Package ‘shinymrp’

October 9, 2025

Title Interface for Multilevel Regression and Poststratification

Version 0.9.0

Author Toan Tran [cre, aut, cph],
Jonah Gabry [aut, cph],
Yajuan Si [aut, cph]

Maintainer Toan Tran <`trannttoan97@gmail.com`>

Description Dual interfaces, graphical and programmatic, designed for intuitive applications of Multilevel Regression and Poststratification (MRP). Users can apply the method to a variety of datasets, from electronic health records to sample survey data, through an end-to-end Bayesian data analysis workflow. The package provides robust tools for data cleaning, exploratory analysis, flexible model building, and insightful result visualization. For more details, see Si et al. (2020) <`https://www150.statcan.gc.ca/n1/en/pub/12-001-x/2020002/article/00003-eng.pdf?st=iF1_Fbrh`> and Si (2025) <doi:10.1214/24-STS932>.

License MIT + file LICENSE

URL `https://mrp-interface.github.io/shinymrp/`

BugReports `https://github.com/mrp-interface/shinymrp/issues`

Depends R (>= 3.6.0)

Imports bsicons, bslib, checkmate, config (>= 0.3.2), dplyr, DT, ggplot2, golem (>= 0.4.1), highcharter, htmlwidgets, httr2, loo, lubridate, magrittr, patchwork, posterior, purrr, qs, R6, RColorBrewer, readr, rlang, scales, shiny (>= 1.8.0), shinyjs, shinyWidgets, stringr, tidyr, waiter

Suggests cmdstanr, knitr, rmarkdown, spelling, testthat (>= 3.0.0)

Additional_repositories `https://stan-dev.r-universe.dev/`

Config/testthat/edition 3

Encoding UTF-8

Language en-US

RoxygenNote 7.3.2

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2025-10-09 12:20:09 UTC

Contents

`example_model` *Return example* `MRPModel` *object with estimation results.*

Description

Return an example `MRPModel` object with estimation results.

Usage

```
example_model(is_timevar = TRUE)
```

Arguments

`is_timevar`	Logical indicating whether the model is fitted to time-varying data.

Value

A `MRPModel` object.

`example_pstrat_data` *Return example poststratification data*

Description

Return example poststratification data accepted by the `$load_pstrat()` method of an `MRPWorkflow` object.

Usage

```
example_pstrat_data()
```

Value

A `data.frame` object.

`example_sample_data` *Return example data*

Description

Return example data based on the specified characteristics.

Usage

```
example_sample_data(
  is_timevar = TRUE,
  is_aggregated = TRUE,
  special_case = NULL,
  family = "binomial"
)
```

Arguments

`is_timevar`	Logical indicating whether the data is time-varying.
`is_aggregated`	Logical indicating whether the data is aggregated.
`special_case`	Optional character string for specific use cases such as COVID data. Options are `NULL`, `"covid"` and `"poll"`. The default is `NULL` which indicates the data is not specific to any supported use case.
`family`	Character string specifying the distribution family for outcome measures. Options are `"binomial"` for binary outcomes and `"normal"` for continuous outcomes.

Value

A `data.frame` object.

`MRPModel` *MRPModel objects*

Description

An `MRPModel` object is an R6 object created by the `$create_model()` method of an `MRPWorkflow` object. Each `MRPModel` object represents a multilevel regression model, providing methods for sampling, diagnostics, and poststratification.

Creates a new instance of the `MRPModel` class. This method is called by the `$create_model()` method of an `MRPWorkflow` object and does not need to be called directly by users.

Arguments

`model_spec`	List containing model effects specification, including intercept, fixed effects, varying effects, and interactions
`mrp_data`	List containing the MRP data structure with input sample data and new post-stratification data
`metadata`	List containing metadata about the analysis, including family, time variables, and special cases
`link_data`	List containing information about data linking, including geography and ACS year
`plot_data`	List containing data prepared for visualization, including dates and GeoJSON objects
`extra`	List containing COVID test sensitivity and specificity

Value

A new `MRPModel` object initialized with the provided model specification and relevant data.

Methods

`MRPModel` objects have the following associated methods, many of which have their own (linked) documentation pages:

Data access:

Method	**Description**
`$model_spec()`	Return model specification.
`$formula()`	Return model formula.
`$metadata()`	Return model metadata.
`$stan_code()`	Return model Stan code.

