

Package ‘ErrorTracer’

May 4, 2026

Type Package

Title Bayesian Error Propagation and Forecast Uncertainty
Decomposition

Version 1.0.2

Date 2026-04-29

Description Provides a full pipeline from regularized or standard regression models (elastic net, linear models, generalized linear models, random forests) to informed Bayesian priors, three-way forecast uncertainty decomposition (parameter / environmental / residual), and forecast shelf life analysis (the quantification of when a forecast becomes uninformative). Designed for ecological and genomic forecasting with climate or environmental covariates. Methods build on Bürkner (2017) <doi:10.18637/jss.v080.i01> for Bayesian regression via 'Stan', Friedman, Hastie, and Tibshirani (2010) <doi:10.18637/jss.v033.i01> for elastic net regularization, Wright and Ziegler (2017) <doi:10.18637/jss.v077.i01> for random forests, and Vehtari, Gelman, and Gabry (2017) <doi:10.1007/s11222-016-9696-4> for leave-one-out cross-validation.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Depends R (>= 4.1.0)

Imports brms (>= 2.20.0), ggplot2 (>= 3.4.0), rlang (>= 1.1.0), stats,
utils

Suggests loo (>= 2.6.0), bayesplot (>= 1.10.0), scales (>= 1.3.0),
tidyr (>= 1.3.0), glmnet (>= 4.1.0), ranger (>= 0.15.0), dplyr
(>= 1.1.0), testthat (>= 3.0.0), knitr, rmarkdown, covr

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Luis Javier Madrigal-Roca [aut, cre],
John Kelly [aut]

Maintainer Luis Javier Madrigal-Roca <madrigalrocalj@yahoo.com>

Repository CRAN

Date/Publication 2026-05-04 11:50:07 UTC

Contents

decompose_uncertainty	2
et_calibrate	3
et_diagnose	5
et_fit	5
et_plot_calibration	7
et_plot_coefficients	7
et_plot_decomposition	8
et_plot_forecast	8
et_plot_prior_posterior	9
et_plot_sensitivity	10
et_plot_shelf_life	10
et_predict	11
et_sensitivity_profile	14
et_sim	16
et_theme	18
extract_priors	19
shelf_life	21
standardize	24
unstandardize	24
Index	25

decompose_uncertainty *Extract or recompute uncertainty decomposition*

Description

Returns a `data.frame` with the three-way uncertainty decomposition stored inside an `et_prediction` object:

param_var Variance of the posterior linear predictor — captures uncertainty in fitted regression coefficients.

env_var Additional variance arising from measurement or prediction uncertainty in the predictor values (estimated via perturbation in `et_predict`). Zero when `env_noise = NULL`.

residual_var Posterior mean of σ^2 — biological process noise, unmeasured drivers, and drift. Using the mean (not median) ensures `param_var + residual_var ≈ total_var` by the law of total variance.

total_var Variance of the full posterior predictive draws (parameter + residual; note that `env_var` is an additive component measured separately from the perturbation step).

Usage

```
decompose_uncertainty(predictions, ...)
```

Arguments

`predictions` An `et_prediction` object from `et_predict`, or an `et_prediction_list` (grouped).
`...` Unused.

Details

All variance components are guaranteed non-negative.

Value

A `data.frame` with columns `obs_id`, `param_var`, `env_var`, `residual_var`, `total_var`. For grouped predictions, a `group` column is prepended.

Examples

```
set.seed(1)
df <- data.frame(y = rnorm(20), x1 = rnorm(20))
fit <- et_fit(y ~ x1, data = df,
             chains = 1, iter = 500, warmup = 250,
             cores = 1, refresh = 0)
new_df <- data.frame(x1 = rnorm(5))
pred <- et_predict(fit, newdata = new_df,
                  env_noise = list(x1 = 0.2),
                  n_draws = 200, n_perturb = 50)
decomp <- decompose_uncertainty(pred)
head(decomp)
```

 et_calibrate

Assess calibration of posterior predictive intervals

Description

Computes observed coverage probability at multiple nominal CI levels. A well-calibrated Bayesian model should produce 90% CIs that contain the true value 90% of the time, etc.

Usage

```
et_calibrate(predictions, observed, response_col = NULL, ci_levels = NULL, ...)
```

Arguments

<code>predictions</code>	An <code>et_prediction</code> or <code>et_prediction_list</code> .
<code>observed</code>	A <code>data.frame</code> with true response values. Must have the same number of rows as <code>predictions\$newdata</code> (rows are matched positionally) and a column with the true response values.
<code>response_col</code>	Character. Name of the response column in <code>observed</code> . Defaults to the left-hand side of the model formula if it can be inferred, otherwise must be specified.
<code>ci_levels</code>	Numeric vector. CI levels to assess. Defaults to all levels present in the <code>et_prediction</code> object.
<code>...</code>	Unused.

