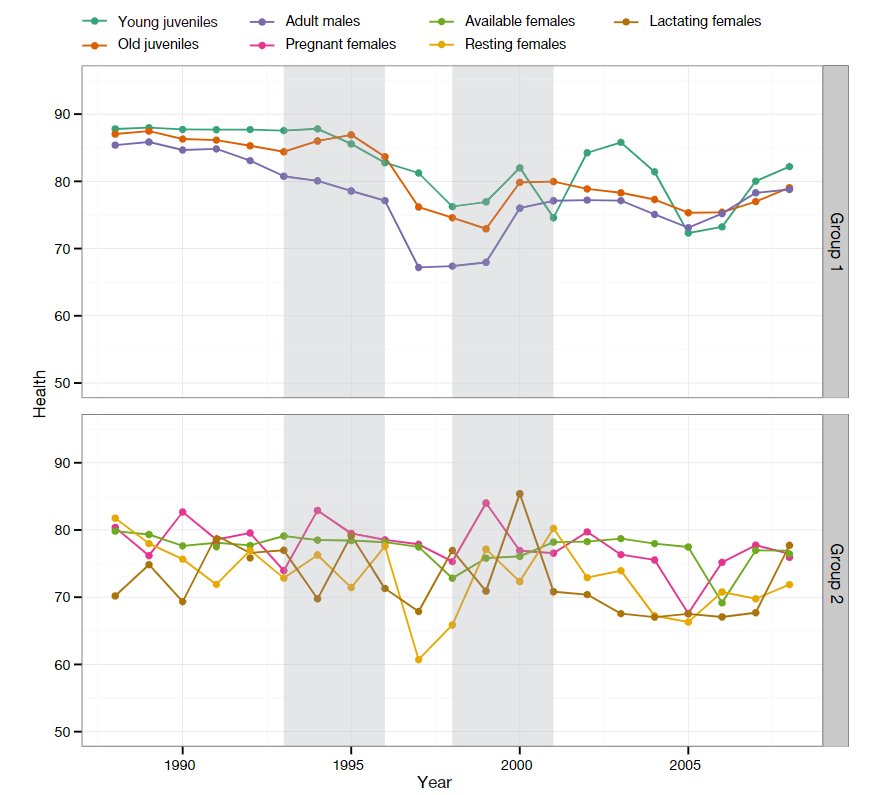
Progression of Entanglement Status

Rob Schick, PhD

2019-01-07

## Introduction

The goal of this writeup is to document the steps we’ve taken to make a new plot to be included in the Knowlton et al. paper. The goal of this analysis and figure is to uncover and highlight a) the progression of animals over time from a clean state to an entangled state, and b) to document if/how animals in a particular state affect the overall population health. Recall the figure from Rolland et al. (2016) that showed the median health of animals in different classes over time:



We had assembled those figures to find out what each group contributes to our overall understanding of health. For example, which group’s health was worst in the late 1990’s?

In the Knowlton et al. manuscript, we’ve shown how survival declines with entanglement. Here our goal is to try and sort out the contribution of entangled animals to the overall health. We will assess this contribution by examining how health of animals varies by entanglement class – minor, moderate, and severe – with the change from the analyses in the manuscript being that here we assume once an animal is entangled it enters a new state in which the health *may* differ from the native unentangled state.

## Methods

The approximate workflow is as follows. For each animal in the population, for each month in the analysis window, assign an entanglement state that corresponds to the injury class. The idea is that the animal starts ‘clear’ or unentangled. Upon becoming entangled, its status gets updated to the overall entanglement status. The key difference from the manuscript is here we assume that as time progresses, the animal only stays in its worst class, and doesn’t recover. We’ll have 5 ordinal classes:

* Class 4 - Unentangled
* Class 3 - Minor Entanglement
* Class 2 - Moderate Entanglement
* Class 1 - Severe Entanglement
* Class 0 - Known Dead (though we include this conceptually, it’s not been populated yet)

For example, every animal starts in class 4. Once it becomes entangled with a minor entanglement its state changes to a 3. While we have been careful to define the temporal extent of the entanglement windows in the manuscript, here we are explicitly assuming the animal is permanently in state 3, i.e. that it doesn’t recover back to an unentangled state. However, should the animal experience a moderate entanglement at some future point, its state will drop to 2 and remain there, even if its health recovers. This idea is one possible way to look at the cumulative impact of entanglement burden.

