Item response models for LLM agent evaluation

Supplement to the BioDisco paper

Anonymous submission

1st August 2025

Table of contents

1 Introduction **1**
Requirements . . . 1

2 Paired comparison modelling **2**
Data wrangling . . . 2
Results . . . 5
Davidson model . . . 10

3 Temporal evaluation **10**

4 Human evaluation **11**
Visualizing Likert scales . . . 13
Bayesian ordinal mixed-effects model . . . 16
Bayesian posterior visualization . . . 22

1 Introduction

This document is a supplement to the "BioDisco" paper. It is used to generate the figures and tables for the evaluation section of the paper.

> **Note**
>
> We don't use the AAAI-2026 template for this document, because it forbids use of certain packages, such as `tabu`, which are useful (when writing with Quarto) for exploratory data analysis.

Requirements

For this analysis we need the following packages:

```
library(BradleyTerry2) # For modelling. Includes {qvcalc}
library(dplyr) # For data manipulation
library(tidyr) # For data reshaping
library(ggplot2) # For plotting
```

```
library(knitr) # For compilation & tables
library(purrr) # For functional programming
library(kableExtra)
theme_set(
    theme_bw(base_family = "Helvetica") +
        theme(
            panel.grid = element_blank(), # remove gridlines
            strip.background = element_blank() # remove shading from facet labels
        )
)
library(likert) # For visualizing Likert scales
library(brms) # For Bayesian modelling
library(tidybayes) # For inference with fitted Bayesian models
```

2 Paired comparison modelling

The data files contain the results from paired comparisons of LLMs evaluated on various metrics. At the time of rendering, available files are as follows:

```
list.files("data")
```

```
[1] "CVD_raw_scores.csv"             "Immunology_raw_scores.csv"
[3] "results_for_BT copy.csv"        "results_for_BT_final.csv"
[5] "results_for_BT_new_prompt.csv" "results_for_BT_testing.csv"
[7] "results_for_BT.csv"             "temporal-eval.csv"
```

```
results <- read.csv("data/results_for_BT_new_prompt.csv")
head(results) |> kable()
```

score	P1	P2	P1_win	P2_win	Tie
novelty	Single LLM	Multi-agent	8	86	6
novelty	Single LLM	Multi+Refine	0	100	0
novelty	Single LLM	Multi+Tools	1	99	0
novelty	Single LLM	BioDisco	0	100	0
novelty	Multi-agent	Multi+Refine	10	89	1
novelty	Multi-agent	Multi+Tools	6	94	0

Data wrangling

We need to convert the data into a format suitable for the Bradley-Terry model. The `P1_win` and `P2_win` columns indicate whether the first or second model won the comparison, respectively. They should be factors with the same levels.

```
all_players <- union(results$P1, results$P2)
results$P1 <- factor(results$P1, levels = all_players)
results$P2 <- factor(results$P2, levels = all_players)
```

The Bradley–Terry model is a paired comparison model that can be used to estimate the "ability" of each player (or LLM agent) based on pairwise comparisons. The model assumes that the probability of one player winning against another is a function of their abilities. The function `BTm` in package `BradleyTerry2` does **not** handle ties by default, so we need to handle them ourselves, by adding half a 'win' to each player in any tied match. However, we can also use the same package to fit a Davidson model, which is a bias-reduced model that handles ties. See Section 2.

```
fit_bradley_terry <- function(data, home_adv = FALSE, bias_reduced = FALSE) {
    data_for_model <- data %>%
        mutate(
            P1_win = P1_win + 0.5 * Tie,
            P2_win = P2_win + 0.5 * Tie
        )

    # Create player objects based on whether home_adv is included
    if (home_adv) {
        P1_obj <- data.frame(team = data_for_model$P1, home_adv = 1)
        P2_obj <- data.frame(team = data_for_model$P2, home_adv = 0)
        bt_formula <- ~ team + home_adv
    } else {
        P1_obj <- data.frame(team = data_for_model$P1)
        P2_obj <- data.frame(team = data_for_model$P2)
        bt_formula <- ~team
    }

    model <- BradleyTerry2::BTm(
        cbind(P1_win, P2_win),
        P1_obj, # Use the created player objects
        P2_obj, # Use the created player objects
        data = data_for_model,
        formula = bt_formula, # Use the dynamic formula
        family = binomial,
        br = bias_reduced,
        id = "team" # 'id' should refer to the column that identifies players/teams
    )

    model
}

fit_to_subset <- function(metric, data, ...) {
    subset_data <- filter(data, score == metric)
    fit_bradley_terry(subset_data, ...)
}

centre_scores <- function(model) {
    coefs <- coef(model)

    # Get player ability scores
    player_coefs <- coefs[grep("^team", names(coefs), value = TRUE)]
    # Get home-advantage coefficient if it exists
```

```
    home_adv_coef <- coefs["home_adv"]
    if (is.na(home_adv_coef)) {
        home_adv_coef <- NULL
    }

    player_scores <- setNames(c(0, player_coefs), all_players)
    player_scores <- player_scores - mean(player_scores)

    c(player_scores, home_adv = home_adv_coef)
}

get_quasi_variances <- function(model) {
    estimates <- centre_scores(model)[all_players]
    qv <- qvcalc::qvcalc(
        model,
        coef.indices = seq_along(all_players) - 1,
        estimate = estimates
    )
    # rownames(qv) <- NULL
    qv$qvframe$rowname <- all_players
    qv$qvframe$estimate <- estimates
    qv$qvframe <- relocate(qv$qvframe, rowname)
    qv
}
```

The following functions provide visualizations of the resulting plots and quasi-variance approximations.

```
centipede_plot <- function(qv_frame) {
    qv_df <- map_dfr(qv_list, "qvframe", .id = "metric") %>%
        # map_dfr(tibble::rownames_to_column, .id = "metric") %>%
        filter(rowname != "home_adv")

    ggplot(qv_df) +
        aes(
            forcats::fct_reorder(rowname, estimate),
            estimate,
            ymin = estimate - 1.96 * quasiSE,
            ymax = estimate + 1.96 * quasiSE
        ) +
        geom_linerange(alpha = .5) +
        geom_point() +
        coord_flip() +
        facet_wrap(~metric, scales = "free_x") +
        labs(x = NULL, y = "Ability score") +
        theme(
            strip.background = element_blank(),
            plot.margin = margin(0, 5, 0, 0),
            # text = element_text(family = "Times New Roman")
        )
}
```

```
beeswarm_plot <- function(qv_list) {
    relative_errors <- map(qv_list, "relerrs")
    data <- purrr::map_dfr(relative_errors, stack, .id = "metric") # values, ind

    ggplot(data) +
        aes(x = metric, y = values) +
        ggbeeswarm::geom_quasirandom() +
        labs(x = NULL, y = "Relative error") +
        coord_flip()
}
```

Results

With home advantage coefficient

Fit the model to the data.

```
metrics <- unique(results$score) %>% setNames(., .)
bt_models <- map(metrics, fit_to_subset, data = results, home_adv = TRUE)
```

```
Warning in eval(family$initialize): non-integer counts in a binomial glm!
Warning in eval(family$initialize): non-integer counts in a binomial glm!
Warning in eval(family$initialize): non-integer counts in a binomial glm!
Warning in eval(family$initialize): non-integer counts in a binomial glm!
```

```
qv_list <- map(bt_models, get_quasi_variances)
```

Present some summary statistics of the fitted models.

```
library(broom)
map_dfr(bt_models, broom::tidy, .id = "metric") |>
    kable()
```

```
Warning: The `tidy()` method for objects of class `BTm` is not maintained by the broom team

This warning is displayed once per session.
```

