Kristina White
ACMS 40950
Final Project
November 20, 2020

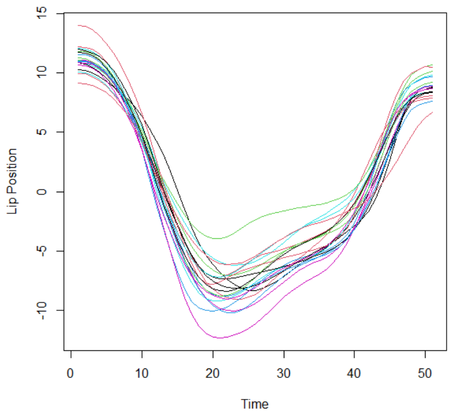
Registration of Lip Motion Data

This project uses the Lip Motion dataset in the fda library in R. The matrix lip contains 51 measurements of the position of the lower lip every 7 milliseconds for 20 repetitions of the syllable 'bob'. There is also a 20x20 matrix lipmarks that contains the positions of the 'leftElbow' and 'rightElbow' for each of the 20 repetitions of the syllable 'bob,' and liptime contains the time in seconds for every 7 milliseconds over the 51 measurements. The project will focus on the landmark registration of the lip data.

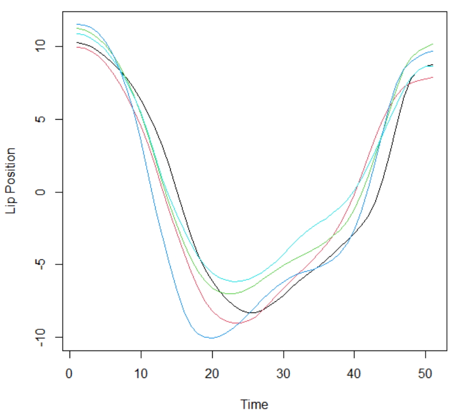
Many approaches in functional data analysis only account for variation in amplitude, but many observed data features also vary in phase or time. Registration aligns these functions so amplitude variation may be better studied. The goal is to transform observed curves x_i using time, or the points on the x-axis, so qualitative features align better. We want to find new time point $s_i = w_i(t)$ so the new $x_i(t)$ is equal to $x(s_i)$. This is the same curve observed at different time points that are better aligned. The transformation function $w_i(t)$ is called a time-warping function. Landmark registration tries to align specific points, called landmarks ($t_{i1},\ldots t_{iK}$). These are usually where $x_i(t)$ crosses some value, the locations of peaks or valleys, or the locations of inflection points. One curve is chosen as a reference (with landmarks $t_{01},\ldots t_{0K}$), and we align other curves to this curve. These define constraints such that $w_i(t_{ij}) = t_{0j}$ for $j = 1,\ldots,K$, or so the warping function for t_{ij} is equivalent to the reference landmark t_{0j}. We then define a smooth function to go between these landmarks.

There are three major constraints on warping functions $w_i(t)$ for $t \in (0, T)$, or for the time points between 0 and some time point T. First, for initial conditions, the starting and ending points should be 0 and T, so $w_i(0) = 0$ and $w_i(T) = T$ to maintain the original interval. Second, $w_i(t_{ij}) = t_{0j.}$ Third, $w_i(t)$ should be monotone, so if $t_1 < t_2$, $w_i(t_1) < w_i(t_2)$.

I investigated two options to use as landmarks to align the lip curves: using the point with the minimum measurement for each of the 20 repetitions of the syllable “bob” and using the lipmarks matrix. I decided to use the points with the minimum measurements. Looking at the lip data, the curves are not aligned. I have included plots using both the full 20 curves and just the first 5 curves to better illustrate how the curves are out of phase. I will perform registration on all 20 curves as well as just the first 5 curves. The goal is to see how registration effects the total variation of the curves for all 20 curves versus just 5 curves.



