Applied Nonparametric Econometrics, 2013 Fall Project 1

Kun Ren*

November 16, 2013

The classical nonparametric tests are known to require strong assumptions such as simple sample (iid). However, most economic and financial data are not generated in an independent manner. Rather, current economic or financial variables are jointly determined not only by current situations but also by history. Therefore, a major part of real-world data does not satisfy the condition required to apply these nonparametric methods only because the data are in the form of time series. To grasp a rough impression on how classical nonparametric methods perform when they test time series data, we conduct a series of Monte Carlo simulations and do comparisons between Mann-Whitney rank test, Wilcoxon signed rank test, Kolmogorov-Sminorov test, t-test and paired t-test. All the simulations are done in R environment[1] and this document is written with the help of `knitr` package.

The simulation model is under simplified settings. We make a benchmark test of two simulated time series of iid normal random variables $\{x_t\}$ and $\{y_t\}$ and test the null hypothesis that the two time series are generated by identical distribution by Mann-Whitney rank test, Wilcoxon signed rank test, Kolmogorov-Sminorov test, t-test and paired t-test, respectively.

To avoid searching the table of critical values, we directly take certain quantiles of the simulated test statistics in these tests as proxy critical values for the comparing tests of time series with interdependent structures.

*Kun Ren<`renkun@outlook.com`>, Student ID: 27720121152622.

[1]The simulations are all run under R 3.0.2 64-bit environment for Windows.

The next steps are central to the problem this document attemps to shed some light on. We simulate two independent MA(1) processes

$$\begin{aligned} x'_t &= \epsilon_t + \theta\epsilon_{t-1} \\ y'_t &= \kappa_t + \theta\kappa_{t-1} \end{aligned}$$

where $\epsilon_t \sim iid\ N\left(0, \frac{1}{\sqrt{1+\theta^2}}\right)$ and $\kappa_t \sim iid\ N\left(0, \frac{1}{\sqrt{1+\theta^2}}\right)$, respectively. Note that the variance of $\{x'_t\}$ and $\{y'_t\}$ are both

$$\left(1+\theta^2\right)\left(\frac{1}{\sqrt{1+\theta^2}}\right)^2 = 1$$

which neutralizes the effect of different variance between normal random variables and the MA(1) processes. Since normally distributed random variables still follow normal distribution when performed addition, the original normal simulation may be regarded as a comparable version of

$$\begin{aligned} x_t &= \phi_t + \theta\tau_t \\ y_t &= \psi_t + \theta\lambda_t \end{aligned}$$

where $\phi_t \sim iid\ N\left(0, \frac{1}{\sqrt{1+\theta^2}}\right)$, $\tau_t \sim iid\ N\left(0, \frac{1}{\sqrt{1+\theta^2}}\right)$, $\psi_t \sim iid\ N\left(0, \frac{1}{\sqrt{1+\theta^2}}\right)$,$\lambda_t \sim iid\ N\left(0, \frac{1}{\sqrt{1+\theta^2}}\right)$ so that $x_t \sim iid\ N\left(0,1\right)$ and $y_t \sim iid\ N\left(0,1\right)$ just as stated at the beginning. To explicitly remark this makes the normal case concepually comparable with the MA(1) case because the the only difference is that the MA(1) case incorporates the serial correlation as an intrinsic interdependent structure. To conduct more experiments, we also test the effect of different sample sizes from the length of 100 to that of 500 and different degrees of interdependece from MA(1) to MA(2) with the same logic applied.

Before we head to solving the problems, we need to set up a simulation environment where proper functions are defined to allow us to do experiments.

