Model fitting, prediction and plotting

Supporting information for 'The Point Process Framework for Integrated Modelling of Biodiversity Data'

Kwaku Peprah Adjei, Philip Mostert, Jorge Sicacha Parada, Emma Skarstein, Robert B. O'Hara

Last updated: October 20 2023 at 22:46:26.

Loading packages:

```
#devtools::install_github("PhilipMostert/PointedSDMs")
library(PointedSDMs) # model fitting
library(ggplot2)     # plotting
library(raster)      # ???? Model fitting fails if I exclude this
library(mapproj)     # map options for plotting
library(INLA)        # functions for specifying mesh
library(dplyr)       # data handling
library(sf)          # spatial stuff
library(showtext)    # font for plot
library(patchwork)   # combining figures
```

Some plot settings:

```
showtext_auto()
f1 <- "Open sans"
font_add_google(f1, f1)
```

1 Downloading data

Before you run this file, make sure you have the following files:

- `data/artsobs_clean_new.rds`
- `data/survey_clean_new.rds`
- `data/environmental_covariates.rds`
- `data/Norwegian_lakes.rds`

This model may take quite a while to run (approx. 6 hours with the final settings that were used for the paper). To run a faster version, it is possible to use less integration points in the mesh, this is described below. Alternatively, the run-time can be reduced by including less than the full four fish species, you can choose which fish species to include in the chunk below. (OBS: You will also need to remove any species you don't want in the chunk where we use "changeComponents" for specific species fields)

```
fishes <- c("Esox_lucius", "Perca_fluviatilis", "Salmo_trutta", "Salvelinus_alpinus")
```

2 Loading Norway map and making mesh

We begin by making a spatial mesh out of a map of Norway.

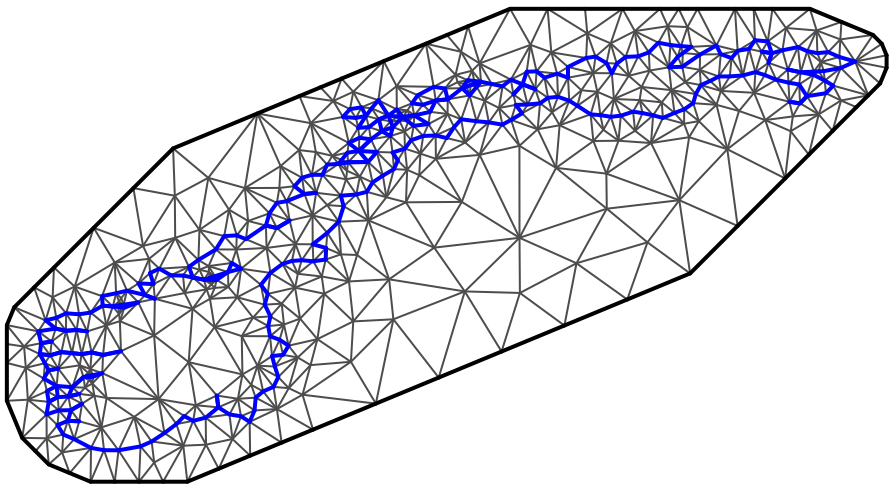
```
#norway <- ggplot2::map_data("world", region = "Norway(?!:Svalbard)")
#norway <- setdiff(norway, dplyr::filter(norway, subregion == "Jan Mayen"))
#norway_sf <- st_as_sf(norway, coords = c("long", "lat"), crs = proj)

proj <- '+proj=utm +zone=32 +datum=WGS84 +units=m +no_defs'
norwayfill <- maps::map("world", "norway", fill=TRUE, plot=FALSE,
                  ylim=c(58,72), xlim=c(4,32))
IDs <- sapply(strsplit(norwayfill$names, ":"), function(x) x[1])
norway.poly <- maptools::map2SpatialPolygons(norwayfill, IDs = IDs,
                                             proj4string = CRS(proj))
```

```
## Please note that 'maptools' will be retired during October 2023,
## plan transition at your earliest convenience (see
## https://r-spatial.org/r/2023/05/15/evolution4.html and earlier blogs
## for guidance);some functionality will be moved to 'sp'.
##  Checking rgeos availability: FALSE
```