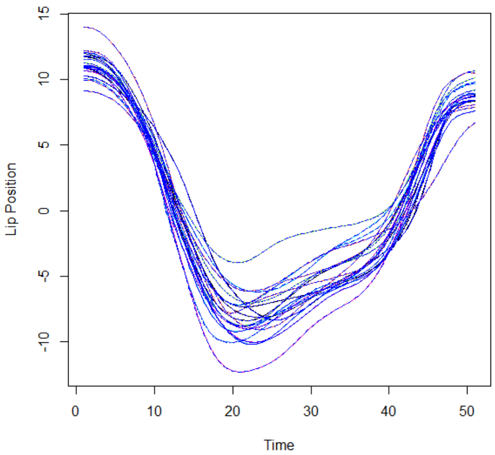
20 curves



5 curves

I will start with the registration of all 20 curves. I first fit a basis to smooth the lip data. I used a B-spline basis of order 4 with knots at the time points defined by 1:dim(lip)[1]. I chose the smoothing parameter lambda to be 1e-12 and penalized the second derivative since I am interested in the original function. I then created a functional data object using smooth.basis:

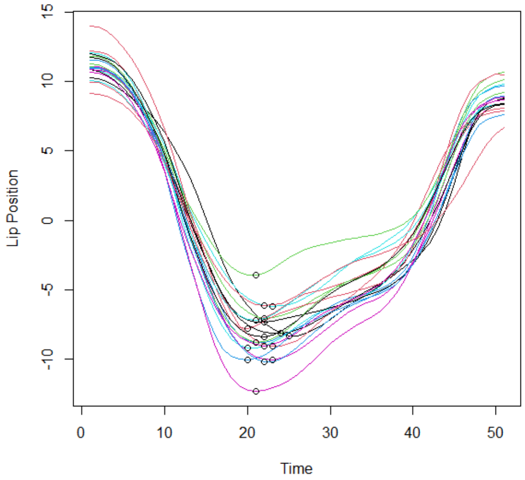
```
time = 1:51

rng = range(time)

knots  <- time

norder <- 4

nbasis <- length(knots) + norder - 2

bbasis <- create.bspline.basis(rng, nbasis, norder, knots)

par = fdPar(bbasis,Lfdobj=int2Lfd(2),lambda=1e-12)

fd = smooth.basis(time, lip, par)$fd
```



20 curves with basis functions

Next, I set the landmarks to be the point of the minimum measurement for each of the 20 curves:

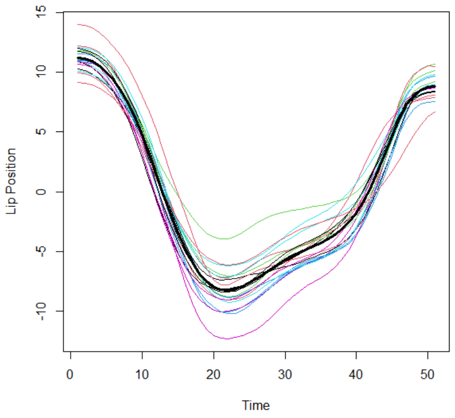
```
landmark = rep(NA,dim(lip)[2])
for(i in 1:dim(lip)[2]){
landmark[i] = which.min(lip[, i])
points(time[landmark[i]], lip[landmark[i],i])
}
landmark
```



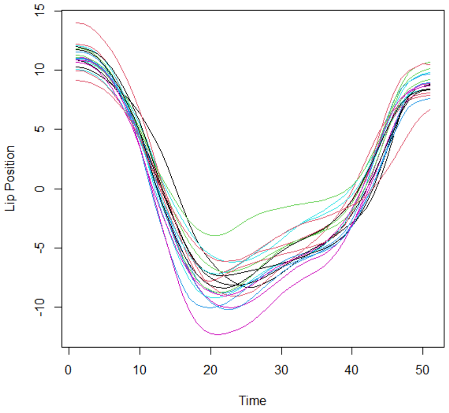
20 curves with landmarks

I then used landmarkreg to align the curves, using the mean of the landmarks as the target function, and plotted the aligned curves and the mean aligned curve:

```
ldmk = landmarkreg(fd, landmark, mean(landmark), par)

r.lip = eval.fd(time, ldmk$regfd)

matplot(time, r.lip, type="l", lty=1,

        xlab="Time", ylab="Lip Position")

mr.lip = apply(r.lip, 1, mean)

lines(time, mr.lip, type="l", lty=1, lwd=3)
```