The first step is to make sure the whole simulation is reproducible, therefore we set a random seed, which allows us to replicate exactly the same series of random numbers by setting up the same random seed. Here we arbitrarily choose seed 123.

```r
set.seed(123)
```

To make comparisons of the performance between the case where independently and identically distributed data are tested and the case where serially correlated time series data are tested, we choose the simplest cases of independently, identically, and normally distributed data and time series of different degrees of moving averages. Below we define the data generating processes of these two types.

```r
iid <- function(n,test) {
  d1 <- rnorm(n)
  d2 <- rnorm(n)
  test(d1,d2)
}
ma <- function(n,ma,test) {
  sd <- 1/sqrt(1+ma^2)
  d1 <- arima.sim(model=list(ma=ma), n=n, sd=sd)
  d2 <- arima.sim(model=list(ma=ma), n=n, sd=sd)
  test(d1,d2)
}
```

The functions above specify how we generate and test the data. More specifically, `iid` function takes two arguments: `n` means how many independent observations do we need to draw from a standard normal distribution whereas `test` means the test function to run against the two produced samples that is expected to produce a numeric vector. We shall see an implementation of test function soon. As for the second function defined above, `ma` receives three arguments: `n` means the length of the sample time series data, `ma` is a numeric vector that specifies the parameters of a moving average process, and `test` indicates a test function as the `iid` function also expects.

Then we define a simple simulation function as shown below:

```r
sim <- function(n,fun) {
  data.frame(t(sapply(1:n,fun)))
}
```

The `sim` function repeat the simulation specified by `fun` function for `n` times and put the output data into a data frame.

To compute the power of the tests, we need the critical values. Here we adopt those values from the simulation of normal data. Once we produce some normal data we may calculate their critical quartiles of the test statistics of the size we specify later.

```
quantiles <- function(result,probs) {
  rq <- sapply(1:ncol(result),
               function(i)quantile(result[,i],probs=probs))
  colnames(rq) <- colnames(result)
  rq
}
```

The `quantiles` function allows us to compute the quantiles of each statistic we are interested in and output a numeric matrix that contains the quantile numbers that will be used to evaluate the performance when we use various test methods against time series data.

The next step is to define the power function that calculates the power of each test in a two-tail manner. Formally, we estimate the probability that we mistakenly reject the null hypothesis$\mathcal{H}_0$ when it is true by computing the percentage of cases where the test statistic lies in the area that indicates an rejection in iid case.

```
powers <- function(result,q) {
  lens <- sapply(1:ncol(result),
                 function(i) {
                   len1 <- length(result[result[,i]<=q[1,i],i])
                   len2 <- length(result[result[,i]>q[2,i],i])
                   (len1+len2)/nrow(result)
                 })
  names(lens) <- colnames(result)
  lens
}
```

The `power` function accepts the simulated data and the benchmark quantile numbers as critical values. Then we define a general-purpose `test` function that do simulations according to the arguments. Here we simplify the way higher order moving average process is defined, i.e. if we specify an `order` greater than 1 (say, 2) we will produce the data by $x_t = \epsilon_t + \theta\epsilon_{t-1} + \theta\epsilon_{t-2}$ where although the process is MA(2), its parameter is only θ as we specify by `theta`, which allows us to focus on the behavior of the test performance for different degree of serial correlation.

```r
test <- function(nsim,nsample,size,theta,test.fun,order=1) {
  probs <- c(size/2,1-size/2)
  iidr <- sim(nsim,function(i) iid(nsample,test.fun))
  iidrq <- quantiles(iidr,probs)
  marp <- NULL
  marpf <- function(x) {
    mar <- sim(nsim,
               function(i) ma(nsample,rep(x,order),test.fun))
    powers(mar,iidrq)
  }
  marps <- sapply(theta, marpf)
  marps <- t(marps)
  rownames(marps) <- theta
  marps
}
```

In our testing procedure, we use the quantiles of the iid sample as the boundaries of test statistics for the five tests rather than use critical values in the table directly, mostly for convenience, which will not result in large errors. Finally we implement a series of tests by defining `tests` function that is required by `test` function to compute statistics as commanded.

```r
tests <- function(d1,d2) {
  mw <- wilcox.test(d1,d2,paired=F)
  wt <- wilcox.test(d1,d2,paired=T)
  ks <- ks.test(d1,d2)
  t <- t.test(d1,d2)
  t.paired <- t.test(d1,d2,paired=T)
  c(mw=mw$statistic,
    wt=wt$statistic,
    ks=ks$statistic,
    unpaired=t$statistic,
    paired=t.paired$statistic)
}
```

