Contents

Introduction

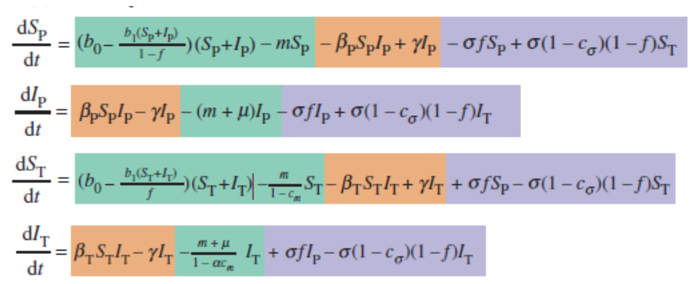
Heavy metal pollution due to urbanization has a serious impact on humans and animals. In addition the exposure to those toxicants can influence negative effects of virus infections on individual hosts, but also altering animal movement. To understand this, a hypothetical flying fox species is infected with a virus and a mathematical model explores the infection dynamics in toxicant-contaminated landscapes.
Flying foxes are often used as bioindicators for heavy metal pollution, because they have a broad range of different habitats. They feed on fruiting plants in human environments, where they face exposure to heavy metals. Another reason why flying fox species are used in this model is because they are reservoir hosts of a virus named henipavirus. This virus can be transmitted to humans and other animals.

Methods

In this section the methods will be discussed. This is divided into two parts: the software and the model. The software part lists the programming language and packages that have been used, with the corresponding versions. The model section lists the differential equations. This is followed by the values and definitions of each parameter of each value is then discussed.

The code used in this project is written used R version 4.1.3. In R the DeSolve package (version 1.32) is imported in order to model the differential equations, which are shown below.

The model is set-up using the following differential equations:



$$\frac{dS_P}{dt} = (b_0 - \frac{b_1(S_P+I_P)}{1-f})(S_P+I_P) - mS_P - \beta_P S_P I_P + \gamma I_P - \sigma f S_P + \sigma(1-c_\sigma)(1-f)S_T$$

$$\frac{dI_P}{dt} = \beta_P S_P I_P - \gamma I_P - (m+\mu)I_P - \sigma f I_P + \sigma(1-c_\sigma)(1-f)I_T$$

$$\frac{dS_T}{dt} = (b_0 - \frac{b_1(S_T+I_T)}{f})(S_T+I_T)| - \frac{m}{1-c_m}S_T - \beta_T S_T I_T + \gamma I_T + \sigma f S_P - \sigma(1-c_\sigma)(1-f)S_T$$

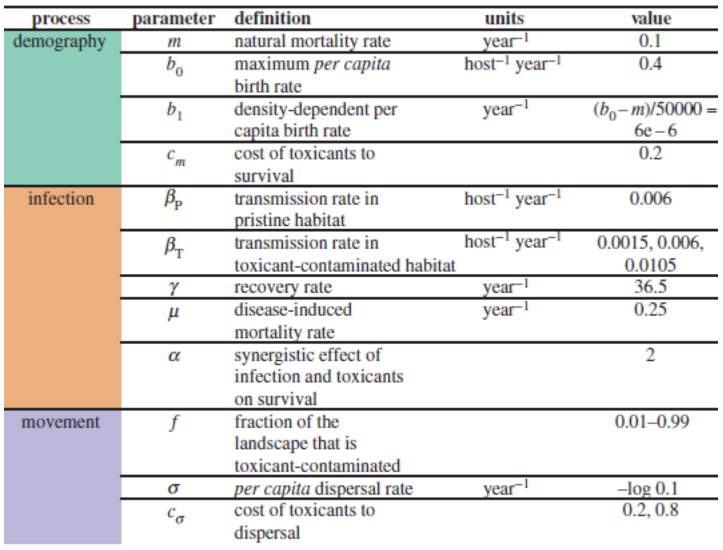
$$\frac{dI_T}{dt} = \beta_T S_T I_T - \gamma I_T - \frac{m+\mu}{1-\alpha c_m} I_T + \sigma f I_P - \sigma(1-c_\sigma)(1-f)I_T$$

Figure 1: caption.....

