Assignment 11; STAT 689

Philip Anderson; panders2@tamu.edu

4/02/2018

```
library("HRW")
library("mgcv")
library("cosso")
library("tidyverse")
```

Question 1

```
data("BostonMortgages")
```

```
str(BostonMortgages)
```

```
## 'data.frame':    2380 obs. of  13 variables:
##  $ dir        : num  0.221 0.265 0.372 0.32 0.36 ...
##  $ hir        : num  0.221 0.265 0.248 0.25 0.35 ...
##  $ lvr        : num  0.8 0.922 0.92 0.86 0.6 ...
##  $ ccs        : int  5 2 1 1 1 1 1 2 2 2 ...
##  $ mcs        : int  2 2 2 2 1 1 2 2 2 1 ...
##  $ pbcr       : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ dmi        : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 2 1 ...
##  $ self       : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ single     : Factor w/ 2 levels "no","yes": 1 2 1 1 1 1 2 1 1 2 ...
##  $ uria       : num  3.9 3.2 3.2 4.3 3.2 ...
##  $ comdominiom: int  0 0 0 0 0 0 1 0 0 0 ...
##  $ black      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ deny       : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 2 1 ...
```

1A

Which model does Cosso select?

```
# create a 1/0 indicator for deny so we know what the procedures are predicting
BostonMortgages$deny_bin <- ifelse(BostonMortgages$deny=="yes", 1, 0)
# corroborate results with contingency table
table(BostonMortgages$deny_bin, BostonMortgages$deny)
```

```
##
##        no  yes
##    0 2095    0
##    1    0  285
```

```
# generate design matrix
# column 5 has a huge outlier - drop it so that cosso will run
BostonMortgages_red <- BostonMortgages %>%
  dplyr::filter(dir < 1)
```

```r
# one-hot-encode the factor variables, dropping the intercept this function makes along the way
fac_to_int <- stats::model.matrix(~ black + pbcr + self + single, BostonMortgages_red)[, -1] %>%
  data.frame()
# grab the non-factor variables
int_vars <- BostonMortgages_red[, c("dir", "lvr")]
# bring everything together
X <- as.matrix(cbind(fac_to_int, int_vars))

# generate response array with custom response variable
y <- BostonMortgages_red[, c("deny_bin")]
```

```r
# run the cosso model
start <- Sys.time()
cosso_mod <- cosso::cosso(x=X, y=y, family=c("Binomial"))
end <- Sys.time()
# runtime
print(end - start)
```

```
## Time difference of 29.4271 secs
```

```r
# we will want to see how our coefficients jointly change with various tuning parameters
tune_matrix <- data.frame(cosso_mod$tune$Mgrid
                  , cosso_mod$tune$L2norm
                )
names(tune_matrix) <- c("M","blackyes", "pbcryes", "selfyes", "singleyes", "dir", "lvr")
print(round(tune_matrix, 2))
```

```
##       M blackyes pbcryes selfyes singleyes  dir  lvr
## 1  0.00     0.00    0.00       0      0.00 0.00 0.00
## 2  0.20     0.00    0.00       0      0.00 1.31 0.00
## 3  0.45     0.00    0.00       0      0.00 1.28 0.00
## 4  0.70     0.00    0.00       0      0.00 1.26 0.00
## 5  0.95     0.00    0.00       0      0.00 1.25 0.00
## 6  1.20     0.00    1.89       0      0.00 1.25 0.00
## 7  1.45     0.00    1.89       0      0.00 1.24 0.00
## 8  1.70     0.00    1.89       0      0.00 1.25 0.00
## 9  1.95     0.00    1.78       0      0.00 1.49 0.69
## 10 2.20     0.89    1.65       0      0.00 1.44 0.62
## 11 2.45     0.89    1.65       0      0.00 1.43 0.63
## 12 2.95     0.89    1.65       0      0.00 1.43 0.64
## 13 3.45     0.89    1.65       0      0.00 1.42 0.65
## 14 3.95     0.84    1.68       0      0.38 1.38 0.63
```

