AI for the Environment: from AI to Ecological Models

Week 10

Rory Gibb & Ella Browning

07/03/2023

Drivers of species occurrence across the Masai Mara

Today we're exploring and analysing some camera trap data from the Masai Mara collected as part of the Biome Health project - see the week 9 lecture slides for a general summary of the data and the project, and see the week 9 workshop for an introduction to spatial data processing and GIS in R.

The goal of today's session is to investigate the distribution and drivers of occupancy for our study species (the Cape hare, *Lepus capensis*) in relation to anthropogenic and environmental factors across the Masai Mara, using the camera trap data and spatial data we processed and extracted in last week's workshop. We will explore fitting and evaluating some generalised linear models, before exploring how extending our models to include nonlinear and geospatial effects can improve our ability to infer ecological drivers. There will be code snippets with short exercises interspersed, along with some larger extension exercises at the end if you have time.

If you are working in RStudio, all the data and environmental layers you'll need for the workshop are in the GitHub, in the "9_AIToEcologicalModels" folder. Please download the whole folder and set this as your working directory, then all the materials you will need are contained within the "data" subfolder. Alternatively, there is an iPython Notebook version of this workshop in the repository that you can run on Google Colabs.

```r
# dependencies
library(dplyr); library(magrittr); library(tibble)
library(sf); library(ggplot2); library(tidyr); library(rstudioapi)
library(mgcv); library(gratia); library(lme4)
library(raster); library(terra) # only required for solutions

# automatically set file path
# (or if this doesn't work,
# manually set your working directory to the folder "9_AItoEcologicalModels")
PATH = dirname(rstudioapi::getSourceEditorContext()$path)
setwd(PATH)
```

Defining our research question

Let's start with a broad question: what is the relationship between level of anthropogenic pressure and spatial occupancy of our focal species? We can define anthropogenic pressure in many ways, but here, let's focus on livestock pressure. We know that one of the major anthropogenic activities in the Mara ecosystem is livestock grazing, which influences vegetation structure and community composition. Hares generally inhabit grassland and pastoral ecosystems, so **if habitat suitability for hares is shaped by pastoral**

activity we might expect a positive relationship with livestock. Alternatively, **if areas with very high levels of livestock activity are too frequently disturbed to provide amenable habitat, we might expect hare occurrence to decline where livestock pressure is highest**. So here we have two alternative, plausible hypotheses.

We might also need to account for other factors that may covary with our drivers of interest and also affect hare presence; here, we'll look at habitat type (proportion of agricultural and closed habitat) and distance to the nearest water body, as well as the conservancy in which the cameras were located.

Environmental and anthropogenic covariates at camera trap locations

Last week, we used the spatial locations of the camera traps along with raster data to extract information about the environmental factors around each camera trap. We have now combined all of these together into a full dataframe of site-level covariates for modelling. *In the solutions*, you can see the full code block that we used to produce this dataframe from the raw data sources, but for this exercise we will just read in the final dataframe for our analyses.

The data contain several site-level covariates: proportion of closed/semi-closed habitat or agriculture land use within a 250m radius, distance to the nearest water source, population density, conservancy, and livestock pressure. Livestock pressure was estimated from the tagged camera trap data (*see the code in the solutions*), and defined as the proportion of surveyed days in which livestock were detected.

```r
## ======== OPTIONAL: full code block to produce a covariates data frame ========

# locations and transform to metres projection
locs = read.csv("./data/kenya/survey/bh_camera_locations.csv") %>%
  sf::st_as_sf(coords = c("Longitude", "Latitude"), crs = 4326) %>%
  sf::st_transform(locs, crs = "+proj=utm +zone=36 +south +datum=WGS84 +units=m +no_defs") %>%
  dplyr::filter(CT_site != "MT34")

# add coordinates columns for XY locations
locs = cbind(locs, sf::st_coordinates(locs))

# extract habitat (land cover) types in a buffer of 250m around each trap
# specify buffer size
buffer_size = 250

# run extraction
hab = raster::raster("./data/kenya/environment/habitat/habitatfinal.tif")
raster::values(hab)[ raster::values(hab) > 200 ] = NA # missing data replace with NA
hab = raster::projectRaster(hab, crs=raster::crs(locs), method="ngb") # project to locs CRS
hab = raster::crop(hab, raster::extent(locs)+10000) # crop to nearby area
locs_buf = sf::st_buffer(locs, dist=buffer_size) # buffer of specified buffer size
locs_ext1 = raster::extract(hab %in% 4:5, locs_buf,  fun = mean) # closed/semi-closed
locs_ext2 = raster::extract(hab %in% 6, locs_buf, fun = mean) # agriculture
locs$closed_lc = as.vector(locs_ext1)
locs$agri_lc = as.vector(locs_ext2)

# extract distance from the nearest "water" grid cell
# water_ras = hab == 2 # water raster
# water_ras[ !water_ras ] = NA # label non-water as NA
# dist_ras = terra::distance(water_ras) # make raster of distance to nearest water
# writeRaster(dist_ras, "./data/kenya/environment/habitat/distance_to_water.tif", format="GTiff")
dist_ras = raster::raster("./data/kenya/environment/habitat/distance_to_water.tif")
```

```r
locs_ext = raster::extract(dist_ras, locs)
locs$distance_to_water = as.vector(locs_ext)/1000 # convert to km

# extract population density
pop = raster::raster("./data/kenya/environment/population/worldpop_ppp_mara.tif") %>%
  raster::aggregate(fact=5, fun="sum", na.rm=TRUE)
area_ras = (raster::area(pop)) / 10^6 # creates raster with cell size in km squared
popdens = pop / area_ras # calculate population density per km2
popdens = log(popdens+1)
names(popdens) = "popdens_log"
locs_reproj = sf::st_transform(locs, crs=crs(popdens)) # transform to match pop raster
locs_pd = raster::extract(popdens, locs_reproj) # extract value per grid cell
locs$popdens_log = as.vector(locs_pd) # add into our "locs" dataframe

# calculate number of days sampled per camera trap
effort = read.csv("./data/kenya/survey/bh_camera_samplingeffort.csv") %>%
  dplyr::filter(effort_class == 1) %>%
  dplyr::group_by(CT_site) %>%
  dplyr::summarise(n_days_sampled = n_distinct(Date))
locs = locs %>% left_join(effort) %>%
  dplyr::mutate(
    # replaces autofill NAs
    n_days_sampled = replace(n_days_sampled, is.na(n_days_sampled), 0)
  )

# calculate livestock detections from camera trap data
# (n=3540 observations of livestock)
ctd_liv = read.csv("./data/kenya/survey/bh_camera_images_mara.csv") %>%
  dplyr::filter(CT_site %in% locs$CT_site) %>% # ensure sites are in location data
  dplyr::mutate(Date = as.Date(Date, format="%Y-%m-%d")) %>%
  dplyr::filter(Species == "livestock") %>%
  dplyr::group_by(CT_site) %>%
  dplyr::summarise(
    n_days_livestock = n_distinct(Date)
  )

# add to locs df and calculate a livestock pressure indicator
# (proportion days when livestock detected)
locs = locs %>%
  dplyr::left_join(ctd_liv) %>%
    dplyr::mutate(
    n_days_livestock = replace(n_days_livestock, is.na(n_days_livestock), 0)
  ) %>%
  dplyr::mutate(
    livestock_pressure = n_days_livestock / n_days_sampled
  )

# drop unnecessary columns and save covariates
ct_data = locs %>%
  st_drop_geometry() %>%
  dplyr::select(-n_days_livestock) %>%
  write.csv("./data/kenya/data_processed/bh_site_covariates_spatial.csv", row.names=FALSE)
```

```
# ======================================================================== #
```