Adjusting the mesh to be coarser is the easiest way to decrease the run-time for the model. With the following mesh, the model fitting takes just about 1 hour for us, but feel free to change the `max.edge` or `cutoff` to get a coarser mesh.

```
mesh <- inla.mesh.2d(boundary = inla.sp2segment(norway.poly),
                     cutoff = 0.3, # smallest allowed distance between points
                     max.edge = c(6, 3), # decrease this for more int. points
                     #max.edge = c(3, 1),
                     offset = c(1, 1),
                     crs = st_crs(proj))
plot(mesh)
```



3 Setting up covariate data

Next we load the environmental data, which will be used as covariates.

```
covariates_raw <- readRDS("data/environmental_covariates.RDS")

covariates <- covariates_raw %>%
```

```
  # Log-transform area of lake
  dplyr::mutate(log_area = log(area_km2)) %>%
  # Remove some uninformative variables
  dplyr::select(-c(ebint, no_vatn_lnr, eb_waterregionID))

# Choose from
# "decimalLatitude", "decimalLongitude",
# "log_area", "perimeter_m", "distance_to_road",
# "eurolst_bio10", "catchment_area_km2", "SCI", "HFP"

Use <- c("log_area", "eurolst_bio10", "SCI")

cov_pixel <- SpatialPixelsDataFrame(
  points = covariates[,c("decimalLongitude","decimalLatitude")],
  data = covariates[,Use],
  proj4string = CRS(proj),
  tol = 0.340571)

# Scale covariates and convert to terra::rast
cov_raster <- scale(terra::rast(cov_pixel))
```

4 Observation data

For this model, we have two observation sets, one which is downloaded from GBIF and one that is a survey dataset (see separate document for download instructions).

```
survey <- readRDS("data/survey_clean_new.rds") %>%
  filter(species %in% fishes)
artsobs <- readRDS("data/artsobs_clean_new.rds") %>%
  filter(species %in% fishes)
```

We can plot the observed data points:

