Precept Five: Model Diagnostics for Binary Outcome Models and Ordered Probit

Rebecca Johnson

March 8th, 2017

Outline

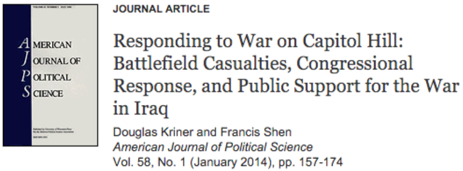
- Replication check-in: questions; advice on constructive feedback
- Follow-up topic for binary outcome models (focus of Pset 4, due March 15th with optional one-week extension to March 22)
  - **Model diagnostics**
    - Separation plots
    - *k-fold* cross validation
- Ordered probit model
  - Conceptual review of latent variable interpretation
  - Derive and optimize log likelihood
  - Practice with `Zelig` to estimate model and if time, automatic version of simulation for QOI

Memo 1 due yesterday; okay? problems/questions?

Tips for constructive feedback

1. Summarize in your own words
2. For a given comment, ask yourself: is there an actionable directive from this? If answer is no, try to reformulate comment so that it contains an actionable directive or flag it as something that they can't address given data/etc., but should mention as a limitation
3. Think both about tips you have for replication (since it seems like most groups are still partway through) and advice for extensions you'd be interested in seeing

Working example throughout precept



JOURNAL ARTICLE

Responding to War on Capitol Hill: Battlefield Casualties, Congressional Response, and Public Support for the War in Iraq

Douglas Kriner and Francis Shen

American Journal of Political Science

Vol. 58, No. 1 (January 2014), pp. 157-174

My House member during this time (Mark Kirk) at my high school (not my picture!)

Many questions in the paper with different models corresponding to DGP for different outcomes

- How does the number of casualties in month $t-1$ predict the *count* of a House member's speeches critical of the war in month t?
  - *Outcome variable*: "monthly sum total of the number of critical House floor speeches identified from the *Congressional record*"
  - *Model*: negative binomial regression
    - *Refresher from lecture:* similar to Poisson in that it models counts but allows for overdispersion by modifying the variance (in lecture notes, Brandon refers to this as NB2 and it can be fit using `MASS` `glm.nb`)
    - **Poisson:**

$$E(Speeches_i|X_i) = \mu_i; V(Speeches_i|X_i) = \mu_i$$

    - **Negative binomial**: for γ (sometimes called dispersion parameter), note that as $\gamma \to \infty$, variance $\to$ mean and negative binomial $\to$ Poisson

$$E(Speeches_i|X_i) = \mu_i; V(Speeches_i|X_i) = \mu_i + \frac{\mu_i^2}{\gamma}$$

Many questions in the paper with different models corresponding to DGP for different outcomes

- Authors then switch from looking at congressman speeches critical of the war as a count *outcome* to congressman speeches as a *predictor* of war-related opinions for constituents from that congressman's district
- Two outcomes with different models:
  1. **Probit** (use to explore model dx): Was the Iraq War a mistake (1 = yes; 0 = no)?
  2. **Ordered probit** (use to review concepts and finding QOI): Should the U.S. stay the course in Iraq?
     - **1 = withdraw immediately**
     - **2 = withdraw in next year**
     - **3 = stay as long as necessary**
     - **4 = increase the number of troops**

