Package 'PTSR'

October 12, 2022

Type Package

Date 2022-02-07

Title Positive Time Series Regression

Version 0.1.2

Depends R (>= 4.0.0)

Imports extraDistr, SuppDists, actuar, numDeriv

Description A collection of functions to simulate, estimate and forecast a wide range of regression based dynamic models for positive time series.
This package implements the results presented in Prass, T.S.; Carlos, J.H.; Taufemback, C.G. and Pumi, G. (2022). ``Positive Time Series Regression" <arXiv:2201.03667>.

License GPL (>= 3)

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 7.1.2

Author Taiane Schaedler Prass [aut, cre, com]
(<https://orcid.org/0000-0003-3136-909X>),
Jonas Hendler Carlos [aut],
Cleiton Guollo Taufemback [aut]

Maintainer Taiane Schaedler Prass <taianeprass@gmail.com>

Repository CRAN

Date/Publication 2022-02-08 07:20:05 UTC

R topics documented:

ddist 2
predict.ptsr 4
print.ptsr 5
ptsr.fit 5
ptsr.link 7
ptsr.sim 8
summary 10

Index **12**

ddist *Reparametrized Distributions*

Description

Density function and random numbers generation for models with support on the positive real line.

Usage

```
d.betap(x, mu, varphi, log = FALSE)

r.betap(n, mu, varphi)

d.F(x, mu, varphi, log = FALSE)

r.F(n, mu, varphi)

d.gamma(x, mu, varphi, log = FALSE)

r.gamma(n, mu, varphi)

d.invGauss(x, mu, varphi, log = FALSE)

r.invGauss(n, mu, varphi)

d.logLogis(x, mu, varphi, log = FALSE)

r.logLogis(n, mu, varphi)

d.logNorm(x, mu, varphi, log = FALSE)

r.logNorm(n, mu, varphi)

d.chi(x, mu, log = FALSE, ...)

r.chi(n, mu, ...)

d.ray(x, mu, log = FALSE, ...)

r.ray(n, mu, ...)
```

Arguments

x	vector of real values
mu	non-negative parameter (the distribution's mean. See 'Details')
varphi	non-negative parameter

log	logical; if TRUE, probabilities p are given as $log(p)$.
n	sample size
...	for compatibility with other functions

Details

- For the reparametrized Beta-Prime distribution, the functions dbetapr and rbetapr are imported from the package extraDistr. The following holds

$$shape1 = mu * varphi$$
$$shape2 = varphi + 1$$
$$scale = 1$$

- For the reparametrized F distribution, the functions df and rf are imported from stats. The following holds

$$df1 = varphi$$
$$df2 = 2 * mu/(mu - 1)$$

  so that the parameter μ must satisfy $\mu > 1$.

- For the reparametrized Gamma distribution, the functions dgamma and rgamma are imported from stats. The following holds

$$shape = varphi$$
$$rate = varphi/mu$$

- For the reparametrized Inverse Gaussian distribution, the functions dinvGauss and rinvGauss are imported from SuppDists. The following holds

$$nu = mu$$
$$lambda = 1/varphi$$

- For the reparametrized Log-logistic distribution, the functions dllogis and rllogis a are imported from actuar. The following holds

$$shape = varphi$$
$$rate = (pi/varphi)/(mu * sin(pi/varphi))$$

- For the reparametrized Log-Normal distribution, the functions dlnorm and rlnorm are imported from stats. The following holds

$$meanlog = log(mu) - varphi^2/2$$
$$sdlog = varphi$$

- For the reparametrized Chi-squared F distribution, the functions dchisq and rchisq are imported from stats. The following holds

$$df = mu$$

- For the reparametrized Rayleigh distribution, the functions drayleigh and rrayleigh are imported from extraDistr. The following holds

$$sigma = mu/sqrt(pi/2)$$

Value

For any avaliable `dist`, `ddist` gives the density and `rdist` generates random deviates.

