```
### KHV MODEL ===== ===== ===== ===== =====

# last updated: January 19, 2023

library(cowplot)
library(deSolve)
library(ggplot2)
library(graphics)
library(reshape2)
library(tidyverse)

rm(list=ls()) # clear environment


### INITIALIZE ===== ===== ===== ===== =====

## Time =====
yrs = 12 # max = 12
tvals = seq(0, yrs*365, by=1)

## Initial Conditions =====
SJ0=0
AJ0=0
IJ0=0
DJ0=0
WJ0=0
SM0=0
AM0=0
IM0=0
DM0=0
WM0=0
NJ_tot0 = 0
NM_tot0 = 0
AJ_tot0 = 0
IJ_tot0 = 0
DJ_tot0 = 0
WJ_tot0 = 0
AM_tot0 = 0
IM_tot0 = 0
DM_tot0 = 0
WM_tot0 = 0
h_tot0 = 0

inits = c(SJ=SJ0, AJ=AJ0, IJ=IJ0, DJ=DJ0, WJ=WJ0,
          SM=SM0, AM=AM0, IM=IM0, DM=DM0, WM=WM0,
          NJ_tot=NJ_tot0, NM_tot=NM_tot0,
          AJ_tot=AJ_tot0, IJ_tot=IJ_tot0, DJ_tot=DJ_tot0, WJ_tot=WJ_tot0,
          AM_tot=AM_tot0, IM_tot=IM_tot0, DM_tot=DM_tot0, WM_tot=WM_tot0,
          h_tot = h_tot0)
```

```
### PARAMETERS: TIME & TEMPERATURE INDEPENDENT ===== ===== ===== ===== =====

# natural death of fish
mu = 0.00014
# transmission from A
betaA = 0.00000088
# transmission from I & D
betaID = 0.0000088
# transmission from W
betaW = 0.05
# diseased induced mortality
xi = 0.2
# shedding from A
rhoA = 0.00001
# shedding from I & D
rhoID = 0.00002
# contamination from external sources
omega = 0.00001
# decay from sanitation
eta = 1/3
# % of market fish that breed
p = 0.05
# survival to juvenile
gb = 0.0006
# survival to market
gJ = 0.00009
# harvesting of market fish
h = 0 #0.01

pars = c(mu=mu, betaA=betaA, betaID=betaID, betaW=betaW, xi=xi,
         rhoA=rhoA, rhoID=rhoID, omega=omega, eta=eta,
         p=p, gb=gb, gJ=gJ, h=h)


### PARAMETERS: TIME & TEMPERATURE DEPENDENT ===== ===== ===== ===== =====

# temperature
tempfunc = function(t){
  temp = 70 + 30*cos(2*pi*t/365)}
# normal distribution
NCfunc = function(temp){
  NC = dnorm(temp,68.9,2.7)}
# symptoms develop
sigmafunc = function(NC){
  sigma = ((1/5.9)/max(NC))*NC}
# symptoms worsen
alphafunc = function(NC){
  alpha = ((1/5.3)/max(NC))*NC}
# become asymptomatic
gammafunc = function(NC){
  gamma = ((1/21)/max(NC))*(-NC + max(NC))}
# import of S & A
mSAfunc = function(t){
  mSA = 100*exp(-0.015*t)}
```

```
# import of I & D
mIDfunc = function(t){
  mID = 10*exp(-0.015*t)}
# no. of eggs from S & A
bSAfunc = function(t){
    ifelse(t>275+0*365 & t<275+0*365+60 | t>275+1*365 & t<275+1*365+60 |
           t>275+2*365 & t<275+2*365+60 | t>275+3*365 & t<275+3*365+60 |
           t>275+4*365 & t<275+4*365+60 | t>275+5*365 & t<275+5*365+60 |
           t>275+6*365 & t<275+6*365+60 | t>275+7*365 & t<275+7*365+60 |
           t>275+8*365 & t<275+8*365+60 | t>275+9*365 & t<275+9*365+60 |
           t>275+10*365 & t<275+10*365+60 | t>275+11*365 & t<275+11*365+60,
           500000, 0)}