Value

A `data.frame` with columns:

ci_level Nominal CI level.

nominal Same as `ci_level`.

observed_coverage Fraction of true values falling inside the CI.

n_obs Number of observations used.

calibration_error Signed difference: `observed - nominal`. Positive = over-coverage (CIs too wide / conservative). Negative = under-coverage (CIs too narrow / overconfident).

sharpness Mean CI width across observations. Sharpness and calibration are complementary: a model can be calibrated but useless if sharpness is poor (very wide CIs).

For grouped predictions, a group column is prepended.

Examples

```
set.seed(1)
df <- data.frame(y = rnorm(20), x1 = rnorm(20))
fit <- et_fit(y ~ x1, data = df,
             chains = 1, iter = 500, warmup = 250,
             cores = 1, refresh = 0)
valid_df <- data.frame(y = rnorm(5), x1 = rnorm(5))
pred <- et_predict(fit, newdata = valid_df,
                 n_draws = 200, n_perturb = 50)
cal <- et_calibrate(pred, observed = valid_df, response_col = "y")
print(cal)
```

et_diagnose	<i>Diagnose a fitted ErrorTracer model</i>
-------------	--

Description

Computes Rhat, effective sample size ratios, divergent transitions, and leave-one-out cross-validation (LOO-CV) for a fitted `et_model`.

Usage

```
et_diagnose(model, loo = TRUE, ...)
```

Arguments

<code>model</code>	An <code>et_model</code> or <code>et_model_list</code> .
<code>loo</code>	Logical. Whether to run LOO-CV (can be slow; default TRUE).
<code>...</code>	Unused.

Value

A list with elements:

convergence List: `rhat_max`, `rhat_all_ok`, `neff_min`, `neff_all_ok`, `n_divergences`.

loo List or NULL: `elpd_loo`, `p_loo`, `looic`, `n_bad_pareto_k`, `loo_object`.

summary Printed summary from `brms::summary()`.

For `et_model_list`, a named list of per-group diagnostic lists plus an aggregated summary `data.frame`.

et_fit	<i>Fit a Bayesian regression model with informed priors</i>
--------	---

Description

Wraps `brms::brm()` and attaches the prior specification, training data reference, and configuration for downstream uncertainty decomposition. Pass `priors` from `extract_priors` to use regularized-model coefficients as prior means; omit it for default (weakly informative) priors.

Usage

```

et_fit(
  formula,
  data,
  priors = NULL,
  chains = 4L,
  iter = 2000L,
  warmup = floor(iter/2),
  cores = min(chains, parallel::detectCores()),
  seed = 42L,
  adapt_delta = 0.95,
  max_treedepth = 12L,
  grouping = NULL,
  eiv = NULL,
  silent = 2L,
  ...
)

```

Arguments

formula	An R formula, e.g. $\text{response} \sim .$ or $y \sim x_1 + x_2$.
data	A data frame with all predictors and the response.
priors	An <code>et_prior_spec</code> object from extract_priors , or a <code>brmsprior</code> object, or <code>NULL</code> for <code>brms</code> defaults.
chains	Integer. Number of MCMC chains (default 4).
iter	Integer. Total iterations per chain, including warmup (default 2000).
warmup	Integer. Warmup iterations per chain (default $\text{floor}(\text{iter} / 2)$).
cores	Integer. Parallel cores (default $\text{min}(\text{chains}, \text{parallel}::\text{detectCores}())$).
seed	Integer. Random seed for reproducibility (default 42).
adapt_delta	Numeric. Target acceptance probability for HMC (default 0.95).
max_treedepth	Integer. Maximum tree depth (default 12).
grouping	Character. Name of a column in <code>data</code> to use for grouping. If non- <code>NULL</code> , one model is fitted per unique group value and an <code>et_model_list</code> is returned.
eiv	Optional errors-in-variables specification. A named list / vector mapping predictor names to either a scalar SD or a vector of per-row SDs (length $\text{nrow}(\text{data})$). For each entry, the formula term for that predictor is rewritten as <code>brms:me(pred, se_pred)</code> (an auxiliary <code>se_<pred></code> column is appended to <code>data</code>), so the posterior reflects measurement error in the predictor as well as coefficient uncertainty. The beta posteriors widen accordingly, which partially absorbs what <code>ErrorTracer</code> 's downstream <code>env_var</code> component would otherwise report. When <code>eiv</code> is supplied together with an <code>et_prior_spec</code> from extract_priors , the informed priors are <i>dropped</i> because they target <code>class = "b"</code> terms and <code>me()</code> terms live under <code>class = "bsp"</code> ; <code>brms</code> defaults are used instead (and a warning is logged).
silent	Integer passed to <code>brms::brm()</code> (default 2, no Stan output).
...	Additional arguments passed to <code>brms::brm()</code> .

Value

An `et_model` object (or an `et_model_list` if grouping is specified).

