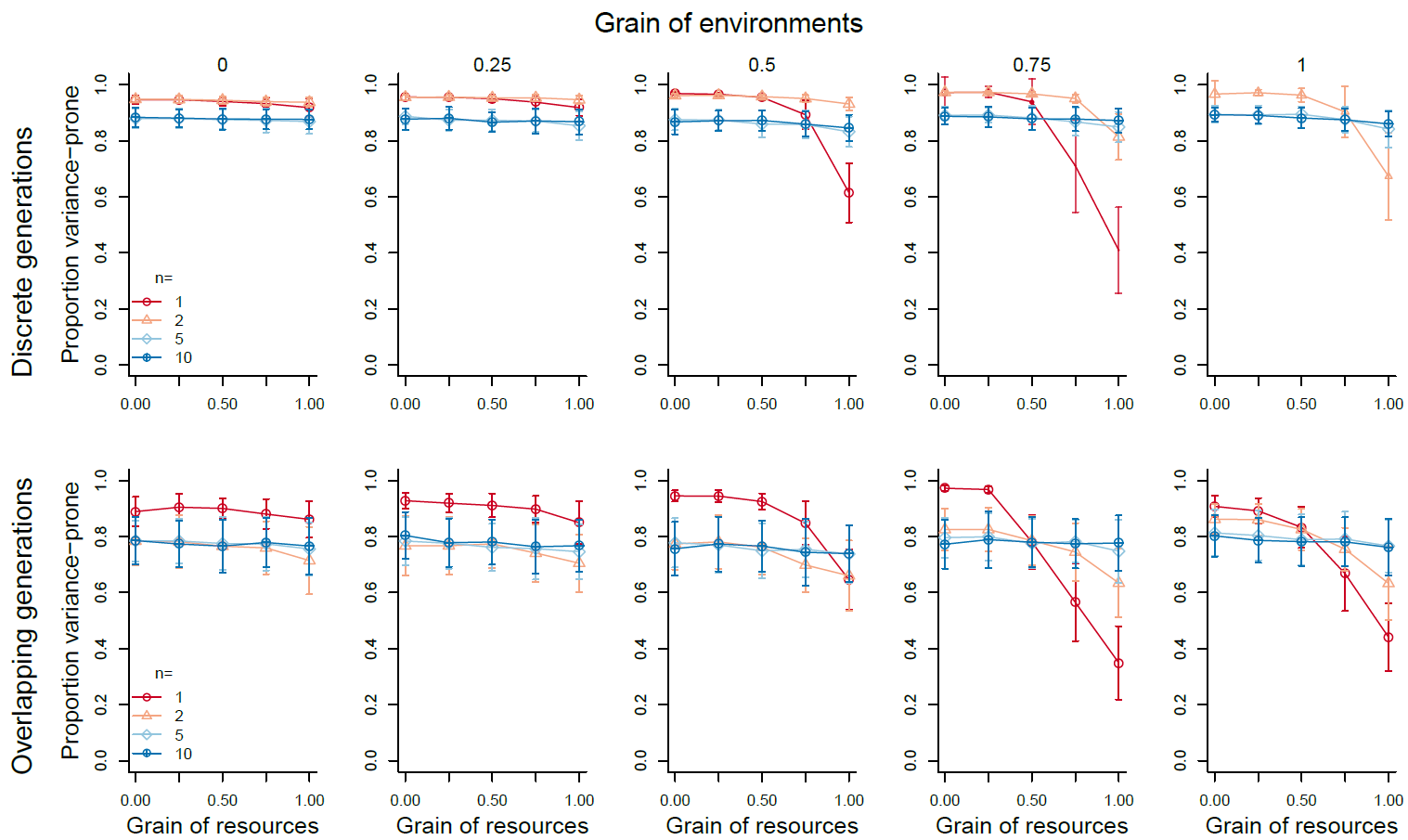
Electronic Supplementary Material (ESM)