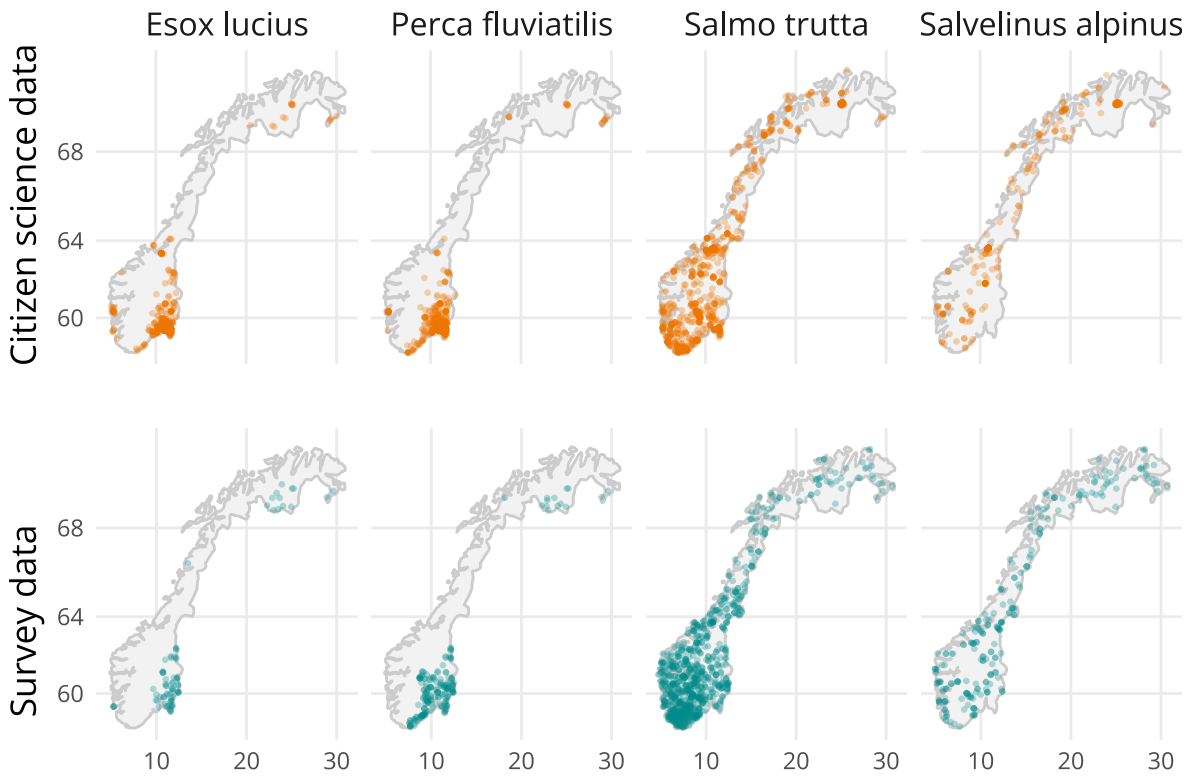
```
norway <- ggplot2::map_data("world", region = "Norway(?!:Svalbard)")
norway <- setdiff(norway, dplyr::filter(norway, subregion == "Jan Mayen"))

p_artsobs <- ggplot(artsobs, aes(x = decimalLongitude, y = decimalLatitude)) +
  geom_polygon(data = norway, aes(long, lat, group = group),
               color="grey80", fill = "grey95") +
  geom_point(color = "darkorange2", size = 0.5, alpha = 0.3) +
  facet_wrap(~species, nrow = 1,
             labeller = labeller(species = function(string) sub("_", " ", string))) +
  coord_map() +
  labs(tag = "Citizen science data") +
  theme_minimal() +
  theme(text = element_text(family = f1),
        strip.text = element_text(family = f1, size = 12),
        plot.tag = element_text(angle = 90, hjust = 0.5),
        plot.tag.position = c(-0.03, 0.45),
        legend.position = "none",
        axis.title = element_blank(),
        axis.text.x = element_blank())

p_survey <- ggplot(survey %>% filter(occurrenceStatus == 1),
```

```
      aes(x = decimalLongitude, y = decimalLatitude)) +
  geom_polygon(data = norway, aes(long, lat, group = group),
               color="grey80", fill = "grey95") +
  geom_jitter(color = "darkcyan", size = 0.5, alpha = 0.3) +
  facet_wrap(~species, nrow = 1) +
  coord_map() +
  xlab("Longitude") +
  ylab("Latitude") +
  labs(tag = "Survey data") +
  theme_minimal() +
  theme(text = element_text(family = f1),
        plot.tag = element_text(angle = 90, hjust = 0.5),
        plot.tag.position = c(-0.03, 0.45),
        legend.position = "none",
        axis.title = element_blank(),
        strip.text = element_blank(),
        plot.margin = margin(l = 30))

p_artsobs / p_survey
```




```
ggsave("figures/presence_points.pdf", height = 5, width = 8)
```

5 Joint model for four fish species

For the presence/absence survey data, we use a Bernoulli distribution, where the presence probability for species $j \in \{$*Salmo trutta, Perca fluvatilis, Esox lucius, Salvelinus alpinus*$\}$ depends on some covariates $x(s)$, along with a spatial field $\xi_j(s)$:

$$Y_{PA,j}(s_i) \sim \text{Bernoulli}(p_{PA,j}(s_i))$$
$$\text{cloglog}(p_{PA,j}(s_i)) = \alpha_{PA,j} + x(s_i)^T\beta_j + \xi_j(s_i).$$

The presence-only data is fitted with a Poisson point process model, where the intensity depends on the same covariates $x(s)$ and the same spatial field $\xi_j(s)$, plus an additional spatial field $\xi_{\text{bias}}(s)$ that is unique to the citizen science data, but shared across all fish species:

$$Y_{PO,j}(s_i) \sim \text{Poisson}(e^{\eta_{PO,j}(s_i)})$$
$$\eta_{PO,j}(s_i) = \alpha_{PO,j} + x(s)^T\beta_j + \xi_j(s_i) + \xi_{\text{bias}}(s_i).$$

So in summary, for this model we have:

- one bias field, based on CS data and common across all species
- four shared fields, shared across the data sets (survey/citizen science), but separate for each fish species.

