Ordinal regression models made easy. A tutorial on parameter interpretation, data simulation, and power analysis.

Filippo Gambarota[1] & Gianmarco Altoè[1]

[1] Department of Developmental Psychology and Socialization, University of Padova, Italy

Author Note

Add complete departmental affiliations for each author here. Each new line herein must be indented, like this line.

Enter author note here.

The authors made the following contributions. Filippo Gambarota: Conceptualization, Writing - Original Draft Preparation, Writing - Review & Editing; Gianmarco Altoè: Writing - Review & Editing, Supervision.

Correspondence concerning this article should be addressed to Filippo Gambarota, Postal address. E-mail: filippo.gambarota@unipd.it

Abstract

Abstract here

Keywords: ordinal, likert, simulations, power

Word count: X

Ordinal regression models made easy. A tutorial on parameter interpretation, data simulation, and power analysis.

Introduction

Psychological research make an extensive use of ordinal data. One of the main reason is probably the usage of Likert scales (Likert, 1932). Ordinal data refers to a specific type of measurement scale (Stevens, 1946) where ordered numbers are assigned to a variable. Compared to nominal scale and as the name suggest the labels are ordered. Compared to interval or ratio scales there is no explicit assumption about the distance between labels. An example is asking people the degree of agreement about a certain statement using a scale from 1 (no agreement) to 7 (total agreement). Answering 4 compared to 2 suggest an higher agreement but we cannot affirm that there is two times the agreement compared to the second answer. Stevens (1946) and Kemp and Grace (2021) suggested that for ordinal variables is appropriate to calculate ranks-based descriptive statistics (e.g., median or percentiles) instead of metric statistics (e.g., mean or standard deviation) and using appropriate inferential tools (Agresti, 2010; e.g., N. Cliff, 1996). This distinction in terms of the appropriateness of certain descriptive statistics is also relevant when modeling data. Treating ordinal data as metric refers to assuming the labels as actual integer numbers thus assuming a fixed and know distance between levels (Liddell & Kruschke, 2018). More generally, Norman Cliff (2016) suggest that most of the research questions in behavioral sciences can be considered as ordinal (*is the score* x higher than the score y?) concerning variables where the most appropriate measurement scale is probably ordinal.

In Psychology especially when using item-based measures (questionnaires, surveys, etc.) the common practice is using a normal linear regression that makes an explicit assumption about metric features of the response variable. Liddell and Kruschke (2018) reviewed the psychological literature using likert-based measures and reported how the majority of papers used metric-based statistical models. In the same work, Liddell and

Kruschke (2018) showed extensive examples and simulations about the potential pitfalls of treating an ordinal variable as metric (but see Robitzsch, 2020 for an alternative perspective). They reported problems in terms of lack of power, inversion of the effects (e.g., finding a negative effect when the true effect is positive) and distorted effect size estimates. Some authors suggested that individual ordinal items and a collection of ordinal (averaged or summed) items can safely be considered and analyzed as metric (Carifio & Perla, 2008; Carifio & Perla, 2007; but see Jamieson, 2004). Despite Liddell and Kruschke (2018) provide some example that averaging ordinal items and applying metric models is not appropriate, in the current paper we discuss only cases where there is a single ordinal outcome (e.g., a single item, question, etc.).

Ordinal regression models

Despite the actual modeling proposal by Liddell and Kruschke (2018), there is a class of regression models taking into account the ordinal nature of the response variable without metric assumptions. We can name this general class of models as *ordinal regression*. The actual statistical nomenclature can be confusing mainly because there are several types of models with different assumptions and structures (Tutz, 2022). Tutz (2022) and Bürkner and Vuorre (2019) provide a clear and updated taxonomy of ordinal regression models. We can identify three main classes: *cumulative models* (Agresti, 2010; CM, McCullagh, 1980), *sequential models* (Tutz, 1990) and *adjacent category models*. The cumulative is the mostly used model assuming the existence of a latent variable that categorized using a set of thresholds produces the observed ordinal variable.

The *sequential model* as suggested by the name is appropriate when modelling sequential processes. Assuming to have five response options, the sequential model assume that responding “3” assume a sequential process where steps “1” and “2” are already reached. A clear example is proposed by Bürkner and Vuorre (2019) where the marriage duration in years is predicted as a function of some explanatory variables. For each level of

the response variable there is a latent distribution where the step between a marriage year $k = 1$ and the next years $k > 1$ is modeled by the sequential model. When comparing k with $k > 1$, everything lower than k is assumed to be already reached (Tutz & Berger, 2020). The adjacent category model compare the category k with $k + 1$ still assuming a latent distribution for each k. As suggested by Tutz (2022) the adjacent-category model can be seen as a series of binary binomial regressions taking into account the order of the categories. Bürkner and Vuorre (2019) suggested that adjacent-category model can be chosen for its mathematical convenience and there is no a clear empirical distinction as for the cumulative vs sequential model.

In the current paper we put the focus on the CM for several reasons. The first reason is that the latent formulation of the model is particularly convenient both for parameter interpretation and data simulation. The second reason is that several psychological variables can be formalized as a latent continuous variable observed as an ordinal item. Furthermore, CM are also used to model data under a signal detection theory framework (e.g., DeCarlo, 2010). The Figure 1 depict the overall structure of the *cumulative* model.

Model notation

In this section we introduce some notation for the CM that is used through the paper and in the R code. We proposed a notation as consistent as possible with the literature and the `ordinal` R package used in the tutorial. We define Y_k as the observed ordinal response with $1, \dots, k$ levels and $Y^\star$ is the underlying latent variable. The latent variable is segmented using $k - 1$ thresholds $\alpha_1, \dots, \alpha_{k-1}$. Similarly to the generalized linear models framework, we define $g(x) = \eta$ as the link function that maps probabilities into the linear predictor η. To transform back η into probabilities we use the inverse of the link function $x = g^{-1}(\eta)$. The specific link function define the type of model and require a different R function. For example, when assuming a Gaussian distribution we are fitting a model with a `probit` link function is the cumulative distribution function $g(x) = \Phi^{-1}(x) = \eta$ and the

inverse of the link function is the inverse cumulative distribution function (or quantile function) defined $x = g^{-1}(\eta) = \Phi(\eta)$. When modelling an ordinal variable in a cumulative link model we actually modelling the cumulative probability $P(Y \leq k), k = 1, ..., k-1$[1]. Equation (1) depict the general cumulative model including predictors $\mathbf{X}$ and regression coefficients β. The minus sign in $\mathbf{X}\beta$ is used to interpret the β as in the standard regression model (Agresti, 2010) where higher β values corresponds to increased probability of responding higher k categories.

$$P(Y \leq k) = g^{-1}(\alpha_k - \mathbf{X}\beta), \quad k = 1, ..., k-1 \tag{1}$$

The $\mathbf{X}\beta$ is the linear predictor η that is the cumulative probability $P(Y \leq k)$ transformed using the link function $g()$. To obtain the probability of a single outcome $P(Y = k)$ we can compute the difference between cumulative probabilities as shown in Equation (2).

$$P(Y = k) = g^{-1}(\alpha_k - \eta) - g^{-1}(\alpha_{k-1} - \eta), \quad k = 1, ..., k-1 \tag{2}$$

There are always two special cases when computing the probability of a single outcome Y that is when $Y = 1$ and $Y = k$. In the first case the cumulative probability is calculated from $-\infty$ that is the same as temporary assuming an $\alpha_0 = -\infty$. In the second case ($Y = 1$) the probability is calculated as $P(Y = k) = 1 - g^{-1}(\alpha_{k-1} - \eta)$ that is the same as assuming a temporary threshold $\alpha_k = +\infty$. The Figure 1 shows how the single probabilities of the ordinal outcome are calculated from cumulative probabilities.

[Figure 1 about here]

[1] As done by Agresti (2010), when referring to $P(Y \leq k)$ we are implicitly conditioning on a particular x value $P(Y \leq k|x_i)$

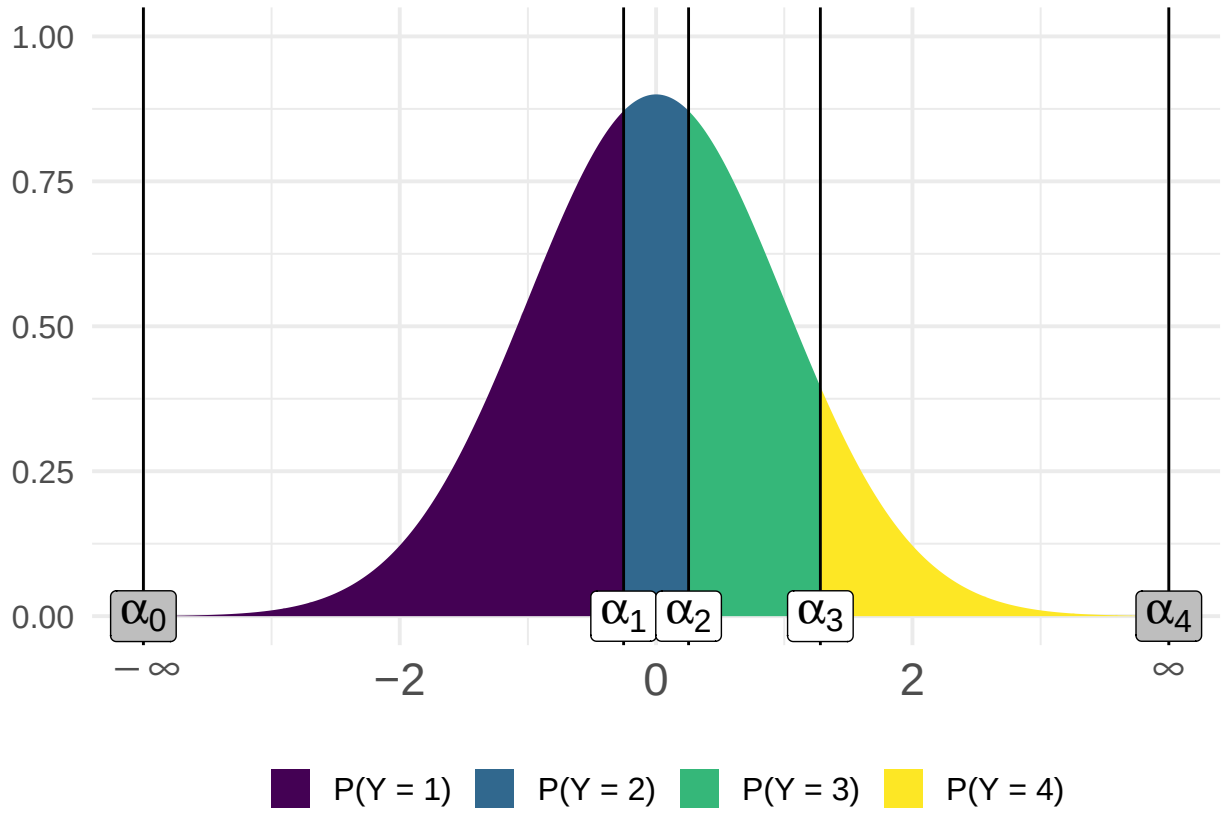

Figure 1. caption here

The same model can be written in the so-called latent formulation as reported in Equation (3). The model is no longer directly about the cumulative probabilities but focused on the continuous latent variable $Y^\star$ as a function of the linear predictor $\eta = \mathbf{X}\beta$ similar to a standard linear regression.

$$Y_i^\star = \eta + \epsilon_i \tag{3}$$

The crucial part is ϵ_i that is the random component of the model coming from a certain probability distribution. For a *probit* model, errors are sampled from a standard Gaussian distribution while for a *logit* model from a standard logistic distribution. Following the notation by Tutz (2022), the observed ordinal value $Y_i = k$ comes from $Y_i^\star$ belonging to the interval defined by the thresholds $Y_i = k \iff \alpha_{k-1} < Y_i^\star < \alpha_k$ where

$-\infty = \alpha_0 < \alpha_1 < \cdots < \alpha_{k-1} < \alpha_k = \infty$.

