Figures

Henry Traynor

2024-04-15

set.seed(1)  
  
att.param <- list(n1 = 500,  
 n2 = 0,  
 b1 = 0.65,  
 b2 = 0.65,  
 k1 = 500,  
 k2 = 500,  
 a12 = 1,  
 a21 = 1,  
 a11 = 1,  
 a22 = 1,  
 del1 = 0,  
 del2 = 2)  
  
singleWindow.time.param <- list(tau = 1/52,  
 time.invade = 0,  
 time.window = 52,  
 time.index = 520)  
  
source("parameter/modelParam.R")  
source("simulations/alphaSim.R")  
source("statistics/singleWindowStat.R")

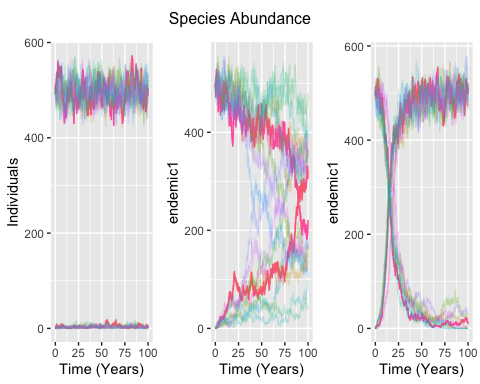
## This is GoeVeg 0.7.2 - build: 2024-02-06

source("simulations/halfAndHalf.R")  
source("statistics/ROCcurveData.R")

## 1. Outcomes of Model

These plots display the three outcomes of the Lotka-Volterra Competition model: endemic survival, coexistence, and invader success.

library(ggplot2)  
library(cowplot)  
library(gridExtra)  
library(grid)  
library(reshape2)  
  
setOfAlphaSimPlot <- function(numRealizations, att.param, singleWindow.time.param, do.prob, ratio.max, ttb, do.win) {  
 time.max = ttb\*2  
 tau = singleWindow.time.param$tau  
 df.data <- data.frame(time=seq(0,time.max, by=tau))  
 i=0  
 while(i<numRealizations) {  
 df.subData = alphaSim(att.param, singleWindow.time.param, do.prob, ratio.max, ttb, do.win)  
 colnames(df.subData) = c("time", paste("endemic",i+1,sep=""), paste("invader",i+1,sep=""), "ratio")  
 df.data = cbind(df.data, df.subData[2:3])  
 i=i+1  
 }  
 plot = ggplot(data=df.data, aes(x=time)) + xlab("Time (Years)")  
 plot = plot + geom\_line(aes(y=endemic1, color="Endemic")) + geom\_line(aes(y=invader1, color="Invader"))  
 df.data = melt(df.data, id.vars="time")  
 plot = plot + geom\_line(data=df.data, aes(time,value,color=variable), alpha=0.2) + theme(legend.position="none")  
 return(plot)  
}  
  
endemic <- setOfAlphaSimPlot(numRealizations=10, att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=0.5, ttb=50, do.win=FALSE)  
coexistence <- setOfAlphaSimPlot(numRealizations=10, att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=1, ttb=50, do.win=FALSE)  
invader <- setOfAlphaSimPlot(numRealizations=10, att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=2.0, ttb =50, do.win=FALSE)  
  
grid.arrange(textGrob("Species Abundance", gp=gpar(size=32)), endemic+ylab("Individuals"), coexistence+theme(legend.position="none"), invader,   
 ncol=3, nrow = 2,   
 layout\_matrix = rbind(c(1,1,1), c(2,3,4)),  
 widths = c(13,13,13), heights = c(0.25, 2.5))

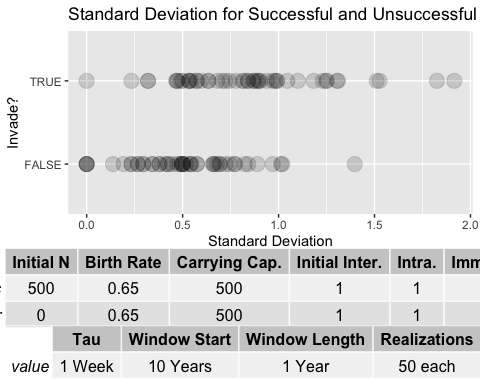


## 2. Two Types of Realizations

library(ggplot2)  
  