The tests function calculates all required nonparametric and parametric test statistics including Mann-Whitney rank test, Wilcoxon signed rank test, Kolmogorov-Sminorov test, *t*-test and paired *t*-test. To better visualize the test statistics with different θ we define a `plot.tests` function.

```
plot.tests <- function(results,titles) {
  cols <- ncol(results[[1]])
  nr <- length(results)
  par(mfrow=c(ceiling(cols/nr),nr))
  for(i in 1:cols) {
    for(j in 1:nr) {
      plot(x=rownames(results[[j]]),
           y=results[[j]][,i],
           type='l',
           main=paste(colnames(results[[j]])[i],titles[j]),
           xlab="theta",
           ylab="pi(theta)",
           cex.main=1.5,
           cex.lab=1.5,
           cex.axis=1.2)
    }
  }
}
```

Now we only need to run `test` function to do parametrized simulations in a convenient way.

First we simulate iid normal data and MA(1) time series both of 100 observations and repeat the simulation for 1000 times for test size 0.01. The MA(1) process is specified as $x_t = \epsilon_t + \theta\epsilon_{t-1}$ for $\theta = -0.9, \ldots, 0, 0.1, \ldots, 0.9$.

```
testr1 <- test(1000,100,0.01,seq(-0.9,0.9,0.1),tests)
testr1

##       mw.W  wt.V  ks.D unpaired.t paired.t
## -0.9 0.000 0.000 0.016      0.000    0.000
## -0.8 0.000 0.000 0.023      0.000    0.000
## -0.7 0.000 0.000 0.016      0.000    0.000
## -0.6 0.000 0.000 0.013      0.000    0.000
## -0.5 0.000 0.000 0.016      0.000    0.000
## -0.4 0.000 0.000 0.007      0.000    0.000
## -0.3 0.000 0.000 0.007      0.000    0.000
## -0.2 0.001 0.000 0.007      0.000    0.000
## -0.1 0.004 0.004 0.005      0.004    0.004
## 0    0.008 0.011 0.006      0.009    0.010
## 0.1  0.020 0.017 0.011      0.020    0.018
## 0.2  0.025 0.024 0.017      0.024    0.025
## 0.3  0.030 0.025 0.020      0.036    0.033
## 0.4  0.042 0.046 0.020      0.050    0.045
## 0.5  0.060 0.056 0.017      0.062    0.058
## 0.6  0.053 0.054 0.022      0.059    0.054
## 0.7  0.050 0.045 0.026      0.054    0.051
## 0.8  0.059 0.052 0.024      0.060    0.053
## 0.9  0.054 0.053 0.031      0.061    0.056
```

From the output above, we clearly see that for $\theta < 0$ the Mann-Whitney rank test, Wilcoxon signed rank test, the unpaired t-test, and the paired t-test all under-reject $\mathcal{H}_0$ whereas for $\theta > 0$ these tests all over-reject $\mathcal{H}_0$. By contrast, Kolmogorov-Sminorov test over-rejects $\mathcal{H}_0$ either for $\theta < 0$ or $\theta > 0$.

