Teaching Size and Power Properties of Hypothesis Tests Through Simulations

Suleyman Taspinar
Graduate Center, City University of New York

Osman Dogan
Istanbul Ulasm A.S.
Teaching Corner

Süleyman Taşpınar* and Osman Doğan

Teaching Size and Power Properties of Hypothesis Tests Through Simulations

DOI 10.1515/jem-2015-0014
Previously published online January 22, 2016

Abstract: In this study, we review the graphical methods suggested in Davidson and MacKinnon (Davidson, Russell, and James G. MacKinnon. 1998. “Graphical Methods for Investigating the Size and Power of Hypothesis Tests.” *The Manchester School* 66 (1): 1–26.) that can be used to investigate size and power properties of hypothesis tests for undergraduate and graduate econometrics courses. These methods can be used to assess finite sample properties of various hypothesis tests through simulation studies. In addition, these methods can be effectively used in classrooms to reinforce students’ understanding of basic hypothesis testing concepts such as Type I error, Type II error, size, power, p-values and under-or-over-sized tests. We illustrate the procedural aspects of these graphical methods through Monte Carlo experiments, and provide the implementation codes written in Matlab and R for the classroom applications.

Keywords: hypothesis tests; P value discrepancy plots; P value plots; power; size; size-power curves.

JEL Classification: C13; C21; C31.

1 Introduction

Undergraduate econometrics textbooks introduce basic hypothesis testing procedures and related concepts. In a sequence of courses on econometrics, students learn various tests to appraise the validity of null hypotheses, but they do not learn how to assess the finite sample properties of these tests for the purpose of ranking them in terms of their performance. As a result, students may cope with the procedural aspect of hypothesis testing, forming hypotheses and testing them, but struggle to understand the mechanism underpinning testing. In this study, we review simple graphical methods that can be used in classrooms to demonstrate to students the effective ways of assessing the finite sample properties of hypothesis tests.

The finite sample properties of specification testing and other forms of hypothesis tests are investigated in the literature through Monte Carlo experiments. It is often challenging to provide the main results of these studies in classrooms as the subject is not well treated in most undergraduate and graduate textbooks (Amemiya 1985; Johnston and Dinardo 1996; Gujarati and Porter 2009; Stock and Watson 2010; Wooldridge 2010, 2012; Dougherty 2011; Greene 2011).1 For example, Wooldridge (2012, Chapter 8) and Greene (2011, Chapter 8) review various tests that can be used to detect the presence of heteroskedasticity in a linear regression model. Although the hypothesis testing concepts and the procedural aspect of various tests are covered in detail, the size and power properties of tests for the purpose of evaluating the performance of these tests

---

1 The size and power properties are covered through simple examples in Martin, Hurn, and Harris (2012). However, the authors do not use any graphical method in the depiction of these properties.

*Corresponding author: Süleyman Taşpınar, Economics Program, Queens College, The City University of New York, USA, E-mail: STaspinar@qc.cuny.edu
Osman Doğan: Istanbul Ulaşım A.Ş., Project Department, Istanbul, Turkey
are not covered. That is, the procedural aspects of assessing the size and power properties of a particular test are not covered as these aspects require simulation techniques.

There are at least two pedagogical reasons to introduce the procedural aspects of examining the finite sample performance of a test. First, students can learn the simulation technique, known as Monte Carlo experiments, to study the exact distribution of a test statistic in finite samples. Monte Carlo experiments, in general, are used to numerically simulate random processes to generate associated sampling distributions for the purpose of evaluating features of distributions (Hendry 1984). This simulation technique can also be used to compare the finite sample distributions of alternative test statistics. Second, the study of this subject can help students to better learn the crucial concepts such as Type I error, Type II error, size, power and over or under-sized test. These concepts usually remain abstract for students as most of the textbooks present them by simply providing formal definitions or formal notational representations without illustrating through computational examples.

In hypothesis testing, there are two kinds of errors. The first one is called the Type I error, and it occurs when a true null hypothesis (H0) is rejected against an alternative hypothesis (H1). The second one is called the Type II error, and it occurs when a false null hypothesis is not rejected against the alternative hypothesis. A decision of whether or not to reject a null hypothesis can be either true or false which can be due to a Type I error or a Type II error. Hence, the truth of a decision can not be determined with certainty. A hypothesis testing rule is designed to minimize the probability of committing a Type I error and a Type II error (Zellner 1983, p. 166). In the Neyman-Pearson (NP) sampling theory, the probability of a Type I error is not minimized instead is predetermined and set to a small number. In this respect, the first step in determining a rejection rule is to choose a level for the probability of Type I error which is usually denoted by \( \alpha \) for a test. The probability of a Type I error is also called a significance level or a nominal size. In the second step, the probability of a Type II error is minimized for a given level of a significance level.

