Package ‘autostsm’

August 12, 2021

Type Package

Title Automatic Structural Time Series Models

Version 1.6

Date 2021-08-01

Author Alex Hubbard

Maintainer Alex Hubbard <hubbard.alex@gmail.com>


License GPL (>= 2)

Imports maxLik (>= 1.4), forecast (>= 8.13), lubridate (>= 1.7), ggplot2 (>= 3.3), gridExtra (>= 2.3), strucchange (>= 1.5), foreach (>= 1.5), doSNOW (>= 1.0), parallel (>= 4.0), zoo (>= 1.8), lmtest (>= 0.9), ggrepel (>= 0.9), progress (>= 1.2), sandwich (>= 3.0),

Depends R (>= 3.5.0), data.table (>= 1.13)

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2021-08-12 14:40:02 UTC
Description

`autostsm` Automatic model selection for structural time series decomposition into trend, cycle, and seasonal components using the Kalman filter. See the package vignette using `vignette("autostsm")` or `browseVignettes("autostsm")` to view it in your browser.

Author(s)

Alex Hubbard
**GDP**

**US GDP Seasonally Adjusted**

**Description**

US GDP Seasonally Adjusted

**Usage**

data(GDP)

**Format**

data.table with columns DATE and GDP, quarterly frequency

**Source**

FRED

---

**NA000334Q**

**US GDP Not Seasonally Adjusted**

**Description**

US GDP Not Seasonally Adjusted

**Usage**

data(NA000334Q)

**Format**

data.table with columns DATE and NA000334Q, quarterly frequency

**Source**

FRED
SP500 | S&P 500

Description

S&P 500

Usage

data(SP500)

Format

data.table with columns DATE and SP500, daily frequency

Source

FRED

stsm_bdiag | Build a block diagonal matrix from two matrices

Description

Build a block diagonal matrix from two matrices

Usage

stsm_bdiag(A, B)

Arguments

A | The top left matrix
B | The bottom right matrix

Value

A block diagonal matrix
**stsm_build_dates**

**Build the date sequence as a Date type**

**Description**

Build the date sequence as a Date type

**Usage**

```
stsm_build_dates(y)
```

**Arguments**

- `y` a list object created from `stsm_detect_frequency`

**Value**

A list with the univariate time series and corrected dates

---

**stsm_check_exo**

**Data check for input exo**

**Description**

Checks for proper input of the table `exo`

**Usage**

```
stsm_check_exo(exo, y)
```

**Arguments**

- `exo` exo datagenous
- `y` input data `y`

**Value**

none
stsm_check_exo_fc  Data check for input exo.fc

Description
Checks for proper input of the table exo.fc

Usage
stsm_check_exo_fc(exo.fc, n.ahead)

Arguments
- exo.fc: exogenous forecast data
- n.ahead: forecast periods

Value
none

stsm_check_y  Data check for input y

Description
Checks for proper input of the table y

Usage
stsm_check_y(y)

Arguments
- y: input data y

Value
none
stsm_constraints

Set the inequality constraints for estimation

Description

Inequality constraints: ineqA

Usage

stsm_constraints(
    prior,
    par,
    freq,
    unconstrained,
    det_trend,
    det_drift,
    det_cycle,
    det_seas,
    det_obs,
    saturating_growth
)

Arguments

prior A data table created by stsm_prior
par parameter values for the state space model
freq Frequency of the data
unconstrained Whether to remove inequality constraints on the trend during estimation
det_trend Set the trend error variance to 0 (deterministic trend)
det_drift Set the drift error variance to 0 (deterministic drift)
det_cycle Set the cycle error variance to 0 (deterministic cycle)
det_seas Set the seasonality error variances to 0 (deterministic seasonality)
det_obs Set the observation equation error variance to 0 (deterministic observation equation)
saturating_growth Force the growth rate to converge to 0 in the long term

Value

list containing the initial values for the Kalman filter
**stsm_coxstuart**  
*Cox-Stuart Test*

**Description**

Taken from the ‘tsutils’ package. Performs the Cox-Stuart test for trend, deviation, or dispersion