Examples

```
set.seed(1)
df <- data.frame(y = rnorm(20), x1 = rnorm(20), x2 = rnorm(20))
ps <- extract_priors(lm(y ~ x1 + x2, data = df))
fit <- et_fit(y ~ x1 + x2, data = df, priors = ps,
             chains = 1, iter = 500, warmup = 250,
             cores = 1, refresh = 0)
print(fit)
```

`et_plot_calibration` *Plot calibration: observed vs nominal coverage*

Description

A well-calibrated model produces points along the 1:1 diagonal. Points above the diagonal indicate over-coverage (conservative); below indicates under-coverage (anti-conservative).

Usage

```
et_plot_calibration(cal)
```

Arguments

`cal` A `data.frame` from [et_calibrate](#).

Value

A `ggplot2` object.

`et_plot_coefficients` *Forest plot of regression coefficients*

Description

Compares Bayesian posterior estimates (95% CI) with the regularized coefficient values used as prior means (shown as crosses).

Usage

```
et_plot_coefficients(model)
```

Arguments

model An `et_model` or `et_model_list`.

Value

A `ggplot2` object.

`et_plot_decomposition` *Plot three-way uncertainty decomposition*

Description

Produces a stacked bar chart showing the relative contributions of parameter uncertainty, environmental uncertainty, and residual variance for each observation.

Usage

```
et_plot_decomposition(decomp, proportional = TRUE, group_col = NULL)
```

Arguments

decomp A data.frame from `decompose_uncertainty` or directly the `$decomposition` slot of an `et_prediction`.

proportional Logical. If TRUE (default), bars are scaled to sum to 1 (proportional contribution). If FALSE, raw variances are shown.

group_col Character. Optional name of a grouping column in `decomp` (present for grouped predictions).

Value

A `ggplot2` object.

`et_plot_forecast` *Plot posterior predictive fan chart*

Description

Overlays nested credible interval ribbons on the median forecast, with optional observed values for calibration assessment.

Usage

```
et_plot_forecast(
  predictions,
  observed = NULL,
  response_col = NULL,
  time_col = NULL
)
```

Arguments

predictions	An et_prediction object.
observed	Optional data.frame with true response values. If provided, points are overlaid.
response_col	Character. Name of the response column in observed.
time_col	Character. Column in predictions\$newdata used as the x-axis. Defaults to observation index.

Value

A ggplot2 object.

et_plot_prior_posterior

Plot prior vs posterior distributions for model coefficients

Description

Overlays prior and posterior density for each predictor coefficient, visualising how much the data update the priors.

Usage

```
et_plot_prior_posterior(model, max_preds = 8L, n_prior_draws = 4000L)
```

Arguments

model	An et_model object.
max_preds	Integer. Maximum number of predictors to show (default 8). Predictors are shown in the order they appear in the prior specification.
n_prior_draws	Integer. Number of random draws for the prior density (default 4000).

Value

A ggplot2 object.

et_plot_sensitivity *Plot a sensitivity profile*

Description

Visualises the output of [et_sensitivity_profile](#): for each noise grid point, shows how the **environmental share** of total variance and the **forecast horizon** respond. Observed horizons are drawn as solid points; projected horizons are hollow points; lower-bound rows are drawn as upward arrows at the last informative time.

Usage

```
et_plot_sensitivity(sens, show = c("horizon", "env_share", "ratio"))
```

Arguments

sens	An et_sensitivity object from et_sensitivity_profile .
show	"horizon" (default) shows the shelf-life horizon; "env_share" shows env_var / total_var; "ratio" shows the mean CI width / plausible range.

Details

The x-axis is the noise fraction (when the grid was built from `fraction_grid`) or the grid step label otherwise. When both a numeric fraction and a descriptive label exist, the fraction is preferred for continuous x-positioning.

Value

A ggplot2 object.

See Also

[et_sensitivity_profile](#)

et_plot_shelf_life *Plot forecast shelf life*

Description

Shows how credible interval width grows over the forecast horizon and marks the threshold beyond which the forecast is uninformative.

Usage

```
et_plot_shelf_life(sl, show_ratio = TRUE)
```

Arguments

sl	An et_shelf_life object from shelf_life .
show_ratio	Logical. If TRUE (default), plots the ratio (CI width / plausible range) rather than raw CI width.

Value

A ggplot2 object.

et_predict	<i>Posterior prediction with uncertainty decomposition</i>
------------	--

Description

Generates posterior predictive draws for new observations, propagates environmental measurement uncertainty through the model, and computes credible intervals. The resulting `et_prediction` object is the input to [decompose_uncertainty](#), [shelf_life](#), and the plotting functions.