In figure 1 four differential equations are shown. Firstly, $\frac{dS_P}{dt}$ calculates the population size of the susceptible (S) flying foxes which are located in a pristine (P) area. Secondly, $\frac{dI_P}{dt}$ calculates the mount of infected (I) flying foxes in a pristine (P) area. Thirdly, $\frac{dS_T}{dt}$ calculates the amount of flying foxes which are susceptible (S) and in a toxic (T) area. Lastly, $\frac{dI_T}{dt}$ calculates the population size of the flying foxes which are infected (I) and located in a toxic area (T)

Additionally, in figure 1 parts of the differential equations are coloured. Each colour represents a different process which may affect the population size with that infection status and habitat. Green is used to indicate demography, orange to indicate infection and purple to indicate movement.

Each equation is broken down into its parameters and listed below:



process	parameter	definition	units	value
demography	m	natural mortality rate	year^{-1}	0.1
	b_0	maximum *per capita* birth rate	host^{-1} year^{-1}	0.4
	b_1	density-dependent per capita birth rate	year^{-1}	$(b_0 - m)/50000 = 6e-6$
	c_m	cost of toxicants to survival		0.2
infection	β_P	transmission rate in pristine habitat	host^{-1} year^{-1}	0.006
	β_T	transmission rate in toxicant-contaminated habitat	host^{-1} year^{-1}	0.0015, 0.006, 0.0105
	γ	recovery rate	year^{-1}	36.5
	μ	disease-induced mortality rate	year^{-1}	0.25
	α	synergistic effect of infection and toxicants on survival		2
movement	f	fraction of the landscape that is toxicant-contaminated		0.01–0.99
	σ	*per capita* dispersal rate	year^{-1}	–log 0.1
	c_σ	cost of toxicants to dispersal		0.2, 0.8

Figure 2: caption.....

As shown in figure 2; β_T, f and population size show variable values. This is based on the different scenarios that are being modulated. In both scenarios (to be discussed shortly) f is being changed, ranging from 0.01 to 0.99 (in steps of 0.01). This is the amount of landscape that is contaminated by toxicants. For each f, the model is run for a timespan of 50 years and the last values are used. This timespan is chosen because all of the differential equations are then at equilibrium. Furthermore, for each scenario the population size, infection prevalence and spillover risk are being calculated using the following formulas:

$$\text{Population size} = S_P + I_P + S_T + I_T$$

$$\text{Infection prevelance} = \frac{I_T}{\text{population size}}$$

$$\text{Spillover risk} = \frac{I_T}{f}$$

Coming back to the scenarios, in the first scenario β_T is being varied. This is done to simulate the effect of different transmission rates in a toxic contaminated habitat with respect to the transmission rate in a pristine habitat. Three different β_T values were used to obtain the following scenarios: *ander woord voor scenarios kiezen*