**Usage**

```r
stsm_coxstuart(
  y,
  type = c("trend", "deviation", "dispersion"),
  sig_level = 0.01
)
```

**Arguments**

- **y**
  - input data
- **type**
  - Type of test: "trend", "deviation", or "dispersion" If type = "trend", test for changes in trend If type = "deviation", test for changes in deviation If type = "dispersion", test for changes in dispersion (range)
- **sig_level**
  - Significance level to determine statistically significant seasonal frequencies

**Value**

list describing the results

---

**stsm_detect_anomalies**  
*Detect Anomalies*

**Description**

Detect anomalies using the estimated structural time series model

**Usage**

```r
stsm_detect_anomalies(
  model,
  y = NULL,
  freq = NULL,
  exo = NULL,
  sig_level = 0.01,
  smooth = TRUE,
  plot = FALSE
)
```
stsm_detect_breaks

Detect Structural Breaks

Description

Detect structural breaks using the estimated structural time series model

Usage

```r
stsm_detect_breaks(
  model,
  y,
  components = c("trend", "cycle", "seasonal"),
)```
stsm_detect_breaks

freq = NULL,
exo = NULL,
sig_level = 0.01,
c1 = 0.8,
smooth = TRUE,
plot = FALSE,
cores = NULL,
show_progress = FALSE
)

Arguments
model Structural time series model estimated using stsm_estimate.
y Univariate time series of data values. May also be a 2 column data frame containing a date column.
components Vector of components to test for structural breaks
freq Frequency of the data (1 (yearly), 4 (quarterly), 12 (monthly), 365.25/7 (weekly), 365.25 (daily)), default is NULL and will be automatically detected
exo Matrix of exogenous variables used for the historical data. Can be used to specify regression effects or other seasonal effects like holidays, etc.
sig_level Significance level to determine statistically significant anomalies
ci Confidence interval, value between 0 and 1 exclusive.
smooth Whether or not to use the Kalman smoother
plot Whether to plot everything
cores Number of cores to use for break detection
show_progress Whether to show progress bar

Value
data table (or list of data tables) containing the dates of detected anomalies from the filtered and/or smoothed series

Examples
## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm") #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[ date >= "1990-01-01", ]
stsm = stsm_estimate(NA000334Q)
breaks = stsm_detect_breaks(model = stsm, y = NA000334Q, plot = TRUE, cores = 2)
## End(Not run)
stsm_detect_cycle

Detect cycle from the data

Description
Detect cycle from the data

Usage

```r
stsm_detect_cycle(
  y,
  freq,
  sig_level = 0.01,
  prior = NULL,
  cl = NULL,
  cores = NULL,
  show_progress = FALSE
)
```

Arguments

- **y**: Univariate time series of data values.
- **freq**: Frequency of the data (1 (yearly), 4 (quarterly), 12 (monthly), 365.25/7 (weekly), 365.25 (daily))
- **sig_level**: Significance level to determine statistically significant seasonal frequencies
- **prior**: A data table created by stsm_prior
- **cl**: A parallel cluster object
- **cores**: Number of cores to use
- **show_progress**: Whether to show progress bar

Value

Numeric value of cycle periodicity

Examples

```r
## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm") #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[date >= "1990-01-01", ]
cycle = stsm_detect_cycle(y = NA000334Q$y, freq = 4)

## End(Not run)
```
stsm_detect_frequency  Detect frequency and dates from the data

Description
Detect frequency and dates from the data

Usage
stsm_detect_frequency(y, freq = NULL)

Arguments
y  Univariate time series of data values. May also be a 2 column data frame containing a date column.
freq  Initial setting for the frequency detection

Value
List giving the dates and frequency of the data

Examples
## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm")  #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[NA000334Q$date >= "1990-01-01", ]
freq = stsm_detect_frequency(y = NA000334Q)
## End(Not run)

stsm_detect_multiplicative
Detect if log transformation is best

Description
Detect if log transformation is best

Usage
stsm_detect_multiplicative(y, freq, sig_level = 0.01, prior = NULL)
**stsm_detect_seasonality**

**Detect seasonality from the data**

**Description**

Detect seasonality from the data

**Usage**

