2022 秋季本科时间序列

第 6 次作业答案

11 月 16 日

1. (a)

$$\boldsymbol{M} = \mathbb{E}\begin{bmatrix}1\\ \boldsymbol{X}_{t-1}\end{bmatrix}\begin{bmatrix}1 & \boldsymbol{X}_{t-1}\end{bmatrix} = \begin{bmatrix}1 & \mathbb{E}\boldsymbol{X}_{t-1}\\ \mathbb{E}\boldsymbol{X}_{t-1} & \mathbb{E}\boldsymbol{X}_{t-1}^2\end{bmatrix}, det(\boldsymbol{M}) = \mathrm{var}\boldsymbol{X}_{t-1}$$

$$\mathrm{var}\boldsymbol{X}_t = \mathrm{var}(\mu + \phi\boldsymbol{X}_{t-1} + \epsilon) = \phi^2\mathrm{var}\boldsymbol{X}_{t-1} + \sigma_\epsilon^2$$

因为 AR(1) 过程 $\boldsymbol{X}_t$ 平稳，故 $\mathrm{var}\boldsymbol{X}_{t-1} = \mathrm{var}\boldsymbol{X}_t = \frac{1}{1-\phi^2} > 0, \boldsymbol{M}$ 满秩

(b) $\gamma(0) = \frac{1}{1-\phi^2}, \gamma(k) = \phi\gamma(k-1) = \phi^k\gamma(0) = \frac{\phi^k}{1-\phi^2}$

$$\boldsymbol{S}_X(\omega) = \sum_{k=-\infty}^{\infty}\gamma(k)e^{-i2\pi\omega k}$$

由于常数项不影响谱密度函数，即求 AR(1) 过程 $\boldsymbol{X}_t = \phi\boldsymbol{X}_{t-1} + \varepsilon_t$ 的谱密度函数，

$\boldsymbol{X}_t - \phi\boldsymbol{X}_{t-1} = \varepsilon_t, A(\mathcal{L})\boldsymbol{X}_t = \varepsilon_t$，则 $\boldsymbol{X}_t = A^{-1}(\mathcal{L})\varepsilon_t, A^{-1}(\mathcal{L}) = \frac{1}{1-\phi\mathcal{L}}$

给定滤波多项式 $C(z) = \frac{1}{1-\phi z}$，按照第五讲滤波序列的谱表示，增益函数 $G(\omega) = \sqrt{C(e^{-i2\pi\omega})C(e^{i2\pi\omega})}$

则 $\boldsymbol{S}_X(\omega) = G^2(\omega)\boldsymbol{S}_\varepsilon(\omega), \boldsymbol{S}_\varepsilon(\omega) = 1$，代入得，

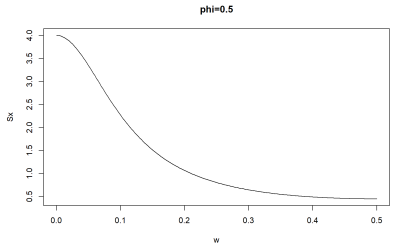
$$\boldsymbol{S}_X(\omega) = \frac{1}{(1-\phi e^{-i2\pi\omega})(1-\phi e^{i2\pi\omega})} = \frac{1}{1-2\phi\cos(2\pi\omega)+\phi^2}$$

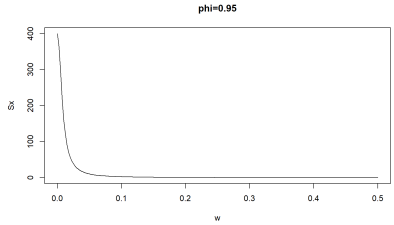
把 $\phi = 0.5, 0.95$ 代入得，$\boldsymbol{S}_X(\omega)|(\phi = 0.5) = \frac{1}{1.25-\cos(2\pi\omega)}, \boldsymbol{S}_X(\omega)|(\phi = 0.95) = \frac{1}{1.9025-1.9\cos(2\pi\omega)}$

代码如下：

```
x <- seq(0,0.5,0.0025)
y <- vector()
z <- vector()
length(x)
for(i in 1:length(x)) {
```

