Appendix II: *Garrulus glandarius* case study

Contents

1 Required packages

In the first step, we load all of the required packages. Note that grainchanger (the package developed for this paper) can be installed using:

```
devtools::install_github("laurajanegraham/grainchanger")
```

Other options in this section set the order and labels for the variables for plotting and tables.

```r
library(captioner)
library(jtools)
library(car)
library(perturb)
library(grainchanger)
library(raster)
library(rgdal)
library(rgeos)
library(sf)
library(knitr)
library(GGally)
library(e1071) # optimising transformations (skewness function)
library(broom)
library(MuMIn)
library(stringr)
library(DHARMa)
library(cowplot)
library(tidyverse)

source("R/ca_glm.R")

# define logit and inv logit functions
logit <- function(value, eps) {
  log((value + eps)/(1 - value + eps))
}

inv.logit <- function(value, eps) {
  eps <- (1-2*eps)
  (eps*(1+exp(value))+(exp(value)-1))/(2*eps*(1+exp(value)))
}

# set up plotting options
theme_set(theme_bw(base_size = 7) + theme(strip.background = element_blank(),
                             panel.grid.major = element_blank(),
                             panel.grid.minor = element_blank()))

# set up some options
# for exploratory plotting
varorder <- c("index10", "winshannon", "lsshannon", "habitat", "urban", "bio1")

varlabel <- c("Abundance (2010 Atlas)", "MW Shannon\n(1km)", "LS Shannon",
              "Forest %", "Urban %", "Temperature")

# for initial tables from models
coefforder <- c("(Intercept)", "winshannon", "lsshannon", "habitat",
               "winshannon:bio1", "lsshannon:bio1",
               "urban", "bio1")

coefflabel <- c("Intercept", "MW Shannon (1km)", "LS Shannon", "Forest %",
                    "MW Shannon : Temperature", "LS Shannon : Temperature",
                    "Urban %", "Temperature")

# required for captioning and numbering tables and figures
tabs <- captioner(prefix = "Table AII.")
figs <- captioner(prefix = "Figure AII.")
```

2 Data

Note that file paths to data sources are hard coded. These will need updating to match folder structure. A search for ~/ in the document will find these.

2.1 Biological data

First we need to load in and spatialise the Jay (*Garrulus glandarius*) data. These were provided by Simon Gillings at the BTO. The data includes the relative abundance indices for the 1990 Atlas (index90) and for the 2010 Atlas (index10). We are using the 2010 Atlas data.

```r
jay_sp <- read_sf("~/DATA/ADMINISTRATIVE/bng/10km_grid_region.shp") %>%
  rename(grid = TILE_NAME) %>%
  inner_join(read_csv("~/DATA/BIOLOGICAL/bto_jays/extract.csv") %>%
               rename(grid = tenkm)) %>%
  select(index10, geometry) %>%
  filter(!is.na(geometry))
```

2.2 Environmental data

The data for which we want to use the upscaling approach on is Land Cover Map 2007, which is the closest match to the 2010 relative abundance index for jays. We will use the moving window to upscale diversity of the two used habitats: Broadleaf and Coniferous forest (LCM codes 1, 2). Eurasian jays use a combination of these habitats: broadleaf for foraging, coniferous for nesting. We will calculate Shannon Diversity on just these two habitats to create a measure of the landscape structure used by jays. We will calculate Shannon diversity using the moving window approach at the 1km scale and without the moving window at the 10km scale. As covariates, we will calculate forest (habitat) cover percentage, urban land cover percentage and from Worldclim mean annual temperature (bio1).

```r
lcm <- raster("~/DATA/LULC/lcm2007/lcm2007_25m_gb.tif")
forest <- c(1, 2) # these are the two forest classes
urban <- c(22, 23)

bng <- as(jay_sp, 'Spatial')

# shannon via moving window
strt <- Sys.time()
bng$winshannon <- winmoveR::winmove_upscale(grid = bng, dat = lcm, radius = 1000,
                                            type = "rectangle", fn = "diversity",
                                            lc_class = forest)
run_time <- difftime(Sys.time(), strt, units = "mins")

save(run_time, file = "results/jays_runtime.Rda")

# shannon without moving window
bng$lsshannon <- nomove_upscale(grid = bng, dat = lcm,
                                fn = "diversity", lc_class = forest)

# other measures from lcm
bng$habitat <- nomove_upscale(grid = bng, dat = lcm, fn = "prop", lc_class = forest)
bng$urban <- nomove_upscale(grid = bng, dat = lcm, fn = "prop", lc_class = urban)

# Bioclimatic variables from worldclim
wc_bio <- getData('worldclim', var = 'bio', path = '~/DATA/CLIMATE/worldclim/', res=5)
bng_pts <- spTransform(gCentroid(bng, byid=TRUE), crs(wc_bio))
jay_sp <- st_as_sf(bng) %>% bind_cols(raster::extract(wc_bio, bng_pts) %>% as.tibble)

save(jay_sp, file="results/jays_covariates.Rda")
```

Upscaling 25m resolution land-cover data for Europe to 10km degrees resolution using a 1km radius window took: 106.7 minutes.

2.3 Exploration and transformation

We have removed cells with 0 abundance because we are only really interested in predicting abundance presuming the species is present. Alternatively we could use all data and a hurdle model, however we chose to take the simpler approach.

```r
load("results/jays_covariates.Rda")
jay_sp <- jay_sp %>% filter(index10 != 0) %>% na.omit
jay_df <- jay_sp %>% as.tibble %>% select(varorder) %>% mutate(bio1 = bio1/10)
# /10 is due to way that worldclim stores the temperature data
```

What do the variables look like spatially?

```r
jay_response <- jay_sp %>%
  mutate_at(.vars = vars(-index10, -geometry), .funs = funs(scale)) %>%
  gather(variable, value, -geometry) %>%
  mutate(facet = "Relative abundance index")

jay_plot <- ggplot(jay_response %>% filter(variable == "index10")) +
  geom_sf(aes(fill = value), colour = NA) +
  coord_sf(crs = st_crs(jay_response), datum = NA) +
  scale_fill_viridis_c(name = "", option = "magma") +
  facet_wrap(~facet) +
  theme(axis.text = element_blank(), axis.line = element_blank(),
        axis.ticks = element_blank(),
        legend.position = "bottom", legend.title.align = 0.5,
        legend.key.height=unit(6,"points"), legend.key.width = unit(1.5, "line"),
        panel.border = element_blank())

jay_covs <- jay_response %>%
  filter(variable %in% varorder[-1]) %>%
  mutate(variable = factor(variable, levels = varorder, labels = varlabel))

cov_plot <- ggplot(jay_covs) +
  geom_sf(aes(fill = value), colour = NA) +
  coord_sf(crs = st_crs(jay_covs), datum = NA) +
  scale_fill_viridis_c(name = "") + facet_wrap(~variable) +
  theme(axis.text = element_blank(), axis.line = element_blank(),
        axis.ticks = element_blank(),
        legend.position = "bottom", legend.title.align = 0.5,
        legend.key.height=unit(6,"points"), legend.key.width = unit(2, "line"),
        panel.border = element_blank())

plot_grid(jay_plot, cov_plot, labels = c("a)", "b)"),
                          label_size = 7, rel_widths = c(1, 1.5))
```

