More advanced probabilistic ordinal model examples

This document will provide more examples how probabilistic models can be used to fit more advanced ordinal regression models.

```
# Load libraries needed for the analysis
library(tidyverse)
library(brms)
library(tidybayes)
library(haven)
library(kableExtra)
library(broom.mixed)
```

```
# Loading the data
census_data_all <- read_sav("../../input/data_processed/combined_census_data.sav") %>%
  filter(exclude == 0) %>%
  mutate(Q2 = haven::as_factor(Q2),
         Age_group = case_when(
          Age < 18 ~ "<18",
          Age > 17 & Age < 30 ~ "18-29",
          Age > 29 & Age < 50 ~ "30-49",
          Age > 49 & Age < 65 ~ "50-64",
          Age > 64 ~ ">64"
        ))
```

Varying thresholds model

Non-hierarchical model

The typical ordinal cumulative model assumes that the thresholds are same for all the explanatory variables. It may sometimes be the case that there is variation at threshold that depends on the explanatory variables.

Let's fit a model that has its own threshold values for every city. This is done by adding the term `thres(gr = City, x = 4)` to the model formula. The code for the model looks quite daunting now, because the priors have to be set separately for all the cities. We will use the same prior for all the cities, but it is also possible to set separate priors for all the cities.

```
# Fitting the varying thresholds model
census_data_all <- census_data_all %>%
  mutate(Q11 = as.numeric(Q11))
fit_vt <- brm(Q11 | thres(gr = City, x = 4) ~ Q2 + Age_group,
                      family = cumulative("logit"), data = census_data_all,
                      seed = 2025, cores = 4, control = list(adapt_delta = 0.99),
                      prior =
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Heidelberg") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Heidelberg") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Heidelberg") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Heidelberg") +
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Helsinki") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Helsinki") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Helsinki") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Helsinki") +
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Lviv") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Lviv") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Lviv") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Lviv") +
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Mannheim") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Mannheim") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Mannheim") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
```

```
                              group = "Mannheim") +
                    prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Vilnius") +
                    prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Vilnius") +
                    prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Vilnius") +
                    prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Vilnius") +
                    prior(normal(0,2), class = "b"),
                  backend = "cmdstanr", silent = 2, refresh = 0)
```

Let's look at the model output.

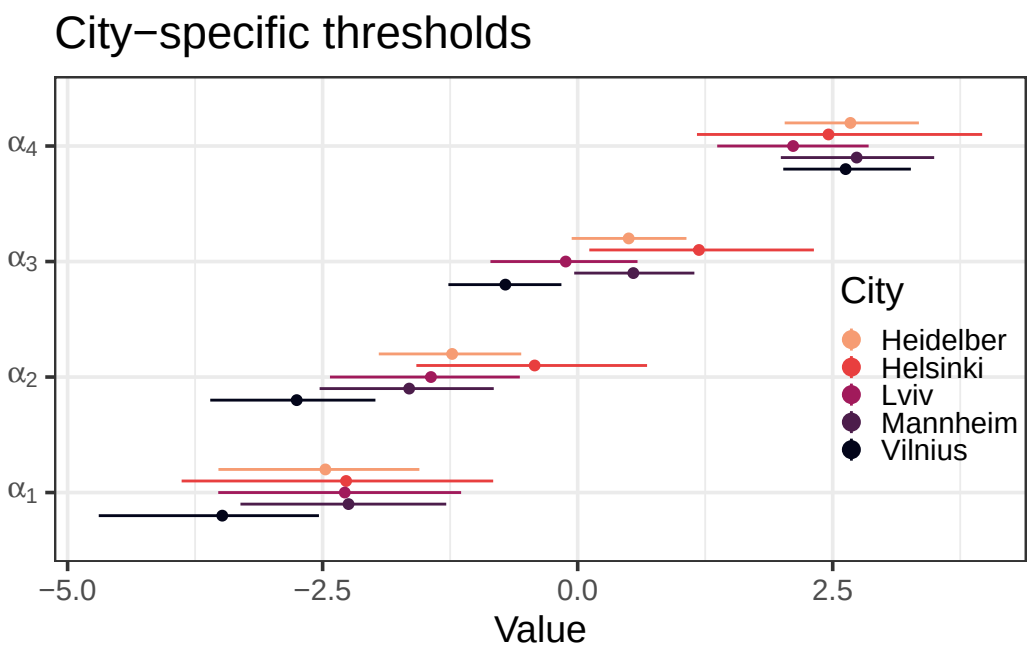
```
# Model output
tidy(fit_vt) %>%
  select(-c(effect, component, group)) %>%
  kable()
```

