Lostine Longitudinal Data Analyses

Raina Plowright, Kezia Manlove, Tom Besser, Pat Matthews, David Paex, Kim Andrews, Lisette Waits, Peter Hudson, Frances Cassirer

10/09/2016

Set-up

Load required packages.

```
require(lme4)
require(graphics)
require(shape)
require(hwde)
```

Data Preparation

This analysis rests on three datasets: a compiled dataset of population-level counts for the Lostine herd through time, a "study-sheep" dataset with time-invariant information on each sampled animal, and a dataset of samples, with information on each animal at each sampling event.

Biological year

For the purposes of this study, we relabeled the biological year for a single May sampling event (a capture of 04LO65 on May 2nd, 2012) as occurring in the 2011 biological year (normally, the cut-off point separating biological years is May 1st).

Age

Age was assigned on handling using horn and tooth eruption for animals first handled at less than four years of age. We classified animals into three broader groups (lambs, yearlings, adults), and then determined more specific ages for all adults.

The basic logic of age assignment was to treat each animal's age as a draw from a multinomial distribution. We combined information on recruitment in each year (e.g., cohort size) with age-specific survival estimates from Manlove et al. 2016 Ecology. For each animal, we identified the most likely birth cohort.

Read in the compiled data, which has cohort sizes for each year in spring, and cut down to just the data from the Lostine.

```
compd.data <- read.csv("./Data/compiled_data_summary_151115b.csv", header = T)
#los <- subset(compd.data, Pop == "Lostine" & year >= 1990)
```

Enter the posterior survival probabilities from Manlove et al. 2016 Ecology

```
krm.surv.probs.he <- c(rep(.99, 2),
                       rep(.90, 4),
                       rep(.84, 6),
                       rep(.85, 7),
```

```
                        rep(0, 8)
                        # fill in 0's to be sure vector is long-enough for  next for loop.
                        )
```

Build a 27x27 cohort matrix, since there are 27 possible birth cohorts for animals in this study.

```
cohort <- matrix(NA, nrow = 27, ncol = 27)
for(i in 1:27){
  k <- subset(los, year == (1989 + i))
    for(j in 1:27){
      cohort[i, j] <- ifelse(j<i, 0, k$Lambs * prod(krm.surv.probs.he[1:(j - i)]))
  }
}
```

Read in sampling data, and individual-level study sheep data.

```
# likelihood an animal comes from a given cohort
samples.full <- read.csv("./Data/LostineSample2016Final_WithClasses_20160929.csv", header = T)
studysheep.in <- read.csv("./Data/Study_sheep_20151215.csv", header = T)
```

At this point, we'll also read in the full.dataset, which is the same as samples

```
#full.data <- read.csv("./Data/LostineSample2016Final_WithClasses_20160929.csv", header = T)
#full.data <- subset(full.data, movi_qpcr != "Indeterminate")
samples.full$qPCRResult <- ifelse(samples.full$movi_qpcr == "Detected" | samples.full$movi_qpcr == "Ind
```

Build an indicator for whether each animal's age is known (e.g., it has a post-mortum tooth age or its age at entry was less than four).

```
studysheep.in$KnownAge <- ifelse(!(is.na(studysheep.in$Tooth_Age)) | studysheep.in$AgeatEntry < 4, 1, 0
```

Loop over all sampled animals. Extract animal-level data for animal i from study sheep. Determine whether the animal needs an age assignment, or whether its age is known. Figure out the range of possible cohorts from which the animal could have come ($kentry_bioyr[1] - k$age_entry[1]). Get a list of possible cohorts for the animal. Pull out the rows from the cohort matrix build above that correspond to each possible birth year, and normalize over all possible birth years for that animal. The most likely birth year is the maximum of those possible year likelihood.

```
animals <- start.cohort <- last.cohort <- omit <- most.likely.cohort <- known.age <- rep(NA, length(leve
cohort.dist <- matrix(0, nrow = length(animals), ncol = 27)
years.list <- vector("list", length(animals))
for(i in 1:length(animals)){
  k <- subset(samples.full, id == levels(samples.full$id)[i])
  studysheep <- subset(studysheep.in, as.character(Animal_ID) == levels(samples.full$id)[i])
  animals[i] <- as.character(levels(samples.full$id)[i])
  known.age[i] <- ifelse(is.na(studysheep$Tooth_Age[1]) == T & k$age_entry[1] >= 4, 0, 1)
  if(k$age_entry[1] <= 3.9){
    start.cohort[i] <- floor(k$entry_bioyr[1] - k$age_entry[1])
    last.cohort[i] <- ceiling(k$entry_bioyr[1] - k$age_entry[1])
  } else{
    start.cohort[i] <- floor(max(k$cap_bioyr) - 20)
```

```
    last.cohort[i] <- ceiling(k$entry_bioyr[1] - k$age_entry[1])
  }
  years.list[[i]] <- seq(floor(start.cohort[i]), ceiling(last.cohort[i]), by = 1)
  omit[i] <- ifelse(max(k$cap_bioyr) <= 2005, 1, 0)

  if(omit[i] == 0){
    cohort.dist[i, years.list[[i]] - 1989] <- cohort[(years.list[[i]] - 1989),
                                                     (k$cap_bioyr[1] - 1989)] / sum(cohort[(years.list[
                                                                                           (k$cap_bioyr
  }
  most.likely.cohort[i] <- seq(1990, 2016, by = 1)[which.max(cohort.dist[i, ])]
}
```

Build a dataframe with these new fields.

```
# build animal dataframe with ages
animal.data <- as.data.frame(cbind(as.character(animals),
                                   start.cohort,
                                   last.cohort,
                                   most.likely.cohort,
                                   omit,
                                   known.age,
                                   round(cohort.dist, 3)))
names(animal.data) <- c("Animal.ID", "FirstPossibleCohort", "LastPossibleCohort", "MostLikelyCohort", "
                        paste("P.", as.character(seq(1990, 2016, by = 1)), sep= ""))
animal.data <- subset(animal.data, Omit == 0)
animal.data$MostLikelyCohortOrig <- animal.data$MostLikelyCohort
```

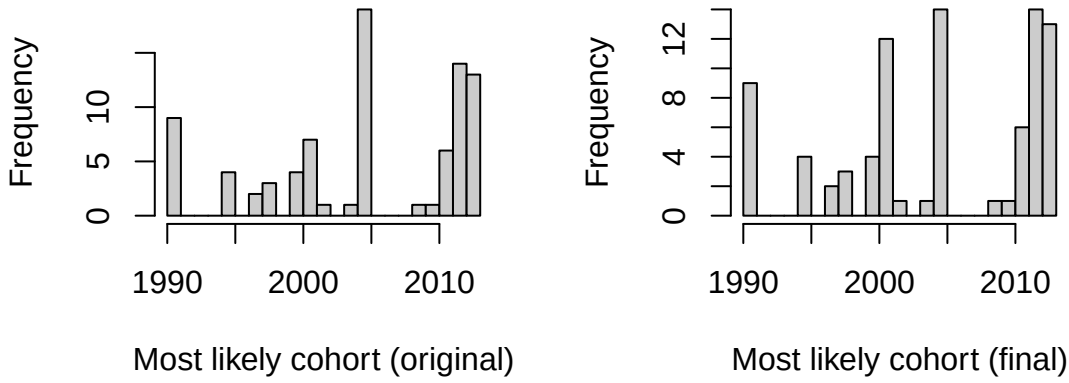
Now, loop over animals again, and figure out whether the most likely cohorts are too full. If a cohort is too full (e.g., there are more animals assigned to that cohort than there were recruits from that cohort in compiled data), reassign the animals with the lowest likelihood of being in that cohort to their next-most-likely cohort. Rinse and repeat until all cohorts are of acceptable sizes (e.g., not bigger than the number of recruits in compiled data).

```
# now, adjust for possible number of animals in each cohort
most.likely.cohort.adj <- rep(NA, length(levels(factor(animal.data$Animal.ID))))
for(i in 1:length(levels(factor(animal.data$Animal.ID)))){
  # 1. subset animal data to get all animals assigned to this animal's most likely cohort.
  k <- subset(animal.data, MostLikelyCohort == animal.data$MostLikelyCohort[i])
  animal.likelihoods <- subset(animal.data, Animal.ID == levels(factor(animal.data$Animal.ID))[i])[, 7:
  # 2. determine how many of those animals have known ages
  known.age.in.cohort <- dim(subset(k, KnownAge == 1))[1]
  # 3. get the max possible cohort size, using the Lambs field in the lostine compiled data
  possible.animals <- subset(los, year == animal.data$MostLikelyCohort[i])$Lambs
  # 4. figure out how many possible ewes of unknown age exist
  possible.ewes.unknown.age <- ceiling((possible.animals - known.age.in.cohort) / 2)
  # 5. if there are more possible animals of unknown age than there are animals of unknown ages whose
  #    most likely cohort is this one, assign all those animals to this cohort.
  if(possible.ewes.unknown.age >= (dim(k)[1] - known.age.in.cohort)){
    most.likely.cohort.adj[i] <- as.numeric(as.character(animal.data$MostLikelyCohort[i]))
  } else {
    # 6. pull out likelihoods for each animal in k in the cohort year
    cohort.name <- paste("P.", k$MostLikelyCohort[1], sep = "")
```

```
    cohort.yr.likelihoods <- subset(k, select = cohort.name)
    likelihood.ranks <- rank((1 - as.numeric(as.character(cohort.yr.likelihoods[ ,1]))))
    in.this.cohort <- k[likelihood.ranks <= possible.ewes.unknown.age, ]
    if(levels(factor(animal.data$Animal.ID))[i] %in% in.this.cohort$Animal.ID){
      most.likely.cohort.adj[i] <- as.numeric(as.character(k$MostLikelyCohort[1]))
    } else {
      next.highest <- strsplit(x = names(animal.likelihoods)[which.max(as.numeric(as.character(as.vecto
      animal.data$MostLikelyCohort[which(animal.data$Animal.ID == levels(factor(animal.data$Animal.ID))
    }
  }
}
#table(is.na(most.likely.cohort.adj))
# iterate through the above for-loop until there are no NAs left.
```

Plot out the age distributions to be sure MostLikelyCohortOrig and Most LikelyCohort aren't identical

```
par(mfrow = c(1, 2))
hist(as.numeric(as.character(animal.data$MostLikelyCohortOrig)), breaks = 20,
     main = "", xlab = "Most likely cohort (original)", col = "grey80")
hist(as.numeric(as.character(animal.data$MostLikelyCohort)), breaks = 20,
     main = "", xlab = "Most likely cohort (final)", col = "grey80")
```



Attach animal cohort data to animal handling dataframe (to associate ages with each sampling event)

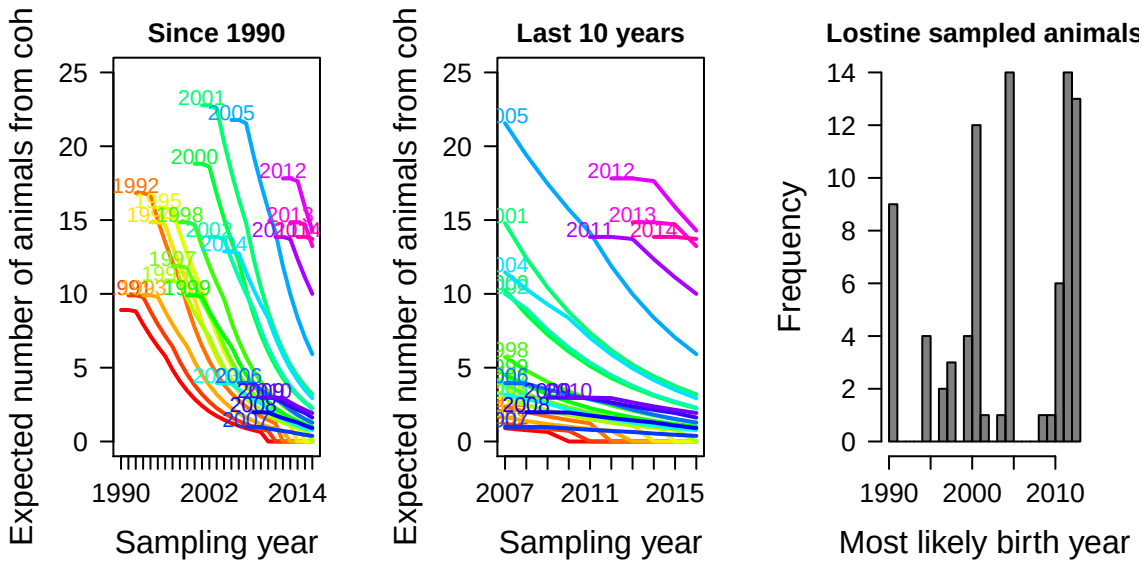
```
samples.full$MostLikelyAge <- rep(NA, dim(samples.full)[1])
for(i in 1:dim(samples.full)[1]){
  k <- subset(animal.data, as.character(Animal.ID) == as.character(samples.full$id)[i])
  samples.full$MostLikelyAge[i] <- as.numeric(as.character(samples.full$cap_bioyr[i])) - as.numeric(as.
}

# cut samples down to just animals that weren't omitted
# samples <- subset(samples.full, id %in% levels(factor(animal.data$Animal.ID)))

#write.csv(samples, "./Data/LostineSampleData_20160505_v1.csv")
```

Cut down to just those animals who have been handled since 2005 (these animals are in the dataframe "samples")