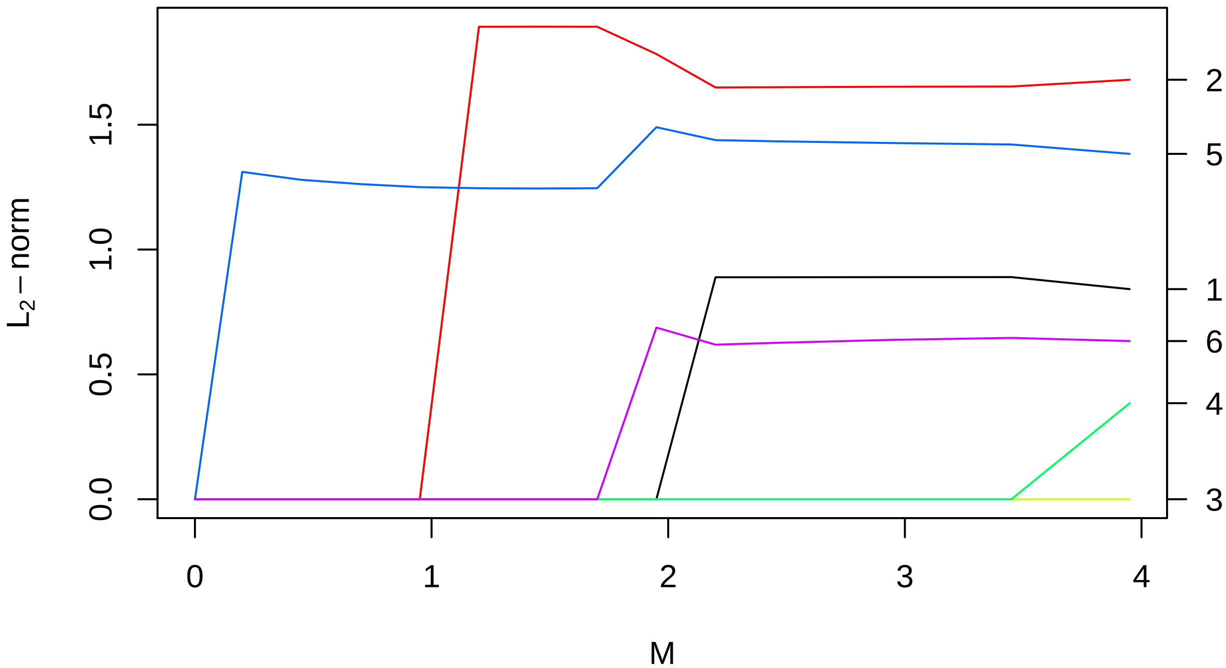
```r
# we will now want to select an optimal tuning paramter
start <- Sys.time()
set.seed(1738)
tune_mod <- cosso::tune.cosso(cosso_mod, 10, FALSE)
fin <- Sys.time()
print(fin - start)
```

```
## Time difference of 55.35386 secs
```

```r
# print out the regularization trace
print(tune_mod$OptM)
```

```
## [1] 3.45
```

```r
cosso::plot.cosso(cosso_mod, M=tune_mod$OptM)
```




```r
# determine what our coefficients for this model will be based on our tuning matrix
tune_matrix %>%
  dplyr::filter(M==tune_mod$OptM) %>%
  print()
```

```
##      M  blackyes  pbcryes selfyes singleyes      dir       lvr
## 1 3.45 0.8896404 1.652929       0         0 1.420734 0.6461124
```

Cosso selects *black*, *pbcr*, *dir*, and *lvr* for our model.

1B

Which model does mgcv select?

I am going to use the exact same data set generated for the cosso problem, but moved from matrix form in to a data frame. Note that this means I am going to keep the outlier removed.

```r
boston_df <- data.frame(cbind(y, X))
# rename our response
names(boston_df)[1] <- "deny_bin"
head(boston_df)
```

```
##   deny_bin blackyes pbcryes selfyes singleyes   dir       lvr
## 1        0        0       0       0         0 0.221 0.8000000
## 2        0        0       0       0         1 0.265 0.9218750
## 3        0        0       0       0         0 0.372 0.9203980
## 4        0        0       0       0         0 0.320 0.8604651
## 5        0        0       0       0         0 0.360 0.6000000
## 6        0        0       0       0         0 0.240 0.5105263
```

```r
mgcv_mod <- mgcv::gam(deny_bin ~ blackyes + pbcryes + selfyes + singleyes + dir + lvr
                      , data=boston_df
```

```r
                      , family=binomial(link="logit")
                      , select=TRUE
                      )
mgcv::summary.gam(mgcv_mod)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## deny_bin ~ blackyes + pbcryes + selfyes + singleyes + dir + lvr
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -6.4130     0.4722 -13.581  < 2e-16 ***
## blackyes      0.9292     0.1586   5.858 4.68e-09 ***
## pbcryes       1.6533     0.1833   9.018  < 2e-16 ***
## selfyes       0.5716     0.1988   2.875  0.00404 **
## singleyes     0.3833     0.1391   2.756  0.00585 **
## dir           5.1953     0.8082   6.428 1.29e-10 ***
## lvr           2.5333     0.4531   5.590 2.26e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  0.151   Deviance explained = 15.6%
## UBRE = -0.38041  Scale est. = 1         n = 2375
```

The mgcv procedure selected all of the variables that we put into the model.

1C

The variables selected by both models are relatively close, but not exact. Given past experiences with both packages, I am more readily inclined to trust mgcv over cosso.