term	estimate	std.error	conf.low	conf.high
(Intercept)[Heidelberg,1]	-2.2698362	0.7823353	-3.8821850	-0.8281444
(Intercept)[Heidelberg,2]	-0.4216905	0.5698219	-1.5812145	0.6802599
(Intercept)[Heidelberg,3]	1.1890099	0.5589836	0.1138926	2.3156308
(Intercept)[Heidelberg,4]	2.4585582	0.7088550	1.1693403	3.9648808
(Intercept)[Helsinki,1]	-3.4837445	0.5620245	-4.6944877	-2.5362105
(Intercept)[Helsinki,2]	-2.7540181	0.4075704	-3.6016350	-1.9824042
(Intercept)[Helsinki,3]	-0.7086089	0.2840038	-1.2671470	-0.1595163
(Intercept)[Helsinki,4]	2.6267407	0.3157271	2.0151780	3.2660430
(Intercept)[Lviv,1]	-2.2829490	0.6129537	-3.5231712	-1.1424980
(Intercept)[Lviv,2]	-1.4380341	0.4732845	-2.4281335	-0.5669286
(Intercept)[Lviv,3]	-0.1171747	0.3698706	-0.8552893	0.5865946
(Intercept)[Lviv,4]	2.1120305	0.3775679	1.3677998	2.8521118
(Intercept)[Mannheim,1]	-2.2462960	0.5146615	-3.3054277	-1.2883145
(Intercept)[Mannheim,2]	-1.6514626	0.4287465	-2.5296147	-0.8225470
(Intercept)[Mannheim,3]	0.5456407	0.3052039	-0.0339780	1.1439208
(Intercept)[Mannheim,4]	2.7349447	0.3834421	1.9913493	3.4944635
(Intercept)[Vilnius,1]	-2.4729609	0.5003243	-3.5213380	-1.5515472
(Intercept)[Vilnius,2]	-1.2302979	0.3503553	-1.9495905	-0.5532764
(Intercept)[Vilnius,3]	0.5000522	0.2943122	-0.0588686	1.0663265
(Intercept)[Vilnius,4]	2.6734615	0.3366340	2.0290550	3.3447455
Q2Male	0.1851185	0.2036432	-0.2220274	0.5909962

term	estimate	std.error	conf.low	conf.high
Q2Anothergenderidentity	-0.7325406	0.5301658	-1.7458975	0.3291252
Age_group18M29	1.2086918	0.2819855	0.6570818	1.7837392
Age_group30M49	0.5406557	0.2806093	-0.0239966	1.0876888
Age_group50M64	0.7056235	0.3609955	-0.0011124	1.4098265

From the output we see that every city now has its own intercept terms. Let's plot the estimates and their intervals and compare the results.

```
# Plotting the thresholds
posterior_summary(fit_vt)[1:20,] %>%
  data.frame() %>%
  mutate(alpha = rep(1:4, times = 5, length.out = 20),
         City = rep(c("Helsinki", "Vilnius", "Lviv", "Mannheim", "Heidelber"),
                    each = 4, length.out = 20)) %>%
  mutate(alpha = str_c("alpha[", alpha, "]"),
         City = factor(City)) %>%
  ggplot(aes(y = Estimate, ymin = Q2.5, ymax = Q97.5, x = alpha, group = City,
             color = City)) +
  geom_pointrange(position = position_dodge(width = -0.5), fatten = 1.5) +
  scale_color_viridis_d("City", option = "F", end = 0.8, direction = -1) +
  scale_x_discrete(NULL, labels = ggplot2:::parse_safe) +
  coord_flip() + labs(title = "City-specific thresholds", y = "Value") +
  theme_bw(14) +
  theme(legend.position = c(0.9, 0.4),
                        legend.background = element_rect(fill = "transparent"),
                        legend.key = element_rect(fill = "transparent"),
                        legend.key.size = unit(0.3, "cm"))
```



From the figure we can see that there is some variation in the threshold values.

