Figures

Henry Traynor

2024-04-18

```
set.seed(1)
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.

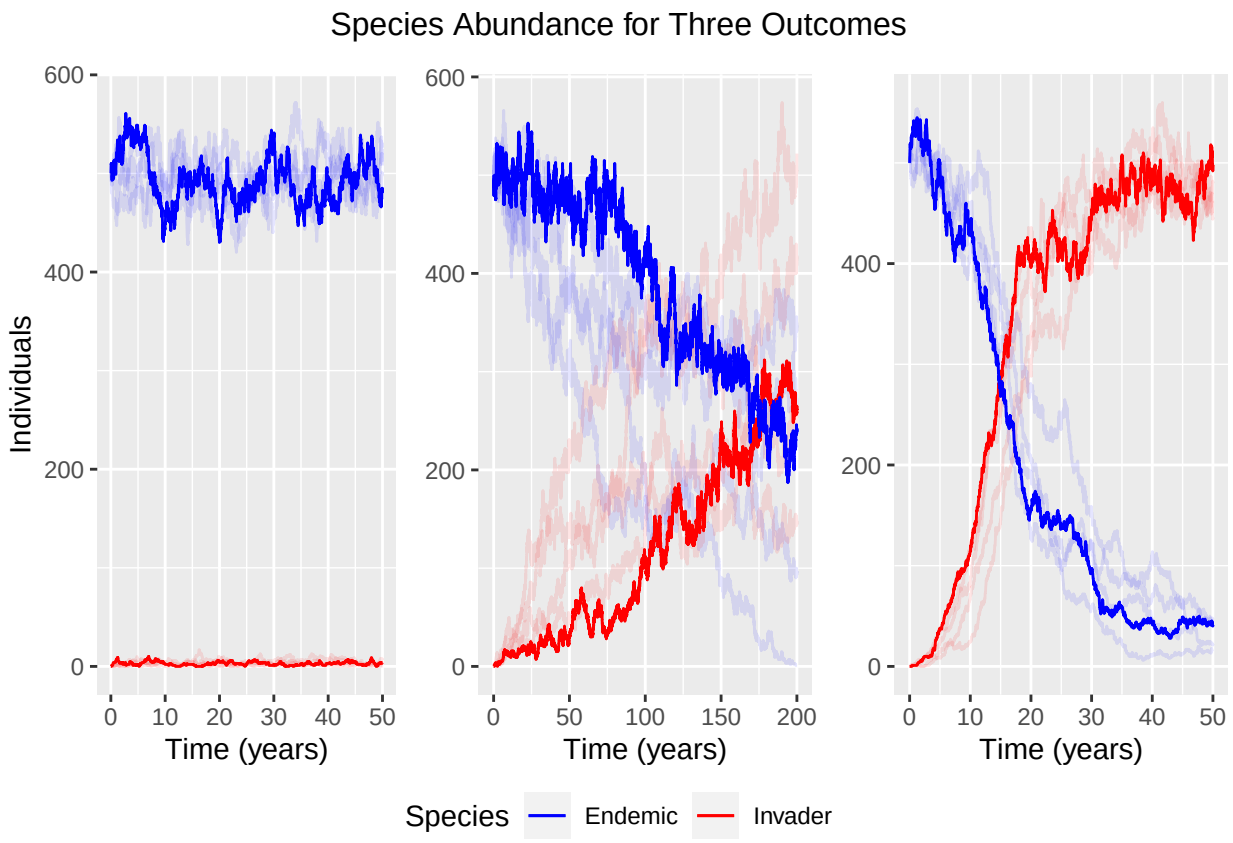
```
library(ggplot2)
library(grid)
library(gridExtra)
library(cowplot)
numRealizations=5
endemic_list <- replicate(numRealizations,
                          alphaSim(att.param, singleWindow.time.param, ratio.max=0.5, ttb=25, do.win=FAl
                          simplify=F)
coexistence_list <- replicate(numRealizations,
                              alphaSim(att.param, singleWindow.time.param, ratio.max=1, ttb=100, do.win=
                              simplify=F)
invader_list <- replicate(numRealizations,
                          alphaSim(att.param, singleWindow.time.param, ratio.max=2.0, ttb=25, do.win=FAl
                          simplify=F)
#initialize empty plots
endemicPlot = ggplot()
coexistencePlot = ggplot()
invaderPlot = ggplot()
for(i in 1:numRealizations) {
  #iterates through and adds the curve for endemic and invader 'numRealizations' times
  endemicPlot = endemicPlot + geom_line(data=endemic_list[[i]], aes(x=time, y=invader), color="red", alp
  coexistencePlot = coexistencePlot + geom_line(data=coexistence_list[[i]], aes(x=time, y=invader), colo
  invaderPlot = invaderPlot + geom_line(data=invader_list[[i]], aes(x=time, y=invader), color="red", alp
}
# add alpha=1.0 curves
```

