

Package: actuarialfitdist (via r-universe)

October 28, 2026

Type Package

Title Actuarial Severity Fitting with Limits and Deductibles

Version 0.2.3

Description Fits actuarial claim severity distributions to ground-up loss data while accounting for deductible truncation, policy-limit censoring, observation weights, optional severity scaling variables, and fixed parameters. Supports single-family and spliced body-tail models, candidate-model comparison, limited expected values, loss elimination ratios, increased limits factors, deductible and limit rating factors, valuation or depreciation adjustments, simulation, diagnostics, and uncertainty estimation. The package is designed for property and casualty actuarial applications in which analysts fit ground-up severity and translate the fitted distribution into pricing factors for limits, deductibles, and valuation provisions. For background on statistical severity modeling, see Klugman, Panjer, and Willmot (2019, ISBN:9781119523789), ``Loss Models: From Data to Decisions'', 5th edition.

License MIT + file LICENSE

URL <https://github.com/gs-actuary/actuarialfitdist>

BugReports <https://github.com/gs-actuary/actuarialfitdist/issues>

Encoding UTF-8

Depends R (>= 4.1.0)

Imports ggplot2, scales, stats, utils

Suggests fitdistrplus, knitr, plotly, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Config/roxygen2/version 8.1.0

Author Greg Sollenberger [aut, cre, cph]

Maintainer Greg Sollenberger <yagreg7@yahoo.com>
Repository https://cran.r-universe.dev
Date/Publication 2026-10-28 11:50:16 UTC
RemoteUrl https://github.com/cran/actuarialfitdist
RemoteRef HEAD
RemoteSha 4a7245034651d871485153b4283b0565d4d12c86

Contents

actuarialfitdist_fit_methods	2
bootstrap_fit	3
candidate_plot_data	4
compare_factors	5
diagnose_fit	6
exceedance_probability	7
expected_payment	7
fit_plot_data	8
fit_severity	9
fit_severity_candidates	11
fit_spliced_severity	13
increased_limits_factor	15
limited_expected_value	16
loss_elimination_ratio	17
plot_candidate_fits	18
plot_fit	19
plot_scaling_fit	21
predict.actuarialfitdist_fit	22
scaling_plot_data	23
severity_factors	24
severity_quantile	25
simulate.actuarialfitdist_fit	26
Index	28

actuarialfitdist_fit_methods
<i>Inspect fitted actuarial severity models</i>

Description

These standard R methods display model results and provide access to coefficients, log-likelihood, information criteria and (when available) the estimated covariance matrix and confidence intervals. `summary()` gives natural-scale estimates and their standard errors. A Hessian-based variance estimate can be unavailable for numerically difficult fits; in that case `vcov()` and `confint()` require a successful refit with `hessian = TRUE`.

Details

`plot(fit)` calls `plot_fit(fit)`. See `?predict.actuarialfitdist_fit` and `?simulate.actuarialfitdist_fit` for prediction and simulation.

Examples

```
set.seed(1)
d <- data.frame(loss = rlnorm(150, 9, 0.8))
fit <- fit_severity(d, "loss", hessian = FALSE)
summary(fit)
coef(fit)
logLik(fit)
```

bootstrap_fit	<i>Bootstrap a fitted severity model</i>
---------------	--

Description

Provides a deliberately lightweight bootstrap. Parametric bootstrap retains the fitted portfolio's deductible, limit, scaling, and weight structure, simulates observed claims conditional on exceeding the deductible, and refits the model. Case bootstrap resamples claim rows with replacement.

Usage

```
bootstrap_fit(
  object,
  R = 200,
  type = c("parametric", "case"),
  seed = NULL,
  refit_threshold = FALSE
)
```

Arguments

<code>object</code>	Fitted model.
<code>R</code>	Number of bootstrap replicates.
<code>type</code>	"parametric" or "case".
<code>seed</code>	Optional random seed.
<code>refit_threshold</code>	For a searched spliced model, whether each bootstrap replicate should repeat threshold search. Default FALSE reuses the selected threshold to keep the bootstrap inexpensive and stable.

Value

An `actuarialfitdist_bootstrap` list containing estimates (one row per replicate, including coefficients, convergence and any error), `success_rate`, `R`, `type`, and `original` (the input fit). Inspect failures before using the empirical distribution of bootstrap estimates.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
boot <- bootstrap_fit(fit, R = 3, type = "case", seed = 1)
boot$success_rate
```

candidate_plot_data	<i>Data for comparing several candidate severity distributions</i>
---------------------	--

Description

Recreates each candidate's row-level deductible truncation and limit censoring, then returns actual and simulated observed claim amounts with the same scale-range filter. Inspect or modify the data with `ggplot2`/`plotly`.

Usage

