Package ‘iAR’

July 3, 2025

Type Package

Title Irregularly Observed Autoregressive Models

Version 1.3.1

Date 2025-07-03

Maintainer Elorrieta Felipe <felipe.elorrieta@usach.cl>

Description Data sets, functions and scripts with examples to implement autoregressive models for irregularly observed time series. The models available in this package are the irregular autoregressive model (Eyheramendy et al.(2018) <doi:10.1093/mnras/sty2487>), the complex irregular autoregressive model (Elorrieta et al.(2019) <doi:10.1051/0004-6361/201935560>) and the bivariate irregular autoregressive model (Elorrieta et al.(2021) <doi:10.1093/mnras/stab1216>).

License GPL-2

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.7),ggplot2,stats, Rdpack, S7, zoo

RdMacros Rdpack

Suggests arfima

URL https://github.com/felipeelorrieta

LinkingTo Rcpp,RcppArmadillo

RoxygenNote 7.3.2

LazyData true

Encoding UTF-8

NeedsCompilation yes

Author Elorrieta Felipe [aut, cre],
Ojeda Cesar [aut],
Eyheramendy Susana [aut],
Palma Wilfredo [aut]

Repository CRAN

Date/Publication 2025-07-03 06:00:02 UTC

Contents

`iAR-package` *iAR: Irregularly Observed Autoregressive Models*

Description

Description: Data sets, functions and scripts with examples to implement autoregressive models for irregularly observed time series. The models available in this package are the irregular autoregressive model (Eyheramendy et al.(2018) <doi:10.1093/mnras/sty2487>), the complex irregular autoregressive model (Elorrieta et al.(2019) <doi:10.1051/0004-6361/201935560>) and the bivariate irregular autoregressive model (Elorrieta et al.(2021) <doi:10.1093/mnras/stab1216>)

Details

"_PACKAGE"

---

agn *Active Galactic Nuclei*

Description

Time series of the AGN MCG-6-30-15 measured in the K-band between 2006 August and 2011 July with the ANDICAM camera mounted on the 1.3 m telescope at Cerro Tololo Inter-American Observatory (CTIO)

Usage

```
agn
```

Format

A data frame with 237 observations on the following 3 variables:

t heliocentric Julian Day - 2450000

m Flux $(10^(-15) ergs/s/cm^2 /A)$

merr measurement error standard deviations.

References

Lira P, Arévalo P, Uttley P, McHardy IMM, Videla L (2015). "Long-term monitoring of the archetype Seyfert galaxy MCG-6-30-15: X-ray, optical and near-IR variability of the corona, disc and torus." *Monthly Notices of the Royal Astronomical Society*, **454**(1), 368-379. ISSN 0035-8711, doi:10.1093/mnras/stv1945.

Examples

```
data(agn)
plot(agn$t,agn$m,type="l",ylab="",xlab="")
```

BiAR *'BiAR' Class*

Description

Represents a bivariate irregular autoregressive (BiAR) time series model. This class extends the 'multidata' class and provides additional properties for modeling, forecasting, and interpolation of bivariate time series data.

Usage

```
BiAR(
  times = integer(0),
  series = integer(0),
  series_esd = integer(0),
  series_names = character(0),
  fitted_values = integer(0),
  loglik = integer(0),
  kalmanlik = integer(0),
  coef = c(0.8, 0),
  tAhead = 1,
  rho = 0,
  forecast = integer(0),
  interpolated_values = integer(0),
  interpolated_times = integer(0),
  interpolated_series = integer(0),
  zero_mean = TRUE,
  standardized = TRUE
)
```

Arguments

`times`	A numeric vector representing the time points.
`series`	A numeric matrix or vector representing the values of the time series.
`series_esd`	A numeric matrix or vector representing the error standard deviations of the time series.
`series_names`	An optional character vector representing the name of the series.
`fitted_values`	A numeric vector containing the fitted values from the model.
`loglik`	A numeric value representing the log-likelihood of the model.
`kalmanlik`	A numeric value representing the Kalman likelihood of the model.
`coef`	A numeric vector containing the estimated coefficients of the model.
`tAhead`	A numeric value specifying the forecast horizon (default: 1).
`rho`	A numeric vector containing the estimated coefficients of the model.
`forecast`	A numeric vector containing the forecasted values.

interpolated_values
: A numeric vector containing the interpolated values.

interpolated_times
: A numeric vector containing the times of the interpolated data points.

interpolated_series
: A numeric vector containing the interpolated series.

zero_mean
: A logical value indicating if the model assumes a zero-mean process (default: TRUE).

standardized
: A logical value indicating if the model assumes a standardized process (default: TRUE).

Details

The ‘BiAR‘ class is designed to handle bivariate irregularly observed time series data using an autoregressive approach. It extends the ‘multidata‘ class to include additional properties for modeling bivariate time series.

Key features of the ‘BiAR‘ class include: - Support for bivariate time series data. - Forecasting and interpolation functionalities for irregular time points. - Assumptions of zero-mean and standardized processes, configurable by the user. - Estimation of model parameters and likelihoods, including Kalman likelihood.

Validation Rules

- ‘@times‘ must be a numeric vector without dimensions and strictly increasing. - ‘@series‘ must be a numeric matrix with two columns (bivariate) or be empty. - The number of rows in ‘@series‘ must match the length of ‘@times‘. - ‘@series_esd‘, if provided, must be a numeric matrix. Its dimensions must match those of ‘@series‘, or it must have one row and the same number of columns. - If ‘@series_esd‘ contains NA values, they must correspond positionally to NA values in ‘@series‘. - ‘@series_names‘, if provided, must be a character vector with length equal to the number of columns in ‘@series‘, and all names must be unique. - ‘@coef‘ must be a numeric vector of length 2, with each element strictly between -1 and 1. - ‘@tAhead‘ must be a strictly positive numeric scalar.

References

Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). “A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series.” *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

Examples

```
o=iAR::utilities()
o<-gentime(o, n=200, distribution = "expmixture", lambda1 = 130, lambda2 = 6.5,p1 = 0.15, p2 = 0.85)
times=o@times
my_BiAR <- BiAR(times = times,coef = c(0.9, 0.3), rho = 0.9)

# Access properties
my_BiAR@coef
```

CiAR *'CiAR' Class*

Description

Represents a complex irregular autoregressive (CiAR) time series model. This class extends the 'unidata' class and provides additional properties for modeling, forecasting, and interpolating irregularly observed time series data with both negative and positive autocorrelation.

Usage

```
CiAR(
  times = integer(0),
  series = integer(0),
  series_esd = integer(0),
  series_names = character(0),
  fitted_values = integer(0),
  kalmanlik = integer(0),
  coef = c(0.9, 0),
  tAhead = 1,
  forecast = integer(0),
  interpolated_values = integer(0),
  interpolated_times = integer(0),
  interpolated_series = integer(0),
  zero_mean = TRUE,
  standardized = TRUE
)
```

Arguments

`times`	A numeric vector representing the time points.
`series`	A complex vector representing the values of the time series.
`series_esd`	A numeric vector representing the error standard deviations of the time series.
`series_names`	An optional character vector of length 1 representing the name of the series.
`fitted_values`	A numeric vector containing the fitted values from the model.
`kalmanlik`	A numeric value representing the Kalman likelihood of the model.
`coef`	A numeric vector of length 2, containing the coefficients of the model. Each value must lie within [-1, 1]. Defaults to 'c(0.9, 0)'.
`tAhead`	A numeric value specifying the forecast horizon (default: 1).
`forecast`	A numeric vector containing the forecasted values.
`interpolated_values`	A numeric vector containing the interpolated values.

interpolated_times
: A numeric vector containing the times of the interpolated data points.

interpolated_series
: A numeric vector containing the interpolated series.

zero_mean
: A logical value indicating if the model assumes a zero-mean process (default: TRUE).

standardized
: A logical value indicating if the model assumes a standardized process (default: TRUE).

