Appendix S1: R code for Application 3.1

Jonsen et al

2022-11-30

Load required R packages

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

Fit `rw` and `crw` SSM's with 12-h `time.step` to southern elephant seal example data

```
## access data from foieGras
data(sese2, package = "foieGras")

## use only 1 seal track
sese2 <- subset(sese2, id == "ct36-E-09")

fit.rw <- fit_ssm(sese2, model = "rw", time.step = 12)
fit.crw <- fit_ssm(sese2, model = "crw", time.step = 12)
```

Calculate One-Step-Ahead (Prediction) residuals for both model fits

```
res.rw <- osar(fit.rw)
res.crw <- osar(fit.crw)
```

Create Figure 1

Panel A - model fits to observed locations

```
## Use sf package to project locations & convert SSM-estimated locations to lines
dat <- grab(fit.rw, "d", as_sf = TRUE) %>%
  sf::st_transform("+proj=stere +lon_0=90 +units=km")

loc.rw <- grab(fit.rw, "f", as_sf = TRUE) %>%
  sf::st_transform("+proj=stere +lon_0=90 +units=km")

line.rw <- loc.rw %>%
  dplyr::group_by(id) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTILINESTRING")

loc.crw <- grab(fit.crw, "f", as_sf = TRUE) %>%
  sf::st_transform("+proj=stere +lon_0=90 +units=km")
```

```r
line.crw <- loc.crw %>%
  dplyr::group_by(id) %>%
  dplyr::summarise(do_union = FALSE, .groups = "drop") %>%
  sf::st_cast("MULTILINESTRING")

line.ssm <- dplyr::bind_rows(line.rw, line.crw) %>%
  dplyr::mutate(SSM = c("rw","crw")) %>%
  dplyr::select(id, SSM, geometry)

## confidence ellipses
ssm.rw <- grab(fit.rw[1,], "f", as_sf=TRUE) %>%
  sf::st_transform("+proj=stere +lon_0=90 +units=km")
xy <- sf::st_coordinates(ssm.rw)
ssm.rw <- bind_cols(ssm.rw, xy) %>%
  rename(x=X, y=Y) %>%
  select(id,date, x, y, everything()) %>%
  split(., .$id)

conf_poly <- lapply(ssm.rw, function(x) {
          conf <- lapply(1:nrow(x), function(i)
            with(x, foieGras:::elps(x[i], y[i], x.se[i], y.se[i], 90)))
          tmp <- lapply(conf, function(x)
            sf::st_polygon(list(x)))
          sf::st_multipolygon(tmp)
        })

conf.rw_sf <- sf::st_as_sf(sf::st_as_sfc(conf_poly))
conf.rw_sf$id <- unique(fit.rw[1,]$id)

ssm.crw <- grab(fit.crw[1,], "f", as_sf=TRUE) %>%
  sf::st_transform("+proj=stere +lon_0=90 +units=km")
xy <- sf::st_coordinates(ssm.crw)
ssm.crw <- bind_cols(ssm.crw, xy) %>%
  rename(x=X, y=Y) %>%
  select(id,date, x, y, everything()) %>%
  split(., .$id)

conf_poly <- lapply(ssm.crw, function(x) {
          conf <- lapply(1:nrow(x), function(i)
            with(x, foieGras:::elps(x[i], y[i], x.se[i], y.se[i], 90)))
          tmp <- lapply(conf, function(x)
            sf::st_polygon(list(x)))
          sf::st_multipolygon(tmp)
        })

conf.crw_sf <- sf::st_as_sf(sf::st_as_sfc(conf_poly))
conf.crw_sf$id <- unique(fit.crw[1,]$id)

conf.ssm <- bind_rows(conf.rw_sf, conf.crw_sf) |>
  mutate(SSM = c("rw","crw")) |>
  select(id, SSM, x) |>
  rename(geometry = x)
sf::st_crs(conf.ssm) <- sf::st_crs(line.ssm)
```

```r
## contruct inset zoom box
zbox <- structure(list(id = c(1,1,1,1,1),
                       lon = c(107, 110.25, 110.25, 107, 107),
                       lat = c(-66.5, -66.5, -64.9, -64.9, -66.5)),
                  class = "data.frame", row.names = c(NA, -5L))

## project zoom box
prj <- sf::st_crs(line.ssm)

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

## get zoom box bounds
bb_zbox <- sf::st_bbox(zbox_sf)

## Plot SSM fits to observed locations
## main map
pA <- ggplot() +
  geom_sf(data = dat,
          col = "orange",
          shape = 3,
          size = 1) +
  geom_sf(data = conf.ssm,
          aes(fill = SSM),
          colour = NA,
          alpha = 0.35) +
  geom_sf(data = line.ssm,
            aes(colour = SSM)) +
  geom_sf(data = zbox_sf, linewidth = 0.5, colour = "black", fill = NA) +
  scale_colour_manual(values = c("firebrick", "dodgerblue")) +
  scale_fill_manual(values = c("firebrick", "dodgerblue"), guide = "none") +
  labs(x = element_blank(), y = element_blank()) +
  theme_minimal() +
  theme(axis.text = element_text(size = 8),
        legend.key.size = unit(c(6, 3), "mm"),
        legend.title = element_text(size = 9),
        legend.text = element_text(size = 8),
        legend.position = c(0.2,0.2))

## inset map
pAsub1 <- ggplot() +
  geom_sf(data = dat,
          col = "orange",
          shape = 3,
          size = 0.8) +
  geom_sf(data = conf.ssm,
          aes(fill = SSM),
          colour = NA,
          alpha = 0.05) +
  geom_sf(data = line.ssm,
          aes(colour = SSM),
          linewidth = 0.75) +
  coord_sf(xlim = c(bb_zbox[1], bb_zbox[3]),
           ylim = c(bb_zbox[2], bb_zbox[4]),
           crs = prj,
           expand = FALSE) +
  scale_colour_manual(values = c("firebrick", "dodgerblue"), guide = "none") +
  scale_fill_manual(values = c("firebrick", "dodgerblue"), guide = "none") +
  labs(x = element_blank(), y = element_blank()) +
  theme_minimal() +
  theme(axis.text = element_blank(),
        panel.background = element_rect(colour = "white"),
        panel.border = element_rect(fill=NA, linewidth=0.5))

## position inset on main map
pAsub1 <- ggplotGrob(pAsub1)

foo <- sf::st_bbox(dat)

pAA <- pA +
  annotation_custom(grob = pAsub1,
                            xmin = foo$xmin + (foo$xmax - foo$xmin) * 0.35,
                            xmax = foo$xmin + (foo$xmax - foo$xmin) * 1.30,
                            ymin = foo$ymin + (foo$ymax - foo$ymin) * 0.35,
                            ymax = foo$ymin + (foo$ymax - foo$ymin) * 1.05)
```