### Data Processing

As far as inputs to this analysis, we have:

1. the posterior estimates of health for all animals
2. the start date of each entanglement event for each animal
3. the matrix of state vectors just described

With these, we can assemble the data required for plotting.

Here are the entanglement injury data:

tangled <- tangleOut %>%   
 select(EGNo, EventNo, StartDateWindow, EndDateWindow, Severity, gear, gearInj, firstCalfidx)  
tangled

## # tibble [910 x 8]  
## EGNo EventNo StartDateWindow EndDateWindow Severity gear  
## \* <int> <int> <dttm> <dttm> <chr> <dbl>  
## 1 1004 1 1987-03-20 00:00:00 1987-06-20 00:00:00 moderate 0  
## 2 3445 1 2004-05-07 00:00:00 2004-08-07 00:00:00 minor 0  
## 3 3351 1 2003-05-12 00:00:00 2003-08-12 00:00:00 minor 0  
## 4 3350 1 2004-05-29 00:00:00 2004-08-29 00:00:00 minor 0  
## 5 1039 1 1997-05-16 00:00:00 1997-08-16 00:00:00 minor 0  
## 6 1032 1 1987-06-11 00:00:00 1987-09-11 00:00:00 minor 0  
## 7 1032 2 1999-04-19 00:00:00 1999-07-19 00:00:00 minor 0  
## 8 1032 3 2004-06-02 00:00:00 2004-09-02 00:00:00 minor 0  
## 9 1019 1 1987-07-02 00:00:00 1987-10-02 00:00:00 minor 0  
## 10 1019 2 1994-12-03 00:00:00 1995-03-03 00:00:00 moderate 0  
## # ... with 900 more rows, and 2 more variables: gearInj <dbl>,  
## # firstCalfidx <int>

In terms of the number of distinct animals in these data, we have:

n\_distinct(tangled$EGNo)

## [1] 464

**Update:** The reason this value is smaller than expected is that so far in the Knowlton et al. manuscript, we only used entanglement events across a window of time. However, there are some number of events that lack a start date.

In order to include those data, i.e. the ones without a valid start date, we need to do a bit of data processing to calculate the extra variables needed. This is because in previous analysis and data preparation steps, we’d discarded these events prior to calculating the window, the timing of the first calf, etc. To include these events without start dates, we need to format the end date properly, and get the temporal index of the first calf.

We start by grabbing these events, and assigning the gear injury status:

idx <- which(is.na(tangleAll$StartDate))   
tangleAll <- tangleAll[idx, ]  
tangleAll <- tangleAll %>%   
 select(EGNo, EventNo, StartDateWindow = StartDate, EndDateWindow = EndDate, Severity, gear) %>%   
 mutate(gearInj = case\_when(Severity == 'minor' & gear == 0 ~ 6,  
 Severity == 'minor' & gear == 1 ~ 4,  
 Severity == 'moderate' & gear == 0 ~ 5,  
 Severity == 'moderate' & gear == 1 ~ 2,  
 Severity == 'severe' & gear == 0 ~ 3,  
 Severity == 'severe' & gear == 1 ~ 1))

Next we add in the first calf index for the reproductively active females.