to

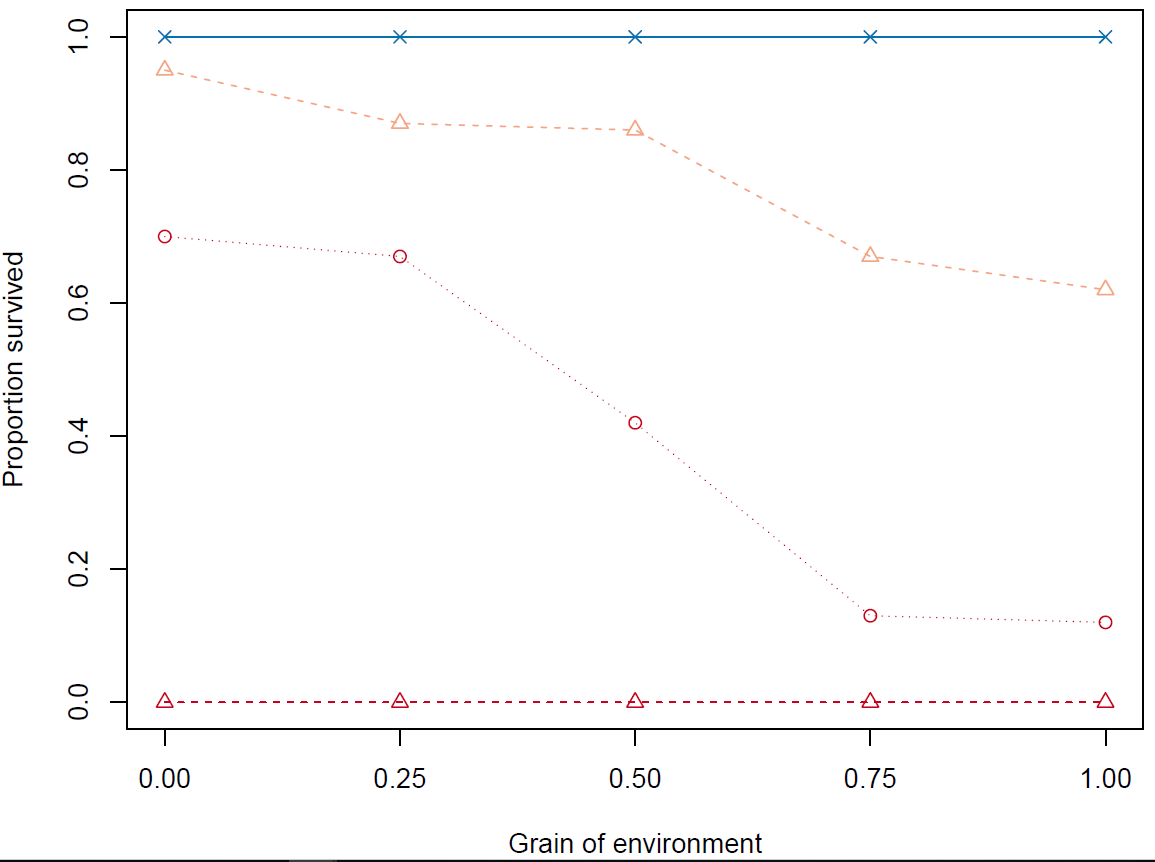
# Bet-hedging across generations can affect the evolution of variance-sensitive strategies within generations

Haaland, Thomas R., Jonathan Wright, Irja I. Ratikainen

## Supplementary figures



**Figure S1:** Mean evolved gene values for the proportion of the population playing variance-prone strategies at the end of the simulations for discrete (top row, between-year mortality *α*=1) and overlapping (right, *α*=0.5) generations, for different grain of environment (*ge*, x-axes), grain of resources (*gr*, given by the number 0 to 1 above each column) and number of decision events prior to reproduction (colours, point types). Points indicate means and error bars indicate standard deviations across 100 replicate populations, and relative point size represents the proportion of populations surviving until the end of the simulation.



**Figure S2:** Proportion of populations that survived until the end of the simulations in Model 2 with discrete generations, for different environmental grains (*ge*). Circles/dotted line: Resource grain *gr* = 0.75; triangles/dashed lines: *gr* = 1. Colors represent number of decision events per lifetime *n*, red: *n* = 1; orange: *n* = 2. Populations at all other parameter combinations (*n* and *gr*) had 100 % survival for all *ge*, here represented by blue crosses/solid line. All populations with overlapping generations, and all populations in Model 1 had 100 % survival as well.

