Appendix S4: Code for Application 3.3

Jonsen et al

2022-06-23

Load required R packages

```
require(tidyverse)
require(ggspatial)
require(patchwork)
require(sf)
require(foieGras)
```

Fit `crw` SSM to harp seal track

```
## load track data
locs <- read.csv("../data/hase_ex33.csv")

fit <- fit_ssm(locs,
               model = "crw",
               time.step = 12,
               vmax = 4,
               control = ssm_control(verbose = 0))
```

Simulate tracks from SSM fit object

```
## load gradient rasters so we can use the potential function in simfit to keep
##  locations off land as much as possible
load(system.file("extdata/grad.rda", package = "foieGras"))

## ensure simulated tracks are same as in main text figure
set.seed(10000)

## simulate 100 tracks from SSM model fit object & apply potential function to
##  keep most locations off land (`grad` is the raster object)
trs <- simfit(fit,
              what = "predicted",
              reps = 100,
              grad = grad,
              beta = c(-350, -350))
```

Filter simulated tracks, keep top 10 % most similar to the SSM-predicted track

```
trs_flt <- sim_filter(trs, keep = 0.1, flag = 2)
```

Re-route any simulated locations occurring on land back to the water

```
trs_flt_rr <- route_path(trs_flt, map_scale = 50, buffer = 20000)
```

Map the simulated, filtered & re-routed tracks & overlay the SSM-predicted track

```
## create custom plots to display results
world_shp <- rnaturalearth::ne_countries(scale = 50, returnclass = "sf")

## To generate a plot with less distortion define a projection -
##  Lambert Azimuthal Equal Area
prj <- "+proj=laea +lat_0=60 +lon_0=-50 +datum=WGS84 +units=m +no_defs +ellps=WGS84"

## project & cast simulated locations to MULTIPOINT for faster map rendering
trs_sf <- trs %>%
  tidyr::unnest(cols = c(sims)) %>%
  sf::st_as_sf(coords = c('lon', 'lat'), crs = 4326) %>%
  sf::st_transform(crs = prj) %>%
  dplyr::group_by(id, rep) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTIPOINT")

## project & cast filtered locations to MULTIPOINT for faster map rendering
trs_flt_sf <- trs_flt %>%
  tidyr::unnest(cols = c(sims)) %>%
  sf::st_as_sf(coords = c('lon', 'lat'), crs = 4326) %>%
  sf::st_transform(crs = prj) %>%
  dplyr::group_by(id, rep) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTIPOINT")

## project & cast re-routed locations to MULTIPOINT for faster map rendering
trs_flt_rr_sf <- trs_flt_rr %>%
  tidyr::unnest(cols = c(sims)) %>%
  sf::st_as_sf(coords = c('lon', 'lat'), crs = 4326) %>%
  sf::st_transform(crs = prj) %>%
  dplyr::group_by(id, rep) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTIPOINT")

## project & cast SSM-predicted locations to MULTIPOINT for faster map rendering
plocs_sf <- grab(fit, "predicted", as_sf = TRUE) %>%
  sf::st_transform(crs = prj) %>%
  dplyr::group_by(id) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTIPOINT")

## project & cast SSM-predicted locations to MULTILINESTRING for track lines
pline_sf <- foieGras::grab(fit, "predicted", as_sf = TRUE) %>%
  sf::st_transform(crs = prj) %>%
  dplyr::group_by(id) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTILINESTRING")
```

```
## define map bounding box from simulated locations
bb_trs <- sf::st_bbox(trs_sf)

## map the simulated & SSM-predicted locations
p1 <- ggplot() +
  theme_minimal() +
  geom_sf(data = world_shp, colour = NA, fill = grey(0.6)) +
  geom_sf(data = trs_sf, size = 0.2, colour = "dodgerblue") +
  geom_sf(data = plocs_sf, size = 0.5, colour = "firebrick") +
  coord_sf(xlim = c(bb_trs[1], bb_trs[3]),
           ylim = c(bb_trs[2], bb_trs[4]),
           crs = prj,
           expand = TRUE) +
  scale_x_continuous(breaks = seq(from = -180, to = 180, by = 20)) +
  scale_y_continuous(breaks = seq(from = 0, to = 90, by = 15)) +
  ggspatial::annotation_scale(height = unit(1.5, "mm"), aes(location = "br"))

## define box to magnify in subsequent map
mbox <- structure(list(id = c(1,1,1,1,1),
                       lon = c(-53, -68, -68, -53, -53),
                       lat = c(45, 45, 51.25, 51.25, 45)),
                  class = "data.frame", row.names = c(NA, -5L))

mbox_sf <- mbox %>%
  sf::st_as_sf(coords = c("lon","lat"), crs = 4326) %>%
  sf::st_transform(crs = prj) %>%
  dplyr::group_by(id) %>%
  dplyr::summarise(geometry = sf::st_combine(geometry)) %>%
  sf::st_cast("POLYGON")

bb_mbox <- sf::st_bbox(mbox_sf)

## map the filtered simulated & SSM-predicted locations
p2 <- ggplot() +
  theme_minimal() +
  geom_sf(data = world_shp, colour = NA, fill = grey(0.6)) +
  geom_sf(data = trs_flt_sf, size = 0.2, colour = "dodgerblue") +
  geom_sf(data = plocs_sf, size = 0.4, colour = "firebrick") +
  geom_sf(data = mbox_sf, size = 0.75, colour = "orange", fill = NA) +
  coord_sf(xlim = c(bb_trs[1], bb_trs[3]),
           ylim = c(bb_trs[2], bb_trs[4]),
           crs = prj,
           expand = TRUE) +
  scale_x_continuous(breaks = seq(from = -180, to = 180, by = 20)) +
  scale_y_continuous(breaks = seq(from = 0, to = 90, by = 15)) +
  ggspatial::annotation_scale(height = unit(1.5, "mm"), aes(location = "br"))
```

```
## use highres land polygons for zoomed-in map
world_shp <- rnaturalearth::ne_countries(scale = 10, returnclass = "sf")

## map the filtered simulated, re-routed & SSM-predicted locations
p3 <- ggplot() +
  theme_minimal() +
  geom_sf(data = world_shp, colour = NA, fill = grey(0.6)) +
  geom_sf(data = trs_flt_sf, colour = "orange", size = 0.8) +
  geom_sf(data = trs_flt_rr_sf, colour = "dodgerblue", size = 0.8) +
  geom_sf(data = pline_sf, size = 0.2, colour = "firebrick") +
  geom_sf(data = plocs_sf, size = 0.4, colour = "firebrick") +
  coord_sf(xlim = c(bb_mbox[1], bb_mbox[3]),
           ylim = c(bb_mbox[2], bb_mbox[4]),
           crs = prj,
           expand = FALSE) +
  scale_x_continuous(breaks = seq(from = -180, to = 180, by = 5)) +
  scale_y_continuous(breaks = seq(from = 0, to = 90, by = 5)) +
  ggspatial::annotation_scale(height = unit(1.5, "mm"), aes(location = "tl"))

## arrange panels & render Figure 4 with `patchwork` package
p1 + p2 + p3 +
  patchwork::plot_layout(ncol = 2) +
  patchwork::plot_annotation(tag_levels = "a") &
  theme(plot.tag = element_text(size = 10))
```