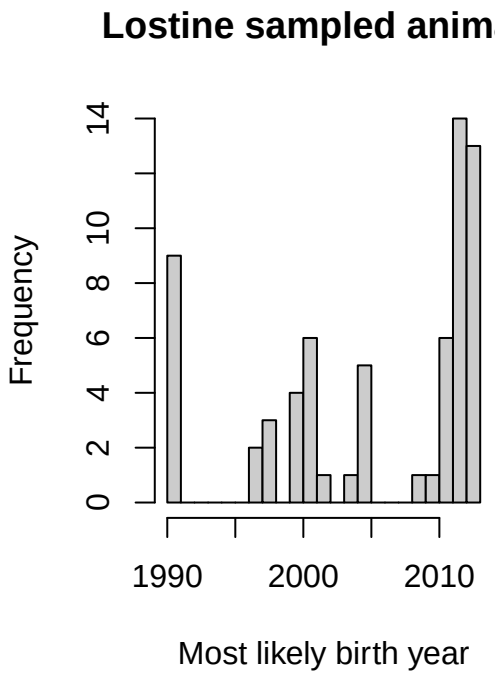
```r
samples <- subset(samples.full, id %in% levels(factor(animal.data$Animal.ID)))
# samples cuts out all animals in samples.full who haven't been sampled since 2005
```

Build some plots to look at age structure.

```r
# KRM age structure plots and analysis of quadrature (DON'T USE MY ANALYSIS OF QUADRATURE; DAVID'S/ALAN
rainbow.cols <- rainbow(n = 27)

par(mfrow = c(1, 3), las = 1, cex.axis = 1.2, cex.lab = 1.5, mar = c(5, 5, 2, 2))
plot(cohort[1, ] ~ seq(1990:2016), type = "l", ylim = c(0, 25),
     col = rainbow.cols[1],
     lwd = 2, xaxt = "n",
     main = "Since 1990",
     xlab = "Sampling year",
     ylab = "Expected number of animals from cohort")
axis(side = 1, at = seq(1990:2016), labels = c(1990:2016))
for(i in 2:27){
  lines(cohort[i, i:27] ~ seq(1990:2016)[-c(1:(i - 1))],
        col = rainbow.cols[i],
        lwd = 2)
  x.label <- (1989 + i)
  text(x = i, y = cohort[i, i] + .5, labels = as.character(x.label),
       col = rainbow.cols[i])
}

plot(cohort[1, 18:27] ~ seq(2007,2016, by = 1), type = "l", ylim = c(0, 25),
     col = rainbow.cols[1],
     lwd = 2, xaxt = "n",
     main = "Last 10 years",
     xlab = "Sampling year",
     ylab = "Expected number of animals from cohort")
axis(side = 1, at = seq(2007,2016, by = 1), labels = c(2007:2016))
for(i in 2:27){
  start <- ifelse(i >= 18, i, 18)
  #  seq.in <- ifelse(i >= 18, seq(2007:2016)[-c(1:(i - 1))], seq())
  lines(cohort[i, start:27] ~ seq(1990,2016, by = 1)[-c(1:(start - 1))],
        col = rainbow.cols[i],
        lwd = 2)
  x.label <- (1989 + i)
  text(x = (1989 + start), y = cohort[i, start] + .5, labels = as.character(x.label),
       col = rainbow.cols[i])
}

known.age.sheep <- subset(animal.data, KnownAge == 1)
hist.breaks <- hist(as.numeric(as.character(animal.data$MostLikelyCohort)),
                    col = rgb(0, 0, 0, alpha = .5), #col = rainbow.cols,
                    breaks = 20, xlab = "Most likely birth year",
                    main = "Lostine sampled animals")$breaks
```

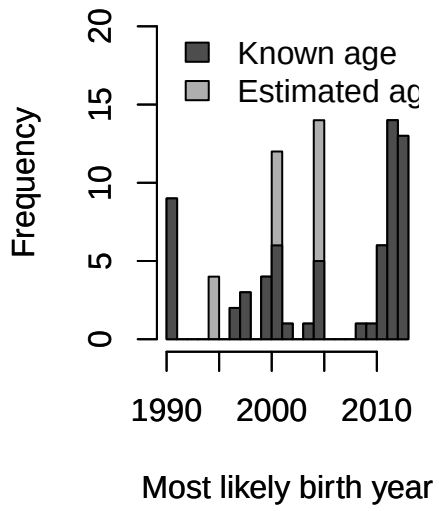



```
hist(as.numeric(as.character(known.age.sheep$MostLikelyCohort)), breaks = 20,
     xlab = "Most likely birth year", main = "Lostine sampled animals", col = "grey80")
```

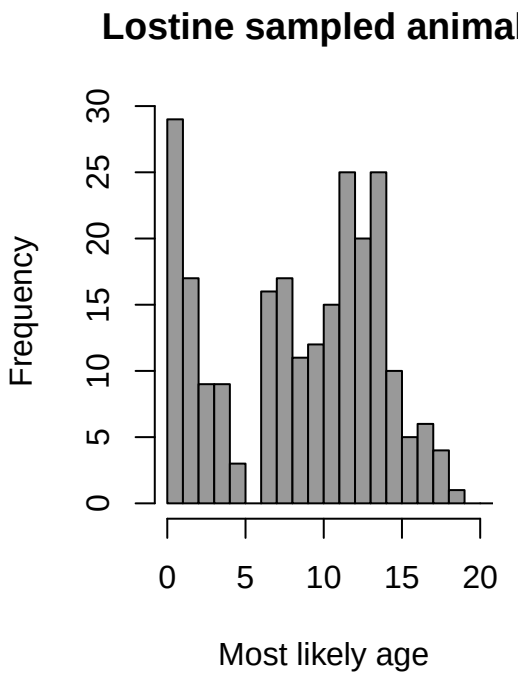



```
par(mfrow = c(1, 1), oma = c(2, 2, 0, 0))
hist(as.numeric(as.character(animal.data$MostLikelyCohort)),
     col = rgb(.1, .1, .1, alpha = .35), #col = rainbow.cols,
     breaks = hist.breaks, xlab = "Most likely birth year",
      ylim = c(0, 20), main = "")
par(new = T)
hist(as.numeric(as.character(known.age.sheep$MostLikelyCohort)),
     col = "grey30", #col = rainbow.cols,
     breaks = hist.breaks, xlab = "Most likely birth year",
     ylim = c(0, 20), main = "")
```

```
leg.text <- c("Known age", "Estimated age")
legend("topleft", leg.text, fill = c("grey30", rgb(.1, .1, .1, alpha = .35)), bty = "n")
```




```
hist(as.numeric(as.character(samples.full$MostLikelyAge)),
     col = "grey60", #col = rainbow.cols,
     breaks = 20, xlab = "Most likely age",
     main = "Lostine sampled animals", xlim = c(0, 20))
```




```
# append most likely age to samples
samples$animal.current.age <- rep(NA, dim(samples)[1])
for(i in 1:dim(samples)[1]){
  k <- subset(animal.data, as.character(Animal.ID) == as.character(samples$id)[i])
  samples$animal.current.age[i] <- as.numeric(as.character(samples$cap_bioyr[i])) - as.numeric(as.chara
}
```

```
# build a two year binned version of age.
samples$animal.current.age.2yrbins <- ifelse(samples$animal.current.age %in% c(0, 1), .5,
                                      ifelse(samples$animal.current.age %in% c(2, 3), 2.5,
                                      ifelse(samples$animal.current.age %in% c(4, 5), 4.5,
                                      ifelse(samples$animal.current.age %in% c(6, 7), 6.5,
                                      ifelse(samples$animal.current.age %in% c(8, 9), 8.5,
                                      ifelse(samples$animal.current.age %in% c(10, 11), 10.5,
                                      ifelse(samples$animal.current.age %in% c(12, 13), 12.5,
                                      ifelse(samples$animal.current.age %in% c(14, 15), 14.5,
                                      ifelse(samples$animal.current.age %in% c(16, 17), 16.5,
                                      ifelse(samples$animal.current.age %in% c(18, 19), 18.5,
                                      ifelse(samples$animal.current.age %in% c(20), 20, NA)))))))))))
```

Calculate time elapsed since beginning of study (December 13, 2012) for each sample.

```
#global.min.date <- min(strptime(as.character(samples$capture_date), format = "%m/%d/%Y"))
global.min.date <- min(strptime(as.character("12/13/2012"), format = "%m/%d/%Y"))
samples$days_into_study <- as.numeric(difftime(strptime(as.character(samples$capture_date), format = "%
                                      global.min.date,
                                      units = "days"
                                      ))
```

Build a subset of samples that omits all indeterminate test results.

```
samples.noind <- subset(samples, movi_qpcr != "Indeterminate")
samples.noind$movi.pcr.pos <- ifelse(samples.noind$movi_qpcr == "Detected", 1,
                                ifelse(samples.noind$movi_qpcr == "Not detected", 0, NA))
```

Pull out juveniles, build a dataset of animals only sampled as adults, and record how many animals were sampled as adults only. Build a different dataset of animals sampled at least once as a lamb or a yearling.

```
not.adults <- subset(samples.noind, age_class != "Adult")
adults <- subset(samples.noind, age_class == "Adult" & !(id %in% levels(factor(not.adults$id))) & cap_b
adultFs <- subset(samples.noind, age_class == "Adult" & !(id %in% levels(factor(not.adults$id))) & cap_
n.adults <- length(levels(factor(adults$id)))
adults$id <- factor(adults$id)
adultFs$id <- factor(adultFs$id)

juvs <- subset(samples.noind, !(id %in% levels(factor(adults$id))))
n.juvs <- length(levels(factor(juvs$id)))
```

Disease status assigments

We used two definitions of "carrier" to determine disease status for each animal. Under the first definition, animals were assigned to the carrier class if they tested positive at least twice in a single year. This assignment was made on an animal-year basis (so a single animal could be a carrier one year, and intermittent or negative in another). Under the second definition, an animal was classified as a carrier if it tested positive in two consecutive years (or in two consecutive sampling events, if events were not in consecutive years – this only applies to 12LO27).

Get animal's classification as neg/inter/carrier on two consec years definition

```
animal.class <- rep(NA, length(levels(factor(adults$id))))
for(i in 1:length(animal.class)){
  k <- subset(adults, id == levels(factor(adults$id))[i])
  animal.class[i] <- as.character(k$carrier_posConsecYears[1])
}

animal.data <- as.data.frame(cbind(levels(factor(adults$id)),
                                   as.character(animal.class)))
names(animal.data) <- c("id", "class")
```

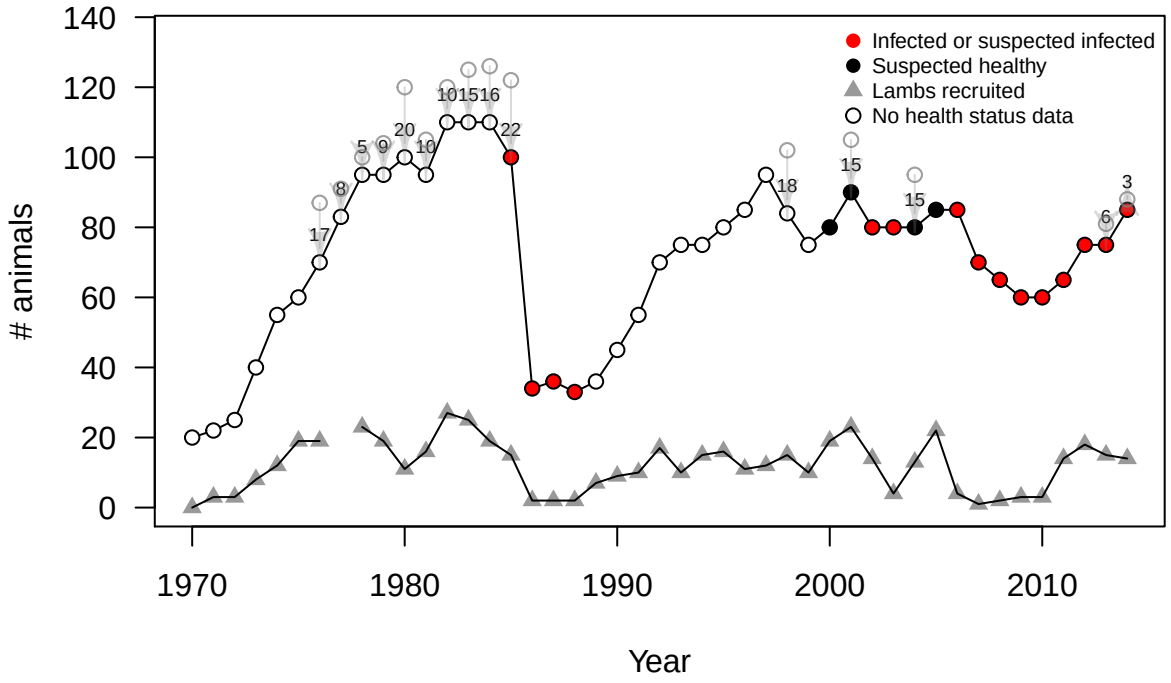
Pull out ids in each disease status group (carriers, inters, negs)

```
carriers <- subset(animal.data, class == "carrier")
inters <- subset(animal.data, class == "intermittent")
negs <- subset(animal.data, class == "negative")
NAs <- subset(animal.data, is.na(class) == T)
```

Figures

Figure 2: Lostine disease status timeseries

```
los <- subset(compd.data, Pop == "Lostine")
los$CLASS[43:44] <- "LAMBS"
pt.col <- ifelse(is.na(los$CLASS) == T, "white",
                 ifelse(los$CLASS == "HEALTHY", "black", "red"))

#svg("./Plots/LostineTimeSeries_20160628_V2.svg", width = 6, height = 5)
plot(los$PopEst ~ los$year, type = "l", ylab = "# animals",
     xlab = "Year", las = 1, ylim = c(0, 135))
points(los$PopEst ~ los$year, col = pt.col, pch = 16)
points(los$PopEst ~ los$year, col = "black", pch = 1)
shp.rem <- subset (los, NoSheepRem != 0 & is.na(NoSheepRem) != T)
shp.rel <- subset (los, NoSheepRel != 0 & is.na(NoSheepRel) != T)
text(x = shp.rem$year[-1], y = shp.rem$PopEst[-1] + 8, shp.rem$NoSheepRem[-dim(shp.rem)[1]],
     cex = .60)
points(shp.rem$PopEst[-1] + shp.rem$NoSheepRem[-dim(shp.rem)[1]] ~
         (shp.rem$year[-1]), col = rgb(.5, .5, .5, .5))
points(shp.rem$PopEst[-1] + shp.rem$NoSheepRem[-dim(shp.rem)[1]] ~
         (shp.rem$year[-1]), col = rgb(.5, .5, .5, .5))

#points(los$Lambs ~ los$year, pch = 5)
points(los$Lambs ~ los$year, pch = 17, col = "grey60")
lines(los$Lambs ~ los$year)
leg.text <- c("Infected or suspected infected", "Suspected healthy", "Lambs recruited", "No health statu
legend("topright", leg.text, col = c("red", "black", "grey60", "black"),
       pch = c(16, 16, 17, 1), pt.cex = c(1, 1, 1, 1), bty = "n", cex = .65)
for(i in 2:dim(shp.rem)[1]){
  Arrows(x0 = shp.rem$year[i], x1 = shp.rem$year[i],
           y0 = (shp.rem$PopEst[i] + shp.rem$NoSheepRem[i-1]),
           y1 = (shp.rem$PopEst[i])+4, cex = .5, code = 2,
         col = rgb(.5, .5, .5, .3))
}
```




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

Figure 3. Sampling intensities

Re-order animal IDs so that they're blocked by carriage status.

