POLSCI 9592

Lecture 5: Ordinal Data Models

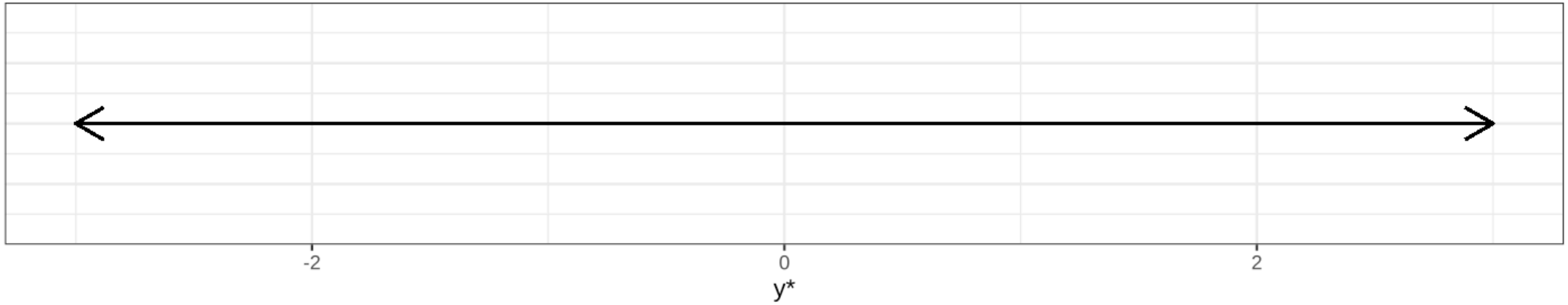
Dave Armstrong

Goals for This Session

1. Ordinal Model Estimation
2. Effects and Effect Displays
3. Model Fit and Evaluation
4. Parallel Regressions Assumption

Basic Idea Behind Ordinal Model

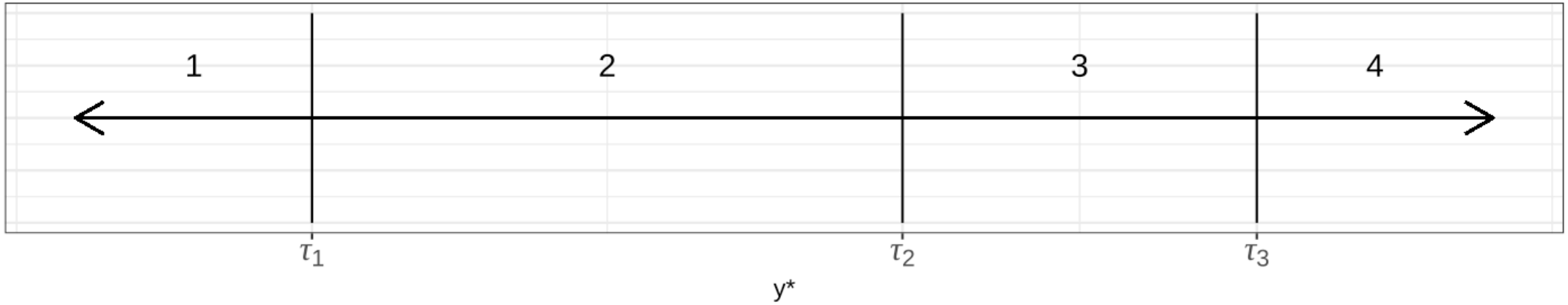
Assume that there is a phenomenon we are trying to explain, let's call it y^*, that lies on a continuum that we don't observe directly.



If we did observe it directly, we would simply be able to do a linear regression on y^*.

What We Actually Observe

What we actually observe as y, a more coarse version of y^*.



Everything that falls in between two vertical lines is coded the same value. For example, everything that falls between τ_1 and τ_2 will be a 2 on y.

Models for Ordinal Data

The mathematical notation for the above is

$$y_i = m \text{ if } \tau_{m-1} \leq y_i^* < \tau_m \text{ for } m = \{1, \ldots, J\}$$

where $\tau_0 = -\infty$ and $\tau_J = \infty$, the other $J - 1$ cut-points are estimated.

In addition to above, we also set $b_0 = 0$ so the model works (i.e., is identified).

Probabilities

To characterize $Pr(y = m|X)$ we need to assume a probability distribution for the errors in the latent variable model.

- Our choices here are really either normal (ordered probit) with $\sigma^2 = 1$ or logistic (ordered logit) with $\sigma^2 = \frac{\pi^2}{3}$. Because we know (or estimate) all of the τ parameters, and we have a probability distribution, say the logistic distribution, then we know:

$$Pr(y \leq m) = \Lambda(\tau_j - Xb) \qquad = \frac{1}{1 + e^{-(\tau_j - Xb)}}$$

Probabilities II

Note that what we get above is the probability of being less than or equal to one particular value. Let's take the observed value 2. We know that:

$$Pr(y \leq 2) = \Lambda(\tau_2 - Xb)$$

What if we wanted to find $Pr(y = 2)$?

Probabilities II

Note that what we get above is the probability of being less than or equal to one particular value. Let's take the observed value 2. We know that:

$$Pr(y \leq 2) = \Lambda(\tau_2 - Xb)$$

What if we wanted to find $Pr(y = 2)$?

We would need to find:

$$\begin{aligned} Pr(y = 2) &= Pr(y \leq 2) - Pr(y \leq 1) \\ &= \Lambda(\tau_2 - Xb) - \Lambda(\tau_1 - Xb) \end{aligned}$$

Likelihood Function

Just like with binary logit models, the likelihood for an individual observation is the probability that the observation takes on its observed value.

$$L(\mathbf{b}|\mathbf{x}_i) = \Lambda(\tau_{y_i} - \mathbf{x}_i b) - \Lambda(\tau_{y_i-1} - \mathbf{x}_i b)$$

And the log-likelihood is just:

$$\log L(\mathbf{b}|\mathbf{x}_i) = \log\big(\Lambda(\tau_{y_i} - \mathbf{x}_i b) - \Lambda(\tau_{y_i-1} - \mathbf{x}_i b)\big)$$

$$LL = \sum_{i=1}^{N} \log\big(\Lambda(\tau_{y_i} - \mathbf{x}_i b) - \Lambda(\tau_{y_i-1} - \mathbf{x}_i b)\big)$$

Example Data

The example data are about state repression.