Since we have two data sets and four species, that means that we in total have eight sub-models.

We first prepare the model using the `intModel` function.

```
fish_model_setup <- intModel(
  survey,                                 # Survey data
  artsobs,                                # Citizen science data
  spatialCovariates = cov_raster,         # Covariates
  speciesName = "species",                # The column containing species name
  speciesSpatial = "copy",                # Copy species fields across data
  Coordinates = c("decimalLongitude", "decimalLatitude"), # Name of coords
  responsePA = "occurrenceStatus",        # Name of response column
  pointsSpatial = NULL,                   # NULL since we use speciesSpatial
  Mesh = mesh,                            # inla mesh object
  Projection = proj)                      # CRS for points and covariates
```

This model has the species specific spatial fields, but we also want a bias field that is shared across the species. We add this using `$addBias`.

```
fish_model_setup$addBias("artsobs")
```

For the species specific fields, the default in PointedSDMs is that these are allowed to be different up to a scaling factor (named beta in INLA) across the data sets. That means that for instance the trout-specific spatial field for the citizen science data set is equal to beta times the trout-specific spatial field for the survey data. In practice, this has to do with the copy-option in INLA. But in our model we want them to be the same, not to vary by a factor. So we manually change this using the `$changeComponents` function, by setting `hyper = list(beta = list(fixed = TRUE))` for each of the four citizen science fields.

```
fish_model_setup$changeComponents(
  ~ Perca_fluviatilis_artsobs_spatial(
      main = geometry,
      copy = "Perca_fluviatilis_survey_spatial",
      hyper = list(beta = list(fixed = TRUE))) +
    Salmo_trutta_artsobs_spatial(
      main = geometry,
      copy = "Salmo_trutta_survey_spatial",
      hyper = list(beta = list(fixed = TRUE))) +
    Salvelinus_alpinus_artsobs_spatial(
      main = geometry,
      copy = "Salvelinus_alpinus_survey_spatial",
      hyper = list(beta = list(fixed = TRUE))) +
    Esox_lucius_artsobs_spatial(
```

```
      main = geometry,
      copy = "Esox_lucius_survey_spatial",
      hyper = list(beta = list(fixed = TRUE))))
```

```
## Components:
## ~-1 + Perca_fluviatilis_survey_spatial(main = geometry, model = Perca_fluviatilis_survey_field) +
##     Salmo_trutta_survey_spatial(main = geometry, model = Salmo_trutta_survey_field) +
##     Salvelinus_alpinus_survey_spatial(main = geometry, model = Salvelinus_alpinus_survey_field) +
##     Esox_lucius_survey_spatial(main = geometry, model = Esox_lucius_survey_field) +
##     Salmo_trutta_artsobs_spatial(main = geometry, copy = "Salmo_trutta_survey_spatial",
##         hyper = list(beta = list(fixed = FALSE))) + Salvelinus_alpinus_artsobs_spatial(main = geometi
##     copy = "Salvelinus_alpinus_survey_spatial", hyper = list(beta = list(fixed = FALSE))) +
##     Esox_lucius_artsobs_spatial(main = geometry, copy = "Esox_lucius_survey_spatial",
##         hyper = list(beta = list(fixed = FALSE))) + Perca_fluviatilis_log_area(main = Perca_fluviatil
##     model = "linear") + Salmo_trutta_log_area(main = Salmo_trutta_log_area,
##     model = "linear") + Salvelinus_alpinus_log_area(main = Salvelinus_alpinus_log_area,
##     model = "linear") + Esox_lucius_log_area(main = Esox_lucius_log_area,
##     model = "linear") + Perca_fluviatilis_eurolst_bio10(main = Perca_fluviatilis_eurolst_bio10,
##     model = "linear") + Salmo_trutta_eurolst_bio10(main = Salmo_trutta_eurolst_bio10,
##     model = "linear") + Salvelinus_alpinus_eurolst_bio10(main = Salvelinus_alpinus_eurolst_bio10,
##     model = "linear") + Esox_lucius_eurolst_bio10(main = Esox_lucius_eurolst_bio10,
##     model = "linear") + Perca_fluviatilis_SCI(main = Perca_fluviatilis_SCI,
##     model = "linear") + Salmo_trutta_SCI(main = Salmo_trutta_SCI,
##     model = "linear") + Salvelinus_alpinus_SCI(main = Salvelinus_alpinus_SCI,
##     model = "linear") + Esox_lucius_SCI(main = Esox_lucius_SCI,
##     model = "linear") + Perca_fluviatilis_intercept(1) + Salmo_trutta_intercept(1) +
##     Salvelinus_alpinus_intercept(1) + Esox_lucius_intercept(1) +
##     artsobs_biasField(main = geometry, model = artsobs_bias_field) +
##     Perca_fluviatilis_artsobs_spatial(main = geometry, copy = "Perca_fluviatilis_survey_spatial",
##         hyper = list(beta = list(fixed = TRUE))) + Salmo_trutta_artsobs_spatial(main = geometry,
##     copy = "Salmo_trutta_survey_spatial", hyper = list(beta = list(fixed = TRUE))) +
##     Salvelinus_alpinus_artsobs_spatial(main = geometry, copy = "Salvelinus_alpinus_survey_spatial",
##         hyper = list(beta = list(fixed = TRUE))) + Esox_lucius_artsobs_spatial(main = geometry,
##     copy = "Esox_lucius_survey_spatial", hyper = list(beta = list(fixed = TRUE)))
## <environment: 0x7f7a961d37b8>
```

