Multinomial and ordered logits

Silje Synnøve Lyder Hermansen

2025-03-17

GLM: A recap

Reminder: What is a GLM?

Regressions aim to describe (a linear) relationship between x and y with one number, β.

- Assumes a continuous and unbounded variable.
- When y is neither (e.g. binary), we relied on a latent continuous variable
- To approximate the latent variable, we calculated the logodds (i.e. we compare)

$\Rightarrow$ *Probability distribution maps unobserved variable to observed outcomes.*

Today: nominal and ordinal variables

Strategies when our outcome variable is categorical

- categorical (e.g. party, profession,. . .) → *multinomial regression*
- ordinal (e.g. attitudes towards topics. . .) → *ordinal regression*

⇒ *Models of choice where we model the chooser's characteristics*

Multinomial logistic regression

Two conceptions of multinomial regression

Two conceptions of multinomial regression

- **Latent variable approach:** Our utility of each choice.
- **A series of binomial logits** with the same reference category.

Latent variable approach

Latent variable approach: Imagine m choices modeled as
$y_m = a_m \times b_m x_i$

- $b_m x_i$ reflects the utility of a choice m for the chooser i with x characteristic. → systematic term
- a_m reflects the baseline utility of that choice → stochastic term

⇒ *The preferred choice is the one with the highest utility*

Example: Party choice

Example: Party choice

Let's consider party choice among voters

- ESS survey round (chap 6, Hermansen, 2023)
- respondents give:
  - preferred party
  - attitudes towards immigration

I can rank parties

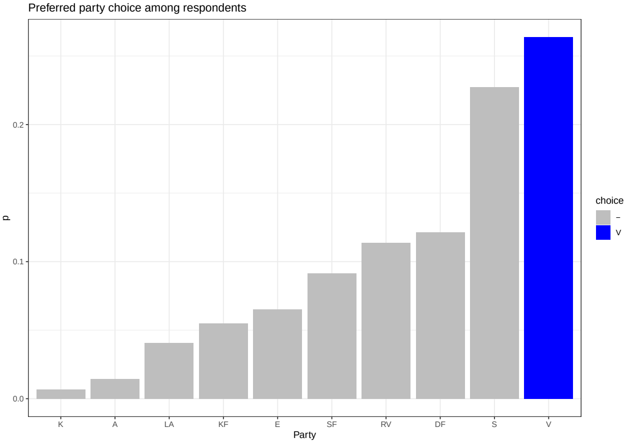
Let’s rank the parties according to the respondents’ choice

```r
tab <-
  df %>%
  #Group by party
  group_by(Party) %>%
  #Number of respondent by party
  reframe(n = n()) %>%
  mutate(
    #Total number of respondents
    N = sum(n),
    #Proportion/probability of group
    p = n/N) %>%
  #Sort just for facility
  arrange(p) %>%
  mutate(
    #Check if it sums up to 1
    cum = cumsum(p),
    #Which is the largest?
    choice = if_else(row.names(.) == which.max(p),
                     Party, "-"))
```

I can rank parties

```
## # A tibble: 10 x 6
##    Party     n     N       p     cum choice
##    <chr> <int> <int>   <dbl>   <dbl> <chr>
##  1 K         8  1179 0.00679 0.00679 -
##  2 A        17  1179 0.0144  0.0212  -
##  3 LA       48  1179 0.0407  0.0619  -
##  4 KF       65  1179 0.0551  0.117   -
##  5 E        77  1179 0.0653  0.182   -
##  6 SF      108  1179 0.0916  0.274   -
##  7 RV      134  1179 0.114   0.388   -
##  8 DF      143  1179 0.121   0.509   -
##  9 S       268  1179 0.227   0.736   -
## 10 V       311  1179 0.264   1       V
```

I can rank parties (figure)



- the most frequent party choice is the most probable outcome

Theoretical link to political science

The assumption is that choosers are rational, and choose a category (m_j) whenever its utility exceeds the alternative (m_d).

$$U(m_j) > U(m_d)$$

$\Rightarrow$ *This is also how we estimate it; through comparisons*

A series of binomial logits

A series of binomial logits with the *same* reference category.

