Tutorial 11: Imputation of Missing Data, Regression Diagnostics, and Simulations

Jan Vogler (jan.vogler@duke.edu)

November 6, 2015

Today's Agenda

1. Imputation of missing data with Amelia
2. Diagnostic techniques: CPR plots and the Ramsey RESET test
3. Model simulations
4. Using model simulations to estimate substantive effects

1. Imputation of missing data with Amelia

We often deal with missing data, meaning that some variable values for some observations are missing. This is problematic for many reasons and data that is missing systematically and not arbitrarily can challenge the validity of our regression results. Insofar, the accuracy of our model profits from the completeness of data.

Imputation allows us to make statistical inferences about missing data values. Our imputed values are based on the data that we have for other observations and for other variables of the unit that has missing values. The most sophisticated imputation techniques compute multiple alternative datasets based on the existing dataset and then make an inference based on those datasets to estimate the final imputed value for the missing values of any given variable (therefore "multiple imputation").

Let us first look at our LDC dataset and how many missing values it has for different variables.

```r
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/")
library(foreign)
LDC = read.dta("LDC_IO_replication.dta")
# summary(LDC)
```

As you can see, we have quite a few NAs in this dataset. Let's impute some of the missing values. For the imputation of missing data, we will use a package by Gary King et al. called *Amelia*. (Interestingly, the guide to the package uses the same dataset that we have been using throughout the semester to illustrate the imputation of missing data.) For the tutorial, we will demonstrate the imputation of missing data with only a small subset of three countries with a small number of variables (due to time constraints).

```r
LDCs = subset(LDC, ctylabel == "SouthAfrica" | ctylabel == "Turkey" | ctylabel ==
    "Indonesia")

# Let us reduce our dataset to some variables that we might be most
# interested in

keep = c("ctylabel", "date", "newtar", "fdignp", "gdp_pc_95d", "polityiv_update2",
    "usheg", "lnpop")
LDCs = LDCs[, keep]
```

Two things that you should be aware of when you use Amelia to impute data yourself:

1. The imputation of missing values in large datasets can take a lot of time.
2. Amelia has problems with variables that are perfectly correlated to other variables.

The content below is based on the Amelia guide. A link is provided at the end.

Now let us do the imputation. We will create only 5 imputed datasets (m=5).

install.packages("Amelia")

```
library(Amelia)
```

```
## Loading required package: Rcpp
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.3, built: 2014-11-14)
## ## Copyright (C) 2005-2015 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
a.out <- amelia(LDCs, m = 5, ts = "date", cs = "ctylabel")
```

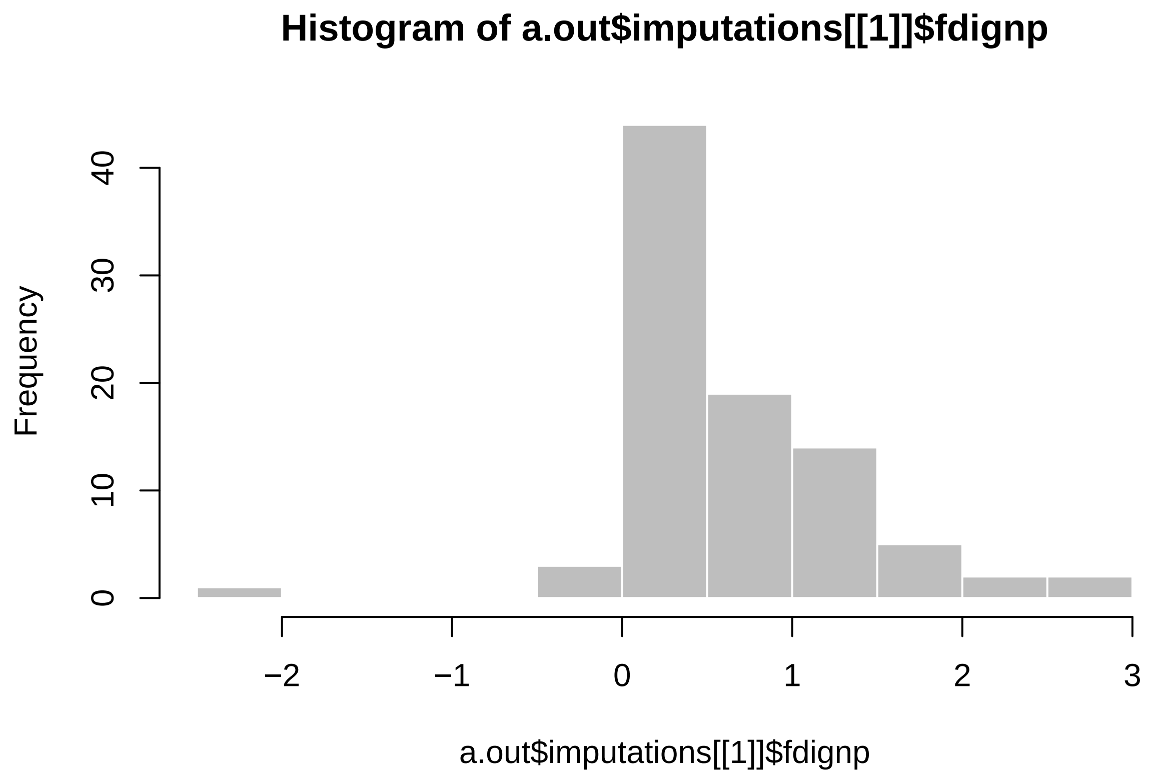
```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106 107 108 109 110 111 112
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
##  81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
##  101 102 103 104 105 106
```

```
##
## -- Imputation 5 --
##
##   1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
##  41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60
##  61
```