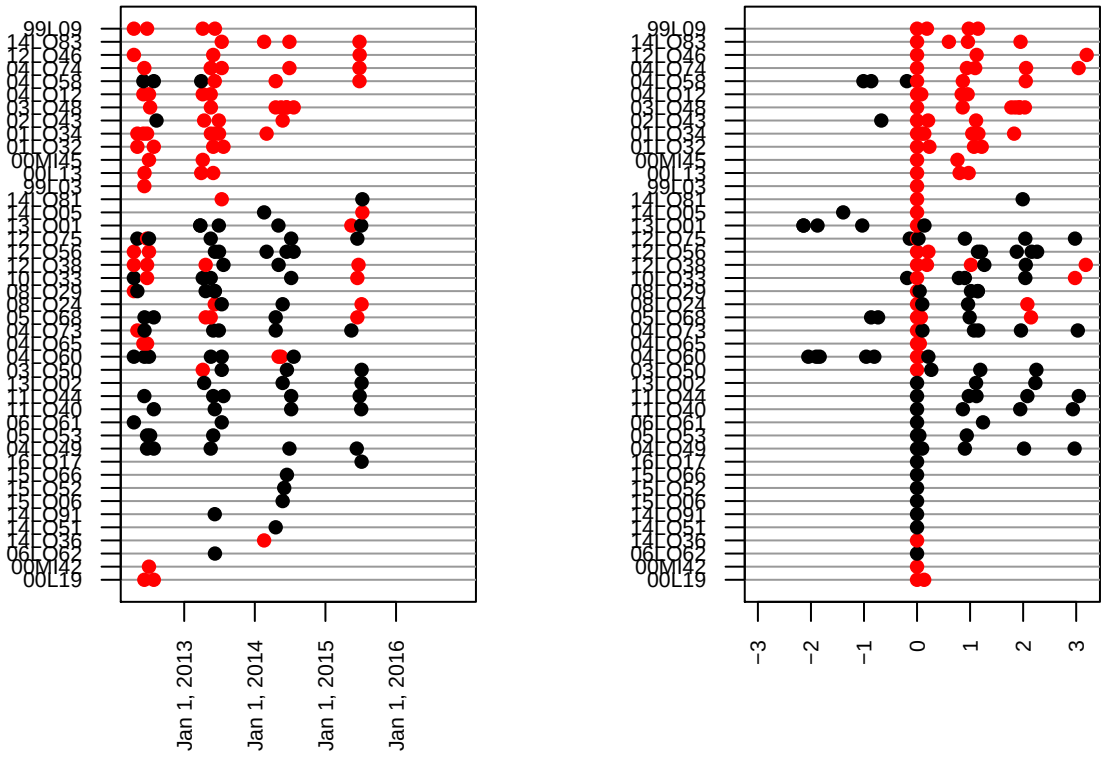
```
adults$id.2 <- factor(adults$id,
                      levels = c(levels(factor(NAs$id)),
                                 levels(factor(negs$id)),
                                 levels(factor(inters$id)),
                                 levels(factor(carriers$id))
                      ))
```

Get time deviations since first positive test for each animal.

```
adults$time.rel.to.1st.pos <- rep(NA, dim(adults)[1])
for(i in 1:dim(adults)[1]){
  k <- subset(adults, id == adults$id[i] & FirstPosInd == 1)
  if(dim(k)[1] == 1){
    adults$time.rel.to.1st.pos[i] <- difftime(time2 = strptime(k$capture_date[1], format = "%m/%d/%Y"),
                                              time1 = strptime(adults$capture_date[i], format = "%m/%d/
    } else {
    m <- subset(adults, id == adults$id[i] & FirstTestInd == 1)
    adults$time.rel.to.1st.pos[i] <- difftime(time2 = strptime(m$capture_date[1], format = "%m/%d/%Y"),
                                              time1 = strptime(adults$capture_date[i], format = "%m/%d/

  }
}
```

Build the sampling intensities for adults figure.

```r
#svg("./Plots/SamplingIntensityFig_20161006_V2.svg", height = 7, width = 6)
par(mfrow = c(1, 2), las = 2, mar = c(6, 6, 1, 1), cex.axis = .7)

# first panel: samples within individuals through time.
plot(x = 0, y = 0, xlim = c(0, 1700), ylim = c(1, n.adults), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.adults){
    abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(adults$id.2 ~ adults$days_into_study, pch = 16,
       col = ifelse(adults$movi_qpcr == "Detected", "red", "black"))
axis(side = 2, at = seq(1, n.adults), labels = levels(factor(adults$id.2)), cex = .4)
axis(side = 1, at = c(260, 260 + 365, 260 + 365*2, 260 + 365*3),
     labels = c("Jan 1, 2013", "Jan 1, 2014", "Jan 1, 2015", "Jan 1, 2016"))

# second panel: first pos = 0
plot(x = 0, y = 0, xlim = c(-365*3, 365*3.2), ylim = c(1, n.adults), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.adults){
  abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(adults$id.2 ~ adults$time.rel.to.1st.pos, pch = 16,
       col = ifelse(adults$movi_qpcr == "Detected", "red", "black"))
axis(side = 2, at = seq(1, n.adults), labels = levels(factor(adults$id.2)), cex = .4)
axis(side = 1, at = c(-365*3, -365*2, -365, 0, 365, 365*2, 365*3),
     labels = c(-3, -2, -1, 0, 1, 2, 3))
```

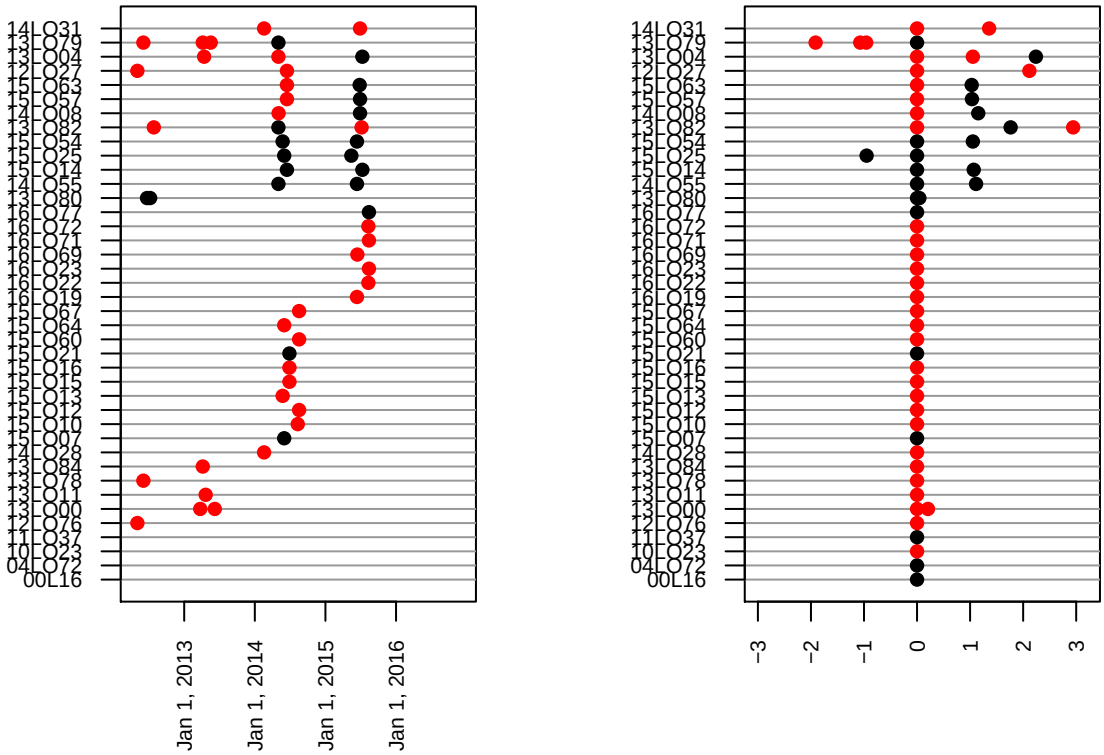


```
#dev.off()
```

Repeat whole process to build sampling intensities for juveniles

```
# get animal's classification as neg/inter/carrier on two consec years definition
juv.class <- rep(NA, length(levels(factor(juvs$id))))
for(i in 1:length(juv.class)){
  k <- subset(juvs, id == levels(factor(juvs$id))[i])
  juv.class[i] <- as.character(k$carrier_posConsecYears[1])
}

juv.data <- as.data.frame(cbind(levels(factor(juvs$id)),
                                as.character(juv.class)))
names(juv.data) <- c("id", "class")

# pull out ids in each group (carriers, inters, negs)
juv.carriers <- subset(juv.data, class == "carrier")
juv.inters <- subset(juv.data, class == "intermittent")
juv.negs <- subset(juv.data, class == "negative")
juv.NAs <- subset(juv.data, is.na(class) == T)

# re-order animal IDs so that they're blocked by carriage status.
juvs$id.2 <- factor(juvs$id,
                    levels = c(levels(factor(juv.NAs$id)),
                               levels(factor(juv.negs$id)),
                               levels(factor(juv.inters$id)),
                               levels(factor(juv.carriers$id))
                    ))

# get time deviations since first positive test for each animal.
juvs$time.rel.to.1st.pos <- rep(NA, dim(juvs)[1])
for(i in 1:dim(juvs)[1]){
  k <- subset(juvs, id == juvs$id[i] & FirstPosInd == 1)
  if(dim(k)[1] == 1){
    juvs$time.rel.to.1st.pos[i] <- difftime(time2 = strptime(k$capture_date[1], format = "%m/%d/%Y"),
                                            time1 = strptime(juvs$capture_date[i], format = "%m/%d/%Y
  } else {
    m <- subset(juvs, id == juvs$id[i] & FirstTestInd == 1)
    juvs$time.rel.to.1st.pos[i] <- difftime(time2 = strptime(m$capture_date[1], format = "%m/%d/%Y"),
                                            time1 = strptime(juvs$capture_date[i], format = "%m/%d/%Y

  }
}

# get time since birth for each animal.
juvs$age_days <- rep(NA, dim(juvs)[1])
for(i in 1:dim(juvs)[1]){
    juvs$age_days[i] <- difftime(time2 = strptime(juvs$date_birth[i], format = "%m/%d/%Y"),
                                 time1 = strptime(juvs$capture_date[i], format = "%m/%d/%Y"))
}

#svg("./Plots/SamplingIntensityFig_JUVS_20161006_V2.svg", height = 5.5, width = 6)
par(mfrow = c(1, 2), las = 2, mar = c(6, 6, 1, 1), cex.axis = .7)
```