Model diagnostics

Reported results in paper

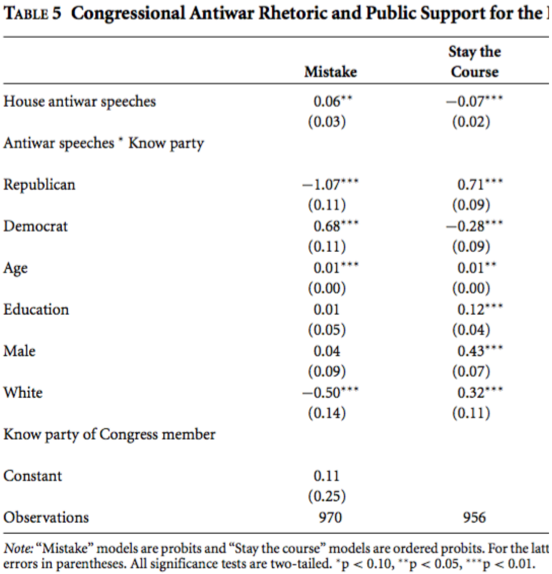


TABLE 5 Congressional Antiwar Rhetoric and Public Support for the

	Mistake	Stay the Course
House antiwar speeches	0.06^{**}	-0.07^{***}
	(0.03)	(0.02)
Antiwar speeches * Know party		
Republican	-1.07^{***}	0.71^{***}
	(0.11)	(0.09)
Democrat	0.68^{***}	-0.28^{***}
	(0.11)	(0.09)
Age	0.01^{***}	0.01^{**}
	(0.00)	(0.00)
Education	0.01	0.12^{***}
	(0.05)	(0.04)
Male	0.04	0.43^{***}
	(0.09)	(0.07)
White	-0.50^{***}	0.32^{***}
	(0.14)	(0.11)
Know party of Congress member		
Constant	0.11	
	(0.25)	
Observations	970	956

Note: "Mistake" models are probits and "Stay the course" models are ordered probits. For the latt
errors in parentheses. All significance tests are two-tailed. $^{*}p < 0.10$, $^{**}p < 0.05$, $^{***}p < 0.01$.

Purpose of model diagnostics

- How well does the model predict the outcome (in this case, correctly classify $1 =$ war was a mistake; $0 =$ not a mistake)?
- *Underlying motivation*: downsides of model complexity (slide 77 lecture), complex models have low bias, high variance and fit the data well but can be 'overfit' to the data's peculiarities and do poorly at predicting outcomes in unseen data
- Note that distinct from robustness checks for identification that authors perform- model that fits well might be a good predictive model but no guarantee it's a causal model:
    - *Focus of model diagnostics we'll review*: how well the fitted model *classifies* or *predicts* the outcomes of $1 =$ yes mistake; $0 =$ not a mistake (which we hope generalizes to respondents out of sample)
    - *Focus of authors' identification-focused robustness checks* (e.g., IV analysis using seniority): are results driven by unobserved features of a district correlated with both congressman position-taking and constituent opinion

Technique one: separation plot (Greenhill, Ward, and Sacks, 2011)

- **Motivation**: in the *post-Tufte* era of statistics, researchers and audiences want not only numerical measures of binary model fit like the pseudo-R^2 $(1 - \frac{\ell(\hat{\beta_{MLE}})}{\ell(\bar{y})})$, but also want intuitive visualizations of model fit
- **Idea**: show how concentrated non-events (0's; not mistake) are in lower range of predicted values v. how concentrated events (1's; mistake) are in higher range of predicted values
  - *Model with no predictive power*: even distribution of events and non-events within range of predicted values
  - *Model with perfect predictive power*: complete partition of events and non-events

Technique one: implementing in R

In your .rmd, do the following:

1. Fit the following probit model that relates an individual's views on whether or not war was a mistake to negative speeches by his or her congressman, controlling for covariates
   - Can use `glm` with `family = binomial(link = "probit")` and `fitted` to get fitted values
   - Model is given by following code in STATA .do file: `probit iraqamistake totantiwarthroughjune16 gop3 dem3 age education4 male white`
2. Use dplyr to arrange dataset from lowest to highest fitted value; use `head` to view. How do these correspond to the respondent's observed value for the iraq war variable?
3. Create a separation plot- either your own or by installing/using the `separationplot` package in R