```r
stsm_detect_seasonality(
  y, freq,
  sig_level = 0.01,
  prior = NULL,
  cl = NULL,
  cores = NULL,
  show_progress = FALSE
)
```

**Arguments**

- **y**: an object created from stsm_detect_frequency
- **freq**: Frequency of the data
- **sig_level**: Significance level to determine statistically significant seasonal frequencies
- **prior**: A data table created by stsm_prior

**Value**

A logical indicating if the model should be multiplicative or not

**Examples**

```r
## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm") #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[date >= "1990-01-01", ]
multiplicative = stsm_detect_multiplicative(y = NA000334Q$y, freq = 4)

## End(Not run)
```
stsm_detect_trend

Detect trend type

Description
Detect trend type

Usage

stsm_detect_trend(
  y,
  freq,
  decomp = "",
  sig_level = 0.01,
  prior = NULL,
  seasons = NULL,
  cycle = NULL
)
stsm_estimate

Arguments

y  Univariate time series of data values. May also be a 2 column data frame containing a date column.
freq  Frequency of the data (1 (yearly), 4 (quarterly), 12 (monthly), 365.25/7 (weekly), 365.25 (daily))
decomp  Decomposition model ("trend-cycle-seasonal", "trend-seasonal", "trend-cycle", "trend-noise")
sig_level  Significance level to determine statistically significant seasonal frequencies
prior  A data table created by stsm_prior
seasons  The seasonal periods
cycle  The cycle period

Value

list with trend type and logical flag for deterministic trend if the trend is determined to have 0 differencing

Examples