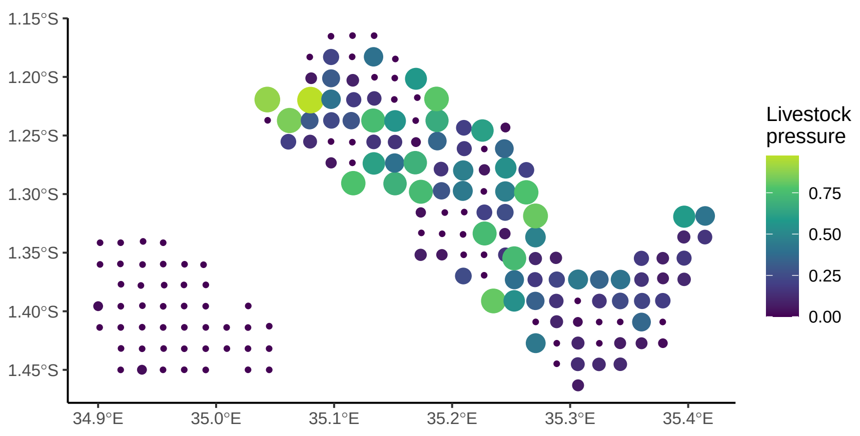
```
# reads in data required for workshop

# read in required data (n=178 camera traps)
# (as produced above)
# create an SF object with the location geometry and site-level covariates
locs = read.csv("./data/kenya/survey/bh_camera_locations.csv") %>%
  sf::st_as_sf(coords = c("Longitude", "Latitude"), crs = 4326) %>%
  sf::st_transform(locs, crs = "+proj=utm +zone=36 +south +datum=WGS84 +units=m +no_defs") %>%
  dplyr::filter(CT_site != "MT34")

# add coordinates columns for XY locations
locs = cbind(locs, sf::st_coordinates(locs))

# covariates for each camera trap
covars = read.csv("./data/kenya/data_processed/bh_site_covariates_spatial.csv") %>%
  dplyr::select(-Conservancy, -X, -Y)

# combine and only keep locations where camera sampled for > 0 days (n=175)
locs = locs %>%
  dplyr::left_join(covars) %>%
  dplyr::filter(n_days_sampled > 0)
```

Exercise 1

- Call head() to familiarise yourself with the data structure of *locs*.
- Explore the distribution of our covariates of interest using histograms and scatterplots, and use *ggplot()* to plot boxplots of covariates of interest across different conservancies (see code below). Will all of these covariates be suitable to include in a model?
- Explore mapping anthropogenic and environmental factors - how do these vary across the study area? We should remember any spatial differences between conservancies later when we are running our analyses.

```
# create a longitudinal dataframe of conservancy vs each covariate
locs_longdf = locs %>%
  sf::st_drop_geometry() %>%
  tidyr::pivot_longer(cols = c("closed_lc", "agri_lc",
                               "distance_to_water", "popdens_log", "livestock_pressure"),
                      names_to="covariate", values_to="value")

# boxplots of covariates across each conservancy - what do you notice?
ggplot(locs_longdf) +
  geom_boxplot(aes(factor(Conservancy), value, group=Conservancy, fill=Conservancy)) +
  theme_minimal() +
  facet_wrap(~covariate, scales="free_y") +
  # sets the x-axis text to print at an angle and not overlapping the plot
  theme(axis.text.x = element_text(angle = 45, hjust = 0.8))
```

```r
# map livestock pressure across the study area
# change this to look at other covariates!
locs %>%
  ggplot() +
  geom_sf(aes(size=livestock_pressure, color=livestock_pressure)) +
  theme_classic() +
  scale_color_viridis_c(end=0.9, name="Livestock\npressure") +
  scale_size(guide="none")
```



Combining environmental covariates with species detections to create a modelling dataframe

Now let's incorporate survey data for our species of interest (*Lepus capensis*) from the camera trap images. As the day is our unit of sampling, we'll calculate the number of days in which the species was detected, and also the proportion of surveyed days.

```r
# specify our species of interest
spp = "hare"

# number of days with hares observed per camera trap
ctd = read.csv("./data/kenya/survey/bh_camera_images_mara.csv") %>%
  dplyr::filter(CT_site %in% locs$CT_site) %>%
  dplyr::mutate(Date = as.Date(Date, format="%Y-%m-%d")) %>%
  dplyr::filter(Species == spp) %>%
  dplyr::group_by(CT_site) %>%
  dplyr::summarise(n_days_detected = n_distinct(Date))

# add to "locs" and replace autofill NAs
locs = locs %>%
  dplyr::left_join(ctd) %>%
  dplyr::mutate(n_days_detected = replace(n_days_detected, is.na(n_days_detected), 0))
```

```
# calculate proportion detected and ensure ranges between 0 and 1
locs$prop_detected = locs$n_days_detected / locs$n_days_sampled
range(locs$prop_detected)

# quick viz
ggplot(locs) +
  geom_sf(aes(size=prop_detected, color=prop_detected), alpha=0.8) +
  theme_classic() +
  scale_color_viridis_c(name="Hare\noccupancy\n(proportion\nsampled\ndays)") +
  scale_size(guide="none")
```