- `sdfac`: State Department Repression score $\{1, 2, 3, 4, 5\}$.
- `cwarcow`: COW civil war indicator.
- `iwarcow`: COW interstate war indicator.
- `logpop`: Log of population.
- `logpcgnp`: Log of per-capita GNP.
- `vanadd`: Additive index of democracy (Vanhanen).

Other notes:

- Alternative summary function is available in the code file. The one from the `MASS` package doesn't provide p-values.

Example

```
library(ordinal)
dat <- rio::import("data/ologit_data.dta")
dat$sd_fac <- factor(dat$sd, levels=1:5)
mod <- clm(sd_fac ~ cwarcow + iwarcow + logpop + logpcgnp + poly(vanadd, 2), data=dat)
summary(mod)
```

```
## formula: sd_fac ~ cwarcow + iwarcow + logpop + logpcgnp + poly(vanadd, 2)
## data:    dat
##
##  link  threshold nobs logLik   AIC     niter max.grad cond.H
##  logit flexible  2550 -2815.96 5651.91 6(0)  4.20e-11 1.9e+06
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## cwarcow             2.99351    0.16109   18.58  < 2e-16 ***
## iwarcow             0.93657    0.19472    4.81 1.51e-06 ***
## logpop              0.42748    0.02723   15.70  < 2e-16 ***
## logpcgnp           -0.34331    0.03478   -9.87  < 2e-16 ***
## poly(vanadd, 2)1 -43.81355    2.67265  -16.39  < 2e-16 ***
## poly(vanadd, 2)2 -25.29074    2.34367  -10.79  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 1|2   3.1857     0.5105   6.241
## 2|3   5.2670     0.5174  10.180
## 3|4   7.3189     0.5301  13.807
## 4|5   9.0755     0.5422  16.738
```

Testing Coefficients

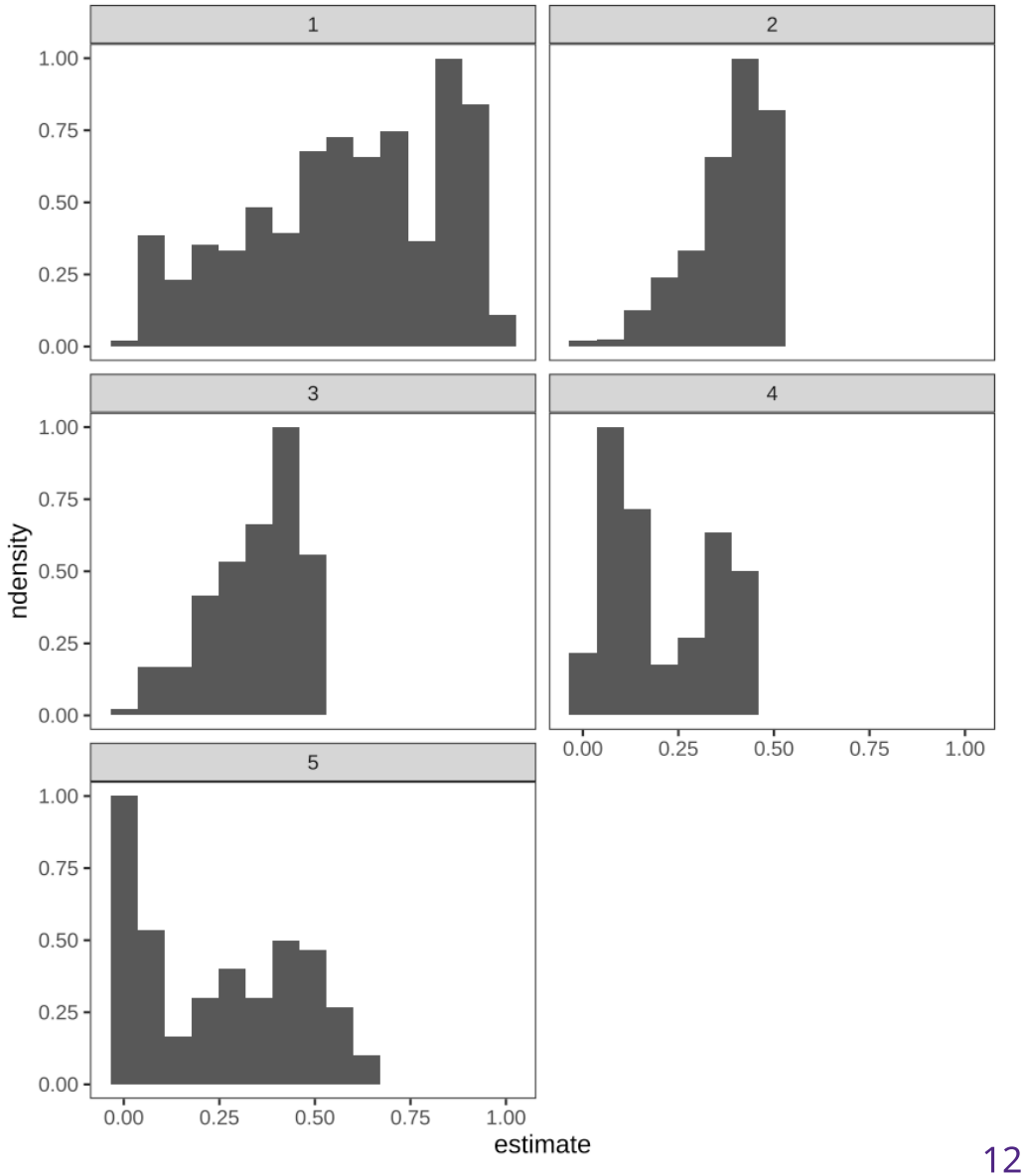
Just like in other GLMs and the linear model, the `Anova` function from the `car` package can be used to evaluate the significance of each of the model's terms.

```
library(car)
Anova(mod)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: sd_fac
##                  Df   Chisq Pr(>Chisq)
## cwarcow           1 345.334  < 2.2e-16 ***
## iwarcow           1  23.134  1.511e-06 ***
## logpop            1 246.530  < 2.2e-16 ***
## logpcgnp          1  97.422  < 2.2e-16 ***
## poly(vanadd, 2)   2 290.521  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Probabilities by Group

