# Package: fasster (via r-universe)

May 30, 2026

**Version** 0.2.0.9000

**Title** Fast Additive Switching of Seasonality, Trend, and Exogenous Regressors

**Description** Implementation of the FASSTER (Forecasting with Additive Switching of Seasonality, Trend, and Exogenous Regressors) model for forecasting time series with multiple seasonal patterns. The model combines state space methodology with a switching component in the observation equation to allow flexible modeling of complex seasonal patterns, including time-varying effects and multiple seasonalities.

**Depends** R (>= 4.1.0), fabletools (>= 0.2.0)

**Imports** dlm, tsibble (>= 0.9.0), purrr, rlang, stats, dplyr (>= 1.0.0), distributional, vctrs

**Suggests** tsibbledata (>= 0.2.0), lubridate, knitr, rmarkdown, testthat, spelling, covr

**URL** https://github.com/tidyverts/fasster, https://fasster.tidyverts.org/

**BugReports** https://github.com/tidyverts/fasster/issues

**License** GPL-3

**Encoding** UTF-8

**ByteCompile** true

**RoxygenNote** 7.3.3

**Roxygen** list(markdown = TRUE, roclets=c('rd', 'collate', 'namespace'))

**Language** en-GB

**VignetteBuilder** knitr

**Config/pak/sysreqs** libicu-dev

**Repository** https://tidyverts.r-universe.dev

**Date/Publication** 2026-02-02 22:25:09 UTC

**RemoteUrl** https://github.com/tidyverts/fasster

**RemoteRef** HEAD

**RemoteSha** f6b6b44916c2e20005cc834a08837d0ba7a88c9e

# Contents



---

`components.FASSTER` *Extract Components from a FASSTER Model*

---

## Description

Decomposes a FASSTER model into its individual components, allowing you to examine the contribution of each term to the fitted values. This is useful for understanding which components drive the model's predictions and how well different aspects of the model fit the data.

## Usage

```
## S3 method for class 'FASSTER'
components(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | A FASSTER model object. |
| `...` | Additional arguments (currently unused). |

## Value

A dable (decomposition table) containing the response variable and each model component as separate columns. The components sum to the response variable.

## Examples

```
if (requireNamespace("tsibbledata", quietly = TRUE)) {
# Fit a FASSTER model and extract components
library(tsibble)
library(dplyr)
fit <- tsibbledata::aus_retail |>
  filter(
    State == "Victoria",
```

```
    Industry == "Cafes, restaurants and catering services"
  ) |>
  model(fasster = FASSTER(Turnover ~ trend(1) + season("year")))

# Extract and view components
components(fit)
}
```

---

FASSTER *Fast Additive Switching of Seasonality, Trend and Exogenous Regressors*

---

## Description

Implements FASSTER

## Usage

```
FASSTER(formula, include = NULL, ...)
```

## Arguments

| | |
|---|---|
| formula | An object of class "formula" (refer to 'Specials' section for usage) |
| include | How many terms should be included to fit the model |
| ... | Not used |

## Details

The fasster model extends commonly used state space models by introducing a switching component to the measurement equation. This is implemented using a time-varying DLM with the switching behaviour encoded in the measurement matrix.

## Value

Returns a mable containing the fitted FASSTER model.

## Specials

The *specials* define the model structure for FASSTER. The model can include trend, seasonal, ARMA, and exogenous regressor components, with optional switching behaviour controlled by grouping factors.

**trend:** The trend special specifies polynomial trend components.

```
trend(n, ...)
```

n The order of the polynomial trend. Use 1 for level, 2 for linear trend, etc.
... Additional arguments passed to dlm::dlmModPoly(). Common arguments include dV (observation variance) and dW (st

**season:** The season special specifies seasonal factors using indicator variables.

```
season(period = NULL, ...)
```

period The seasonal period. If NULL, automatically detected from the data (uses the smallest frequency). Can be a number o
... Additional arguments passed to dlm::dlmModSeas(). Common arguments include dV and dW.

**fourier:** The fourier special specifies seasonal components using Fourier terms (trigonometric functions).

