

# Package: itsmr (via r-universe)

October 31, 2024

**Type** Package

**Title** Time Series Analysis Using the Innovations Algorithm

**Version** 1.10

**Date** 2022-07-27

**Author** George Weigt

**Maintainer** George Weigt <g808391@icloud.com>

**Description** Provides functions for modeling and forecasting time series data. Forecasting is based on the innovations algorithm. A description of the innovations algorithm can be found in the textbook ``Introduction to Time Series and Forecasting" by Peter J. Brockwell and Richard A. Davis.  
<<https://link.springer.com/book/10.1007/b97391>>.

**License** FreeBSD

**LazyLoad** yes

**NeedsCompilation** no

**URL** <https://georgeweigt.github.io/itsmr-refman.pdf>

**Date/Publication** 2022-08-06 06:10:02 UTC

**Repository** <https://georgeweigt.r-universe.dev>

**RemoteUrl** <https://github.com/cran/itsmr>

**RemoteRef** HEAD

**RemoteSha** 8312b117d05799dcba45e2a919601cb876d067ce

## Contents

itsmr-package . . . . .	2
aacvf . . . . .	4
acvf . . . . .	5
airpass . . . . .	5
ar.inf . . . . .	6
arar . . . . .	7

arma . . . . .	8
autofit . . . . .	9
burg . . . . .	10
check . . . . .	11
deaths . . . . .	11
dowj . . . . .	12
forecast . . . . .	12
hannan . . . . .	13
hr . . . . .	14
ia . . . . .	15
lake . . . . .	16
ma.inf . . . . .	16
periodogram . . . . .	17
plota . . . . .	18
plotc . . . . .	18
plots . . . . .	19
Resid . . . . .	19
season . . . . .	21
selftest . . . . .	21
sim . . . . .	22
smooth.exp . . . . .	23
smooth.fft . . . . .	23
smooth.ma . . . . .	24
smooth.rank . . . . .	25
specify . . . . .	25
strikes . . . . .	26
Sunspots . . . . .	26
test . . . . .	27
trend . . . . .	28
wine . . . . .	28
yw . . . . .	29
<b>Index</b>	<b>30</b>

---

 itsmr-package

*Time Series Analysis Using the Innovations Algorithm*


---

### Description

Provides functions for modeling and forecasting time series data. Forecasting is based on the innovations algorithm. A description of the innovations algorithm can be found in the textbook *Introduction to Time Series and Forecasting* by Peter J. Brockwell and Richard A. Davis.

## Details

Package: itsmr  
Type: Package  
Version: 1.10  
Date: 2022-07-27  
License: FreeBSD  
LazyLoad: yes  
URL: <https://georgeweigt.github.io/itsmr-refman.pdf>

## Author(s)

George Weigt  
Maintainer: George Weigt <g808391@icloud.com>

## References

Brockwell, Peter J., and Richard A. Davis. *Introduction to Time Series and Forecasting*. 2nd ed. Springer, 2002.

## Examples

```
plotc(wine)

## Define a suitable data model
M = c("log", "season", 12, "trend", 1)

## Obtain residuals and check for stationarity
e = Resid(wine, M)
test(e)

## Define a suitable ARMA model
a = arma(e, p=1, q=1)

## Obtain residuals and check for white noise
ee = Resid(wine, M, a)
test(ee)

## Forecast future values
forecast(wine, M, a)
```

---

aacvf                      *Autocovariance of ARMA model*

---

**Description**

Autocovariance of ARMA model

**Usage**

```
aacvf(a, h)
```

**Arguments**

a	ARMA model
h	Maximum lag

**Details**

The ARMA model is a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Value**

Returns a vector of length h+1 to accomodate lag 0 at index 1.

**See Also**

[arma](#)

**Examples**

```
a = arma(Sunspots,2,0)
aacvf(a,40)
```

---

acvf	<i>Autocovariance of data</i>
------	-------------------------------

---

**Description**

Autocovariance of data

**Usage**

```
acvf(x, h = 40)
```

**Arguments**

x	Time series data
h	Maximum lag

**Value**

Returns a vector of length  $h+1$  to accomodate lag 0 at index 1.