- Data consists of many groups, but I only compare two groups → data/variation intensive model choice.
- Categories/choice are mutually exclusive → Different β for each choice

⇒ *All choices are given a probability and they sum up to one.*

Example: ESS survey round

Let's do an intercept-only model

Logit transformation:

$$logit(p_m) = log(\frac{p_m}{p_d})$$

```
tab <-
  df %>%
  #Group by party
  group_by(Party) %>%
  #Number of respondent by party
  reframe(n = n()) %>%
  mutate(
    #Total number of respondents
    N = sum(n),
    #Proportion/probability of group
    p = n/N,
    #Pick Social democrats as reference category
    p_ref = p[Party == "S"],
    #Odds
    odds = p/p_ref,
    #Logodds
    logodds = log(odds))
```

Example: ESS survey

- intercept-only model
- . . . where the reference-level (S) is effectively left out

```
## # A tibble: 10 x 7
##    Party     n     N       p p_ref   odds logodds
##    <chr> <int> <int>   <dbl> <dbl>  <dbl>   <dbl>
##  1 A        17  1179 0.0144  0.227 0.0634  -2.76
##  2 DF      143  1179 0.121   0.227 0.534   -0.628
##  3 E        77  1179 0.0653  0.227 0.287   -1.25
##  4 K         8  1179 0.00679 0.227 0.0299  -3.51
##  5 KF       65  1179 0.0551  0.227 0.243   -1.42
##  6 LA       48  1179 0.0407  0.227 0.179   -1.72
##  7 RV      134  1179 0.114   0.227 0.5     -0.693
##  8 S       268  1179 0.227   0.227 1        0
##  9 SF      108  1179 0.0916  0.227 0.403   -0.909
## 10 V       311  1179 0.264   0.227 1.16     0.149
```

In R: set a reference level

- We set a reference level p_d: That's the leave-one-out trick.

```
df <-
  df %>%
  #I use the Social democrats
  mutate(Party = relevel(as.factor(Party), ref = "S"))
```

- Estimate the model

```
library(nnet)
mod.cat <- multinom(Party ~
                      1,
                    df)
```

```
## # weights:  20 (9 variable)
## initial  value 2714.747825
## iter  10 value 2332.511892
## final  value 2326.831829
## converged
```

Results table

The result is a series of equations, one for each party

Table 1:

	Dependent variable:						
	A	DF	E	K	KF	LA	RV
	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Constant	-2.76^{***} (0.25)	-0.63^{***} (0.10)	-1.25^{***} (0.13)	-3.51^{***} (0.36)	-1.42^{***} (0.14)	-1.72^{***} (0.16)	-0.69^{*} (0.11
Akaike Inf. Crit.	4,671.66	4,671.66	4,671.66	4,671.66	4,671.66	4,671.66	4,671.

Note:

With predictors

Let’s regress party choice on scepticism towards immigration

```
library(nnet)
mod.cat <- multinom(Party ~
                      Skepsis,
                    df)
```

```
## # weights:  30 (18 variable)
## initial  value 2705.537484
## iter  10 value 2304.290245
## iter  20 value 2246.392642
## final  value 2246.301290
## converged
```

Table 2:

	Dependent variable:						
	A	DF	E	K	KF	LA	RV
	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Skepsis	0.23	0.56***	−0.04	−0.18	0.04	0.07	−0.30***
	(0.15)	(0.07)	(0.08)	(0.24)	(0.09)	(0.10)	(0.07)
Constant	−3.89***	−3.69***	−1.04**	−2.70**	−1.58***	−2.06***	0.62*
	(0.85)	(0.40)	(0.41)	(1.09)	(0.44)	(0.51)	(0.32)
Akaike Inf. Crit.	4,528.60	4,528.60	4,528.60	4,528.60	4,528.60	4,528.60	4,528.60
Note:							*p

