

# Examining correlates of female-limited polymorphism using MCMCglmm (incl. diagnostics)

Beatriz Willink

Rachel Blow

May 2020

Set your working directory to the parent directory containing all of your MCMC analyses

```
knitr::opts_knit$set(root.dir = '..' )
```

## 1) Read in data

Call the “ape” package for reading trees

```
require(ape)
```

Read in tree samples file

```
tres <- read.nexus("MCMCglmm/sample.2100.trees")
length(tres)
```

```
## [1] 2100
```

Read in data from literature

```
isch_dat <- read.csv("MCMCglmm/ischnura_distribution.csv")
```

Get list of countries by ISO code

```
ischISO <- unique(unlist(c(strsplit(as.character(isch_dat$Countries),
  split = ","))))
```

Calculate area for all countries with Ischnura species using .shp files (maps downloaded from GADM.org)

```
require(raster)
x <- shapefile('MCMCglmm/maps/countries.shp')
area <- vector(length = length(ischISO))

for (i in 1:length(ischISO)){
  area[i] <- raster::area(x[which(x$ISO == paste0(ischISO[i])),])/1000000
}
```

Get area for all states/provinces/territories in Australia, Brazil, Canada, China, India, Russia, and US

```
AUS <- shapefile('MCMCglmm/maps/AUS.shp')
```

```
## Warning in rgdal::readOGR(dirname(x), fn, stringsAsFactors =
  stringsAsFactors, :
## Dropping null geometries: 3
```

```

for (i in 1:length(AUS$NAME_1)){
  area <- append(area, raster::area(AUS[which(AUS$NAME_1 == paste0(AUS$
    NAME_1[i])),,])/1000000)
}

BRA <- shapefile('MCMCglmm/maps/BRA.shp')
for (i in 1:length(BRA$NAME_1)){
  area <- append(area, raster::area(BRA[which(BRA$NAME_1 == paste0(BRA$
    NAME_1[i])),,])/1000000)
}

CAN <- shapefile('MCMCglmm/maps/CAN.shp')
for (i in 1:length(CAN$NAME_1)){
  area <- append(area, raster::area(CAN[which(CAN$NAME_1 == paste0(CAN$
    NAME_1[i])),,])/1000000)
}

CHN <- shapefile('MCMCglmm/maps/CHN.shp')
for (i in 1:length(CHN$NAME_1)){
  area <- append(area, raster::area(CHN[which(CHN$NAME_1 == paste0(CHN$
    NAME_1[i])),,])/1000000)
}

IND <- shapefile('MCMCglmm/maps/IND.shp')
for (i in 1:length(IND$NAME_1)){
  area <- append(area, raster::area(IND[which(IND$NAME_1 == paste0(IND$
    NAME_1[i])),,])/1000000)
}

IDN <- shapefile('MCMCglmm/maps/IDN.shp')
for (i in 1:length(IDN$NAME_1)){
  area <- append(area, raster::area(IDN[which(IDN$NAME_1 == paste0(IDN$
    NAME_1[i])),,])/1000000)
}

RUS <- shapefile('MCMCglmm/maps/RUS.shp')
for (i in 1:length(RUS$NAME_1)){
  area <- append(area, raster::area(RUS[which(RUS$NAME_1 == paste0(RUS$
    NAME_1[i])),,])/1000000)
}

TUR <- shapefile('MCMCglmm/maps/TUR.shp')
for (i in 1:length(TUR$NAME_1)){
  area <- append(area, raster::area(TUR[which(TUR$NAME_1 == paste0(TUR$
    NAME_1[i])),,])/1000000)
}

US <- shapefile('MCMCglmm/maps/USA.shp')
for (i in 1:length(US$NAME_1)){
  area <- append(area, raster::area(US[which(US$NAME_1 == paste0(US$NAME_
    1[i])),,])/1000000)
}

area.dat <- data.frame("division" = c(ischISO, AUS$NAME_1, BRA$NAME_1, CAN

```

```
$NAME_1, CHN$NAME_1, IND$NAME_1, IDN$NAME_1, RUS$NAME_1, TUR$NAME_1, US
$NAME_1), "area" = area)
```

Add areas for each species

```
Range <- vector()
for (j in 1:length(isch_dat$Taxon)){
  divs <- unlist(c(strsplit(as.character(isch_dat$Countries[j]), split = "
    ,"),
    strsplit(paste0(isch_dat$Other_division[j]), split = ",,")))
  ar <- 0
  for (i in divs){
    temp <- area.dat$area[which(area.dat$division == paste0(i))]
    ar <- ar + temp
  }
  Range[j] <- ar
}

isch_dat <- cbind(isch_dat[2:5], Range)
```

I. ezoin is endemic of Ogasawara islands in Japan, according to IUCN the estimated extent of occurrence is 258 km<sup>2</sup>

```
isch_dat$Range[which(isch_dat$Taxon == "Ischnura_ezoin")] <- 258
```

Call the “geiger” package for “name.check” function

```
require(geiger)
```

Check that species names from literature data match species names in tree file

```
Taxa <- data.frame(Taxon = unique(isch_dat$Taxon))
rownames(Taxa) <- Taxa$Taxon
name.check(tres[[1]], Taxa)
```

```
## [1] "OK"
```

Call the “MCMCglmm” package for creating MCMC objects

```
require(MCMCglmm)
```

Calculate inverted relatedness matrix

```
inv.phylo <- inverseA(tres[[1]], nodes="TIPS", scale=TRUE)
```