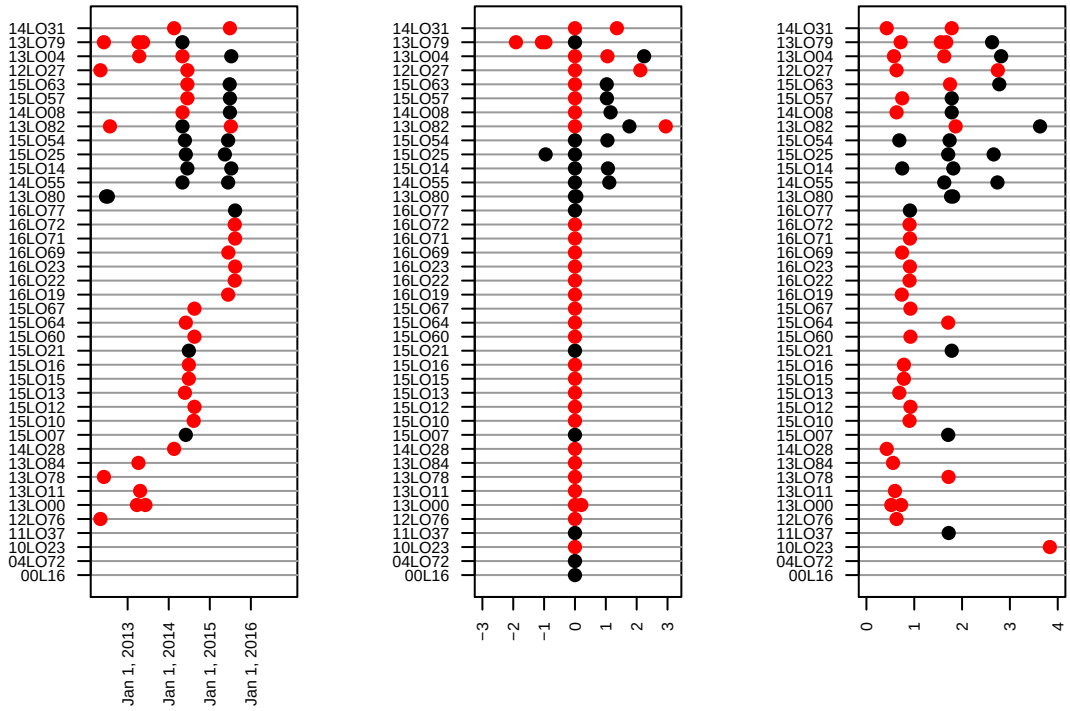
```r
# first panel: samples within individuals through time.
plot(x = 0, y = 0, xlim = c(0, 1700), ylim = c(1, n.juvs), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.juvs){
  abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(juvs$id.2 ~ juvs$days_into_study, pch = 16,
       col = ifelse(juvs$movi_qpcr == "Detected", "red", "black"))
axis(side = 2, at = seq(1, n.juvs), labels = levels(factor(juvs$id.2)), cex = .4)
axis(side = 1, at = c(260, 260 + 365, 260 + 365*2, 260 + 365*3),
     labels = c("Jan 1, 2013", "Jan 1, 2014", "Jan 1, 2015", "Jan 1, 2016"))

# second panel: first pos = 0
plot(x = 0, y = 0, xlim = c(-365*3, 365*3.2), ylim = c(1, n.juvs), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.juvs){
  abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(juvs$id.2 ~ juvs$time.rel.to.1st.pos, pch = 16,
       col = ifelse(juvs$movi_qpcr == "Detected", "red", "black"))
axis(side = 2, at = seq(1, n.juvs), labels = levels(factor(juvs$id.2)), cex = .4)
axis(side = 1, at = c(-365*3, -365*2, -365, 0, 365, 365*2, 365*3),
     labels = c(-3, -2, -1, 0, 1, 2, 3))
```




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

Build a three-panel version of juveniles sampling figure.

```r
#svg("./Plots/SamplingIntensityFig_3PanelJUVS_20161006_V1.svg", height = 6, width = 9)
par(mfrow = c(1, 3), las = 2, mar = c(6, 6, 1, 1), cex.axis = .7)

# first panel: samples within individuals through time.
plot(x = 0, y = 0, xlim = c(0, 1700), ylim = c(1, n.juvs), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.juvs){
  abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(juvs$id.2 ~ juvs$days_into_study, pch = 16,
       col = ifelse(juvs$movi_qpcr == "Detected", "red", "black"), cex = 1.5)
axis(side = 2, at = seq(1, n.juvs), labels = levels(factor(juvs$id.2)))
axis(side = 1, at = c(260, 260 + 365, 260 + 365*2, 260 + 365*3),
     labels = c("Jan 1, 2013", "Jan 1, 2014", "Jan 1, 2015", "Jan 1, 2016"))

# second panel: first pos = 0
plot(x = 0, y = 0, xlim = c(-365*3, 365*3.2), ylim = c(1, n.juvs), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.juvs){
  abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(juvs$id.2 ~ juvs$time.rel.to.1st.pos, pch = 16,
       col = ifelse(juvs$movi_qpcr == "Detected", "red", "black"), cex = 1.5)
axis(side = 2, at = seq(1, n.juvs), labels = levels(factor(juvs$id.2)))
axis(side = 1, at = c(-365*3, -365*2, -365, 0, 365, 365*2, 365*3),
     labels = c(-3, -2, -1, 0, 1, 2, 3))

# third panel: birth day = 0
plot(x = 0, y = 0, xlim = c(0, 365*4), ylim = c(1, n.juvs), cex = 0,
     xaxt = "n", yaxt = "n", xlab = "", ylab = "")
for(i in 1:n.juvs){
  abline(h = i, lty = 1, lwd = 1, col = "grey60")
}
points(juvs$id.2 ~ juvs$age_days, pch = 16,
       col = ifelse(juvs$movi_qpcr == "Detected", "red", "black"), cex = 1.5)
axis(side = 2, at = seq(1, n.juvs), labels = levels(factor(juvs$id.2)))
axis(side = 1, at = c(0, 365, 365*2, 365*3, 365*4),
     labels = c( 0, 1, 2, 3, 4))
```




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

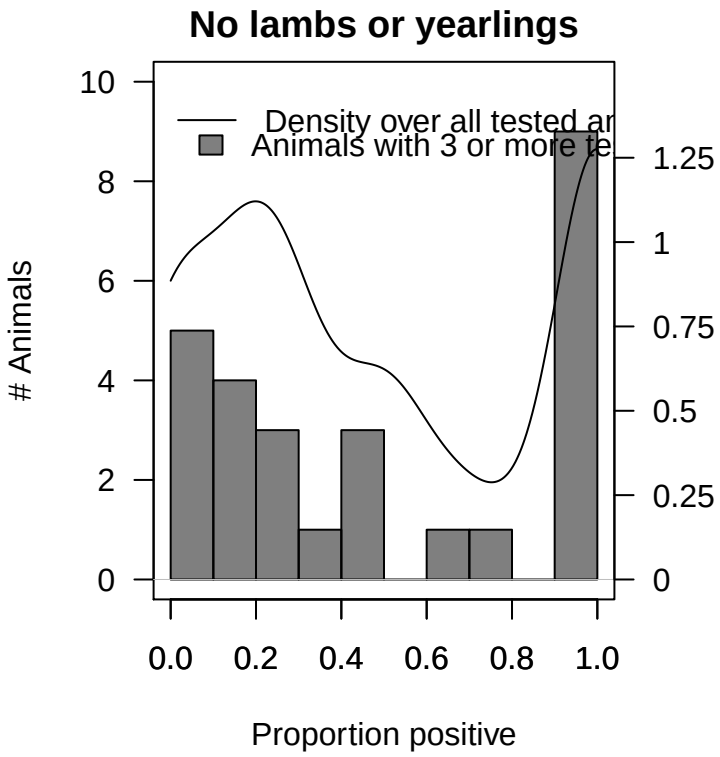
Figure 4a. Proportion positive tests within animal, with KDE

```
# remove all animals with only one observation
observed.once <- names(table(adults$id))[which(table(adults$id) == 1 )]
observed.twice <-  names(table(adults$id))[which(table(adults$id) == 2 )]

pcr.by.id <- table(adults$qPCRResult, adults$id)
prop.pos <- prop.table(pcr.by.id, margin = 2)[1, ]
d <- density((1-prop.pos),
             weights = as.vector(table(adults$id))/sum(table(adults$id)),
             from = 0, to = 1,
             bw = .1)

# bootstrapped CI for minimum of d
nboot <- 10000
resamp.min <- rep(NA, nboot)
for(i in 1:nboot){
  resamp.dat <- pcr.by.id[ ,sample(1:31, rep = T)]
  resamp.prop.pos <- prop.table(resamp.dat, margin = 2)[1, ]
  resamp.d <- density((1-resamp.prop.pos),
                      weights = as.vector(resamp.dat[1,] + resamp.dat[2, ])/sum(resamp.dat[1,] + resamp
                      from = 0, to = 1,
                      bw = .1)
  resamp.min[i] <- resamp.d$x[which.min(resamp.d$y)]
  # print(i)
}
```

```r
quantile(resamp.min, c(0.1, 0.5, 0.9))
```

```
##       10%       50%       90%
## 0.5009785 0.7416830 0.8023483
```

```r
# reduced to just animals with 3 or more samples
data <- subset(adults, !(id %in% observed.once | id %in% observed.twice))
data$id <- factor(data$id)

pcr.by.id.full <- table(adults$qPCRResult, adults$id)
prop.pos.full <- prop.table(pcr.by.id.full, margin = 2)[1, ]

pcr.by.id <- table(data$qPCRResult, data$id)
prop.pos <- prop.table(pcr.by.id, margin = 2)[1, ]
#rbind(pcr.by.id, 1-prop.pos)

#svg("./Plots/PropPositiveFrequencies_20160623.svg", height = 5, width = 6)
par(mfrow = c(1, 1), mar = c(4, 4, 2, 4))
hist(1 - prop.pos, breaks = 10, col = rgb(0, 0, 0, alpha = .5),
     xlim = c(0, 1), ylim = c(0, 10),
     ylab = "# Animals",
     xlab = "Proportion positive",
     las = 1,
     main = "No lambs or yearlings")
leg.text <- "Animals with 3 or more tests"
legend(y = 9.5, x = 0, leg.text, fill = "grey50", bty = "n")

par(new = T)
plot(d, xlim = c(0, 1), ylim = c(0, max(d$y)+.2), yaxt = "n", xlab = "",
     main = "", ylab = "")
mtext(side = 4, line = 3, "")
axis(side = 4, at = seq(0, max(d$y), by = .25), labels = seq(0, max(d$y), by = .25), las = 1)
leg.text <- "Density over all tested animals"
legend(y = max(d$y) + .2, x = -.05, leg.text, lty = 1, bty = "n")
```

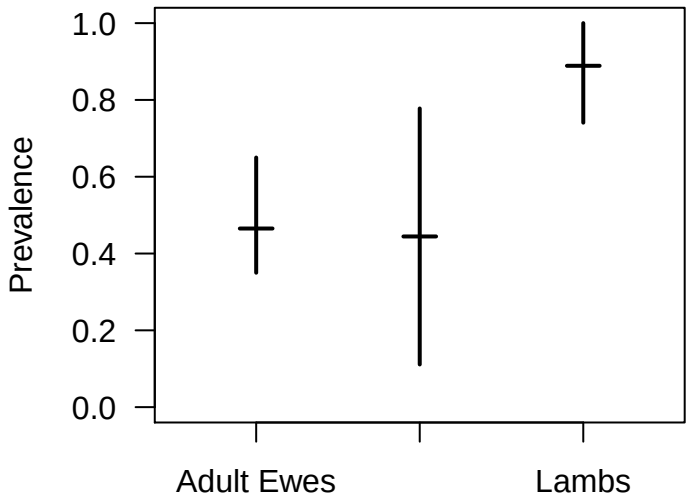



```
#dev.off()
```

Figure 4b. Prevalence by age

```
# Movi prevalence estimation
# samples <- read.csv("./Data/LostineSample2016Final_WithClasses_20160927.csv", header = T)
# samples <- subset(samples, movi_qpcr %in% c("Detected", "Not detected"))
# samples$qPCRResult <- ifelse(samples$movi_qpcr == "Detected" | samples$movi_qpcr == "Indeterminate",
# samples <- subset(samples, cap_bioyr >= 2008)

# prev based on first sampling event for EVERYBODY
# strip out each animal's first sampling event
first.event <- vector("list", length(levels(factor(samples.noind$id))))
for(i in 1:length(levels(factor(samples.noind$id)))){
  k <- subset(samples.noind, as.character(id) == as.character(levels(factor(samples.noind$id))[i]) & ca
  first.event[[i]] <- k[which.min(as.numeric(as.Date(k$capture_date, format = "%m/%d/%Y"))), ]
}

first.event.frame <- do.call("rbind", first.event)

# subset adults/yrs/lambs
first.event.ads <- subset(first.event.frame, age_class == "Adult" & sex == "F")
first.event.lamb <- subset(first.event.frame, age_class == "Lamb")
first.event.yrl <- subset(first.event.frame, age_class == "Yearling")
first.event.frame$juv.ind <- ifelse(first.event.frame$age_class == "Adult", "adult", "juv")

# bootstrapped prevalence across all years in adult females
```

```r
nboot <- 1000
boot.ewe.prev <- rep(NA, nboot)
boot.lamb.prev <- rep(NA, nboot)
boot.yrl.prev <- rep(NA, nboot)
for(i in 1:nboot){
  k <- first.event.ads[sample(1:dim(first.event.ads)[1], size = dim(first.event.ads)[1], replace = T), ]
  boot.ewe.prev[i] <- table(k$movi_qpcr, useNA = "always")[1]/dim(first.event.ads)[1]
}

for(i in 1:nboot){
  k <- first.event.lamb[sample(1:dim(first.event.lamb)[1], size = dim(first.event.lamb)[1], replace = T)
  boot.lamb.prev[i] <- table(k$movi_qpcr, useNA = "always")[1]/dim(first.event.lamb)[1]
}

for(i in 1:nboot){
  k <- first.event.yrl[sample(1:dim(first.event.yrl)[1], size = dim(first.event.yrl)[1], replace = T), ]
  boot.yrl.prev[i] <- table(k$movi_qpcr, useNA = "always")[1]/dim(first.event.yrl)[1]
}

adewe.prev.all.yrs <- quantile(boot.ewe.prev, c(0.025, 0.975))
lamb.prev.all.yrs <- quantile(boot.lamb.prev, c(0.025, 0.975))
yrl.prev.all.yrs <- quantile(boot.yrl.prev, c(0.025, 0.975))

# Comparison of WITHIN-YEAR prevalence
# need first sampling event of each animal in each year

prev.xtab <- table(first.event.frame$age_class, first.event.frame$movi_qpcr)
age.spec.prevs <- prev.xtab[, 1] / table(first.event.frame$age_class)

#svg("./Plots/AgeSpecPrev_20160711.svg", height = 4, width = 3)
plot(x = 0, y = 0, xlim = c(.5, 3.5), ylim = c(0, 1), cex = 0,
     xaxt = "n", ylab = "Prevalence", las = 1, xlab = "")
segments(x0 = 1, x1 = 1, y0 = adewe.prev.all.yrs[1], y1 = adewe.prev.all.yrs[2], lwd = 2)
segments(x0 = 2, x1 = 2, y0 = yrl.prev.all.yrs[1], y1 = yrl.prev.all.yrs[2], lwd = 2)
segments(x0 = 3, x1 = 3, y0 = lamb.prev.all.yrs[1], y1 = lamb.prev.all.yrs[2], lwd = 2)
segments(x0 = 0.9, x1 = 1.1, y0 = age.spec.prevs[1], y1 = age.spec.prevs[1], lwd = 2)
segments(x0 = 1.9, x1 = 2.1, y0 = age.spec.prevs[3], y1 = age.spec.prevs[3], lwd = 2)
segments(x0 = 2.9, x1 = 3.1, y0 = age.spec.prevs[2], y1 = age.spec.prevs[2], lwd = 2)
axis(side = 1, at = c(1, 2, 3), labels = c("Adult Ewes", "Yearlings", "Lambs"))
```

