HW 5_updated

BBC: Bichay, Bovee, Coeman

3/21/2018

Graph 1: What do People in Other Countries think of Trump's Policies?

Load in the data and the tidyverse, and countrycode package, and the car package

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --
```

```
## <U+221A> ggplot2 2.2.1     <U+221A> purrr   0.2.4
## <U+221A> tibble  1.3.4     <U+221A> dplyr   0.7.4
## <U+221A> tidyr   0.7.2     <U+221A> stringr 1.2.0
## <U+221A> readr   1.1.1     <U+221A> forcats 0.2.0
```

```
## -- Conflicts ------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(countrycode)
library(car)
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
climate <- read.csv("/Users/nickbichay/Desktop/ /aPLS 900/Week 7/trump-world-trust/TRUMPWORLD-issue-1.cs

wall <- read.csv("/Users/nickbichay/Desktop/ /aPLS 900/Week 7/trump-world-trust/TRUMPWORLD-issue-2.csv"

Iran <- read.csv("/Users/nickbichay/Desktop/ /aPLS 900/Week 7/trump-world-trust/TRUMPWORLD-issue-3.csv"

trade <- read.csv("/Users/nickbichay/Desktop/ /aPLS 900/Week 7/trump-world-trust/TRUMPWORLD-issue-4.csv

travel <- read.csv("/Users/nickbichay/Desktop/ /aPLS 900/Week 7/trump-world-trust/TRUMPWORLD-issue-5.csv
```

Part 2: Data Manipulation

First, merge.

```
##Merge
  #first add a unique variable to each dataset identifying the dataset
climate$dataset = c("climate")
```

```
wall$dataset = c("wall")
Iran$dataset = c("Iran")
trade$dataset = c("trade")
travel$dataset = c("travel")

#then merge based on the Dataset variable
finaldata <-rbind(climate, Iran, wall, trade, travel)
```

Create a region variable using the country code package. Then, tweak the variable

```
#Use the country code package to change the region variable into REGIONS
finaldata$region <- countrycode(finaldata$country, "country.name", "region")

#Create a region Key to convert the old regions
regionKey = data.frame(dirty=unique(finaldata$region), stringsAsFactors = FALSE)
regionKey$clean = NA
regionKey$clean[grepl('Europe',regionKey$dirty)] = 'Europe\nand Russia'
regionKey$clean[grepl('Africa',regionKey$dirty)] = 'Middle East and Africa'
regionKey$clean[grepl('Asia',regionKey$dirty)] = 'Asia'
regionKey$clean[regionKey$dirty %in% c('South America', 'Central America')] = 'South America and Mexico

##Merge new clean region variable, check to make sure that it worked
finaldata$regionClean = regionKey$clean[match(finaldata$region, regionKey$dirty)]
finaldata$regionClean[finaldata$country %in% c('Israel','Jordan','Lebanon')] = 'Middle East and Africa'