## 2) Analysis

### Run 1

Set prior

```
prior.1 <- list(B = list(mu = c(0,0), V = diag(2) * (1 + pi^2/3)),
  G=list(G1=list(V=1, nu=1000, alpha.mu=rep(0,1), alpha.V=diag
    (1))),
  R = list(V = 1, fix = 1))
```

Start run

```
m1.bin.start <- MCMCglmm(FemState.bin ~ Range,
                        random = ~ Taxon,
                        rcov = ~ units,
                        ginverse=list(Taxon=inv.phylo$Ainv),
                        family = "categorical", data = isch_dat,
                        prior = prior.1, pl=TRUE,slice=TRUE,
                        nitt=2000, burnin=0, thin=1,verbose = F)
```

Loop over multiple trees, saving the last iteration of 10,000 iterations for each tree -100

```
m1.bin.multiplyMCMC <- m1.bin.start

for(i in 1:length(tres)){
  IN.tree <- inverseA(tres[[i]],nodes="TIPS")
  start <- list(Liab = m1.bin.multiplyMCMC$Liab[1,],
               R=list(R1=matrix(ncol=1,nrow=1,m1.bin.multiplyMCMC$VCV
                               [1,2])),
               G=list(G1=matrix(ncol=1,nrow=1,m1.bin.multiplyMCMC$VCV
                               [1,1])))

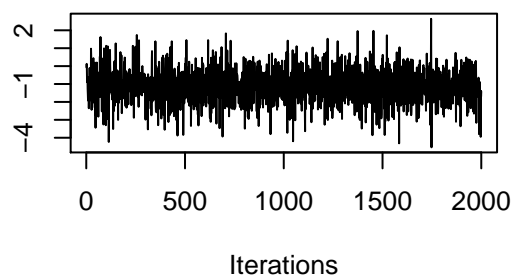
  m1.bin.multiplyMCMC <- MCMCglmm(FemState.bin ~ Range,
                                  random = ~ Taxon,
                                  rcov = ~ units,
                                  ginverse=list(Taxon=inv.phylo$Ainv),
                                  family = "categorical", data = isch_dat,
                                  prior = prior.1,
                                  pl=TRUE,slice=TRUE, nitt=10000, thin=1,
                                  burnin=9999, start=start, verbose=F)

  if(i>100){
    m1.bin.start$VCV[i-100,] <- m1.bin.multiplyMCMC$VCV[1,]
    m1.bin.start$Sol[i-100,] <- m1.bin.multiplyMCMC$Sol[1,]
    m1.bin.start$Liab[i-100,] <- m1.bin.multiplyMCMC$Liab[1,]
  }
}
```

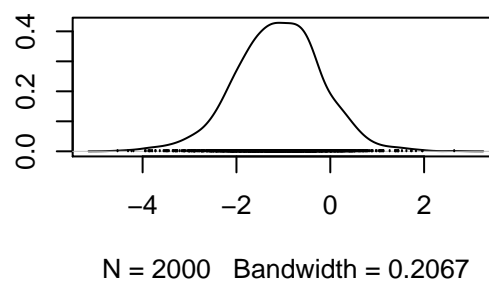
Visual check:

```
plot(m1.bin.start)
```

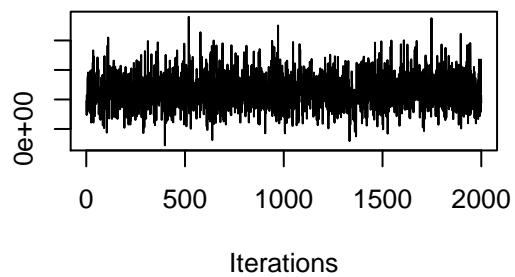
**Trace of (Intercept)**



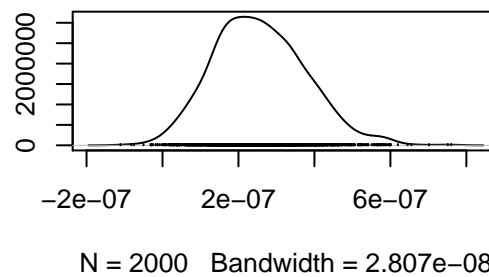
**Density of (Intercept)**

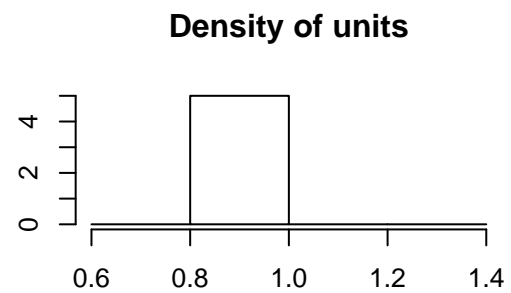
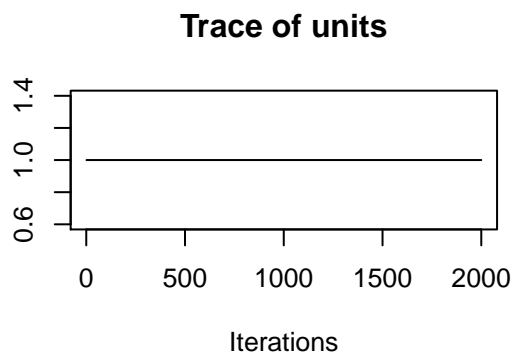
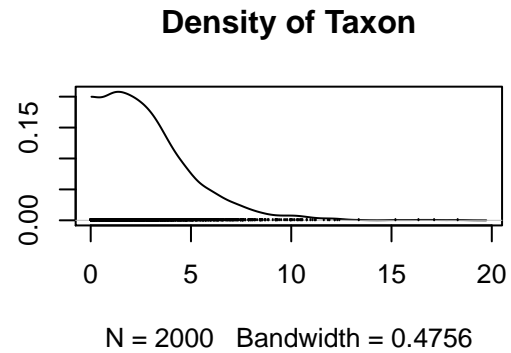
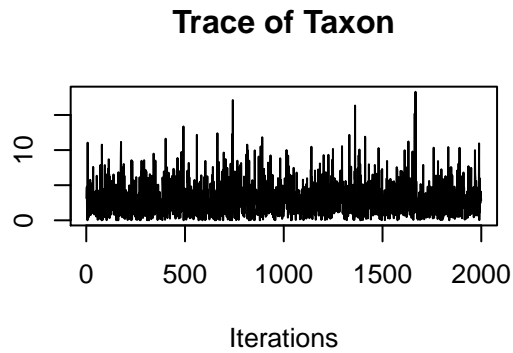