Model fitting:

Method	**Description**
`$fit()`	Fit multilevel regression model using CmdStanR.
`$check_fit_exists()`	Check if model has been fitted.
`$check_estimate_exists()`	Check if poststratification has been performed.

Posterior summary & diagnostics:

Method	**Description**
`$summary()`	Return posterior summary table.
`$diagnostics()`	Return sampling diagnostics.

Post-processing:

Method	Description
`$ppc()`	Create input for posterior predictive check.
`$log_lik()`	Create input for leave-one-out cross-validation.
`$poststratify()`	Run poststratification to generate population estimates.

Saving model object:

Method	Description
`$save()`	Save model object to file.

Examples

```
library(shinymrp)

# Initialize workflow
workflow <- mrp_workflow()

# Load example data
sample_data <- example_sample_data()

# Preprocess sample data
workflow$preprocess(
  sample_data,
  is_timevar = TRUE,
  is_aggregated = TRUE,
  special_case = NULL,
  family = "binomial"
)

# Link to ACS data at ZIP code level
workflow$link_acs(
  link_geo = "zip",
  acs_year = 2021
)

# Create and fit multiple models
model <- workflow$create_model(
  intercept_prior = "normal(0, 4)",
  fixed = list(
    sex = "normal(0, 2)"
  ),
  varying = list(
    race = "normal(0, 2)",
    age = "normal(0, 2)",
    time = "normal(0, 2)"
  )
)
```

```
# Run MCMC
model$fit(n_iter = 500, n_chains = 2, seed = 123)

# Estimates summary and diagnostics
posterior_summary <- model$summary()

# Sampling diagnostics
model_diagnostics <- model$diagnostics()
```

`MRPModel-method-check_estimate_exists`
Check if poststratification has been performed

Description

The `$check_estimate_exists()` method of an `MRPModel` object checks whether poststratification has been performed. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
check_estimate_exists()
```

Value

Logical indicating whether poststratification has been performed.

`MRPModel-method-check_fit_exists`
Check if model has been fitted

Description

The `$check_fit_exists()` method of an `MRPModel` object checks whether the model has been fitted. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
check_fit_exists()
```

Value

Logical indicating whether the model has been fitted.

`MRPModel-method-diagnostics` *Return sampling diagnostics*

Description

The `$diagnostics()` method of an `MRPModel` object returns MCMC diagnostics, including convergence statistics and sampling efficiency measures. Check out this official Stan guide for more information on interpreting these metrics. For usage examples, refer to the More examples of R6 classes vignette.

Usage

```
diagnostics(summarize = TRUE)
```

Arguments

`summarize`	Logical indicating whether to return a summarized version of the diagnostics (default is TRUE)

Value

A data.frame object if `summarize` is TRUE, otherwise a list of raw diagnostics.

`MRPModel-method-fit` *Fit multilevel regression model using cmdstanr*

Description

The `$fit()` method of an `MRPModel` object fits the model using Stan's main Markov chain Monte Carlo (MCMC) algorithm. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
fit(n_iter = 2000, n_chains = 4, seed = NULL, ...)
```

Arguments

`n_iter`	Number of MCMC iterations per chain (including warmup iterations). Default is 2000.
`n_chains`	Number of MCMC chains to run. Default is 4.
`seed`	Random seed for reproducibility. Default is `NULL`.
`...`	Additional arguments passed to CmdStanR `$sample()` method.

Value

No return value, called for side effects.

`MRPModel-method-formula`

Return model formula

Description

The `$formula()` method of an `MRPModel` object returns the lme4-style formula constructed from the given model specification. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
formula()
```

Value

A character string of the model formula.

`MRPModel-method-log_lik`

Create inputs for leave-one-out cross-validation

Description

The `$log_lik()` method of an `MRPModel` object runs Stan's standalone generated quantities and extracts log-likelihood values for leave-one-out cross-validation. This method is called by the `$compare_models()` method of an `MRPWorkflow` object and does not need to be called directly by users.

Usage

```
log_lik()
```

Value

A data.frame object containing log-likelihood values.

`MRPModel-method-metadata`

Return model metadata.