Details

The ‘CiAR‘ class is designed to handle irregularly observed time series data with either negative or positive autocorrelation using an autoregressive approach. It extends the ‘unidata‘ class to include functionalities specific to the ‘CiAR‘ model.

Key features of the ‘CiAR‘ class include: - Support for irregularly observed time series data with negative or positive autocorrelation. - Forecasting and interpolation functionalities for irregular time points. - Configurable assumptions of zero-mean and standardized processes.

Validation

- Inherits all validation rules from the ‘unidata‘ class: - ‘@times‘, ‘@series‘, and ‘@series_esd‘ must be numeric vectors. - ‘@times‘ must not contain ‘NA‘ values and must be strictly increasing. - The length of ‘@series‘ must match the length of ‘@times‘. - The length of ‘@series_esd‘ must be 0, 1, or equal to the length of ‘@series‘. - ‘NA‘ values in ‘@series‘ must correspond exactly (positionally) to ‘NA‘ values in ‘@series_esd‘. - ‘@series_names‘, if provided, must be a character vector of length 1.

- ‘@coef‘ must be a numeric vector of length 2 with no dimensions. - Each value in ‘@coef‘ must be in the interval [-1, 1]. - ‘@tAhead‘ must be a strictly positive numeric scalar.

References

Elorrieta, F, Eyheramendy, S, Palma, W (2019). “Discrete-time autoregressive model for unequally spaced time-series observations.” *A&A*, **627**, A120. doi:10.1051/00046361/201935560.

Examples

```
o=iAR::utilities()
o<-gentime(o, n=200, distribution = "expmixture", lambda1 = 130, lambda2 = 6.5,p1 = 0.15, p2 = 0.85)
times=o@times
my_CiAR <- CiAR(times = times,coef = c(0.9, 0))

# Access properties
my_CiAR@coef
```

clcep *Classical Cepheid*

Description

Time series of a classical cepheid variable star obtained from HIPPARCOS.

Usage

```
clcep
```

Format

A data frame with 109 observations on the following 3 variables:

t heliocentric Julian Day

m magnitude

merr measurement error of the magnitude (in mag).

Details

The frequency computed by GLS for this light curve is 0.060033386. Catalogs and designations of this star: HD 1989: HD 305996 TYCHO-2 2000:TYC 8958-2333-1 USNO-A2.0:USNO-A2 0225-10347916 HIP: HIP-54101

Examples

```
data(clcep)
f1=0.060033386
foldlc(clcep,f1)
```

cvnovag *ZTF g-band Cataclysmic Variable/Nova*

Description

Time series of a cataclysmic variable/nova object observed in the g-band of the ZTF survey and processed by the ALeRCE broker.ZTF Object code: ZTF18aayzpbr

Usage

```
cvnovag
```

Format

A data frame with 67 observations on the following 3 variables:

t heliocentric Julian Day - 2400000

m magnitude

merr measurement error standard deviations.

References

Förster F, Cabrera-Vives G, Castillo-Navarrete E, Estévez PA, Sánchez-Sáez P, Arredondo J, Bauer FE, Carrasco-Davis R, Catelan M, Elorrieta F, Eyheramendy S, Huijse P, Pignata G, Reyes E, Reyes I, Rodríguez-Mancini D, Ruz-Mieres D, Valenzuela C, Álvarez-Maldonado I, Astorga N, Borissova J, Clocchiatti A, Cicco DD, Donoso-Oliva C, Hernández-García L, Graham MJ, Jordán A, Kurtev R, Mahabal A, Maureira JC, Muñoz-Arancibia A, Molina-Ferreiro R, Moya A, Palma W, Pérez-Carrasco M, Protopapas P, Romero M, Sabatini-Gacitua L, Sánchez A, Martín JS, Sepúlveda-Cobo C, Vera E, Vergara JR (2021). "The Automatic Learning for the Rapid Classification of Events (ALeRCE) Alert Broker." *The Astronomical Journal*, **161**(5), 242. doi:10.3847/15383881/abe9bc.

Examples

```
data(cvnovag)
plot(cvnovag$t,cvnovag$m,type="l",ylab="",xlab="",col="green")
```

---

cvnovar — *ZTF r-band Cataclysmic Variable/Nova*

Description

Time series of a cataclysmic variable/nova object observed in the r-band of the ZTF survey and processed by the ALeRCE broker.ZTF Object code: ZTF18aayzpbr

Usage

```
cvnovar
```

Format

A data frame with 65 observations on the following 3 variables:

t heliocentric Julian Day - 2400000

m magnitude

merr measurement error standard deviations.

References

Förster F, Cabrera-Vives G, Castillo-Navarrete E, Estévez PA, Sánchez-Sáez P, Arredondo J, Bauer FE, Carrasco-Davis R, Catelan M, Elorrieta F, Eyheramendy S, Huijse P, Pignata G, Reyes E, Reyes I, Rodríguez-Mancini D, Ruz-Mieres D, Valenzuela C, Álvarez-Maldonado I, Astorga N, Borissova J, Clocchiatti A, Cicco DD, Donoso-Oliva C, Hernández-García L, Graham MJ, Jordán A, Kurtev R, Mahabal A, Maureira JC, Muñoz-Arancibia A, Molina-Ferreiro R, Moya A, Palma W, Pérez-Carrasco M, Protopapas P, Romero M, Sabatini-Gacitua L, Sánchez A, Martín JS, Sepúlveda-Cobo C, Vera E, Vergara JR (2021). "The Automatic Learning for the Rapid Classification of Events (ALeRCE) Alert Broker." *The Astronomical Journal*, **161**(5), 242. doi:10.3847/15383881/abe9bc.

Examples

```
data(cvnovar)
plot(cvnovar$t,cvnovar$m,type="l",ylab="",xlab="",col="red")
```

---

dmcep *Double Mode Cepheid.*

Description

Time series of a double mode cepheid variable star obtained from OGLE.

Usage

```
dmcep
```

Format

A data frame with 191 observations on the following 3 variables:

t heliocentric Julian Day

m magnitude

merr measurement error of the magnitude (in mag).

Details

The dominant frequency computed by GLS for this light curve is 0.7410152. The second frequency computed by GLS for this light curve is 0.5433353. OGLE-ID:175210

Examples

```
data(dmcep)
f1=0.7410152
foldlc(dmcep,f1)
#fit=harmonicfit(dmcep,f1)
#f2=0.5433353
#foldlc(cbind(dmcep$t,fit$res,dmcep$merr),f2)
```

dscut — *Delta Scuti*

Description

Time series of a Delta Scuti variable star obtained from HIPPARCOS.

Usage

```
dscut
```

Format

A data frame with 116 observations on the following 3 variables:

t heliocentric Julian Day

m magnitude

merr measurement error of the magnitude (in mag).

Details

The frequency computed by GLS for this light curve is 14.88558646. Catalogs and designations of this star: HD 1989: HD 199757 TYCHO-2 2000: TYC 7973-401-1 USNO-A2.0: USNO-A2 0450-39390397 HIP: HIP 103684

Examples

```
data(dscut)
f1=14.88558646
foldlc(dscut,f1)
```

eb — *Eclipsing Binaries (Beta Lyrae)*

Description

Time series of a Beta Lyrae variable star obtained from OGLE.

Usage

```
eb
```

Format

A data frame with 470 observations on the following 3 variables:

t heliocentric Julian Day

m magnitude

merr measurement error of the magnitude (in mag).

Details

The frequency computed by GLS for this light curve is 1.510571586. Catalogs and designations of this star:OGLE051951.22-694002.7

Examples

```
data(eb)
f1=1.510571586
foldlc(eb,f1)
```

`fit` *Fitted Values for iAR, CiAR, and BiAR Classes*

Description

Fitted Values for the provided data. This method is implemented for: 1. Irregular Autoregressive models (‘iAR‘) 2. Complex Irregular Autoregressive models (‘CiAR‘) 3. Bivariate Autoregressive models (‘BiAR‘)

Usage

```
fit(x, ...)
```

Arguments

`x` An object of class `iAR`, `CiAR`, or `BiAR`, containing the model specification and parameters:

- For `iAR`:
  - `family`: The distribution family of the iAR model (one of "norm", "t", or "gamma").
  - `series`: A numeric vector representing the time series to be fitted.
  - `coef`: The coefficient(s) of the iAR model.
  - `times`: A numeric vector specifying the time points of the series.
  - `zero_mean`: Logical, whether to fit a zero-mean model.
  - `standardized`: Logical, whether the model output should be standardized (for "norm" family).
  - `mean`: The mean parameter (only for "gamma" family).