Exercise 2

- Plot scatterplots of the relationship between our response variable (proportion hare detections, *"prop_det"*) and our covariates of interest, in particular livestock activity, closed habitat, distance to water and conservancy. Do you see any obvious evidence of relationships in the raw plots?

```
# exercise 2 solution

# visualise
ggplot(locs) + geom_point(aes(livestock_pressure, prop_detected)) + theme_classic()
ggplot(locs) + geom_point(aes(closed_lc, prop_detected)) + theme_classic()
ggplot(locs) + geom_point(aes(distance_to_water, prop_detected)) + theme_classic()
ggplot(locs) + geom_jitter(aes(Conservancy, prop_detected)) + theme_classic()
```

Fitting logistic (binomial) regression models to estimate probability of hare occupancy

Let's investigate our research question using generalised linear models. Our response variable is binomial - i.e. the species either was detected or not detected, during each sampling window. Each camera trap sampled for a particular number of nights ("trials") with a particular number of detections ("successes"), and we are interested in how our covariates affect the *probability* of success. We model this using **logistic regression** with a binomial likelihood and logit link function, where we estimate the linear effects of covariates $X_1 : X_n$ on the log odds of hare occurrence.

The model would be formulated as:

$Y_i \sim Binom(n_i, p_i)$

$log(p_i/(1-p_i)) = \beta_0 + X\beta$

where Y_i is the proportion of successful outcomes (hare detections), n_i is the number of trials (the number of sampled days), and p_i is the probability of success, which we are estimating as a function of covariates. β is a vector of slope parameters, and X is a matrix of covariates.

Our covariates are all on different scales of magnitude to each other, so slope estimates are difficult to compare between covariates. To deal with this we centre and scale covariates - subtract the mean and divide by the standard deviation. This way, slope parameters always describe the change in Y for 1 standard deviation change in X, regardless of what units X was measured in.

```
# scale linear covariates for comparability and save as new variables "_s"
# (denoting scaled)
locs$closed_lc_s = scale(locs$closed_lc)
```

```
locs$livestock_pressure_s = scale(locs$livestock_pressure)
locs$distance_to_water_s = scale(locs$distance_to_water)

# plot histograms of our scaled covariates
# what do you notice in comparison to the unscaled versions?
```

We use the *glm()* function to fit a generalised linear model, defining this formula as "Y ~ covariate + covariate + . . . ", specifying a binomial likelihood. Since the conservancies showed markedly different levels of livestock activity and habitat factors, we include conservancy as a covariate *a priori* in the model, to account for these coarse spatial differences across the study area.

```
# logistic regression model with livestock and conservancy
# response is the proportion detections
# "weights" argument provides the model with number of trials
m1 = glm(prop_detected ~ livestock_pressure_s + Conservancy,
         family=binomial(link="logit"),
         weights = n_days_sampled,
         data=locs)

# summary information
summary(m1)
```

- Call *summary()* on the model to see summary information including residuals and fitted parameters (coefficients table). What does the slope estimate suggest about the relationship with livestock pressure?

The model summary also shows some goodness-of-fit statistics based on the log-likelihood:

- **Residual deviance**, a measure of how much of the total variation in the observed data is explained by the model (*lower values = more variation explained = better model*). A significant reduction from the null deviance supports the inclusion of covariates in the model.
- **AIC (Akaike Information Criterion)** accounts for the improvement in log-likelihood provided by including more parameters, while penalising overfitting to the data (*lower values = better fitting model*).

We should also look at the distribution of residuals (the unexplained error not accounted for by the model) to check our model is adhering to assumptions. These plots get tricky to interpret visually for GLMs because different likelihoods make different assumptions about how error is distributed (see: https://bookdown.org/ltupper/340f21_notes/deviance-and-residuals.html). Plotting the model shows the Pearson residuals; generally these should be evenly distributed around the fitted line (shown in red).

```
# AIC
AIC(m1)

# plot model
plot(m1)
```

Exercise 3

What other factors might be influencing hare occurrence and also covary with livestock occurrence? It may be important to include these too, in case they explain some of this relationship.

- Fit another model called *m2*, also including closed habitat and distance to water as covariates.
- Call *summary()* to examine the model. Compare the AIC and residual deviance; does adding these covariates improve the model?

```r
# exercise 3 solution

# adding additional covariates
m2 = glm(prop_detected ~ livestock_pressure_s + Conservancy + distance_to_water_s + closed_lc_s,
         family=binomial(link="logit"),
         weights = n_days_sampled,
         data=locs)

# summary
summary(m2)
```

- The function below extracts the fitted parameter estimates, calculates the 95% confidence intervals and visulises them. Use this to plot the slope estimates for *m2*. What does the model suggest about the relationship between hare occupancy, livestock and habitat metrics? What about how hare occupancy differs between conservancies?