Description

The `$metadata()` method of an `MRPModel` object returns the metadata associated with the model, including metadata inherited from a workflow object and model fitting parameters. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
metadata()
```

Value

A list containing the model metadata.

`MRPModel-method-model_spec`

Return model specification

Description

The `$model_spec()` method of an `MRPModel` object returns the model specification list. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
model_spec()
```

Value

A list containing the model specification including intercept, fixed effects, varying effects, and interactions.

`MRPModel-method-poststratify` *Run poststratification to generate population estimates*

Description

The `$poststratify()` method of an `MRPModel` object runs Stan's standalone generated quantities and extracts posterior samples for poststratified estimates. This method is called by the `$poststratify()` method of a `MRPWorkflow` object and does not need to be called directly by users.

Usage

```
poststratify(interval = 0.95)
```

Arguments

`interval`	Confidence interval (a numeric value between 0 and 1) or standard deviation (`"1sd"` or `"2sd"`) for the estimates (default is 0.95).

Value

A data.frame object containing the poststratified estimates and their corresponding uncertainty intervals.

`MRPModel-method-ppc` *Create input for posterior predictive check*

Description

The `$ppc()` method of an `MRPModel` object runs Stan's standalone generated quantities to draw from the posterior predictive distribution. This method is called by the `$pp_check()` method of a `MRPWorkflow` object and does not need to be called directly by users.

Usage

```
ppc()
```

Value

A data.frame object containing samples from the posterior predictive distribution.

`MRPModel-method-save` *Save model object to file*

Description

The `$save()` method of an `MRPModel` object saves a fitted MRPModel object to a file for later use. `qs::qsave()` is used internally, and it is customary to use the `.qs` file extension. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
save(file)
```

Arguments

`file`	File path where the model should be saved.

Value

No return value, called for side effects.

`MRPModel-method-stan_code` *Return model Stan code.*

Description

The `$stan_code()` method of an `MRPModel` object returns the model Stan code. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
stan_code()
```

Value

A character string containing the model Stan code.

`MRPModel-method-summary` *Return posterior summary table*

Description

The `$summary()` method of an `MRPModel` object returns tables containing the summary of posterior samples for the model parameters and diagnostics. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
summary()
```

Value

A list of data.frame objects containing posterior sample summary and diagnostics for model parameters:

- fixed effects (`fixed`)
- standard deviations of varying effects (`varying`)
- standard deviations of residuals (`other`)

`MRPWorkflow` *MRPWorkflow objects*

Description

A `MRPWorkflow` object is an R6 object created by the `mrp_workflow()` function. This class provides methods for all steps in the workflow, from data preparation and visualization to model fitting.

Methods

`MRPWorkflow` objects have the following associated methods with their own (linked) documentation pages:

Data preparation:

Method	**Description**
`$preprocess()`	Preprocess sample data.
`$preprocessed_data()`	Return preprocessed sample data.
`$link_acs()`	Link sample data to ACS data.
`$load_pstrat()`	Load custom poststratification data.

Model fitting & diagnostics:

Method	Description
`$create_model()`	Create a `MRPModel` object.
`$pp_check()`	Perform posterior predictive check.
`$compare_models()`	Compare models using LOO-CV.

Visualization:

Method	Description
`$demo_bars()`	Create demographic comparison bar plots.
`$covar_hist()`	Create geographic covariate distribution histograms.
`$sample_size_map()`	Create sample size map.
`$outcome_plot()`	Create summary plots of raw outcome measure.
`$outcome_map()`	Visualize raw outcome measure by geography.
`$estimate_plot()`	Visualize estimates for demographic groups.
`$estimate_map()`	Visualize estimates for geographic areas.

Examples

```
library(shinymrp)

# Initialize the MRP workflow
workflow <- mrp_workflow()

# Load example data
sample_data <- example_sample_data()

### DATA PREPARATION

# Preprocess sample data
workflow$preprocess(
  sample_data,
  is_timevar = TRUE,
  is_aggregated = TRUE,
  special_case = NULL,
  family = "binomial"
)

# Link data to the ACS
# and obtain poststratification data
workflow$link_acs(
  link_geo = "zip",
  acs_year = 2021
)

### DESCRIPTIVE STATISTICS

# Visualize demographic distribution of data
```

```
sex_bars <- workflow$demo_bars(demo = "sex")

