Analyze\_transmission

# Preamble

Title: *Script B for Objective 2: effects of seasonal network differences on pathogen transmission*  
Author: Marie Gilbertson  
Date: “08/30/2020”

**What this code does:**  
1. Analyzes differences between wet and dry season epidemics

#### Clear Environment ####  
remove(list=ls())  
  
#### Load libraries ####  
library(ggplot2)  
library(dplyr)  
library(stringr) # for "str\_split"  
library(viridis)  
library(ggpubr)  
  
#### Set seed ####  
set.seed(3268)

# Load data

Start by loading parameter sets and preparing to load data of interest.

# load parameter sets  
param.sets <- get(load("../Data/simulation\_paramsets.Rdata"))  
  
# set which parameter sets to analyze  
param.ids <- c(1:420)  
  
# empty dataframe to store results  
all.results <- NULL  
summ.stats <- data.frame(matrix(nrow = length(param.ids), ncol = 7))  
colnames(summ.stats) <- c("param.set", "m.dur.time", "m.total.i", "m.final.i",   
 "m.max.i", "m.max.it\_s", "prop.fail")

Next, loop through parameter sets of interest, loading those simulation results.

# loop through and load all results  
for(i in 1:length(param.ids)){  
 param.set.num <- param.ids[i]  
   
 full.name <- paste("../Output/Simulation\_Results/full set results\_paramset ", param.set.num, ".Rdata", sep = "")  
 set\_results <- get(load(full.name))  
   
 all.results <- rbind(all.results, set\_results)  
   
 #### store summary stats ####  
 summ.stats$param.set[i] <- set\_results$param.set[1]  
   
 # proportion failed epidemics; if only 1 individual is ever infected, consider it a "failed" epidemic  
 bad.draws <- sum(set\_results$num.failed)  
 only.1 <- nrow(set\_results[set\_results$total.i<=(1/33),])  
 # number of bad draws + number only infecting 1 / simulations + number of bad draws  
 summ.stats$prop.fail[i] <- (bad.draws + only.1)/(nrow(set\_results) + bad.draws)  
   
 ## now, of only the "successful" epidemics:  
 set\_results\_succ <- set\_results[set\_results$total.i>(1/33),]  
 # mean duration  
 summ.stats$m.dur.time[i] <- mean(set\_results\_succ$dur.time)  
 # mean total ever infected  
 summ.stats$m.total.i[i] <- mean(set\_results\_succ$total.i)  
 # mean final infected  
 summ.stats$m.final.i[i] <- mean(set\_results\_succ$final.i)  
 # mean max infected  
 summ.stats$m.max.i[i] <- mean(set\_results\_succ$max.i)  
 # mean start point of max infected  
 summ.stats$m.max.it\_s[i] <- mean(set\_results\_succ$max.it\_s)  
}

Update formatting, subset, and save for ease in plotting later.

# assign appropriate model type and gamma labeling for plotting  
param.sets$model.type <- NA  
param.sets$gamma <- NA  
for(i in 1:nrow(param.sets)){  
 temp.params <- param.sets[i,]  
 model.type\_gamma <- unlist(str\_split(temp.params$model.type\_gamma, "\_"))  
 param.sets$model.type[i] <- model.type\_gamma[1] # type of disease process; options are "SI", "SIR", or "SIS"  
   
 if(model.type\_gamma[1]=="SI"){  
 param.sets$gamma[i] <- 0  
 }else{  
 param.sets$gamma[i] <- as.numeric(model.type\_gamma[2])  
 }  
}  
  
param.sets$param.set <- paste("set\_", param.sets$set.num, sep = "")  
summ.stats <- left\_join(summ.stats, param.sets, by = "param.set")  
all.results <- left\_join(all.results, param.sets, by = "param.set")  
  
# split data and save as subsets of interest  
weight.data <- subset(summ.stats, summ.stats$weights==T)  
binary.data <- subset(summ.stats, summ.stats$weights==F)  
  
w.si <- subset(weight.data, weight.data$model.type=="SI")  
b.si <- subset(binary.data, binary.data$model.type=="SI")  
  
w.sis <- subset(weight.data, weight.data$model.type=="SIS")  
b.sis <- subset(binary.data, binary.data$model.type=="SIS")  
   
w.sir <- subset(weight.data, weight.data$model.type=="SIR")  
b.sir <- subset(binary.data, binary.data$model.type=="SIR")  
  