```
library(marginaleffects)
probs <- predictions(mod, newdata=dat)
probs <- probs %>%  filter(group == sd)
ggplot(probs, aes(x=estimate)) +
  geom_histogram(aes(y=after_stat(ndensity)), bins=15) +
  facet_wrap(~sd, nrow=3) +
  theme_bw() +
  theme(panel.grid=element_blank())
```



Discrete Change

For any variable (and particularly qualitative ones), the discrete change is:

$$\Delta Pr(y = m|x) = Pr(y = m|x, x_k = x_k^{\text{end}}) - Pr(y = m|x, x_k = x_k^{\text{start}})$$

This simply gives the difference in predicted probability for a discrete change in one X, holding the other x's constant at particular values.

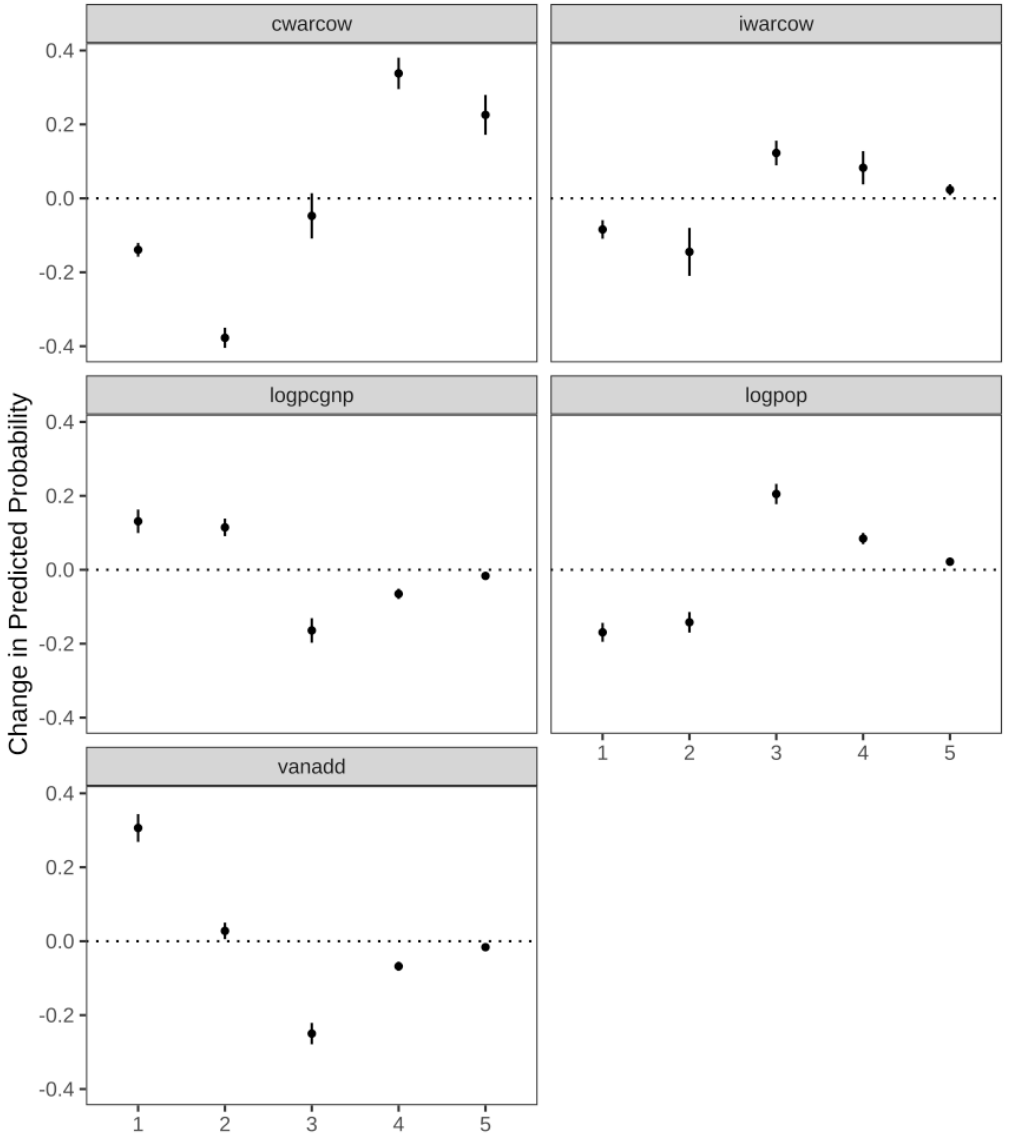
Example: MER Approach

```
library(tidyr)
mer_comps <- comparisons(mod, newdata = datagrid("median"),
                         variables=list(cwarcow = c(0,1),
                                        iwarcow = c(0,1),
                                        logpop = "2sd",
                                        logpcgnp = "2sd",
                                        vanadd = "2sd"),
                         type="prob")
mer_comps %>%
  mutate(estimate = sprintf("%.2f%s", estimate, ifelse(sign(conf.low) == sign(conf.high), "*", ""))) %>%
  select(term, contrast, group, estimate) %>%
  pivot_wider(names_from = "group", values_from = "estimate")
```

```
## # A tibble: 5 × 7
##   term     contrast            `1`    `2`    `3`    `4`    `5`
##   <chr>    <chr>               <chr>  <chr>  <chr>  <chr>  <chr>
## 1 cwarcow  1 - 0               -0.14* -0.38* -0.05  0.34*  0.23*
## 2 iwarcow  1 - 0               -0.08* -0.14* 0.12*  0.08*  0.02*
## 3 logpcgnp (x + sd) - (x - sd) 0.13*  0.11*  -0.16* -0.07* -0.02*
## 4 logpop   (x + sd) - (x - sd) -0.17* -0.14* 0.20*  0.08*  0.02*
## 5 vanadd   (x + sd) - (x - sd) 0.31*  0.03*  -0.25* -0.07* -0.02*
```

Plot