```
endemicPlot = endemicPlot + geom_line(data=endemic_list[[1]], aes(x=time, y=invader), color="red") + geo
coexistencePlot = coexistencePlot + geom_line(data=coexistence_list[[1]], aes(x=time, y=invader), color=
invaderPlot = invaderPlot + geom_line(data=invader_list[[1]], aes(x=time, y=invader, color="Invader")) +

legend = get_legend(invaderPlot)
```

```
## 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()`).
## 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()`).
## 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()`).
```

```
grid.arrange(textGrob("Species Abundance for Three Outcomes", gp=gpar(size=32)),
             endemicPlot+ylab("Individuals"),
             coexistencePlot+theme(axis.title.y=element_blank()),
             invaderPlot+theme(axis.title.y=element_blank()) + theme(legend.position="none"),
             legend,
             ncol=3, nrow = 3,
             layout_matrix = rbind(c(1,1,1), c(2,3,4), c(5,5,5)),
             widths = c(13,13,13), heights = c(0.25, 2.5, 0.25)
             )
```

```
## 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()`).
## 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()`).
## 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()`).
## 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()`).
## 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()`).
## 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()`).
```



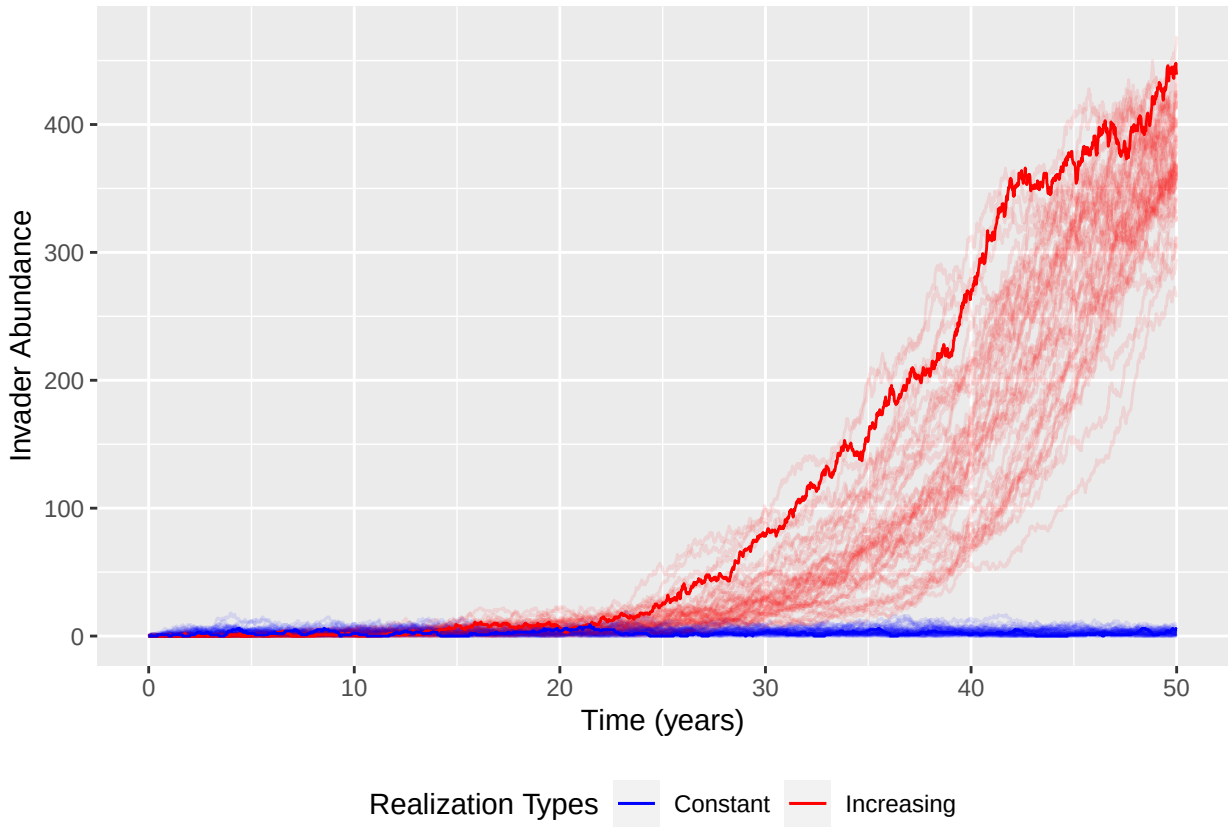
2. Two Types of Realizations

