Figures

Henry Traynor

2024-03-22

```r
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
```

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

1. Outcomes of Model

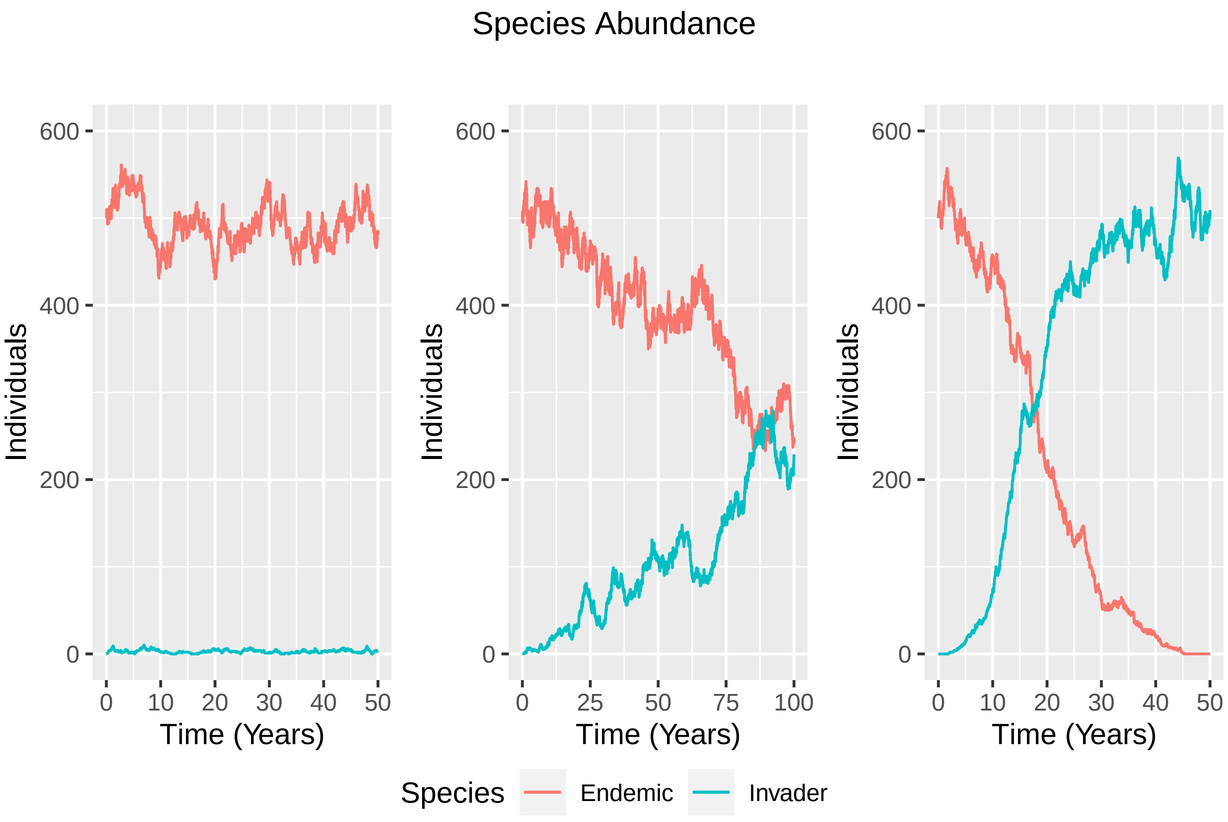
These plots display the three outcomes of the Lotka-Volterra Competition model: endemic survival, coexistence, and invader success.

```r
library(ggplot2)
library(cowplot)
library(gridExtra)
library(grid)

df.endemic <- alphaSim(att.param, singleWindow.time.param, ratio.max=0.5, ttb=25, do.win=FALSE)
df.coexistence <- alphaSim(att.param, singleWindow.time.param, ratio.max=1, ttb=50, do.win=FALSE)
df.invader <- alphaSim(att.param, singleWindow.time.param, ratio.max=2.0, ttb =25, do.win=FALSE)

endemic.plot <- ggplot(data=df.endemic, aes(x=time)) +
                  geom_line(aes(y=endemic, color="Endemic")) +
                  geom_line(aes(y=invader, color="Invader")) +
                  xlab("Time (Years)") +
                  ylab("Individuals") + theme(legend.position = "none") + ylim(c(0,600))

coexistence.plot <- ggplot(data=df.coexistence, aes(x=time)) +
                      geom_line(aes(y=endemic, color="Endemic")) +
                      geom_line(aes(y=invader, color="Invader")) +
                      xlab("Time (Years)") +
                      ylab("Individuals") + theme(legend.position = "none") + ylim(c(0,600))

invader.plot <- ggplot(data=df.invader, aes(x=time)) +
                  geom_line(aes(y=endemic, color="Endemic")) +
                  geom_line(aes(y=invader, color="Invader")) +
                  xlab("Time (Years)") +
                  ylab("Individuals") + theme(legend.position="bottom") + guides(color= guide_legend(ti


legend <- get_legend(invader.plot)
```

```
## Warning: Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
```

```
grid.arrange(textGrob("Species Abundance", gp=gpar(size=32)), endemic.plot, coexistence.plot, invader.pl
             ncol=3, nrow = 3,
             layout_matrix = rbind(c(1,1,1), c(2,3,4), c(5,5,5)),
             widths = c(10,10,10), heights = c(0.5, 2.5, 0.2))
```

