Package 'fable.intermittent'

June 18, 2026

Title Forecasting Models for Intermittent Time Series

Version 0.1.0

Description Extends the 'fable' framework to support forecasting methods specifically designed for intermittent time series data, where demand occurs sporadically with many zero values. All methods produce probabilistic forecasts returned as 'distributional' objects. The returned forecasts can be used to evaluate accuracy, plot and print the results seamlessly with 'fable'. The methods include:
Harvey, Fernandes (1989) <doi:10.1080/07350015.1989.10509750>,
Willemain, Smart, Schwarz (2004) <doi:10.1016/S0169-2070(03)00013-X>,
Zhou, Viswanathan (2011) <doi:10.1016/j.ijpe.2010.09.021>,
Snyder, Ord, Beaumont (2012) <doi:10.1016/j.ijforecast.2011.03.009>,
Kolassa (2016) <doi:10.1016/j.ijforecast.2015.12.004>,
Hasni, Aguir, Babai, Jemai (2019) <doi:10.1080/00207543.2018.1424375>,
Damato, Azzimonti, Corani (2025) <doi:10.1016/j.ijforecast.2025.10.001>,
Sbrana (2025) <doi:10.1080/01605682.2025.2569661>.

License LGPL (>= 3)

Encoding UTF-8

LazyData true

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 4.1.0), fabletools (>= 0.6.0)

Imports tsibble (>= 1.0.0), distributional (>= 0.3.0), nloptr (>= 2.0.0), rlang (>= 1.0.0), Rcpp (>= 1.0.0)

Suggests dplyr, testthat (>= 3.3.0), tweedie, ggtime, ggplot2, tibble, knitr, rmarkdown

VignetteBuilder knitr

URL https://github.com/StefanoDamato/fable.intermittent

BugReports https://github.com/StefanoDamato/fable.intermittent/issues

Config/roxygen2/version 8.0.0

RoxygenNote 7.3.3

NeedsCompilation yes

Author Stefano Damato [aut, cre, cph] (ORCID:
<https://orcid.org/0009-0001-0225-3736>),
Lorenzo Zambon [aut] (ORCID: <https://orcid.org/0000-0002-8939-993X>),
Dario Azzimonti [aut] (ORCID: <https://orcid.org/0000-0001-5080-3061>)

Maintainer Stefano Damato <stefanodamato128@gmail.com>

Repository CRAN

Date/Publication 2026-06-18 10:00:02 UTC

Contents

auto . . . 3
BETANBB . . . 4
dist_tweedie . . . 5
EMPDISTR . . . 6
fitted.BETANBB . . . 7
fitted.EMPDISTR . . . 8
fitted.GAMPOISB . . . 8
fitted.HSPES . . . 9
fitted.MARWAL . . . 10
fitted.NEGBINES . . . 10
fitted.STATICDISTR . . . 11
fitted.TWEES . . . 12
fitted.VZ . . . 12
fitted.WSS . . . 13
forecast.BETANBB . . . 14
forecast.EMPDISTR . . . 14
forecast.GAMPOISB . . . 15
forecast.HSPES . . . 16
forecast.MARWAL . . . 17
forecast.NEGBINES . . . 17
forecast.STATICDISTR . . . 18
forecast.TWEES . . . 19
forecast.VZ . . . 20
forecast.WSS . . . 20
GAMPOISB . . . 21
generate.BETANBB . . . 22
generate.EMPDISTR . . . 23
generate.GAMPOISB . . . 24
generate.HSPES . . . 24
generate.MARWAL . . . 25
generate.NEGBINES . . . 26
generate.STATICDISTR . . . 27
generate.TWEES . . . 27
generate.VZ . . . 28
generate.WSS . . . 29
HSPES . . . 30
MARWAL . . . 31

model_sum.BETANBB 32
model_sum.EMPDISTR 32
model_sum.GAMPOISB 33
model_sum.HSPES 34
model_sum.MARWAL 34
model_sum.NEGBINES 35
model_sum.STATICDISTR 36
model_sum.TWEES 36
model_sum.VZ 37
model_sum.WSS 38
NEGBINES 38
raf 39
residuals.BETANBB 40
residuals.EMPDISTR 41
residuals.GAMPOISB 41
residuals.HSPES 42
residuals.MARWAL 43
residuals.NEGBINES 43
residuals.STATICDISTR 44
residuals.TWEES 45
residuals.VZ 45
residuals.WSS 46
STATICDISTR 47
tweedie 48
TWEES 49
VZ 50
WSS 51

Index **53**

`auto` *Automotive Spare Parts Demand Dataset*

Description

A dataset of monthly demand for automotive spare parts. The data contains 3000 intermittent time series, each spanning 24 monthly periods from January 2010 to December 2011.

