

## Appendix S3: Community LDG Model

*Alan Fecchio, Jeffrey A. Bell, Mariane Bosholn, Jefferson A. Vaughan, Vasyl V. Tkach, Holly L. Lutz, Victor R. Cueto, Cristian A. Gorosito, Daniel González-Acuña, Chad Stromlund, Danielle Kvasager, Kiba J. M. Comiche, Karin Kirchgatter, João B. Pinho, Jacob Beru, Marina Anciães, Carla S. Fontana, Kristof Zyskowski, Sidnei Sampaio, Janice H. Dispoto, Spencer C. Galen, Jason D. Weckstein, Nicholas J. Clark*

Load the Bayesian distribution of *Leucocytozoon* trees

```
paras_trees_reduced <- ape::read.nexus("Parasite phylogenies/Leuco_trees_reduced.nexus")
```

Import the prepared region x lineage matrix

```
load("Processed data/site_paras_mat.rda")
```

Identify parasites in the matrix that aren't in the tree (these sequences were deemed too short for phylogeny estimation)

```
rand_tree <- sample(paras_trees_reduced,  
                     size = 1)[[1]]  
not_in_tree <- setdiff(colnames(site_paras_mat),  
                       rand_tree$tip.label)
```

Drop these lineages from the matrix

```
regions <- site_paras_mat`Area Endemism`  
library(dplyr)  
site_paras_mat_reduced <- site_paras_mat %>%  
  dplyr::select(-dplyr::matches(paste0(not_in_tree,  
                                    collapse = "|")))
```

Double check (this should be zero)

```
setdiff(colnames(site_paras_mat_reduced),  
        rand_tree$tip.label)
```

```
## character(0)
```

Likewise, drop parasites from the trees that are not in matrix

```
not_in_mat <- setdiff(rand_tree$tip.label,  
                      colnames(site_paras_mat_reduced))  
paras_trees_reduced <- lapply(paras_trees_reduced,  
                               ape::drop.tip, tip = (not_in_mat))  
class(paras_trees_reduced) <- "multiPhylo"
```

Double check

```
rand_tree <- sample(paras_trees_reduced,  
                     size = 1)[[1]]  
setdiff(rand_tree$tip.label, colnames(site_paras_mat_reduced))
```

```
## character(0)
```

Transpose the matrix and add column names for regions

```
site_paras_mat_reduced <- data.frame(t(site_paras_mat_reduced))
colnames(site_paras_mat_reduced) <- regions
```

Calculate interpolated phylogenetic diversity for each region across each tree. Regions with fewer than 6 parasites recorded will be omitted as previous tests revealed that diversity could not be estimated for these

```
divs_list <- lapply(seq_len(100), function(x) {
  cat("Processing tree", x, "out of 100...\n")
  tree <- ape::write.tree(paras_trees_reduced[[x]],
    file = "", append = FALSE,
    digits = 10)
  tree <- ade4::newick2phylog(tree)

  paras_names <- names(tree$leaves)
  site_paras_mat_reduced <- site_paras_mat_reduced[paras_names,
    ]
  too_few <- which(colSums(site_paras_mat_reduced) <
    6)

  divs <- iNextPD::iNextPD(x = site_paras_mat_reduced[,
    -too_few], labels = paras_names,
    phy = tree, q = 0, datatype = "abundance")
  divs <- data.frame(divs$AsyPDEst) %>%
    purrr::set_names(c("Area Endemism",
      "q", "Measure", "Diversity")) %>%
    dplyr::filter(q == "q = 1") %>%
    dplyr::filter(Measure == "Estimator") %>%
    dplyr::mutate(Diversity = as.vector(scale(Diversity))) %>%
    dplyr::select(-q, -Measure)
})
diversities <- do.call(rbind, divs_list)
```