Interpretation

Interpretation

All the possibilities of the binomial logit are open, but the backtransformation is a hack

⇒ *However, you want to decide which story you want to tell*

Different approaches

- With respect to the reference category
    - the regression table (logodds): direction and statistical significance
    - marginal effects (partial back-transformation): relative change
- Predicted outcomes per category
    - predicted probability of each category (transformation of latent variable): when one increases, the other decrease
    - predicted choice (total back-transformation): most probable choice

Marginal effects

The marginal effects are interpreted with reference to the reference level:

- A one-unit increase in skepticism decreases the probability of voting Alternativet rather than Social democrats with:
  - (1 - exp(0.23) $\times$100) = -25%

Predicted probabilities

The results can be read as a series of equations, one for each category m

$$Pr(y = m) \sim log(odds)$$
$$log(odds) = a_m + b_m x$$

- predictions for each category → *separate slopes and intercept*

$$log(odds) = -3.89 + 0.23x$$

⇒ *The "latent" variable is here represented by the logodds*

Predicted probabilities (cont.)

The manual backtransformation requires more manual work

1. set a scenario (e.g. x = 5)
2. backtransform: divide the odds for the relevant category by the sum of the odds for all categories (incl. the reference) within each scenario

- calculate the logodds by hand for all categories within the scenario, sum over and exponentiate
- . . . or use the `predict()` function in R

```
preds <- predict(mod.cat, newdata = data.frame(Skepsis = 5), type =
preds
```

```
##           S           A          DF           E           K
## 0.238097927 0.015061927 0.096815641 0.067125356 0.006603907 0.058
##          LA          RV          SF           V
## 0.043437070 0.100558519 0.093078122 0.280621538
```

⇒ *The probability that a respondent with moderate view on immigration votes Alternativet is 2 %*

Predicted probabilities using R

Predictions give latent probability of voting for a party, given the scenario.

- quickly many predictions

```
predict(mod.cat, newdata = data.frame(Skepsis = 0:10), type = "probs")
```

```
##             S           A          DF          E           K         KF
## 1  0.20156152 0.004111702 0.005024230 0.07102726 0.013573578 0.04142711
## 2  0.22132873 0.005661793 0.009642847 0.07458954 0.012481772 0.04715935
## 3  0.23629023 0.007579911 0.017993633 0.07615679 0.011159273 0.05219498
## 4  0.24492042 0.009852479 0.032598957 0.07549367 0.009686500 0.05608684
## 5  0.24588798 0.012403951 0.057203370 0.07248457 0.008143871 0.05837492
## 6  0.23809793 0.015061927 0.096815641 0.06712536 0.006603907 0.05859999
## 7  0.22090279 0.017523801 0.156998939 0.05956003 0.005130955 0.05636326
## 8  0.19463550 0.019362051 0.241781563 0.05018784 0.003785915 0.05148373
## 9  0.16132571 0.020124962 0.350275938 0.03978347 0.002627871 0.04423892
## 10 0.12492356 0.019542405 0.474085494 0.02946227 0.001704106 0.03551389
## 11 0.09028141 0.017710636 0.598847491 0.02036305 0.001031341 0.02660757
##            LA         RV         SF         V
## 1  0.02581331 0.37409713 0.13163687 0.1317273
## 2  0.03042339 0.30551451 0.13044641 0.1627516
## 3  0.03486177 0.24258118 0.12567952 0.1955027
## 4  0.03878487 0.18700519 0.11756233 0.2280087
## 5  0.04179347 0.13963145 0.10651356 0.2575629
## 6  0.04343707 0.10055852 0.09307812 0.2806215
## 7  0.04325535 0.06938758 0.07793232 0.2929450
## 8  0.04090670 0.04546947 0.06196735 0.2904199
## 9  0.03639232 0.02802972 0.04635204 0.2708491
## 10 0.03024714 0.01614272 0.03239172 0.2359867
## 11 0.02346832 0.00867657 0.02118576 0.1912932
```