**See Also**

[plota](#)

**Examples**

```
acvf(Sunspots)
```

---

airpass	<i>Number of international airline passengers, 1949 to 1960</i>
---------	---

---

**Description**

Number of international airline passengers, 1949 to 1960

**Examples**

```
plotc(airpass)
```

---

ar.inf                      *Compute AR infinity coefficients*

---

**Description**

Compute AR infinity coefficients

**Usage**

```
ar.inf(a, n = 50)
```

**Arguments**

a	ARMA model
n	Order

**Details**

The ARMA model is a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Value**

Returns a vector of length n+1 to accomodate coefficient 0 at index 1.

**See Also**

[ma.inf](#)

**Examples**

```
a = yw(Sunspots,2)
ar.inf(a)
```

---

arar *Forecast using ARAR algorithm*

---

**Description**

Forecast using ARAR algorithm

**Usage**

```
arar(y, h = 10, opt = 2)
```

**Arguments**

y	Time series data
h	Steps ahead
opt	Display option (0 silent, 1 tabulate, 2 plot and tabulate)

**Value**

Returns the following list invisibly.

pred	Predicted values
se	Standard errors
l	Lower bounds (95% confidence interval)
u	Upper bounds

**See Also**

[forecast](#)

**Examples**

```
arar(airpass)
```

---

`arma`*Estimate ARMA model coefficients using maximum likelihood*

---

**Description**

Estimate ARMA model coefficients using maximum likelihood

**Usage**

```
arma(x, p = 0, q = 0)
```

**Arguments**

<code>x</code>	Time series data
<code>p</code>	AR order
<code>q</code>	MA order

**Details**

Calls the standard R function `arima` to estimate AR and MA coefficients. The innovations algorithm is used to estimate white noise variance.

**Value**

Returns an ARMA model consisting of a list with the following components.

<code>phi</code>	Vector of AR coefficients (index number equals coefficient subscript)
<code>theta</code>	Vector of MA coefficients (index number equals coefficient subscript)
<code>sigma2</code>	White noise variance
<code>aicc</code>	Akaike information criterion corrected
<code>se.phi</code>	Standard errors for the AR coefficients
<code>se.theta</code>	Standard errors for the MA coefficients

**See Also**

[autofit](#) [burg](#) [hannan](#) [ia](#) [yw](#)

**Examples**

```
M = c("diff",1)
e = Resid(dowj,M)
a = arma(e,1,0)
print(a)
```



---

`autofit`*Find the best model from a range of possible ARMA models*

---

**Description**

Find the best model from a range of possible ARMA models

**Usage**

```
autofit(x, p = 0:5, q = 0:5)
```

**Arguments**

<code>x</code>	Time series data (typically residuals from <code>Resid</code> )
<code>p</code>	Range of AR orders
<code>q</code>	Range of MA orders

**Details**

Tries all combinations of `p` and `q` and returns the model with the lowest AICC. The arguments `p` and `q` should be small ranges as this function can be slow otherwise. The innovations algorithm is used to estimate white noise variance.

**Value**

Returns an ARMA model consisting of a list with the following components.

<code>phi</code>	Vector of AR coefficients (index number equals coefficient subscript)
<code>theta</code>	Vector of MA coefficients (index number equals coefficient subscript)
<code>sigma2</code>	White noise variance
<code>aicc</code>	Akaike information criterion corrected
<code>se.phi</code>	Standard errors for the AR coefficients
<code>se.theta</code>	Standard errors for the MA coefficients

**See Also**

[arma](#)

**Examples**

```
M = c("diff",1)
e = Resid(dowj,M)
a = autofit(e)
print(a)
```

---

**burg***Estimate AR coefficients using the Burg method*

---

**Description**

Estimate AR coefficients using the Burg method

**Usage**

```
burg(x, p)
```

**Arguments**

x	Time series data (typically residuals from Resid)
p	AR order

**Details**

The innovations algorithm is used to estimate white noise variance.