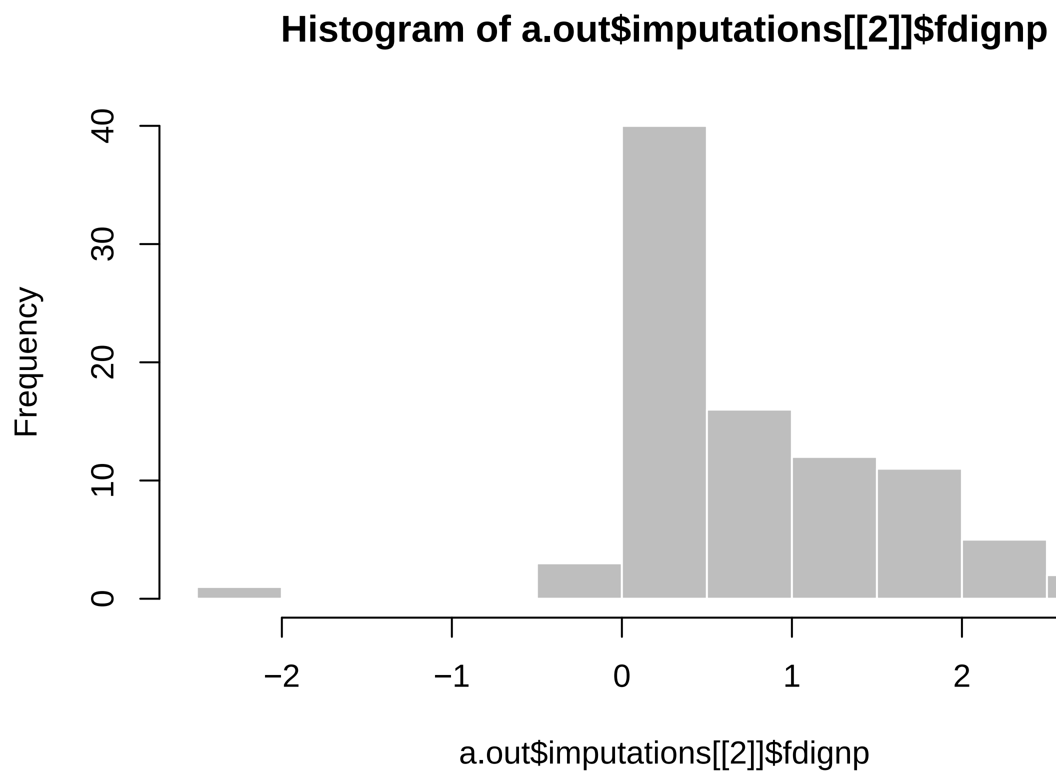
```
# It is very important to include ts as the time variable and cs as the unit
# variable. Otherwise Amelia would treat all observations as independent
```

Let us have a look at the imputed values of the first three datasets that were generated.

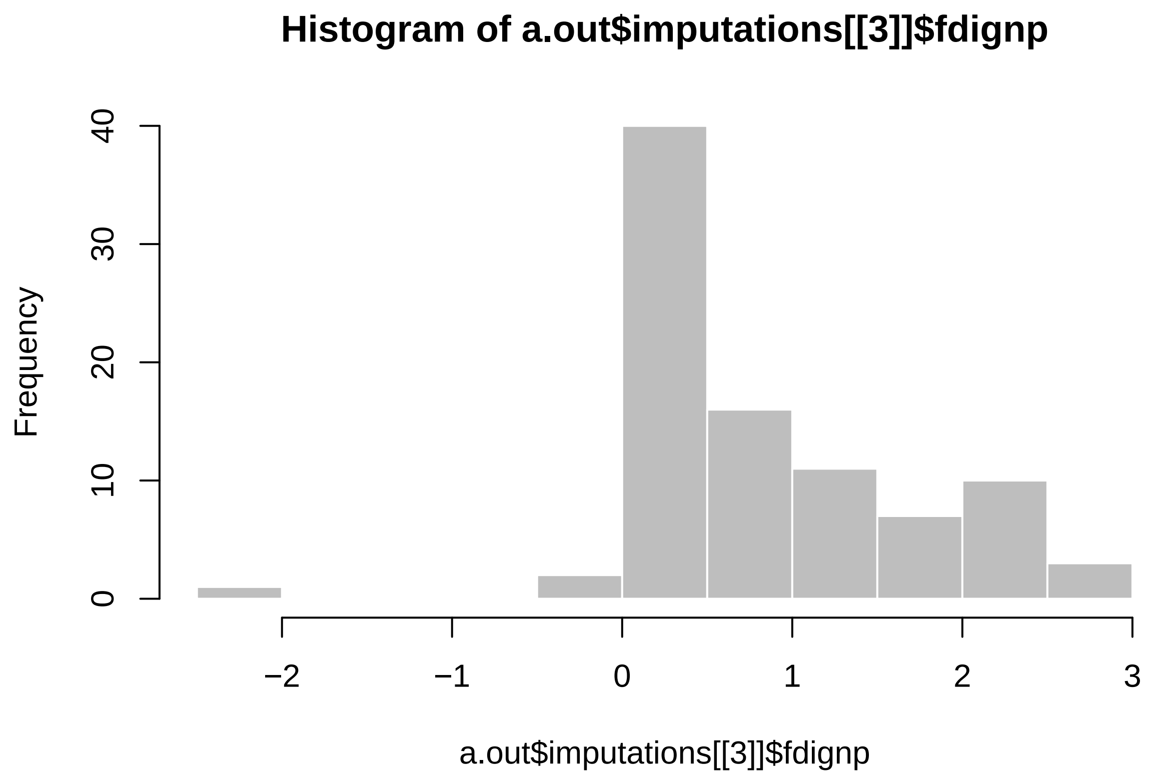
```
hist(a.out$imputations[[1]]$fdignp, col = "grey", border = "white")
```



```
hist(a.out$imputations[[2]]$fdignp, col = "grey", border = "white")
```



```r
hist(a.out$imputations[[3]]$fdignp, col = "grey", border = "white")
```