Usage

```
auto
```

Format

A tsibble with 72,000 rows and 3 variables:

series_id Character. Unique identifier for each time series.

index Date (yearmonth). The monthly time index.

value Numeric. The demand quantity for the given month.

Source

Turkmen, A. C., Januschowski, T., Wang, Y., & Cemgil, A. T. (2021). Forecasting intermittent and sparse time series: A unified probabilistic framework via deep renewal processes. *PLOS ONE*, 16(11), e0259764.

Available at https://github.com/canerturkmen/gluon-ts/tree/intermittent-datasets/datasets.

Examples

```
library(tsibble)
auto
```

BETANBB *Beta-Negative Binomial Bayesian Dynamic Model*

Description

Conjugate Bayesian dynamic model for count time series with a negative binomial observation distribution and a Beta prior on the success probability parameter. This extends the conjugate updating framework of Harvey & Fernandes (1989) to the Beta-Negative Binomial family. The Beta prior is updated at each time step using a discount factor `w`. Forecasts are available as samples simulating from the model forward in time.

Usage

```
BETANBB(formula, ...)
```

Arguments

`formula`	Model specification.
`...`	Not used.

Value

A model specification.

References

Harvey, A. C., & Fernandes, C. (1989). Time series models for count or qualitative observations. *Journal of Business & Economic Statistics*, 7(4), 407–417. doi:10.1080/07350015.1989.10509750.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(BETANBB(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

`dist_tweedie` *Tweedie Distribution*

Description

Construct a Tweedie distribution object using the compound Poisson–Gamma parameterisation with power parameter in $(1, 2)$. The Tweedie family is a subclass of exponential dispersion models that naturally produces exact zeros (via the Poisson count component) mixed with continuous positive values (via the Gamma severity component), making it well suited to intermittent demand data.

Usage

```
dist_tweedie(mean = 1, dispersion = 1, power = 1.5)
```

Arguments

`mean`	Mean parameter $\mu > 0$.
`dispersion`	Dispersion parameter $\phi > 0$.
`power`	Power parameter $p \in (1, 2)$.

Details

The density is evaluated using the series expansion of Dunn & Smyth (2005), implemented in C++ for performance.

Value

A `distributional` distribution object of class `dist_tweedie`.

References

Dunn, P. K., & Smyth, G. K. (2005). Series evaluation of Tweedie exponential dispersion model densities. *Statistics and Computing*, 15(4), 267–280. doi:10.1007/s112220054070y.

Examples

```
d <- dist_tweedie(mean = 2, dispersion = 0.8, power = 1.5)
d |> mean()
d |> quantile(c(0.5, 0.9))
d |> density(c(0, 1.5, 3))
d |> distributional::variance()
d |> distributional::generate(10)
```

EMPDISTR *Empirical Distribution Resampling*

Description

Naive non-parametric baseline for intermittent demand forecasting. The predictive distribution at every horizon is simply the empirical distribution of the observed values: forecasts are produced by resampling with replacement from the historical series. Point forecasts are the sample mean.

Usage

```
EMPDISTR(formula, hot_start = FALSE, ...)
```

Arguments

`formula`	Model specification.
`hot_start`	Logical. If `TRUE`, leading zeros are removed from the time series before fitting.
`...`	Not used.

Value

A model specification.

References

Hasni, M., Aguir, M. S., Babai, M. Z., & Jemai, Z. (2019). Spare parts demand forecasting: a review on bootstrapping methods. *International Journal of Production Research*, 57(15–16), 4791–4804. doi:10.1080/00207543.2018.1424375.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(EMPDISTR(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

`fitted.BETANBB` *Extract fitted values from a BETANBB model*

Description

Extract fitted values from a BETANBB model

Usage

```
## S3 method for class 'BETANBB'
fitted(object, ...)
```

Arguments

`object`	A model for which fitted values are required.
`...`	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, BETANBB(value))
fitted(fit)
```

fitted.EMPDISTR *Extract fitted values from an EMPDISTR model*

Description

Extract fitted values from an EMPDISTR model

Usage

```
## S3 method for class 'EMPDISTR'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, EMPDISTR(value))
fitted(fit)
```

fitted.GAMPOISB *Extract fitted values from a GAMPOISB model*

Description

Extract fitted values from a GAMPOISB model

Usage