## Supplementary table

**Table S1.** List of mathematical notation and parameter values

|  |  |  |
| --- | --- | --- |
| Parameter or variable | Description | Range or values |
| Model 1 | | |
| *R* | Overall state of resource availability (determining whether the variable strategy gives good or bad payoffs) | {0, 1} |
| *gr* | Grain of resources | {0, 0.25, 0.5, 0.75, 1} |
| *ri* | Local resource availability for individual *i* | {0, 1} |
| *µ* | Mean payoff of the variable strategy | 2 (baseline) |
| *a* | Proportional reduction in payoff for the safe strategy relative to variable strategy | [0, 1), set to 0.1 in figure 2 and 3. |
| *b* | Proportional variability in payoff for the variable strategy | (0, 1], set to 0.9 in figure 2 and 3. |
| Model 2 | | |
| *E* | Overall state of environmental quality (determining energetic state) | [0, 1] |
| *ge* | Grain of environment | {0, 0.25, 0.5, 0.75, 1} |
| *xi* | Energetic state of individual *i* | [0, 1] |
| Simulation parameters | | |
| *n* | Number of decision events prior to reproduction | {1, 2, 5, 10} |
| *α* | Between-season mortality | Set to 1 or 0.5 in fig. 2 and S1, and 0.5 in fig. 3. |
| *K* | Carrying capacity | 5000 |
| *mp* | Mutation rate | Baseline 0.005 |
| *mσ* | Mutational size | Baseline 0.05 |

## R code

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

## VARIANCE-SENSITIVITY & BET-HEDGING ##

## HAALAND, WRIGHT & RATIKAINEN, 2019 ##

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

## Preamble

rm(list=ls())

library(dplyr)

library(cowplot)

library(scatterpie)

library(grid)

library(RColorBrewer)

theme\_set(theme\_cowplot())

# Set your directory: use find-and-replace-all on 'MyDirectory'

## Utility function - sigmoid relationship between energetic state and fitness

Utility <- function(x,r) r/(1+exp(-5\*(x-0.5))) #x is state; r is upper asymptote.

## sim(): Individual-based simulation.

## Arguments:

## n: number of time steps/instances before reproduction, over which to gather resources.

## alpha: Between-year mortality

## m.rate: Mutation rate. Default 0.005

## m.size: Mutation size. Default 0.05

## K: Carrying capacity. Default 5000. Density dependence is 'ceiling' type.

## T: Number of years/seasons.

## reps: Number of replicate simulations. Default 1

## plot: Logical. Whether to plot population trajectories, or not (default not).

## Variant:1 (default): Resources gained relate linearly to fitness.

## 2: Energetic state random, but risky foraging can change it. Sigmoid relation with fitness.

## grain\_res: 0: No correlation among patch quality. 1: All patches have same quality. [0,1]

## grain\_env: Only implemented in variant 2. Correlation among individual states in a time step. [0,1]

## mu: Average reproductive rate

## a: Penalty of choosing constant patch (only implemented in Variant 1). Default 0.1.

## b: Variability of variable patch. Default 0.1. a and b represent proportions of mu. [0,1]

## Title: Plot title. Default blank

sim <- function(n,alpha=0.2,m.rate=0.005,m.size=0.05,K=5000,T=2000,reps=1,plot=FALSE,Variant=1,

grain\_res=0,grain\_env=0.1,mu=2,b=0.1,a=0.1,title=""){

if(plot){

plot(1:T,rep(1,T),type="n",ylim=c(0,1),ylab="Proportion variance-prone",xlab="Year",main=title)

}

if(Variant==1){

w.s <- c(mu\*(1-b),mu\*(1+b)) # Payoffs of variable patch, c(bad,good)

w.n <- mu\*(1-a) # Payoff of constant patch (non-sensitive)

}else if(Variant==2){

w.s <- c(-0.1,0.1) # Change in state at variable patch, c(bad,good)

w.n <- 0 # Change in state at constant patch

}

nstorage <- zstorage <- matrix(NA,reps,T) # Storage matrices for population sizes and gene values

for(rep in 1:reps){

#Initiate population

N <- K #Starting pop. size is at carrying capacity

pop <- runif(N) #Initiate genes for probability of playing risky strategy.

for(Time in 1:T){

#Record population traits

zstorage[rep,Time] <- mean(as.numeric(pop)) #Record phenotype

nstorage[rep,Time] <- N #Record population size.

strategy <- ifelse(pop>runif(N),TRUE,FALSE) #Strategy played. TRUE=Risky, FALSE=Safe

if(Variant==1){

W <- rep(0,N)

for(Step in 1:n){

this.W <- rep(0,N)

cond <- ifelse(runif(1)<0.5,TRUE,FALSE) #Good (TRUE) or bad (FALSE) resource conditions?

if(grain\_res<1){

conds <- sample(c(cond,!cond),size=N,prob=c(0.5+grain\_res/2,0.5-grain\_res/2),replace=TRUE)

this.W[which(strategy==TRUE & conds==cond)] <- w.s[as.numeric(cond)+1]

this.W[which(strategy==TRUE & conds!=cond)] <- w.s[as.numeric(!cond)+1]

this.W[which(strategy==FALSE)] <- w.n #Payoff for playing safe strategy doesn't depend on cond.