```
library(ggplot2)

numRealizations=50
inc_list = replicate(n=numRealizations,
                     expr=alphaSim(att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=2, ttb=25
                     simplify=F)
const_list = replicate(n=numRealizations,
                       expr=alphaSim(att.param, singleWindow.time.param, do.prob=TRUE, ratio.max=0.5, t
                       simplify=F)

plot = ggplot()
for(i in 1:numRealizations) {
  plot = plot + geom_line(data=inc_list[[i]], aes(x=time, y=invader), color="red", alpha=0.1)
  plot = plot + geom_line(data=const_list[[i]], aes(x=time, y=invader), color="blue", alpha=0.1)
}
plot = plot + geom_line(data=inc_list[[1]], aes(x=time, y=invader, color="Increasing")) + geom_line(data

plot + ylab("Invader Abundance") + xlab("Time (years)") + theme(legend.position="bottom")
```

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

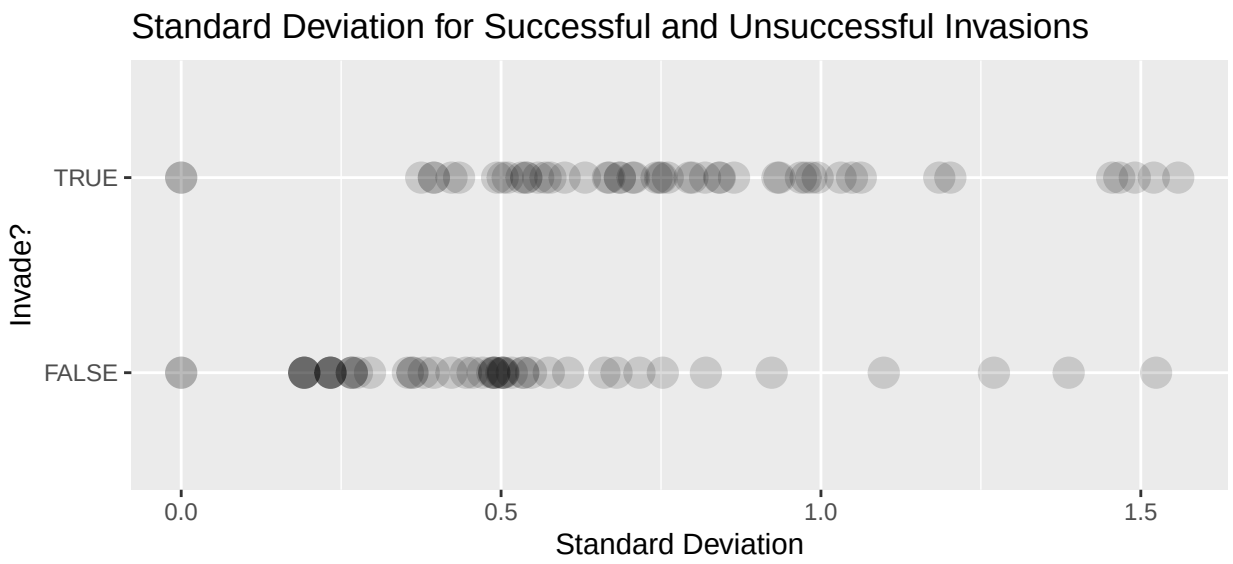
```
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## Removed 1 row containing missing values (‘geom_line()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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()‘).
## 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],at
colnames(endemic.data) = c('Initial N', 'Birth Rate', 'Carrying Cap.', 'Initial Inter.', 'Intra.', 'Imm
invader.data = data.frame(invader=c(att.param[2],att.param[4],att.param[6],att.param[8],att.param[10],a
colnames(invader.data) = c('Initial N', 'Birth Rate', 'Carrying Cap.', 'Initial Inter.', 'Intra.', 'Imm
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))
)
```



	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

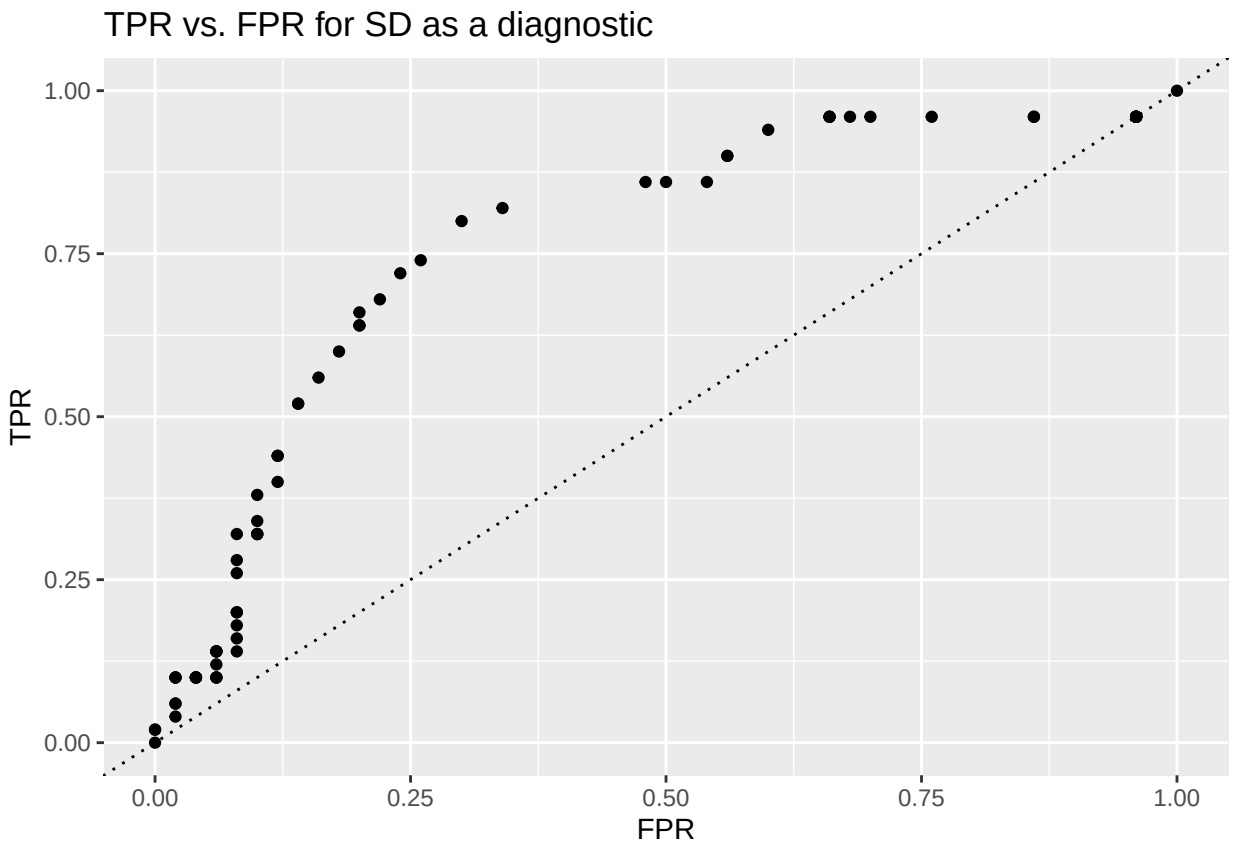
	Tau	Window Start	Window Length	Realizations
value	1 Week	10 Years	1 Year	50 each

ROC Curve for Single Stat

