Appendix S2: Code for Application 3.2

Jonsen et al

2022-11-30

Load required R packages

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

Fit `crw` SSM with 5-min `time.step` to time-regularise little penguin tracks

```
## Load GPS location data from .csv file
lipe <- read.csv("data/lipe_gps_ex32.csv")

## Load prey capture data from .csv file
lipe.pc <- read.csv("data/lipe_pc_ex32.csv")

## fit 6 `crw` SSMs, using: 1) speed filter (vmax) of 5 m/s to exclude any extreme
##   observations; 2) excluding any locations occurring < 5 s apart in time (min.dt);
##   3) 0.5, 1, 2, 4, 8, 16-min time.steps
ts <- c(0.5,1,2,4,8,16)
fits <- lapply(1:6, function(i) {
  fit_ssm(lipe, vmax=5, min.dt=5, model="crw", time.step=ts[i]/60)
})
```

Fit move persistence models with `fit_mpm` to SSM-predicted locations

```
## use `jmpm` model to fit jointly across the 4 penguin tracks
fmps <- lapply(fits, function(x) {
  fit_mpm(x, what = "predicted", model = "jmpm")
  })
```

Function to merge track & predation events

```
merge.tracks.preds <- function(tracks, preds){
  # drop tracks not in events
  tracks$id <- as.character(tracks$id)
  tracks <- tracks[tracks$id %in% unique(preds$id),]
  tracks$preyCount <- 0

  predsl <- split(preds, with(preds, interaction(id)), drop = TRUE)
  # Unique events
```

```
  uniqueEventsl2 <- lapply(predsl,
                          function(x) x[round(diff(as.POSIXct(x$date))) > 2,])

  # Loop events
  tracks <- split(tracks, tracks$id)
  for (i in 1:length(tracks)){
    # go through events and find the nearest track timestamp for each
    events <- uniqueEventsl2[[tracks[[i]]$id[1]]]
    events$idx <- sapply(events$date, function(dt)
      which.min(abs(as.numeric(difftime(dt, tracks[[i]]$date, unit='sec')))))
    # populate tracks
    for (idx in unique(events$idx)){
      tracks[[i]]$preyCount[idx] <- sum(events$idx == idx)
    }
  }
  # merge tracks back together
  tracks <- do.call(rbind, tracks)
  row.names(tracks) <- NULL
  return(tracks)
}
```

Aggregate prey capture events & append to locations

```
## grab SSM-predicted locations (use 2-min predictions)
peng.ssm_sf <- grab(fits[[3]], "p", as_sf = TRUE)

## aggregate prey capture events to location times & append
peng.ssm_sf <- merge.tracks.preds(peng.ssm_sf, lipe.pc)

## append move persistence estimates
peng.ssm_sf <- peng.ssm_sf %>% mutate(g = grab(fmps[[3]], "f")$g)
```

Plot move persistence time-series for 5-min prediction interval & map along SSM-predicted tracks

```
## plot move persistence (using 2-min) time-series for all 4 penguins, drop legend
p <- plot(fmps[[3]], ncol = 2, pages = 1, pal = "Plasma") &
  theme(legend.position = "none",
        plot.title = element_blank(),
        axis.text = element_text(size = 6),
        panel.grid.minor = element_blank())

## adjust x-axis date labels for clean manuscript figure
p[[1]] <- p[[1]] &
  scale_x_datetime(date_breaks = "3 hours", date_labels = "%H:%M")
p[[2]] <- p[[2]] &
  scale_x_datetime(date_breaks = "3 hours", date_labels = "%H:%M")
p[[3]] <- p[[3]] &
  scale_x_datetime(date_breaks = "4 hours", date_labels = "%H:%M")
p[[4]] <- p[[4]] &
  scale_x_datetime(date_breaks = "5 hours", date_labels = "%H:%M")
```

