Killer whale Clock

Caila Kucheravy

2024-01-04

Run the killer whale skin clock developed by Parsons et al. (2023).

Prep:

```r
setwd("~/Documents/Master's/Analysis/Epigenetic Aging/Killer Whales")

library(tidyverse)
library(glmnet)
```

Load updated sample sheet:

```r
sample_sheet <- readRDS('output/updated_sample_sheet_combined_KillerWhale_array.rds')
```

Load the killer whale skin clock, select the correct columns and filter out NAs to have only CpGs used in the clock:

```r
clock <- read_csv('input/Table.WhaleS3.SkinClockCoef.csv')

kw_clock <- clock %>%
  select(var, Coef.Killerwhale.Skin.Sqrt) %>%      # Select the correct column for the KW clock
  filter(!is.na(Coef.Killerwhale.Skin.Sqrt))       # Filter out CpGs not used for this clock (50 CpGs)
```

Load normalized beta values:

```r
kw_betas <- readRDS('output/tbetas_corrected_combined_KillerWhale_array.rds')

# Filter for CpGs used in clock - 50 for killer whales
kw_betas_filtered <- kw_betas %>%
  select(any_of(kw_clock$var))
```

The age transformation used in the paper is: sqrt(Age+1)=Age. Need to back-transform for final age.

Form a weighted linear combination of the CpGs for killer whales:

```r
# Pivot clock wider to match beta table
kw_clock_wide <- kw_clock %>%
  pivot_wider(names_from = var, values_from = Coef.Killerwhale.Skin.Sqrt)

# Multiply beta values by the clock weights:
est_ages_kw <- data.frame(mapply('*', kw_betas_filtered, kw_clock_wide[,2:51]))

# Sum values, add intercept
est_ages_kw <- est_ages_kw %>%
  mutate(Sum = rowSums(est_ages_kw)) %>%
  mutate(Intercept = kw_clock_wide$`(Intercept)`)

# Sum intercept and weighted beta values
```

```
est_ages_kw <- est_ages_kw %>%
  mutate(Ages = rowSums(est_ages_kw[,c("Sum", "Intercept")]))

# Age transformation: DNAmAge = F^(-1) (x*beta)
est_ages_kw <- est_ages_kw %>%
  mutate(Age_Transformed = (est_ages_kw$Ages^2) - 1)

est_ages_kw$Age_Transformed
```

```
##   [1] 14.256375  3.650937 17.570467  6.414204  3.434888  3.913672 25.289485
##   [8]  7.734235 23.257862  7.470849  6.786813  8.218794 11.350434 10.369020
##  [15]  8.731627  9.948638 12.000812  9.315785  7.140396  6.627101  7.932776
##  [22] 12.712197 15.930368  5.030084  5.437012 18.925497  7.789091  5.988418
##  [29]  7.347518  5.765844  4.183960  3.177588 12.108077  4.612530 12.841012
##  [36] 15.634721 10.667820 12.701103  3.385554  8.779063  9.176903  4.298312
##  [43]  7.294429 14.988458  9.314062  2.887993 22.468361 23.615180  9.065264
##  [50] 11.790230 20.186232 11.200262  6.956046  5.922878 29.251329  3.102473
##  [57]  5.459565  4.480056  4.984759  8.101788  9.068439  6.308725 10.764665
##  [64] 12.387284  6.667874  2.960607 14.696543  3.009963  3.344104 12.910515
##  [71] 12.964907  6.070538  9.738125  8.691357 11.665770 11.033129 14.122635
##  [78]  8.734187 10.284856  9.164254 16.533763 15.735843  8.255537  9.297317
##  [85] 12.497050 11.071359 10.038349  7.628992  9.070211  5.612963  7.631208
##  [92] 13.087478 14.853402  9.355239 13.765679 14.110114  2.756090 15.160602
##  [99] 12.051822
```

Add ages to sample sheet:

```
# Add column with basename back to dataframe with ages
est_ages_kw$chip.ID.loc <- sample_sheet$chip.ID.loc

# Select chip ID and ages to join with sample sheet
DNAm_ages_kw <- est_ages_kw %>%
  select(chip.ID.loc, Age_Transformed)

# Join ages with sample sheet
kw_ages <- sample_sheet %>%
  left_join(DNAm_ages_kw, by = "chip.ID.loc")
```