We can save those datasets either as a single large dataset or as multiple datasets containing each imputation.

```r
setwd("C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/w11/")
save(a.out, file = "imputations.RData")
write.amelia(obj = a.out, file.stem = "LDCs", format = "csv")
```

We can now run an analysis with our imputed dataset. The Amelia package is integrated with the Zelig package (both are by Gary King), so we will use this package to estimate a new regression with the imputed data.

install.packages("Zelig")

library(Zelig)

```r
z.out.imp <- zelig(polityiv_update2 ~ gdp_pc_95d + fdignp, data = a.out$imputations,
    model = "ls")
```

```
##
##
##  How to cite this model in Zelig:
##   Kosuke Imai, Gary King, and Olivia Lau. 2015.
##   "ls: Least Squares Regression for Continuous Dependent Variables"
##   in Kosuke Imai, Gary King, and Olivia Lau, "Zelig: Everyone's Statistical Software,"
##   http://gking.harvard.edu/zelig
##
```

```r
summary(z.out.imp)
```

```
##
##   Model: ls
##   Number of multiply imputed data sets: 5
##
## Combined results:
##
## Call:
## lm(formula = formula, weights = weights, model = F, data = data)
##
## Coefficients:
##                    Value   Std. Error    t-stat      p-value
## (Intercept) -4.588669508 0.8765385666 -5.234989 1.967630e-07
## gdp_pc_95d   0.003596105 0.0003784857  9.501296 6.164615e-11
## fdignp      -3.193129348 0.6026081315 -5.298849 1.204105e-07
##
## For combined results from datasets i to j, use summary(x, subset = i:j).
## For separate results, use print(summary(x), subset = i:j).
```

Amelia has a graphical interface that *might* make it easier for you to use it. When I tried to use the graphical interface, however, my R session crashed multiple times and I had to restart it completely. So be cautious and save your work before you use the following command.

```r
# AmeliaView()
```

For more information on how to use the package (for your own research) see the very helpful introduction:

https://cran.r-project.org/web/packages/Amelia/vignettes/amelia.pdf

2. Diagnostic techniques: CPR plots and the Ramsey RESET test

Diagnostic techniques are important tools to assess the accuracy and robustness of our regression. In the last tutorial, outlier plots were introduced. Those plots help us to find out whether or not our results are driven by just a few units, which is bad.

Let us look at a regression that is related to the model that we have used in tutorial 5 on interpretation. (This is a shortened version though)

```r
lm_newtar = lm(newtar ~ l1polity + l1fdi + l1usheg + factor(ctylabel) - 1, data = LDC)
# summary(lm_newtar)
```

The car package ("Companion to Applied Regression") includes commands for many diagnostic techniques. You should always apply these techniques to your own regression results to make sure that they are robust.