The length of the result is determined by `n` for `rdist`, and is the maximum of the lengths of the numerical arguments for `rdist`.

The numerical arguments other than `n` are recycled to the length of the result. Only the first elements of the logical arguments are used.

`predict.ptsr` *Predict method for PTSR*

Description

Predicted values based on ptsr object.

Usage

```
## S3 method for class 'ptsr'
predict(object, newdata, nnew = 0, ...)
```

Arguments

`object`	Object of class inheriting from `"ptsr"`
`newdata`	A matrix with new values for the regressors. If omitted and `"xreg"` is present in the model, the fitted values are returned. If the model does not include regressors, the functions will use the value of `nnew`.
`nnew`	number of out-of-sample forecasts required. If `newdata` is provided, `nnew` is ignored.
`...`	further arguments passed to or from other methods.

Details

`predict.ptsr` produces predicted values, obtained by evaluating the regression function in the frame `newdata`.

If `newdata` is omitted the predictions are based on the data used for the fit.

For now, prediction intervals are not provided.

Value

A list with the following arguments

`series`	The original time series yt.
`xreg`	The original regressors (if any).
`fitted.values`	The in-sample forecast given by μ_t.
`etat`	In-sample values of $g(\mu[t])$.

error	The error term
residuals	The (in-sample) residuals, that is, the observed minus the predicted values.
forecast	The predicted values for yt.

`print.ptsr` *Print Method of class PTSR*

Description

Print method for objects of class `ptsr`.

Usage

```
## S3 method for class 'ptsr'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	object of class `ptsr`.
digits	minimal number of significant digits, see `print.default`.
...	further arguments to be passed to or from other methods. They are ignored in this function

Details

Users are not encouraged to call these internal functions directly. Internal functions for package PTSR.

Value

Invisibly returns its argument, `x`.

`ptsr.fit` *Title Function to fit a PTSR model*

Description

Title Function to fit a PTSR model

Usage

```
ptsr.fit(start, yt, xreg = NULL, xregar = TRUE, fit.alpha = TRUE,
  p = 0, q = 0, arlag = NULL, malag = NULL, ddist = d.gamma,
  link1 = "log", link2 = "identity", g1 = NULL, g1.inv = NULL,
  g2 = NULL, method = "L-BFGS-B", ...)
```

Arguments

start	a vector with the starting values for the non-fixed coefficients of the model.
yt	the time series
xreg	optionally, a vector or matrix of external regressors. Default is NULL
xregar	logical, if FALSE, the regressors are not included in the AR component of the model. Default is TRUE.
fit.alpha	logical, if FALSE, alpha is set to zero. Default is TRUE
p	order of the AR polinomial
q	order of the MA polinomial
arlag	the lags to be included in the AR polinomial. Default is NULL, meaning that all lags will be included.
malag	the lags to be included in the MA polinomial. Default is NULL, meaning that all lags will be included.
ddist	function, the density function to be used
link1	character indicating which link must be used for μ_t. See ptsr.link for available links. Default is 'log'.
link2	character indicating which link must be used for y_t in the AR recursion. See ptsr.link for available links. Default is 'identity'.
g1	optionally, a link function to be used for μ_t. Default is NULL, so that it is calculated internally, using link1.
g1.inv	optionally, a the inverse link function to be used for η_t. It must the the ivnerse of g1. Default is NULL, so that it is calculated internally, using link1.
g2	optionally, a link function to be used for y_t. Default is NULL, so that it is calculated internally, using link2.
method	The method to be used. See [optim][stats::optim] for details.
...	Further arguments to be passed to optim.

Value

The same arguments return by optim plus a the following arguments

- coefficients: a vector with the estimated coefficients;
- sll: the sum of the log-likelihood for the fitted model;
- series: the original time series;
- xreg: the regressors (if any);
- fitted.values: the conditional mean, which corresponds to the in-sample forecast, also denoted fitted values;
- residuals: the observed minus the fitted values;
- model: a list with the configurations used to fit the model.

Examples