Take a look at the duplicates:

```
# Note that KW-2019-06 was a technical replicate (from the same DNA sample), while the others were from
duplicates <- kw_ages %>%
  arrange(block) %>%
  group_by(block) %>%
  filter(n() > 1)
duplicates
```

```
## # A tibble: 43 x 16
## # Groups:   block [20]
##    Order block  sampleId Species  Year Location Sex     lab plate chip.id stripe
##    <int> <chr>  <chr>    <chr>   <int> <chr>    <chr> <dbl> <dbl> <chr>   <chr>
##  1    29 ARPI-~ ARPI-20~ Orcinu~  2013 Eclipse~ F         1     2 207222~ R05C01
##  2    37 ARPI-~ KW-2020~ Killer~  2020 Cumberl~ F         2     3 206139~ R01C01
##  3    32 ARPI-~ ARPI-20~ Orcinu~  2013 Eclipse~ F         1     2 207222~ R02C02
##  4    33 ARPI-~ ARPI-20~ Orcinu~  2013 Eclipse~ F         1     2 207222~ R03C02
##  5    48 ARPI-~ ARPI-20~ Orcinu~  2013 Eclipse~ F         1     2 207222~ R06C02
##  6    24 ARPI-~ KW-2020~ Killer~  2020 Cumberl~ F         2     3 206116~ R06C02
##  7    35 ARPI-~ KW-2020~ Killer~  2020 Cumberl~ F         2     3 206139~ R05C02
##  8    55 ARPI-~ ARPI-20~ Orcinu~  2018 Eclipse~ F         1     1 207222~ R01C02
##  9    56 ARPI-~ ARPI-20~ Orcinu~  2018 Eclipse~ F         1     1 207222~ R02C02
## 10    23 ARPI-~ ARPI-20~ Orcinu~  2018 Eclipse~ F         1     1 207222~ R05C02
## # i 33 more rows
## # i 5 more variables: row <chr>, column <chr>, chip.ID.loc <chr>,
## #   Basename <chr>, Age_Transformed <dbl>
```

Most of the ages estimate for duplicate samples are quite close, but some of the recaptures don't reflect the difference in age between captures.

We also have a few known-age individuals. Unfortunately, two of the known-age individuals (ARSQ-xx-1379, 34yo and KW-CH-2011, 35 yo) had to be removed in the quality control stage.

We can compare the DNAm estimated age to the known age (determined from GLGs):

```
known_ages <- data.frame(
  sampleId = c("ARRB-xx-1291", "ARSQ-xx-1397", "KW-2022-PI-01"),
  Known_Age = c(28, 6, "In progress")
)

est_ages <- kw_ages %>%
  select(sampleId, Age_Transformed) %>%
  filter(sampleId %in% c("ARRB-xx-1291", "ARSQ-xx-1397", "KW-2022-PI-01"))

compare_known_ages <- known_ages %>%
  left_join(est_ages, by = "sampleId")
compare_known_ages
```

```
##        sampleId   Known_Age Age_Transformed
## 1  ARRB-xx-1291          28       29.251329
## 2  ARSQ-xx-1397           6        3.102473
## 3  ARSQ-xx-1397           6        5.459565
## 4 KW-2022-PI-01 In progress        8.779063
```

