Package ‘deliberr’

December 19, 2025

Type Package

Title Methods for Deliberation Analysis

Version 0.1.0

URL https://github.com/gumbelino/deliberr,
https://gumbelin.shinyapps.io/deliberr

BugReports https://github.com/gumbelino/deliberr/issues

Description An implementation of deliberative reasoning index (DRI) and related tools for analysis of deliberation survey data. Calculation of DRI, plot of intersubjective correlations (IC), generation of large-language model (LLM) survey data, and permutation tests are supported. Example datasets and a graphical user interface (GUI) are also available to support analysis. For more information, see Niemeyer and Veri (2022) <doi:10.1093/oso/9780192848925.003.0007>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports stats, dplyr, shiny, tibble, ggplot2, rstatix, grid, psych, tidyr, rlang, glue, httr, purrr, readr, jsonlite, lifecycle, uuid,

RoxygenNote 7.3.3

Suggests DT, gridExtra, knitr, readxl, rmarkdown

Depends R (>= 3.5)

NeedsCompilation no

Author Gustavo Kreia Umbelino [aut, cre] (ORCID: <https://orcid.org/0000-0001-7754-9606>),
Francesco Veri [aut] (ORCID: <https://orcid.org/0000-0002-9295-6776>),
Swiss National Science Foundation (SNSF) [fnd]

Maintainer Gustavo Kreia Umbelino <gustavo.kreiaumbelino@zda.uzh.ch>

Repository CRAN

Date/Publication 2025-12-19 14:50:02 UTC

Contents

`format_dri_survey` *Format DRI survey*

Description

`format_dri_survey` helps transform raw survey data into useful objects for further manipulation with `deliberr`

Usage

```
format_dri_survey(
  survey_info = list(type = NA_character_, order = NA_integer_, statement =
    NA_character_, name = NA_character_, scale_max = NA_integer_, q_method = NA)
)
```

Arguments

`survey_info` survey information needed to format DRI survey

Value

A list of survey info, including `name`, `considerations` data, `policies` data, `scale_max` or the upper bound of Likert-scale survey questions, and `q_method` which flags whether the survey uses Q methodology

See Also

surveys for raw survey data formatting

Examples

```
dri_survey <- format_dri_survey(surveys[surveys$name == "acp", ])

dri_survey$name
dri_survey$considerations
```

get_dri *Get DRI from a Group of Participants*

Description

`get_dri` calculates the deliberation reasoning index (DRI) for a group of deliberation participants

Usage

```
get_dri(ic, adjusted = TRUE)
```

Arguments

`ic`	dataframe generated by `get_dri_ic(data)`
`adjusted`	a logical indicating whether to use the original or adjusted DRI calculation formula

Value

the group-level DRI value

See Also

`get_dri_ic()` to generate `ic` parameter

Other IC methods: `get_dri_ind()`

Examples

```
# get pre-deliberation (stage_id == 1) data from BEP case
data <- human_data[human_data$stage_id == 1 & human_data$case == "BEP", ]

# calculate IC
ic <- get_dri_ic(data)

# generate DRI
get_dri(ic)
```

```
# same as the mean of individual DRIs
mean(get_dri_ind(ic)$dri)
```

get_dri_alpha *Get DRI Cronbach's Alpha*

Description

get_dri_alpha calculates the internal consistency of DRI survey responses using Cronbach's alpha

Usage

```
get_dri_alpha(data)
```

Arguments

data	the raw DRI survey response data

Value

a dataframe with alpha_c, alpha_p, and alpha_all with values of Cronbach's alpha for considerations, policy preferences, and both, respectively

See Also

human_data for raw survey response data formatting

psych::alpha() for details on Cronbach's alpha calculation

Other DRI survey methods: get_dri_case(), get_dri_ic()

Examples

```
# get pre-deliberation (stage_id == 1) data from Mayo case
data <- human_data[human_data$stage_id == 1 & human_data$case == "Mayo", ]
get_dri_alpha(data)
```

get_dri_case *Get DRI from Case*

Description

get_dri_case calculates the pre- and post-deliberation DRI from a specific deliberation case

Usage

```
get_dri_case(
  case,
  adjusted = TRUE,
  method = "wilcox",
  alternative = "greater",
  data = NULL
)
```

