# Box 5 Projecting continuous biomass dynamics with pulsed removals

## Box overview

Inland freshwater fisheries systems are commonly harvested over short periods, representing a pulsed biomass removal. This is commonly the case when invasive species like common carp *Cyprinus carpio* and bigheaded carp *Hypophthalmichthys nobilis* use commercial fisheries. Additionally, biomass can be removed from a recreational fishery over a short period when short seasons occur within an annual time period. This box demonstrates how to project semidiscrete biomass dynamics models, which are continuous biomass dynamics models that allow for pulsed dynamics like biomass removals (Colvin et al. 2012).

## Libraries needed for analysis

The libraries listed in the code chunk below are required to numerically integrate the biomass dynamics models.

library(deSolve)

## Projecting semidiscrete biomass dynamics

To project semidiscrete biomass dynamics an additional function is needed to simulate the addition or removal from the system. First a function that specifies the ordinary differential equation the describes the biomass dynamics is specified in the code chunk below. This box assumes the Fox biomass dynamics model (Fox 1970). Note that in the code chunk, differs from previous box examples as the change in Yield over each time step is set to 0 because the amount of biomass removed from the system is a pulsed event the occurs at specific times. The way that numerical integrators work, simply specifying the removal amount in the function that defines the ODE is not sufficient because the corresponding change in biomass and harvest yield is adjusted by the time step (). By fixing dY to 0 in the function we can keep track of the amount of biomass harvested from the system for evaluating management policies.

ode\_fox<-function(t,x,parms)  
 {  
 # set the state variable   
 B<-x [1]  
 Y<-x [2]  
 # set the parameters  
 r<-parms[1]  
 Bmax<-parms[2]  
  
 # the Fox model of biomass dynamics as a   
 # ordinary differential equation  
 yield<- F\*B  
 dB<- r\*B\*(log(Bmax/B))   
 dY<- 0  
 return(list(c(dB,dY)))  
 }

To simulate the removal events another function is needed that is applied as specified time steps. The function has the same input arguments as ode\_fox. In the function instantaneous changes in biomass can be specified. In this example the fishing mortality is applied at a single instance and the biomass is reduced by the removal amount as B-B\*F. Additionally, the amount removed is specified as the biomass times the fishing mortality B\*F. The function is set up to keep track of biomass and yield. Defining the outputs to work with ode\_fox and solving using ode is a bit different. The values the function returns needs to be the same length as the initial values, so a vector ini is specified below that specifies the initial biomass at 10000 and harvest yield at 0. As you will see below this is a bit of a work around to keep track of the amount of biomass removed.

harvestFun <- function(t, x, parms){  
 B<- x[1]  
 F<- parms['F']  
 x[1] <- B-B\*F # biomass - removed biomass  
 x[2] <- B\*F # biomass removed  
 return(x)  
 }

Here the model parameters are specified for the intrinsic growth rate maximum biomass and fishing mortality. A vector of initial biomass amount is specified and harvest yield as an object ini.

parms<-c(r=0.3,Bmax=10000,F=0.1)  
ini<- c(B=10000,Y=0)

The final part of setting up the model to project biomass dynamics with pulsed removals requires specifying when biomass will be removed and the time to evaluate the model. In this example, biomass is pulsed out of the system at the beginning of year 1, 2, and 4. The model will project the biomass dynamics for 10 years using a timestep of 0.1. Experience using events in the ode function had found that at times floating points can result in an event not occurring and therefore rounding the times to the corresponding digits of the timestep minimizes this occurrence.

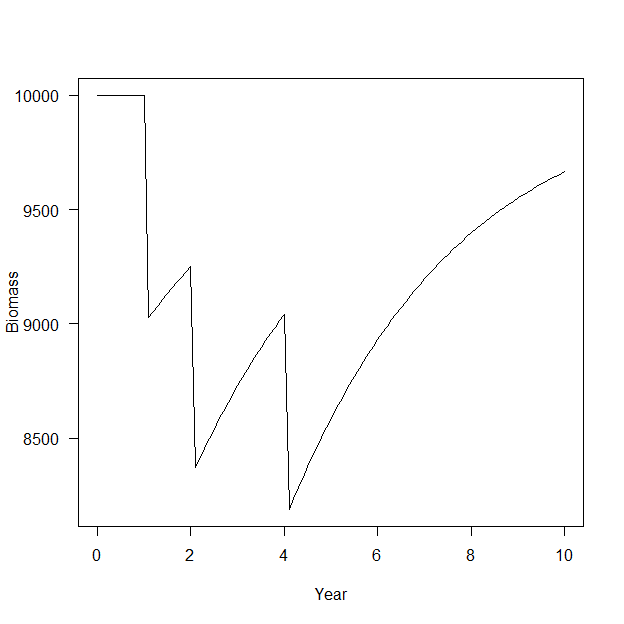
evtimes<-c(1,2,4) # when removals occur  
times<-round(seq(0,10,by=0.1),1) # round to time step digits