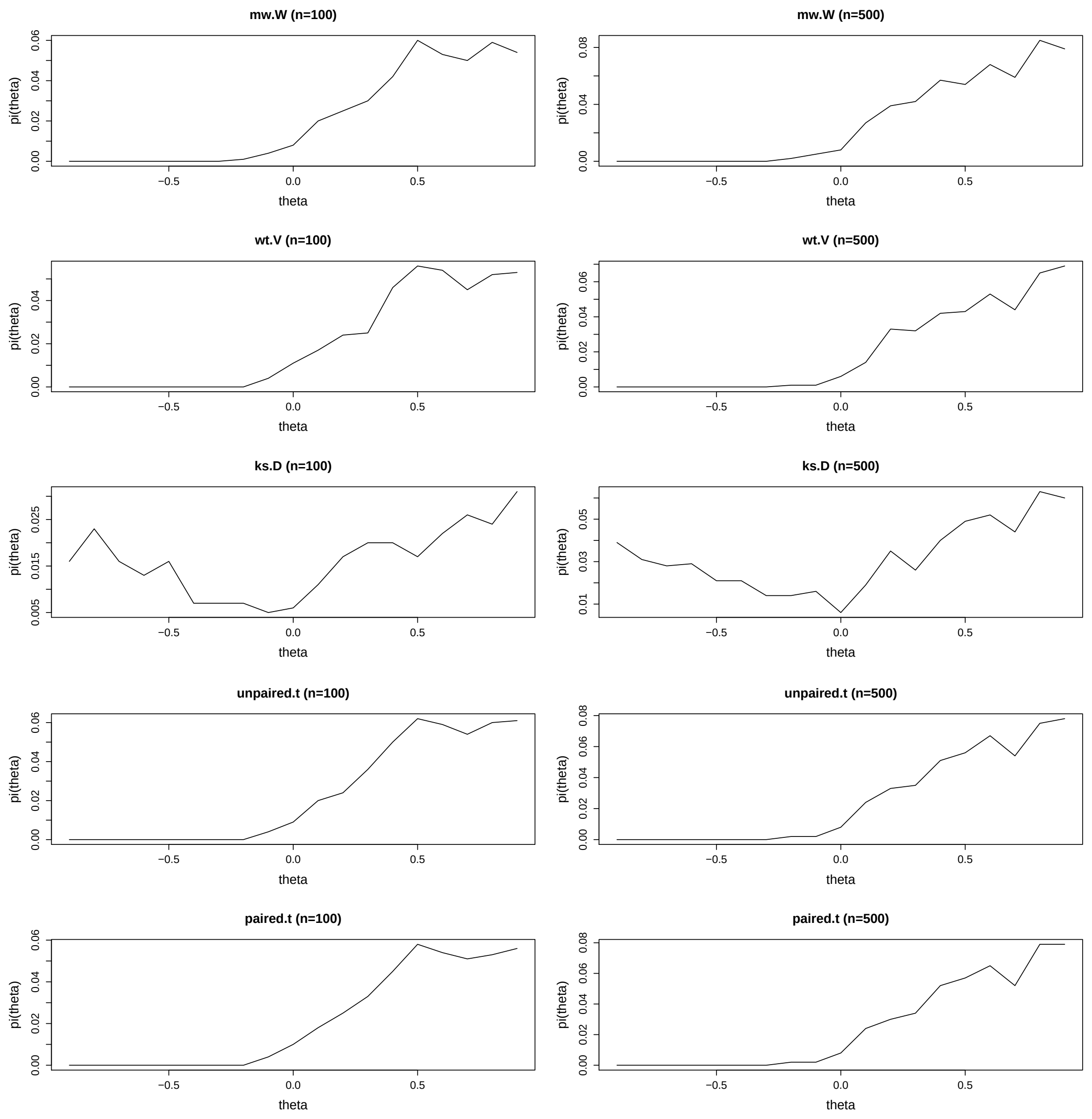
To address whether the phenomena we observe is the previous tests still persist, we conduct another test with 500 observations in each sample. To implement it, we simulate iid normal data and MA(1) time series both of 500 observations and repeat the simulation for 1000 times for test size 0.01. The MA(1) process is specified as $x_t = \epsilon_t + \theta\epsilon_{t-1}$ for $\theta = -0.9, \ldots, 0, 0.1, \ldots, 0.9$.

```r
testr2 <- test(1000,500,0.01,seq(-0.9,0.9,0.1),tests)
testr2

##       mw.W  wt.V  ks.D unpaired.t paired.t
## -0.9 0.000 0.000 0.039      0.000    0.000
## -0.8 0.000 0.000 0.031      0.000    0.000
## -0.7 0.000 0.000 0.028      0.000    0.000
## -0.6 0.000 0.000 0.029      0.000    0.000
## -0.5 0.000 0.000 0.021      0.000    0.000
## -0.4 0.000 0.000 0.021      0.000    0.000
## -0.3 0.000 0.000 0.014      0.000    0.000
## -0.2 0.002 0.001 0.014      0.002    0.002
## -0.1 0.005 0.001 0.016      0.002    0.002
## 0    0.008 0.006 0.006      0.008    0.008
## 0.1  0.027 0.014 0.019      0.024    0.024
## 0.2  0.039 0.033 0.035      0.033    0.030
## 0.3  0.042 0.032 0.026      0.035    0.034
## 0.4  0.057 0.042 0.040      0.051    0.052
## 0.5  0.054 0.043 0.049      0.056    0.057
## 0.6  0.068 0.053 0.052      0.067    0.065
## 0.7  0.059 0.044 0.044      0.054    0.052
## 0.8  0.085 0.065 0.063      0.075    0.079
## 0.9  0.079 0.069 0.060      0.078    0.079
```