```
ggplot(mer_comps, aes(x=as.factor(group), y=estimate,
                      ymin=conf.low, ymax=conf.high)) +
  geom_pointrange(size=.1) +
  geom_hline(yintercept=0, linetype=3) +
  facet_wrap(~term, ncol=2) +
  theme_bw() +
  theme(panel.grid=element_blank()) +
  labs(x="", y="Change in Predicted Probability")
```



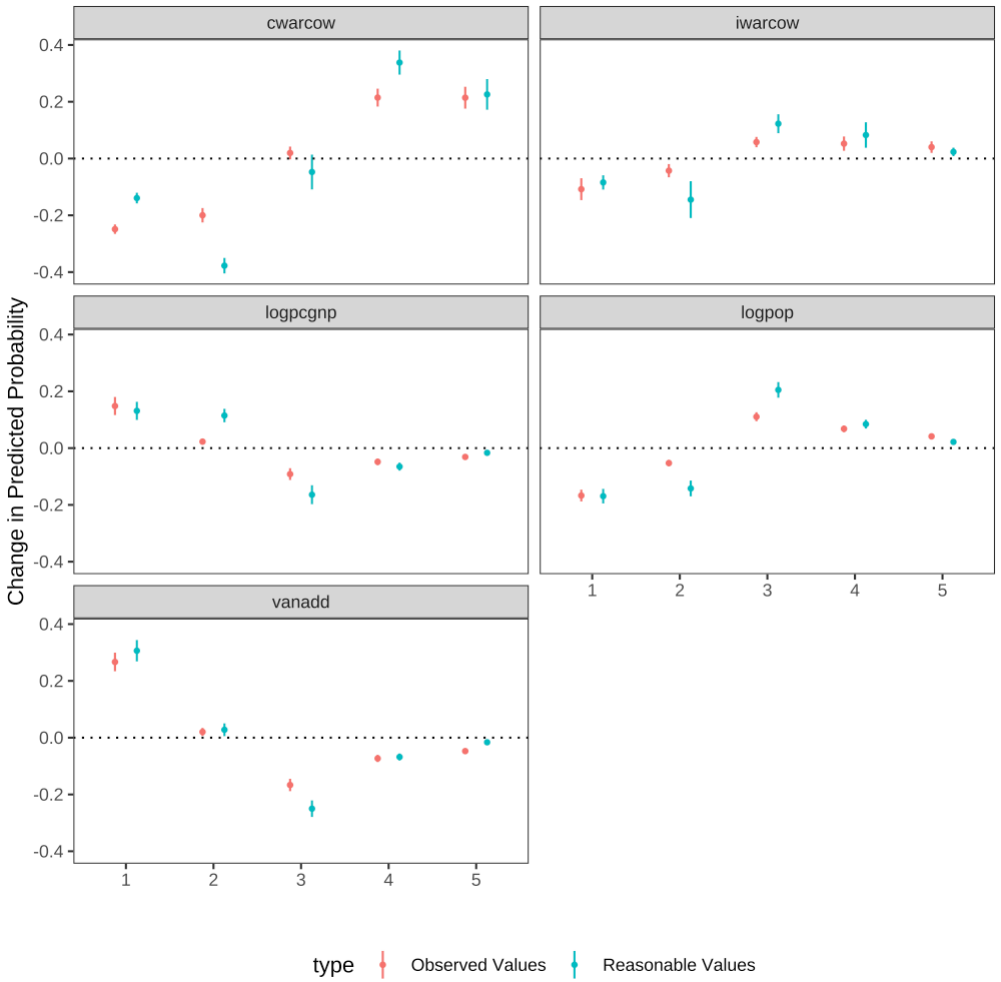
Example: AME Approach

```
ave_comps <- avg_comparisons(mod,
                          variables=list(cwarcow = c(0,1),
                                         iwarcow = c(0,1),
                                         logpop = "2sd",
                                         logpcgnp = "2sd",
                                         vanadd = "2sd"),
                          type="prob")
ave_comps %>%
  mutate(estimate = sprintf("%.2f%s", estimate, ifelse(sign(conf.low) == sign(conf.high), "*", ""))) %>%
  select(term, contrast, group, estimate) %>%
  pivot_wider(names_from = "group", values_from = "estimate")
```