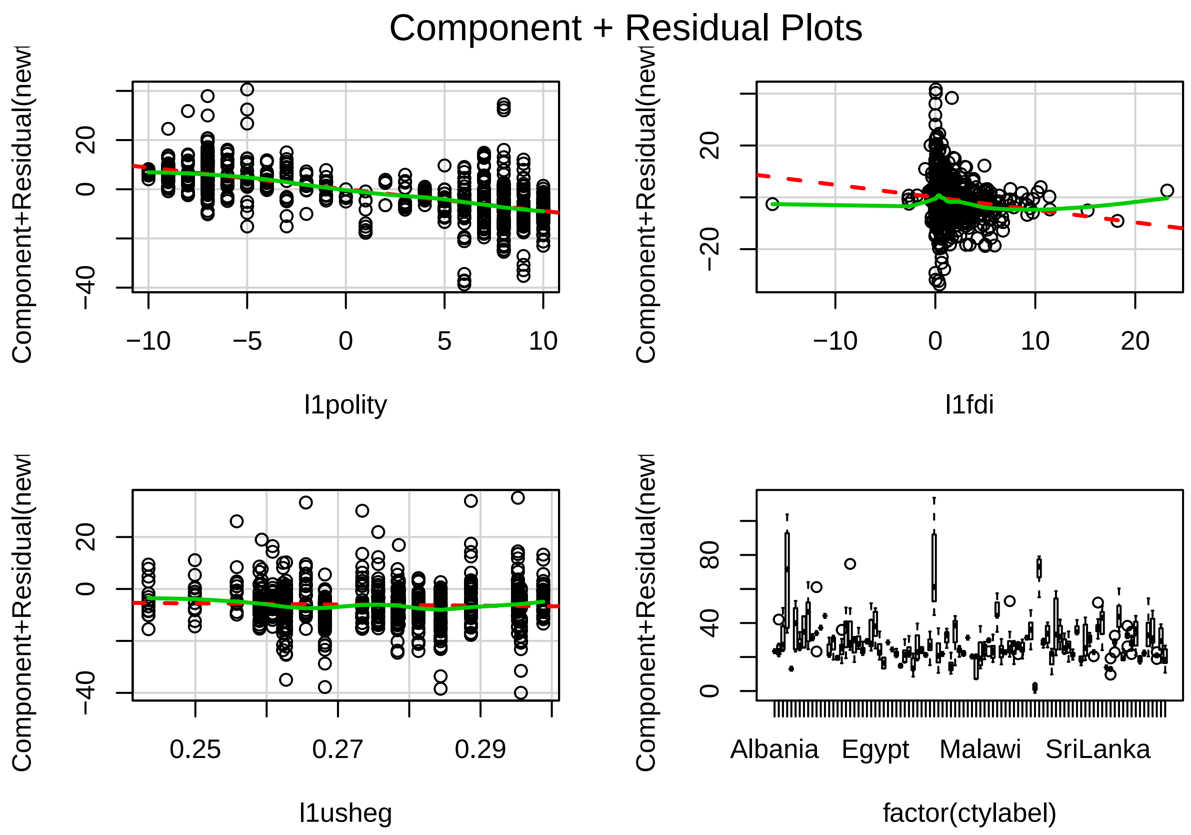
install.packages("car")

Let us first focus on the component plus residual plots that were introduced in the lecture.

```r
library(car)
```

```
##
## Attaching package: 'car'
##
## The following object is masked from 'package:boot':
##
##     logit
```

```
crPlots(lm_newtar)
```



What can we tell from these regressions? What can we say about each of our independent variables?

Let us move on to the Ramsey RESET Test for Functional Form Misspecification. For this test we use the “lmtest” package.

install.packages(“lmtest”)

```
library(lmtest)
```

```
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
resettest(lm_newtar, power = 2:3, type = c("regressor"), data = LDC)
```

```
##
##  RESET test
##
## data:  lm_newtar
## RESET = 3.4925, df1 = 6, df2 = 617, p-value = 0.0021
```

How would we interpret this output?

Many more diagnostic techniques can be found here:

http://www.statmethods.net/stats/rdiagnostics.html

3. Model simulations

Credit to Professor Chris Johnston. The code for the simulation approach introduced here is from his class.

We will look at new data that we have not used previously: data on attitudes toward the United States Supreme Court. This is a topic that students of political behavior and identities might be interested in.

```r
courtdata <- read.table("courtdata.txt", header = TRUE)
# summary(courtdata)
```

What is the meaning of the variables in the data set?

Independent Variables

college = dummy for college education (1 = college educated)

ideo = 5-point ideological self-identification, ranging from "very liberal" (1) to "very conservative" (5)

soph = 10-item political knowledge scale (0 = low, 10 = high)

black, hispanic = dummies for respective categories (1 = hispanic)

income = 15-point household income category scale (1 = low, 15 = high)

ruling = dummy indicating respondent correctly identified the Affordable Care Act ruling (1)

male = dummy indicating male gender (1 = male)

Dependent Variable

We are interested in attitudes of US citizens toward the Supreme Court. For this purpose we look at the variable **sclaw** (standing for "Supreme Court Law"). The variable was based on the following statement:

"The Supreme Court should be allowed to throw out any law it deems unconstitutional"

The variable ranges from "strongly agree" to "strongly disagree".

Let us estimate a linear model.

```r
model1 <- lm(sclaw ~ ruling + ideo + soph + age + male + black + hisp + college +
    income, data = courtdata)
summary(model1)
```

```
##
## Call:
## lm(formula = sclaw ~ ruling + ideo + soph + age + male + black +
##     hisp + college + income, data = courtdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9065 -1.1406 -0.3534  0.8163  3.2797
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.919930   0.240242  12.154  < 2e-16 ***
## ruling      -0.133350   0.135592  -0.983  0.32567
## ideo        -0.135789   0.045475  -2.986  0.00291 **
## soph        -0.036333   0.020474  -1.775  0.07633 .
## age          0.004780   0.003167   1.509  0.13161
## male        -0.165054   0.100291  -1.646  0.10019
## black        0.071503   0.153606   0.465  0.64170
## hisp        -0.123167   0.170601  -0.722  0.47052
## college     -0.039149   0.105591  -0.371  0.71091
## income       0.009154   0.012308   0.744  0.45721
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.34 on 835 degrees of freedom
## Multiple R-squared:  0.02962,    Adjusted R-squared:  0.01916
## F-statistic: 2.832 on 9 and 835 DF,  p-value: 0.002746
```

When we estimate a linear model, all our coefficients are assumed to be normally distributed random variables with a mean and a variance that depends on the data. We can make use of this fact by simulating different versions of the model in which the coefficients are normally distributed around the original model.

