Bootstrapping Simulation

Yiyang Hu
Instructor: Yinxiao Huang

December 31, 2011

1 Introduction

This paper investigates the bootstrap method by using the statistics software *R*. Generally speaking, bootstrapping is a kind of re-sampling method. When we sample from an approximating distribution, we will get an estimator from the estimating properties each time. The most common way for the approximating distribution is the empirical distribution. One of the significant advantages of bootstrapping is that it is very simple, without too much complicated parameters and easy to control the stability. Moreover, bootstrapping is especially helpful in the following cases:

1. When we need to perform power calculations, but only inadequate pilot sample is available.

2. When the underlying distribution is known but the sample size is so small that cannot fully represent the forthright statistical inference.

3. When the distribution of the given statistics is very abstract or unknown.

2 The Simulation of X

Initially we use a normal distribution simulation for an autoregressive model with order 1,

$$X_t = \phi X_{t-1} + W_t, \tag{1}$$

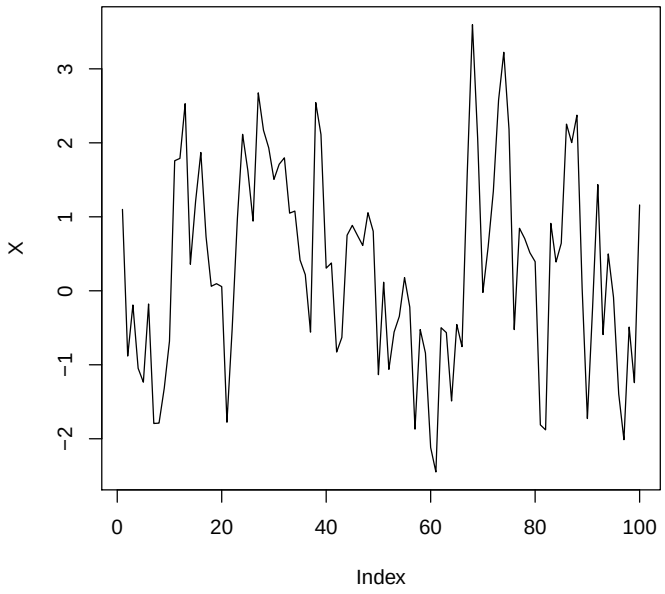

Figure 1: Sequence X_t

where the W_t are individually independent white noise. We start with

$$X_1 \sim N(0,1)$$
$$\phi = 0.57$$

and let the sequence $X_1, X_2, \ldots$ be defined by equation (1) where the W_t are randomly chosen real numbers with the command $rnorm(1)$. This produces a sequence shown in Figure 1, where 100 sample points were used.

3 Autoregession

Next, we try to estimate the coefficient ϕ from the sequence $X_1, X_2, \ldots$ alone. This is accomplished via the R function ar, and produces an estimated coefficient $\hat{\phi}$. We repeat this process of generating a sequence $X_1, X_2, \ldots$ from (1) and estimating the coefficient ϕ to get a sequence of estimates $\hat{\phi}_1, \hat{\phi}_2, \ldots$.

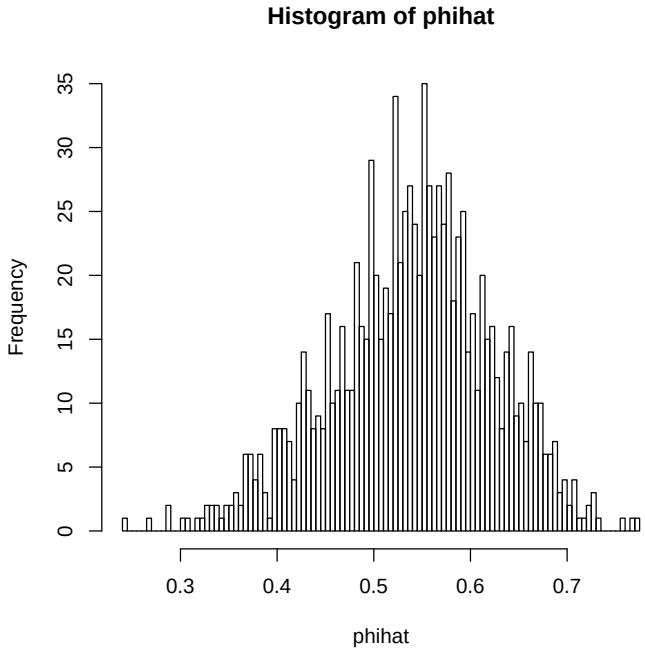

Figure 2: Distribution of $\hat{\phi}_n$ for normally distribution white noise

When we repeat this process for a large number of repetitions (eg, $N = 1000$), the sample distribution of $\hat{\phi}_1, \hat{\phi}_2, \ldots$ should be close to normal. From the graph of the distribution shown in Figure 2, we can see that it is very close to normal, with a mean of 0.5395119 and variance of 0.007056072.

We now repeat this process for student's t-distribution white noise (generated with the command *rt(1,4,0)*) with $\phi = 0.9$. In order to observe the normal discribution in the sequence $\hat{\phi}_1, \hat{\phi}_2, \ldots$, the sample size had to be increased from 100 to 1000. As we can see from Figure 3, the distribution is nearly symmetric about its mean of 0.8955467. The variation was 0.0002029488. The Central Limit Theorem tells us that the mean and sum of N independent and identically distributed random variables will be closely to normal when N is large.