The rejection rule becomes operational once a test statistic is defined. Let \( T \) be a test statistic and \( t \) be an estimated value of \( T \) for a particular sample. A test statistic, as a function of a random sample, is a random variable with a distribution function. The finite sample distribution of the test statistic, in general, is unknown and is approximated by an asymptotic distribution. The distribution of the test statistic and the predetermined significance level are used to determine a critical value, \( c \), for the purpose of defining a rejection region. For example, assume that the alternative hypothesis is two-sided and \( \alpha = 0.05 \) then the critical value is the 97.5th percentile in the distribution of \( T \) under the assumption that the null hypothesis is true. Here, the rejection region consists of values of \( t \) that are larger than \( c \). The \( t \) values that fall in this region lead the rejection of the null hypothesis.

The above illustration shows that the critical value, determined through the distribution of the test statistic, specifies the rejection region. As stated, the distribution used to determine a critical value is generally an asymptotic distribution that is assumed to be approximated to the finite sample distribution of the test statistic well enough. In order to ascertain whether the asymptotic approximation provides an accurate approximation to the finite sample distribution, the size and power properties of tests are usually investigated. The size properties can be delineated through simulation studies by investigating the rejection frequencies of a test when the data generating process (DGP) satisfies the null hypothesis. Through this exercise the actual size, i.e. the rejection frequency of a test, can be obtained and then be compared with the nominal size. A smaller discrepancy between the nominal size and the actual size suggests that the asymptotic distribution used to determine the critical value approximates the finite sample distribution of the test statistic well enough. If the actual size is greater than the nominal size, the test is over-sized and leads to the rejection of the null hypothesis more often than predicted by the asymptotic distribution. In the case where the actual size is less than the nominal size, the test is under-sized and results in rejection of the null hypothesis less often than predicted by the asymptotic distribution.

2 As stated in Hendry (1984, p. 942), the term “simulation” can be used interchangeably with the phrase “Monte Carlo experiments”.

DE GRUYTER
The power of a test is related to the probability of Type II error. Formally, the power of a test equals one minus the probability of committing to a Type II error. Hence, the minimization of Type II error results in maximization of the power. In simulation studies, the calculation of (size-adjusted) power first requires a simulation study in which the DGP satisfies the null hypothesis. Through this first simulation, the actual critical values corresponding to various size levels are determined. In a second simulation, the DGP satisfies the alternative hypothesis. In this second simulation, the rejection rates calculated by using the actual critical values obtained from the first simulation give the power of the test at various size levels (Martin et al. 2012).

In this study, we provide simple steps that can be used to investigate the size and power properties of hypothesis tests. In particular, we review important graphical methods that can be used to study these properties and provide implementation codes to make these methods easy for the classroom applications. These graphical methods, suggested by Davidson and MacKinnon (1998), are based on the empirical distributions of p-values of test statistics and can be used to demonstrate simulation results on the size and power properties in a much more informative way. We illustrate these methods for testing the existence of potential autoregressive conditional heteroskedasticity (ARCH) effects in financial time series.

The remaining of this article is organized in the following way. In Section 2, we review the graphical methods for the investigation of the size and power properties of tests. In this Section, we also provide algorithms for easy implementations. In Section 3, we illustrate the implementation of these graphical methods through a simulation study. Section 4 concludes.

2 Size and Power Properties of Hypothesis Tests

The conventional way of reporting simulation results on the size and power properties of hypothesis tests is through tables rather than graphs. Tables span multiple pages and require much more attention to discern some meaningful patterns. The disadvantages of the use of tables are discussed in Davidson and MacKinnon (1998) and Arribas-Bel, Koschinsky, and Amaral (2012) in details. Davidson and MacKinnon (1998) suggest three graphical methods for presenting the simulation results. For the size properties, the P value and P value discrepancy plots are suggested. The power properties are illustrated through the size-power curves. As we will show, these plots visualize the empirical distributions of p-values generated under either the null hypothesis or the alternative hypothesis in different ways.

2.1 Graphical Methods for Investigating Size Properties

The P value and P value discrepancy plots suggested in Davidson and MacKinnon (1998) are based on the empirical distribution functions (EDF) of p-values. Let \( t_j \) for \( j = 1, \ldots, R \) be the \( R \) realizations of \( T \) generated in a Monte Carlo experiment. A Monte Carlo experiment is simply a numerical simulation to calculate the statistic of interest, i.e. \( t_j \), through many replications of the DGP. Let \( G(x) \) be the cumulative distribution function (CDF) of the asymptotic distribution of \( T \) evaluated at \( x \). Then, the p-value associated with \( t_j \) denoted by \( p(t_j) \) is given by \( p(t_j) = 1 - G(t_j) \). Intuitively, the p-value associated with \( t_j \) is the significance level of the test when the critical value is \( t_j \). Let \( F(x) \) be the CDF of \( p(T) \) evaluated at \( x \). An estimate of \( F \) can be constructed simply from the EDF of \( p(t_j) \) (Hendry 1984, p. 958). Consider a sequence of points denoted by \( x_i \) for \( i = 1, \ldots, m \) from the interval \((0, 1)\). Then, the estimate of the CDF of \( p(T) \) is given by

\[
\hat{F}(x_i) = \frac{1}{R} \sum_{j=1}^{R} \mathbf{1}(p(t_j) \leq x_i)
\]

