**Supporting information S1**

*R* code for Model 3.

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

# Function to maximise using MLE

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

# p = starting values of three parameters p[1], p[2], p[3]

# index = interval cover value for year site combination

# n.y = number if years

######

loglike10 <-function(p, index, n.y) {

sd <- p[2]

loglike.res<-rep(0, length(n.y))

for (j in 1:length(n.y)){

mu <- p[3]\*n.y[j] + p[1]

indexdf<-index

boundl<-rep(0,length(indexdf[,j]))

boundr<-rep(0,length(indexdf[,j]))

left<-rep(0,length(indexdf[,j]))

right<-rep(1,length(indexdf[,j]))

# Specify current boundaries for each interval class

boundl[indexdf[,j]==0]<-logit(0.0001)

boundr[indexdf[,j]==0]<-logit(0.001)

boundl[indexdf[,j]==1]<-logit(0.001)

boundr[indexdf[,j]==1]<-logit(0.01)

boundl[indexdf[,j]==2]<-logit(0.01)

boundr[indexdf[,j]==2]<-logit(0.03)

boundl[indexdf[,j]==3]<-logit(0.03)

boundr[indexdf[,j]==3]<-logit(0.05)

boundl[indexdf[,j]==4]<-logit(0.05)

boundr[indexdf[,j]==4]<-logit(0.10)

boundl[indexdf[,j]==5]<-logit(0.10)

boundr[indexdf[,j]==5]<-logit(0.25)

boundl[indexdf[,j]==6]<-logit(0.25)

boundr[indexdf[,j]==6]<-logit(0.33)

boundl[indexdf[,j]==7]<-logit(0.33)

boundr[indexdf[,j]==7]<-logit(0.50)

boundl[indexdf[,j]==8]<-logit(0.50)

boundr[indexdf[,j]==8]<-logit(0.75)

boundl[indexdf[,j]==9]<-logit(0.75)

boundr[indexdf[,j]==9]<-logit(0.95)

boundl[indexdf[,j]==10]<-logit(0.95)

boundr[indexdf[,j]==10]<-logit(0.99)

left<-pnorm(boundl, mu, sd)

right<-pnorm(boundr, mu, sd)

lik<-right-left

negloglik <- (-2\*sum(log(lik[!is.na(indexdf[,j])])))

loglike.res[j] <- negloglik

}

return(sum(loglike.res))

}

**Supporting information S2**

JAGS code for Model 4.

######

# isCensored[r] indexes which observations are censored

# y[r] is NA when the observation is censored

# censorLimitVec[r,] provides the lower and upper bounds for censored observation r

######

model {

for( r in 1 : Ndata ) {

isCensored[r] ~ dinterval( y[r] , censorLimitVec[r,] )

y[r] ~ dnorm( mu[r] , tau )

mu[r] <- b0[site[r]] + b1[site[r]] \* x[r]

}

for ( s in 1 : Nsite ) {

b0[s] ~ dnorm( mu0G , tau0G )

b1[s] ~ dnorm( mu1G , tau1G )

}

p.decrease <- 1 - step(mu1G)

tau ~ dgamma(0.1 , 0.1)

mu0G ~ dnorm(0 , 0.01)

tau0G ~ dgamma(0.1 , 0.1)

mu1G ~ dnorm(0 , 0.01)

tau1G ~ dgamma(0.1 , 0.1)

}