From the output data above, we see that the phenomena previously presented still persist and somewhat become more obvious as the extent of over-rejection is even higher with larger samples. To make a contrast between these two cases, we then plot the probability of rejection under $\mathcal{H}_0$ for each θ.

```r
plot.tests(list(testr1,testr2),c("(n=100)","(n=500)"))
```



As we can see, $\theta = 0$ is a benchmark where the MA process degenerates to iid case, resulting in the proper probability of correct rejection of $\mathcal{H}_0$ while for other θ the pattern of under-rejection and over-rejection still holds.

To address the issue where data suffers from a higher order of serial correlation, we also simulate MA(2) time series both of 500 observations and repeat the simulation for 1000 times for test size 0.01. The MA(2) process is specified as $x_t = \epsilon_t + \theta\epsilon_{t-1} + \theta\epsilon_{t-2}$ for $\theta = -0.9, \ldots, 0, 0.1, \ldots, 0.9$.

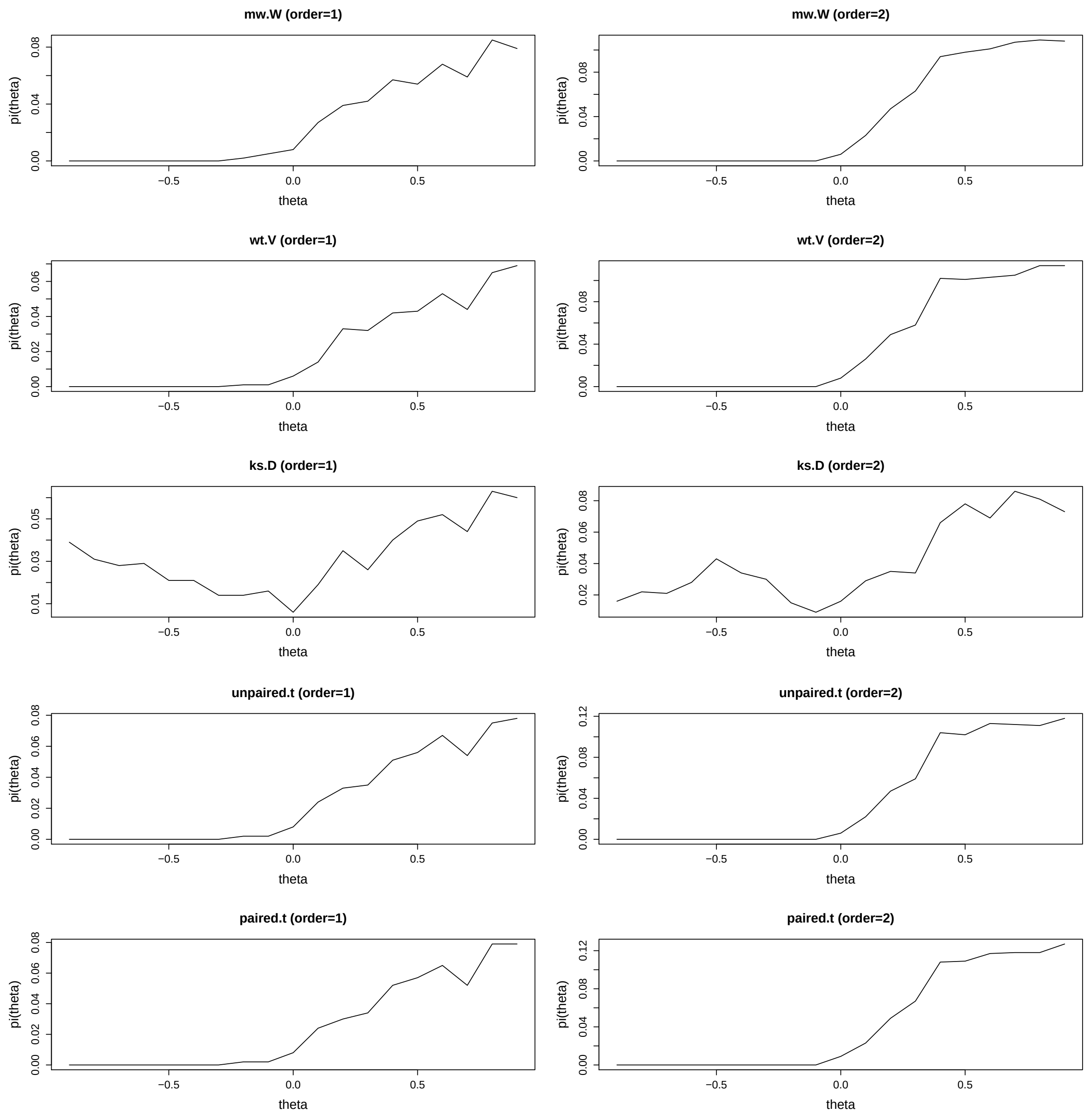
```
testr3 <- test(1000,500,0.01,seq(-0.9,0.9,0.1),tests,order=2)
testr3

##         mw.W  wt.V  ks.D unpaired.t paired.t
## -0.9 0.000 0.000 0.016      0.000    0.000
## -0.8 0.000 0.000 0.022      0.000    0.000
## -0.7 0.000 0.000 0.021      0.000    0.000
## -0.6 0.000 0.000 0.028      0.000    0.000
## -0.5 0.000 0.000 0.043      0.000    0.000
## -0.4 0.000 0.000 