**SUPPLEMENTARY MATERIAL: R-functions for Bayesian estimation of the Gaussian and Weibull Survival Models.**

**1. Gaussian**

### GAUSSIAN MODEL ESTIMATION

# by Moltchanova et al

# the data frame dt has been pre-formatted to contain the following variables

# dish = dish ID number

# t.start = start of the observation period

# t.end = end of the observation period

# x = the number of seeds that have germinated during the observational period

# wp = Water Potential

# temp = temperature

# the parameters theta, Tb, psib50, sdpsib, k and Td are

# the HTT model parameters

# the HTT model inside the distribution function

inside.fun <- function(theta,Tb,psib50,sdpsib,t,wp,temp,k,Td){

 psib50.coef <- psib50+k\*(temp-Td)\*(temp>=Td)

 fg <- (wp-theta/((temp-Tb)\*t)-psib50.coef)/sdpsib

return(fg)

}

# the gaussian probit function

g.gau.full <- function(theta,Tb,psib50,sdpsib,t,wp,temp,k,Td){

 psib50.coef <- psib50+k\*(temp-Td)\*(temp>=Td)

 g <- pnorm((wp-theta/((temp-Tb)\*t)-psib50.coef)/sdpsib)

return(g)

}

# the survival likelihood based on the gaussian probit function

surv.lik.gau.fun <- function(theta,Tb,psib50,sdpsib,k,dt,Td){

g.end <- g.gau.full(theta,Tb,psib50[dt$dish],

 sdpsib,t=dt$t.end,wp=dt$wp,temp=dt$temp,k,Td)

g.end[dt$t.end==Inf] <- 1

gr <- g.end-g.gau.full(theta,Tb,psib50[dt$dish],

 sdpsib,t=dt$t.start,wp=dt$wp,temp=dt$temp,k,Td)

gr[gr==0] <- 0.00000001

loglik <- sum(dt$x\*log(gr))

return(loglik)

}

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

### MH - ALGORITHM ###

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

### initial values

theta <- 10.7

Tb <- 3.2

mupsib50 <- -.36

taupsib50 <- 1

psib50 <- rep(mupsib50,Ndish)

sdpsib <- 0.25

k <- 0.026

Td <- 18.6

### monitors

ITER <- 10^6

mon.theta <- mon.Tb <- mon.mupsib50 <- mon.taupsib50 <-

mon.sdpsib <- mon.k <- mon.Td <- numeric(ITER)

mon.psib50 <- array(dim=c(Ndish,ITER/10))

# for DIC:

mon.psib50mean <- numeric(Ndish)

mon.loglik <- numeric(ITER)

system.time({

for(iter in 1:ITER){print(iter)

# sampling theta

theta.new <- rnorm(1,theta,.05)

logR <- surv.lik.gau.fun(theta.new,Tb,psib50,sdpsib,k,dt=dt.surv,Td)-

 surv.lik.gau.fun(theta ,Tb,psib50,sdpsib,k,dt=dt.surv,Td)+

 dnorm(theta.new,0,100,log=T)-dnorm(theta,0,100,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){theta <- theta.new}

# sampling Tb

Tb.new <- rnorm(1,Tb,.025)

logR <- surv.lik.gau.fun(theta,Tb.new,psib50,sdpsib,k,dt=dt.surv,Td)-

 surv.lik.gau.fun(theta,Tb ,psib50,sdpsib,k,dt=dt.surv,Td)+

 dnorm(Tb.new,0,100,log=T)-dnorm(Tb,0,100,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){Tb <- Tb.new}

# sampling mupsib50 (Gibbs)

mupsib50 <- rnorm(1,sum(psib50)\*taupsib50/(Ndish\*taupsib50+10^(-4)),1/sqrt(Ndish\*taupsib50+10^(-4)))

# sampling taupsib50 (Gibbs)

taupsib50 <- rgamma(1,0.01+Ndish/2,0.01+.5\*sum((psib50-mupsib50)^2))

# sampling psib50

psib50.new <- rnorm(Ndish,psib50,.0025)

logR <- surv.lik.gau.fun(theta,Tb,psib50.new,sdpsib,k,dt=dt.surv,Td)-

 surv.lik.gau.fun(theta,Tb,psib50 ,sdpsib,k,dt=dt.surv,Td)+

 sum(dnorm(psib50.new,mupsib50,1/sqrt(taupsib50),log=T))-

 sum(dnorm(psib50 ,mupsib50,1/sqrt(taupsib50),log=T))

logU <- log(runif(1,0,1))

if(logU<logR){psib50 <- psib50.new}

# sampling sdpsib

sdpsib.new <- exp(rnorm(1,log(sdpsib),.025))

logR <- surv.lik.gau.fun(theta,Tb,psib50,sdpsib.new,k,dt=dt.surv,Td)-

 surv.lik.gau.fun(theta,Tb,psib50,sdpsib ,k,dt=dt.surv,Td)+

 dnorm(log(sdpsib.new),0,100,log=T)-dnorm(log(sdpsib),0,100,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){sdpsib <- sdpsib.new}