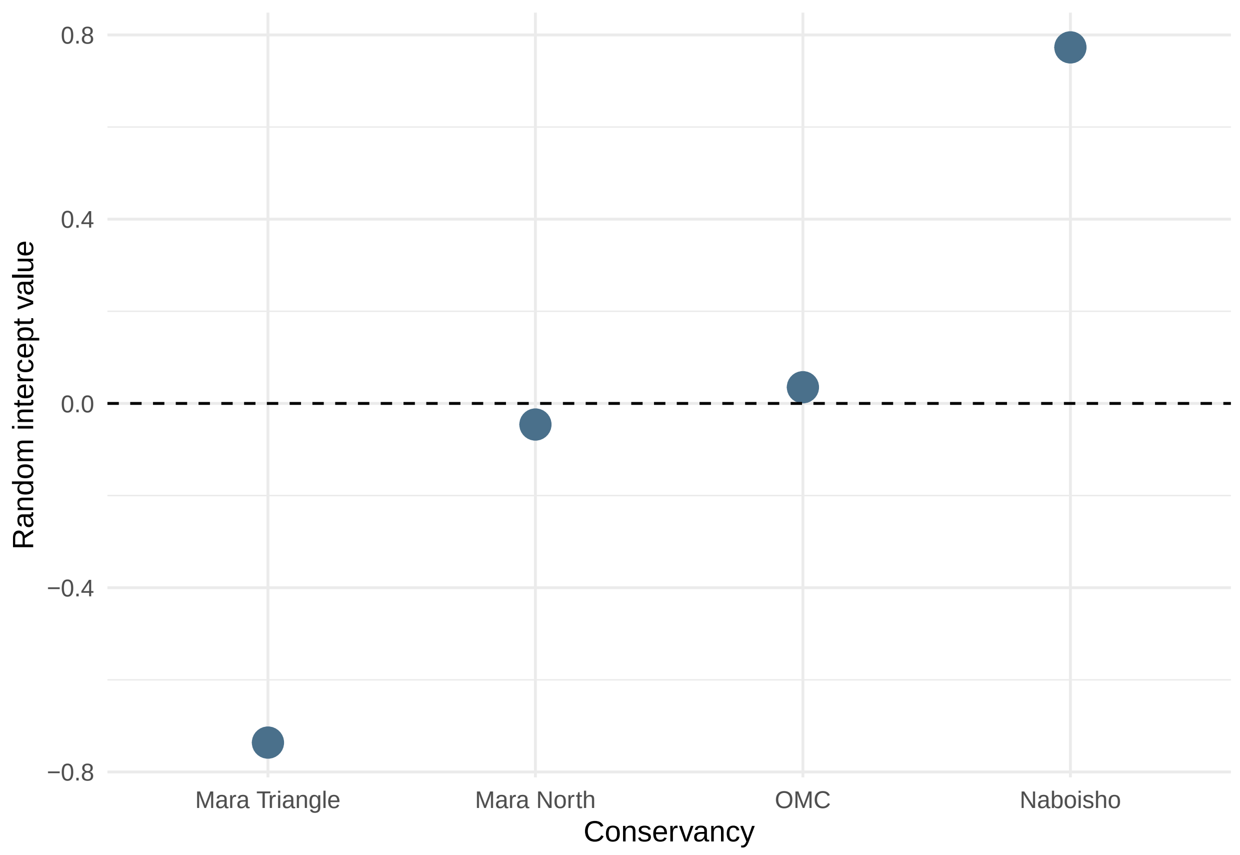
```r
# Function to plot coefficients and confidence intervals.
# The intercept is often at a different scale to the slope parameters
# so creates a separate sub-plot for the intercept).

plotFixedEffects = function(model){

  plot_df = coef(summary(model)) %>% # extract parameters table from model
    as.data.frame() %>% # convert to df
    tibble::rownames_to_column(var="param") %>% # make column "param" from row names
    # classify param as either Intercept or Slope
    dplyr::mutate(param_type = ifelse(param == "(Intercept)", "Intercept", "Slope")) %>%
    dplyr::rename("se"=3) # rename std error variable because easier to work with

  plot = ggplot(plot_df) +
    geom_point(aes(param, Estimate), size=3) + # point estimate
    # 95% confidence interval (1.96 * standard error)
    geom_linerange(aes(param, ymin=Estimate-(1.96*se), ymax=Estimate+(1.96*se))) +
    geom_hline(yintercept=0, lty=2) + # horizontal line marking zero (i.e. no effect)
    theme_minimal() +
    facet_wrap(~param_type, scales="free") + # split plot by parameter type
    theme(axis.text = element_text(size=12),
          axis.title = element_text(size=12),
          strip.text = element_text(size=14)) +
    xlab("Parameter") + ylab("Estimate (95% confidence interval)") +
    coord_flip() # flip so the plot is horizontal

  return(plot)
}

# plot
plotFixedEffects(m2)
```

Note that conservancy is a 4-level categorical variable, so the model is estimating how the intercept differs between each category (i.e. are hare detections on average higher or lower in each conservancy?). One of the categories is incorporated into the intercept (the base factor), and the 3 other parameters estimate the log odds difference in hare occurrence between the base factor and each of the other conservancies.

Since there are only 4 levels, it was reasonable to include conservancy as a categorical fixed effect. An alternative, especially if there were a large number of conservancies, would be to design this as a **mixed effects (multilevel) model** and specify a *random intercept* for conservancy. In the solutions there is code which shows how you can do this.

- *If you like, you could also explore what happens to the slope covariates if you fit a model excluding conservancy. What does this suggest about the importance of accounting for spatially-clustered sampling within the model?*

```r
# bonus exercise: specifying conservancy as a random intercept
# fitting model as a glmer() in the lme4 package

# random intercept specified as "+ (1|Conservancy)"
m3.glmer = lme4::glmer(prop_detected ~ livestock_pressure_s + distance_to_water_s + closed_lc_s + (1|Con
                       family=binomial(link="logit"),
                       weights = n_days_sampled,
                       data=locs)

# plot fixed effects
plotFixedEffects(m3.glmer)

# extract fitted random intercepts and plot
# same general findings as with m3 categorical variable:
# high in Naboisho, low in Mara Triangle
lme4::ranef(m3.glmer) %>%
  as.data.frame() %>%
  ggplot() +
  geom_point(aes(grp, condval), size=5, color="skyblue4") +
  theme_minimal() +
  geom_hline(yintercept=0, lty=2) +
  xlab("Conservancy") + ylab("Random intercept value")
```



Investigating nonlinear relationships between livestock pressure and hare occupancy in 'mgcv'

Linear models make simplifying assumptions that the relationship between covariate X and response variable Y is monotonic, which may not adequately reflect biological reality - for example, many biological processes follow hump-shaped or saturated curves. Such nonlinear interactions can occur in complex ecosystems, so our models need to be flexible enough to account for them.

So far, using logistic regression we found some evidence for a positive relationship between livestock pressure and the probability of hare occupancy. We earlier hypothesised that hare occupancy might either be positively or negatively related to livestock pressure. But **alternatively, what if both are partially true, and that the relationship is nonlinear, with hare occupancy peaking at intermediate livestock activity (which creates amenable habitat) and declining at very high levels (due to excessive disturbance)?**

Many approaches are available for fitting nonlinear functions to data, and models that combine mixtures of linear and nonlinear terms are generally called **generalised additive models** (GAMs). We'll use the R package *mgcv*, which provides a fast and flexible framework to fit many types of nonlinear model. (For a great beginner's introduction to the principles of GAMs in mgcv, see: https://noamross.github.io/gams-in-r-course/chapter1).

We'll again fit a logistic regression model, with the only difference that we are now fitting a smooth function $f(X_1)$ of our covariate. We can still also include linear effects.