```
## S3 method for class 'GAMPOISB'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, GAMPOISB(value))
fitted(fit)
```

`fitted.HSPES` *Extract fitted values from a HSPES model*

Description

Extract fitted values from a HSPES model

Usage

```
## S3 method for class 'HSPES'
fitted(object, ...)
```

Arguments

`object`	A model for which fitted values are required.
`...`	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, HSPES(value))
fitted(fit)
```

fitted.MARWAL *Extract fitted values from a MARWAL model*

Description

Extract fitted values from a MARWAL model

Usage

```
## S3 method for class 'MARWAL'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, MARWAL(value))
fitted(fit)
```

fitted.NEGBINES *Extract fitted values from a NEGBINES model*

Description

Extract fitted values from a NEGBINES model

Usage

```
## S3 method for class 'NEGBINES'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, NEGBINES(value))
fitted(fit)
```

---

fitted.STATICDISTR *Extract fitted values from a STATICDISTR model*

Description

Extract fitted values from a STATICDISTR model

Usage

```
## S3 method for class 'STATICDISTR'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, STATICDISTR(value))
fitted(fit)
```

fitted.TWEES *Extract fitted values from a TWEES model*

Description

Extract fitted values from a TWEES model

Usage

```
## S3 method for class 'TWEES'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, TWEES(value))
fitted(fit)
```

fitted.VZ *Extract fitted values from a VZ model*

Description

Extract fitted values from a VZ model

Usage

```
## S3 method for class 'VZ'
fitted(object, ...)
```

Arguments

object	A model for which fitted values are required.
...	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, VZ(value))
fitted(fit)
```

`fitted.WSS` *Extract fitted values from a WSS model*

Description

Extract fitted values from a WSS model

Usage

```
## S3 method for class 'WSS'
fitted(object, ...)
```

Arguments

`object`	A model for which fitted values are required.
`...`	Not used.

Value

A numeric vector of fitted values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, WSS(value))
fitted(fit)
```

forecast.BETANBB *Forecast a BETANBB model*

Description

Produces forecast distributions from a fitted BETANBB model using simulation.

Usage

```
## S3 method for class 'BETANBB'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
times	The number of sample paths to use in estimating the forecast distribution.
...	Other arguments passed to methods

Value

A distribution vector of class dist_sample.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, BETANBB(value))
forecast(fit, h = "7 days")
```

forecast.EMPDISTR *Forecast an EMPDISTR model*

Description

Produces forecast distributions by repeating the empirical distribution estimated from the training data at each forecast horizon.

Usage

```
## S3 method for class 'EMPDISTR'
forecast(object, new_data, specials = NULL, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A distribution vector of class dist_sample.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, EMPDISTR(value))
forecast(fit, h = "7 days")
```

---

forecast.GAMPOISB *Forecast a GAMPOISB model*

Description

Produces forecast distributions from a fitted GAMPOISB model using simulation.

Usage

```
## S3 method for class 'GAMPOISB'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
times	The number of sample paths to use in estimating the forecast distribution.
...	Other arguments passed to methods

Value

A distribution vector of forecasts: for h=1 the vector class is dist_negative_binomial; for h>1 the vector class is dist_sample.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, GAMPOISB(value))
forecast(fit, h = "7 days")
```

`forecast.HSPES` *Forecast a HSPES model*

Description

Produces forecast distributions from a fitted HSPES model using simulation.

Usage

```
## S3 method for class 'HSPES'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

`object`	A model for which forecasts are required.
`new_data`	A tsibble containing future index values to forecast.
`specials`	Passed by `fabletools::forecast.mdl_df()`.
`times`	The number of sample paths to use in estimating the forecast distribution.
`...`	Other arguments passed to methods

Value

A distribution vector of forecasts: for h=1 the vector class is `dist_inflated` (hurdle-shifted Poisson); for h>1 the vector class is `dist_sample`.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, HSPES(value))
forecast(fit, h = "7 days")
```

forecast.MARWAL *Forecast a MARWAL model*

Description

Produces forecast distributions from a fitted MARWAL model.

Usage

```
## S3 method for class 'MARWAL'
forecast(object, new_data, specials = NULL, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A distribution vector of class dist_normal.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, MARWAL(value))
forecast(fit, h = "7 days")
```

forecast.NEGBINES *Forecast a NEGBINES model*

Description

Produces forecast distributions from a fitted NEGBINES model using simulation.