In the basic version of the model, the thresholds α_k are considered as fixed and being part of the measurement procedure (Liddell & Kruschke, 2018) and do not vary as a function of the predictors. In a more sophisticated version of the model called location-shift (Tutz, 2022), both the location μ and the thresholds α_k can vary as a function of the predictors. Tutz (2022) described also another version of the model called location-scale (Cox, 1995; Rigby & Stasinopoulos, 2005; Tutz, 2022) where the location μ and the scale σ^2 of the distribution can vary as a function of the predictors.

The model can be also formalized in alternative ways. Liddell and Kruschke (2018) and Kruschke (2015) proposed a Bayesian version of the model with a different threshold parametrization. Gelman, Hill, and Vehtari (2020) proposed three alternative parametrizations focusing on different definition of the thresholds.

Link function

The cumulative link model implemented in Equations (1) and (3) can be considered the general formulation that requires specifying the link function $g(\cdot)$ or the errors distribution $\epsilon_i \sim D(\mu, \sigma^2)$. Among several available functions the *logit* and *probit* models are the most common. The *logit* model assume a *logit* link function and a logistic distribution as latent variable. On the other side, the *probit* model assume a Gaussian distribution.

The two models provide similar results with a different parameters interpretation. In the next sections we will illustrate the differences and simulation strategies. Figure 2 depicts the two distributions while Table 1 summarise the presented cumulative models with the proposed link function and the corresponding R code.

In terms of parameters, both distributions can be defined with a location μ and a scale s parameter. The standard normal distribution has $\mu = 0$ and $s = 1$. Furthermore

the variance corresponds to the scale $s^2 = \sigma^2 = 1$. The variance of the logistic distribution is $\sigma^2 = \frac{s^2\pi^2}{3}$. The standard logistic distribution has $\mu = 0$ and $s^2 = 1$ thus the standard deviation simplified to $\frac{\pi}{\sqrt{3}} \approx 1.81$. In practical terms, fixing μ and s lead to an higher standard deviation for the logistic distribution.

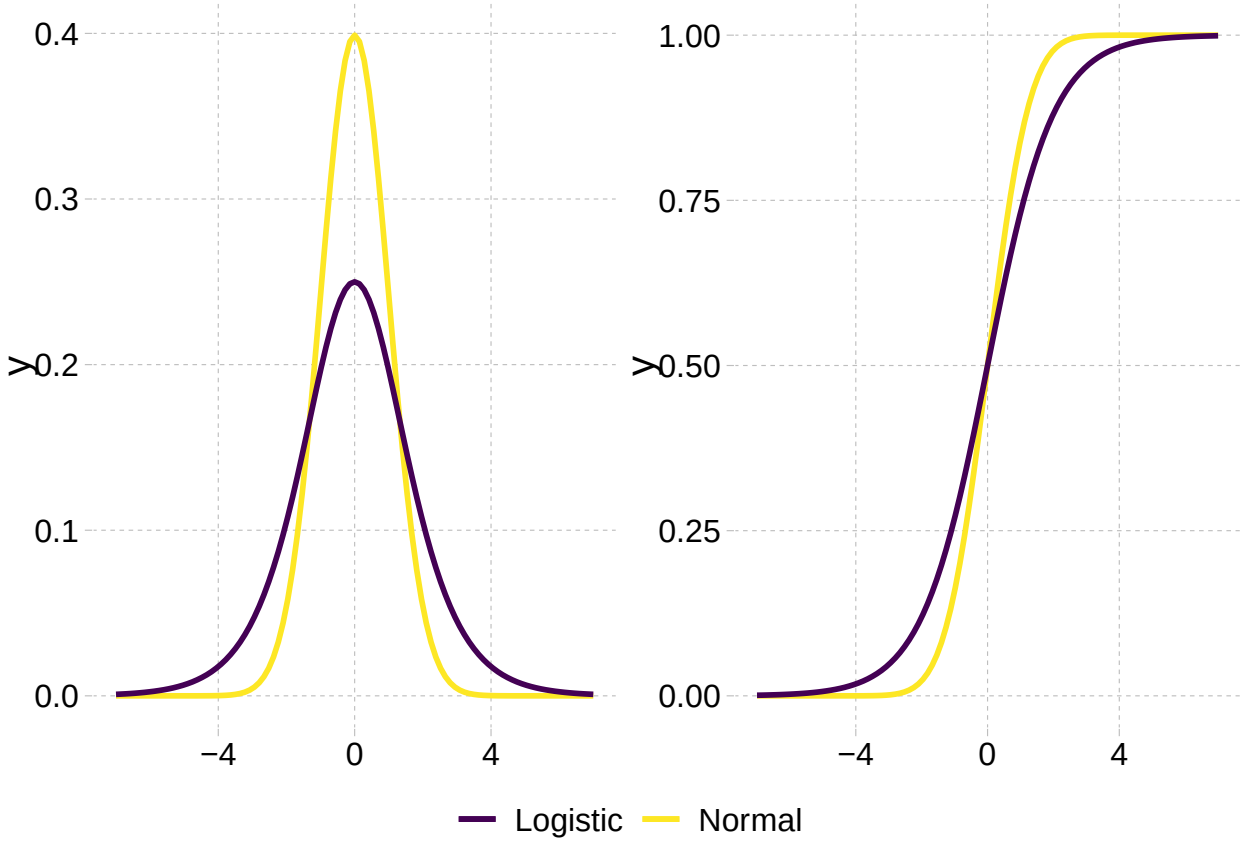

Figure 2. caption here

[Figure 2 about here]

Model fitting

For fitting the cumulative models we used the `ordinal` package (Christensen, 2019). Despite the presence of other possibilities the `ordinal` package provide the most complete and intuitive way to implement the ordinal models. The syntax is very similar to standard

Table 1

caption here

	Link Function		Inverse Link Function	
Model	Equation	R Code	Equation	R Code
Cumulative Logit	$\text{logit}(p) = \log(p/(1-p))$	‘qlogis()‘	$e^{\text{logit}(p)}/(1+e^{\text{logit}(p)})$	‘plogis()‘
Cumulative Probit	$z = \Phi(p)$	‘qnorm()‘	$\Phi^{-1}(z)$	‘pnorm()‘

linear models in R and default functions to calculate predictions, perform model comparison, extract relevant model information are implemented similarly to standard regression modelling.[2].

The function to fit the model is `clm()` and the model equation is specified using the R formula syntax `y ~ x` where `y` is the ordinal dependent variable and `x` is one or more predictors eventually including also the interactions. The package also implements mixed-effects models (see the `clmm()` function) including random intercepts and slopes but the topic is beyond the scope of the current tutorial.

When fitting the model the crucial arguments are the `formula`, the `link` function and the `data`. More advanced arguments are the `nominal`, `scale` and `threshold`.

- `formula`: the formula `y ~ x` with the dependent variable and predictors.
- `link`: is the link function. In this tutorial we consider only the *logit* and *probit* link but other link functions are available.
- `data`: is the dataset where with the variables included in the `formula`
- `nominal`: formula with predictors where the proportional odds assumption (See Section) is relaxed (i.e., partial or non proportional odds)

[2] For a very complete overview of the ordinal package see Christensen (2019) and the Package documentation https://cran.r-project.org/web/packages/ordinal/ordinal.pdf

- `scale`: formula with predictors for the scale (standard deviation) parameter. This argument allow to fit a scale-location model (see Section). The main `formula` argument refers to predictors on the location parameter (i.e., the mean μ).
- `threshold`: different structures for estimating the thresholds. The default is `threshold = "flexible"` where $k-1$ threshold (where k is the number of ordinal levels for Y) are estimated.

We can start by fitting a simple model, highlighting the crucial parameters where the detailed explanation will be expanded in the next sections. Table contains simulated data from $n = 100$ participants rating the agreement about a certain item with $k = 4$ ordered options. The participants are divided into two groups (predictor x). We can fit a cumulative link model with `clm()` function and check the model summary.

Table 2

caption here

group	mean	median	sd	Y1	Y2	Y3	Y4
a	2.42	2	1.13	14 (p = 0.28)	12 (p = 0.24)	13 (p = 0.26)	11 (p = 0.22)
b	3	3	1.16	9 (p = 0.18)	6 (p = 0.12)	11 (p = 0.22)	24 (p = 0.48)

```
fit <- clm(y ~ x, data = dat, link = "logit")
summary(fit)
#> formula: y ~ x
#> data:    dat
#>
#>  link  threshold nobs logLik  AIC    niter max.grad cond.H
#>  logit flexible  100  -132.25 272.51 4(0)  1.08e-08 2.0e+01
#>
#> Coefficients:
#>    Estimate Std. Error z value Pr(>|z|)
#> xb   0.9587     0.3718   2.579  0.00992 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#>
#> Threshold coefficients:
#>     Estimate Std. Error z value
#> 1|2  -0.8103     0.2814  -2.879
#> 2|3   0.0672     0.2636   0.255
#> 3|4   1.1158     0.2900   3.848
```

The two main sections of the model summary are the *Coefficients* section reporting the regression coefficients β and the *Threshold* section reporting the α estimation. Given that $k = 4$ we have $k - 1 = 3$ thresholds and a single β associated with the x effect. As in standard regression models, when x is a categorical predictor with j level, we will estimate $j - 1$ regression coefficients where the interpretation depends on the contrast coding [see Schad, Vasishth, Hohenstein, and Kliegl (2020)][3]

Interpreting parameters

Logit Model. Odds and odds ratio

To understand the logit model we need to introduce odds and odds ratio. The odds of a probability p is defined as $\frac{p}{1-p}$ thus the probability of success divided by the probability of failure. The odds takes value ranging from 0 to ∞. For example with a probability of $p = 0.8$ we have an odds of 4, thus we have a 4 successes for each failure. The same as having $p = 0.2$ and an odds of 0.25 means that for each 0.25 successes we have a failure or that we have 4 failures for each success. When comparing two groups or conditions we can compare the two odds calculating an odds ratio. The odds ratio is the mostly used statistics to compare groups or conditions on a binary outcome. An odds ratio of 4 means that the odds of success at the numerator is 4 times higher than the odds of success at the

[3] In R the default is the dummy coding where a factor of j levels is converted into $j - 1$ dummy variables. By default, the first level of the factor is taken as the reference level and the $j - 1$ coefficients represent the comparison between the other levels and the reference.

denominator. The Figure 3 shows the relationship between probabilities and odds. The logit transformation is about taking the logarithm of the odds creating a symmetric function ranging from $-\infty$ to ∞ with $p = 0.5$ as the midpoint because $\log(\frac{0.5}{1-0.5}) = 0$. The standard logistic regression with two outcome model the logit transformed probability.