Question 2

```r
data("femSBMD")
names(femSBMD) <- tolower(names(femSBMD))
str(femSBMD)
```

```
## 'data.frame':    1003 obs. of  7 variables:
##  $ idnum    : int  1 1 1 1 2 2 2 2 3 3 ...
##  $ spnbmd   : num  0.719 0.732 0.776 0.781 0.62 0.627 0.759 0.79 0.641 0.622 ...
##  $ age      : num  11.2 12.2 13.2 14.3 12.7 13.8 14.8 15.8 10.9 11.9 ...
##  $ ethnicity: Factor w/ 4 levels "Asian","Black",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ black    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hispanic : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ white    : int  1 1 1 1 1 1 1 1 1 1 ...
```

2A

Define a new identification variable, given as 2 times the existing identification variable.

```r
femSBMD$idnum2 <- 2*femSBMD$idnum
str(femSBMD)
```

```
## 'data.frame':    1003 obs. of  8 variables:
##  $ idnum    : int  1 1 1 1 2 2 2 2 3 3 ...
##  $ spnbmd   : num  0.719 0.732 0.776 0.781 0.62 0.627 0.759 0.79 0.641 0.622 ...
##  $ age      : num  11.2 12.2 13.2 14.3 12.7 13.8 14.8 15.8 10.9 11.9 ...
##  $ ethnicity: Factor w/ 4 levels "Asian","Black",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ black    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hispanic : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ white    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ idnum2   : num  2 2 2 2 4 4 4 4 6 6 ...
```

2B

Rerun the gamm given in class, using first the original idnum variable and then the new one. Ensure that we are getting the same results with each.

```r
# original fit
class_fit <- mgcv::gamm(spnbmd ~ s(age) + black + hispanic + white
                        , random=list(idnum = ~1) # intercept allowed to vary randomly
                        , data=femSBMD
                         )
# new fit
hw_fit <- mgcv::gamm(spnbmd ~ s(age) + black + hispanic + white
                        , random=list(idnum2 = ~1)
                        , data=femSBMD
                         )
```

```r
# print the original results
summary(class_fit$gam)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## spnbmd ~ s(age) + black + hispanic + white
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.92538    0.01243  74.444  < 2e-16 ***
## black        0.08191    0.01718   4.769 2.13e-06 ***
## hispanic    -0.01516    0.01754  -0.864    0.388
## white        0.01503    0.01748   0.860    0.390
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df     F p-value
## s(age) 7.201  7.201 225.6  <2e-16 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.519
##   Scale est. = 0.0013551  n = 1003
```

```r
# print out the updated results
summary(hw_fit$gam)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## spnbmd ~ s(age) + black + hispanic + white
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.92538    0.01243  74.444  < 2e-16 ***
## black        0.08191    0.01718   4.769 2.13e-06 ***
## hispanic    -0.01516    0.01754  -0.864    0.388
## white        0.01503    0.01748   0.860    0.390
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##          edf Ref.df     F p-value
## s(age) 7.201  7.201 225.6  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.519
##   Scale est. = 0.0013551  n = 1003
```

The specific number of the id number variable does not appear to matter in the gamm function's grouping of observations.

Question 3

3A

```r
(version_info <- version)
```

```
##                  _
## platform       x86_64-apple-darwin15.6.0
## arch           x86_64
## os             darwin15.6.0
## system         x86_64, darwin15.6.0
## status
## major          3
## minor          4.3
## year           2017
## month          11
```

```
## day            30
## svn rev        73796
## language       R
## version.string R version 3.4.3 (2017-11-30)
## nickname       Kite-Eating Tree
```

```
version_num <- paste0(version_info$major, ".", version_info$minor)
cat("\n")
```

```
print(paste0("version num: ", version_num))
```

```
## [1] "version num: 3.4.3"
```

3C

```
library("rstan")
```

```
## Loading required package: StanHeaders
```

```
## rstan (Version 2.17.3, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
##
## Attaching package: 'rstan'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

RStan has loaded successfully.

Question 4

```
pig.weights <- read.csv("/Users/panders2/Documents/schools/tamu/stat_689/homework/semiparametric-regres
names(pig.weights) <- tolower(names(pig.weights))
str(pig.weights)
```