Arguments

case	a character string specifying the name of the case in human_data
adjusted	a logical indicating whether you want the original or adjusted DRI formula
method	a character string specifying the method for statistical testing, must be one of "wilcox" (default) or "t.test"
alternative	a character string specifying the alternative hypothesis, must be one of "greater" (default), "two.sided" or "less". You can specify just the initial letter
data	a survey responses dataframe, must include pre- (stage_id == 1) and post-deliberation (stage_id == 2) data

Value

a tibble with with the following components: case, pre, post, delta, p_value, and significance

See Also

human_data for raw survey response data formatting

Other DRI survey methods: get_dri_alpha(), get_dri_ic()

Examples

```
get_dri_case("Activate")

# same as
get_dri_case("Activate", data = human_data)
```

get_dri_ic *Get DRI Intersubjective Consistency (IC)*

Description

get_dri_ic calculates the intersubjective consistency (IC) between deliberation participants using their DRI survey responses

Usage

```
get_dri_ic(data)
```

Arguments

data	the raw DRI survey response data

Value

dataframe with pnums or participant numbers, pnum1 and pnum2, or the unique number of participant 1 and 2, respectively, ccor and pcor, or the correlation between considerations statement ratings and policy preference rankings, respecively, and dj, or the modal orthogonal distance d for a given pair j

See Also

human_data for raw survey response data formatting

Other DRI survey methods: get_dri_alpha(), get_dri_case()

Examples

```
# get post-deliberation (stage_id == 2) data from Zukunft case
data <- human_data[human_data$stage_id == 2 & human_data$case == "Zukunft", ]
get_dri_ic(data)
```

get_dri_ind *Get DRI from Individual Participants*

Description

get_dri_ind calculates the DRI for each individual participant at a deliberation

Usage

```
get_dri_ind(ic, adjusted = TRUE)
```

Arguments

`ic`	dataframe generated by `get_dri_ic(data)`
`adjusted`	a logical indicating whether to use the original or adjusted DRI calculation formula

Value

tibble with `pnum` as participant number and their respective individual `dri`

See Also

`get_dri_ic()` to generate `ic` parameter

Other IC methods: `get_dri()`

Examples

```
# get post-deliberation (stage_id == 2) data from Zukunft case
data <- human_data[human_data$stage_id == 2 & human_data$case == "Zukunft", ]

# generate IC
ic <- get_dri_ic(data)

# get individual DRIs
get_dri_ind(ic)
```

`get_dri_llm_response` *Get DRI LLM Response*

Description

`get_dri_llm_response` uses `https://openrouter.ai` to generate artificial LLM responses to DRI survey questions

Usage

```
get_dri_llm_response(
  model_id,
  survey_info = list(type = NA_character_, order = NA_integer_, statement =
    NA_character_, name = NA_character_, scale_max = NA_integer_, q_method = NA),
  api_key = Sys.getenv("OPENROUTER_API_KEY"),
  role_info = list(uid = NA_character_, role = NA_character_, description =
    NA_character_),
  n = 1,
  request_log_path = NA_character_
)
```

Arguments

`model_id` a model_id string from openrounter.ai

`survey_info` a list with survey question information, including `type`, `order`, `statement`, `name`, `scale_max`, and `q_method`

`api_key` the API key generated by OpenRouter

`role_info` a named list with basic data of a role (i.e., `uid`, `role`, `description`)

`n` the number of responses requested (default = 1)

`request_log_path`
an optional path to a file where the request texts are saved

Value

a dataframe with `n` survey responses by `model_id`, including a unique identifier, `uuid`, a creation timestamp, `created_at_utc`, the time it took to generate the response, `time_s`, the estimated cost in USD, `est_cost_usd`, whether the response is valid, `is_valid`, and the reason it is not, `invalid_reason`

See Also

`get_model_ids()` for all currently available model ids from openrouter.ai

Other LLM methods: `get_model_ids()`, `make_dri_llm_prompts()`

Examples

```
# get DRI survey
survey_info <- surveys[surveys$name == "acp",]

# select a model from openrouter
model_id <- "google/gemini-2.5-flash-lite"

# send request to openrouter API
## Not run:
llm_data <- get_dri_llm_response(model_id, survey_info)
## End(Not run)
```