Map SSM-predicted locations with move persistence & prey capture estimates

```
## use foieGras::map to merge SSM & MPM model fits (SSM = fit, MPM = fmp);
##  use map tiles for better coastline resolution (Montague Is not in
##  `rnaturalearthhires` polygon data)

## define bounding box for map & expand limits to get desired map boundary
bb <- sf::st_bbox(peng.ssm_sf)
bb["xmin"] <- bb["xmin"] - diff(bb[c(1,3)]) * 0.7/2
bb["xmax"] <- bb["xmax"] + diff(bb[c(1,3)]) * 0.3/2
bb["ymin"] <- bb["ymin"] - diff(bb[c(2,4)]) * 0.2/2
bb["ymax"] <- bb["ymax"] + diff(bb[c(2,4)]) * 0.2/2

## customize mapping aesthetics
my.aes <- aes_lst(conf = F, line = T, mp_pal = hcl.colors(n=100, "Plasma"))
my.aes$df$size[1] <- 1

m <- foieGras::map(
  fits[[3]],
  fmps[[3]],
  what = "p",
  aes = my.aes,
  map_type = "cartolight",
  zoomin = 1,
  normalise = FALSE,
  silent = TRUE
) +
  geom_sf(data=peng.ssm_sf %>% filter(preyCount > 0),
          aes(size = preyCount, fill = g),
          shape = 21,
          stroke = 0.3,
          inherit.aes = TRUE) +
  scale_size(breaks = c(1,10,20),
             range = c(1, 4),
             name = "prey\ncaptures",
             guide = "none") +
  scale_fill_viridis_c(option = "C",
                       begin = min(peng.ssm_sf$g),
                       end = max(peng.ssm_sf$g),
                       guide = "none") +
  ggspatial::annotation_scale(height = unit(0.15, "cm"),
                              aes(location = "br")) +
  xlab(element_blank()) +
  ylab(element_blank()) +
  coord_sf(xlim = bb[c(1,3)],
           ylim = bb[c(2,4)],
           expand = TRUE,
           crs = sf::st_crs(peng.ssm_sf)) +
  scale_x_continuous(breaks = pretty(seq(150.13, 150.26, l = 4), n = 3)) +
  scale_y_continuous(breaks = pretty(seq(-36.5, -36.24, l = 5), n = 4)) +
  theme(legend.position = c(0.25,0.05),
        legend.direction = "horizontal",
        legend.key.width = unit(5, "mm"),
        legend.title = element_text(size = 8),
```

```r
        legend.text = element_text(size = 6),
        axis.text = element_text(size = 6),
        panel.grid = element_line(colour = "grey40"),
        panel.background = element_blank(),
        panel.ontop = TRUE)

## define track labels for map annotations
label.df <- data.frame(tag = c("a","b","c","d"),
                       x = c(0.3, 0.8, 0.15, 0.83) *
                        (bb["xmax"] - bb["xmin"]) + bb["xmin"],
                       y = c(0.85, 0.15, 0.4, 0.5) *
                        (bb["ymax"] - bb["ymin"]) + bb["ymin"])
names.df <- data.frame(tag = c("Montague\nIsland"),
                      x = 0.92 * (bb["xmax"] - bb["xmin"]) + bb["xmin"],
                      y = 0.91 * (bb["ymax"] - bb["ymin"]) + bb["ymin"])

m <- m +
  geom_text(data = label.df, aes(x,y,label=tag), size = 3) +
  geom_text(data = names.df, aes(x,y,label=tag), size = 2.5)

## make custom prey capture legend for location symbol size
pc.title <- data.frame(tag = "prey\ncaptures",
                      x = 0.05 *
                        (bb["xmax"] - bb["xmin"]) + bb["xmin"],
                      y = 0.1 *
                        (bb["ymax"] - bb["ymin"]) + bb["ymin"])
pc.df <- data.frame(x = c(0.15, 0.205, 0.26, 0.33) *
                        (bb["xmax"] - bb["xmin"]) + bb["xmin"],
                   y = rep(0.11, 4) *
                        (bb["ymax"] - bb["ymin"]) + bb["ymin"],
                   size = c(1, 1, 6, 15))
pc.labels <- data.frame(x = c(0.15, 0.205, 0.26, 0.33) *
                        (bb["xmax"] - bb["xmin"]) + bb["xmin"],
                   y = rep(0.08, 4) *
                        (bb["ymax"] - bb["ymin"]) + bb["ymin"],
                   tag = c("0", "1", "5", "10"))

m <- m +
  geom_text(data = pc.title,
            aes(x, y, label = tag),
            size = 2,
            inherit.aes = FALSE) +
  geom_point(data = pc.df[1,],
             aes(x, y),
             shape = 21,
             size = 1,
             fill = "#EDB300",
             colour = "#EDB300",
             inherit.aes = FALSE,
             show.legend = FALSE) +
  geom_point(data = pc.df[2:4,],
             aes(x, y, size = size),
             shape = 21,
```

```
             stroke = 0.3,
             fill = NA,
             inherit.aes = FALSE,
             show.legend = FALSE) +
  geom_text(data = pc.labels,
            aes(x, y, label = tag),
            size = 2,
            inherit.aes = FALSE)
```