- $\beta_T < \beta_P$
- $\beta_T = \beta_P$
- $\beta_T > \beta_P$

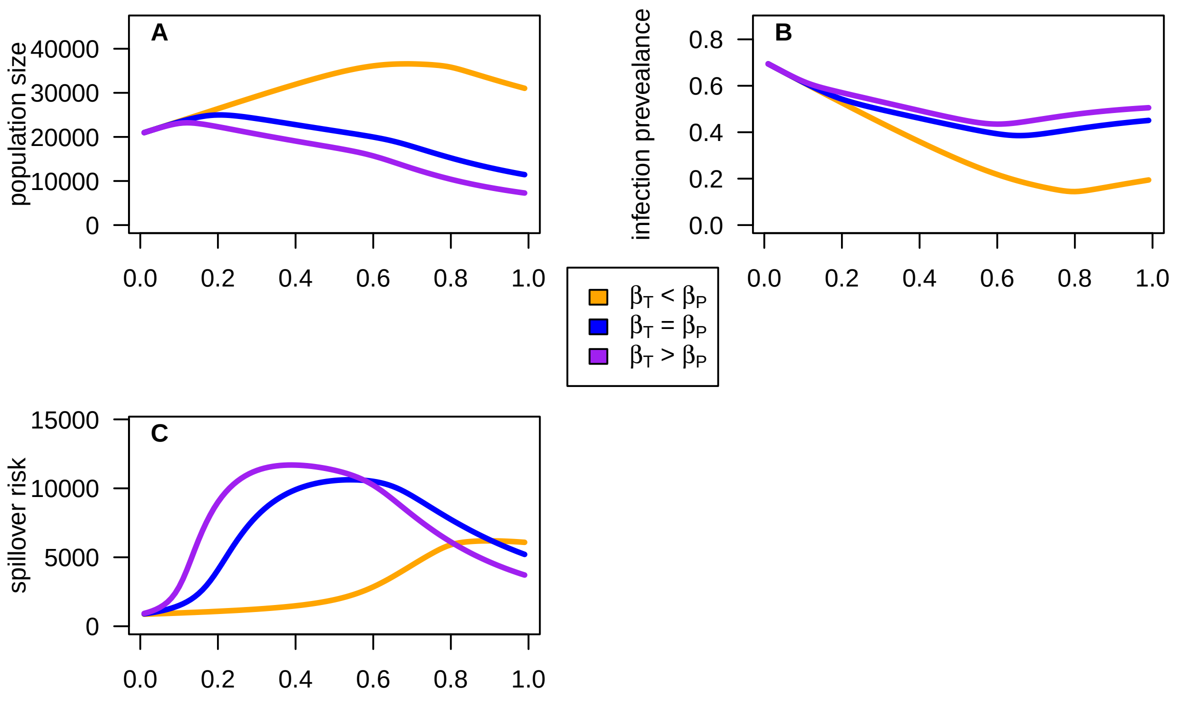
In the second scenario, the population size is being varied. This is done to see what the effect is of the population size when also the fraction of the toxic-contaminated habitat is changed. The following population sizes were used:

- population size = ...

Results

As mentioned earlier, two scenarios have been modulated. This section is divided according to these scenarios.

Scenario 1: beta_t



In figure # the population size, infection prevalence and spillover risk are shown with varying β_t values.

In figure #A $\beta_t = \beta_p$ and $\beta_t > \beta_p$ show a similar trend. When the fraction of contaminant habitat increases (f), the population size first increases ($f < 0.15$) a little and then declines slowly ($f > 0.15$). However, the population size of $\beta_t = \beta_p$ has a higher increase until it declines resulting in a higher population size at different f's. Interestingly, when looking at $\beta_t < \beta_p$ an increase can be seen until an f of 0.7 is reached, which is way higher when compared to the other scenarios where it increased until 0.15. After an f of 0.7 it declines.

In figure #B $\beta_t = \beta_p$ and $\beta_t > \beta_p$ are again very similar. A slow decline of infection prevalence can be seen until an %f% of approximately 0.65 is reached, after which it increases. Furthermore, $\beta_t < \beta_p$ shows a steeper and longer decrease as f increases. It then increases in the same way as$\beta_t = \beta_p$ and $\beta_t > \beta_p$.

In figure #C $\beta_t = \beta_p$ and $\beta_t > \beta_p$ repeatedly show similar trends. They both have a sharp S-shaped increase with respect to an increasing value of f. Hereafter it declines. The same S-shaped increase can be seen when looking at $\beta_t < \beta_p$. However, this increases is less steep and the optimum is reached at a higher f. Furthermore, when the optimum is reached it does not decline at all.

Altogether it seems like a similar trend in all graphs can be seen. $\beta_t = \beta_p$ and $\beta_t > \beta_p$ show very similar curves, where $\beta_t = \beta_p$ is delayed or more spread out. When looking at $\beta_t < \beta_p$ the same phenomenon can be seen, but more extreme. The increases and decreases are less steep and take longer with respect to an increasing f.