Usage

```
et_predict(
  model,
  newdata,
  env_noise = NULL,
  env_cov = NULL,
  env_dist = NULL,
  n_draws = 2000L,
  ci_levels = c(0.5, 0.8, 0.9, 0.95),
  n_perturb = NULL,
  interval_type = c("predictive", "linpred"),
  include_env_in_ci = FALSE,
  ...
)
```

Arguments

model	An et_model or et_model_list object from et_fit .
newdata	A data.frame containing the predictor columns named in the model formula. For grouped models, must also contain the grouping column.
env_noise	Environmental measurement / prediction uncertainty. Can be: <ul style="list-style-type: none"> • NULL (default): no environmental noise. • A single numeric: applied as a fraction of each predictor's empirical SD in newdata (e.g. 0.1 means 10% noise, constant across all observations). • A named list or named numeric vector with one scalar per predictor: constant absolute noise SD per predictor, e.g. list(Tmean = 0.5, PPT = 10).

- A named list where each entry is a **numeric vector of length** `nrow(newdata)`: *time-varying* (per-row) noise SDs. Use this when predictor uncertainty grows with forecast horizon, e.g. from a GCM ensemble spread that increases over time: `list(Tmean = 0.30 + 0.01 * (years - base_year))`. Entries not supplied default to zero (no noise for that predictor).
- `env_cov` Correlation structure of the environmental noise. The *magnitudes* of the noise come from `env_noise`; `env_cov` supplies the *correlation* between predictors, so that a perturbation on row i is drawn from $\mathcal{N}(0, D_i R D_i)$ with $D_i = \text{diag}(\sigma_{i1}, \dots, \sigma_{ip})$ and R = the correlation matrix. One of:
- NULL (default): independent noise, $R = I$ — equivalent to ErrorTracer behaviour prior to this feature and the right choice when predictor measurement errors are genuinely independent (e.g. separate instruments on unrelated variables).
 - "empirical": compute the correlation of the predictor columns in the **training data** (`model$data`). Use this when predictor *errors* are expected to inherit the correlation structure of the predictors themselves — e.g. temperature and humidity that co-vary in the underlying climate system.
 - "newdata": compute the correlation of the predictor columns in `newdata`. Useful when the forecast window has a different covariance structure than training (e.g. scenario runs).
 - A numeric $p \times p$ matrix with `dimnames` matching the model's predictors. Entries with an off-diagonal exceeding 1 are rescaled to a correlation matrix. Use this to supply an independent estimate of the *error* correlation structure (e.g. from a reanalysis product or a sensor covariance report).
- A correlation derived from training data is a working assumption: the structure of the *errors* is assumed to mirror the structure of the *values*. When this is implausible, pass a matrix directly.
- `env_dist` Distributional form of the per-predictor noise. The `env_noise` SDs set the *magnitude* of the perturbation; `env_dist` sets its *shape*. For every distribution other than "gaussian", the noise is calibrated so that (approximately) $E[\tilde{x}] = x$ and $\text{Var}[\tilde{x}] = \sigma^2$, using a Gaussian copula to honour `env_cov`. One of:
- NULL (default): "gaussian" for every predictor — additive Gaussian noise, legacy behaviour.
 - A single string ("gaussian", "lognormal", "gamma", "beta"): applied to all predictors.
 - A named list / character vector with one entry per predictor to override the default, e.g. `list(PPT = "gamma", tmax = "gaussian")`.
- Distributions:
- "gaussian" Additive normal noise ($\tilde{x} = x + \varepsilon, \varepsilon \sim N(0, \sigma^2)$). Appropriate for symmetric measurement error on a continuous, potentially negative scale (temperature, anomalies).
- "lognormal" Multiplicative noise: $\log \tilde{x} \sim N(\log x - s^2/2, s^2)$ with $s^2 = \log(1 + (\sigma/x)^2)$. Preserves positivity; right-tail skewed. Natural for strictly positive continuous variables whose error scales with magnitude (e.g. enzyme activity, biomass). Rows with $x \leq 0$ are left unperturbed.

"gamma" $\tilde{x} \sim \text{Gamma}(\text{shape} = (x/\sigma)^2, \text{rate} = x/\sigma^2)$. Positive support, right-skewed, analytic mean/variance match. Natural for precipitation, rates, and other non-negative continuous variables. Rows with $x \leq 0$ are left unperturbed.

"beta" $\tilde{x} \sim \text{Beta}(\alpha, \beta)$ with $\alpha + \beta = x(1-x)/\sigma^2 - 1$. Support in $(0, 1)$; appropriate for proportions and probabilities (allele frequencies, presence rates). Rows with $x \notin (0, 1)$ or $\sigma^2 \geq x(1-x)$ are left unperturbed.