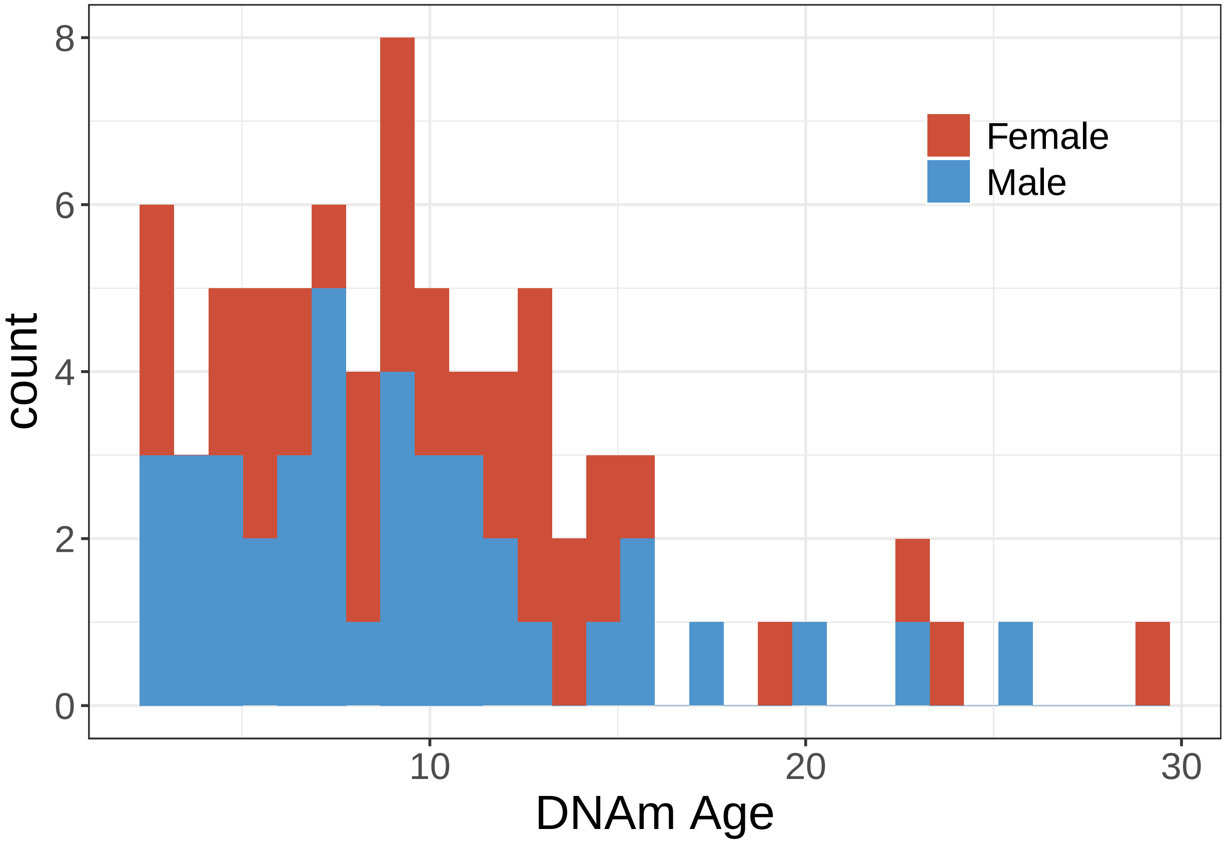
Remove the duplicates and add the difference between year and 2022:

```
# Taking the first age for now - but might want to take average or something for final ages.
kw_ages_dupsRemoved <- kw_ages %>%
  mutate(duplicate = duplicated(block)) %>%
  filter(!duplicate == "TRUE")

# Add the difference in years so that the age structure represents what it would be in 2022:
kw_ages_dupsRemoved <- kw_ages_dupsRemoved %>%
  select(!duplicate) %>%
  mutate(diffYear = 2022 - kw_ages_dupsRemoved$Year) %>%
  mutate(adj_age = Age_Transformed + diffYear)

#Write the age file to csv:
#write.csv(kw_ages_dupsRemoved, "kw_ages.csv")

# Keep only Cumberland Sound & Northern Baffin Island samples ("High Arctic" group):
Locations <- c("Cumberland Sound", "Eclipse Sound", "Newfoundland", "Saint Pierre et Miquelon")

kw_ages_HA <- kw_ages_dupsRemoved %>%
  filter(Location %in% Locations) %>%
  # And the one 2013 CS sample that grouped with the Greenland samples
  filter(!sampleId == "ARPG-2013-01")
```

Plot the data (unadjusted age):

```
cols <- c("tomato3", "steelblue3")

ggplot(kw_ages_dupsRemoved, aes(x = Age_Transformed, fill = Sex)) +
  geom_histogram() +
  xlab("DNAm Age") +
  #scale_x_continuous("DNAm Age", limits = c(0,55), breaks = c(0,5,10,15,20,25,30,35,40,45,50,55)) +
  scale_fill_manual(values = cols,
                    labels = c("Female", "Male")) +
  theme_bw() +
  theme(axis.text = element_text(size=14),
        axis.title = element_text(size=18),
        legend.title = element_blank(),
        legend.text = element_text(size = 14),
        legend.position = c(0.82,0.8),
        legend.background = element_blank())
```