Now that diversity estimates have been calculated for each of the 100 parasite trees, we will test for the presence of a latitudinal diversity gradient. First, calculate means and sds for each of the community-level covariates. These will allow us to model their effects using latent variables to account for uncertainty

```
area_descriptors <- paras_reg_dat %>%
  dplyr::mutate(Latitude = as.vector(scale(abs(Latitude))),
    Altitude = as.vector(scale(Altitude))) %>%
  dplyr::group_by(`Area Endemism`) %>%
  dplyr::summarise(mean_lat = mean(Latitude,
    na.rm = T), sd_lat = sd(Latitude,
    na.rm = T), mean_alt = mean(Altitude,
    na.rm = T), sd_alt = sd(Altitude,
    na.rm = T), mean_ndvi = mean(NDVI,
    na.rm = T), sd_ndvi = sd(NDVI,
    na.rm = T), mean_ndwi = mean(NDWI,
    na.rm = T), sd_ndwi = sd(NDWI,
    na.rm = T)) %>% dplyr::ungroup() %>%
  dplyr::filter(`Area Endemism` %in%
    unique(diversities$`Area Endemism`)) %>%
  dplyr::mutate(`Area Endemism` = as.factor(`Area Endemism`)) %>%
  dplyr::arrange(`Area Endemism`)
```

Gather necessary variables for the JAGS model

```

diversity <- diversities$Diversity
region <- diversities`Area Endemism`
n_region <- nrow(area_descriptors)

mean_lat <- area_descriptors$mean_lat
sd_lat <- area_descriptors$sd_lat
mean_alt <- area_descriptors$mean_alt
sd_alt <- area_descriptors$sd_alt
mean_ndvi <- area_descriptors$mean_ndvi
sd_ndvi <- area_descriptors$sd_ndvi
mean_ndwi <- area_descriptors$mean_ndwi
sd_ndwi <- area_descriptors$sd_ndwi

n_obs <- nrow(diversities)

jags_dat <- list(diversity = diversity,
  region = as.numeric(region), n_region = n_region,
  mean_lat = mean_lat, sd_lat = sd_lat,
  mean_alt = mean_alt, sd_alt = sd_alt,
  mean_ndvi = mean_ndvi, sd_ndvi = sd_ndvi,
  mean_ndwi = mean_ndwi, sd_ndwi = sd_ndwi,
  n_obs = n_obs)

```

The following model is JAGS syntax for estimating predictors of phylogenetic community diversity

```

cat("
  model {
    #Formulate normal distributions for drawing site-level predictors
    for(i in 1:n_obs){
      latitude[i] ~ dnorm(mean_lat[region[i]], pow(sd_lat[region[i]] + 0.01, -2));
      altitude[i] ~ dnorm(mean_alt[region[i]], pow(sd_alt[region[i]] + 0.01, -2));
      ndvi[i] ~ dnorm(mean_ndvi[region[i]], pow(sd_ndvi[region[i]] + 0.01, -2));
      ndwi[i] ~ dnorm(mean_ndwi[region[i]], pow(sd_ndwi[region[i]] + 0.01, -2));
    }

    #Priors for beta coefficients
    beta_lat ~ dnorm(0, tau_lat)
    beta_alt ~ dnorm(0, tau_alt)
    beta_ndvi ~ dnorm(0, tau_ndvi)
    beta_ndwi ~ dnorm(0, tau_ndwi)

    tau_lat <- pow(sigma_lat, -2); sigma_lat ~ dexp(0.1)
    tau_alt <- pow(sigma_alt, -2); sigma_alt ~ dexp(0.1)
    tau_ndvi <- pow(sigma_ndvi, -2); sigma_ndvi ~ dexp(0.1)
    tau_ndwi <- pow(sigma_ndwi, -2); sigma_ndwi ~ dexp(0.1)

    #Estimate varying intercepts among regions from normal distributions
    #Redundant parameter expansion allows the sampler to move more freely and converge
    #more quickly
    alpha[1] <- 0 #set the reference category to zero
    alpha_raw[1] <- 0
    for (j in 2:n_region){
      alpha_raw[j] ~ dnorm(0, tau_alpha_raw);
    }
  }
")