Scenario 2: population size

```
## FOR LOOPJES OBTAINING DATA

# for (population_size in c(1000, 50000, 100000)){
#   df <- run_model(beta_t = 0.0015, population = population_size)
#   if (population_size == 1000){
#     N_all <- data.frame('25000' = df$N)
#     p_all <- data.frame('25000' = df$p)
#     rho_all <- data.frame('25000' = df$rho)
#   }
#   if (population_size == 50000){
#     N_all <- data.frame(N_all, '50000' = df$N)
#     p_all <- data.frame(p_all, '50000' = df$p)
#     rho_all <- data.frame(rho_all, '50000' = df$rho)
#   }
#   if (population_size == 100000){
#     N_all <- data.frame(N_all, '75000' = df$N)
#     p_all <- data.frame(p_all, '75000' = df$p)
#     rho_all <- data.frame(rho_all, '75000' = df$rho)
#   }
# }
```

```
# population_data <- list(N_all, p_all, rho_all)
# population_labels <- c('population size', 'infection prevealance', 'spillover risk')
# population_cols <- c('orange', 'blue', 'purple')
#
# plot_scenario(datasets = population_data, y_labels = population_labels, line_cols = population_cols)
```

```
#
# run_model_sensitivity <- function(population = 50000, infected = 100, m = 0.1, b0 = 0.4,
#                         c_m = 0.2, beta_p = 0.006, beta_t = 0.006, gamma = 36.5,
#                         mu = 0.25, alpha = 2, sigma = -log(0.1), c_sigma = 0.2, f = 0.1){
#
#   # Create a data frame to store needed data
#   df <- data.frame(matrix(nrow = 0, ncol = 2))
#   colnames(df) <- c('mu', 'N')
#
#   # Calculate b1 based on the function parameters
#   b1 <- (b0 - m) / population
#
#   # Calculate infection status per habitat
#   Sp <- (population - infected) * (1 - f)
#   Ip <- infected * (1 - f)
#   St <- (population - infected) * f
#   It <- infected * f
#
#   # Creating a vector with the parameter values
#   parameters <- c(m = m, b0 = b0, b1 = b1, c_m = c_m, beta_p = beta_p,
#                   beta_t = beta_t, gamma = gamma, mu = mu, alpha = alpha,
#                    f = f, sigma = sigma, c_sigma = c_sigma)
#
#   # Creating a vector with the initial values
#   state<- c(Sp = Sp, Ip = Ip, St = St, It = It)
#
#   # Time frame of 50 years
#   times <- seq(0, 50, 0.05)
#   print(c_m)
#   out <- tail(ode(y = state, times = times, func = wildlife_urbanization_model, parms = parameters),
#
#   df[nrow(df) + 1,] <- c(c_m, (out[3] + out[5])/sum(out[2:5]))
#
#   return(df)
# }
```

```
# df_mu <- data.frame(matrix(nrow=0, ncol=2))
# for (mu in seq(0.05, 0.95, 0.02)){
#   df_mu[nrow(df_mu) + 1,] <- run_model_sensitivity(c_m = mu)
# }
# plot(df_mu$X2 ~ df_mu$X1, type = 'l', ylim = c(0, 1))
```