**Trace of Range**



**Density of Range**





Summary:

```
summary(m1.bin.start)
```

```
##
## Iterations = 1:2000
## Thinning interval = 1
## Sample size = 2000
##
## DIC: 40.47319
##
## G-structure: ~Taxon
##
##      post.mean  1-95% CI u-95% CI eff.samp
## Taxon      2.916 1.555e-05   7.267     2000
##
## R-structure: ~units
##
##      post.mean 1-95% CI u-95% CI eff.samp
## units          1      1      1         0
##
## Location effects: FemState.bin ~ Range
##
##      post.mean  1-95% CI  u-95% CI  eff.samp  pMCMC
## (Intercept) -1.107e+00 -3.010e+00  6.127e-01    2000 0.213
## Range       2.561e-07  1.664e-08  4.764e-07    2000 0.014 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Hypothesis test: what is the posterior probability that the probability of polymorphism does not increase with distribution range

```
PMCMC <- sum(m1.bin.start$Sol[, 'Range'] <= 0)/length((m1.bin.start$Sol[, 'Range'] <= 0))
```

```
PMCMC
```

```
## [1] 0.007
```

Posterior mean and HPD interval for the effect of an increase of 1 million Km2 on the probability of being polymorphic

```
PM <- plogis(mean(m1.bin.start$Sol[,1]) + mean(m1.bin.start$Sol[,2])* 1e6) -  
      plogis(mean(m1.bin.start$Sol[,1]))
```

```
PM
```

```
## [1] 0.05082946
```

```
lwr <- plogis(mean(m1.bin.start$Sol[,1]) + HPDinterval(m1.bin.start$Sol[,2])[1]* 1e6) -  
      plogis(mean(m1.bin.start$Sol[,1]))
```

```
lwr
```

```
## [1] 0.00311974
```

```
upr <- plogis(mean(m1.bin.start$Sol[,1]) + HPDinterval(m1.bin.start$Sol[,2])[2]* 1e6) -  
      plogis(mean(m1.bin.start$Sol[,1]))
```

```
upr
```

```
## [1] 0.09895277
```

Obtain estimate using “predict” function

```
Pred <- predict.MCMCglmm(object = m1.bin.start, newdata = isch_dat, type = "response",  
                          interval = "confidence")
```

```
HPDinterval(m1.bin.start$Sol[,2])
```

```
##           lower           upper  
## var1 1.663537e-08 4.763575e-07  
## attr(,"Probability")  
## [1] 0.95
```

Make the fitted data frame

```
Pdat <- cbind(isch_dat, Pred)
```

```
for(i in 1:nrow(Pdat)){  
  if(Pdat$FemState.bin[i] == "P"){  
    Pdat$P[i] <- 1  
  }  
  else {Pdat$P[i] <- 0}  
}
```

## Run 2

Start run

```
m2.bin.start <- MCMCglmm(FemState.bin ~ Range,
                        random = ~ Taxon,
                        rcov = ~ units,
                        ginverse=list(Taxon=inv.phylo$Ainv),
                        family = "categorical", data = isch_dat,
                        prior = prior.1, pl=TRUE,slice=TRUE,
                        nitt=2000, burnin=0, thin=1,verbose = F)
```

Loop over multiple trees, saving the last iteration of 10,000 iterations for each tree -100

```
m2.bin.multiplyMCMC <- m2.bin.start

for(i in 1:length(tres)){
  IN.tree <- inverseA(tres[[i]],nodes="TIPS")
  start <- list(Liab = m2.bin.multiplyMCMC$Liab[1,],
               R=list(R1=matrix(ncol=1,nrow=1,m2.bin.multiplyMCMC$VCV
                               [1,2])),
               G=list(G1=matrix(ncol=1,nrow=1,m2.bin.multiplyMCMC$VCV
                               [1,1])))

  m2.bin.multiplyMCMC <- MCMCglmm(FemState.bin ~ Range,
                                  random = ~ Taxon,
                                  rcov = ~ units,
                                  ginverse=list(Taxon=inv.phylo$Ainv),
                                  family = "categorical", data = isch_dat,
                                  prior = prior.1,
                                  pl=TRUE,slice=TRUE, nitt=10000, thin=1,
                                  burnin=9999, start=start, verbose=F)

  if(i>100){
    m2.bin.start$VCV[i-100,]<- m2.bin.multiplyMCMC$VCV[1,]
    m2.bin.start$Sol[i-100,]<- m2.bin.multiplyMCMC$Sol[1,]
    m2.bin.start$Liab[i-100,]<- m2.bin.multiplyMCMC$Liab[1,]
  }
}
```

Put them into a list