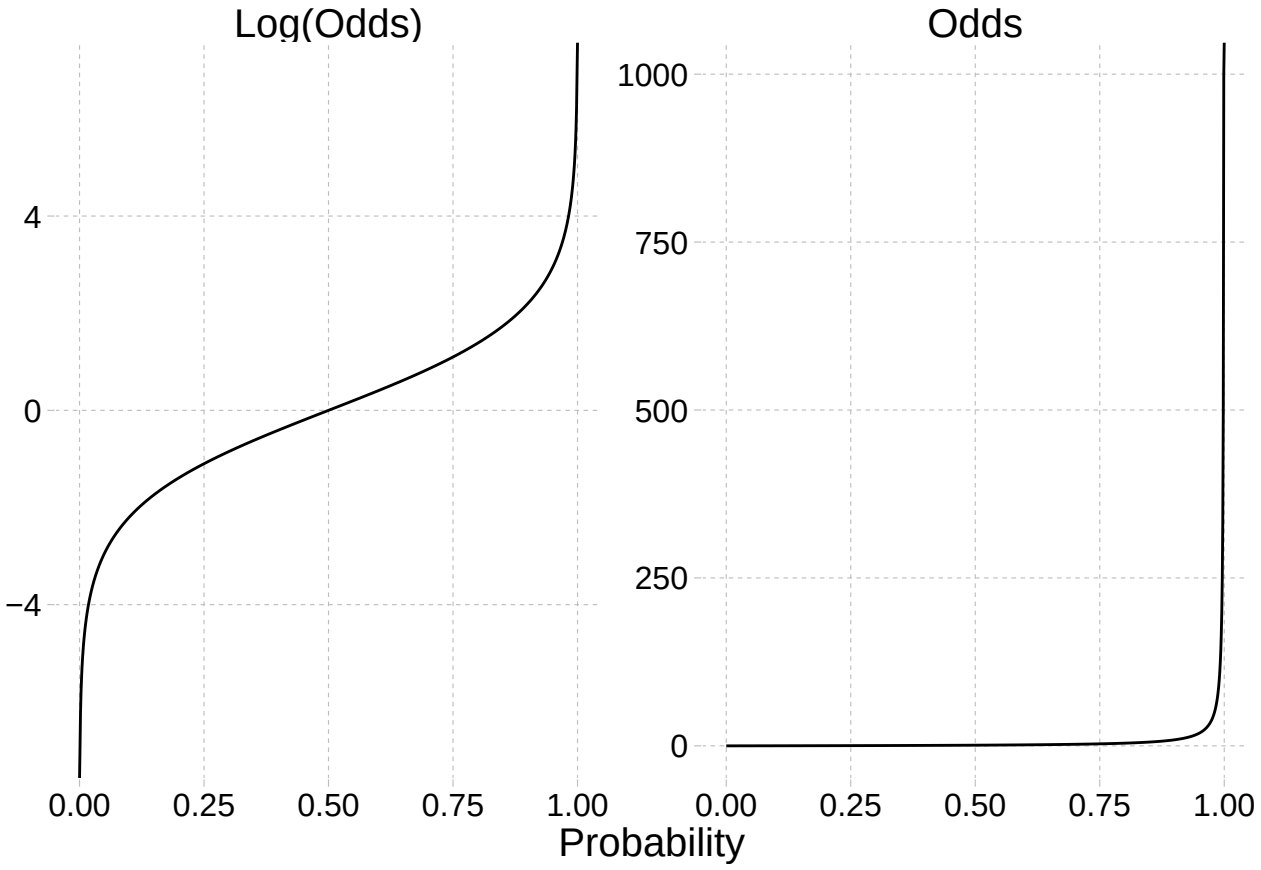

Figure 3. caption here

[Figure 3 about here]

Odds and odds ratios are clearly defined with $k = 2$ outcomes. With an ordinal variable and a cumulative model we can use the cumulative odds ratio. With e.g. $k = 4$ outcomes we have $k - 1$ models determined by the cumulative probability in terms of $P(Y \leq 1), \ldots, P(Y \leq k - 1)$. We can manually calculate the odds ratio starting from the previous dataset.

```r
# function to calculate odds
odds <- function(p) p / (1 - p)

# probability of Y = 1 for the group a
dat$yn <- as.integer(dat$y) # better as number here
(p1a <- mean(dat$yn[dat$x == "a"] == 1))
#> [1] 0.28

# thus P(Y > 1) = 1 - p1a
odds(p1a)
#> [1] 0.3888889

# odds ratio Y = 1 a vs b
(p1b <- mean(dat$yn[dat$x == "b"] == 1))
#> [1] 0.18
odds(p1a) / odds(p1b)
#> [1] 1.771605
```

For example the probability of responding $Y = 1$ in the group "a" is 0.28 corresponding to an odds of 0.39. When calculating the odds ratio comparing "a" vs "b" for $Y = 1$ we obtain that the group "a" has 1.77 times the odds of responding $Y = 1$ compared to the group "b".

For an ordinal variable we have k levels and we can calculate $k - 1$ cumulative probabilities (excluding the last associated with $p = 1$). Thus we have $k - 1$ odds ratio or what is called a cumulative odds ratio. Basically we repeat the previous steps but on the cumulative probability $P(Y \leq k), k = 1, \ldots, k - 1$. The three odds ratios are very similar correspond to the model estimate e_1^{β}, in R `exp(fit$beta)` 2.61. In fact, the logit model essentially estimate the cumulative odds ratio.

```r
# the dummy_ord() is a custom function creating k - 1 dummy variable for the cumulative probabilities

# data.frame with the group (x) and the k - 1 dummy variables
cum_p <- cbind(x = dat$x, dummy_ord(dat$y))

# calculating the cumulative probability for k - 1 variables.
# Taking the average of a series of 0-1 is the same as computing the proportion of 1s.
```

```r
(group_a <- apply(cum_p[cum_p$x == "a", -1], 2, mean))
#> y1vs234 y12vs34 y123vs4
#>    0.28    0.52    0.78
(group_b <- apply(cum_p[cum_p$x == "b", -1], 2, mean))
#> y1vs234 y12vs34 y123vs4
#>    0.18    0.30    0.52

# calculating k - 1 odds ratios on the cumulative probabilities as a/b
odds(group_a) / odds(group_b)
#>  y1vs234  y12vs34  y123vs4
#> 1.771605 2.527778 3.272727
```

Proportional odds assumption

From the previous example, we noticed that the $k-1$ odds ratios are very similar (the similarity increase if n increase). This is not a coincidence but an implicit (until now) assumption made by the CM called *proportional odds* (POA). Following again the taxonomy by Tutz (2022), each of the presented ordinal regression model has a basic version making the (POA). There are more advanced versions of the model relaxing this assumption completely (*non proportional odds*) and partially (*partial proportional odds*). The proportional odds assumption is a crucial point when interpreting model parameters. Can be formalized as in Equation ().

Basically if we use the *logit* link function $g()$ the βs are interpreted as standard odds ratio in the logistic regression. Given that we have $k > 2$ alternatives there is the need of $k-1$ equations. The POA is formalized in Equation (4). Basically the cumulative log odds ratio comparing $P(Y_k|x_0)$ with $P(Y_k|x_1)$ is the same regardless the specific k or threshold α.

The proportional odds is assuming that the regression coefficients β are independent from the thresholds α thus the effect is the same regardless of the Y level. Thus $\beta_1 = \beta_2 \cdots = \beta_{k-1}$. The Figure (4) depicts the proportional odds assumption for the $k-1$ logistic curves both for probabilities and linear predictors η.

$$\text{logit}(\frac{P(Y \leq 1|x_1)}{P(Y \leq 1|x_0)}) = \cdots = \text{logit}(\frac{P(Y \leq k-1|x_1)}{P(Y \leq k-1|x_0)}) \tag{4}$$

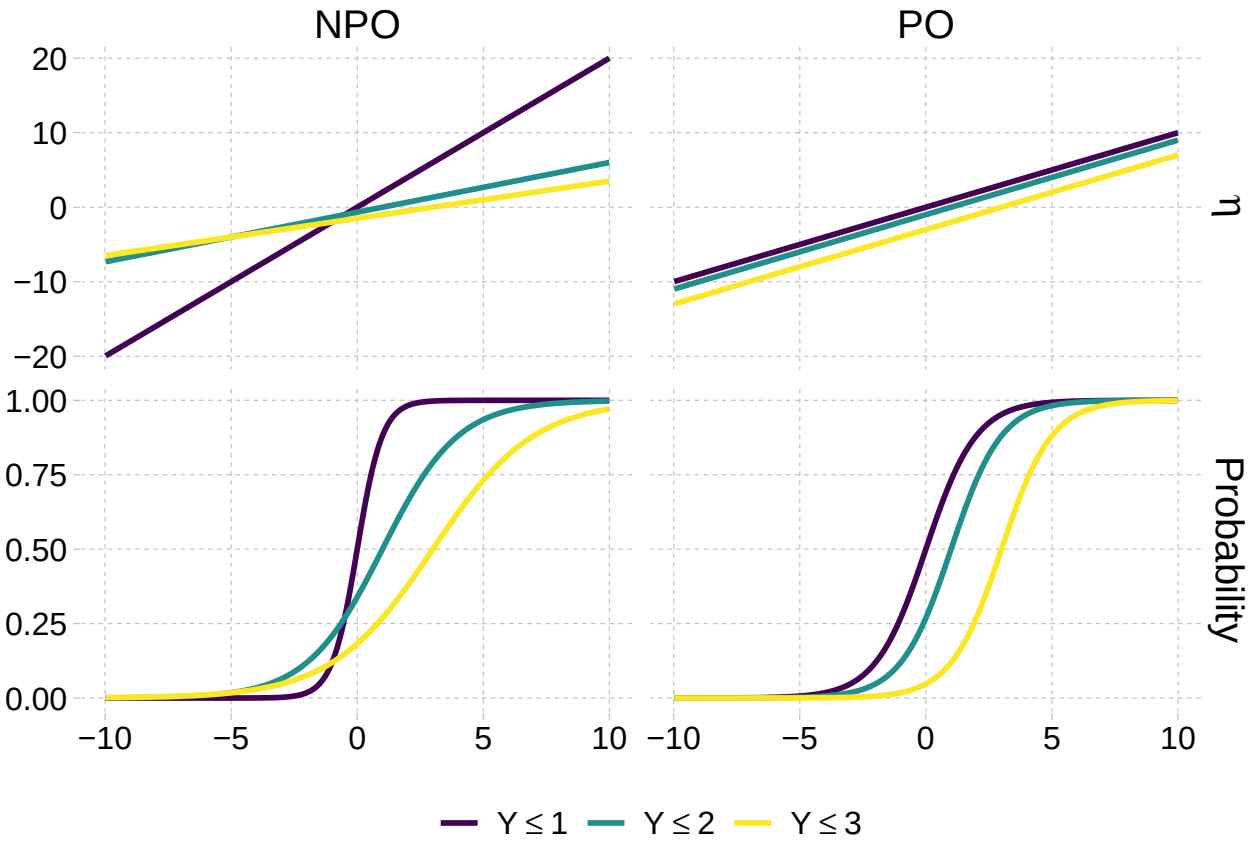

Figure 4. caption here

[Figure 4 about here]

The proportional odds assumption is convenient because regardless of k, the β_j (j being the number of regression coefficients) effect is assumed to be the same. The model is more parsimonious compared to estimating $k-1$ coefficients for each β_j as in the multinomial regression or the non-proportional model.