```
candidate_plot_data(
  object,
  scale_range = NULL,
  nsim = 3,
  seed = NULL,
  output = c("raw", "plot_ready"),
  type = c("density", "histogram", "cdf", "survival"),
  bins = 40,
  points = 160,
  x_scale = c("log10", "linear"),
  xlim = NULL
)
```

Arguments

<code>object</code>	Output of <code>fit_severity_candidates()</code> .
<code>scale_range</code>	Optional range of <code>scale_by</code> values.
<code>nsim</code>	Number of observed-portfolio simulations per candidate.
<code>seed</code>	Optional reproducibility seed.
<code>output</code>	"raw" (default) returns claim-level simulations; "plot_ready" returns deterministic bin densities/CDF/survival values consumed by <code>plot_candidate_fits(method = "analytic")</code> .
<code>type</code>	Plot-ready summary type.
<code>bins</code>	Histogram bin count.
<code>points</code>	Density/CDF grid intervals.
<code>x_scale</code>	"log10" (default) or "linear" plot coordinate.
<code>xlim</code>	Optional plot-ready grid range; not a claim filter.

Value

A data frame with a distribution column for each successful candidate. output = "raw" returns actual and simulated claim-level observations with observed_loss, censoring flags and weights. output = "plot_ready" returns source, x, value, lower, upper, and distribution, matching the default candidate plot; source = "modeled" denotes expected portfolio probabilities, not Monte Carlo simulations.

Examples

```
set.seed(11)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
candidates <- fit_severity_candidates(d, loss = "loss",
  distributions = c("lognormal", "gamma"), hessian = FALSE)
head(candidate_plot_data(candidates, nsim = 1, seed = 1))
```

compare_factors

Compare factor outputs side by side

Description

Evaluates the same rating scenarios for each successful candidate fit and reshapes one selected factor metric into a distribution-by-distribution table.

Usage

```
compare_factors(object, metric = "rating_factor", ...)
```

Arguments

object	A candidate-comparison object.
metric	One output column from severity_factors() to compare.
...	Arguments passed to severity_factors().

Value

A wide data frame with the coverage scenario columns used by severity_factors() and one value.<distribution> column for each successful candidate, for the chosen metric.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
candidates <- fit_severity_candidates(d, "loss",
  distributions = c("lognormal", "gamma"), hessian = FALSE)
compare_factors(candidates, deductible = 1000, limit = c(25000, 50000))
```

`diagnose_fit`*Diagnose fit within a severity layer*

Description

Compares actual and simulated claim behavior within a user-selected observed severity layer, optionally restricted to a range of scaling values.

Usage

```
diagnose_fit(  
  object,  
  lower = 0,  
  upper = Inf,  
  scale_range = NULL,  
  nsim = 100,  
  seed = NULL  
)
```

Arguments

<code>object</code>	Fitted model.
<code>lower</code>	Lower observed-severity bound.
<code>upper</code>	Upper observed-severity bound.
<code>scale_range</code>	Optional range of scaling values.
<code>nsim</code>	Number of simulated observed portfolios.
<code>seed</code>	Optional random seed.

Value

A data frame with one row for actual and one for simulated observed claims. Columns include `source`, `rows`, `layer_count`, `layer_proportion`, `layer_mean`, and `censoring_rate`. Simulated counts are averaged over replicates; results reflect observed (possibly censored) severity, not uncensored ground-up losses.

Examples

```
set.seed(123)  
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))  
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)  
diagnose_fit(fit, lower = 1000, upper = 30000, nsim = 1)
```

exceedance_probability

Exceedance probabilities from a fitted model

Description

Returns fitted ground-up probabilities of exceeding specified severity thresholds, optionally at supplied values of the scaling variable.

Usage

```
exceedance_probability(object, threshold, scale_value = NULL)
```

Arguments

object	A fitted actuarialfitdist model.
threshold	Ground-up severity threshold.
scale_value	Optional scaling value for a model fitted with scale_by.

Value

Numeric vector of fitted probabilities that positive ground-up severity exceeds each threshold.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(100000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
exceedance_probability(fit, 100000)
```

expected_payment

Expected payment under deductible, limit, and valuation terms

Description

Computes the expectation of $\text{pmin}(\text{pmax}(v * X - d, 0), L)$, where X is positive ground-up severity, v is valuation_factor, d is deductible, and L is limit.

Usage

```
expected_payment(
  object,
  deductible = 0,
  limit = Inf,
  scale_value = NULL,
  valuation_factor = 1
)
```

Arguments

object	A fitted model.
deductible	Deductible amount(s).
limit	Payment limit(s); Inf is allowed.
scale_value	Optional scaling value(s).
valuation_factor	Nonnegative multiplicative valuation factor applied to ground-up severity before deductible and limit.

Value

Numeric vector of expected insurer payments, in loss currency units, calculated as the expectation of $\text{pmin}(\text{pmax}(\text{valuation_factor} * X - \text{deductible}, 0), \text{limit})$ for positive ground-up severity X . Here *limit* is a payment cap *after* deductible, not the ground-up observation censoring limit used while fitting.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
expected_payment(fit, deductible = 1000, limit = 25000)
```

fit_plot_data

Data underlying observed-versus-modeled severity plots

Description

Produces a censoring- and truncation-aware comparison data set. Modeled values are simulated under the fitted model using each selected row's actual deductible, limit, and scaling value. This avoids the misleading practice of overlaying one unconditional fitted density on a heterogeneous portfolio of deductible-truncated and limit-censored claims.