$Y_i \sim Binom(n_i, p_i)$

$log(p_i/(1 - p_i)) = \beta_0 + f(X_1)$

We can use mgcv to fit both the same linear model as above, and a model that includes a nonlinear effect of livestock pressure using a penalised thin-plate regression spline. Let's look at the difference between the two.

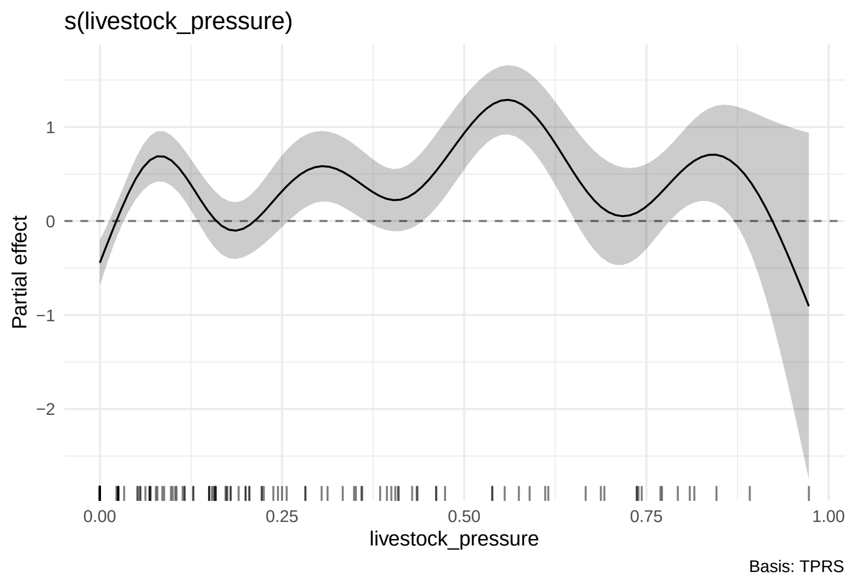
```r
# using mgcv's gam function to fit a linear model
# (parameter estimates are almost the same as the glm)
m4 = mgcv::gam(prop_detected ~ Conservancy + livestock_pressure_s,
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# adding a nonlinear function of livestock occ using s()
m5 = mgcv::gam(prop_detected ~ Conservancy + s(livestock_pressure),
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# call summary() to look at the fitted model
```

- Compare the AIC of the two models; does including the nonlinear term provide a better fit to the data?

```r
# substantial improvement from nonlinear term
# but is it biologically plausible?
data.frame(
  model = c("linear", "nonlinear"),
  AIC = c(AIC(m4), AIC(m5))
)
```

You can call *plot()* on the fitted model to visualise the fitted partial effect of livestock pressure, or alternatively the gratia package provides some great tools for visualising GAMs.

```r
# using plot
#plot(m5)

# using gratia's draw() function
gratia::draw(m5) + theme_minimal() + geom_hline(yintercept=0, lty=2, alpha=0.5)
```



- What does your biological intuition suggest to you about how generalisable or realistic this fitted relationship is? How does this compare to what the AIC suggests about which is the better model?

During fitting, the GAM function seeks to optimise the "wiggliness" of the fitted spline to avoid overfitting to the data. Alternatively, we can manually change parameters to determine this, through changing the number of knots (basis functions) or the smoothing parameter that penalises overfitting (which is normally optimised when we use the method='REML' argument).

- *If you like, try experimenting with the code below to investigate what happens to the shape of the fitted spline when you reduce or increase the number of knots (k) from the default of 9, or when you manually increase or decrease the smoothing penalty (sp). Do these functions look more ecologically reasonable? What does the AIC suggest?*

```
# modify k parameter to increase/decrease knots (functional complexity)
# modify sp parameter to increase/decrease smoothing penalty
m5.1 = mgcv::gam(prop_detected ~ Conservancy +
                   s(livestock_pressure, k=9, sp=0.005),
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled)

# plot the function
gratia::draw(m5.1)

# compare AIC to m5 where sp was optimised
AIC(m5); AIC(m5.1)
```

Accounting for spatial dependence among observations using geospatial models

Why is the fitted spline unrealistically wiggly? It's possible it is picking up some residual spatial structure in the data we have not properly accounted for. *Remember*, any model only knows what we tell it explicitly, and our model currently knows nothing about the arrangement of camera trap sites beyond what conservancy they belong to.

Statistical models assume errors are independent, but this assumption can be violated if observations closer together in space are more closely related to each other - **spatial autocorrelation** - for example, because animals move between neighbouring cameras, or because of unmeasured covariates influencing the system. Geospatial statistics provides tools to explicitly account for such spatial dependencies.

In this last section we will explore using mgcv to fit a *Gaussian process* spatial random effect across the study area, and see whether this improves our ability to infer the relationship between livestock pressure and hare occupancy. We have covered some key geospatial statistics concepts in the lectures, including Gassian processes, but the aim here is to provide a (very) brief exploration of their use in practice.

First, let's map the residuals (error) from our fitted GAM across the study area. Does there seem to be any spatial clustering in the residual error that would indicate spatial autocorrelation?

```
# extract residuals
resid = locs %>%
  dplyr::mutate(resid = resid(m5, "deviance"))

# map residuals with diverging colour scale and size to absolute magnitude
resid %>%
  ggplot() +
  geom_sf(aes(color=resid, size=abs(resid))) +
  scale_color_gradient2() +
  theme_classic() +
  ggtitle("m5 residuals")
```

Next, we'll fit a Gaussian process two-dimensional field to the hare occupancy data in mgcv. We use the s() function, giving the X and Y coordinates, and setting the basis as "gp". The m argument specifies the covariance function for the Gaussian process - i.e. the shape and range of how the correlation between observations decays by distance. Here, we specify a Matern covariance function with a range of 3 kilometres (which we previously estimated as optimal for these data based on some exploratory analyses), which determines at what distance observations are considered effectively uncorrelated.

- Compare the AIC between the model with spatial random effect (but no covariates) and m5 - does the spatial random effect improve the model fit?