3 In the literature, QQ plots are also used to compare quantiles of asymptotic and finite sample distributions. There are at least two disadvantages of this plot: (i) there is no natural scale for the axes of this plot, and (ii) it can not depict how the actual size values are related to the nominal size values. For details, see Davidson and MacKinnon (1998).
where \(I(\cdot)\) is the indicator function. Note that the evaluation points, \(x_i\), can be considered as the significant levels. Following Davidson and MacKinnon (1998), we choose the following sequence for \(x_i\)

\[
x_i = \{0.1 \times 10^{-8}, 0.2 \times 10^{-8}, 0.5 \times 10^{-8}, \ldots, 0.1 \times 10^{-3}, 0.2 \times 10^{-3}, 0.5 \times 10^{-3}, 0.001, 0.002, \ldots, 0.010, 0.015, 0.020, \ldots, 0.990, 0.991, \ldots, 0.999\}.
\]

(2)

Given the estimate in (1), the P value plot is defined as the plot of \(\hat{F}(x_i)\) against \(x_i\) under the assumption that the true DGP is characterized by the null hypothesis. Similarly, the P value discrepancy plot is defined as the plot of \(\hat{F}(x_i) - x_i\) against \(x_i\) under the assumption that the true DGP satisfies the null hypothesis. The interpretation of these plots is based on the fact that \(p(T)\) has a uniform distribution over \((0, 1)\). To establish this fact, we show that \(\Pr(p(T) \leq y) = y\) for \(0 < y < 1\). For this purpose, we assume that \(G\) is strictly increasing. Then,

\[
\Pr(p(T) \leq y) = \Pr(1 - G(T) \leq y) = \Pr(G(T) \geq 1 - y) = \Pr(G^{-1}(G(T)) \geq G^{-1}(1 - y)) = \Pr(T \geq G^{-1}(1 - y)) = 1 - G(G^{-1}(1 - y)) = y.
\]

(3)

The above result indicates that if \(G\) used to calculate p-values is correct, \(p(T)\) has a uniform distribution over \((0, 1)\). This result also indicates that when \(\hat{F}(x)\) is plotted against \(x\), the resulting plot should be close to the 45 degree line. Hence, the P value plot obtained by a plot of \(\hat{F}(x_i)\) against \(x_i\) should be close to the 45 degree line. Therefore, the P value discrepancy plot highlights the differences between the empirical distribution function and the 45 degree line. The large discrepancies from the horizontal axis in a P value discrepancy plot for a test statistic suggest that the finite sample distribution of the test statistic differs from the asymptotic distribution used to determine the critical values.

To assess the significance of discrepancies in a P value discrepancy plot, we follow Anselin et al. (1996) and Anselin (2001) to construct a 95% confidence interval for a nominal size by using a normal approximation to the binomial distribution. Let \(\alpha\) be a nominal size value, and \(R\) be the number of resamples in a Monte Carlo experiment. Using a normal approximation to the binomial distribution, a 95% confidence interval centered on \(\alpha\) would be given by \(\alpha \pm 1.96(\alpha(1 - \alpha)/R)^{1/2}\), and thus it would include rejection rates between \(\alpha - 1.96(\alpha(1 - \alpha)/R)^{1/2}\) and \(\alpha + 1.96(\alpha(1 - \alpha)/R)^{1/2}\). We use this approach to insert a 95% point-wise confidence interval for each \(x_i\) in a P value discrepancy plot.

We summarize the required steps to construct the P value and P value discrepancy plots in the following algorithm.

**Algorithm 1**
1. Define the number of resamples, \(j = 1, \ldots, R\), e.g. \(R = 1e4\).
2. Define the set of evaluation points, \(x_i\).
3. For each resample \(j = 1, \ldots, R\) generate the DGP that satisfies the null hypothesis and obtain the test statistic \(T_i\). For each test statistic, calculate \(p(t)\) by using the asymptotic distribution of the test statistic. Then, calculate \(I(p(t) \leq x_i)\) for each \(x_i\).
4. Calculate the empirical CDF using (1), i.e. \(\hat{F}(x_i)\) for each \(x_i\).
5. Obtain the P value plots by plotting corresponding \(\hat{F}(x_i)\) against \(x_i\).
6. For the P value discrepancy plots, plot \((\hat{F}(x_i) - x_i)\) against \(x_i\).

### 2.2 Graphical Methods for Investigating Power Properties

For the investigation of the power properties, size adjusted power curves are used in the literature. In this approach, the actual critical values generated in a simulation where the DGP satisfies the null hypothesis are

\[\text{Note that the strictly increasing assumption is required to ensure that } G^{-1} \text{ is well defined, and hence } G^{-1}G(x) = x. \text{ If } G \text{ is not strictly increasing then we can define } G^{-1}(x) = \inf\{x: G(x) \geq y\}, \text{ which is well defined for } 0 < y < 1. \text{ For details, see Casella and Berger (2002, pp. 54–55).}\]
used to calculate the power curves. Davidson and MacKinnon (1998) suggest an alternative approach, where
the power curve is generated against the actual size obtained under the corresponding null hypothesis.
Again, there are two experiments in this approach. In the first experiment, the DGP belonging to the alterna-
tive hypothesis is used to generate the EDF of p-values. We denote the resulting EDF by $\hat{F}(x)$. The DGP used
in the second experiment satisfies the null hypothesis, and let $\hat{F}(x)$ be the resulting EDF of p-values. Then,
the size-power curve is obtained by plotting $\hat{F}(x_i)$ against $\hat{F}(x_i)$ for $i=1, \ldots, m$.