# no. of eggs from I
bIfunc = function(bSA){
  bI = 0.5*bSA}


### MODEL ===== ===== ===== ===== =====

Model = function(t, inits, pars){

  # time & temperature dependent parameters
  temp = tempfunc(t)
  NC = NCfunc(temp)
  sigma = sigmafunc(NC)
  alpha = alphafunc(NC)
  gamma = gammafunc(NC)
  mSA = mSAfunc(t)
  mID = mIDfunc(t)
  bSA = bSAfunc(t)
  bI = bIfunc(bSA)

  with(as.list(c(inits, pars)),{

    ## Juvenile Equations =====
    dSJ = -mu*SJ - betaA*AJ*SJ - betaID*(IJ + DJ)*SJ - betaW*WJ*SJ - gJ*SJ +
          p*gb*(bSA*(SM + AM) + bI*IM)
    dAJ = -mu*AJ + betaA*AJ*SJ + betaID*(IJ + DJ)*SJ + betaW*WJ*SJ -
          sigma*AJ + gamma*IJ + gamma*DJ - gJ*AJ
    dIJ = -mu*IJ + sigma*AJ - gamma*IJ - alpha*IJ - gJ*IJ
    dDJ = -mu*DJ + alpha*IJ - gamma*DJ - xi*DJ - gJ*DJ
    dWJ = rhoA*AJ + rhoID*(IJ + DJ) + omega - eta*WJ

    ## Market Equations =====
    dSM = -mu*SM - betaA*AM*SM - betaID*(IM + DM)*SM - betaW*WM*SM + gJ*SJ +
          mSA - h*SM
    dAM = -mu*AM + betaA*AM*SM + betaID*(IM + DM)*SM + betaW*WM*SM -
          sigma*AM + gamma*IM + gamma*DM + gJ*AJ + mSA - h*AM
    dIM = -mu*IM + sigma*AM - gamma*IM - alpha*IM + gJ*IJ + mID
    dDM = -mu*DM + alpha*IM - gamma*DM - xi*DM + gJ*DJ + mID
    dWM = rhoA*AM + rhoID*(IM + DM) + omega - eta*WM
```

```
    ## Totals =====
    dNJ_tot = p*gb*(bSA*(SM + AM) + bI*IM)
    dNM_tot = 2*mSA + 2*mID + gJ*(SJ + AJ + IJ + DJ)
    dAJ_tot = betaA*AJ*SJ
    dIJ_tot = betaID*IJ*SJ
    dDJ_tot = betaID*DJ*SJ
    dWJ_tot = betaW*WJ*SJ
    dAM_tot = betaA*AM*SM
    dIM_tot = betaID*IM*SM
    dDM_tot = betaID*DM*SM
    dWM_tot = betaW*WM*SM
    dh_tot = h*(SM + AM + IM + DM)

    list(c(dSJ, dAJ, dIJ, dDJ, dWJ,
           dSM, dAM, dIM, dDM, dWM,
           dNJ_tot, dNM_tot,
           dAJ_tot, dIJ_tot, dDJ_tot, dWJ_tot,
           dAM_tot, dIM_tot, dDM_tot, dWM_tot,
           dh_tot))
  })
}

# solve model
sol = as.data.frame(ode(y=inits, times=tvals, func=Model, parms=pars))


### PLOTS ===== ===== ===== ===== =====

## Parameters: Time & Temperature Dependent =====

temp_plot = tempfunc(tvals)
temp_df = data.frame(tvals, temp_plot)
temp_plot2 = ggplot(temp_df, aes(x=tvals, y=temp_plot)) + geom_line() +
  ggtitle("(a) water temperature") + xlab("days") + ylab("degrees F")

NC_plot = NCfunc(temp_plot)
NC_df = data.frame(temp_plot, NC_plot)
NC_plot2 = ggplot(NC_df, aes(x=temp_plot, y=NC_plot)) + geom_line() +
  ggtitle("(b) normal curve") + xlab("temperature") + ylab("")

sigma_plot = sigmafunc(NC_plot)
sigma_df = data.frame(tvals, sigma_plot)
sigma_plot2 = ggplot(sigma_df, aes(x=tvals, y=sigma_plot)) + geom_line() +
  ggtitle("(c) sigma") + xlab("days") + ylab("per day")

alpha_plot = alphafunc(NC_plot)
alpha_df = data.frame(tvals, alpha_plot)