Total backtransformation

To predict party choice, I identify the party with the highest probability within each scenario/respondent

- I let the scenario vary (or I can do in-sample prediction) and predict probabilities

```
preds <- predict(mod.cat, newdata = data.frame(Skepsis = 0:10), type = "probs")
```

- I identify the most likely outcome for scenario 1

```
## RV
##  8
```

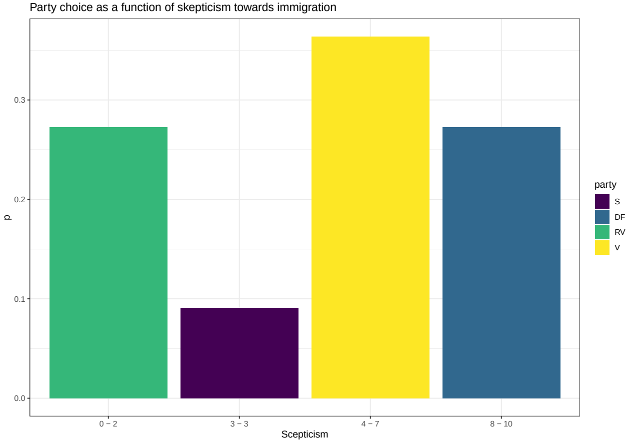
- R can also do it for me

```
preds <- predict(mod.cat, newdata = data.frame(Skepsis = 0:10), type = "class")
preds
```

```
##  [1] RV RV RV S  V  V  V  V  DF DF DF
## Levels: S A DF E K KF LA RV SF V
```

Total backtransformation (cont.)

Predicted categorical outcomes can also be illustrated in a barplot, with the predictor on the x-axis



Main assumption: IIA

Independence of irrelevant alternatives:

- there are no choices beyond what is modeled
- consistency: if we prefer A $>$ B and B $>$ C, then also A $>$ C

$\Rightarrow$ *The β does not depend on on other values of y (other alternatives).*

Testing the main assumption:

The Hausmann-McFadden test: Removes an alternative (supposed to be irrelevant) and check if β changes.

- Restricted model (a choice is removed) vs. unrestricted model (original)
- if IIA holds, then unrestricted model has smaller variance.

$\Rightarrow$ χ^2*-test with smaller value indicates IIA holds.*

Prediction testing

- **Predict outcome**
  - predicted outcome/choice is the one with the highest probability/utility
  - confusion matrix (Proportion of correct predictions: $\frac{\textit{sum of diagonal}}{\textit{N observations}}$)
- **Probability of all outcomes separately**: ROC curve and separation plots

$\Rightarrow$ *as in binomial regression, where you have one category vs. the rest*

Ordered logistic regression

What is an ordered variable?

A ranked variable with unknown distance between categories.

- Often the result of binning: Close connection to latent formulation.
- We can choose how to treat it: As linear, categorical or **ordinal**.

$\Rightarrow$ *estimate a single set of regression parameters, but keep the information on the order without assuming a continuous variable.*

Two conceptions of ordered logisitc regression

There are two ways of understanding the ordered logit:

- Latent variable: useful for interpretation.
- Parallel regressions: useful for understanding and checking estimation.

Latent variable approach: cutpoints

Cutpoints

We rely on cutpoints to slice up the latent variable and determine outcomes

- **Binomial logistic:** One cutpoint. $\rightarrow$ Rarely estimated.
- **Ordinal logistic:** Serveral cutpoints. $\rightarrow$ Explicit.

