Portmanteau Test Statistics

Esam Mahdi
Islamic University of Gaza

A. Ian McLeod
University of Western Ontario

Abstract

In this vignette, we briefly describe the portmanteau test statistics given in the portes package based on the asymptotic chi-square distribution and Monte-Carlo significance test. Some illustrative applications are given.

**Keywords:** ARMA models, VARMA models, SARIMA models, GARCH models, ARFIMA models, TAR models, Monte-Carlo significance test, Portmanteau test, Parallel computing.

1. Box and Pierce portmanteau test

In the univariate time series, Box and Pierce (1970) introduced the portmanteau statistic

\[ Q_m = n \sum_{\ell=1}^{m} \hat{r}_\ell^2, \]  

(1)

where \( \hat{r}_\ell = \sum_{t=\ell+1}^{n} \hat{a}_t \hat{a}_{t-\ell} / \sum_{t=1}^{n} \hat{a}_t^2 \), and \( \hat{a}_1, \ldots, \hat{a}_n \) are the residuals. This test statistic is implemented in the R function `BoxPierce()`, where it can be used with the multivariate case as well. \( Q_m \) has a chi-square distribution with \( k^2(m - p - q) \) degrees of freedom where \( k \) represents the dimension of the time series. The usage of this function is extremely simple:

`BoxPierce(obj, lags=seq(5,30,5), order=0, season=1, squared.residuals=FALSE),`

where `obj` is a univariate or multivariate series with class "numeric", "matrix", "ts", or ("mts" "ts"). It can be also an object of fitted time-series model (including time series regression) with class "ar"1, "arima0"2, "Arima"3, ("ARIMA" "Arima")4, "lm"5, ("glm" "lm")6, "varest"7. `obj` may also an object with class "list" from any fitted model using the built in R functions, such as the functions `FitAR()`, `FitARz()`, and `FitARP()` from the FitAR R package (McLeod, Zhang, and Xu 2013), the function `garch()` from the R package `{tseries}` (Trapletti, Hornik, and LeBaron 2017), the function `garchFit()` from the R package

---

1The functions `ar()`, `ar.burg()`, `ar.yw()`, `ar.mle()`, and `ar.ols()` in the R package `stats` produce an output with class "ar".

2The function `arima0()` in the R package `stats` produces an output with class "arima0".

3The function `arima()` in the R package `stats` produces an output with class "Arima".

4The functions `Arima()` and `auto.arima()` in the R package `forecast` produce an output with class ("ARIMA" "Arima")

5The function `lm()` in the R package `stats` produces an output with class "lm".

6The function `glm()` in the R package `stats` produces an output with class ("glm" "lm")

7The function `VAR()` in the R package `vars` produces an output with class "varest".
Portmanteau Test Statistics

fGarch (Wuertz and core team members 2016), the function fracdiff() from the R package fracdiff (Fraley, Leisch, Maechler, Reisen, and Lemonte 2012), the function tar() from the R package TSA (Chan and Ripley 2012), etc. lags is a vector of numeric integers represents the lag values, m, at which we need to check the adequacy of the fitted model.

It is important, as indicated by McLeod (1978), to use this test statistic for testing the seasonality with seasonal period s in many applications. The test for seasonality may obtained by replacing the lag ℓ in the test statistics given in Equation 1 by ℓs, which is implemented in our package. In this case, the seasonal period s is entered via the argument season, where season = 1 is used for usual test with no seasonality check.

The argument order is used for degrees of freedom of asymptotic chi-square distribution. If obj is a fitted time-series model with class "ar", "arima0", "Arima", ("ARIMA", "Arima"), "lm", ("glm", "lm"), "varest", or "list" then no need to enter the value of order as it will be automatically determined from the original fitted model of the object obj. In general order = p + q, where p and q are the orders of the autoregressive (or vector autoregressive) and moving average (or vector moving average) models respectively. In SARIMA models order = p + q + ps + qs, where ps and qs are the orders of the seasonal autoregressive and seasonal moving average respectively. season is the seasonality period which is needed for testing the seasonality cases. Default is season = 1 for testing the non seasonality cases. Finally, when squared.residuals = TRUE, then apply the test on the squared values to check for Autoregressive Conditional Heteroscedastic, ARCH, effects. When squared.residuals = FALSE, then apply the test on the usual residuals.

Note that the function portest() with the arguments test = "BoxPierce", MonteCarlo = FALSE, order = 0, season = 1, and squared.residuals=FALSE will gives the same results of the function BoxPierce(). The Monte-Carlo version of this test statistic is implemented in the function portest() as an argument test = "BoxPierce" provided that MonteCarlo = TRUE is selected.

```
portest(obj, lags=seq(5,30,5), test="BoxPierce", fn=NULL, squared.residuals=FALSE, MonteCarlo=TRUE, innov.dist=c("Gaussian","t","stable","bootstrap"), ncores=1, nrep=1000, model=list(sim.model=NULL, fit.model=NULL), pkg.name=NULL, set.seed=123, season=1, order=0)
```