- For CiAR:
  - coef: The real and imaginary parts of the CiAR model's coefficients.
  - series: A numeric vector representing the time series to be fitted.
  - times: A numeric vector specifying the time points of the series.
  - zero_mean: Logical, whether to fit a zero-mean model.
  - standardized: Logical, whether the model output should be standardized.
  - c: A scaling parameter for the CiAR model.
- For BiAR:
  - coef: The coefficients of the BiAR model (real and imaginary parts).
  - series: A numeric matrix with two columns representing the bivariate time series to be fitted.
  - times: A numeric vector specifying the time points of the series.
  - series_esd: A numeric matrix for the error structure (optional, used internally).
  - zero_mean: Logical, whether to fit a zero-mean model.

... Additional arguments (unused).

Details

This method fits the specified time series model to the data contained in the object. Depending on the class of the input object:

- For iAR, the function supports three distribution families:
- "norm" for normal distribution.
- "t" for t-distribution.
- "gamma" for gamma distribution.
- For CiAR, the function uses complex autoregressive processes.
- For BiAR, the function fits a bivariate autoregressive process.

All required parameters (e.g., coefficients, time points) must be set before calling this method.

Value

An updated object of class iAR, CiAR, or BiAR, where the fitted_values property contains the fitted time series values.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.,Elorrieta, F, Eyheramendy, S, Palma, W (2019). "Discrete-time autoregressive model for unequally spaced time-series observations." *A&A*, **627**, A120. doi:10.1051/00046361/201935560.,Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). "A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series." *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

Examples

```
# Example 1: Fitting a normal iAR model
library(iAR)
n=100
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_norm <- iAR(family = "norm", times = times, coef = 0.9)
model_norm <- sim(model_norm)
model_norm <- kalman(model_norm)
model_norm <- fit(model_norm)
plot(model_norm@times, model_norm@series, type = "l", main = "Original Series")
lines(model_norm@times, model_norm@fitted_values, col = "red", lwd = 2)
plot_fit(model_norm)

# Example 2: Fitting a CiAR model
set.seed(6714)
model_CiAR <- CiAR(times = times,coef = c(0.9, 0))
model_CiAR <- sim(model_CiAR)
y=model_CiAR@series
y1=y/sd(y)
model_CiAR@series=y1
model_CiAR@series_esd=rep(0,n)
model_CiAR <- kalman(model_CiAR)
print(model_CiAR@coef)
model_CiAR <- fit(model_CiAR)
yhat=model_CiAR@fitted_values

# Example 3: Fitting a BiAR model
n=80
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_BiAR <- BiAR(times = times,coef = c(0.9, 0.3), rho = 0.9)
model_BiAR <- sim(model_BiAR)
y=model_BiAR@series
y1=y/apply(y,2,sd)
model_BiAR@series=y1
model_BiAR@series_esd=matrix(0,n,2)
model_BiAR <- kalman(model_BiAR)
print(model_BiAR@coef)
model_BiAR <- fit(model_BiAR)
print(model_BiAR@rho)
yhat=model_BiAR@fitted_values
```

`foldlc` *Plotting folded light curves*

Description

This function plots a time series folded on its period.

Usage

```
foldlc(file, f1, plot = TRUE)
```

Arguments

`file`	Matrix with the light curve observations. The first column must have the irregular times, the second column must have the brightness magnitudes and the third column must have the measurement errors.
`f1`	Frequency (1/Period) of the light curve.
`plot`	logical; if TRUE, the function returns the plot of folded time series.

Value

A matrix whose first column has the folded (phased) observational times.

Examples

```
data(clcep)
f1=0.060033386
foldlc(clcep,f1)
```

`forecast` *Forecast for iAR, CiAR, and BiAR Classes*

Description

Generates forecasts for the specified time series model. This method is implemented for: 1. Irregular Autoregressive models (‘iAR‘) 2. Complex Irregular Autoregressive models (‘CiAR‘) 3. Bivariate Autoregressive models (‘BiAR‘)

Usage

```
forecast(x, ...)
```

Arguments

`x` An object of class `iAR`, `CiAR`, or `BiAR`, containing the model specification and parameters:

- For `iAR`:
  - `family`: The distribution family of the iAR model (one of "norm", "t", or "gamma").
  - `series`: A numeric vector representing the time series to forecast.

  - `coef`: The coefficient(s) of the iAR model.
  - `zero_mean`: Logical, whether the model assumes a zero-mean series.
  - `standardized`: Logical, whether the model uses standardized data (only for "norm" family).
  - `mean`: The mean parameter (only for "gamma" family).
  - `tAhead`: Integer, the number of steps ahead to forecast.
- For `CiAR`:
  - `coef`: The real and imaginary parts of the CiAR model's coefficients.
  - `series`: A numeric vector representing the time series to forecast.
  - `times`: A numeric vector specifying the time points of the series.
  - `zero_mean`: Logical, whether the model assumes a zero-mean series.
  - `standardized`: Logical, whether the model output should be standardized.
  - `tAhead`: Integer, the number of steps ahead to forecast.
- For `BiAR`:
  - `coef`: The coefficients of the BiAR model (real and imaginary parts).
  - `series`: A numeric matrix with two columns representing the bivariate time series to forecast.
  - `times`: A numeric vector specifying the time points of the series.
  - `tAhead`: Integer, the number of steps ahead to forecast.

`...` Additional arguments (unused).

Details

This method generates forecasts for the specified time series model. Depending on the class of the input object:

- For `iAR`, the function supports three distribution families:
- "norm" for normal distribution.
- "t" for t-distribution.
- "gamma" for gamma distribution.
- For `CiAR`, the function uses complex autoregressive processes.
- For `BiAR`, the function generates forecasts for a bivariate autoregressive process.

All required parameters (e.g., coefficients, time points) must be set before calling this method.

Value

An updated object of class `iAR`, `CiAR`, or `BiAR`, where the `forecast` property contains the forecasted values.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.,Elorrieta, F, Eyheramendy, S, Palma, W (2019). "Discrete-time autoregressive model for unequally spaced time-series observations." *A&A*, **627**, A120. doi:10.1051/00046361/201935560.,Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). "A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series." *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

Examples

```
# Example 1: Forecasting with a normal iAR model
library(iAR)
n=100
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_norm <- iAR(family = "norm", times = times, coef = 0.9)
model_norm <- sim(model_norm)
model_norm <- kalman(model_norm)
model_norm@tAhead=1.3
model_norm <- forecast(model_norm)
plot(times, model_norm@series, type = "l", main = "Original Series with Forecast")
points(max(times)+ model_norm@tAhead, model_norm@forecast, col = "blue", pch = 16)
plot_forecast(model_norm)

# Example 2: Forecasting with a CiAR model
set.seed(6714)
model_CiAR <- CiAR(times = times,coef = c(0.9, 0))
model_CiAR <- sim(model_CiAR)
y=model_CiAR@series
y1=y/sd(y)
model_CiAR@series=y1
model_CiAR@series_esd=rep(0,n)
model_CiAR <- kalman(model_CiAR)
print(model_CiAR@coef)
model_CiAR@tAhead=1.3
model_CiAR <-forecast(model_CiAR)
model_CiAR@forecast

# Example 3: Forecasting with a BiAR model
n=80
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_BiAR <- BiAR(times = times,coef = c(0.9, 0.3), rho = 0.9)
model_BiAR <- sim(model_BiAR)
```

```
y=model_BiAR@series
y1=y/apply(y,2,sd)
model_BiAR@series=y1
model_BiAR@series_esd=matrix(0,n,2)
model_BiAR <- kalman(model_BiAR)
print(model_BiAR@coef)
model_BiAR@tAhead=1.3
model_BiAR <-forecast(model_BiAR)
model_BiAR@forecast
```

`gentime` *Generating Irregularly spaced times*

Description

A method for generating time points based on a statistical distribution. The results are stored in the ‘times‘ slot of the ‘utilities‘ object.