Table 2: Summary of Bradley-Terry models for each metric

metric	term	estimate	std.error	statistic	p.value
novelty	teamMulti-agent	2.8985177	0.4482469	6.4663416	0.0000000
novelty	teamMulti+Refine	5.8221805	0.6733227	8.6469390	0.0000000
novelty	teamMulti+Tools	6.4027395	0.9113220	7.0257708	0.0000000
novelty	teamBioDisco	9.0899822	1.1560232	7.8631487	0.0000000
novelty	home_adv	0.8026602	0.3604814	2.2266342	0.0259717

metric	term	estimate	std.error	statistic	p.value
relevance	teamMulti-agent	-0.0946304	0.1399995	-0.6759339	0.4990826
relevance	teamMulti+Refine	-0.3874814	0.1715601	-2.2585749	0.0239098
relevance	teamMulti+Tools	-0.0855453	0.2138123	-0.4000953	0.6890863
relevance	teamBioDisco	-0.2764607	0.2623261	-1.0538820	0.2919369
relevance	home_adv	0.0771199	0.1428515	0.5398608	0.5892931
significance	teamMulti-agent	1.5425902	0.2670156	5.7771539	0.0000000
significance	teamMulti+Refine	2.6454052	0.3782274	6.9942183	0.0000000
significance	teamMulti+Tools	3.7037721	0.4994204	7.4161404	0.0000000
significance	teamBioDisco	4.9452247	0.6586507	7.5081144	0.0000000
significance	home_adv	0.0676008	0.2400569	0.2816033	0.7782477
verifiability	teamMulti-agent	0.1024227	0.1510424	0.6781058	0.4977046
verifiability	teamMulti+Refine	-0.6218918	0.1784806	-3.4843660	0.0004933
verifiability	teamMulti+Tools	-0.5711104	0.2250495	-2.5377101	0.0111580
verifiability	teamBioDisco	-0.1385173	0.2753485	-0.5030618	0.6149208
verifiability	home_adv	0.4392820	0.1491073	2.9460804	0.0032183

```
map_dfr(bt_models, broom::tidy, .id = "metric") |>
    filter(term == "home_adv") |>
    # rename_with(tools::toTitleCase) |>
    kable("latex", booktabs = TRUE, digits = 2) %>%
    # collapse_rows(valign = "top") %>%
    cat()
```

```
\begin{table}

\caption{\label{tab:model summary}Summary of Bradley-Terry models for each metric}
\centering
\begin{tabular}[t]{llrrrr}
\toprule
metric & term & estimate & std.error & statistic & p.value\\
\midrule
novelty & home\_adv & 0.80 & 0.36 & 2.23 & 0.03\\
relevance & home\_adv & 0.08 & 0.14 & 0.54 & 0.59\\
significance & home\_adv & 0.07 & 0.24 & 0.28 & 0.78\\
verifiability & home\_adv & 0.44 & 0.15 & 2.95 & 0.00\\
\bottomrule
\end{tabular}
\end{table}
```

Make a centipede plot of the results.

```
centipede_plot(qv_list)
```

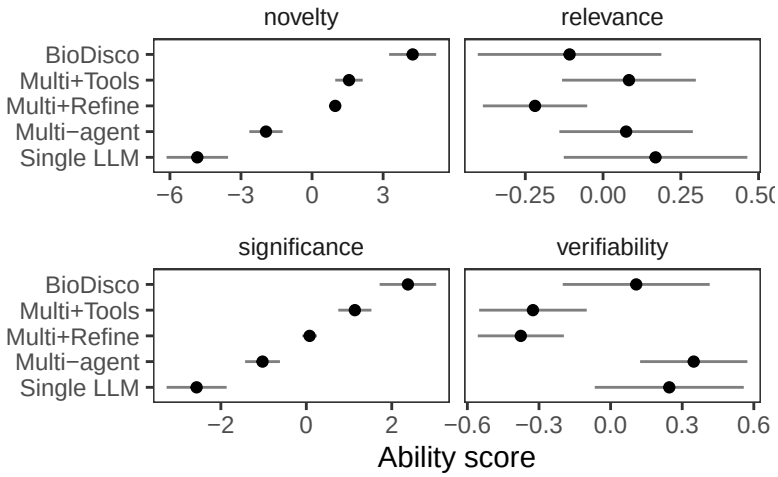


Figure 1: Centipede plot of Bradley-Terry model scores for each metric

Make a table of tables.

```
library(kableExtra)
qv_tbl <- qv_list %>%
    map_dfr("qvframe", .id = "metric") %>%
    as_tibble() %>%
    rename(model = rowname) %>%
    # mutate(metric = tools::toTitleCase(metric)) %>%
    rename_with(tools::toTitleCase)

qv_tbl %>%
    kable(digits = 2)
```

Table 3: Quasi-variance and point estimates for each metric

Metric	Model	Estimate	SE	quasiSE	quasiVar
novelty	Single LLM	-4.84	0.00	0.66	0.44
novelty	Multi-agent	-1.94	0.45	0.36	0.13
novelty	Multi+Refine	0.98	0.67	0.10	0.01
novelty	Multi+Tools	1.56	0.91	0.30	0.09
novelty	BioDisco	4.25	1.16	0.51	0.26
relevance	Single LLM	0.17	0.00	0.15	0.02
relevance	Multi-agent	0.07	0.14	0.11	0.01
relevance	Multi+Refine	-0.22	0.17	0.09	0.01
relevance	Multi+Tools	0.08	0.21	0.11	0.01
relevance	BioDisco	-0.11	0.26	0.15	0.02
significance	Single LLM	-2.57	0.00	0.36	0.13
significance	Multi-agent	-1.02	0.27	0.21	0.04
significance	Multi+Refine	0.08	0.38	0.09	0.01
significance	Multi+Tools	1.14	0.50	0.20	0.04
significance	BioDisco	2.38	0.66	0.34	0.11
verifiability	Single LLM	0.25	0.00	0.16	0.03
verifiability	Multi-agent	0.35	0.15	0.11	0.01
verifiability	Multi+Refine	-0.38	0.18	0.09	0.01

Metric	Model	Estimate	SE	quasiSE	quasiVar
verifiability	Multi+Tools	-0.33	0.23	0.11	0.01
verifiability	BioDisco	0.11	0.28	0.16	0.02

```
if (!is_latex_output()) {
    qv_tbl %>%
        kable("latex", booktabs = TRUE, digits = 2) %>%
        collapse_rows(valign = "top") %>%
        cat()
}
```

Make a beeswarm plot of the relative errors in the quasi-variance approximation for each metric.

```
beeswarm_plot(qv_list)
```

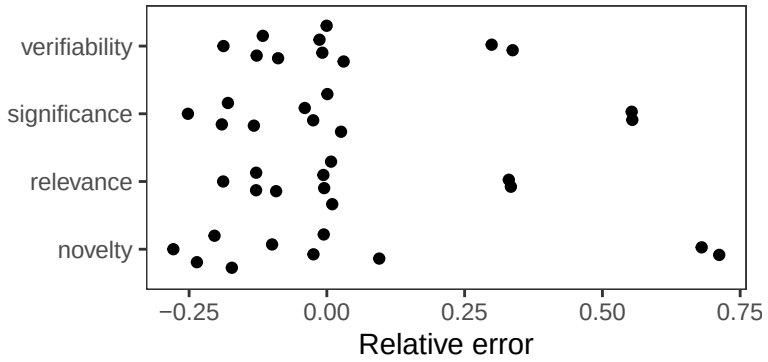


Figure 2: Beeswarm plot of relative errors for each metric

Without home advantage coefficient

Just given for reference. Will require some adjustment to the figure dimensions to make it look good in the PDF.

```
bt_models2 <- map(metrics, fit_to_subset, data = results, home_adv = FALSE)
```

```
Warning in eval(family$initialize): non-integer counts in a binomial glm!
Warning in eval(family$initialize): non-integer counts in a binomial glm!
Warning in eval(family$initialize): non-integer counts in a binomial glm!
Warning in eval(family$initialize): non-integer counts in a binomial glm!
```

```
qv_list2 <- map(bt_models2, get_quasi_variances)
centipede_plot(qv_list2)
```

