Title of article: The loss of foundation species revisited

Author's names: Allyson L. Degrassi[1,†], Steven Brantley[2], Carrie R. Levine[3], Robert J. Miller[4], Jacqueline Mohan[5], Sydne Record[6], and Aaron M. Ellison[7]

Author's affiliation addresses:

[1] Department of Biology, University of Vermont, 120 Marsh Life Science, Burlington VT 05468 USA

[2] Joseph W. Jones Ecological Research Center, 3988 Jones Center Drive, Newton, GA 39870 USA

[3] Department of Environmental Science, Policy, and Management, University of California, Berkeley, 130 Mulford Hall, Berkeley, CA 94720 USA

[4] Marine Science Institute, University of California Santa Barbara, Santa Barbara CA 93109-6150 USA

[5] Odum School of Ecology, University of Georgia Athens, GA 30601 USA

[6] Department of Biology, Bryn Mawr College, Park 209 Bryn Mawr, PA 19010 USA

[7] Harvard University, Harvard Forest, 324 North Main Street, Petersham, MA 01366 USA

Corresponding author's e-mail address:

† E-mail: adegrass@uvm.edu

Journal Submission: PeerJ

Date: 2015

The raw data are available from the Harvard Forest Data Archive, file HF-259 http://harvardforest.fas.harvard.edu/data-archive).

```
############################################################

Foundation Species Citation Review
Product of LTER Working Group 2012
 Collaborative project with:
 Allyson Degrassi <adegrass@uvm.edu>,
Steven Brantley   <sbrantle@umn.edu>,
Robert Miller <miller@msi.ucsb.edu>,
Carrie R Levine <crlevine@berkeley.edu>,
Sydne Record <sydne.record@gmail.com>,
Jacqueline Mohan <jmohan@uga.edu>,
Aaron Ellison <aellison@fas.harvard.edu>

Date: 30 October 2014 - 30 June 2015
Primary: A. Degrassi


############################################################

 MASTER SCRIPT

For RStudio

library(plyr) # to merge data frame

FSMeta <- read.csv("FSMeta_HF Archive_v2.csv")


####################Find out literature types: Reviews, Commentary, Letters, ETC##############

FSNonPrimary <-aggregate(FSMeta$LiteratureType, by=list(FSMeta$LiteratureType), FUN=length)

names(FSNonPrimary)[names(FSNonPrimary)=="Group.1"] <- "Literature Type"

names(FSNonPrimary)[names(FSNonPrimary)=="x"] <- "Studies"

FSNonPrimary
```

```
##################### Select primary articles only ######################################
FSMetaPrimary <- subset(FSMeta, LiteratureType == "Primary")
nrow(FSMetaPrimary) #should be 331

##################### Find out definitions from FSDefined ######################
FSDefine <-aggregate(FSMetaPrimary$FSDefined, by=list(FSMetaPrimary$FSDefined), FUN=length)
names(FSDefine)[names(FSDefine)=="Group.1"] <- "Definition"
names(FSDefine)[names(FSDefine)=="x"] <- "Studies"

chisq.test(FSDefine$Studies)

names(FSDefine)[names(FSDefine)=="V3"] <- "Percent"
FSDefine$Percent <- FSDefine$Studies/sum(FSDefine$Studies)
FSDefine

##################### Find out FS Claim ######################
FSClaim <-aggregate(FSMetaPrimary$FSClaim, by=list(FSMetaPrimary$FSClaim), FUN=length)
names(FSClaim)[names(FSClaim)=="Group.1"] <- "FSClaim"
names(FSClaim)[names(FSClaim)=="x"] <- "Studies"
chisq.test(FSClaim$Studies)

names(FSClaim)[names(FSClaim)=="V3"] <- "Percent"
FSClaim$Percent <- FSClaim$Studies/sum(FSDefine$Studies)
FSClaim
```

```
################## Find out strength of Influence ##############

Strong <- subset(FSMetaPrimary, Strong == 1)

Moderate <- subset(FSMetaPrimary, Moderate == 1)

Marginal <- subset(FSMetaPrimary, Marginal == 1)

Strong <-aggregate(Strong$Strong, by=list(Strong$Strong), FUN=length)

Moderate <-aggregate(Moderate$Moderate, by=list(Moderate$Moderate), FUN=length)

Marginal <-aggregate(Marginal$Marginal, by=list(Marginal$Marginal), FUN=length)