alpha_plot2 = ggplot(alpha_df, aes(x=tvals, y=alpha_plot)) + geom_line() +
  ggtitle("(d) alpha") + xlab("days") + ylab("per day")

gamma_plot = gammafunc(NC_plot)
gamma_df = data.frame(tvals, gamma_plot)
gamma_plot2 = ggplot(gamma_df, aes(x=tvals, y=gamma_plot)) + geom_line() +
  ggtitle("(e) gamma") + xlab("days") + ylab("per day")
```

```
mSA_plot = mSAfunc(tvals)
mSA_df = data.frame(tvals, mSA_plot)
mSA_plot2 = ggplot(mSA_df, aes(x=tvals, y=mSA_plot)) + geom_line() +
  ggtitle("(f) m_SA") + xlab("days") + ylab("ind. per day")

mID_plot = mIDfunc(tvals)
mID_df = data.frame(tvals, mID_plot)
mID_plot2 = ggplot(mID_df, aes(x=tvals, y=mID_plot)) + geom_line() +
  ggtitle("(g) m_ID") + xlab("days") + ylab("ind. per day")

bSA_plot = bSAfunc(tvals)
bSA_df = data.frame(tvals, bSA_plot)
bSA_plot2 = ggplot(bSA_df, aes(x=tvals, y=bSA_plot)) + geom_line() +
  ggtitle("(h) b_SA") + xlab("days") + ylab("eggs")

bI_plot = bIfunc(bSA_plot)
bI_df = data.frame(tvals, bI_plot)
bI_plot2 = ggplot(bI_df, aes(x=tvals, y=bI_plot)) + geom_line() +
  ggtitle("(i) b_I") + xlab("days") + ylab("eggs")

plot_grid(temp_plot2, NC_plot2, sigma_plot2, alpha_plot2, gamma_plot2,
          mSA_plot2, mID_plot2, bSA_plot2, bI_plot2, ncol = 3, nrow = 3)

## Disease Classes =====

J_plot = sol %>% gather(key, individuals, SJ, AJ, IJ, DJ) %>%
ggplot(aes(x=time, y=individuals, color=key)) + geom_line() +
  ggtitle("(a) Juvenile Pool") + xlab("days") + ylab("individuals") +
  theme(legend.position=c(1,1),legend.justification=c("right","top"),
        legend.margin=margin(5,5,5,5))

M_plot = sol %>% gather(key, individuals, SM, AM, IM, DM) %>%
ggplot(aes(x=time, y=individuals, color=key)) + geom_line() +
  ggtitle("(a) Market Pool") + xlab("days") + ylab("individuals") +
  theme(legend.position=c(1,1),legend.justification=c("right","top"),
        legend.margin=margin(5,5,5,5))

WJ_plot = sol %>% gather(key, individuals, WJ) %>%
ggplot(aes(x=time, y=individuals, color=key)) + geom_line() +
  ggtitle("(b) Water Contamination") + xlab("days") + ylab("percentage") +
  theme(legend.position=c(1,1),legend.justification=c("right","top"),
        legend.margin=margin(5,5,5,5))

WM_plot = sol %>% gather(key, individuals, WM) %>%
  ggplot(aes(x=time, y=individuals, color=key)) + geom_line() +
  ggtitle("(b) Water Contamination") + xlab("days") + ylab("percentage") +
  theme(legend.position=c(1,1),legend.justification=c("right","top"),
        legend.margin=margin(5,5,5,5))

plot_grid(J_plot, WJ_plot, ncol = 1, nrow = 2)
plot_grid(M_plot, WM_plot, ncol = 1, nrow = 2)
```

```
## Zoomed vs. Temperature =====

J_plot2 = sol %>% gather(key, individuals, AJ, IJ, DJ) %>%
  ggplot(aes(x=time, y=individuals, color=key)) + geom_line() +