$\Rightarrow$ *Model estimates both regression parameters (β) and cutpoints (τ).*

A series of cutpoints

You are in the category m when the latent variable is between its two cutpoints: $\tau_{m-1} < y^{\star} < \tau_m$

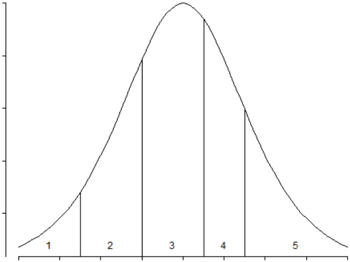


Figure 1: Slicing up a latent variable

The regression coefficients

The model calculates the odds of being lower than τ_m

- The first cutpoint (τ_0) is 0 ($-inf$): you cant be lower than the lowest.
- The last cutpoint is 1 ($+inf$): all observations are in some category.
- You end up with $m-1$ cutpoints.

The regression output

The regression output reports both β and τ

- **Regression coefficient** β is reported in relation to *upper* cutpoint of the category: $\tau_m - \beta x_i$
- **Cutpoints** serve also as intercepts.

The predicted value

The predicted probability of being in category m:

$$Pr(y_i = m) = \frac{exp(\tau_m - \beta x_i)}{1 + exp(\tau_m - \beta x_i)} - \frac{exp(\tau_{m-1} - \beta x_i)}{1 + exp(\tau_{m-1} - \beta x_i)} \tag{1}$$

An example: Attitudes towards redistribution

An example:

ESS respondents (that voted V or DF) are asked to what extent they believe the state should engage in redistribution (1 = disagree; 5 = agree).

```
#Load in data
download.file(
  url("https://siljehermansen.github.io/teaching/beyond-linear-models/kap10
  destfile = "kap10.rda"
)
df <- kap10

#Check distribution
barplot(table(df$Udjaevn))
```

An example:

ESS respondents (that voted V or DF) are asked to what extent they believe the state should engage in redistribution (1 = disagree; 5 = agree).

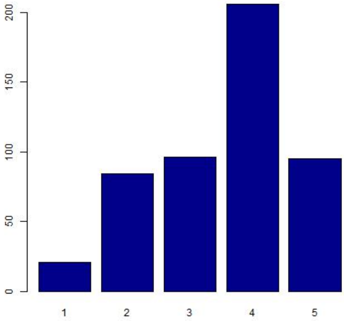


Figure 2: Attitudes towards redistribution is an ordered variable

Attitudes towards redistribution as a function of income

```r
#Library for ordinal regression
library(MASS)
#Recode into ordered factor
df$Udjaevn.ord <- as.ordered(as.factor(df$Udjaevn))
#Run regression
mod.ord <- polr(Udjaevn.ord ~ Indtaegt,
                df,
                method = "logistic",
                Hess = TRUE)
summary(mod.ord)
```

Attitudes towards redistribution as a function of income

```
## Call:
## polr(formula = Udjaevn.ord ~ Indtaegt, data = df, Hess = TRUE,
##     method = "logistic")
##
## Coefficients:
##           Value Std. Error t value
## Indtaegt 0.1153    0.03155   3.653
##
## Intercepts:
##     Value   Std. Error t value
## 1|2 -2.4186  0.2903    -8.3306
## 2|3 -0.6008  0.2179    -2.7566
## 3|4  0.3069  0.2150     1.4277
## 4|5  2.2276  0.2403     9.2686
##
## Residual Deviance: 1298.396
## AIC: 1308.396
## (51 observations deleted due to missingness)
```

We learn two things from the regression output

Regression coefficient reports effect of x on probability to be placed one category higher

- Effect in logodds: `0.115`
- We can backtransform to one unit increase in x: $(exp(\beta) - 1) \times 100$ $= 12\%$ increase in likelihood of a higher category.