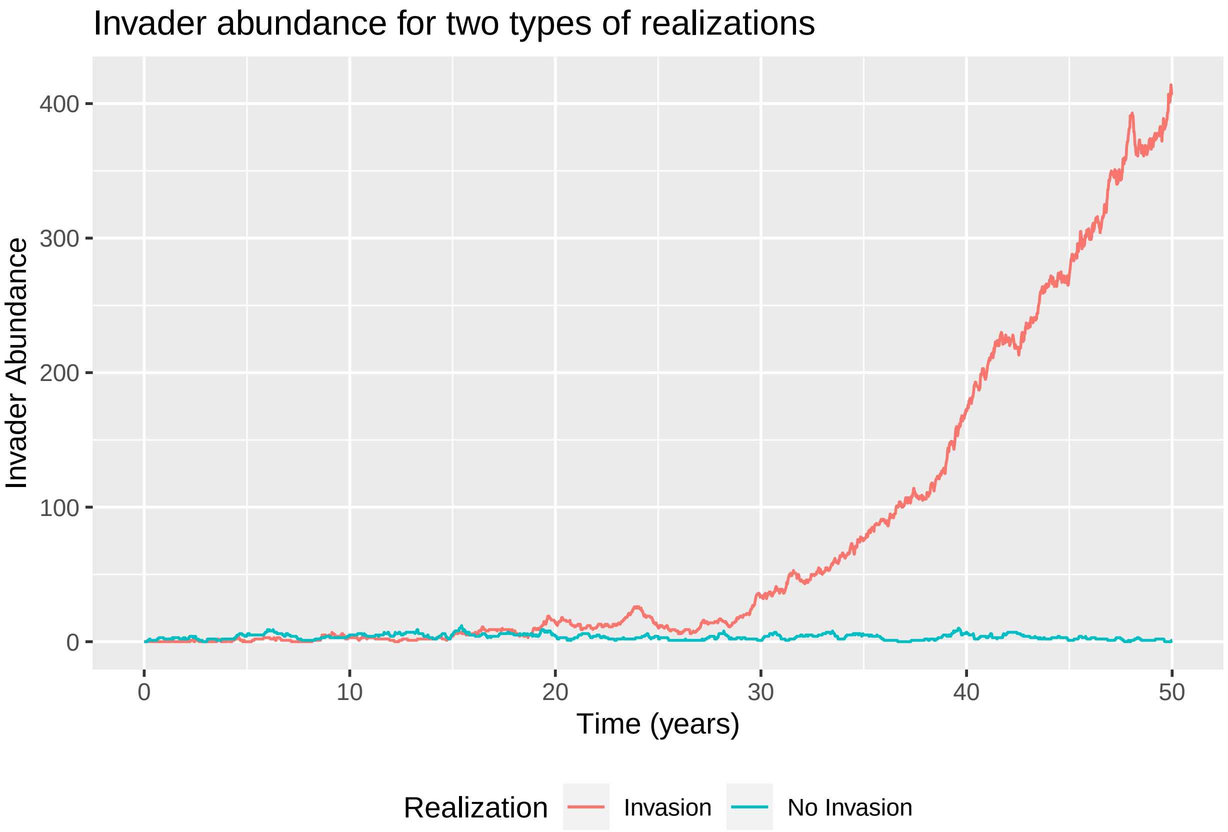
```
## Warning: Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
```