Given the relatively strong assumption, there are several methods for testing if data are supporting the POA (see Liu, He, Tu, & Tang, 2023 for an overview). Tutz and Berger (2020) suggested a trade-off between assuming/relaxing the POA in terms of implementing

location-shift or *location-scale* models. Basically these methods should guarantee more flexibility in modelling the observed probabilities reducing the number of parameters. However, these methods and models are outside the scope of the tutorial. In the next section, we provide more details only about the *location-scale* model in terms of parameters interpretation and data simulation.

This can be shown by fitting the model with `clm()` that by default assume the POA and using the predicted probabilities to compute the odds ratios. In the following code we computed the odds ratio comparing x_a and x_b when $k \leq 1$ and $k \leq 2$.

```
# fitting the model
fit <- clm(y ~ x, data = dat, link = "logit")

# extracting the predicted probabilities for the two groups
pr <- predict(fit, data.frame(x = unique(dat$x)))$fit

# y <= 1
y1a <- pr[1, 1]
y1b <- pr[2, 1]

# y <= 2
y12a <- sum(pr[1, 1:2])
y12b <- sum(pr[2, 1:2])

# odds ratio y <= 1 (a vs b) VS y <= 2 (a vs b)
odds(y1a) / odds(y1b)
#> [1] 2.608304
odds(y12a) / odds(y12b)
#> [1] 2.608304
```

In this case the odds ratio is exactly the same because the model (thus the predicted probabilities) assume the POA. When estimating the $k-1$ odds ratio from the data results could be different. In fact, the discrepancy between the odds ratio is an index of POA violation. metti qualche riferimento, magari a supplementary.

Proportional odds and *probit* model. The proportional odds assumption is relevant only for the *logit* model because of the link function that put β as log odds. However the concept of the proportional odds that is more generally called parallel slopes assumption is assumed also for the probit model. Equation (5) depict the parallel slopes assumption for a probit model. Basically the difference in z for a unit increase in x (i.e., the slope) is the same regardless the Y level. This can be directly demonstrated in the same way as for the logit model. We use the same dataset of the previous simulation. The difference between the z scores comparing the two groups calculated on the cumulative probability is the same regardless the $k-1$ level.

$$z_{x_1-x_0} = \Phi(P \leq 1|x_0) - \Phi(P \leq 1|x_1) = \cdots = \Phi(P \leq k-1|x_0) - \Phi(P \leq k-1|x_1) \quad (5)$$

```r
# fitting the model
fit <- clm(y ~ x, data = dat, link = "probit")

# extracting the predicted probabilities for the two groups
pr <- predict(fit, data.frame(x = unique(dat$x)))$fit

# y <= 1
y1a <- pr[1, 1]
y1b <- pr[2, 1]

# y <= 2
y12a <- sum(pr[1, 1:2])
y12b <- sum(pr[2, 1:2])

# z score difference y <= 1 (a vs b) VS y <= 2 (a vs b)
qnorm(y1a) - qnorm(y1b)
#> [1] 0.5644823
qnorm(y12a) - qnorm(y12b)
#> [1] 0.5644823
```

Probit Model z scores

The alternative to fitting a cumulative logit model is using a standard normal distribution thus fitting a *probit* model. The main difference is that the latent distribution is now a Gaussian distribution with $\mu = 0$ and $\sigma^2 = 1$ and the link function is the cumulative Gaussian distribution function Φ implemented in R with the `pnorm()` function. The inverse of the Gaussian cumulative distribution function is Φ^{-1} implemented in R with the `qnorm()` function. The logistic and normal distribution are very similar, with the logistic having more variance.

The main difference regards the parameters interpretation. In the logit model the β are the log odds ratio. For categorical variables they represents the increase in the log odds of moving from one level to the other while for numerical variables is the increase in the log odds for a unit increase in x.

For the *probit* model, the β is the increase in terms of z scores for a unit increase in x. This is very convenient especially for categorical variables because parameters can be interpreted as a Cohen's d like measure. Thinking about the latent distributions, the β is the shift in the latent mean comparing two or more groups or the slope of latent scores as a function of a numeric x. More formally the shift in the latent distribution is $\frac{\beta}{\sigma}$ (for the *probit* model $\sigma = 1$). The interpretation in terms of shifting the latent mean holds also for the logistic model. However, the standard deviation of the standard logistic regression is $\frac{\pi}{\sqrt{3}} \approx 1.81$[4]. The β for the logistic distribution can be interpreted as the location shift of the latent logistic distribution by $\beta/(\frac{\pi}{\sqrt{3}})$ standard deviations.

---

[4] Actually the variance of the logistic distribution is $\frac{s^2\pi^2}{3}$ and the standard deviation $\frac{s\pi}{\sqrt{3}}$ where s is the scale of the distribution. For the standard logistic distribution $s = 1$ (as for the standard normal distribution).

Simulating data

In this tutorial we present two method for simulating ordinal data. The first method concerns calculating the probabilities of each k of Y as a function of predictors and then sampling from a multinomial distribution. The second method simulate the latent variable $Y^\star$ as a function of predictors.

Simulating from a multinomial distribution. For the first method we need to calculate $g^{-1}(\eta)$ as a function of predictors and then sample from a multinomial distribution[5]. This is similar to the general method to simulate data for a generalized linear model where the linear predictor η are calculated and data are generated from the corresponding probability distribution of the random component.

As a simple example, let's simulate two groups or condition with $n = 100$ participants responding to an item with $k = 4$ ordered options. We simulate that the second group has more probability of responding higher categories of Y. As explained in the previous sections, we can summarise the effect size of a CM (assuming POA) using a single $\beta = \log(OR)$. We can assume that the odds ratio is $OR = 2$. For simplicity, the probabilities of the first group $x = 0$ are uniform thus $P(Y = 1|x_0) = ...P(Y = k|x_0) = 1/k$. The following code summarise the first steps of the simulation. Basically we define the simulation parameters, we calculate the $k - 1$ thresholds α and we apply Equations (1) and (2). In this way we calculated the true probability of each Y level for the two groups.

An important step is converting from α to probability and the opposite. This steps are implemented in the `alpha_to_prob()` and `prob_to_alpha()` functions. Basically given the input and the link function, these functions return the thresholds associated with k probabilities or the $k - 1$ thresholds.

---

[5] In R we are using the `sample()` function that simulate values from a *categorical* distribution. The *categorical* distribution is considered a special case of the multinomial distribution

```r
k <- 5
(p <- rep(1/k, k)) # uniform
#> [1] 0.2 0.2 0.2 0.2 0.2
names(p) <- paste0("y", 1:k)

(alpha <- prob_to_alpha(p, link = "logit")) # or prob_to_alpha(p, "probit")
#>         y1         y2         y3         y4
#> -1.3862944 -0.4054651  0.4054651  1.3862944
alpha_to_prob(alpha, link = "logit")
#> [1] 0.2 0.2 0.2 0.2 0.2
```

```r
## SIMULATION PARAMETERS

N <- 100 # sample size
or <- 2 # odds ratio
k <- 4 # number of ordinal alternatives
probs0 <- rep(1/k, k) # probabilities for the first group
alpha <- prob_to_alpha(probs0, link = "logit")
dat <- data.frame(x = rep(c(0, 1), each = N/2))

## LINEAR PREDICTOR

# calculate linear predictor using equation TODO put equation
# obtaining k - 1 equations

lp <- lapply(alpha, function(a) a - log(or) * dat$x)
names(lp) <- sprintf("lp_leq%s", 1:(k - 1)) # giving appropriate names
lp <- data.frame(lp)
head(lp)
#>     lp_leq1 lp_leq2  lp_leq3
#> 1 -1.098612       0 1.098612
#> 2 -1.098612       0 1.098612
#> 3 -1.098612       0 1.098612
#> 4 -1.098612       0 1.098612
#> 5 -1.098612       0 1.098612
#> 6 -1.098612       0 1.098612

## CUMULATIVE PROBABILITIES

# apply the inverse of the link function (invlogit) to calculate cumulative probabilities
cump <- lapply(lp, plogis)
```

```r
cump <- data.frame(cump)
names(cump) <- sprintf("cump_leq%s", 1:(k - 1)) # giving appropriate names
head(cump)
#>   cump_leq1 cump_leq2 cump_leq3
#> 1      0.25       0.5      0.75
#> 2      0.25       0.5      0.75
#> 3      0.25       0.5      0.75
#> 4      0.25       0.5      0.75
#> 5      0.25       0.5      0.75
#> 6      0.25       0.5      0.75

## PROBABILITIES OF Y

# for each row, we can calculate P(Y = k) using equation TODO put equation
# P(Y = 1) = P(Y <= 1)
# P(Y = 2) = P(Y <= 2) - P(Y <= 1)
# P(Y = 3) = P(Y <= 3) - P(Y <= 2)
# P(Y = 4) = 1 - P(Y <= 3)

# adding a columns of 0 and 1, then diff() for adjacent differences
cump <- cbind(0, cump, 1)
p <- apply(cump, 1, diff, simplify = FALSE)

p <- data.frame(do.call(rbind, p)) # collapse list of rows into a dataframe
names(p) <- sprintf("p%s", 1:k) # giving appropriate names
head(p)
#>     p1   p2   p3   p4
#> 1 0.25 0.25 0.25 0.25
#> 2 0.25 0.25 0.25 0.25
#> 3 0.25 0.25 0.25 0.25
#> 4 0.25 0.25 0.25 0.25
#> 5 0.25 0.25 0.25 0.25
#> 6 0.25 0.25 0.25 0.25

# probabilities for id = 1 (x = 0) and id = 51 (x = 1)
p[c(1, 51), ]
#>           p1        p2        p3   p4
#> 1  0.2500000 0.2500000 0.2500000 0.25
#> 51 0.1428571 0.1904762 0.2666667 0.40

## CUMULATIVE ODDS RATIO
```

```r
# calculate the (cumulative) odds ratio

x0 <- cump[1, 2:k]
x1 <- cump[51, 2:k]

# this is the same as the or (the b1) and we are assuming POA
odds(x0) / odds(x1)
#>   cump_leq1 cump_leq2 cump_leq3
#> 1         2         2         2
```

```r
## SAMPLING FROM THE MULTINOMIAL DISTRIBUTION

# example of a random Y outcome based on the probabilities
sample(x = 1:k, size = 1, prob = p[1, ])
#> [1] 3

# we can apply it to the full dataset, this step lead to different results
# each time we run the code because we are sampling from a distribution

dat$y <- apply(p, 1, function(ps) sample(1:k, size = 1, prob = ps))
head(dat)
#>   x y
#> 1 0 3
#> 2 0 3
#> 3 0 1
#> 4 0 4
#> 5 0 3
#> 6 0 2

# let's compute the observed probabilities, to be compared to the true
# probabilities

# observed
(op <- prop.table(table(dat$x, dat$y), margin = 1))
#>
#>        1    2    3    4
#>   0 0.24 0.34 0.22 0.20
#>   1 0.16 0.14 0.30 0.40

# true (just selecting a row from x = 0 and x = 1)
```

```
p[c(1, 51), ]
#>           p1        p2        p3   p4
#> 1  0.2500000 0.2500000 0.2500000 0.25
#> 51 0.1428571 0.1904762 0.2666667 0.40

# similarly we can compute the observed cumulative odds ratios

(cum_op <- apply(op, 1, cumsum))
#>
#>        0    1
#>   1 0.24 0.16
#>   2 0.58 0.30
#>   3 0.80 0.60
#>   4 1.00 1.00
odds(cum_op[, 1]) / odds(cum_op[, 2])
#>        1        2        3        4
#> 1.657895 3.222222 2.666667      NaN

# the odds ratios are not the same as the parameter. as we increase N
# the parameter will converge to the true value
```

```
dat$y <- ordered(dat$y) # make an ordered factor in R where 1 < 2 < 3 < 4
fit <- clm(y ~ x, data = dat, link = "probit")
summary(fit)
#> formula: y ~ x
#> data:    dat
#>
#>  link   threshold nobs logLik  AIC    niter max.grad cond.H
#>  probit flexible  100  -134.51 277.02 5(0)  6.26e-14 1.6e+01
#>
#> Coefficients:
#>   Estimate Std. Error z value Pr(>|z|)
#> x   0.5424     0.2193   2.474   0.0134 *
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Threshold coefficients:
#>     Estimate Std. Error z value
#> 1|2  -0.6064     0.1727  -3.510
#> 2|3   0.1084     0.1649   0.657
#> 3|4   0.8175     0.1786   4.576
```