```
mh.list <- mcmc.list(m1.bin.start$Sol, m2.bin.start$Sol)
```

## 3) Plotting

Call “ggplot2” for plotting, “wesanderson” for colour scheme and “scales” for formatting axis scales

```
require(ggplot2)
require(wesanderson)
require(scales)
```

Plot probability of polymorphism against range size

```
p1 <- ggplot(data = Pdat, aes(x=Range, y = fit))+
  geom_line(size=1, show.legend = NA, color = "black")+
  theme_minimal()
```

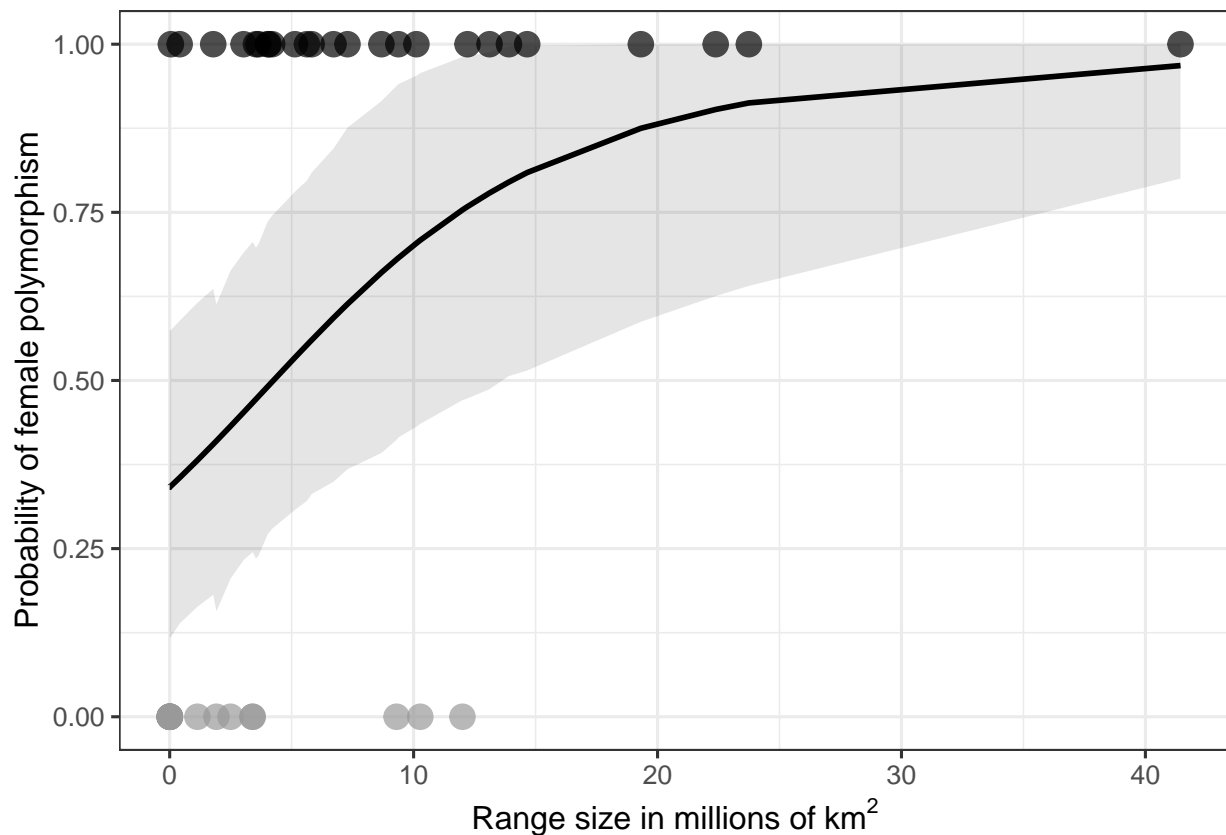


```

theme_bw(base_size = 12)+
geom_ribbon(aes(ymax = upr, ymin = lwr, x = Range), fill = "black",
  color=NA,
  alpha = 0.1)+
geom_point(data = Pdat, aes(color = FemState.bin, fill = FemState.bin, y
  = P),
  alpha = 0.7 , size = 4, show.legend = FALSE )+
ylab(label = "Probability of female polymorphism")+
xlab(label=c(expression("Range size in millions of" ~km^2))) +
scale_color_manual(breaks = c("M", "P"),
  values = c("grey60", "black")) +
scale_fill_manual(breaks = c("M", "P"),
  values = c("grey60", "black")) +
scale_x_continuous(labels = number_format(scale = 1e-6))
theme(panel.grid.major = element_blank(), panel.grid.minor = element_
  blank())

```

p1



Monomorphic: light grey

Polymorphic: dark grey

Call plotting packages

```

require(gridExtra)
require(gtable)
require(ggtree)

```

Plot range size per species and probability of polymorphism against range size

```
tre1 <- read.nexus(file = "StarBEAST2/summary.tree")
range_dat <- data.frame(Species = isch_dat[,1], State = isch_dat[,3],
  Range = isch_dat$Range)

for (i in 1:length(range_dat$Species)){
  if (range_dat$Range[i] < 100000){
    range_dat$Range[i] <- 100000
  }
}