We summarize steps needed to construct the size-power curves in the following algorithm.

Algorithm 2
1. Define the number of resamples, $R$, e.g. $R=1e4$.
2. Define the set of evaluation points, $x_i$s.
3. For each resample $j=1, \ldots, R$ generate the DGP that satisfies the alternative hypothesis and obtain the test
statistic $t_j$. For each test statistic, calculate $p(t_j)$ by using the asymptotic distribution of the test statistic.
Then, calculate $1(p(t_j) \leq x_i)$ for each $x_i$.
4. Calculate the empirical CDF using (1), i.e. $\hat{F}(x_i)$ for each $x_i$.
5. Obtain the size-power curves by plotting $\hat{F}(x_i)$ against $\hat{F}(x_i)$, where $\hat{F}(x_i)$ is the empirical CDF gener-
ated in Algorithm 1.

3 Demonstrations

We will illustrate the use of P value, P values discrepancy and size-power curve plots through an application
in financial time series analysis. It is well-known in financial time series that return series usually show no or
little autocorrelation. However, serial independence between the squared values of the return series is often
rejected, indicating the existence of nonlinear relationships between subsequent observations. Furthermore,
volatility of the return series appears to be clustered in the sense that small movements in returns tend to be
followed by small returns in the next period, whereas large movements in returns tend to be followed by large
returns in the next period. The idea behind an ARCH model is to account for these characteristics of the return
series by modeling the autocorrelation structure in the volatility (variance) of returns (Engle 1982). The ARCH
specification allows the return series to be serially uncorrelated but dependent. This dependence is formu-
lated by a simple quadratic function of the lagged values of the return series. More formally, suppose that
the return series $\{y_t\}$ is weakly stationary (or covariance stationary) such that $\mathbb{E}(y_t^2)<\infty$. Then, the ARCH(m)
model is given by

$$
\begin{align*}
    y_t &= \sigma_t \epsilon_t \\
    \sigma_t^2 &= \alpha_0 + \alpha_1 y_{t-1}^2 + \cdots + \alpha_m y_{t-m}^2
\end{align*}
$$


where $\theta=(\alpha_0, \alpha_1, \ldots, \alpha_m)'$ is a vector of parameters to be estimated, and $\{\epsilon_t\}$ is a sequence of i.i.d. random
variables with mean 0 and variance 1. The parameter vector can be estimated by the maximum likelihood
estimator.5

Engle (1982) suggests a Lagrange Multiplier (LM) test for the detection of ARCH effects. For our model, the
null and the alternative hypotheses are stated as

$$
H_0: \alpha_1 = \alpha_2 = \cdots = \alpha_m = 0,
H_1: \alpha_1 \neq 0 \text{ for some } i \in \{1, \ldots, m\}.
$$

Under $H_0$, we have the restricted model in which there is no ARCH effect and $y_t$ simply has mean zero,
and constant conditional variance of $\alpha_0$. The LM test is based on the idea that the sample gradients evaluated

5 For details on estimation, among others see Greene (2011), Francq and Zakoian (2011) and Tsay (2013).
at the restricted parameter estimates should be close to zero, when the restrictions are valid. The LM test statistic can be computed as $T \times R^2$, where $R^2$ value is obtained from the least squares regression of $y_i^2$ on an intercept and $m$ lagged values of $y_i^2$. Under $H_0$, we have

$$T \times R^2 \sim \chi_m^2.$$  

(5)

A second test known as the Ljung–Box test is based on the autocorrelation in the squared return series $\{y_i^2\}$. The idea of the test is based on the fact that the sample autocorrelation function (ACF) can capture the dynamic dependence of the data (McLeod and Li 1983). The lag-$k$ autocorrelation of $y_i^2$ is given by

$$\rho_k = \text{cov}(y_i^2, y_{i-k}^2) / \text{var}(y_i^2).$$

Then, the ACF of $y_i^2$ is defined as the sequence of autocorrelations, $\{\rho_k\}$. Suppose we have a sample of $T$ observations on $\{y_i^2\}$. Then, the lag-$k$ sample autocorrelation of $y_i^2$ is defined as

$$\hat{\rho}_k = \frac{\sum_{t=k+1}^T (y_{t-k}^2 - \bar{y}_t^2)(y_t^2 - \bar{y}_t^2)}{\sum_{t=1}^T (y_t^2 - \bar{y}_t^2)^2}.$$  

(6)

for $0 \leq k < T-1$. The Ljung–Box statistic tests the joint significance of $m$ lags of $y_i^2$. More specifically, the null and the alternative hypotheses are

$$H_0: \rho_1 = \rho_2 = \cdots = \rho_m = 0$$

$$H_1: \rho_i \neq 0 \text{ for some } i \in \{1, \ldots, m\}.$$  

Then, the test statistic is given by (Ljung and Box 1978; McLeod and Li 1983)

$$Q(m) = T(T+2) \sum_{j=1}^m \frac{\hat{\rho}_j^2}{T-j} \sim \chi_m^2.$$  

(7)