We can generalize the previous workflow as:

1. Define simulation parameters, baseline probabilities, etc. as in the previous workflow
2. Calculate the linear predictors η using $k-1$ equations combining the predictors $\mathbf{X}$ and regression coefficients β.
3. Apply the inverse of the link function $g^{-1}(\eta)$ on the linear predictor and calculate the cumulative probabilities $p(y \leq 1|x), p(y \leq 2|x), ...p(y \leq k-1|x)$.
4. Calculate for each observation the probability of the k outcome as the difference between cumulative probabilities.
5. Sample n outcomes from a multinomial distribution, choosing between k alternatives with the calculated probabilities.
6. Fit the appropriate model using `ordinal::clm()`

A general suggestion is to increase the sample size n to a very large value reducing the random variability to check if simulated parameters are recovered.

Simulating from the latent distribution. An equivalent but more efficient way to simulate an ordinal outcome is using the latent formulation of the model. This require simulating a standard linear regression using the appropriate data generation function (*logistic* or *normal*) and the cutting the latent values according to the thresholds α. The workflow is slightly different compared to the previous approach.

1. Define n (number of observations) and k (number of ordinal outcomes)
2. Define the regression coefficients β
3. Define the baseline probabilities when $x = 0$. These probabilities can be converted to the corresponding thresholds α choosing the appropriate link function (*logit* or *probit*).
4. Calculate the linear predictors η using $k-1$ equations combining the predictors $\mathbf{X}$, the regression coefficients β and the error sampled from the appropriate probability distribution.

5. Cut the latent variable into k areas using the thresholds α and assign the corresponding ordinal value. This step simply checks if the latent value $Y_i^\star$ is between two threshold values and assign the corresponding value. This can be done using the `cut()` or the `findInterval()` functions.

```
## SIMULATION PARAMETERS

N <- 1e3 # sample size
or <- 4 # odds ratio, higher here just for a more clear plot
k <- 4 # number of ordinal alternatives
probs0 <- rep(1/k, k) # probabilities for the first group
alpha <- prob_to_alpha(probs0, link = "logit")
dat <- data.frame(x = rep(c(0, 1), each = N/2))

## LINEAR PREDICTOR

# calculate the linear predictor using equation TODO put equation
dat$lp <- log(or) * dat$x

# add the random part by sampling errors from a standard logistic (or normal) distribution
dat$ystar <- dat$lp + rlogis(N, location = 0, scale = 1)

# cut the latent distribution. The + 1 because the first category is 0 by default.
dat$y <- findInterval(dat$ystar, alpha) + 1

head_tail(dat, n = 3)
#>      x       lp       ystar y
#> 1    0 0.000000  5.69014212 4
#> 2    0 0.000000 -0.46455234 2
#> 3    0 0.000000  0.46477194 3
#> 998  1 1.386294  1.77268372 4
#> 999  1 1.386294  0.31432885 3
#> 1000 1 1.386294  0.09155736 3
```

The Figure 5 depict the simulated Y^* and the corresponding ordinal value. As for the previous simulation we can fit the model using `clm()` and check the estimated parameters.

[Figure 5 about here]

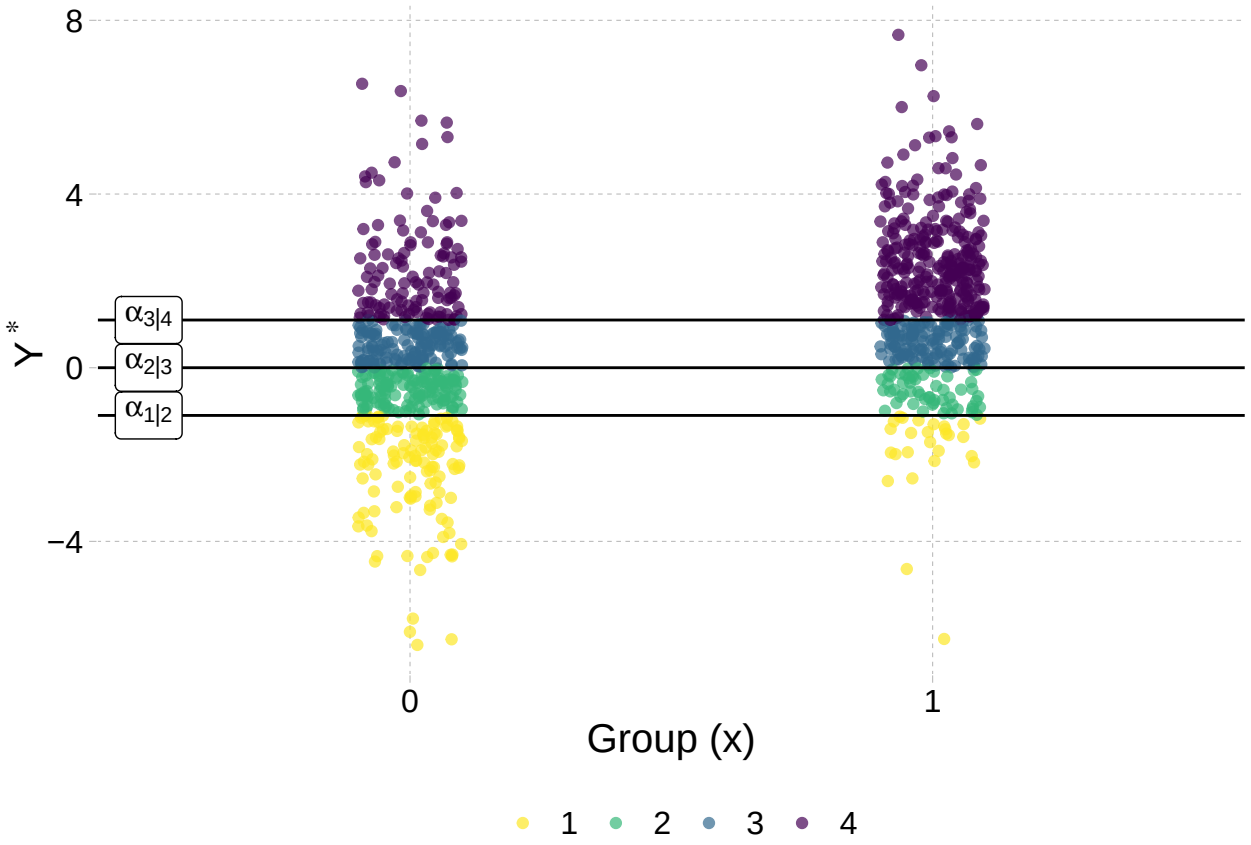

Figure 5. caption

```
dat$y <- ordered(dat$y) # make an ordered factor in R where 1 < 2 < 3 < 4
fit <- clm(y ~ x, data = dat, link = "logit")
summary(fit)
#> formula: y ~ x
#> data:    dat
#>
#>  link  threshold nobs logLik   AIC     niter max.grad cond.H
#>  logit flexible  1000 -1230.46 2468.92 4(0)  1.50e-07 1.5e+01
#>
#> Coefficients:
#>   Estimate Std. Error z value Pr(>|z|)
#> x    1.532      0.124   12.35   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Threshold coefficients:
#>     Estimate Std. Error z value
```

```
#> 1|2 -1.19697    0.09975  -12.00
#> 2|3 -0.01383    0.08657   -0.16
#> 3|4  1.21659    0.09628   12.64
```

The simulation using the latent formulation of the model is implemented in the `sim_ord_latent()` function. Details about the function can be found on the Github repository LINK.

```
N <- 100 # sample size
or <- 4 # odds ratio, higher here just for a more clear plot
k <- 4 # number of ordinal alternatives
probs0 <- rep(1/k, k) # probabilities for the first group
alpha <- prob_to_alpha(probs0, link = "logit")
dat <- data.frame(x = rep(c(0, 1), each = N/2))

# same as the previous simulation
dat <- sim_ord_latent(~x, By = log(or), prob = probs0, data = dat, link = "logit")

head_tail(dat, n = 3)
#>     x y         ys
#> 1   0 3  0.2137525
#> 2   0 2 -0.8345836
#> 3   0 3  0.3450251
#> 98  1 3  1.0198405
#> 99  1 3  0.5007378
#> 100 1 2 -0.2334367
```

Choosing parameters values.

Thresholds α**.** The previous simulation can be easily extended by adding predictors and their interactions. The crucial part is setting appropriate and empirically meaningful parameters. Thresholds α are usually not of main interest (but see the location-shift models Tutz, 2022) and can be considered as intercepts in standard linear regression. The thresholds are quantiles of the latent distribution that produced a certain probability distribution of the ordinal variable. To set meaningful α values we can convert probabilities into thresholds (`alpha_to_prob()`). The function `show_alpha()` create a

meaningful visual representation of using a specific set of thresholds (see Figure 6). The function can be used as `show_alpha(prob = rep(1/4, 4), link = "probit", plot = c("latent", "probs"))`. In regression terms, the thresholds determine the probabilities when $x = 0$ (as the intercept). Thus with two groups for example, the thresholds are the k probabilities of the ordinal variable Y for the reference group.