```

```

alpha[j] <- xi_alpha*(alpha_raw[j] - mean(alpha_raw[]))
}
xi_alpha ~ dunif(0.01, 100)
tau_alpha_raw <- pow(sigma_alpha_raw, -2); sigma_alpha_raw ~ dexp(0.5)
sigma_alpha <- xi_alpha*sigma_alpha_raw

#Gaussian regression likelihood w/ varying intercepts for each region
for(i in 1:n_obs){
  diversity[i] ~ dnorm(mu[i], tau)
  mu[i] <- alpha[region[i]] + beta_lat * latitude[i] +
    beta_alt * altitude[i] +
    beta_ndvi * ndvi[i] + beta_ndwi * ndwi[i]
}
tau <- pow(sigma_overall, -2); sigma_overall ~ dexp(0.5)

#Regional predictions for easy regional plotting
for(j in 1:n_region){
  region_pred[j] <- alpha[j] + beta_lat * mean_lat[j] +
    beta_alt * mean_alt[j] +
    beta_ndvi * mean_ndvi[j] +
    beta_ndwi * mean_ndwi[j]
}

#Posterior predictive check
for(i in 1:n_obs){
  pred[i] <- mu[i];
  residual[i] <- diversity[i] - pred[i];
  sq[i] <- pow(residual[i], 2);
  sim[i] ~ dnorm(mu[i], tau);
  sq_sim[i] <- pow(sim[i] - pred[i], 2)
}

fit <- sum(sq[])
fit_sim <- sum(sq_sim[])
test <- step(fit_sim - fit)
ppc <- mean(test)
}
",
file = (div.jags.mod <- tempfile()))

```

State the parameters to monitor in each model iteration

```
param <- c("alpha", "beta_lat", "beta_alt",
  "beta_ndvi", "beta_ndwi", "sigma_overall",
  "sigma_lat", "sigma_alt", "region_pred",
  "sigma_ndvi", "sigma_ndwi", "ppc",
  "latitude", "ndvi", "sim")
```

Provide a function to give initial values to each modelled parameter

```
jags_inits <- function() {
  list(xi_alpha = 1)
}
```

Model run parameters

```
na = 1000 # adaptation
nb = 1e+05 # burnin
ni = 4000 # samples
nc = 2 # chains
```

Run the model (note, on a 120gb Linux system, this took ~35 mins to complete)

```
library(rjags)
load.module("glm")
rjags_mod <- jags.model(div.jags.mod,
  data = jags_dat, inits = jags_inits,
  n.chains = nc, n.adapt = na)
update(rjags_mod, nb)
out <- jags.samples(rjags_mod, param,
  ni, thin = 4)
save(out, file = "Processed data/div_out.rda")
```

Post-processing the model output

```
load("Processed data/div_out.rda")
source("Functions/hpd.R")
```

Some model convergence diagnostics (need values < 1.2)

```
gelman.diag(out[["sigma_overall"]],
  multivariate = F)

## Potential scale reduction factors:
##
##           Point est. Upper C.I.
## sigma_overall      1.01      1.06

gelman.diag(out[["sigma_lat"]], multivariate = F)

## Potential scale reduction factors:
##
##           Point est. Upper C.I.
## sigma_lat        1.05      1.05
```

Bayesian p-value (values close to 0.5 are good, closer to 0.7 or 0.3 are signs of discrepancies)

```
mean(out$ppc)
```

```
## [1] 0.5295
```

Extract regression coefficients for random regional intercepts

```
alphas <- lapply(seq(2, 9), function(x) {
  region <- as.character(area_descriptors$`Area Endemism`[x])
  cred_int <- hpd(out$alpha[x, ,
    ], 0.95)
  data.frame(Region = stringi::stri_trans_totitle(gsub("_",
    " ", region)), lower_alpha = cred_int[3],
    mode_alpha = cred_int[2], upper_alpha = cred_int[1])
})
```

```
do.call(rbind, alphas)
```