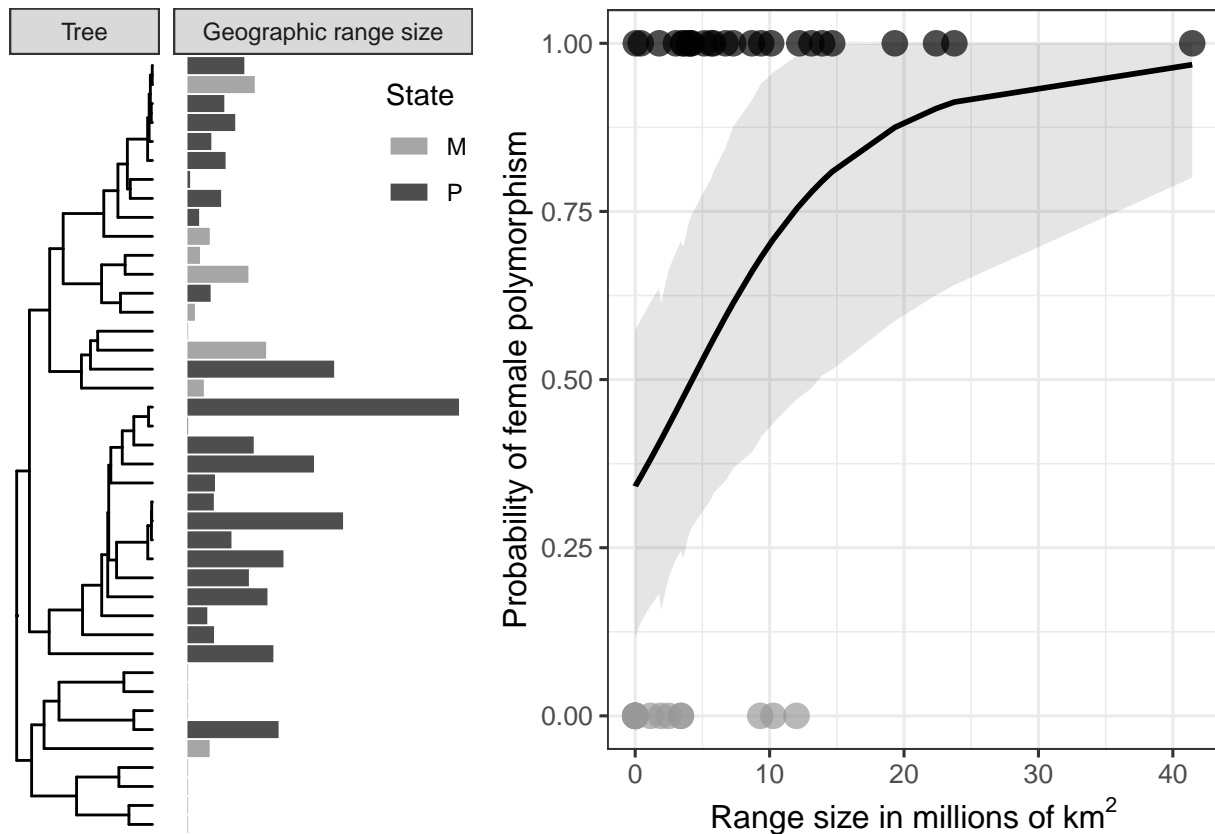
p <- ggtree(tre1)

p2 <- facet_plot(p, panel="Geographic_range_size", data=range_dat, geom=
  geom_segment, mapping = aes(x=0, xend=Range, y=y, yend=y, color = State
  ), size = 3, alpha = 0.7) + scale_color_manual(values = c("grey50", "
  black")) + theme(legend.position = c(.90, .88))

gt = ggplot_gtable(ggplot_build(p2))
gt$layout$1[grep('panel-1', gt$layout$name)]

## [1] 5 7

gt$widths[5] = 0.5*gt$widths[5]
lay <- rbind(c(1,1,2,2,2))
Fig5 <- grid.arrange(gt, p1, ncol = 2, layout_matrix = lay)
```



Save plot as a PDF

```
ggsave("Fig5.pdf", plot = Fig5, device = NULL, path = "Figures",
       scale = 1, width = 7, height = 5, dpi = 600, limitsize = TRUE,
       units = "in")
```

## Diagnostics

a) **Gelman-Rubin statistic** Gelman-rubin diagnostic of convergence

```
GR <- gelman.diag(mh.list, confidence = 0.95, transform=TRUE, autoburnin=
  FALSE,
                  multivariate=T)
```

GR