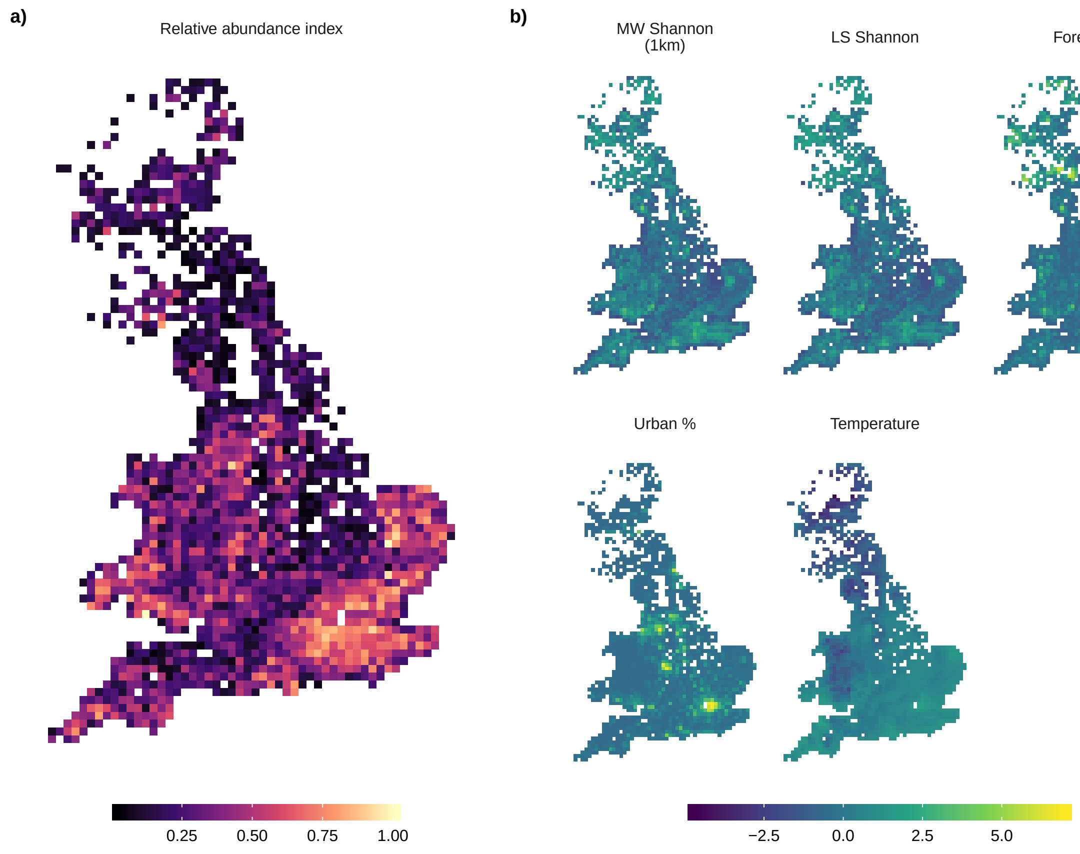

Figure AII. 1: A case study of the effect of habitat structure on Garrulus glandarius abundance. Study area showing the spatial distribution of (a) relative abundance index for G. glandarius and (b) scaled covariates (mean = 0; standard deviation = 1).

How are the variables distributed and where are the correlations?

```r
ggpairs(
  jay_df %>% select(varorder),
  upper = list(
    continuous = wrap('cor', method = "spearman")
  ),
  lower = list(
    continuous = wrap('points', alpha = 0.3, size =0.1)
  ),
  columnLabels = varlabel
)
```

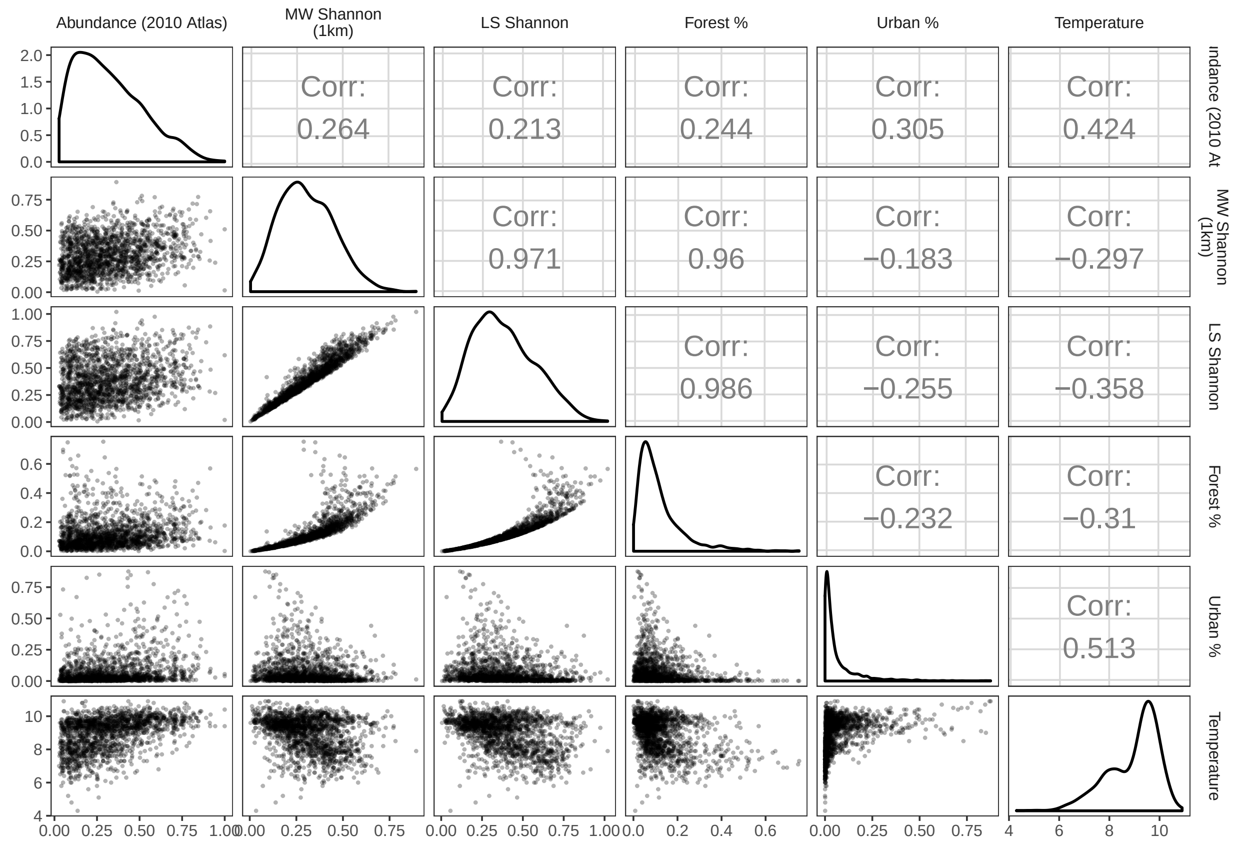

Figure AII. 2: Pairs plot of *Garrulus glandarius* relative abundance and covariates. Correlations shown are Spearman's ρ.