1.1. Example 1

First a simple univariate example is provided. We fit an AR(2) model to the logarithms of Canadian lynx trappings from 1821 to 1934. Data is available from the R package datasets under the name lynx. This model was selected using the BIC criterion. The asymptotic distribution and the Monte-Carlo version of \( Q_m \) statistic are given in the following R code for lags m = 5, 10, 15, 20, 25, 30.

```
> library("portes")

> require("FitAR")
> lynxData <- log(lynx)
> p <- SelectModel(lynxData, ARModel = "AR", Criterion = "BIC", Best = 1)
```
```r
fit <- FitAR(lynxData, p, ARModel = "AR")
res <- fit$res
BoxPierce(res, order=p) ## The asymptotic distribution of BoxPierce test

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6.748225</td>
<td>3</td>
<td>0.08037069</td>
</tr>
<tr>
<td>10</td>
<td>15.856081</td>
<td>8</td>
<td>0.04448698</td>
</tr>
<tr>
<td>15</td>
<td>22.631444</td>
<td>13</td>
<td>0.04631764</td>
</tr>
<tr>
<td>20</td>
<td>30.304179</td>
<td>18</td>
<td>0.03459211</td>
</tr>
<tr>
<td>25</td>
<td>34.157210</td>
<td>23</td>
<td>0.06291892</td>
</tr>
<tr>
<td>30</td>
<td>37.963103</td>
<td>28</td>
<td>0.09909886</td>
</tr>
</tbody>
</table>

## Use FitAR from FitAR R package with Monte-Carlo version of BoxPierce test, users may write their own two R functions. See the following example:

fit.model <- function(data){
  p <- SelectModel(data, ARModel = "AR", Criterion = "BIC", Best = 1)
  fit <- FitAR(data, p, ARModel = "AR")
  res <- fit$res
  phiHat <- fit$phiHat
  sigsqHat <- fit$sigsqHat
  list(res=res,order=p,phiHat=phiHat,sigsqHat=sigsqHat)
}
Fit <- fit.model(lynxData)
BoxPierce(Fit) ## The asymptotic distribution of BoxPierce statistic

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6.748225</td>
<td>3</td>
<td>0.08037069</td>
</tr>
<tr>
<td>10</td>
<td>15.856081</td>
<td>8</td>
<td>0.04448698</td>
</tr>
<tr>
<td>15</td>
<td>22.631444</td>
<td>13</td>
<td>0.04631764</td>
</tr>
<tr>
<td>20</td>
<td>30.304179</td>
<td>18</td>
<td>0.03459211</td>
</tr>
<tr>
<td>25</td>
<td>34.157210</td>
<td>23</td>
<td>0.06291892</td>
</tr>
<tr>
<td>30</td>
<td>37.963103</td>
<td>28</td>
<td>0.09909886</td>
</tr>
</tbody>
</table>

sim.model <- function(parSpec){
  phi <- parSpec$phiHat
  n <- length(parSpec$res)
  sigma <- parSpec$sigsqHat
  ts(SimulateGaussianAR(phi, n = n, InnovationVariance = sigma))
}
portest(Fit,test = "BoxPierce", ncores = 4,
  model=list(sim.model=sim.model,fit.model=fit.model),pkg.name="FitAR")

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6.748225</td>
<td>0.05294705</td>
</tr>
<tr>
<td>10</td>
<td>15.856081</td>
<td>0.02497502</td>
</tr>
<tr>
<td>15</td>
<td>22.631444</td>
<td>0.02297702</td>
</tr>
</tbody>
</table>
```
Portmanteau Test Statistics

20  30.304179  0.01298701
25  34.157210  0.02797203
30  37.963103  0.03196803

For lags \( m > 5 \), the Monte-Carlo version of Box and Pierce test and the asymptotic chi-square suggests that the model maybe inadequate. Fitting a subset autoregressive using the \( \text{BIC} \) (McLeod and Zhang 2008), the portmanteau test based on both methods, Monte-Carlo and asymptotic distribution suggest model adequacy.

```r
> SelectModel(log(lynx),lag.max=15,ARModel="ARp",Criterion="BIC",Best=1)
[1]  1  2  4 10 11

> FitsubsetAR <- function(data){
+   FitsubsetAR <- FitARp(data, c(1, 2, 4, 10, 11))
+   res <- FitsubsetAR$res
+   phiHat <- FitsubsetAR$phiHat
+   p <- length(phiHat)
+   sigsqHat <- FitsubsetAR$sigsqHat
+   list(res=res,order=p,phiHat=phiHat,sigsqHat=sigsqHat)
+ }

> SimsubsetARModel <- function(parSpec){
+   phi <- parSpec$phiHat
+   n <- length(parSpec$res)
+   sigma <- parSpec$sigsqHat
+   ts(SimulateGaussianAR(phi, n = n, InnovationVariance = sigma))
+ }

> Fitsubset <- FitsubsetAR(lynxData)
> BoxPierce(Fitsubset)

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2.382300</td>
<td>0</td>
<td>NA</td>
</tr>
<tr>
<td>10</td>
<td>4.258836</td>
<td>0</td>
<td>NA</td>
</tr>
<tr>
<td>15</td>
<td>6.532786</td>
<td>4</td>
<td>0.162736</td>
</tr>
<tr>
<td>20</td>
<td>9.887818</td>
<td>9</td>
<td>0.359643</td>
</tr>
<tr>
<td>25</td>
<td>13.258935</td>
<td>14</td>
<td>0.506244</td>
</tr>
<tr>
<td>30</td>
<td>16.172499</td>
<td>19</td>
<td>0.645739</td>
</tr>
</tbody>
</table>

> portest(Fitsubset,test = "BoxPierce", ncores = 4,
+     model=list(sim.model=SimsubsetARModel,fit.model=FitsubsetAR),pkg.name="FitAR")

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2.382300</td>
<td>0.5654346</td>
</tr>
<tr>
<td>10</td>
<td>4.258836</td>
<td>0.7822178</td>
</tr>
<tr>
<td>15</td>
<td>6.532786</td>
<td>0.8481518</td>
</tr>
<tr>
<td>20</td>
<td>9.887818</td>
<td>0.8211788</td>
</tr>
<tr>
<td>25</td>
<td>13.258935</td>
<td>0.7952048</td>
</tr>
<tr>
<td>30</td>
<td>16.172499</td>
<td>0.7972028</td>
</tr>
</tbody>
</table>
It is important to indicate that the p-values associated with the Monte-Carlo significance tests are always exit and do not depend on the degrees of freedom, while the p-value based on the asymptotic chi-square distribution tests are defined only for positive degrees of freedom.

1.2. Example 2

In this example we consider the monthly log stock returns of Intel corporation data from January 1973 to December 2003. First we apply the $Q_m$ statistic directly on the returns using the asymptotic distribution and the Monte-Carlo significance test. The results suggest that returns data behaves like white noise series as no significant serial correlations found.

```r
> monthintel <- as.ts(monthintel)
> BoxPierce(monthintel)

lags statistic df  p-value
5     4.666889  5  0.45786938
10    14.364748 10  0.15699489
15    23.120348 15  0.08161787
20    24.000123 20  0.24238680
25    29.617977 25  0.2391229
30    31.943703 30  0.37015020

> portest(monthintel, test = "BoxPierce", ncores = 4)

lags statistic df  p-value
5     4.666889  5  0.45554446
10    14.364748 10  0.13186813
15    23.120348 15  0.07292707
20    24.000123 20  0.19380619
25    29.617977 25  0.19180819
30    31.943703 30  0.26573427

> BoxPierce(monthintel, squared.residuals = TRUE)

lags statistic df  p-value
5     40.78073  5  1.039009e-07
10    49.57872 10  3.189915e-07
15    81.90133 15  3.131517e-11
20    86.50575 20  3.006796e-10
25    87.54737 25  7.161478e-09
30    88.55017 30  1.087505e-07
```

After that we apply the $Q_m$ statistic on the squared returns. The results suggest that the monthly returns are not serially independent and the return series may suffers of ARCH effects.

