Package: cycleTrendR (via r-universe)

May 8, 2026

Type Package

Title Adaptive Cycle and Trend Analysis for Irregular Time Series

Version 0.2.0

URL https://PietroPiu-labstats.github.io/cycleTrendR,
https://github.com/PietroPiu-labstats/cycleTrendR

Description Provides adaptive trend estimation, cycle detection, Fourier harmonic selection, bootstrap confidence intervals, change-point detection, and rolling-origin forecasting. Supports LOESS, GAM, and GAMM models, and automatically handles irregular sampling using the Lomb–Scargle periodogram. Designed for biomedical, environmental, and engineering time-series analysis.

License GPL-3 | file LICENSE

Encoding UTF-8

Imports blocklength, fANCOVA, ggplot2, lomb, gridExtra, changepoint, mgcv, dplyr, nortest, officer, flextable, nlme, magrittr, tseries

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev libfribidi-dev make libharfbuzz-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libwebp-dev libxml2-dev libssl-dev

Repository https://pietropiu-labstats.r-universe.dev

Date/Publication 2026-03-09 17:58:17 UTC

RemoteUrl https://github.com/pietropiu-labstats/cycletrendr

RemoteRef HEAD

RemoteSha fc55354c34d663c4e7615dd72de0b40690daa3d2

Contents

adaptive_cycle_trend_analysis 2

Index **7**

`adaptive_cycle_trend_analysis`
Adaptive Trend and Cycle Analysis for Time Series

Description

Performs adaptive trend estimation, cycle detection, Fourier harmonic selection, bootstrap confidence intervals, change points detection, and rolling-origin forecasting. Supports LOESS, GAM, and GAMM models, and automatically handles irregular sampling using the Lomb Scargle periodogram.

Usage

```
adaptive_cycle_trend_analysis(
  signal,
  dates,
  normalize = FALSE,
  trendmethod = c("loess", "gam"),
  usefourier = FALSE,
  fourierK = 2,
  auto_fourier_select = TRUE,
  fourier_selection_criterion = c("AICc", "BIC"),
  fourierK_max = 6,
  cimethod = c("model", "bootstrapiid", "bootstrapmbb"),
  nboot = 1000,
  blocksize = NULL,
  seasonalfrequency = 7,
  stlrobust = TRUE,
  specspans = c(7, 7),
  auto_seasonality = TRUE,
  lagmax = NULL,
  loess_span_mode = c("auto_aicc", "auto_gcv", "cv", "fixed"),
  loess_span_fixed = NULL,
  loess_span_grid = seq(0.15, 0.6, by = 0.05),
  loess_cv_k = 5,
  blocklength_mode = c("auto_pwsd", "heuristic", "fixed"),
  blocklength_fixed = NULL,
  robust = TRUE,
  use_gamm = FALSE,
  group_var = NULL,
  group_values = NULL,
  random_effect = NULL,
```

```
  cor_struct = c("none", "ar1", "arma"),
  arma_p = 1,
  arma_q = 0,
  forecast_holdout_h = 0,
  forecast_origin_mode = c("expanding", "sliding"),
  train_window = NULL,
  forecast_lock_K = TRUE,
  exportdocx = FALSE,
  exportplot = FALSE,
  outputpath = "analysisreport.docx",
  logopath = NULL,
  logowidth = 1.5,
  logoheight = 0.5,
  project_id = NULL,
  cohort_id = NULL,
  assay_version = NULL,
  analyst = NULL,
  run_date = Sys.Date(),
  notes = NULL,
  include_parameters_appendix = TRUE
)
```