### save these datasets for making manuscript figures ###  
# save(w.si, file = "../Output/Manuscript\_Figures/data\_for\_plotting/w.si.Rdata")  
# save(b.si, file = "../Output/Manuscript\_Figures/data\_for\_plotting/b.si.Rdata")  
# save(w.sis, file = "../Output/Manuscript\_Figures/data\_for\_plotting/w.sis.Rdata")  
# save(b.sis, file = "../Output/Manuscript\_Figures/data\_for\_plotting/b.sis.Rdata")  
# save(w.sir, file = "../Output/Manuscript\_Figures/data\_for\_plotting/w.sir.Rdata")  
# save(b.sir, file = "../Output/Manuscript\_Figures/data\_for\_plotting/b.sir.Rdata")

# Automated generation of results heatmaps

Loop through all model types, generating results heatmaps for epidemic outcomes of interest.

model.types <- c("w.si", "b.si" , "w.sir", "b.sir", "w.sis", "b.sis")  
  
for(i in 1:length(model.types)){  
   
 temp.model <- model.types[i]  
 print(temp.model)  
   
   
 # read in data  
 name <- paste("../Output/Manuscript\_Figures/data\_for\_plotting/", temp.model, ".Rdata", sep = "")  
 temp.data <- get(load(name))  
   
 temp.data.d <- subset(temp.data, temp.data$season=="Dry")  
 temp.data.w <- subset(temp.data, temp.data$season=="Wet")  
 temp.data\_diff <- temp.data.d  
 temp.data\_diff[,c("m.dur.time", "m.total.i","prop.fail")] <- temp.data.d[,c("m.dur.time", "m.total.i","prop.fail")] - temp.data.w[,c("m.dur.time", "m.total.i","prop.fail")]  
   
   
   
 #### MEAN DURATION ####  
 if(temp.model %in% c("w.sir", "b.sir", "w.sis", "b.sis")){  
  
 g1\_st <- ggplot(temp.data, aes(x = c.rate, y = prob)) + geom\_tile(aes(fill = m.dur.time), colour = "white") +  
 scale\_fill\_viridis(discrete=F, option = "B", direction = 1, limits = c(1, 26)) +   
 ylab("Probability of transmission") +  
 xlab("Edge weight scaling")   
   
   
 gg1\_st <- g1\_st + facet\_grid(gamma ~ season) +  
 theme(text = element\_text(size=15)) +   
 theme(axis.text.x = element\_text(size = 10),  
 legend.position = "bottom") +  
 labs(fill = "Mean duration (weeks)")  
 gg1\_st  
   
 # differences  
 g2\_st <- ggplot(temp.data\_diff, aes(x = c.rate, y = prob)) + geom\_tile(aes(fill = m.dur.time), colour = "white") +  
 scale\_fill\_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0, breaks = c(-2, 0, 2, 4), limits = c(-2.6, 4.4)) +  
 ylab("Probability of transmission") +  
 xlab("Edge weight scaling")   
   
   
 gg2\_st <- g2\_st + facet\_grid(gamma~.) +  
 theme(text = element\_text(size=15)) +   
 theme(axis.text.x = element\_text(size = 10),  
 legend.position = "bottom") +  
 labs(fill = "Dry mean difference (weeks)")  
 gg2\_st  
   
   
 g <- ggarrange(gg1\_st, gg2\_st, labels = c("A", "B"), nrow = 1, widths = c(1.5, 1))  
 g  
   
 dur.name <- paste("../Output/Manuscript\_Figures/", temp.model,"\_duration\_full figure.jpeg", sep = "")  
 ggsave(g, file = dur.name, width = 11, height = 11, units = "in", dpi = 300)  
 }  
   
 #### MEAN TOTAL INFECTED ####  
 g1\_st <- ggplot(temp.data, aes(x = c.rate, y = prob)) + geom\_tile(aes(fill = m.total.i), colour = "white") +  
 scale\_fill\_viridis(discrete=F, option = "B", direction = 1, limits = c(0.06, 1)) +   
 ylab("Probability of transmission") +  
 xlab("Edge weight scaling")   
   
 gg1\_st <- g1\_st + facet\_grid(gamma ~ season) +  
 theme(text = element\_text(size=15)) +   
 theme(axis.text.x = element\_text(size = 10),  
 legend.position = "bottom", legend.key.width = unit(1, "cm")) +  
 labs(fill = "Mean total proportion infected")  
 gg1\_st  
   
   
 g2\_st <- ggplot(temp.data\_diff, aes(x = c.rate, y = prob)) + geom\_tile(aes(fill = m.total.i), colour = "white") +  
 scale\_fill\_gradient2(low = "blue", mid = "white", high = "red", midpoint = 0, breaks = c(-0.08, 0, 0.08, 0.16), limits = c(-0.08, 0.2)) +  
 ylab("Probability of transmission") +  
 xlab("Edge weight scaling")   
   