Usage

```
gentime(x, ...)
```

Arguments

`x` An object of class ‘utilities‘.

`...` Additional arguments for generating time points:

n A positive integer. Length of observation times.

distribution A character string specifying the distribution of the observation times. Default is ‘"expmixture"‘. Available options are: - ‘"expmixture"‘: A mixture of two exponential distributions. - ‘"uniform"‘: A uniform distribution. - ‘"exponential"‘: A single exponential distribution. - ‘"gamma"‘: A gamma distribution.

lambda1 Mean (1/rate) of the exponential distribution or the first exponential distribution in a mixture of exponential distributions. Default is ‘130‘.

lambda2 Mean (1/rate) of the second exponential distribution in a mixture of exponential distributions. Default is ‘6.5‘.

p1 Weight of the first exponential distribution in a mixture of exponential distributions. Default is ‘0.15‘.

p2 Weight of the second exponential distribution in a mixture of exponential distributions. Default is ‘0.85‘.

a Shape parameter of a gamma distribution or lower limit of the uniform distribution. Default is ‘0‘.

b Scale parameter of a gamma distribution or upper limit of the uniform distribution. Default is ‘1‘.

Value

An updated ‘utilities‘ object with the generated observation times stored in the ‘times‘ slot.

References

Eyheramendy S, Elorrieta F, Palma W (2018). “An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves.” *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.

Examples

```
set.seed(12917)
o1=iAR::utilities()
o1<-gentime(o1, n=200, distribution = "expmixture", lambda1 = 130,
lambda2 = 6.5,p1 = 0.15, p2 = 0.85)
st=o1@times
mean(diff(st))

o1=iAR::utilities()
o1<-gentime(o1, n=200, distribution = "expmixture", lambda1 = 15,
lambda2 = 2.5,p1 = 0.15, p2 = 0.85)
st=o1@times
mean(diff(st))
```

---

harmonicfit — *Harmonic Fit to Time Series*

Description

This function fit an k-harmonic function to time series data.

Usage

```
harmonicfit(
  file,
  f1,
  nham = 4,
  weights = NULL,
  print = FALSE,
  remove_trend = TRUE
)
```

Arguments

`file`	A matrix with two columns. The first column corresponds to the observations times, and the second column corresponds to the measures.
`f1`	Frequency (1/Period) of the time series
`nham`	Number of harmonic components in the model
`weights`	An array with the weights of each observation
`print`	logical; if true, the summary of the harmonic fitted model will be printed. The default value is false.
`remove_trend`	logical; if true, the linear trend of time series will be removed before the the harmonic model is fitted.

Value

A list with the following components:

`res`	Residuals to the harmonic fit of the time series.
`t`	Observations times.
`R2`	Adjusted R-Squared.
`MSE`	Mean Squared Error.
`coef`	Summary of the coefficients estimated by the harmonic model.

Examples

```
data(clcep)
f1=0.060033386
#results=harmonicfit(file=clcep[,1:2],f1=f1)
#results$R2
#results$MSE
#results=harmonicfit(file=clcep[,1:2],f1=f1,nham=3)
#results$R2
#results$MSE
#results=harmonicfit(file=clcep[,1:2],f1=f1,weights=clcep[,3])
#results$R2
#results$MSE
```

`iAR` *'iAR' Class*

Description

Represents a univariate irregular autoregressive (iAR) time series model. This class extends the "unidata" class and includes additional properties for modeling, forecasting, and interpolation.

Usage

```
iAR(
  times = integer(0),
  series = integer(0),
  series_esd = integer(0),
  series_names = character(0),
  family = "norm",
  fitted_values = integer(0),
  loglik = integer(0),
  kalmanlik = integer(0),
  coef = 0.9,
  df = 3,
  sigma = 1,
  mean = 0,
  variance = 1,
  tAhead = 1,
  forecast = integer(0),
  interpolated_values = integer(0),
  interpolated_times = integer(0),
  interpolated_series = integer(0),
  zero_mean = TRUE,
  standardized = TRUE,
  hessian = FALSE,
  summary = list()
)
```

Arguments

`times`	A numeric vector representing the time points.
`series`	A numeric vector representing the values of the time series.
`series_esd`	A numeric vector representing the error standard deviations of the time series.
`series_names`	An optional character vector of length 1 representing the name of the series.
`family`	A character string indicating the distribution family of the model. Must be one of ‘"norm"‘, ‘"t"‘, or ‘"gamma"‘. Defaults to ‘"norm"‘.
`fitted_values`	A numeric vector containing the fitted values from the model.
`loglik`	A numeric value representing the log-likelihood of the model.
`kalmanlik`	A numeric value representing the Kalman likelihood of the model.
`coef`	A numeric vector containing the estimated coefficients of the model (default: 0.9).
`df`	A numeric value representing the degrees of freedom (‘t‘ distribution) (default: 3).
`sigma`	A numeric value representing the scale parameter (‘t‘ distribution) (default: 1).
`mean`	A numeric value representing the estimated mean of the model (‘gamma‘ parameter).

`variance` A numeric value representing the estimated variance of the model (‘gamma‘ parameter).

`tAhead` A numeric value specifying the forecast horizon (default: 1).

`forecast` A numeric vector containing the forecasted values.

`interpolated_values` A numeric vector containing the interpolated values.

`interpolated_times` A numeric vector containing the times of the interpolated data points.

`interpolated_series` A numeric vector containing the interpolated series.

`zero_mean` A logical value indicating if the model assumes a zero-mean process (default: TRUE).

`standardized` A logical value indicating if the model assumes a standardized process (default: TRUE).

`hessian` A logical value indicating whether the Hessian matrix is computed during estimation (default: FALSE).

`summary` A list containing the summary of the model fit, including diagnostics and statistical results.

Details

The ‘iAR‘ class is designed to handle irregularly observed time series data using an autoregressive approach. It extends the "unidata" class to include additional modeling and diagnostic capabilities. Key functionalities include forecasting, interpolation, and model fitting.

The class also supports advanced modeling features, such as: - Different distribution families for the data (e.g., Gaussian, ‘t‘-distribution). - Optional computation of the Hessian matrix for parameter estimation. - Standardized or zero-mean process assumptions.

Validation Rules

- Inherits all validation rules from the ‘unidata‘ class: - ‘@times‘, ‘@series‘, and ‘@series_esd‘ must be numeric vectors. - ‘@times‘ must not contain ‘NA‘ values and must be strictly increasing. - The length of ‘@series‘ must match the length of ‘@times‘. - The length of ‘@series_esd‘ must be 0, 1, or equal to the length of ‘@series‘. - ‘NA‘ values in ‘@series‘ must correspond exactly (positionally) to ‘NA‘ values in ‘@series_esd‘. - ‘@series_names‘, if provided, must be a character vector of length 1.

- ‘@family‘ must be one of ‘"norm"‘, ‘"t"‘, or ‘"gamma"‘. - ‘@coef‘ must be a numeric scalar strictly between 0 and 1. - ‘@df‘ must be a positive integer scalar **only if** ‘family = "t"‘; otherwise, it should not be specified. - ‘@sigma‘ must be a strictly positive numeric scalar (used in ‘"t"‘ family). - ‘@mean‘ and ‘@variance‘ must be strictly positive numeric scalars **only if** ‘family = "gamma"‘. - ‘@tAhead‘ must be a strictly positive numeric scalar.

References

Eyheramendy S, Elorrieta F, Palma W (2018). “An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves.” *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.