Panel B - plot time-series of prediction residuals for `rw` model

```r
## use `foieGras::plot.osar()`; modify with additional ggplot2 functions
pB <- plot(res.rw, "ts") +
  xlab(element_blank()) +
  ylab("Residuals") +
  theme(axis.text.x = element_text(size=7, angle=45),
        axis.title.y = element_text(size = 8),
        strip.text.y = element_blank(),
        axis.title.x = element_blank())
```

Panel C - plot autocorrelation functions of prediction residuals for `rw` model

```r
## use `foieGras::plot.osar()`; modify with additional ggplot2 functions
pC <- plot(res.rw, "acf") +
  theme(strip.text.y = element_blank(),
        axis.text = element_text(size = 8),
        axis.title = element_text(size = 8))
```

Panel D - plot time-series of prediction residuals for `crw` model

```r
## use `foieGras::plot.osar()`; modify with additional ggplot2 functions
pD <- plot(res.crw, "ts") +
  xlab(element_blank()) +
  ylab(element_blank()) +
  theme(axis.text.x = element_text(size=7, angle=45),
        strip.text.y = element_blank(),
        axis.title.x = element_blank())
```

Panel E - plot autocorrelation functions of prediction residuals for `crw` model

```
## use `foieGras::plot.osar()`; modify with additional ggplot2 functions
pE <- plot(res.crw, "acf") +
  ylab(element_blank()) +
  theme(strip.text.y = element_blank(),
        axis.text = element_text(size = 8),
        axis.title = element_text(size = 8))
```

Arrange panels & render figure 1 using the `patchwork` package

```
pAA / (pB | pD) / (pC | pE) +
  patchwork::plot_layout(widths = c(3,3,3),
              heights = c(3,1,1)) +
  patchwork::plot_annotation(tag_levels = "a") &
  theme(plot.tag = element_text(size = 9))
```