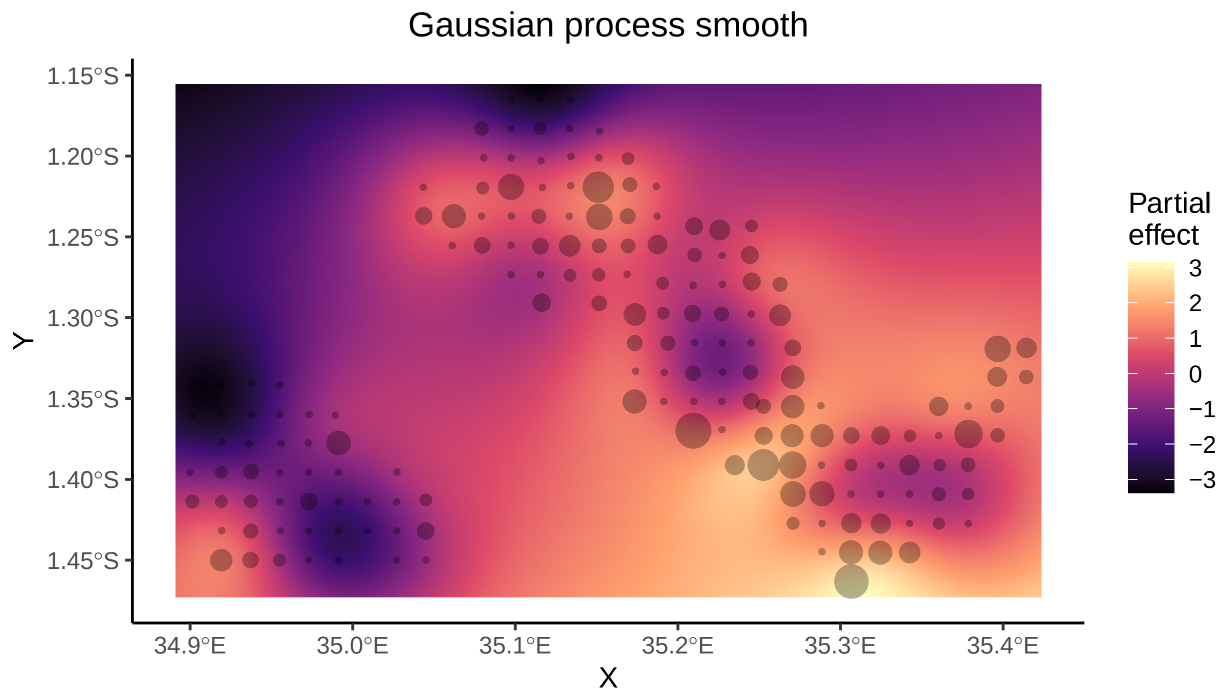
```
# fit model (with no covariates for now)
m6 = mgcv::gam(prop_detected ~ 1 +
                s(X, Y, bs="gp", m=c(3, 3000)), # matern w/ kappa=1.5 and range=3000m
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# call summary
summary(m6)
```

To understand what is going on we can visualise the fitted spatial effect. The function below uses the model to predict the value of the Gaussian process effect in each grid cell of a template raster, and then plot it across the study area.

- Call the plotGPSmooth() function on the model to visualise it. What do you notice about the relationship between nearby observations?

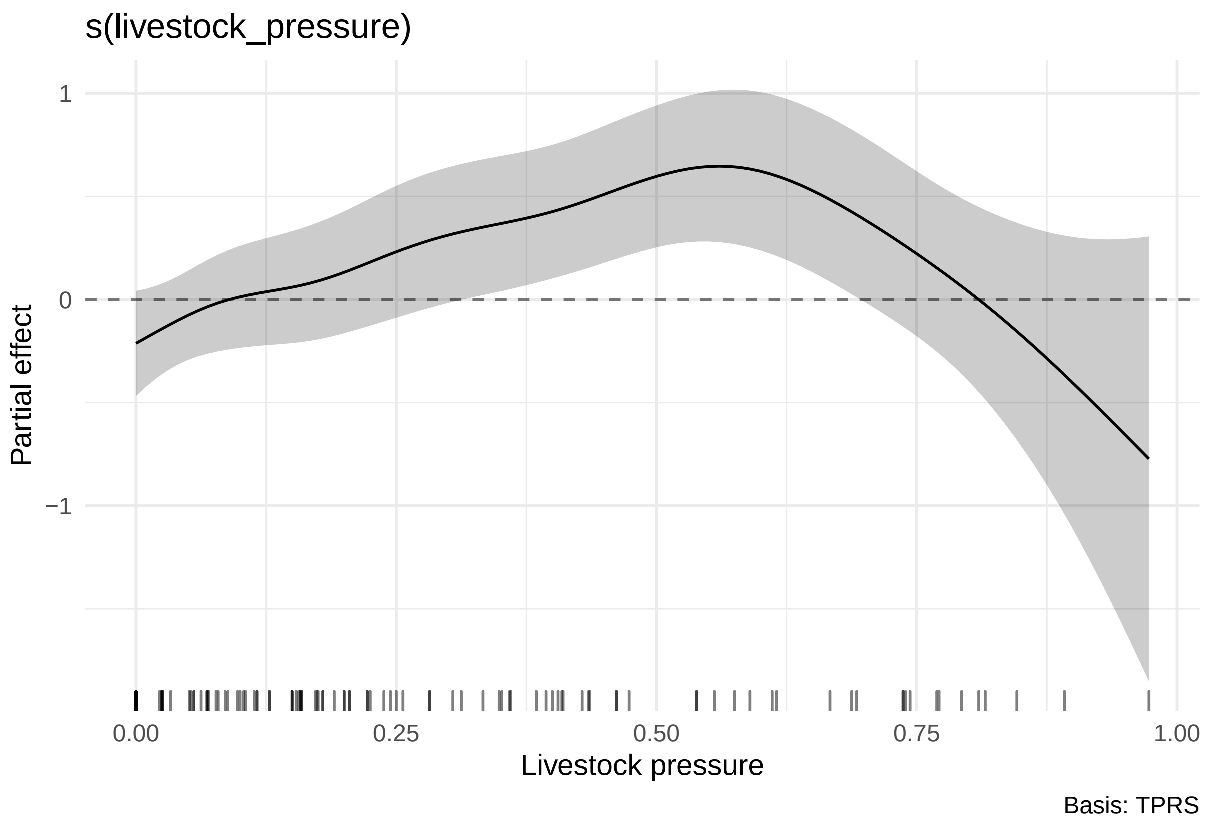
```r
plotGPSmooth = function(model, plot.title="Gaussian process smooth"){

  # grid locations to predict to with XY coordinates
  # (based on creating a template raster of the area)
  new_dat = read.csv("./data/kenya/environment/habitat/template_raster_cells.csv")

  # predict to raster locations
  new_dat$gp_pred = predict.gam(object = model,
                                newdata = new_dat,
                                type = "terms",
                                terms = "s(X,Y)",
                                se.fit = F)

  # plot spatial field and overlay observations
  sp_plot = new_dat %>%
    ggplot() +
    geom_raster(aes(X, Y, fill=gp_pred)) +
    scale_fill_viridis_c(option="magma", name="Partial\neffect") +
    geom_sf(data=locs, aes(size=prop_detected), pch=16, alpha=0.3) +
    theme_classic() +
    ggtitle(plot.title) +
    theme(plot.title = element_text(size=13, hjust=0.5)) +
    scale_size(guide="none")

  return(sp_plot)
}
```

```r
# call the function on our fitted model
plotGPSmooth(m6)
```