0.034      0.000    0.000
## -0.3 0.000 0.000 0.030      0.000    0.000
## -0.2 0.000 0.000 0.015      0.000    0.000
## -0.1 0.000 0.000 0.009      0.000    0.000
## 0    0.006 0.008 0.016      0.006    0.009
## 0.1  0.023 0.026 0.029      0.022    0.023
## 0.2  0.047 0.049 0.035      0.047    0.049
## 0.3  0.063 0.058 0.034      0.059    0.067
## 0.4  0.094 0.102 0.066      0.104    0.108
## 0.5  0.098 0.101 0.078      0.102    0.109
## 0.6  0.101 0.103 0.069      0.113    0.117
## 0.7  0.107 0.105 0.086      0.112    0.118
## 0.8  0.109 0.114 0.081      0.111    0.118
## 0.9  0.108 0.114 0.073      0.118    0.127
```

From the output data above, we see the same pattern of under-rejection and over-rejection each test presents as stated in the previous cases. However, with higher order of autocorrelation, captured by MA(2) process, the magnitude of over-rejection enlarges for each test. This can be interpreted as data with higher order of positive autocorrelation will be uniformly over-rejected by the tests we examined.

To illustrate this point, we again make a contrast of the test performance of higher order MA process, thus we plot the graphs as following:

```r
plot.tests(list(testr2,testr3),c("(order=1)","(order=2)"))
```



As we can see, the magnitude for each group of tests get greater when the order of autocorrelation gets larger as we observe in the table.