```
## # A tibble: 5 × 7
##   term     contrast            `1`    `2`    `3`    `4`    `5`
##   <chr>    <chr>               <chr>  <chr>  <chr>  <chr>  <chr>
## 1 cwarcow  1 - 0               -0.25* -0.20* 0.02   0.21*  0.21*
## 2 iwarcow  1 - 0               -0.11* -0.04* 0.06*  0.05*  0.04*
## 3 logpcgnp (x + sd) - (x - sd) 0.15*  0.02*  -0.09* -0.05* -0.03*
## 4 logpop   (x + sd) - (x - sd) -0.17* -0.05* 0.11*  0.07*  0.04*
## 5 vanadd   (x + sd) - (x - sd) 0.27*  0.02*  -0.17* -0.07* -0.05*
```

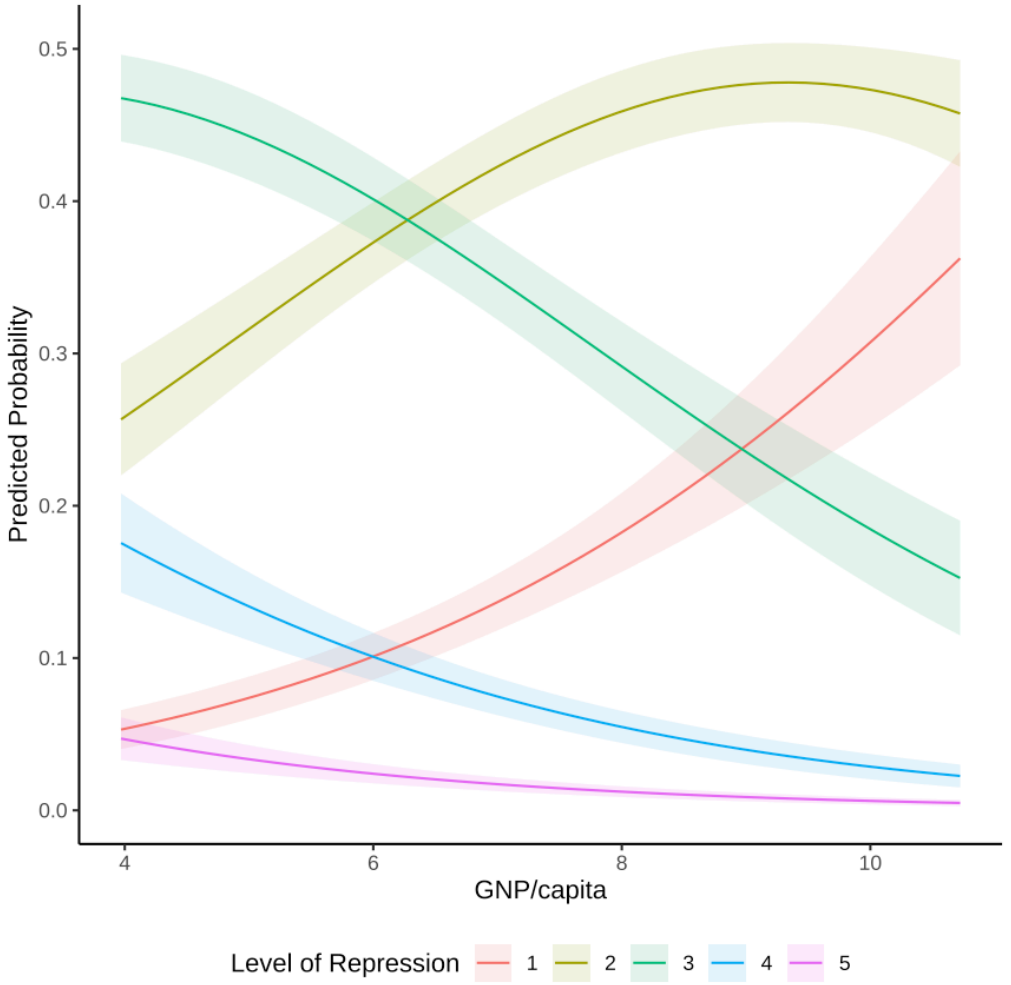
Plot

```
bcomps <- bind_rows(
  mer_comps %>%
    select(term, group, estimate, conf.low, conf.high) %>%
    mutate(type = "Reasonable Values"),
  ave_comps %>%
    select(term, group, estimate, conf.low, conf.high) %>%
    mutate(type = "Observed Values"))


ggplot(bcomps, aes(x=as.factor(group), y=estimate,
                       ymin=conf.low, ymax=conf.high,
                   colour=type)) +
  geom_pointrange(size=.05, position=position_dodge(width=.5)) +
  geom_hline(yintercept=0, linetype=3) +
  facet_wrap(~term, ncol=2) +
  theme_bw() +
  theme(panel.grid=element_blank(),
        legend.position="bottom") +
  labs(x="", y="Change in Predicted Probability")
```



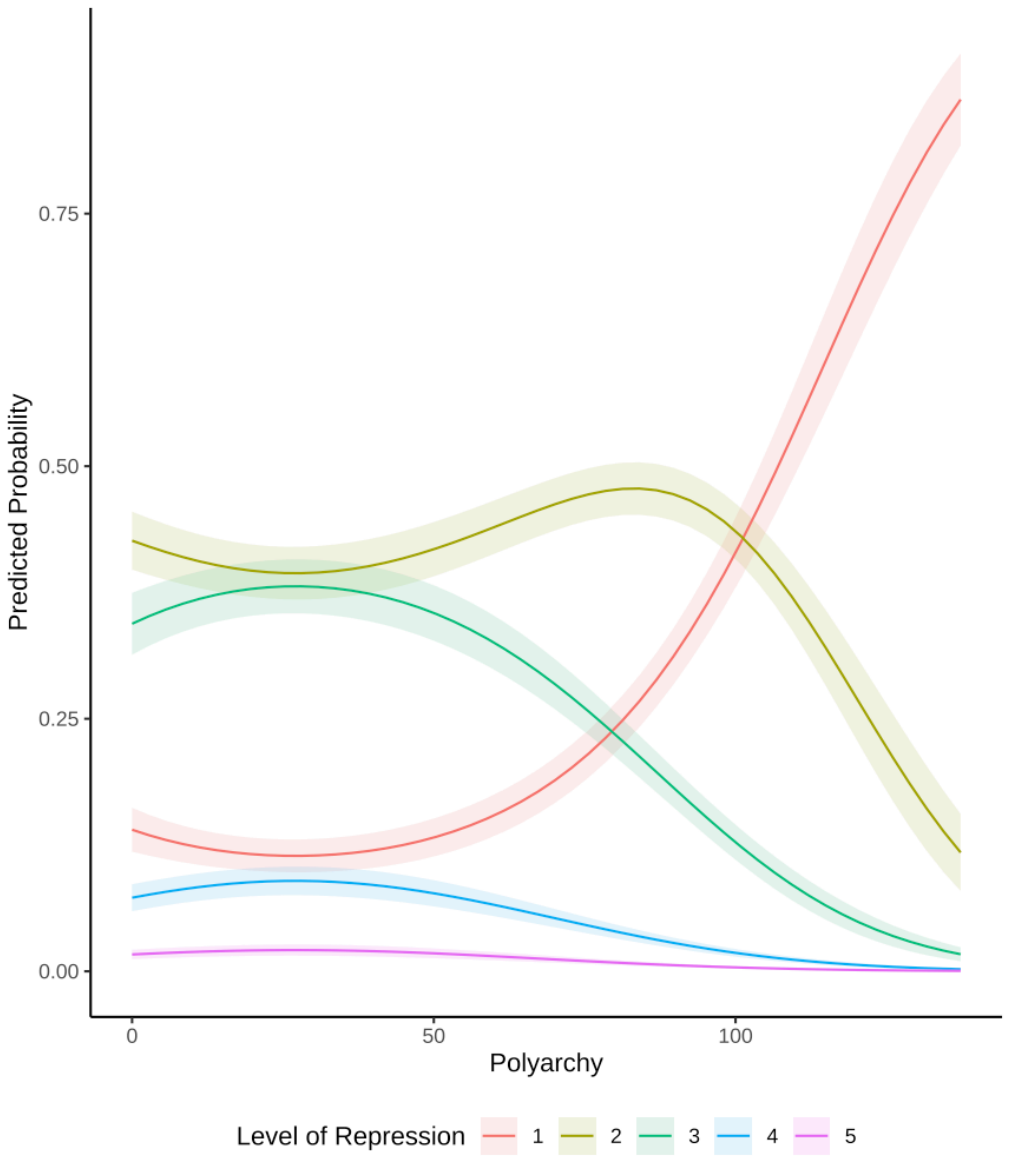
MER Effects Plot