} else{

this.W[which(strategy==TRUE)] <- w.s[as.numeric(cond)+1]

this.W[which(strategy==FALSE)] <- w.n

}

W <- W+this.W

}

}

else if(Variant==2){

W <- rep(0,N)

for(Step in 1:n){

this.W <- rep(0,N)

cond <- runif(1) # Mean environmental condition

lower <- grain\_env\*cond

upper <- cond + (1-grain\_env)\*(1-cond)

states <- runif(N,lower,upper) # 'Energetic states' of individuals depend on env.conditions, but can differ more or less depending on grain\_env

gamblers <- which(strategy==TRUE & states<0.5) # Only risky individuals in the accel. part of the curve gamble.

env <- ifelse(runif(1)<0.5,1,2) #Resource quality, bad (1) or good (2)

if(grain\_res<1){

this.W[gamblers] <- sample(c(w.s[env],w.s[which(w.s!=w.s[env])]),size=length(gamblers),

prob=c(0.5+grain\_res/2,0.5-grain\_res/2),replace=TRUE)

} else{

this.W[gamblers] <- w.s[env]

}

#Risk averse or non-sensitive have unchanged states.

W <- W+Utility(states+this.W,2\*mu) #Fitness payoffs are determined by sigmoid utility function

}

}

if(sum(W)==0){

print(c("Extinct at time",Time))

break

}

#Selection

if(alpha==1){ # If discrete generations (all adults die)

NextN <- ifelse(sum(W)>K,K,round(sum(W)/n))

pop <- sample(pop,size=NextN,prob=W,replace=TRUE)

#Mutation

mut <- which(runif(NextN)<m.rate) #Select mutated individuals

pop[mut] <- rnorm(length(mut),pop[mut],m.size)

}

else{ # If overlapping generations (some adults survive)

Alive <- runif(N)>alpha

Alive <- which(Alive==TRUE) # Alive is a sequence of numbers of the individuals who survived.

#How many 'slots' should be filled (i.e. how many offspring should be produced?)

Offspring <- min(sum(W),K-length(Alive))

#Reproduction. IDs of those who get offspring. If mortality happens before breeding:

IDs <- sample(Alive,size=Offspring,prob=W[Alive],replace=TRUE)

##If mortality happens after breeding (hashtag away if applying):

#IDs <- sample(1:N,size=Offspring,prob=W,replace=TRUE)

#Next generation is a combination of new and old individuals.

pop <- c(pop[IDs],pop[Alive])

#Mutation: Some of the offspring mutate

mut <- which(runif(Offspring)<m.rate) #Select mutated individuals

pop[mut] <- rnorm(length(mut),pop[mut],m.size)

NextN <- length(pop)

}

pop[pop<0] <- 0 # Make sure noone has negative phenotypic values

pop[pop>1] <- 1

#Update population size

N <- NextN

}

print(rep)

if(plot){

lines(1:T,zstorage[rep,],col=rgb(1,0,0,0.1))

}

}

return(list(zstorage,nstorage))

}

#Running the simulation (example)

test <- sim(n=1,alpha=1,reps=20,m.rate=0.1,plot=TRUE,Variant=1,grain\_res=0.2,grain\_env=1,b=0.9,a=0.1,mu=2)

for(i in 1:20){lines(1:2000,test[[2]][i,1:2000]/5000,col=rgb(0,0,1,0.2))}

## TimePlots: Plots all evolutionary trajectories. Useful for investigating what's going on

TimePlot <- function(Variant,n,alpha,grain\_res,grain\_env,subfolder=NULL,axis=FALSE,title="",popsize=FALSE){

tmp <- readRDS(file=switch(Variant,

"1"=paste0("MyDirectory\\",subfolder,ifelse(is.null(subfolder),"","\\"),"Variant=1, alpha=",alpha,", n=",n,

", grain\_res=",grain\_res,", z.Rdata"),

"2"=paste0("MyDirectory\\",subfolder,ifelse(is.null(subfolder),"","\\"),"Variant 2, alpha=",alpha,", n=",n,

", grain\_res=",grain\_res,", grain\_env=",grain\_env,", z.Rdata")))