```
## Potential scale reduction factors:
##
##           Point est. Upper C.I.
## (Intercept)           1         1.01
## Range              1         1.00
##
## Multivariate psrf
##
## 1
```

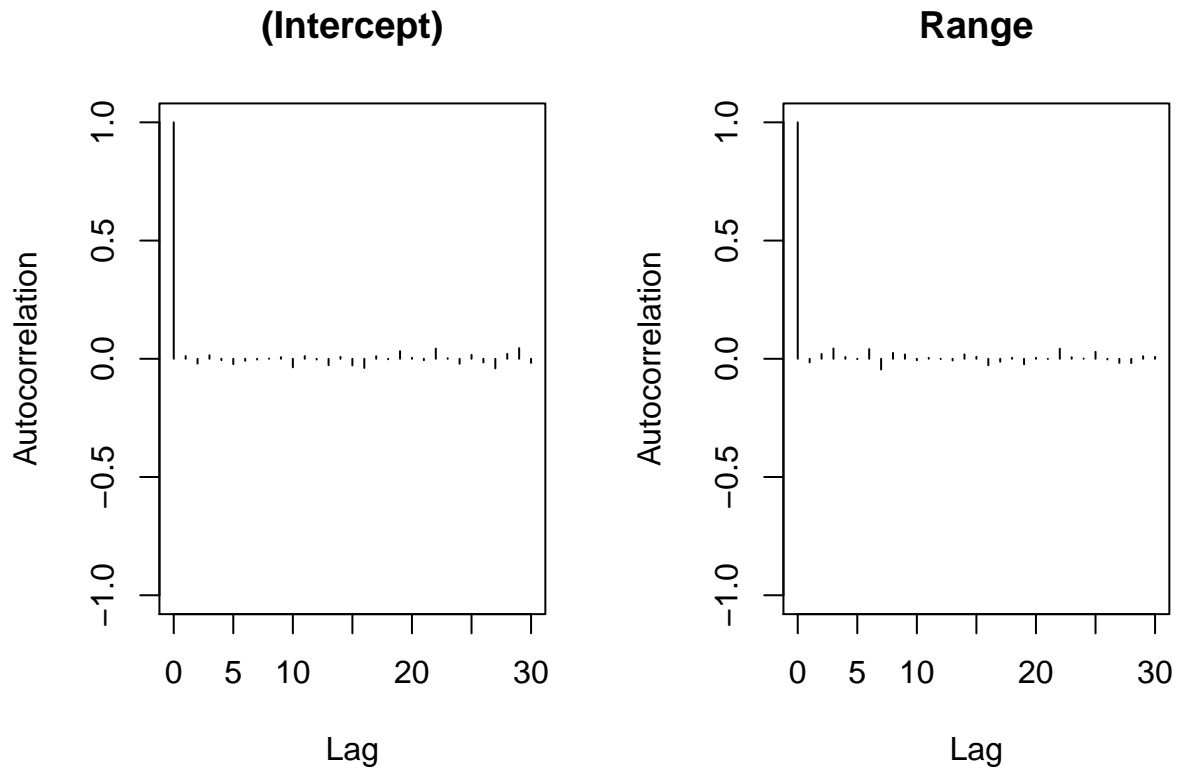
b) **Autocorrelation** Calculate autocorrelation between draws

```
diag(autocorr(m1.bin.start$Sol)[2, , ])
```

```
## (Intercept)      Range
##  0.01137690 -0.01508488
```

Present autocorrelation in plot form

```
autocorr.plot(m1.bin.start$Sol)
```



c) **Effective sample size** Calculate effective sample size

```
effectiveSize(m1.bin.start$Sol)
```

```
## (Intercept)      Range
##          2000      2000
```

d) **Visual diagnostic of trace convergence and mixing** Prepare data for plotting multiple traces

```
colnames(m1.bin.start$Sol)[1] <- "Intercept"
```

```
for (i in 1:length(colnames(m1.bin.start$Sol))){
  for (j in 1:2){
    temp <- eval(parse(text=paste("m", j, ".bin.start$Sol[,", i, "]", sep = "
")))
    assign(paste("trace_", "mcmc", j, colnames(m1.bin.start$Sol)[i], sep="
"), temp)
  }
}
```

```
temp_list = mcmc.list()
for (i in 1:length(colnames(m1.bin.start$Sol))){
  for (j in 1:2){
    temp_list[[j]] <- eval(parse(text=paste("trace_mcmc", j ,
                                             colnames(m1.bin.start$Sol)[i]
                                             ], sep = "")))
  }
}
```

```

    assign(paste("trace_", colnames(m1.bin.start$Sol)[i], sep = ""), temp_
          list)
  }
}