```
pred_gnp <- plot_predictions(mod,
                             newdata="median",
                             condition="logpcgnp",
                             draw=FALSE)
ggplot(pred_gnp, aes(x=logpcgnp, y=estimate,
                     ymin=conf.low,
                     ymax=conf.high,
                     fill=as.factor(group),
                     colour=as.factor(group))) +
  geom_ribbon(alpha=.15, colour="transparent") +
  geom_line() +
  theme_classic() +
  theme(legend.position="bottom") +
  labs(x="GNP/capita", y="Predicted Probability",
       colour = "Level of Repression",
       fill = "Level of Repression")
```



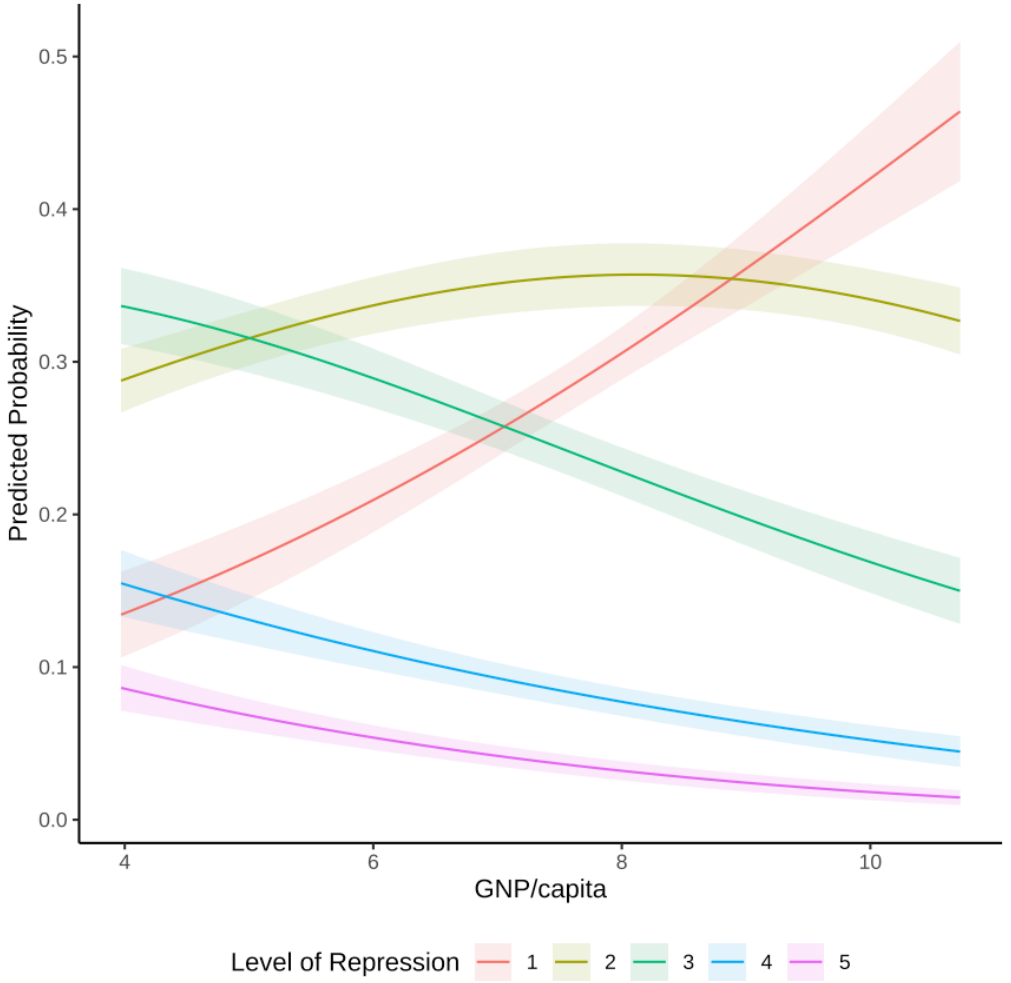
MER Approach: Polynomial

```
pred_dem <- plot_predictions(mod,
                         condition="vanadd",
                         newdata = "median",
                         draw=FALSE)