The model is project using the ode function. However a new argument is included, events. The events argument can take different forms, in this example the function that specifies how removals occur and the times those events occur are specified as a list.

modelout <-ode(  
 y = ini,   
 times = times,   
 func = ode\_fox,   
 parms = parms,  
 events = list(func= harvestFun, time=evtimes),  
 method = "lsoda")  
modelout<-as.data.frame(modelout)

The results are plotted using the code chunk below. You can see the change in biomass at times 1, 2, and 4 that result from the pulsed removal.

plot(B~time,modelout,type='l',ylab="Biomass",xlab="Year",las=1)

 To fully interpret the amount of biomass removed we need to modify the output a bit. In short the model keeps track of biomass and harvest yield with changes governed by the ODE. However when removal events occur the state variables (biomass, harvest yield) are changed for that instant. Looking at the results of the output below you can see where the harvest event was triggered at time 1, 2, and 4 but the harvest yield values then remain the same until the next event. This is because the change in harvest yield dY was set to 0. The harvest yield is this rest at the next event.

modelout

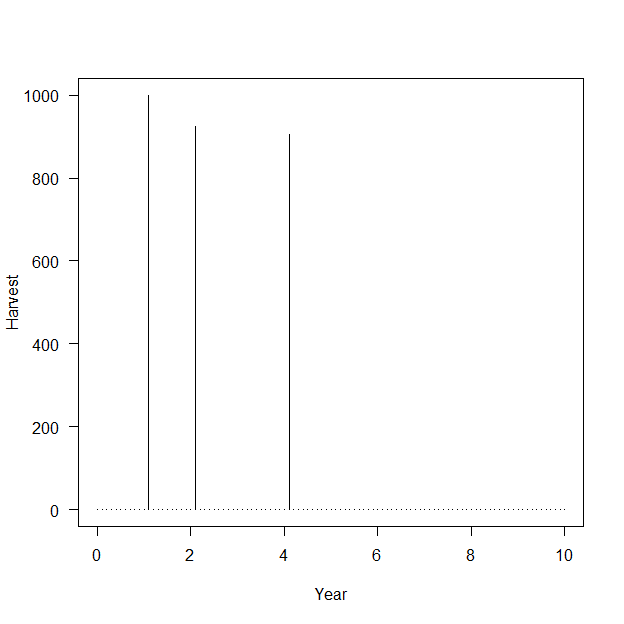
## time B Y  
## 1 0.0 10000.000 0.0000  
## 2 0.1 10000.000 0.0000  
## 3 0.2 10000.000 0.0000  
## 4 0.3 10000.000 0.0000  
## 5 0.4 10000.000 0.0000  
## 6 0.5 10000.000 0.0000  
## 7 0.6 10000.000 0.0000  
## 8 0.7 10000.000 0.0000  
## 9 0.8 10000.000 0.0000  
## 10 0.9 10000.000 0.0000  
## 11 1.0 10000.000 0.0000  
## 12 1.1 9028.070 1000.0000  
## 13 1.2 9055.394 1000.0000  
## 14 1.3 9081.990 1000.0000  
## 15 1.4 9107.872 1000.0000  
## 16 1.5 9133.060 1000.0000  
## 17 1.6 9157.571 1000.0000  
## 18 1.7 9181.420 1000.0000  
## 19 1.8 9204.623 1000.0000  
## 20 1.9 9227.197 1000.0000  
## 21 2.0 9249.157 1000.0000  
## 22 2.1 8369.488 924.9157  
## 23 2.2 8413.633 924.9157  
## 24 2.3 8456.696 924.9157  
## 25 2.4 8498.695 924.9157  
## 26 2.5 8539.652 924.9157  
## 27 2.6 8579.588 924.9157  
## 28 2.7 8618.522 924.9157  
## 29 2.8 8656.474 924.9157  
## 30 2.9 8693.465 924.9157  
## 31 3.0 8729.513 924.9157  
## 32 3.1 8764.639 924.9157  
## 33 3.2 8798.862 924.9157  
## 34 3.3 8832.201 924.9157  
## 35 3.4 8864.675 924.9157  
## 36 3.5 8896.304 924.9157  
## 37 3.6 8927.106 924.9157  
## 38 3.7 8957.100 924.9157  
## 39 3.8 8986.304 924.9157  
## 40 3.9 9014.735 924.9157  
## 41 4.0 9042.413 924.9157  
## 42 4.1 8187.874 904.2413  
## 43 4.2 8236.400 904.2413  
## 44 4.3 8283.766 904.2413  
## 45 4.4 8329.991 904.2413  
## 46 4.5 8375.097 904.2413  
## 47 4.6 8419.103 904.2413  
## 48 4.7 8462.030 904.2413  
## 49 4.8 8503.897 904.2413  
## 50 