Usage

```
fit_plot_data(
  object,
  scale_range = NULL,
  nsim = 20,
  seed = NULL,
  output = c("raw", "plot_ready"),
  type = c("density", "histogram", "cdf", "survival"),
  bins = 40,
  points = 160,
  x_scale = c("log10", "linear"),
  xlim = NULL
)
```

Arguments

object	Fitted model.
scale_range	Optional two-element range of fitted scaling values.
nsim	Number of simulated observed portfolios.
seed	Optional random seed.
output	"raw" (default) preserves individual actual/simulated rows; "plot_ready" returns the exact deterministic summary consumed by the default plotting functions, with columns source, x, value, lower, and upper. value is probability density in the displayed x coordinate (per log10 severity when requested), or a CDF/survival probability.
type	Summary type when output = "plot_ready".
bins	Number of histogram bins.
points	Number of grid intervals for density/CDF/survival summaries.
x_scale	"log10" or "linear" for plot-ready summaries.
xlim	Optional grid interval. Unlike data filtering, the probabilities are computed relative to the complete selected portfolio.

Value

A data frame. For output = "raw", each selected original claim and each simulated replicate has row, source, simulation, observed_loss, censored, deductible, limit, scale_value, and weight. For output = "plot_ready", columns source, x, value, lower, and upper give the deterministic observed and modeled values used in analytic plots. Histograms/densities use density per displayed coordinate (per log10-severity unit when x_scale = "log10"); CDF/survival values are probabilities. The analytic output does not depend on nsim or seed.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
head(fit_plot_data(fit, nsim = 1, seed = 1))
```

fit_severity	<i>Fit an actuarial severity distribution</i>
--------------	---

Description

Fits a parametric severity distribution to ground-up insurance losses using maximum likelihood. Claims below the deductible are treated as unobserved (left truncation) and observations at the policy limit are treated as right-censored.

Usage

```
fit_severity(
  data,
  loss,
  deductible = NULL,
  limit = NULL,
  distribution = "lognormal",
  scale_by = NULL,
  scale_intercept = FALSE,
  weights = NULL,
  fixed = list(),
  control = list(),
  hessian = TRUE,
  invalid_rows = c("warn_drop", "error")
)
```

Arguments

<code>data</code>	A data frame containing claim-level observations.
<code>loss</code>	Character scalar naming the ground-up loss column.
<code>deductible</code>	Optional character scalar naming the deductible column; omitted means no deductible.
<code>limit</code>	Optional character scalar naming the ground-up censoring-limit column. Omission means no censoring; use <code>Inf</code> for individually unlimited claims.
<code>distribution</code>	Distribution family. See <code>?actuarialfitdist</code> .
<code>scale_by</code>	Optional character scalar naming a numeric severity-scaling variable. For homeowners this may be Coverage A; for other lines it may be TIV, vehicle value, or a user-created transformation. The package treats it generically. Supported families are parameterized so finite mean severity is proportional to <code>scale_coef * scale_by + scale_intercept</code> .
<code>scale_intercept</code>	Logical. If <code>FALSE</code> , the scale intercept is fixed exactly at zero. If <code>TRUE</code> , it is estimated subject to positive fitted scales for all observations used in fitting.
<code>weights</code>	Optional character scalar naming nonnegative likelihood weights. A weight of 2 contributes twice the row log-likelihood and is equivalent to two identical observations. Noninteger weights define a weighted or pseudo-likelihood.
<code>fixed</code>	Named list of natural-scale parameters to hold fixed.
<code>control</code>	List passed to <code>stats::optim()</code> . When an additive scale intercept is estimated, the package sets the optimizer's <code>parscale</code> for that coordinate to <code>max(1000, 0.1 * median(loss))</code> ; other free coordinates use 1. Explicit <code>control\$parscale</code> overrides this heuristic and must follow the order in <code>\$optim_par</code> . This affects numerical optimization, not the statistical scale model or reported coefficients. The fitter checks a zero-intercept anchor where applicable and retries numerically sensitive families. Invalid apparent convergence is rejected.
<code>hessian</code>	Logical; calculate a numerical Hessian and covariance matrix.

`invalid_rows` "warn_drop" excludes anomalous rows with a warning (default); "error" stops and identifies affected original positions. Anomalies include ground-up loss at/below deductible and above a finite censoring limit; investigate the valuation basis before fitting.

Value

An `actuarialfitdist_fit` object. Important components include coefficients (named estimates on their natural scales), `logLik`, AIC, BIC, convergence, and `optimizer_counts`. `data` contains the retained observations; `excluded_rows` records original row positions excluded during validation, and `excluded_details` explains why. `vcov` is a covariance matrix when it can be estimated (`hessian = TRUE`); otherwise it is `NULL`. `convergence = 0` is an optimizer diagnostic, not proof that the chosen distribution adequately describes the claim tail.

Critical ground-up-loss assumption

loss must contain ground-up claim severities on a consistent valuation basis. It must not contain insurer payments, losses net of deductible, excess-of-deductible amounts, or ACV/depreciated payments unless the analyst has first converted those observations to the intended ground-up basis. The package deliberately does not guess or reconstruct this basis.

Observation mechanism versus payment factors

For fitting, `deductible` and `limit` are **column names** giving each claim's left-truncation threshold and *ground-up censoring threshold*. A claim recorded exactly at its finite limit is right-censored. In `expected_payment()` and `severity_factors()`, `limit` instead means a **post-deductible insurer payment cap**, supplied as an amount. These definitions must not be interchanged.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal")
summary(fit)
```