Hierarchical model

The varying thresholds model can also be fitted in the hierarchical framework. This is done by adding the term `(1 | City)` to the model formula.

```
# Fitting the model
fit_vth <- brm(Q11 | thres(gr = City, x = 4) ~ Q2 + Age_group + (1 | City),
                       family = cumulative("logit"), data = census_data_all,
                       seed = 2025, cores = 4, control = list(adapt_delta = 0.99),
                       prior =
                         prior(normal(-1.4,1), class = "Intercept", coef = "1",
                               group = "Heidelberg") +
                         prior(normal(-0.4,1), class = "Intercept", coef = "2",
                               group = "Heidelberg") +
                         prior(normal(0.4,1), class = "Intercept", coef = "3",
                               group = "Heidelberg") +
                         prior(normal(1.4,1), class = "Intercept", coef = "4",
                               group = "Heidelberg") +
                         prior(normal(-1.4,1), class = "Intercept", coef = "1",
                               group = "Helsinki") +
                         prior(normal(-0.4,1), class = "Intercept", coef = "2",
                               group = "Helsinki") +
```

```
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Helsinki") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Helsinki") +
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Lviv") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Lviv") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Lviv") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Lviv") +
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Mannheim") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Mannheim") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Mannheim") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Mannheim") +
                        prior(normal(-1.4,1), class = "Intercept", coef = "1",
                              group = "Vilnius") +
                        prior(normal(-0.4,1), class = "Intercept", coef = "2",
                              group = "Vilnius") +
                        prior(normal(0.4,1), class = "Intercept", coef = "3",
                              group = "Vilnius") +
                        prior(normal(1.4,1), class = "Intercept", coef = "4",
                              group = "Vilnius") +
                        prior(normal(0,2), class = "b") +
                        prior(normal(0,1), class = "sd"),
                      backend = "cmdstanr", silent = 2, refresh = 0)
```

Let's look at the model output.

```
# Model output
tidy(fit_vth) %>%
  select(-c(effect, component, group)) %>%
  kable()
```

term	estimate	std.error	conf.low	conf.high
(Intercept)[Heidelberg,1]	-2.4229929	0.7512240	-3.9297742	-1.0047657
(Intercept)[Heidelberg,2]	-0.6409857	0.6072731	-1.8250045	0.5483168
(Intercept)[Heidelberg,3]	0.9898746	0.6016763	-0.1991195	2.1488893
(Intercept)[Heidelberg,4]	2.3384651	0.7496242	0.9178453	3.8809390
(Intercept)[Helsinki,1]	-3.0844504	0.6717646	-4.4693052	-1.8065245
(Intercept)[Helsinki,2]	-2.2163246	0.5787478	-3.3522507	-1.0332402
(Intercept)[Helsinki,3]	0.0188336	0.5754979	-1.0202075	1.2068830
(Intercept)[Helsinki,4]	3.3608185	0.5995628	2.2859493	4.5507035
(Intercept)[Lviv,1]	-2.1530491	0.6456578	-3.5090607	-0.9440828
(Intercept)[Lviv,2]	-1.2766911	0.5313301	-2.3081387	-0.2225463
(Intercept)[Lviv,3]	0.1110473	0.5059745	-0.8152913	1.1525645
(Intercept)[Lviv,4]	2.3635483	0.5362387	1.3360988	3.4509400
(Intercept)[Mannheim,1]	-2.2466972	0.5644227	-3.4046332	-1.1989710
(Intercept)[Mannheim,2]	-1.6504280	0.5064629	-2.6527500	-0.6903134
(Intercept)[Mannheim,3]	0.5807775	0.4813588	-0.3211288	1.5450023
(Intercept)[Mannheim,4]	2.7693166	0.5064855	1.8215463	3.7996122
(Intercept)[Vilnius,1]	-2.4701467	0.5752439	-3.6300680	-1.3908395
(Intercept)[Vilnius,2]	-1.2257083	0.4737339	-2.1814375	-0.2899427
(Intercept)[Vilnius,3]	0.4904327	0.4746683	-0.4620940	1.3931625
(Intercept)[Vilnius,4]	2.6582845	0.4975343	1.6860857	3.6232710
Q2Male	0.1477156	0.2039260	-0.2429777	0.5493738
Q2Anothergenderidentity	-0.7415184	0.5342113	-1.7731820	0.3133917
Age_group18M29	0.7460026	0.4472017	-0.1761158	1.5788012
Age_group30M49	0.0597720	0.4391687	-0.8479506	0.8435771
Age_group50M64	0.2442747	0.4877355	-0.7479772	1.1701905
sd___(Intercept)	0.9396291	0.4928416	0.1039942	2.0073952

Now there is the variance term for the random effects.