MW Shannon a stronger correlate of relative abundance than LS Shannon. Although both Shannon measures correlate with the amount of habitat, MW Shannon is least correlated with this (still very high). Right skew to habitat percentage, urban percentage and precipitation, and left skew to temperature.

Despite skew in the data, we have not transformed these variables because model assumptions are met without doing so (ease of interpretation). Because the abundance index is proportional, we transform this using a logit transform with the smallest non-zero value added to the numerator and denominator due to presence of 1s in data:

```
# smallest relative abundance
eps <- min(jay_df$index10)
jay_df_t <- mutate(jay_df,
                   index10logit = logit(index10, eps))

jay_narrow_t <- gather(jay_df_t, variable, value) %>%
  mutate(variable = factor(variable,
         levels = varorder, labels = varlabel))

# get mean and sd values from the transformed data for back-scaling
means <- jay_df_t %>%
  summarise_all(funs(mean, sd)) %>%
  gather() %>%
  separate(key, into=c("covariate", "measure")) %>%
  spread(measure, value)
```

The abundance score for two cells was equal to 1, so the smallest non-zero percentage response (0.03) was added to the index before applying the logit function to avoid divide by zero issues.

We scaled the data (mean = 0, SD = 1) to make the covariates comparable due to the wide range in measurement scales.

```
scale_this <- function(x) as.vector(scale(x))
jay_df_t <- mutate_at(jay_df_t, .vars = vars(-index10logit, -index10), .funs = funs(scale_this))
```

3 Statistical model

Our sample size is n = 1719.

Our model contains the following variables: MW Shannon, LS Shannon, Forest %, Urban % and Temperature. We are also including the interaction term between Temperature and MW & LS Shannon.

```
mod_global <- lm(index10logit ~ winshannon + lsshannon + habitat + urban + bio1 +
                   bio1:lsshannon + bio1:winshannon,
                 data = jay_df_t, na.action = na.fail)

res_global <- mod_global %>% coef %>% enframe(name = "variable", value = "coef") %>%
  left_join(mod_global %>% confint %>% as_tibble(rownames = "variable")) %>%
  mutate(fvariable = factor(variable, levels = coefforder, labels = coefflabel)) %>%
  arrange(fvariable)

# check this calculation of R2
globalr2 = glance(mod_global)$r.square

res_global %>% select(fvariable, coef, `2.5 %`, `97.5 %`) %>% kable(digits=3)
```

fvariable	coef	2.5 %	97.5 %
Intercept	-0.848	-0.889	-0.808
MW Shannon (1km)	0.457	0.293	0.621
LS Shannon	-0.031	-0.213	0.150
Forest %	-0.014	-0.087	0.059
MW Shannon : Temperature	0.344	0.217	0.471
LS Shannon : Temperature	-0.294	-0.427	-0.161
Urban %	0.135	0.097	0.173
Temperature	0.499	0.455	0.544

Table AII. 1: Results of linear regression for the full model

Partial regression coefficient for window-based Shannon is larger than the landscape-based Shannon coefficient, habitat (marginally) and the interaction terms.

The model explains 36.95% of the variance in jay abundance.

4 Model validation

```
jay_df_t$resids <- mod_global$residuals
jay_df_t$fitted <- mod_global$fitted.values
```

```r
# fitted values on the data scale
fitted <- inv.logit(jay_df_t$fitted, eps)

plot_grid(
  ggplot(jay_df_t, aes(x = fitted, y = resids)) +
    geom_point(size = 0.1) + geom_hline(yintercept = 0),
  ggplot(jay_df_t, aes(x = winshannon, y = resids)) +
    geom_point(size = 0.1) + geom_hline(yintercept = 0) + xlab("MW Shannon"),
  ggplot(jay_df_t, aes(x = lsshannon, y = resids)) +
    geom_point(size = 0.1) + geom_hline(yintercept = 0) + xlab("LS Shannon"),
  ggplot(jay_df_t, aes(x = habitat, y = resids)) +
    geom_point(size = 0.1) + geom_hline(yintercept = 0) + xlab("Forest %"),
  ggplot(jay_df_t, aes(x = urban, y = resids)) +
    geom_point(size = 0.1) + geom_hline(yintercept = 0) + xlab("Urban %"),
  ggplot(jay_df_t, aes(x = bio1, y = resids)) +
    geom_point(size = 0.1) + geom_hline(yintercept = 0) + xlab("Temperature")
)
```