Arguments

`signal`	Numeric vector of observed values.
`dates`	Date vector of the same length as `signal`.
`normalize`	Logical; if TRUE, Z score normalization is applied.
`trendmethod`	Character; `"loess"` or `"gam"`.
`usefourier`	Logical; whether to include Fourier harmonics.
`fourierK`	Integer; fixed number of harmonics if auto selection disabled.
`auto_fourier_select`	Logical; if TRUE, selects K via AICc/BIC.
`fourier_selection_criterion`	`"AICc"` or `"BIC"`.
`fourierK_max`	Maximum K to consider during selection.
`cimethod`	`"model"`, `"bootstrapiid"`, or `"bootstrapmbb"`.
`nboot`	Number of bootstrap samples.
`blocksize`	Block size for MBB bootstrap.
`seasonalfrequency`	Seasonal frequency for STL (regular sampling).
`stlrobust`	Logical; robust STL decomposition.
`specspans`	Smoothing spans for spectral estimation.
`auto_seasonality`	Logical; if TRUE, uses dominant period.

`lagmax` Maximum lag for ACF and Ljung Box tests.

`loess_span_mode` `"auto_aicc"`, `"auto_gcv"`, `"cv"`, `"fixed"`.

`loess_span_fixed` Numeric; fixed LOESS span.

`loess_span_grid` Grid of spans for CV.

`loess_cv_k` Number of folds for blocked CV.

`blocklength_mode` `"auto_pwsd"`, `"heuristic"`, `"fixed"`.

`blocklength_fixed` Fixed block length.

`robust` Logical; robust LOESS or robust GAM family.

`use_gamm` Logical; fit GAMM instead of GAM.

`group_var` Character; grouping variable for random intercepts.

`group_values` Optional vector to attach as grouping variable.

`random_effect` Optional random effects list for `mgcv::gamm`.

`cor_struct` `"none"`, `"ar1"`, `"arma"`.

`arma_p`, `arma_q` ARMA orders.

`forecast_holdout_h` Holdout horizon for forecasting.

`forecast_origin_mode` `"expanding"` or `"sliding"`.

`train_window` Training window for sliding origin.

`forecast_lock_K` Logical; lock Fourier K across origins.

`exportdocx` Logical; export DOCX report.

`exportplot` Logical; export plots.

`outputpath` Path for DOCX export.

`logopath` Optional logo for DOCX.

`logowidth`, `logoheight` Logo dimensions.

`project_id`, `cohort_id`, `assay_version`, `analyst`, `run_date`, `notes` Metadata.

`include_parameters_appendix` Logical; include appendix in DOCX.

Value

A list containing:

- Trend estimates
- Confidence intervals

- Residuals and diagnostics
- Fourier selection results
- Change-point locations
- Spectral analysis
- Forecast results (if enabled)
- ggplot2 objects for visualization

Examples

```
## Not run:
set.seed(1)
dates <- as.Date("2020-01-01") + cumsum(sample(1:3, 300, replace = TRUE))
signal <- sin(2*pi*as.numeric(dates)/20) + rnorm(300, 0, 0.3)

res_gam <- adaptive_cycle_trend_analysis(
  signal = signal,
  dates = dates,
  usefourier = TRUE,
  trendmethod = "gam"
)

dates <- as.Date("2020-01-01") + cumsum(sample(1:3, 150, replace = TRUE))
signal <- sin(2*pi*as.numeric(dates)/25) + rnorm(150, 0, 0.3)
group  <- rep(letters[1:4], length.out = length(signal))

res_gamm <- adaptive_cycle_trend_analysis(
  signal = signal,
  dates = dates,
  trendmethod = "gam",
  use_gamm = TRUE,
  group_var = "subject",
  group_values = group,
  usefourier = FALSE,
  nboot = 20
)

plot(res_gamm$Plot$Trend)

dates <- as.Date("2020-01-01") + 1:120
signal <- sin(2*pi*(1:120)/20) + rnorm(120, 0, 0.2)

res_loess <- adaptive_cycle_trend_analysis(
  signal = signal,
  dates = dates,
  trendmethod = "loess",
  usefourier = TRUE,
  auto_fourier_select = TRUE,
  nboot = 50
)

plot(res_loess$Plot$Trend)
```

```
## End(Not run)
```

Index

adaptive_cycle_trend_analysis, 2