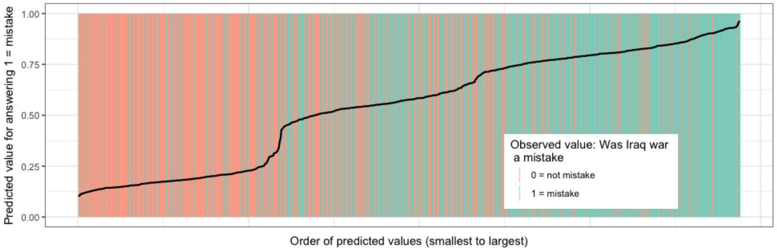
Technique one: our implementation

```
##arrange in ascending order by predicted value
probit_df_order <- probit_df_all %>%
                  arrange(probit_pred_out) %>%
                  select(probit_outcome_predicted,
                         probit_outcome_observed)
```

Technique one: our implementation

```
##plot results
ggplot(data=probit_df_order) +
  geom_rect(aes(xmin = 0,
    xmax = seq(length.out = length(probit_outcome_observed)),
      ymin = 0, ymax = 1),
    fill = "#FEE8C8") +
  geom_linerange(aes(color = factor(probit_outcome_observed,
            levels = c(0, 1),
            labels = c("0 = not mistake",
          "1 = mistake")),
    ymin = 0, ymax = 1, x = seq(length.out =
    length(probit_outcome_observed))),
    alpha = 0.4) +
  labs(color = "Observed value: Was Iraq war \n a mistake")  +
  geom_line(aes(y = probit_outcome_predicted,
                x = seq(length.out =
                length(probit_outcome_observed))), lwd = 0.8) +
  theme_bw() + xlab("Order of predicted values (smallest to largest)") +
  ylab("Predicted value for answering 1 = mistake") +
  theme(axis.text.x=element_blank(), axis.ticks.x=element_blank(),
  legend.position = c(0.75, 0.25))
```

Technique one: interpretation

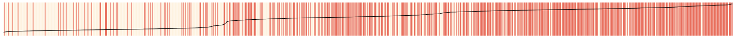


Highlights fairly high concentration of non-events (war not a mistake) in lower range of predicted values and fairly high concentration of events (war was a mistake) in higher range of predicted values, but obviously not perfect separation

Technique one: implementation using separationplot package

```
#feed separationplot the predicted v. observed outcomes
#from the binary model
separationplot(probit_df_order$probit_outcome_predicted,
               probit_df_order$probit_outcome_observed,
      type = "line", line = T, lwd1 = 0.5, lwd2 = 0.5,
xlab = "Predicted values in order", shuffle = T,
width = 9, height = 1.2,
col0 = "#FEF0D9", col1 = "#E34A33", flag = NULL, flagcol = 1,
file = "~/Dropbox/Working materials soc 504/precept5 draft/
auto_sep_plot", newplot = TRUE, locate = NULL, rectborder = NA,
show.expected = F, zerosfirst = T, BW = F,
heading = "Separation plot for Was Iraq a Mistake
predicted outcome: package version")
```

Technique one: results from separationplot package


Separation plot for Was Iraq a Mistake predicted outcome: package version

Technique two: k-fold cross-validation

- **Motivation:** we want to 'train' (fit) our model on one set of data (training set) and then evaluate how well model performs in predicting outcomes in test set
    - **One approach:** split our data into two subsets; use one as training, other as test
    - **How k-fold validation differs:** especially for small data sets, idea that by only training on, for instance, one half the data and testing on the other half, we 'waste' the data by only using it for one purpose; instead, let's split the data into k partitions so that each observation is sometimes in the training set and sometimes in test set
- *Acknowledgments (but no liability)*: Alex's great stats reading group presentation on the topic!

