ESM 232 - Rabbit Population Matrix

Jennifer Truong, Maggie Brickner, and Mauricio Collado

05/14/2021

1. Introduction

A small city with a large urban park has decided to introduce a rare species of rabbits into this park - Rabbits are cute, and the kids love them, and giving a rare species a new home sounds like a good idea. The urban park manager is concerned about how this rabbit population might grow over the next few decades. Rabbits have no natural predators in the region where the park is situated. The manager would like to know, approximately how many rabbits there will be 20 years from now if the rabbits are introduced as planned. The manager reviewed the literature and found the following estimates for survival and fertility rates for the rare rabbit population for 4 different age classes.

To answer the manager's question, we built a function to estimate the rabbit population considering its age groups. Then we perform a Sobel Sensitivity analysis to test the sensitivity of the total population to the survivability of young and sub-adult individuals.

The current version employs the updated evolve_pop function!

2. Using Leslie Matrices to Evolve Populations

Our first step is building a square matrix that holds the fertility and survivability for four age classes (young, sub-adults, adults and aged) of rabbit population.

```
# Set up number of age classes
nclasses = 4

# create a growth matrix to store fertility and survivability information
gmatrix=matrix(nrow=nclasses, ncol=nclasses)
#gmatrix

# change NAs to zero
gmatrix[]=0.0
#gmatrix

# assign values for fertility for each age class
# fertility numbers are big here because they are RABBITS!!
fert =  c(0,2,6,1)

# enter into our matrix
gmatrix[1,]=fert

 # now add survivability
 # survivability (to the next class) is also per time step
gmatrix[2,1]=0.8
gmatrix[3,2]=0.85
gmatrix[4,3]=0.65
```

```
# we also want to to account for the oldest population group - they don't transfer to another group
# but they do die - this will be survivability per time step but they just stay in their class/group
gmatrix[4,4]=0.1

gmatrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]  0.0 2.00 6.00  1.0
## [2,]  0.8 0.00 0.00  0.0
## [3,]  0.0 0.85 0.00  0.0
## [4,]  0.0 0.00 0.65  0.1
```

We test our matrix values for the populations in years 1 and 2. We observe a reduced population for adults and aged groups in years 1 and 2. However, the sub-adult population grows significantly in year 2. We can expect changes in the population structure until each age group finds its stable state. However, it is important to highlight that **we assumed 10 individuals for all age groups**. In the next exercise, we will evaluate for 10 adults.

```
# start with an initial population of 10
p0 = rep(10, times=nclasses)

# advance to the next time step
# note the use of matrix multiplication
p1 = gmatrix %*% p0
p1
```

```
##      [,1]
## [1,] 90.0
## [2,]  8.0
## [3,]  8.5
## [4,]  7.5
```

```
# has the total number of individuals changed?
sum(p1)
```

```
## [1] 114
```

```
sum(p0)
```

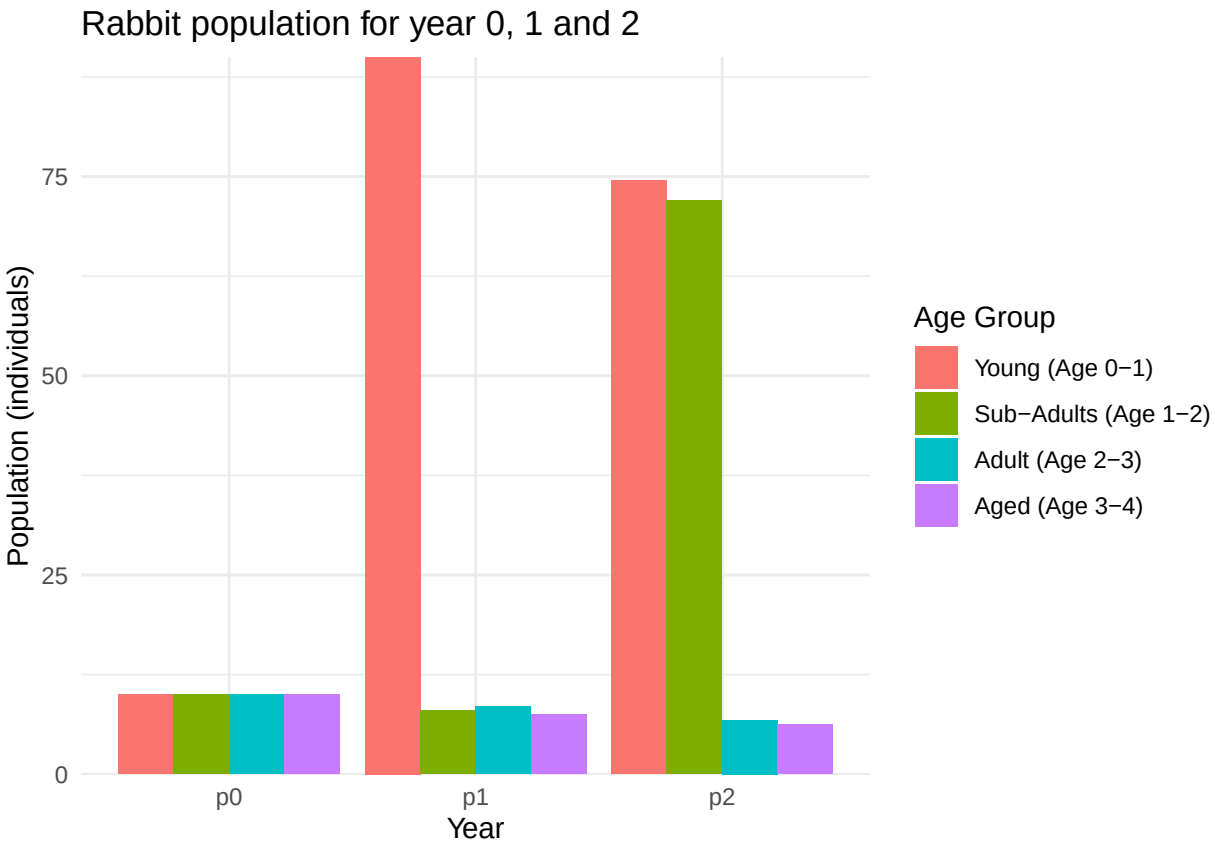
```
## [1] 40
```

```
# growth rate
sum(p1)/sum(p0)
```

```
## [1] 2.85
```

```
#add another year
p2 = gmatrix %*% p1

# combined
pop = cbind.data.frame(p0,p1,p2)
pop$age = c("Young (Age 0-1)",
            "Sub-Adults (Age 1-2)",
            "Adult (Age 2-3)",
            "Aged (Age 3-4)") %>%
  as_factor() # Change to factor class
```

```r
popl = pop %>%
  gather(key="timestep",value="pop",-age)

ggplot(popl, aes(timestep, pop,fill=as.factor(age)))+
  geom_col(position="dodge")+
  labs(title="Rabbit population for year 0, 1 and 2",
       y="Population (individuals)",
       x="Year",
       fill="Age Group") +
  theme_minimal() +
  scale_y_continuous(expand = c(0,0))
```



