Package 'cosmic'

May 8, 2026

Title Conditional Ordinal Stereotype Model for Incident-Level Comparison

Version 0.5

Description Implements the Conditional Ordinal Stereotype Model for Incident-Level Comparison (COSMIC), a method for analyzing ordinal outcomes observed across multiple actors within shared events. The model uses a conditional likelihood to remove event-level confounding and estimate actor-specific propensities relative to their peers. Efficient computation is achieved via a dynamic programming algorithm for the Poisson-multinomial normalization term, enabling scalable estimation with Markov chain Monte Carlo. The package provides tools for data preparation, model fitting using Stan, and extraction of posterior summaries for comparative inference. Estimation of police officer propensity to escalate force is the primary motivation for the model. For more details see Ridgeway (2026) ``A Conditional Ordinal Stereotype Model to Estimate Police Officers' Propensity to Escalate Force'' <doi:10.1080/01621459.2025.2597050>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Suggests arrangements, cmdstanr (>= 0.9.0), future, future.apply, ggplot2, kableExtra, knitr, MASS, progressr, rmarkdown, testthat (>= 3.0.0), xtable

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

Config/testthat/edition 3

VignetteBuilder knitr

Imports dplyr, posterior, stats

NeedsCompilation no

Author Greg Ridgeway [aut, cre]

Maintainer Greg Ridgeway <gridge@upenn.edu>

Repository CRAN

Date/Publication 2026-05-04 19:00:08 UTC

Contents

`cosmic` *Fit the Conditional Ordinal Stereotype Model (COSMIC)*

Description

Fits the Conditional Ordinal Stereotype Model for Identification and Comparison (COSMIC) to ordinal outcomes observed across multiple actors within shared events (e.g., officers within incidents). The model uses a conditional likelihood to remove event-level confounding and estimate actor-specific latent propensities relative to their peers.

Usage

```
cosmic(
  data,
  incidentID,
  officerID,
  y,
  priorSD_lambda = 2,
  priorSD_sDiff = 1,
  iter = 2000,
  chains = 4,
  cores = 1,
  threads = 8
)
```

Arguments

`data`	A data frame containing one row per actor-event observation
`incidentID`	A column (unquoted) identifying the event or incident
`officerID`	A column (unquoted) identifying the actor (e.g., officer)
`y`	A column (unquoted) giving the ordinal outcome. Values must be consecutive integers starting at 1 (e.g., 1, 2, 3, ...)
`priorSD_lambda`	Prior standard deviation for the actor-specific parameters λ. Default is 2
`priorSD_sDiff`	Prior standard deviation for the differences between adjacent ordinal scale parameters

iter	Number of MCMC iterations per chain
chains	Number of Markov chains
cores	Number of chains to run in parallel. When using within-chain parallelization (e.g., via reduce_sum), this should be set so that cores * threads_per_chain does not exceed the number of available CPU cores
threads	Number of threads per chain used for within-chain parallelization

Details

The likelihood is evaluated using a dynamic programming algorithm for the Poisson-multinomial normalization term, enabling efficient computation for incidents involving multiple actors. Posterior inference is performed via Markov chain Monte Carlo using **cmdstanr**.

Because **cmdstanr** is not distributed on CRAN, users may need to install it from the Stan R-universe repository before fitting models:

```
install.packages("cmdstanr",
                 repos = c("https://stan-dev.r-universe.dev",
                           getOption("repos")))
cmdstanr::install_cmdstan()
```

The COSMIC model is an ordinal regression model that estimates actor-specific latent propensities while conditioning on the set of outcomes observed within each event. This conditioning removes all event-level factors that are shared across actors, allowing for comparisons that are invariant to environmental differences.

The normalization term in the conditional likelihood involves a sum over permutations consistent with the observed counts of outcome categories. This is computed using a dynamic programming algorithm to avoid explicit enumeration.

Parallel computation is supported both across chains and within chains. When specifying cores and threads, users should ensure that total CPU usage remains within hardware limits.