```
fourier(period = NULL, K = floor(period/2), ...)
```

period The seasonal period. If NULL, automatically detected from the data.
K The number of Fourier terms (harmonics) to include. Maximum is floor(period/2). More harmonics capture mor
... Additional arguments passed to dlm::dlmModTrig().

**ARMA:** The ARMA special includes autoregressive moving average components.

```
ARMA(...)
```

... Arguments passed to dlm::dlmModARMA(). Typically includes ar (vector of AR coefficients) and ma (vector of MA coef

**xreg:** The xreg special includes exogenous regressors in the model.

```
xreg(...)
```

... Bare expressions for the exogenous regressors. These are evaluated in the context of the data, so you can use transforma

**custom:** The custom special allows you to specify custom DLM structures.

```
custom(...)
```

... Arguments passed to dlm::dlm() to create a custom DLM component.

%S% **(Switching operator):** The %S% operator creates switching models where different model structures apply to different groups defined by a factor variable.

```
group
```

group A factor variable (or expression that evaluates to a factor) defining the groups. Each level of the factor will have its ow
spec The model specifications to replicate for each group. This can be any combination of `trend()`, `season()`, `fourier()`

For example, `group %S% trend(1)` creates a separate level for each value of `group`, allowing the mean to switch between groups.

`%?%` **(Conditional operator):** The `%?%` operator creates conditional models where a component is only active when a logical condition is TRUE.

`condition`

condition A logical variable or expression. The model component will only contribute when this is TRUE.
spec The model component to conditionally include.

For example, `(year > 2000) %?% trend(1)` includes a level component only for observations after 2000.

## Heuristic

The model parameters are estimated using the following heuristic:

1. Filter the data using the specified model with non-zero state variances
2. Obtain smoothed states $(\theta^{(s)}t = \theta_t|D_T)$ to approximate correct behaviour
3. The initial state parameters taken from the first smoothed state: $m_0 = E(\theta_0^{(s)})$, $C_0 = Var(\theta_0^{(s)})$
4. Obtain state noise variances from the smoothed variance of $w_t$: $W = Var(w_t^{(s)}) = Var(\theta_t^{(s)} - G\theta_{t-1}^{(s)})$ Obtain measurement noise variance from smoothed variance of $v_t$: $V = Var(v_t^{(s)}) = Var(y_t - F_t\theta_t^{(s)})$
5. Repair restricted state variances for seasonal factors and ARMA terms

## Examples

```
# Basic model with trend and seasonality
cbind(mdeaths, fdeaths) |>
  as_tsibble(pivot_longer = FALSE) |>
  model(FASSTER(mdeaths ~ trend(1) + fourier(12)))

# Model with exogenous regressor
cbind(mdeaths, fdeaths) |>
  as_tsibble(pivot_longer = FALSE) |>
  model(FASSTER(mdeaths ~ fdeaths + trend(1) + fourier(12)))

# Switching model with different trends for different periods
cbind(mdeaths, fdeaths) |>
  as_tsibble(pivot_longer = FALSE) |>
  model(
```

```
  FASSTER(mdeaths ~ (lubridate::year(index) > 1977) %S% trend(1) + fourier(12))
)
```

---

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

---

## Description

Returns the one-step-ahead fitted values from a FASSTER model, calculated during the Kalman filtering process. These represent the model's predictions at each time point using only information available up to that point.

## Usage

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

## Arguments

| | |
|---|---|
| `object` | A fitted FASSTER model object. |
| `...` | Additional arguments (currently unused). |

## Value

A numeric vector of fitted values with the same length as the training data.

## Examples

```
library(tsibble)
fit <- as_tsibble(mdeaths) |>
  model(FASSTER(value ~ trend(1) + fourier(12)))