Examples

```
# Create an `iAR` object
o=iAR::utilities()
o<-gentime(o, n=200, distribution = "expmixture", lambda1 = 130, lambda2 = 6.5,p1 = 0.15, p2 = 0.85)
times=o@times
my_iAR <- iAR(family = "norm", times = times, coef = 0.9,hessian=TRUE)

my_iAR@family
my_iAR@coef
```

`iARPermutation` *Test for the significance of the autocorrelation estimated by the iAR package models*

Description

This function perform a test for the significance of the autocorrelation estimated by the iAR package models. This test is based in to take N disordered samples of the original data.

Usage

```
iARPermutation(
  series,
  times,
  series_esd = 0,
  iter = 100,
  coef,
  model = "iAR",
  plot = TRUE,
  xlim = c(-1, 0),
  df = 3
)
```

Arguments

`series`	Array with the time series observations.
`times`	Array with the irregular observational times.
`series_esd`	Array with the variance of the measurement errors.
`iter`	Number of disordered samples of the original data (N).
`coef`	autocorrelation estimated by one of the iAR package models.
`model`	model used to estimate the autocorrelation parameter ("iAR", "iAR-Gamma", "iAR-T", "CiAR" or "BiAR").
`plot`	logical; if true, the function return a density plot of the distribution of the bad fitted examples; if false, this function does not return a plot.

xlim	The x-axis limits (x1, x2) of the plot. Only works if plot='TRUE'. See plot.default for more details.
df	degrees of freedom parameter of the iAR-T model.

Details

The null hypothesis of the test is: The autocorrelation coefficient estimated for the time series belongs to the distribution of the coefficients estimated on the disordered data, which are assumed to be uncorrelated. Therefore, if the hypothesis is accepted, it can be concluded that the observations of the time series are uncorrelated.The statistic of the test is log(phi) which was contrasted with a normal distribution with parameters corresponding to the log of the mean and the variance of the phi computed for the N samples of the disordered data. This test differs for iARTest in that to perform this test it is not necessary to know the period of the time series.

Value

A list with the following components:

coef	MLE of the autocorrelation parameter of the model.
bad	MLEs of the autocorrelation parameters of the models that has been fitted to the disordered samples.
norm	Mean and variance of the normal distribution of the disordered data.
z0	Statistic of the test (log(abs(phi))).
pvalue	P-value computed for the test.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.

See Also

Planets,iARTest

Examples

```
data(Planets)
t<-Planets[,1]
res<-Planets[,2]
y=res/sqrt(var(res))
#res3=iARloglik(y,t,standardized=TRUE)[1]
#res3$coef
#set.seed(6713)
#require(ggplot2)
#test<-iARPermutation(series=y,times=t,coef=res3$coef,model="iAR",plot=TRUE,xlim=c(-9.6,-9.45))
```

iARTest — *Test for the significance of the autocorrelation estimated by the iAR package models in periodic irregularly observed time series*

Description

This function perform a test for the significance of the autocorrelation estimated by the iAR package models. This test is based on the residuals of the periodical time series fitted with an harmonic model using an incorrect period.

Usage

```
iARTest(
  series,
  times,
  series_esd = 0,
  f,
  coef,
  model = "iAR",
  plot = TRUE,
  xlim = c(-1, 0),
  df = 3
)
```

Arguments

`series`	Array with the time series observations.
`times`	Array with the irregular observational times.
`series_esd`	Array with the variance of the measurement errors.
`f`	Frequency (1/Period) of the raw time series.
`coef`	autocorrelation estimated by one of the iAR package models.
`model`	model used to estimate the autocorrelation parameter ("iAR", "iAR-Gamma", "iAR-T", "CiAR" or "BiAR").
`plot`	logical; if true, the function return a density plot of the distribution of the bad fitted examples; if false, this function does not return a plot.
`xlim`	The x-axis limits (x1, x2) of the plot. Only works if plot='TRUE'. See `plot.default` for more details.
`df`	degrees of freedom parameter of the iAR-T model.

Details

The null hypothesis of the test is: The autocorrelation estimated in the time series belongs to the distribution of the coefficients estimated for the residuals of the data fitted using wrong periods. Therefore, if the hypothesis is rejected, it can be concluded that the residuals of the harmonic model do not remain a time dependency structure.The statistic of the test is log(phi) which was contrasted with a normal distribution with parameters corresponding to the log of the mean and the variance of the phi computed for the residuals of the bad fitted light curves.

Value

A list with the following components:

`phi`	MLE of the autocorrelation parameter of the iAR/CiAR model.
`bad`	A matrix with two columns. The first column contains the incorrect frequencies used to fit each harmonic model. The second column has the MLEs of the autocorrelation parameters of the iAR/CiAR model that has been fitted to the residuals of the harmonic model fitted using the frequencies of the first column.
`norm`	Mean and variance of the normal distribution of the bad fitted examples.
`z0`	Statistic of the test (log(abs(phi))).
`pvalue`	P-value computed for the test.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.

See Also

`clcep`, `harmonicfit`,`iARPermutation`

Examples

```
data(clcep)
f1=0.060033386
#results=harmonicfit(file=clcep,f1=f1)
#y=results$res/sqrt(var(results$res))
#st=results$t
#res3=iARloglik(y,st,standardized=TRUE)[1]
#res3$coef
#require(ggplot2)
#test<-iARTest(series=clcep[,2],times=clcep[,1],f=f1,coef=res3$coef,
#model="iAR",plot=TRUE,xlim=c(-10,0.5))
#test
```

`interpolation` *Interpolation for iAR, CiAR, and BiAR Classes*

Description

This method performs imputation of missing values in a time series using an autoregressive model. The imputation is done iteratively for each missing value, utilizing available data and model coefficients. Depending on the model family, the interpolation is performed differently: - For `norm`: A standard autoregressive model for normally distributed data. - For `t`: A model for time series with t-distributed errors. - For `gamma`: A model for time series with gamma-distributed errors. - For `CiAR`: A complex irregular autoregressive model. - For `BiAR`: A bivariate autoregressive model.

Usage

```
interpolation(x, ...)
```

Arguments

`x` An object of class `iAR`, `CiAR`, or `BiAR`, containing the model specification and parameters:

- For `iAR`:
  - `family`: The distribution family of the iAR model (one of "norm", "t", or "gamma").
  - `series`: A numeric vector representing the time series to interpolate.
  - `coef`: The coefficients of the iAR model.
  - `zero_mean`: Logical, whether the model assumes a zero-mean series.
  - `standardized`: Logical, whether the model uses standardized data (only for "norm" family).
  - `mean`: The mean parameter (only for "gamma" family).
- For `CiAR`:
  - `coef`: The coefficients of the CiAR model.
  - `series_esd`: The series of error standard deviations for the CiAR model.
  - `zero_mean`: Logical, whether the model assumes a zero-mean series.
  - `standardized`: Logical, whether the model uses standardized data.
  - `seed`: Optional seed for random number generation.
- For `BiAR`:
  - `coef`: The coefficients of the BiAR model.
  - `series_esd`: The series of error standard deviations for the BiAR model.
  - `zero_mean`: Logical, whether the model assumes a zero-mean series.
  - `yini1`: Initial value for the first time series (for BiAR models).
  - `yini2`: Initial value for the second time series (for BiAR models).
  - `seed`: Optional seed for random number generation.

`...` Additional arguments (unused).

Details

Performs interpolation on time series with missing values. This method is implemented for: 1. Irregular Autoregressive models (iAR) 2. Complex Irregular Autoregressive models (CiAR) 3. Bivariate Autoregressive models (BiAR)

The method handles missing values (NA) in the time series by imputing them iteratively. For each missing value, the available data is used to fit the autoregressive model, and the missing value is imputed based on the model's output. For the `CiAR` and `BiAR` models, the error standard deviations and initial values are also considered during imputation.

Value

An object of the same class as `x` with interpolated time series.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.,Elorrieta, F, Eyheramendy, S, Palma, W (2019). "Discrete-time autoregressive model for unequally spaced time-series observations." *A&A*, **627**, A120. doi:10.1051/00046361/201935560.,Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). "A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series." *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

See Also

`forecast` for forecasting methods for these models.