`fit_severity_candidates`

Fit and compare candidate severity distributions

Description

Fits several supported families to the same ground-up-loss data and returns both the fitted models and a comparison table with per-distribution optimization runtimes, evaluation counts, and convergence codes. The function intentionally does not choose a winner automatically; actuarial tail behavior and rating factor stability can matter even when information criteria are close.

Usage

```
fit_severity_candidates(
  data, loss, deductible = NULL, limit = NULL,
  distributions = c("weibull", "gamma", "lognormal",
    "pareto2", "burr", "normal", "exponential",
    "loglogistic", "invgauss", "gpd"),
  scale_by = NULL, scale_intercept = FALSE,
  weights = NULL, fixed = list(), control = list(),
  hessian = TRUE, invalid_rows = c("warn_drop", "error")
)
```

Arguments

<code>data</code>	A data frame containing claim-level observations.
<code>loss</code>	Character scalar naming the ground-up loss column.
<code>deductible</code>	Optional character scalar naming the deductible column; omitted means no deductible.
<code>limit</code>	Optional character scalar naming the ground-up censoring-limit column. Omission means no censoring; use <code>Inf</code> for individually unlimited claims.
<code>distributions</code>	Character vector of candidate families. Defaults to all supported single-family distributions.
<code>scale_by</code>	Optional character scalar naming a numeric severity-scaling variable. For homeowners this may be Coverage A; for other lines it may be TIV, vehicle value, or a user-created transformation. The package treats it generically. Supported families are parameterized so finite mean severity is proportional to $\text{scale_coef} * \text{scale_by} + \text{scale_intercept}$.
<code>scale_intercept</code>	Logical. If <code>FALSE</code> , the scale intercept is fixed exactly at zero. If <code>TRUE</code> , it is estimated subject to positive fitted scales for all observations used in fitting.
<code>weights</code>	Optional character scalar naming nonnegative likelihood weights. A weight of 2 contributes twice the row log-likelihood and is equivalent to two identical observations. Noninteger weights define a weighted or pseudo-likelihood.
<code>fixed</code>	Named list of natural-scale parameters to hold fixed.
<code>control</code>	List passed to <code>stats::optim()</code> . When an additive scale intercept is estimated, the package sets the optimizer's <code>par</code> scale for that coordinate to $\max(1000, 0.1 * \text{median}(\text{loss}))$; other free coordinates use 1. Explicit <code>control\$parscale</code> overrides this heuristic and must follow the order in <code>\$optim_par</code> . This affects numerical optimization, not the statistical scale model or reported coefficients. The fitter checks a zero-intercept anchor where applicable and retries numerically sensitive families. Invalid apparent convergence is rejected.
<code>hessian</code>	Logical; calculate a numerical Hessian and covariance matrix.
<code>invalid_rows</code>	"warn_drop" excludes anomalous rows with a warning (default); "error" stops and identifies affected original positions. Anomalies include ground-up loss at/below deductible and above a finite censoring limit; investigate the valuation basis before fitting.

Value

An actuarialfitdist_candidates list with fits (named fitted objects or errors), comparison (one row per requested family), excluded_rows, and excluded_details. comparison includes converged, logLik, AIC, BIC, error, and per-family fitting time and optimizer evaluations. Failed fits have missing fit statistics. No winning model is selected automatically; compare tail behavior and payment factors as well.

Choosing a distribution

Fit criteria are comparable only for fits using the same data and observation likelihood. Check converged and error before comparing AIC/BIC. Prefer a distribution with credible tail and layer behavior, rather than selecting the smallest AIC mechanically.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
candidates <- fit_severity_candidates(d, "loss",
  distributions = c("lognormal", "gamma"), hessian = FALSE)
candidates$comparison
candidates$fits$lognormal
plot_candidate_fits(candidates)
```

fit_spliced_severity *Fit a spliced body-tail severity model*

Description

Fits a two-component severity model with one family below a threshold and a second family above it. Each component is normalized on its side of the threshold and a fitted splice probability controls the probability mass in the body. Density continuity at the threshold is not imposed.

Usage

```
fit_spliced_severity(
  data,
  loss,
  deductible = NULL,
  limit = NULL,
  body = "lognormal",
  tail = "pareto2",
  threshold = NULL,
  threshold_method = c("fixed", "search"),
  threshold_probs = c(0.65, 0.95),
  threshold_grid = 15L,
  scale_by = NULL,
  scale_intercept = FALSE,
```