Below we provide Matlab scripts that implement Algorithms 1 and 2. For the data generating process in both algorithms, we utilize monthly log returns of Intel stock taken from Tsay (2013, p. 182) and historical stock prices from Yahoo Finance. The sample involves monthly observations from January 1973 through June 2015. In both algorithms, we first fit an ARCH(3) specification to the return series and obtain the standardized residuals, $\hat{\epsilon}_i$. Then, using these residuals, Algorithm 1 resamples under the null hypothesis of no ARCH effects, whereas Algorithm 2 resamples under the alternative hypothesis of ARCH effects, generating a process following an ARCH(3) model in each resample.

**Algorithm 1**

1. Clear the memory and set the seed for future reference.

```matlab
clear all; clc;
rng(20150424,'twister');
```

2. Define the number of resamples ($R$), sample size ($T$), and the set of evaluation points $x_j$.

```matlab
R = 1e4; % number of resamples
T = 500; % sample size
% define the set of evaluation points
eval1 = [1e-9,2e-9,5e-9,1e-8,2e-8,5e-8,1e-7,2e-7,5e-7,1e-6,6e-6,6e-6,...
4e-6,5e-5,5e-5,1e-4,2e-4,4e-4];
 eval2 = [0.001:.001:.001,.015:.005:.990,.991:.001:.999];
 ep = [eval1,eval2];
```

---

6 MATLAB® Version 7.11.0. To save space, we leave the R scripts to the Appendix.
3. Count the total number of $x_i$s and define it as $k$. Generate two $R \times k$ matrices (of zeros) to save the rejection frequencies.

```matlab
k = length(ep); % for counting the rejection frequencies
test_LJ0 = zeros(R,k);
test_LM0 = zeros(R,k);
```

4. Upload the data set, calculate the simple log returns and demand the return series.

```matlab
% monthly log returns of Intel stock
st = importdata('m-intcsp7315.txt',','1);
st = st.data; % pull the data from the structure
st = log(1+st(:,2));
st = st - mean(st);
```

5. Fit an ARCH(3) model to the log return series and calculate the standardized residuals, $\hat{\varepsilon}_i$s. We use the `garchpq()` function for the estimation which is available in the UCSD GARCH toolbox written by Sheppard (2007).

```matlab
% set some options for the optimizer
options = optimset('fmincon');
options = optimset(options,'Display','off');
% use the UCSD GARCH toolbox written by Kevin Sheppard (2002)
[par,LL,ht] = garchpq(st,3,0,[],options);
ress = st./sqrt(ht); % the standardized residuals
```

6. Resample under the null hypothesis $R$ times. In each resample, test for ARCH using the LM and Ljung–Box tests. We use the `lmtest2()` and `ljq2()` functions, which are available in the UCSD GARCH toolbox. If the p-value of the test statistic returned by `lmtest2()` is less than the level of the test $x_i$, replace corresponding entry in matrix, `test_LM0` with 1. Repeat the same process with `ljq2()` and `test_LJ0`.

```matlab
% under the null no ARCH effects count the rejections of a true null
for jj=1:R
    dt = randsample(ress,T,true);
    rlm = lmtest2(dt,6);
    test_LM0(jj,rlm.pval(end)<ep) = 1;
    rlj = ljq2(dt,6);
    test_LJ0(jj,rlj.pval(end)<ep) = 1;
end
```

7. Save the result matrices. They will also be used to generate the size power plots.

```matlab
save('H0_LM','test_LM0');
save('H0_LJ','test_LJ0');
```

8. Obtain the empirical CDF of $p(T)$, i.e. $\hat{F}(\chi_i)$.

```matlab
% obtain the empirical c.d.f.
LJ = sum(test_LJ0)/R;
LM = sum(test_LM0)/R;
```

---

7 The UCSD GARCH toolbox is available at http://www.kevinsheppard.com/UCSD_GARCH.
9. Obtain a 95% confidence interval for a nominal size by using the normal approximation to a binomial distribution.

\[
\begin{align*}
\text{lb} &= \text{ep} - 1.96 \times \text{sqrt}(\text{ep} \times (1-\text{ep}) / \text{mit}); \\
\text{ub} &= \text{ep} + 1.96 \times \text{sqrt}(\text{ep} \times (1-\text{ep}) / \text{mit});
\end{align*}
\]