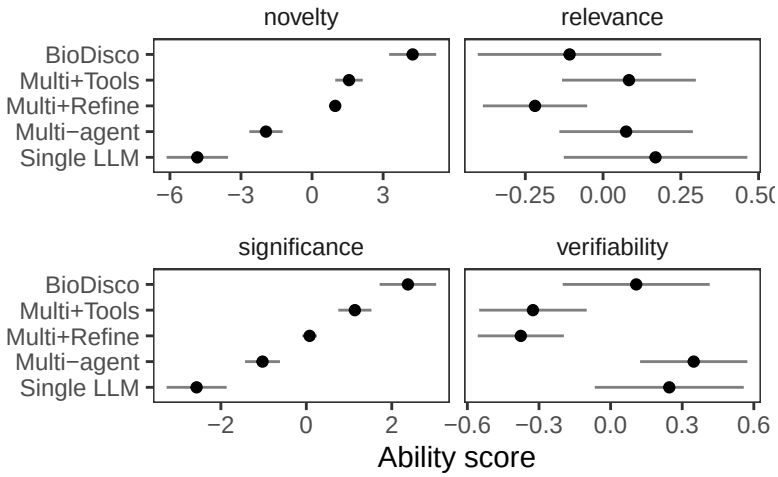


Figure 3: Centipede plot of Bradley-Terry model scores for each metric without home advantage

Beeswarm and tables:

```
beeswarm_plot(qv_list2)
```

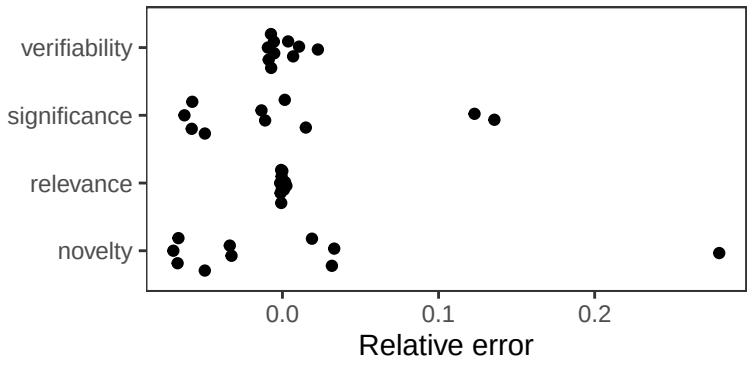


Figure 4: Relative errors for quasi-variance approximation without home advantage

```
qv_tbl2 <- qv_list2 %>%
    map_dfr("qvframe", .id = "metric") %>%
    as_tibble() %>%
    rename(model = rowname) %>%
    # mutate(metric = tools::toTitleCase(metric)) %>%
    rename_with(tools::toTitleCase)

qv_tbl2 %>%
    kable(digits = 2)
```

Metric	Model	Estimate	SE	quasiSE	quasiVar
novelty	Single LLM	-3.67	0.00	0.32	0.11
novelty	Multi-agent	-1.48	0.30	0.22	0.05
novelty	Multi+Refine	1.00	0.37	0.12	0.02
novelty	Multi+Tools	0.99	0.37	0.12	0.02

Metric	Model	Estimate	SE	quasiSE	quasiVar
novelty	BioDisco	3.17	0.42	0.22	0.05
relevance	Single LLM	0.23	0.00	0.09	0.01
relevance	Multi-agent	0.10	0.13	0.09	0.01
relevance	Multi+Refine	-0.22	0.13	0.09	0.01
relevance	Multi+Tools	0.05	0.13	0.09	0.01
relevance	BioDisco	-0.17	0.13	0.09	0.01
significance	Single LLM	-2.48	0.00	0.20	0.04
significance	Multi-agent	-0.99	0.21	0.13	0.02
significance	Multi+Refine	0.08	0.23	0.11	0.01
significance	Multi+Tools	1.10	0.25	0.13	0.02
significance	BioDisco	2.29	0.28	0.17	0.03
verifiability	Single LLM	0.60	0.00	0.10	0.01
verifiability	Multi-agent	0.51	0.13	0.10	0.01
verifiability	Multi+Refine	-0.36	0.14	0.09	0.01
verifiability	Multi+Tools	-0.50	0.14	0.10	0.01
verifiability	BioDisco	-0.25	0.14	0.09	0.01

Relative errors for quasi-variance approximation without home advantage

```
if (!is_latex_output()) {
    qv_tbl2 %>%
        kable("latex", booktabs = TRUE, digits = 2) %>%
        collapse_rows(valign = "top") %>%
        cat()
}
```

Davidson model

> **Caution**
>
> A Davidson model was fitted in an earlier version of this document (`pairwise.Rmd`), and may eventually be copied over. However it is not included in the paper.

3 Temporal evaluation

We produce a violin plot to compare the gold–gold (unrelated) hypotheses with the gold–generated (paired) hypotheses.

```
temporal <- read.csv("data/temporal-eval.csv") %>%
    setNames(c("vs. unrelated gold", "vs. generated")) %>%
    pivot_longer(everything()) %>%
    filter(!is.na(value))

ggplot(temporal, aes(x = name, y = value, fill = name)) +
    geom_violin() +
    geom_boxplot(width = .1) +
```

```
    labs(x = NULL, y = "Cosine similarity") +
    theme(legend.position = "none") +
    scale_fill_manual(values = c("#EC619F", "#6ABFA3")) +
    theme(plot.margin = margin(0, 0, 0, 0)) +
    coord_flip()
```

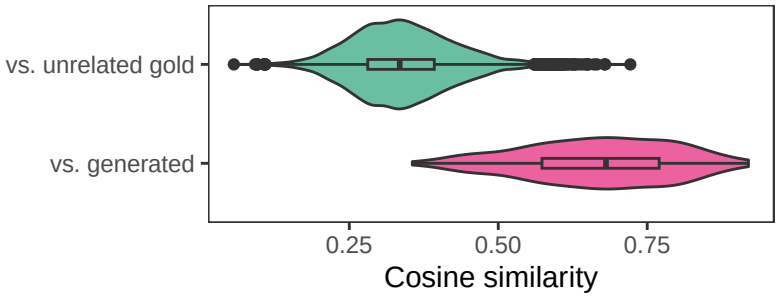


Figure 5: Violin plot of cosine similarities for the temporal evaluation

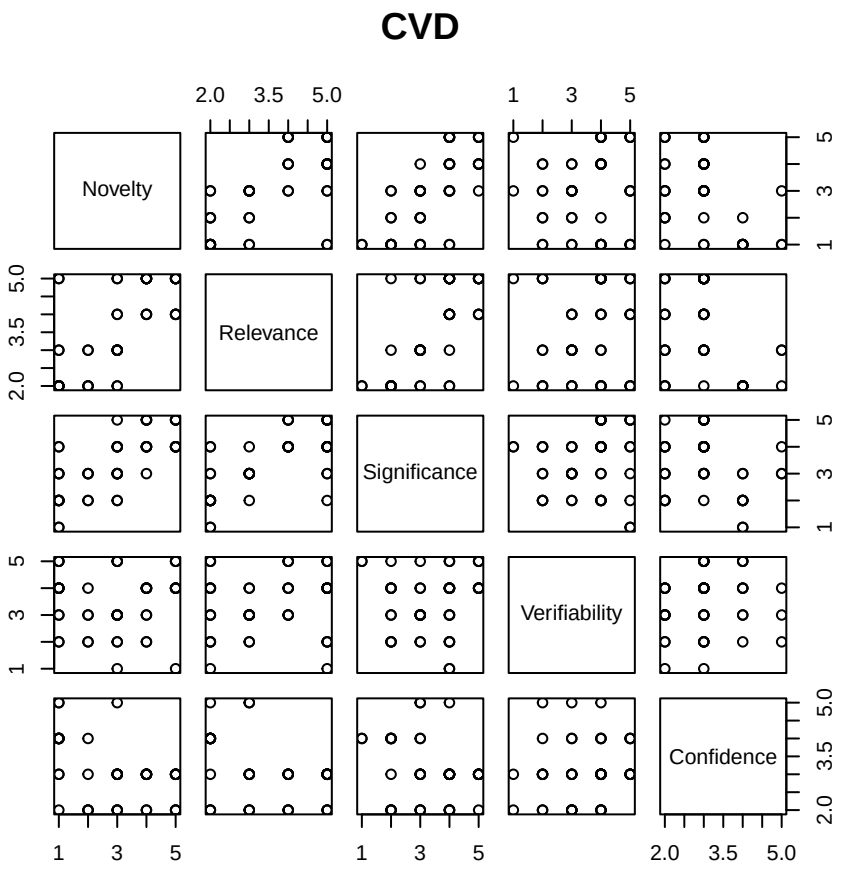
4 Human evaluation

The results of the human evaluation survey live in the following files:

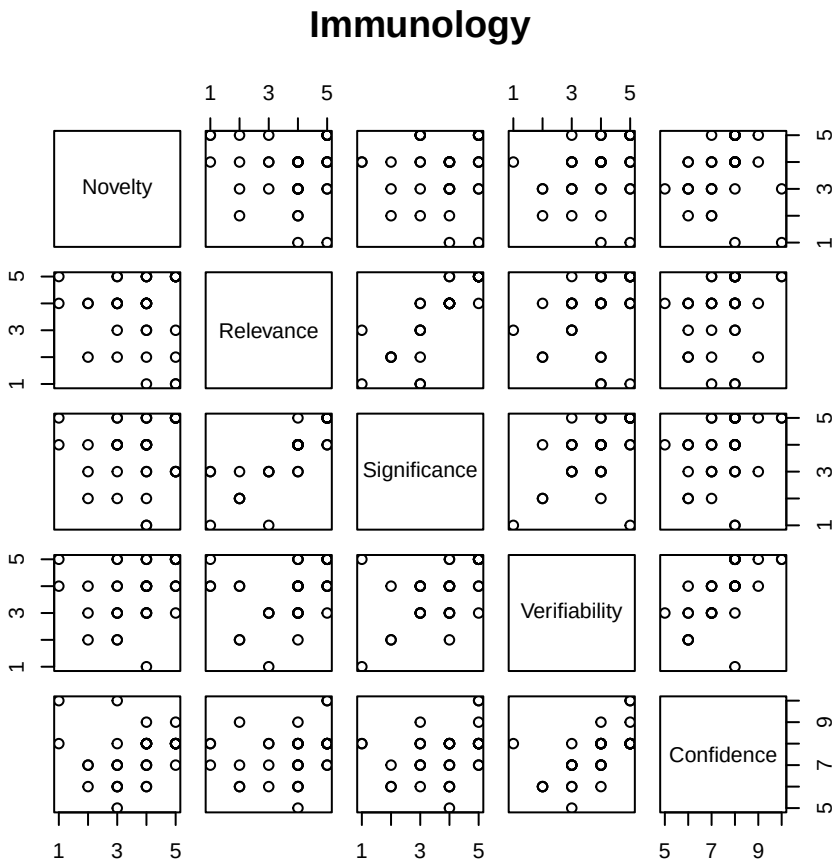
```
human_cvd <- read.csv("data/CVD_raw_scores.csv") |> mutate(topic = "CVD")
human_imm <- read.csv("data/Immunology_raw_scores.csv") |> mutate(topic = "Immunology")
human <- bind_rows(human_cvd, human_imm)
```

Look at correlations.

```
plot(human_cvd %>% select(Novelty:Confidence), main = "CVD")
```




```
plot(human_imm %>% select(Novelty:Confidence), main = "Immunology")
```



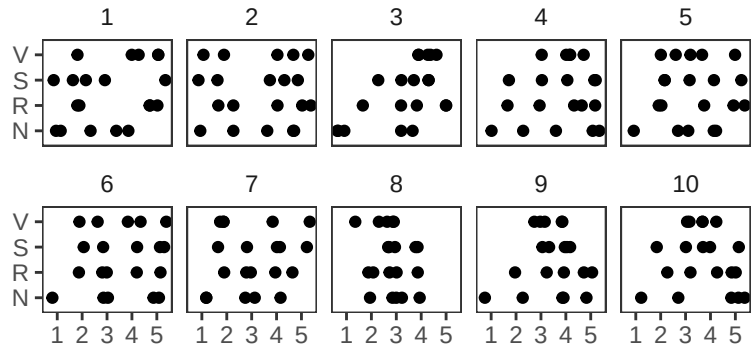
Get the data into a long format.

```
human_long <- human %>%
    select(-Confidence_num) %>%
    group_by(topic, Hypothesis) %>%
    mutate(rater_id = row_number()) %>%
    ungroup() %>%
    pivot_longer(Novelty:Confidence, names_to = "metric", values_to = "rating") %>%
    mutate(metric_abbr = stringr::str_sub(metric, 1, 1))
```

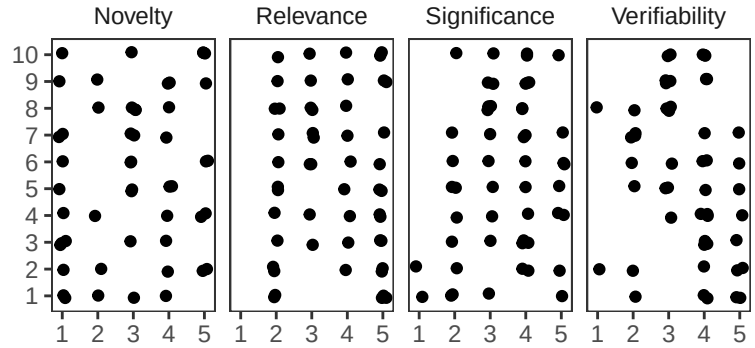
Visualizing Likert scales

Now we can try to plot the results.

```
# Plot by metric, facet by hypothesis
ggplot(human_long %>% filter(metric != "Confidence", topic == "CVD")) +
    aes(metric_abbr, rating) +
    geom_jitter(width = 0) +
    facet_wrap(~Hypothesis, nrow = 2) +
    coord_flip() +
    labs(x = NULL, y = NULL)
```




```
# Plot by hypothesis, facet by metric
ggplot(human_long %>% filter(metric != "Confidence", topic == "CVD")) +
    aes(as.factor(Hypothesis), rating) +
    geom_jitter(width = 0.1, height = 0.1) +
    facet_wrap(~metric, nrow = 1) +
    coord_flip() +
    labs(x = NULL, y = NULL)
```