3. Rabbit Population in 20 Years

We use the function to evolve a population through time considering:

- Inputs = survivability, fertility, initial population, time steps
- Output = final population matrix
- A dynamic model - difference equations - similar to our diffusion model

The manager indicated us to **assume that we start with 10 adult rabbits**.

We consider the rabbit parameters are annual information. Thus, we run a function to calculate the population (See **Appendix 1**) for 20 time steps (20 years).

```r
# call the evolve population function
source(here("R/evolve_pop2.R"))
```

```r
# fertility rates
F1 = 0 #young
F2 = 2 #sub adult
F3 = 6 #adult
F4 = 1 #aged

# survivability
p12 = 0.8  #young
p23 = 0.85 #sub adult
p34 = 0.65 #adult
p44 = 0.1 #aged

# initial population parameters
ini = c(0, 0, 10, 0) # start with 10 adult rabbits
nyears = 20 # number of years (time step) to run
fert_rabbit = c(F1, F2, F3, F4) # fertility for each age class
surv_rabbit = c(p12, p23, p34, p44) # survivability for each age class

#run the equation evolve_pop(fertility, survivability, initial pop, years)
rabbit_pop=evolve_pop(fert_rabbit, surv_rabbit, ini, nyears)

#check the results
head(rabbit_pop)
```

```
## $popbyage
##      [,1] [,2]  [,3]   [,4]     [,5]      [,6]      [,7]      [,8]      [,9]
## [1,]    0 60.0  6.50 96.650 255.2650 207.68650 808.28165 1417.0515 2257.7659
## [2,]    0  0.0 48.00  5.200  77.3200 204.21200 166.14920  646.6253 1133.6412
## [3,]   10  0.0  0.00 40.800   4.4200  65.72200 173.58020  141.2268  549.6315
## [4,]    0  6.5  0.65  0.065  26.5265   5.52565  43.27187  117.1543  103.5129
##          [,10]     [,11]     [,12]     [,13]     [,14]      [,15]     [,16]
## [1,] 5668.5843 9761.6072 18944.518 39810.638 72750.524 145566.477 287659.30
## [2,] 1806.2127 4534.8675  7809.286 15155.614 31848.510  58200.419 116453.18
## [3,]  963.5950 1535.2808  3854.637  6637.893 12882.272  27071.234  49470.36
## [4,]  367.6118  663.0979  1064.242  2611.939  4575.824   8831.059  18479.41
##          [,17]      [,18]     [,19]     [,20]
## [1,] 548207.91 1088169.78 2118523.3 4111679.4
## [2,] 230127.44  438566.33  870535.8 1694818.7
## [3,]  98985.20  195608.32  372781.4  739955.4
## [4,]  34003.67   67740.75  133919.5  255699.8
##
## $poptot
##  [1]      10.0000      66.5000      55.1500     142.7150     363.5315
##  [6]     483.1462    1191.2829    2322.0579    4044.5514    8806.0038
## [11]   16494.8533   31672.6830   64216.0834  122057.1304  239669.1889
## [16]  472062.2456  911324.2250 1790085.1785 3495760.0330 6802153.3673
```

