Ashy Storm-Petrel Catch-Per-Unit-Effort Estimation for Channel Islands National Park

Amelia J. DuVall & Emma Kelsey

v. 2025-09-19

This document builds on the metadata file ("sessions.csv") that documents mist-netting efforts for Ashy Storm-Petrels for the Channel Islands National Park (CHIS) Seabird Monitoring Program Database. Several new fields are added to the data file, including * Apparent Sunset ("app_sunset") * Standard Ending 5.3 hours Post-Sunset ("std_ending") * Total Effort in Minutes ("min") * Total Effort in Minutes with Standard Ending ("min_std") * Number of ASSP Captures ("ASSP") * Number of ASSP Captures with Standard Ending ("ASSPstd") * Catch-Per-Unit-Effort ("CPUEraw") * Catch-Per-Unit-Effort with Standard Ending ("CPUEstd")

This is v.2025-09-19

Read-in data

```
sessions <- readRDS(here::here("Data", "cleaned_data", "sessions.RDS")) # mistnet session data
captures <- readRDS(here::here("Data", "cleaned_data", "captures.RDS")) # banding data

# Check for duplicated session_IDs in sessions.
#This will affect joining data frames by session_ID.
any(duplicated(sessions$session_ID))
```

```
## [1] FALSE
```

```
# Check for duplicated catch_IDs in captures.
any(duplicated(captures$catch_ID))
```

```
## [1] FALSE
```

```
# Filter to ASSP spp only
# remove same night recaptures (SNRs) and unbanded individuals
ASSP <- captures %>%
  filter(species == "ASSP" & recapture != "SNR" & band_no != "notbanded")

notbanded <- captures %>%
  filter(species == "ASSP" & band_no == "notbanded")

LESP <- captures %>%
  filter(species == "LHSP" & recapture != "SNR" & band_no != "notbanded") %>%
  group_by(session_ID) %>%
  summarize(n = n())
```

Add new fields

Format sessions data

```
## function to convert difftime to minutes
mins <- function(x) lubridate::time_length(x, unit = "minutes")

## create dataframe to run sunset function on
sun_vec <- sessions %>%
  transmute(date = session_date, lat = lat, lon = long,
            site_code = site_code, session_ID = session_ID)

## add new variables to cpue df
cpue_v1 <- getSunlightTimes(data = sun_vec,
                            keep = "sunset", # compute sunset time
                            tz = "America/Los_Angeles") %>%
  mutate(std_ending = sunset + hours(5) + minutes(18), # 5.3 hours after sunset
         app_sunset = sunset) %>%
  select(-sunset) %>%  # keep only app_sunset
  left_join(sun_vec, by = c("date", "lat", "lon")) %>%
  rename(long = lon) %>%
  left_join(sessions, by = c("session_ID", "site_code", "lat", "long")) %>%
  # number of minutes that net was open (nets open/closed up to 5x/night)
  mutate(min_1 = mins(net_close_1 - net_open_1),
         min_2 = mins(net_close_2 - net_open_2),
         min_3 = mins(net_close_3 - net_open_3),
         min_4 = mins(net_close_4 - net_open_4),
         min_5 = mins(net_close_5 - net_open_5)) %>%
  # replace NAs in min columns with 0s
  mutate(across(all_of(paste0("min_", 1:5)), ~coalesce(.x, 0))) %>%
  # which came first, net close or 5.3 hours after sunset? use earlier
  mutate(net_close_1_std = pmin(net_close_1, std_ending),
         net_close_2_std = pmin(net_close_2, std_ending),
         net_close_3_std = pmin(net_close_3, std_ending),
         net_close_4_std = pmin(net_close_4, std_ending),
         net_close_5_std = pmin(net_close_5, std_ending)) %>%
  # number of minutes net was open using standard ending (5.3 hours after sunset)
  mutate(min_1_std = mins(net_close_1_std - net_open_1),
         min_2_std = mins(net_close_2_std - net_open_2),
         min_3_std = mins(net_close_3_std - net_open_3),
         min_4_std = mins(net_close_4_std - net_open_4),
         min_5_std = mins(net_close_5_std - net_open_5)) %>%
  # fill NAs with 0 and change any negative values to 0
  mutate(across(all_of(paste0("min_", 1:5, "_std")), ~pmax(coalesce(.x, 0), 0))) %>%
  # totals across all segments
  mutate(
    #total minutes of effort (uncapped)
    min = rowSums(across(all_of(paste0("min_", 1:5))), na.rm = FALSE),
    # total minutes capped at standed ending (5.3 hours after sunset)
    min_std = rowSums(across(all_of(paste0("min_", 1:5, "_std"))), na.rm = FALSE),
    # making sure there are no negative totals
    min = if_else(min < 0, 0, min),
    min = if_else(is.na(net_open_1), 0, min),
```

```
    # treat 0 mins (for std) as NA to indicate there was no effort
    min_std = if_else(min_std == 0, NA_real_, min_std),
    # if net did not open for the first time, then 0 minutes
    min_std = if_else(is.na(net_open_1), 0, min_std)) %>%
  # drop just the unneccessary columns
  select(-all_of(c(paste0("min_", 1:5), paste0("min_", 1:5, "_std")))) %>%
  arrange(session_year)

## check minute values
summary(cpue_v1$min/60) # more intuitive to think in terms of hours
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   3.200   4.833   4.164   5.283   8.833
```

```
summary(cpue_v1$min_std/60)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.000   3.026   3.990   3.476   4.426   5.284       2
```