# sampling k

k.new <- rnorm(1,k,.0025)

logR <- surv.lik.gau.fun(theta,Tb,psib50,sdpsib,k.new,dt=dt.surv,Td)-

 surv.lik.gau.fun(theta,Tb,psib50,sdpsib,k ,dt=dt.surv,Td)+

 dnorm(k.new,0,10,log=T)-dnorm(k,0,10,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){k <- k.new}

# sampling Td

Td.new <- rnorm(1,Td,.025)

logR <- surv.lik.gau.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td.new)-

 surv.lik.gau.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td )+

 dnorm(Td.new,20,5,log=T)-dnorm(Td,20,5,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){Td <- Td.new}

# updating the monitors

mon.theta[iter] <- theta

mon.Tb[iter] <- Tb

mon.mupsib50[iter] <- mupsib50

mon.taupsib50[iter] <- taupsib50

mon.sdpsib[iter] <- sdpsib

mon.k[iter] <- k

mon.Td[iter] <- Td

if(iter%%10==0){

mon.psib50[,iter/10] <- psib50}

# for DIC

if(iter > .5\*ITER){mon.psib50mean <- mon.psib50mean+psib50/(.5\*ITER)}

mon.loglik[iter] <- surv.lik.gau.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td)

} # end of iterations

}) # end of system time

# the monitors can then be used for posterior inference

# evaluating the DIC

(mean.log <- mean(mon.loglik[seq(.5\*ITER+1,ITER,length.out=1001)]))

log.mean <- surv.lik.gau.fun(

 mean(mon.theta[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mean(mon.Tb[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mon.psib50mean,

 mean(mon.sdpsib[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mean(mon.k[seq(.5\*ITER+1,ITER,length.out=1001)]),dt=dt.surv,

 mean(mon.Td[seq(.5\*ITER+1,ITER,length.out=1001)]))

(DIC <- -2\*(2\*mean.log-log.mean))

**2. Weibull**

### GAUSSIAN MODEL ESTIMATION

# by Moltchanova et al

# the data frame dt has been pre-formatted to contain the following variables

# dish = dish ID number

# t.start = start of the observation period

# t.end = end of the observation period

# x = the number of seeds that have germinated during the observational period

# wp = Water Potential

# temp = temperature

# the parameters theta, Tb, psib50, sdpsib, k and Td are

# the HTT model parameters

# the HTT model inside the distribution function

inside.fun <- function(theta,Tb,psib50,sdpsib,t,wp,temp,k,Td){

 psib50.coef <- psib50+k\*(temp-Td)\*(temp>=Td)

 fg <- (wp-theta/((temp-Tb)\*t)-psib50.coef)/sdpsib

return(fg)

}

# the weibull cdf function

g.wei.full <- function(theta,Tb,psib50,sdpsib,t,wp,temp,k,Td,nu){

 psib50.coef <- psib50+k\*(temp-Td)\*(temp>=Td)

 g <- pweibull((wp-theta/((temp-Tb)\*t)-psib50.coef)/sdpsib,

 shape=nu)

return(g)

}

# the survival likelihood based on the weibull cdf function

surv.lik.wei.fun <- function(theta,Tb,psib50,sdpsib,k,dt,Td,nu){

g.end <- g.wei.full(theta,Tb,psib50[dt$dish],

 sdpsib,t=dt$t.end,wp=dt$wp,temp=dt$temp,k,Td,nu)

g.end[dt$t.end==Inf] <- 1

gr <- g.end-g.wei.full(theta,Tb,psib50[dt$dish],

 sdpsib,t=dt$t.start,wp=dt$wp,temp=dt$temp,k,Td,nu)

gr[gr==0] <- 0.00000001

loglik <- sum(dt$x\*log(gr))

return(loglik)

}

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

### MH - ALGORITHM ###

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

### initial values

theta <- 11

Tb <- 1.5

mupsib50 <- -1.15

taupsib50 <- 1

psib50 <- rep(mupsib50,Ndish)

sdpsib <- 0.6

k <- 0.026

Td <- 25.6

nu <- 1.5

### monitors

ITER <- 10^6

mon.theta <- mon.Tb <- mon.mupsib50 <- mon.taupsib50 <-

mon.sdpsib <- mon.k <- mon.Td <- mon.nu <- numeric(ITER)

#mon.psib50 <- array(dim=c(Ndish,ITER))

# for DIC:

mon.psib50mean <- numeric(Ndish)

mon.loglik <- numeric(ITER)

system.time({

for(iter in 1:ITER){print(iter)

