# Projecting continous biomass dynamics

This box demonstrates how to:

1. project continuous biomass dynamics from an ordinary differential equation using R,
2. determine the appropriate time step for the numerical integrator, and
3. the effect of different numerical integrators.

## Libraries needed for analysis

The code below loads the deSolve package that can be used to solve oridinary differential equations (ODEs) or difference equestions in R (Soetaert et al. 2010).

library(deSolve)

## Setting up the biomass dynamics model

Projecting biomass dynamics as an ordinary differential equation (ode) or a difference model can be done using a function. The function has a a couple of parts to it that are common regardless of the model (e.g., Graham Schaefer, Pella Tomlinson, Fox) or the type of model (ode or difference). The 4 parts within the function are:

1. Defining the state variables (e.g., biomass, harvest)
2. Defining the parameters (e.g., intrinsic growth rate, carrying capacity, fishing mortality)
3. Definging the ode or difference equation. This part differs between the 2 types and is demonstrated below.
4. A return() function that is a list of the state variables and other quantities that are of interest like the amount of fish harvested.

## Graham Schaefer Model

The function below sets up the Graham Schaefer model Graham (1935) as an ODE where and that can be used with a numerical integrator from the deSolve package. The function takes 3 input arguments, t which is the timestep, x which is a vector of the state variables, and parms which is a vector of the parameters.

# Graham Schaefer model specified as an ODE  
ode\_gs<-function(t,x,parms)  
 {  
 # 1. set the state variable   
 B<-x   
   
 # 2. set the parameters  
 r<-parms["r"]  
 K<-parms["K"]  
 F<-parms["F"]  
   
 # 3. set up the Graham-Schaefer model of biomass dynamics as an ode  
 dB<- r\*B\*(1-(B/K))-F\*B  
   
 # 4. return the relevant quantites for the time step  
 return(list(dB))  
 }

To run the model and project biomass the values for each parameter must be specified in a vector.

parameters<-c(r=0.6,K=10000,F=0.1)

The initial biomass must also be set to give the projected biomass dynamics as starting point. Here the initial biomass is set to 10,000.

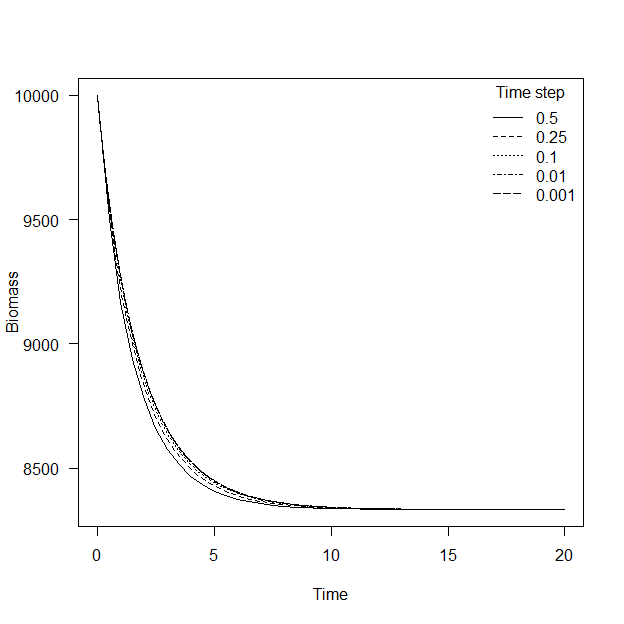
initial\_biomass<- c(B=10000)

The ode() function from the deSolve package provides several numerical integrators (e.g., Euler, Runge-Kutta 4) that can be used to project biomass dynamics. There is no set rules for which, in general if biomass dynamics are smooth over time the type of integrator is a tradeoff of the resolution of the timestep and whether dynamics are the similar throughout the projection.

dt<-c(0.5,0.25, 0.1, 0.01,0.001)  
out<-data.frame()  
for(i in 1:length(dt))  
 {  
 solution<- ode(  
 y=initial\_biomass,   
 times=seq(0,20,by=dt[i]),   
 func=ode\_gs,   
 parms=parameters,   
 method="euler")  
 solution<-as.data.frame(solution)  
 solution$dt<-dt[i]  
 out<-rbind(out,solution)  
 }

The code below plots the solutions for the projected biomass for decreasing time steps use for the numerican integration.

plot(B~time,solution,  
 ylab="Biomass",  
 xlab="Time",  
 las=1,  
 type='n',  
 main="")  
points(B~time,out,subset=dt==0.5,type="l",lty=1)  
points(B~time,out,subset=dt==0.25,type="l",lty=2)  
points(B~time,out,subset=dt==0.1,type="l",lty=3)  
points(B~time,out,subset=dt==0.01,type="l",lty=4)  
points(B~time,out,subset=dt==0.001,type="l",lty=5)  
legend("topright",title="Time step",c("0.5","0.25","0.1","0.01","0.001"),  
 lty=c(1,2,3,4,5),bty="n")