Usage

```
## S3 method for class 'NEGBINES'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
times	The number of sample paths to use in estimating the forecast distribution.
...	Other arguments passed to methods

Value

A distribution vector of forecasts: for h=1 the vector class is dist_negative_binomial; for h>1 the vector class is dist_sample.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, NEGBINES(value))
forecast(fit, h = "7 days")
```

---

forecast.STATICDISTR *Forecast a STATICDISTR model*

Description

Produces forecast distributions from a fitted STATICDISTR model.

Usage

```
## S3 method for class 'STATICDISTR'
forecast(object, new_data, specials = NULL, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A distribution vector. The class depends on the static distribution fitted by the STATICDISTR method.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, STATICDISTR(value))
forecast(fit, h = "7 days")
```

`forecast.TWEES` *Forecast a TWEES model*

Description

Produces forecast distributions from a fitted TWEES model using simulation.

Usage

```
## S3 method for class 'TWEES'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

`object`	A model for which forecasts are required.
`new_data`	A tsibble containing future index values to forecast.
`specials`	Passed by `fabletools::forecast.mdl_df()`.
`times`	The number of sample paths to use in estimating the forecast distribution.
`...`	Other arguments passed to methods

Value

A distribution vector of forecasts: for h=1 the vector class is `dist_tweedie`; for h>1 the vector class is `dist_sample`.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, TWEES(value))
forecast(fit, h = "7 days")
```

forecast.VZ *Forecast a VZ model*

Description

Produces forecast distributions from a fitted VZ model using simulation.

Usage

```
## S3 method for class 'VZ'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
times	The number of sample paths to use in estimating the forecast distribution.
...	Other arguments passed to methods

Value

A distribution vector of forecasts of class dist_sample.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, VZ(value))
forecast(fit, h = "7 days")
```

forecast.WSS *Forecast a WSS model*

Description

Produces forecast distributions from a fitted WSS model using simulation.

Usage

```
## S3 method for class 'WSS'
forecast(object, new_data, specials = NULL, times = 10000, ...)
```

Arguments

object	A model for which forecasts are required.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
times	The number of sample paths to use in estimating the forecast distribution.
...	Other arguments passed to methods

Value

A distribution vector of forecasts of class dist_sample.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, WSS(value))
forecast(fit, h = "7 days")
```

GAMPOISB *Gamma-Poisson Bayesian Dynamic Model*

Description

Conjugate Bayesian dynamic model for count time series with a Poisson observation distribution and a Gamma prior on the rate parameter, following Harvey & Fernandes (1989). The Gamma prior is updated at each time step using a discount factor w that controls how quickly past information decays. The first-step forecast follows a Negative Binomial distribution, and multi-step forecasts are obtained by simulating from the model forward in time.

Usage

```
GAMPOISB(formula, ...)
```

Arguments

formula	Model specification.
...	Not used.

Value

A model specification.

References

Harvey, A. C., & Fernandes, C. (1989). Time series models for count or qualitative observations. *Journal of Business & Economic Statistics*, 7(4), 407–417. doi:10.1080/07350015.1989.10509750.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(GAMPOISB(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

`generate.BETANBB` *Generate sample paths from a BETANBB model*

Description

Generate sample paths from a BETANBB model

Usage

```
## S3 method for class 'BETANBB'
generate(x, new_data, specials = NULL, ...)
```

Arguments

`x`	A fitted `BETANBB` model object.
`new_data`	A tsibble containing future index values to forecast.
`specials`	Passed by `fabletools::forecast.mdl_df()`.
`...`	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, BETANBB(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

---

generate.EMPDISTR *Generate sample paths from an EMPDISTR model*

Description

Generate sample paths from an EMPDISTR model

Usage

```
## S3 method for class 'EMPDISTR'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted EMPDISTR model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A new_data tibble with a .sim column of simulated values.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, EMPDISTR(value))
generate(fit, h = 7)
```

generate.GAMPOISB *Generate sample paths from a GAMPOISB model*

Description

Generate sample paths from a GAMPOISB model

Usage

```
## S3 method for class 'GAMPOISB'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted GAMPOISB model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, GAMPOISB(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

generate.HSPES *Generate sample paths from a HSPES model*

Description

Generate sample paths from a HSPES model

Usage

```
## S3 method for class 'HSPES'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted HSPES model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, HSPES(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

---

generate.MARWAL *Generate sample paths from a MARWAL model*

Description

Generate sample paths from a MARWAL model

Usage

```
## S3 method for class 'MARWAL'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted MARWAL model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, MARWAL(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

`generate.NEGBINES` *Generate sample paths from a NEGBINES model*

Description

Generate sample paths from a NEGBINES model

Usage

```
## S3 method for class 'NEGBINES'
generate(x, new_data, specials = NULL, ...)
```