`get_llm_response` *Get LLM Response from OpenRouter.ai*

Description

`get_llm_response` sends a prompt to a specified large language model (LLM) through the OpenRouter.ai API and returns the text response. It can also manage conversation history

Usage

```
get_llm_response(
  user_prompt,
  model_id = "x-ai/grok-3-mini",
  system_prompt = NA_character_,
  context = NULL,
  temperature = 0,
  api_key = Sys.getenv("OPENROUTER_API_KEY")
)
```

Arguments

`user_prompt` a string containing the prompt or question for the model

`model_id` a string specifying the model to use (e.g., "google/gemini-flash-1.5"). You can find model names on the OpenRouter.ai website

`system_prompt` A string defining the role or behavior of the model. This is only used for the first message in a conversation (when ’context’ is NULL)

`context` a list representing the conversation history. If provided, the ’system_prompt’ is ignored, as the context is assumed to contain the full history. Defaults to NULL for a new conversation

`temperature` a numeric value between 0 and 2 that controls the randomness of the model’s output. Higher values mean more "creative" responses

`api_key` a string containing your OpenRouter.ai API key. It is strongly recommended to use the default, which retrieves the key from an environment variable named `OPENROUTER_API_KEY`

Value

A list containing three elements: `response`, `context`, and `cost` `cost` is a list containing `prompt_cost`, `completion_cost`, and `total_cost` in USD

Examples

```
## Not run:
# Make sure to set your API key first
# Sys.setenv(OPENROUTER_API_KEY = "your_api_key_here")

# First turn of the conversation
first_turn <- get_llm_response(
  user_prompt = "What are the three main benefits of using R for data analysis?",
  model_id = "x-ai/grok-3-mini",
  system_prompt = "You are a helpful assistant who provides concise answers."
)
cat("--- Initial Response ---\n")
cat(first_turn$response)
cat(paste0("\n--- Total Cost: $",
format(first_turn$cost$total_cost, scientific = FALSE), " ---\n"))
```

```
# Follow-up question using the context from the first turn
second_turn <- get_llm_response(
  user_prompt = "Can you elaborate on the second benefit you mentioned?",
  model_id = "x-ai/grok-3-mini",
  context = first_turn$context
)
cat("\n\n--- Follow-up Response ---\n")
cat(second_turn$response)
cat(paste0("\n--- Total Cost: $", format(second_turn$cost$total_cost,
scientific = FALSE), " ---\n"))

## End(Not run)
```

`get_model_ids` *Get Model IDs*

Description

`get_model_ids` uses OpenRouter to get `provider` and `model` names. The `model_id` can be recreated as `provider/model`

Usage

```
get_model_ids()
```

Value

a dataframe with columns `provider` and `model`

See Also

Other LLM methods: `get_dri_llm_response()`, `make_dri_llm_prompts()`

Examples

```
get_model_ids()
```

`human_data` *Human data*

Description

Pre- and post-deliberation DRI survey data from 24 deliberation cases around the world. Some cases used the same survey.

Usage

```
human_data
```

Format

A data frame with 67 variables, including `survey`, `case`, `stage_id`, `C1...C50` and `P1...P10`.

`make_dri_llm_prompts` *Make DRI LLM Prompts*

Description

`make_dri_llm_prompts` creates the system and user prompts used for generating LLM DRI survey data

Usage

```
make_dri_llm_prompts(
  dri_survey,
  role_info = list(uid = NA_character_, role = NA_character_, description =
    NA_character_)
)
```

Arguments

`dri_survey`	a list of formatted DRI survey questions
`role_info`	information about a specific role, including unique identifier `uid`, `role` name, and role `description`

Value

a list of lists with four variables: `system`, `considerations`, `policies`, and `reason` prompts

