PARAMETER ESTIMATION BIAS WITH DIFFERENT SAMPLE SIZE

RAN ZHAO

ABSTRACT. This paper investigates the bias of parameter estimation, given different sets of parameter true value and sample size. We found that non-zero α improves the unbiasedness of parameter estimation in AR(1) model, but the level of σ does not contribution to unbiasedness. Another improvement of parameter estimation raises when the AR(1) becomes stationary, say changing the β from 1 to 0.95. Increasing the sample size makes the distribution of β's t-statistic closer to be normally distributed.

1. MODEL SPECIFICATION

Consider the AR(1) model with dynamic

$$p_t = \alpha + \beta p_{t-1} + \epsilon_t, \quad \epsilon_t \sim N(0, \sigma^2) \tag{1}$$

where the error terms, ϵ_t, are i.i.d.

Given the true value of the parameters and the initial value of the log stock price, the full log stock price process, $\{p_t\}_{t=0}^{T}$, can be simulated by the following algorithm,

Algorithm 1 Estimate Bias

procedure SIMULATION
set sample size N and simulation length T
loop i from 1 to N:
    initialize $p(0,i)$
    loop t from 1 to T:
        draw random normal variable $u \sim N(0, \sigma^2)$
        $p(t,i) \leftarrow \alpha + \beta p(t,i) + norm$
    next t
    estimate parameter bias $Bias_{(\theta,i)}[\hat{\theta}] = \hat{\theta}_i - \theta$, for $\theta = \{\alpha, \beta, \sigma\}$
next i
$Bias_{(\theta)}[\hat{\theta}] = \sum_i Bias_{(\theta,i)}/N$
end procedure

2. ESTIMATION ANALYSIS

2.1. $\alpha = 0, \beta = 1, \sigma = 0.2$. First, we select sample size $T = 50$ with 10,000 paths to estimate the bias of parameters estimation. Given the parameter set as $\theta = \{\alpha = 0, \beta = 1, \sigma = 0.2\}$, the biases are shown in Table 1. The initial value of the log stock price is set to $\log(100) \approx 4.605$. The bias for α estimation is 0.4739, the bias for β estimation is -0.1029, and the bias for σ is -0.0049.

When bumping the true value of α, the bias of all the parameters reduce significantly. That is, a non-zero trending benefits the unbiasedness of the parameter estimation. However, bumping σ up and down does not improve the bias of parameter estimation.

To test $\beta = 1$, the t-statistic is calculated as $t = (\hat{\beta} - 1)/se(\hat{\beta})$. With the sample size of 50, the 1% and 5% t-statistics are -3.5265 and -2.9083, respectively.

Table 1 The bias on parameter estimation and t-statistic on β estimation are provided, given different sets of true parameter values. The model specification is as Equation 1. Simulation procedure follows Algorithm 1.

Run Tag	α	β	σ	T	$Bias(\hat{\alpha})$	$Bias(\hat{\beta})$	$Bias(\hat{\sigma})$	$t_\beta(1\%)$	$t_\beta(5\%)$
1	0	1	0.2	50	0.4739	-0.1029	-0.0049	-3.5265	-2.9083
2	-0.2	1	0.2	50	-0.0049	-0.0026	-0.0012	-2.6658	-1.9607
3	0.2	1	0.2	50	0.0266	-0.0024	-0.0015	-2.5619	-1.8682
4	0	1	0.1	50	0.4754	-0.1033	-0.0025	-3.5554	-2.9260
5	0	1	0.3	50	0.4860	-0.1051	-0.0078	-3.6126	-2.9336
6	0	1	0.2	600	0.0413	-0.0090	-0.0004	-3.5049	-2.8769
7	0	0.95	0.2	50	0.0183	-0.0126	-0.0011	-4.7079	-3.9881

When sample size increases to 600, the 1% and 5% t-statistics are -2.6658 and -1.9607, respectively. With the increase of sample size, the t-statistics are closer to normal distributed statistics, with narrower tail-distribution. When sample size is smaller, the fat-fail is more obvious.

2.2. **$\alpha = 0, \beta = 0.95, \sigma = 0.2$.** Changing true value of β to be 0.95 instead of 1, or α estimation is 0.0183, the bias for β estimation is -0.0126, and the bias for σ is -0.0011. The biases when $\beta = 0.95$ are consistently lower than the biases when $\beta = 1$, showing an improving on unbiasedness when the model is stationary.