ggplot(pred_dem, aes(x=vanadd, y=estimate,
                     ymin=conf.low,
                     ymax=conf.high,
                     fill=as.factor(group),
                     colour=as.factor(group))) +
  geom_ribbon(alpha=.15, colour="transparent") +
  geom_line() +
  theme_classic() +
  theme(legend.position="bottom") +
  labs(x="Polyarchy", y="Predicted Probability",
       colour = "Level of Repression",
       fill = "Level of Repression")
```



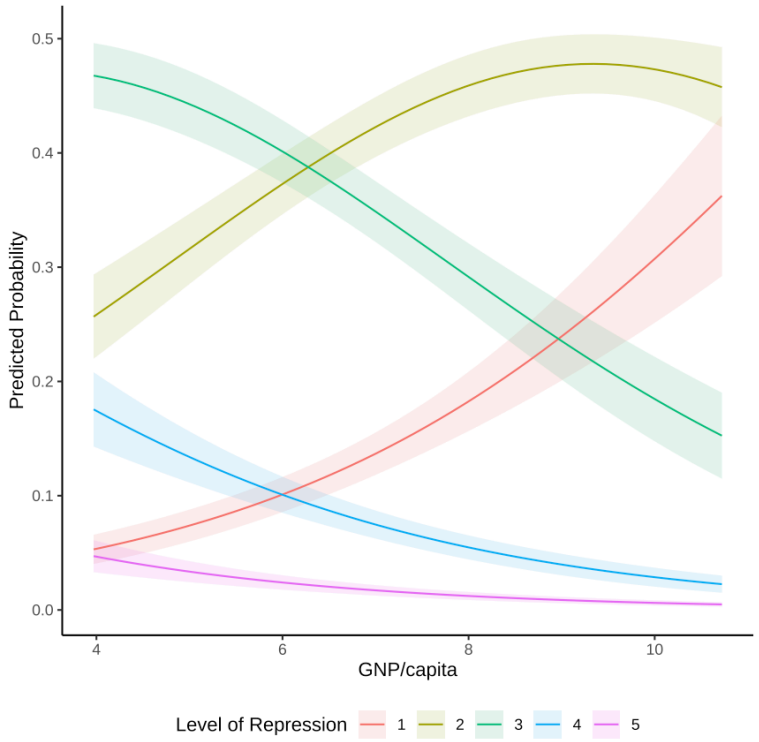
Plotting Predicted Probabilities: AME Approach

```
seq_range <- function(x,n=100){
  mn <- min(x, na.rm=TRUE)
  mx <- max(x, na.rm=TRUE)
  seq(mn, mx, length=n)
}
pred_gnp2 <- avg_predictions(mod,
                        variables=list(logpcgnp=seq_range(dat$log

ggplot(pred_gnp2, aes(x=logpcgnp, y=estimate,
                      ymin=conf.low,
                      ymax=conf.high,
                      fill=as.factor(group),
                      colour=as.factor(group))) +
  geom_ribbon(alpha=.15, colour="transparent") +
  geom_line() +
  theme_classic() +
  theme(legend.position="bottom") +
  labs(x="GNP/capita", y="Predicted Probability",
       colour = "Level of Repression",
       fill = "Level of Repression")
```

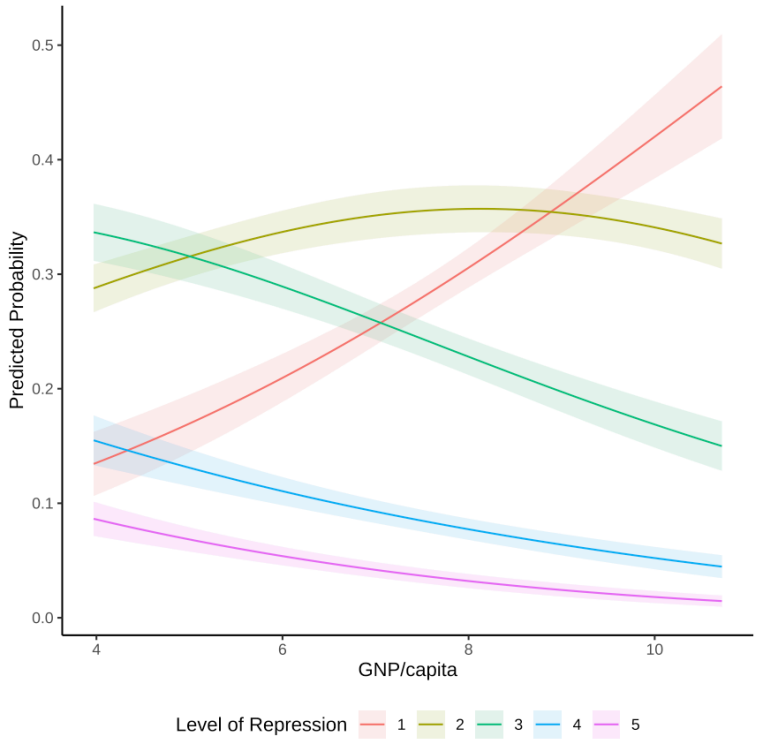


MER vs AME

MER



AME



Model Fit

```
library(DAMisc)
ordfit(mod, data=dat)
```

```
##                       Estimate
## Count R2              0.529
## Count R2 (Adj)        0.321
## ML R2                 0.497
## McFadden R2           0.237
## McFadden R2 (Adj)     0.235
## McKelvey & Zavoina R2 0.533
```

Model Fit II

```
p <- MASS::polr(formula(mod), data=dat)
pre(p, data=dat)
```

```
## mod1:  sd_fac ~ cwarcow + iwarcow + logpop + logpcgnp + poly(vanadd, 2)
## mod2:  sd_fac ~ 1
##
## Analytical Results
##  PMC =  0.306
##  PCP =  0.529
##  PRE =  0.321
## ePMC =  0.255
## ePCP =  0.404
## ePRE =  0.200
```

Comparative Model Fit

The same options that exist for the binary model work here, too.

- Likelihood ratio tests for nested models.
- AIC, AICc and BIC for non-nested models.
- Clarke Test for non-nested models.

Ordered Model vs Linear Model

Sometimes, we wonder whether we *could* estimate a linear model rather than the ordered model.

- Interpretation is much easier.
- Effects are not conditional unless specified as such.

Problem:

- No information in categorical variable to estimate the residual variance from the linear model.
- Residual variance from the linear model estimated on the categories will not be the residual variance from the linear model.

Methematically

Assume the latent variable y^* has the following equation:

$$y^* = b_0 + b_1 x + b_2 z + e \quad e \sim N(0, \sigma^2)$$

If we knew the residual variance, we could assume that in our model and all would be good. Instead we set it at 1 (with probit). If we wanted to re-write the above to force the residual variance to 1, we would write.

$$y^* = b_0 + b_1 x + b_2 z + \sigma e \quad e \sim N(0, 1)$$
$$\frac{y^*}{\sigma} = \frac{b_0}{\sigma} + \frac{b_1}{\sigma} x + \frac{b_2}{\sigma} z + e \quad e \sim N(0, 1)$$

So, what we are actually estimating in the ordinal model is a scaled version of the true coefficients, but the scale factor is unknown.

Parallel Regressions Assumption

There is one additional assumption we make with these types of models - the *Parallel Regressions Assumption*.

- Assumes that the relationship between x and y doesn't change from one value of y to another.
- The coefficient relating x to y is the same for all categories.

The Null and Alternative Hypotheses are:

- H_0: The parallel regressions assumption holds.
- H_A: The parallel regressions assumption doesn't hold.

Brant Test