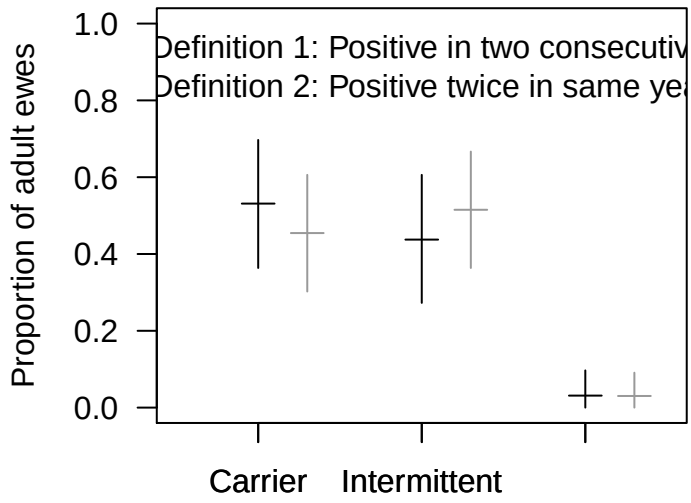



```
#dev.off()
```

Figure 4c. Estimated proportion in each disease state

```
# Movi prevalence estimation
#samples <- read.csv("./Data/LostineSample2016Final.csv", header = T)
#length(levels(factor(samples$id)))
#
#samples.small <- subset(samples, age_class == "Adult" & sex == "F")

# pull off one obs per animal
ind.carriage.class.1 <- ind.carriage.class.2 <- ind.age.class <- ind.id <- rep(NA, length(levels(adultF
for(i in 1:length(levels(adultFs$id))){
  k <- subset(samples.small, as.character(id) == as.character(levels(adultFs$id))[i])
  ind.carriage.class.1[i] <- as.character(k$carriage_twoPosOneYear)[1]
  ind.carriage.class.2[i] <- as.character(k$carrier_posConsecYears)[1]
  ind.id[i] <- as.character(levels(adultFs$id))[i]
}

carriage.status.props.1 <- table(na.omit(ind.carriage.class.1))/sum(table(na.omit(ind.carriage.class.1)
carriage.status.props.2 <- table(na.omit(ind.carriage.class.2))/sum(table(na.omit(ind.carriage.class.2)

nboot <- 1000
carriage.1.boot <- carriage.2.boot <- matrix(NA, nrow = nboot, ncol = 3)
ind.carriage.class.1.nonas <- na.omit(ind.carriage.class.1)
ind.carriage.class.2.nonas <- na.omit(ind.carriage.class.2)
for(i in 1:nboot){
  k <- sample(1:length(ind.carriage.class.2.nonas), size = length(ind.carriage.class.2.nonas), replace =
  carriage.1.boot[i, ] <- table(factor(ind.carriage.class.1.nonas[k], levels = c("carrier", "intermitte
  carriage.2.boot[i, ] <- table(factor(ind.carriage.class.2.nonas[k], levels = c("carrier", "intermitte
}

def1.carrier.cis <- quantile(carriage.1.boot[, 1], c(0.025, 0.975))
def1.intermittent.cis <- quantile(carriage.1.boot[, 2], c(0.025, 0.975))
def1.negative.cis <- quantile(carriage.1.boot[, 3], c(0.025, 0.975))
```

```
def2.carrier.cis <- quantile(carriage.2.boot[, 1], c(0.025, 0.975))
def2.intermittent.cis <- quantile(carriage.2.boot[, 2], c(0.025, 0.975))
def2.negative.cis <- quantile(carriage.2.boot[, 3], c(0.025, 0.975))

# svg("./Plots/PropEwesInEachCarriageState_20160712.svg", height = 4, width = 3)
plot(x = 0, y = 0, xlim = c(.5, 3.5), ylim = c(0, 1), xaxt = "n",
     xlab = "", ylab = "Proportion of adult ewes", las = 1)
segments(x0 = .9, x1 = 1.1, y0 = carriage.status.props.1[1], y1 = carriage.status.props.1[1])
segments(x0 = 1.9, x1 = 2.1, y0 = carriage.status.props.1[2], y1 = carriage.status.props.1[2])
segments(x0 = 2.9, x1 = 3.1, y0 = carriage.status.props.1[3], y1 = carriage.status.props.1[3])
segments(x0 = 1, x1 = 1, y0 = def1.carrier.cis[1], y1 = def1.carrier.cis[2])
segments(x0 = 2, x1 = 2, y0 = def1.intermittent.cis[1], y1 = def1.intermittent.cis[2])
segments(x0 = 3, x1 = 3, y0 = def1.negative.cis[1], y1 = def1.negative.cis[2])
axis(side = 1, at = c(1, 2, 3), labels = c("Carrier", "Intermittent", "Negative"))

segments(x0 = 1.2, x1 = 1.4, y0 = carriage.status.props.2[1], y1 = carriage.status.props.2[1], col = "g
segments(x0 = 2.2, x1 = 2.4, y0 = carriage.status.props.2[2], y1 = carriage.status.props.2[2], col = "g
segments(x0 = 3.2, x1 = 3.4, y0 = carriage.status.props.2[3], y1 = carriage.status.props.2[3], col = "g
segments(x0 = 1.3, x1 = 1.3, y0 = def2.carrier.cis[1], y1 = def2.carrier.cis[2], col = "grey60")
segments(x0 = 2.3, x1 = 2.3, y0 = def2.intermittent.cis[1], y1 = def2.intermittent.cis[2], col = "grey6
segments(x0 = 3.3, x1 = 3.3, y0 = def2.negative.cis[1], y1 = def2.negative.cis[2], col = "grey60")
axis(side = 1, at = c(1, 2, 3), labels = c("Carrier", "Intermittent", "Negative"))
leg.text <- c("Definition 1: Positive in two consecutive years", "Definition 2: Positive twice in same y
legend("top", leg.text, col = c("black", "grey60"), bty = "n", lty = c(1, 1))
```




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

Figure 5. Infection duration bootstrap plots.

First, run bootstrap.

```
### THIS WILL FAIL IF THE INPUT CSV IS NOT SORTED AS INCREASING IN CAPTURE DATE
# (EITHER WITHIN INDIVIDUAL OR ACROSS ALL INDIVIDUALS)
observed.once <- names(table(samples.noind$id))[which(table(samples.noind$id) == 1)]
```

```r
data <- subset(samples.noind, !(id %in% observed.once))
data$id <- factor(data$id)
# data$movi_qpcr <- ifelse(data$movi_qpcr == "Indeterminate", "Detected", as.character(data$movi_qpcr))
# get string lengths of all consecutive same outcomes
individ.list <- vector("list", length(levels(factor(data$id))))
string.length <- string.status <- vector("list", length(levels(data$id)))
for(i in 1:length(levels(data$id))){
  individ.list[[i]] <- subset(data, id == levels(data$id)[i])
  switches <- which(individ.list[[i]]$movi_qpcr[-dim(individ.list[[i]])[1]] != individ.list[[i]]$movi_q
  switches.full <- c(1, switches, dim(individ.list[[i]])[1])
  string.length[[i]] <- string.status[[i]] <- rep(NA, length(switches.full) - 1)
  if(length(switches.full) == 1){
     string.length[[i]][1] <- difftime(time1 = strptime(as.numeric(as.character(individ.list[[i]]$captu
                                       time2 = strptime(as.numeric(as.character(individ.list[[i]]$captu
     )
    string.status[[i]][1] <- individ.list[[i]]$movi_qpcr[1]

  } else {
    for(j in 1:(length(switches.full) - 1)){
          string.length[[i]][j] <- difftime(time1 = strptime(individ.list[[i]]$capture_date[switches.ful
                                            time2 = strptime(individ.list[[i]]$capture_date[switches.ful
          )
          string.status[[i]][j] <- as.character(individ.list[[i]][switches.full[j], ]$movi_qpcr)
    }
  }
 # print(i)
}

full.string.length.orig <- do.call("c", string.length)
full.string.status.orig <- do.call("c", string.status)
orig.mean.length <- mean(full.string.length.orig)
orig.mean.length.detected <- mean(full.string.length.orig[which(full.string.status.orig == "Detected")]
orig.mean.length.undetected <- mean(full.string.length.orig[which(full.string.status.orig == "Not detec

# get string lengths of consecutive same outcomes for animals classified as CARRIERS
carrier1 <- subset(data, carrier_posConsecYears == "carrier")
carrier1$id <- factor(carrier1$id)
individ.list.carrier1 <- vector("list", length(levels(factor(carrier1$id))))
string.length.carrier1 <- string.status.carrier1 <- vector("list", length(levels(carrier1$id)))
for(i in 1:length(levels(factor(carrier1$id)))){
  individ.list.carrier1[[i]] <- subset(carrier1, id == levels(carrier1$id)[i])
  switches.carrier1 <- which(individ.list.carrier1[[i]]$movi_qpcr[-dim(individ.list.carrier1[[i]])[1]]
  switches.full.carrier1 <- c(1, switches.carrier1, dim(individ.list.carrier1[[i]])[1])
  string.length.carrier1[[i]] <- string.status.carrier1[[i]] <- rep(NA, length(switches.full.carrier1) -
  if(length(switches.full.carrier1) == 1){
    string.length.carrier1[[i]][1] <- difftime(time1 = strptime(individ.list.carrier1[[i]]$capture_date
                                      time2 = strptime(individ.list.carrier1[[i]]$capture_date[1], forma
    )
    string.status.carrier1[[i]][1] <- individ.list.carrier1[[i]]$movi_qpcr[1]

  } else {
    for(j in 1:(length(switches.full.carrier1) - 1)){
      string.length.carrier1[[i]][j] <- difftime(time1 = strptime(individ.list.carrier1[[i]]$capture_da
```