2. Two Types of Realizations

```
library(ggplot2)
df.invaderWin <- alphaSim(att.param, singleWindow.time.param, ratio.max=2, ttb=25, do.win=TRUE)
df.invaderLose <- alphaSim(att.param, singleWindow.time.param, ratio.max=0.5, ttb=25, do.win=FALSE)

df.realizations <- cbind(df.invaderWin, df.invaderLose)
colnames(df.realizations) <- c("time1", "endemic1", "invader1", "ratio1", "time2", "endemic2", "invader2

ggplot(data=df.realizations, aes(x=time1)) + geom_line(aes(y=invader1, color="Invasion")) + geom_line(ae
```

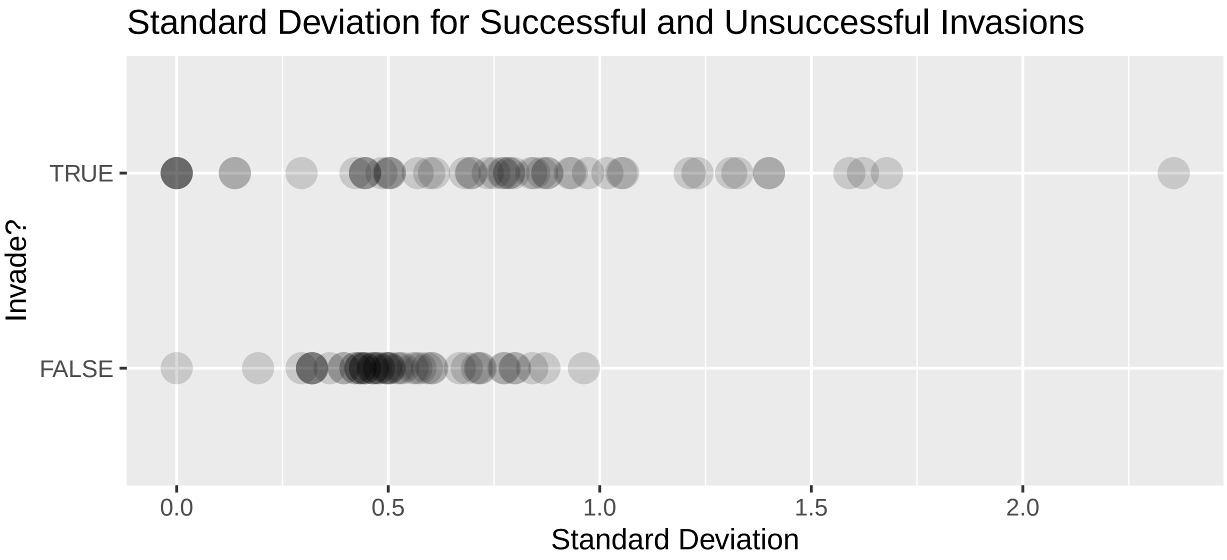
```
## Warning: Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
```



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
colnames(endemic.data) = c('Initial N', 'Birth Rate', 'Carrying Cap.', 'Initial Inter.', 'Intra.', 'Immi
invader.data = data.frame(invader=c(att.param[2],att.param[4],att.param[6],att.param[8],att.param[10],at
colnames(invader.data) = c('Initial N', 'Birth Rate', 'Carrying Cap.', 'Initial Inter.', 'Intra.', 'Immi
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')
```