# Visualize geographic distribution of data
ss_map <- workflow$sample_size_map()

# Visualize outcome measure
raw_outcome_plot <- workflow$outcome_plot()

### MODEL BUILDING

# Create new model objects
model <- workflow$create_model(
  intercept_prior = "normal(0, 4)",
  fixed = list(
    sex = "normal(0, 2)",
    race = "normal(0, 2)"
  ),
  varying = list(
    age = "",
    time = ""
  )
)

# Run MCMC
model$fit(n_iter = 500, n_chains = 2, seed = 123)

# Estimates summary and diagnostics
model$summary()

# Sampling diagnostics
model$diagnostics()

# Posterior predictive check
workflow$pp_check(model)

### VISUALIZE RESULTS

# Plots of overall estimates, estimates for demographic groups, and geographic areas
workflow$estimate_plot(model, group = "sex")

# Choropleth map of estimates for geographic areas
workflow$estimate_map(model, geo = "county")
```

`MRPWorkflow-method-compare_models` *Compare models using LOO-CV*

Description

The `$compare_models()` method compares multiple fitted `MRPModel` objects using leave-one-out cross-validation to assess relative model performance. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
compare_models(..., suppress = NULL)
```

Arguments

`...`	Multiple MRPModel objects to compare.
`suppress`	Character string specifying output to suppress during comparison.

Value

A data.frame object containing the comparison results.

`MRPWorkflow-method-covar_hist`

Create geographic covariate distribution histogram

Description

The `covar_hist()` method creates histogram plots showing the distribution of geographic covariates across ZIP codes. Refer to the More on data preparation for their definitions. This method is only available for COVID data. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
covar_hist(covar, file = NULL, ...)
```

Arguments

`covar`	Character string specifying the geographic covariate. Options are `"college"`, `"poverty"`, `"employment"`, `"income"`, `"urbanicity"`, and `"adi"`.
`file`	Optional file path to save the plot.
`...`	Additional arguments passed to `ggsave`, such as `width` and `height`.

Value

A ggplot object showing the covariate distribution histogram.

`MRPWorkflow-method-create_model`

Create a new MRPModel object

Description

The `$create_model()` method creates a new MRPModel object with Stan code generated from the model specification list. CmdStanR objects are used internally to interface with Stan to compile the code and run its MCMC algorithm. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
create_model(
  intercept_prior = NULL,
  fixed = NULL,
  varying = NULL,
  interaction = NULL,
  sens = 1,
  spec = 1
)
```

Arguments

`intercept_prior`	Character string specifying the prior distribution for the overall intercept. Check *Details* for more information about prior specification.
`fixed`	List of the fixed effects in the model and their prior distributions. Check *Details* for more information about prior specification.
`varying`	List of the varying effects in the model and the prior distributions of their standard deviations. Check *Details* for more information about prior specification.
`interaction`	List of the interactions in the model and their prior distributions. Interaction names are created by concatenating the names of the interacting variables with a colon (e.g., "sex:age"). Currently, only two-way interactions are supported. Check *Details* for more information about prior specification.
`sens`	Sensitivity adjustment in the COVID-19 test results. Check *Details* for more information.
`spec`	Specificity adjustment in the COVID-19 test results. Check *Details* for more information.

Details

Prior specification:

The syntax for the prior distributions is similar to that of Stan. The following are currently supported:

- normal(mu, sigma)
- student_t(nu, mu, sigma)
- structured*

The last one is a custom prior syntax for the structured prior distribution developed by Si et al. (2020).

The following default prior distributions are assigned to effects with empty strings (`""`) in the model specification list:

- Overall intercept: normal(0,5)
- Coefficient: normal(0,3)

The model assumes varying effects follow a normal distribution with an unknown standard deviation, which will be assigned with priors.

- Standard deviation (main effect): normal+(0,3)
- Standard deviation (interaction): normal+(0,1)

Testing sensitivity and specificity:

For COVID data, we allow users to specify the PCR testing sensitivity and specificity. Let p_k be the probability that person i in group k tests positive. The analytic incidence p_k is a function of the test sensitivity δ, specificity γ, and the true incidence π_k for individuals in group k:

$$p_k = (1-\gamma)(1-\pi_k) + \delta\pi_k.$$

Value

A new MRPModel object.