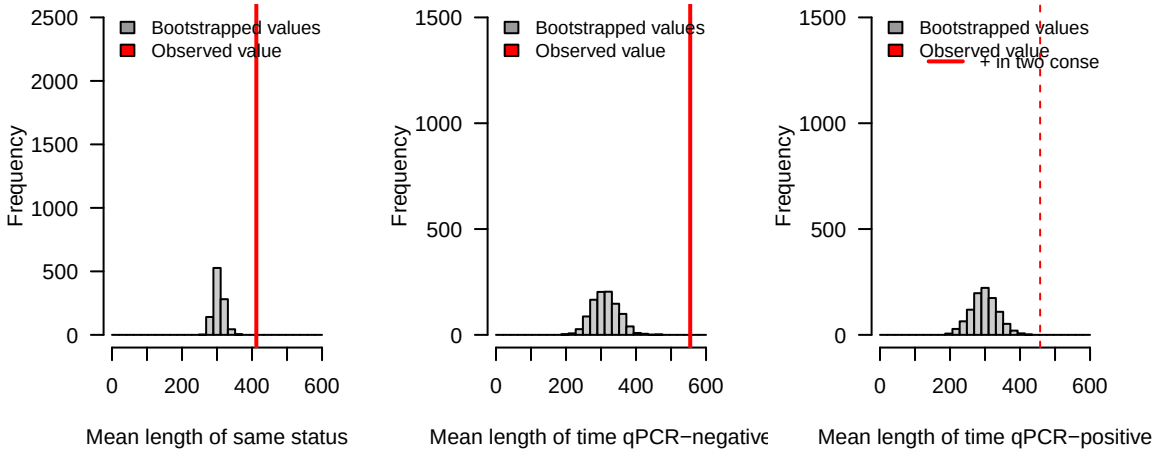
```
                                        time2 = strptime(individ.list.carrier1[[i]]$capture_date[switche
      )
      string.status.carrier1[[i]][j] <- as.character(individ.list.carrier1[[i]][switches.full.carrier1[
    }
  }
 # print(i)
}

full.string.length.orig.carrier1 <- do.call("c", string.length.carrier1)
full.string.status.orig.carrier1 <- do.call("c", string.status.carrier1)
orig.mean.length.carrier1 <- mean(full.string.length.orig.carrier1)
orig.mean.length.detected.carrier1 <- mean(full.string.length.orig.carrier1[which(full.string.status.ori
orig.mean.length.undetected.carrier1 <- mean(full.string.length.orig.carrier1[which(full.string.status.

# get string lengths of consecutive same outcomes for animals classified as CARRIERS (pos two consec ye
carrier2 <- subset(data, carrier_posConsecYears == "carrier")
carrier2$id <- factor(carrier2$id)
individ.list.carrier2 <- vector("list", length(levels(factor(carrier2$id))))
string.length.carrier2 <- string.status.carrier2 <- vector("list", length(levels(carrier2$id)))
for(i in 1:length(levels(factor(carrier2$id)))){
  individ.list.carrier2[[i]] <- subset(carrier2, id == levels(carrier2$id)[i])
  switches.carrier2 <- which(individ.list.carrier2[[i]]$movi_qpcr[-dim(individ.list.carrier2[[i]])[1]]
  switches.full.carrier2 <- c(1, switches.carrier2, dim(individ.list.carrier2[[i]])[1])
  string.length.carrier2[[i]] <- string.status.carrier2[[i]] <- rep(NA, length(switches.full.carrier2) -
  if(length(switches.full.carrier2) == 1){
    string.length.carrier2[[i]][1] <- difftime(time1 = strptime(individ.list.carrier2[[i]]$capture_date
                                               time2 = strptime(individ.list.carrier2[[i]]$capture_date
    )
    string.status.carrier2[[i]][1] <- individ.list.carrier2[[i]]$movi_qpcr[1]

  } else {
    for(j in 1:(length(switches.full.carrier2) - 1)){
      string.length.carrier2[[i]][j] <- difftime(time1 = strptime(individ.list.carrier2[[i]]$capture_dat
                                                 time2 = strptime(individ.list.carrier2[[i]]$capture_dat
      )
      string.status.carrier2[[i]][j] <- as.character(individ.list.carrier2[[i]][switches.full.carrier2[
    }
  }
#  print(i)
}

full.string.length.orig.carrier2 <- do.call("c", string.length.carrier2)
full.string.status.orig.carrier2 <- do.call("c", string.status.carrier2)
orig.mean.length.carrier2 <- mean(full.string.length.orig.carrier2)
orig.mean.length.detected.carrier2 <- mean(full.string.length.orig.carrier2[which(full.string.status.ori
orig.mean.length.undetected.carrier2 <- mean(full.string.length.orig.carrier2[which(full.string.status.

# generate bootstrapped distribution
data <- subset(data, is.na(movi_qpcr) == F & !(movi_qpcr == ""))
data.boot.fun <- function(nboot, data){
  # build storage objects
  full.string.length <- full.string.status <- data.list <- vector("list", nboot)
  boot.mean.length <- boot.mean.length.detected <- boot.mean.length.undetected <- rep(NA, nboot)
```

```
  individ.list <- vector("list", length(levels(factor(data$id))))
  string.length <- string.status <- vector("list", length(levels(data$id)))

  for(n in 1:nboot){
    data.list[[n]] <- data
    data.list[[n]]$movi_qpcr <- sample(data$movi_qpcr, size = dim(data)[1], replace = F)
  for(i in 1:length(levels(data.list[[n]]$id))){
    individ.list[[i]] <- subset(data.list[[n]], id == levels(data.list[[n]]$id)[i])
    switches <- which(individ.list[[i]]$movi_qpcr[-dim(individ.list[[i]])[1]] != individ.list[[i]]$movi
    switches.full <- c(1, switches, dim(individ.list[[i]])[1])
    string.length[[i]] <- string.status[[i]] <- rep(NA, length(switches.full) - 1)
    if(length(switches.full) == 1){
      string.length[[i]][1] <- difftime(time1 = strptime(individ.list[[i]]$capture_date[dim(individ.lis
                                         time2 = strptime(individ.list[[i]]$capture_date[1], format = "%
      )
      string.status[[i]][1] <- individ.list[[i]]$movi_qpcr[1]

    } else {
      for(j in 1:(length(switches.full) - 1)){
        string.length[[i]][j] <- difftime(time1 = strptime(individ.list[[i]]$capture_date[switches.full
                                           time2 = strptime(individ.list[[i]]$capture_date[switches.full
        )
        string.status[[i]][j] <- as.character(individ.list[[i]][switches.full[j], ]$movi_qpcr)
      }
    }
  }
  full.string.length[[n]] <- do.call("c", string.length)
  full.string.status[[n]] <- do.call("c", string.status)

  boot.mean.length[n] <- mean(full.string.length[[n]])
  boot.mean.length.detected[n] <- mean(full.string.length[[n]][which(full.string.status[[n]] == "Detect
  boot.mean.length.undetected[n] <- mean(full.string.length[[n]][which(full.string.status[[n]] == "Not

#  print(n)
  }
  out.list <- list(full.string.length = full.string.length,
                   full.string.status = full.string.status,
                   boot.mean.length = boot.mean.length,
                   boot.mean.length.detected = boot.mean.length.detected,
                   boot.mean.length.undetected = boot.mean.length.undetected)
  return(out.list)
}

data.boot.test <- data.boot.fun(nboot = 1000, data = data)
table(data.boot.test$boot.mean.length.detected >= orig.mean.length.detected)
```

```
##
## FALSE  TRUE
##   282   718
```

Then, plot results

```
#svg("./Plots/StatusPermutationTests_20160930.svg", height = 3, width = 7)
par(mfrow = c(1, 3), las = 1)
hist(data.boot.test$boot.mean.length,
     breaks = seq(0, 600, length.out = 30), col = "grey80",
     main = "", xlab = "Mean length of same status", ylim = c(0, 2500))
abline(v = orig.mean.length, lwd = 2, col = "red")
leg.text <- c("Bootstrapped values", "Observed value")
legend("topleft", leg.text, fill = c("grey60", "red"), bty = "n", cex = .9)

hist(data.boot.test$boot.mean.length.undetected, ylim = c(0, 1500),
     breaks = seq(0, 600, length.out = 30), col = "grey80",
     main = "", xlab = "Mean length of time qPCR-negative")
abline(v = orig.mean.length.undetected, lwd = 2, col = "red")
leg.text <- c("Bootstrapped values", "Observed value")
legend("topleft", leg.text, fill = c("grey60", "red"), bty = "n", cex = .9)

hist(data.boot.test$boot.mean.length.detected,
     breaks = seq(0, 600, length.out = 30), col = "grey80", ylim = c(0, 1500),
     main = "", xlab = "Mean length of time qPCR-positive")
leg.text <- c("Bootstrapped values", "Observed value")
legend("topleft", leg.text, fill = c("grey60", "red"), bty = "n", cex = .9)
#abline(v = orig.mean.length.detected, lwd = 2, col = "red")
abline(v = orig.mean.length.detected.carrier1, lwd = 1, col = "red", lty = 2)
#abline(v = orig.mean.length.detected.carrier2, lwd = 2, col = "red", lty = 3)
leg.text.2 <- c("+ in two consec years")
legend(x = 90, y = 1400, leg.text.2, lty = c(1, 2, 3), lwd = rep(2, 3), col = rep("red", 3), bty = "n",
```




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

Figure 6. Test status by age.

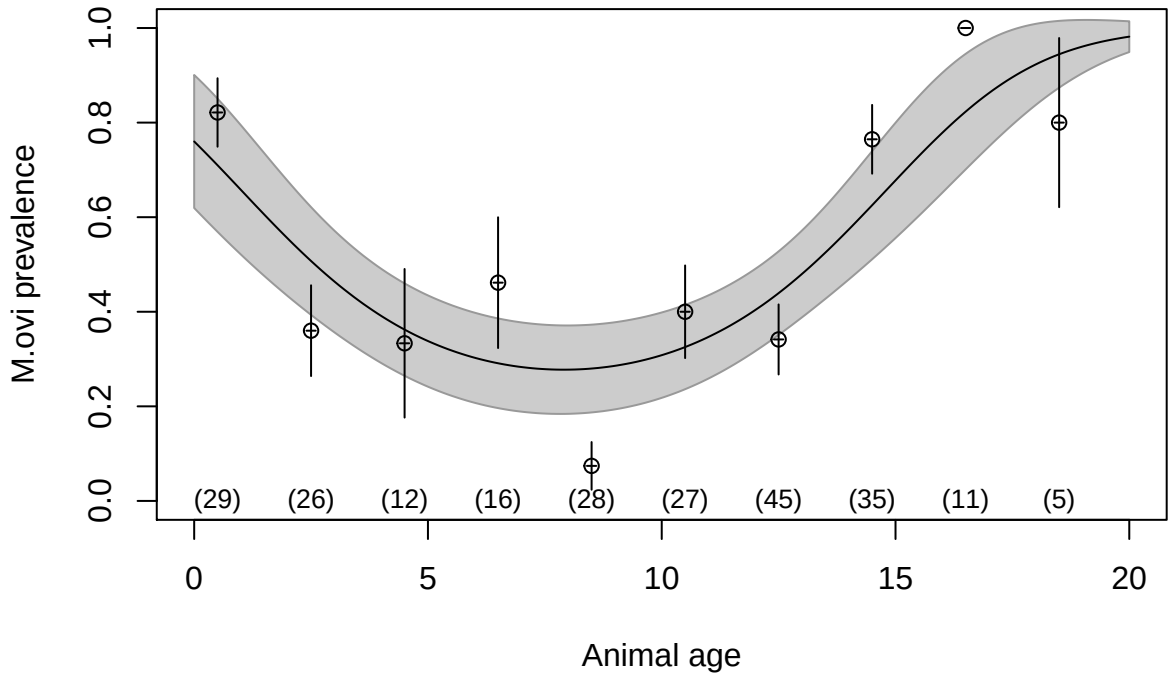
Quadratic fit for age-prevalence.

```
logist.cat <- glm(movi.pcr.pos ~ factor(animal.current.age),
                  family = "binomial", data = samples.noind)
logist.cat.2yr <- glm(movi.pcr.pos ~ factor(animal.current.age.2yrbins),
                 family = "binomial", data = samples.noind)
```

```r
newdata <- data.frame(animal.current.age.2yrbins = as.numeric(as.character(levels(factor(samples.noind$a
logist.cat.preds <- predict(logist.cat.2yr, newdata, type = "response", se = T)
logist.cat.preds <- predict(logist.cat.2yr, newdata, type = "response", se = T)
logist.quad <- glm(movi.pcr.pos ~ animal.current.age + I(animal.current.age^2),
                   family = "binomial", data = samples.noind)
newdata.quad <- data.frame(animal.current.age = seq(0, 20, length.out = 200))
logist.quad.pred <- predict(logist.quad, newdata.quad, type = "response", se = T)
prev.by.age <- table(samples.noind$movi.pcr.pos, samples.noind$animal.current.age.2yrbins)[2, ]/table(sa
ages <- as.numeric(as.character(names(table(samples.noind$animal.current.age.2yrbins))))
```

Plot of age-by-prevalence (with error bars not adjusted for uncertainty in age… David's is more appropriate.)