**Value**

Returns an ARMA model consisting of a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	0
sigma2	White noise variance
aicc	Akaike information criterion corrected
se.phi	Standard errors for the AR coefficients
se.theta	0

**See Also**

[arma hannan ia yw](#)

**Examples**

```
M = c("diff",1)
e = Resid(dowj,M)
a = burg(e,1)
print(a)
```

---

check	<i>Check for causality and invertibility</i>
-------	--

---

**Description**

Check for causality and invertibility

**Usage**

check(a)

**Arguments**

a                    ARMA model

**Details**

The ARMA model is a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Value**

None

**Examples**

```
a = specify(ar=c(0,0,.99))
check(a)
```

---

deaths	<i>USA accidental deaths, 1973 to 1978</i>
--------	--

---

**Description**

USA accidental deaths, 1973 to 1978

**Examples**

```
plotc(deaths)
```

---

 dowj

*Dow Jones utilities index, August 28 to December 18, 1972*


---

**Description**

Dow Jones utilities index, August 28 to December 18, 1972

**Examples**

```
plotc(dowj)
```

---

forecast

*Forecast future values*


---

**Description**

Forecast future values

**Usage**

```
forecast(x, M, a, h = 10, opt = 2, alpha = 0.05)
```

**Arguments**

x	Time series data
M	Data model
a	ARMA model
h	Steps ahead
opt	Display option (0 silent, 1 tabulate, 2 plot and tabulate)
alpha	Level of significance

**Details**

The data model can be NULL for none. Otherwise M is a vector of function names and arguments.

Example:

```
M = c("log", "season", 12, "trend", 1)
```

The above model takes the log of the data, then subtracts a seasonal component of period 12, then subtracts a linear trend component.

These are the available functions:

diff	Difference the data. Has a single argument, the lag.
hr	Subtract harmonic components. Has one or more arguments, each specifying the number of observations per harmonic.

log        Take the log of the data, has no arguments.  
 season    Subtract a seasonal component. Has a single argument, the number of observations per season.  
 trend     Subtract a trend component. Has a single argument, the order of the trend (1 linear, 2 quadratic, etc.)

At the end of the model there is an implicit subtraction of the mean operation. Hence the resulting time series always has zero mean.

All of the functions are inverted before the forecast results are displayed.

### Value

Returns the following list invisibly.

pred	Predicted values
se	Standard errors (not returned for data models with log)
l	Lower bounds (95% confidence interval)
u	Upper bounds

### See Also

[arma Resid test](#)

### Examples

```
M = c("log", "season", 12, "trend", 1)
e = Resid(wine, M)
a = arma(e, 1, 1)
forecast(wine, M, a)
```

---

hannan

*Estimate ARMA coefficients using the Hannan-Rissanen algorithm*

---

### Description

Estimate ARMA coefficients using the Hannan-Rissanen algorithm

### Usage

```
hannan(x, p, q)
```

### Arguments

x	Time series data (typically residuals from Resid)
p	AR order
q	MA order ( $q > 0$ )

**Details**

The innovations algorithm is used to estimate white noise variance.

**Value**

Returns an ARMA model consisting of a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance
aicc	Akaike information criterion corrected
se.phi	Standard errors for the AR coefficients
se.theta	Standard errors for the MA coefficients

**See Also**

[arma burg ia yw](#)

**Examples**

```
M = c("diff",12)
e = Resid(deaths,M)
a = hannan(e,1,1)
print(a)
```

---

 hr

---

*Estimate harmonic components*


---

**Description**

Estimate harmonic components

**Usage**

```
hr(x, d)
```

**Arguments**

x	Time series data
d	Vector of harmonic periods

**Value**

Returns a vector the same length as x. Subtract from x to obtain residuals.