Varying dispersion model

Non-hierarchical model

If there is a reason to suspect that there is additional variance that is not taken into account by the normal cumulative model, a model with varying dispersion can be fit. The model allows modeling additional variance with the explanatory variables.

Let's fit a model that has varying dispersion based on the gender variable. This is done by wrapping the normal formula inside function `bf()` while the varying dispersion part is specified inside function `lf()`. It is important to set `cmc = FALSE` inside the `lf()` function.

```r
# Fitting the model
fit_vd <- brm(bf(Q11 | thres(x = 4)  ~  Age_group + Q2) +
                        lf(disc ~ 0 + Q2, cmc = FALSE),
                      family = cumulative("logit"), data = census_data_all,
                      seed = 2025, cores = 4, control = list(adapt_delta = 0.99),
                      prior = prior(normal(-1.39,1), class = "Intercept", coef = "1") +
                        prior(normal(-0.45,1), class = "Intercept", coef = "2") +
                        prior(normal(0.45,1), class = "Intercept", coef = "3") +
                        prior(normal(1.39,1), class = "Intercept", coef = "4") +
                        prior(normal(0,2), class = "b") +
                        prior(normal(0,1), class = "b", dpar = "disc"),
                      backend = "cmdstanr", silent = 2, refresh = 0)
```

Let's take a look at the model output.

```r
# Model output
tidy(fit_vd) %>%
  select(-c(effect, component, group)) %>%
  kable()
```

term	estimate	std.error	conf.low	conf.high
(Intercept)[1]	-4.3145740	0.5711842	-5.4408695	-3.2342147
(Intercept)[2]	-3.1975661	0.4858480	-4.1810068	-2.2616340
(Intercept)[3]	-1.1834736	0.4348310	-2.0433042	-0.3301400
(Intercept)[4]	1.4869770	0.4420990	0.6242110	2.3897837
Age_group18M29	0.0141763	0.4380401	-0.8566367	0.8955342
Age_group30M49	-0.5068745	0.4415900	-1.3748342	0.3633992
Age_group50M64	-0.1986401	0.4767572	-1.1280395	0.7460828
Q2Male	-0.0519771	0.2070357	-0.4590906	0.3639125
Q2Anothergenderidentity	-1.0857742	0.7974776	-2.6591637	0.5370771
disc_Q2Male	0.0057510	0.1010666	-0.1938605	0.2062159
disc_Q2Anothergenderidentity	-0.6130064	0.2634649	-1.1611100	-0.1225572

The terms `disc_Q2Male` and `disc_Q2Anothergenderidentity` describe how the variance of these terms compare to the reference category female. The output has to be transformed to a form where it makes sense.

```r
# Variace terms
tibble(Gender = c("Female", "Male", "Anothergenderidentity"),
       sigma = 1/exp(c(0, fixef(fit_vd)["disc_Q2Male",1],
```

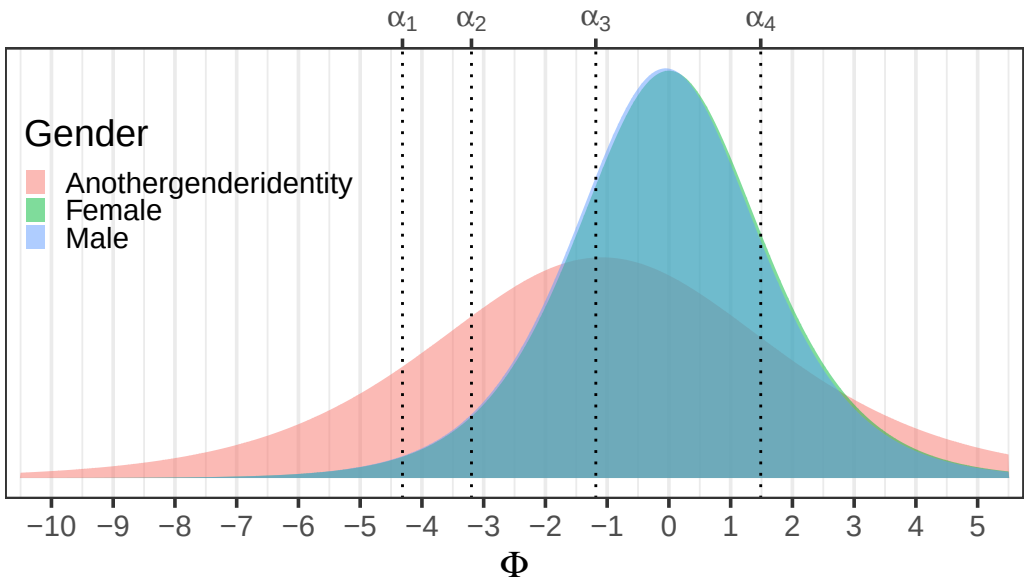
```
                        fixef(fit_vd)["disc_Q2Anothergenderidentity",1]))) %>%
  kable()
```

Gender	sigma
Female	1.0000000
Male	0.9942655
Anothergenderidentity	1.8459728

We see that people with another gender identity have a higher variance when compared to females.

