Simulated data: spatial smoothing

Natasha Klappstein

2024-01-12

This document contains code to reproduce the spatial smoothing example in Appendix B of Klappstein et al. (2024). Note that these results may be slightly different to those in the paper, as we used the `hmmSSF` package in the original analysis, but are using `amt` here for reader clarity. The general idea is to use a spatial smooth to capture select for an "unknown" centre of attraction.

Data and setup

```
# load packages

library(mgcv)
library(gratia)
library(ggplot2)
library(wesanderson)
library(dplyr)
library(terra)
library(cowplot)
library(amt)
```

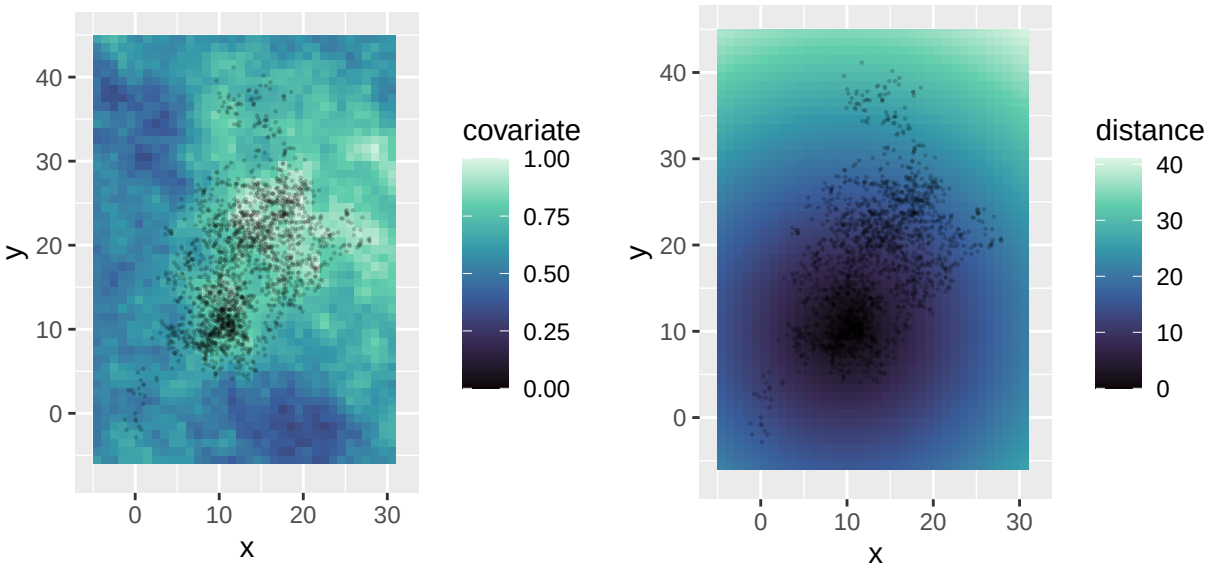
The data were simulated from a step selection function with selection for two covariates: i) one randomly generated covariate (called `cov`) with selection $\beta_{cov} = 5$, and ii) distance to a centre of attraction at coordinates (10, 10) (denoted `centre`) with selection $\beta_{centre} = -0.075$. We can load both of these data sources (location and covariate) and translate the covariates back into raster format.

```
# load and view location data
data <- readRDS("data/sim_data.RData")
head(data)
```

```
  ID           x          y                time
1  1  0.00000000  0.0000000 2020-01-01 00:00:00
2  1  0.09250911 -0.9318068 2020-01-01 01:00:00
3  1 -1.23033812 -0.7097249 2020-01-01 02:00:00
4  1  0.04876421 -0.7114203 2020-01-01 03:00:00
5  1  0.72878785 -1.9954392 2020-01-01 04:00:00
6  1 -1.02086198 -1.9767650 2020-01-01 05:00:00
```

```
# load and process raster data
cov_data <- readRDS("data/cov_data.RData")
cov <- rast(cov_data[[1]], type = "xyz")
centre <- rast(cov_data[[2]], type = "xyz")
names(cov) <- "cov"
```

Below are plots of the simulated locations overlaid onto each covariate raster.



Generating random points

In this example, we will use the `amt` package to generate random points. First, we need to transform our dataset into class `track_xy*`, then we can use `random_steps` to generate 20 random points for each observed location. We use the amt default, which uses a gamma distribution for random step lengths.

```
# create track_xy*
track <- make_track(data, x, y) |>
  steps()

# get random steps
set.seed(55)
ssf_data <- random_steps(track, n = 20)

# get covariate values
ssf_data <- extract_covariates(ssf_data, cov)
head(ssf_data)
```

```
# A tibble: 6 x 9
     x1_    x2_    y1_    y2_   sl_     ta_ case_ step_id_   cov
*  <dbl>  <dbl>  <dbl>  <dbl> <dbl>   <dbl> <lgl>    <dbl> <dbl>
1 0.0925 -1.23  -0.932 -0.710 1.34  -1.84   TRUE         3 0.559
2 0.0925 -0.223 -0.932 -0.412 0.608 -2.70   FALSE        3 0.516
3 0.0925  0.297 -0.932 -2.19  1.27   0.0625 FALSE        3 0.633
4 0.0925  0.344 -0.932 -0.633 0.391  2.34   FALSE        3 0.519
5 0.0925  0.121 -0.932 -1.09  0.165  0.0732 FALSE        3 0.556
6 0.0925 -0.926 -0.932 -1.57  1.20  -1.11   FALSE        3 0.572
```