Arguments

`x`	A fitted `NEGBINES` model object.
`new_data`	A tsibble containing future index values to forecast.
`specials`	Passed by `fabletools::forecast.mdl_df()`.
`...`	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, NEGBINES(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

generate.STATICDISTR *Generate sample paths from a STATICDISTR model*

Description

Generate sample paths from a STATICDISTR model

Usage

```
## S3 method for class 'STATICDISTR'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted STATICDISTR model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, STATICDISTR(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

generate.TWEES *Generate sample paths from a TWEES model*

Description

Generate sample paths from a TWEES model

Usage

```
## S3 method for class 'TWEES'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted TWEES model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, TWEES(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

---

generate.VZ *Generate sample paths from a VZ model*

Description

Generate sample paths from a VZ model

Usage

```
## S3 method for class 'VZ'
generate(x, new_data, specials = NULL, ...)
```

Arguments

x	A fitted VZ model object.
new_data	A tsibble containing future index values to forecast.
specials	Passed by fabletools::forecast.mdl_df().
...	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, VZ(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

`generate.WSS` *Generate sample paths from a WSS model*

Description

Generate sample paths from a WSS model

Usage

```
## S3 method for class 'WSS'
generate(x, new_data, specials = NULL, ...)
```

Arguments

`x`	A fitted `WSS` model object.
`new_data`	A tsibble containing future index values to forecast.
`specials`	Passed by `fabletools::forecast.mdl_df()`.
`...`	Other arguments passed to methods

Value

A vector of future paths from a dataset using a fitted model.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, WSS(value))
generate(fit, new_data = tsibble::new_data(ts, 7))
```

HSPES *Hurdle-Shifted Poisson Exponential Smoothing*

Description

Exponential smoothing state space model for intermittent demand with a hurdle-shifted Poisson observation distribution, following the framework of Snyder, Ord & Beaumont (2012). The model decomposes demand into two components: a Bernoulli occurrence process (probability of non-zero demand) and a shifted Poisson demand-size process. Both components are driven by independent (optionally damped) exponential smoothing state equations. The first-step forecast follows an hurdle-shifted Poisson distribution, multi-step forecasts are obtained by simulating from the model forward in time.

Usage

```
HSPES(formula, damped = TRUE, ...)
```

Arguments

`formula`	Model specification.
`damped`	Logical. If `TRUE` (default), both the occurrence and demand smoothing components use a damping parameter.
`...`	Not used.

Value

A model specification.

References

Snyder, R. D., Ord, J. K., & Beaumont, A. (2012). Forecasting the intermittent demand for slow-moving inventories: A modelling approach. *International Journal of Forecasting*, 28(2), 485–496. doi:10.1016/j.ijforecast.2011.03.009.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(HSPES(value)) |>
  forecast(h = "7 days")

fc_ts |> print()
```

```
if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

MARWAL *Markov Chain Model with Random Walk dynamic*

Description

A Random Walk model on the demand composed by a switching constant and an ARMA model, where the changes are controlled by a Markov Chain on the occurrence process. Parameters are estimated in closed-form on deseasonalized data, and forecasts are returned as Gaussian distributions for each time step.

Usage

```
MARWAL(formula, ...)
```

Arguments

`formula`	Model specification.
`...`	Not used.

Value

A model specification.

References

Sbrana, G. (2025). Markov Walk and Walmart sales prediction. *Journal of the Operational Research Society*, 1–12. doi:10.1080/01605682.2025.2569661.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(MARWAL(value)) |>
  forecast(h = "7 days")

fc_ts |> print()
```

```
if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

`model_sum.BETANBB` *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'BETANBB'
model_sum(x)
```

Arguments

`x`	A fitted `fable` model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, BETANBB(value))
model_sum(fit[[1]][[1]])
```

`model_sum.EMPDISTR` *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'EMPDISTR'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, EMPDISTR(value))
model_sum(fit[[1]][[1]])
```

---

model_sum.GAMPOISB *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'GAMPOISB'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, GAMPOISB(value))
model_sum(fit[[1]][[1]])
```

model_sum.HSPES *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'HSPES'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, HSPES(value))
model_sum(fit[[1]][[1]])
```

model_sum.MARWAL *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'MARWAL'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, MARWAL(value))
model_sum(fit[[1]][[1]])
```

---

model_sum.NEGBINES *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'NEGBINES'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, NEGBINES(value))
model_sum(fit[[1]][[1]])
```

model_sum.STATICDISTR *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'STATICDISTR'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, STATICDISTR(value))
model_sum(fit[[1]][[1]])
```

model_sum.TWEES *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'TWEES'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, TWEES(value))
model_sum(fit[[1]][[1]])
```

`model_sum.VZ` *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'VZ'
model_sum(x)
```

Arguments

`x`	A fitted `fable` model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, VZ(value))
model_sum(fit[[1]][[1]])
```

model_sum.WSS *Return model name*

Description

Return model name

Usage

```
## S3 method for class 'WSS'
model_sum(x)
```

Arguments

x	A fitted fable model object.

