

Package: **voiageR** (via r-universe)

October 12, 2026

Type Package

Title Value of Information Analysis in R

Version 2.2.0

Config/voiage/Release-Version 2.2.0

Description An R interface to the Rust-backed voiage core for Value of Information analysis, with a direct Rust C ABI EVPI path and helper functions for Python environment management where advanced methods still use reticulate. Provides functions for calculating Expected Value of Perfect Information (EVPI), Expected Value of Partial Perfect Information (EVPPI), and Expected Value of Sample Information (EVSI).

URL <https://github.com/edithatogo/voiage>

BugReports <https://github.com/edithatogo/voiage/issues>

License Apache License (>= 2.0)

Encoding UTF-8

NeedsCompilation yes

Roxygen list(markdown = TRUE, roclets = c("`rd", "`namespace",
`srr::srr_stats_roclet"))

Config/testthat/edition 3

SystemRequirements Cargo and Rust (>= 1.85) for source installation;
Python (>= 3.12, optional, for EVPPI and EVSI through
reticulate)

Suggests reticulate (>= 1.46.0), knitr (>= 1.51), rmarkdown (>= 2.31),
testthat (>= 3.3.2)

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

Config/pak/sysreqs python3 libclang-dev

Repository <https://edithatogo.r-universe.dev>

Date/Publication 2026-09-12 06:33:49 UTC

RemoteUrl <https://github.com/edithatogo/voiage>

```
RemoteRef HEAD
RemoteSha ca3f05cc8adcc74b99fe236152327d0cd3679553
RemoteSubdir r-package/voiageR
```

Contents

enbs	2
evpi	3
evppi	4
evsi	5
init_voiage	7
is_voiage_available	7
set_voiage_env	8
Index	9

enbs	<i>Calculate Expected Net Benefit of Sampling (ENBS)</i>
------	--

Description

This function calculates the Expected Net Benefit of Sampling as EVSI - research_cost, optionally scaling EVSI to a population before subtracting the research cost.

Usage

```
enbs(  
  evsi_result,  
  research_cost,  
  population = NULL,  
  time_horizon = NULL,  
  discount_rate = NULL  
)
```

Arguments

- evsi_result A finite numeric value representing the expected value of sample information.
- research_cost A non-negative numeric value representing the total cost of the proposed re-
search study.
- population Optional population size for scaling the EVSI result.
- time_horizon Optional time horizon for scaling the EVSI result.
- discount_rate Optional discount rate for scaling the EVSI result.

Value

The calculated ENBS value.

Examples

```
# Calculate ENBS for a study with EVSI = 12.5 and cost = 3.0
enbs_value <- enbs(12.5, 3.0)
print(enbs_value)
```

evpi

Calculate Expected Value of Perfect Information (EVPI)

Description

This function calculates the Expected Value of Perfect Information using the `voiage` library.

Usage

```
evpi(
  net_benefits,
  population = NULL,
  time_horizon = NULL,
  discount_rate = NULL
)
```

Arguments

<code>net_benefits</code>	A matrix or data frame with rows representing PSA samples and columns representing strategies.
<code>population</code>	Optional population size for scaling the result.
<code>time_horizon</code>	Optional time horizon for scaling the result.
<code>discount_rate</code>	Optional discount rate for scaling the result.

Value

The calculated EVPI value.

Examples

```
# Create sample net benefit data
net_benefits <- matrix(rnorm(2000), nrow = 1000, ncol = 2)

# Calculate EVPI
evpi_value <- evpi(net_benefits)
print(evpi_value)
```

evppi

Calculate Expected Value of Partial Perfect Information (EVPPI)

Description

This function calculates the Expected Value of Partial Perfect Information using the `voiage` library.

Usage

```
evppi(
  net_benefits,
  parameter_samples,
  parameters_of_interest = NULL,
  population = NULL,
  time_horizon = NULL,
  discount_rate = NULL
)
```

Arguments

`net_benefits` A matrix or data frame with rows representing PSA samples and columns representing strategies.

`parameter_samples` A named list or data frame with parameter samples.

`parameters_of_interest` Optional character vector of parameter names to retain.

`population` Optional population size for scaling the result.

`time_horizon` Optional time horizon for scaling the result.

`discount_rate` Optional discount rate for scaling the result.

Value

The calculated EVPPI value.