```r
> BoxPierce(monthintel, squared.residuals = TRUE)

lags statistic df  p-value
5     40.78073  5  1.039009e-07
10    49.57872 10  3.189915e-07
15    81.90133 15  3.131517e-11
20    86.50575 20  3.006796e-10
25    87.54737 25  7.161478e-09
30    88.55017 30  1.087505e-07
```
Portmanteau Test Statistics

> portest(monthintel,test="BoxPierce",ncores=4,squared.residuals=TRUE)

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>40.78073</td>
<td>0.000999001</td>
</tr>
<tr>
<td>10</td>
<td>49.57872</td>
<td>0.000999001</td>
</tr>
<tr>
<td>15</td>
<td>81.90133</td>
<td>0.000999001</td>
</tr>
<tr>
<td>20</td>
<td>86.50575</td>
<td>0.000999001</td>
</tr>
<tr>
<td>25</td>
<td>87.54737</td>
<td>0.000999001</td>
</tr>
<tr>
<td>30</td>
<td>88.55017</td>
<td>0.000999001</td>
</tr>
</tbody>
</table>

1.3. Example 3

In this example we implement the portmanteau statistic on an econometric model of aggregate demand in the U.K. to show the usefulness of using these statistics in testing the seasonality. The data are quarterly, seasonally unadjusted in 1958 prices, covering the period 1957/3-1967/4 (with 7 series each with 42 observations), as published in Economic Trends and available from our package with the name EconomicUK. This data were disused by Prothero and Wallis (1976), where they fit several models to each series and compared their performance with a multivariate model (See (Prothero and Wallis 1976, Tables 1-7)).

For simplicity, we select the first series, Cn: Consumers’ expenditure on durable goods, and the first model 1a as fitted by Prothero and Wallis (1976) in Table 1.

> require("forecast")
> cd <- EconomicUK[,1]
> cd.fit <- Arima(cd, order=c(0,1,0), seasonal=list(order=c(0,1,1), period=4))

After that we apply the usual $Q_m$ test statistic as well as the seasonal version of $Q_m$ test statistic. We implement both cases using the asymptotic distribution and the Monte-Carlo procedures. The results suggest that the model is good.

> BoxPierce(cd.fit,lags=c(5,10),season=1) ## Asympt. dist. for usual check

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2.509718</td>
<td>4</td>
<td>0.6428964</td>
</tr>
<tr>
<td>10</td>
<td>5.252716</td>
<td>9</td>
<td>0.8117454</td>
</tr>
</tbody>
</table>

> BoxPierce(cd.fit,lags=c(5,10),season=4) ## Asympt. dist. check for seasonality

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1.307341</td>
<td>4</td>
<td>0.8601288</td>
</tr>
<tr>
<td>10</td>
<td>1.918594</td>
<td>9</td>
<td>0.9926904</td>
</tr>
</tbody>
</table>

> portest(cd.fit,lags=c(5,10),test="BoxPierce",ncores=4) ## MC check for seasonality

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2.509718</td>
<td>0.5184815</td>
</tr>
<tr>
<td>10</td>
<td>5.252716</td>
<td>0.2267732</td>
</tr>
</tbody>
</table>
2. Ljung and Box portmanteau test

Ljung and Box (1978) modified Box and Pierce (1970) test statistic by

\[
\hat{Q}_m = n(n+2) \sum_{\ell=1}^{m} (n-\ell)^{-1} \hat{r}_\ell^2.
\]

This test statistic is also asymptotically chi-square with the same degrees of freedom of BoxPierce and it is implemented in the contribution R function LjungBox(),

\[
\text{LjungBox}(\text{obj}, \text{lags=}\text{seq}(5,30,5), \text{order=}0, \text{season=}1, \text{squared.residuals=}\text{FALSE}),
\]

where the arguments of this function are described as before.

In stats R, the function Box.test() was built to compute the Box and Pierce (1970) and Ljung and Box (1978) test statistics only in the univariate case where we can not use more than one single lag value at a time. The functions BoxPierce() and LjungBox() are more general than Box.test() and can be used in the univariate or multivariate time series at vector of different lag values as well as they can be applied on an output object from a fitted model described in the description of the function BoxPierce().

Note that the function portest() with the arguments test = "LjungBox", MonteCarlo = FALSE, order = 0, season = 1, and squared.residuals=FALSE will gives the same results of the function LjungBox(). The Monte-Carlo version of this test statistic is implemented in the function portest() as an argument test = "LjungBox" provided that MonteCarlo = TRUE is selected.

\[
\text{portest}(\text{obj}, \text{lags=}\text{seq}(5,30,5), \text{test=}\text{"LjungBox"}, \text{fn=}\text{NULL}, \text{squared.residuals=}\text{FALSE}, \\
\text{MonteCarlo=}\text{TRUE}, \text{innov.dist=}\text{c("Gaussian","t","stable","bootstrap"), ncores=}1, \\
\text{nrep=}1000, \text{model=}\text{list(sim.model=}\text{NULL}, \text{fit.model=}\text{NULL}), \text{pkg.name=}\text{NULL}, \\
\text{set.seed=}123, \text{season=}1, \text{order=}0)
\]

2.1. Example 4

The built-in R function auto.arima() in the package forecast (Hyndman, Athanasopoulos, Razbash, Schmidt, Zhou, Khan, Bergmeir, and Wang 2017) is used to fit the best ARIMA model based on the AIC criterion to the numbers of users connected to the Internet through a server every minute WWWusage dataset of length 100 that is available from the forecast package,

\[
> \text{library("forecast")}
> \text{FitWWW} \leftarrow \text{auto.arima(WWWusage)}
\]

Then the LjungBox portmanteau test is applied on the residuals of the fitted model at lag values m = 5, 10, 15, 20, 25, and 30 which yields that the assumption of the adequacy in the fitted model is fail to reject.
Portmanteau Test Statistics

> LjungBox(FitWWW) ## The asymptotic distribution of LjungBox test

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4.091378</td>
<td>3</td>
<td>0.2517645</td>
</tr>
<tr>
<td>10</td>
<td>7.833827</td>
<td>8</td>
<td>0.4498687</td>
</tr>
<tr>
<td>15</td>
<td>11.985102</td>
<td>13</td>
<td>0.5288659</td>
</tr>
<tr>
<td>20</td>
<td>19.736039</td>
<td>18</td>
<td>0.3478749</td>
</tr>
<tr>
<td>25</td>
<td>28.147803</td>
<td>23</td>
<td>0.2102440</td>
</tr>
<tr>
<td>30</td>
<td>33.460065</td>
<td>28</td>
<td>0.2192169</td>
</tr>
</tbody>
</table>

> portest(FitWWW, nrep = 500, test = "LjungBox", ncores = 4)

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>4.091378</td>
<td>0.2834331</td>
</tr>
<tr>
<td>10</td>
<td>7.833827</td>
<td>0.5089820</td>
</tr>
<tr>
<td>15</td>
<td>11.985102</td>
<td>0.5568862</td>
</tr>
<tr>
<td>20</td>
<td>19.736039</td>
<td>0.3632735</td>
</tr>
<tr>
<td>25</td>
<td>28.147803</td>
<td>0.2335329</td>
</tr>
<tr>
<td>30</td>
<td>33.460065</td>
<td>0.2315369</td>
</tr>
</tbody>
</table>

> detach(package:forecast)

3. Hosking portmanteau test

Hosking (1980) generalized the univariate portmanteau test statistics given in eqns. (1, 2) to the multivariate case. He suggested the modified multivariate portmanteau test statistic

\[ \tilde{Q}_m = n^2 \sum_{\ell=1}^{m} (n-\ell)^{-1} \tilde{r}_\ell' (\tilde{R}_0^{-1} \otimes \tilde{R}_0^{-1}) \tilde{r}_\ell, \]  

where \( \tilde{r}_\ell = \text{vec} \tilde{R}_\ell \) is a \( 1 \times k^2 \) row vector with rows of \( \tilde{R}_\ell \) stacked one next to the other, and \( m \) is the lag order. The \( \otimes \) denotes the Kronecker product (http://en.wikipedia.org/wiki/Kronecker_product), \( \tilde{R}_\ell = L \tilde{\Gamma}_\ell L \), \( LL' = \tilde{\Gamma}_0^{-1} \) where \( \tilde{\Gamma}_\ell = n^{-1} \sum_{t=\ell+1}^{n} \hat{a}_t \hat{a}_t' \) is the lag \( \ell \) residual autocovariance matrix.

The asymptotic distributions of \( \tilde{Q}_m \) is chi-squared with the same degrees of freedom of BoxPierce and LjungBox. In portest package, this statistic is implemented in the function Hosking():

\[ \text{Hosking}(\text{obj}, \text{lags=seq}(5,30,5), \text{order}=0, \text{season}=1, \text{quared.residuals}=\text{FALSE}), \]

where the arguments of this function is described as before. Note that the function portest() with the arguments test = "Hosking", MonteCarlo = FALSE, order = 0, season = 1, and squared.residuals=FALSE will gives the same results of the function Hosking(). The Monte-Carlo version of this test statistic is implemented in the function portest() as an argument test = "Hosking" provided that MonteCarlo = TRUE is selected.
portest(obj, lags=seq(5, 30, 5), test="Hosking", fn=NULL, squared.residuals=FALSE, MonteCarlo=TRUE, innov.dist=c("Gaussian","t","stable","bootstrap"), ncores=1, nrep=1000, model=list(sim.model=NULL, fit.model=NULL), pkg.name=NULL, set.seed=123, season=1, order=0)

3.1. Example 5

In this example, we consider fitting a \( \text{VAR}(k) \), \( k = 1, 3, 5 \) model to the monthly log returns of the IBM stock and the S&P 500 index from January 1926 to December 2008 with 996 observations (Tsay 2010, chapter 8). The p-values for the modified portmanteau test of Hosking (1980), \( \tilde{Q}_m \), are computed using the Monte-Carlo test procedure with \( 10^3 \) replications. For additional comparisons, the p-values for \( \tilde{Q}_m \) are also evaluated using asymptotic approximations.