Format capture data (ASSP only)

```
ASSP_std <- ASSP %>%
  # remove these fields since they exist in sessions w/ diff values
  dplyr::select(-c("flagged", "notes")) %>%
  # join ASSP captures and cpue_v1 dataframes
  left_join(cpue_v1, by = c("site_code", "session_ID", "lat", "long", "session_date",
                            "session_month", "session_day", "session_year", "island_code",
                            "subisland_code")) %>%
  # if bird was caught before std_ending give it a "1", otherwise give it a "0"
  mutate(std = if_else(std_ending > capture_date, "1", "0"),
         # assume individual is a breeder if brood patch (BP) indicates de/re-feathering
         assumeBreed = mosaic::derivedFactor(
           "Y" = (BP == "B" | BP == "2" | BP == "3" | BP == "4"),
           "N" = (BP == "D" | BP == "d" | BP == "0" | BP == "5" |
                    BP == "PD" | BP == "1" | BP == "1.5" | BP == "4.5"),
           .default = NA),
         # capture time (min) past sunset
         catchPastSS = capture_date - app_sunset) %>%
  filter(TRUE)

# summarize catches for each night
cpue <- ASSP_std %>%
  group_by(session_ID) %>%
  summarise(
            # total number of ASSP captured
            ASSP = n(),
            # total number of ASSP captured before standard ending
            ASSPstd = sum(std == "1"),
            # total number of brood patches recorded
            BPct = count(!is.na(BP)),
```

```
            # number of birds that have a broodpatch (2-4, B)
            BP_Y = sum(assumeBreed == "Y", na.rm = TRUE),
            # number of birds that don't have a broodpatch (1, 1.5, 4.5, 5, PB, D)
            BP_N = sum(assumeBreed == "N", na.rm = TRUE)) %>%
  ungroup() %>%
  right_join(cpue_v1, by= c("session_ID")) %>%
  mutate(
    # raw CPUE (number of ASSP captured per minute)
    CPUEraw =  ASSP/min,
    # standard CPUE (number of ASSP captured per min within standard ending)
    CPUEstd = ASSPstd/min_std,
    # frequency of birds that have a brood patch
    BPfreq_Y = BP_Y/BPct,
    # frequency of birds that don't have a brood patch
    BPfreq_N = BP_N/BPct) %>%
  dplyr::select("session_ID", "island_code", "subisland_code", "site_code", "site_name",
                "lat",  "long", "session_date", "session_year", "session_month",
                "session_day",  "series_ID",    "app_sunset",   "std_ending",   "net_open_1",
                "net_close_1",  "net_open_2", "net_close_2",    "net_open_3",   "net_close_3",
                "net_open_4",   "net_close_4", "net_open_5",    "net_close_5",  "min",
                "min_std", "ASSP",  "ASSPstd",  "CPUEraw",  "CPUEstd", "BPfreq_Y",
                "BPfreq_N", "net_mesh", "net_dim",  "spp_audio_file",   "dB_level",
                "speaker_system",   "org", "notes", "flagged",  "flagged_notes") %>%
  arrange(desc(session_year))

# Check new fields
chk <- cpue %>%
  filter(is.na(ASSP))

sIDs <- unique(chk$session_ID)

chkcaps <- captures %>%
  filter(session_ID %in% c(sIDs)) # no ASSP captures (banded) during these sessions

# Check code for missing BP scores
BPsNA <- ASSP %>%
  filter(is.na(BP))

BPsIDS <- unique(BPsNA$session_ID) # session_IDs with NA for BP scores

# Look at one of these sessions
chk.cpue <- cpue %>%
  filter(session_ID == "2018-08-15_SBI_ESP")

chk.cap <- ASSP %>%
  filter(session_ID == "2018-08-15_SBI_ESP") #%>%
  # group_by(BP) %>%
  # summarize(n = n())

chk.std <- ASSP_std %>%
  filter(session_ID == "2018-08-15_SBI_ESP")

# check CPUEstd
```

```r
summary(cpue$CPUEstd)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
## 0.003678 0.040904 0.071757      Inf 0.113211      Inf       21
```

Export out

```r
# Exporting as RDS will preserve date/time fields
#saveRDS(cpue, here("Data", "cleaned_data", "cpue.RDS"))
#write.csv(cpue, here("Data", "cleaned_data", "cpue.csv"), row.names = FALSE)
```