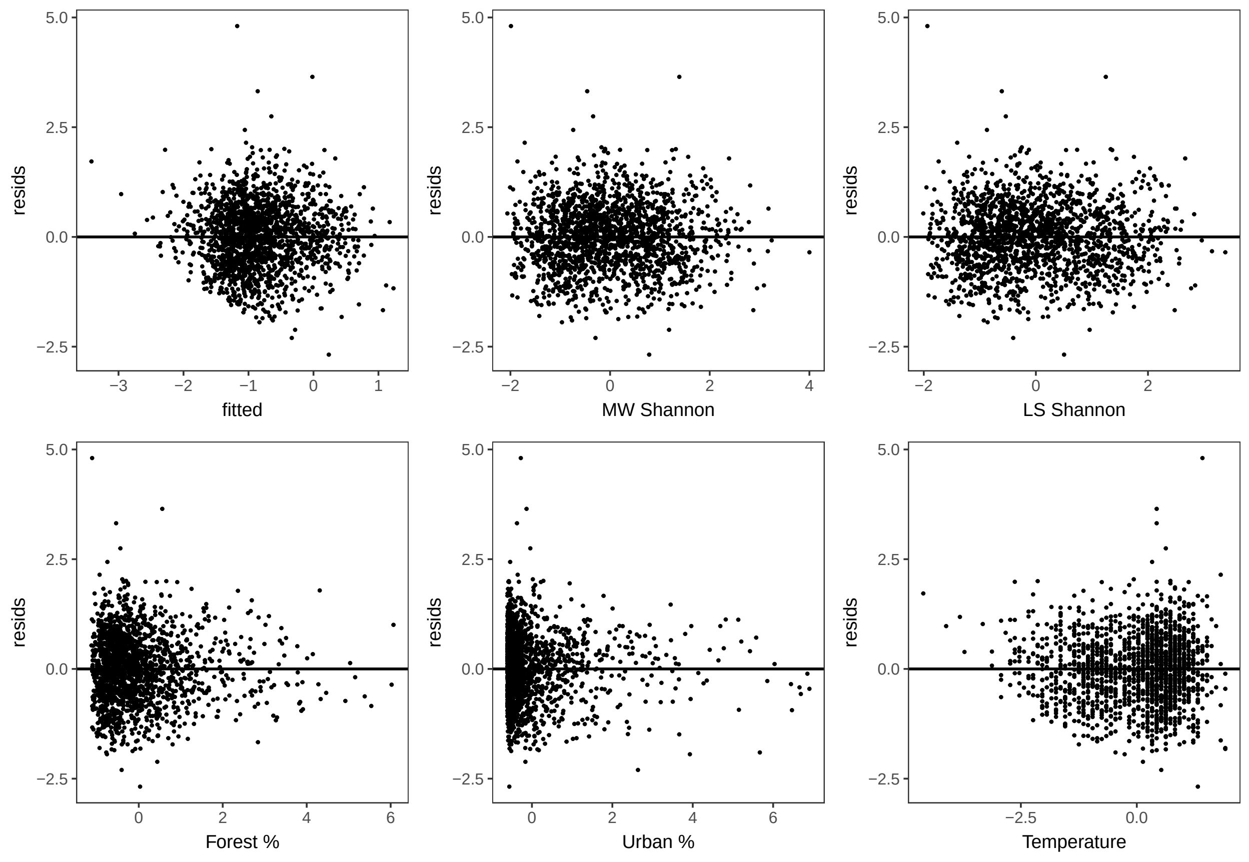

Figure AII. 3: Residuals plotted against fitted values and all predictors

There is some patterning in the residuals, but overall a reasonable model fit. The lowest fitted value is 0 and the highest is 0.79. Much better conformity to assumptions and range of predicted values than either when fit using untransformed response with either Gaussian or binomial distribution.

5 Collinearity diagnostics

Due to the high correlations between the two Shannon measures and Forest %, we will use some in-depth collinearity diagnostics

1. Variance inflation factor (not overly relevant due to large sample size)
2. Condition index and variance decomposition
3. Stability under perturbation analysis
4. Stability under data sub-sampling

5.1 Variance inflation factor

```
vif(mod_global) %>% kable(digits = 2, col.names = c("VIF"))
```

	VIF
winshannon	20.52
lsshannon	25.00
habitat	4.03
urban	1.12
bio1	1.48
lsshannon:bio1	14.27
winshannon:bio1	12.59

Table AII. 2: Variance inflation factors for each term in the global model

Although these variance inflation factors may seem high, due to the large sample size (and that we mostly still detect effects) we do not see these as problematic; although it may explain why we do not detect an effect of LS Shannon. See O'Brien 2007 for warnings on rule-of-thumb application of multicollinearity diagnostics.

5.2 Condition index and variance decomposition

See Callaghan and Chen 2008 for information on how to interpret condition indices and variance decomposition (but use D. Belsley, E. Kuh, and R. Welsch (1980). Regression Diagnostics. Wiley. as citation). In short, rows in the below table with a high condition index (first column, high is > 10 - moderate to strong collinearity, > 30 - severe collinearity) which are associated with high variance of regression estimate (the rest of the table) will cause a problem in the analysis.

```
mod_diag <- colldiag(mod_global) %>% lapply(as.data.frame)
mod_diag <- do.call("cbind", mod_diag)
names(mod_diag) <- c("CI", "Intercept", "MW Shannon", "LS Shannon",
                     "Forest %", "Urban %", "Temp")
kable(mod_diag, digits = 3)
```

CI	Intercept	MW Shannon	LS Shannon	Forest %	Urban %	Temp
1.000	0	0.006	0.005	0.024	0.009	0.023
1.642	0	0.003	0.001	0.006	0.490	0.187
1.731	1	0.000	0.000	0.000	0.000	0.000
2.148	0	0.002	0.000	0.001	0.500	0.683
3.730	0	0.065	0.018	0.848	0.000	0.016
10.478	0	0.925	0.976	0.121	0.001	0.091

Table AII. 3: Condition index and variance decomposition

The highest condition index is 10.48; this puts us in the “weak-to-moderate collinearity” range. There is high variance decomposition associated with the MW & LS Shannon (particularly LS) coefficient estimates. We will check the effect of this with some further tests on the data.

5.3 Stability under perturbation analysis

We will add some random noise to the two environmental heterogeneity measures to evaluate collinearity.

```r
attach(jay_df_t)
perturb_mod <- perturb(mod_global, pvars = c("winshannon", "lsshannon"), prange=c(0.1,0.1))
detach(jay_df_t)

perturb_mod$coeff.table %>% as.tibble %>%
  gather(variable, value) %>%
  group_by(variable) %>%
  summarise(mean_val = mean(value),
            sd_val = sd(value),
            min_val = min(value),
            max_val = max(value)) %>%
  mutate(variable = factor(variable, levels = coefforder,
                           labels = gsub("\n", " ", coefflabel))) %>%
  arrange(variable) %>%
  kable(digits = 3)
```

variable	mean_val	sd_val	min_val	max_val
Intercept	-0.839	0.003	-0.848	-0.831
MW Shannon (1km)	0.365	0.043	0.231	0.480
LS Shannon	0.055	0.047	-0.068	0.198
Forest %	-0.006	0.007	-0.022	0.014
MW Shannon : Temperature	0.268	0.031	0.182	0.348
LS Shannon : Temperature	-0.213	0.033	-0.294	-0.123
Urban %	0.136	0.002	0.129	0.139
Temperature	0.497	0.004	0.486	0.507

Table AII. 4: Results of perturbation analysis. Random noise was added to the two Shannon measures to the range of 0.1

Again, there may be some issues with LS Shannon.

5.4 Stability under data sub-sampling

Now let’s test this by checking the model results are stable under sub-sampling.