```

weights = NULL,
fixed = list(),
control = list(),
hessian = TRUE,
invalid_rows = c("warn_drop", "error"),
min_tail_exact = 25L
)

```

Arguments

<code>data</code>	A data frame containing claim-level observations.
<code>loss</code>	Character scalar naming the ground-up loss column.
<code>deductible</code>	Optional character scalar naming the deductible column; omitted means no deductible.
<code>limit</code>	Optional character scalar naming the ground-up censoring-limit column. Omission means no censoring; use <code>Inf</code> for individually unlimited claims.
<code>body</code>	Distribution family used below the splice threshold.
<code>tail</code>	Distribution family used above the splice threshold.
<code>threshold</code>	Numeric threshold for <code>threshold_method = "fixed"</code> . For search mode, it may optionally supply candidate thresholds.
<code>threshold_method</code>	Either <code>"fixed"</code> or <code>"search"</code> .
<code>threshold_probs</code>	For search mode when explicit thresholds are not supplied, two probabilities defining the range of empirical loss quantiles searched. Defaults to 0.65 through 0.95.
<code>threshold_grid</code>	Number of candidate thresholds used in search mode.
<code>scale_by</code>	Optional character scalar naming a numeric variable (such as Coverage A) that scales the body and tail family parameters. Because the splice threshold is in absolute severity units, a spliced model's overall mean need not be exactly proportional to the scaling variable.
<code>scale_intercept</code>	Logical. If <code>FALSE</code> , the scale intercept is fixed exactly at zero. If <code>TRUE</code> , it is estimated subject to positive fitted scales for all observations used in fitting.
<code>weights</code>	Optional character scalar naming nonnegative likelihood weights. A weight of 2 contributes twice the row log-likelihood and is equivalent to two identical observations. Noninteger weights define a weighted or pseudo-likelihood.
<code>fixed</code>	Optional list. Use <code>body = list(...)</code> , <code>tail = list(...)</code> , and optionally <code>splice_prob = ...</code> to fix parameters.
<code>control</code>	List passed to <code>stats::optim()</code> . When an additive scale intercept is estimated, the package sets the optimizer's <code>parscale</code> for that coordinate to <code>max(1000, 0.1 * median(loss))</code> ; other free coordinates use 1. Explicit <code>control\$parscale</code> overrides this heuristic and must follow the order in <code>\$optim_par</code> . This affects numerical optimization, not the statistical scale model or reported coefficients. The fitter checks a zero-intercept anchor where applicable and retries numerically sensitive families. Invalid apparent convergence is rejected.

<code>hessian</code>	Logical; calculate a numerical Hessian and covariance matrix.
<code>invalid_rows</code>	"warn_drop" excludes anomalous rows with a warning (default); "error" stops and identifies affected original positions. Anomalies include ground-up loss at/below deductible and above a finite censoring limit; investigate the valuation basis before fitting.
<code>min_tail_exact</code>	Minimum uncensored observations strictly above a threshold. Candidates failing this condition are skipped and described in <code>threshold_search</code> .

Details

The same critical ground-up-loss assumption as `fit_severity()` applies. The loss column must contain ground-up severities. Deductibles define left truncation and limits define right censoring.

Value

An `actuarialfitdist_fit` object with body and tail coefficients, fitted splice probability, chosen threshold, fit statistics, and the excluded-row audit. If `threshold_method = "search"`, `threshold_search` records attempted thresholds, exact uncensored tail counts, convergence, fit statistics, and reasons for rejecting candidates. A successful optimizer code does not guarantee a stable extrapolated tail.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(50, log(10000), 0.8))
fit <- fit_spliced_severity(d, "loss", body = "lognormal", tail = "pareto2",
  threshold = as.numeric(stats::quantile(d$loss, 0.75)), hessian = FALSE,
  min_tail_exact = 5)
fit$threshold
```

`increased_limits_factor`

Increased limits factor

Description

Computes the ratio of expected payments at a target limit to expected payments at a base limit, holding the deductible and valuation basis fixed.

Usage

```
increased_limits_factor(
  object,
  limit,
  base_limit,
  scale_value = NULL,
  deductible = 0,
  valuation_factor = 1
)
```

Arguments

<code>object</code>	A fitted model.
<code>limit</code>	Target payment limit.
<code>base_limit</code>	Base payment limit.
<code>scale_value</code>	Optional scaling value.
<code>deductible</code>	Deductible applied to both target and base coverage.
<code>valuation_factor</code>	Multiplicative valuation factor applied before deductible.

Value

Numeric vector of increased limits factors: expected payment at each target `limit` divided by the payment at `base_limit`, for the same deductible and valuation assumptions.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
increased_limits_factor(fit, limit = 50000, base_limit = 25000)
```

limited_expected_value

Limited expected severity

Description

Computes the expectation of $\text{pmin}(X, \text{limit})$ for positive ground-up severity X . For models fitted with a scaling variable, supply the corresponding `scale_value`.

Usage

```
limited_expected_value(object, limit, scale_value = NULL)
```

Arguments

<code>object</code>	A fitted model.
<code>limit</code>	Positive severity limit.
<code>scale_value</code>	Optional scaling value.

Value

Numeric vector of expected ground-up severity limited at each supplied `limit`, in the same currency units as the loss data. `Inf` requests the unlimited mean when it exists.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
limited_expected_value(fit, 25000)
```

loss_elimination_ratio

Loss elimination ratio

Description

Calculates one minus the expected payment after deductible divided by the expected positive ground-up severity. Infinite-mean models return NA when the denominator is not finite.