T <- dim(tmp)[2]

runs <- sample(2:dim(tmp)[1],dim(tmp)[1]/2)

plot(1:T,tmp[1,],type="l",col=rgb(1,0,0,0.1),ylab="",ylim=c(0,1),xlab="",main=as.expression(title))

if(popsize){

tmp2 <- readRDS(file=switch(Variant,

"1"=paste0("MyDirectory\\",subfolder,ifelse(is.null(subfolder),"","\\"),"Variant=1, alpha=",alpha,", n=",n,

", grain\_res=",grain\_res,", n.Rdata"),

"2"=paste0("MyDirectory\\",subfolder,ifelse(is.null(subfolder),"","\\"),"Variant 2, alpha=",alpha,", n=",n,

", grain\_res=",grain\_res,", grain\_env=",grain\_env,", n.Rdata")))

tmp2 <- tmp2/max(tmp2,na.rm=TRUE)

lines(1:T,tmp2[1,],col=rgb(0,0,1,0.1))

}

if(axis=="Both"||axis=="Side"){

mtext("Proportion risk-prone", side=2, line=2.5,cex=0.9)

}

if(axis=="Both"||axis=="Bottom"){

mtext("Year",side=1, line=2.5,cex=0.9)

}

for(i in runs){

if(popsize){

lines(1:T,tmp2[i,],col=rgb(0,0,1,0.1))

}

lines(1:T,tmp[i,],col=rgb(1,0,0,0.1))

}

}

par(mfrow=c(3,2),mar=c(4,4,0.5,0.5))

## Run simulations and save data

# Paper presents results for alpha=0.5 and 1, vary other parameters at your leisure

#Model 1

for(i in c(0,0.25,0.5,0.75,1)){ # For different resource grains

for(j in c(1,2,5,10)){ # For different number of decision events

tmp <- sim(n=j,alpha=1,reps=100,T=2000,plot=TRUE,Variant=1,grain\_res=i,title=i)

saveRDS(tmp[[1]],file=paste0("MyDirectory\\Variant 1, alpha=1, n=",j,", grain\_res=",i,", z.Rdata"))

saveRDS(tmp[[2]],file=paste0("MyDirectory\\Variant 1, alpha=1, n=",j,", grain\_res=",i,", n.Rdata"))

}

}

#Model 2

for(i in c(0,0.25,0.5,0.75,1)){ # For different environmental grains

for(j in c(0,0.25,0.5,0.75,1)){ # For different resource grains

for(k in c(1,2,5,10)){ #For different number of decision events

tmp <- sim(n=k,grain\_env=i,alpha=0.5,reps=100,T=2000,plot=TRUE,Variant=2,grain\_res=j,title=j,a=0.1,b=0.9,mu=2)

saveRDS(tmp[[1]],file=paste0("MyDirectory\\Variant 2, alpha=0.5, n=",k,", grain\_res=",j,", grain\_env=",i,", z.Rdata"))

saveRDS(tmp[[2]],file=paste0("MyDirectory\\Variant 2, alpha=0.5, n=",k,", grain\_res=",j,", grain\_env=",i,", n.Rdata"))

print(k)

}

}

}

## Plotting results (outcomes at the end of the simulations) on a continuous axis.

Continuous <- function(Variant,grain\_env=NULL,alpha,subfolder=NULL,axis=FALSE,title="",legend=TRUE){

if(Variant==2){

mydata <- readRDS(file=paste0("MyDirectory\\",subfolder,ifelse(is.null(subfolder),"","\\"),"Variant=2, alpha=",alpha,", grain\_env=",grain\_env,", PieSummary continuous.Rdata"))

} else if(Variant==1){

mydata <- readRDS(file=paste0("MyDirectory\\",subfolder,ifelse(is.null(subfolder),"","\\"),"Variant=1, alpha=",alpha,", PieSummary continuous.Rdata"))

}

grains <- unique(mydata$Grain\_res)

ns <- unique(mydata$n)

pal <- brewer.pal(length(ns),"RdBu")

plot(grains,grains,ylim=c(0,1),type="n",xlab="",ylab="",xaxt="n",main=as.expression(title),bty="L")

axis(side=1,at=grains)

if(axis=="Both"||axis=="Side"){

mtext("Proportion variance-prone", side=2, line=2.5,cex=1)

}

if(axis=="Both"||axis=="Bottom"){

mtext("Grain of resources",side=1, line=2.5,cex=1)