```
## 'data.frame':    432 obs. of  3 variables:
##  $ id.num   : int  1 1 1 1 1 1 1 1 1 2 ...
##  $ num.weeks: int  1 2 3 4 5 6 7 8 9 1 ...
##  $ weight   : num  24 32 39 42.5 48 54.5 61 65 72 22.5 ...
```

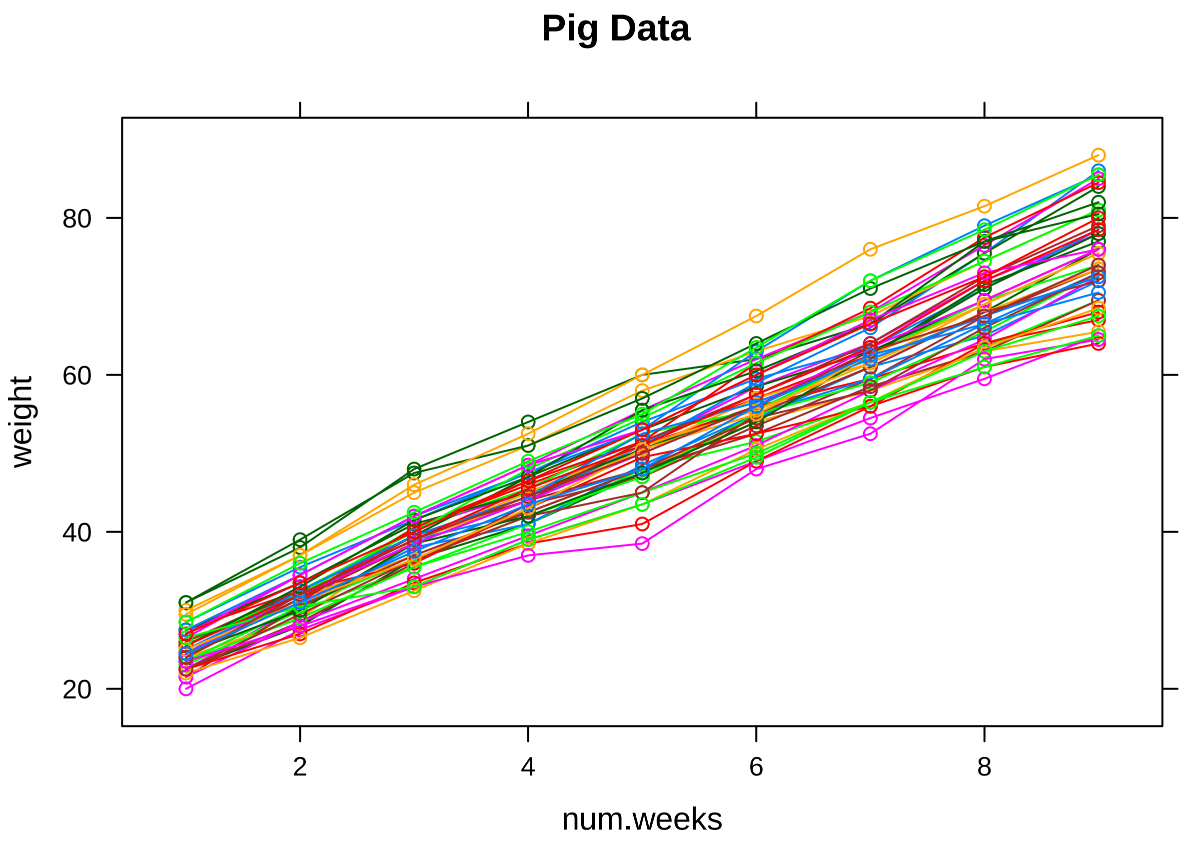
```
library("lattice")
```

4A

Display the lattice plot of the pig data.

```
lattice::xyplot(weight~num.weeks
                , data=pig.weights
```

```
               , groups=id.num
               , type="b"
               , main="Pig Data"
               )
```



4B

Looking at the data, it appears that a random-intercept model holds for these data. Note that the slope of each pig's data appears to be the same - the main difference between the lines tends to be the point where they started.

4C

Fit the random-intercept model and give your code. Also, do a summary and show your results.

```
random_int <- mgcv::gamm(weight ~ num.weeks
                          , random=list(id.num = ~1)
                          , data=pig.weights
                          )
# first grab the model coefficients
summary(random_int$gam)
```

```
##
## Family: gaussian
## Link function: identity
##
```

```
## Formula:
## weight ~ num.weeks
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.3556     0.5988   32.32   <2e-16 ***
## num.weeks     6.2099     0.0391  158.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =   0.93
##   Scale est. = 4.3833    n = 432
```

```
# now grab confidence interval information
intervals(random_int$lme)
```

```
## Approximate 95% confidence intervals
##
##  Fixed effects:
##                  lower      est.     upper
## X(Intercept) 18.181008 19.355613 20.530219
## Xnum.weeks    6.133191  6.209896  6.286601
## attr(,"label")
## [1] "Fixed effects:"
##
##  Random Effects:
##   Level: id.num
##                    lower     est.    upper
## sd((Intercept)) 3.130774 3.849349 4.732852
##
##  Within-group standard error:
##    lower     est.    upper
## 1.950670 2.093625 2.247056
```

The between-person (intercept) variance is captured by the 95% interval given by (3.13, 4.73), which does not contain zero and is thus significant. The within-person (random error) variance is captured by the 95% interval given by (1.95, 2.25), which also does not contain zero and is thus significant.