Usage

```
loss_elimination_ratio(
  object,
  deductible,
  scale_value = NULL,
  valuation_factor = 1
)
```

Arguments

object	A fitted model.
deductible	Deductible amount.
scale_value	Optional scaling value.
valuation_factor	Multiplicative valuation factor applied before the deductible.

Value

Numeric vector of loss elimination ratios: one minus expected post-deductible payment divided by expected payment without a deductible, holding the valuation factor fixed. Returns NA when the denominator is not finite and positive.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
loss_elimination_ratio(fit, deductible = 1000)
```

plot_candidate_fits *Compare observed and fitted severity across candidate distributions*

Description

Facets or overlays observed severity against modeled severity histograms, densities, CDFs or survival curves. The default uses expected probabilities, not random simulations. Uses a shared log10 axis by default. Fitted comparisons reproduce the observation mechanism, not unconditional ground-up densities. Returned ggplot can be customized.

Usage

```
plot_candidate_fits(
  object,
  type = c("histogram", "density", "cdf", "survival"),
  scale_range = NULL,
  nsim = 3,
  seed = NULL,
  bins = 40,
  x_scale = c("log10", "linear"),
  xlim = NULL,
  facet = TRUE,
  method = c("analytic", "simulation"),
  points = 160,
  x_breaks = NULL,
  x_text_angle = if (facet) 45 else 0,
  x_text_size = 9,
  y_text_size = 9
)
```

Arguments

object	Result of fit_severity_candidates().
type	Plot type: "histogram", "density", "cdf", or "survival".
scale_range	Optional scale-by filter.
nsim	Number of simulated portfolios when method = "simulation"; ignored by the deterministic analytic default.
seed	Random seed.
bins	Histogram bins.
x_scale	"log10" (default) or "linear".
xlim	Optional visible x range.
facet	Logical; facet models (default) or overlay them.
method	"analytic" (default) uses fast deterministic modeled bin probabilities; "simulation" uses the older row-level simulation.

points	Number of density and CDF grid intervals in analytic mode.
x_breaks	Optional explicit severity-axis tick positions.
x_text_angle	Axis label rotation (degrees); defaults to 45 in facets.
x_text_size	Horizontal-axis tick font size in points.
y_text_size	Vertical-axis tick font size in points.

Value

A ggplot2 plot, faceted by distribution unless facet = FALSE. Only successfully fitted candidates are plotted. The default is analytic (deterministic); candidate_plot_data(output = "plot_ready") returns the exact plotting values for custom charts.

Interpretation

Analytic curves compare the empirical **observed** severity distribution to the fitted distribution after the same observation rules. They are not unconditional ground-up density curves. nsim and seed affect only method = "simulation". Models that fail fitting are omitted, and their errors remain in object\$comparison.

Examples

```
set.seed(2)
d <- data.frame(loss = rlnorm(150, log(9000), 0.9))
fits <- fit_severity_candidates(d, "loss",
  distributions = c("lognormal", "weibull"), hessian = FALSE)
plot_candidate_fits(fits)
```

plot_fit	<i>Plot censoring-aware goodness of fit</i>
----------	---

Description

The default log10 severity axis reveals the body of heavy-tailed claim distributions without discarding large losses. For histograms, ggplot computes density in log-severity coordinates, ensuring simulated and observed bars use the same normalization. Axis limits only zoom the view.

Usage

```
plot_fit(
  object,
  type = c("histogram", "density", "cdf", "survival"),
  scale_range = NULL,
  nsim = 20,
  seed = NULL,
  bins = 30,
  x_scale = c("log10", "linear"),
  xlim = NULL,
```

```

method = c("analytic", "simulation"),
points = 160,
x_breaks = NULL,
x_text_angle = 0,
x_text_size = 9,
y_text_size = 9
)

```

Arguments

object	Fitted model.
type	"histogram", "density", "cdf", or "survival".
scale_range	Optional two-element range of fitted scaling values.
nsim	Number of simulated observed portfolios.
seed	Optional random seed.
bins	Number of histogram bins.
x_scale	"log10" (default) or "linear".
xlim	Optional visible x-axis range (without dropping observations).
method	"analytic" (default) computes fitted expected bin counts and CDF directly, including censoring atoms. "simulation" retains the earlier simulation-based diagnostic; nsim is used only in this mode.
points	Number of intervals for analytic density and CDF summaries.
x_breaks	Optional user-supplied x-axis tick positions.
x_text_angle	Axis tick label rotation in degrees.
x_text_size	Horizontal-axis tick label font size in points.
y_text_size	Vertical-axis tick label font size in points.

Value

A ggplot2 plot comparing observed severity with modeled observations on the same deductible, limit and scaling basis. The default analytic method uses deterministic expected probabilities; choose method = "simulation" for a Monte Carlo comparison. To customize the underlying analytic values, use `fit_plot_data(output = "plot_ready")`.

Analytic versus simulation plots

method = "analytic" (the default) calculates expected observed-claim probabilities across the actual mix of deductibles, limits, and scaling values. It is deterministic and ignores nsim and seed. Use method = "simulation" when you want to see Monte Carlo variability; nsim controls the number of simulated portfolios in that mode.

Examples