```r
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))
)
```



	Initial N	Birth Rate	Carrying Cap.	Initial Inter.	Intra.	Immigration
endemic	500	0.65	500	1	1	0
invader	0	0.65	500	1	1	2

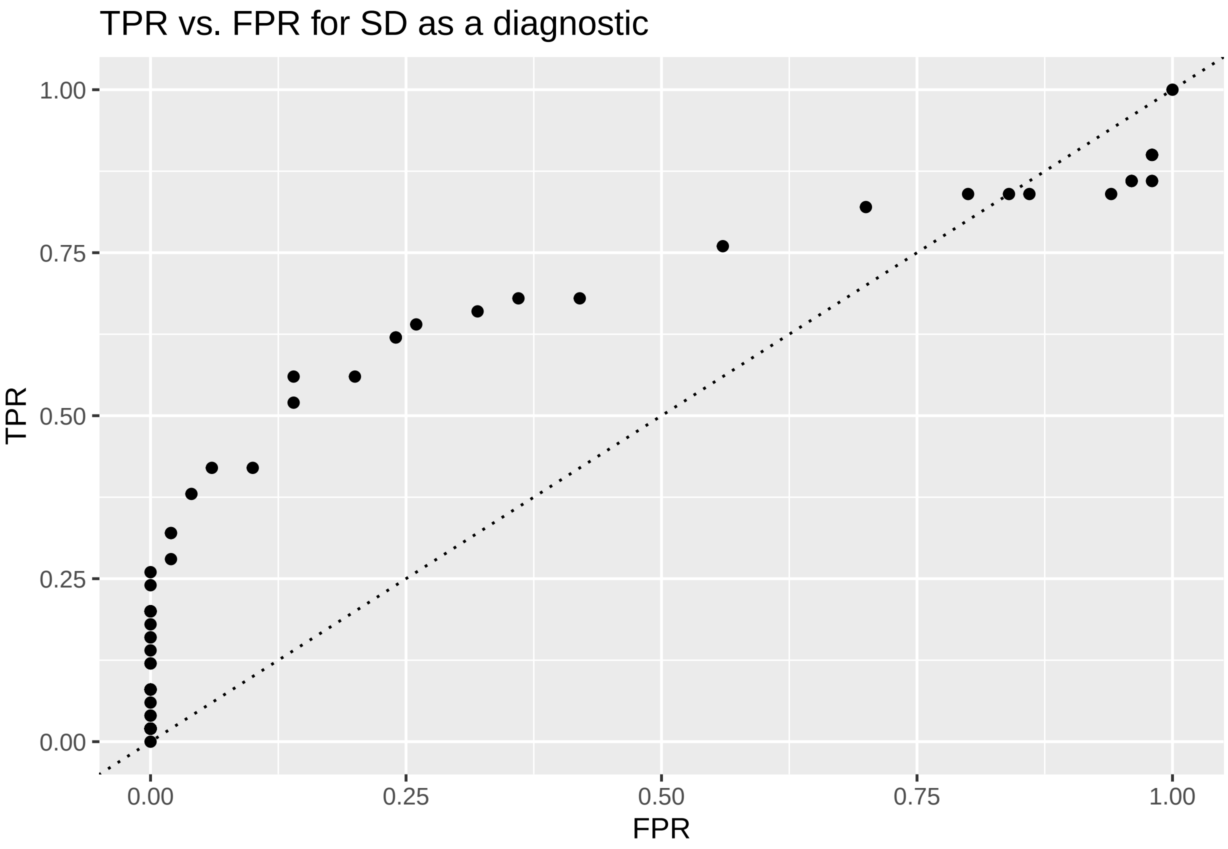
	Tau	Window Start	Window Length	Realizations
value	1 Week	10 Years	1 Year	50 each

## ROC Curve for Single Stat

```r
library(DescTools)
df.rates <- ROCcurveData(df.SD, numThresholds = 75, inequality = ">")
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)
```


TPR vs. FPR for SD as a diagnostic

## 4. Several Statistics and Realization Types

```
library(goeveg)
library(moments)
# increasing ratio
df.increasing <- halfAndHalf(att.param, singleWindow.time.param, ttb=25, halfSimulations=50, win.ratio=
# constant ratio: a12 > a21
df.constant <- halfAndHalf(att.param, singleWindow.time.param, ttb=25, halfSimulations=50, win.ratio=2,