We may look at which terms are included in each of the eight sub-models by calling `$updateFormula` with the data sets as the arguments.

```
fish_model_setup$updateFormula(datasetName = "survey")
```

```
## $Perca_fluviatilis
## occurrenceStatus ~ Perca_fluviatilis_log_area + Perca_fluviatilis_eurolst_bio10 +
##     Perca_fluviatilis_SCI + Perca_fluviatilis_intercept + Perca_fluviatilis_survey_spatial
## <environment: 0x7f7a9678ef60>
##
## $Salmo_trutta
## occurrenceStatus ~ Salmo_trutta_log_area + Salmo_trutta_eurolst_bio10 +
##     Salmo_trutta_SCI + Salmo_trutta_intercept + Salmo_trutta_survey_spatial
## <environment: 0x7f7a9678ef60>
##
## $Salvelinus_alpinus
## occurrenceStatus ~ Salvelinus_alpinus_log_area + Salvelinus_alpinus_eurolst_bio10 +
##     Salvelinus_alpinus_SCI + Salvelinus_alpinus_intercept + Salvelinus_alpinus_survey_spatial
## <environment: 0x7f7a9678ef60>
```

```
##
## $Esox_lucius
## occurrenceStatus ~ Esox_lucius_log_area + Esox_lucius_eurolst_bio10 +
##     Esox_lucius_SCI + Esox_lucius_intercept + Esox_lucius_survey_spatial
## <environment: 0x7f7a9678ef60>
```

```
fish_model_setup$updateFormula(datasetName = "artsobs")
```

```
## $Salvelinus_alpinus
## geometry ~ Salvelinus_alpinus_log_area + Salvelinus_alpinus_eurolst_bio10 +
##     Salvelinus_alpinus_SCI + Salvelinus_alpinus_intercept + Salvelinus_alpinus_artsobs_spatial +
##     artsobs_biasField
## <environment: 0x7f7a9678ef60>
##
## $Esox_lucius
## geometry ~ Esox_lucius_log_area + Esox_lucius_eurolst_bio10 +
##     Esox_lucius_SCI + Esox_lucius_intercept + Esox_lucius_artsobs_spatial +
##     artsobs_biasField
## <environment: 0x7f7a9678ef60>
##
## $Salmo_trutta
## geometry ~ Salmo_trutta_log_area + Salmo_trutta_eurolst_bio10 +
##     Salmo_trutta_SCI + Salmo_trutta_intercept + Salmo_trutta_artsobs_spatial +
##     artsobs_biasField
## <environment: 0x7f7a9678ef60>
##
## $Perca_fluviatilis
## geometry ~ Perca_fluviatilis_log_area + Perca_fluviatilis_eurolst_bio10 +
##     Perca_fluviatilis_SCI + Perca_fluviatilis_intercept + Perca_fluviatilis_artsobs_spatial +
##     artsobs_biasField
## <environment: 0x7f7a9678ef60>
```