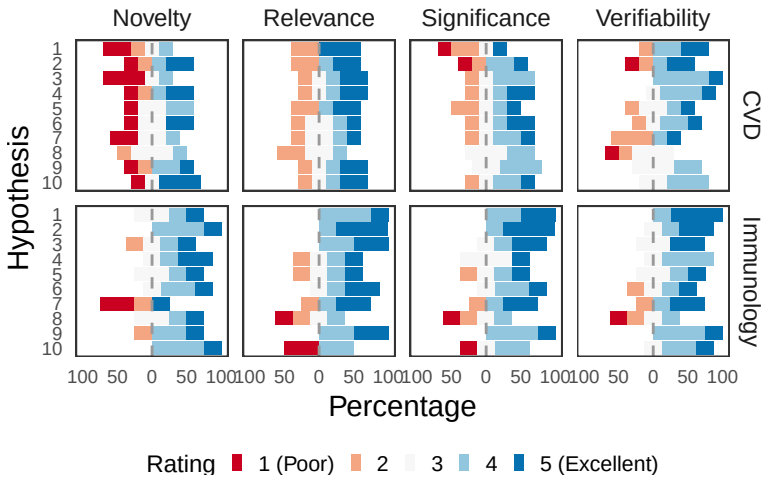


Not especially satisfying, to be honest. But we can try a stacked bar chart visualization inspired by those from the `likert` package.

```
likert_data <- human_long %>%
    filter(metric != "Confidence") %>%
    group_by(topic) %>%
    count(metric, Hypothesis, rating) %>%
    group_by(topic, metric, Hypothesis) %>%
    mutate(frac = n / sum(n)) %>%
    ungroup() %>%
    mutate(
        plot_value = case_when(
            rating %in% 1:2 ~ -frac,
            rating %in% 4:5 ~ frac,
            rating == 3 ~ frac / 2
        ),
        rating_label = factor(rating,
            levels = 1:5,
            labels = c("1 (Poor)", "2", "3", "4", "5 (Excellent)"),
            ordered = TRUE
```

```
        )
    ) %>%
    # Duplicate the 'neutral' rows and make their values negative
    # so that they appear on the left side of the plot
    bind_rows(
        filter(., rating == 3) %>%
            mutate(plot_value = -plot_value)
    )

plot_likert <- function(data, facet_by = "topic") {
    ggplot(data) +
        aes(
            x = plot_value,
            y = factor(Hypothesis, levels = 10:1),
            fill = rating_label
        ) +
        # Bars to the left
        geom_col(
            position = position_stack(reverse = FALSE),
            data = data %>% filter(plot_value < 0)
        ) +
        # Bars to the right
        geom_col(
            position = position_stack(reverse = TRUE),
            data = data %>% filter(plot_value > 0)
        ) +
        # Add a vertical line to mark the centre
        geom_vline(
            xintercept = 0,
            linetype = "dashed",
            colour = "grey60"
        ) +
        facet_grid(reformulate("metric", facet_by)) +
        scale_fill_brewer(
            type = "div", direction = 1, name = "Rating",
            palette = 5
        ) +
        scale_x_continuous("Percentage",
            breaks = seq(-1, 1, by = 0.5),
            labels = c(100, 50, 0, 50, 100),
            limits = c(-1, 1)
        ) +
        scale_y_discrete("Hypothesis") +
        theme(
            strip.background = element_blank(),
            legend.position = "bottom",
            panel.grid = element_blank(),
            plot.margin = margin(0, 0, 0, 0),
            axis.text = element_text(size = unit(7, "pt")),
            axis.ticks = element_blank(),
            legend.text = element_text(
```

```
                size = unit(8, "pt")
            ),
            legend.title = element_text(size = unit(9, "pt")),
            legend.margin = margin(0, 0, 0, 0),
            legend.key.size = unit(5, "pt"),
            legend.box.margin = margin(0, 0, 0, 0)
        )
}

plot_likert(likert_data)
```



> **Caution**
>
> Not implemented.

Bayesian ordinal mixed-effects model

But an even better approach for this might be to fit a statistical model. An graded response model (a generalization of a Rasch model in item-response theory) can take the following form:

Let Y_{ijm} be the ordinal rating (on a scale of 1 to 5) given by rater i to hypothesis j on metric m. The model estimates the cumulative probability $P(Y_{ijm} \leq k)$ for each rating category $k \in \{1, 2, 3, 4\}$. Using the probit link function (Φ^{-1}, the inverse of the standard normal cdf), the model is defined:

$$P(Y_{ijm} \leq k) = \Phi(\tau_k - \eta_{ijm}),$$

where the linear predictor contains the fixed and random effects:

$$\eta_{ijm} = \underbrace{\beta_m}_{\text{metric effect}} + \underbrace{u_i}_{\text{rater effect}} + \underbrace{v_{jm}}_{\text{hypo.-on-metric effect}}$$

and where τ_k are thresholds of the ordinal rating categories on the latent scale; β_m is a fixed effect representing the average 'quality' or 'easiness' for metric m, $u_i \sim N(0, \sigma_u^2)$ is a random

effect for rater i, representing their overall leniency or severity; $v_{jm} \sim N(0, \sigma_v^2)$ is the random effect for the quality of hypothesis j specifically on metric m.

We fit a single **Bayesian cumulative ordinal mixed-effects model** for all metrics, which has the advantage (over separately modelling each metric) of pooling ratings for more robust estimates, however with the strong assumption that the effect of 'rater leniency' is constant across the four metrics.

```
message("We have ", parallel::detectCores(), " cores available.")
```

```
We have 20 cores available.
```

```
options(mc.cores = min(4, parallel::detectCores()))
```

Model for CVD

```
brm_cvd <- brm(
    rating ~ metric + (1 | rater_id) + (1 | metric:Hypothesis),
    data = human_long %>% filter(metric != "Confidence", topic == "CVD"),
    family = cumulative("probit"),
    seed = 123,
    file = "brm_cvd.rds"
)

summary(brm_cvd)
```

```
 Family: cumulative
  Links: mu = probit; disc = identity
Formula: rating ~ metric + (1 | rater_id) + (1 | metric:Hypothesis)
   Data: human_long %>% filter(metric != "Confidence", topi (Number of observations: 200)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Multilevel Hyperparameters:
~metric:Hypothesis (Number of levels: 40)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.23      0.13     0.01     0.51 1.00     1068     1677

~rater_id (Number of levels: 5)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     1.22      0.53     0.54     2.53 1.00     1395     2193

Regression Coefficients:
                  Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept[1]         -1.38      0.54    -2.42    -0.27 1.00     1301     1972
Intercept[2]         -0.31      0.53    -1.32     0.80 1.00     1292     1951
Intercept[3]          0.58      0.53    -0.43     1.71 1.00     1292     1949
Intercept[4]          1.59      0.54     0.58     2.75 1.00     1304     2016
```

```
metricRelevance         0.74      0.25     0.24     1.24 1.00     3069     2426
metricSignificance      0.52      0.25     0.02     1.00 1.00     2586     2742
metricVerifiability     0.59      0.25     0.10     1.07 1.01     2919     2885

Further Distributional Parameters:
     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
disc     1.00      0.00     1.00     1.00   NA       NA       NA

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
# Extract and plot the estimated parameters for hypotheses and raters
random_effects_cvd <- ranef(brm_cvd)

# Hypothesis 'quality' scores
random_effects_cvd$`metric:Hypothesis`
```

```
, , Intercept

                    Estimate Est.Error       Q2.5     Q97.5
Novelty_1       -0.174208686 0.2600932 -0.8100555 0.2050408
Novelty_10       0.192178677 0.2674922 -0.1920698 0.8609321
Novelty_2        0.098324808 0.2279781 -0.3008343 0.6343681
Novelty_3       -0.230731284 0.2869937 -0.9418240 0.1675903
Novelty_4        0.098472113 0.2340740 -0.3182797 0.6557278
Novelty_5       -0.001288264 0.2134584 -0.4516745 0.4547069
Novelty_6        0.103991858 0.2375933 -0.3384210 0.6718175
Novelty_7       -0.139371409 0.2536009 -0.7516938 0.2649301
Novelty_8        0.008096726 0.2142932 -0.4439355 0.4539571
Novelty_9        0.038783907 0.2219880 -0.4072708 0.5451447
Relevance_1      0.070909573 0.2309552 -0.3582440 0.6253969
Relevance_10     0.046418074 0.2223345 -0.3903633 0.5456073
Relevance_2      0.006227210 0.2211086 -0.4583735 0.4865676
Relevance_3      0.053520315 0.2239131 -0.3900176 0.5672490
Relevance_4      0.058409593 0.2185753 -0.3659470 0.5696934
Relevance_5      0.010101799 0.2183460 -0.4458115 0.4967464
Relevance_6     -0.050663509 0.2231571 -0.5756033 0.3612223
Relevance_7     -0.050752722 0.2163720 -0.5534184 0.3791399
Relevance_8     -0.182441019 0.2597005 -0.8221887 0.2021294
Relevance_9      0.050399603 0.2279691 -0.3906243 0.5764610
Significance_1  -0.173319453 0.2603665 -0.8025484 0.2208724
Significance_10  0.039477451 0.2220135 -0.4038634 0.5466982
Significance_2  -0.051192956 0.2190629 -0.5519859 0.3777137
Significance_3  -0.012436722 0.2230540 -0.5012015 0.4702164
Significance_4   0.097819477 0.2353543 -0.3161001 0.6564078
Significance_5  -0.037782210 0.2195514 -0.5195294 0.3945542
Significance_6   0.097732831 0.2269104 -0.3106714 0.6352560
Significance_7   0.044110765 0.2144341 -0.3764254 0.5317611
Significance_8  -0.012981185 0.2179821 -0.4979517 0.4455145
Significance_9   0.028682482 0.2162001 -0.3950287 0.5139362
```

```
Verifiability_1   0.117427039 0.2434902 -0.2944853 0.7137811
Verifiability_10  0.007781476 0.2159063 -0.4248299 0.4759452
Verifiability_2  -0.018552081 0.2187454 -0.4922789 0.4202779
Verifiability_3   0.151051337 0.2511946 -0.2406575 0.7616373
Verifiability_4   0.108786694 0.2348762 -0.2907253 0.6732105
Verifiability_5  -0.022130029 0.2185272 -0.4954176 0.4295264
Verifiability_6   0.025490253 0.2158849 -0.4065804 0.5185785
Verifiability_7  -0.097264960 0.2237372 -0.6283535 0.2884183
Verifiability_8  -0.236518881 0.2805733 -0.9217775 0.1494829
Verifiability_9  -0.031660522 0.2138983 -0.5022446 0.4011680
```

```
# Rater 'leniency' scores
random_effects_cvd$rater_id
```

```
, , Intercept

    Estimate Est.Error        Q2.5       Q97.5
1 -1.1177497 0.5203233 -2.17680004 -0.05537351
2  0.7424988 0.5204033 -0.26208934  1.84408680
3 -0.6186062 0.5162044 -1.61661104  0.44897783
4  0.2915631 0.5142442 -0.68560434  1.39334919
5  1.0232271 0.5272035  0.01525896  2.15731689
```