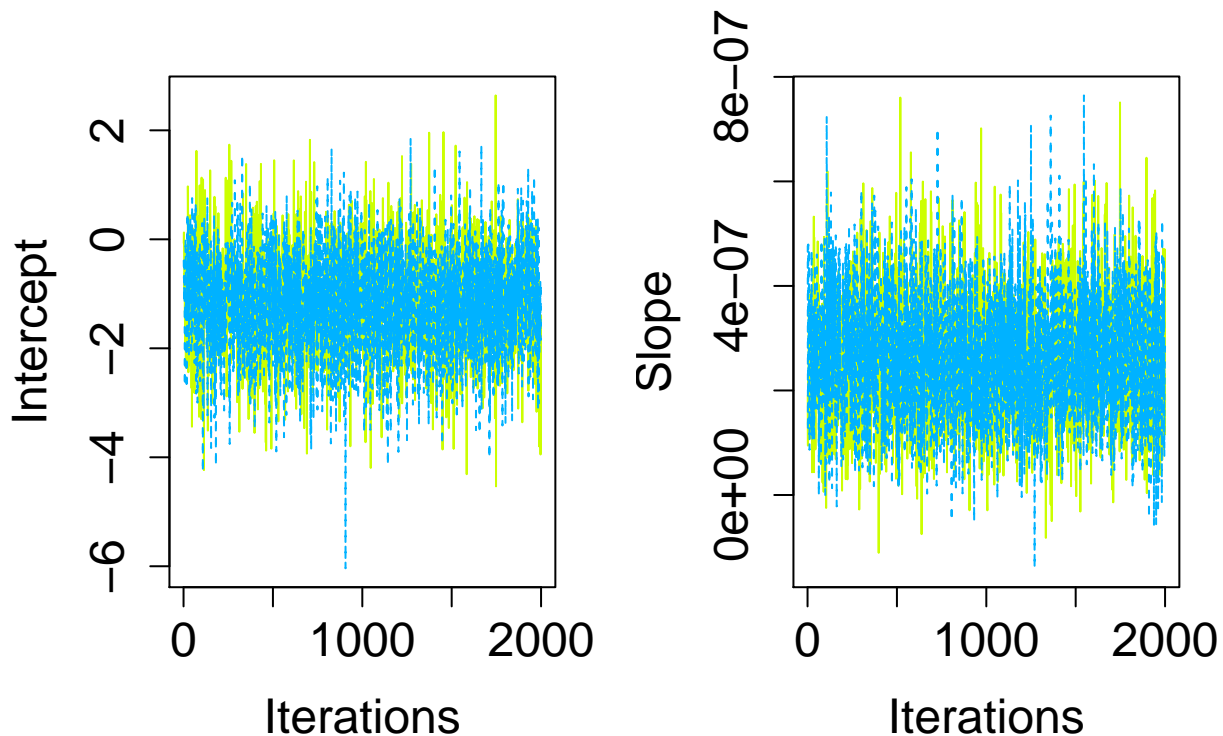
```

Plot traces

```

ylabs <- c("Intercept", "Slope")
par(mfrow=c(1,2))
for (i in 1:2){
  temp <- eval(parse(text=paste("trace_", colnames(m1.bin.start$Sol)[i],
    sep = "")))
  traceplot(temp,col = rainbow(3, start=0.2, end=0.9), ylab = ylabs[i],
    cex.lab=1.5, cex.axis=1.5)
}

```



Prepare data for plotting multiple density curves

```

colnames(m2.bin.start$Sol)[1] <- "Intercept"

reshaping <- function(x) {
  y <- rbind(t(x))
  result <- setNames(as.data.frame.table(y),c("variable","run","value"))
}

for (i in 1:2){
  temp <- eval(parse(text=paste("m", i, ".bin.start$Sol", sep = "")))

```

```

  reshape <- reshaping(temp)
  reshape$run <- as.factor(rep(i, nrow(reshape)))
  assign(paste("reshape", i, sep = ""), reshape)
}

```

```
plot_df <- rbind(reshape1, reshape2)
```

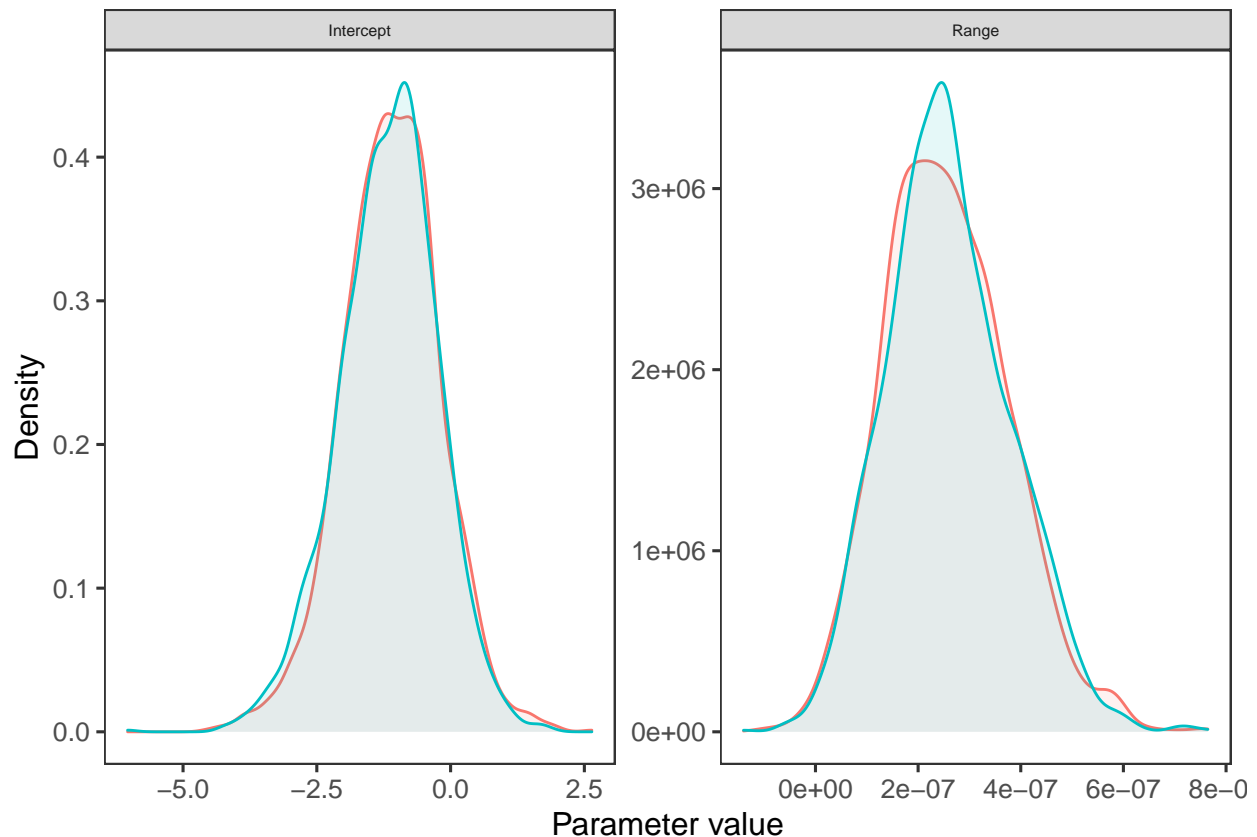
Plot by each parameter