# Extract fitted values
fitted(fit)
```

`forecast.FASSTER` *Forecast a FASSTER model*

## Description

Produces forecasts from a trained FASSTER model using the Kalman filter framework. This method generates point forecasts and prediction intervals by propagating the state space forward through time.

## Usage

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

## Arguments

| | |
|---|---|
| `object` | A trained FASSTER model object. |
| `new_data` | A tsibble containing future time points to forecast. Must be regularly spaced and contain any required exogenous regressors. |
| `specials` | A list of special formulations generated from the model formula, used to construct the design matrix for time-varying components. |
| `...` | Additional arguments (currently unused). |

## Details

The forecast method implements a Kalman filter to propagate the state space model forward in time:

1. For each forecast horizon, constructs time-varying system matrices (FF, GG, V, W) using exogenous variables from `new_data`
2. Computes the state forecast: $a_{t+1} = G_t a_t$
3. Computes the state covariance: $R_{t+1} = G_t R_t G_t' + W_t$
4. Computes the observation forecast: $f_t = F_t a_{t+1}$
5. Computes the forecast variance: $Q_t = F_t R_{t+1} F_t' + V_t$

The method handles switching components by matching exogenous variables in `new_data` with the model's design matrix, adding zero columns for any missing levels.

## Value

A distribution vector (from the distributional package) containing normal distributions with forecasted means and standard errors for each future time point. This integrates with fable's forecast distribution structure.

`glance.FASSTER` *Glance at a FASSTER model*

## Description

Constructs a single-row summary of the model's goodness-of-fit statistics. This method follows the broom package conventions and is used by fabletools to provide model selection metrics.

## Usage

```
## S3 method for class 'FASSTER'
glance(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | A FASSTER model object. |
| `...` | Additional arguments (currently unused). |

## Value

A one-row tibble containing:

**sigma2** The estimated observation variance (V). If the model has multivariate observations, this is a list containing the variance matrix.

**log_lik** The log-likelihood of the model.

**AIC** Akaike Information Criterion.

**AICc** Corrected AIC for small sample sizes.

**BIC** Bayesian Information Criterion.

## Examples

```
library(tsibble)
fit <- as_tsibble(mdeaths) |>
  model(FASSTER(value ~ trend(1) + fourier(12)))

# Get model fit statistics
glance(fit)
```

`interpolate.FASSTER` *Interpolate missing values in a FASSTER model*

## Description

Fills in missing values in the response variable using the model's fitted values. This method only works for interpolating data used to estimate the model and cannot be used for new data.

## Usage

```
## S3 method for class 'FASSTER'
interpolate(object, new_data, specials, ...)
```

## Arguments

| | |
|---|---|
| `object` | A fitted FASSTER model object. |
| `new_data` | A tsibble containing the data to interpolate. Must be the same data used to fit the model. |
| `specials` | A list of special terms (passed by fabletools). |
| `...` | Additional arguments (currently unused). |

## Details

This method identifies missing values (NAs) in the response variable and replaces them with the corresponding fitted values from the model. It only works when `new_data` has the same length as the data used to fit the model.

## Value

A tsibble with missing values in the response variable replaced by fitted values.

## Examples

```
library(tsibble)
library(dplyr)

# Create data with missing values
deaths_na <- as_tsibble(mdeaths) |>
  mutate(value = if_else(row_number() %in% c(10, 20, 30), NA_real_, value))

# Fit model
fit <- deaths_na |>
  model(FASSTER(value ~ trend(1) + fourier(12)))

# Interpolate missing values
interpolate(fit, deaths_na)
```

`refit.FASSTER` *Refit a FASSTER model*

## Description

Applies a fitted FASSTER model to a new dataset.