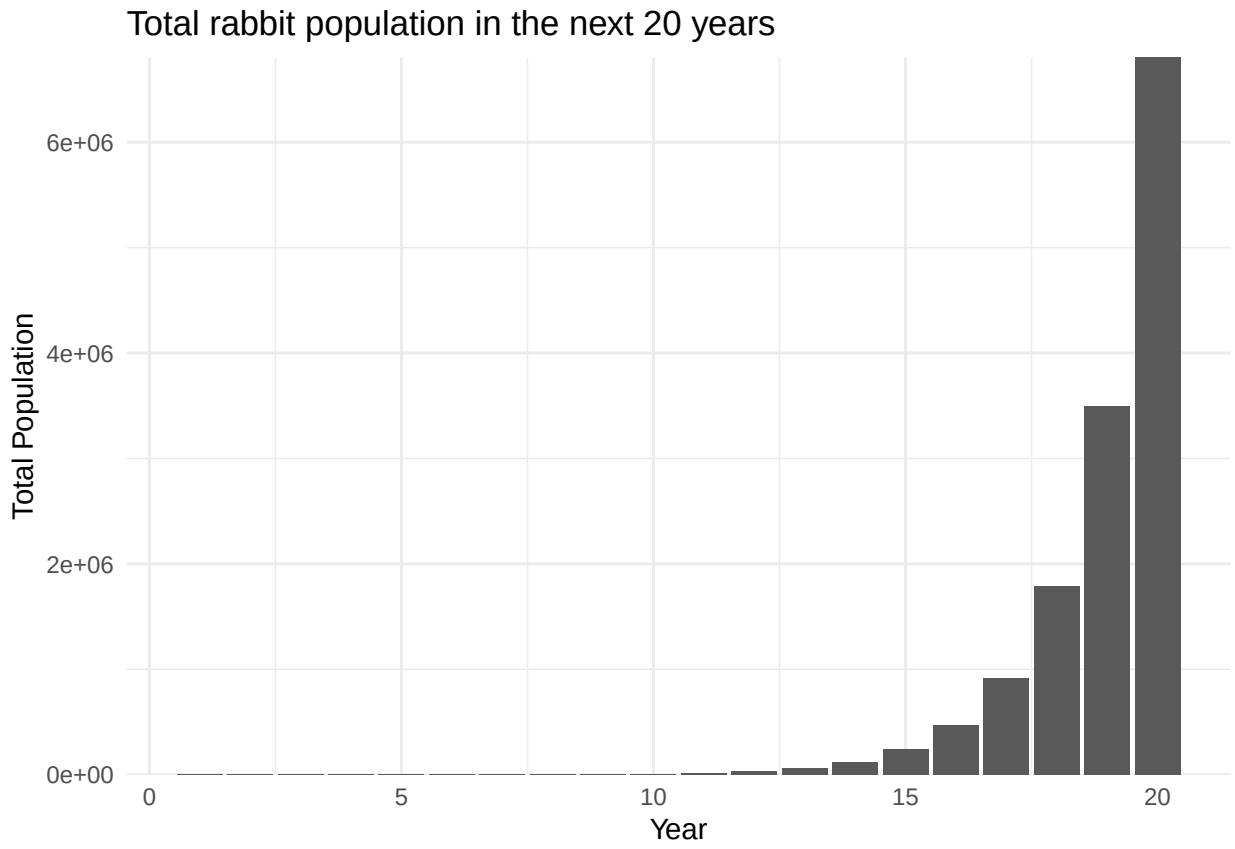
```r
# keep the results for each decade
# graph different components of the output
```

```r
# add year
year = seq(from=1, to=nyears)

################################################
# total population kept in dataframe
```

```
rabbit_tot = cbind.data.frame(year=year, poptot=rabbit_pop$poptot)

# keep result for year 20
pop20 =  format(rabbit_tot$poptot[nyears],big.mark=",",scientific=FALSE)
```

In 20 years, the total rabbit population reach **6,802,153** individuals.

```
# plot total population per decade
ggplot(rabbit_tot, aes(year, poptot))+
  geom_col() +
  labs(title="Total rabbit population in the next 20 years",
       y="Total Population",
       x="Year") +
  theme_minimal() +
  scale_y_continuous(expand = c(0,0))
```