Let's also plot the distributions for the different genders and compare the results.

```
# Plotting the distributions
tibble(Q2 = c("Female", "Male", "Anothergenderidentity"),
       mu = c(0, fixef(fit_vd)["Q2Male",1],
              fixef(fit_vd)["Q2Anothergenderidentity",1]),
       sigma = 1/exp(c(0, fixef(fit_vd)["disc_Q2Male",1],
                       fixef(fit_vd)["disc_Q2Anothergenderidentity",1]))) %>%
    expand(nesting(Q2, mu, sigma),
           x = seq(from = -10.5, to = 5.5, length.out = 500)) %>%
    mutate(d = dlogis(x, mu, sigma)) %>%
    ggplot(aes(x = x, y = d, fill = Q2)) +
    geom_area(alpha = 1/2, position = "identity") +
    geom_vline(xintercept = fixef(fit_vd)[1:4, 1], linetype = 3) +
    scale_x_continuous(expression(Phi), breaks = -10:5,
                       sec.axis = dup_axis(
                           name = NULL,
                           breaks = fixef(fit_vd)[1:4, 1] %>% as.double(),
                           labels = parse(text = str_c("alpha[", 1:4, "]"))
                       )) +
    scale_y_continuous(NULL, breaks = NULL) +
    coord_cartesian(xlim = c(-10, 5)) +
    theme_bw(base_size = 14) +
    theme(legend.position = c(0.18,0.7),
          legend.background = element_rect(fill = "transparent"),
          legend.key = element_rect(fill = "transparent"),
          legend.key.size = unit(0.3, "cm")) +
    labs(subtitle = "Varying dispersion model", fill = "Gender")
```



From the figure it is clear that there is really no difference between males and females. However, people with another gender identity seem to have a lower mean response and higher variance.

Hierarchical model

The varying dispersion model can also be fitted inside the hierarchical framework. The dispersion itself can have a hierarchical effect. This done by adding `(1 | City)` term to the dispersion formula inside function `lf()`.

```
# Fitting the model
fit_vdh <- brm(bf(Q11 | thres(x = 4)  ~  Age_group + Q2 + (1 | City)) +
                          lf(disc ~ 0 + (1 | City), cmc = FALSE),
                        family = cumulative("logit"), data = census_data_all,
                        seed = 2025, cores = 4, control = list(adapt_delta = 0.99),
                        prior = prior(normal(-1.39,1), class = "Intercept", coef = "1") +
                          prior(normal(-0.45,1), class = "Intercept", coef = "2") +
                          prior(normal(0.45,1), class = "Intercept", coef = "3") +
                          prior(normal(1.39,1), class = "Intercept", coef = "4") +
                          prior(normal(0,2), class = "b") +
                          prior(normal(0,1), class = "sd", dpar = "disc"),
                        backend = "cmdstanr", silent = 2, refresh = 0)
```

The model output now has a term for the dispersion variance term.