```
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)
```



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=
# constant ratio: a12 > a21
df.constant <- halfAndHalf(att.param, singleWindow.time.param, ttb=25, halfSimulations=100, 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

```
library(tidyverse)
```

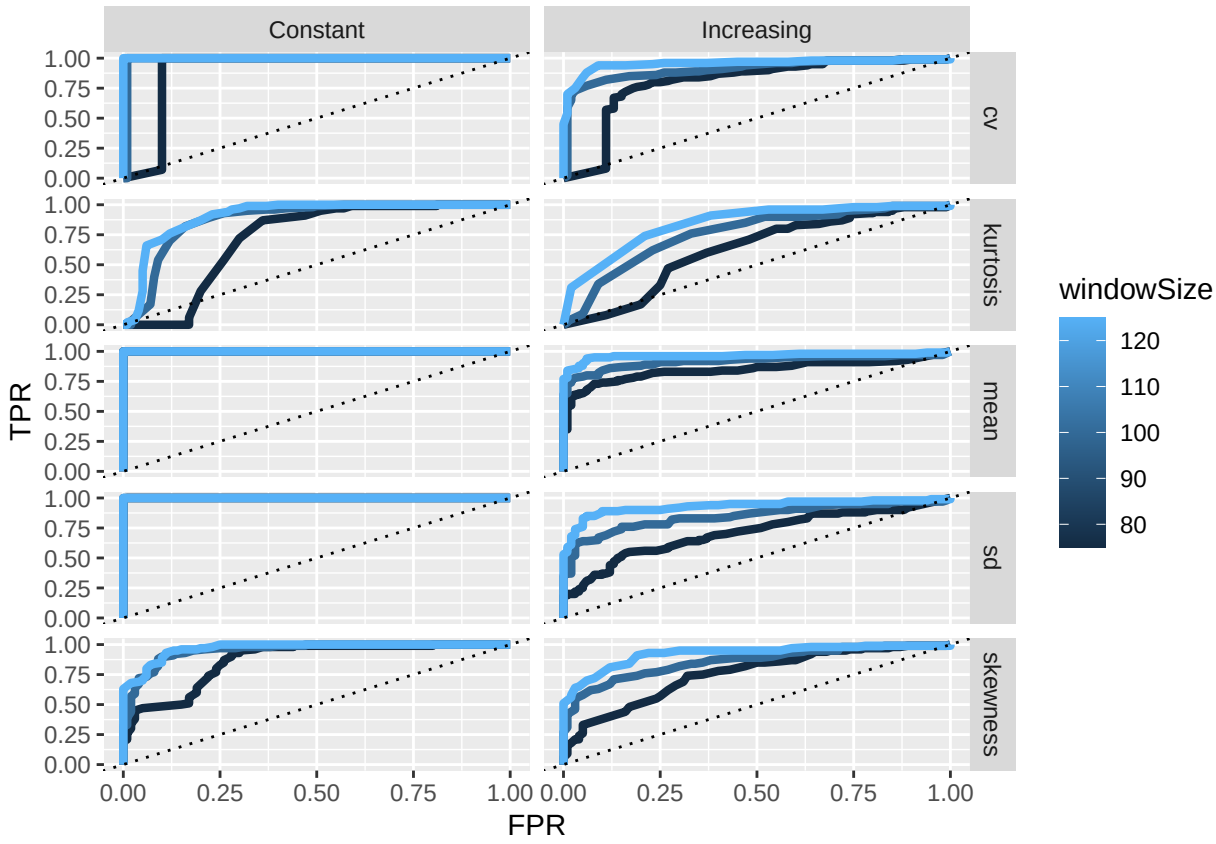
```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::combine()   masks gridExtra::combine()
## x dplyr::filter()    masks stats::filter()
## x dplyr::lag()       masks stats::lag()
## x 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, inequal
    }
  }
  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(nu
  colnames(roc.df) = columns
  return(roc.df)
}