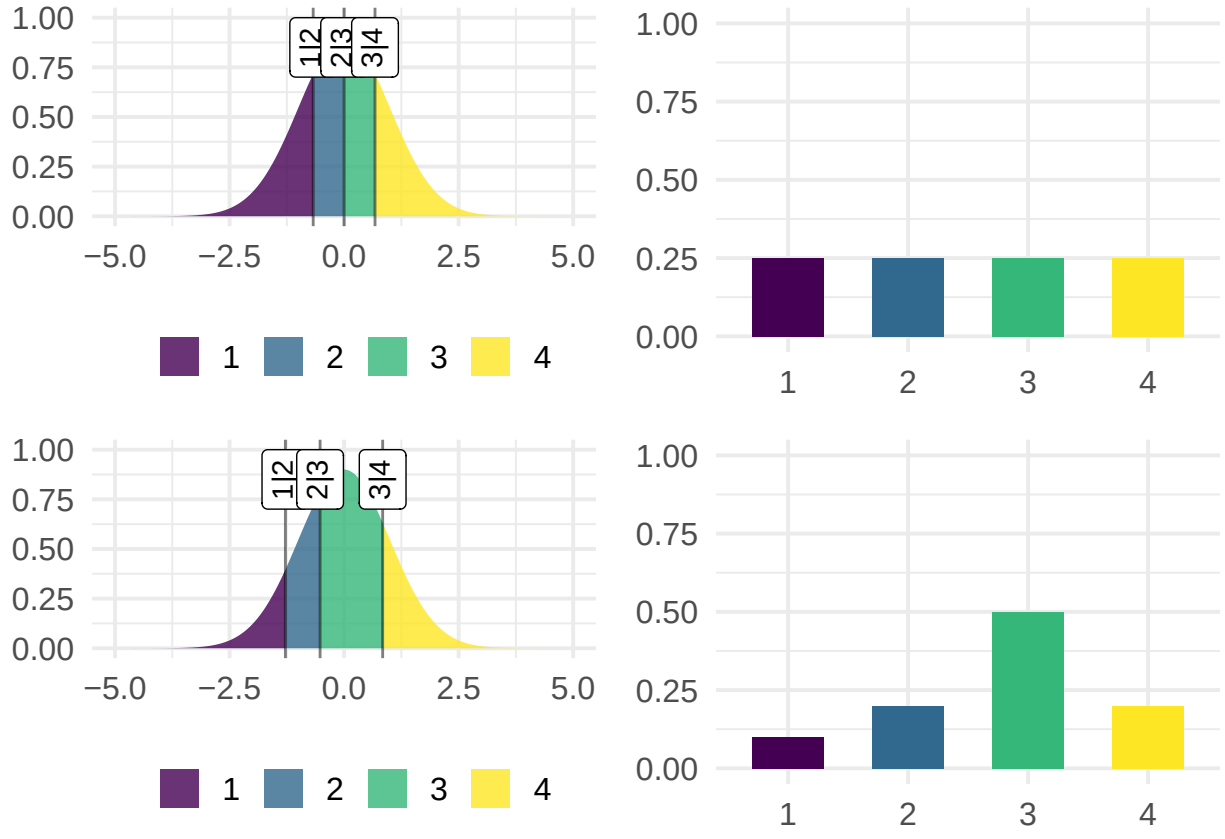

Figure 6. caption

[Figure 6 about here]

Regression coefficients β***.*** Regression coefficients are the effect size parameters. For *probit* models we can set the β_j to be in standardized (i.e., Cohen's d-like) units. For a categorical variable as the previous example with the group, β is the degree of separation in standard deviation unit between the two latent distributions. For *logit* models we can set the odds ratio. Meaningful odds ratios can be derived from previous literature,

meta-analyses or converting to other effect sizes. For example, Sánchez-Meca, Marín-Martínez, and Chacón-Moscoso (2003) proposed some equations to convert between odds ratios and Cohen's d. Using their approach, a Cohen's d of 0.5 usually considered a plausible medium effect size corresponds to and odds ratio of ≈ 2.47.

We can also calculate and plot the predicted probabilities (i.e., $g^{-1}(\eta)$, without actually sampling) given the predictors and the chosen regression coefficients. In this way we can try different values and see if predicted probabilities are plausible or not. The `cat_latent_plot()` and `num_latent_plot()` functions can be for respectively a categorical (Figures 7) and numerical predictor (Figures 8).

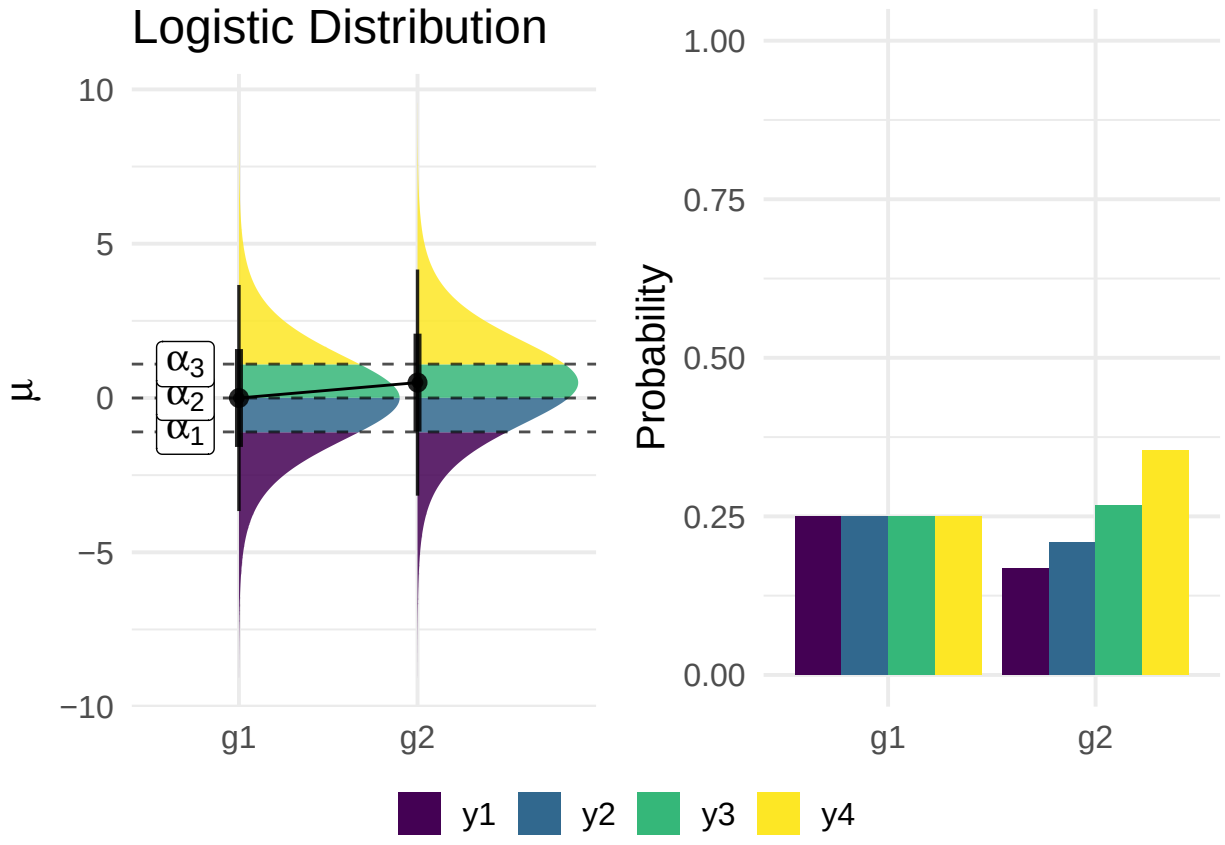

Figure 7. `cat_latent_plot(m = c(0, 0.5), s = 1, probs = rep(1/4, 4), link = "logit", plot = "both")`

[Figure 7 about here]

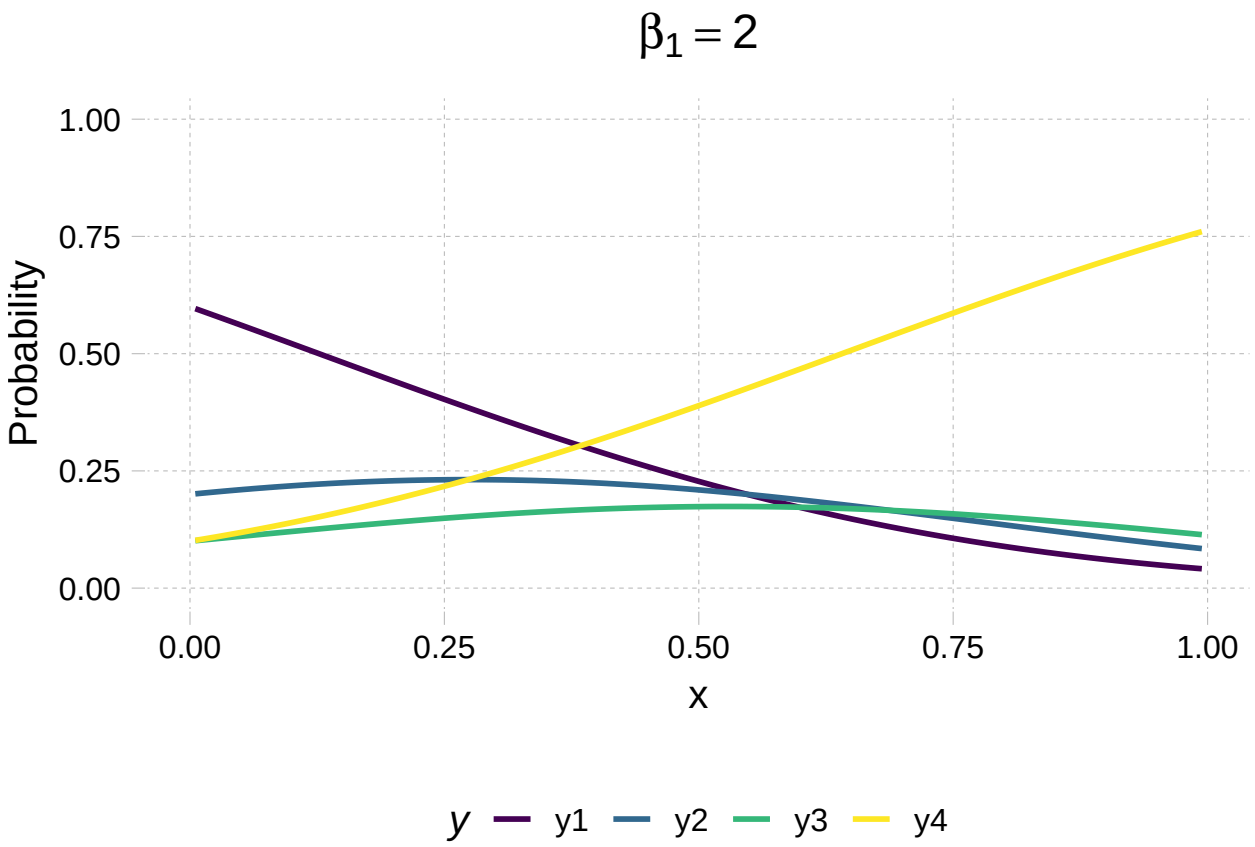

Figure 8. `num_latent_plot(x = runif(100), b1 = 2, probs = c(0.6, 0.2, 0.1, 0.1), link = "probit")`

[Figure 8 about here]

2x2 interaction

A common research design could be a 2x2 factorial design. In this example we have 2 main effects and the interaction. Regardless the link function, we can focus the model equation on the linear predictor. Equation depicts the linear predictor for a 2x2 interaction. With a 2x2 design, a good strategy is to use sum to zero contrasts for the two factors. By default R use dummy coding but here we set the contrasts for X_1 and X_2 as 0.5 and -0.5. In this way β_1 will be the main effect of X_1, β_2 the main effect of X_2 and β_3 the interaction (thus the difference of differences). Equation (6) depict the model formula.

$$P(Y \leq k) = g^{-1}(\alpha_k - \beta_1 X_{1_i} + \beta_2 X_{2_i} + \beta_3 X_{1_i} X_{2_i}) \tag{6}$$

```r
n <- 100
k <- 3
betas <- c(b1 = 0, b2 = 1, b3 = 0.5) # b1 = main effect X1, b2 = main effect X2, b3 = interaction

dat <- expand.grid(x1 = c("a", "b"), x2 = c("c", "d"), n = 1:n)
dat$x1 <- factor(dat$x1)
dat$x2 <- factor(dat$x2)

# sum to 0 coding
contrasts(dat$x1) <- c(0.5, -0.5)
contrasts(dat$x2) <- c(0.5, -0.5)
probs0 <- rep(1/k, k)

dat <- sim_ord_latent(~ x1 * x2, By = c(0, 1, 0.5), prob = probs0, link = "probit", data = dat)
fit <- clm(y ~ x1 * x2, data = dat, link = "probit")
summary(fit)
#> formula: y ~ x1 * x2
#> data:    dat
#>
#>  link   threshold nobs logLik  AIC    niter max.grad cond.H
#>  probit flexible  400  -402.44 814.87 5(0)  1.38e-07 2.6e+01
#>
#> Coefficients:
#>         Estimate Std. Error z value Pr(>|z|)
#> x11      -0.1513     0.1165  -1.299   0.1940
#> x21       0.9056     0.1177   7.691 1.45e-14 ***
#> x11:x21   0.4015     0.2330   1.723   0.0849 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Threshold coefficients:
#>     Estimate Std. Error z value
#> 1|2 -0.43622    0.06724  -6.487
#> 2|3  0.32019    0.06626   4.832
```

The parameters interpretation is the same as introduced in the general case. The thresholds α are fixed representing the probabilities $P(Y = k)$ when all predictors are zero.