Correlation (env_cov) is applied to the latent standard-normal draws before the marginal quantile transform, so rank correlations are preserved across distributions.

n_draws	Integer. Number of posterior draws to use (default 2000; capped at the number of draws available in the fit).
ci_levels	Numeric vector. Credible interval levels to compute (default c(0.5, 0.8, 0.9, 0.95)).
n_perturb	Integer. Number of draws used for the environmental perturbation step (default min(500, n_draws)). Reducing this speeds up computation.
interval_type	Character. Which draws to use when computing credible intervals: <ul style="list-style-type: none"> • "predictive" (default): draws from posterior_predict, which include sigma (residual noise). Use this when forecasting individual observations — e.g. a single population's allele frequency, one site's ozone reading on a specific day. • "linpred": draws from posterior_linpred, which capture only parameter uncertainty (no sigma). Use this when forecasting the mean response — e.g. the expected ozone across many similar days, or mean delta f across replicate populations. These intervals are always narrower; they will under-cover individual observations unless sigma is negligible. <p>The decomposition components and posterior_predict/posterior_linpred matrices are always computed regardless of this setting.</p>
include_env_in_ci	Logical. When TRUE and interval_type = "predictive", credible intervals are constructed from environmentally inflated draws $\tilde{y} = \tilde{\text{lp}} + \varepsilon$, with $\tilde{\text{lp}}$ the perturbed linear predictor and $\varepsilon \sim N(0, \sigma^2)$ using posterior draws of σ . This folds the environmental uncertainty component back into the CI, which is typically what you want for sensitivity analyses or whenever the reported interval should cover predictor-measurement error. When FALSE (default, backward compatible), CIs are based on posterior_predict only — parameter + residual, without predictor noise.
...	Passed to methods.

Value

An et_prediction object (list) containing:

posterior_predict Matrix [n_draws x n_obs]: full posterior predictive draws (parameter + residual uncertainty).

posterior_linpred Matrix [n_draws x n_obs]: linear predictor draws (parameter uncertainty only).

lp_perturbed Matrix [n_perturb x n_obs]: linear predictor computed on environmentally perturbed inputs.

sigma_draws Numeric vector: posterior draws of sigma.

credible_intervals data.frame with columns row_id, ci_level, lower, median, upper, width.

decomposition data.frame from decompose_uncertainty(): param_var, env_var, residual_var, total_var.

newdata The input newdata.

model Reference to the et_model used.

env_cov The $p \times p$ correlation matrix actually used for perturbation (identity for env_cov = NULL).

env_dist Named character vector mapping each predictor to the distribution actually used for its perturbation.

See Also

[decompose_uncertainty](#), [shelf_life](#), [et_calibrate](#)

et_sensitivity_profile

Sensitivity profile of the environmental uncertainty component

Description

Sweeps a grid of noise magnitudes, re-runs [et_predict](#) for each, and summarises how the environmental variance component and the forecast shelf life respond. This turns the subjective choice of env_noise into an auditable *curve*: instead of reporting a single shelf life at one (often hand-picked) measurement-error level, the user sees how the horizon degrades as assumed noise grows.

Usage

```
et_sensitivity_profile(
  model,
  newdata,
  plausible_range,
  fraction_grid = NULL,
  absolute_grid = NULL,
  env_cov = NULL,
  env_dist = NULL,
  include_env_in_ci = TRUE,
  ci_level = 0.9,
  ci_levels = c(0.5, 0.8, 0.9, 0.95),
  threshold = 1,
  time_col = NULL,
  n_draws = 2000L,
```

```

    n_perturb = NULL,
    max_extrapolation_factor = 10,
    verbose = TRUE
  )

```

Arguments

model	An <code>et_model</code> from <code>et_fit</code> .
newdata	<code>data.frame</code> passed to <code>et_predict</code> .
plausible_range	Two-element numeric vector, as in <code>shelf_life</code> .
fraction_grid	Optional numeric vector of scalar noise fractions.
absolute_grid	Optional list of <code>env_noise</code> arguments (each a named list / vector). If supplied together with <code>fraction_grid</code> , <code>fraction_grid</code> is ignored.
env_cov, env_dist	Passed through to <code>et_predict</code> .
include_env_in_ci	Logical. If TRUE (default), credible intervals — and hence the shelf-life horizon — are computed from environmentally inflated draws (<code>et_predict(..., include_env_in_ci = TRUE)</code>). Set FALSE to see how <code>env_var</code> evolves while holding the CI fixed at parameter + residual.
ci_level	Numeric. Used for shelf life and CI width tracking (default 0.90). Must be in <code>ci_levels</code> .
ci_levels	Numeric vector of CI levels computed per iteration (default <code>c(0.5, 0.8, 0.9, 0.95)</code>).
threshold	Numeric. Shelf-life threshold (default 1.0).
time_col	Character. Optional time column for shelf life.
n_draws, n_perturb	Passed to <code>et_predict</code> .
max_extrapolation_factor	Passed to <code>shelf_life</code> .
verbose	Logical. If TRUE (default), logs each iteration.

Details

Three argument styles are supported for the grid:

- `fraction_grid`: scalar fractions of each predictor's SD in `newdata`. These are passed straight to the scalar form of `et_predict`'s `env_noise`. Use this for an apples- to-apples sweep that scales all predictors together.
- `absolute_grid`: a list of named numeric lists / vectors; each list element becomes a single `env_noise` argument (so you can sweep absolute SDs for one predictor while holding others fixed).
- Neither supplied (NULL): a default log-spaced fraction grid `c(0, 0.01, 0.025, 0.05, 0.1, 0.2, 0.4, 0.8)` is used.