4.9 8544.725 904.2413  
## 51 5.0 8584.534 904.2413  
## 52 5.1 8623.344 904.2413  
## 53 5.2 8661.174 904.2413  
## 54 5.3 8698.045 904.2413  
## 55 5.4 8733.977 904.2413  
## 56 5.5 8768.988 904.2413  
## 57 5.6 8803.099 904.2413  
## 58 5.7 8836.328 904.2413  
## 59 5.8 8868.695 904.2413  
## 60 5.9 8900.219 904.2413  
## 61 6.0 8930.919 904.2413  
## 62 6.1 8960.812 904.2413  
## 63 6.2 8989.918 904.2413  
## 64 6.3 9018.254 904.2413  
## 65 6.4 9045.838 904.2413  
## 66 6.5 9072.687 904.2413  
## 67 6.6 9098.819 904.2413  
## 68 6.7 9124.250 904.2413  
## 69 6.8 9148.998 904.2413  
## 70 6.9 9173.079 904.2413  
## 71 7.0 9196.508 904.2413  
## 72 7.1 9219.303 904.2413  
## 73 7.2 9241.477 904.2413  
## 74 7.3 9263.048 904.2413  
## 75 7.4 9284.029 904.2413  
## 76 7.5 9304.435 904.2413  
## 77 7.6 9324.281 904.2413  
## 78 7.7 9343.581 904.2413  
## 79 7.8 9362.349 904.2413  
## 80 7.9 9380.598 904.2413  
## 81 8.0 9398.342 904.2413  
## 82 8.1 9415.593 904.2413  
## 83 8.2 9432.365 904.2413  
## 84 8.3 9448.670 904.2413  
## 85 8.4 9464.520 904.2413  
## 86 8.5 9479.927 904.2413  
## 87 8.6 9494.902 904.2413  
## 88 8.7 9509.458 904.2413  
## 89 8.8 9523.605 904.2413  
## 90 8.9 9537.353 904.2413  
## 91 9.0 9550.715 904.2413  
## 92 9.1 9563.699 904.2413  
## 93 9.2 9576.316 904.2413  
## 94 9.3 9588.577 904.2413  
## 95 9.4 9600.490 904.2413  
## 96 9.5 9612.065 904.2413  
## 97 9.6 9623.312 904.2413  
## 98 9.7 9634.238 904.2413  
## 99 9.8 9644.854 904.2413  
## 100 9.9 9655.167 904.2413  
## 101 10.0 9665.186 904.2413

To clean up this artifact the harvest yield at times other than when the event occurred are set to 0 using the code below. Note that while the event was triggered at time 1, 2, and 4 the harvest yield is recorded in timestep 1.1, 2.1, and 4.1. The code below sets values other than the event times (i.e., 1, 2, 4) plus the time step to 0 as harvest did not occur at those times, but the ode function carried the harvest yield over those steps because dY was set to 0 so that the harvest yield could be tracked in the projection. The round function is used to ensure the right times are selected.

modelout[!(modelout$time%in%(round(evtimes+0.1,1))),]$Y<-0

The plot below illustrates the pulsed harvest removals that occurred at time 1, 2, and 4.

plot(Y~time,modelout,type="h",ylab="Harvest",xlab="Year",las=1)



## References

Colvin, M. E., C. L. Pierce, and T. W. Stewart. 2012. Semidiscrete biomass dynamic modeling: An improved approach for assessing fish stock responses to pulsed harvest events. Canadian Journal of Fisheries and Aquatic Sciences 69:1710–1721.

Fox, W. W. 1970. An exponential surplus-yield model for optimizing exploited fish populations. Transactions of the American Fisheries Society 99:80–88.