**Examples**

```
y = hr(deaths,c(12,6))
plotc(deaths,y)
```

---

ia *Estimate MA coefficients using the innovations algorithm*

---

**Description**

Estimate MA coefficients using the innovations algorithm

**Usage**

```
ia(x, q, m = 17)
```

**Arguments**

x	Time series data (typically residuals from Resid)
q	MA order
m	Recursion level

**Details**

Normally *m* should be set to the default value. The innovations algorithm is used to estimate white noise variance.

**Value**

Returns an ARMA model consisting of a list with the following components.

phi	0
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance
aicc	Akaike information criterion corrected
se.phi	0
se.theta	Standard errors for the MA coefficients

**See Also**

[arma burg hannan yw](#)

**Examples**

```
M = c("diff",1)
e = Resid(dowj,M)
a = ia(e,1)
print(a)
```

---

lake	<i>Level of Lake Huron, 1875 to 1972</i>
------	--

---

**Description**

Level of Lake Huron, 1875 to 1972

**Examples**

```
plotc(lake)
```

---

ma.inf	<i>Compute MA infinity coefficients</i>
--------	---

---

**Description**

Compute MA infinity coefficients

**Usage**

```
ma.inf(a, n = 50)
```

**Arguments**

a	ARMA model
n	Order

**Details**

The ARMA model is a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Value**

Returns a vector of length n+1 to accomodate coefficient 0 at index 1.

**See Also**

[ar.inf](#)



**Examples**

```
M = c("diff", 12)
e = Resid(deaths, M)
a = arma(e, 1, 1)
ma.inf(a, 10)
```

---

periodogram

*Plot a periodogram*

---

**Description**

Plot a periodogram

**Usage**

```
periodogram(x, q = 0, opt = 2)
```

**Arguments**

x	Time series data
q	MA filter order
opt	Plot option (0 silent, 1 periodogram only, 2 periodogram and filter)

**Details**

The filter q can be a vector in which case the overall filter is the composition of MA filters of the designated orders.

**Value**

The periodogram vector divided by  $2\pi$  is returned invisibly.

**See Also**

[plots](#)

**Examples**

```
periodogram(Sunspots, c(1, 1, 1, 1))
```

---

plota *Plot data and/or model ACF and PACF*

---

**Description**

Plot data and/or model ACF and PACF

**Usage**

```
plota(u, v = NULL, h = 40)
```

**Arguments**

u, v	Data and/or ARMA model in either order
h	Maximum lag

**Value**

None

**Examples**

```
plota(Sunspots)
a = yw(Sunspots, 2)
plota(Sunspots, a)
```

---

plotc *Plot one or two time series*

---

**Description**

Plot one or two time series

**Usage**

```
plotc(y1, y2 = NULL)
```

**Arguments**

y1	Data vector (plotted in blue with knots)
y2	Data vector (plotted in red, no knots)

**Value**

None

**Examples**

```
plotc(uspop)
y = trend(uspop,2)
plotc(uspop,y)
```

---

plots	<i>Plot spectrum of data or ARMA model</i>
-------	--

---

**Description**

Plot spectrum of data or ARMA model

**Usage**

```
plots(u)
```

**Arguments**

u                      Data vector or an ARMA model

**Value**

None

**See Also**

[periodogram](#)

**Examples**

```
a = specify(ar=c(0,0,.99))
plots(a)
```

---

Resid	<i>Compute residuals</i>
-------	--------------------------

---

**Description**

Compute residuals

**Usage**

```
Resid(x, M = NULL, a = NULL)
```

**Arguments**

x	Time series data
M	Data model
a	ARMA model

**Details**

The data model can be NULL for none. Otherwise M is a vector of function names and arguments.

Example:

```
M = c("log", "season", 12, "trend", 1)
```

The above model takes the log of the data, then subtracts a seasonal component of period 12, then subtracts a linear trend component.

These are the available functions:

diff	Difference the data. Has a single argument, the lag.
hr	Subtract harmonic components. Has one or more arguments, each specifying the number of observations per harmonic.
log	Take the log of the data, has no arguments.
season	Subtract a seasonal component. Has a single argument, the number of observations per season.
trend	Subtract a trend component. Has a single argument, the order of the trend (1 linear, 2 quadratic, etc.)