Value

The model name as a string

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, WSS(value))
model_sum(fit[[1]][[1]])
```

NEGBINES *Negative Binomial Exponential Smoothing*

Description

Exponential smoothing state space model for intermittent demand with a negative binomial observation distribution, as proposed by Snyder, Ord & Beaumont (2012). The conditional mean of the negative binomial is governed by a (optionally damped) exponential smoothing process. The probability parameter is estimated to maximise the likelihood. The first-step forecast follows a Negative Binomial distribution, and multi-step forecasts are obtained by simulating from the model forward in time.

Usage

```
NEGBINES(formula, damped = TRUE, ...)
```

Arguments

formula	Model specification.
damped	Logical. If TRUE (default), the exponential smoothing component uses a damping parameter.
...	Not used.

Value

A model specification.

References

Snyder, R. D., Ord, J. K., & Beaumont, A. (2012). Forecasting the intermittent demand for slow-moving inventories: A modelling approach. *International Journal of Forecasting*, 28(2), 485–496. doi:10.1016/j.ijforecast.2011.03.009.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(NEGBINES(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

---

raf *RAF Spare Parts Demand Dataset*

Description

A dataset of monthly demand for spare parts from the Royal Air Force (RAF). The data contains 5000 intermittent time series, each spanning 84 monthly periods from January 1996 to December 2002. This is a widely used benchmark dataset for intermittent demand forecasting.

Usage

```
raf
```

Format

A tsibble with 420,000 rows and 3 variables:

series_id Character. Unique identifier for each time series.

index Date (yearmonth). The monthly time index.

value Numeric. The demand quantity for the given month.

Source

Syntetos, A. A., & Boylan, J. E. (2005). The accuracy of intermittent demand estimates. *International Journal of Forecasting*, 21(2), 303–314.

Available at https://github.com/canerturkmen/gluon-ts/tree/intermittent-datasets/datasets.

Examples

```
library(tsibble)
raf
```

`residuals.BETANBB` *Extract residuals from a BETANBB model*

Description

Extract residuals from a BETANBB model

Usage

```
## S3 method for class 'BETANBB'
residuals(object, ...)
```

Arguments

`object`	A model for which residuals are required.
`...`	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, BETANBB(value))
residuals(fit)
```

residuals.EMPDISTR *Extract residuals from an EMPDISTR model*

Description

Extract residuals from an EMPDISTR model

Usage

```
## S3 method for class 'EMPDISTR'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, EMPDISTR(value))
residuals(fit)
```

residuals.GAMPOISB *Extract residuals from a GAMPOISB model*

Description

Extract residuals from a GAMPOISB model

Usage

```
## S3 method for class 'GAMPOISB'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, GAMPOISB(value))
residuals(fit)
```

---

residuals.HSPES *Extract residuals from a HSPES model*

Description

Extract residuals from a HSPES model

Usage

```
## S3 method for class 'HSPES'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, HSPES(value))
residuals(fit)
```

residuals.MARWAL *Extract residuals from a MARWAL model*

Description

Extract residuals from a MARWAL model

Usage

```
## S3 method for class 'MARWAL'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, MARWAL(value))
residuals(fit)
```

residuals.NEGBINES *Extract residuals from a NEGBINES model*

Description

Extract residuals from a NEGBINES model

Usage

```
## S3 method for class 'NEGBINES'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, NEGBINES(value))
residuals(fit)
```

---

residuals.STATICDISTR *Extract residuals from a STATICDISTR model*

Description

Extract residuals from a STATICDISTR model

Usage

```
## S3 method for class 'STATICDISTR'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, STATICDISTR(value))
residuals(fit)
```

residuals.TWEES *Extract residuals from a TWEES model*

Description

Extract residuals from a TWEES model

Usage

```
## S3 method for class 'TWEES'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, TWEES(value))
residuals(fit)
```

residuals.VZ *Extract residuals from a VZ model*

Description

Extract residuals from a VZ model

Usage

```
## S3 method for class 'VZ'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, VZ(value))
residuals(fit)
```

---

residuals.WSS *Extract residuals from a WSS model*

Description

Extract residuals from a WSS model

Usage

```
## S3 method for class 'WSS'
residuals(object, ...)
```

Arguments

object	A model for which residuals are required.
...	Not used.