#get rid of Canada, Australia/New Zealand because they are not in final viz
finaldata = na.omit(finaldata)
```

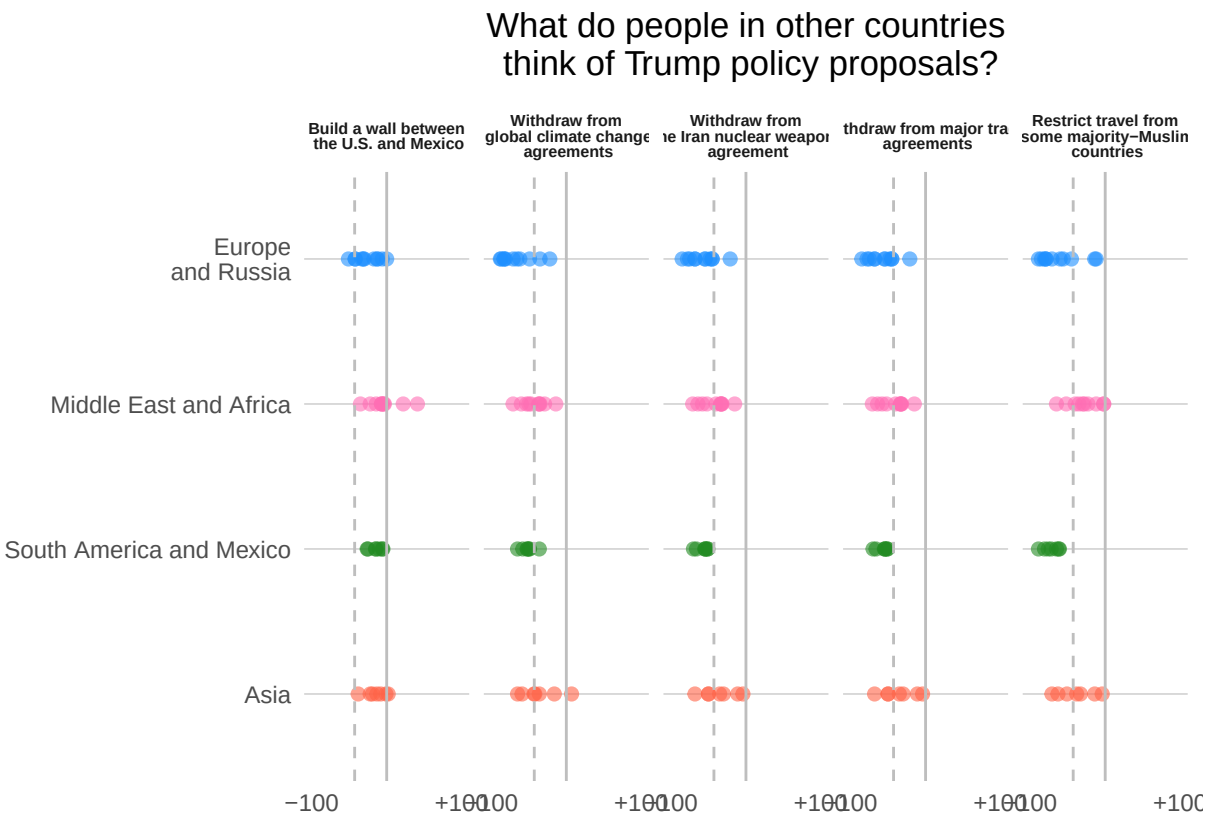
Building visualization

```
#First calculate the mean for net approval
finaldata$mean_approval = mean(finaldata$net_approval)

#acknowledge the factor level order for the regionClean variable in the final viz
finaldata$regionClean = factor(finaldata$regionClean, levels=rev(unique(finaldata$regionClean)[c(1,3,4,

#Facet Label Prep
    ##get ready to change labels on facets by making a list of eventual facet names
    facet_names <- list(
      'climate'="Withdraw from\n global climate change\n agreements",
      'Iran'="Build a wall between\n the U.S. and Mexico",
      'trade'="Withdraw from\n the Iran nuclear weapons\n agreement",
      'travel'="Withdraw from major trade\n agreements",
      'wall' = "Restrict travel from\n some majority-Muslim\n countries"
    )
    #Facet labeler function
    facet_labeler <- function(variable,value){
      return(facet_names[value])
    }

#make the visualization
finalViz <- ggplot(finaldata, aes(x = net_approval, y = regionClean, color=regionClean)) +
  geom_point(alpha=.6, size=2) +
  #change the point colors to match the viz online
  scale_color_manual(values=c("tomato", "forestgreen", "hotpink1", "dodgerblue1"))+
```