}

for(i in ns){

points(grains,mydata$Mean[mydata$n==i],pch=i,col=pal[which(ns==i)],cex=1.2\*mydata$Survived[mydata$n==i])

lines(grains,mydata$Mean[mydata$n==i],col=pal[which(ns==i)])

for(j in grains){ #Error bars

y <- mydata$Mean[mydata$n==i & mydata$Grain\_res==j]

arrows(j,y,j,y+mydata$Sd[mydata$n==i & mydata$Grain\_res==j],angle=90,col=alpha(pal[which(ns==i)],mydata$Survived[mydata$n==i]),length=0.02)

arrows(j,y,j,y-mydata$Sd[mydata$n==i & mydata$Grain\_res==j],angle=90,col=alpha(pal[which(ns==i)],mydata$Survived[mydata$n==i]),length=0.02)

}

}

if(legend){legend("bottomleft",legend=ns,col=pal,pch=ns,lty=1,title="n=",bty="n")}

return(mydata)

}

setwd("MyDirectory\\")

pdf("Variant 2 continuous.pdf",width=10,height=6)

par(mfrow=c(2,5),mar=c(4,4,1,0.4),oma=c(0,3,3,0.4))

Continuous(Variant=2,grain\_env=0,alpha=1,axis="Side",title=0)

for(i in c(0.25,0.5,0.75,1)){

Continuous(Variant=2,grain\_env=i,alpha=1,axis=FALSE,legend=FALSE,title=i)

}

Continuous(Variant=2,grain\_env=0,alpha=0.5,axis="Both")

for(i in c(0.25,0.5,0.75,1)){

Continuous(Variant=2,grain\_env=i,alpha=0.5,axis="Bottom",legend=FALSE)

}

mtext("Grain of environments",side=3,outer=TRUE,cex=1.2,line=1)

mtext("Overlapping generations Discrete generations", side=2,outer=TRUE,cex=1.2,line=1)

dev.off()

pdf("Variant 1 continuous.pdf",width=8,height=4)

par(mfrow=c(1,2),mar=c(4,4,2,1))

Continuous(Variant=1,grain\_res=1,subfolder="a0.1b0.9mu2",alpha=1,title="Discrete generations",axis="Both")

Continuous(Variant=1,grain\_res=1,subfolder="a0.1b0.9mu2",alpha=0.5,title="Overlapping generations",axis="Bottom")

dev.off()

#### Geometric mean calculations for Model 1 ####

# Calculate according to equation 1 and inequality 2

g.mean.calc <- function(mu=1,a=0.1,b=0.1,n=1,type="gm\_var"){

store <- numeric(n+1)

for(m in 0:n){

store[m+1] <- ((mu\*(1-b))\*(n-m)+(mu\*(1+b))\*m)^dbinom(m,n,0.5)

}

gm\_var <- prod(store)

gm\_const <- n\*mu\*(1-a)

return(switch(type,"gm\_var"=gm\_var, #Geometric mean of variable patch

"gm\_const"=gm\_const, #Geometric mean of constant patch

"cond\_a"=1-(gm\_var/(n\*mu)), #Condition for a below which the constant patch is better

"cond\_b"=gm\_const/(n\*mu))) #Condition for b above which variable patch is better

}

### Heatmap:

ns <- 1:8

bs <- (1:10)/10

store <- matrix(0,length(ns),length(bs))

for(i in 1:8){

for(j in 1:10){

store[i,j] <- g.mean.calc(mu=2,b=bs[j],n=i,type="cond\_a")

}

}

setwd("MyDirectory\\")

pdf("Predictions figure.pdf",height=5.5,width=10)

par(mfrow=c(1,2),mar=c(4,4,2,0.1))

breaks <- c(seq(0,0.01,by=0.001),seq(0.02,1,by=0.02))

image(x=ns,y=bs,z=store,breaks=breaks,col=grey.colors(60,start=0.05,end=1),bty="L",

xlab=expression("Number of decision events prior to reproduction, "\*italic(n)),

ylab=expression("Magnitude of variation at variable patch, "\*italic(b)),xaxt="n",yaxt="n")

axis(side=1,at=1:8)

axis(side=2,at=1:10/10)

labs <- c(0.005,0.01,0.025,0.05,0.1,0.25,0.5)

contour(ns,bs,store,add=TRUE,levels=labs,col="White")

#Scale bar

test <- matrix(60,10,60)

test[1,] <- 1:60

image(x=1:10,y=1:60,z=test,col=grey.colors(60,start=0.05,end=1),xlab="",ylab="",xaxt="n",yaxt="n",bty="n")

labs <- c(0.002,0.02,breaks[21],0.5,0.9)

text(x=rep(3,length(labs)),y=which(breaks %in% labs),labs)

dev.off()