```

set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
plot_fit(fit, x_scale = "log10")

```

plot_scaling_fit	<i>Plot the fitted scaling relationship</i>
------------------	---

Description

Compares empirical observed mean severity by scale bin with simulated observed means and the fitted ground-up mean relationship.

Usage

```
plot_scaling_fit(  
  object,  
  bins = 10,  
  bin_method = c("quantile", "width"),  
  scale_range = NULL,  
  nsim = 50,  
  seed = NULL,  
  x_breaks = NULL,  
  x_text_angle = 0,  
  x_text_size = 9,  
  y_text_size = 9  
)
```

Arguments

object	Model fitted with scale_by.
bins	Number of bins.
bin_method	"quantile" or "width".
scale_range	Optional scaling range.
nsim	Number of simulated portfolios.
seed	Optional random seed.
x_breaks	Optional numeric x tick positions.
x_text_angle	Rotation of the scale-axis labels in degrees.
x_text_size	Size of the scale-axis tick labels.
y_text_size	Size of the mean-severity-axis tick labels.

Value

A ggplot2 object showing observed means, simulated observed means, and fitted ground-up means by scaling-variable bin. The plotted data can be obtained with `scaling_plot_data(output = "plot_ready")`.

Examples

```
set.seed(2)
d <- data.frame(cov_a = rep(c(200000, 400000, 600000), each = 80))
d$loss <- rlnorm(nrow(d), log(.07*d$cov_a), .8)
f <- fit_severity(d, "loss", scale_by = "cov_a", hessian = FALSE)
plot_scaling_fit(f, bins = 3, nsim = 1)
```

predict.actuarialfitdist_fit

Predict severity quantities

Description

Evaluates fitted means, quantiles, survival probabilities, CDF values, or densities for single-family and spliced severity models.

Usage

```
## S3 method for class 'actuarialfitdist_fit'
predict(
  object,
  newdata = NULL,
  scale_value = NULL,
  type = c("mean", "quantile", "survival", "cdf", "density"),
  p = 0.5,
  x = NULL,
  ...
)
```

Arguments

object	Fitted model.
newdata	Optional data frame containing the original scaling column.
scale_value	Optional numeric scaling value(s). Used instead of newdata.
type	One of mean, quantile, survival, cdf, or density.
p	Probability for quantiles.
x	Severity value for CDF, survival, or density predictions.
...	Unused.

Value

Numeric fitted values. Means and quantiles are in ground-up severity units; survival and CDF results are probabilities, and density results are probability densities. Results are conditional on positive ground-up severity.

Examples

```
set.seed(1)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", hessian = FALSE)
predict(fit, type = "mean")
predict(fit, type = "quantile", p = 0.95)
predict(fit, type = "survival", x = 25000)
```

scaling_plot_data

Data underlying the scaling relationship diagnostic

Description

Bins the fitted scaling variable and compares actual mean observed severity with the mean of simulated observed severities generated under the same row- level deductibles and limits. It also reports the fitted ground-up mean at a representative scaling value in each bin. The first two series are directly comparable on the observed/censored basis; the ground-up mean is included to display the model's implied severity-scaling relationship.

Usage

```
scaling_plot_data(
  object,
  bins = 10,
  bin_method = c("quantile", "width"),
  scale_range = NULL,
  nsim = 50,
  seed = NULL,
  output = c("wide", "plot_ready")
)
```

Arguments

object	Model fitted with scale_by.
bins	Number of bins.
bin_method	"quantile" or "width".
scale_range	Optional scaling range.
nsim	Number of simulated portfolios.
seed	Optional random seed.
output	"wide" (default) returns summary columns; "plot_ready" returns the exact long data used by plot_scaling_fit().

Value

A data frame with scale bins. For output = "wide", columns include bin, scale_value, n, actual_observed_mean, modeled_observed_mean, and fitted_ground_up_mean. For output = "plot_ready", columns bin, scale_value, mean, and series give the exact long-format data used by plot_scaling_fit(). Observed and modeled observed means incorporate policy terms; the ground-up mean does not.

Examples

```
set.seed(2)
d <- data.frame(cov_a = rep(c(200000, 400000, 600000), each = 80))
d$loss <- rlnorm(nrow(d), log(.07*d$cov_a), .8)
f <- fit_severity(d, "loss", scale_by = "cov_a", hessian = FALSE)
scaling_plot_data(f, bins = 3, nsim = 1)
```

severity_factors

*Generate actuarial severity rating factors***Description**

Takes either one fitted severity model or a candidate-comparison object to the business endpoint. Outputs expected payments, limited expected severity, LERs, ILFs, and base-relative combined rating factors. If a candidate object is supplied, every successful candidate is evaluated using the same coverage scenarios so factor sensitivity can be compared directly.

Usage