load("E:/rob/Documents/research/projects/rightWhaleEntanglement/src/data/calfTable.rdata")  
startYr <- 1970  
stopYr <- 2016  
yrvec <- startYr:stopYr  
nyr <- length(yrvec)  
monYr <- cbind(rep(c(1:12), nyr), rep(yrvec, each = 12))  
  
tangleAll$firstCalf <- as.Date('2500-01-01', '%Y-%m-%d')  
tangleAll$firstCalfidx <- NA  
  
for (egno in unique(tangleAll$EGNo)) {  
   
 if (length(calfTable[calfTable$EGNo == egno, 'CalvingYear']) > 0) {  
   
 year1 <- calfTable[calfTable$EGNo == egno, 'CalvingYear'][which.min(calfTable[calfTable$EGNo == egno, 'CalvingYear'])]  
 subdate <- as.Date(paste(year1, '-01-01', sep = ''), '%Y-%m-%d')  
 tangleAll$firstCalf[tangleAll$EGNo == egno] <- subdate  
 tangleAll$firstCalfidx[tangleAll$EGNo == egno] <- match(year1, monYr[, 2])   
   
 }  
   
}

Now we add these data to the events that have a defined start date, and I’m pulling out 1045, since she’s a special case:

tangledNew <- bind\_rows(tangled, tangleAll)  
  
tangled <- tangledNew %>%   
 select(-firstCalf) %>%   
 dplyr::filter(EGNo != 1045)

With the raw data, we can now build up a data structure that defines the fates of the animals. We populate this with 4’s under the assumption that all animals start unentangled.

dateVec <- seq.Date(as.Date("1970/01/01"), as.Date("2016/12/01"), by = 'month')  
fate <- matrix(4, nrow = length(ID), ncol = length(dateVec))

Now let’s loop over individuals, and within individuals, entanglement events, to fill in this state matrix.

**UPDATE** This changes slightly from the last document in that, I’m now starting the change at EndDateWindow instead of StartDateWindow. I don’t anticipate a major change. I’ve also corrected how I update the indexing on the fate matrix.

**UPDATE January 07, 2019** We’ll add the imputed and known deaths into this matrix now. This will be consistent with the survival curve analysis in the paper, and it will also make the health calculations a bit more correct - in the sense that after the month of most probable death, we’ll assume all health values are NA.

for (i in 1:length(unique(tangled$EGNo))){  
 eg <- unique(tangled$EGNo)[i]  
 anidx <- which(ID == eg)  
 dsub <- filter(tangled, EGNo == eg) %>%   
 select(EventNo, Severity, gearInj, StartDateWindow, EndDateWindow, firstCalfidx) %>%   
 arrange(StartDateWindow)  
   
 for(j in 1:nrow(dsub)){  
 startPos <- which.min(abs(dateVec - as.Date(dsub$EndDateWindow[j])))  
 valStart <- dsub$gearInj[j]  
 if(valStart %in% c(6, 4)) val <- 3  
 if(valStart %in% c(5, 2)) val <- 2  
 if(valStart %in% c(3, 1)) val <- 1  
 fate[anidx, startPos:ncol(fate)] <- val  
 }  
  
 # Update Health to exclude health of repro females   
 # from first calf (t1) to  
 # remaining time in unentangled state (t2), i.e. fate has to be a 4  
 if(!any(is.na(dsub$firstCalfidx))){  
   
 t1 <- min(dsub$firstCalfidx)  
 t2 <- max(which(fate[anidx, ] == 4))  
   
 if(t1 < t2){  
 timeAfter <- t1:t2  
 healthmean[anidx, timeAfter] <- NA  
 }  
   
 }  
   
 # Update to include death states  
 didx <- which.max(deathyr[anidx, ])  
 if(didx < dcut) {  
 fate[anidx, didx:ncol(fate)] <- 0   
 }  
   
   
}

A few things to note in the code. We start by looping over individuals based on their EGNo. We subset the entanglement data to just have the one or more events for each individual animal. When we subset (the dsub data frame), we pull out the information on the individual events, severity, time of start/stop, etc.

We then loop over these individual events to find the ~~start~~ **end** position of the event. We then calculate the severity, lumping by category, i.e. minor - with gear (valStart == 4) and minor without gear (valStart == 6) get collapsed to a minor fate with a value of 3.