Each iteration calls `et_predict(..., env_noise = g)` then `shelf_life` and records:

- Mean parameter / environmental / residual / total variance across forecast observations.
- The horizon description ("observed", "projected", or "lower_bound") and numeric horizon if any.
- Mean / max CI width / plausible-range ratio.

Value

An `et_sensitivity` object: a data.frame with one row per grid point and columns

`grid_id` 1-indexed step.

`label` Short descriptor of the `env_noise` used (e.g. "fraction = 0.05").

`fraction` Scalar fraction when applicable; NA for absolute-grid rows.

`param_var`, `env_var`, `residual_var`, `total_var` Mean across forecast observations.

`env_share` $\text{env_var} / (\text{env_var} + \text{total_var})$ — the fraction of combined predictive variance attributable to predictor noise (bounded in $[0, 1]$).

`ci_width_mean`, `ci_width_max` At `ci_level`.

`ratio_mean`, `ratio_max` CI width / plausible range.

`horizon_type` One of "observed", "projected", "lower_bound".

`horizon` Numeric horizon (observed or projected) or NA for lower-bound rows.

`horizon_description` The long description returned by `shelf_life`.

The returned object carries the original call and `plausible_range` as attributes for use by `et_plot_sensitivity`.

See Also

[et_predict](#), [shelf_life](#), [et_plot_sensitivity](#)

et_sim

Simulated allele-frequency-change dataset for ErrorTracer tutorials

Description

A named list containing training data, forecast predictors, validation responses, and ground-truth parameters for two synthetic SNP clusters (A and B) at a hypothetical mountain plant site. The dataset is designed to exercise every step of the ErrorTracer pipeline and to support parameter-recovery validation (true coefficients are known).

Usage

`et_sim`

Format

A named list with seven elements:

`train` `data.frame` with 40 rows and 6 columns (2 clusters \times 20 training years, 1995–2014):

year Integer. Calendar year.

cluster_id Character. SNP cluster identifier: "A" or "B".

Tmean Numeric. Standardised mean growing-season temperature (original unit: °C; standardised using training-period mean and SD).

PPT Numeric. Standardised total growing-season precipitation (original unit: mm).

SWE Numeric. Standardised peak snow water equivalent (original unit: mm).

z_diff Numeric. Simulated allele-frequency change on the arcsin-sqrt scale ($\arcsin(\sqrt{f})$ transformation). This transformation is unbounded and has no fixed $[-1, 1]$ constraint; the plausible range should be derived from the training data or from the theoretical bounds $([0, \pi/2] \setminus \{0\})$ on the arcsin scale).

`forecast` `data.frame` with 10 rows and 5 columns (2 clusters \times 5 forecast years, 2015–2019).

Same columns as `train` except `z_diff` is absent — these are the prediction targets. Predictors are standardised using the training-period statistics stored in `standardization`.

`validation` `data.frame` with 10 rows and 3 columns (`year`, `cluster_id`, `z_diff`). True response values for the forecast period; used with `et_calibrate` to assess posterior predictive coverage.

`true_params` Named list with one element per cluster. Each element is a named numeric vector of the true data-generating parameters: `intercept`, `Tmean`, `PPT`, `SWE`, and `sigma` (residual SD).

Cluster A `intercept = 0.00`, `Tmean = 0.50`, `PPT = -0.30`, `SWE = 0.20`, `sigma = 0.40`

Cluster B `intercept = 0.10`, `Tmean = 0.30`, `PPT = -0.20`, `SWE = -0.10`, `sigma = 0.50`

`env_noise` Named list. Suggested per-predictor environmental noise SDs (on the standardised scale) for use with the `env_noise` argument of `et_predict`: `Tmean = 0.30`, `PPT = 0.20`, `SWE = 0.15`.

`standardization` Named list with one element per predictor. Each element is a named numeric vector `c(mean = ..., sd = ...)` giving the training-period statistics used to standardise the data. Needed if you wish to back-transform predictions to the original (unstandardised) scale with `unstandardize`.

`description` Character string. Human-readable description of the dataset and how it was generated.

Details

The climate time series are simulated as AR(1) processes with a warming trend in `Tmean` ($+0.05$ °C yr^{-1}). `SWE` is derived from `Tmean` and `PPT` to introduce physically motivated covariance (colder winters produce more snowpack; wetter winters also increase snowpack). All predictors are standardised using training-period statistics only, which is the correct procedure when predictors in the forecast period may fall outside the training range.

Responses are generated from a linear model with Gaussian noise:

$$z_diff_t = \alpha + \beta_1 Tmean_t + \beta_2 PPT_t + \beta_3 SWE_t + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \sigma^2)$$

The true parameters are stored in `et_sim>true_params` for parameter-recovery validation.

The dataset was generated with `set.seed(111)`. The full generation script is at `data-raw/generate_et_sim.R`.