```
  #add a line for the mean net approval
  geom_vline(xintercept = mean(finaldata$net_approval), linetype="dashed", color = "gray") +
  geom_vline(xintercept=0, color='gray', linetype='solid') +
  #Cleaning up the theme: no gray background
  theme_bw() +
  scale_x_continuous('', limits=c(-100,100), breaks=c(-100, 100), labels=c('-100', '+100')) +
  guides(fill=FALSE) +
  labs(
    y='',
    title='What do people in other countries\n think of Trump policy proposals?', size = 5
  ) +
  #move position of title
  theme(plot.title = element_text(hjust=0.5)) +
  #no axis ticks, no borders, no legend
  theme(
    axis.ticks=element_blank(),
    panel.border=element_blank(),
    legend.position="none",
    #creating a horizontal line for each region, eliminating all other grid lines
    panel.grid.major.y = element_line(color="gray", size = .2),
    panel.grid.minor.y = element_blank(),
    panel.grid.major.x = element_blank(),
    panel.grid.minor.x = element_blank()
  )+
      facet_wrap(~dataset, nrow=1, labeller=facet_labeler) +
  theme(strip.text = element_text(face="bold", size=5.5),
        strip.background = element_blank())
```

```
## Warning: The labeller API has been updated. Labellers taking `variable`and
## `value` arguments are now deprecated. See labellers documentation.
```

```
finalViz
```



Graph 2 Trump Approval Rating

First, load in the data and library ggplot2

```
library(ggplot2)

approval_polllist <- read.csv( file='/Users/nickbichay/Desktop/approval_polllist.csv',
                     stringsAsFactors=FALSE )

approval_topline <- read.csv( file='/Users/nickbichay/Desktop/approval_topline.csv',
                              stringsAsFactors=FALSE )
```

Second, data manipulation:

Subset the data by all polls to make it easier to deal with. Convert the dates in the data from characters to dates so they can be used in the graphs

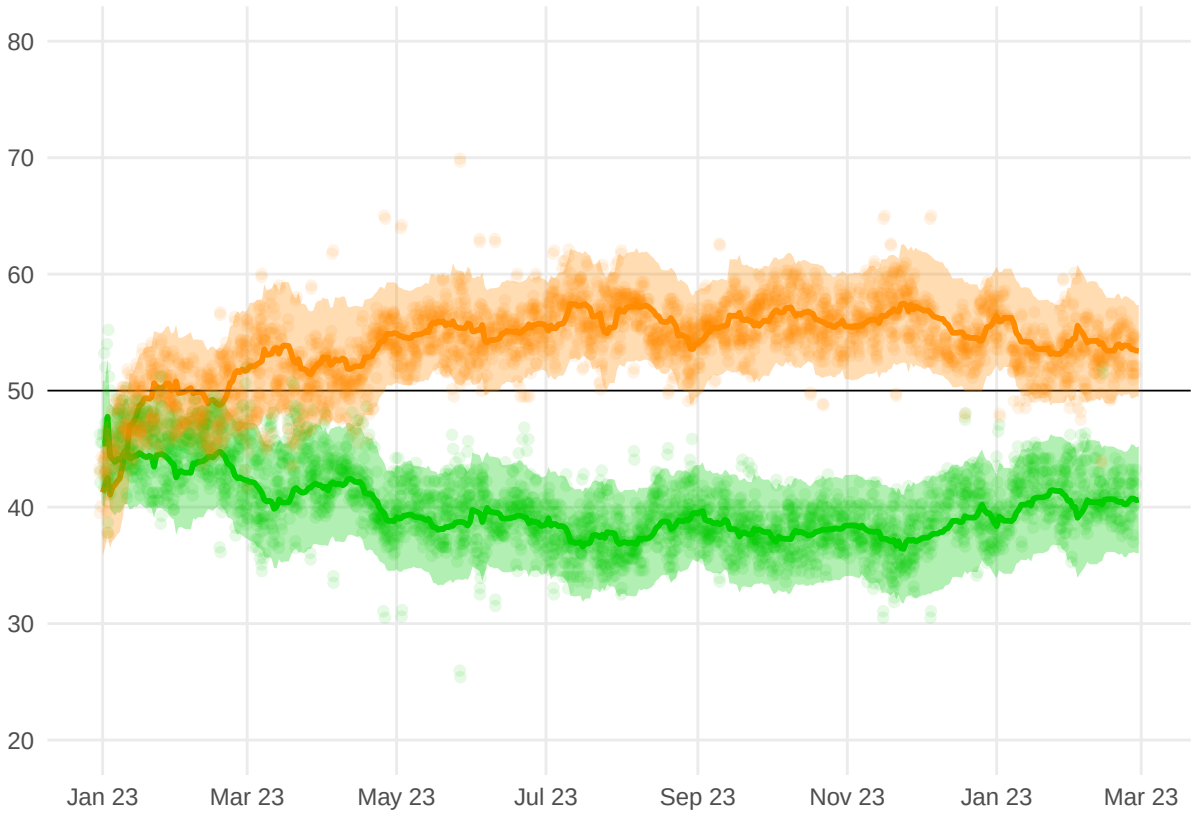
```
allpolls <- approval_topline[approval_topline$subgroup == "All polls",]
allpolls$date <- as.Date(allpolls$modeldate, "%m/%d/%Y")
approval_polllist$date <- as.Date(approval_polllist$enddate, "%m/%d/%Y")
```

Finally, create the graph

Finally, create the graph