```
#-------------------------------------------------------------------
# Gamma-ARMA(1,1) model with no regressors
#-------------------------------------------------------------------

simu = ptsr.sim(n = 3000, burn = 50,
                varphi = 20, alpha = 0,
                phi = 0.35, theta = 0.2,
                seed = 1234, rdist = r.gamma,
                link1 = "log", link2 = "log")

fit1 = ptsr.fit(start =  c(0,0,0,10), yt = simu$yt,
               fit.alpha = TRUE, p = 1, q = 1,
               ddist = d.gamma, link1 = "log",
               link2 = "log", method = "L-BFGS-B")
summary(fit1)

# removing alpha from the model
fit2 = ptsr.fit(start =  c(0,0,10), yt = simu$yt,
               fit.alpha = FALSE, p = 1, q = 1,
               ddist = d.gamma, link1 = "log",
               link2 = "log", method = "L-BFGS-B")
summary(fit2)
```

`ptsr.link` *Create a Link for PTSR models*

Description

Given the name of a link, this function returns a link function, an inverse link function, the derivative $d\eta/d\mu$ and the derivative $d\mu/d\eta$.

Usage

```
ptsr.link(link = "log")
```

Arguments

`link`	character; one of `"log"`, `"log1"`. See 'Details'.

Details

The available links are:

log: $f(x) = log(x)$

log1: $f(x) = log(x - 1)$

Value

An object of class "link-ptsr", a list with components

linkfun	Link function function(mu)
linkinv	Inverse link function function(eta)
linkdif	Derivative function(mu) $d\eta/d\mu$
mu.eta	Derivative function(eta) $d\mu/d\eta$
name	a name to be used for the link

---

ptsr.sim *Function to simulate a PTSR model*

Description

Function to simulate a PTSR model

Usage

```
ptsr.sim(n = 1, burn = 0, xreg = NULL, xregar = TRUE, varphi = 1,
  alpha = 0, beta = NULL, phi = NULL, theta = NULL,
  seed = stats::runif(1, 1000, 10000), rdist = r.gamma, link1 = "log",
  link2 = "identity", g1 = NULL, g1.inv = NULL, g2 = NULL)
```

Arguments

n	a strictly positive integer. The sample size of yt (after burn-in). Default is 1.
burn	a non-negative integer. length of "burn-in" period. Default is 0.
xreg	optionally, a vector or matrix of external regressors. For simulation purposes, the length of xreg must be n+burn. Default is NULL.
xregar	logical, if FALSE, the regressors are not included in the AR component of the model. Default is TRUE.
varphi	non-negative parameter. Default is 1.
alpha	a numeric value corresponding to the intercept. Default is 0.
beta	optionally, a vector of coefficients corresponding to the regressors in xreg. Default is NULL.
phi	optionally, for the simulation function this must be a vector of size p, corresponding to the autoregressive coefficients (including the ones that are zero), where p is the AR order. Default is NULL.
theta	optionally, for the simulation function this must be a vector of size q, corresponding to the moving average coefficients (including the ones that are zero), where q is the MA order. Default is NULL. that $g_2(y_t) = 0$, for $t < 1$.

seed	optionally, an integer which gives the value of the fixed seed to be used by the random number generator. If missing, a random integer is chosen uniformly from 1,000 to 10,000.
rdist	function, the random number generator to be used
link1	character indicating which link must be used for μ_t. See ptsr.link for available links. Default is 'log'.
link2	character indicating which link must be used for y_t in the AR recursion. See ptsr.link for available links. Default is 'identity'.
g1	optionally, a link function to be used for μ_t. Default is NULL, so that it is calculated internally, using link1.
g1.inv	optionally, a the inverse link function to be used for η_t. It must the the ivnerse of g1. Default is NULL, so that it is calculated internally, using link1.
g2	optionally, a link function to be used for y_t. Default is NULL, so that it is calculated internally, using link2.

Details

The function ptsr.sim generates a random sample from a positive time series regression model, with a given distribution.

Value

Returns a list with the following components

- yt: the simulated time series
- mut: the conditional mean
- etat: the linear predictor $g(\mu_t)$
- error: the error term.

Examples