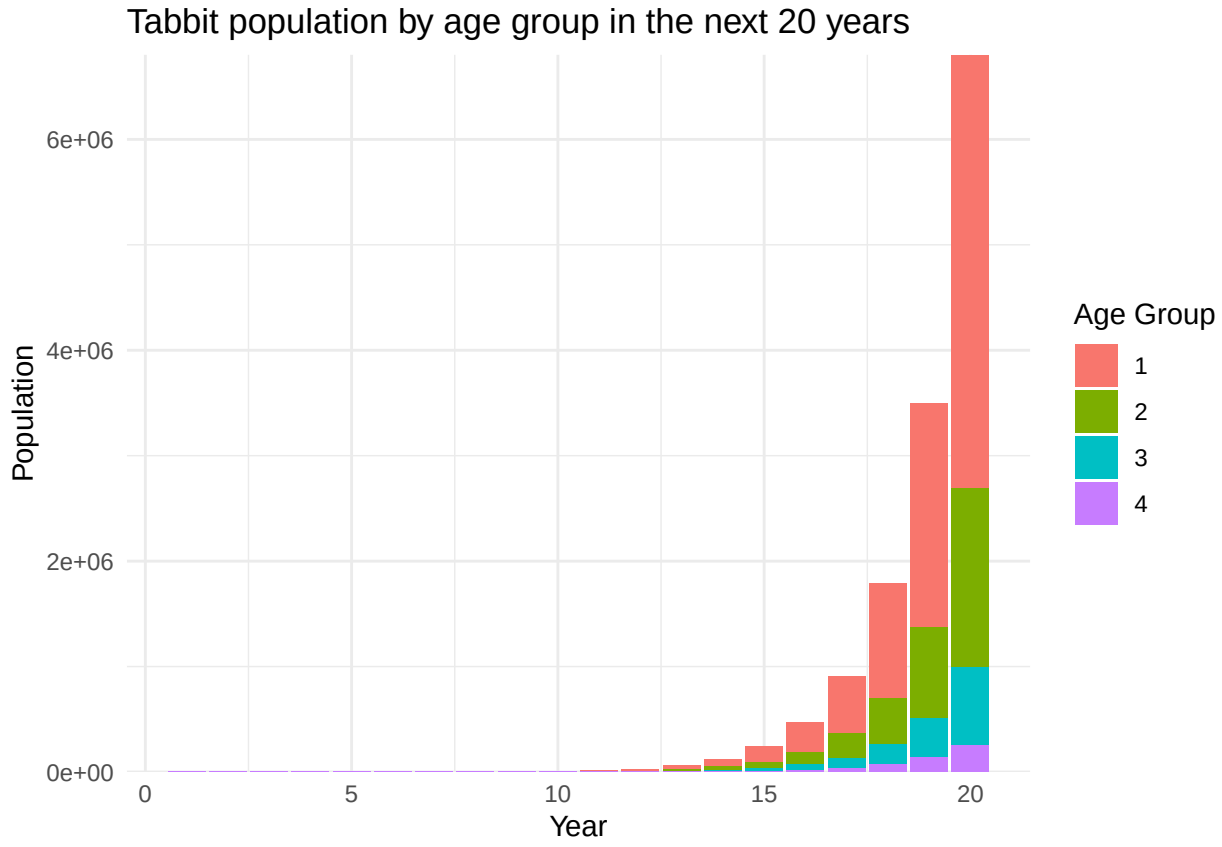



```
##############################################
# population by age group kept in dataframe
rabbit_ages = cbind.data.frame(year=year, t(rabbit_pop$popbyage))

rabbit_agesl = rabbit_ages %>%
  gather(key="agecat", value="pop",-year)

# keep the result for young on year 20
rabbit_young <- rabbit_agesl %>%
  filter(agecat==1)
```

```
young20=  format(rabbit_young$pop[nyears], big.mark=",",scientific=FALSE)
```

Also in year 20, **4,111,679** individuals belong to the first age class (young).

```
# plot information about ages
ggplot(rabbit_agesl, aes(year, pop, fill=agecat))+
  geom_col()+
  labs(title="Tabbit population by age group in the next 20 years",
       y="Population",
       x="Year",
       fill="Age Group") +
  theme_minimal() +
  scale_y_continuous(expand = c(0,0))
```



4. Rabbit Population with Hawks

Hawks generally only eat younger rabbits- thus they reduce the survivability of the young and sub-adults age classes (the first two classes). The estimates are that survivability reduced to between 0.65 and 0.75 for Ages 0-1 and between 0.75 and 0.8 for Ages 1-2. We assume that distributions are uniform.

Our first step is to generate the samples for the Sobel analysis.

```
library(sensitivity)

# survivability - based on mortality rates per thousand per decade
nsample=200

# fertility rates
```

```
F1 = 0 #young
F2 = 2 #sub adult
F3 = 6 #adult
F4 = 1 #aged

# survivability
p12 = 0.8  #original value young
p23 = 0.85 #original value sub adult
p34 = 0.65 #adult
p44 = 0.1  #aged


# we do not vary our ferility parameters
fs = cbind.data.frame(F1=F1,
                      F2=F2,
                      F3=F3,
                      F4=F4)

# create our two samples for Sobel
# first do our survivability, youngs and sub adults have uniform distribution
ps1 = cbind.data.frame(p12 = runif(min=0.65, max=0.75, n=nsample),
                       p23 = runif(min=0.75, max=0.8, n=nsample),
                       p34 = p34,
                       p44 = p44)

ps2 = cbind.data.frame(p12 = runif(min=0.65, max=0.75, n=nsample),
                       p23 = runif(min=0.75, max=0.8, n=nsample),
                       p34 = p34,
                       p44 = p44)

# put servivability and fertility together
allp1 = cbind.data.frame(ps1,fs)
allp2 = cbind.data.frame(ps2,fs)

# get sobel samples
sens_rabbit=soboljansen(model = NULL, allp1, allp2, nboot = 100)

head(sens_rabbit$X)
```

```
##         p12       p23  p34 p44 F1 F2 F3 F4
## 1 0.6915139 0.7560720 0.65 0.1  0  2  6  1
## 2 0.7472212 0.7708327 0.65 0.1  0  2  6  1
## 3 0.7111297 0.7982432 0.65 0.1  0  2  6  1
## 4 0.6526501 0.7876674 0.65 0.1  0  2  6  1
## 5 0.6721526 0.7783478 0.65 0.1  0  2  6  1
## 6 0.6746181 0.7975472 0.65 0.1  0  2  6  1
```

```
nsim=nrow(sens_rabbit$X)
```