10. Generate P value plots by plotting \( \hat{F}(x) \) against \( x \). Note that, the main attention is typically paid to the Type I errors which are set at levels smaller than 20%. Therefore, this plot and the subsequent plots only show the rejection frequencies for the case where \( x \leq 0.20 \).

```
% p value plot
plot (ep, LJ, '—dg', 'MarkerSize', 6, 'LineWidth', 1.3, 'MarkerEdgeColor', 'k');
hold on;
plot (ep, LM, '—ob', 'MarkerSize', 6, 'LineWidth', 1.3, 'MarkerEdgeColor', 'k');
plot (0:0.001:0.204, 0:0.001:0.204, 'r', 'LineWidth', 1.3)
hold off; grid on;
axis([-0.002 0.202 -0.002 0.204]);
xlabel('Nominal size', 'FontSize', 12);
ylabel('Actual size', 'FontSize', 12);
set(gca, 'XTick', [0:.04:2], 'XTickLabel', {'0', '0.04', '0.08', '0.12', '0.16', '0.20'});
set(gca, 'YTick', [0:.04:2], 'YTickLabel', {'0', '0.04', '0.08', '0.12', '0.16', '0.20'});
legend('Ljung-Box', 'LM', 'location', 'northwest');
```

11. Generate a P value discrepancy plot by plotting \( \hat{F}(x) - x \) against \( x \).

```
% size discrepancy plot
plot (ep, LJ-ep, '—dg', 'MarkerSize', 6, 'LineWidth', 1.3, 'MarkerEdgeColor', 'k');
hold on;
plot (ep, LM-ep, '—ob', 'MarkerSize', 6, 'LineWidth', 1.3, 'MarkerEdgeColor', 'k');
plot (ep, lb-ep, 'r', 'LineWidth', 1.3)
hold off; grid on;
axis([-0.002 0.202 -0.02 0.05]);
xlabel('Nominal size', 'FontSize', 12);
ylabel('Size discrepancy', 'FontSize', 12);
set(gca, 'XTick', [0:.04:2], 'XTickLabel', {'0', '0.04', '0.08', '0.12', '0.16', '0.20'});
set(gca, 'YTick', [-0.02 0.02 0.04], 'YTickLabel', {'-0.02', '0', '0.02', '0.04'});
legend('Ljung-Box', 'LM', 'location', 'northeast');
```

Algorithm 2

1. Clear the memory and set the seed for future reference.

```
clear all; clc;
rng(20150424, 'twister');
```
2. Define the number of resamples ($R$), sample size ($T$), and the set of evaluation points $x_i$s.

\begin{verbatim}
R = 1e4;  % number of resamples
T = 500;  % sample size
% define the set of evaluation points
eval1 = [1e-9, 2e-9, 5e-9, 1e-8, 2e-8, 5e-8, 1e-7, 2e-7, 5e-7, 1e-6, 2e-6, 5e-6, ...
        1e-5, 2e-5, 5e-5, 1e-4, 2e-4, 5e-4];
eval2 = [.001:.001:.010, .015:.005:.990, .991:.001:.999];
ep = [eval1, eval2];
\end{verbatim}

3. Count the total number of $x_i$s and define it as $k$. Generate two $R \times k$ matrices (of zeros) to save the rejection frequencies.

\begin{verbatim}
k = length(ep);
% for counting the rejection frequencies
test_LJ1 = zeros(R,k);
test_LMI = zeros(R,k);
\end{verbatim}

4. Upload the data set, calculate the log simple returns and demean the return series.

\begin{verbatim}
% monthly log returns of Intel stock
st = importdata('m-intoep7315.txt', '', 1);
st = st.data;  % pull the data from the structure
st = log(1+st(:,2));
st = st - mean(st);
\end{verbatim}

5. Fit an ARCH(3) model to the log return series and calculate the standardized residuals, $\hat{\epsilon}_i$s.

\begin{verbatim}
% set some options for the optimizer
options = optimset('fmincon');
options = optimset(options,'Display','off');
% use the UCSD GARCH toolbox written by Kevin Sheppard (2002)
[par,LL,ht] = garchpq(st,3,0,[],options);
ress = st./sqrt(ht);  % the standardized residuals
\end{verbatim}

6. Resample under the alternative hypothesis $R$ times. More specifically, in each resample generate a process from an ARCH(3) specification and test for ARCH effects. If the p value of the test statistic is less than the level of the test $x_i$, replace corresponding entry of the result matrix with 1.

\begin{verbatim}
% under the alternative: ARCH(3) model
uvar = par(1)/(1-par(2)-par(3)-par(4));  % unconditional variance
for jj=1:R
    dt = zeros(T+3+500,1);
    dt(1:3,1) = sqrt(uvar);
    for tt = 4:(T+3+500)
        res_e = randsample(ress,1);
        ht = par'*[1;dt(tt-(1:3)).^2];
        dt(tt) = sqrt(ht)*res_e;
    end
    dt = dt(4+500:end);  % discard the first 500 as burn-ins
    rlm = lmtree2(dt,6);
    test_LMI(jj, rlm.pval(end)<ep) = 1;
    rlj = liq2(dt,6);
    test_LJ1(jj, rlj.pval(end)<ep) = 1;
end
\end{verbatim}
7. Obtain the empirical CDF of $p(T)$, i.e. $\hat{F}(x_i)$.

```matlab
% obtain the empirical c.d.f.
LJ1 = sum(test_LJ1)/R;
LM1 = sum(test_LM1)/R;
```

8. Reload the empirical CDF $\hat{F}(x_i)$ generated in Algorithm 1.

```matlab
% reload the cdf under the null
tmp = load('H0_LJ');
LJ0 = sum(tmp.test_LJ0)/R;
tmp = load('H0_LM');
LM0 = sum(tmp.test_LM0)/R;
```