Code Appendix

```
setwd('C:\\Users\\ranzhao\\Documents\\Empirical-Asset-Pricing\\Assignment 3')
setwd('D:\\PhD FE\\Empirical-Asset-Pricing\\Assignment 3')

ar.parameter.inference <- function(n=50,alpha=0,beta=1,sigma=0.2,p0=log(100),N=10000){
  # simulate the log stock price process
  alpha.bias = rep(0,N)
  beta.bias = rep(0,N)
  sigma.bias = rep(0,N)
  t.stats = rep(0,N)

  # simulation loop
  for (j in 1:N){
    p.series = rep(0, n)
    # time loop
    for (i in 1:n){
      if (i == 1){
        p.series[i] = alpha + beta * p0 + rnorm(1, 0, sigma)
      }
      else{
        p.series[i] = alpha + beta * p.series[i-1] + rnorm(1, 0, sigma)
      }
    }
    # fit the parameters using ols
    fitted.model = lm(p.series[2:n]~p.series[1:(n-1)])
    summ = summary(fitted.model)
    # fitted parameters
    alpha.fit = as.numeric(fitted.model$coefficients[1])
    beta.fit = as.numeric(fitted.model$coefficients[2])
    sigma.fit = sqrt(var(fitted.model$residuals)*(n-1)/(n-2))
    # bias parameters
    alpha.bias[j] = alpha.fit - alpha
    beta.bias[j] = beta.fit - beta
    sigma.bias[j] = sigma.fit - sigma
    t.stats[j] = as.numeric((summ$coefficients[2]-1)/summ$coefficients[4])
  }

  #output
  p.out = c()
  p.out$bias = cbind(alpha.bias, beta.bias, sigma.bias)
  p.out$t.stats = t.stats
  return(p.out)
}

# question (a) i, iii
T50alpha0beta1sigma0p2 = ar.parameter.inference(n=50,alpha=0,beta=1,sigma=0.2,p0=log(100),N=10000)
c(mean(T50alpha0beta1sigma0p2$bias[,1]), mean(T50alpha0beta1sigma0p2$bias[,2]), mean(T50alpha0beta1sigma0p2$bias[,3]))
quantile(T50alpha0beta1sigma0p2$t.stats,c(0.01,0.05))

# question (a) ii
alpha.bump.results1 = ar.parameter.inference(n=50,alpha=0.2,beta=1,sigma=0.2,p0=log(100),N=10000)
c(mean(alpha.bump.results1$bias[,1]), mean(alpha.bump.results1$bias[,2]), mean(alpha.bump.results1$bias[,3]))
quantile(alpha.bump.results1$t.stats,c(0.01,0.05))
```

```r

alpha.bump.results2 = ar.parameter.inference(n=50,alpha=-0.2,beta=1,sigma=0.2,p0=log(100),N=10000)
c(mean(alpha.bump.results2$bias[,1]), mean(alpha.bump.results2$bias[,2]), mean(alpha.bump.results2$bias[,3]))
quantile(alpha.bump.results2$t.stats,c(0.01,0.05))

sigma.bump.results1 = ar.parameter.inference(n=50,alpha=0,beta=1,sigma=0.1,p0=log(100),N=10000)
c(mean(sigma.bump.results1$bias[,1]), mean(sigma.bump.results1$bias[,2]), mean(sigma.bump.results1$bias[,3]))
quantile(sigma.bump.results1$t.stats,c(0.01,0.05))

sigma.bump.results2 = ar.parameter.inference(n=50,alpha=0,beta=1,sigma=0.3,p0=log(100),N=10000)
c(mean(sigma.bump.results2$bias[,1]), mean(sigma.bump.results2$bias[,2]), mean(sigma.bump.results2$bias[,3]))
quantile(sigma.bump.results2$t.stats,c(0.01,0.05))

# question (a) iv
T600alpha0beta1sigma0p2 = ar.parameter.inference(n=600,alpha=0,beta=1,sigma=0.2,p0=log(100),N=10000)
c(mean(T600alpha0beta1sigma0p2$bias[,1]), mean(T600alpha0beta1sigma0p2$bias[,2]), mean(T600alpha0beta1sigma0p2$bias[,3]))
quantile(T600alpha0beta1sigma0p2$t.stats,c(0.01,0.05))

# question (b) i
T50alpha0beta0p95sigma0p2 = ar.parameter.inference(n=50,alpha=0,beta=0.95,sigma=0.2,p0=log(100),N=10000)
c(mean(T50alpha0beta0p95sigma0p2$bias[,1]), mean(T50alpha0beta0p95sigma0p2$bias[,2]), mean(T50alpha0beta0p95sigma0p2$bias[,3]))
quantile(T50alpha0beta0p95sigma0p2$t.stats,c(0.01,0.05))
```

assignment3.R