Source

Generated by data-raw/generate_et_sim.R with set.seed(111).

Examples

```
data(et_sim)

# Inspect structure
str(et_sim, max.level = 2)

# Training data
head(et_sim$train)

# True parameters for cluster A
et_sim$true_params$A

# Suggested noise SDs for et_predict()
et_sim$env_noise
```

et_theme

Minimal ggplot2 theme for ErrorTracer plots

Description

Minimal ggplot2 theme for ErrorTracer plots

Usage

```
et_theme(base_size = 12)
```

Arguments

base_size Base font size (default 12).

Value

A ggplot2 theme object.

extract_priors	<i>Extract brms prior specification from a regularized or standard regression model</i>
----------------	---

Description

Converts a fitted model object into a brms prior specification suitable for `et_fit`. The coefficient estimates (or importance weights for **ranger**) from the regularized or standard fit are used as informative prior means, so the Bayesian model starts close to the regularized or standard solution while remaining open to data-driven revision.

Usage

```
extract_priors(  
  model,  
  multiplier = 2,  
  min_sd = 0.1,  
  intercept_prior_sd = NULL,  
  sigma_prior_scale = 1,  
  ...  
)  
  
## S3 method for class 'lm'  
extract_priors(  
  model,  
  multiplier = 2,  
  min_sd = 0.1,  
  intercept_prior_sd = NULL,  
  sigma_prior_scale = 1,  
  ...  
)  
  
## S3 method for class 'glm'  
extract_priors(  
  model,  
  multiplier = 2,  
  min_sd = 0.1,  
  intercept_prior_sd = NULL,  
  sigma_prior_scale = 1,  
  ...  
)  
  
## S3 method for class 'cv.glmnet'  
extract_priors(  
  model,  
  multiplier = 2,  
  min_sd = 0.1,
```

```

    intercept_prior_sd = NULL,
    sigma_prior_scale = 1,
    lambda = "lambda.min",
    ...
)

## S3 method for class 'glmnet'
extract_priors(
  model,
  multiplier = 2,
  min_sd = 0.1,
  intercept_prior_sd = NULL,
  sigma_prior_scale = 1,
  s = 1L,
  ...
)

## S3 method for class 'ranger'
extract_priors(
  model,
  multiplier = 2,
  min_sd = 0.1,
  intercept_prior_sd = NULL,
  sigma_prior_scale = 1,
  ...
)

```

Arguments

model	A fitted model. Supported classes: <ul style="list-style-type: none"> • <code>cv.glmnet / glmnet</code> — elastic net / lasso • <code>lm</code> — ordinary least squares • <code>glm</code> — generalized linear model • <code>ranger</code> — random forest (importance-scaled flat priors)
multiplier	Numeric scalar. Prior SD is set to <code>multiplier * coef </code> (for signed-coefficient methods) or <code>multiplier * importance_normalised</code> (for <code>ranger</code>). Default 2.0.
min_sd	Numeric scalar. Minimum prior SD to avoid degenerate (spike) priors on near-zero coefficients. Default 0.1.
intercept_prior_sd	Optional prior SD for the intercept term. The default NULL emits <i>no</i> explicit intercept prior and lets brms pick its data-aware default (<code>student_t(3, median(y), mad(y))</code>). If you supply a numeric value, the prior becomes <code>normal(0, intercept_prior_sd)</code> — this is only appropriate when the response has been centred (or nearly so) before fitting, since brms's <code>class = "Intercept"</code> refers to the intercept at the centre of the predictors, whose posterior mean equals $E[y]$ there. Supplying a small <code>intercept_prior_sd</code> for an uncentred response will pull the intercept toward zero and cripple the fit.

sigma_prior_scale	Scale parameter for the half-Cauchy prior on the residual SD sigma. Default 1.0.
...	Additional arguments passed to methods.
lambda	Which lambda to extract coefficients from when the model is a cv.glmnet object. Either "lambda.min" (default) or "lambda.1se".
s	Index (integer) or value of lambda to extract coefficients at when the model is a plain glmnet object. Defaults to the first lambda (smallest regularisation).

Details

For ranger models, signed coefficients are not available. Priors are centred at zero (direction unknown) and the prior SD for each predictor is set to `multiplier * importance_normalised`, where importance is normalised to the `[min_sd, 1]` interval. Only variables with positive permutation importance are included.

Value

An `et_prior_spec` list containing:

`prior` A `brmsprior` object for use in `brms::brm()`.

`pred_names` Character vector of included predictor names.

`coefs` Named numeric vector of regularized coefficients (NULL for ranger, which uses importance instead).

`method` Character: the dispatch method used.

`multiplier, min_sd` Settings echo.

Examples

```
fit_lm <- lm(mpg ~ wt + hp + cyl, data = mtcars)
ps <- extract_priors(fit_lm, multiplier = 2, min_sd = 0.1)
print(ps)
```

shelf_life

Compute the forecast shelf life

Description

Quantifies *when* a forecast becomes uninformative by comparing the width of credible intervals to a plausible response range. A forecast is uninformative when its CI width exceeds `threshold * plausible_range`.

