Package: EMDSSAForecast (via r-universe)

October 15, 2026

Type Package

Title Hybrid EMD-SSA Forecasting with Artificial Neural Networks for Time Series

Version 0.1.0

Description Implements a hybrid empirical mode decomposition, singular spectrum analysis, and artificial neural network forecasting framework for time series data. The framework extracts the trend from the first intrinsic mode function using singular spectrum analysis, forecasts the extracted trend and remaining decomposition components using artificial neural networks, and combines the component forecasts to obtain the final forecast. For methodological details, refer to Kumar et al. (2026) <doi:10.1007/s10791-026-10076-2>.

License MIT + file LICENSE

Encoding UTF-8

Imports Rlibeemd, Rssa, neuralnet

Suggests spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

RoxygenNote 7.3.3

Language en-US

NeedsCompilation no

Author Prabhat Kumar [aut, cre], Girish Kumar Jha [aut], Rajeev Ranjan Kumar [aut], Santosha Rathod [aut], R N Singh [aut]

Maintainer Prabhat Kumar <prabhatkkv@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-15 17:00:24 UTC

RemoteUrl https://github.com/cran/EMDSSAForecast

RemoteRef HEAD

RemoteSha 3aded3aa5af7bab40fef9b89621d92eec18e4d5a

Contents

accuracy_metrics 2
annTSforecast 3
emd_decompose 4
emd_ssa_annTSforecast 5
plot.emd_ssa_annTSforecast 7
plot_imfs 8
plot_ssa_trend 9
reconstruct_forecast 10
ssa_trend 11
tune_emd_ssa_annTSforecast 13

Index **15**

`accuracy_metrics` *Forecast Accuracy Metrics*

Description

Calculate common forecast accuracy metrics including RMSE, MAE, MAPE, and R-squared.

Usage

```
accuracy_metrics(actual, forecast)
```

Arguments

`actual`	Numeric vector containing the observed values.
`forecast`	Numeric vector containing the forecasted values.

Details

Missing values are removed pairwise before calculating the metrics. Observations with an actual value equal to zero are excluded from the MAPE calculation to avoid division by zero. If all actual values are zero, MAPE is returned as `NA`.

R-squared is calculated as

$$R^2 = 1 - \frac{\sum(y_t - \hat{y}_t)^2}{\sum(y_t - \bar{y})^2}.$$

Value

A one-row data frame containing:

RMSE Root Mean Squared Error.

MAE Mean Absolute Error.

MAPE Mean Absolute Percentage Error in percent.

R_squared Coefficient of determination (R-squared).

Examples

```
# Example using the built-in AirPassengers dataset
actual <- as.numeric(AirPassengers)

# Illustrative forecasts
forecast <- actual + 1

accuracy_metrics(
  actual = actual,
  forecast = forecast
)
```

---

annTSforecast *ANN-Based Time-Series Forecast*

Description

Fit a feed-forward artificial neural network to lagged time-series observations and produce recursive multi-step forecasts.

Usage

```
annTSforecast(
  series,
  h = 1,
  lag = 5,
  hidden = c(70, 20),
  maxit = 2500,
  seed = 123
)
```

Arguments

`series`	Numeric univariate training time series.
`h`	Forecast horizon. Default is 1.
`lag`	Number of lagged observations used as ANN inputs. Default is 5.
`hidden`	Hidden-layer sizes. The default configuration is `c(70, 20)`.
`maxit`	Maximum number of neural-network training steps. Default is 2500.
`seed`	Random seed for reproducible model fitting. Default is 123.

Details

The function converts the univariate time series into a supervised learning dataset using lagged observations. The input variables and response are scaled to the interval from 0 to 1 before fitting the artificial neural network. The fitted network is then used to generate recursive multi-step forecasts, where each predicted value is appended to the historical series and used as an input for subsequent forecasts.

