

# Package ‘KRONX’

May 7, 2026

**Type** Package

**Title** Clock of Regimes for Regime-Switching Fragility Analysis

**Version** 0.1.0

**Date** 2026-04-22

**Description** Implements the Clock of Regimes (KRONX) framework for regime-switching fragility analysis of financial time series. The package fits Gaussian and Student-t Hidden Markov Models (HMMs) to return data, constructs a hazard-adjusted transition operator  $Q$ , derives the associated generator  $K = Q - I$ , and computes the fundamental matrix  $N = -K$  inverse to characterize expected residence times under structural fragility.

**License** GPL (>= 3)

**Encoding** UTF-8

**Depends** R (>= 4.0.0)

**Imports** stats, utils

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown

**VignetteBuilder** utils

**Config/testthat/edition** 3

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** Oscar Linares [aut, cre]

**Maintainer** Oscar Linares <olinares@umich.edu>

**Repository** CRAN

**Date/Publication** 2026-04-24 20:20:27 UTC

## Contents

compute_log_returns . . . . .	2
fit_hmm_em . . . . .	2
read_close_series . . . . .	3
run_kronx . . . . .	4
viterbi_decode . . . . .	6

**Index****8**

---

compute_log_returns	<i>Compute log returns from a close-price vector.</i>
---------------------	---

---

**Description**

Compute log returns from a close-price vector.

**Usage**

```
compute_log_returns(close)
```

**Arguments**

close            Numeric vector of close prices.

**Value**

Numeric vector of log returns (length = length(close) - 1).

---

fit_hmm_em	<i>Fit a Gaussian or Student-t HMM via EM with multiple random restarts.</i>
------------	--

---

**Description**

Fit a Gaussian or Student-t HMM via EM with multiple random restarts.

**Usage**

```
fit_hmm_em(  
  x,  
  K = 3L,  
  model = c("gaussian", "student"),  
  n_starts = 5L,  
  max_iter = 200L,  
  tol = 1e-06,  
  nu_grid = c(3:30, 40, 60, 100),  
  verbose = TRUE  
)
```

**Arguments**

<code>x</code>	Numeric vector of observations (log returns).
<code>K</code>	Integer number of hidden states.
<code>model</code>	"gaussian" or "student".
<code>n_starts</code>	Number of random restarts.
<code>max_iter</code>	Maximum EM iterations per restart.
<code>tol</code>	Log-likelihood convergence tolerance.
<code>nu_grid</code>	Candidate degrees-of-freedom for Student-t grid search.
<code>verbose</code>	Logical; if TRUE print iteration progress.

**Value**

A list with components:

- model** Model type string.
- A**  $K \times K$  transition matrix.
- delta** Length- $K$  initial state probability vector.
- params** List with  $\mu$ ,  $\sigma$ , and (Student-t)  $\nu$ .
- gamma**  $T \times K$  posterior state probabilities.
- xi**  $(T-1) \times K \times K$  posterior joint transition probabilities.
- logLik** Scalar log-likelihood at convergence.
- n** Number of observations.
- K** Number of states.

---

<code>read_close_series</code>	<i>Read a close-price series from a CSV file.</i>
--------------------------------	---

---

**Description**

The function is case-insensitive: it accepts `close`, `Close`, `CLOSE`, or any capitalisation. The first matching column is used.

**Usage**

```
read_close_series(file, close_col = "close")
```

**Arguments**

<code>file</code>	Path to the CSV file.
<code>close_col</code>	Name of the close-price column (default "close").

**Value**

A numeric vector of finite close prices (length  $\geq 10$ ).

run\_kronx

*Run the full Clock of Regimes (KRONX) analysis.***Description**

Runs the KRONX pipeline on either a CSV file of prices or a numeric vector of returns supplied directly. When `file` is used, the function reads the CSV file, extracts the close-price column (case-insensitive match on "close" or "Close"), and computes log returns. When `x` is supplied, it is treated as a numeric return series and the file-ingestion step is skipped.

**Usage**

```
run_kronx(
  file = NULL,
  x = NULL,
  close_col = "close",
  K = 3L,
  n_starts = 5L,
  max_iter = 200L,
  tol = 1e-06,
  seed = 123L,
  epsilon_min = 0.01,
  c_hazard = 0.05,
  tail_alpha = 0.01,
  ruin_horizon = 250L,
  nu_grid = c(3:30, 40, 60, 100),
  verbose = TRUE,
  output_dir = "kronx_output"
)
```

**Arguments**

<code>file</code>	Path to a CSV file containing a close-price column. Ignored if <code>x</code> is supplied. Default: NULL.
<code>x</code>	Optional numeric vector of returns. If supplied, <code>file</code> is ignored and no CSV is read. This is intended in particular for small, self-contained examples and tests. Default: NULL.
<code>close_col</code>	Name of the close-price column. The function first tries the exact name, then a case-insensitive match, and finally falls back to any column whose name contains "close". Default: "close".
<code>K</code>	Number of hidden states. Default: 3L.
<code>n_starts</code>	Number of EM random restarts. Default: 5L.
<code>max_iter</code>	Maximum EM iterations per restart. Default: 200L.
<code>tol</code>	Log-likelihood convergence tolerance. Default: 1e-6.
<code>seed</code>	Random seed for reproducibility. Default: 123L.