For the simulations we need the package “arm”. Please make sure to install it via the following command:

install.packages(“arm”)

```r
library(arm)
```

```
## Loading required package: Matrix
## Loading required package: lme4
##
## arm (Version 1.8-6, built: 2015-7-7)
##
## Working directory is C:/Users/Jan/OneDrive/Documents/GitHub/ps630_lab/W11
##
##
## Attaching package: 'arm'
##
## The following object is masked from 'package:car':
```

```
##
##     logit
##
## The following object is masked from 'package:Zelig':
##
##     sim
##
## The following object is masked from 'package:boot':
##
##     logit
```

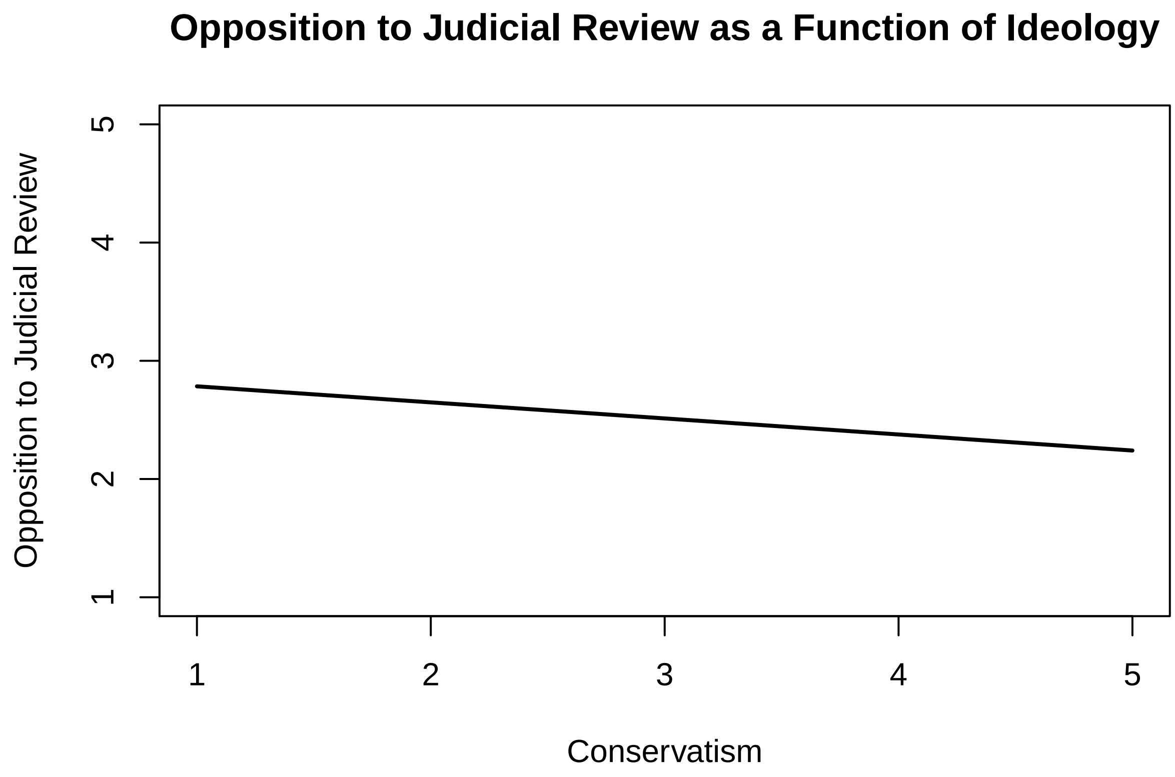
```
model1.sims <- sim(model1, n.sims = 1000)
```

How did we create these simulations? How could that be useful to us?