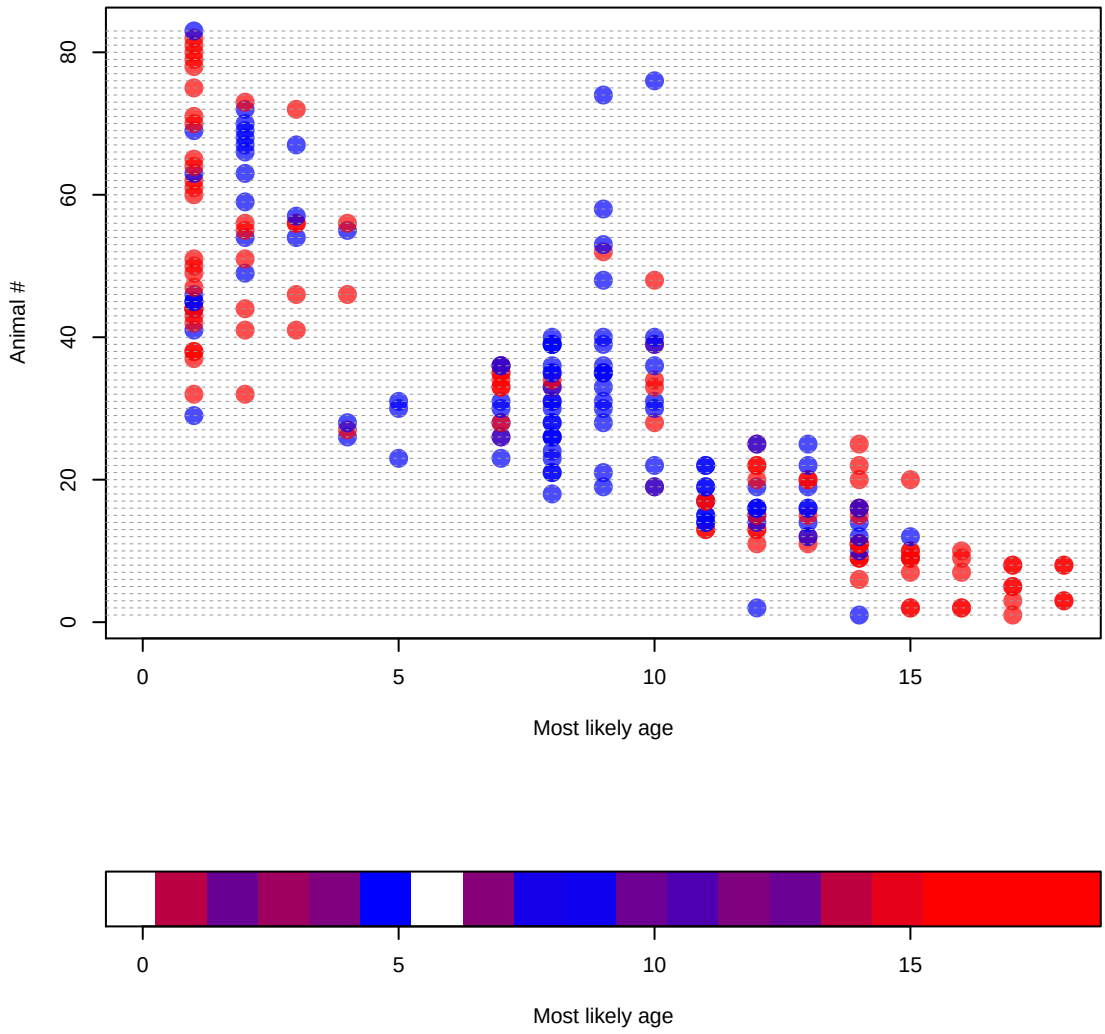
```r
# svg("./Plots/AgePrevalenceCurve_20160505.svg", height = 5, width = 6)
par(mfrow = c(1,1))
plot(as.numeric(as.character(prev.by.age)) ~ ages, xlim = c(0, 20), ylim = c(0, 1),
     ylab = "M.ovi prevalence", xlab = "Animal age")
for(i in 1:dim(newdata)[1]){
  segments(x0 = newdata$animal.current.age[i], x1 = newdata$animal.current.age[i],
        y0 = logist.cat.preds$fit[i] - logist.cat.preds$se.fit[i],
        y1 = logist.cat.preds$fit[i] + logist.cat.preds$se.fit[i])
  segments(x0 = newdata$animal.current.age[i] - .1, x1 = newdata$animal.current.age[i] + .1,
           y0 = logist.cat.preds$fit[i], y1 = logist.cat.preds$fit[i])
  text(x = newdata$animal.current.age[i], y = 0.0, cex = .8,
       paste("(", table(samples$animal.current.age.2yrbins)[i], ")", sep = ""))
}
lines(logist.quad.pred$fit ~ newdata.quad$animal.current.age, type = "l")
polygon(x = c(newdata.quad$animal.current.age, rev(newdata.quad$animal.current.age)),
        y = c(logist.quad.pred$fit - 1.96 * logist.quad.pred$se.fit,
              rev(logist.quad.pred$fit + 1.96 * logist.quad.pred$se.fit)),
        col = rgb(0, 0, 0, alpha = .2), border = rgb(0, 0, 0, alpha = .4))
```



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

And here's the color-block version (as per Frances's Sheep and Goat Council talk)

```
# age-spec prevalence
full.data <- read.csv("./Data/LostineSample2016Final_WithClasses_20160927.csv", header = T)
full.data <- subset(full.data, movi_qpcr %in% c("Detected", "Not detected"))
full.data$qPCRResult <- ifelse(full.data$movi_qpcr == "Detected" | full.data$movi_qpcr == "Indeterminat
full.data <- subset(full.data, cap_bioyr >= 2008)

full.data$DaysSinceFirstCap <- full.data$NumberTests <- full.data$DaysSinceFirstOverallCap <- full.data
for(i in 1:dim(full.data)[1]){
  k <- subset(full.data, id == full.data$id[i])
  full.data$DaysSinceFirstCap[i] <- difftime(as.POSIXlt(strptime(full.data$capture_date[i], format = "%
                                             min(as.POSIXlt(strptime(k$capture_date, format = "%m/%d/%Y
  full.data$DaysSinceFirstOverallCap[i] <- difftime(as.POSIXlt(strptime(full.data$capture_date[i], forma
                                             min(as.POSIXlt(strptime(full.data$capture_date, format = "
  full.data$NumberTests[i] <- dim(k)[1]
  full.data$AgeRank[i] <- min(k$age_capture)
  full.data$leq5.ind[i] <- ifelse(full.data$age_capture[i] <= 5, 1, 0)
}

full.data$pt.col <- ifelse(full.data$qPCRResult == "P", rgb(1, 0, 0, alpha = .7),
                 ifelse(full.data$qPCRResult == "N", rgb(0, 0, 1, alpha = .7),
                        rgb(.6, .6, .6, alpha = .5)))

full.data$MostLikelyAge <- ifelse(full.data$MostLikelyAge == 0, 1, full.data$MostLikelyAge)
full.data$id <- ifelse(full.data$id == "99L03", "000L03",
                       ifelse(full.data$id == "99L09", "000L09",
                              as.character(full.data$id)))
prev.tab <- prop.table(table(full.data$MostLikelyAge, full.data$pt.col), margin = 1)
prev.col <- rgb(prev.tab[,2], 0, prev.tab[,1], alpha = 1)
prev.col.in <- c(prev.col[1:5], "white", prev.col[6:18])

#svg("./Plots/AgeByPrev_IndividAndMarginal_20160929.svg", height = 7, width = 5)
layout(matrix(c(1, 1, 1, 2), ncol = 1, byrow = T))
#par(mfrow = c(2, 1))
plot((as.numeric(factor(full.data$id))) ~
       as.numeric(as.character(full.data$MostLikelyAge)),
     xlab = "Most likely age", ylab = "Animal #",
     cex = 2, #cex = (reduced.data$Movi.ELISA/100 * 5),
     col = full.data$pt.col,
     pch = 16, xlim = c(0, 18), ylim = c(1, 83))
for(i in 1:length(levels(factor(full.data$id)))){
  abline(h = i, lty = 2, lwd = .5, col = rgb(.6, .6, .6))
}

plot(rep(1, 18) ~ seq(1, 18, by = 1), col = prev.col.in[1:18], pch = 15, cex = 8, yaxt = "n",
     xlab = "Most likely age", ylab = "", xlim = c(0, 18))
```




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

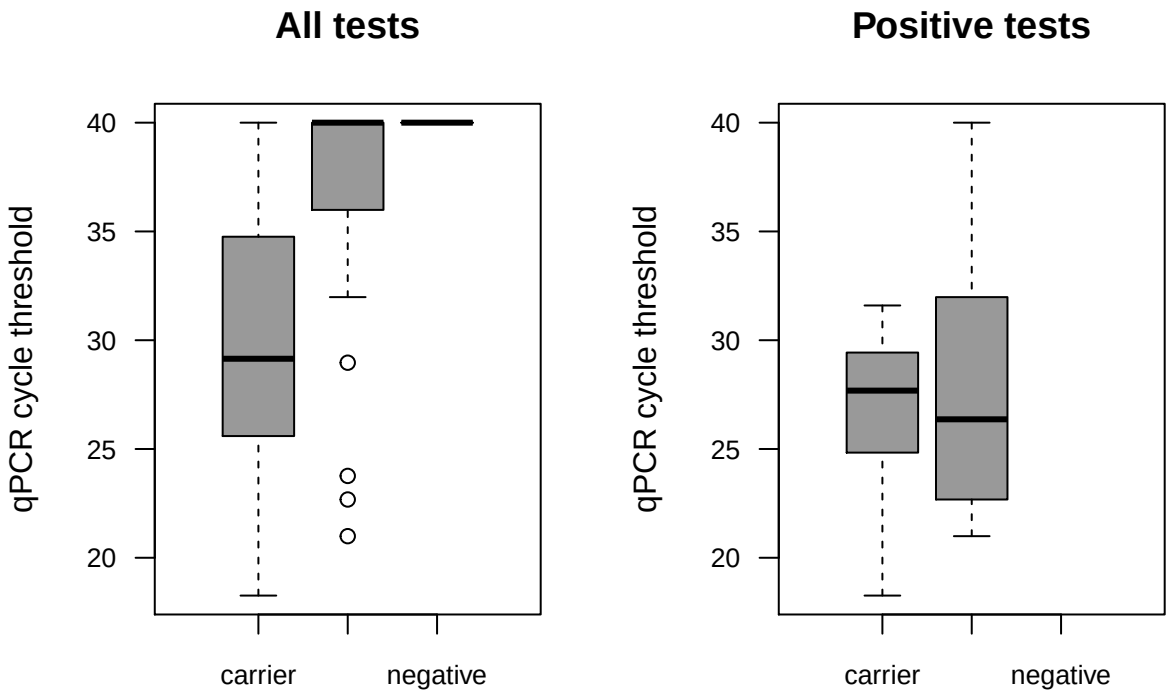
Figure 7.

Other Calculations

Power analysis for load difference between intermittents and carriers.

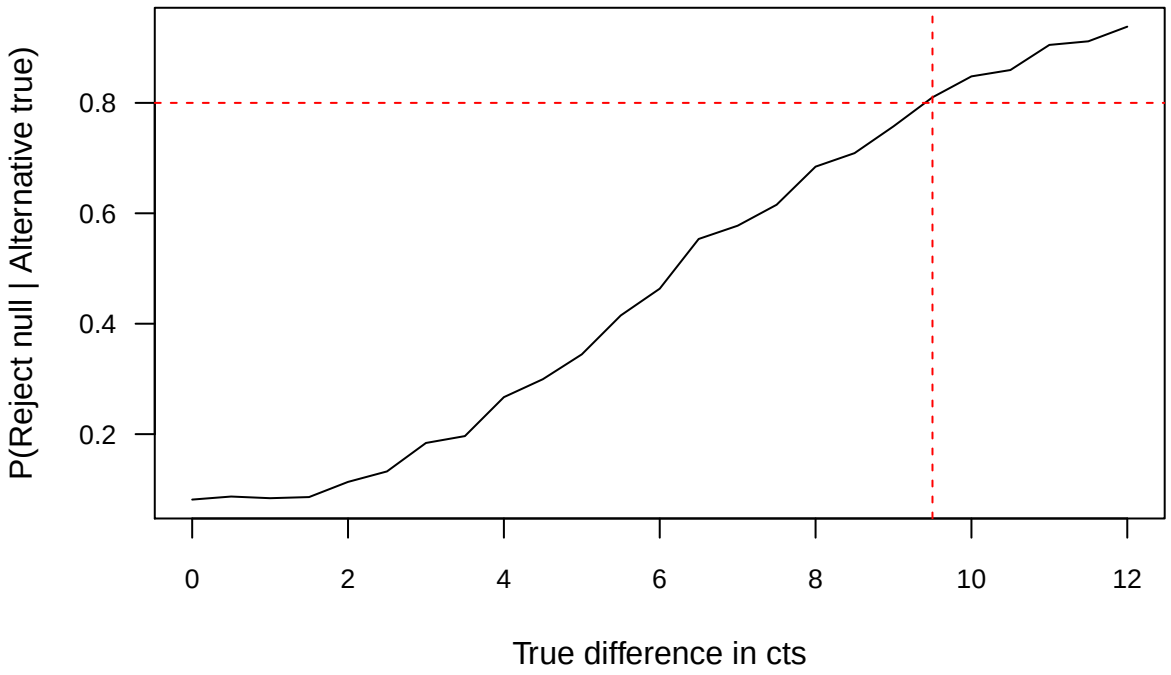
```
samplesWithqPCR <- read.csv("./Data/LostineSample2016Final.csv", header = T)

qpcrs <- subset(samplesWithqPCR, is.na(movi_direct_pcr) == F)
qpcrs_testpos <- subset(samplesWithqPCR, is.na(movi_direct_pcr) == F & movi_qpcr %in% c("Detected"))
qpcrs_testpos$carriage_twoPosOneYear <- factor(qpcrs_testpos$carriage_twoPosOneYear,
                                               levels = c("carrier", "intermittent", "negative"))


par(mfrow = c(1, 2), las = 1, cex.axis = .8)
boxplot(qpcrs$movi_direct_pcr ~ as.numeric(as.factor(qpcrs$carriage_twoPosOneYear)), col = "grey60",
```

```
        xlim = c(0, 4), xaxt = "n", ylab = "qPCR cycle threshold",
        main = "All tests")
axis(side = 1, at = c(1, 2, 3),
     labels = c("carrier", "intermittent", "negative"), cex = .6)
boxplot(qpcrs_testpos$movi_direct_pcr ~ as.numeric(qpcrs_testpos$carriage_twoPosOneYear), col = "grey60
        xlim = c(0, 4), xaxt = "n", ylab = "qPCR cycle threshold",
        main = "Positive tests")
axis(side = 1, at = c(1, 2, 3),
     labels = c("carrier", "intermittent", "negative"), cex = .6)
```




```
# test to compare qPCRs among test-pos carriers and test-pos intermittents
wilcox.testpos.out <- wilcox.test(qpcrs_testpos$movi_direct_pcr ~ qpcrs_testpos$carriage_twoPosOneYear)

# power analysis to examine differences
# effect might be two doublings shift...
alpha <- 0.05
# beta <- 0.80
# sample.size <-
# variance <-
# effect <-
# effect.size <- effect / variance

# tabulate values on raw data
carrier.testpos <- subset(qpcrs_testpos, carriage_twoPosOneYear == "carrier")
inter.testpos <- subset(qpcrs_testpos, carriage_twoPosOneYear == "intermittent")
n.carrier <- dim(carrier.testpos)[1]
n.inter <- dim(inter.testpos)[1]
sd.carrier.ct <- sd(carrier.testpos$movi_direct_pcr)
sd.inter.ct <- sd(inter.testpos$movi_direct_pcr)
variance.in <- (sqrt(sd.carrier.ct/n.carrier + sd.inter.ct/n.inter))^2
```

```r
# generate new data
baseline.ct <- 1
effects <- seq(0, 12, length.out = 25)
n.reps <- 2000
reject <- matrix(NA, nrow = length(effects), ncol = n.reps)
simmed.carriers <- simmed.inters <- wilcox.out <- vector("list", length(effects))
for(i in 1:length(effects)){
  simmed.carriers[[i]] <- simmed.inters[[i]] <- vector("list", n.reps)
  for(j in 1:n.reps){
    simmed.carriers[[i]][[j]] <- rnorm(mean = baseline.ct, sd = sd.carrier.ct, n = n.carrier)
    simmed.inters[[i]][[j]] <- rnorm(mean = baseline.ct - effects[i], sd = sd.inter.ct, n = n.inter)
    simmed.cts <- c(simmed.carriers[[i]][[j]], simmed.inters[[i]][[j]])
    simmed.states <- c(rep("carrier", n.carrier), rep("inter", n.inter))
    wilcox.out[[i]][[j]] <- wilcox.test(simmed.cts ~ simmed.states)
    reject[i, j] <- ifelse(wilcox.out[[i]][[j]]$p.value <= alpha, 1, 0)
  }
#  print(i)
}

# table rejection rates for each effect size
rejection.prop <- matrix(NA, ncol = 2, nrow = length(effects))
for(i in 1:length(effects)){
  rejection.prop[i, ] <- table(reject[i, ])/dim(reject)[2]
}

# "1"s are correct rejections
par(mfrow = c(1, 1))
plot(rejection.prop[,2] ~ effects,
     ylab = "P(Reject null | Alternative true)",
     xlab = "True difference in cts", type = "l")
abline(h = .8, lty = 2, col = "red")
abline(v = 9.5, lty = 2, col = "red")
```