Value

A list containing:

fitted In-sample fitted values. The first `lag` observations are returned as `NA`.

forecast Recursive multi-step forecasts.

model Fitted neural network object.

lag Number of lagged observations used as inputs.

hidden Hidden-layer configuration.

maxit Maximum number of training steps.

seed Random seed used for model fitting.

scale_min Minimum value used for min-max scaling.

scale_max Maximum value used for min-max scaling.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

fit <- annTSforecast(
  series = series,
  h = 3,
  lag = 5,
  hidden = c(10, 5),
  maxit = 500,
  seed = 123
)

fit$forecast
```

`emd_decompose` *Empirical Mode Decomposition*

Description

Decompose a univariate time series into intrinsic mode functions (IMFs) and a residual component using `Rlibeemd::emd()`.

Usage

```
emd_decompose(train, S_number = 4, num_siftings = 50)
```

Arguments

`train`	Numeric univariate time series.
`S_number`	S-number stopping criterion used by `Rlibeemd::emd()`. Default is 4.
`num_siftings`	Maximum number of sifting iterations. Default is 50.

Details

Empirical Mode Decomposition (EMD) decomposes a time series into a set of intrinsic mode functions and a residual component. The resulting components can subsequently be used for further analysis or hybrid forecasting.

Value

A numeric matrix whose columns contain IMF1, IMF2, ..., and Residual. Each row corresponds to an observation in the original time series.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

# Perform EMD decomposition
result <- emd_decompose(
  train = series,
  S_number = 4,
  num_siftings = 50
)

# View the dimensions and component names
dim(result)
colnames(result)
```

---

emd_ssa_annTSforecast *Complete Hybrid EMD-SSA-ANN Forecast*

Description

Run the complete hybrid forecasting workflow: EMD -> SSA trend extraction from IMF1 -> ANN component forecasting -> ensemble reconstruction.

Usage

```
emd_ssa_annTSforecast(
  train,
  h = 1,
  L = 12,
  trend_rate = 0.9,
  lag = 5,
  hidden = c(70, 20),
  maxit = 2500,
  S_number = 4,
  num_siftings = 50,
  seed = 123
)
```

Arguments

`train`	Numeric univariate training time series.
`h`	Forecast horizon. Default is 1.
`L`	SSA window length. Default is 12.
`trend_rate`	Proportion of SSA variance used for trend extraction. Default is 0.90.
`lag`	Number of lagged observations used as ANN inputs. Default is 5.
`hidden`	Hidden-layer sizes for the ANN. Default is `c(70, 20)`.
`maxit`	Maximum number of neural-network training steps. Default is 2500.
`S_number`	S-number stopping criterion for EMD. Default is 4.
`num_siftings`	Maximum number of EMD sifting iterations. Default is 50.
`seed`	Random seed for reproducible ANN fitting. Default is 123.

Details

The hybrid forecasting framework consists of four main stages. First, Empirical Mode Decomposition (EMD) decomposes the training series into intrinsic mode functions and a residual component. Second, Singular Spectrum Analysis (SSA) is applied to IMF1 to extract its trend component. Third, the reconstructed components are forecast using an artificial neural network based on lagged observations. Finally, the individual component forecasts are aggregated to obtain the overall forecast.

Value

An object of class `emd_ssa_annTSforecast` containing:

call The matched function call.

train The training time series.

h Forecast horizon.

imfs EMD components including the residual.

ssa SSA trend extraction results.

trend SSA-extracted trend component.

fitted Overall in-sample fitted values.

forecast Overall forecasts.

component_fitted Fitted values for individual components.

component_forecasts Forecasts for individual components.

parameters List containing all model parameters.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

fit <- emd_ssa_annTSforecast(
  train = series,
  h = 5,
```

```
  L = 12,
  trend_rate = 0.90,
  lag = 5,
  hidden = c(10, 5),
  maxit = 500,
  seed = 123
)

# Display the forecasts
fit$forecast

# Display the model summary
fit

# Plot the training series and forecasts
plot(fit)
```

`plot.emd_ssa_annTSforecast`

Plot Hybrid EMD-SSA-ANN Forecast

Description

Plot the training series together with forecasts obtained from the hybrid EMD-SSA-ANN forecasting model.

Usage

```
## S3 method for class 'emd_ssa_annTSforecast'
plot(x, ...)
```

Arguments

`x`	An object of class `emd_ssa_annTSforecast`.
`...`	Additional graphical arguments.

Value

Invisibly returns the supplied model object.

Examples