 gg2\_st <- g2\_st + facet\_grid(gamma~.) +  
 theme(text = element\_text(size=15)) +   
 theme(axis.text.x = element\_text(size = 10),  
 legend.position = "bottom", legend.key.width = unit(1, "cm")) +  
 labs(fill = "Dry mean difference")  
 gg2\_st  
   
 g <- ggarrange(gg1\_st, gg2\_st, labels = c("A", "B"), nrow = 1, widths = c(1.5, 1))  
 g  
   
   
 if(temp.model %in% c("w.si", "b.si")){  
 total.name <- paste("../Output/Manuscript\_Figures/", temp.model,"\_total infected\_full figure.jpeg", sep = "")  
 ggsave(g, file = total.name, width = 11, height = 5, units = "in", dpi = 300)  
 }else{  
 total.name <- paste("../Output/Manuscript\_Figures/", temp.model,"\_total infected\_full figure.jpeg", sep = "")  
 ggsave(g, file = total.name, width = 11, height = 11, units = "in", dpi = 300)  
 }  
   
   
 #### MEAN PROPORTION FAILED ####  
   
 g1\_st <- ggplot(temp.data, aes(x = c.rate, y = prob)) + geom\_tile(aes(fill = prop.fail), colour = "white") +  
 scale\_fill\_viridis(discrete=F, option = "B", direction = -1, limits = c(0, 1)) +   
 ylab("Probability of transmission") +  
 xlab("Edge weight scaling")   
   
 gg1\_st <- g1\_st + facet\_grid(gamma ~ season) +  
 theme(text = element\_text(size=15)) +   
 theme(axis.text.x = element\_text(size = 10),  
 legend.position = "bottom", legend.key.width = unit(1, "cm")) +  
 labs(fill = "Proportion failed")  
 gg1\_st  
   
   
   
   
 g2\_st <- ggplot(temp.data\_diff, aes(x = c.rate, y = prob)) + geom\_tile(aes(fill = prop.fail), colour = "white") +  
 scale\_fill\_gradient2(low = "red", mid = "white", high = "blue", midpoint = 0, breaks = c(-0.2, -0.1, 0, 0.1), limits = c(-0.22, 0.12)) +  
 ylab("Probability of transmission") +  
 xlab("Edge weight scaling")   
   
 gg2\_st <- g2\_st + facet\_grid(gamma~.) +  
 theme(text = element\_text(size=15)) +   
 theme(axis.text.x = element\_text(size = 10),  
 legend.position = "bottom") +   
 labs(fill = "Dry difference")  
 gg2\_st  
   
   
 g <- ggarrange(gg1\_st, gg2\_st, labels = c("A", "B"), nrow = 1, widths = c(1.5, 1))  
 g  
   
 if(temp.model %in% c("w.si", "b.si")){  
 fail.name <- paste("../Output/Manuscript\_Figures/", temp.model,"\_prop failed\_full figure.jpeg", sep = "")  
 ggsave(g, file = fail.name, width = 11, height = 5, units = "in", dpi = 300)  
 }else{  
 fail.name <- paste("../Output/Manuscript\_Figures/", temp.model,"\_prop failed\_full figure.jpeg", sep = "")  
 ggsave(g, file = fail.name, width = 11, height = 11, units = "in", dpi = 300)  
 }  
   