<code>epsilon_min</code>	Minimum baseline Talebian hazard. Default: 0.01.
<code>c_hazard</code>	Hazard sensitivity to tail thickness. Default: 0.05.
<code>tail_alpha</code>	Left-tail probability used for the empirical loss threshold. Default: 0.01.
<code>ruin_horizon</code>	Horizon, in observations, used in the risk bound. Default: 250L.
<code>nu_grid</code>	Candidate degrees-of-freedom values for Student-t fitting. Default: c(3:30, 40, 60, 100).
<code>verbose</code>	Logical; if TRUE, print progress and a summary. Default: TRUE.
<code>output_dir</code>	Directory for output files. Created if it does not exist. Pass NULL to suppress file output. Default: "kronx_output".

## Details

The function then fits Gaussian and Student-t Hidden Markov Models (HMMs), builds the KRONX operator chain  $Q \rightarrow K \rightarrow N$ , and computes a residence-weighted upper bound on extreme downside risk. Results are returned as a named list and, optionally, written to `output_dir`.

## Value

A named list containing:

**close\_px** Numeric vector of raw close prices, or NULL when `x` is supplied directly.

**ret** Numeric vector of log returns or supplied returns.

**K** Number of hidden states.

**gauss\_fit** Fitted Gaussian HMM object.

**t\_fit** Fitted Student-t HMM object.

**A\_gauss** Gaussian transition matrix.

**A\_t** Student-t transition matrix.

**epsilon** Per-state Talebian hazard vector.

**Q** Hazard-adjusted operator.

**K\_mat** KRONX generator matrix.

**N\_mat** Fundamental matrix.

**pi\_qs** Quasi-stationary distribution.

**pi\_res** Residence-weight vector.

**loss\_threshold** Empirical left-tail return threshold.

**q\_state** Per-state tail probability.

**bar\_q** Residence-weighted tail probability.

**bar\_lambda** Residence-weighted hazard.

**ruin\_bound** Upper bound over `ruin_horizon`.

**tail\_alpha** The `tail_alpha` value used.

**ruin\_horizon** The `ruin_horizon` value used.

**state\_summary** A data.frame summarising per-state quantities.

**vpath\_t** Integer vector of Viterbi-decoded Student-t state labels.

**output\_files** Character vector of paths to written output files, or NULL if `output_dir` is NULL.

**input\_source** A short label describing the data source used.

**Examples**

```

# Small self-contained example suitable for automatic checking
set.seed(123)
x <- rnorm(120, mean = 0, sd = 0.01)
res <- run_kronx(
  x = x,
  K = 2L,
  n_starts = 1L,
  max_iter = 10L,
  tol = 1e-4,
  tail_alpha = 0.05,
  ruin_horizon = 25L,
  verbose = FALSE,
  output_dir = NULL
)
res$ruin_bound
dim(res$A_t)
# Self-contained file-based example
tmp <- tempfile(fileext = ".csv")
dat <- data.frame(
  date = sprintf("2026-01-%02d", 1:20),
  close = c(
    100.0, 100.2, 100.1, 100.5, 100.4,
    100.8, 101.0, 100.9, 101.3, 101.1,
    101.6, 101.5, 101.9, 102.0, 101.8,
    102.2, 102.4, 102.3, 102.7, 102.9
  )
)
utils::write.csv(dat, tmp, row.names = FALSE)
res_file <- run_kronx(
  file = tmp,
  close_col = "close",
  K = 2L,
  n_starts = 1L,
  max_iter = 8L,
  tol = 1e-4,
  tail_alpha = 0.10,
  ruin_horizon = 10L,
  verbose = FALSE,
  output_dir = NULL
)
res_file$ruin_bound

```

---

viterbi\_decode

---

*Decode the most likely hidden-state path via the Viterbi algorithm.*


---

**Description**

Decode the most likely hidden-state path via the Viterbi algorithm.

**Usage**

```
viterbi_decode(x, fit)
```

**Arguments**

<code>x</code>	Numeric vector of observations (log returns).
<code>fit</code>	A fitted HMM object returned by <a href="#">fit_hmm_em</a> .

**Value**

An integer vector of length `length(x)` with state labels (1-indexed).

# Index

`compute_log_returns`, [2](#)

`fit_hmm_em`, [2](#), [7](#)

`read_close_series`, [3](#)

`run_kronx`, [4](#)

`viterbi_decode`, [6](#)