Examples

```
# Interpolation for iAR model
library(iAR)
n=100
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_norm <- iAR(family = "norm", times = times, coef = 0.9)
model_norm <- sim(model_norm)
y=model_norm@series
y1=y/sd(y)
model_norm@series=y1
model_norm@series_esd=rep(0,n)
model_norm <- kalman(model_norm)
print(model_norm@coef)
napos=10
model_norm@series[napos]=NA
model_norm <- interpolation(model_norm)
interpolation=na.omit(model_norm@interpolated_values)
```

```
mse=as.numeric(y1[napos]-interpolation)^2
print(mse)
plot(times,y,type='l',xlim=c(times[napos-5],times[napos+5]))
points(times,y,pch=20)
points(times[napos],interpolation*sd(y),col="red",pch=20)

# Interpolation for CiAR model
model_CiAR <- CiAR(times = times,coef = c(0.9, 0))
model_CiAR <- sim(model_CiAR)
y=model_CiAR@series
y1=y/sd(y)
model_CiAR@series=y1
model_CiAR@series_esd=rep(0,n)
model_CiAR <- kalman(model_CiAR)
print(model_CiAR@coef)
napos=10
model_CiAR@series[napos]=NA
model_CiAR <- interpolation(model_CiAR)
interpolation=na.omit(model_CiAR@interpolated_values)
mse=as.numeric(y1[napos]-interpolation)^2
print(mse)
plot(times,y,type='l',xlim=c(times[napos-5],times[napos+5]))
points(times,y,pch=20)
points(times[napos],interpolation*sd(y),col="red",pch=20)
# Interpolation for BiAR model
model_BiAR <- BiAR(times = times,coef = c(0.9, 0.3), rho = 0.9)
model_BiAR <- sim(model_BiAR)
y=model_BiAR@series
y1=y/apply(y,2,sd)
model_BiAR@series=y1
model_BiAR@series_esd=matrix(0,n,2)
model_BiAR <- kalman(model_BiAR)
print(model_BiAR@coef)
napos=10
model_BiAR@series[napos,1]=NA
model_BiAR@series_esd[napos,1]=NA
model_BiAR <- interpolation(model_BiAR)
interpolation=na.omit(model_BiAR@interpolated_values[,1])
mse=as.numeric(y1[napos,1]-interpolation)^2
print(mse)
par(mfrow=c(2,1))
plot(times,y[,1],type='l',xlim=c(times[napos-5],times[napos+5]))
points(times,y[,1],pch=20)
points(times[napos],interpolation*apply(y,1,sd)[1],col="red",pch=20)
plot(times,y[,2],type='l',xlim=c(times[napos-5],times[napos+5]))
points(times,y[,2],pch=20)
```

`kalman` *Maximum Likelihood Estimation of Parameters for iAR, CiAR, and BiAR Models using the Kalman Filter*

Description

Performs Maximum Likelihood Estimation (MLE) of the parameters of the iAR, CiAR, and BiAR models by maximizing the likelihood function using the Kalman filter. This method applies the Kalman filter to compute the likelihood and estimate the model parameters that maximize the likelihood for each model type.

Usage

```
kalman(x, ...)
```

Arguments

`x` An object of class `iAR`, `CiAR`, or `BiAR` containing the model specification and parameters:

- For `iAR` models, the object should contain:
  - `series`: A numeric vector of the time series data.
  - `times`: A numeric vector specifying the time points of the series.
  - `series_esd`: A numeric vector specifying the error structure.
  - `zero_mean`: A logical value indicating if the series should be zero-centered.
  - `standardized`: A logical value indicating if the series should be standardized.
- For `CiAR` models, the object should contain:
  - `series`: A numeric vector of the time series data.
  - `times`: A numeric vector specifying the time points of the series.
  - `series_esd`: A numeric vector specifying the error structure.
  - `zero_mean`: A logical value indicating if the series should be zero-centered.
  - `standardized`: A logical value indicating if the series should be standardized.
  - `c`: A numeric value specifying the scale parameter for the CiAR model.
  - `niter`: An integer specifying the number of iterations for the Kalman filter.
  - `seed`: An integer seed for random number generation (optional).
- For `BiAR` models, the object should contain:
  - `series`: A numeric matrix containing two columns for bivariate time series data.
  - `times`: A numeric vector specifying the time points of the series.
  - `series_esd`: A numeric matrix specifying the error structure for both series.
  - `zero_mean`: A logical value indicating if the series should be zero-centered.
  - `niter`: An integer specifying the number of iterations for the Kalman filter.
  - `seed`: An integer seed for random number generation (optional).

`...` Additional arguments (unused).

Details

This function applies the Kalman filter to perform Maximum Likelihood Estimation (MLE) of the parameters of the autoregressive models (iAR, CiAR, BiAR). The Kalman filter is used to maximize the likelihood function based on the given time series data, and the parameters that maximize the likelihood are estimated.

- For `iAR`, the Kalman filter is applied to estimate the model parameters by maximizing the likelihood. - For `CiAR`, the Kalman filter is applied to estimate the parameters of the complex autoregressive model by maximizing the likelihood. - For `BiAR`, the Kalman filter is applied to estimate the parameters of the bivariate autoregressive model by maximizing the likelihood.

The method returns the updated model object, including the estimated parameters and the log-likelihood value.

Value

An updated object of class `iAR`, `CiAR`, or `BiAR`, where the `coef` property is updated with the estimated model parameters (using MLE) and the `kalmanlik` property contains the log-likelihood value of the model.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.,Elorrieta, F, Eyheramendy, S, Palma, W (2019). "Discrete-time autoregressive model for unequally spaced time-series observations." *A&A*, **627**, A120. doi:10.1051/00046361/201935560.,Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). "A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series." *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

Examples

```
# Example 1: Applying Kalman filter for MLE of iAR model parameters
library(iAR)
n=100
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_norm <- iAR(family = "norm", times = times, coef = 0.9,hessian=TRUE)
model_norm <- sim(model_norm)
model_norm <- kalman(model_norm)
print(model_norm@coef)  # Access the estimated coefficients
print(model_norm@kalmanlik)  # Access the Kalman likelihood value

# Example 2: Applying Kalman filter for MLE of CiAR model parameters
set.seed(6714)
model_CiAR <- CiAR(times = times,coef = c(0.9, 0))
model_CiAR <- sim(model_CiAR)
```

```
y=model_CiAR@series
y1=y/sd(y)
model_CiAR@series=y1
model_CiAR@series_esd=rep(0,n)
model_CiAR <- kalman(model_CiAR)
print(model_CiAR@coef)
print(model_CiAR@kalmanlik)

# Example 3: Applying Kalman filter for MLE of BiAR model parameters
set.seed(6714)
model_BiAR <- BiAR(times = times,coef = c(0.9, 0.3), rho = 0.9)
model_BiAR <- sim(model_BiAR)
y=model_BiAR@series
y1=y/apply(y,2,sd)
model_BiAR@series=y1
model_BiAR@series_esd=matrix(0,n,2)
model_BiAR <- kalman(model_BiAR)
print(model_BiAR@coef)
print(model_BiAR@kalmanlik)
```

---

loglik — *Maximum Likelihood Estimation for iAR Models*

Description

Maximum Likelihood Estimation for irregular autoregressive (iAR) models, supporting different distribution families: normal (‘iAR‘), t (‘iAR-T‘), and gamma (‘iAR-Gamma‘).

Usage

```
loglik(x, ...)
```

Arguments

x — An object of class `iAR`, containing the model specification, parameters, and the time series to be evaluated:

- `series`: The observed time series.
- `times`: A numeric vector specifying the time points of the series.
- `series_esd`: (Optional) A standardized version of the series.
- `zero_mean`: Logical. Indicates whether the series should be mean-centered.
- `standardized`: Logical. Indicates whether the series is standardized.
- `hessian`: Logical. If `TRUE`, the function computes the Hessian matrix for parameter estimation.
- `family`: The distribution family of the iAR model (one of "norm", "t", or "gamma").
- `df`: Degrees of freedom for the t-distribution (only for `family = "t"`).