```
approval <- ggplot(allpolls, aes(x = date)) +
  #approval line
  geom_line(aes(y = approve_estimate, group = 1), color = "green3", size = 1) +
  #disapproval line
```

```
  geom_line(aes(y = disapprove_estimate, group = 1), color = "darkorange", size = 1) +
  #solid line showing 50% mark
  geom_hline(aes(yintercept=50), color = "black", size = .3) +
  #approval ribbon
  geom_ribbon(aes(ymin = approve_lo, ymax = approve_hi), alpha = .3, fill = "green3") +
  #disapproval ribbon
  geom_ribbon(aes(ymin = disapprove_lo, ymax = disapprove_hi), alpha = .3, fill = "darkorange") +
  #approval poll data, jittered
  geom_jitter(data = approval_polllist,
              aes(y = adjusted_approve), color = "green3", alpha = .1) +
  #disapproval poll data, jittered
  geom_jitter(data = approval_polllist,
              aes(y = adjusted_disapprove), color = "darkorange", alpha = .1) +
  #set the scale for the y axis, range from 20 to 80%, show every 10%
  scale_y_continuous(breaks = seq(20, 80, 10), limits = c(20, 80)) +
  #set the scale for the x axis, starting at beginning of data, show every two months
  scale_x_date(
    breaks = seq(as.Date("2017-01-23"),
                 as.Date("2018-03-23"), by="2 months"),
    date_labels = "%b %d"
    ) +
  #ditch the axes labels
  labs(x = " ", y = "") +
  #get rid of the gray background
  theme_bw() +
  #remove tick marks, the grid border and gridlines for non-labeled lines
  theme(
    axis.ticks=element_blank(),
    panel.border=element_blank(),
    panel.grid.minor=element_blank()
  )

approval
```



Substantive Effects Simulations

First we read in the data files needed for the model.

```
dyad <- read.table("/Users/nickbichay/Desktop/ /aPLS 900/Midterm/Dyadicdata.tab.tsv", sep="\t", header="

polity <- read.csv("/Users/nickbichay/Desktop/ /aPLS 900/Midterm/p4v2016.csv", header = TRUE, stringsAs

lji <- read.table("/Users/nickbichay/Desktop/ /aPLS 900/Midterm/LJI-estimates-20140422.tab.tsv", sep="\
```

Second, we merge the datasets, keeping only the variables of interest. We also create a lagged-DV.

