Appendix B **R code for implementation of predator-prey system models**

#Load essential libraries in R#

library(ggplot2)

library(xlsx)

library(rstan)

#MC paramaters#

nch=3 #number of chains

iter=30000 #number of iterations

bin=20000 #size of burn-in

nth=1 #size of thinning

### #### Chrysoperla-Myzus (C-M) system example

#Load data from file#

loc\_dir<-"C:\\Users\\..." #local working directory that data files are found

setwd(loc\_dir)

# read sort time data

newdata\_2 <- read.xlsx(file = "Chrysopidae.xlsx",sheetName = "2hr")

#read long time data

newdata\_24 <- read.xlsx(file = "Chrysopidae.xlsx",sheetName = "24hr")

# deterministic or stochastic type

type<-"stochastic"

#create initial values function

# hours prey exposed to predator 2 hours (short term) and 24 hours (long term)

N<-24

# number of predators

inum<-1

#import corresponding data

newdata<-(get(paste("newdata",N,sep = "\_") )[,c(3,5,4,2,8)])

#Sort original data according to density and missing values

if (N==24) newdata<-newdata[order(newdata$Density.,newdata$NA..1,newdata$Replicate),]

#data size

NN<- nrow(newdata)

#number of odes in the system

M<-1

#number of trials

n\_trials<- nrow(newdata)

#initial time value

t0<-0

#time vector

t <- 1:N

#number of parameters in the ode equation

n\_pars<-2

#number of predators in the model. Plug in as constant coefficeients

pe <- as.array(inum)

#Prey density at the end of the study

Ne <- as.integer(newdata[,2\*inum])

#initial values of densities of prays

A<- as.integer(newdata[,(2\*inum-1)])

#parameters of interest

if (type=="deterministic") parms<- c("theta","dev")

else parms<- c("theta","phi","s\_sq","sigma","dev")

### #### Propylea–Aphis (P-A) system example

locat<-'C:\\...\\'

setwd(locat)

newdata <- read.table("newdata.txt", header = T)

####recursive for Beddington-DeAngelis model ############

# number of predators

inum<-2 #3 or 4

# hours prey exposed to predator

N<-6

#data size

NN<- nrow(newdata)

#number of odes in the system

M<-1

#number of trials

n\_trials<- nrow(newdata)

#initial time value

t0<-0

#time vector

t <- 1:N

#number of parameters in the ode equation predators

n\_pars<-3

#number of predators in the model. Plug in as constant coefficeients

pe <- as.array(inum)

#Prey density at the end of the study

Ne <- as.integer(newdata[,2\*inum])

#initial values of densities of prays

A<- as.integer(newdata[,(2\*inum-1)]) #initial values of densities of prays

#parameters of interest

parms<- c("theta","dev")

###common for all examples

#Input data vector for rstan

my\_data <-

list(t=t,n\_trials=n\_trials,n\_pars=n\_pars,Ne=Ne,A=A,t0=t0,M=M,N=N,NN=NN,pe=pe)

brfun<- function (num\_id=1) {list(kappa=matrix(data = runif(NN\*2,0,1),2,NN),

theta=c(rexp(n\_pars,0.1)),phi=rgamma(1,0.1,0.1),s\_sq=1/rgamma(1,10,10))}

inits<-lapply(1:nch, function(id) brfun(num\_id=id))

####H2 MODEL####

if (type=="deterministic") cfile<-"hol2.stan" else cfile<-"hol\_ou\_twoinstant.stan"

model\_obj<- paste0("mod",inum,"\_hol")

####SSS MODEL####

if(type=="deterministic") cfile="Sss.stan" else cfile="SS\_binom\_ou\_twoinstant.stan"

####BD MODEL####

cfile<-"Beddington.stan"

model\_obj<- paste0("mod",inum,"\_BD")

####CM MODEL####

cfile<-"CrowelyMartin.stan.stan"

model\_obj<- paste0("mod",inum,"\_CM")

####HV MODEL####

cfile<-" Hassell.stan"

model\_obj<- paste0("mod",inum,"\_HV")

###HMC method via STAN###

assign(model\_obj ,

stan(file = cfile,

data = my\_data,

init = inits,

control = list(adapt\_delta = 0.9),

chains = nch,

warmup = bin,

iter = iter,

thin=nth))

result<- as.data.frame(summary(eval(parse(text= model\_obj)),pars=param)$summary)

print(result, d=3)

**#### Create posterior predictive plots ####**

pnames<-c(parms,"y\_fit")

param.sample<-as.data.frame(summary(eval(parse(text=model\_obj)),pars=pnames)$summary)

yypred<-as.matrix(eval(parse(text= model\_obj)),pars="y\_fit")

# vector with predictor x values

xv<-newdata$Density.