We use weakly informative or flat default priors on all parameters and variance components:

```
get_prior(brm_cvd) |> kable()
```

prior	class	coef	group	resp	dpar	nlpar	lb	ub	source
	b								default
	b	metricRelevance							default
	b	metricSignificance							default
	b	metricVerifiability							default
student_t(3, 0, 2.5)	Intercept								default
	Intercept	1							default
	Intercept	2							default
	Intercept	3							default
	Intercept	4							default
student_t(3, 0, 2.5)	sd						0		default
	sd		metric:Hypothesis						default
	sd	Intercept	metric:Hypothesis						default
	sd		rater_id						default
	sd	Intercept	rater_id						default

Model for immunology

```r
brm_imm <- brm(
    rating ~ metric + (1 | rater_id) + (1 | metric:Hypothesis),
    data = human_long %>% filter(metric != "Confidence", topic == "Immunology"),
    family = cumulative("probit"),
    seed = 123,
    file = "brm_immunology.rds"
)

summary(brm_imm)
```

```
Warning: There were 2 divergent transitions after warmup. Increasing
adapt_delta above 0.8 may help. See
http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

 Family: cumulative
  Links: mu = probit; disc = identity
Formula: rating ~ metric + (1 | rater_id) + (1 | metric:Hypothesis)
   Data: human_long %>% filter(metric != "Confidence", topi (Number of observations: 160)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Multilevel Hyperparameters:
~metric:Hypothesis (Number of levels: 40)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.51      0.16     0.18     0.83 1.00     1087     1365

~rater_id (Number of levels: 4)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.87      0.53     0.29     2.30 1.00     1168     1926

Regression Coefficients:
                   Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept[1]          -1.71      0.52    -2.70    -0.64 1.00     1461     1587
Intercept[2]          -1.12      0.50    -2.07    -0.10 1.00     1457     1532
Intercept[3]          -0.38      0.49    -1.31     0.63 1.00     1435     1595
Intercept[4]           0.76      0.50    -0.14     1.82 1.00     1466     1607
metricRelevance        0.13      0.34    -0.55     0.80 1.00     1986     2504
metricSignificance     0.10      0.34    -0.58     0.78 1.00     2319     2579
metricVerifiability    0.16      0.34    -0.50     0.84 1.00     2251     2328

Further Distributional Parameters:
     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
disc     1.00      0.00     1.00     1.00   NA       NA       NA

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

```r
# Extract and plot the estimated parameters for hypotheses and raters
random_effects_imm <- ranef(brm_imm)

# Hypothesis 'quality' scores
random_effects_imm$`metric:Hypothesis`
```

```
, , Intercept

                   Estimate Est.Error       Q2.5     Q97.5
Novelty_1       -0.01714607 0.3733817 -0.7885958 0.7220313
Novelty_10       0.18811948 0.4073772 -0.5708601 1.0623589
Novelty_2        0.21941148 0.4007942 -0.5123484 1.0950897
Novelty_3       -0.10259792 0.3962874 -0.9389457 0.6507671
Novelty_4        0.25739632 0.4007957 -0.4769289 1.0893964
Novelty_5       -0.01583872 0.3875008 -0.7986799 0.7422470
Novelty_6        0.10174314 0.3770353 -0.6448195 0.8565316
Novelty_7       -0.61499737 0.4418887 -1.5825764 0.1442686
Novelty_8       -0.03846695 0.3835545 -0.8229786 0.7290512
Novelty_9        0.01422520 0.3814932 -0.7456399 0.7865617
Relevance_1      0.16260877 0.3893897 -0.5787430 0.9757335
Relevance_10    -0.61918085 0.4394792 -1.5618654 0.1312227
Relevance_2      0.51607535 0.4520130 -0.2452689 1.5215085
Relevance_3      0.33627516 0.4063849 -0.4228695 1.2041623
Relevance_4     -0.20754564 0.3806156 -1.0055346 0.5064340
Relevance_5     -0.16230226 0.3891687 -0.9755119 0.5729635
Relevance_6      0.20910494 0.3972571 -0.5532194 1.0449688
Relevance_7      0.11705229 0.3913395 -0.6269293 0.9212945
Relevance_8     -0.63442093 0.4453027 -1.5900412 0.1397700
Relevance_9      0.33260059 0.4148600 -0.3982563 1.1940518
Significance_1   0.35027789 0.4206193 -0.4137849 1.2424178
Significance_10 -0.41901108 0.4036512 -1.2910124 0.3160683
Significance_2   0.52009923 0.4590361 -0.2727464 1.5115439
Significance_3   0.21072601 0.4141468 -0.5457782 1.0829526
Significance_4  -0.22435823 0.3846736 -1.0125483 0.5086315
Significance_5  -0.15885157 0.3845575 -0.9596277 0.5672138
Significance_6   0.05149650 0.3884395 -0.7199727 0.8492237
Significance_7   0.12869750 0.3950031 -0.6260680 0.9501686
Significance_8  -0.61304458 0.4398538 -1.5600618 0.1425085
Significance_9   0.17413130 0.3856669 -0.5694239 0.9828561
Verifiability_1  0.51385534 0.4346423 -0.2358371 1.4613383
Verifiability_10 0.02928033 0.3842312 -0.7214610 0.8472767
Verifiability_2  0.20474652 0.4045138 -0.5503377 1.0787975
Verifiability_3  0.07410771 0.3857263 -0.6646271 0.8928726
Verifiability_4 -0.16376665 0.3766157 -0.9633135 0.5477089
Verifiability_5 -0.09497502 0.3854926 -0.8847993 0.6616186
Verifiability_6 -0.17520421 0.3841025 -0.9768839 0.5349562
Verifiability_7  0.10674152 0.3975771 -0.6650187 0.9194357
Verifiability_8 -0.63516765 0.4498972 -1.5801917 0.1539082
Verifiability_9  0.15131648 0.3966486 -0.6142805 0.9872977
```