- `sigma`: The scale parameter for the t-distribution (only for `family = "t"`).
- `mean`: The mean parameter for the gamma distribution (only for `family = "gamma"`).
- `variance`: The variance parameter for the gamma distribution (only for `family = "gamma"`).

`...` Additional arguments (unused).

Details

This method estimates the parameters of an iAR model using the Maximum Likelihood Estimation (MLE) approach. Depending on the chosen distribution family, the corresponding likelihood function is maximized:

- "norm" maximizes the likelihood for a normally-distributed series.
- "t" maximizes the likelihood for a t-distributed series.
- "gamma" maximizes the likelihood for a gamma-distributed series.

The function updates the `iAR` object with the estimated parameters, the log-likelihood value, and a summary table that includes standard errors and p-values.

Value

An updated `iAR` object with the following additional attributes:

- `coef`: Estimated model coefficients.
- `loglik`: Log-likelihood value of the model.
- `summary`: A summary table containing parameter estimates, standard errors, and p-values.
- `sigma`: For t and gamma families, the estimated scale parameter.
- `mean`: For the gamma family, the estimated mean parameter.
- `variance`: For the gamma family, the estimated variance parameter.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.

Examples

```
# Example: Estimating parameters for a normal iAR model
library(iAR)
times <- 1:100
model <- iAR(family = "norm", times = times, coef = 0.9, hessian = TRUE)
model <- sim(model)  # Simulate the series
model <- loglik(model)  # Estimate parameters using MLE
print(model@coef)  # Access the estimated coefficients
print(model@loglik)  # Access the computed log-likelihood
```

`multidata` *Multidata Class*

Description

The ‘multidata‘ class is an S7 class designed to represent multidimensional time series models, including the main time series and additional series (e.g., error standard deviations or related variables).

Usage

```
multidata(
  times = integer(0),
  series = integer(0),
  series_esd = integer(0),
  series_names = character(0)
)
```

Arguments

`times`	A numeric vector representing the time points of the time series.
`series`	A numeric vector or matrix representing the main time series.
`series_esd`	A numeric vector or matrix representing the additional series, such as error standard deviations or other related data.
`series_names`	An optional character vector representing the name of the series.

Validation Rules

- ‘@times‘ must be a numeric vector and strictly increasing. - ‘@series‘ must be a numeric matrix or empty. Its number of rows must match the length of ‘@times‘. - ‘@series_esd‘ must be a numeric matrix or empty. If provided and not empty, its dimensions must match those of ‘@series‘. - If ‘@series_names‘ is provided, it must be a character vector of the same length as the number of columns of ‘@series‘, and contain unique values.

See Also

[BiAR]

Examples

```
# Create a multidata object
multidata_instance <- multidata(
  times = c(1, 2, 3, 4),
  series = matrix(c(10, 20, 15, 25,
                    12, 18, 17, 23),
                  nrow = 4, ncol = 2),
  series_esd = matrix(c(1, 1.5, 1.2, 1.8,
```

```
                              0.9, 1.3, 1.1, 1.7),
                            nrow = 4, ncol = 2)
)
```

`pairingits` *Pairing two irregularly observed time series*

Description

This method pairs the observational times of two irregularly observed time series.

Usage

```
pairingits(x, ...)
```

Arguments

`x`	An object of class ‘utilities‘.
`...`	Additional arguments for pairing time series: **lc1** A data frame with three columns corresponding to the first irregularly observed time series. **lc2** A data frame with three columns corresponding to the second irregularly observed time series. **tol** A numeric value indicating the tolerance parameter.

Details

The method checks the observational times in both input time series and pairs the measurements if they fall within the specified tolerance (‘tol‘). If a measurement in one series cannot be paired, it is filled with ‘NA‘ values for the corresponding columns of the other series.

Value

An object of class ‘utilities‘ with two slots:

`series`	A matrix containing the paired time series, where unmatched measurements are filled with ‘NA‘.
`series_esd`	A matrix containing the paired error standard deviations of the time series, where unmatched measurements are filled with ‘NA‘.
`times`	A numeric vector with the paired observational times.

References

Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). “A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series.” *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

Examples

```
data(cvnovag)
data(cvnovar)
datag=cvnovag
datar=cvnovar
o1=iAR::utilities()
o1<-pairingits(o1, datag,datar,tol=0.1)
pargr1=na.omit(o1@paired)
st=apply(pargr1[,c(1,4)],1,mean)
model_BiAR <- BiAR(times = st,series=pargr1[,c(2,5)],series_esd=pargr1[,c(3,6)])
model_BiAR <- kalman(model_BiAR)
```

`Planets` *Transit of an extrasolar planet*

Description

Time series corresponding to the residuals of the parametric model fitted by Jordan et al (2013) for a transit of an extrasolar planet.

Usage

```
Planets
```

Format

A data frame with 91 observations on the following 2 variables:

t Time from mid-transit (hours).

r Residuals of the parametric model fitted by Jordan et al (2013).

References

Jordán A, Espinoza N, Rabus M, Eyheramendy S, Sing D~K, Désert J, Bakos G~Á, Fortney J~J, López-Morales M, Maxted P~F~L, Triaud A~H~M~J, Szentgyorgyi A (2013). "A Ground-based Optical Transmission Spectrum of WASP-6b." *The Astrophysical Journal*, **778**, 184. doi:10.1088/0004637X/778/2/184, 1310.6048.

Examples

```
data(Planets)
#plot(Planets[,1],Planets[,2],xlab='Time from mid-transit (hours)',ylab='Noise',pch=20)
```

plot — *Plot Method for unidata Objects*

Description

This method allows visualizing ‘unidata‘ and ‘multidata‘ objects using the ‘plot.zoo‘ function from the ‘zoo‘ package. It converts the ‘unidata‘ (‘multidata‘) object into a ‘zoo‘ object and then applies the plotting function to the time series contained in the object.

Arguments

`x`	An object of class ‘unidata‘ or ‘multidata‘. This object must contain the time series in the ‘series‘ slot and the associated time points in the ‘times‘ slot.
`...`	Additional arguments passed to the ‘plot.zoo‘ function for customizing the plot (e.g., titles, colors, etc.).

plot_fit — *Plot Fit Method for unidata Objects*

Description

This method visualizes the ‘unidata‘ and ‘multidata‘ objects by plotting both the original time series and the fitted values. The ‘unidata‘ (‘multidata‘) object is converted into a ‘zoo‘ object for plotting, and the fitted values are added as a secondary line with a different style and color.

Usage

```
plot_fit(x,...)
```

Arguments

`x`	An object of class ‘unidata‘ or ‘multidata‘. This object must contain the time series in the ‘series‘ slot, the associated time points in the ‘times‘ slot, and the fitted values in the ‘fitted_values‘ slot.
`...`	Additional arguments passed to the ‘plot.zoo‘ function for customizing the plot (e.g., titles, colors, etc.).

plot_forecast *Plot Forecast Method for unidata Objects*

Description

This method visualizes the ‘unidata‘ and ‘multidata‘ objects by plotting both the original time series and the forecasted values. The ‘unidata‘ (‘multidata‘) object is converted into a ‘zoo‘ object for plotting, and the forecasted values are added as points with a different color.

Usage

```
plot_forecast(x,...)
```

Arguments

x	An object of class ‘unidata‘ or ‘multidata‘. This object must contain the time series in the ‘series‘ slot, the associated time points in the ‘times‘ slot, the forecasted values in the ‘forecast‘ slot, and the forecast horizon in the ‘tAhead‘ slot.
...	Additional arguments passed to the ‘plot.zoo‘ function for customizing the plot (e.g., titles, colors, etc.).

sim *Simulate Time Series for Multiple iAR Models*

Description

Simulates a time series for irregular autoregressive (iAR) models, including: 1. Normal iAR model (‘iAR‘) 2. T-distribution iAR model (‘iAR-T‘) 3. Gamma-distribution iAR model (‘iAR-Gamma‘)

Usage

```
sim(x, ...)
```

Arguments

x — An object of class `iAR`, `CiAR`, or `BiAR`, containing the model specification and parameters:

- For `iAR` (irregular AR models), the model family could be "norm", "t", or "gamma", where:
  - `family`: The distribution family of the iAR model (one of "norm", "t", or "gamma").
  - `coef`: The coefficient(s) of the iAR model.
  - `times`: A numeric vector specifying the time points of the series.
  - `df`: Degrees of freedom for the t-distribution (only for `family = "t"`).