Value

An object of class "cosmic_fit" containing:

fit	The fitted CmdStanMCMC object.
model	The compiled CmdStanModel used for sampling.
data	The processed data passed to Stan, including an officer_lookup table that maps sequential idOff values back to the original officerID values supplied by the user.

References

Ridgeway, G. (2026). A Conditional Ordinal Stereotype Model to Estimate Police Officers' Propensity to Escalate Force. *Journal of the American Statistical Association*, 1–12. https://www.tandfonline.com/doi/full/10.1080/01621459.2025.2597050

See Also

posterior, summary.cosmic_fit

Examples

```
d <- data.frame(
  id = c(1,1,2,2),
  idOff = c(1,2,1,2),
  y = c(1,2,1,3)
)

fit <- cosmic(d, id, idOff, y,
              iter = 300,
              chains = 1,
              cores = 1,
              threads = 1)

print(fit)
```

---

diagnose *Diagnose a COSMIC fit*

Description

Reports convergence and Hamiltonian Monte Carlo diagnostics for a fitted `"cosmic_fit"` object. The report flags parameters with large $\hat{R}$, small effective sample size, and sampler pathologies such as divergences, saturated tree depth, or low E-BFMI.

Usage

```
diagnose(object, ...)

## S3 method for class 'cosmic_fit'
diagnose(
  object,
  rhat_threshold = 1.01,
  ess_threshold_per_chain = 100,
  ebfmi_threshold = 0.2,
  ...
)
```

Arguments

`object`	A fitted object of class `"cosmic_fit"`.
`...`	Additional arguments passed to the underlying summary method for the Stan fit.
`rhat_threshold`	Threshold above which $\hat{R}$ is flagged. Default is `1.01`.

`ess_threshold_per_chain`
: Minimum effective sample size per chain. Default is `100`; the total threshold used is this value multiplied by the number of chains.

`ebfmi_threshold`
: Threshold below which E-BFMI is flagged. Default is `0.2`.

Value

`diagnose.cosmic_fit()` prints a compact diagnostic report and returns a `"cosmic_diagnostics"` object invisibly.

Examples

```
d <- data.frame(
  id = c(1,1,2,2),
  idOff = c(1,2,1,2),
  y = c(1,2,1,3)
)

fit <- cosmic(d, id, idOff, y, iter = 300, chains = 1, cores = 1, threads = 1)
diagnose(fit)
```

`officer_summary` *Summarize officers relative to their peer groups*

Description

Computes posterior summary measures for each officer in a fitted COSMIC model. The returned data frame includes the size of each officer's peer group, both the original and sequential officer IDs, tail-rank probabilities within that group, posterior contrasts against the mean of peers, and counts by force category.

Usage

```
officer_summary(object, pct_threshold = 0.25, pct_tail = 0.05)
```

Arguments

`object`
: A fitted object of class `"cosmic_fit"`.

`pct_threshold`
: Threshold used to define peer groups from posterior variances. Officer `j` is treated as a peer of officer `i` when the posterior variance of `lambda_i - lambda_j` is less than `pct_threshold * 2 * priorSD_lambda^2`. Default is `0.25`.

`pct_tail`
: Tail fraction used when computing top-tail and bottom-tail rank probabilities. Default is `0.05`, tracking the probability of being in the top or bottom 5 percent.

Details

Officers are treated as peers when the posterior variance of $\lambda_i - \lambda_j$ is smaller than `pct_threshold * 2 * priorSD_lambda^2`.

When the **future.apply** package is installed, computations are distributed using the active **future** plan. If **progressr** is also installed, progress updates are emitted while officers are processed.

Parallel workers are not configured by `officer_summary()` itself. Instead, the function respects the **future** plan that is already active. For example, users can request four background R sessions with `future::plan(future::multisession, workers = 4)` before calling `officer_summary()`. If no multi-worker plan is active, computation falls back to sequential evaluation. To see progress updates in an interactive session, call `progressr::handlers("cli")` before running `officer_summary()`.