NB This will reduce the power to detect significant relationships (increase Type II errors).

```r
model_subsample <-function(x, dat, n) {

  dat <- sample_n(dat, n)

  mod<- lm(index10logit ~ winshannon + lsshannon + urban + bio1 +
                 bio1:lsshannon + bio1:winshannon,
                data = dat, na.action = na.fail)

  res <- mod %>% coef %>% enframe(name = "variable", value = "coef") %>%
    left_join(mod %>% confint %>% as_tibble(rownames = "variable")) %>%
    mutate(fvariable = factor(variable, levels = coefforder,
                              labels = gsub("\n", " ", coefflabel))) %>%
    arrange(fvariable)

  r2 <- glance(mod)$r.square

  res %>% select(fvariable, coef, `2.5 %`, `97.5 %`) %>% mutate(R2 = r2)
}

n <- 1500
nsim <- 100
out <- map_dfr(1:nsim, ~model_subsample(.x, jay_df_t, n))

out_narrow <- out %>% gather(measure, value, -R2, -fvariable) %>%
  mutate(measure = factor(measure, labels = c("LCI", "UCI", "Coefficient")),
         measure = factor(measure, levels = c("LCI", "Coefficient", "UCI"))) %>%
  filter(grepl("Shannon", fvariable))

res_narrow <- res_global %>% gather(measure, value, -fvariable, -variable) %>%
  mutate(measure = factor(measure, labels = c("LCI", "UCI", "Coefficient")),
         measure = factor(measure, levels = c("LCI", "Coefficient", "UCI"))) %>%
  filter(grepl("Shannon", fvariable))
```

```r
ggplot(out_narrow, aes(x = "fvariable", y = value)) +
  geom_jitter(size = 0.1) +
  facet_grid(measure~fvariable) +
  geom_hline(data = res_narrow, aes(yintercept = value), colour = "red") +
  labs(x = "Variable", y = "Coefficient estimate") +
  theme(axis.text.x=element_blank())
```

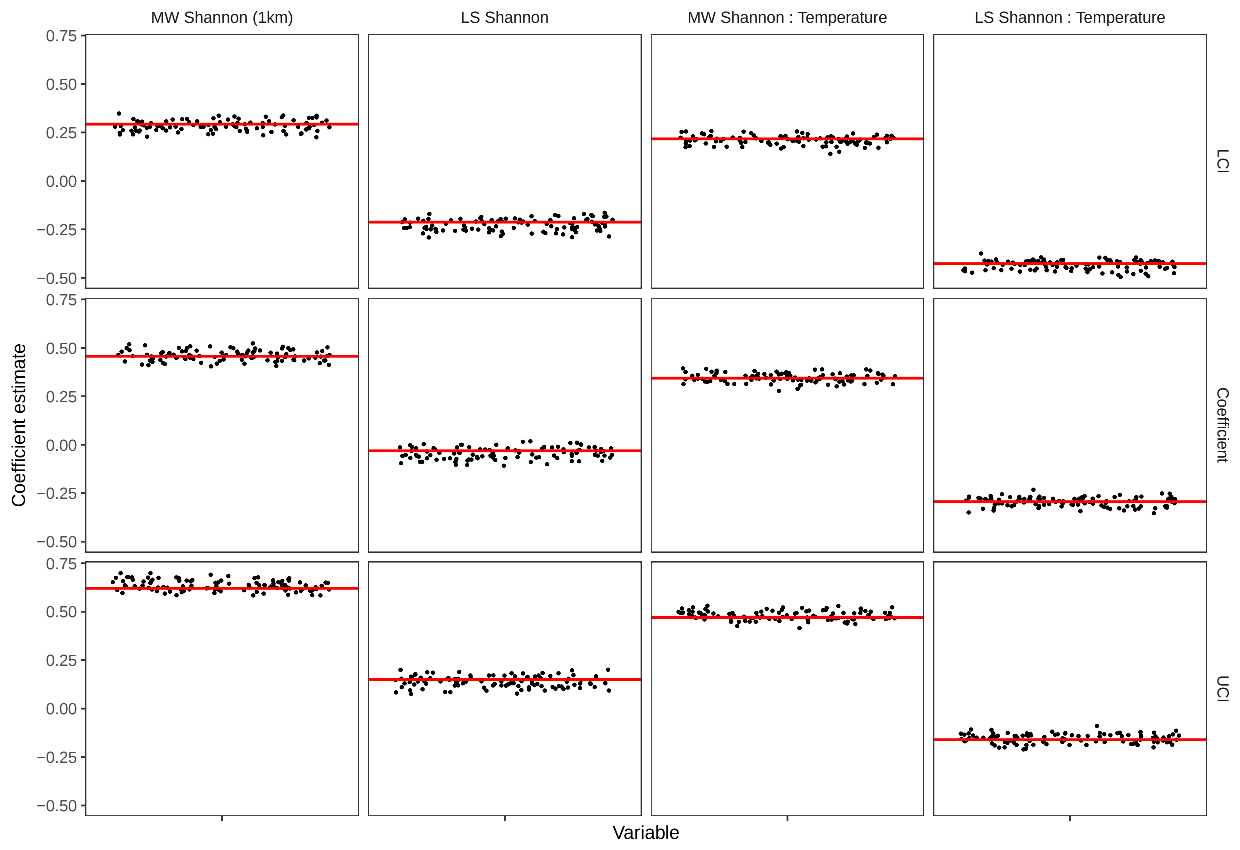

Figure AII. 4: Results of subsampling analysis. For 100 replicates, 1500 points were subsampled from the total dataset and the regression calculated. Red lines show the value of the coefficient when the regression is calculated using the full dataset

In Figure AII. 4, each point is the estimate for the model parameterised with one of the 100 samples of n = 1500 from the full dataset. The red line shows the estimate from the model parameterised using the full dataset (n = 1719). The estimates are reasonably robust to sub-sampling the data (not much variation in estimates and very few changes to non-significant). As is R^2.

```
ggplot(data = out, aes(x = R2)) +
  geom_histogram() +
  geom_vline(xintercept = globalr2)
```

Figure AII. 5: R^2 distribution for each replicate in the subsampling analysis.

5.5 Collinearity in regression coefficients

Finally, we need to check that we do not have extreme collinearity in the parameter estimates.

```
coef_cov <- cov2cor(vcov(mod_global))

coef_cov%>%
  as.tibble(rownames = "variable") %>%
  kable(digits = 2)
```

variable	(Intercept)	winshannon	lsshannon	habitat	urban	bio1	lsshannon:bio1	winshannon:bio1
(Intercept)	1.00	-0.18	0.14	0.12	0.01	-0.12	0.37	-0.27

variable	(Intercept)	winshannon	lsshannon	habitat	urban	bio1	lsshannon:bio1	winshannon:bio1
winshannon	-0.18	1.00	-0.92	0.01	-0.04	-0.12	-0.45	0.41
lsshannon	0.14	-0.92	1.00	-0.34	0.04	0.18	0.37	-0.36
habitat	0.12	0.01	-0.34	1.00	0.00	-0.03	0.18	-0.12
urban	0.01	-0.04	0.04	0.00	1.00	-0.26	0.02	-0.01
bio1	-0.12	-0.12	0.18	-0.03	-0.26	1.00	-0.24	0.19
lsshannon:bio1	0.37	-0.45	0.37	0.18	0.02	-0.24	1.00	-0.96
winshannon:bio1	-0.27	0.41	-0.36	-0.12	-0.01	0.19	-0.96	1.00

Table AII. 5: Pairwise correlations between coefficient estimates from the full model

Here we have problems. MW Shannon and LS Shannon coefficient estimates, as well as the interaction terms, are extremely correlated. Note that this is not necessarily an issue because we would not want to estimate both in reality.