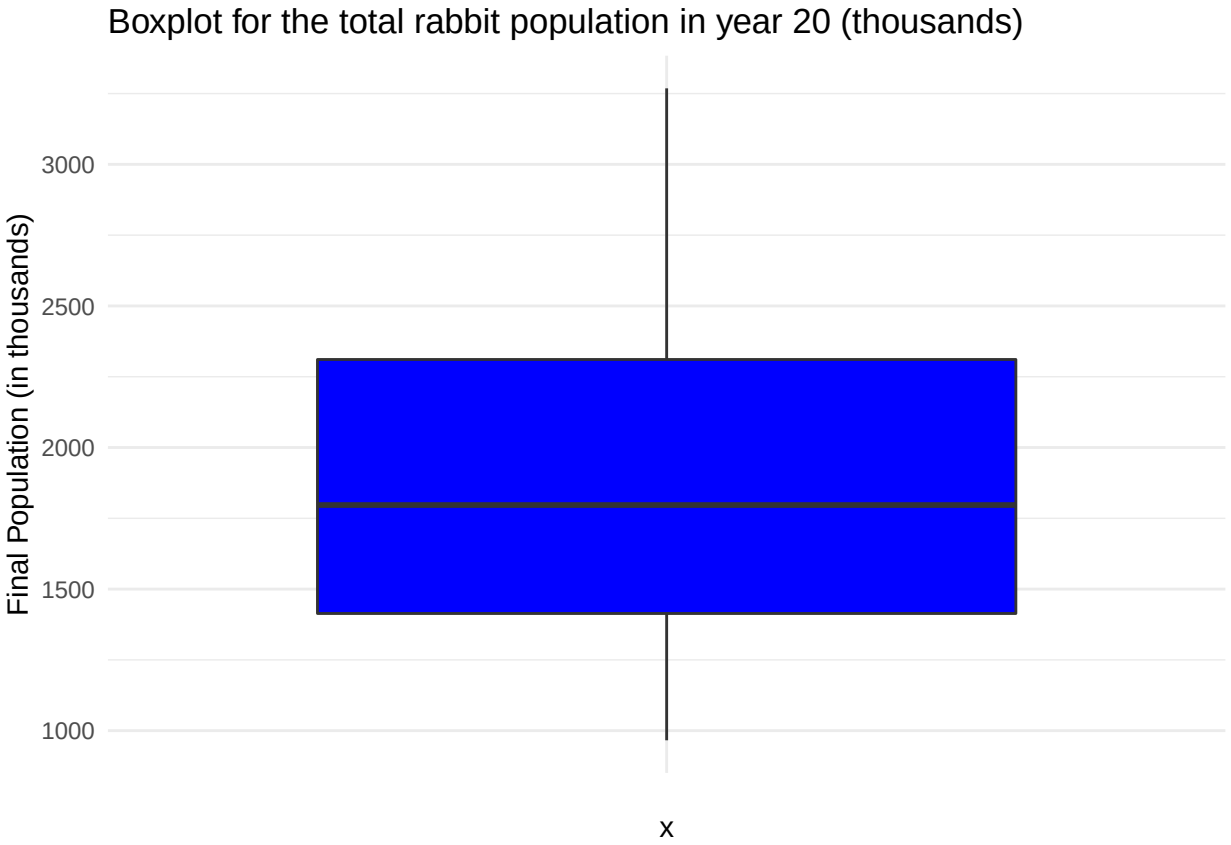
Our second step is to create our wrapper function that contains our evol population function, and the parameter set selected by Sobel.

```
# run model and save what we care about: final population after 2 decades
# this is already output by evolve_pop so we don't need a compute_metric function
```

```
ini = c(0, 0, 10,0) # 10 adult rabbits
nyears = 20 # number of years


# parameter set, with code to extract our metric of interest (final population)
p_wrapper = function(p12, p23, p34, p44, F1, F2, F3, F4, use_func, initialpop, nstep ) {

fertility=c(F1,F2, F3, F4) #fertility data
survivability= c(p12, p23, p34, p44) #survivability data

res = use_func(survivability =survivability,
               fertility = fertility,
               initialpop=initialpop,
               nstep=nstep)
# now return the final population total

return(finalpop=res$poptot[nstep])
}

# use pmap here so we can specify rows of our sensitivity analysis parameter object
res = as.data.frame(sens_rabbit$X) %>% pmap_dbl(p_wrapper,
                                                initialpop=ini,
                                                nstep=nyears,
                                                use_func=evolve_pop)
```