- `sigma`: The scale parameter for the t-distribution (only for `family = "t"`).
- `mean`: The mean parameter for the gamma distribution (only for `family = "gamma"`).
- `variance`: The variance parameter for the gamma distribution (only for `family = "gamma"`).

• For `CiAR` (complex irregular autoregressive models):
  - `coef`: The real and imaginary parts of the CiAR model's coefficients.
  - `times`: A numeric vector specifying the time points of the series.
  - `rho`: The correlation parameter for the CiAR model.
  - `c`: The scale parameter for the CiAR model.

• For `BiAR` (BiAR models):
  - `coef`: The coefficients of the BiAR model (real and imaginary).
  - `times`: A numeric vector specifying the time points of the series.
  - `rho`: The correlation parameter for the BiAR model.
  - `series_esd`: The series for the error structure (optional, used internally).

`...` Additional arguments for generating irregular times. This is used only if no time points have been provided in the `times` argument:

n A positive integer. Length of observation times. Default is 100. This is used only if no time points are provided in `times`.

distribution A character string specifying the distribution of the observation times. Default is ‘"expmixture"‘. Available options are:

expmixture A mixture of two exponential distributions.

uniform A uniform distribution.

exponential A single exponential distribution.

gamma A gamma distribution.

lambda1 Mean (1/rate) of the exponential distribution or the first exponential distribution in a mixture of exponential distributions. Default is ‘130‘.

lambda2 Mean (1/rate) of the second exponential distribution in a mixture of exponential distributions. Default is ‘6.5‘.

p1 Weight of the first exponential distribution in a mixture of exponential distributions. Default is ‘0.15‘.

p2 Weight of the second exponential distribution in a mixture of exponential distributions. Default is ‘0.85‘.

a Shape parameter of a gamma distribution or lower limit of the uniform distribution. Default is ‘0‘.

b Scale parameter of a gamma distribution or upper limit of the uniform distribution. Default is ‘1‘.

Details

This function simulates time series based on the specified model and its parameters. Depending on the class of the input object:

- For `iAR` models, it supports three distribution families:
- "norm" for normal distribution.
- "t" for t-distribution.
- "gamma" for gamma distribution.
- For `CiAR` models, it uses complex autoregressive processes to generate the time series.
- For `BiAR` models, it simulates a BiAR process using specified coefficients and correlation.

The coefficients and any family-specific parameters must be set before calling this function.

Value

An updated object of class `iAR`, `CiAR`, or `BiAR`, where the `series` property contains the simulated time series.

References

Eyheramendy S, Elorrieta F, Palma W (2018). "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves." *Monthly Notices of the Royal Astronomical Society*, **481**(4), 4311-4322. ISSN 0035-8711, doi:10.1093/mnras/sty2487, https://academic.oup.com/mnras/article-pdf/481/4/4311/25906473/sty2487.pdf.,Elorrieta, F, Eyheramendy, S, Palma, W (2019). "Discrete-time autoregressive model for unequally spaced time-series observations." *A&A*, **627**, A120. doi:10.1051/00046361/201935560.,Elorrieta F, Eyheramendy S, Palma W, Ojeda C (2021). "A novel bivariate autoregressive model for predicting and forecasting irregularly observed time series." *Monthly Notices of the Royal Astronomical Society*, **505**(1), 1105-1116. ISSN 0035-8711, doi:10.1093/mnras/stab1216, https://academic.oup.com/mnras/article-pdf/505/1/1105/38391762/stab1216.pdf.

Examples

```
# Example 1: Simulating a normal iAR model
library(iAR)
n=100
set.seed(6714)
o=iAR::utilities()
o<-gentime(o, n=n)
times=o@times
model_norm <- iAR(family = "norm", times = times, coef = 0.9,hessian=TRUE)
model_norm <- sim(model_norm)
plot(model_norm, type = "l", main = "Simulated iAR-Norm Series")

# Example 2: Simulating a CiAR model
set.seed(6714)
model_CiAR <- CiAR(times = times,coef = c(0.9, 0))
model_CiAR <- sim(model_CiAR)
plot(model_CiAR , type = "l", main = "Simulated CiAR Series")

# Example 3: Simulating a BiAR model
set.seed(6714)
model_BiAR <- BiAR(times = times,coef = c(0.9, 0.3), rho = 0.9)
model_BiAR <- sim(model_BiAR)
plot(times, model_BiAR@series[,1], type = "l", main = "Simulated BiAR Series")
```

summary — *Summary Method for unidata Objects*

Description

This method provides a summary of the ‘unidata‘ object, which represents an iAR (irregular autoregressive) model. The summary includes information about the model family (e.g., "normal", "t", "gamma"), the coefficients, standard errors, t-values, and p-values. The output is formatted to display the relevant statistics based on the model family.

Arguments

`object`	An object of class ‘unidata‘. This object contains the fitted iAR model, including parameters such as coefficients (‘coef‘), standard errors (‘stderr‘), t-values (‘tvalue‘), p-values (‘pvalue‘), family type (‘family‘), and other model-specific parameters.
`...`	Additional arguments (unused).

unidata — *Unidata Class*

Description

The ‘unidata‘ class is an S7 class designed to represent univariate irregularly observed time series models with associated times, values, and optional error standard deviations.

Usage

```
unidata(
  times = integer(0),
  series = integer(0),
  series_esd = integer(0),
  series_names = character(0)
)
```

Arguments

`times`	A numeric vector representing the time points.
`series`	A numeric vector representing the values of the time series.
`series_esd`	A numeric vector representing the error standard deviations of the time series.
`series_names`	An optional character vector of length 1 representing the name of the series.

Validation Rules

- ‘@times‘, ‘@series‘, and ‘@series_esd‘ must be numeric vectors. - ‘@times‘ must not contain ‘NA‘ values and must be strictly increasing. - The length of ‘@series‘ must match the length of ‘@times‘. - The length of ‘@series_esd‘ must be 0, 1, or equal to the length of ‘@series‘. - ‘NA‘ values in ‘@series‘ must correspond exactly (positionally) to ‘NA‘ values in ‘@series_esd‘. - ‘@series_names‘, if provided, must be a character vector of length 1.

See Also

[iAR], [CiAR]

Examples

```
# Create a unidata object
unidata_instance <- unidata(
  times = c(1, 2, 3, 4),
  series = c(10, 20, 15, 25),
  series_esd = c(1, 1.5, 1.2, 1.8),
  series_names = "my_series")
```

---

utilities *Utilities Class*

Description

The ‘utilities‘ class is an S7 class designed to group utility functions that are not directly tied to other specific objects in the package. These include the functions ‘gentime‘, ‘paringits‘, ‘harmonicfit‘, and ‘foldlc‘.

Usage

```
utilities(
  times = integer(0),
  series = integer(0),
  series_esd = integer(0),
  paired = integer(0)
)
```

Arguments

times	A numeric vector storing the time points.
series	A numeric vector representing the values of the time series.
series_esd	A numeric vector representing the error standard deviations of the time series.
paired	Data Frame with the paired datasets.

Description

This class acts as a container for standalone methods that perform independent operations within the package. By grouping them under a single class, the package achieves better modularity and organization, facilitating maintenance and extensibility.

Available Methods

- ‘gentime‘: Generates time points based on a specified statistical distribution. - ‘paringits‘: A method for pairing irregular time series. - ‘harmonicfit‘: A method for fitting harmonic models to data. - ‘foldlc‘: A method for folding light curves in time series analysis.

Examples

```
# Create a utilities object
```

Index