Examples

```
## Not run:
# Create sample net benefit data
net_benefits <- matrix(rnorm(2000), nrow = 1000, ncol = 2)

# Create parameter samples
param_samples <- list(
  param1 = rnorm(1000),
  param2 = rnorm(1000)
)

# Calculate EVPPI
```

```

evppi_value <- evppi(net_benefits, param_samples)
print(evppi_value)

## End(Not run)

```

evsi

Calculate Expected Value of Sample Information (EVSI)

Description

This function calculates Expected Value of Sample Information through the Python voyage bridge.

Usage

```

evsi(
  model_func,
  prior_samples,
  trial_design,
  population = NULL,
  time_horizon = NULL,
  discount_rate = NULL,
  method = c("two_loop", "regression", "efficient", "moment_based"),
  n_outer_loops = 100,
  n_inner_loops = 1000,
  metamodel = "linear",
  seed = NULL
)

```

Arguments

<code>model_func</code>	A function that takes parameter samples and returns net benefits.
<code>prior_samples</code>	A named list, data frame, or matrix with prior parameter samples.
<code>trial_design</code>	A list containing an arms list. Each arm must provide a unique name and positive integer <code>sample_size</code> .
<code>population</code>	Optional population size for scaling the result.
<code>time_horizon</code>	Optional time horizon for scaling the result.
<code>discount_rate</code>	Optional discount rate for scaling the result.
<code>method</code>	EVSI estimator. <code>two_loop</code> is the corrected built-in joint-normal study model; the other values are compatibility estimators.
<code>n_outer_loops</code>	Number of simulated trial data sets for <code>two_loop</code> .
<code>n_inner_loops</code>	Number of joint posterior draws per simulated trial.
<code>metamodel</code>	Metamodel used by compatible approximation methods.
<code>seed</code>	Optional non-negative integer seed for reproducible <code>two_loop</code> simulation.

Details

The built-in `two_loop` method uses a fitted joint multivariate-normal prior and a known-variance arm-mean likelihood. `prior_samples` must contain one parameter named `mean_<normalised arm>` for each trial arm, where normalisation lowercases the arm name and replaces spaces with under-scores. It must also contain a finite, strictly positive `sd_outcome` whose value is fixed across all prior draws.

Python-only custom `trial_simulator` and `posterior_sampler` callbacks are not converted or executed by the R facade, which therefore does not claim parity for custom two-loop models.

Value

The calculated EVSI value.

Examples

```
## Not run:
model_func <- function(params) {
  cbind(
    Treatment = 50000 * params$mean_treatment - 3000,
    Control = 50000 * params$mean_control
  )
}

set.seed(17)
draws <- 1000L
prior_samples <- list(
  mean_treatment = rnorm(draws, mean = 0.06, sd = 0.03),
  mean_control = rnorm(draws, mean = 0.00, sd = 0.02),
  sd_outcome = rep(1.0, draws)
)

trial_design <- list(
  arms = list(
    list(name = "Treatment", sample_size = 100L),
    list(name = "Control", sample_size = 100L)
  )
)

evsi_value <- evsi(
  model_func,
  prior_samples,
  trial_design,
  method = "two_loop",
  n_outer_loops = 100L,
  n_inner_loops = 1000L,
  seed = 20260724L
)
print(evsi_value)

## End(Not run)
```

init_voiage	<i>Initialize the voiage Python module</i>
-------------	--

Description

This function initializes the voiage Python module using reticulate.

Usage

```
init_voiage()
```

Value

None (invisible)

Examples

```
## Not run:  
init_voiage()  
  
## End(Not run)
```

is_voiage_available	<i>Check if voiage Python package is available</i>
---------------------	--

Description

This function checks if the voiage Python package is available and can be imported.

Usage

```
is_voiage_available()
```

Value

Logical indicating whether the voiage package is available.

Examples

```
## Not run:  
is_available <- is_voiage_available()  
print(is_available)  
  
## End(Not run)
```

set_voiage_env	<i>Set Python environment for voiage</i>
----------------	--

Description

This function sets the Python environment to use for the voiage package.

Usage

```
set_voiage_env(env, type = c("virtualenv", "conda"))
```

Arguments

env	The Python environment to use (virtualenv or conda environment name).
type	The type of environment ("virtualenv" or "conda").

Value

None (invisible)

Examples

```
## Not run:  
# Use a virtual environment  
set_voiage_env("myenv", type = "virtualenv")  
  
# Use a conda environment  
set_voiage_env("myenv", type = "conda")  
  
## End(Not run)
```

Index

enbs, [2](#)

evpi, [3](#)

evppi, [4](#)

evsi, [5](#)

init_voiage, [7](#)

is_voiage_available, [7](#)

set_voiage_env, [8](#)