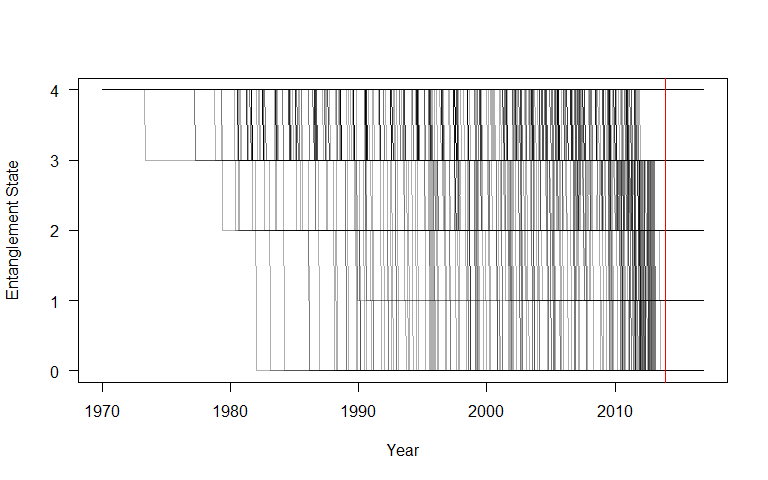
From this position in time until the end of the modeling time (ncol(fate)), we then set the states equal to the lumped severity category. If a later entanglement event has a lower category, i.e. it’s a worse entanglement, then the fate gets updated from the start of *that* event until the end of modeling time.

The two nested if() statements make sure we do not count reproductively active but unentangled females in the reference category. To do this, for any female we find the month of her first calf, and, using some indexing, we set all her health values to NA for the time following her first calf up until the time she encounters her first entanglement. This ensures that the health of all animals in the “unentangled” state does not include natural variation from health in reproductive females.

Note that when doing this updating of the health, I first check that the first calf (t1) comes before the first entanglement (t2). If yes, then the health for this period – reproductively active but not *yet* entangled – is not included in the unentangled reference class.

And then we plot it:

plot(dateVec, fate[1, ], type = 'l', ylim = c(0, 4), col = "#00000050", ylab = 'Entanglement State', las = 1, xlab = 'Year')  
for(i in 2:nrow(fate)){  
 lines(dateVec, fate[i, ], col = "#00000050")  
}  
abline(v = dateVec[dcut], col = 'red')



I’m not sure what all to make analytically out of this plot, but one thing we might want to compare and/or examine further is the rate of change from state to state over the years to see how it lines up with the breaking strength information. For example, it seems like there was a big increase in minor to moderate status in the late 1990’s, and a bit of an increase in moderate to severe in the early 00’s.

## Sample and Plot Health

With the fate vectors complete for each individual animal, we use those in conjunction with the health data, i.e. the posterior estimates of individual health, to gauge how health of different classes of entangled animals changes.

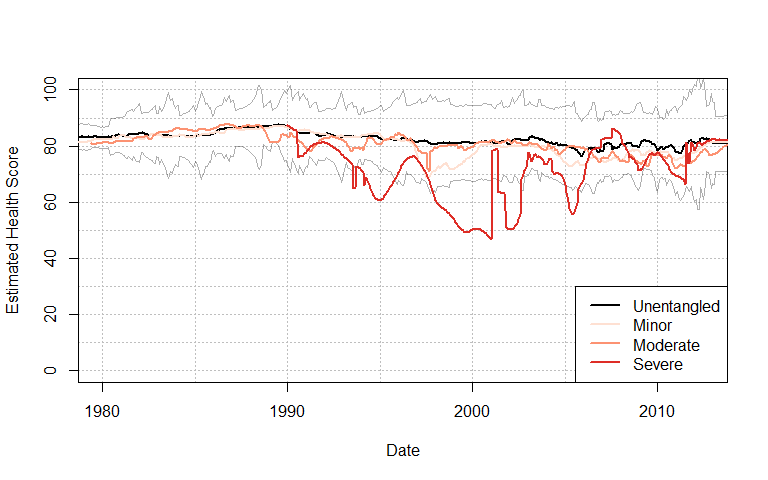
To get the fate data married with the health data, we:

1. loop over the columns in that health matrix (n.b. that each row corresponds to an animal, each column corresponds to a specific month within the modeling domain)
2. tally the number of animals in each class from fate
3. intersect fate with health to get the summary statistics (mean, sd) from each grouping at each time

dfAll <- numeric(0)  
health <- healthmean[noquote(rownames(healthmean)) %in% ID, ]  
  
for (i in 1:ncol(health)){  
 df <- data.frame(id = ID, health = health[, i], fate = fate[, i], dateTime = dateVec[i])  
 dfSummary <- df %>%   
 group\_by(fate) %>%   
 summarize(median = median(health, na.rm = TRUE),   
 sd = sd(health, na.rm = TRUE),  
 num\_an = n()) %>%   
 mutate(time = i) %>%  
 mutate(dateTime = dateVec[i])  
 dfAll <- bind\_rows(dfAll, dfSummary)  
}

Ok, with it assembled, let’s try and plot it out.

mypal <- brewer.pal(3, 'Reds')  
unentangled <- dfAll %>% filter(fate == 4)  
minor <- dfAll %>% filter(fate == 3)  
moderate <- dfAll %>% filter(fate == 2)  
severe <- dfAll %>% filter(fate == 1)  
  
plot(unentangled$dateTime, unentangled$median, type = 'l', lwd = 2, ylim = c(0, 100),  
 panel.first =   
 c(abline(h = seq(0, 100, 10), lty = 3, col = 'grey'),   
 abline(v = as.Date(c("1980-01-01", "1985-01-01", "1990-01-01", "1995-01-01",   
 "2000-01-01", "2005-01-01", "2010-01-01")),   
 lty = 3, col = 'grey')),   
 xlim = c(as.Date('1980-01-01'), as.Date("2012-07-01")),  
 xlab = 'Date', ylab = "Estimated Health Score")  
lines(unentangled$dateTime, unentangled$median + unentangled$sd, col = "#00000050")  
lines(unentangled$dateTime, unentangled$median - unentangled$sd, col = "#00000050")  
  
lines(minor$dateTime, minor$median, lty = 1, col = mypal[1], lwd = 2)  
# lines(minor$dateTime, minor$mean + minor$sd, lty = 2, col = mypal[1])  
# lines(minor$dateTime, minor$mean - minor$sd, lty = 2, col = mypal[1])  
  
lines(moderate$dateTime, moderate$median, lty = 1, col = mypal[2], lwd = 2)  
# lines(moderate$dateTime, moderate$mean + moderate$sd, lty = 2, col = mypal[2])  
# lines(moderate$dateTime, moderate$mean - moderate$sd, lty = 2, col = mypal[2])  
  
lines(severe$dateTime, severe$median, lty = 1, col = mypal[3], lwd = 2)  
# lines(severe$dateTime, severe$mean + severe$sd, lty = 2, col = mypal[3])  
# lines(severe$dateTime, severe$mean - severe$sd, lty = 2, col = mypal[3])  
legend( "bottomright", col = c('black', mypal), legend = c("Unentangled", 'Minor', "Moderate", 'Severe'), lty = 1, lwd = 2)

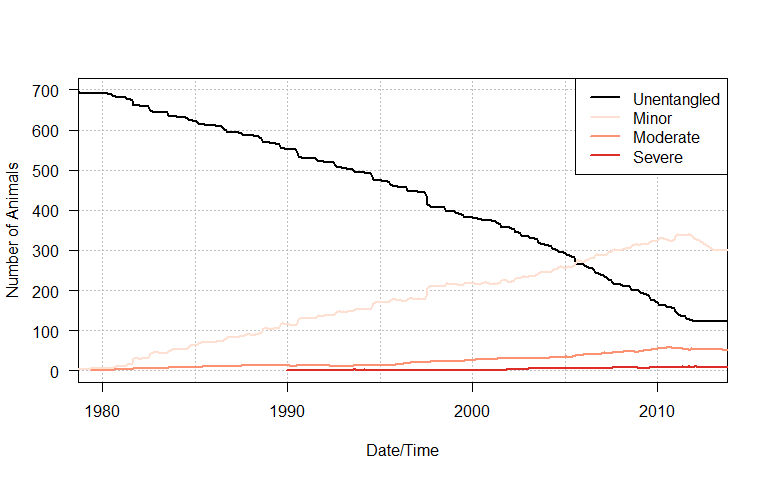


The above needs additional data for context – namely how many whales are in each state at each time. Let’s add that below. First we tally it and then convert it to a normalized matrix that shows the percentage of animals in each class.