Usage

```
shelf_life(
  predictions,
  plausible_range,
  ci_level = 0.9,
  threshold = 1,
  time_col = NULL,
  min_slope_for_projection = 1e-04,
  max_extrapolation_factor = 10,
  ...
)
```

Arguments

predictions	An et_prediction or et_prediction_list.
plausible_range	Numeric vector of length 2 (c(min, max)) giving the plausible range of the response. For unbounded responses (e.g. \ arcsin-transformed allele-frequency change) use range(training_data\$response) or a biologically motivated interval. The effective range is diff(plausible_range).
ci_level	Numeric. The credible interval level to use (default 0.90). Must be present in the et_prediction object.
threshold	Numeric. CI width / plausible range above which the forecast is uninformative (default 1.0).
time_col	Character. Optional column in predictions\$newdata to use as the time axis. If NULL, observation index is used.
min_slope_for_projection	Numeric. Minimum linear slope (of ratio vs. \ time) required to attempt extrapolation when all periods are informative. Below this value the shelf life is reported as a lower bound only. Default 1e-4.
max_extrapolation_factor	Numeric. Cap on how far the linear projection may reach beyond the observed window. If the projected crossing time exceeds max(time) + max_extrapolation_factor * (max(time) - min(time)), the result is reported as a lower bound instead of a projection. Set to Inf to disable the cap. Default 10.
...	Unused.

Details

The function operates in three modes depending on the available data and whether the uninformative threshold is crossed within the forecast window:

Observed The threshold is crossed within the forecast/validation window. The shelf life is the first time point at which $\text{ratio} \geq \text{threshold}$.

Projected All forecast periods remain informative but the CI/range ratio is trending upward. A linear trend is fitted to the ratios and extrapolated to estimate when the threshold would be reached.

Lower bound All forecast periods are informative with no upward trend in the ratio. The shelf life is reported as a lower bound: > last observed time.

The intended workflow is:

1. Fit the model (`et_fit`).
2. Predict on held-out data or a future time window (`et_predict`).
3. Call `shelf_life()` on those predictions.
4. If the threshold is not crossed in the held-out window, the *projected* mode extrapolates the horizon automatically.

Value

An `et_shelf_life` object (a `data.frame`) with columns:

obs_id Observation index.

time Time axis value.

ci_width Width of the credible interval at `ci_level`.

plausible_range Effective plausible range (scalar).

ratio CI width / plausible range.

informative Logical; TRUE when `ratio < threshold`.

For grouped predictions a group column is prepended. The object carries three attributes that drive `print()`:

`horizon` Named list with elements `value`, `type` ("observed", "projected", or "lower_bound"), `last_informative`, and `description`.

`horizon_by_group` For grouped objects: named list of per-group horizon lists.

`threshold` The threshold value used.

Examples

```
set.seed(1)
df <- data.frame(y = rnorm(20), year = 2001:2020, x1 = rnorm(20))
fit <- et_fit(y ~ x1, data = df,
             chains = 1, iter = 500, warmup = 250,
             cores = 1, refresh = 0)
new_df <- data.frame(x1 = rnorm(10), year = 2021:2030)
pred <- et_predict(fit, newdata = new_df,
                  n_draws = 200, n_perturb = 50)
sl <- shelf_life(pred,
                 plausible_range = range(df$y),
                 ci_level = 0.90,
                 threshold = 1.0,
                 time_col = "year",
                 max_extrapolation_factor = 10)
print(sl)
```

standardize	<i>Standardize a numeric vector (mean-centre, unit-variance)</i>
-------------	--

Description

Standardize a numeric vector (mean-centre, unit-variance)

Usage

```
standardize(x)
```

Arguments

x	Numeric vector.
---	-----------------

Value

Standardized numeric vector. Returns zeros if variance is zero.

unstandardize	<i>Reverse standardization</i>
---------------	--------------------------------

Description

Reverse standardization

Usage

```
unstandardize(z, mu, s)
```

Arguments

z	Standardized values.
mu	Original mean.
s	Original standard deviation.

Value

Values on the original scale.

Index

* datasets

et_sim, 16

decompose_uncertainty, 2, 8, 11, 14

et_calibrate, 3, 7, 14, 17

et_diagnose, 5

et_fit, 5, 11, 15, 19, 23

et_plot_calibration, 7

et_plot_coefficients, 7

et_plot_decomposition, 8

et_plot_forecast, 8

et_plot_prior_posterior, 9

et_plot_sensitivity, 10, 16

et_plot_shelf_life, 10

et_predict, 2, 3, 11, 14–17, 23

et_sensitivity_profile, 10, 14

et_sim, 16

et_theme, 18

extract_priors, 5, 6, 19

shelf_life, 11, 14–16, 21

standardize, 24

unstandardize, 17, 24