Plot move persistence time-series for the different time scales

```
ggp <- lapply(1:4, function(i) {
  g05 <- grab(fmps[[1]], "f") %>% filter(id == unique(id)[i])
  g1 <- grab(fmps[[2]], "f") %>% filter(id == unique(id)[i])
  g2 <- grab(fmps[[3]], "f") %>% filter(id == unique(id)[i])
  g4 <- grab(fmps[[4]], "f") %>% filter(id == unique(id)[i])
  g8 <- grab(fmps[[5]], "f") %>% filter(id == unique(id)[i])
  g16 <- grab(fmps[[6]], "f") %>% filter(id == unique(id)[i])
  gg <- bind_rows(g05, g1, g2, g4, g8, g16) %>%
    mutate(interval = rep(ts,
                          c(
                            nrow(g05), nrow(g1), nrow(g2),
                            nrow(g4), nrow(g8), nrow(g16)
                          ))) %>%
    mutate(interval = as.factor(interval))

  p <- ggplot() +
    geom_path(data = gg, aes(date, g, colour = interval,
                             group = interval)) +
    geom_path(
      data = gg %>% filter(interval == 2),
      aes(date, g),
      colour = "#FB8072",
      linewidth = 1
    ) +
    scale_colour_manual(
      values = c(
        "#FFFFB3",
        "#BEBADA",
        "#FB8072",
        "#8DD3C7",
        "#FDB462",
        "#80B1D3"
      ),
      name = "prediction\ninterval (min)"
    ) +
    ylim(0, 1) +
    theme_grey() +
    theme(
      legend.title = element_text(size = 6),
      axis.title.y = element_text(size = 10),
      legend.position = "none",
      axis.text = element_text(size = 6),
```

```
      panel.grid.minor = element_blank()
    ) +
    xlab(element_blank()) +
    ylab(expression(gamma[t]))

  if(i==4) {
    p <- p + theme(legend.position = "bottom", #c(0.5, 0.12),
      legend.direction = "horizontal",
      legend.key.width = unit(0.03, "npc"),
      legend.key.height = unit(0.0001, "npc"),
      legend.text = element_text(size = 6),
      legend.background = element_blank())
  }
  return(p)
})

ggp[[1]] <- ggp[[1]] +
    scale_x_datetime(date_breaks = "3 hours", date_labels = "%H:%M")
ggp[[2]] <- ggp[[2]] +
    scale_x_datetime(date_breaks = "3 hours", date_labels = "%H:%M")
ggp[[3]] <- ggp[[3]] +
    scale_x_datetime(date_breaks = "4 hours", date_labels = "%H:%M")
ggp[[4]] <- ggp[[4]] +
    scale_x_datetime(date_breaks = "5 hours", date_labels = "%H:%M")
```

Render final plot

```
design <- "
1299
3499
5699
7899
"

## plot movement persistence 1-D time-series and map along 2-D tracks
ggp[[1]] + p[[1]] + ggp[[2]] + p[[2]] + ggp[[3]] + p[[3]] + ggp[[4]] + p[[4]] + m +
  patchwork::plot_layout(design = design) +
  patchwork::plot_annotation(tag_levels = "a") &
  theme(plot.tag = element_text(size = 9))
 grid::grid.draw(grid::textGrob("Number of prey captures per 2 min",
                                x = 0.55, y = 0.55, rot = 270,
                                gp = grid::gpar(fontsize = 10)))
```