⇒ *Hypothesis testing as in a binomial logit*

We learn two things from the regression output

We have one intercept per cutpoint

- e.g.: intercept of passing from 1 to 2 is `-2.419`
- e.g.: intercept is reported as significant (with standard errors)

$\Rightarrow$ *The model does a fair job in distinguishing between categories.*

Predicted scenarios

We interpret predicted probability by choosing one level of x and one category (two cutpoints) of y: What is the probability of m?

$$Pr(y_i = m) = \frac{exp(\tau_m - \beta x_i)}{1 + exp(\tau_m - \beta x_i)} - \frac{exp(\tau_{m-1} - \beta x_i)}{1 + exp(\tau_{m-1} - \beta x_i)} \tag{2}$$

Example

Let’s choose low-income respondents (x = 1) and category 3 (diff between cutpoints 2 and 3)

```
z = mod.ord$zeta
x = 1

logodds1 <- z[3] - coefficients(mod.ord) * x
logodds2 <- z[3-1] - coefficients(mod.ord) * x
## Probabilities
p1 <- exp(logodds1)/(1 + exp(logodds1)) #3/4 or lower
p2 <- exp(logodds2)/(1 + exp(logodds2)) #2/3 or lower
## Difference between cutpoints
p1 - p2 #cat 3
```

An example

Predicted proportion in category

```
paste(round((p1-p2)*100),
"% of low-income respondents are predicted to answer x = 3 ('neutral')." )
```

[1] "22 % of low-income respondents are predicted to answer x = 3 ('neutral')."

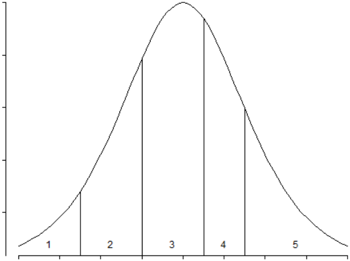
Cumulative probability

```
paste(round((p1)*100),
"% of low-income respondents are predicted to answer x = 3 ('neutral') or l
```

[1] "55 % of low-income respondents are predicted to answer x = 3 ('neutral') or lower to the question of whether they support redistribution."

Two ways of viewing the slicing

We can report the probability (e.g. `0.22`) of ending up between two cutpoints, or the *cumulative* probability (e.g. `0.55`) to be below each



point.

Exercice:

Increase the τ (z) within each value of Income (x)

```
##Create empty plot
plot(y = 0,
     x = 0,
     axes = FALSE,
     xlim = c(1,4),
     ylim = c(0,1),
     ylab = "Probability of z or below",
     xlab = "Thresholds",
     main = "Cumulative probability \nof support for redistribution",
     type = "n")
axis(1, at = 1:length(p1),
     labels = names(p1))
axis(2)
```

Exercice:

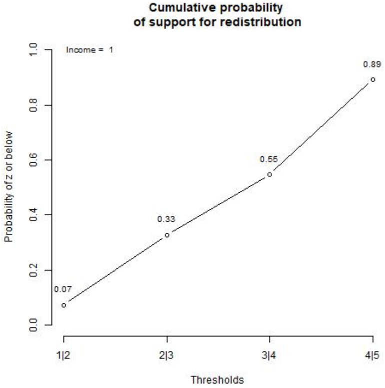
Increase the τ (z) within each value of Income (x)

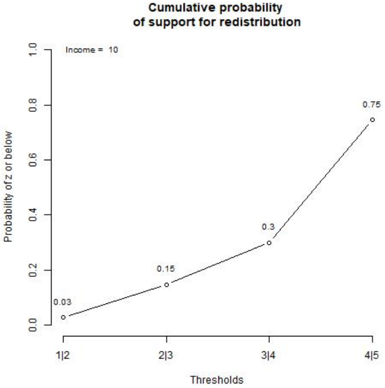
```
#Set values for prediction
x = 10 #Let this go from 1 to 10; check the shape of 10
z = mod.ord$zeta
#Logodds
logodds1 <- z - coefficients(mod.ord) * x
#Probabilities
p1 <- exp(logodds1)/(1 + exp(logodds1)) #3/4 or lower

#Plot probabilities
lines(y = p1,
      x = 1:length(p1),
      type = "b")
#Set legend (report x-value)
legend("topleft",
       bty = "n",
       cex = 0.8,
     paste("Income = ", x))

#Plot probabilities
text(y = p1+0.06,
```