See Also

`format_dri_survey()` for how to format `dri_survey`

prompts for how prompts are formatted

Other LLM methods: `get_dri_llm_response()`, `get_model_ids()`

Examples

```
# get ccps as an example survey
dri_survey <- format_dri_survey(surveys[surveys$name == "ccps",])

# create an example role from scratch
role_info <- list(
  uid = "sur",
  role = "surfer",
  description = "likes the ocean"
)

make_dri_llm_prompts(dri_survey, role_info)
```

open_gui *Open Graphical User Interface (GUI)*

Description

open_gui uses shiny to open an interactive interface for code-free DRI analysis

Usage

```
open_gui()
```

Value

NA

Examples

```
## Not run:
open_gui()

## End(Not run)
```

permute_dri *Permute DRI*

Description

permute_dri tests whether the links between considerations and policy preferences are consistent or likely due to chance

Usage

```
permute_dri(data, iterations = 10000, verbose = FALSE, summary = TRUE)
```

Arguments

`data`	raw DRI survey dataframe
`iterations`	number permutations to generate
`verbose`	a logical flag to print time of permutation
`summary`	a logical indicating whether to return the raw data or summary of test results; raw data is optimal for plotting permutation results

Value

dataframe with permutation test results, raw or summarized. Summarized results include the number of participants, `n`, the observed DRI, `obs_dri`, the number of permutations conducted, `n_perm`, the mean permutation DRI, `mean_perm_dri`, and the frequency which the permutation DRI is greater or equal to the observed DRI, `p`

Examples

```
# get pre-deliberation (stage_id == 1) data from Zukunft case
data <- human_data[human_data$stage_id == 1 & human_data$case == "Zukunft", ]

# permute DRI 100 times
permute_dri(data, iterations = 100)
```

`plot_dri_ic` *Plot DRI Intersubjective Consistency (IC)*

Description

`plot_dri_ic` creates a dot plot of deliberation IC where each dot represents a pair of participants

Usage

```
plot_dri_ic(
  ic,
  title = NA_character_,
  suffix = NA_character_,
  dri = NA_real_,
  caption = NULL
)
```

Arguments

`ic`	dataframe generated by `get_dri_ic(data)`
`title`	title of the plot
`suffix`	string to be added after the title separated by `:`
`dri`	numeric value generated by `get_dri(ic)`; if omitted, `get_dri` is called by default
`caption`	a string to be displayed under the plot

Value

an IC plot

See Also

`get_dri_ic()` for how to generate the `ic` parameter

`get_dri()` for how to generate the `dri` parameter

Examples

```
# get post-deliberation (stage_id == 2) data from Zukunft case
data <- human_data[human_data$stage_id == 2 & human_data$case == "Zukunft", ]

# set plot optional parameters
title <- "Case Zukunft"
suffix <- "Post-Deliberation IC Plot"
caption <- "this is an example plot"

# calculate ic
ic <- get_dri_ic(data)

plot_dri_ic(ic, title, suffix, caption = caption)
```

---

prompts *Prompts*

Description

Prompts used in DRI surveys for humans and LLMs.

Usage

```
prompts
```

Format

A data frame with two variables: `type` and `prompt`

`roles` *Roles*

Description

Roles used to generate LLM role-playing data.

Usage

```
roles
```

Format

A data frame with five variables: `uid`, `type`, `article`, `role` and `description`.

`summarize_perm_dri` *Summarize DRI Permutation Test Results*

Description

`summarize_perm_dri` summarizes the results of a permutation test done using `permute_dri(..., summary = FALSE)`; useful for summarizing results after plotting permutation results

Usage

```
summarize_perm_dri(perms, type = "common")
```

Arguments

`perms`	results of the permutation test generated by `permute_dri()`
`type`	which type of statistics to summarize (e.g., `"common"`, `"robust"`, `"mean"`)

Value

summary of permutation test

See Also

`permute_dri()` for generating the `perms` parameter

`rstatix::get_summary_stats()` for values of `type`

Examples

```
# get pre-deliberation (stage_id == 1) data from Zukunft case
data <- human_data[human_data$stage_id == 1 & human_data$case == "Zukunft", ]

# create permutations
perms <- permute_dri(data, iterations = 100, summary = FALSE)

summarize_perm_dri(perms)
```

---

surveys *Surveys*

Description

All survey data used in deliberations.

Usage

```
surveys
```

Format

A data frame with six variables: `type`, `order`, `statement`, `name`, `scale_max` and `q_method`.

Index