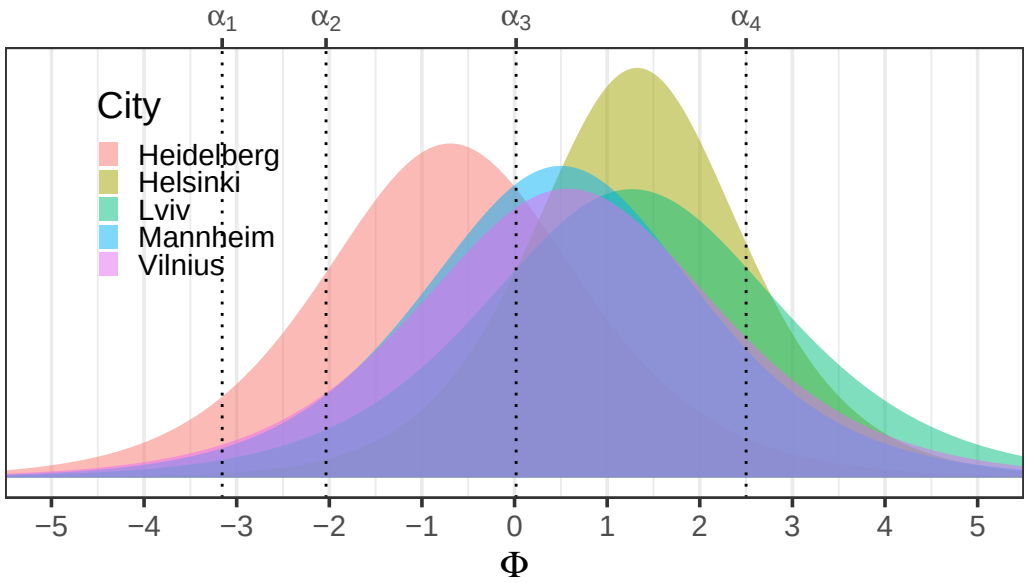
```r
# Model output
tidy(fit_vdh) %>%
  select(-c(effect, component, group)) %>%
  kable()
```

term	estimate	std.error	conf.low	conf.high
(Intercept)[1]	-3.1566799	0.6657579	-4.4786033	-1.8382170
(Intercept)[2]	-2.0354138	0.6011494	-3.1947935	-0.8173712
(Intercept)[3]	0.0169688	0.5996974	-1.0987815	1.2627953
(Intercept)[4]	2.5001717	0.6881068	1.3215935	3.9405853
Age_group18M29	0.2296032	0.4065378	-0.5711817	1.0463628
Age_group30M49	-0.3919397	0.3869590	-1.1483227	0.3747893
Age_group50M64	-0.2095030	0.4234866	-1.0376465	0.6204308
Q2Male	0.0905691	0.1861388	-0.2662994	0.4573221
Q2Anothergenderidentity	-0.5775121	0.4960379	-1.5614825	0.3901047
sd___(Intercept)	1.3165885	0.6986090	0.3831148	3.0155173
sd_*disc*(Intercept)	0.2772493	0.1694131	0.0622315	0.7397092

We can now plot the distributions for the different cities.

```r
# Plotting distributions
as_draws_df(fit_vdh) %>%
    select(starts_with("r_City["), starts_with("r_City__disc[")) %>%
    pivot_longer(cols = everything(), names_to = "term", values_to = "value") %>%
    mutate(
        effect_type = case_when(
            grepl("^r_City__disc", term) ~ "r_City__disc",
            grepl("^r_City", term) ~ "r_City"
        ),
        city = sub(".*\\[(.*),.*", "\\1", term)
    ) %>%
    group_by(city, effect_type) %>%
    summarize(mean_value = mean(value), .groups = "drop") %>%
    pivot_wider(id_cols = city, names_from = effect_type, values_from = mean_value) %>%
    mutate(sigma = 1/exp(r_City__disc)) %>%
    expand(nesting(city, r_City, sigma),
           x = seq(from = -5.5, to = 5.5, length.out = 500)) %>%
    mutate(d = dlogis(x, r_City, sigma)) %>%
    ggplot(aes(x = x, y = d, fill = city)) +
    geom_area(alpha = 1/2, position = "identity") +
```

```
    geom_vline(xintercept = fixef(fit_vdh)[1:4, 1], linetype = 3) +
    scale_x_continuous(expression(Phi), breaks = -10:5,
                       sec.axis = dup_axis(
                           name = NULL,
                           breaks = fixef(fit_vdh)[1:4, 1] %>% as.double(),
                           labels = parse(text = str_c("alpha[", 1:4, "]"))
                       )) +
    scale_y_continuous(NULL, breaks = NULL) +
    coord_cartesian(xlim = c(-5, 5)) +
    theme_bw(base_size = 14) +
    theme(legend.position = c(0.18,0.7),
          legend.background = element_rect(fill = "transparent"),
          legend.key = element_rect(fill = "transparent"),
          legend.key.size = unit(0.3, "cm")) +
    labs(subtitle = "Varying hierarchical dispersion model", fill = "City")
```



We can see that there is some variation in the means of the distributions and in their dispersions.

Conclusion

In this document we gave an overview of how the probabilistic programming framework can be used to perform more advanced forms of analysis of ordinal survey data. We covered varying

thresholds and dispersion models.

If you want an overview of the basics of how to do probabilistic analysis of ordinal survey data, you can check out the file `prob_example.qmd`.