fateEnt <- fate  
fateEnt[fateEnt == 0] <- NA  
tallyList <- lapply(split(fateEnt, col(fateEnt)), table)  
  
tmat <- matrix(NA, nrow = 4, ncol(fateEnt))  
for(i in 1:ncol(fateEnt)){  
 vdat <- as.vector(unlist(tallyList[[i]]))  
 vidx <- noquote(names(unlist(tallyList[[i]])))  
 tmat[match(vidx, 1:4), i] <- vdat  
}  
tpct <- 100 \* (tmat / colSums(tmat, na.rm = TRUE))

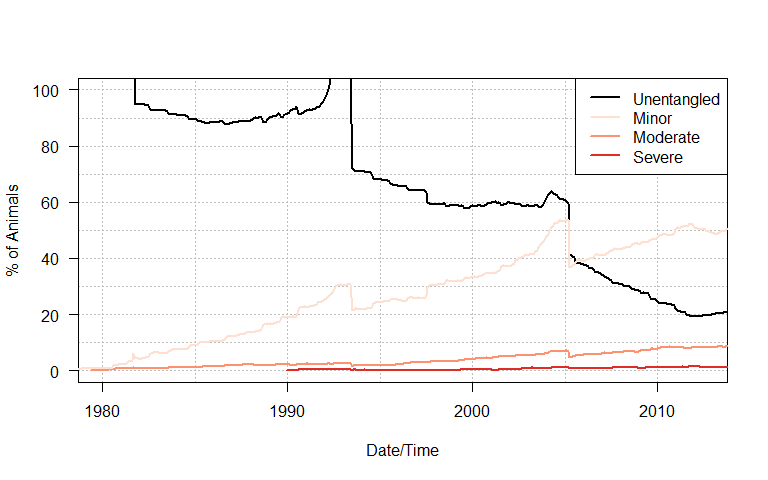
With that assembled we can plot the raw numbers:

plot(dateVec, tmat[4, ], ylim = c(0, 700), type = 'l', panel.first =   
 c(abline(h = seq(0, 700, 100), lty = 3, col = 'grey')   
 ,abline(v = as.Date(c("1980-01-01", "1985-01-01", "1990-01-01", "1995-01-01", "2000-01-01", "2005-01-01", "2010-01-01")),   
 lty = 3, col = 'grey')),  
 xlim = c(as.Date('1980-01-01'), as.Date("2012-07-01")),   
 xlab = 'Date/Time', ylab = 'Number of Animals', las = 1, lwd = 2)  
lines(dateVec, tmat[1, ], ylim = c(0, 700), col = mypal[3], lwd = 2)  
lines(dateVec, tmat[2, ], ylim = c(0, 700), col = mypal[2], lwd = 2)  
lines(dateVec, tmat[3, ], ylim = c(0, 700), col = mypal[1], lwd = 2)  
legend( "topright", col = c('black', mypal), legend = c("Unentangled", 'Minor', "Moderate", 'Severe'), lty = 1, lwd = 2)



and the normalized values:

plot(dateVec, tpct[4, ], ylim = c(0, 100), type = 'l', panel.first =   
 c(abline(h = seq(0, 100, 10), lty = 3, col = 'grey')   
 ,abline(v = as.Date(c("1980-01-01", "1985-01-01", "1990-01-01", "1995-01-01", "2000-01-01", "2005-01-01", "2010-01-01")),   
 lty = 3, col = 'grey')),xlim = c(as.Date('1980-01-01'), as.Date("2012-07-01")),   
 xlab = 'Date/Time', ylab = '% of Animals', las = 1, lwd = 2)  
lines(dateVec, tpct[1, ], col = mypal[3], lwd = 2)  
lines(dateVec, tpct[2, ], col = mypal[2], lwd = 2)  
lines(dateVec, tpct[3, ], col = mypal[1], lwd = 2)  
legend( "topright", col = c('black', mypal), legend = c("Unentangled", 'Minor', "Moderate", 'Severe'), lty = 1, lwd = 2)