Value

A numeric vector of residuals.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
fit <- model(ts, WSS(value))
residuals(fit)
```

`STATICDISTR` *Static Count Distribution Model*

Description

Static (IID) count distribution model for intermittent demand, following Kolassa (2016). The method fits several candidate distributions — Poisson, hurdle-shifted Poisson, negative binomial, and hurdle-shifted negative binomial — to the observed series and selects the best by AIC. A mixture option that blends all four predictive distributions is also available.

Usage

```
STATICDISTR(
  formula,
  distr = c("auto", "pois", "hsp", "nbinom", "hsnb", "mixture"),
  hot_start = FALSE,
  criterion = c("aic", "bic"),
  ...
)
```

Arguments

`formula`	Model specification.
`distr`	Distribution choice: one of `"auto"`, `"pois"`, `"hsp"`, `"nbinom"`, `"hsnb"`, or `"mixture"`.
`hot_start`	Logical. If `TRUE`, leading zeros are removed from the time series before fitting.
`criterion`	Information criterion to use for model selection when `distr = "auto"`. One of `"aic"` or `"bic"`.
`...`	Not used.

Value

A model specification.

References

Kolassa, S. (2016). Evaluating predictive count data distributions in retail sales forecasting. *International Journal of Forecasting*, 32(3), 788–803. doi:10.1016/j.ijforecast.2015.12.004.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)
```

```
fc_ts <- ts |>
  model(STATICDISTR(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

`tweedie` *Tweedie Distribution Functions*

Description

Density, distribution function, quantile function and random generation for the Tweedie distribution with mean equal to `mean`, dispersion equal to `dispersion`, and power equal to `power`.

Usage

```
rtweedie(n, mean = 1, dispersion = 1, power = 1.5)

dtweedie(x, mean = 1, dispersion = 1, power = 1.5, log = FALSE)

ptweedie(
  q,
  mean = 1,
  dispersion = 1,
  power = 1.5,
  lower.tail = TRUE,
  log.p = FALSE
)

qtweedie(
  p,
  mean = 1,
  dispersion = 1,
  power = 1.5,
  lower.tail = TRUE,
  log.p = FALSE
)
```

Arguments

`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.

mean	vector of means.
dispersion	vector of dispersion parameters.
power	vector of power parameters.
x, q	vector of quantiles.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$; otherwise, $P[X > x]$.
p	vector of probabilities.

Details

If mean, dispersion, or power are not specified they assume the default values of 1, 1, and 1.5, respectively.

The Tweedie distribution used here follows the compound Poisson-Gamma parameterisation with power parameter in $(1, 2)$. It has $\mathbb{E}[X] = \mu$ and $\mathrm{Var}(X) = \phi\mu^p$, where μ is mean, ϕ is dispersion, and p is power.

Value

dtweedie gives the density, ptweedie gives the distribution function, qtweedie gives the quantile function, and rtweedie generates random samples.

The length of the result is determined by n for rtweedie, and is the maximum of the lengths of the numerical arguments for the other functions.

The numerical arguments other than n are recycled to the length of the result. Only the first elements of the logical arguments are used.

References

Dunn, P. K., & Smyth, G. K. (2005). Series evaluation of Tweedie exponential dispersion model densities. *Statistics and Computing*, 15(4), 267–280. doi:10.1007/s112220054070y.

TWEES *Tweedie Exponential Smoothing*

Description

Exponential smoothing state space model for intermittent demand with a Tweedie observation distribution. The conditional mean of the Tweedie is governed by a (optionally damped) exponential smoothing process. The power and dispersion parameter are estimated to maximise the likelihood. The Tweedie family naturally models both zeros and large spikes via its compound Poisson-Gamma nature. The first-step forecast follows a Tweedie distribution, and multi-step forecasts are obtained by simulating from the model forward in time. The model parameters are estimated by

Usage