aligned

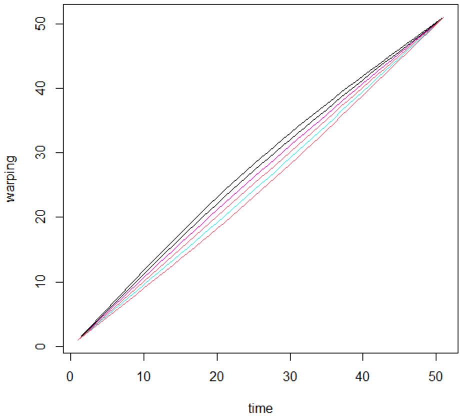


unaligned

The registered curves look to be more aligned than the original curves.

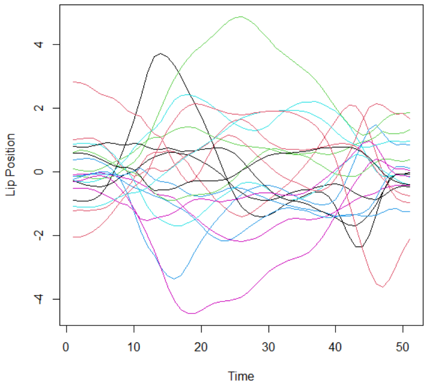
I also plotted the warping function used to transform the curves:

```
w.lip = eval.fd(time, ldmk$warpfd)

matplot(time, w.lip, type="l", lty=1, xlab="time", ylab="warping")
```

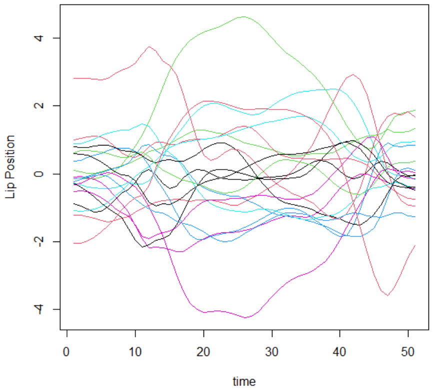


I used center functions to see the change in variation resulting from registration:

```
cfd = center.fd(fd)

c.lip = eval.fd(time, cfd)

matplot(time, c.lip, type="l", lty=1,

        xlab="Time", ylab="Lip Positions")


rcfd = center.fd(ldmk$regfd)

rc.lip = eval.fd(time, rcfd)

matplot(time, rc.lip, type="l", lty=1,

        xlab="time", ylab="Lip Positions")
```



unregistered



registered

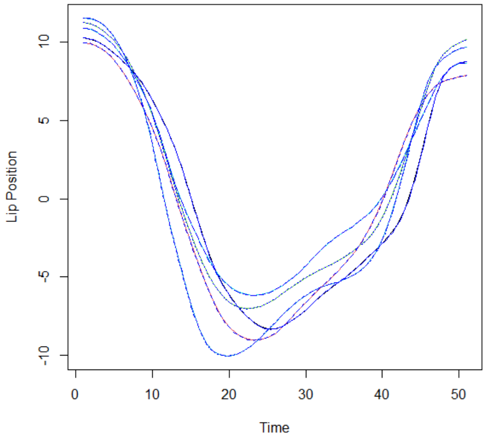
The center functions center each of the curves on the mean function by subtracting the pointwise mean from each of the functions. Looking at the centered functions, some of the curves seem to vary less, and some seem to vary more. The fraction "varprop" is a fraction of the total variation of the curves after registration to that before registration. This fraction can be calculated by:

varprop = sum(diag(inprod(rcfd,rcfd)))/sum(diag(inprod(cfd,cfd)))