4. Using model simulations to estimate substantive effects

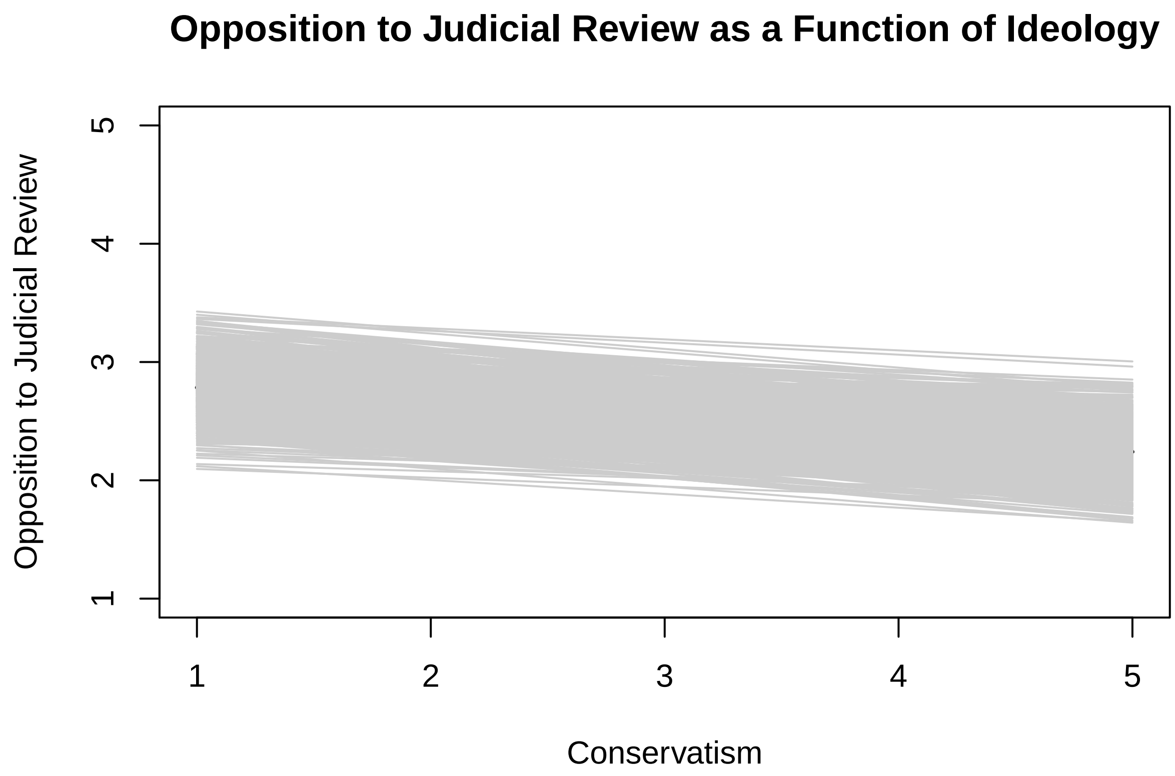
The following plot is generated by our knowledge about the regression. We access the first coefficient (the intercept) and the third coefficient (ideology). We let ideology vary from 1 to 5. If we plug in the right formula, then we will get the predicted values when all other variables are at the value 0. (This is similar to our predictions from previous tutorials, where we held all variables at their mean.)

```
curve(coef(model1)[1] + coef(model1)[3] * x, from = 1, to = 5, ylim = c(1, 5),
    xlab = "Conservatism", ylab = "Opposition to Judicial Review", main = "Opposition to Judicial Revie
    lwd = 2)
```



Now let's look instead at the lines that are the result of the 1000 simulations that we created above.

```
curve(coef(model1)[1] + coef(model1)[3] * x, from = 1, to = 5, ylim = c(1, 5),
    xlab = "Conservatism", ylab = "Opposition to Judicial Review", main = "Opposition to Judicial Revie
    lwd = 2)
for (i in 1:1000) {
    curve(coef(model1.sims)[i, 1] + coef(model1.sims)[i, 3] * x, add = TRUE,
        col = "gray80")
}
```