In this case by doing `alpha_to_prob(fit$alpha, link = "probit")` we should recover the `probs0` vector. `x1` is the main effect thus the difference in z scores between a and b averaging over `x2`. The same holds for `x2`. `x1:x2` is the difference of differences in z scores (i.e., the interaction). Figure 9 depicts the model results.

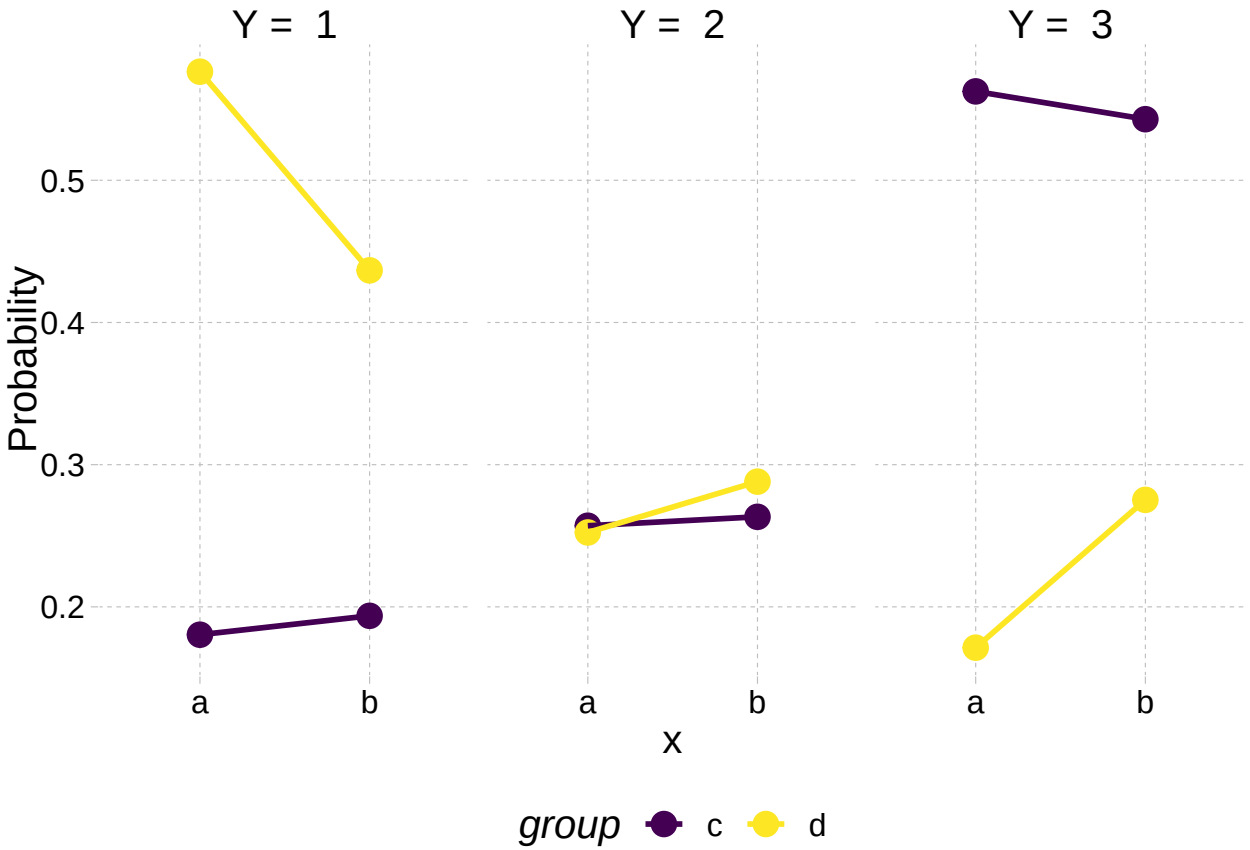

Figure 9

[Figure 9 about here]

Numerical by categorical interaction

Another common scenario is to compute and interaction between a numerical variable x and a categorical variable g. For simplicity we simulate the factor with two levels and the numerical variable sampled from an uniform distribution between 0 and 1.

```r
n <- 100
k <- 3
dat <- data.frame(x = runif(n),
                  g = rep(c("a", "b"), each = n/2))
dat$g <- factor(dat$g)
contrasts(dat$g) <- c(0.5, -0.5)
probs0 <- rep(1/k, k)

dat <- sim_ord_latent(~ x * g, By = c(0.3, 0.5, 0.5), prob = probs0, link = "probit", data = dat)
fit <- clm(y ~ x * g, data = dat, link = "probit")
summary(fit)
#> formula: y ~ x * g
#> data:    dat
#>
#>  link   threshold nobs logLik  AIC    niter max.grad cond.H
#>  probit flexible  100  -100.58 211.15 6(0)  2.24e-13 9.7e+01
#>
#> Coefficients:
#>      Estimate Std. Error z value Pr(>|z|)
#> x     0.39237    0.38698   1.014   0.3106
#> g1    0.02712    0.44200   0.061   0.9511
#> x:g1  1.43293    0.77516   1.849   0.0645 .
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Threshold coefficients:
#>     Estimate Std. Error z value
#> 1|2  -0.2440     0.2289  -1.066
#> 2|3   0.4631     0.2308   2.006
```

Again, `x` is the slope of the numerical predictor averaging over `g` (given that `g` has been coded with sum to zero contrasts) thus the increase in z scores for a unit increase in `x`. `g1` is the main effect of the factor evaluated when $x = 0$. To change the interpretation of `g1` a possibility is to center `x` differently (e.g., mean centering). The `x:g1` is the difference in slopes between the two groups. Figure 10 depicts the model results.

[Figure 10 about here]

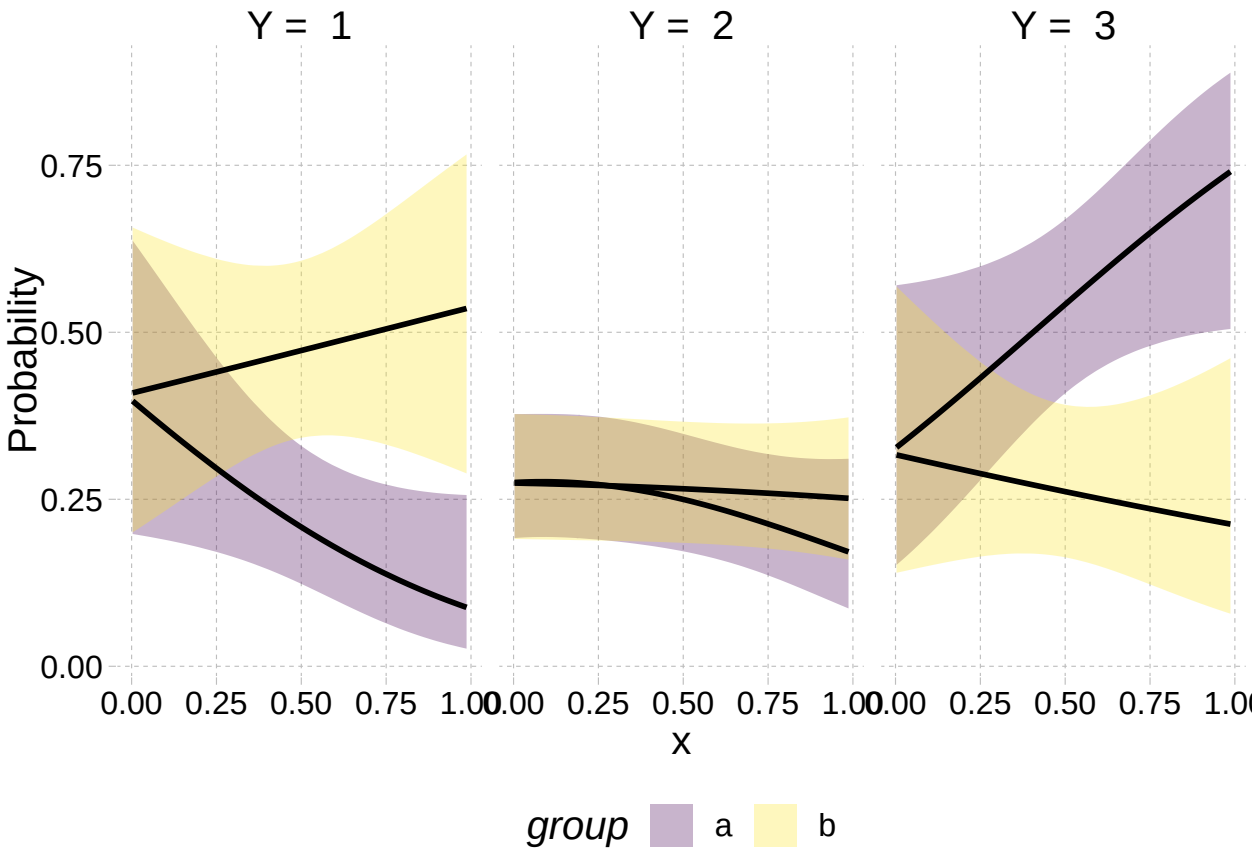

Figure 10

Power Analysis

In this section we introduce how to estimate the statistical power. Formally the power is defined as the probability of correctly rejecting the null hypothesis H_0. In some simple case, the power can be computed analytically however using data simulation we can handle every scenario. We present very general example that, as for the previous simulations, can be extended and modified.