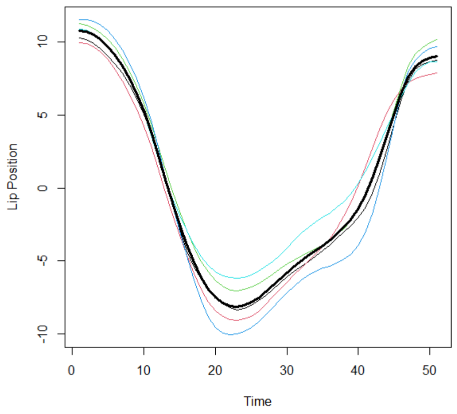
In this case, varprop = 1.002854. Since this fraction is greater than 1, this suggests the total variation of the curves actually increased after registration. Because registration did not reduce the total variation of the curves, it does not provide meaningful results in this case.

I would like to compare the registration of all 20 curves to the registration of just the first 5. We have already seen the unalignment of the curves is more pronounced when considering just these 5 curves. Will registration have a meaningful effect and reduce the total variation when looking at just 5 curves?

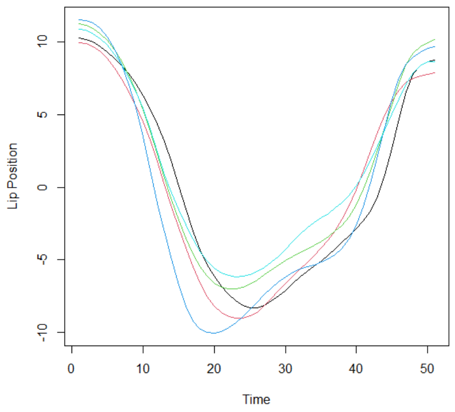
I fit the 5 curves using the same B-spline basis with the same smoothing parameter and penalty:



I used the same method to find the landmarks as the point with the minimum measurement for each of the 5 curves and used landmarkreg to align the curves, again using the mean of the landmarks as the reference function. I then plotted the aligned curves and their mean:

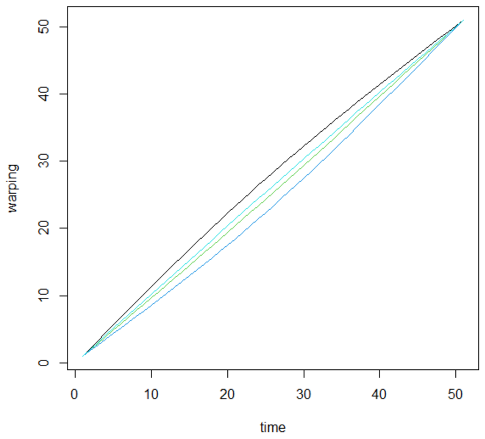


aligned

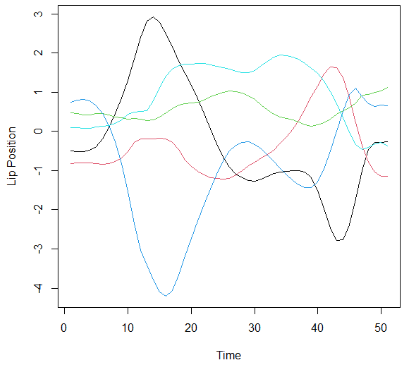


unaligned

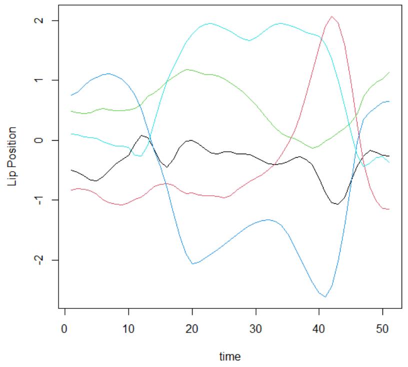
The registered curves are clearly more in phase than the unregistered curves. I also plotted the warping functions:



Plotting the centered functions shows the curves have less variation after registration:



unregistered



registered

For the 5 curves, varprop = 0.6744768, which means only about 67% of the total variation in the curves remained after registration. Therefore, about 33% of the variation can be accounted for by the shift registration.