At the end of the model there is an implicit subtraction of the mean operation. Hence the resulting time series always has zero mean.

**Value**

Returns a vector of residuals the same length as x.

**See Also**

[test](#)

**Examples**

```
M = c("log", "season", 12, "trend", 1)
e = Resid(wine, M)

a = arma(e, 1, 1)
ee = Resid(wine, M, a)
```

---

season	<i>Estimate seasonal component</i>
--------	------------------------------------

---

**Description**

Estimate seasonal component

**Usage**

```
season(x, d)
```

**Arguments**

x	Time series data
d	Number of observations per season

**Value**

Returns a vector the same length as x. Subtract from x to obtain residuals.

**See Also**

[trend](#)

**Examples**

```
y = season(deaths,12)
plotc(deaths,y)
```

---

selftest	<i>Run a self test</i>
----------	------------------------

---

**Description**

Run a self test

**Usage**

```
selftest()
```

**Details**

This function is a useful check if the code is modified.

**Value**

None

**Examples**

```
selftest()
```

---

sim

*Generate synthetic observations*

---

**Description**

Generate synthetic observations

**Usage**

```
sim(a, n = 100)
```

**Arguments**

a	ARMA model
n	Number of synthetic observations required

**Details**

The ARMA model is a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Value**

Returns a vector of n synthetic observations.

**Examples**

```
a = specify(ar=c(0,0,.99))
x = sim(a,60)
plotc(x)
```

---

smooth.exp	<i>Apply an exponential filter</i>
------------	------------------------------------

---

**Description**

Apply an exponential filter

**Usage**

```
smooth.exp(x, alpha)
```

**Arguments**

x	Time series data
alpha	Smoothness setting, 0-1

**Details**

Zero is maximum smoothness.

**Value**

Returns a vector of smoothed data the same length as x.

**Examples**

```
y = smooth.exp(strikes,.4)
plotc(strikes,y)
```

---

smooth.fft	<i>Apply a low pass filter</i>
------------	--------------------------------

---

**Description**

Apply a low pass filter

**Usage**

```
smooth.fft(x, f)
```

**Arguments**

x	Time series data
f	Cut-off frequency, 0-1

**Details**

The cut-off frequency is specified as a fraction. For example,  $c = .25$  passes the lowest 25% of the spectrum.

**Value**

Returns a vector the same length as  $x$ .

**Examples**

```
y = smooth.fft(deaths,.1)
plotc(deaths,y)
```

---

smooth.ma

*Apply a moving average filter*

---

**Description**

Apply a moving average filter

**Usage**

```
smooth.ma(x, q)
```

**Arguments**

x	Time series data
q	Filter order

**Details**

The averaging function uses  $2q+1$  values.

**Value**

Returns a vector the same length as  $x$ .

**Examples**

```
y = smooth.ma(strikes,2)
plotc(strikes,y)
```



---

smooth.rank	<i>Apply a spectral filter</i>
-------------	--------------------------------

---

**Description**

Apply a spectral filter

**Usage**

```
smooth.rank(x, k)
```

**Arguments**

x	Time series data
k	Number of frequencies

**Details**

Passes the mean and the k frequencies with the highest amplitude. The remainder of the spectrum is filtered out.

**Value**

Returns a vector the same length as x.

**Examples**

```
y = smooth.rank(deaths,2)
plotc(deaths,y)
```

---

specify	<i>Specify an ARMA model</i>
---------	------------------------------

---

**Description**

Specify an ARMA model

**Usage**

```
specify(ar = 0, ma = 0, sigma2 = 1)
```

**Arguments**

ar	Vector of AR coefficients (index number equals coefficient subscript)
ma	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Value**

Returns an ARMA model consisting of a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	Vector of MA coefficients (index number equals coefficient subscript)
sigma2	White noise variance

**Examples**

```
specify(ar=c(0,0,.99))
```

---

strikes	<i>USA union strikes, 1951-1980</i>
---------	-------------------------------------