Technique two: k-fold cross-validation

General procedure:

1. Split data into K equally-sized partitions
2. For $k = 1, 2....K$:
    - 2.1 Use kth partition as test data; other $k-1$ partitions as training data
    - 2.2 Fit model on training data
    - 2.3 Evaluate how well model performs on test data...in precept today, performance = round predicted value to nearest integer (0 or 1) and see what fraction match observed values but other techniques: AUC under receiver operator curve (see Alex's presentation link previous slide); methods in `cv.glm` function in `boot package`, etc.[1]
    - 2.4 Repeat process for each k partition

[1] As discussed in precept, we can also call this performance metric a loss function, and when we shift to cases of using cross-validation for continuous outcomes, we may want to define this loss function in terms of $(\hat{y} - y)^2$ since distance between prediction and observed is especially important in cases without the $0, 1$ misclassification of binary outcomes case

Technique two: use 10-fold cross-validation to compare three model fits

- In your .rmd, use $K = 10$ to use cross-validation to compare the fit of three models for predicting views of Iraq war as a mistake:
    1. Model estimated in the paper: congressman speech + covariates outlined on earlier slide
    2. Same model as #1 but exclude congressman speech
    3. Same model as #1 but exclude the two party affiliation variables (gop3 and dem3)
- Remember that when defining 'compare the fit', we're using the following metric of performance for this example and that performance is in *test* set:
    1. Round the predicted value to the nearest integer (0 or 1)
    2. Compare to observed value for viewing Iraq as mistake (will be 0 or 1)
    3. Find fraction that match
- Compare the average of that metric across folds for each of the three models

Technique two (10-fold cross-val): setting up data/folds and formulae

```
#shuffle data and divide into 10 folds
war_shuffle <- war_nomiss %>%
                 sample_n(nrow(war_nomiss)) %>%
               mutate(fold_indic =
               rep_len(1:10, nrow(war_shuffle)))

#setting up formulae
form_withspeech <- formula(iraqamistake ~
totantiwarthroughjune16 + gop3 + dem3 + age + education4 +
male + white)

form_nospeech <- formula(iraqamistake ~  gop3 + dem3 +
age + education4 + male + white)

form_nopol <- formula(iraqamistake ~
totantiwarthroughjune16 + age + education4 + male + white)
```

Technique two (10-fold cross-val): storing the function

```
crossval.mod <- function(k, data, form_of_interest){
  ##divide data into training and test set
  test_data <- data %>% filter(fold_indic == k)
  train_data <- data %>% filter(fold_indic != k)

  ##fit model on training data
  mod_train <-  glm(form_of_interest, data = train_data,
            family = binomial(link = "probit"))

  ##use that model to predict outcomes in test df
  predict_test <- predict(mod_train, newdata = test_data, type = "response")

  ##bind with observed values in test data
  ##round, and code if it equals the observed, then summarise
  frac_match_res <- test_data %>%
              mutate(pred_out = predict_test,
                     round_pred_out = round(pred_out, 0),
                     match_obs_pred = ifelse(round_pred_out ==
                    iraqamistake, 1, 0)) %>%
              summarise(frac_match = sum(match_obs_pred)/nrow(test_data))
    ##return the fraction that match
    return(frac_match_res)
}
```

Technique two (10-fold cross-val): applying the function

```
kfold_res_speech <- sapply(1:10, crossval.mod,
                    data = war_shuffle,
                    form_of_interest = form_withspeech)
paper_mod <- mean(unlist(kfold_res_speech))

kfold_res_nospeech <- sapply(1:10, crossval.mod,
                    data = war_shuffle,
                    form_of_interest = form_nospeech)
nospeech <- mean(unlist(kfold_res_nospeech))

kfold_res_nopol <- sapply(1:10, crossval.mod,
                    data = war_shuffle,
                    form_of_interest = form_nopol)
nopol <- mean(unlist(kfold_res_nopol))
```

Technique two (10-fold cross-val): results

Model	Average rounded match with observed across 10 folds
Model in paper	0.7289
Model without speech predictor	0.7268
Model without respondent pol. affil	0.5608

- Cross-validation shows that excluding respondent's political affiliation from the model is associated with greater reduction in model fit in test sample than excluding the respondent's exposure to speeches
- Highlights contrast between good predictors (in this case, political affiliation) and the predictors that may be of causal interest