```
TWEES(formula, damped = TRUE, scaling = TRUE, ...)
```

Arguments

formula	Model specification.
damped	Logical. If TRUE (default), the exponential smoothing component uses a damping parameter.
scaling	Logical. If TRUE (default), the time series is divided by its maximum value before fitting and predictions are back-transformed. This improves numerical stability.
...	Not used.

Value

A model specification.

References

Damato, S., Azzimonti, D., & Corani, G. (2025). Forecasting intermittent time series with Gaussian Processes and Tweedie likelihood. *International Journal of Forecasting* (in press). doi:10.1016/j.ijforecast.2025.10.001.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(TWEES(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

VZ *Viswanathan–Zhou Bootstrap Method*

Description

Non-parametric bootstrap forecasting method for intermittent demand proposed by Zhou & Viswanathan (2011). The time series is decomposed into demand sizes and inter-demand intervals via Croston's decomposition. Forecasts are generated by independently resampling from the empirical distributions of both components, preserving the marginal structure of the original series.

Usage

```
VZ(formula, ...)
```

Arguments

formula	Model specification.
...	Not used.

Value

A model specification.

References

Zhou, Chenxi, Viswanathan, S., (2011). Comparison of a new bootstrapping method with parametric approaches for safety stock determination in service parts inventory systems. *International Journal of Production Economics* 133(1) 481–485. doi:10.1016/j.ijpe.2010.09.021

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(VZ(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

WSS *Willemain–Smart–Schwarz Bootstrap Method*

Description

Markov-chain bootstrap forecasting method for intermittent demand proposed by Willemain, Smart & Schwarz (2004). Occurrence of non-zero demand is modelled with a two-state Markov chain whose transition probabilities are estimated from the observed binary occurrence sequence. Demand sizes are resampled from past non-zero observations with Gaussian jittering to smooth the empirical distribution.

Usage

```
WSS(formula, ...)
```

Arguments

formula	Model specification.
...	Not used.

Value

A model specification.

References

Willemain, T. R., Smart, C. N., & Schwarz, H. F. (2004). A new approach to forecasting intermittent demand for service parts inventories. *International Journal of Forecasting*, 20(3), 375–387. doi:10.1016/S01692070(03)00013X.

Examples

```
ts <- tsibble::tsibble(
  time = as.Date("2026-01-01") + seq_len(40),
  value = rnbinom(40, size = 1, prob = 0.3),
  index = time
)

fc_ts <- ts |>
  model(WSS(value)) |>
  forecast(h = "7 days")

fc_ts |> print()


if (requireNamespace("ggtime", quietly = TRUE)) {
  library(ggtime)
  fc_ts |> autoplot(ts)
}
```

Index

∗ **datasets**
    auto, 3
    raf, 39

auto, 3

BETANBB, 4

dist_tweedie, 5
dtweedie *(*tweedie*)*, 48

EMPDISTR, 6

fabletools::forecast.mdl_df(), *14–29*
fitted.BETANBB, 7
fitted.EMPDISTR, 8
fitted.GAMPOISB, 8
fitted.HSPES, 9
fitted.MARWAL, 10
fitted.NEGBINES, 10
fitted.STATICDISTR, 11
fitted.TWEES, 12
fitted.VZ, 12
fitted.WSS, 13
forecast.BETANBB, 14
forecast.EMPDISTR, 14
forecast.GAMPOISB, 15
forecast.HSPES, 16
forecast.MARWAL, 17
forecast.NEGBINES, 17
forecast.STATICDISTR, 18
forecast.TWEES, 19
forecast.VZ, 20
forecast.WSS, 20

GAMPOISB, 21
generate.BETANBB, 22
generate.EMPDISTR, 23
generate.GAMPOISB, 24
generate.HSPES, 24
generate.MARWAL, 25
generate.NEGBINES, 26
generate.STATICDISTR, 27
generate.TWEES, 27
generate.VZ, 28
generate.WSS, 29

HSPES, 30

MARWAL, 31
model_sum.BETANBB, 32
model_sum.EMPDISTR, 32
model_sum.GAMPOISB, 33
model_sum.HSPES, 34
model_sum.MARWAL, 34
model_sum.NEGBINES, 35
model_sum.STATICDISTR, 36
model_sum.TWEES, 36
model_sum.VZ, 37
model_sum.WSS, 38

NEGBINES, 38

ptweedie *(*tweedie*)*, 48

qtweedie *(*tweedie*)*, 48

raf, 39
residuals.BETANBB, 40
residuals.EMPDISTR, 41
residuals.GAMPOISB, 41
residuals.HSPES, 42
residuals.MARWAL, 43
residuals.NEGBINES, 43
residuals.STATICDISTR, 44
residuals.TWEES, 45
residuals.VZ, 45
residuals.WSS, 46
rtweedie *(*tweedie*)*, 48

STATICDISTR, 47

tweedie, 48
TWEES, 49

VZ, 50

WSS, 51