```
series <- as.numeric(AirPassengers)

fit <- emd_ssa_annTSforecast(
  train = series,
  h = 5,
```

```
  L = 12,
  trend_rate = 0.90,
  lag = 5,
  hidden = c(10, 5),
  maxit = 500,
  seed = 123
)

plot(fit)
```

`plot_imfs` *Plot EMD Components*

Description

Plot the intrinsic mode functions (IMFs) and residual obtained from empirical mode decomposition.

Usage

```
plot_imfs(imfs, main = "EMD Decomposition")
```

Arguments

`imfs`	Numeric matrix containing EMD components. Columns represent individual IMFs and the residual.
`main`	Main title of the plot. Default is "EMD Decomposition".

Details

Each EMD component is displayed in a separate panel. Different colours are used to distinguish the components visually.

Value

The function produces a multi-panel plot and invisibly returns NULL.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

imfs <- emd_decompose(
  train = series,
  S_number = 4,
  num_siftings = 50
)

plot_imfs(imfs)
```

plot_ssa_trend *Plot SSA-Extracted Trend from an IMF*

Description

Plot the original IMF together with the trend extracted using singular spectrum analysis (SSA).

Usage

```
plot_ssa_trend(imf, trend, main = "SSA Trend Extraction from IMF1")
```

Arguments

imf	Numeric vector containing the original IMF used for SSA trend extraction.
trend	Numeric vector containing the SSA-extracted trend.
main	Main title of the plot. Default is "SSA Trend Extraction from IMF1".

Details

The original IMF is displayed in grey and the SSA-extracted trend is displayed in red. This visualization allows the extracted trend to be compared directly with the corresponding IMF.

Value

The function produces a plot and invisibly returns NULL.

Examples

```
series <- as.numeric(AirPassengers)

imfs <- emd_decompose(
  train = series,
  S_number = 4,
  num_siftings = 50
)

result <- ssa_trend(
  imf = imfs[, "IMF1"],
  L = 12,
  trend_rate = 0.90
)

plot_ssa_trend(
  imf = imfs[, "IMF1"],
  trend = result$trend
)
```

reconstruct_forecast *Forecast and Reconstruct EMD-SSA Components*

Description

Forecast the SSA-extracted IMF1 trend and each remaining EMD component using artificial neural networks, then combine their forecasts to obtain the final hybrid forecast.

Usage

```
reconstruct_forecast(
  imfs,
  trend,
  h,
  lag = 5,
  hidden = c(70, 20),
  maxit = 2500,
  seed = 123
)
```

Arguments

`imfs`	Numeric matrix or data frame containing EMD components, typically returned by `emd_decompose()`.
`trend`	Numeric vector containing the SSA-extracted trend from IMF1.
`h`	Forecast horizon.
`lag`	Number of lagged observations used as ANN inputs. Default is 5.
`hidden`	Hidden-layer sizes for the ANN. Default is `c(70, 20)`.
`maxit`	Maximum number of neural-network training steps. Default is 2500.
`seed`	Random seed for reproducible ANN fitting. Default is 123.

Details

The SSA-extracted trend from IMF1 is forecast separately using an artificial neural network. Each remaining EMD component, including the residual component, is also forecast separately. The component forecasts are then summed to obtain the final hybrid forecast.

Value

A list containing:

fitted Final reconstructed in-sample fitted values.

forecast Final reconstructed forecasts.

component_fitted A list containing fitted values for the SSA-extracted trend and individual EMD components.

component_forecasts A list containing forecasts for the SSA-extracted trend and individual EMD components.

component_models A list containing the fitted ANN model objects for each component.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

# Perform EMD decomposition
imfs <- emd_decompose(
  train = series,
  S_number = 4,
  num_siftings = 50
)

# Extract the trend from IMF1 using SSA
ssa <- ssa_trend(
  imfs[, "IMF1"],
  L = 12,
  trend_rate = 0.90
)

# Forecast and reconstruct the components
result <- reconstruct_forecast(
  imfs = imfs,
  trend = ssa$trend,
  h = 5,
  lag = 5,
  hidden = c(10, 5),
  maxit = 500,
  seed = 123
)

# Display the final forecasts
result$forecast
```

`ssa_trend` *Extract an IMF Trend Using Singular Spectrum Analysis*

Description

Apply Singular Spectrum Analysis (SSA) to an intrinsic mode function and reconstruct the minimum number of eigentriples required to reach a specified Trend Rate.

Usage

