse(Nt>Kk,Nt+1,Kk)

  # Derive probability of illegal harvest

  psi.t <- IllegalHarvestF(compliance_level = user.compliance,
                           user_target = U.TGT,
                           population_size = Nt,
                           carrying_capacity = CC)

  # Derive illegal harvest based on max harvest and probability

  Y.t <- sum(rbinom(n=max.h.t,size=1,prob=psi.t))

  if (Y.t==q.prime.t){
    h.t <- q.prime.t
  }

  else{
    h.t <- max(c(q.prime.t,Y.t))
  }

}

# Create results data.frame

results   <- matrix(dat = NA, nrow = nsteps, ncol = 9)
```

```
results[1,1] <- sim1$resource_vector;                 # Number of
resources
results[1,2] <- sim1$observation_vector;     # Observed number of
resources
results[1,3] <- q.t                       # Harvesting quota before
lobbying
results[1,4] <- q.prime.t                  # Harvesting quota after
lobbying
results[1,5] <- h.t                          # Harvest after illegal
offtake
results[1,6] <- manager.bias                            # Manager bias
level
results[1,7] <- user.compliance                     # User compliance
level
results[1,8] <- 1                                                  #
Time step
results[1,9] <- UB[1]                                            # User
budget

# Apply harvest

sim1$X <- sim1$resource_vector-h.t

#####################################
### Run through rest of time steps ##
#####################################

for (time_step in 2:nsteps){

  # Run gmse_apply

  sim_new <- tryCatch(gmse_apply(old_list = sim1,
                                 res_mod = pop_model,
                                 obs_mod = obs_model,
                                 get_res = "Full",
                                 user_budget = UB[time_step]),
                      error=function(err) NA)

  if (is.na(sim_new)==T){     # If the resource goes extinct

    # Add results to data.frame

    results[time_step:nsteps,1] <- 0
    results[time_step:nsteps,2] <- 0
    results[time_step:nsteps,3] <- NA
    results[time_step:nsteps,4] <- NA
    results[time_step:nsteps,5] <- NA
    results[time_step:nsteps,6] <- decision.bias
    results[time_step:nsteps,7] <- user.compliance
    results[time_step:nsteps,8] <- time_step:nsteps
    results[time_step:nsteps,9] <- NA

    print("Extinction")

    break

  }
```

```
  if (class(sim_new)=="list"){     # If the resource does not go
extinct

    # Extract resource abundance

    Nt <- sim_new$resource_vector

    # Extract cost

    c.t <- sim_new$manager_vector

    # Derive quota

    q.t <- floor(UB[time_step]/c.t)

    # Minimum possible quota

    min.q <- 0

    # Derive probability of harvest ban

    phi.t <- ClobbyingF(bias_level = decision.bias,
                        conservation_target = C.TGT,
                        population_size = Nt)

    # Is harvest banned?

    P.t <- rbinom(n=1,size=1,prob=phi.t)

    # If P(t)=0

    if (P.t == 0){
      q.prime.t <- q.t
    }

    # Otherwise...

    if (P.t==1){

      # The new quota is then derived as

      q.prime.t <- min.q

    }

    # Derive maximum possible harvest at minimum cost

    max.h.t <- floor(UB[time_step]/minc)

    # If the maximum possible harvest is smaller or equal to lobbied
quota
    # No need to poach as harvesting is unregulated

    if (max.h.t <= q.prime.t){
      h.t <- max.h.t
    }
```

```
    else{

      # Slight correction to Kk in the case that stochasticity
overshoots
      # the carrying capacity

      CC <- ifelse(Nt>Kk,Nt+1,Kk)

      # Derive probability of illegal harvest

      psi.t <- IllegalHarvestF(compliance_level = user.compliance,
                               user_target = U.TGT,
                               population_size = Nt,
                               carrying_capacity = CC)

      # Derive illegal harvest based on max harvest and probability

      Y.t <- sum(rbinom(n=max.h.t,size=1,prob=psi.t))

      if (Y.t==q.prime.t){
        h.t <- q.prime.t
      }

      else{
        h.t <- max(c(q.prime.t,Y.t))
      }

    }

    # Add results to data frame
    results[time_step,1] <- sim_new$resource_vector;
    results[time_step,2] <- sim_new$observation_vector;
    results[time_step,3] <- q.t
    results[time_step,4] <- q.prime.t
    results[time_step,5] <- h.t
    results[time_step,6] <- decision.bias
    results[time_step,7] <- user.compliance
    results[time_step,8] <- time_step
    results[time_step,9] <- UB[time_step]

    # Apply harvest

    sim_new$X <- sim_new$resource_vector-h.t;

    sim1 <- sim_new

  }

}
```

Literature cited

Duthie A. B., J. J. Cusack, I. J. Jones, J. Minderman, E. B. Nilsen, R. A. Pozo, R. A., S. Rakotonarivo, B. Van Moorter, and N. Bunnefeld. 2018. GMSE: an R package for generalised management strategy evaluation. *Methods in Ecology and Evolution* 9: 2396-2401.