```

p <- ggplot(plot_df, aes(x= value, fill = run, colour = run)) +
  geom_density(alpha = 0.1) +
  theme_bw(base_size = 12)+
  theme(legend.position = "none", strip.text.x = element_text(size = 6),
        panel.grid.major = element_blank(), panel.grid.minor = element_
        blank())+
  labs(x="Parameter_value", y = "Density") +
  facet_wrap(~variable, scales = "free", ncol = 3, nrow = 5)

```

p



## Run with log

Start run

```

m3.bin.start <- MCMCglmm(FemState.bin ~ log(Range),
  random = ~ Taxon,

```

```
rcov = ~ units,
ginverse=list(Taxon=inv.phylo$Ainv),
family = "categorical", data = isch_dat,
prior = prior.1, pl=TRUE,slice=TRUE,
nitt=2000, burnin=0, thin=1,verbose = F)
```

Loop over multiple trees, saving the last iteration of 10,000 iterations for each tree -100

```
m3.bin.multiplyMCMC <- m3.bin.start

for(i in 1:length(tres)){
  IN.tree <- inverseA(tres[[i]],nodes="TIPS")
  start <- list(Liab = m3.bin.multiplyMCMC$Liab[1,],
               R=list(R1=matrix(ncol=1,nrow=1,m3.bin.multiplyMCMC$VCV
                             [1,2])),
               G=list(G1=matrix(ncol=1,nrow=1,m3.bin.multiplyMCMC$VCV
                             [1,1])))

  m3.bin.multiplyMCMC <- MCMCglmm(FemState.bin ~ log(Range),
                                random = ~ Taxon,
                                rcov = ~ units,
                                ginverse=list(Taxon=inv.phylo$Ainv),
                                family = "categorical", data = isch_dat,
                                prior = prior.1,
                                pl=TRUE,slice=TRUE, nitt=10000, thin=1,
                                burnin=9999, start=start, verbose=F)

  if(i>100){
    m3.bin.start$VCV[i-100,]<- m3.bin.multiplyMCMC$VCV[1,]
    m3.bin.start$Sol[i-100,]<- m3.bin.multiplyMCMC$Sol[1,]
    m3.bin.start$Liab[i-100,]<- m3.bin.multiplyMCMC$Liab[1,]
  }
}
```

Hypothesis test: what is the posterior probability that the probability of polymorphism does not increase with distribution range

```
PMCMC <- sum(m3.bin.start$Sol[,2] <= 0)/length((m3.bin.start$Sol[,2] <= 0)
)
PMCMC
```

```
## [1] 0.0235
```

Posterior mean and HPD interval for the effect of an increase of 1 million Km2 on the probability of being polymorphic

```
PM <- plogis(mean(m3.bin.start$Sol[,1]) + mean(m3.bin.start$Sol[,2])) -
  plogis(mean(m3.bin.start$Sol[,1]))
PM
```

```
## [1] 0.01322369
```

```
lwr <- plogis(mean(m3.bin.start$Sol[,1]) + HPDinterval(m3.bin.start$Sol
[,2])[1]) -
  plogis(mean(m3.bin.start$Sol[,1]))
lwr
```

```
## [1] 0.0007493114

upr <- plogis(mean(m3.bin.start$Sol[,1]) + HPDinterval(m3.bin.start$Sol
[,2])[2]) -
  plogis(mean(m3.bin.start$Sol[,1]))
upr

## [1] 0.02963553

Obtain estimate using "predict" function
Pred3 <- predict.MCMCglmm(object = m3.bin.start, newdata = isch_dat, type
= "response",
                           interval = "confidence")
HPDinterval(m3.bin.start$Sol[,2])

##           lower      upper
## var1 0.01631059 0.51327
## attr(,"Probability")
## [1] 0.95

Make the fitted data frame
Pdat3 <- cbind(isch_dat, Pred3)

for(i in 1:nrow(Pdat3)){
  if(Pdat3$FemState.bin[i] == "P"){
    Pdat3$P[i] <- 1
  }
  else {Pdat3$P[i] <- 0}
}

Plot

p3 <- ggplot(data = Pdat3, aes(x=Range, y = fit))+
  geom_line(size=1, show.legend = NA, color = "black")+
  theme_bw(base_size = 12)+
  geom_ribbon(aes(ymax = upr, ymin = lwr, x = Range), fill = "black",
    color=NA,
    alpha = 0.1)+
  geom_point(data = Pdat3, aes(color = FemState.bin, fill = FemState.bin,
    y = P),
    alpha = 0.7 , size = 4, show.legend = FALSE )+
  ylab(label = "Probability of female polymorphism")+
  xlab(label=c(expression("Range size in millions of" ~km^2))) +
  scale_color_manual(breaks = c("M", "P"),
    values = c("grey60", "black")) +
  scale_fill_manual(breaks = c("M", "P"),
    values = c("grey60", "black")) +
  scale_x_continuous(labels = number_format(scale = 1e-6))
theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank
())

## List of 2
## $ panel.grid.major: list()
## .. attr(*, "class")= chr [1:2] "element_blank" "element"
```



```
## $ panel.grid.minor: list()
## ..- attr(*, "class")= chr [1:2] "element_blank" "element"
## - attr(*, "class")= chr [1:2] "theme" "gg"
## - attr(*, "complete")= logi FALSE
## - attr(*, "validate")= logi TRUE
```

p3