6 Commonality analysis

Commonality analysis allows us to understand the importance of different variables despite collinearity in the model. This method also does not suffer from issues such as the order of covariate entry which affects stepwise regression techniques. See Kraha et al. 2012 and Ray-Mukherjee et al. 2014 for explanations of this approach. For each variable we will calculate:

1. *Beta coefficients* the standardised partial regression coefficients
2. *Structural coefficients* Squared Pearson correlation between the variable and the fitted values from the model
3. *Unique variance* Amount of variance uniquely accounted for by the variable
4. *Common variance* Amount of variance in common with other variables
5. *Total variance* Total amount of variance accounted for by the variable

```r
ca_table <- res_global %>%
  filter(fvariable != "Intercept") %>%
  # structural coefficients
  mutate(r_s = map_dbl(variable, function(x) {
    out <- cor(model.matrix(mod_global)[,x], mod_global$fitted.values)
  }),
  r_s2 = r_s^2) %>%
  # unique, common and total variance
  inner_join(calc_commonality(mod_global)$CCTotalbyVar %>%
               as.tibble(rownames = "variable")) %>%
  select(fvariable, beta = coef, `2.5 %`, `97.5 %`, r_s, r_s2, Unique, Common, Total) %>%
  arrange(fvariable)

write_csv(ca_table, "tables/jays_commonality.csv")

kable(ca_table, digits = 3)
```

fvariable	beta	2.5 %	97.5 %	r_s	r_s2	Unique	Common	Total
MW Shannon (1km)	0.457	0.293	0.621	0.449	0.202	0.011	0.064	0.074
LS Shannon	-0.031	-0.213	0.150	0.357	0.128	0.000	0.047	0.047
Forest %	-0.014	-0.087	0.059	0.256	0.066	0.000	0.024	0.024
MW Shannon : Temperature	0.344	0.217	0.471	0.265	0.070	0.010	0.016	0.026
LS Shannon : Temperature	-0.294	-0.427	-0.161	0.293	0.086	0.007	0.025	0.032

fvariable	beta	2.5 %	97.5 %	r_s	r_s2	Unique	Common	Total
Urban %	0.135	0.097	0.173	0.391	0.153	0.018	0.039	0.057
Temperature	0.499	0.455	0.544	0.676	0.457	0.182	-0.013	0.169

Table AII. 6: Commonality analysis for the global model

LS Shannon has no unique variance explained associated. The structure coefficient is much larger than the beta, which suggests suppression (or, in reality, that it's measuring the same thing as MW Shannon, but less effectively). MW Shannon explains 6.35% of the total variance in the data (more than the 4.71 explained by LS Shannon). Temperature explains by far the most variance in the data (3.89). We will also get rid of Forest % because it explains very little of the variance, has no unique variance explained, and was insignificant in the model.

7 Final results and plots

The results we will display along with the commonality analysis will be the results of a MW-only version of the full model. We have also removed Forest % because this also did not uniquely explain any variance and was adding to potential collinearity issues.

```r
mod_mw <- lm(index10logit ~ urban + winshannon * bio1,
                 data = jay_df_t, na.action = na.fail)

res_mw <- mod_mw %>% coef %>% enframe(name = "variable", value = "coef") %>%
  left_join(mod_mw %>% confint %>% as_tibble(rownames = "variable")) %>%
  mutate(fvariable = factor(variable, levels = coefforder,
                             labels = gsub("\n", " ", coefflabel))) %>%
  arrange(fvariable)

mwr2 <- glance(mod_mw)$r.square

res_mw %>% select(fvariable, coef, `2.5 %`, `97.5 %`) %>% kable(digits=3)
```

fvariable	coef	2.5 %	97.5 %
Intercept	-0.817	-0.855	-0.779
MW Shannon (1km)	0.416	0.378	0.454
MW Shannon : Temperature	0.071	0.035	0.108
Urban %	0.135	0.097	0.174
Temperature	0.463	0.423	0.503

Table AII. 7: Results of linear regression for the final reduced model

The model explains 36.14% of the variance in jay abundance.

For those variables involved in an interaction, we plot the interaction plot; for those whose main effects we wish to interpret, we plot the effect plot.

```r
# need to predict from the model then convert to the original scales
pred_vals <- bind_rows(
  make_predictions(mod_mw, pred = "urban", interval = TRUE)$predicted %>%
    select(index10logit, ymax, ymin, pred = urban) %>%
    mutate(covariate = "urban"),
  make_predictions(mod_mw, pred = "bio1", modx = "winshannon", interval = TRUE)$predicted %>%
    select(index10logit, ymax, ymin, modx_group, pred = bio1) %>%
    mutate(covariate = "bio1", modx = "winshannon"),
  make_predictions(mod_mw, pred = "winshannon", modx = "bio1", interval = TRUE)$predicted %>%
    select(index10logit, ymax, ymin, modx_group, pred = winshannon) %>%
    mutate(covariate = "winshannon", modx = "bio1")) %>%
  as.tibble %>%
  mutate_at(c("index10logit", "ymin", "ymax"), inv.logit, eps) %>%
  left_join(means) %>%
  left_join(means, by = c("modx" = "covariate"), suffix = c("_pred", "_modx")) %>%
  mutate(pred = pred*sd_pred + mean_pred,
         modx_value = case_when(modx_group == "- 1 SD" ~ mean_modx - sd_modx,
                                modx_group == "Mean" ~ mean_modx,
                                modx_group == "+ 1 SD" ~ mean_modx + sd_modx,
                                TRUE ~ 0),
         modx_label = factor(paste0(modx_group, " (", round(modx_value, 2), ")")),
         modx_label = fct_reorder(modx_label, modx_value))

# plot the results
winshannon_plot <- ggplot(pred_vals %>% filter(covariate == "winshannon"),
                          aes(x = pred, y = index10logit, group = modx_label)) +
  geom_line(aes(lty = modx_label)) +
  scale_linetype_manual(name = expression("Temperature (" * degree * "C)"),
                        values = c("dashed", "solid", "dotdash")) +
  scale_y_continuous(limits = c(0, 1), breaks = c(0, 0.25, 0.5, 0.75, 1)) +
  geom_ribbon(aes(ymin = ymin, ymax = ymax), alpha = 0.2) +
  theme(legend.position = c(0.25, 0.8), legend.key.height = unit(0.1, "line")) +
  labs(x = "MW Shannon (1km)",
       y = expression(paste(italic("G. glandarius"),
                            " abundance index (" %+-% "95% CI)")))

bio1_plot <- ggplot(pred_vals %>% filter(covariate == "bio1"),
                    aes(x = pred, y = index10logit, group = modx_label)) +
  geom_line(aes(lty = modx_label)) +
  scale_linetype_manual(name = "MW Shannon (1km)",
                        values = c("dashed", "solid", "dotdash")) +
  scale_y_continuous(limits = c(0, 1), breaks = c(0, 0.25, 0.5, 0.75, 1)) +
  geom_ribbon(aes(ymin = ymin, ymax = ymax), alpha = 0.2) +
  theme(legend.position = c(0.3, 0.8), legend.key.height = unit(0.1, "line")) +
  labs(x = expression("Temperature (" * degree * "C)"),
       y = "")

urban_plot <- ggplot(pred_vals %>% filter(covariate == "urban"),
                     aes(x = pred, y = index10logit)) +
  geom_line() +
  scale_y_continuous(limits = c(0, 1), breaks = c(0, 0.25, 0.5, 0.75, 1)) +
  geom_ribbon(aes(ymin = ymin, ymax = ymax), alpha = 0.2) +
  labs(x = "Urban %",
       y = "")

plot_grid(winshannon_plot, bio1_plot, urban_plot, nrow = 1)
```