We show the results of the last year in a boxplot. We observe the average population is between 1.5 and 2 million individuals. A **significant reduction** if we compare it to the 6.8 million of the previous result. The result is not surprising if we consider that hawks reduced the upper range of survivability for young (from 0.8 to 0.75) and sub-adult (from 0.85 to 0.8) individuals. In addition, these age groups also admit lower ranges of survivability (0.65 for youngs and 0.75 for sub-adults).

```
# transform our result into a data frame

ggplot(data.frame(finalpop=res), aes(x="", y=finalpop/1000) )+
  geom_boxplot(fill="blue")+
  theme(axis.title.x = element_blank())+
  labs(title = "Boxplot for the total rabbit population in year 20 (thousands)",
    y="Final Population (in thousands)") +
  theme_minimal()
```



Finally, the main effect analysis shows that the rabbit has higher sensitivity to the young survivability (ignoring interaction with other parameters). For the total effect (taking into account the interaction with other variables), the young survivability is also predominant. Logically, the sub-adult survivability is the next relevant one. The rest of the parameters are fixed values.

```
# give our results to sensitivity structure

sens_rabbit=tell(sens_rabbit, res)

# loot at results
sens_rabbit$S
```

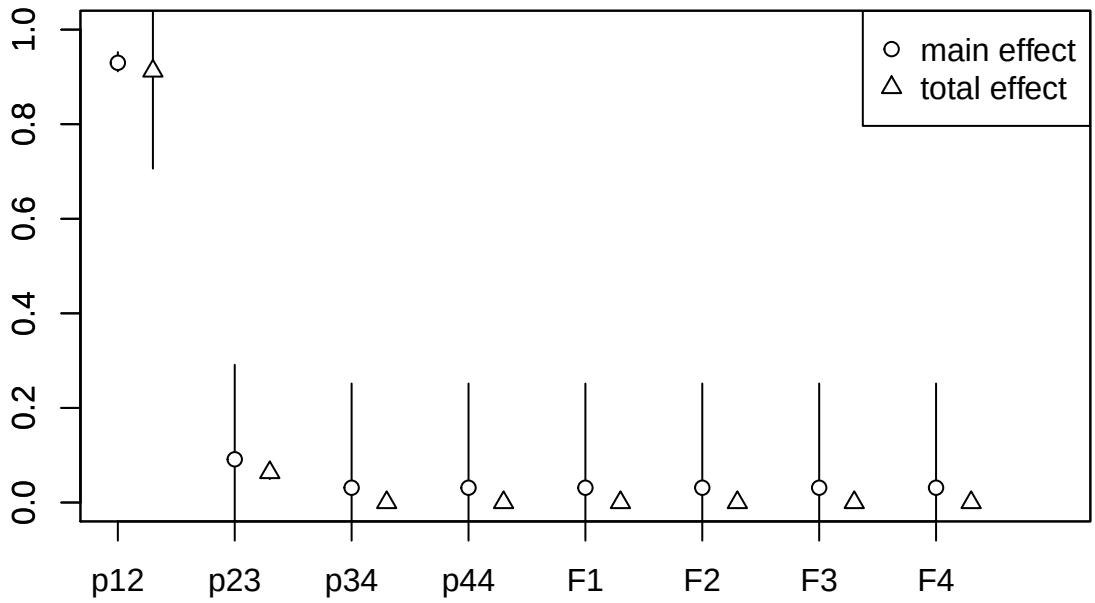
```
##       original          bias  std. error   min. c.i. max. c.i.
## p12 0.92939963 -0.0004428386 0.009419067  0.91178544 0.9523503
## p23 0.08199151 -0.0093421019 0.092158403 -0.07760374 0.2912570
## p34 0.01769804 -0.0135734429 0.101155548 -0.13476785 0.2515906
## p44 0.01769804 -0.0135734429 0.101155548 -0.13476785 0.2515906
## F1  0.01769804 -0.0135734429 0.101155548 -0.13476785 0.2515906
## F2  0.01769804 -0.0135734429 0.101155548 -0.13476785 0.2515906
## F3  0.01769804 -0.0135734429 0.101155548 -0.13476785 0.2515906
## F4  0.01769804 -0.0135734429 0.101155548 -0.13476785 0.2515906
```

```
sens_rabbit$T
```

```
##       original         bias  std. error  min. c.i.  max. c.i.
## p12 0.92290764 0.0104069006 0.096118006 0.70609154 1.09524917
## p23 0.06374465 0.0003523184 0.006612204 0.04954567 0.07847969
## p34 0.00000000 0.0000000000 0.000000000 0.00000000 0.00000000
```

```
## p44 0.00000000 0.0000000000 0.000000000 0.00000000 0.00000000
## F1  0.00000000 0.0000000000 0.000000000 0.00000000 0.00000000
## F2  0.00000000 0.0000000000 0.000000000 0.00000000 0.00000000
## F3  0.00000000 0.0000000000 0.000000000 0.00000000 0.00000000
## F4  0.00000000 0.0000000000 0.000000000 0.00000000 0.00000000
```

```
# graph the most sensitive parameter
plot(sens_rabbit)
```