For example, by simulating a model fixing a specific parameter to zero, we can estimate the type-1 error rate (wrongly rejecting H_0) or assessing an eventual bias in model estimates. The general workflow for a power simulation can be summarized as:

1. Specify the research design, e.g., 2x2 factorial design

2. Specify the effect/parameter of interest, e.g., the interaction effect
3. Define the simulation conditions, e.g., a range of sample size, effect size, etc.
4. Implement one of the simulation workflow described in the previous sections
5. Repeat the simulation a large number of times (e.g, 10000) and store the relevant values from each simulation
6. Summarise the simulation results

To make a practical example, we simulate the power of detecting a group difference of $d = 0.4$ (assuming a *probit* model). Participants respond to an ordinal variable with $k = 5$. The iteration is performed using a `for` loop that repeat 1000 times the data simulation, model fitting and extract the p-value for the β_1. The power is then estimated as the number of p-values lower than the critical level over the number of simulations.

```
n <- 40 # sample size
k <- 5  # number of ordinal variables
d <- 0.4 # effect size (i.e., our regression coefficients)
nsim <- 1e3 # higher is better, here using 1000 for an example
probs0 <- rep(1/k, k)
alpha <- 0.05 # critical alpha

p <- rep(NA, nsim) # preallocation to improve loop computational efficiency
dat <- data.frame(group = rep(c("a", "b"), each = n)) # data frame

head_tail(dat, n = 3)
#>    group
#> 1      a
#> 2      a
#> 3      a
#> 78     b
#> 79     b
#> 80     b

for(i in 1:nsim){
  sim <- sim_ord_latent(~group, By = d, prob = probs0, link = "probit", data = dat)
  fit <- clm(y ~ group, data = sim, link = "probit")
  p[i] <- summary(fit)$coefficients["groupb", "Pr(>|z|)"] # extract the pvalue
```

```
}

# estimate the power
mean(p <= alpha, na.rm = TRUE)
#> [1] 0.404
```

Despite useful, calculating power curves instead a single value is more informative. For this reason we repeat the previous simulation but for different sample sizes. Figure 11 depict the power curve resulting from the simulation.

```
n <- c(20, 40, 60, 100, 200)
power <- rep(NA, length(n))

for(i in 1:length(n)){
  p <- rep(NA, nsim) # preallocation for speed
  dat <- data.frame(group = rep(c("a", "b"), each = n[i]))
  for(j in 1:nsim){
    sim <- sim_ord_latent(~group, By = d, prob = probs0, link = "probit", data = dat)
    fit <- clm(y ~ group, data = sim, link = "probit")
    p[j] <- summary(fit)$coefficients["groupb", "Pr(>|z|)"]
  }
  power[i] <- mean(p <= alpha)
}

power
#> [1] 0.231 0.405 0.521 0.768 0.969
```

[Figure 11 about here]

Disclaimer about the functions

The current paper proposed a simplified way with some functions to generate ordinal data. For more complex simulations such as simulating correlated ordinal data the `simstudy` package https://kgoldfeld.github.io/simstudy/articles/ordinal.html proposed a very comprehensive set of data simulation function also for ordinal data.

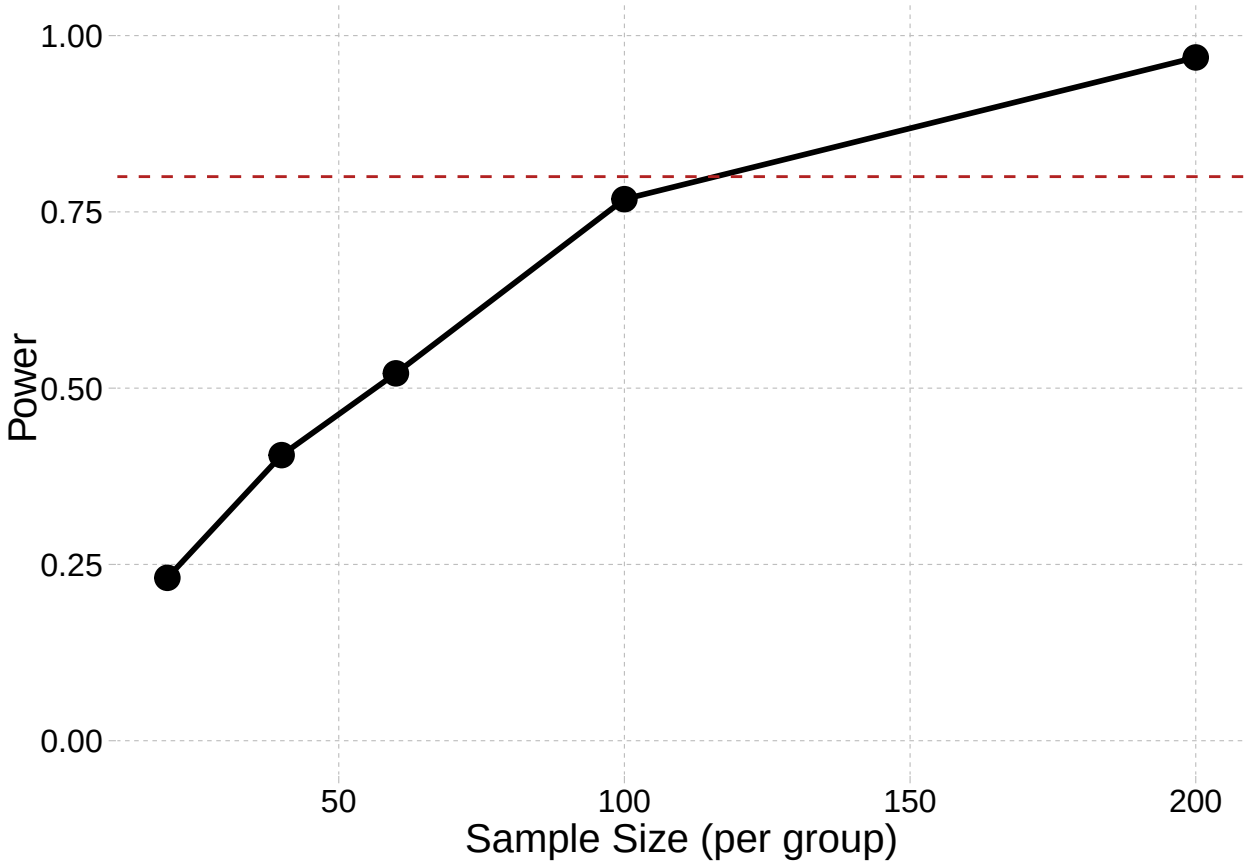

Figure 11. caption

References

Agresti, A. (2010). *Analysis of ordinal categorical data.* Hoboken, NJ, USA: John Wiley & Sons, Inc. https://doi.org/10.1002/9780470594001

Bürkner, P.-C., & Vuorre, M. (2019). Ordinal regression models in psychology: A tutorial. *Advances in Methods and Practices in Psychological Science*, *2*, 77–101. https://doi.org/10.1177/2515245918823199

Carifio, J., & Perla, R. (2008). Resolving the 50-year debate around using and misusing likert scales. *Medical Education*, *42*, 1150–1152. https://doi.org/10.1111/j.1365-2923.2008.03172.x

Carifio, J., & Perla, R. J. (2007). Ten common misunderstandings, misconceptions, persistent myths and urban legends about likert scales and likert response formats and

their antidotes. *Journal of Social Sciences*, *3*, 106–116. https://doi.org/10.3844/jssp.2007.106.116

Christensen, R. H. B. (2019). *Ordinal—regression models for ordinal data*.

Cliff, N. (1996). Answering ordinal questions with ordinal data using ordinal statistics. *Multivariate Behavioral Research*, *31*, 331–350. https://doi.org/10.1207/s15327906mbr3103_4

Cliff, Norman. (2016). *Ordinal methods for behavioral data analysis*. London, England: Psychology Press.

Cox, C. (1995). Location-scale cumulative odds models for ordinal data: A generalized non-linear model approach. *Statistics in Medicine*, *14*, 1191–1203. https://doi.org/10.1002/sim.4780141105

DeCarlo, L. T. (2010). On the statistical and theoretical basis of signal detection theory and extensions: Unequal variance, random coefficient, and mixture models. *Journal of Mathematical Psychology*, *54*, 304–313. https://doi.org/10.1016/j.jmp.2010.01.001

Gelman, A., Hill, J., & Vehtari, A. (2020). *Regression and other stories*. Cambridge University Press. https://doi.org/10.1017/9781139161879

Jamieson, S. (2004). Likert scales: How to (ab)use them. *Medical Education*, *38*, 1217–1218. https://doi.org/10.1111/j.1365-2929.2004.02012.x

Kemp, S., & Grace, R. C. (2021). Using ordinal scales in psychology. *Methods in Psychology (Online)*, *5*, 100054. https://doi.org/10.1016/j.metip.2021.100054

Kruschke, J. K. (2015). *Doing bayesian data analysis*. https://doi.org/10.1016/c2012-0-00477-2

Liddell, T. M., & Kruschke, J. K. (2018). Analyzing ordinal data with metric models: What could possibly go wrong? *Journal of Experimental Social Psychology*, *79*, 328–348. https://doi.org/10.1016/j.jesp.2018.08.009

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of Psychology*, *22*, 55. Retrieved from https://psycnet.apa.org/record/1933-01885-001

Liu, A., He, H., Tu, X. M., & Tang, W. (2023). On testing proportional odds assumptions for proportional odds models. *General Psychiatry*, *36*, e101048. https://doi.org/10.1136/gpsych-2023-101048

McCullagh, P. (1980). Regression models for ordinal data. *Journal of the Royal Statistical Society*, *42*, 109–127. https://doi.org/10.1111/j.2517-6161.1980.tb01109.x

Rigby, R. A., & Stasinopoulos, D. M. (2005). Generalized additive models for location, scale and shape. *Journal of the Royal Statistical Society. Series C, Applied Statistics*, *54*, 507–554. https://doi.org/10.1111/j.1467-9876.2005.00510.x

Robitzsch, A. (2020). Why ordinal variables can (almost) always be treated as continuous variables: Clarifying assumptions of robust continuous and ordinal factor analysis estimation methods. *Frontiers in Education*, *5*, 589965. https://doi.org/10.3389/feduc.2020.589965

Sánchez-Meca, J., Marín-Martínez, F., & Chacón-Moscoso, S. (2003). Effect-size indices for dichotomized outcomes in meta-analysis. *Psychological Methods*, *8*, 448–467. https://doi.org/10.1037/1082-989X.8.4.448

Schad, D. J., Vasishth, S., Hohenstein, S., & Kliegl, R. (2020). How to capitalize on a priori contrasts in linear (mixed) models: A tutorial. *Journal of Memory and Language*, *110*, 104038. https://doi.org/10.1016/j.jml.2019.104038

Stevens, S. S. (1946). On the theory of scales of measurement. *Science (New York, N.Y.)*, *103*, 677–680. https://doi.org/10.1126/science.103.2684.677

Tutz, G. (1990). Sequential item response models with an ordered response. *The British Journal of Mathematical and Statistical Psychology*, *43*, 39–55. https://doi.org/10.1111/j.2044-8317.1990.tb00925.x

Tutz, G. (2022). Ordinal regression: A review and a taxonomy of models. *Wiley Interdisciplinary Reviews. Computational Statistics*, *14*. https://doi.org/10.1002/wics.1545

Tutz, G., & Berger, M. (2020). Non proportional odds models are widely dispensable –

sparser modeling based on parametric and additive location-shift approaches. *arXiv [Stat.ME]*. Retrieved from http://arxiv.org/abs/2006.03914