```r
> data("IbmSp500")
> ibm <- log(IbmSp500[, 2] + 1) * 100
> sp5 <- log(IbmSp500[, 3] + 1) * 100
> z <- data.frame(cbind(ibm, sp5))
> FitIBMSP5001 <- ar.ols(z, aic = FALSE, intercept = TRUE, order.max = 1)
> Hosking(FitIBMSP5001)

lags statistic df    p-value
   5  44.60701  16 0.0001594110
  10  63.92523  36 0.0028210050
  15  79.63965  56 0.0206430161
  20 122.76400  76 0.0005488958
  25 152.14275  96 0.0002315766
  30 172.10164 116 0.0005612691

> portest(FitIBMSP5001, test = "Hosking", ncores = 4)

lags statistic    p-value
   5  44.60701 0.000999001
  10  63.92523 0.002821008
  15  79.63965 0.020979021
  20 122.76400 0.000999001
  25 152.14275 0.000999001
  30 172.10164 0.000999001

> FitIBMSP5003 <- ar.ols(z, aic = FALSE, intercept = TRUE, order.max = 3)
> Hosking(FitIBMSP5003)

lags statistic df    p-value
   5  21.46968  8 0.0005999073
  10 40.36636 28 0.061317366
  15 55.14693 48 0.222617147
Portmanteau Test Statistics

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>92.49612</td>
<td>0.025796818</td>
</tr>
<tr>
<td>25</td>
<td>121.00241</td>
<td>0.011311937</td>
</tr>
<tr>
<td>30</td>
<td>138.44693</td>
<td>0.025694805</td>
</tr>
</tbody>
</table>

> portest(FitIBMSP5003, test = "Hosking", ncores = 4)

lags statistic p-value
5  21.46968 0.008991009
10 40.36636 0.065934066
15 55.14693 0.204795205
20 92.49612 0.024975025
25 121.00241 0.010989011
30 138.44693 0.019980020

> FitIBMSP5005 <- ar.ols(z, aic = FALSE, intercept = TRUE, order.max = 5)
> Hosking(FitIBMSP5005)

lags statistic df p-value
5  0.2076267 0 0.0000000
10 19.2862036 20 0.5032986
15 36.8697754 40 0.6119561
20 73.5270586 60 0.1126691
25 98.7210756 80 0.0763671
30 115.5525028 100 0.1369843

> portest(FitIBMSP5005, test = "Hosking", ncores = 4)

lags statistic p-value
5  0.2076267 0.91008991
10 19.2862036 0.48051948
15 36.8697754 0.6119561
20 73.5270586 0.1126691
25 98.7210756 0.0763671
30 115.5525028 0.1369843

All results reject the fitted VAR(1) and VAR(3) whereas the results suggest that the VAR(5) models is maybe an adequate model.