Registration failed to produce meaningful results for all 20 curves but did reduce the total variation when just the first 5 curves were registered. When looking at the original 20 curves versus the original 5 curves, the phase unalignment was more pronounced for just the 5 curves. Some of the 20 curves did seem well aligned. Therefore, when working with all of the repetition curves, registration is not necessary, but it may be helpful when only interested in a smaller subset of the repetition curves.

R code:

```
library(fda)
data(lip)
head(lip)

time = 1:51
rng = range(time)
knots  <- time
norder <- 4
nbasis <- length(knots) + norder – 2
bbasis <- create.bspline.basis(rng, nbasis, norder, knots)

par = fdPar(bbasis,Lfdobj=int2Lfd(2),lambda=1e-12)
fd = smooth.basis(time, lip, par)$fd

matplot (time, lip, type="l", lty=1, cex=2,

        lwd=1,  xlab = "Time", ylab="Lip Position")
lines(fd, col="blue")

landmark = rep(NA,dim(lip)[2])
for(i in 1:dim(lip)[2]){

  landmark[i] = which.min(lip[, i])

  points(time[landmark[i]], lip[landmark[i],i])

}

landmark

ldmk = landmarkreg(fd, landmark, mean(landmark), par)

r.lip = eval.fd(time, ldmk$regfd)
matplot(time, r.lip, type="l", lty=1, xlab="Time", ylab="Lip Position")
```

```
mr.lip = apply(r.lip, 1, mean)
lines(time, mr.lip, type="l", lty=1, lwd=3)

w.lip = eval.fd(time, ldmk$warpfd)
matplot(time, w.lip, type="l", lty=1, xlab="time", ylab="warping")

cfd = center.fd(fd)
c.lip = eval.fd(time, cfd)
matplot(time, c.lip, type="l", lty=1, xlab="Time", ylab="Lip Position")

rcfd = center.fd(ldmk$regfd)
rc.lip = eval.fd(time, rcfd)
matplot(time, rc.lip, type="l", lty=1, xlab="time", ylab="Lip Position")

varprop = sum(diag(inprod(rcfd,rcfd)))/sum(diag(inprod(cfd,cfd)))
varprop

par = fdPar(bbasis,Lfdobj=int2Lfd(2),lambda=1e-12)
fd = smooth.basis(time, lip[, 1:5], par)$fd

matplot (time, lip[, 1:5], type="l", lty=1, cex=2,

        lwd=1,  xlab = "Time", ylab="Lip Position")
lines(fd, col="blue")

landmark = rep(NA,5)
for(i in 1:5){

  landmark[i] = which.min(lip[, i])

  points(time[landmark[i]], lip[landmark[i],i])

}
landmark

ldmk = landmarkreg(fd, landmark, mean(landmark), par)

r.lip = eval.fd(time, ldmk$regfd)
matplot(time, r.lip, type="l", lty=1, xlab="Time", ylab="Lip Position")
mr.lip = apply(r.lip, 1, mean)
lines(time, mr.lip, type="l", lty=1, lwd=3)

w.lip = eval.fd(time, ldmk$warpfd)
matplot(time, w.lip, type="l", lty=1, xlab="time", ylab="warping")

cfd = center.fd(fd)
c.lip = eval.fd(time, cfd)
matplot(time, c.lip, type="l", lty=1, xlab="Time", ylab="Lip Position")
```

```
rcfd = center.fd(ldmk$regfd)
rc.lip = eval.fd(time, rcfd)
matplot(time, rc.lip, type="l", lty=1, xlab="time", ylab="Lip Position")

varprop = sum(diag(inprod(rcfd,rcfd)))/sum(diag(inprod(cfd,cfd)))
varprop
```