```r
## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm")  #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[date >= "1990-01-01", ]
trend = stsm_detect_trend(y = NA000334Q$y, freq = 4)
## End(Not run)
```

stsm_estimate

Trend cycle seasonal decomposition using the Kalman filter.

Description

Estimates a structural time series model using the Kalman filter and maximum likelihood. The seasonal and cycle components are assumed to be of a trigonometric form. The function checks three trend specifications to decompose a univariate time series into trend, cycle, and/or seasonal components plus noise. The function automatically detects the frequency and checks for a seasonal and cycle component if the user does not specify the frequency or decomposition model. This can be turned off by setting freq or specifying decomp. State space model for decomposition follows 
\[ Y_t = T_t + C_t + S_t + A^*X_t + e_t, e_t \sim N(0, \sigma^2_e) \] 
Y is the data T is the trend component C is the cycle component S is the seasonal component X is the exogenous data with parameter vector B e is the observation error.
Usage

\texttt{stsm\_estimate(}
\texttt{  \hspace{0.5cm} y,}
\texttt{  \hspace{0.5cm} exo = NULL,}
\texttt{  \hspace{0.5cm} freq = NULL,}
\texttt{  \hspace{0.5cm} decomp = NULL,}
\texttt{  \hspace{0.5cm} trend = NULL,}
\texttt{  \hspace{0.5cm} unconstrained = FALSE,}
\texttt{  \hspace{0.5cm} saturating\_growth = FALSE,}
\texttt{  \hspace{0.5cm} multiplicative = NULL,}
\texttt{  \hspace{0.5cm} par = NULL,}
\texttt{  \hspace{0.5cm} seasons = NULL,}
\texttt{  \hspace{0.5cm} cycle = NULL,}
\texttt{  \hspace{0.5cm} cores = NULL,}
\texttt{  \hspace{0.5cm} det\_obs = FALSE,}
\texttt{  \hspace{0.5cm} det\_trend = NULL,}
\texttt{  \hspace{0.5cm} det\_seas = FALSE,}
\texttt{  \hspace{0.5cm} det\_drift = FALSE,}
\texttt{  \hspace{0.5cm} det\_cycle = FALSE,}
\texttt{  \hspace{0.5cm} sig\_level = 0.01,}
\texttt{  \hspace{0.5cm} optim\_methods = c("BFGS", "NM", "CG", "SANN"),}
\texttt{  \hspace{0.5cm} maxit = 10000,}
\texttt{  \hspace{0.5cm} verbose = FALSE}
\texttt{)}

Arguments

\texttt{y} \hspace{0.5cm} Univariate time series of data values. May also be a 2 column data frame containing a date column.

\texttt{exo} \hspace{0.5cm} Matrix of exogenous variables. Can be used to specify regression effects or other seasonal effects like holidays, etc.

\texttt{freq} \hspace{0.5cm} Frequency of the data (1 (yearly), 4 (quarterly), 12 (monthly), 365.25/7 (weekly), 365.25 (daily)), default is NULL and will be automatically detected

\texttt{decomp} \hspace{0.5cm} Decomposition model ("tend-cycle-seasonal", "trend-seasonal", "trend-cycle", "trend-noise")

\texttt{trend} \hspace{0.5cm} Trend specification ("random-walk", "random-walk-drift", "double-random-walk", "random-walk2"). The default is NULL which will choose the best of all specifications based on the maximum likelihood. "random-walk" is the random walk trend. "random-walk-drift" is the random walk with constant drift trend. "double-random-walk" is the random walk with random walk drift trend. "random-walk2" is a 2nd order random walk trend as in the Hodrick-Prescott filter. If trend is "random-walk", the trend model is \( T_t = T_{t-1} + e_t, e_t \sim N(0, \text{sig}_t^2) \) If trend is "random-walk-drift", the trend model is \( T_t = T_{t-1} + D_{t-1} + e_t, e_t \sim N(0, \text{sig}_t^2) \) with \( D_t = d + \text{phi}_d D_{t-1} + n_t, n_t \sim N(0, \text{sig}_d^2) \) If trend is "double-random-walk", the trend model is \( T_t = M_{t-1} + T_{t-1} + e_t, e_t \sim N(0, \text{sig}_t^2) \) with \( M_t = M_{t-1} + n_t, n_t \sim N(0, \text{sig}_d^2) \) If trend
is "random-walk2", the trend model is $T_t = 2T_{t-1} - T_{t-2} + e_t$, $e_t \sim N(0, \sigma_t^2)$

unconstrained Logical whether to remove inequality constraints on the trend during estimation

saturating_growth Force the growth rate to converge to 0 in the long term

multiplicative If data should be logged to create a multiplicative model. If multiplicative = TRUE, then the data is logged and the original model becomes multiplicative ($Y_t = T_t * C_t * S_t * B X_t * e_t$)

par Initial parameters, default is NULL

seasons The seasonal periods: i.e. c(365.25, 7 if yearly and weekly seasonality). Default is NULL and will be estimated via wavelet analysis. Can set to FALSE if want no seasonality

cycle, The period for the longer-term cycle. Default is NULL and will be estimated via wavelet analysis. Can set to FALSE if want no cycle.

cores Number of cores to use for seasonality and cycle detection

det_obs Set the observation equation error variance to 0 (deterministic observation equation) If det_obs = TRUE then the error variance of the observation equation ($\sigma_e$) is set to 0

det_trend Set the trend error variance to 0 (deterministic trend) If det_trend = TRUE then the error variance of the trend equation ($\sigma_t$) is set to 0 and is referred to as a smooth trend

det_seas Set the seasonality error variances to 0 (deterministic seasonality) If det_seas = TRUE then the error variance all seasonality frequency j equations ($\sigma_s$) are set to 0 and is referred to as deterministic seasonality

det_drift Set the drift error variance to 0 (deterministic drift) If det_drift = TRUE then the error variance of the drift equation ($\sigma_d$) is set to 0 and is referred to as a deterministic drift

det_cycle Set the cycle error variance to 0 (deterministic cycle) If det_cycle = TRUE then the error variance of the cycle equation ($\sigma_c$) is set to 0 and is referred to as a deterministic cycle

sig_level Significance level to determine statistically significant seasonal frequencies

optim_methods Vector of 1 to 3 optimization methods in order of preference ("NR", "BFGS", "CG", "BHHH", or "SANN")

maxit Maximum number of iterations for the optimization

verbose Logical whether to print messages or not

Value

List of estimation values including a data table with coefficients, convergence code, frequency, decomposition, seasonality, cyclicity, and trend specification as well as the a data table with the original data with dates. Any exogenous data given is also returned.
Examples

```r
## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm") #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[ date >= "1990-01-01", ]
stsm = stsm_estimate(NA000334Q)

## End(Not run)
```

### stsm_fixed_pars

**Fixed parameter setting**

#### Description

Fixed parameter setting

#### Usage