4. Li and McLeod portmanteau test

Li and McLeod (1981) suggested the multivariate modified portmanteau test statistic

\[ \hat{Q}_m^{(L)} = n \sum_{t=1}^{m} \hat{r}_t (\hat{R}_0^{-1} \otimes \hat{R}_0^{-1}) \hat{r}_t + \frac{k^2 m (m + 1)}{2n}, \]  

(4)

which is distributed as chi-squared with the same degrees of freedom of BoxPierce, LjungBox, and Hosking. In portes package, the test statistic \( \hat{Q}_m^{(L)} \) is implemented in the function LiMcLeod().
LiMcLeod(obj, lags=seq(5,30,5), order=0, season=1, squared.residuals=FALSE),
where the arguments of this function is described as before. Note that the function portest() with the arguments test = "LiMcLeod", MonteCarlo = FALSE, order = 0, season = 1, and squared.residuals=FALSE will gives the same results of the function LiMcLeod(). The Monte-Carlo version of this test statistic is implemented in the function portest() as an argument test = "LiMcLeod" provided that MonteCarlo = TRUE is selected.

portest(obj,lags=seq(5,30,5),test="LiMcLeod",fn=NULL,squared.residuals=FALSE,
MonteCarlo=TRUE,innov.dist=c("Gaussian","t","stable","bootstrap"),ncores=1,
nrep=1000,model=list(sim.model=NULL,fit.model=NULL),pkg.name=NULL,
set.seed=123,season=1,order=0)

4.1. Example 6
The trivariate quarterly time series, 1960–1982, of West German investment, income, and consumption was discussed by Lütkepohl (2005, §3.23). So n = 92 and k = 3 for this series. As in Lütkepohl (2005, §4.24) we model the logarithms of the first differences. Using the AIC and FPE, Lütkepohl (2005, Table 4.25) selected a var(2) for this data. All diagnostic tests reject simple randomness, var(0). The asymptotic distribution and the Monte-Carlo tests for var(1) suggests model inadequacy supports the choice of the var(2) model. However, testing for nonlinearity using the squared residuals suggest inadequacy in the var(2) model,

```r
> data("WestGerman")
> DiffData <- matrix(numeric(3 * 91), ncol = 3)
> for (i in 1:3) DiffData[, i] <- diff(log(WestGerman[, i]), lag = 1)
> FitWG <- ar.ols(DiffData, aic = FALSE, order.max = 2, intercept = FALSE)
> LiMcLeod(FitWG, lags = c(5, 10, 15))

lags statistic df   p-value
  5  30.65934 27  0.2853557
 10  72.38418 72  0.4651266
 15 122.08588 117  0.3552372

> portest(FitWG, lags = c(5, 10, 15), test = "LiMcLeod", ncores = 4)

lags statistic   p-value
  5  30.65934 0.3506494
 10  72.38418 0.5314685
 15 122.08588 0.3656344

> LiMcLeod(FitWG, lags = c(5, 10, 15), squared.residuals = TRUE)

lags statistic df   p-value
  5  35.12685 27  0.13568171
 10  91.04927 72  0.064231096
 15 169.14303 117  0.001161299
```
5. Generalized variance portmanteau test

Peña and Rodríguez (2002) proposed a univariate portmanteau test of goodness-of-fit test based on the $m$-th root of the determinant of the $m$-th Toeplitz residual autocorrelation matrix

$$\hat{R}_m = \begin{pmatrix} \hat{r}_0 & \hat{r}_1 & \cdots & \hat{r}_m \\ \hat{r}_{-1} & \hat{r}_0 & \cdots & \hat{r}_{m-1} \\ \vdots & \ddots & \ddots & \vdots \\ \hat{r}_{-m} & \hat{r}_{-m+1} & \cdots & \hat{r}_0 \end{pmatrix},$$

(5)

where $\hat{r}_0 = 1$ and $\hat{r}_{-\ell} = \hat{r}_{\ell}$, for all $\ell$. They approximated the distribution of their proposed test statistic by the gamma distribution and provided simulation experiments to demonstrate the improvement of their statistic in comparison with the one that is given in Eq. (2).

Peña and Rodríguez (2006) suggested to modify this test by taking the log of the $(m+1)$-th root of the determinant in Eq. (5). They proposed two approximations by using the Gamma and Normal distributions to the asymptotic distribution of this test and indicated that the performance of both approximations for checking the goodness-of-fit in linear models is similar and more powerful for small sample size than the previous one. Lin and McLeod (2006) introduced the Monte-Carlo version of this test as they noted that it is quite often that the generalized variance portmanteau test does not agree with the suggested Gamma approximation and the Monte-Carlo version of this test is more accurate. Mahdi and McLeod (2012) generalized both methods to the multivariate time series. Their test statistic

$$D_m = \frac{-3n}{2m+1} \log |\hat{R}_m|,$$

(6)

where

$$\hat{R}_m = \begin{pmatrix} \mathbb{I}_k & \hat{R}_1 & \cdots & \hat{R}_m \\ \hat{R}_{-1} & \mathbb{I}_k & \cdots & \hat{R}_{m-1} \\ \vdots & \ddots & \ddots & \vdots \\ \hat{R}_{-m} & \hat{R}_{-m+1} & \cdots & \mathbb{I}_k \end{pmatrix}.$$  

(7)

Replacing $\hat{R}_m$ that is given in Equation refMahdiMcLoed by $\hat{R}_m(s)$ will easily extend to test for seasonality with period $s$, where

$$\hat{R}_m(s) = \begin{pmatrix} \mathbb{I}_k & \hat{R}_s & \hat{R}_{2s} & \cdots & \hat{R}_{ms} \\ \hat{R}_s & \mathbb{I}_k & \hat{R}_s & \cdots & \hat{R}_{(m-1)s} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \hat{R}_{ms} & \hat{R}_{(m-1)s} & \hat{R}_{(m-2)s} & \cdots & \mathbb{I}_k \end{pmatrix}.$$  

(8)

The null distribution is approximately $\chi^2$ with $k^2(1.5m(m+1)(2m+1)^{-1} - o)$ degrees of freedom where $o = p + q + ps + qs$ denotes the order of the series as described before. This test statistics is implemented in the contributed R function MahdiMcLeod(),

MahdiMcLeod(obj, lags=seq(5,30,5), order=0, season=1, squared.residuals=FALSE),
where the arguments of this function are described as before. Note that the function `portest()` with the arguments `test = "MahdiMcLeod", MonteCarlo = FALSE, order = 0, season = 1, and squared.residuals=FALSE` will give the same results of the function `MahdiMcLeod()`. The Monte-Carlo version of this test statistic is implemented in the function `portest()` as an argument `test = "MahdiMcLeod"` provided that `MonteCarlo = TRUE` is selected.

```
portest(obj,lags=seq(5,30,5),test="MahdiMcLeod",fn=NULL,squared.residuals=FALSE,
       MonteCarlo=TRUE,innov.dist=c("Gaussian","t","stable","bootstrap"),ncores=1,
       nrep=1000,model=list(sim.model=NULL,fit.model=NULL),pkg.name=NULL,
       set.seed=123,season=1,order=0)
```