9. Generate a size power curve by plotting $\hat{F}(x_i)$ against $\hat{F}(x_j)$.

```matlab
% size-power curve plot
plot(LJ0,LJ1,'--dg','LineWidth',1.3,'MarkerSize',6,'MarkerEdgeColor','k');
hold on;
plot(LM0,LM1,'--ob','LineWidth',1.3,'MarkerSize',6,'MarkerEdgeColor','k');
hold off; grid on;
axis([-0.001 0.001 0.08 0.20]);
xlabel('Size','FontSize',12);
ylabel('Power','FontSize',12);
set(gca,'XTick',0:0.04:0.2);
    'XTickLabel',{'.0','0.04','0.08','0.12','0.16','0.20'});
set(gca,'YTick',0:0.2:1);
    'YTickLabel',{'.0','0.2','0.4','0.6','0.8','1'});
legend('Ljung-Box','LM','Location','northwest');
```

Algorithms 1 and 2 produce Figures 1–3. The p value plots are given in Figure 1, which shows the rejection frequencies when the DGP satisfies the null hypothesis. For both tests, there are large discrepancies from the 45 degree line. For example, when the nominal size is 0.04, both test reject the corresponding null hypothesis.
around 8% of times. Both tests are over-sized for the nominal size values smaller than 12% and under-sized for the nominal size values larger than 12%. The P value discrepancy plots provided in Figure 2 highlight the degree of discrepancy from the 45 degree line. It is clear that the rejection rates are not in the 95% point-wise confidence interval in almost all cases. This figure also makes it clear that the Ljung–Box test is slightly more over-sized for the nominal size values smaller than 12%. Overall, Figures 1 and 2 suggest that the empirical distributions, i.e. the finite sample distributions, of both tests differ from the asymptotic \( \chi^2 \) distribution used to establish the critical values. In other words, the asymptotic distributions established under the corresponding null hypothesis for these tests are not approximating the corresponding empirical distributions well enough for the sample size used in our experiment.
The power properties can be analyzed through the size-power curves given in Figure 3. As shown in the figure, the rejection rates of both tests for the corresponding false null hypothesis are around 90% when the actual size is around 4%. The Ljung–Box test has slightly more power than the ML test for the case where the actual size is smaller than 10%. Both tests achieve more power, with rejection rates converging to 100%, when the actual size gradually increases to 20%.

4 Conclusion

Although econometrics textbooks introduce criteria such as bias, consistency and precision to assess various attributes of estimators, majority of them present hypothesis tests without providing the procedural aspects that can be used to evaluate their finite sample properties. The finite sample properties of hypothesis tests can be investigated by the size and power properties. There are many journal articles in which these properties for various tests are presented and investigated. However, there are few textbooks in which these properties are covered to assess the finite sample properties of tests.

In this study, we illustrate these properties through graphical methods and provide detailed implementation algorithms, which can make this subject to be easily adopted for the classroom applications for both undergraduate and graduate students. The three graphical methods that we covered are based on the EDF of p-values of tests, and we show how to interpret these plots to evaluate the size and power properties of hypothesis tests. We present two tests that are used to detect the ARCH effects in financial time series for the demonstrations. We show how to use effectively P values plots, P value discrepancy plots and size-power curves to study the size and power properties of these tests.

Acknowledgments: This research was supported, in part, by a grant of computer time from the City University of New York High Performance Computing Center under NSF Grants CNS-0855217 and CNS-0958379.

Appendix

Listing 1: R Codes for P Value Plots and P Value Discrepancy Plots.

```r
# Install fGarch package
#install.packages("fGarch")
rm(list=ls(all=TRUE))
library(fGarch)
#
#install.packages("FinTS")
library(FinTS)
#
set.seed(20150424, kind="Mersenne-Twister")
#
mit = 1e4 # number of resamples
T = 500 # sample size
# define the levels of the test
eval1 = c(1e-9,2e-9,5e-9,1e-8,2e-8,8e-8,5e-8,8e-8,
1e-7,2e-7,5e-7,1e-6,2e-6,6e-6,
1e-5,2e-5,5e-5,1e-4,2e-4,4e-4)
eval2 = c(seq(from=0.001,to=0.010,by=0.001),seq(from=0.015,to=0.999,by=0.005),
seq(from=0.991,to=0.999, by=0.001))
ep = c(eval1, eval2)
l = length(ep)
#
# count the rejections
test_LJ0 = matrix(0,mit,l)
test_LM0 = matrix(0,mit,l)
#
# monthly log returns of Intel stock
```
Listing 1: (continued)

dx=read.table("m-intcsp7315.txt",header=T)
dx = dx[,2]
dt = log(1+dx)
dt = dt-mean(dt)

# fit an ARCH(3) first and calculate
# the standardized residuals
ml=garchFit("garch(3,0),data=dx,include.mean=F,trace=F"

res=dt/sqrt(ml$h)