Shifting gears to ordered probit[2]

1. Review of latent variable interpretation
2. Programming the log-likelihood and optimizing
3. Practice with Zelig to find QOI

[2] Slides owe great debt to generations of TFs in Gov 2011 and (a) generation of 504

Motivation for ordered probit (or logit): ordered response category

- Authors don't report question wording and it was difficult to track down, but probably something like: "What should the U.S. do with the troops in Iraq?"
  - **1 = withdraw immediately**
  - **2 = withdraw in next year**
  - **3 = stay as long as necessary**
  - **4 = increase the number of troops**
- This has 1, 2, 3, 4– why not `lm`?
  - `lm` assumes a continuous outcome variable that is on an interval scale (distance between 1 and 2 is equivalent to distance between 3 and 4)
  - May be violated for scales like above where, for instance, when we map the categories onto a latent continuous variable that underlies each individual's response, there's a larger distance between withdrawing next year v. staying as long as necessary than between withdrawing immediately v. withdrawing next year

Latent variable interpretation applied to present example

- Suppose there is a latent (unobserved) data distribution, $Y^* \sim f_{stn}(y^*|\mu_i)$.
- This latent distribution has a systematic component, $\mu_i = x_i\beta$.
- Any realizations, y_i^*, are completely unobserved.
- In present case, this latent Y^* might represent something like a respondent's feelings about the appropriate deployment of troops in Iraq and when Gallup presents that respondent with the survey with the four categories from the previous slide, the respondent translates those latent feelings into an observed category choice
- What you *do* observe is whether y_i^* is between some threshold parameters

Latent variable interpretation: where do these thresholds come from?

- Shorter answer: since these thresholds are unobserved, we need to *estimate* them (treat them as parameters in our maximum likelihood estimation)
- So, for instance, if we pretend our latent variable about war Y^* ranges from 0-4, the thresholds might correspond to the following:
  - **1 = withdraw immediately**: $\tau_1 = 0.5$
  - **2 = withdraw in next year**: $\tau_2 = 1.8$
  - **3 = stay as long as necessary**: $\tau_3 = 2$
  - **4 = increase the number of troops**
- Or, they might be:
  - **1 = withdraw immediately**: $\tau_1 = 1$
  - **2 = withdraw in next year**: $\tau_2 = 1.7$
  - **3 = stay as long as necessary**: $\tau_3 = 2.2$
  - **4 = increase the number of troops**

Ordered probit: deriving the likelihood

In equation form,

$$y_{ij} = \begin{cases} 1 & \text{if } \tau_{j-i} < y_i^* \leq \tau_j \\ 0 & \text{otherwise} \end{cases}$$

Our stochastic component is still Bernoulli:

$$Pr(Y_{ij}|\pi) = \pi_{i1}^{y_{i1}} \pi_{i2}^{y_{i2}} \pi_{i3}^{y_{i3}} \dots$$

where $\sum_{j=1}^{M} \pi_{ij} = 1$

Ordered probit: Deriving the likelihood

Like the regular probit and logit, the key here is deriving π_{ij}.

You use this to derive the likelihood that y_i^* will fall into category j:

$$\begin{aligned}
\pi_{ij} = Pr(Y_{ij} = 1) &= Pr(\tau_{j-1} < y_i^* < \tau_j) \\
&= \int_{\tau_{j-1}}^{\tau_j} f(y_i^*|\mu_i) dy_i^* \\
&= \int_{\tau_{j-1}}^{\tau_j} f(y_i^*|x_i\beta) dy_i^* \\
&= F(\tau_j|x_i\beta) - F(\tau_{j-1}|x_i\beta)
\end{aligned}$$

where F is the cumulative normal density with variance 1.