Model fitting

Then, we can add a constant `times` variable for the Cox PH implementation and fit the model in `mgcv`. Our response is `times` in conjunction with the stratum ID (`step_id_`). We're including linear effects for step length `sl_` and the habitat covariate `cov`. However, rather than including the centre of attraction, we're assuming that we do not have that information. Instead, we will include a spatial smooth to account for any remaining spatial heterogeneity not explained by `cov`.

```r
# add dummy variable for times
ssf_data$times <- 1

# fit in mgcv
fit <- gam(cbind(times, step_id_) ~
             sl_ +
             cov +
             s(x2_, y2_),
           data = ssf_data,
           family = cox.ph,
           weights = case_)
```

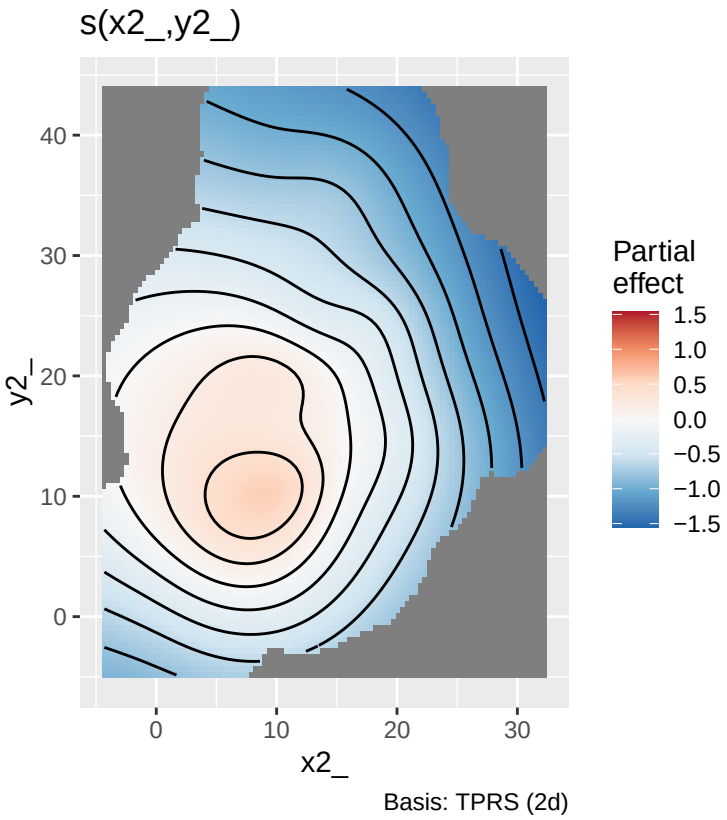
```r
summary(fit)
```

```
Family: Cox PH
Link function: identity

Formula:
cbind(times, step_id_) ~ sl_ + cov + s(x2_, y2_)

Parametric coefficients:
    Estimate Std. Error z value Pr(>|z|)
sl_  0.03327    0.02723   1.222    0.222
cov  4.92774    0.58652   8.402   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
           edf Ref.df Chi.sq p-value
s(x2_,y2_) 9.97  14.69  17.16   0.292

Deviance explained = -3.19%
-REML = 6035.3  Scale est. = 1         n = 41958
```

Model interpretation

Our summary indicates that we estimate the effect of the habitat covariate quite well, as the estimate is close to the truth ($\beta_{cov} = 5$). We can also plot the spatial smooth quickly with `gratia`.

```r
gratia::draw(fit, rug = FALSE)
```



We can also reproduce the plot in the paper with the following code, which allows us to better visualise how the spatial smooth is capturing the centre of attraction.

```
# get spatial smooth estimates
spatial <- smooth_estimates(fit, smooth = "s(x2_,y2_)")

# plot covariate
p1a <- ggplot() +
  geom_spatraster(data = cov, aes(fill = cov)) +
  coord_equal() +
  scale_fill_viridis_c(option = "mako", name = "covariate") +
  ylim(c(min(data$y) -4, max(data$y)+4)) +
  xlim(c(min(data$x)-4, max(data$x)+4)) +
  xlab("x") + ylab("y") +
  xlab("x") + ylab("y")

# plot centre of attraction
p2a <- ggplot() +
  geom_spatraster(data = centre, aes(fill = focal_sum)) +
  coord_equal() +
  scale_fill_viridis_c(option = "mako", limits = c(0, 41), name = "distance") +
  ylim(c(min(data$y) -4, max(data$y)+4)) +
  xlim(c(min(data$x)-4, max(data$x)+4)) +
  xlab("x") + ylab("y") +
  xlab("x") + ylab("y")

# plot spatial smooth
p3 <- ggplot(spatial, aes(x = x2_, y = y2_, fill = est)) +
```

```
  geom_raster() + coord_equal() +
  scale_fill_viridis_c(option = "rocket") +
  geom_point(data = data, aes(x = x, y = y, fill = ID),
             alpha = 0.2, size = 0.1)

# plot altogether
plot_grid(p1a, p2a, p3, nrow = 1)
```