Finally, let's reintroduce our livestock covariate. Does accounting for the spatial dependency among sites lead to a more ecologically plausible nonlinear relationship with hare occupancy?

```
# fit the model
m7 = mgcv::gam(prop_detected ~ 1 +
                 s(X, Y, bs="gp", m=c(3, 3000)) +
                 s(livestock_pressure),
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# plot the fitted function
gratia::draw(m7, select="s(livestock_pressure)") + xlab("Livestock pressure") +
  geom_hline(yintercept=0, lty=2, alpha=0.5) + theme_minimal()
```



Exercise 4

Do the data provide stronger evidence for a linear or a nonlinear relationship between livestock pressure and hare occupancy, after we have accounted for space?

- Modify the code above to fit a model with a **linear** effect of livestock pressure. Using AIC, compare between the three models, either including no effect (*m6*), a nonlinear effect (*m7*), or a linear effect - which is the best supported?

```r
# fit model with linear effect of livestock
m8 = mgcv::gam(prop_detected ~ 1 +
                 s(X, Y, bs="gp", m=c(3, 3000)) +
                 livestock_pressure_s,
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# summary - positive linear effect but quite uncertain
summary(m8)

# compare AICs - the nonlinear model is better supported!
data.frame(
  model = c("gp_only", "livestock_linear", "livestock_nonlinear"),
  AIC = c(AIC(m6), AIC(m8), AIC(m7))
)
```

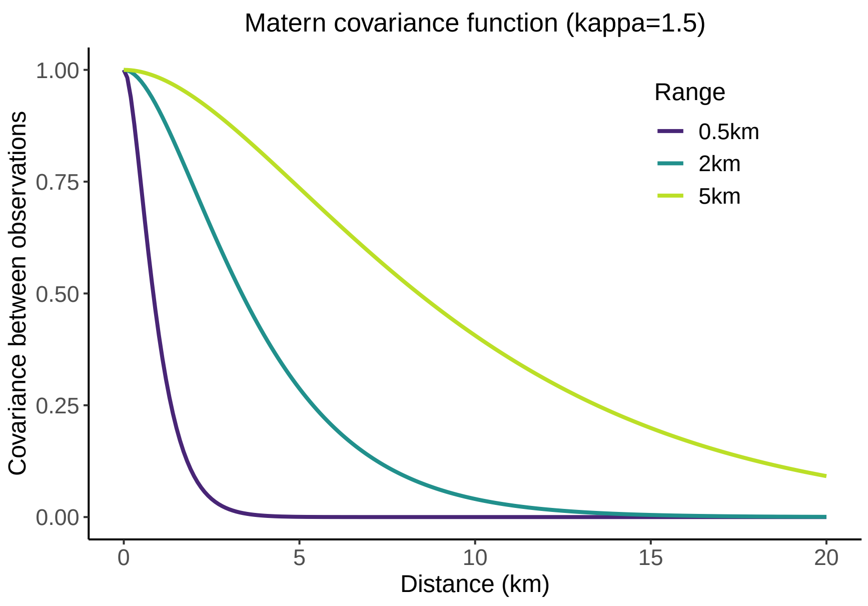
Extension exercises

Exercise 5

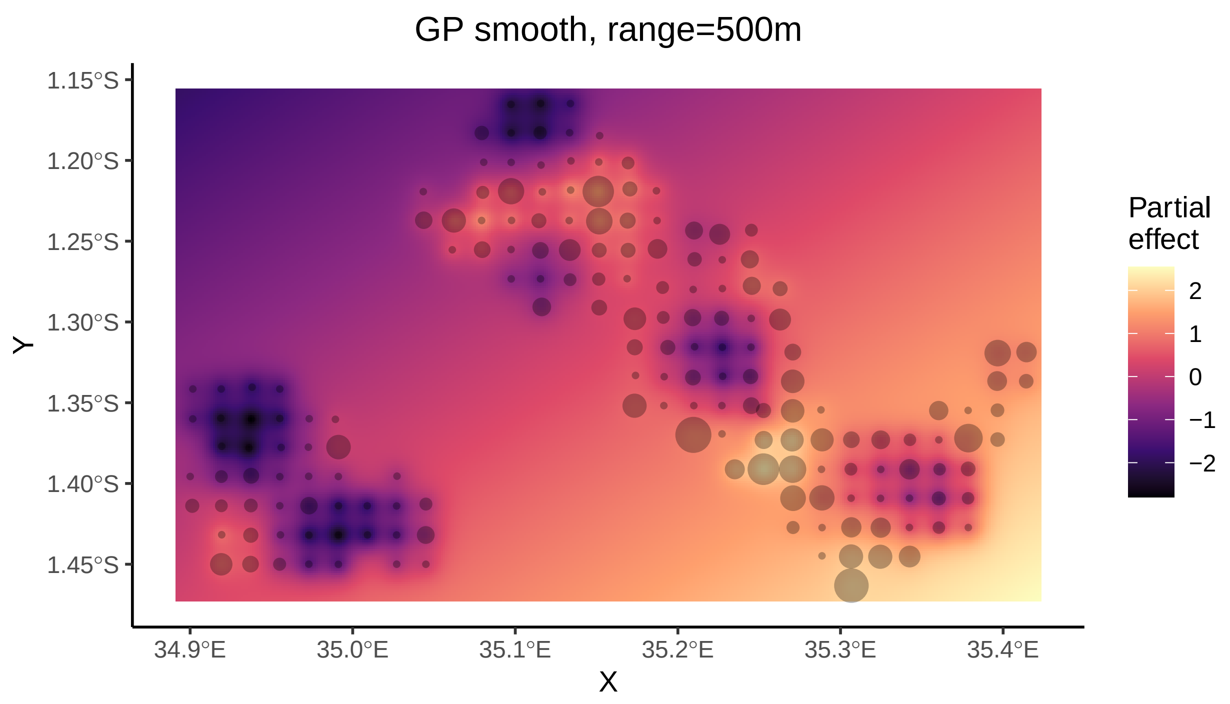
Within the Gaussian process model, the shape of the covariance function determines how the correlation between observations declines with distance. In particular, the *range* parameter, which is set to 3 kilometres in our models above, determines the distance at which observations become effectively uncorrelated (see the figure below for an illustration for the Matern covariance function with varying ranges).