	Region	lower_alpha	mode_alpha	upper_alpha
## 1	Boreal Forest	-1.3884595	-0.7594635	-0.8679231
## 2	Huallaga	0.7236598	0.9909212	0.9729180
## 3	Mesoamerica	1.7164622	1.7837757	1.8155092
## 4	Pantanal	-0.7523100	-0.6955345	-0.6976408
## 5	Parkland Prairie	-2.0731158	-1.6210322	-1.7011400
## 6	Southwest	-0.8693900	-0.7092780	-0.7124570
## 7	Temperate Forest	0.3258167	0.6821221	0.6488839
## 8	Temperate Grassland	-0.6127550	-0.4102769	-0.4088088

Extract beta coefficients

```
beta_lat <- plyr::adply(out$beta_lat[,,
  1], c(1))[, -1]
beta_lat <- c(beta_lat, plyr::adply(out$beta_lat[,,
  2], c(1))[, -1])

beta_alt <- plyr::adply(out$beta_alt[,,
  1], c(1))[, -1]
beta_alt <- c(beta_alt, plyr::adply(out$beta_alt[,,
  2], c(1))[, -1])

beta_ndvi <- plyr::adply(out$beta_ndvi[,,
  1], c(1))[, -1]
beta_ndvi <- c(beta_ndvi, plyr::adply(out$beta_ndvi[,,
  2], c(1))[, -1])

beta_ndwi <- plyr::adply(out$beta_ndwi[,,
  1], c(1))[, -1]
beta_ndwi <- c(beta_ndwi, plyr::adply(out$beta_ndwi[,,
  2], c(1))[, -1])
```

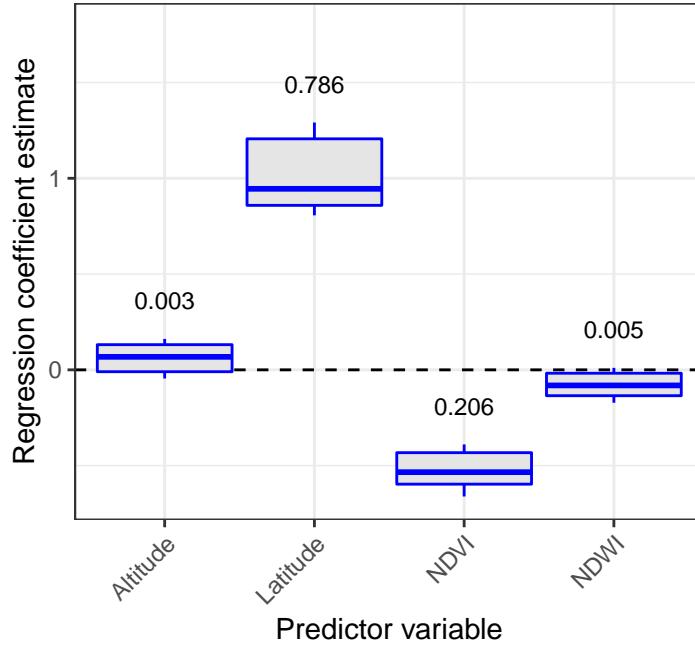
Summarise beta coefficients for plotting

```
betas <- data.frame(Estimate = c(beta_lat,
  beta_alt, beta_ndvi, beta_ndwi),
  Coefficient = c(rep("Latitude",
  length(beta_lat)), rep("Altitude",
  length(beta_alt)), rep("NDVI",
  length(beta_ndvi)), rep("NDWI",
  length(beta_ndwi)))) %>% dplyr::group_by(Coefficient) %>%
dplyr::summarise(ymin = min(Estimate),
  lower = hpd(Estimate, 0.95)[1],
  middle = hpd(Estimate, 0.95)[2],
  upper = hpd(Estimate, 0.95)[3],
  ymax = max(Estimate))

# Extract mean variance explained
# for each covariate
alt_var <- mean(out$beta_alt)^2
lat_var <- mean(out$beta_lat)^2
ndvi_var <- mean(out$beta_ndvi)^2
ndwi_var <- mean(out$beta_ndwi)^2
total_var <- alt_var + lat_var + ndvi_var +
  ndwi_var
var_exp <- c(round(alt_var/total_var,
  3), round(lat_var/total_var, 3),
  round(ndvi_var/total_var, 3), round(ndwi_var/total_var,
  3))
```