```
# subset only US and variables of interest
dyad <- dyad[dyad$ccode1==2,c("ccode2", "year", "absidealdiff")]

### merge in institutional model variables

# subset only variables of interest
polity <- polity[,c("ccode", "year", "polity2")]
lji <- lji[,c("ccode", "year", "LJI")]

# gen dummies
polity$democracy <- ifelse(polity$polity2 >= 6, 1, 0)
polity$autocracy <- ifelse(polity$polity2 <= -6, 1, 0)

# merge, keeping only observations in both datasets
```

```
dyad <- merge(dyad, polity, by.x=c("ccode2","year"), by.y=c("ccode","year"), all=FALSE)
dyad <- merge(dyad, lji, by.x=c("ccode2","year"), by.y=c("ccode","year"), all=FALSE)

# create lag
library(DataCombine)
dyad <- slide(dyad, Var = "absidealdiff", GroupVar = "ccode2", slideBy = -1)
```

```
##
## Remember to order dyad by ccode2 and the time variable before running.
```

```
##
## Lagging absidealdiff by 1 time units.
```

```
# rename ccode and lagged dv
colnames(dyad)[colnames(dyad) == "ccode2"] <- "ccode"
colnames(dyad)[colnames(dyad) == "absidealdiff-1"] <- "lagdv"

# drop NAs
dyad <- na.omit(dyad)
```

Here we run the model, save the coefficents, and create a vector of coefficients based on their variance

```
inst_lm <- lm(absidealdiff ~  democracy + autocracy + LJI + lagdv, data=dyad)