# under the null no ARCH effects
# count the rejections of a true null

for(jj in 1:mit){
dt=sample(res,1,replace=TRUE)
LM=ArchTest(dt,lag=6,demean=FALSE)
test_LM0[jj,LM$pv.value<ep] = 1;
rij=Box.test(dt,lag=6,type="Ljung")
test_LJ0[jj,rij$p.value<ep] = 1;
}

# obtain the empirical c.d.f.

LJ = colSums(test_LJ0)/mit
LM = colSums(test_LM0)/mit

write.table(LJ, file="LJ0.txt", sep="\t", row.names=F, col.names=F);
write.table(LM, file="LM0.txt", sep="\t", row.names=F, col.names=F);

# p value plot

plot(ep,LJ,col="green",type="b",lwd=2,lty=2, xlab='Nominal size ',ylab='Actual size ',
     xlim=c(0,20), ylim=c(0,20),pch=0)
lines(ep,LM,col="blue",lwd=2,lty=2,pch=1,type="b")
lines(ep,ep,col="red",lwd=2)
abline(h=0,lty=1,col="black")
grid(lwd = 1, col = "blue")

legend("topleft",# places a legend at the appropriate place
     c("Ljung-Box","LM"), # puts text in the legend
     lty=c(2,2), # gives the legend appropriate symbols (lines
     pch=c(0,1),
     lwd=c(2,2),col=c("green","blue")) # gives the legend lines the correct color and width

# P value discrepancy plot

plot(ep,LJ-ep,col="green",type="b",lwd=2,lty=2,xlab='Nominal size ',ylab='Actual size ',
     xlim=c(0,20), ylim=c(-.05,.08),pch=0)
lines(ep,LM-ep,col="blue",lwd=2,lty=2,pch=1,type="b")
lines(ep,ub-ep,col="red",lwd=2)
lines(ep,ub-ep,col="red",lwd=2)
abline(h=0,lty=1,col="black")
grid(lwd = 1, col = "blue")

legend("topright",# places a legend at the appropriate place
     c("Ljung-Box","LM"), # puts text in the legend
     lty=c(2,2), # gives the legend appropriate symbols (lines
     pch=c(0,1),
     lwd=c(2,2),col=c("green","blue")) # gives the legend lines the correct color and width
Listing 2: R Codes for Size-Power Curves.

```r
# Install fGarch package
# install.packages("fGarch")
rm(list=ls(all=TRUE))
library(fGarch)

# install.packages("FinTS")
library(FinTS)
# set seed
set.seed(20150424, kind="Mersenne-Twister")
#
mit = 1e4 # number of resamples
T = 500 # sample size
# define the levels of the test
evall = c(1e-9.2e-9.5e-9.1e-8.2e-8.5e-8.8e-8, 1e-7.2e-7.5e-7.1e-6.2e-6.5e-6, 1e-6.2e-5.5e-5.1e-4.2e-4.5e-4)
eval2 = c(seq(from=0.001, to=0.010, by=0.001),
          seq(from=0.015, to=0.990, by=0.005),
          seq(from=0.991, to=0.999, by=0.001))
ep = c(eval1, eval2)
n = length(ep)
#
# count the rejections

test_LJ1 = matrix(0, mit, n)
test_LMI = matrix(0, mit, n)
#
# monthly log returns of Intel stock

da=read.table("m-intctp315.txt", header=T)
st = da[,2]
st = log(1+st)
st = st - mean(st)
#
# fit an ARCH(3) first and calculate

ml=garchFit("garch(3,0), data=st, include.mean=F, trace=F")
#
# extract the standardized residuals

ress=sqrt(ml@h.t)
#
# under the alternative: ARCH(3) model

par=ml@fit$matcoef[1:4,]

uvar = par[1]/(1-par[2]-par[3]-par[4])

for(jj in 1:mit){
dt1 = matrix(0,T+3+500,1)
dt1[1:3,1] = sqrt(uvar)
for(tt in 4:(T+3+500)){
  res.s=sample(ress, 1)
  ht = t(par)*q*(res.s[1], [tt-(1:3),1]^2)
  dt1[tt,1] = sqrt(ht)*res.s
}
}
dt1 = dt1[504:1003,1]
rlm = ArchTest(dt1, lags=6, demean=F)
test_LMI[jj, rlmp.value<ep] = 1
rlj = Box.test(dt1*dt1, lag=6, type="Ljung")
test_LJ1[jj, rljp.value<ep] = 1
```
Listing 2: (continued)
}
# LJJ = colSums(test._LJJ)/mit
LM1 = colSums(test._LM1)/mit
# LJO = as.matrix(read.table("LJO.txt",sep="\t"));
LM0 = as.matrix(read.table("LM0.txt",sep="\t"));

# size=plot curves
plot(LJO,LM1,col="green",type="b",lwd=2,lt=2,xlab='Size',ylab='Power',
     xlim=c(0,20),ylim=c(0.1),pch=0)
lines(LM0,LM1,col="blue",lwd=2,lt=2,pch=1,type="b")
# abline(h=0,lt=1,col="black")
grid(lw=1,col="blue")
legend("topleft",c("Ljung–Box","LM"),col="green",lty=1,pch=2)

References