Plot to create **Figure XX**

```
library(ggplot2)
Betas_plot <- ggplot(data = betas,
  aes(x = Coefficient)) + theme_bw() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_boxplot(aes(ymin = ymin, lower = lower,
  middle = middle, upper = upper,
  ymax = ymax), stat = "identity",
  fill = "gray90", colour = "blue") +
  ylab("Regression coefficient estimate") +
  xlab("Predictor variable") + annotate("text",
  x = "Altitude", y = max(betas$ymax[1]) +
    0.2, label = sprintf("%0.3f",
    var_exp[1]), size = 3) + annotate("text",
  x = "Latitude", y = max(betas$ymax[2]) +
    0.2, label = sprintf("%0.3f",
    var_exp[2]), size = 3) + annotate("text",
  x = "NDVI", y = max(betas$ymax[3]) +
    0.2, label = sprintf("%0.3f",
    var_exp[3]), size = 3) + annotate("text",
  x = "NDWI", y = max(betas$ymax[4]) +
    0.2, label = sprintf("%0.3f",
    var_exp[4]), size = 3) + scale_y_continuous(limits = c(min(betas$ymin),
  max(betas$ymax) + 0.5)) + theme(legend.position = ""),
  theme(axis.text.x = element_text(angle = 45,
  vjust = 1, hjust = 1))
```



It is clear that NDVI and Latitude are strong predictors of community phylogenetic diversity. We can use the regional predictions of diversity for investigating these patterns

```

preds <- lapply(seq(1, 9), function(x) {
  pred <- plyr::adply(out$region_pred[x,
    , 1], c(1))[, -1]
  pred <- c(pred, plyr::adply(out$region_pred[x,
    , 2], c(1))[, -1])
  data.frame(Region = levels(region)[x],
             Estimate = pred)
})

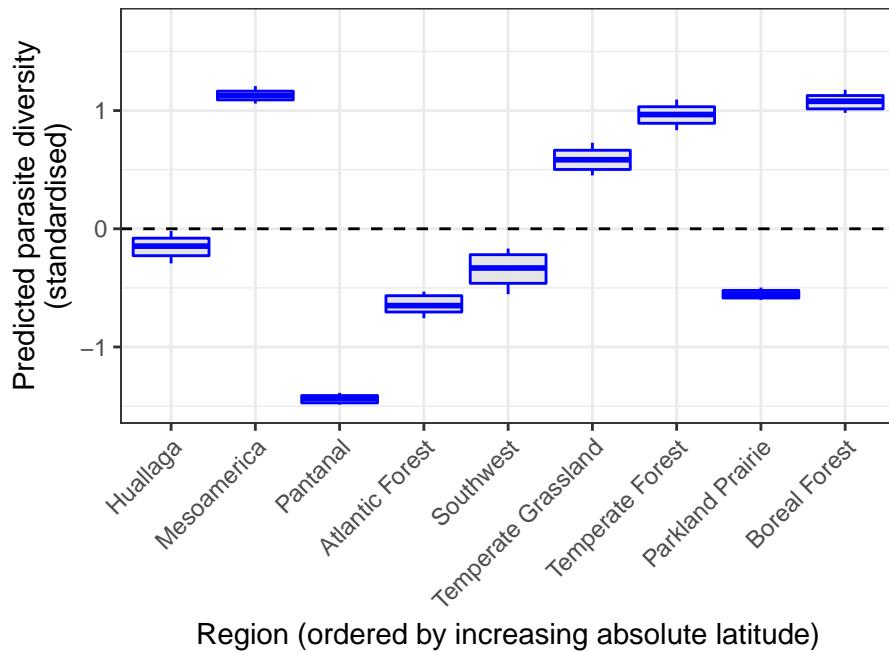
preds <- do.call(rbind, preds) %>%
  dplyr::group_by(Region) %>% dplyr::summarise(ymin = min(Estimate),
  lower = hpd(Estimate, 0.95)[1],
  middle = hpd(Estimate, 0.95)[2],
  upper = hpd(Estimate, 0.95)[3],
  ymax = max(Estimate)) %>% dplyr::mutate(Region = stringi::stri_trans_totitle(gsub("_",
  " ", Region)))

preds$Latitude <- area_descriptors$mean_lat
preds$NDVI <- area_descriptors$mean_ndvi
preds$Region <- factor(preds$Region,
  levels = preds$Region[order(preds$Latitude)])