Strong[,1] = "Strong"

Moderate[,1] = "Moderate"

Marginal [,1] = "Marginal"

Influence <- join_all(list(Strong, Moderate, Marginal), by = 'Group.1', type = 'full')

names(Influence)[names(Influence)=="Group.1"] <- "Strength"

names(Influence)[names(Influence)=="x"] <- "Studies"

Influence

chisq.test(Influence$Studies)


names(Influence)[names(Influence)=="V3"] <- "Percent"

Influence$Percent <- Influence$Studies/sum(Influence$Studies)

Influence


# Filter for primary papers and papers that claimed to study foundation species

FSMetaPrimaryFoundation <- subset(FSMetaPrimary, FSClaim == "Foundation Species")

nrow(FSMetaPrimaryFoundation)


################ Find out FS Role ############################
```

```
FSRole <-aggregate(FSMetaPrimaryFoundation$FSRole, by=list(FSMetaPrimaryFoundation$FSRole), FUN=length)

names(FSRole)[names(FSRole)=="Group.1"] <- "Role"

names(FSRole)[names(FSRole)=="x"] <- "Studies"


chisq.test(FSRole$Studies)

names(FSRole)[names(FSRole)=="V3"] <- "Percent"

FSRole$Percent <- FSRole$Studies/sum(FSRole$Studies)

FSRole



################## Find out Threat to FS  ####################

Climate <- subset(FSMetaPrimaryFoundation, ClimateChange == 1)

InvasiveSpecies <- subset(FSMetaPrimaryFoundation, InvasiveSpp == 1)

HabitatDeg <- subset(FSMetaPrimaryFoundation, HabitatDegradation == 1)

Exploitation <- subset(FSMetaPrimaryFoundation, Exploitation == 1)

DiseasePathogen <- subset(FSMetaPrimaryFoundation, DiseasePathogen == 1)

NoThreat<- subset(FSMetaPrimaryFoundation, NoThreat == 1)

Climate <-aggregate(Climate$ClimateChange, by=list(Climate$ClimateChange), FUN=length)

InvasiveSpecies <-aggregate(InvasiveSpecies$InvasiveSpp, by=list(InvasiveSpecies$InvasiveSpp), FUN=length)

HabitatDeg <-aggregate(HabitatDeg$HabitatDegradation, by=list(HabitatDeg$HabitatDegradation), FUN=length)

Exploitation <-aggregate(Exploitation$Exploitation, by=list(Exploitation$Exploitation), FUN=length)

DiseasePathogen <-aggregate(DiseasePathogen$DiseasePathogen, by=list(DiseasePathogen$DiseasePathogen), FUN=length)

NoThreat <-aggregate(NoThreat$NoThreat, by=list(NoThreat$NoThreat), FUN=length)
```

```
Climate[,1] = "Climate Change"

InvasiveSpecies[,1] = "Invasive Species"

HabitatDeg [,1] = "Habitat Degradation"

Exploitation [,1] = "Exploitation"

DiseasePathogen [,1] = "Disease or Pathogen"

NoThreat [,1] = "No Threat"

Threat <- join_all(list(Climate, InvasiveSpecies, HabitatDeg, Exploitation, DiseasePathogen, NoThreat), by = 'Group.1', type = 'full')

names(Threat)[names(Threat)=="Group.1"] <- "Threat"

names(Threat)[names(Threat)=="x"] <- "Studies"


chisq.test(Threat$Studies)

names(Threat)[names(Threat)=="V3"] <- "Percent"

Threat$Percent <- Threat$Studies/sum(Threat$Studies)

Threat


#################### MAP ################################

source("C:\\Users\\Ally\\Documents\\UVM\\Projects\\FS Meta Analysis\\40-Software\\R Source Scripts\\projectFSfunctions_FSMap.R")


library(maps)

library(plyr)

library(mapproj)

library(rworldmap)

library(plotrix)
```

```
# Select studies that did claim the organism was a FS

FSClaimYes <-subset(FSMeta, FSClaim == "Foundation Species")

ClaimYesCount <-nrow(FSClaimYes) # is the number of studies that did claim FS

ClaimYesCount


ClaimYesMap <- subset(FSClaimYes, CountryID1 != "North and South America, Europe, Asia and New Zealand") # n = 1

ClaimYesMap <- subset(ClaimYesMap, CountryID1 != "global") # n = 1

ClaimYesMap <- subset(ClaimYesMap, CountryID1 != "Most of Europe") # n = 1


YesClaimMapAttributes1 <-aggregate(ClaimYesMap$CountryID1, by=list(ClaimYesMap$CountryID1), FUN=length)