roc.df <- ROCdata(statsList, numThresholds, stats, windows, inequalities)
```

```
#NOT FINISHED -- need to think about how to interface with facet
library(tidyverse)
ROCjStat <- function(roc.df, windows) {
  optimums = list()
  for(w in windows) {
    tempW = roc.df %>% filter(windowSize==m)
    for(i in 1:length(roc.data.list)) {
      temp = roc.data.list[[i]]
      diff = vector("numeric", length=length(temp[,1]))
      for(j in 1:length(diff)) {
        diff[j] = temp$TPR[j] - temp$FPR[j]
      }
      optimums[[i]] = c(temp$TPR[which.max(diff)], temp$FPR[which.max(diff)])
    }
  }
  return(optimums)
}
```

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), linewidth=1.5) +
  facet_grid(Statistic ~ realizationType) +
  geom_abline(slope=1, intercept=0, linetype=3)
```



7. AUC vs. TTB

```
library(DescTools)
AUCvsTTB <- function(ttbRange, numTTB, stats, inequalities, halfSimulations, numThresholds, windowSize)
  ttbSeq = seq(from=ttbRange[1], to=ttbRange[2], length=numTTB)
  columns = c("AUC", "TTB", "Statistic")
  df.auc = data.frame(matrix(nrow = 0, ncol = length(columns)))
  colnames(df.auc) = columns

  for(i in 1:length(stats)) {
    for(ttb in ttbSeq) {
      df.data = halfAndHalf(att.param, singleWindow.time.param, ttb, halfSimulations, win.ratio=2.0, fa

      df.stat = singleWindowStat(df.data, time.window=windowSize, time.index=ttb*52/2, func=stats[i])

      df.rates = ROCcurveData(df.stat, numThresholds, inequalities[i], stats[i])

      auc = AUC(x=df.rates$FPR, y=df.rates$TPR, method='trapezoid')
      bind = c(auc, ttb, stats[i])
      df.auc = rbind(df.auc, bind)
      colnames(df.auc) = columns
    }
  }
  return(df.auc)
}
```

```
#df.auc <- AUCvsTTB(ttbRange=c(10,25),
#                    numTTB=15,
#                    stats,
#                    inequalities,
#                    halfSimulations=10,
#                    numThresholds=10,
#                    windowSize=104)
```

```
#ggplot(data=df.auc, aes(x=TTB, y=AUC))+ geom_line(aes(group=Statistic)) + facet_grid(Statistic ~ .)
```