```
#---------------------------------------------------------------------
# Generating a sample of a Gamma-ARMA(1,1) model with no regressors
#---------------------------------------------------------------------

simu = ptsr.sim(n = 300, burn = 50,
                varphi = 20, alpha = 0,
                phi = 0.35, theta = 0.2,
                seed = 1234, rdist = r.gamma,
                link1 = "log", link2 = "log")

names(simu)
plot.ts(simu$yt)
lines(simu$mut, col= "red")
```

summary *Summary Method of class PTSR*

Description

summary method for class "ptsr".

Usage

```
## S3 method for class 'ptsr'
summary(object, ...)

## S3 method for class 'summary.ptsr'
print(x, digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"), ...)
```

Arguments

object	object of class "ptsr".
...	further arguments passed to or from other methods.
x	an object of class "summary.ptsr", usually, a result of a call to summary.ptsr.
digits	minimal number of significant digits, see print.default.
signif.stars	logical. If TRUE, 'significance stars' are printed for each coefficient.

Details

print.summary.btsr tries to be smart about formatting the coefficients, standard errors, etc. and additionally provides 'significance stars'.

Value

The function summary.ptsr computes and returns a list of summary statistics of the fitted model given in object. Returns a list of class summary.ptsr, which contains the following components:

residuals	the residuals of the model.
coefficients	a $k \times 4$ matrix with columns for the estimated coefficient, its standard error, z-statistic and corresponding (two-sided) p-value.
sigma.res	the square root of the estimated variance of the random error $$\hat{\sigma}^2 = \frac{1}{n-k}\sum_i r_i^2,$$ where r_i is the i-th residual, residuals[i].
df	degrees of freedom, a 3-vector $(k, n-k, k*)$, the first being the number of non-aliased coefficients, the last being the total number of coefficients.

`vcov`	a $k \times k$ matrix of (unscaled) covariances. The inverse ov the information matrix.
`loglik`	the sum of the log-likelihood values
`aic`	the AIC value. $AIC = -2 * loglik + 2 * k$.
`bic`	the BIC value. $BIC = -2 * loglik + log(n) * k$.
`hqc`	the HQC value. $HQC = -2 * loglik + log(log(n)) * k$.

Index

d.betap *(ddist)*, 2
d.chi *(ddist)*, 2
d.F *(ddist)*, 2
d.gamma *(ddist)*, 2
d.invGauss *(ddist)*, 2
d.logLogis *(ddist)*, 2
d.logNorm *(ddist)*, 2
d.ray *(ddist)*, 2
dbetapr, *3*
dchisq, *3*
ddist, 2
df, *3*
dgamma, *3*
dinvGauss, *3*
dllogis, *3*
dlnorm, *3*
drayleigh, *3*

extraDistr, *3*

predict.ptsr, 4
print.default, *5*, *10*
print.ptsr, 5
print.summary.ptsr *(summary)*, 10
ptsr.fit, 5
ptsr.link, *6*, 7, *9*
ptsr.sim, 8

r.betap *(ddist)*, 2
r.chi *(ddist)*, 2
r.F *(ddist)*, 2
r.gamma *(ddist)*, 2
r.invGauss *(ddist)*, 2
r.logLogis *(ddist)*, 2
r.logNorm *(ddist)*, 2
r.ray *(ddist)*, 2
rbetapr, *3*
rchisq, *3*
rdist *(ddist)*, 2
rf, *3*
rgamma, *3*
rinvGauss, *3*
rllogis, *3*
rlnorm, *3*
rrayleigh, *3*

stats, *3*
summary, 10