- Experiment with varying the range parameter in *m6* and plotting the Gaussian process smooth. What happens to the shape of the fitted spatial field when we reduce the range to a very short distance, such as 500m, or a very long distance, such as 10km? How ecologically realistic do they seem?

```
# code to produce example covariance functions using geoR's "matern" function

library(geoR)

# create dataframe of matern with kappa 1.5 and phi (range) from 0.5 to 5km
matern_example = data.frame(distance = seq(0, 20, by=0.1)) %>%
  dplyr::mutate(
    `0.5km` = geoR::matern(distance, phi=0.5, k = 1.5),
    `2km` = geoR::matern(distance, phi=2, k = 1.5),
    `5km` =  geoR::matern(distance, phi=5, k = 1.5)
  )

# plot
matern_example %>%
  tidyr::pivot_longer(cols = 2:4) %>%
  ggplot() +
  geom_line(aes(distance, value, group=name, color=name), size=1) +
  theme_classic() +
  xlab("Distance (km)") + ylab("Covariance between observations") +
  theme(legend.position = c(0.8, 0.8),
        plot.title = element_text(hjust=0.5, size=14),
        legend.text = element_text(size=12),
        legend.title = element_text(size=13),
        axis.title = element_text(size=13),
        axis.text = element_text(size=12)) +
  scale_color_viridis_d(begin=0.1, end=0.9, name="Range") +
  ggtitle("Matern covariance function (kappa=1.5)")
```

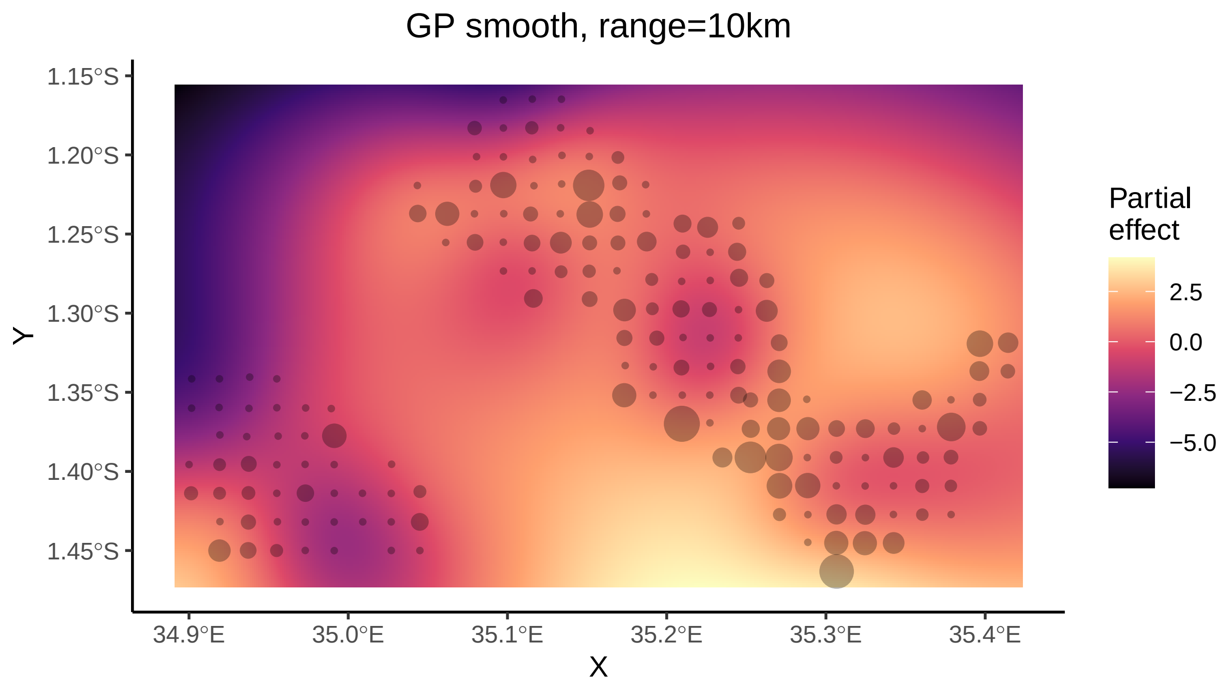


```r
# exercise 5 solution:

# fit GP model with range parameter reduced to 500m
m6.1 = mgcv::gam(prop_detected ~ 1 +
                 s(X, Y, bs="gp", m=c(3, 500)),
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# forces a very short range of spatial correlation
# effectively treating each point as independent
# (similar to a random intercept)
plotGPSmooth(m6.1, plot.title = "GP smooth, range=500m")
```



```r
# AIC suggests a worse fit
AIC(m6); AIC(m6.1)
```

```r
# fit GP model with range parameter increased to 10km
m6.2 = mgcv::gam(prop_detected ~ 1 +
                 s(X, Y, bs="gp", m=c(3, 10000)),
               data = locs,
               family = binomial(link="logit"),
               weights = n_days_sampled,
               method = "REML")

# forces a very long range of spatial correlation
# loses some finer scale spatial heterogenity - not complex enough
plotGPSmooth(m6.2, plot.title = "GP smooth, range=10km")
```



```
# AIC suggests a worse fit
AIC(m6); AIC(m6.2)
```

Exercise 6

There are several other wildlife species included in the camera trap tagged images data frame. Try developing your own models and hypotheses based on these other species data and the available environmental and social covariates.

- How different are your findings for different species? Are there any consistent patterns?