5.1. Example 7

Consider again the log numbers of Canadian lynx trappings univariate series from 1821 to 1934, where the $AR(2)$ model is selected based on the $BIC$ criterion using the function `SelectModel` in the R package `FitAR` (McLeod et al. 2013) as a first step in the analysis. Now, we apply the statistic $D_m$ on the fitted model based on the asymptotic distribution and the Monte-Carlo significance test.

```
> require("FitAR")
> lynxData <- log(lynx)
> p <- SelectModel(lynxData, ARModel = "AR", Criterion = "BIC",Best = 1)
> fit <- FitAR(lynxData, p, ARModel = "AR")
> res <- fit$res
> MahdiMcLeod(res,order=p) # The asymptotic distribution of MahdiMcLeod test

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5.984989</td>
<td>2.090909</td>
<td>0.054687987</td>
</tr>
<tr>
<td>10</td>
<td>10.036630</td>
<td>5.857143</td>
<td>0.11522212</td>
</tr>
<tr>
<td>15</td>
<td>21.447021</td>
<td>9.612903</td>
<td>0.014964682</td>
</tr>
<tr>
<td>20</td>
<td>31.810564</td>
<td>13.365854</td>
<td>0.003100578</td>
</tr>
<tr>
<td>25</td>
<td>38.761595</td>
<td>17.117647</td>
<td>0.002040281</td>
</tr>
<tr>
<td>30</td>
<td>43.936953</td>
<td>20.868852</td>
<td>0.002252062</td>
</tr>
</tbody>
</table>
```

> # Use FitAR in FitAR package with Monte-Carlo version of MahdiMcLeod test,
> # users may write their own two R functions. See the following example:
> fit.model <- function(data){
+   p <- SelectModel(data, ARModel = "AR", Criterion = "BIC",Best = 1)
+   fit <- FitAR(data, p, ARModel = "AR")
+   res <- fit$res
+   phiHat <- fit$phiHat
+   sigsqHat <- fit$sigsqHat
+   list(res=res,order=p,phiHat=phiHat,sigsqHat=sigsqHat)
+ }
> Fit <- fit.model(lynxData)
> MahdiMcLeod(Fit) # The asymptotic distribution of MahdiMcLeod statistic
### Portmanteau Test Statistics

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5.984989</td>
<td>2.090909</td>
<td>0.054687987</td>
</tr>
<tr>
<td>10</td>
<td>10.036630</td>
<td>5.857143</td>
<td>0.11522212</td>
</tr>
<tr>
<td>15</td>
<td>21.447021</td>
<td>9.612903</td>
<td>0.014964682</td>
</tr>
<tr>
<td>20</td>
<td>31.810564</td>
<td>13.365854</td>
<td>0.003100578</td>
</tr>
<tr>
<td>25</td>
<td>38.761595</td>
<td>17.117647</td>
<td>0.002040281</td>
</tr>
<tr>
<td>30</td>
<td>43.936953</td>
<td>20.868852</td>
<td>0.002252062</td>
</tr>
</tbody>
</table>

```r
> sim.model <- function(parSpec){
  + phi <- parSpec$phiHat
  + n <- length(parSpec$res)
  + sigma <- parSpec$sigsqHat
  + ts(SimulateGaussianAR(phi, n = n, InnovationVariance = sigma))
+ }
> portest(Fit,test = "MahdiMcLeod", ncores = 4,
  + model=list(sim.model=sim.model,fit.model=fit.model),pkg.name="FitAR")

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5.984989</td>
<td>0.033966034</td>
</tr>
<tr>
<td>10</td>
<td>10.036630</td>
<td>0.059940060</td>
</tr>
<tr>
<td>15</td>
<td>21.447021</td>
<td>0.005994006</td>
</tr>
<tr>
<td>20</td>
<td>31.810564</td>
<td>0.000999001</td>
</tr>
<tr>
<td>25</td>
<td>38.761595</td>
<td>0.000999001</td>
</tr>
<tr>
<td>30</td>
<td>43.936953</td>
<td>0.000999001</td>
</tr>
</tbody>
</table>

> SelectModel(log(lynx),lag.max=15,ARModel="ARp",Criterion="BIC",Best=1)
[1] 1 2 4 10 11

After that, we fit the subset autoregressive AR\(_{(1,2,4,10,11)}\) using the BIC and then we apply \(\mathcal{D}_m\) as before,

```r
> FitsubsetAR <- function(data){
  + FitsubsetAR <- FitARp(data, c(1, 2, 4, 10, 11))
  + res <- FitsubsetAR$res
  + phiHat <- FitsubsetAR$phiHat
  + p <- length(phiHat)
  + sigsqHat <- FitsubsetAR$sigsqHat
  + list(res=res,order=p,phiHat=phiHat,sigsqHat=sigsqHat)
+ }
> SimsubsetARModel <- function(parSpec){
  + phi <- parSpec$phiHat
  + n <- length(parSpec$res)
  + sigma <- parSpec$sigsqHat
  + ts(SimulateGaussianAR(phi, n = n, InnovationVariance = sigma))
+ }
> Fitsubset <- FitsubsetAR(lynxData)
> MahdiMcLeod(Fitsubset)
```
The Monte-Carlo version of the statistic $D_m$ and its approximation asymptotic distribution suggest that the subset AR model is an adequate model.

### 5.2. Example 8

Consider again fitting a VAR ($k$), $k = 1, 3, 5$ model to the monthly log returns of the IBM stock and the S&P 500 index from January 1926 to December 2008 with 996 observations (Tsay 2010, chapter 8).