```
severity_factors(
  object,
  deductible,
  limit,
  scale_value = NULL,
  valuation_factor = 1,
  base_deductible = 0,
  base_limit = Inf,
  base_valuation_factor = 1,
  grid = c("cross", "paired")
)
```

Arguments

object	A fitted model or candidate-comparison object.
deductible	Vector of target deductibles.
limit	Vector of target payment limits; Inf is allowed.
scale_value	Optional vector of scaling values.
valuation_factor	Vector of multiplicative valuation factors; 1 means no depreciation adjustment.

base_deductible	Base deductible for rating_factor.
base_limit	Base payment limit for rating_factor and ILF calculations.
base_valuation_factor	Base valuation factor.
grid	"cross" (default) forms the Cartesian product of inputs; "paired" recycles vectors to matching scenario rows.

Details

A valuation_factor below one applies a multiplicative ACV/depreciation adjustment to ground-up replacement-cost severity before the deductible and limit. Fitting remains agnostic to depreciation: historical losses must have already been placed on the analyst's chosen ground-up basis.

Value

A long-format data frame with one row per coverage scenario and successful distribution. Columns include distribution, model, scale_value, deductible, limit, valuation_factor, expected_payment, limited_expected_severity, unlimited_mean, LER, ILF, rating_factor, and the three base_* assumptions. A value can be NA when the relevant expectation or denominator does not exist. With grid = "cross", input vectors form a Cartesian product; with "paired", they form positionally matched scenarios.

What the factor limit means

Here limit is the maximum **payment after applying the deductible**. It is not the ground-up censoring threshold in fit_severity(). rating_factor compares each scenario's expected payment against the supplied base deductible, limit and valuation terms. grid = "cross" can create many scenarios when several input vectors are long.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
severity_factors(fit, deductible = c(500, 1000),
  limit = c(25000, 50000), grid = "paired")
```

severity_quantile	<i>Severity quantiles from a fitted model</i>
-------------------	---

Description

Returns ground-up severity quantiles implied by a fitted single-family or spliced model, including scale-dependent fits when scale_value is supplied.

Usage

```
severity_quantile(object, p, scale_value = NULL)
```

Arguments

object	A fitted actuarialfitdist model.
p	Probability or vector of probabilities.
scale_value	Optional scaling value for a model fitted with scale_by.

Value

Numeric vector of fitted positive ground-up severity quantiles at probabilities p, in the original loss units.

Examples

```
set.seed(123)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", distribution = "lognormal", hessian = FALSE)
severity_quantile(fit, c(.5, .9, .99))
```

```
simulate.actuarialfitdist_fit
```

Simulate from a fitted severity model

Description

Simulates either positive ground-up severities or the observed claims that result after reproducing row-level deductible truncation and limit censoring.

Usage

```
## S3 method for class 'actuarialfitdist_fit'
simulate(
  object,
  nsim = 1,
  seed = NULL,
  type = c("observed", "ground_up"),
  newdata = object$data,
  n = NULL,
  scale_value = NULL,
  ...
)
```

Arguments

object	Fitted model.
nsim	Number of simulated replicates.
seed	Optional random seed.
type	"observed" reproduces deductible truncation and limit censoring using rows of newdata; "ground_up" simulates positive ground-up severity.
newdata	Optional data frame. Defaults to the fitting data.
n	Number of ground-up draws when type = "ground_up". Defaults to the number of rows in newdata.
scale_value	Optional scale values for ground-up simulation.
...	Unused.

Value

A data frame. In type = "observed", rows contain row, simulation, observed_loss, censored, deductible, limit, and scale_value after reproducing each row's observation mechanism. In type = "ground_up", rows contain simulation, ground_up, and scale_value without censoring or truncation.

Examples

```
set.seed(1)
d <- data.frame(loss = rlnorm(120, log(10000), 0.8))
fit <- fit_severity(d, "loss", hessian = FALSE)
head(simulate(fit, nsim = 1, seed = 123))
head(simulate(fit, nsim = 1, seed = 123, type = "ground_up", n = 5))
```

Index

actuarialfitdist_fit_methods, [2](#)
AIC.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
BIC.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
bootstrap_fit, [3](#)
candidate_plot_data, [4](#)
coef.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
compare_factors, [5](#)
confint.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
diagnose_fit, [6](#)
exceedance_probability, [7](#)
expected_payment, [7](#)
fit_plot_data, [8](#)
fit_severity, [9](#)
fit_severity_candidates, [11](#)
fit_spliced_severity, [13](#)
increased_limits_factor, [15](#)
limited_expected_value, [16](#)
logLik.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
loss_elimination_ratio, [17](#)
plot.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
plot_candidate_fits, [18](#)
plot_fit, [19](#)
plot_scaling_fit, [21](#)
predict.actuarialfitdist_fit, [22](#)
print.actuarialfitdist_candidates
 (actuarialfitdist_fit_methods),
 [2](#)
print.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
print.summary.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
scaling_plot_data, [23](#)
severity_factors, [24](#)
severity_quantile, [25](#)
simulate.actuarialfitdist_fit, [26](#)
summary.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)
vcov.actuarialfitdist_fit
 (actuarialfitdist_fit_methods),
 [2](#)