  ggtitle("(a) Juvenile Pool") + xlab("days") + ylab("individuals") +
  theme(legend.position=c(1,1),legend.justification=c("right","top"),
        legend.margin=margin(5,5,5,5))

M_plot2 = sol %>% gather(key, individuals, AM, IM, DM) %>%
  ggplot(aes(x=time, y=individuals, color=key)) + geom_line() +
  ggtitle("(a) Market Pool") + xlab("days") + ylab("individuals") +
  theme(legend.position=c(1,1),legend.justification=c("right","top"),
        legend.margin=margin(5,5,5,5))

temp_plot = tempfunc(tvals)
temp_df = data.frame(tvals, temp_plot)
temp_plot3 = ggplot(temp_df, aes(x=tvals, y=temp_plot)) + geom_line() +
  ggtitle("(b) Water Temperature") + xlab("days") + ylab("degrees F")

plot_grid(J_plot2, temp_plot3, ncol = 1, nrow = 2)
plot_grid(M_plot2, temp_plot3, ncol = 1, nrow = 2)


### TOTALS ===== ===== ===== ===== =====

# total number of fish
NJ_total = round(sol$NJ_tot[length(sol$NJ_tot)], digits=0)
NM_total = round(sol$NM_tot[length(sol$NM_tot)], digits=0)

# total number of cases due to each class
AJ_total = round(sol$AJ_tot[length(sol$AJ_tot)], digits=0)
IJ_total = round(sol$IJ_tot[length(sol$IJ_tot)], digits=0)
DJ_total = round(sol$DJ_tot[length(sol$DJ_tot)], digits=0)
WJ_total = round(sol$WJ_tot[length(sol$WJ_tot)], digits=0)
AM_total = round(sol$AM_tot[length(sol$AM_tot)], digits=0)
IM_total = round(sol$IM_tot[length(sol$IM_tot)], digits=0)
DM_total = round(sol$DM_tot[length(sol$DM_tot)], digits=0)
WM_total = round(sol$WM_tot[length(sol$WM_tot)], digits=0)

# total number of cases
NJ_cases_total = AJ_total + IJ_total + DJ_total + WJ_total
NM_cases_total = AM_total + IM_total + DM_total + WM_total

# total number of fish harvested
h_total = round(sol$h_tot[length(sol$h_tot)], digits=0)

# percentages of total number of fish
NJ_cases_per = round((NJ_cases_total/NJ_total)*100, digits=0)
NM_cases_per = round((NM_cases_total/NM_total)*100, digits=0)
AJ_per_total = round((AJ_total/NJ_total)*100, digits=0)
IJ_per_total = round((IJ_total/NJ_total)*100, digits=0)
DJ_per_total = round((DJ_total/NJ_total)*100, digits=0)
WJ_per_total = round((WJ_total/NJ_total)*100, digits=0)
AM_per_total = round((AM_total/NM_total)*100, digits=0)
```

```
IM_per_total = round((IM_total/NM_total)*100, digits=0)
DM_per_total = round((DM_total/NM_total)*100, digits=0)
WM_per_total = round((WM_total/NM_total)*100, digits=0)
h_per_total = round((h_total/NM_total)*100, digits=0)

# percentages of total number of cases
AJ_per_cases = round((AJ_total/NJ_cases_total)*100, digits=0)
IJ_per_cases = round((IJ_total/NJ_cases_total)*100, digits=0)
DJ_per_cases = round((DJ_total/NJ_cases_total)*100, digits=0)
WJ_per_cases = round((WJ_total/NJ_cases_total)*100, digits=0)
AM_per_cases = round((AM_total/NM_cases_total)*100, digits=0)