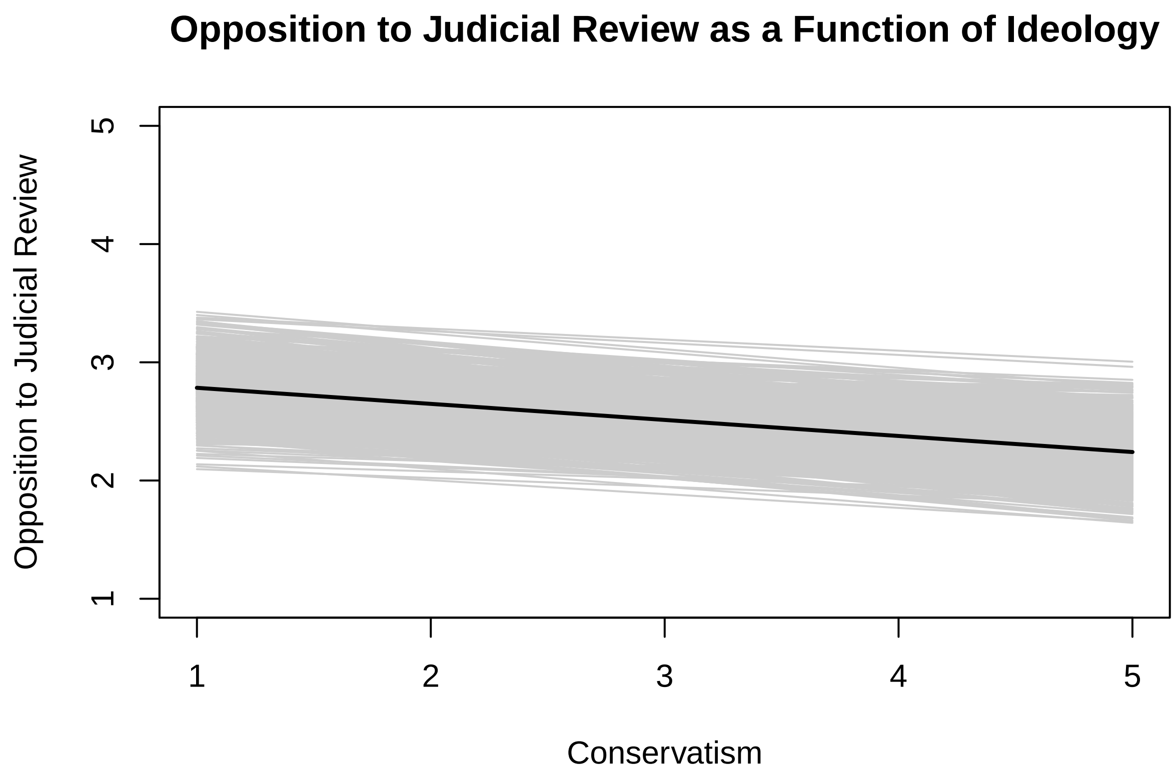


What we can see in this plot is that the simulated coefficients are very similar to the one that we already had. In fact, the simulation predictions are approximately normally distributed around our original regression line.

Let's run the original command one more time, so we get our regression line back on top of everything else.

```
curve(coef(model1)[1] + coef(model1)[3] * x, from = 1, to = 5, ylim = c(1, 5),
    xlab = "Conservatism", ylab = "Opposition to Judicial Review", main = "Opposition to Judicial Revie
    lwd = 2)
for (i in 1:1000) {
    curve(coef(model1.sims)[i, 1] + coef(model1.sims)[i, 3] * x, add = TRUE,
        col = "gray80")
}

curve(coef(model1)[1] + coef(model1)[3] * x, col = "black", lwd = 2, add = TRUE)
```



Let us now utilize our model simulations. They can help us to make an average predictive comparison. We use our model simulations in combination with draws from the data to create a so-called "average predictive comparison". These help us to evaluate the substantive impact that a variable has in comparison with other variables. Our results will include confidence intervals.

In order to do this, we first create an array that we fill with data. We have 1000 simulations of the model and X observations or data points. We will apply each of those 1000 simulations to all our data points to estimate the average predicted effect.

```r
d.ideo <- array(NA, c(1000, length(courtdata$sclaw)))
m.ideo <- array(NA, 1000)
```

Now we run a for loop. In this for loop we go through each of the 1000 simulations of the model that we have generated based on the variance-covariance matrix. An in each of the 1000 iterations, we also go through the data in its entirety. In each of these iterations, we subtract the effect of holding our ideology at the minimum from holding it at the maximum. This will give us an idea of what the effect is across many possible models and all of the empirical data that is available to us - a so-called "average prediction" of the effect.

```r
# Remember our model was: sclaw ~ ruling + ideo + soph + age + male + black
# + hisp + college + income

summary(courtdata$ideo)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   3.000   3.128   4.000   5.000
```

```
for (i in 1:1000) {
    d.ideo[i, ] <- (coef(model1.sims)[i, 1] + coef(model1.sims)[i, 2] * courtdata$ruling +
        coef(model1.sims)[i, 3] * 5 + coef(model1.sims)[i, 4] * courtdata$soph +
        coef(model1.sims)[i, 5] * courtdata$age + coef(model1.sims)[i, 6] *
        courtdata$male + coef(model1.sims)[i, 7] * courtdata$black + coef(model1.sims)[i,
        8] * courtdata$hisp + coef(model1.sims)[i, 9] * courtdata$college +
        coef(model1.sims)[i, 10] * courtdata$income) - (coef(model1.sims)[i,
        1] + coef(model1.sims)[i, 2] * courtdata$ruling + coef(model1.sims)[i,
        3] * 0 + coef(model1.sims)[i, 4] * courtdata$soph + coef(model1.sims)[i,
        5] * courtdata$age + coef(model1.sims)[i, 6] * courtdata$male + coef(model1.sims)[i,
        7] * courtdata$black + coef(model1.sims)[i, 8] * courtdata$hisp + coef(model1.sims)[i,
        9] * courtdata$college + coef(model1.sims)[i, 10] * courtdata$income)
    m.ideo[i] <- mean(d.ideo[i, ])
}

mean(m.ideo)
```

```
## [1] -0.6724957
```

```
sd(m.ideo)
```

```
## [1] 0.2321447
```

```
quantile(m.ideo, probs = c(0.025, 0.16, 0.84, 0.975))
```

```
##       2.5%        16%        84%      97.5%
## -1.1157230 -0.9003192 -0.4301574 -0.2063324
```