setOfIncAlphaSimPlot <- function(numRealizations, att.param, singleWindow.time.param, do.prob, ttb) {  
 time.max = ttb\*2  
 tau = singleWindow.time.param$tau  
 df.data <- data.frame(time=seq(0,time.max, by=tau))  
 i=0  
 while(i<numRealizations) {  
 df.subData1 = alphaSim(att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=2, ttb=50, do.win=TRUE)  
 colnames(df.subData1) = c("time", paste("endemic",i+1,sep=""), paste("invaderInc",i+1,sep=""), "ratio")  
 df.subData2 = alphaSim(att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=0.5, ttb=50, do.win=FALSE)  
 colnames(df.subData2) = c("time", paste("endemic",i+1,sep=""), paste("invaderConstant",i+1,sep=""), "ratio")  
 df.data = cbind(df.data, df.subData1[3], df.subData2[3])  
 i=i+1  
 }  
 plot = ggplot(data=df.data, aes(x=time)) + xlab("Time(Years)")  
 plot = plot + geom\_line(aes(y=invaderInc1, color="Increasing")) + geom\_line(aes(y=invaderConstant1, color="Invader"))  
 df.data = melt(df.data, id.vars="time")  
 plot = plot + geom\_line(data=df.data, aes(time,value,color=variable), alpha=0.2) + theme(legend.position="none") + ylab("Abundance")  
 return(plot)  
}  
  
typesPlot <- setOfIncAlphaSimPlot(10, att.param, singleWindow.time.param, do.prob=TRUE, ttb=50)

## 3. Single Window Statistics

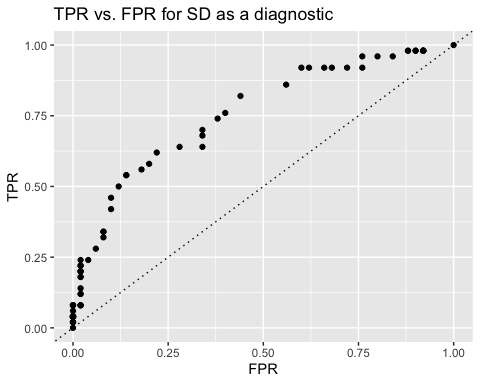
library(gridExtra)  
library(ggplot2)  
df.test <- halfAndHalf(att.param,  
 singleWindow.time.param,  
 ttb=25,  
 halfSimulations=50,  
 win.ratio=2,  
 fail.ratio=0.1)  
  
df.SD <- singleWindowStat(df.data=df.test,  
 time.window=52,  
 time.index=520,  
 func="sd")  
  
endemic.data = data.frame(endemic=c(att.param[1],att.param[3],att.param[5],att.param[7],att.param[9],att.param[11]))  
colnames(endemic.data) = c('Initial N', 'Birth Rate', 'Carrying Cap.', 'Initial Inter.', 'Intra.', 'Immigration')  
invader.data = data.frame(invader=c(att.param[2],att.param[4],att.param[6],att.param[8],att.param[10],att.param[12]))  
colnames(invader.data) = c('Initial N', 'Birth Rate', 'Carrying Cap.', 'Initial Inter.', 'Intra.', 'Immigration')  
parameters = rbind(endemic.data, invader.data)  
rownames(parameters) = c("endemic", 'invader')  
  
time.parameters = as.data.frame(t(c('1 Week', '10 Years', '1 Year', '50 each')))  
colnames(time.parameters) = c('Tau', 'Window Start', 'Window Length', 'Realizations')  
rownames(time.parameters) = c('value')  
  
plot <-ggplot(data=df.SD,  
 aes(x=stat, y=do.win)) +  
 geom\_point(alpha=0.15, size=5) +  
 labs(x='Standard Deviation', y="Invade?") +  
 ggtitle("Standard Deviation for Successful and Unsuccessful Invasions")  
  
grid.arrange(  
 arrangeGrob(plot),  
 tableGrob(parameters),  
 tableGrob(time.parameters),  
 layout\_matrix = cbind(c(1,1,1,1,2,3),  
 c(1,1,1,1,2,3),  
 c(1,1,1,1,2,3),  
 c(1,1,1,1,2,3))  
)

 ## ROC Curve for Single Stat

library(DescTools)  
df.rates <- ROCcurveData(df.SD, numThresholds = 75, inequality = ">", statistic="sd")  
auc <- AUC(x=df.rates$FPR, y=df.rates$TPR, method='trapezoid')

## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):  
## collapsing to unique 'x' values

ggplot(data=df.rates, aes(x=FPR, y=TPR)) + geom\_point() + geom\_abline(slope=1, intercept=0, linetype=3) + ggtitle("TPR vs. FPR for SD as a diagnostic") + theme(legend.position="bottom")

 ## 4. Several Statistics and Realization Types

library(goeveg)  
library(moments)  
# increasing ratio  
df.increasing <- halfAndHalf(att.param, singleWindow.time.param, ttb=25, halfSimulations=100, win.ratio=2, fail.ratio=0.1, increasing=TRUE)  
# constant ratio: a12 > a21  
df.constant <- halfAndHalf(att.param, singleWindow.time.param, ttb=25, halfSimulations=100, win.ratio=2, fail.ratio=0.1, increasing=FALSE)  
  
#statistics  
stats <- c("mean", "sd", "cv", "kurtosis", "skewness")  
inequalities <- c(">", ">", "<", "<", "<")  
windows <- c(52,104,156)  
time.index=520  
  
singleStatWindows <- function(df.data, windows, time.index, statsNum) {  
 df1 <- singleWindowStat(df.data, time.window=windows[1], time.index, func=stats[statsNum])  
 df2 <- singleWindowStat(df.data, time.window=windows[2], time.index, func=stats[statsNum])  
 df3 <- singleWindowStat(df.data, time.window=windows[3], time.index, func=stats[statsNum])  
 df.stat.type <- cbind(df1,df2,df3)  
 colnames(df.stat.type) = c("do.win", "stat1","do.win2", "stat2","do.win3", "stat3")  
 return(df.stat.type)  
}  
  
dfListing <- function(df.increasing, df.constant, stats) {  
 statsList = list()  
 for(i in 1:length(stats)) {  
 df.stat.increasing = singleStatWindows(df.increasing, windows, time.index, i)  
 df.stat.increasing = subset(df.stat.increasing, select = -c(do.win2, do.win3))  
 df.stat.constant = singleStatWindows(df.constant, windows,time.index, i)  
 df.stat.constant = subset(df.stat.constant, select = -c(do.win2, do.win3))  
 statsList[[i]] = df.stat.increasing  
 statsList[[length(stats)+i]] = df.stat.constant  
 }  
 return(statsList)  
}  
  
statsList <- dfListing(df.increasing, df.constant, stats)

## 5. ROC Data

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ lubridate 1.9.3 ✔ tibble 3.2.1  
## ✔ purrr 1.0.2 ✔ tidyr 1.3.1  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::combine() masks gridExtra::combine()  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ lubridate::stamp() masks cowplot::stamp()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

numThresholds=200  
  
ROCdata <- function(statsList, numThresholds, stats, windows, inequalities) {  
 columns = c("threshold", "TPR", "FPR", "Statistic", "realizationType", "windowSize")  
 roc.df = data.frame(matrix(nrow =0, ncol = length(columns)))  
 colnames(roc.df) = columns  
 ineqList = rep(1:5,2)  
 for(i in 1:length(statsList)) {  
 for(j in 1:length(windows)) {  
 roc.df = rbind(roc.df, arrange(.data=ROCcurveData(statsList[[i]][c(1,j+1)], numThresholds, inequalities[ineqList[i]], stats[ineqList[i]]), by\_group=TPR))  
 }  
 }  
 roc.df$realizationType = c(matrix(replicate(15\*numThresholds, "Increasing")),  
 matrix(replicate(15\*numThresholds, "Constant")))  
 roc.df$windowSize = c(replicate(2,replicate(5,cbind(replicate(numThresholds,ttb.times[1]),replicate(numThresholds,ttb.times[2]),replicate(numThresholds,ttb.times[3])))))  
 colnames(roc.df) = columns  
 return(roc.df)  
}  
  
roc.df <- ROCdata(statsList, numThresholds, stats, windows, inequalities)

## 6. ROC Plots

library(ggplot2)  
library(tidyverse)  
  
## with facet  
ggplot(data=roc.df, aes(x=FPR, y=TPR, color=windowSize)) +   
 geom\_line(aes(group=windowSize)) +   
 facet\_grid(Statistic ~ realizationType) +  
 geom\_abline(slope=1, intercept=0, linetype=3)