library(MASS)
```

```
##
## Attaching package: 'MASS'
```

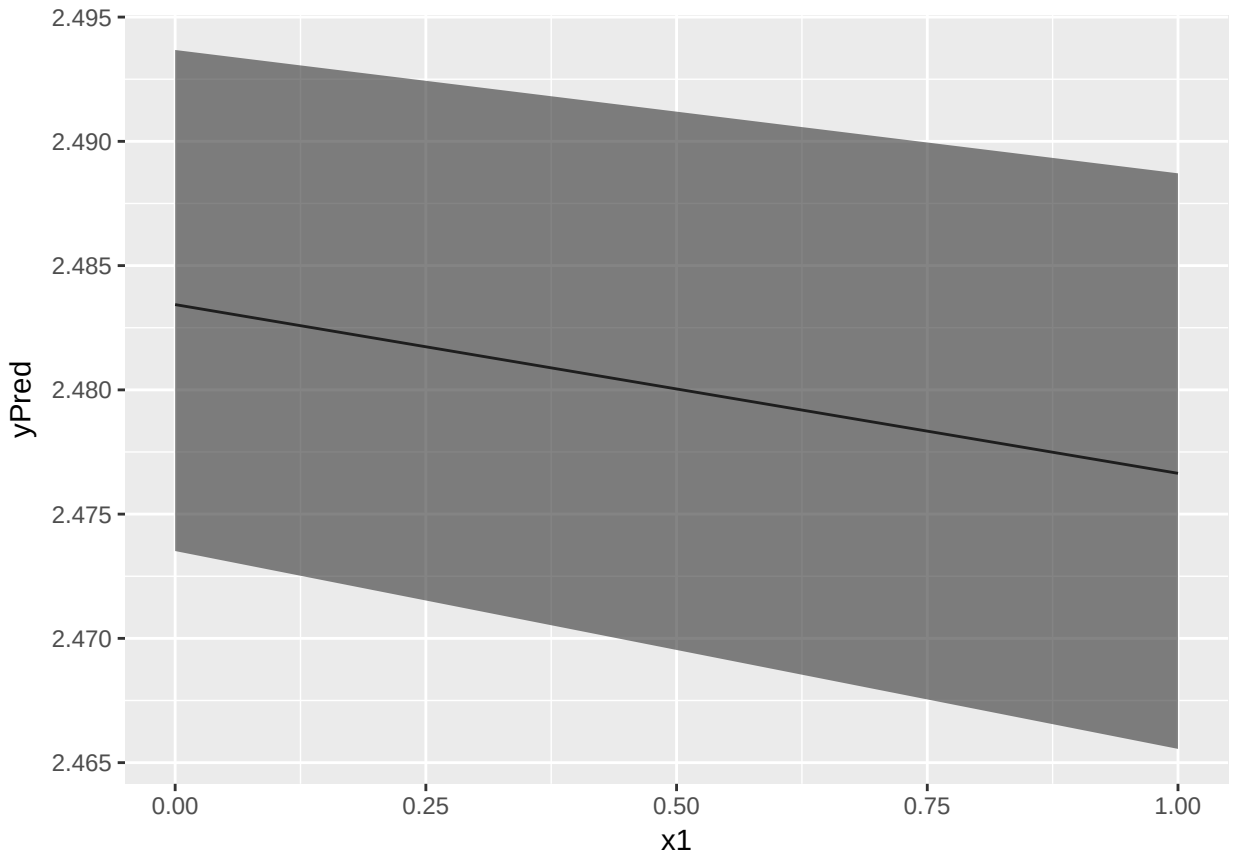
```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
betaMean <- coef(inst_lm)
betaDist <- vcov(inst_lm)
betaDraws <- mvrnorm(1000, betaMean, betaDist)
```

Next we simulate changes in each variable and create a graph to show the change as the variable of interest changes from its min to its max value, while the remaining variables stay at their mean (or, for the democracy/autocracy dummies: at zero)

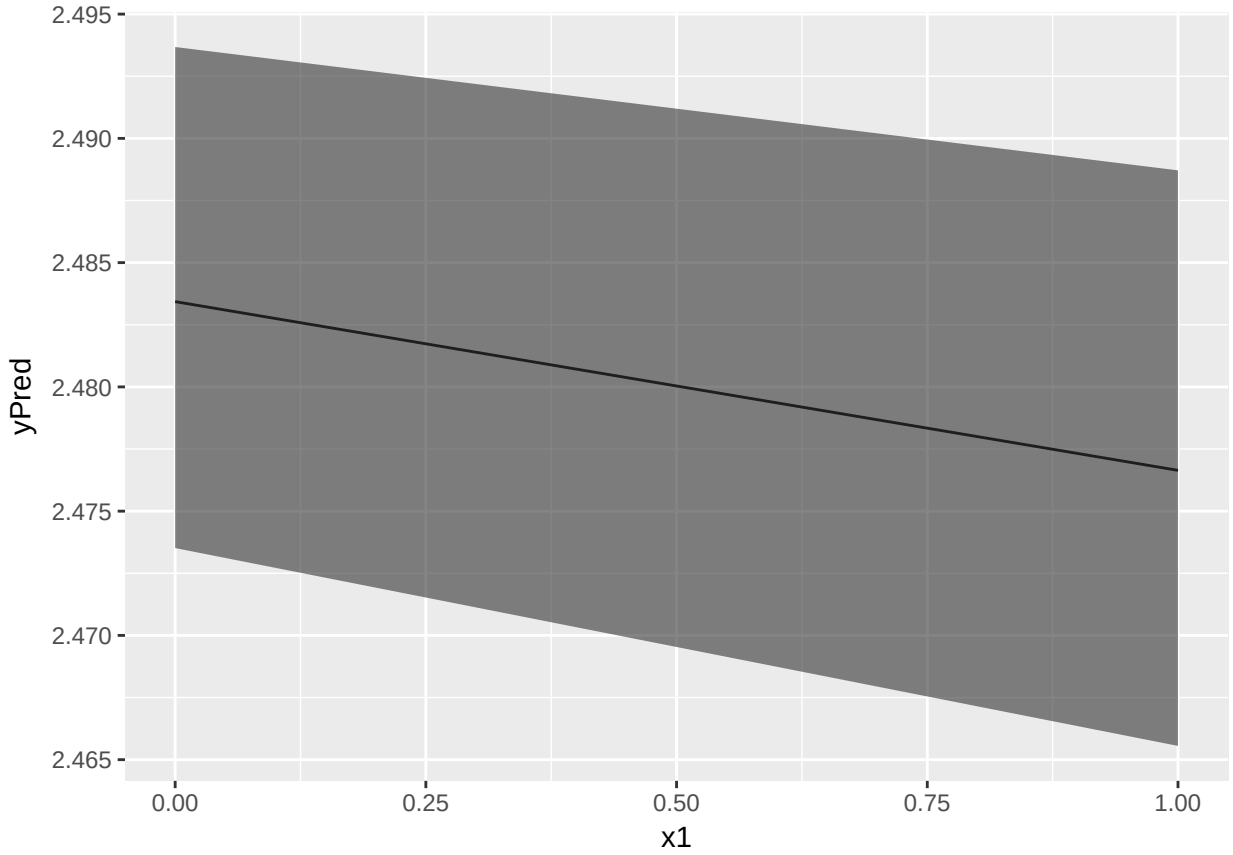
```
### sim for democracy dummy