---

**Description**

USA union strikes, 1951-1980

**Examples**

```
plotc(strikes)
```

---

Sunspots	<i>Number of sunspots, 1770 to 1869</i>
----------	---

---

**Description**

Number of sunspots, 1770 to 1869

**Examples**

```
plotc(Sunspots)
```

---

test	<i>Test residuals for stationarity and randomness</i>
------	---

---

**Description**

Test residuals for stationarity and randomness

**Usage**

```
test(e)
```

**Arguments**

e                    Time series data (typically residuals from Resid)

**Details**

Plots ACF, PACF, residuals, and QQ. Displays results for Ljung-Box, McLeod-Li, turning point, difference-sign, and rank tests. The plots can be used to check for stationarity and the other tests check for white noise.

**Value**

None

**See Also**

[Resid](#)

**Examples**

```
M = c("log", "season", 12, "trend", 1)
e = Resid(wine, M)
test(e) ## Is e stationary?
a = arma(e, 1, 1)
ee = Resid(wine, M, a)
test(ee) ## Is ee white noise?
```

---

trend	<i>Estimate trend component</i>
-------	---------------------------------

---

**Description**

Estimate trend component

**Usage**

```
trend(x, p)
```

**Arguments**

x	Time series data
p	Polynomial order (1 linear, 2 quadratic, etc.)

**Value**

Returns a vector the same length as x. Subtract from x to obtain residuals. The returned vector is the least squares fit of a polynomial to the data.

**See Also**

[season](#)

**Examples**

```
y = trend(uspop,2)
plotc(uspop,y)
```

---

wine	<i>Australian red wine sales, January 1980 to October 1991</i>
------	--

---

**Description**

Australian red wine sales, January 1980 to October 1991

**Examples**

```
plotc(wine)
```

---

 yw

*Estimate AR coefficients using the Yule-Walker method*


---

**Description**

Estimate AR coefficients using the Yule-Walker method

**Usage**

```
yw(x, p)
```

**Arguments**

x	Time series data (typically residuals from Resid)
p	AR order

**Details**

The innovations algorithm is used to estimate white noise variance.

**Value**

Returns an ARMA model consisting of a list with the following components.

phi	Vector of AR coefficients (index number equals coefficient subscript)
theta	0
sigma2	White noise variance
aicc	Akaike information criterion corrected
se.phi	Standard errors for the AR coefficients
se.theta	0

**See Also**

[arma](#) [burg](#) [hannan](#) [ia](#)

**Examples**

```
M = c("diff",1)
e = Resid(dowj,M)
a = yw(e,1)
```

# Index

- \* **datasets**
  - airpass, 5
  - deaths, 11
  - dowj, 12
  - lake, 16
  - strikes, 26
  - Sunspots, 26
  - wine, 28
- \* **package**
  - itsmr-package, 2
- aacvf, 4
- acvf, 5
- airpass, 5
- ar.inf, 6, 16
- arar, 7
- arma, 4, 8, 9, 10, 13–15, 29
- autofit, 8, 9
  
- burg, 8, 10, 14, 15, 29
  
- check, 11
  
- deaths, 11
- dowj, 12
  
- forecast, 7, 12
  
- hannan, 8, 10, 13, 15, 29
- hr, 14
  
- ia, 8, 10, 14, 15, 29
- itsmr (itsmr-package), 2
- itsmr-package, 2
  
- lake, 16
  
- ma.inf, 6, 16
  
- periodogram, 17, 19
- plota, 5, 18
  
- plotc, 18
- plots, 17, 19
  
- Resid, 13, 19, 27
  
- season, 21, 28
- selftest, 21
- sim, 22
- smooth.exp, 23
- smooth.fft, 23
- smooth.ma, 24
- smooth.rank, 25
- specify, 25
- strikes, 26
- Sunspots, 26
  
- test, 13, 20, 27
- trend, 21, 28
  
- wine, 28
  
- yw, 8, 10, 14, 15, 29