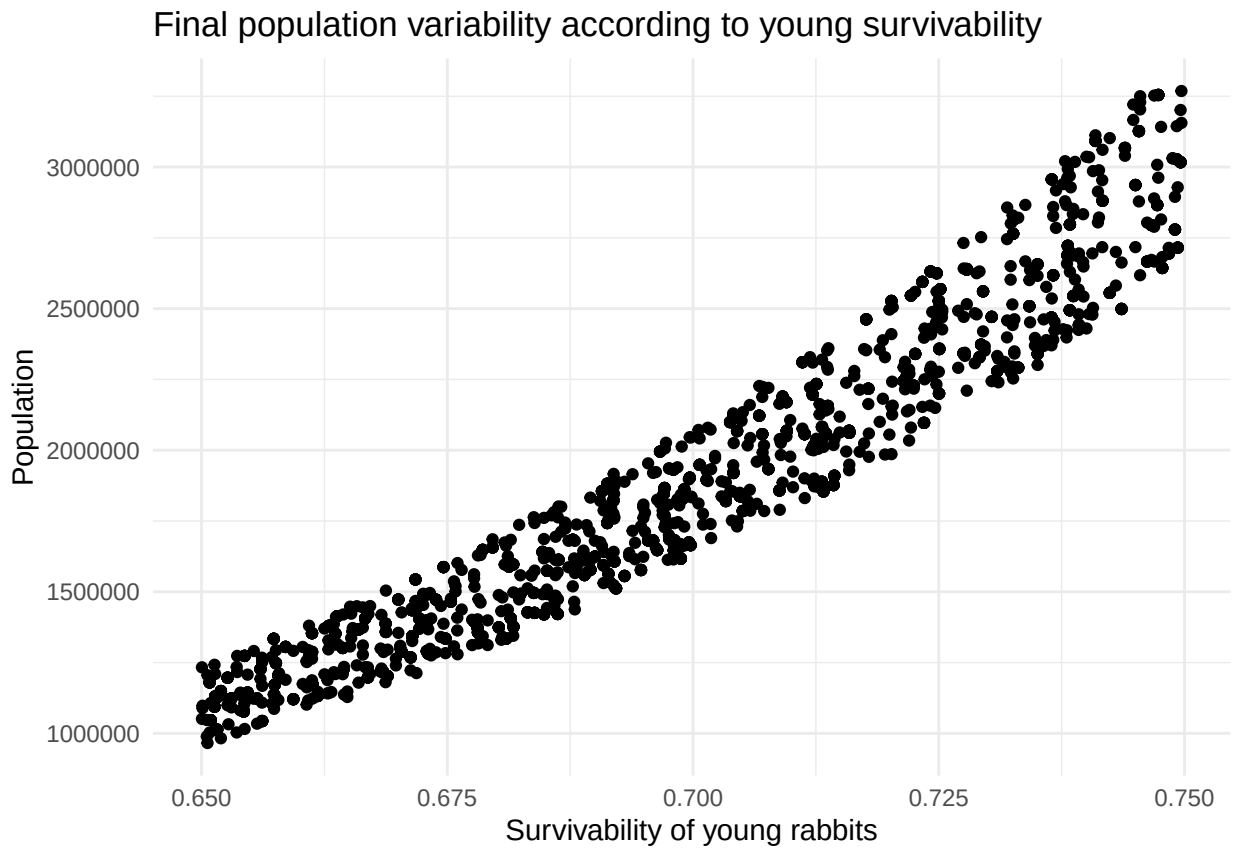
We observe a positive correlation between the rabbit population and tested survivability parameters (youngs and sub-adults). However, we observe the influence of sub-adults is less relevant because its range of potential population values presents lesser sensitivity. For sub-adult survivability between 0.75 and 0.76, the population ranges between 1 and 2.75 million. For sub-adult survivability between 0.79 and 0.8, the population can be between 1.2 and 3.2 million. Consequently, the rabbit population has less sensitivity to sub-adult survivability.

In contrast, young survivability admits "narrower" population ranges along with its values. For example, lower young survivability values have a population range between 1 and 1.5 million; and higher survivability values have populations between 2.5 and 3.2 million.