YesClaimMapAttributes2 <-aggregate(ClaimYesMap$CountryID2, by=list(ClaimYesMap$CountryID2), FUN=length)


# Join the two data frames together


YesClaimMapAttributes <- join_all(list(YesClaimMapAttributes1, YesClaimMapAttributes2), by = 'Group.1', type = 'full')



names(YesClaimMapAttributes)[names(YesClaimMapAttributes)=="Group.1"] <- "Country"

names(YesClaimMapAttributes)[names(YesClaimMapAttributes)=="x"] <- "FSYesStudies"


###### select studies that did NOT claim the organism was a FS

FSClaimNo <-subset(FSMeta, FSClaim == "Not Foundation Species")
```

```
ClaimNoCount <-nrow(FSClaimNo) # number of studies that did not claim FS

ClaimNoMap <- subset(FSClaimNo, CountryID1 != "global" & CountryID1 != "Most of Europe" & CountryID1 != "NA")


NoClaimMapAttributes1 <-aggregate(ClaimNoMap$CountryID1, by=list(ClaimNoMap$CountryID1), FUN=length)

NoClaimMapAttributes2 <-aggregate(ClaimNoMap$CountryID2, by=list(ClaimNoMap$CountryID2), FUN=length)


# Join the two data frames together


NoClaimMapAttributes <- join_all(list(NoClaimMapAttributes1, NoClaimMapAttributes2), by = 'Group.1', type = 'full')

names(NoClaimMapAttributes)[names(NoClaimMapAttributes)=="Group.1"] <- "Country"

names(NoClaimMapAttributes)[names(NoClaimMapAttributes)=="x"] <- "FSNoStudies"

ClaimMapAttributes <- join_all(list(YesClaimMapAttributes, NoClaimMapAttributes), by = 'Country', type = 'full')

# replace NA's with 0's

ClaimMapAttributes[is.na(ClaimMapAttributes)] <- 0

#blank map dataset

blankmap <- getMap(resolution = "low")


mapAttributes <- data.frame(blankmap$NAME, blankmap$LON, blankmap$LAT)

names(mapAttributes) <- c("Country", "Lon", "Lat")

ClaimMapAttributes <- join_all(list(ClaimMapAttributes, mapAttributes), by = 'Country', type = 'left')


FSYesColor <- "limegreen"
```

```
FSNoColor <- "blue"

TextSize <- 1.45


###############

png(filename="FSMap11.png",

   type="cairo",

   units="in",

   width=15,

   height=10,

   pointsize=12,

   res=300)

#################


plot(blankmap)

plotPieCharts(ClaimMapAttributes, 2, 10, FSYesColor, FSNoColor)


legend(x = -30, y = -30,

      legend= c("FS Studied", "FS Not Studied"),

      cex = TextSize,

      col = c("white", "white"),

      pt.cex = cex,

      box.lty = 0

)


######
```

```
floating.pie(-25, -38, legendSlice, radius = 3, col = FSYesColor)

floating.pie(-25, -46, legendSlice, radius = 3, col = FSNoColor)

#########

legend(x = -180, y = 35,

      legend= c("# Studies"),

      cex = TextSize,

      pt.cex = cex,

      box.lty = 0

)



##########################

legendSlice <- c(1)

legendColor <- "grey51"

legendLat <- -152


floating.pie(legendLat, -65, legendSlice, radius = 10, col = legendColor)

legend(legendLat, -57,legend= "223", cex = TextSize, pt.cex = cex, box.lty = 0, bg = "transparent")

floating.pie(legendLat, -43, legendSlice, radius = 8, col = legendColor)

legend(legendLat, -35,legend= "13", cex = TextSize, pt.cex = cex, box.lty = 0, bg = "transparent")

floating.pie(legendLat, -25, legendSlice, radius = 7, col = legendColor)

legend(legendLat+2, -17,legend= "8", cex = TextSize, pt.cex = cex, box.lty = 0, bg = "transparent")

floating.pie(legendLat, -8, legendSlice, radius = 6, col = legendColor)

legend(legendLat+2, 1.5,legend= "3", cex = TextSize, pt.cex = cex, box.lty = 0, bg = "transparent")

floating.pie(legendLat, 5, legendSlice, radius = 4, col = legendColor)
```

```
legend(legendLat+1.5, 14,legend= "2", cex = TextSize, pt.cex = cex, box.lty = 0, bg = "transparent")

floating.pie(legendLat, 15, legendSlice, radius = 2, col = legendColor)

legend(legendLat, 25,legend= "1", cex = TextSize, pt.cex = cex, box.lty = 0, bg = "transparent")


########

dev.off()
```