#data frames with the percentiles vectors

ymeddata<-data.frame(x = xv, y = apply(yypred,2, median)) #for median

yquant1data<-data.frame(x = xv, y = apply(yypred,2, function(x) quantile(x, c(0.025)) )) #for 1st qr

yquant2data<-data.frame(x = xv, y = apply(yypred,2, function(x) quantile(x, c(0.975)) )) #for 2nd qr

yquant3data<-data.frame(x = xv, y = apply(yypred,2, function(x) quantile(x, c(0.25)) )) #for 1st qr

yquant4data<-data.frame(x = xv, y = apply(yypred,2, function(x) quantile(x, c(0.75)) )) #for 2nd qr

#real data dataframe

ndata<-data.frame(x=newdata$Density.,y=newdata$Aphids..Eaten.)

#new dataframe with augmented results

yndata<-ymeddata

# convert grouping variables to factor

yndata$Legend<-as.factor(yndata$x)

yndata$y1 <- yquant1data$y

yndata$median <- ymeddata$y

yndata$y2 <- yquant2data$y

yndata$y3 <- yquant3data$y

yndata$y4 <- yquant4data$y

Legend<-factor(yndata$x) #the temperature levels

colval<-colorRampPalette(c("royalblue","red"))

scale\_color\_brewer(palette="Dark2")

theme\_update(plot.title = element\_text(hjust = 0.5))

ggplot() + ggtitle("Default is now set to centered")

if (N==2){

ggplot(data = yndata, aes(x = x, y = y)) +

#ylim(-0.75, 0.12)+

geom\_point(data=ndata, size = 1.5, aes(x = x, y = y, colour = "#000000"),alpha=1) +

# geom\_errorbar(aes(ymin=y1, ymax=y2),width=.6,

# position=position\_dodge(0.5))+

geom\_line(aes(x = x, y = median, colour="median"), size=1)+

geom\_ribbon(aes(x = x,ymin=y1,ymax=y2),alpha=0.25, fill="#81DAF5")+

geom\_ribbon(aes(x = x,ymin=y3,ymax=y4),alpha=0.25, fill="#145fa7")+

labs(

#x = expression(atop("Temperature"~degree\*C , paste("(a) Bieri"))),

x = expression(atop("Prey abundance", paste("for short term response (2h)"))),

y = expression("Prey consumed")

,

title = "Fitness of Chrysoperla agilis on Myzus persicae: Posterior predictive Cr.I. vs data",

subtitle = paste0(type," Holling-II model")

)+

#subtitle = "for Tetranychus urticae mites")+

#panel\_bg(fill = "gray95", color = NA) +

#grid\_lines(color = "gray")+

scale\_colour\_manual("Legend",labels = c("Data","Median"),

values = c("black","red"),

guide = guide\_legend(override.aes = list(

linetype = c(rep("blank", 1), "solid"),

shape = c(16, NA,NA),

fill=c("#81DAF5", "#81DAF5"))))+

guides( color = guide\_legend(

order = 1,

override.aes = list(

color = c("black", "red"),

fill = c("white", "white"),

linetype = c("blank", "solid"),

shape = c(16, NA)))) +

theme\_bw() +

# remove legend key border color & background

theme(text = element\_text(size=16),

plot.title = element\_text(size=14),

plot.subtitle = element\_text(size=14,hjust = 0.5),

legend.key = element\_rect(colour = NA, fill = NA),

legend.box.background = element\_blank())+

geom\_point(aes(x = x, y = y, size = "95% Cr.I.", shape = NA)) +

geom\_point(aes(x = x, y = y, size = "50% Cr.I.", shape = NA)) +

guides(size = guide\_legend(NULL,

order = 2,

override.aes = list(shape = c(15,15),

color = c("#145fa7","#81DAF5"),

size = c(6,6) )))

}else

{ggplot(data = yndata, aes(x = x, y = y)) +

#ylim(-0.75, 0.12)+

geom\_point(data=ndata, size = 1.5, aes(x = x, y = y, colour = "#000000"),alpha=1) +

# geom\_errorbar(aes(ymin=y1, ymax=y2),width=.6,

# position=position\_dodge(0.5))+

geom\_line(aes(x = x, y = median, colour="median"), size=1)+

geom\_ribbon(aes(x = x,ymin=y1,ymax=y2),alpha=0.25, fill="#81DAF5")+

geom\_ribbon(aes(x = x,ymin=y3,ymax=y4),alpha=0.25, fill="#145fa7")+

labs(

#x = expression(atop("Temperature"~degree\*C , paste("(a) Bieri"))),

x = expression(atop("Prey abundance", paste("for long term response (24h)"))),

y = expression("Prey consumed")

,

title = "Fitness of Chrysoperla agilis on Myzus persicae: Posterior predictive Cr.I. vs data",

subtitle = paste0(type," Holling-II model")