Discussion

References

Apendix

```r
## ODE FUNCTION

# Load deSolve package
library(deSolve)

# Function with the models differential equations
wildlife_urbanization_model <- function(t, state, parameters) {
  with(as.list(c(state, parameters)),{

    dSp <- (b0 - (b1 * (Sp + Ip)) /(1-f) ) * (Sp + Ip) - m * Sp -  # demography
      beta_p * Sp * Ip + gamma * Ip -  # infection
      sigma * f * Sp + sigma * (1 - c_sigma) * (1 - f) * St  # movement


    dIp <-  beta_p * Sp * Ip - gamma * Ip -  # infection
      (m + mu) * Ip -  # demography
      sigma * f * Ip + sigma * (1 - c_sigma) * (1 - f) * It  # movement


    dSt <- (b0 - (b1 * (St + It) / f)) * (St + It) - (m / (1 - c_m)) * St -  # demography
      beta_t * St * It + gamma * It +  # infection
      sigma * f * Sp - sigma * (1 - c_sigma) * (1 - f) * St  # movement


    dIt <- beta_t * St * It - gamma * It -  # infection
      ((m + mu) / (1 - alpha * c_m)) * It +  # demography
      sigma * f * Ip - sigma * (1 - c_sigma) * (1 - f) * It  # movement


    list(c(dSp, dIp, dSt, dIt))
  })
}
```

```r
## MODEL FUNCTION

run_model <- function(population = 50000, infected = 100, m = 0.1, b0 = 0.4,
                      c_m = 0.2, beta_p = 0.006, beta_t = 0.006, gamma = 36.5,
                      mu = 0.25, alpha = 2, sigma = -log(0.1), c_sigma = 0.2){

  # Create a data frame to store needed data
  df <- data.frame(matrix(nrow = 0, ncol = 8))
  colnames(df) <- c('f', 'Sp', 'Ip', 'St', 'It', 'N', 'p', 'rho')

  for (f in seq(0.01, 0.99, 0.01)){
    # Calculate b1 based on the function parameters
    b1 <- (b0 - m) / population

    # Calculate infection status per habitat
    Sp <- (population - infected) * (1 - f)
    Ip <- infected * (1 - f)
    St <- (population - infected) * f
    It <- infected * f

    # Creating a vector with the parameter values
    parameters <- c(m = m, b0 = b0, b1 = b1, c_m = c_m, beta_p = beta_p,
                    beta_t = beta_t, gamma = gamma, mu = mu, alpha = alpha,
                    f = f, sigma = sigma, c_sigma = c_sigma)

    # Creating a vector with the initial values
    state<- c(Sp = Sp, Ip = Ip, St = St, It = It)

    # Time frame of 50 years
    times <- seq(0, 50, 0.05)

    out <- tail(ode(y = state, times = times, func = wildlife_urbanization_model, parms = parameters), 

    df[nrow(df) + 1,] <- c(f, out[2:5], sum(out[2:5]), (out[3] + out[5])/sum(out[2:5]), out[5]/f)
    }

  return(df)
}
```

```
## PLOT FUNCTION
plot_scenario <- function(datasets, y_labels, line_cols, legends){
  plot_labels = c("A", "B", "C")
  for (item in 1:length(datasets)){
    data <- datasets[[item]]
    for (scenario in 1:3){
      if (scenario == 1){
        plot(data[,scenario] ~ df$f, col = line_cols[scenario],
             type = 'l', lwd = 3, ylim = c(0, max(data) * 1.25),
             xlab = 'f', las = 1, ylab = "")
        title(ylab = y_labels[item], line = 3.3)
        text(x=0.05, y=max(data) * 1.2, labels = plot_labels[item], font = 2)
      }
      else{
        lines(data[,scenario] ~ df$f, col = line_cols[scenario], lwd = 3)
      }
    }
  }
}
```

```
## FOR LOOPJES OBTAINING DATA

for (beta_t in c(0.0015, 0.006, 0.0105)){
  df <- run_model(beta_t = beta_t)
  if (beta_t == 0.0015){
    N_all <- data.frame('0.0015' = df$N)
    p_all <- data.frame('0.0015' = df$p)
    rho_all <- data.frame('0.0015' = df$rho)
  }
  if (beta_t == 0.006){
    N_all <- data.frame(N_all, '0.006' = df$N)
    p_all <- data.frame(p_all, '0.006' = df$p)
    rho_all <- data.frame(rho_all, '0.006' = df$rho)
  }
  if (beta_t == 0.0105){
    N_all <- data.frame(N_all, '0.0105' = df$N)
    p_all <- data.frame(p_all, '0.0105' = df$p)
    rho_all <- data.frame(rho_all, '0.0105' = df$rho)
  }
}
```

```
beta_t_data <- list(N_all, p_all, rho_all)
beta_t_labels <- c('population size', 'infection prevealance', 'spillover risk')
beta_t_cols <- c('orange', 'blue', 'purple')
beta_t_legends <- c(expression(paste(beta[T], " < ", beta[P])),expression(paste(beta[T], " = ", beta[P]
                    expression(paste(beta[T], " > ", beta[P])))
par(mfrow=c(2,2), mar=c(3.1,5.1,3.1,2.1))
plot_scenario(datasets = beta_t_data, y_labels = beta_t_labels, line_cols = beta_t_cols,
              legends = beta_t_legends)
legend(1.1, 26000, legend = beta_t_legends, fill = beta_t_cols, xpd=NA)
```