IM_per_cases = round((IM_total/NM_cases_total)*100, digits=0)
DM_per_cases = round((DM_total/NM_cases_total)*100, digits=0)
WM_per_cases = round((WM_total/NM_cases_total)*100, digits=0)


### R0 ===== ===== ===== ===== =====

# total population sizes
NJ = sol$SJ + sol$AJ + sol$IJ + sol$DJ
NM = sol$SM + sol$AM + sol$IM + sol$DM

# time & temperature dependent parameters
temp_R0 = tempfunc(tvals)
NC_R0 = NCfunc(temp_R0)
sigma_R0 = sigmafunc(NC_R0)
alpha_R0 = alphafunc(NC_R0)
gamma_R0 = gammafunc(NC_R0)

# R0 Calculation
R0J = (NJ*((gamma_R0 + mu + xi)*(betaA*eta*(gamma_R0 + mu) +
      betaID*eta*sigma_R0 + betaW*(gamma_R0*rhoA + mu*rhoA +
      rhoID*sigma_R0)) + alpha_R0*(betaA*eta*(gamma_R0 + mu + xi) +
      betaID*eta*sigma_R0 + betaW*(gamma_R0*rhoA + mu*rhoA + xi*rhoA +
      rhoID*sigma_R0)))) /
      (eta*(alpha_R0*gamma_R0*mu + alpha_R0*(mu + xi)*(mu + sigma_R0) +
      mu*(gamma_R0 + mu + xi)*(gamma_R0 + mu + sigma_R0)))

R0M = (NM*((gamma_R0 + mu + xi)*(betaA*eta*(gamma_R0 + mu) +
      betaID*eta*sigma_R0 + betaW*(gamma_R0*rhoA + mu*rhoA +
      rhoID*sigma_R0)) + alpha_R0*(betaA*eta*(gamma_R0 + mu + xi) +
      betaID*eta*sigma_R0 + betaW*(gamma_R0*rhoA + mu*rhoA + xi*rhoA +
      rhoID*sigma_R0)))) /
      (eta*(alpha_R0*gamma_R0*mu + alpha_R0*(mu + xi)*(mu + sigma_R0) +
      mu*(gamma_R0 + mu + xi)*(gamma_R0 + mu + sigma_R0)))

R0J_init = round(R0J[min(which(R0J > 1))], digits = 0)
R0M_init = round(R0M[min(which(R0M > 1))], digits = 0)
```

```
### RESULTS TABLE ===== ===== ===== ===== =====

results = c(NJ_total, NJ_cases_total, NJ_cases_per,
            AJ_total, AJ_per_total, AJ_per_cases,
            IJ_total, IJ_per_total, IJ_per_cases,
            DJ_total, DJ_per_total, DJ_per_cases,
            WJ_total, WJ_per_total, WJ_per_cases,
            0, 0, R0J_init,
            NM_total, NM_cases_total, NM_cases_per,
            AM_total, AM_per_total, AM_per_cases,
            IM_total, IM_per_total, IM_per_cases,
            DM_total, DM_per_total, DM_per_cases,
            WM_total, WM_per_total, WM_per_cases,
            h_total, h_per_total, R0M_init)

results_tab = matrix(results, nrow=18, ncol=2)

rownames(results_tab) =
  c("total fish: ", "total cases: ", "total cases (% of total # fish): ",
  "cases from A: ", "cases from A (% of total # fish): ",
  "cases from A (% of total # cases): ",
  "cases from I: ", "cases from I (% of total # fish): ",
  "cases from I (% of total # cases): ",
  "cases from D: ", "cases from D (% of total # fish): ",
  "cases from D (% of total # cases): ",
  "cases from W: ", "cases from W (% of total # fish): ",
  "cases from W (% of total # cases): ",
  "harvested: ", "harvested %: ", "R0: ")

colnames(results_tab) = c("J", "M")

cat("Results Table \n")
print(results_tab)
```