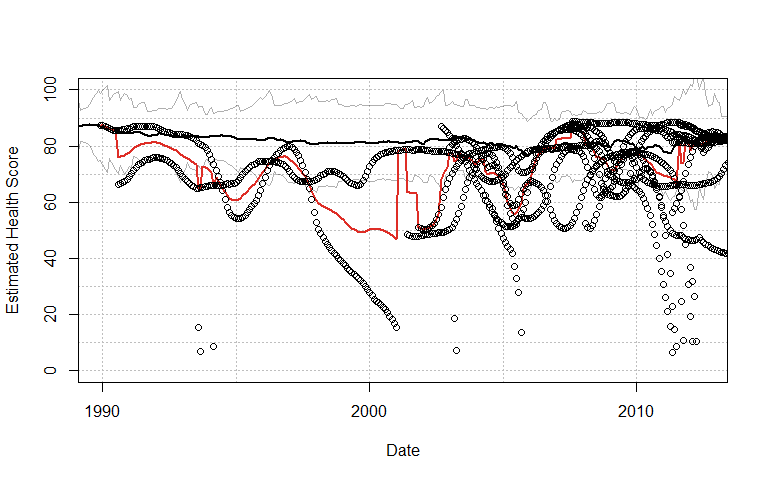


**UPDATE** We want to look at the spread of individual health values around the severe curve - especially in the early 2000’s to get a feel for what is causing the spread. So I’m going to just plot the severe curve, but also add in the scores from each individual to show the variation.

dfAll <- numeric(0)  
health <- healthmean[noquote(rownames(healthmean)) %in% ID, ]  
  
for (i in 1:ncol(health)){  
 df <- data.frame(id = ID, health = health[, i], fate = fate[, i], dateTime = dateVec[i])  
 dfSummary <- df %>%   
 filter(fate == 1) %>%   
 mutate(time = i)  
 dfAll <- bind\_rows(dfAll, dfSummary)  
}

Ok, then we need to plot it out with the severe data as a line, and the individual whales as points around that line. This is going to be a mess:

plot(unentangled$dateTime, unentangled$median, type = 'l', lwd = 2, ylim = c(0, 100),  
 panel.first = c(abline(h = seq(0, 100, 10), lty = 3, col = 'grey'),   
 abline(v = as.Date(c("1990-01-01", "1995-01-01",   
 "2000-01-01", "2005-01-01", "2010-01-01")),   
 lty = 3, col = 'grey')),   
 xlim = c(as.Date('1990-01-01'), as.Date("2012-07-01")),  
 xlab = 'Date', ylab = "Estimated Health Score")  
lines(unentangled$dateTime, unentangled$median + unentangled$sd, col = "#00000050")  
lines(unentangled$dateTime, unentangled$median - unentangled$sd, col = "#00000050")  
lines(severe$dateTime, severe$median, lty = 1, col = mypal[3], lwd = 2)  
points(dfAll$dateTime, dfAll$health)



So just a quick take on that, especially for the period in the early 2000’s. In that first few years (2000-2002) we only have two severely entangled whales, and one of them dies (February 2001), so the health jumps up since the one animal was doing ok (and would actually carry on being ok for many years). In June of 2001, we have another severe entanglement, and that animal’s health is poor, so the average health jumps from 78 down to 63. In November 2001, we have a 3rd severely entangled whale whose health is also poor, and we see a jump from 63 down to 51.

I think the code is doing what we want it to do, but we have a situation whereby there are only a few animals with severe entanglements, so their averages are jumping around al over the place.