Result


Cumulative probability
of support for redistribution
Income = 1
Probability of z or below
0.0
0.2
0.4
0.6
0.8
1.0
0.07
0.33
0.55
0.89
1|2
2|3
3|4
4|5
Thresholds


Cumulative probability
of support for redistribution
Income = 10
Probability of z or below
0.0
0.2
0.4
0.6
0.8
1.0
0.03
0.15
0.3
0.75
1|2
2|3
3|4
4|5
Thresholds

Parallel regressions approach: for assessment

Parallel regressions approach

The parallel regression approach is useful to understand how the model is estimated

- The y is recoded into $m-1$ dummy variables indicating if $y \leq m$
- Run a series of regressions where all β are fixed (i.e.: the same).

$\Rightarrow$ *This is also useful when we assess the model*

How good is our model?

The basic assumption

The basic assumption is that all parallel regressions have (about) the same regression coefficient

- Check the mean of the predictor for each value of y. Does it trend?

```
df %>%
  filter(!is.na(Udjaevn)) %>%
  group_by(Udjaevn) %>%
  reframe(mean(Indtaegt, na.rm = T))
```

```
## # A tibble: 5 x 2
##   Udjaevn ‘mean(Indtaegt, na.rm = T)‘
##     <dbl>                       <dbl>
## 1       1                        4.8
## 2       2                        5.58
## 3       3                        5.96
## 4       4                        6.41
## 5       5                        6.75
```

- Run parallel regressions without constraint on β. Are they similar?

An example of parallel regressions

Recode into dummies

The dummies flag cases below a cumulative threshold of *outcomes*

```
##
df$ut1 <- ifelse(df$Udjaevn > 1, 1 , 0) #2 or above
df$ut2 <- ifelse(df$Udjaevn > 2, 1 , 0) #3 or above
df$ut3 <- ifelse(df$Udjaevn > 3, 1 , 0) #4 or above
df$ut4 <- ifelse(df$Udjaevn > 4, 1 , 0) #5
```

$\Rightarrow$ *The model then runs 4 regressions where* β *reports an aggregated value from all 4 coefficients (think: weigted mean).*

Run four regressions

Let's examplify with the parallel regressions without fixed β:

```
##Parallel regressions:
mod1 <- glm(ut1 ~ Indtaegt, df, family = "binomial")
mod2 <- glm(ut2 ~ Indtaegt, df, family = "binomial")
mod3 <- glm(ut3 ~ Indtaegt, df, family = "binomial")
mod4 <- glm(ut4 ~ Indtaegt, df, family = "binomial")
```

Compare coefficients from four regressions

```
## 
## ======================================================
##                           Dependent variable:
##                   ------------------------------------
##                     ut1      ut2      ut3       ut4
##                     (1)      (2)      (3)       (4)
## ------------------------------------------------------
## Indtaegt          0.189**  0.125*** 0.110***  0.094**
##                   (0.085)  (0.041)  (0.035)   (0.045)
## 
## Constant          2.048*** 0.552**   -0.270  -2.082***
##                   (0.474)  (0.260)  (0.231)   (0.319)
## 
## ------------------------------------------------------
## Observations        459      459      459       459
## Log Likelihood    -79.653  -234.669 -303.983 -217.674
## Akaike Inf. Crit. 163.306  473.338  611.967   439.348
## ======================================================
## Note:                      *p<0.1; **p<0.05; ***p<0.01
```

Coefficient should be a weighted average from four regressions

These βs are weighted by the number of observations in each category:

```r
table(df$Udjaevn)
```

```
##
##   1   2   3   4   5
##  21  84  96 206  95
```