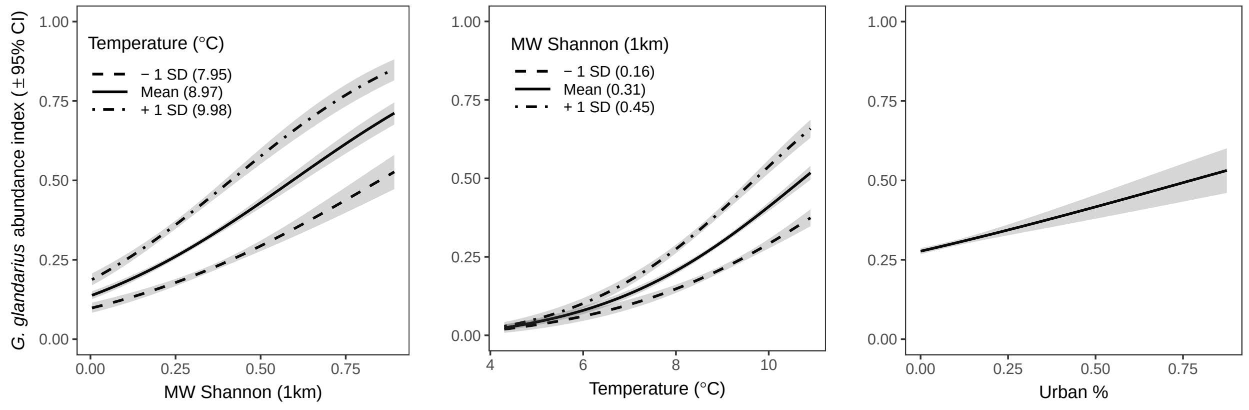

Figure AII. 6: Main effect estimates for the full model

We can also calculate the Johnson-Neyman interval to understand the range of moderator values for which the interaction term is significant. We then convert back to the measurement scale

```
sim_slopes(mod_mw,
           pred = "winshannon",
           modx = "bio1",
           johnson_neyman = TRUE)$jn
```

```
## [[1]]
## JOHNSON-NEYMAN INTERVAL
##
## When bio1 is OUTSIDE the interval [-12.18, -3.80], the slope of
## winshannon is p < .05.
##
## Note: The range of observed values of bio1 is [-4.60, 1.91]
```

```
# significant for -3.80 to 1.91 - HC'd so update if model changes
winshannon_slope <- paste(round(c(-3.80, 1.91) *
                                  filter(means, covariate == "bio1") %>% pull(sd) +
                                  filter(means, covariate == "bio1") %>% pull(mean), 2),
                          collapse = ", ")

sim_slopes(mod_mw,
           pred = "bio1",
           modx = "winshannon",
           johnson_neyman = TRUE)$jn
```

```
## [[1]]
## JOHNSON-NEYMAN INTERVAL
##
## When winshannon is OUTSIDE the interval [-13.66, -4.19], the slope of
## bio1 is p < .05.
##
## Note: The range of observed values of winshannon is [-2.06, 4.00]
```

```
# significant for full range of data - HC'd so update if model changes
temp_slope <- paste(round(c(-2.06, 4.00) *
                            filter(means, covariate == "winshannon") %>% pull(sd) +
                            filter(means, covariate == "winshannon") %>% pull(mean), 2),
                    collapse = ", ")
```

On the measurement scale, MW Shannon (1km) slope is significant in Temperature range 5.11, 10.9 (all but the lowest temperatures) and Temperature slope is significant in MW Shannon (1km) range 0, 0.89 (the full range of the data)

8 Collinearity diagnostics of the final model

We do not expect collinearity in the final model due to low correlations between variables, however we have included the variance inflation factor and condition index/variance decomposition tests to clarify.

8.1 Variance inflation factor

```
vif(mod_mw) %>% kable(digits = 2, col.names = c("VIF"))
```

	VIF
urban	1.11
winshannon	1.10
bio1	1.23
winshannon:bio1	1.03

Table AII. 8: Variance inflation factors for each term in the final model

8.2 Condition index and variance decomposition

```
mod_diag <- colldiag(mod_mw) %>% lapply(as.data.frame)
mod_diag <- do.call("cbind", mod_diag)
names(mod_diag) <- c("CI", "Intercept", "MW Shannon", "Urban %", "Temp")
kable(mod_diag, digits = 3)
```

CI	Intercept	MW Shannon	Urban %	Temp
1.000	0	0.179	0.167	0.235
1.227	1	0.000	0.000	0.000
1.318	0	0.469	0.575	0.001
1.547	0	0.351	0.258	0.764

Table AII. 9: Condition index and variance decomposition

9 Session Info

```
session <- devtools::session_info()
session[[1]]
```

```
##  setting  value
##  version  R version 3.5.0 (2018-04-23)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United Kingdom.1252
##  tz       Europe/London
##  date     2018-06-22
```

```
session[[2]] %>% kable
```