`MRPWorkflow-method-demo_bars`

Create demographic comparison bar plots

Description

Creates bar plots for comparing demographic distributions between sample data and poststratification data. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
demo_bars(demo, file = NULL, ...)
```

Arguments

`demo`	Character string specifying the demographic variable to plot.
`file`	Optional file path to save the plot.
`...`	Additional arguments passed to `ggsave`, such as `width` and `height`.

Value

A ggplot object showing demographic comparisons

`MRPWorkflow-method-estimate_map`

Create a choropleth map of MRP estimates

Description

The `$estimate_map()` method creates interactive choropleth maps that show MRP estimates by geographic region. This method cannot be used if either the sample or the poststratification data contains no geographic information. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
estimate_map(
  model,
  geo = NULL,
  time_index = NULL,
  interval = 0.95,
  file = NULL
)
```

Arguments

`model`	Fitted MRPModel object
`geo`	Character string specifying the geographic level for mapping. Options include geography for data linking and those at larger scales. A "linking" geography is required to use this method. It is either specified as `geo` in the `$link_acs()` method or the smallest common geographic scale between the sample data and the custom poststratification data input using `$load_pstrat()`.
`time_index`	Integer specifying the time index for time-varying data.
`interval`	Confidence interval (a numeric value between 0 and 1) or standard deviation (`"1sd"` or `"2sd"`) for the estimates (default is 0.95).
`file`	Optional file path with .html extension to save the interactive map. Expand the hamburger menu in the top right corner of the map to access other export options.

Value

A highcharter map object showing MRP estimates by geography.

MRPWorkflow-method-estimate_plot

Visualize estimates for demographic groups

Description

The `$estimate_plot()` method creates plots showing overall MRP estimates or estimates for different demographic groups. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
estimate_plot(
  model,
  group = NULL,
  interval = 0.95,
  show_caption = TRUE,
  file = NULL,
  ...
)
```

Arguments

`model`	Fitted MRPModel object
`group`	Character string specifying the demographic group. If left as `NULL`, overall estimates are plotted.
`interval`	Confidence interval (a numeric value between 0 and 1) or standard deviation (`"1sd"` or `"2sd"`) for the estimates (default is 0.95).
`show_caption`	Logical indicating whether to show the caption in the plot (default is TRUE).
`file`	Optional file path to save the plot.
`...`	Additional arguments passed to `ggsave`, such as `width` and `height`.

Value

A ggplot object showing MRP estimates.

`MRPWorkflow-method-link_acs`

Link sample data to ACS data

Description

The `$link_acs()` method obtains poststratification data by linking the preprocessed sample data to the American Community Survey based on given geographic granularity and year. See the More on data preparation vignette for more information on data linking. For usage examples, refer to the More examples of R6 classes vignette.

Usage

```
link_acs(link_geo = NULL, acs_year = 2023)
```

Arguments

`link_geo`	Character string specifying the geographic level for linking. Options are `"zip"`, `"county"`, and `"state"`.
`acs_year`	Numeric value specifying the last year of the data collection period for the target ACS dataset.

Value

No return value, called for side effects.

`MRPWorkflow-method-load_pstrat`

Load custom poststratification data

Description

The `$load_pstrat()` method processes and stores input poststratification data. The object is subject to the same data preprocessing steps as the sample data. See the More on data preparation vignette for more information on data processing. For usage examples, refer to the More examples of R6 classes vignette.

Usage

```
load_pstrat(pstrat_data, is_aggregated = TRUE)
```

Arguments

`pstrat_data`	An object of class `data.frame` (or one that can be coerced to that class).
`is_aggregated`	Logical indicating whether the poststratification data is already aggregated.

Value

No return value, called for side effects.

`MRPWorkflow-method-outcome_map` *Visualize raw outcome measure by geography*

Description

The `$outcome_map()` method creates maps showing the average outcome values by geography for cross-sectional data, or the highest/lowest temporal average for time-varying data. The sample and poststratification data must contain geographic information for this method to work. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
outcome_map(summary_type = NULL, file = NULL)
```

Arguments

`summary_type`	Character string, for time-varying data, indicating whether to display the highest (`"max"`) or lowest (`"min"`) temporal average. Leave as `NULL` for cross-sectional data.
`file`	Optional file path with .html extension to save the interactive map. Expand the hamburger menu in the top right corner of the map to access other export options.

Value

A highcharter map object showing average outcome measure by geography.