```r
stsm_fixed_pars(par, y, det_obs = FALSE, det_trend = FALSE, det_drift = FALSE, det_cycle = FALSE, det_seas = FALSE, saturating_growth = FALSE, exo = NULL)
```

#### Arguments

- **par**: Initial parameters
- **y**: Vector of univariate time series
- **det_obs**: Set the observation equation error variance to 0 (deterministic observation equation) If det_obs = TRUE then the error variance of the observation equation (sig_e) is set to 0
- **det_trend**: Set the trend error variance to 0 (deterministic trend) If det_trend = TRUE then the error variance of the trend equation (sig_t) is set to 0 and is referred to as a smooth trend
**det_drift**
Set the drift error variance to 0 (deterministic drift) If det_drift = TRUE then the error variance of the drift equation (sig_d) is set to 0 and is referred to as a deterministic drift.

**det_cycle**
Set the cycle error variance to 0 (deterministic cycle) If det_cycle = TRUE then the error variance of the cycle equation (sig_c) is set to 0 and is referred to as a deterministic cycle.

**det_seas**
Set the seasonality error variances to 0 (deterministic seasonality) If det_seas = TRUE then the error variance all seasonality frequency j equations (sig_s) are set to 0 and is referred to as deterministic seasonality.

**saturating_growth**
Force the growth rate to converge to 0 in the long term.

**exo**
Matrix of exogenous variables. Can be used to specify regression effects or other seasonal effects like holidays, etc.

---

**stsm_forecast**

**Kalman Filter and Forecast**

**Description**

Kalman filter and forecast an estimated model from stsm_estimate output.

**Usage**

```r
stsm_forecast(
  model,
  y,
  n.ahead = 0,
  freq = NULL,
  exo = NULL,
  exo.fc = NULL,
  ci = 0.8,
  plot = FALSE,
  plot.decomp = FALSE,
  plot.fc = FALSE,
  n.hist = NULL,
  smooth = TRUE,
  dampen_cycle = FALSE
)
```

**Arguments**

- **model**
  Structural time series model estimated using stsm_estimate.

- **y**
  Univariate time series of data values. May also be a 2 column data frame containing a date column.

- **n.ahead**
  the number of periods to forecast.
### Description

Format the exo table

### Usage

```r
stsm_format_exo(exo, dates, range)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>exo</td>
<td>exogenous data</td>
</tr>
<tr>
<td>dates</td>
<td>dates vector</td>
</tr>
<tr>
<td>range</td>
<td>range of data to include</td>
</tr>
</tbody>
</table>

Value

a data table

Description

Get initial parameter estimates for estimation

Usage

```r
stsm_init_pars(
  y, freq, trend, cycle, decomp = "", seasons = NULL, prior = NULL, sig_level = 0.01
)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>an object created from stsm_detect_frequency</td>
</tr>
<tr>
<td>freq</td>
<td>Frequency of the data</td>
</tr>
<tr>
<td>trend</td>
<td>Trend specification (&quot;random-walk&quot;, &quot;random-walk-drift&quot;, &quot;double-random-walk&quot;, &quot;random-walk2&quot;).</td>
</tr>
<tr>
<td>cycle</td>
<td>The period for the longer-term cycle</td>
</tr>
<tr>
<td>decomp</td>
<td>Decomposition model (&quot;tend-cycle-seasonal&quot;, &quot;trend-seasonal&quot;, &quot;trend-cycle&quot;, &quot;trend-noise&quot;)</td>
</tr>
<tr>
<td>seasons</td>
<td>The seasonal lengths to split the seasonality into</td>
</tr>
<tr>
<td>prior</td>
<td>A data table created by stsm_prior</td>
</tr>
<tr>
<td>sig_level</td>
<td>Significance level for statistical tests</td>
</tr>
</tbody>
</table>

Value

named vector containing the initial parameter estimates for estimation
Description

Simplified version taken from the `imputeTS` package. Uses Kalman Smoothing on structural time series models for imputation. It uses "StructTS" to build a "basic structural model" if the frequency of \( y \) is greater than 1. Otherwise, it uses a local trend model.

Usage

```r
stsm_na_kalman(y)
```

Arguments

- **y**
  - Univariate time series

Description

Return a naive model prior decomposition

Usage