```
library(brant)
dat$vascale <- scale(dat$vanadd)
bmod  <- MASS::polr(sd_fac ~ cwarcow + iwarcow + logpop +
        logpcgnp + vascale + I(vascale^2), data=dat)
```

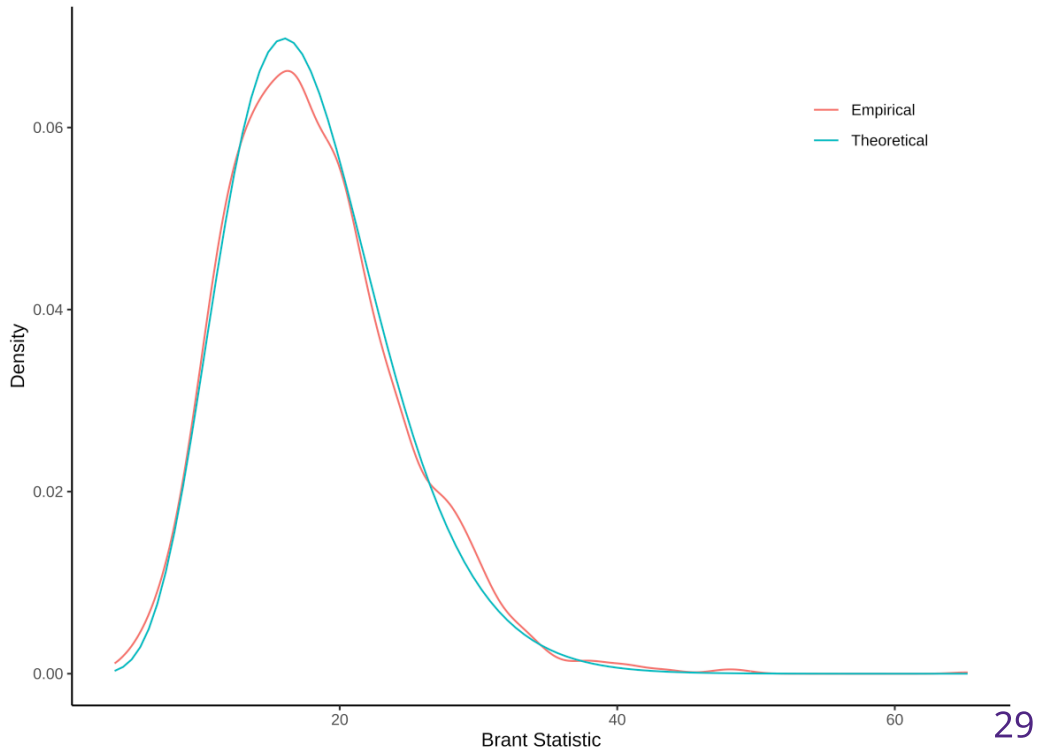
```
library(brant)
brant(bmod)
```

```
## ------------------------------------------------
## Test for    X2    df    probability
## ------------------------------------------------
## Omnibus        227.21    18    0
## cwarcow         3.81    3    0.28
## iwarcow         7.08    3    0.07
## logpop         20.63    3    0
## logpcgnp    67.36    3    0
## vascale        56.27    3    0
## I(vascale^2)    27.06    3    0
## ------------------------------------------------
##
## H0: Parallel Regression Assumption holds
```

Simulation for Brant Test

```
X <- model.matrix(bmod)[,-1]
b <- coef(bmod)

xb <- X %*% b
tau <- c(-Inf, bmod$zeta, Inf)
q <- sapply(tau, function(t)plogis(t-xb))
D <- matrix(0, nrow=ncol(q), ncol = ncol(q)-1)
D[c(1,2), 1] <- c(-1,1)
D[c(2,3), 2] <- c(-1,1)
D[c(3,4), 3] <- c(-1,1)
D[c(4,5), 4] <- c(-1,1)
D[c(5,6), 5] <- c(-1,1)

probs <- q %*% D

brant_stats <- NULL
sink(tempfile())
for(i in 1:2500){
  newy <- as.factor(apply(probs, 1, function(x)
            which.max(rmultinom(1, 1, x))))
  u <- update(bmod, newy ~ .)
  brant_stats <- c(brant_stats, brant(u)[1])
}
sink()
```

```
load("data/brant_stats.rda")
ggplot() +
  stat_density(aes(x=brant_stats, colour="Empirical"), geom="line
  stat_function(fun = function(x)dchisq(x, 18), aes(colour="Theor
  theme_classic() +
  theme(legend.position="inside",
        legend.position.inside = c(.85, .85)) +
  labs(x="Brant Statistic", y = "Density", colour="")
```



What Next?

If, as above, we reject H_0 for the Brant Test, we must re-estimate a more flexible model.

- We can do this with what we'll learn next - multinomial logit.

Review

1. Ordinal Model Estimation
2. Effects and Effect Displays
3. Model Fit and Evaluation
4. Parallel Regressions Assumption

Exercises

1. Choose some variables as independent variables and estimate an ordered logit model of `demsat` using your chosen independent variables.
   - How does the model fit?
   - What are the effects of the variables in the model?
2. Does this model meet the assumptions of the ordered logit model (particularly, the parallel regressions assumption)?

- `vote` Party of vote
- `votenum` Party number of chosen party
- `male` R gender (0=female, 1=male)
- `age` Age of R
- `urban` size of city in which R lives
- `soclass` Subjective social class
- `hhincome` Household income (in francs/month)
- `union` Member of a union
- `retnat` Retrospective national economic evaluations
- `demsat` satisfaction with the functioning of democracy
- `eusup` Good choice for France to belong to EU
- `lrself` position on left-right scale (0-10)