`MRPWorkflow-method-outcome_plot` *Create summary plots of the outcome measure*

Description

The `$outcome_plot()` method creates plots of the average outcome values. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
outcome_plot(file = NULL, ...)
```

Arguments

`file`	Optional file path to save the plot.
`...`	Additional arguments passed to ggsave, such as `width` and `height`.

Value

A ggplot object showing the outcome measure distribution.

`MRPWorkflow-method-pp_check`

Perform posterior predictive check

Description

The `$pp_check()` method creates posterior predictive check plots to assess model fit by comparing observed data to replicated data from the posterior predictive distribution. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
pp_check(model, file = NULL, ...)
```

Arguments

`model`	Fitted MRPModel object.
`file`	Optional file path to save the plot.
`...`	Additional arguments passed to ggsave, such as `width` and `height`.

Value

A ggplot object showing the posterior predictive check result.

`MRPWorkflow-method-preprocess`

Preprocess sample data

Description

The `$preprocess()` method runs the preprocessing pipeline that includes data standardization, filtering, imputation, and aggregation. See the More on data preparation vignette for more information about data processing. For usage examples, refer to the More examples of R6 classes vignette.

Usage

```
preprocess(
  data,
  is_timevar = FALSE,
  is_aggregated = FALSE,
  special_case = NULL,
  family = NULL,
  time_freq = NULL,
  freq_threshold = 0
)
```

Arguments

`data`	An object of class `data.frame` (or one that can be coerced to that class) that satisfies the requirements specified in the More on data preparation vignette.
`is_timevar`	Logical indicating whether the data contains time-varying components.
`is_aggregated`	Logical indicating whether the data is already aggregated.
`special_case`	Character string specifying special case handling. Options are `NULL` (the default), `"covid"`, and `"poll"`.
`family`	Character string specifying the distribution family for the outcome variable. Options are `"binomial"` for binary outcome measures and `"normal"` for continuous outcome measures.
`time_freq`	Character string specifying the time indexing frequency or time length for grouping dates (YYYY-MM-DD) in the data. Options are `NULL` (the default), `"week"`, `"month"`, and `"year"`. This parameter must be `NULL` for cross-sectional data or time-varying data that already has time indices.
`freq_threshold`	Numeric value specifying the minimum frequency threshold for including observations. Values with lower frequency will cause the entire row to be removed. The default value is 0 (no filtering).

Value

No return value, called for side effects.

`MRPWorkflow-method-preprocessed_data`

Return preprocessed sample data

Description

The `$preprocessed_data()` method returns the preprocessed sample data. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
preprocessed_data()
```

Value

A data.frame object containing the preprocessed sample data.

`MRPWorkflow-method-sample_size_map` *Create sample size map*

Description

The `$sample_size_map()` method creates interactive choropleth maps showing data distribution with respect to geography. This method cannot be used if either the sample or the poststratification data contains no geographic information. Check out the More examples of R6 classes vignette for usage examples.

Usage

```
sample_size_map(file = NULL)
```

Arguments

`file`	Optional file path with .html extension to save the interactive map. Expand the hamburger menu in the top right corner of the map to access other export options.

Value

A highcharter map object showing sample size distribution.

`mrp_workflow` *Create a new MRPWorkflow object*

Description

Create a new `MRPWorkflow` object that implements the Bayesian data analysis workflow common in applications of Multilevel Regression and Post-stratification (MRP).

Usage

```
mrp_workflow()
```

Value

A `MRPWorkflow` object.

`run_app` *Run the Shiny Application*

Description

Run the Shiny Application

Usage

```
run_app(
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  launch.browser = TRUE,
  ...
)
```

Arguments

`onStart`	A function that will be called before the app is actually run. This is only needed for `shinyAppObj`, since in the `shinyAppDir` case, a `global.R` file can be used for this purpose.
`options`	Named options that should be passed to the `runApp` call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify `width` and `height` parameters which provide a hint to the embedding environment about the ideal height/width for the app.
`enableBookmarking`	Can be one of `"url"`, `"server"`, or `"disable"`. The default value, `NULL`, will respect the setting from any previous calls to `enableBookmarking()`. See `enableBookmarking()` for more information on bookmarking your app.
`uiPattern`	A regular expression that will be applied to each `GET` request to determine whether the `ui` should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.
`launch.browser`	Logical; if TRUE (default) open in an external browser even when running inside RStudio. If FALSE, use RStudio Viewer (when available).
`...`	arguments passed into golem options via `with_golem_options()`. See `?golem::get_golem_options` for details.

Index