}

## [1] "w.si"

## [1] "b.si"

## [1] "w.sir"

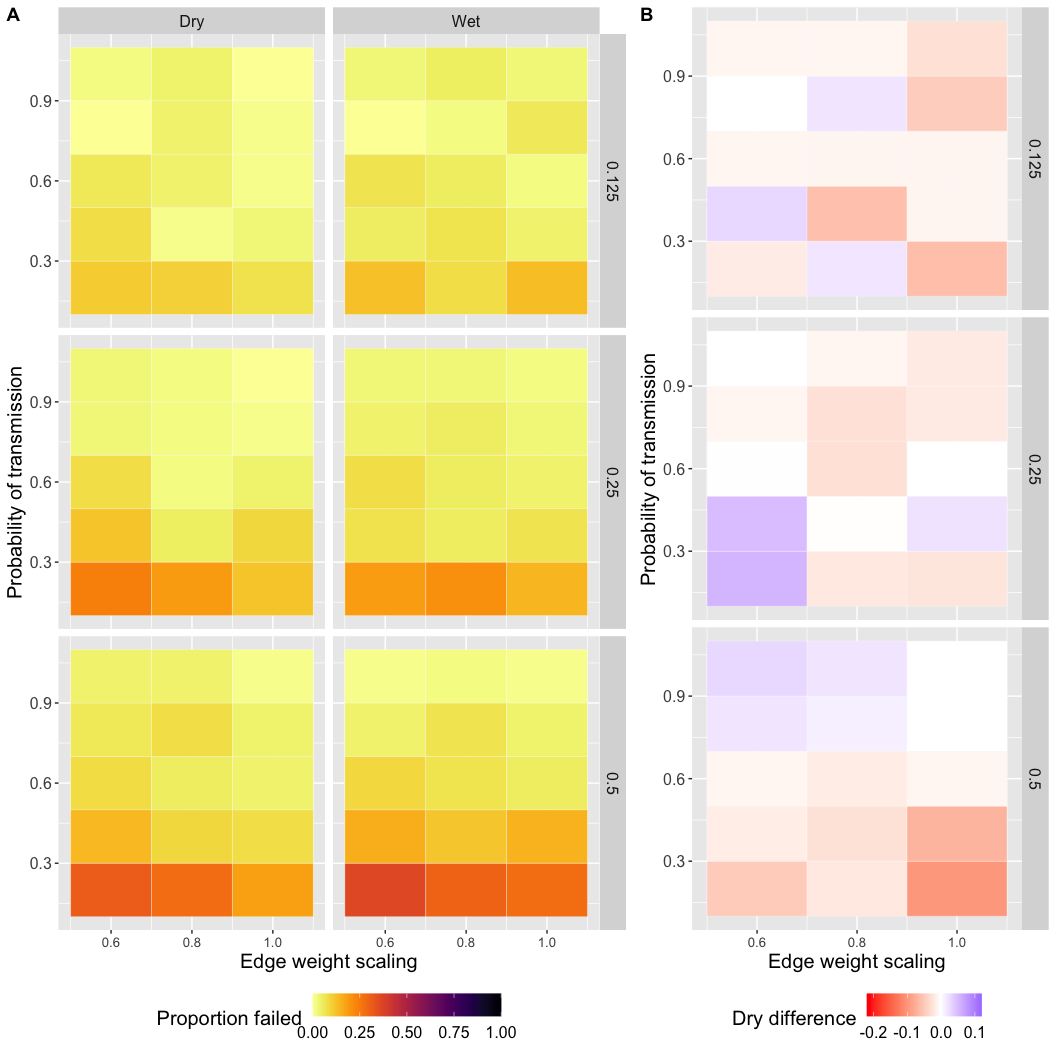
## [1] "b.sir"

## [1] "w.sis"

## [1] "b.sis"

For illustration purposes, view example output.

print(g)



For reproducibility, view session info.

sessionInfo()

## R version 3.6.3 (2020-02-29)  
## Platform: x86\_64-apple-darwin15.6.0 (64-bit)  
## Running under: macOS 10.16  
##   
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib  
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib  
##   
## locale:  
## [1] en\_US.UTF-8/en\_US.UTF-8/en\_US.UTF-8/C/en\_US.UTF-8/en\_US.UTF-8  
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## other attached packages:  
## [1] ggpubr\_0.4.0 viridis\_0.5.1 viridisLite\_0.3.0 stringr\_1.4.0   
## [5] dplyr\_1.0.4 ggplot2\_3.3.3   
##   
## loaded via a namespace (and not attached):  
## [1] tidyselect\_1.1.0 xfun\_0.24 purrr\_0.3.4 haven\_2.2.0   
## [5] carData\_3.0-4 colorspace\_2.0-0 vctrs\_0.3.6 generics\_0.1.0   
## [9] htmltools\_0.5.1.1 yaml\_2.2.1 rlang\_0.4.10 pillar\_1.4.7   
## [13] foreign\_0.8-76 glue\_1.4.2 withr\_2.4.1 DBI\_1.1.1   
## [17] readxl\_1.3.1 lifecycle\_1.0.0 munsell\_0.5.0 ggsignif\_0.6.0   
## [21] gtable\_0.3.0 cellranger\_1.1.0 zip\_2.1.0 evaluate\_0.14   
## [25] labeling\_0.4.2 knitr\_1.28 rio\_0.5.16 forcats\_0.5.0   
## [29] curl\_4.3 broom\_0.7.2 Rcpp\_1.0.6 scales\_1.1.1   
## [33] backports\_1.1.6 abind\_1.4-5 farver\_2.0.3 gridExtra\_2.3   
## [37] hms\_1.0.0 digest\_0.6.27 stringi\_1.5.3 openxlsx\_4.1.5   
## [41] rstatix\_0.6.0 cowplot\_1.1.0 grid\_3.6.3 tools\_3.6.3   
## [45] magrittr\_2.0.1 tibble\_3.0.6 crayon\_1.4.1 tidyr\_1.0.2   
## [49] car\_3.0-9 pkgconfig\_2.0.3 ellipsis\_0.3.1 data.table\_1.12.8  
## [53] rmarkdown\_2.9 R6\_2.5.0 compiler\_3.6.3