## Usage

```
## S3 method for class 'FASSTER'
refit(object, new_data, specials = NULL, reestimate = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `object` | A fitted FASSTER model. |
| `new_data` | A tsibble containing the new data. |
| `specials` | (passed by `fabletools::refit.mdl_df()`). |
| `reestimate` | If `TRUE`, the model parameters will be re-estimated to suit the new data using the heuristic approach. If `FALSE`, the existing model structure and parameters are applied to the new data without modification. |
| `...` | Additional arguments passed to the training function. |

## Value

A refitted FASSTER model.

## Examples

```
library(tsibble)

# Fit model to male deaths
fit_male <- as_tsibble(mdeaths) |>
  model(FASSTER(value ~ trend(1) + fourier(12)))

# Refit to female deaths without re-estimating parameters
refit(fit_male, as_tsibble(fdeaths), reestimate = FALSE)

# Refit to female deaths with re-estimated parameters
refit(fit_male, as_tsibble(fdeaths), reestimate = TRUE)
```

`report.FASSTER` *Report on a FASSTER model*

## Description

Prints a detailed report of the estimated variance parameters for a FASSTER model. This includes the state noise variances (W) for each model component and the observation noise variance (V).

## Usage

```
## S3 method for class 'FASSTER'
report(object, ...)
```

## Arguments

| | |
|---|---|
| `object` | A FASSTER model object. |
| `...` | Additional arguments (currently unused). |

## Details

The report displays:

- State noise variances (W): The variance of the random innovations for each state component, grouped by model term.
- Observation noise variance (V): The variance of the measurement error.

## Value

Invisibly returns NULL. Called for its side effect of printing the variance report to the console.

## Examples

```
library(tsibble)
fit <- as_tsibble(mdeaths) |>
  model(FASSTER(value ~ trend(1) + fourier(12)))

# Print variance report
report(fit)
```

## residuals.FASSTER *Extract residuals from a FASSTER model*

### Description

Returns the one-step-ahead forecast errors (residuals) from a FASSTER model.

### Usage

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

### Arguments

| | |
|---|---|
| object | A fitted FASSTER model object. |
| ... | Additional arguments (currently unused). |

### Value

A numeric vector of residuals with the same length as the training data.

### Examples

```
library(tsibble)
fit <- as_tsibble(mdeaths) |>
  model(FASSTER(value ~ trend(1) + fourier(12)))

# Extract residuals
residuals(fit)
```

## stream.FASSTER *Stream new data through a FASSTER model*

### Description

Extends a fitted FASSTER model by filtering new observations through the existing state space model. The model's states and parameters are updated sequentially as new data arrives, allowing for online learning without refitting from scratch.

### Usage

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

## Arguments

| | |
|---|---|
| `object` | A fitted FASSTER model object |
| `new_data` | A tsibble containing new observations to stream through the model |
| `specials` | A list of special terms (switching variables, etc.) parsed from the model formula |
| `...` | Additional arguments (currently unused) |

## Details

The streaming process:

1. Constructs the design matrix from new data
2. Applies the Kalman filter to sequentially update states
3. Updates model variance based on all residuals
4. Prepares the model for subsequent forecasting or streaming

## Value

An updated FASSTER model object with:

- Extended state estimates incorporating the new data
- Updated model variance
- Appended fitted values and residuals
- Updated DLM components for future forecasting

## Examples

```
library(tsibble)
library(fasster)

# Fit initial model on training data
fit <- as_tsibble(head(USAccDeaths, -12)) |>
  model(fasster = FASSTER(value ~ trend() + season("year")))
tidy(fit)
tail(fitted(fit), 20)

# Stream new data through the model
fit_updated <- fit |>
  stream(as_tsibble(tail(USAccDeaths, 12)))
tidy(fit_updated)
tail(fitted(fit_updated), 20)
```

`tidy.FASSTER` *Extract coefficients from a FASSTER model*

## Description

Obtains the mean and variance of the estimated initial states from a FASSTER model. Values in the `estimate` column are contains the mean, and the `std.error` column contains the standard deviation of the initial states.

## Usage

```
## S3 method for class 'FASSTER'
tidy(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | An object containing a FASSTER model. |
| `...` | Unused. |

## Value

A tibble with three columns:

**term** The name of each state variable in the model.

**estimate** The mean of the estimated initial state for each term.

**std.error** The standard deviation of the estimated initial state for each term.

# Index