Ordered probit: Deriving the likelihood

But this is the likelihood of one observation falling in one of the categories. We want to generalize to all observations and all categories

$$
\begin{aligned}
L(\tau, \beta | y) &= \prod_{i=1}^{n} \left\{ \prod_{j=1}^{m} [\pi_{ij}]^{y_{ij}} \right\} \\
L(\tau, \beta | y) &= \prod_{i=1}^{n} \left\{ \prod_{j=1}^{m} [F(\tau_j | x_i \beta) - F(\tau_{j-1} | x_i \beta)]^{y_{ij}} \right\}
\end{aligned}
$$

where τ is a vector of threshold parameters that you'll have to estimate.

Then we take the log to get the log-likelihood

$$
lnL(\tau, \beta | y) = \sum_{i=1}^{n} \sum_{j=1}^{m} y_{ji} ln[F(\tau_j | x_i \beta) - F(\tau_{j-1} | x_i \beta)]
$$

Two approaches to estimating the model

1. More manual: program that log-likelihood (together) and use `optim` to optimize
2. More automatic: check our work in #1 using `Zelig`

General steps for ordered probit log-likelihood

1. Create a matrix (we call it Z in the code) that for each respondent, indicates the choice of an outcome category we observe in the data (e.g., withdraw immediately versus send more troops)
2. In log-likelihood equation, we'll estimate both the unobserved thresholds for each category τ and how covariates relate to a particular respondent falling within each threshold

Step one: create matrix indicating each respondent's observed choice

```
##first make an empty matrix Z indicating
##what category a respondent falls in
##rows = number of respondents
##columns = number of levels of ordered response
levels <- length(unique(war_noNA$staythecourseindex))
Z <- matrix(NA, nrow(war_noNA), levels)
y <- war_noNA$staythecourseindex
y0 <- sort(unique(y))

##then, populate that Z matrix with each respondent
##and indicating whether they do (TRUE) or
##don't (FALSE) fall into that level
for(j in 1:levels){
  Z[, j] <- y == y0[j]
}
```

Step 1.5: create our X matrix without an intercept

```
##create X matrix- no intercept because
##we fix tau_1 to be cutoff point and intercept
##by not adding an intercept to the X,
##we're estimating that tau_1
X <- as.matrix(war_noNA %>%
              select(totantiwarthroughjune16,
                     gop3, dem3, age, education4,
                     male, white))
```

Step two: program the log-likelihood

```
loglik.probit <- function(par, Z, X){

    ##first, we separate the parameters
    ##vector into two types of parameters- beta
    ##and tau thresholds
    beta <- par[1:ncol(X)]
    tau <- par[(ncol(X)+1):length(par)]

    ##creates each respondent's
    ##latent variable y* as product
    ##of coefficients and covariate vals
    ystarmu <- X%*%beta

    ##levels (m in notation) is # of
    ##thresholds + 1
    levels <- length(tau) + 1

    ##create matrix for first 3 level prob
    cprobs = matrix(nrow=length(ystarmu), ncol=levels)

    ##create matrix to eventually store all level prob
    probs <- matrix(nrow=nrow(X), ncol=levels)
```

Step two: program the log-likelihood continued

```
    ##for levels 1-3, estimate each
    ##respondent's difference in cumul.prob
    ##of y* falling within thresholds for that level
    for (j in 1:(levels-1))
          cprobs[,j] <- pnorm(tau[j]- ystarmu)

    ##for last level, probability is 1 - probability of 3rd level
    probs[,levels] <- 1-cprobs[,levels-1]

    ##add in estimated prob for first level
    probs[,1] <- cprobs[,1]

    ##iterate through the 2nd and 3rd level of the ordinal variable and
    ## compute probability as c2-c1 (for second level)
    ##and c3-c2 (for third level) results in a mtrix with respondent prob
    ##of falling in each level filled in each respondent's probability sums to 1
    for (j in 2:(levels-1))
          probs[,j] <- cprobs[,j] - cprobs[,(j-1)]

    ##sum the logged probabilities to get
    ##log.lik
    sum(log(probs[Z]))
}
```