We can plot the βs for comparison:

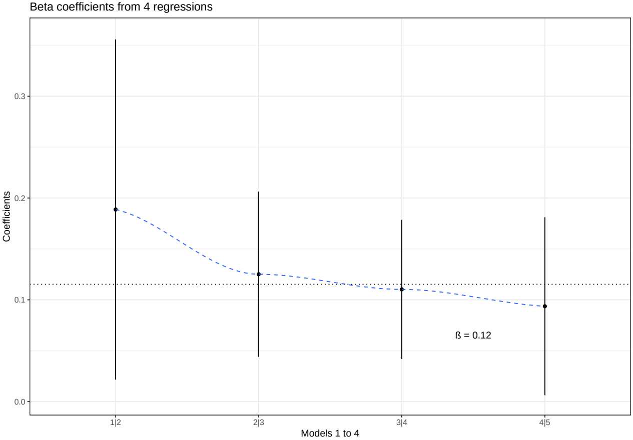
```
slope <- rbind(summary(mod1)$coefficients[2, c(1)],
               summary(mod2)$coefficients[2, c(1)],
               summary(mod3)$coefficients[2, c(1)],
               summary(mod4)$coefficients[2, c(1)])
se.slope <- rbind(summary(mod1)$coefficients[2, c(2)],
               summary(mod2)$coefficients[2, c(2)],
               summary(mod3)$coefficients[2, c(2)],
               summary(mod4)$coefficients[2, c(2)])
threshold <- c("1|2","2|3","3|4","4|5")
```

We can plot the βs for comparison:

```
data.frame(slope,
           se.slope,
           threshold) %>%
  ggplot +
  geom_point(aes(x = threshold,
                 y = slope)) +
  geom_errorbar(aes(x = threshold,
                    ymax = slope + 1.96 * se.slope,
                    ymin = slope - 1.96 * se.slope),
                width = 0) +
  geom_hline(yintercept = mod.ord$coefficients,
             lty = 3) +
  geom_text(aes(y = mod.ord$coefficients-0.05,
                x = 3.5,
                label = paste("\u03b2 =", round(mod.ord$coefficients,2))
                ),
            parse = F) +
  labs(title = "Slope coefficients from 4 regressions") +
  ylab("Coefficients") +
  xlab("Models 1 to 4")
```

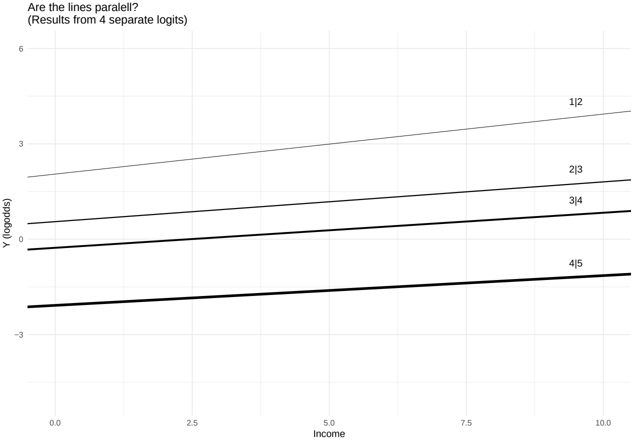
We can plot the βs for comparison:

The overall β is `0.12`. If the ordered model describes the data well, then all the unconstrained βs should resemble that description.



A visual inspection

A more visual way of checking the "parallel lines assumption" is to inspect if the regression lines are parallel.



When is it smart to run an ordered logit?

- You have few ordered categories
- The effect is approximately the same across the categories (parallel lines assumption)

What do I do if the assumption doesn't hold?

- Run an OLS/linear model:
  - if you have many categories
  - fairly equal spread of observations between categories
- Run a multinomial model:
  - i.e. estimate different β for each regression/threshold