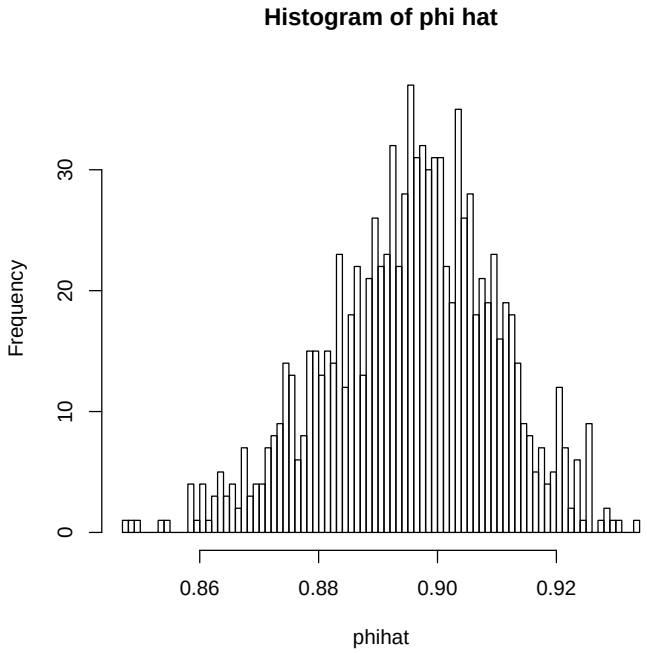

Figure 3: Distribution of $\hat{\phi}_n$ for student's t-distribution white noise

4 Bootstrapping

The last step is to analyze a dataset. We are given a file containing 100 data points which is believed to follow an $AR(1)$ model, but of which the coefficient ϕ is unknown. We would like to use the bootstrap method to simulate the distribution of $\hat{\phi}$. First we use *mydata=scan("bootstrap.txt")* to read 100 items from the given file. Then using *ar(mydata,order=1)* we find the estimated coefficient $\hat{\phi} = 0.9572$. Because now we already know the coefficient and each term $X_1, \ldots, X_{100}$, we can use these to find the residuals $e_1, e_2 \ldots\ldots e_{100}$ from our order 1 autoregression model. Next, we randomly sample with replacement from the set we already have and obtain the sample innovation $e_1^*, e_2^*, \ldots, e_{100}^*$. Then we generate a bootstrapped data set sequentially by

$$X_1^* = e_1^*,$$
$$X_t^* = \hat{\phi} X_{t-1}^* + e_t^*.$$

The R code used to accomplish this is:

```
mydata=scan("bootstrap.txt")
fit=ar(mydata,order=1)
phi=fit$ar
resid=fit$resid[-1]
x.star=array(dim=100)
phi.star=array(dim=999)
for(i in 1:999){
    resid.star=sample(resid,replace=TRUE,size=100)
    x.star[1]=resid.star[1]
    for(t in 1:99){
        x.star[t+1]=phi*x.star[t]+resid.star[t+1]
    }
    phi.star[i]=ar(x.star,order=1)$ar
}
hist(phi.star)
```

The whole idea of bootstrapping is that after we get the sequence $e_1^*, e_2^*, \ldots,$ we take the data back and obtain the sequence $X_1^*, X_2^*, \ldots$. As a result of repeating this process, we can obtain a sequence $\hat{\phi}_1^*, \hat{\phi}_2^*, \ldots$ of estimated coefficients and compare it with a normal distribution. From the histogram in

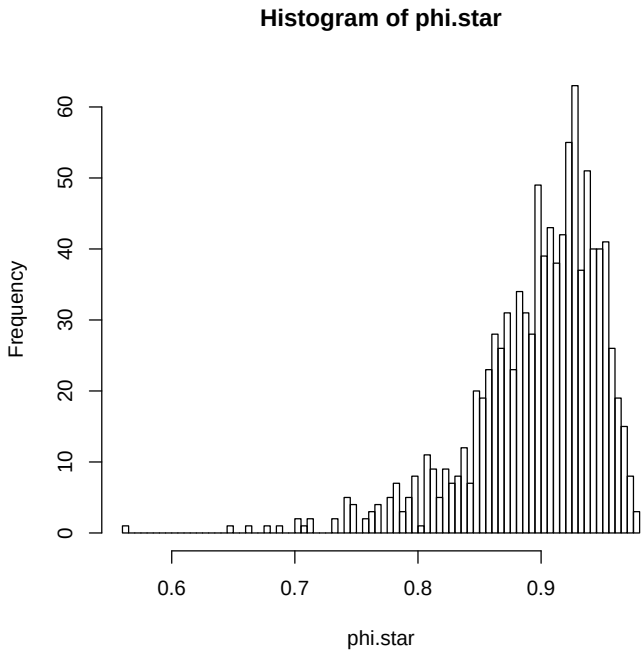

Figure 4: Distribution of $\hat{\phi}_n^*$ from bootstrapping

Figure 4 we can clearly see that the graph is not symmetric about its mean. This result might be due to the following reasons:

1. Using the bootstrapping method oversimplified the distribution which does not provide a comprehensive finite-sample guarantees. The error from the first $e_1^*, e_2^*, \ldots$ sequence may lead to another in the succeeding sequence. After 100 times, the accumulative errors make it deviate from a normal distribution.

2. The underlying true model is not driven by normally distributed white noise, which is common in a real life data set.