```r
> data("IbmSp500")
> ibm <- log(IbmSp500[, 2] + 1) * 100
> sp5 <- log(IbmSp500[, 3] + 1) * 100
> z <- data.frame(cbind(ibm, sp5))
> FitIBMSP5001 <- ar.ols(z, aic = FALSE, intercept = TRUE, order.max = 1)
> MahdiMcLeod(FitIBMSP5001)

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>30.94638</td>
<td>12.36364</td>
<td>0.002451865</td>
</tr>
<tr>
<td>10</td>
<td>54.96232</td>
<td>27.42857</td>
<td>0.001374481</td>
</tr>
<tr>
<td>15</td>
<td>71.92499</td>
<td>42.45161</td>
<td>0.003150107</td>
</tr>
<tr>
<td>20</td>
<td>92.18933</td>
<td>57.46341</td>
<td>0.002479329</td>
</tr>
<tr>
<td>25</td>
<td>113.50448</td>
<td>72.47059</td>
<td>0.001479682</td>
</tr>
<tr>
<td>30</td>
<td>131.84170</td>
<td>87.47541</td>
<td>0.001535085</td>
</tr>
</tbody>
</table>
```

```r
> portest(FitIBMSP5001, test = "MahdiMcLeod", ncores = 4)
```
Portmanteau Test Statistics

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>30.94638</td>
<td>0.000999001</td>
</tr>
<tr>
<td>10</td>
<td>54.96232</td>
<td>0.001998002</td>
</tr>
<tr>
<td>15</td>
<td>71.92999</td>
<td>0.003996004</td>
</tr>
<tr>
<td>20</td>
<td>92.18933</td>
<td>0.002997003</td>
</tr>
<tr>
<td>25</td>
<td>113.50448</td>
<td>0.001998002</td>
</tr>
<tr>
<td>30</td>
<td>131.84170</td>
<td>0.000999001</td>
</tr>
</tbody>
</table>

```r
> FitIBMSP5003 <- ar.ols(z, aic = FALSE, intercept = TRUE, order.max = 3)
> MahdiMcLeod(FitIBMSP5003)
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>8.204407</td>
<td>4.363636</td>
<td>0.10439490</td>
</tr>
<tr>
<td>10</td>
<td>26.338795</td>
<td>19.428571</td>
<td>0.13491932</td>
</tr>
<tr>
<td>15</td>
<td>41.032583</td>
<td>34.451613</td>
<td>0.20425337</td>
</tr>
<tr>
<td>20</td>
<td>59.566550</td>
<td>49.463415</td>
<td>0.15389576</td>
</tr>
<tr>
<td>25</td>
<td>80.445483</td>
<td>64.470588</td>
<td>0.08650118</td>
</tr>
<tr>
<td>30</td>
<td>98.529183</td>
<td>79.475410</td>
<td>0.07265450</td>
</tr>
</tbody>
</table>

```r
> portest(FitIBMSP5003, test = "MahdiMcLeod", ncores = 4)
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>8.204407</td>
<td>0.02397602</td>
</tr>
<tr>
<td>10</td>
<td>26.338795</td>
<td>0.03896104</td>
</tr>
<tr>
<td>15</td>
<td>41.032583</td>
<td>0.09090909</td>
</tr>
<tr>
<td>20</td>
<td>59.566550</td>
<td>0.07592408</td>
</tr>
<tr>
<td>25</td>
<td>80.445483</td>
<td>0.04195804</td>
</tr>
<tr>
<td>30</td>
<td>98.529183</td>
<td>0.03565404</td>
</tr>
</tbody>
</table>

```r
> FitIBMSP5005 <- ar.ols(z, aic = FALSE, intercept = TRUE, order.max = 5)
> MahdiMcLeod(FitIBMSP5005)
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.1240808</td>
<td>0.000000</td>
<td>NA</td>
</tr>
<tr>
<td>10</td>
<td>7.6633386</td>
<td>11.428571</td>
<td>0.7738564</td>
</tr>
<tr>
<td>15</td>
<td>19.3087716</td>
<td>26.45161</td>
<td>0.8397923</td>
</tr>
<tr>
<td>20</td>
<td>35.8167000</td>
<td>41.46341</td>
<td>0.7178773</td>
</tr>
<tr>
<td>25</td>
<td>55.0094785</td>
<td>56.47059</td>
<td>0.5301989</td>
</tr>
<tr>
<td>30</td>
<td>71.9562981</td>
<td>71.47541</td>
<td>0.4618016</td>
</tr>
</tbody>
</table>

```r
> portest(FitIBMSP5005, test = "MahdiMcLeod", ncores = 4)
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.1240808</td>
<td>0.9210789</td>
</tr>
<tr>
<td>10</td>
<td>7.6633386</td>
<td>0.5814186</td>
</tr>
<tr>
<td>15</td>
<td>19.3087716</td>
<td>0.6113886</td>
</tr>
<tr>
<td>20</td>
<td>35.8167000</td>
<td>0.4315684</td>
</tr>
<tr>
<td>25</td>
<td>55.0094785</td>
<td>0.2467532</td>
</tr>
<tr>
<td>30</td>
<td>71.9562981</td>
<td>0.2157842</td>
</tr>
</tbody>
</table>
While the fitted $\text{VAR}(1)$ model is rejected, the $D_m$ test based on the asymptotic distribution suggests that the fitted $\text{VAR}(3)$ and $\text{VAR}(5)$ maybe consider to be an adequate model, whereas the Monte-Carlo version of this test is only supports the claim that the fitted $\text{VAR}(5)$ is an adequate model.

5.3. Example 9

In this example, we consider the quarterly time series, 1960–1982, of West German investment, income, and consumption studied before.

We apply the statistic $D_m$ on the fitted $\text{VAR}(2)$ model based on the asymptotic distribution and the Monte-Carlo significance test,

```r
> data("WestGerman")
> DiffData <- matrix(numeric(3 * 91), ncol = 3)
> for (i in 1:3) DiffData[, i] <- diff(log(WestGerman[, i]), lag = 1)
> FitWG <- ar.ols(DiffData, aic = FALSE, order.max = 2, intercept = FALSE)
> MahdiMcLeod(FitWG, lags = c(5, 10, 15))

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>20.90960</td>
<td>18.81818</td>
<td>0.3310523</td>
</tr>
<tr>
<td>10</td>
<td>52.17337</td>
<td>52.71429</td>
<td>0.4951414</td>
</tr>
<tr>
<td>15</td>
<td>91.80348</td>
<td>86.51613</td>
<td>0.3283405</td>
</tr>
</tbody>
</table>

> portest(FitWG, lags = c(5, 10, 15), test = "MahdiMcLeod", ncores = 4)

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>20.90960</td>
<td>0.2837163</td>
</tr>
<tr>
<td>10</td>
<td>52.17337</td>
<td>0.5624376</td>
</tr>
<tr>
<td>15</td>
<td>91.80348</td>
<td>0.5854146</td>
</tr>
</tbody>
</table>

After that we apply the MahdiMcLeod test on the squared residuals of the fitted $\text{VAR}(2)$ model to check for heteroskedasticity,

```r
> MahdiMcLeod(FitWG, lags = c(5, 10, 15), squared.residuals = TRUE)
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>df</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>41.91791</td>
<td>18.81818</td>
<td>0.0016724112</td>
</tr>
<tr>
<td>10</td>
<td>85.20565</td>
<td>52.71429</td>
<td>0.0030577666</td>
</tr>
<tr>
<td>15</td>
<td>137.96484</td>
<td>86.51613</td>
<td>0.0003672143</td>
</tr>
</tbody>
</table>

> portest(FitWG, lags = c(5, 10, 15), test = "MahdiMcLeod", squared.residuals = TRUE, ncores = 4)

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>41.91791</td>
<td>0.2967033</td>
</tr>
<tr>
<td>10</td>
<td>85.20565</td>
<td>0.2267732</td>
</tr>
<tr>
<td>15</td>
<td>137.96484</td>
<td>0.1318681</td>
</tr>
</tbody>
</table>
The asymptotic chi-square distribution of MahdiMcLeod test suggest that to reject that null hypothesis of constant variance, whereas the Monte-Carlo version does not show any heteroskedasticity.

5.4. Example 10

Consider again the econometric model of aggregate demand in the U.K. where we chose the Cn: Consumers’ expenditure on durable goods series and the first model 1a as fitted by Prothero and Wallis (1976) in Table 1 to EconomicUK data.