Optimize

Note that for these data, optimization is very sensitive to choice of starting parameters and as another note, care also needs to be taken when optimizing to make sure τ remain in the same order

```
##try optimizing
##for parameters,
##use the lm val
reg_lm <- lm(staythecourseindex ~ totantiwarthroughjune16 +
gop3 + dem3 + age + education4 + male + white,
             data = war_noNA)
coef_lm <- coef(reg_lm)[-1]
par <- c(coef_lm, 0, 1, 2)

optim(par, loglik.probit,
      Z = Z, X = X, method = "BFGS",
      control = list(fnscale = -1))
```

Way two: checking work using oprobit within Zelig/ZeligChoice

```
ordered.prob <- zelig(factor(staythecourseindex) ~
totantiwarthroughjune16 + gop3 +
              dem3 + age + education4 + male + white,
              data = war, model = "oprobit")
```

Motivation for simulation to get QOI's

- The more manual log-likelihood and Zelig each yield the following estimates for the relationship between the covariate of interest (House rep. speeches), other covariates, and the respondent's choice of ordered category on the Iraq war question
- Shows unsurprisingly, that republicans (gop3) have a higher probability of choosing a higher category

	Coef	SE
totantiwarthroughjune16	-0.067	0.026
gop3	0.712	0.093
dem3	-0.283	0.087
age	0.006	0.002
education4	0.120	0.038
male	0.426	0.073
white	0.319	0.103

- But that general coefficient could be driven by different parts of the category distribution– for instance, by republicans being *much less likely* to choose one of the withdrawal options than dems, or instead by republicans being *much more likely* to advocate for increased troops for instance– simulation helps us explore via *first differences* between two groups

General steps for using Zelig for simulation for QOI process we've been doing manually

1. Use `setx` to create data.frames of interest (in this case, one for gop3 $= 1 =$ republican; another for gop3 $= 0 =$ democrat)

   ```
   x.repub <- setx(ordered.prob, gop3 = 1)
   x.dem <- setx(ordered.prob, gop3 = 0)
   ```

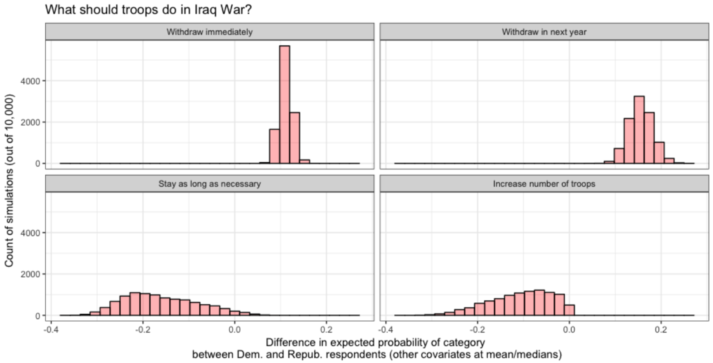
2. Use `sim` to use simulation for QOI- here, we feed it the model, our first data.frame of interest, our second one of interest (since we want it to return first differences among other things), and tell it to run 10,000 simulations

   ```
   sim.oprobit <- sim(ordered.prob,
                      x = x.repub,
                      x1 = x.dem,
                      num = 10000)
   ```

3. Once we have those simulations, see .rmd solutions for code to extract the first differences results for plotting in `ggplot` rather than `Zelig's` built-in graphics (accessed via `plot(simobject)`) for plot we show on next slide

Simulation results: distribution of category differences driving significant coefficient

First differences shows that significant coefficient on Republican largely driven by Democrats' much higher probability of choosing a withdrawal response (with distribution > 0), and less by Republicans' probability of choosing a keeping or increasing the troops response (where distributions overlap with zero)



Summing up

- **Model diagnostics**
    - Separation plots
    - *k-fold* cross validation
- Ordered probit model
    - Conceptual review of latent variable interpretation
    - Derive and optimize log likelihood
    - Practice with `Zelig` to estimate model and if time, automatic version of simulation for QOI