```
# Rater 'leniency' scores
random_effects_imm$rater_id
```

```
, , Intercept

      Estimate Est.Error        Q2.5     Q97.5
1  0.873628515 0.4718731  0.06989941 1.9523065
2  0.007396534 0.4516199 -0.83901421 1.0069429
3 -0.311557314 0.4487088 -1.21167982 0.6700423
4 -0.113443828 0.4419953 -0.94653638 0.8395621
```

Bayesian posterior visualization

From this fitted model we obtain the following inferences. It's handy to use `tidybayes` for this, and to get the right variable names we use `variables(model)` (output omitted).

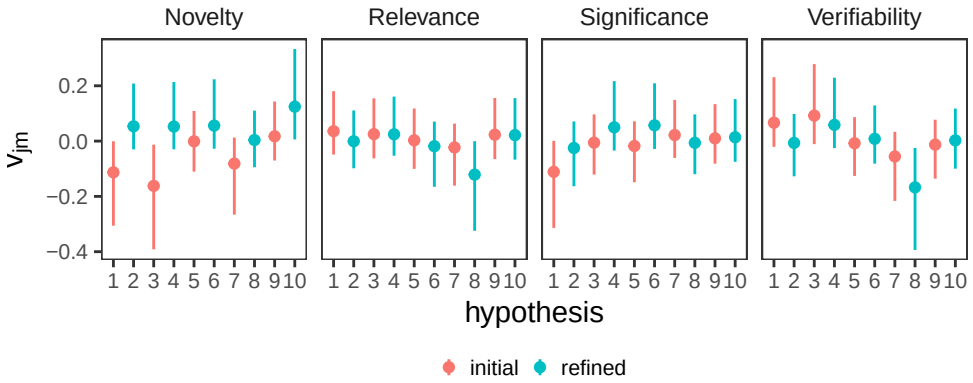
```
variables(brm_cvd) # check variable name syntax for spread_draws
```

Hypothesis quality scores by metric

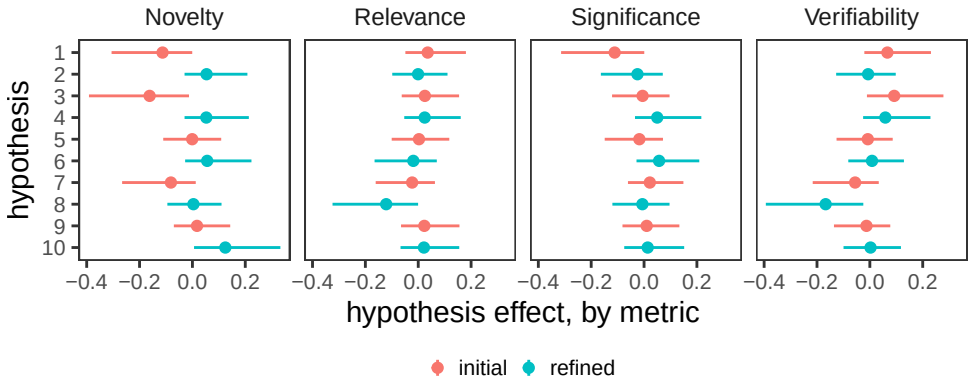
For cardiovascular disease:

```
# Helper functions for posterior credible intervals
# e.g. 50%, 80%, 95%
qlower <- function(x) quantile(x, 0.25)
qupper <- function(x) quantile(x, 0.75)

prepare_hypothesis_by_metric <- function(model, combo) {
    model %>%
        tidybayes::spread_draws(`r_metric:Hypothesis`[combo, term]) %>%
        tidyr::separate_wider_delim(combo, "_", names = c("metric", "hypothesis")) %>%
        group_by(metric, hypothesis) %>% # .chain
        summarise(
            lower = qlower(`r_metric:Hypothesis`),
            upper = qupper(`r_metric:Hypothesis`),
            median = median(`r_metric:Hypothesis`),
            .groups = "drop"
        ) %>%
        mutate(
            hypothesis = reorder(hypothesis, as.numeric(as.character(hypothesis))),
            refined = if_else(as.numeric(as.character(hypothesis)) %% 2 == 0, "refined", "in
        )
}

hypothesis_by_metric <- list(CVD = brm_cvd, Immunology = brm_imm) %>%
    map(prepare_hypothesis_by_metric) %>%
    bind_rows(.id = "topic")

plot_hypothesis_by_metric <- function(data) {
```

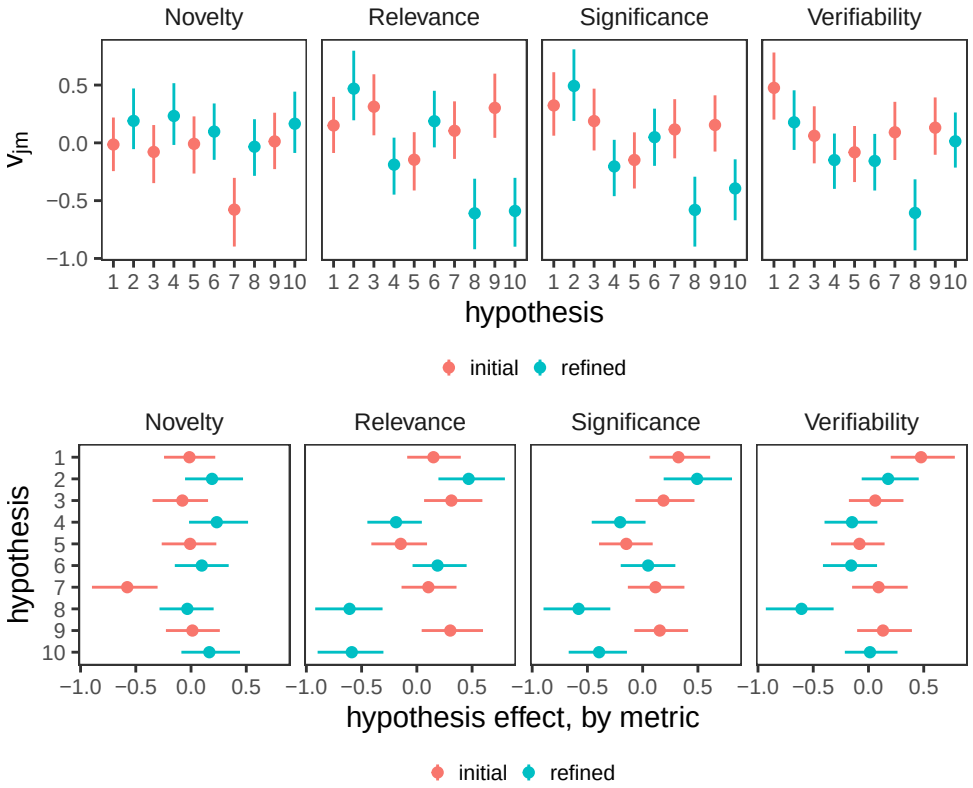
```
    ggplot(data) +
        aes(hypothesis, median, ymin = lower, ymax = upper, colour = refined) +
        geom_linerange() +
        geom_point() +
        facet_wrap(~metric, nrow = 1) +
        ylab(expression(v[jm])) +
        theme(
            legend.position = "bottom",
            panel.grid = element_blank(),
            plot.margin = margin(0, 0, 0, 0),
            axis.text = element_text(size = unit(8, "pt")),
            legend.text = element_text(
                size = unit(8, "pt")
            ),
            legend.title = element_text(size = unit(9, "pt")),
            legend.margin = margin(0, 0, 0, 0),
            legend.key.size = unit(5, "pt"),
            legend.box.margin = margin(0, 0, 0, 0)
        ) +
        scale_colour_discrete(NULL)
}