Value

A data frame with one row per officer. When the fitted model includes an `officer_lookup` table, the summary includes both `idOffOrig` and `idOff`. The returned object has class `c("cosmic_officer_summary", "data.frame")`.

Examples

```
d <- data.frame(
  id = c(1,1,2,2),
  idOff = c(1,2,1,2),
  y = c(1,2,1,3)
)
fit <- cosmic(d, id, idOff, y, iter = 300, chains = 1, cores = 1, threads = 1)

# use plan() and handlers() to run officer_summary() in parallel
# future::plan(future::multisession, workers = 4)
# progressr::handlers("cli")
off_summary <- officer_summary(fit)
outlier_report(off_summary)
```

---

outlier_report *Create an outlier-focused report table*

Description

Filters and reorders officer summaries to show only officers whose top-tail or bottom-tail posterior rank probability exceeds a chosen cutoff. The returned data frame is designed to be passed directly to `knitr::kable()`, `kableExtra::kbl()`, or `xtable::xtable()`.

Usage

```
outlier_report(x, prob_outlier = 0.8)
```

Arguments

x	A "cosmic_officer_summary" object returned by officer_summary.
prob_outlier	Probability cutoff used to flag an outlier. Default is 0.8. Set to 0 to return all officers.

Value

A filtered data frame with class c("cosmic_outlier_report", "data.frame"). When available, the report includes both idOffOrig and idOff.

Examples

```
d <- data.frame(
  id = c(1,1,2,2),
  idOff = c(1,2,1,2),
  y = c(1,2,1,3)
)
fit <- cosmic(d, id, idOff, y, iter = 300, chains = 1, cores = 1, threads = 1)
off_summary <- officer_summary(fit)

outliers <- outlier_report(off_summary)
outliers

# For a nicer looking report use kable()
# knitr::kable(outliers)
```

---

posterior *Extract posterior draws from a COSMIC model*

Description

Extracts posterior samples from a fitted COSMIC model. This is a convenience wrapper around CmdStan draws extraction that avoids direct dependence on **cmdstanr** in user code.

Usage

```
posterior(object, ...)

## S3 method for class 'cosmic_fit'
posterior(object, pars = NULL, tidy = FALSE, ...)
```

Arguments

object	A fitted object of class "cosmic_fit".
...	Additional arguments passed to objectfitdraws().
pars	Optional character vector of parameter names to extract (e.g., "lambda", "sDelta"). Defaults to all parameters.
tidy	Logical; if FALSE (default), returns a list of arrays as a named list of posterior draw arrays. If TRUE, returns a data frame with one row per draw.

Value

If tidy = FALSE, a named list of posterior draws.

If tidy = TRUE, a data frame with one row per posterior draw and columns corresponding to parameters (expanded with indices where needed).

Examples

```
d <- data.frame(
  id = c(1,1,2,2),
  idOff = c(1,2,1,2),
  y = c(1,2,1,3)
)

fit <- cosmic(d, id, idOff, y, iter = 300, chains = 1, cores = 1, threads = 1)

# raw posterior draws
draws <- posterior(fit)

# only lambda
lambda_draws <- posterior(fit, pars = "lambda")

# tidy format
df <- posterior(fit, tidy = TRUE)
```

---

summary.cosmic_fit *Print and summarize a COSMIC fit*

Description

Methods for inspecting fitted objects of class "cosmic_fit". print() shows a compact posterior summary for the main model parameters, while summary() returns the full summary matrix from the underlying CmdStan fit object.

Usage

```
## S3 method for class 'cosmic_fit'
print(x, ...)

## S3 method for class 'cosmic_fit'
summary(object, ...)
```

Arguments

x	A fitted object of class "cosmic_fit".
...	Additional arguments passed to the underlying method.
object	A fitted object of class "cosmic_fit".

Value

print.cosmic_fit() returns x invisibly.

summary.cosmic_fit() returns the summary matrix produced by the underlying fit summary method.

Index