Let's compare this to the effect that sophistication has. We need to also estimate the average predictive effects for sophistication. Please note that this time we let ideology vary with the data. What we keep constant here is the sophistication variable at its minimum and maximum to assess the average effect that it has on attitudes toward the Supreme Court.

```
summary(courtdata$soph)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   4.000   7.000   6.724   9.000  10.000
```

```
d.soph <- array(NA, c(1000, length(courtdata$sclaw)))
m.soph <- array(NA, 1000)

for (i in 1:1000) {
    d.soph[i, ] <- ((coef(model1.sims)[i, 1] + coef(model1.sims)[i, 2] * courtdata$ruling +
        coef(model1.sims)[i, 3] * courtdata$ideo + coef(model1.sims)[i, 4] *
        10 + coef(model1.sims)[i, 5] * courtdata$age + coef(model1.sims)[i,
        6] * courtdata$male + coef(model1.sims)[i, 7] * courtdata$black + coef(model1.sims)[i,
        8] * courtdata$hisp + coef(model1.sims)[i, 9] * courtdata$college +
        coef(model1.sims)[i, 10] * courtdata$income) - (coef(model1.sims)[i,
        1] + coef(model1.sims)[i, 2] * courtdata$ruling + coef(model1.sims)[i,
        3] * courtdata$ideo + coef(model1.sims)[i, 4] * 0 + coef(model1.sims)[i,
        5] * courtdata$age + coef(model1.sims)[i, 6] * courtdata$male + coef(model1.sims)[i,
        7] * courtdata$black + coef(model1.sims)[i, 8] * courtdata$hisp + coef(model1.sims)[i,
        9] * courtdata$college + coef(model1.sims)[i, 10] * courtdata$income))
    m.soph[i] <- mean(d.soph[i, ])
}

mean(m.soph)
```

```
## [1] -0.3674008
```

```
sd(m.soph)
```

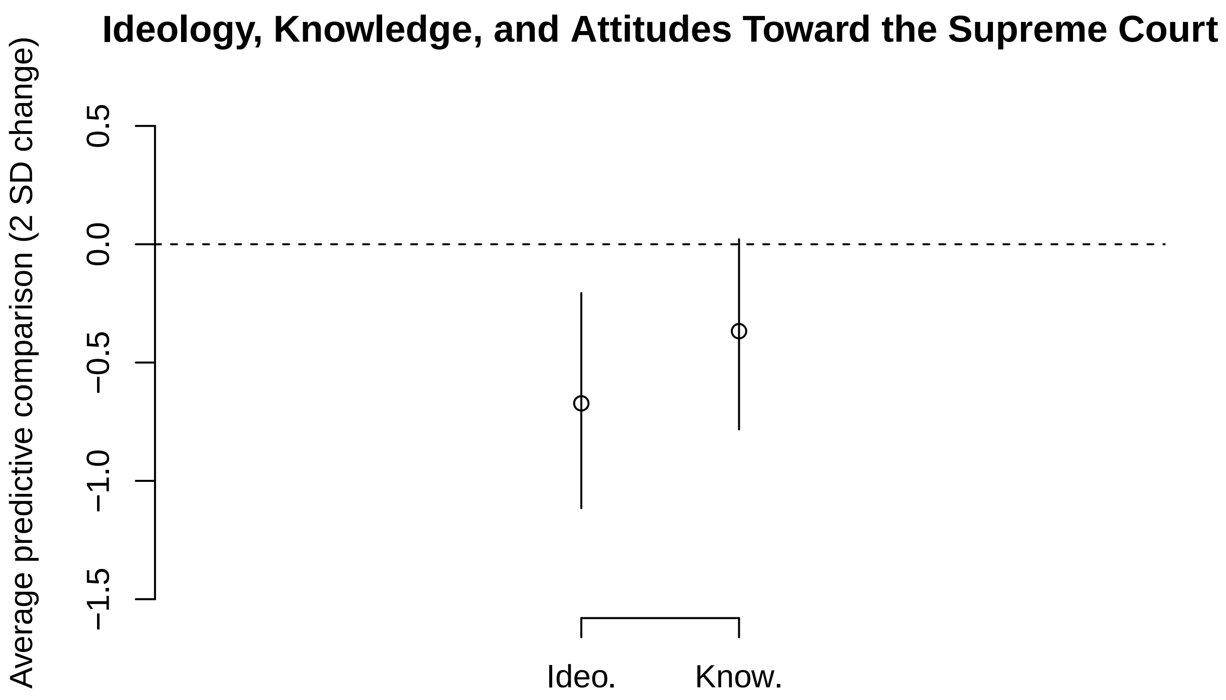
```
## [1] 0.2065123
```

```
quantile(m.soph, probs = c(0.025, 0.16, 0.84, 0.975))
```

```
##        2.5%         16%         84%       97.5%
## -0.78288912 -0.57989463 -0.16255687  0.02126657
```

Let us plot these two in comparison.

```
plot(1:2, c(mean(m.ideo), mean(m.soph)), type = "p", ylim = c(-1.5, 0.5), xlab = "",
    main = "Ideology, Knowledge, and Attitudes Toward the Supreme Court", ylab = "Average predictive co
    asp = 1.5, axes = FALSE)
axis(1, at = c(1, 2), labels = c("Ideo.", "Know."))
axis(2, at = c(-1.5, -1, -0.5, 0, 0.5))
abline(h = 0, lty = 2)
segments(1, quantile(m.ideo, probs = c(0.025)), 1, quantile(m.ideo, probs = c(0.975)))
segments(2, quantile(m.soph, probs = c(0.025)), 2, quantile(m.soph, probs = c(0.975)))
```



Which variable has the greater substantive effect? What can we say about the average influence of the knowledge variable?