#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)
```

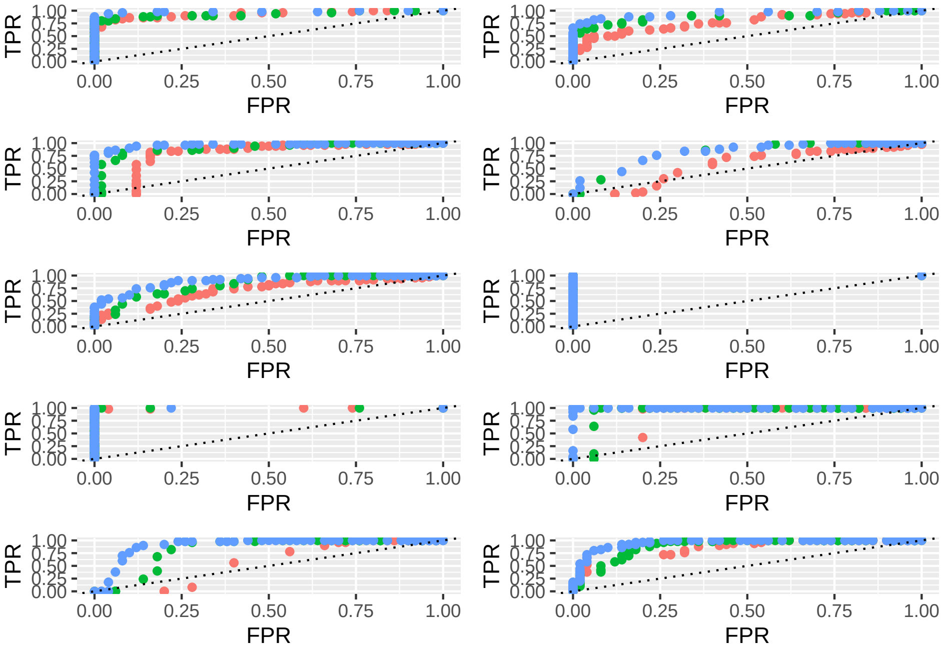
5. ROC Data

```
numThresholds=100

ROCdataList <- function(statsList, numThresholds, stats, windows, inequalities) {
  rocList = list()
  ineqList = rep(1:5,2)
  for(i in 1:length(statsList)) {
    temp.list = list()
    for(j in 1:length(windows)) {
      temp.list[[j]] = ROCcurveData(statsList[[i]][c(1,j+1)], numThresholds, inequalities[ineqList[i]])
    }
    df.stat.type.roc = cbind(temp.list[[1]],
                             temp.list[[2]],
                             temp.list[[3]])
    colnames(df.stat.type.roc) = c("threshold", "TPR1", "FPR1", "threshold2", "TPR2", "FPR2", "threshol
    df.stat.type.roc = subset(df.stat.type.roc, select=-c(threshold2, threshold3))
    rocList[[i]] = df.stat.type.roc
  }
  return(rocList)
}

roc.list <- ROCdataList(statsList, numThresholds, stats, windows, inequalities)
```

6. ROC Plots

```
library(ggplot2)
plotList <- function(roc.list) {
  roc.plot.list = list()
  for(i in 1:length(roc.list)) {
    curve.stat.type = ggplot() +
      geom_point(data=roc.list[[i]], aes(x=FPR1, y=TPR1, color="1 year")) +
      geom_point(data=roc.list[[i]], aes(x=FPR2, y=TPR2, color="2 years")) +
      geom_point(data=roc.list[[i]], aes(x=FPR3, y=TPR3, color="3 years")) +
      geom_abline(slope=1, intercept=0, linetype=3) +
      theme(legend.position = "none") +
      xlab("FPR") + ylab("TPR")
    roc.plot.list[[i]] = curve.stat.type
  }
  return(roc.plot.list)
}

roc.plot.list <- plotList(roc.list)
```

```
library(gridExtra)
library(ggplot2)
grid.arrange(
  roc.plot.list[[1]],
  roc.plot.list[[2]],
  roc.plot.list[[3]],
  roc.plot.list[[4]],
  roc.plot.list[[5]],
  roc.plot.list[[6]],
  roc.plot.list[[7]],
  roc.plot.list[[8]],
  roc.plot.list[[9]],
  roc.plot.list[[10]],
  ncol=2
)
```


TPR
FPR