### Interpretation

The ending biomass is the same for all the 5 time steps evaluated. However, there was some discrepancy in projected biomass up to year 10. The 0.5 time step resulted in biomass projections that were lower than the 0.01 and 0.001 time steps. The take away from this example is that the time step can be important to accurately project biomass dynamics. As a rule of thumb the smallest time step within the computational burden allowed should be used. In other words, as time step decreases in size the amount of computer time needed to project the dynamics increases, therefore pick a time step that is sufficiently accurate for the needs of the analysis and the model runs in a reasonable amount of time.

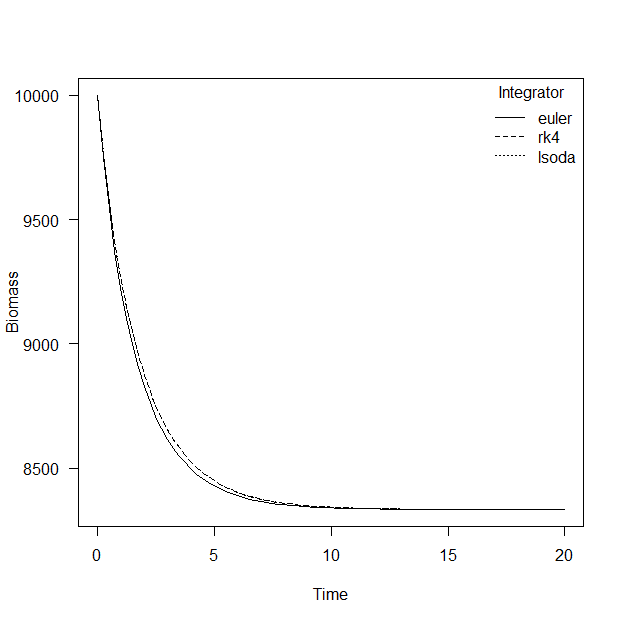
## Comparing numerical integrators

Similar to the issue with time step, the accuracy of numerical integrations varies with the numerical integrator used. The most basic numerical integrator is the Euler (euler) while the Runge-Kutta 4 (rk4) and and the Livermore family (lsoda) are much more sophisticated and in many cases more accurate at a coarser time step than the Euler (Soetaert and Herman 2009, Stevens 2009). The code chunk below loops over the 3 numerican integrates using the parameters and intial biomass of the previous example to with a time step of 0.25.

integrator<-c("euler","rk4","lsoda")  
out<-data.frame()  
for(i in 1:length(integrator))  
 {  
 solution<- ode(  
 y=initial\_biomass,   
 times=seq(0,20,by=0.25),   
 func=ode\_gs,   
 parms=parameters,   
 method=integrator[i])  
 solution<-as.data.frame(solution)  
 solution$integrator<-integrator[i]  
 out<-rbind(out,solution)  
 }

The code below plots the solutions for the 3 integrators. Similar to the previous results where the euler integrator was used for decreasing time steps, the projections using the rk4 and the lsoda integrators are identical but slighly higher than the euler.

plot(B~time,solution,  
 ylab="Biomass",  
 xlab="Time",  
 las=1,  
 type='n',  
 main="")  
points(B~time,out,subset=integrator=="euler",type="l",lty=1)  
points(B~time,out,subset=integrator=="rk4",type="l",lty=2)  
points(B~time,out,subset=integrator=="lsoda",type="l",lty=3)  
legend("topright",title="Integrator", c("euler","rk4","lsoda"),  
 lty=c(1,2,3),bty="n")



## Take home

Any integrator can be used to numerically solve and project biomass dynamics using ODEs, but sone intial effort should be used to evaluate which integrator will be used and what time step is sufficient for the problem. The most accurate projects will use small time steps but there are diminishing returns in accuracy that trade off with model run times. In closing a time step is sufficiently small if decreasing it does not appreiciable change the biomass dynamics over time simulated.

Graham, M. 1935. Modern theory of exploiting a fishery and applications to north sea trawling. J. Cons. Int. Exp. Mer 10:264–274.

Soetaert, K., and P. M. J. Herman. 2009. A practical guide to ecological modelling : Using r as a simulation platform. book, Springer, Dordrecht.

Soetaert, K., T. Petzoldt, and R. Setzer. 2010. Solving differential equations in r: Package deSolve. Journal of Statistical Software 33:1–25.

Stevens, M. H. H. 2009. A primer of ecology with r. book, Springer, New York.