Finally, we actually fit the model using `fitISDM`.

```
fish_model <- fitISDM(fish_model_setup,
                      options = list(
                        control.inla = list(int.strategy = 'eb', cmin = 0),
                        safe = TRUE,
                        inla.mode = 'experimental'))
```

We may then examine the model summary and save the model for future use.

```
summary(fish_model)

saveRDS(fish_model, "results/fish_model.rds")
```

6 Predictions and plots

Once the model has been fit, we can look at the predictions from the species-specific shared fields and the bias field.

We define a function that will do species-specific predictions, and save the species predictions, since these take a little time to compute.

```
predict_species <- function(model, species, mask, mesh){
  sharedfield <- predict(model,
                         mesh = mesh,
```

```
                          mask = mask,
                          format = 'sp',
                          spatial = TRUE,
                          fun = 'linear',
                          species = species,
                          n.samples = 1000)
  file_name <- paste0("results/sharedfield_", species, ".rds")
  saveRDS(sharedfield, file_name)
  return(sharedfield)
}
```

```
prediction_list <- list()
for(fish in fishes) {
  prediction_list[[fish]] <- predict_species(
    model = fish_model,
    mesh = mesh,
    mask = norway.poly,
    species = fish
  )
}
```

Once we have the predictions, we can make some plots. We similarly define a function that makes a plot for one species, and then run this for all four species.

```
plot_preferences <- list(scale_fill_distiller(palette = "BrBG", direction = 1),
                          coord_map(),
                          xlab(""), ylab(""), labs(fill = ""),
                          theme_minimal(),
                          theme(text = element_text(family = f1),
                                title = element_text(family = f1, size = 10),
                                legend.key.height = unit(0.3, "cm"),
                                legend.position = "bottom",
                                plot.margin = margin(10, 10, 0, 10, "points"))
)

plot_species <- function(predictions, species_to_plot, plot_preferences){
  p <- ggplot() +
    geom_polygon(data = norway, aes(long, lat, group = group),
                 color="grey80", fill = "grey95") +
    gg(predictions$speciesPredictions[[species_to_plot]]) +
    labs(title = sub("_", " ", species_to_plot)) +
    plot_preferences
  return(p)
}
```

```
plot_list <- list()
for(fish in fishes) {
  plot_list[[fish]] <- plot_species(
    predictions = readRDS(paste0("results/sharedfield_", fish, ".rds")),
    species = fish,
    plot_preferences = plot_preferences)
}

patchwork::wrap_plots(plot_list, nrow = 1)
ggsave("figures/fishplot_four_species.pdf", height = 2, width = 8)
```

And finally we predict and plot the bias field, which is shared between all the fish, as it describes the human sampling more than the distribution of the fish.

```
fish_biasfield <- predict(fish_model,
                          mesh = mesh,
                          mask = norway.poly,
                          format = 'sp',
                          biasfield = TRUE,
                          fun = 'linear',
                          n.samples = 1000) # Should "spatial" be false here?

saveRDS(fish_biasfield, "results/fish_biasfield.rds")
fish_biasfield <- readRDS("results/fish_biasfield.rds")

ggplot() +
  geom_polygon(data = norway, aes(long, lat, group = group),
               color="grey80", fill = "grey95") +
  gg(fish_biasfield$biasFields$artsobs) +
  labs(title = "Bias field") +
  plot_preferences

ggsave("figures/fishplot_biasfield.pdf", width = 3, height = 3)
```

This document took 0.01 hours to compile.