plot_hypothesis_by_metric2 <- function(data) {
    ggplot(data %>% mutate(hypothesis = factor(hypothesis, levels = 10:1))) +
        aes(hypothesis, median, ymin = lower, ymax = upper, colour = refined) +
        geom_linerange() +
        geom_point() +
        # scale_x_discrete(breaks = seq(2, 10, by = 2)) +
        ylab("hypothesis effect, by metric") +
        coord_flip() +
        facet_wrap(~metric, nrow = 1) +
        theme(
            legend.position = "bottom",
            panel.grid = element_blank(),
            plot.margin = margin(0, 0, 0, 0),
            axis.text = element_text(size = unit(8, "pt")),
            legend.text = element_text(
                size = unit(8, "pt")
            ),
            legend.title = element_text(size = unit(9, "pt")),
            legend.margin = margin(0, 0, 0, 0),
            legend.key.size = unit(5, "pt"),
            legend.box.margin = margin(0, 0, 0, 0)
        ) +
        scale_colour_discrete(NULL)
}

plot_hypothesis_by_metric(hypothesis_by_metric %>% filter(topic == "CVD"))
```



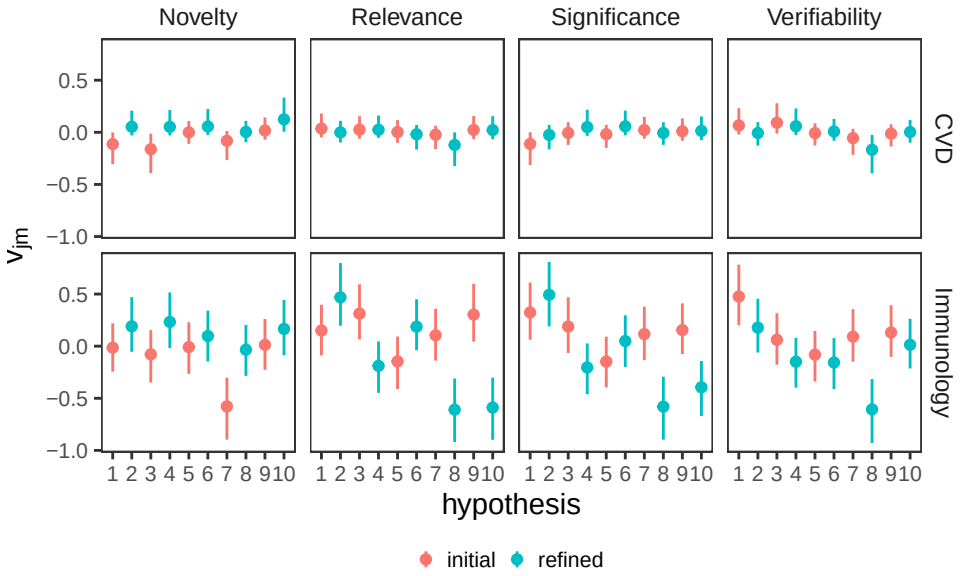

```
plot_hypothesis_by_metric2(hypothesis_by_metric %>% filter(topic == "CVD"))
```

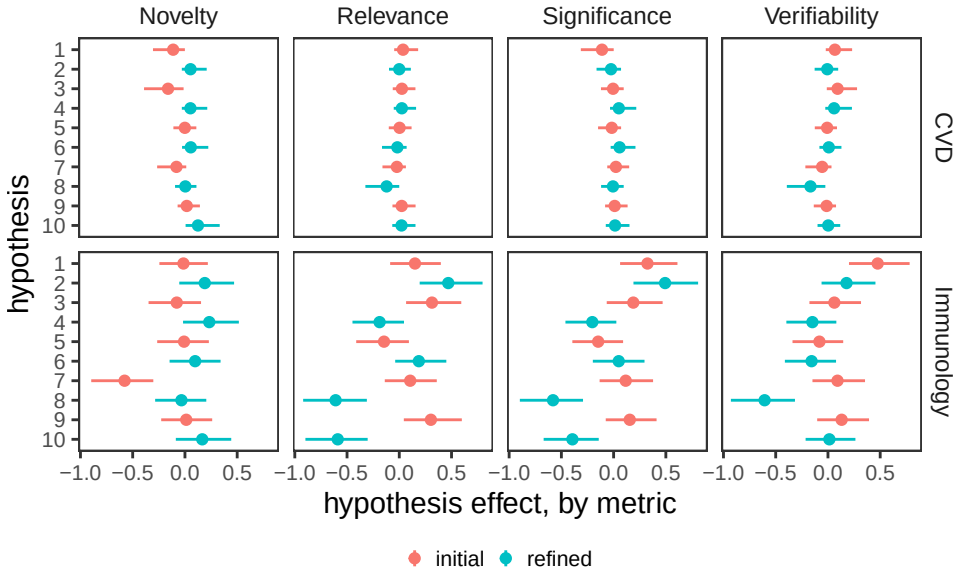


For immunology:



For both:



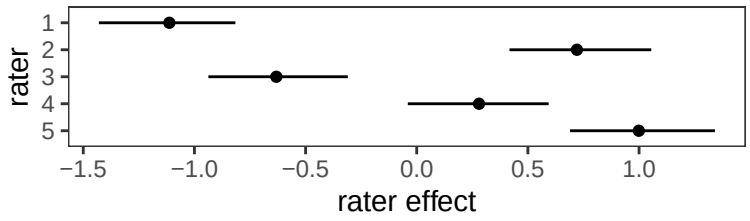


Rater leniency/severity

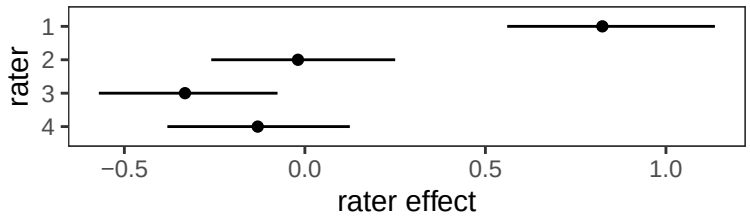
For CVD:

```
prepare_rater_leniency <- function(data) {
    data %>%
        spread_draws(r_rater_id[rater, term]) %>%
        group_by(rater) %>%
        summarise(
            lower = qlower(r_rater_id),
            upper = qupper(r_rater_id),
            median = median(r_rater_id),
            .groups = "drop"
        ) %>%
        mutate(rater = reorder(rater, as.numeric(rater), decreasing = TRUE))
}
```

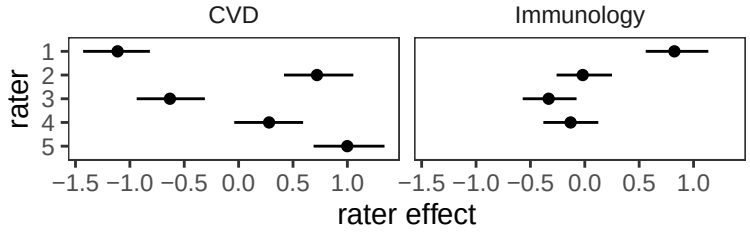
```
rater_leniency <- list(CVD = brm_cvd, Immunology = brm_imm) %>%
    map(prepare_rater_leniency) %>%
    bind_rows(.id = "topic")

ggplot(rater_leniency %>% filter(topic == "CVD")) +
    aes(rater, median, ymin = lower, ymax = upper) +
    geom_linerange() +
    geom_point() +
    ylab("rater effect") +
    coord_flip()
```



For immunology:



For both:



Selected hypotheses

We can also use `brms::posterior_predict()` or `tidybayes::add_predicted_draws()` to generate the expected distribution of ratings for selected hypotheses of interest, for example the best and worst for novelty. We would expect the 'best' hypothesis to show a high probability of receiving 4s or 5s, and the 'worst' hypothesis might show a high(er) probability of receiving 1s and 2s.

> **Caution**
>
> Not implemented.