package	*	version	date	source
abind		1.4-5	2016-07-21	CRAN (R 3.5.0)
assertthat		0.2.0	2017-04-11	CRAN (R 3.5.0)
backports		1.1.2	2017-12-13	CRAN (R 3.5.0)
base	*	3.5.0	2018-04-23	local
bindr		0.1.1	2018-03-13	CRAN (R 3.5.0)
bindrcpp	*	0.2.2	2018-03-29	CRAN (R 3.5.0)
broom	*	0.4.4	2018-03-29	CRAN (R 3.5.0)
captioner	*	2.2.3.9000	2018-06-19	Github (adletaw/captioner@5f2b435)
car	*	3.0-0	2018-04-02	CRAN (R 3.5.0)
carData	*	3.0-1	2018-03-28	CRAN (R 3.5.0)
cellranger		1.1.0	2016-07-27	CRAN (R 3.5.0)
class		7.3-14	2015-08-30	CRAN (R 3.5.0)
classInt		0.2-3	2018-04-16	CRAN (R 3.5.0)
cli		1.0.0	2017-11-05	CRAN (R 3.5.0)
codetools		0.2-15	2016-10-05	CRAN (R 3.5.0)
colorspace		1.3-2	2016-12-14	CRAN (R 3.5.0)
compiler		3.5.0	2018-04-23	local
cowplot	*	0.9.2	2017-12-17	CRAN (R 3.5.0)
crayon		1.3.4	2017-09-16	CRAN (R 3.5.0)
curl		3.2	2018-03-28	CRAN (R 3.5.0)
data.table		1.11.2	2018-05-08	CRAN (R 3.5.0)
datasets	*	3.5.0	2018-04-23	local
DBI		1.0.0	2018-05-02	CRAN (R 3.5.0)
devtools		1.13.5	2018-02-18	CRAN (R 3.5.0)
DHARMa	*	0.1.6	2018-03-18	CRAN (R 3.5.0)
digest		0.6.15	2018-01-28	CRAN (R 3.5.0)
dplyr	*	0.7.4	2017-09-28	CRAN (R 3.5.0)
e1071	*	1.6-8	2017-02-02	CRAN (R 3.5.0)
evaluate		0.10.1	2017-06-24	CRAN (R 3.5.0)
forcats	*	0.3.0	2018-02-19	CRAN (R 3.5.0)
foreach		1.4.4	2017-12-12	CRAN (R 3.5.0)
foreign		0.8-70	2017-11-28	CRAN (R 3.5.0)
GGally	*	1.4.0	2018-05-17	CRAN (R 3.5.0)
ggplot2	*	2.2.1.9000	2018-05-23	Github (tidyverse/ggplot2@eecc450)
glue		1.2.0	2017-10-29	CRAN (R 3.5.0)
grainchanger	*	0.0.0.9000	2018-06-06	local (laurajanegraham/grainchanger@NA)
graphics	*	3.5.0	2018-04-23	local
grDevices	*	3.5.0	2018-04-23	local
grid		3.5.0	2018-04-23	local
gtable		0.2.0	2016-02-26	CRAN (R 3.5.0)
haven		1.1.1	2018-01-18	CRAN (R 3.5.0)
highr		0.6	2016-05-09	CRAN (R 3.5.0)
hms		0.4.2	2018-03-10	CRAN (R 3.5.0)
htmltools		0.3.6	2017-04-28	CRAN (R 3.5.0)
httr		1.3.1	2017-08-20	CRAN (R 3.5.0)
iterators		1.0.9	2017-12-12	CRAN (R 3.5.0)

package	*	version	date	source
jsonlite		1.5	2017-06-01	CRAN (R 3.5.0)
jtools	*	1.0.0	2018-05-08	CRAN (R 3.5.0)
knitr	*	1.20	2018-02-20	CRAN (R 3.5.0)
labeling		0.3	2014-08-23	CRAN (R 3.5.0)
lattice		0.20-35	2017-03-25	CRAN (R 3.5.0)
lazyeval		0.2.1	2017-10-29	CRAN (R 3.5.0)
lubridate		1.7.4	2018-04-11	CRAN (R 3.5.0)
magrittr		1.5	2014-11-22	CRAN (R 3.5.0)
Matrix		1.2-14	2018-04-13	CRAN (R 3.5.0)
memoise		1.1.0	2017-04-21	CRAN (R 3.5.0)
methods	*	3.5.0	2018-04-23	local
mnormt		1.5-5	2016-10-15	CRAN (R 3.5.0)
modelr		0.1.2	2018-05-11	CRAN (R 3.5.0)
MuMIn	*	1.40.4	2018-01-30	CRAN (R 3.5.0)
munsell		0.4.3	2016-02-13	CRAN (R 3.5.0)
nlme		3.1-137	2018-04-07	CRAN (R 3.5.0)
openxlsx		4.0.17	2017-03-23	CRAN (R 3.5.0)
parallel		3.5.0	2018-04-23	local
perturb	*	2.05	2012-02-19	CRAN (R 3.5.0)
pillar		1.2.2	2018-04-26	CRAN (R 3.5.0)
pkgconfig		2.0.1	2017-03-21	CRAN (R 3.5.0)
plyr		1.8.4	2016-06-08	CRAN (R 3.5.0)
psych		1.8.4	2018-05-06	CRAN (R 3.5.0)
purrr	*	0.2.4	2017-10-18	CRAN (R 3.5.0)
R6		2.2.2	2017-06-17	CRAN (R 3.5.0)
raster	*	2.6-7	2017-11-13	CRAN (R 3.5.0)
RColorBrewer		1.1-2	2014-12-07	CRAN (R 3.5.0)
Rcpp		0.12.16	2018-03-13	CRAN (R 3.5.0)
readr	*	1.1.1	2017-05-16	CRAN (R 3.5.0)
readxl		1.1.0	2018-04-20	CRAN (R 3.5.0)
reshape		0.8.7	2017-08-06	CRAN (R 3.5.0)
reshape2		1.4.3	2017-12-11	CRAN (R 3.5.0)
rgdal	*	1.2-20	2018-05-07	CRAN (R 3.5.0)
rgeos	*	0.3-27	2018-06-01	CRAN (R 3.5.0)
rio		0.5.10	2018-03-29	CRAN (R 3.5.0)
rlang		0.2.0	2018-02-20	CRAN (R 3.5.0)
rmarkdown		1.9	2018-03-01	CRAN (R 3.5.0)
RPostgreSQL		0.6-2	2017-06-24	CRAN (R 3.5.0)
rprojroot		1.3-2	2018-01-03	CRAN (R 3.5.0)
rstudioapi		0.7	2017-09-07	CRAN (R 3.5.0)
rvest		0.3.2	2016-06-17	CRAN (R 3.5.0)
scales		0.5.0	2017-08-24	CRAN (R 3.5.0)
sf	*	0.6-2	2018-04-25	CRAN (R 3.5.0)
sp	*	1.2-7	2018-01-19	CRAN (R 3.5.0)
spData		0.2.8.3	2018-03-25	CRAN (R 3.5.0)
stats	*	3.5.0	2018-04-23	local
stats4		3.5.0	2018-04-23	local
stringi		1.1.7	2018-03-12	CRAN (R 3.5.0)
stringr	*	1.3.1	2018-05-10	CRAN (R 3.5.0)
tibble	*	1.4.2	2018-01-22	CRAN (R 3.5.0)
tidyr	*	0.8.0	2018-01-29	CRAN (R 3.5.0)
tidyselect		0.2.4	2018-02-26	CRAN (R 3.5.0)

package	*	version	date	source
tidyverse	*	1.2.1	2017-11-14	CRAN (R 3.5.0)
tools		3.5.0	2018-04-23	local
udunits2		0.13	2016-11-17	CRAN (R 3.5.0)
units		0.5-1	2018-01-08	CRAN (R 3.5.0)
utils	*	3.5.0	2018-04-23	local
viridisLite		0.3.0	2018-02-01	CRAN (R 3.5.0)
withr		2.1.2	2018-05-15	Github (jimhester/withr@79d7b0d)
xml2		1.2.0	2018-01-24	CRAN (R 3.5.0)
yaml		2.1.19	2018-05-01	CRAN (R 3.5.0)