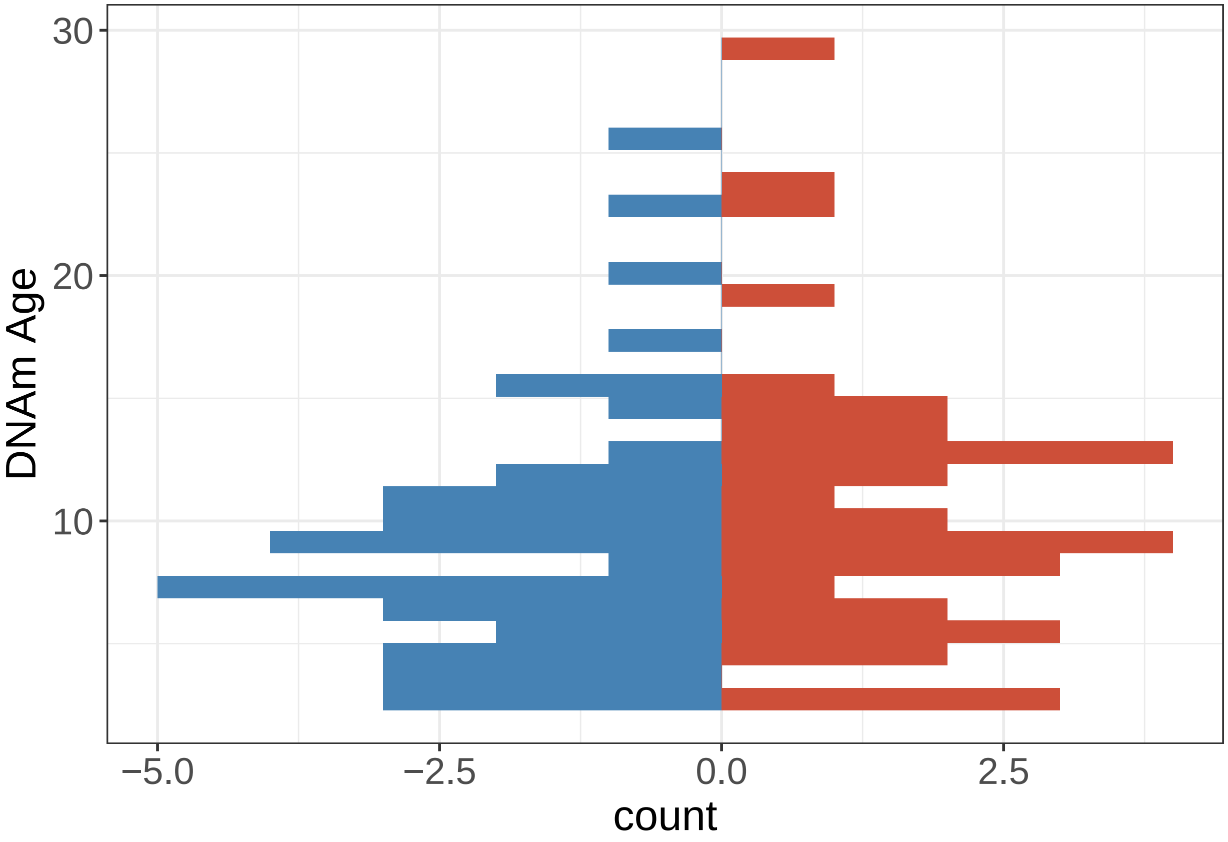
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
males <- kw_ages_dupsRemoved %>%
  filter(Sex == "M")
females <- kw_ages_dupsRemoved %>%
  filter(Sex == "F")
```

```
g <- ggplot() +
  geom_histogram( data = females,
    aes(x = Age_Transformed, y = after_stat(count)),
    fill="tomato3") +
  geom_histogram( data = males,
    aes(x = Age_Transformed, y = -after_stat(count)),
    fill= "steelblue") +
  theme_bw()

#tiff("Age_Distribution_2.tiff", units="in", width=8, height=5, res=400)

g +
  coord_flip() +
  xlab("DNAm Age") +
  #scale_x_continuous(limits = c(0,55), breaks = c(0,10,20,30,40,50)) +
  #scale_y_continuous("DNAm Age", limits = c(-5,5), breaks = c(-5,-3,-1,1,3,5))
  theme(axis.text = element_text(size=14),
        axis.title = element_text(size=16))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