```r
stsm_prior(y, freq, decomp = ",", seasons = NULL, cycle = NULL)
```

Arguments

- **y**
  - an object created from `stsm_detect_frequency`
- **freq**
  - Frequency of the data
- **decomp**
  - decomposition string
- **seasons**
  - The seasonal periods to split the seasonality into
- **cycle**
  - The cycle periods

Value

data table containing a naive decomposition using STL
Examples

## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm") #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[date >= "1990-01-01", ]
prior = stsm_prior(y = NA000334Q$y, freq = 4)

## End(Not run)

### stsm_ssm

**State space model**

**Description**

Creates a state space model in list form \( y_t = H \cdot B + e_t \cdot B = F \cdot B_{t-1} + u_t \)

**Usage**

```r
stsm_ssm(par = NULL, yt = NULL, decomp = NULL, trend = NULL, init = NULL, model = NULL, prior = NULL, freq = NULL, seasons = NULL, cycle = NULL)
```

**Arguments**

- **par**: Vector of named parameter values, includes the harmonics
- **yt**: Univariate time series of data values
- **decomp**: Decomposition model ("tend-cycle-seasonal", "trend-seasonal", "trend-cycle", "trend-noise")
- **trend**: Trend specification ("random-walk", "random-walk-drift", "double-random-walk", "random-walk2"). The default is NULL which will choose the best of all specifications based on the maximum likelihood. "random-walk" is the random walk trend. "random-walk-drift" is the random walk with constant drift trend. "double-random-walk" is the random walk with random walk drift trend. "random-walk2" is a 2nd order random walk trend as in the Hodrick-Prescott filter.
init: Initial state values for the Kalman filter
model: a stsm_estimate model object
prior: Model prior built from stsm_prior. Only needed if prior needs to be built for initial values
freq: Frequency of the data. Only needed if prior needs to be built for initial values and prior = NULL
seasons: Numeric vector of seasonal frequencies. Only needed if prior needs to be built for initial values and prior = NULL
cycle: Numeric value for the cycle frequency. Only needed if prior needs to be built for initial values and prior = NULL

Value
List of space space matrices

Examples

## Not run:
#GDP Not seasonally adjusted
library(autostsm)
data("NA000334Q", package = "autostsm") #From FRED
NA000334Q = data.table(NA000334Q, keep.rownames = TRUE)
colnames(NA000334Q) = c("date", "y")
NA000334Q[, "date" := as.Date(date)]
NA000334Q[, "y" := as.numeric(y)]
NA000334Q = NA000334Q[date >= "1990-01-01", ]
sts = stsm_estimate(NA000334Q)
ssm = stsm_ssm(model = stsm)

## End(Not run)

---

**UNRATE**

*Unemployment Rate Seasonally Adjusted*

**Description**

Unemployment Rate Seasonally Adjusted

**Usage**

data(UNRATE)

**Format**

data.table with columns DATE and UNRATE, monthly frequency

**Source**

FRED
Unemployment Rate Not Seasonally Adjusted

Description
Unemployment Rate Not Seasonally Adjusted

Usage
data(UNRATENSA)

Format
data.table with columns DATE and UNRATENSA, monthly frequency

Source
FRED
Index

* datasets
  GDP, 3
  NA000334Q, 3
  SP500, 4
  UNRATE, 24
  UNRATENSA, 25

autostsm, 2

GDP, 3

NA000334Q, 3

SP500, 4

stsm_bdiag, 4

stsm_build_dates, 5

stsm_check_exo, 5

stsm_check_exo_fc, 6

stsm_check_y, 6

stsm_constraints, 7

stsm_coxstuart, 8

stsm_detect_anomalies, 8

stsm_detect_breaks, 9

stsm_detect_cycle, 11

stsm_detect_frequency, 12

stsm_detect_multiplicative, 12

stsm_detect_seasonality, 13

stsm_detect_trend, 14

stsm_estimate, 15

stsm_fixed_pars, 18

stsm_forecast, 19

stsm_format_exo, 20

stsm_init_pars, 21

stsm_na_kalman, 22

stsm_prior, 22

stsm_ssm, 23

UNRATE, 24

UNRATENSA, 25