# sampling theta

theta.new <- rnorm(1,theta,.05)

logR <- surv.lik.wei.fun(theta.new,Tb,psib50,sdpsib,k,dt=dt.surv,Td,nu)-

 surv.lik.wei.fun(theta ,Tb,psib50,sdpsib,k,dt=dt.surv,Td,nu)+

 dnorm(theta.new,0,100,log=T)-dnorm(theta,0,100,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){theta <- theta.new}

# sampling Tb

Tb.new <- rnorm(1,Tb,.025)

logR <- surv.lik.wei.fun(theta,Tb.new,psib50,sdpsib,k,dt=dt.surv,Td,nu)-

 surv.lik.wei.fun(theta,Tb ,psib50,sdpsib,k,dt=dt.surv,Td,nu)+

 dnorm(Tb.new,0,100,log=T)-dnorm(Tb,0,100,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){Tb <- Tb.new}

# sampling mupsib50 (Gibbs)

mupsib50 <- rnorm(1,sum(psib50)\*taupsib50/(Ndish\*taupsib50+10^(-4)),1/sqrt(Ndish\*taupsib50+10^(-4)))

# sampling taupsib50 (Gibbs)

taupsib50 <- rgamma(1,0.01+Ndish/2,0.01+.5\*sum((psib50-mupsib50)^2))

# sampling psib50

psib50.new <- rnorm(Ndish,psib50,.0025)

logR <- surv.lik.wei.fun(theta,Tb,psib50.new,sdpsib,k,dt=dt.surv,Td,nu)-

 surv.lik.wei.fun(theta,Tb,psib50 ,sdpsib,k,dt=dt.surv,Td,nu)+

 sum(dnorm(psib50.new,mupsib50,1/sqrt(taupsib50),log=T))-

 sum(dnorm(psib50 ,mupsib50,1/sqrt(taupsib50),log=T))

logU <- log(runif(1,0,1))

if(logU<logR){psib50 <- psib50.new}

# sampling sdpsib

sdpsib.new <- exp(rnorm(1,log(sdpsib),.025))

logR <- surv.lik.wei.fun(theta,Tb,psib50,sdpsib.new,k,dt=dt.surv,Td,nu)-

 surv.lik.wei.fun(theta,Tb,psib50,sdpsib ,k,dt=dt.surv,Td,nu)+

 dnorm(log(sdpsib.new),0,100,log=T)-dnorm(log(sdpsib),0,100,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){sdpsib <- sdpsib.new}

# sampling k

k.new <- rnorm(1,k,.0025)

logR <- surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k.new,dt=dt.surv,Td,nu)-

 surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k ,dt=dt.surv,Td,nu)+

 dnorm(k.new,0,10,log=T)-dnorm(k,0,10,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){k <- k.new}

# sampling Td

Td.new <- rnorm(1,Td,.025)

logR <- surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td.new,nu)-

 surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td ,nu)+

 dnorm(Td.new,20,5,log=T)-dnorm(Td,20,5,log=T)

logU <- log(runif(1,0,1))

if(logU<logR){Td <- Td.new}

# sampling nu

nu.new <- rnorm(1,nu,.025)

logR <- surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td,nu.new)-

 surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td,nu )+

 dgamma(nu.new,4,1,log=T)-dgamma(nu,4,1,log=T)+

 log(nu.new)-log(nu)

logU <- log(runif(1,0,1))

if(logU<logR){nu <- nu.new}

# updating the monitors

mon.theta[iter] <- theta

mon.Tb[iter] <- Tb

mon.mupsib50[iter] <- mupsib50

mon.taupsib50[iter] <- taupsib50

mon.sdpsib[iter] <- sdpsib

mon.k[iter] <- k

mon.Td[iter] <- Td

mon.nu[iter] <- nu

#mon.psib50[,iter] <- psib50

# for DIC

if(iter > .5\*ITER){mon.psib50mean <- mon.psib50mean+psib50/(.5\*ITER)}

mon.loglik[iter] <- surv.lik.wei.fun(theta,Tb,psib50,sdpsib,k,dt=dt.surv,Td,nu)

} # end of iterations

}) # end of system time

# the monitors can then be used for posterior inference

# evaluating the DIC

(mean.log <- mean(mon.loglik[seq(.5\*ITER+1,ITER,length.out=1001)]))

log.mean <- surv.lik.wei.fun(

 mean(mon.theta[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mean(mon.Tb[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mon.psib50mean,

 mean(mon.sdpsib[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mean(mon.k[seq(.5\*ITER+1,ITER,length.out=1001)]),dt=dt.surv,

 mean(mon.Td[seq(.5\*ITER+1,ITER,length.out=1001)]),

 mean(mon.nu[seq(.5\*ITER+1,ITER,length.out=1001)]))

(DIC <- -2\*(2\*mean.log-log.mean))