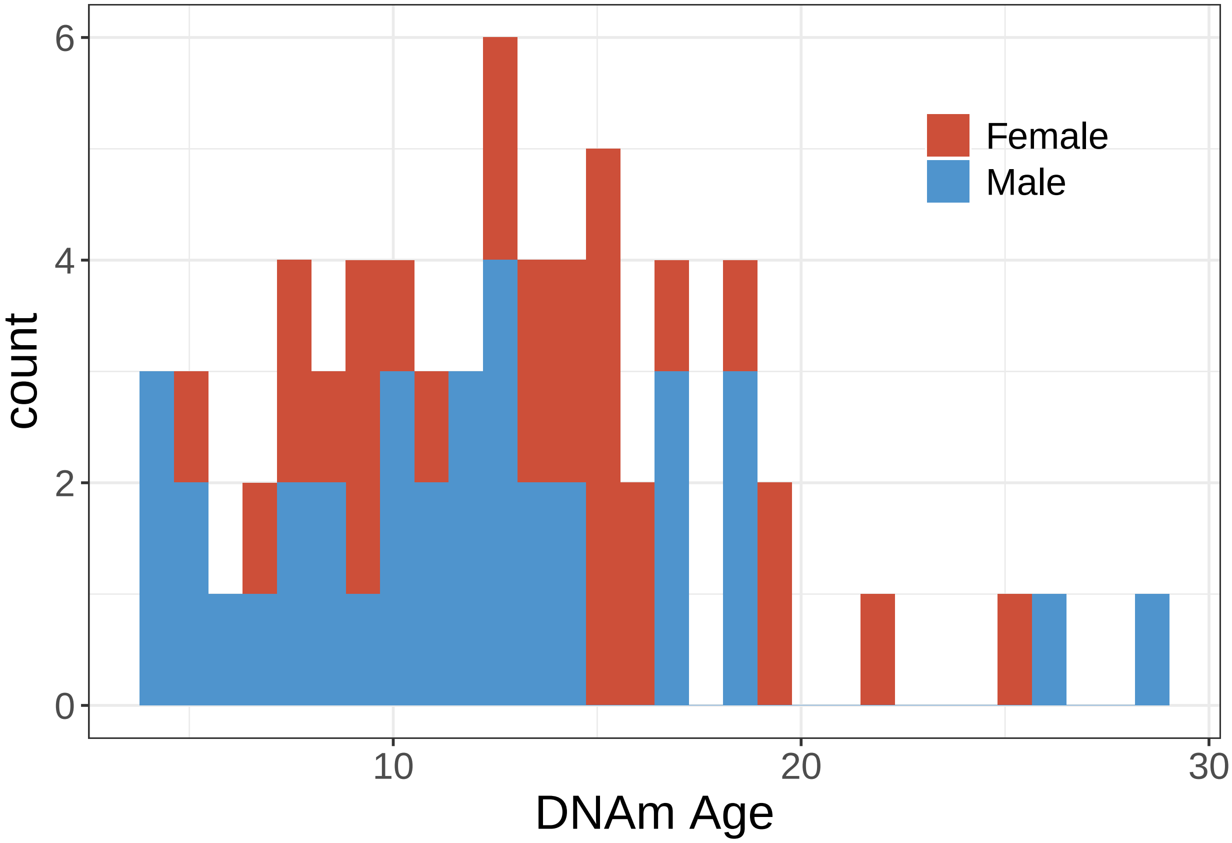
```
#dev.off()
```

Plot the data (adjusted 2022 age):

```
#tiff("age_structure_no.calves.tiff", units="in", width=6, height=4, res=500)
```

```
ggplot(kw_ages_HA, aes(x = adj_age, fill = Sex)) +
  geom_histogram() +
  xlab("DNAm Age") +
  #scale_x_continuous("DNAm Age", limits = c(0,55), breaks = c(0,5,10,15,20,25,30,35,40,45,50,55)) +
  scale_fill_manual(values = cols,
                    labels = c("Female", "Male")) +
  theme_bw() +
  theme(axis.text = element_text(size=14),
        axis.title = element_text(size=18),
        legend.title = element_blank(),
        legend.text = element_text(size = 14),
        legend.position = c(0.82,0.8),
        legend.background = element_blank())
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
#dev.off()
```

Summarize data:

```
males <- kw_ages_dupsRemoved %>%
  filter(Sex == "M")

females <- kw_ages_dupsRemoved %>%
  filter(Sex == "F")

length(males$Age_Transformed)
```

```
## [1] 40
```

```
summary(males$Age_Transformed)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.756   5.700   8.501   9.392  11.317  25.289
```

```
summary(males$adj_age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.888   8.278  11.986  12.667  15.175  30.186
```

```
length(females$Age_Transformed)
```

```
## [1] 36
```

```
summary(females$Age_Transformed)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.961   6.308   9.632  10.813  13.257  29.251
```

```
summary(females$adj_age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.961   9.581  14.631  14.924  17.219  42.251
```

```
# Juveniles
kw_ages_dupsRemoved %>%
  dplyr::count(adj_age < 10)
```

```
##   adj_age < 10  n
## 1        FALSE 52
## 2         TRUE 24
```

```
24/76
```

```
## [1] 0.3157895
```

```
# Reproductive adults
kw_ages_dupsRemoved %>%
  dplyr::count(adj_age < 35)
```

```
##   adj_age < 35  n
## 1        FALSE  1
## 2         TRUE 75
```

```
(75-24)/76
```

```
## [1] 0.6710526
```

```
# Post-reproductive adults
kw_ages_dupsRemoved %>%
  dplyr::count(adj_age > 35)
```

```
##   adj_age > 35  n
## 1        FALSE 75
## 2         TRUE  1
```

```
1/76
```

```
## [1] 0.01315789
```