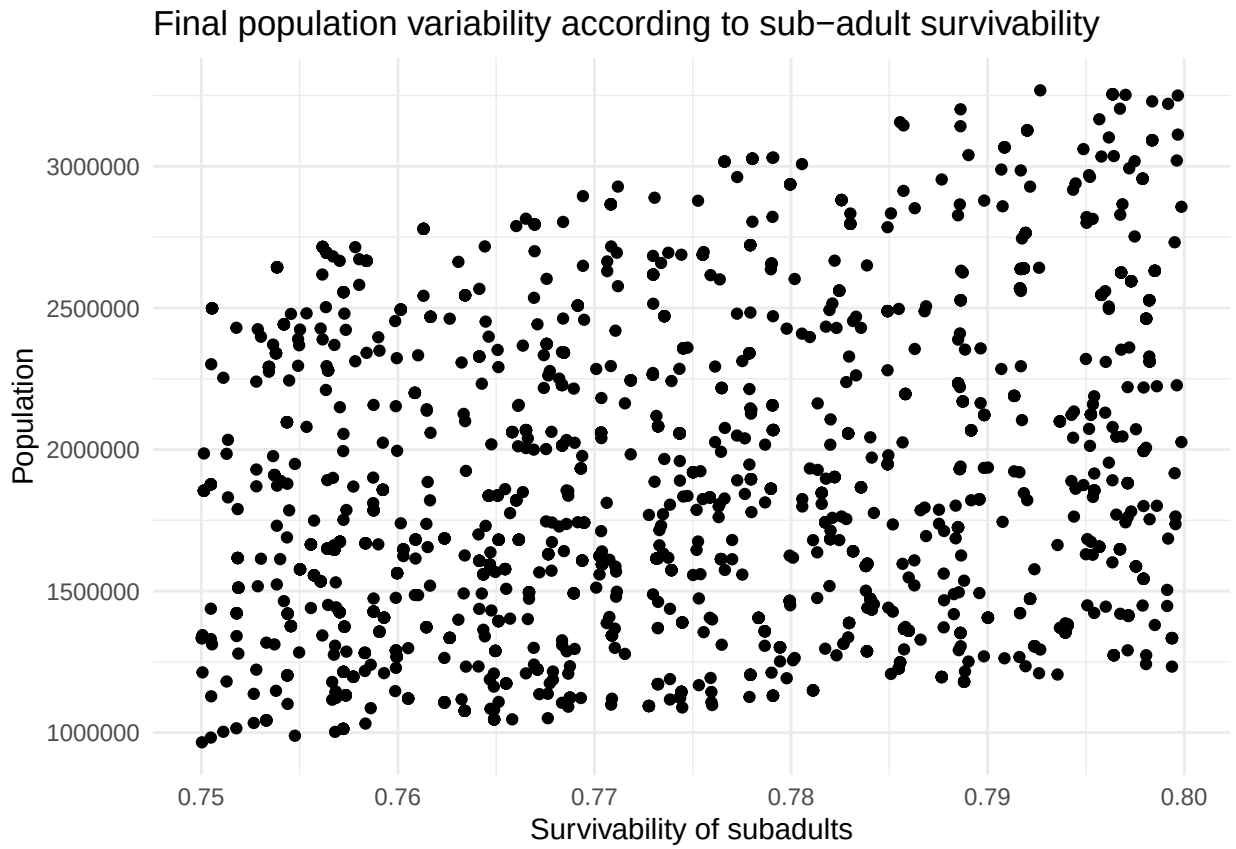
However, both cases present less average and maximum population than our analysis without hawk predation. As we said before, the hawks lower the maximum survivability value for these age groups and incorporate uncertainty.

```
# plot the pop variability to each age class survivability
tmp = cbind.data.frame(sens_rabbit$X, pop12=sens_rabbit$y)

ggplot(tmp, aes(p12, pop12))+
  geom_point()+
```

```
  labs(title="Final population variability according to young survivability",
       x="Survivability of young rabbits",
       y="Population") +
  theme_minimal()
```




```
ggplot(tmp, aes(p23, pop12))+
  geom_point()+
  labs(title="Final population variability according to sub-adult survivability",
       x="Survivability of subadults",
       y="Population") +
  theme_minimal()
```



Appendix: population evolution function

```
#' Population Evolution using Leslie Matrix
#' Evolve a population
#' @param fertility fertility rates
#' @param survivability survivability rates
#' @param initialpop initial population
#' @param nstep number of time steps
#' @return population structure for each time step (OR error message if population cannot be defined)
#' Authors: Naomi Tague and Rachel Torres


evolve_pop = function(fertility, survivability, initialpop, nstep) {


  nclasses = length(fertility)

  # make sure inputs are in the right format
  if ((nclasses!=length(survivability) ))
  { return(sprintf("fertility %d doesn't match survivability %d",
                   nclasses, length(survivability))) }

  if ((nclasses!=length(initialpop) ))
  { return(sprintf("population initialization %d  doesn't match fertility %d ", length(initialpop),
                   length(fertility))) }
```

```
  #initialize the Leslie matrix
  leslie_matrix = matrix(nrow=nclasses, ncol=nclasses)
  leslie_matrix[,] = 0.0
  leslie_matrix[1,] = fertility

  for (i in 1:(nclasses-1)) {
    leslie_matrix[i+1,i] = survivability[i]
  }
  leslie_matrix[nclasses,nclasses] = survivability[nclasses]

  # create an matrix to store population structure
  pop_structure = matrix(nrow=nclasses, ncol=nstep)
  total_pop = rep(0, times=nstep)
  pop_structure[,1] = initialpop
  total_pop[1] = sum(initialpop)

  for (i in 2:nstep) {

    pop_structure[,i] = leslie_matrix %*% pop_structure[,i-1]
    total_pop[i]=sum(pop_structure[,i])

  }

  return(list(popbyage=pop_structure, poptot=total_pop))
}
```