```

Arranging regions by `Latitude` and plotting 95% HPD intervals allows us to visualise how diversity varies over latitudinal gradients

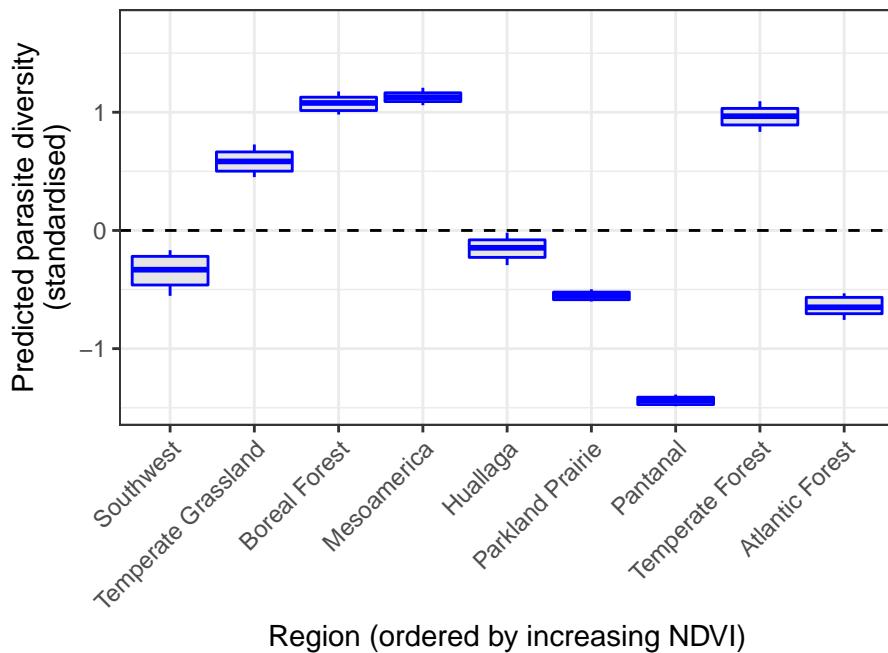
```
regions_lat_plot <- ggplot(data = preds,
  aes(x = Region)) + theme_bw() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_boxplot(aes(ymin = ymin, lower = lower,
    middle = middle, upper = upper,
    ymax = ymax), stat = "identity",
    fill = "gray90", colour = "blue") +
  ylab("Predicted parasite diversity\n(standardised)") +
  xlab("Region (ordered by increasing absolute latitude)") +
  scale_y_continuous(limits = c(min(preds$ymin),
    max(preds$ymax) + 0.5)) + theme(legend.position = "") +
  theme(axis.text.x = element_text(angle = 45,
    vjust = 1, hjust = 1))
```



There is a trend of increasing diversity at higher absolute latitudes, though some regions such as **Mesoamerica** and **Parkland Prairie** buck this trend. What about across NDVI gradients?

```
preds$Region <- factor(preds$Region,
  levels = preds$Region[order(preds$NDVI)])
regions_ndvi_plot <- ggplot(data = preds,
  aes(x = Region)) + theme_bw() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  geom_boxplot(aes(ymin = ymin, lower = lower,
    middle = middle, upper = upper,
    ymax = ymax), stat = "identity",
    fill = "gray90", colour = "blue") +
  ylab("Predicted parasite diversity\n(standardised)") +
  xlab("Region (ordered by increasing NDVI)") +
  scale_y_continuous(limits = c(min(preds$ymin),
    max(preds$ymax) + 0.5)) + theme(legend.position = "") +
  theme(axis.text.x = element_text(angle = 45,
```

```
vjust = 1, hjust = 1))
```



Communities are moderately less diverse with increasing summer vegetation cover, though this trend is not as clear as the latitudinal diversity gradient.