Hardy-Weinberg Figures

First, data set-up.

```
# remove all animals with only one observation
observed.once <- names(table(samples.noind$id))[which(table(samples.noind$id) == 1)]

data <- subset(samples.noind, !(id %in% observed.once) & age_class == "Adult")
data$id <- factor(data$id)
# data$qPCRResult <- ifelse(data$movi_qpcr == "Detected" | data$movi_qpcr == "Indeterminate", "P", "N")

individ.bioyr <- unique(subset(data, select = c(id, cap_bioyr)))

# build list of all tests for each animal within a biological year
individ.bioyr.list <- vector("list", dim(individ.bioyr)[1])
number.tests.per.individ.bioyr <- rep(NA, dim(individ.bioyr)[1])

for(i in 1:length(individ.bioyr.list)){
  individ.bioyr.list[[i]] <- subset(data, id == individ.bioyr[i, 1] & cap_bioyr == individ.bioyr[i, 2])
  number.tests.per.individ.bioyr[i] <- dim(individ.bioyr.list[[i]])[1]
}

individ.bioyr.list.reduced <- individ.bioyr.list[number.tests.per.individ.bioyr != 1]
number.tests <- tests.pos <- tests.neg <- genotype.vsn <- rep(NA, length(individ.bioyr.list.reduced))
for(i in 1:length(individ.bioyr.list.reduced)){
  number.tests[i] <- dim(individ.bioyr.list.reduced[[i]])[1]
  tests.pos[i] <- dim(subset(individ.bioyr.list.reduced[[i]], movi_qpcr == "Detected"))[1]
  tests.neg[i] <- dim(subset(individ.bioyr.list.reduced[[i]], movi_qpcr == "Not detected"))[1]
  genotype.vsn[i] <- paste(individ.bioyr.list.reduced[[i]]$qPCRResult[1], "/", individ.bioyr.list.reduc
}
```

```r
# first test results
firsttests <- subset(samples.noind, FirstTestInd == 1 & age_class == "Adult")
table(firsttests$movi_qpcr)
```

```
##
##      Detected Indeterminate  Not detected
##            18             0            33
```

```r
prop.pos <- 18 / (18+30)
prop.neg <- 30/(18+30)

# Expected number with 2 pos if system at Hardy-Weinberg keq
# expected pos:
n.pos.pos <- (prop.pos ^ 2) * 41
n.neg.neg <- (prop.neg ^ 2) * 41
n.switch <- 2 * prop.pos * prop.neg * 41
expected <- c(n.neg.neg, n.switch, n.pos.pos)
observed <- table(tests.pos[number.tests == 2])
observed <- c(19, 9, 13)

obs.0.ci <- binom.test(x = observed[1], n = sum(observed))$conf.int * sum(observed)
obs.1.ci <- binom.test(observed[2], sum(observed))$conf.int * sum(observed)
obs.2.ci <- binom.test(observed[3], sum(observed))$conf.int * sum(observed)

exp.0.ci <- binom.test(x = ceiling(expected[1]), n = sum(observed))$conf.int * sum(observed)
exp.1.ci <- binom.test(ceiling(expected[2]), sum(observed))$conf.int * sum(observed)
exp.2.ci <- binom.test(ceiling(expected[3]), sum(observed))$conf.int * sum(observed)

hw.test <- hwexact(obs.hom1 = 19, obs.hets = 9, obs.hom2 = 13)

#---------------------------#
#-- OLD VSN ----------------#
#---------------------------#
# sum all test results to get overall distribution
prop.pos <- sum(tests.pos) / sum(number.tests)
prop.neg <- sum(tests.neg) / sum(number.tests)
sum(table(tests.pos[number.tests == 2]))
```

```
## [1] 32
```

```r
table(tests.pos[number.tests == 3])
```

```
##
## 0 1 2 3
## 4 1 1 2
```

```r
# Expected number with 2 pos if system at Hardy-Weinberg keq
# expected pos:
n.pos.pos <- (prop.pos ^ 2) * 36
n.neg.neg <- (prop.neg ^ 2) * 36
n.switch <- 2 * prop.pos * prop.neg * 36
```

```
expected <- c(n.neg.neg, n.switch, n.pos.pos)
observed <- table(tests.pos[number.tests == 2])

obs.0.ci <- binom.test(x = observed[1], n = sum(observed))$conf.int * sum(observed)
obs.1.ci <- binom.test(observed[2], sum(observed))$conf.int * sum(observed)
obs.2.ci <- binom.test(observed[3], sum(observed))$conf.int * sum(observed)

exp.0.ci <- binom.test(x = ceiling(expected[1]), n = sum(observed))$conf.int * sum(observed)
exp.1.ci <- binom.test(ceiling(expected[2]), sum(observed))$conf.int * sum(observed)
exp.2.ci <- binom.test(ceiling(expected[3]), sum(observed))$conf.int * sum(observed)
```

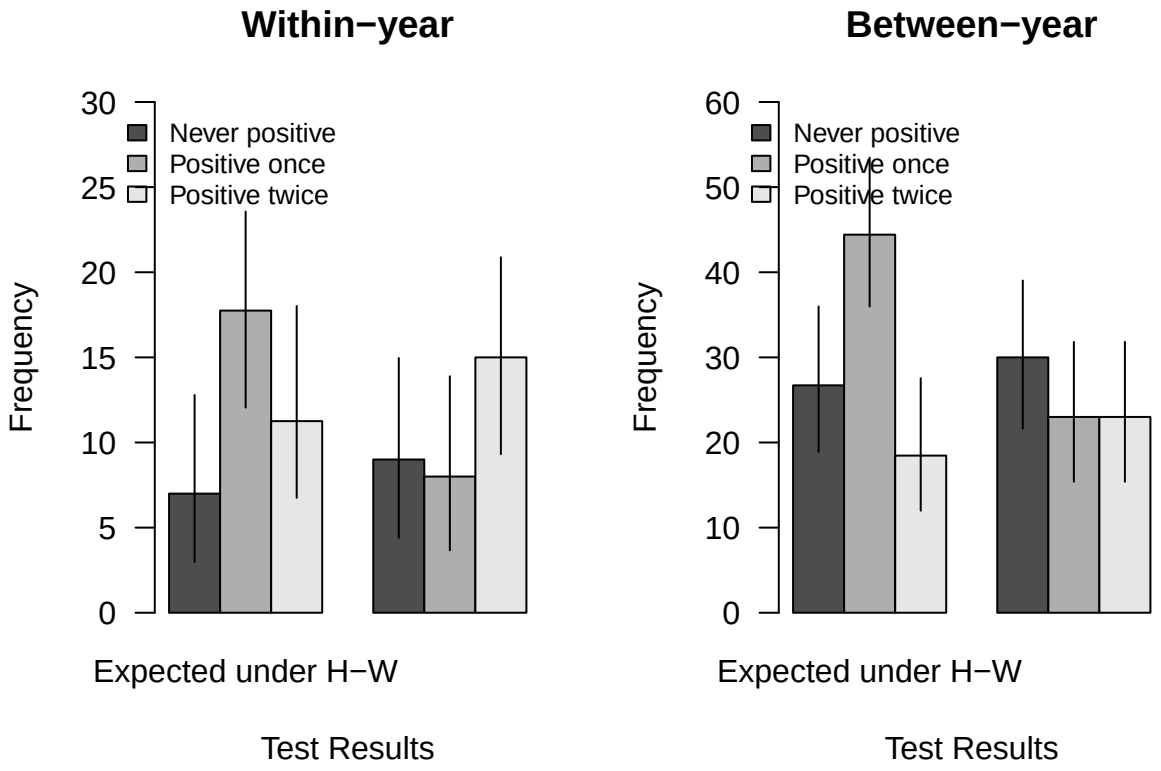
X-year Hardy-Weinberg

```
prop.pos.xyr <- (23 * 2 + 12 + 11) / (2*(30 + 11 + 8 + 21))
prop.neg.xyr <- (30 * 2 + 12 + 11) / (2*(30 + 11 + 8 + 21))

n.pos.pos.xyr <- (prop.pos.xyr ^ 2) * 76
n.neg.neg.xyr <- (prop.neg.xyr ^ 2) * 76
n.switch.xyr <- 2 * prop.pos.xyr * prop.neg.xyr * 76
expected.xyr <- c(n.neg.neg.xyr, n.switch.xyr, n.pos.pos.xyr)
observed.xyr <- c(30, 23, 23)

obs.0.ci.xyr <- binom.test(x = observed.xyr[1], n = sum(observed.xyr))$conf.int * sum(observed.xyr)
obs.1.ci.xyr <- binom.test(observed.xyr[2], sum(observed.xyr))$conf.int * sum(observed.xyr)
obs.2.ci.xyr <- binom.test(observed.xyr[3], sum(observed.xyr))$conf.int * sum(observed.xyr)

exp.0.ci.xyr <- binom.test(x = ceiling(expected.xyr[1]), n = sum(observed.xyr))$conf.int * sum(observed
exp.1.ci.xyr <- binom.test(ceiling(expected.xyr[2]), sum(observed.xyr))$conf.int * sum(observed.xyr)
exp.2.ci.xyr <- binom.test(ceiling(expected.xyr[3]), sum(observed.xyr))$conf.int * sum(observed.xyr)

hw.tab.xyr <- rbind(expected.xyr, observed.xyr)
```

Build figure

```
# svg("./Plots/HWPlots_20160623.svg", height = 5, width = 6)
par(mfrow = c(1, 2))
hw.tab <- rbind(expected, observed)
barplot(t(hw.tab), beside = T, ylim = c(0, 30), las = 1,
        ylab = "Frequency", xlab = "Test Results",
        names.arg = c("Expected under H-W", "Observed"),
        legend.text = c("Never positive", "Positive once", "Positive twice"),
        args.legend = list(y = 30, x = 4.5, bty = "n", cex = .8),
        main = "Within-year")
segments(x0 = 1.5, x1 = 1.5, y0 = exp.0.ci[1], y1 = exp.0.ci[2])
segments(x0 = 2.5, x1 = 2.5, y0 = exp.1.ci[1], y1 = exp.1.ci[2])
segments(x0 = 3.5, x1 = 3.5, y0 = exp.2.ci[1], y1 = exp.2.ci[2])

segments(x0 = 5.5, x1 = 5.5, y0 = obs.0.ci[1], y1 = obs.0.ci[2])
segments(x0 = 6.5, x1 = 6.5, y0 = obs.1.ci[1], y1 = obs.1.ci[2])
segments(x0 = 7.5, x1 = 7.5, y0 = obs.2.ci[1], y1 = obs.2.ci[2])

barplot(t(hw.tab.xyr), beside = T, ylim = c(0, 60), las = 1,
        ylab = "Frequency", xlab = "Test Results",
```

```
        names.arg = c("Expected under H-W", "Observed"),
        legend.text = c("Never positive", "Positive once", "Positive twice"),
        args.legend = list(y = 60, x = 4.5, bty = "n", ncol = 1, cex = .8),
        main = "Between-year")
segments(x0 = 1.5, x1 = 1.5, y0 = exp.0.ci.xyr[1], y1 = exp.0.ci.xyr[2])
segments(x0 = 2.5, x1 = 2.5, y0 = exp.1.ci.xyr[1], y1 = exp.1.ci.xyr[2])
segments(x0 = 3.5, x1 = 3.5, y0 = exp.2.ci.xyr[1], y1 = exp.2.ci.xyr[2])

segments(x0 = 5.5, x1 = 5.5, y0 = obs.0.ci.xyr[1], y1 = obs.0.ci.xyr[2])
segments(x0 = 6.5, x1 = 6.5, y0 = obs.1.ci.xyr[1], y1 = obs.1.ci.xyr[2])
segments(x0 = 7.5, x1 = 7.5, y0 = obs.2.ci.xyr[1], y1 = obs.2.ci.xyr[2])
```




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