```
ssa_trend(imf, L = 12, trend_rate = 0.9)
```

Arguments

`imf`	Numeric IMF series, normally IMF1 obtained from `emd_decompose()`.
`L`	SSA window length. Default is 12.
`trend_rate`	Proportion of the cumulative SSA contribution retained for trend reconstruction. Must be between 0 and 1. Default is 0.90.

Details

Singular Spectrum Analysis is applied to the supplied IMF series. The relative contribution of each SSA component is calculated from the singular values, and the minimum number of leading eigentriples whose cumulative contribution reaches the specified `trend_rate` is selected for reconstruction.

A `trend_rate` of 0 returns a zero-valued trend, corresponding to complete removal of the supplied IMF. A `trend_rate` of 1 returns the original IMF unchanged.

Value

A list containing:

trend Reconstructed SSA trend component.

ssa_model Fitted SSA model.

L SSA window length.

trend_rate Specified Trend Rate.

eigentriples Number of eigentriples retained for trend reconstruction.

lambda SSA eigenvalues.

contribution Relative contribution of the SSA components.

cumulative_contribution Cumulative relative contribution of the SSA components.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

# Perform EMD decomposition
imfs <- emd_decompose(
  train = series,
  S_number = 4,
  num_siftings = 50
)

# Extract the trend from IMF1 using SSA
result <- ssa_trend(
  imf = imfs[, "IMF1"],
  L = 12,
  trend_rate = 0.90
)
```

```
# Display the extracted trend
result$trend

# Display the number of retained eigentriples
result$eigentriples
```

`tune_emd_ssa_annTSforecast`

Tune SSA Window and Trend Rate

Description

Evaluate combinations of SSA window lengths and Trend Rate values using a supplied hold-out test set.

Usage

```
tune_emd_ssa_annTSforecast(
  train,
  test,
  L_values = c(12, 36, 48),
  trend_rates = c(0.85, 0.9, 0.95),
  lag = 5,
  hidden = c(70, 20),
  maxit = 2500,
  S_number = 4,
  num_siftings = 50,
  seed = 123
)
```

Arguments

`train`	Numeric training time series.
`test`	Numeric hold-out observations used for evaluating forecasts.
`L_values`	Numeric vector of SSA window lengths to evaluate.
`trend_rates`	Numeric vector of Trend Rate values to evaluate. Each value must be between 0 and 1.
`lag`	Number of lagged observations used as ANN inputs. Default is 5.
`hidden`	Hidden-layer sizes for the ANN. Default is `c(70, 20)`.
`maxit`	Maximum number of neural-network training steps. Default is 2500.
`S_number`	S-number stopping criterion for EMD. Default is 4.
`num_siftings`	Maximum number of EMD sifting iterations. Default is 50.
`seed`	Random seed for reproducible ANN fitting. Default is 123.

Details

The function first decomposes the training series using EMD. For each combination of SSA window length and Trend Rate, SSA is applied to IMF1 to extract the trend component. The resulting hybrid EMD-SSA-ANN model is then used to forecast the complete hold-out period. Forecast accuracy is evaluated using RMSE, MAE, MAPE, and R-squared.

Value

A data frame containing the evaluated combinations of SSA window length and Trend Rate, together with the forecast horizon and accuracy measures RMSE, MAE, MAPE, and R-squared. The returned data frame is ordered by RMSE, MAE, and MAPE.

Examples

```
# Example using the built-in AirPassengers dataset
series <- as.numeric(AirPassengers)

# Use the first 132 observations for training and the final
# 12 observations as the hold-out test set
train <- series[1:132]
test <- series[133:144]

tuning <- tune_emd_ssa_annTSforecast(
  train = train,
  test = test,
  L_values = c(12, 24),
  trend_rates = c(0.90, 0.95),
  lag = 5,
  hidden = c(10, 5),
  maxit = 500,
  seed = 123
)

tuning
```

Index

accuracy_metrics, 2
annTSforecast, 3

emd_decompose, 4
emd_ssa_annTSforecast, 5

plot.emd_ssa_annTSforecast, 7
plot_imfs, 8
plot_ssa_trend, 9

reconstruct_forecast, 10

ssa_trend, 11

tune_emd_ssa_annTSforecast, 13