```r
> require("forecast")
> cd <- EconomicUK[,1]
> cd.fit <- Arima(cd,order=c(0,1,0),seasonal=list(order=c(0,1,1),period=4))
```

After fitting SARIMA \((0,1,0)(0,1,1)_4\), we apply the usual \(D_m\) test statistic as well as the seasonal version of \(D_m\) test statistic. The asymptotic distribution and the Monte-Carlo significance test suggest that the model is good.

```r
> MahdiMcLeod(cd.fit,lags=c(5,10),season=1) ## Asympt. dist. for usual check

   lags statistic   df  p-value
      5   1.700823  3.090909  0.6532001
     10   3.714068  6.857143  0.7999453

> MahdiMcLeod(cd.fit,lags=c(5,10),season=4) ## Asympt. dist. for seasonal check

   lags statistic   df  p-value
      5   0.6612291  3.090909  0.8918977
     10   1.5718612  6.857143  0.9771575

> portest(cd.fit,lags=c(5,10),ncores=4) ## MC check for seasonality

   lags statistic  p-value
      5   1.700823  0.6003996
     10   3.714068  0.4795205

> detach(package:forecast)
```

6. Generalized Durbin-Watson test statistic

The classical test statistic that is very useful in diagnostic checking in time series regression and model selection is the Durbin-Watson statistic (Durbin and Watson 1950, 1951, 1971). This test statistic may be written as

\[
d = \frac{\sum_{t=2}^{n}(\hat{e}_t - \hat{e}_{t-1})^2}{\sum_{t=1}^{n} \hat{e}_t^2},
\]  
(9)
where $\hat{e}_t, t = 1, 2, \ldots, n$ are the OLS residuals.

Under the null hypothesis of the absence of the autocorrelation of the disturbances, in particular at lag 1, the test statistic, $d$, is a linear combination of chi-squared variables and should be close to 2, whereas small values of $d$ indicate positive correlation.

In econometric data, we have many cases in which the error distribution is not normal with a higher-order autocorrelation than AR(1) or the exogenous variables are nonstochastic where the dependent variable is in a lagged form as an independent variable. With these cases, the Durbin-Watson test statistic using the asymptotic distribution is no accurate. For such cases, we include, in our package portes, the two arguments `test = "other"` and `fn`, so that the Monte-carlo version of the generalized Durbin-Watson test statistic at lag $\ell$ can be calculated.

### 6.1. Example 11

Consider the annual U.S. macroeconomic data from the year 1963 to 1982 with two variables, `consumption`: the real consumption and `gnp`: the gross national product. Data was studied by Greene (1993, Chapter 7, p. 221, Table 7.7) and is available from the package `lmtest` (Hothorn, Zeileis, Farebrother, Cummins, Millo, and Mitchell 2017) under the name `USDistLag`.

First, we fit the distributed lag model as discussed in Greene (1993, Example 7.8) as follows,

```r
> # install.packages("lmtest") is needed
> require("lmtest")
> data("USDistLag")
> usdl <- stats::na.contiguous(cbind(USDistLag, lag(USDistLag, k = -1)))
> colnames(usdl) <- c("con", "gnp", "con1", "gnp1")
> fm1 <- lm(con ~ gnp + con1, data = usdl)
```

Then we write R code function `fn()` returns the generalized Durbin-Watson test statistic so that we can pass it to the argument `fn` inside the function `portest()`.

```r
> fn <- function(obj, lags){
+   test.stat <- numeric(length(lags))
+   for (i in 1:length(lags))
+     test.stat[i] <- -sum(diff(obj, lag=lags[i])^2)/sum(obj^2)
+   test.stat
+ }
```

After that we apply the Monte-carlo version of the generalized Durbin-Watson test statistic at lags 1, 2, and 3, using the nonparametric bootstrap residual, which clearly detects a significant positive autocorrelation at lag 1.

```r
> portest(fm1, lags=1:3, test = "other", fn = fn, ncores = 4, innov.dist= "bootstrap")
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.356622</td>
<td>0.03096903</td>
</tr>
<tr>
<td>2</td>
<td>2.245157</td>
<td>0.73426573</td>
</tr>
<tr>
<td>3</td>
<td>2.488189</td>
<td>0.92907093</td>
</tr>
</tbody>
</table>
When residual autocorrelation is detected, sometimes simply taking first or second differences is all that is needed to remove the effect of autocorrelation (McLeod, Yu, and Mahdi 2012).

```r
> fm2 <- lm(con ~ gnp + con1, data = diff(usdl,differences=1))
```

After differencing, the Monte-Carlo version of the Durbin-Watson test statistic fail to reject the reject the null hypothesis of no autocorrelation and suggest that the differencing model is an adequate one.

```r
> portest(fm2, lags=1:3, test = "other", fn = fn, ncores = 4, innov.dist= "bootstrap")
```

<table>
<thead>
<tr>
<th>lags</th>
<th>statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.346099</td>
<td>0.719281</td>
</tr>
<tr>
<td>2</td>
<td>1.404779</td>
<td>0.183816</td>
</tr>
<tr>
<td>3</td>
<td>1.335600</td>
<td>0.232767</td>
</tr>
</tbody>
</table>

```r
> detach(package:lmtest)
```

### References


**Affiliation:**

Esam Mahdi  
Department of Mathematics  
Islamic University of Gaza  
E-mail: emahdi@iugaza.edu.ps  
URL: http://site.iugaza.edu.ps/emahdi  

A. Ian McLeod  
Department of Statistical and Actuarial Sciences  
University of Western Ontario  
E-mail: aim@stats.uwo.ca  
URL: http://www.stats.uwo.ca/mcleod