# create matrix of hypothetical x values to simulate over
x1Values = c(0, 1)
scenario = cbind(intercept=1, democracy=x1Values, autocracy=0, LJI=mean(dyad$LJI), lagdv=mean(dyad$lagdv

# generate predictions

library(dplyr)

yPred  <-  scenario %*% betaMean

yPredUncert <- scenario %*% t(betaDraws)
yPredInt  <-  apply(yPredUncert, 1, function(x){
    quantile(x, c(0.025, 0.975), na.rm=TRUE) }) %>% t()
```

```r
simAnalysis  <- data.frame(x1=x1Values, yPred=yPred, yPredInt)
names(simAnalysis)[3:4] = c('q95lo','q95hi')

# graph

library(ggplot2)

ggplot(simAnalysis, aes(x=x1, y=yPred)) +
    geom_line() +
    geom_ribbon(aes(ymin=q95lo, ymax=q95hi), alpha=.6)
```




```r
### sim for autocracy dummy

# create matrix of hypothetical x values to simulate over
x1Values_aut = c(0, 1)
scenario_aut = cbind(intercept=1, autocracy=x1Values_aut, democracy=0, LJI=mean(dyad$LJI), lagdv=mean(d

# generate predictions

yPred_aut  <-  scenario_aut %*% betaMean

yPredUncert_aut <- scenario_aut %*% t(betaDraws)
yPredInt_aut  <-  apply(yPredUncert_aut, 1, function(x){
    quantile(x, c(0.025, 0.975), na.rm=TRUE) }) %>% t()

simAnalysis_aut  <- data.frame(x1=x1Values_aut, yPred=yPred_aut, yPredInt_aut)
names(simAnalysis_aut)[3:4] = c('q95lo','q95hi')
```

```r
# graph

ggplot(simAnalysis_aut, aes(x=x1, y=yPred)) +
    geom_line() +
    geom_ribbon(aes(ymin=q95lo, ymax=q95hi), alpha=.6)
```




```r
### sim for lji

# create matrix of hypothetical x values to simulate over
x1Values_lji = seq(min(dyad$LJI), max(dyad$LJI), length.out=50)
scenario_lji = cbind(intercept=1, autocracy=0, democracy=0, LJI=x1Values_lji, lagdv=mean(dyad$lagdv) )

# generate predictions

yPred_lji  <-  scenario_lji %*% betaMean

yPredUncert_lji <- scenario_lji %*% t(betaDraws)
yPredInt_lji  <-  apply(yPredUncert_lji, 1, function(x){
    quantile(x, c(0.025, 0.975), na.rm=TRUE) }) %>% t()

simAnalysis_lji  <- data.frame(x1=x1Values_lji, yPred=yPred_lji, yPredInt_lji)
names(simAnalysis_lji)[3:4] = c('q95lo','q95hi')

# graph

ggplot(simAnalysis_lji, aes(x=x1, y=yPred)) +
```

```
    geom_line() +
    geom_ribbon(aes(ymin=q95lo, ymax=q95hi), alpha=.6)
```