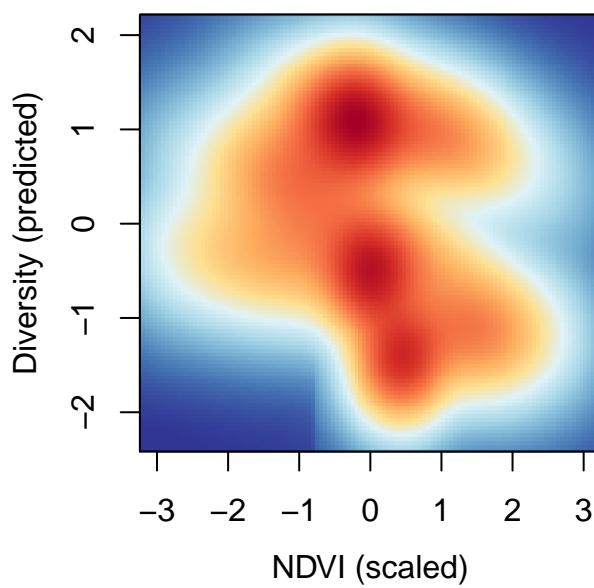
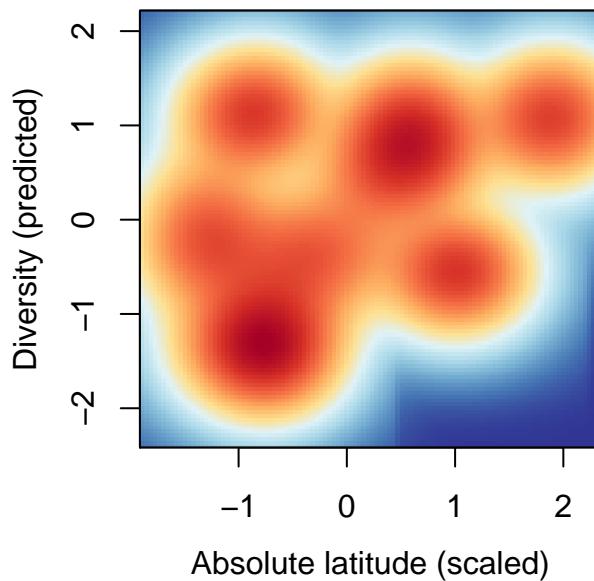
We can use bivariate heatmaps as another approach for plotting these two important covariates

```
library(RColorBrewer)
`Diversity (predicted)` <- c(unlist(c(plyr::adply(out$sim[, 1], c(1)))[-1]), unlist(c(plyr::adply(out$sim[, 2], c(1)))[-1]))

`Absolute latitude (scaled)` <- as.vector(scale(c(unlist(c(plyr::adply(out$latitude[, 1], c(1)))[-1]), unlist(c(plyr::adply(out$latitude[, 2], c(1)))[-1]))))

`NDVI (scaled)` <- as.vector(scale(c(unlist(c(plyr::adply(out$ndvi[, 1], c(1)))[-1]), unlist(c(plyr::adply(out$ndvi[, 2], c(1)))[-1]))))

my_cols <- rev(brewer.pal(10, "RdYlBu"))
par(mfrow = c(2, 1), mar = c(4, 4, 1.5, 1.5))
smoothScatter(x = `Absolute latitude (scaled)`, y = `Diversity (predicted)`, nrpoints = 0, bandwidth = c(0.35, 0.3), colramp = colorRampPalette(my_cols), mgp = c(2.5, 1, 0))
smoothScatter(x = `NDVI (scaled)`, y = `Diversity (predicted)`, bandwidth = c(0.35, 0.3), nrpoints = 0, colramp = colorRampPalette(my_cols), mgp = c(2.5, 1, 0), xlim = c(-3, 3))
```



Here it is more clear that **Latitude** is a stronger predictor of community diversity