)+

#subtitle = "for Tetranychus urticae mites")+

#panel\_bg(fill = "gray95", color = NA) +

#grid\_lines(color = "gray")+

scale\_colour\_manual("Legend",labels = c("Data","Median"),

values = c("black","red"),

guide = guide\_legend(override.aes = list(

linetype = c(rep("blank", 1), "solid"),

shape = c(16, NA,NA),

fill=c("#81DAF5", "#81DAF5"))))+

guides( color = guide\_legend(

order = 1,

override.aes = list(

color = c("black", "red"),

fill = c("white", "white"),

linetype = c("blank", "solid"),

shape = c(16, NA)))) +

theme\_bw() +

# remove legend key border color & background

theme(text = element\_text(size=16),

plot.title = element\_text(size=14),

plot.subtitle = element\_text(size=14,hjust = 0.5),

legend.key = element\_rect(colour = NA, fill = NA),

legend.box.background = element\_blank())+

geom\_point(aes(x = x, y = y, size = "95% Cr.I.", shape = NA)) +

geom\_point(aes(x = x, y = y, size = "50% Cr.I.", shape = NA)) +

guides(size = guide\_legend(NULL,

order = 2,

override.aes = list(shape = c(15,15),

color = c("#145fa7","#81DAF5"),

size = c(6,6) )))}

#Calculate and record information criteria and record method elapsed time

library(loo)

ttime<-get\_elapsed\_time(eval(parse(text= model\_obj)))

array.stan <- as.array(eval(parse(text= model\_obj)))

for( i in 1:ncol(array.stan) ){

tempo <- as.data.frame(array.stan[,i,])

tempo$chain <- i

tempo$iteration <- 1:nrow(tempo)

if ( i==1 ) df.res <- tempo

else df.res <- rbind(df.res, tempo)

}

rm(tempo, array.stan, i)

mean.Deviance <- mean(df.res$dev)

var.Deviance <- var(df.res$dev)

df.input<-as.data.frame(eval(parse(text= model\_obj))) # stan fit as data frame

# values of number of prey consumed and initial densities of prey

Ne <- as.integer(newdata[,2\*inum])

A<- as.integer(newdata[,(2\*inum-1)])

if (type=="deterministic") res<-summary(eval(parse(text= model\_obj)),pars="p")$summary else

res<-summary(eval(parse(text= model\_obj)),pars="lambda")$summary[(NN+1):(2\*NN),]

lp<-as.vector(res[,"mean"])

Deviance.mean <- -2\*sum(with(df.input, dbinom(Ne, A, lp, log = T) ))

pD <- mean.Deviance - Deviance.mean

DIC\_res<-c(mean.Dev=mean.Deviance, Dev.mean=Deviance.mean, pD=mean.Deviance-Deviance.mean, DIC=pD+mean.Deviance, DIC\_est=mean.Deviance+var.Deviance/2)

param<-ifelse((parms==c("theta","dev")),n\_pars+1,n\_pars+1+2)[1] #deterministic case uses +1 params for p, deterministic case uses +3 params for p,k,mu

AIC\_BIC<-c(AIC=mean(df.res$dev)+2\*param, BIC=mean(df.res$dev)+param\*log(length(Ne)))

lik\_sss<-extract\_log\_lik( eval(parse(text= paste0("mod",inum,"\_sss"))), parameter\_name = "log\_lik",

merge\_chains = TRUE)

criteria<-rbind(as.data.frame(loo(lik\_sss)$estimates),as.data.frame(waic(lik\_sss)$estimates))

result1<-as.data.frame(summary(eval(parse(text= paste0("mod",inum,"\_sss"))),pars=parms)$summary) # another way to write the summary results of the stan

# sample size info of the model

nmodel<- model\_obj

result2<-data.frame(c(nmodel,nch,iter,bin,nth,nch\*(iter-bin)/nth), row.names = c("Model Name","Number of chains","iterrations","Burn in","Thinning ratio","sample size"))

colnames(result2)="data"

dim(as.array(eval(parse(text= model\_obj)))) #check the sample size within chains

fold<-"C:\\Users\\...\\"

if (type=="deterministic") ccfile<-paste0(fold, model\_obj,N,"\_no\_",format(Sys.Date(), "%d%m%Y"),".xls") else ccfile<-paste0(fold, model\_obj,N," \_ou\_",format(Sys.Date(), "%d%m%Y"),".xls")[1]

# write results in excel file

write.xlsx(result2, ccfile)

write.xlsx(result1, ccfile, append = TRUE, sheetName="results")

write.xlsx(ttime/60,ccfile, append = TRUE, sheetName="elapsed\_time")

write.xlsx(AIC\_BIC, ccfile, append = TRUE, sheetName="AIC\_BIC")

write.xlsx(DIC\_res, ccfile, append = TRUE, sheetName="DIC")

write.xlsx(criteria, ccfile, append = TRUE, sheetName="criteria")

write.xlsx(newdata, ccfile, append = TRUE, sheetName="data")