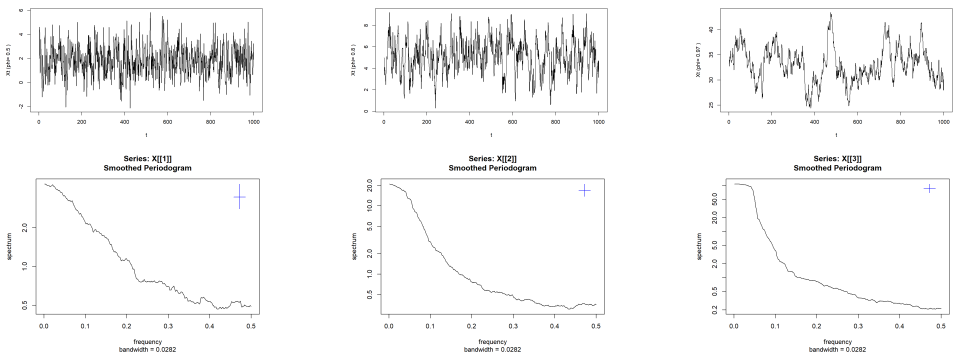
```
  y[i] <- 1/(1+0.95^2-2*0.95*cos(2*pi*x[i]))
  z[i] <- 1/(1+0.5^2-2*0.5*cos(2*pi*x[i]))
}
plot(x,z,type="l",main="phi=0.5",xlab="w",ylab="Sx")
plot(x,y,type="l",main="phi=0.95",xlab="w",ylab="Sx")
```





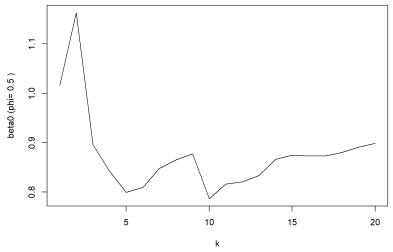
(c) 代码如下:

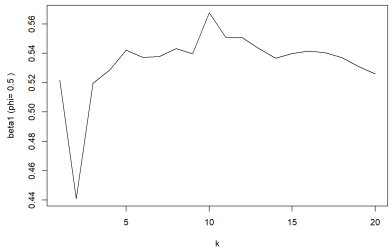
```
mu <- 1
phi <- c(0.5, 0.8,0.97)
X <- tibble()
for(i in 1:3){
 X[1, i] = mu/(1 - phi[i])
 for(j in 1:1000){
  X[j+1, i] = mu + phi[i]*X[j, i] + rnorm(1,0,1)
 }
 plot(X[[i]], type = "l", xlab = "t",
 ylab = paste('Xt (phi=', phi[i],')'))
}
spec.pgram(X[[1]],kernel("daniell", c(5,7)), taper = 0.2)
spec.pgram(X[[2]],kernel("daniell", c(5,7)), taper = 0.1)
spec.pgram(X[[3]],kernel("daniell", c(5,7)), taper = 0.2)
```

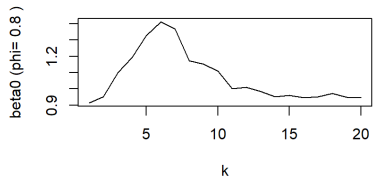


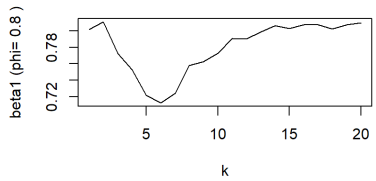
(d) 代码如下：

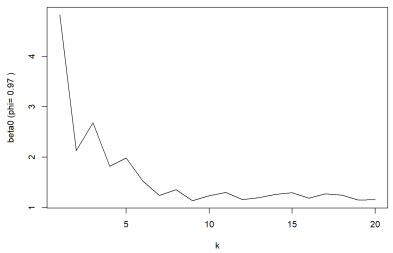
```
beta0 <- tibble()
beta1 <- tibble()
for(i in 1:3){
  for(k in 1:20){
    y <- unlist(X[2:(50*k+1), i])
    x <- unlist(X[1:(50*k), i])
    ols <- coefficients(lm(y ~ x))
    beta0[k, i] = ols[1]
    beta1[k, i] = ols[2]
  }
  plot(beta0[[i]], type = "l", xlab = "k",
  ylab = paste('beta0 (phi=', phi[i],')'))
  plot(beta1[[i]], type = "l", xlab = "k",
  ylab = paste('beta1 (phi=', phi[i],')'))
}
```

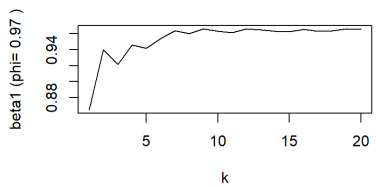










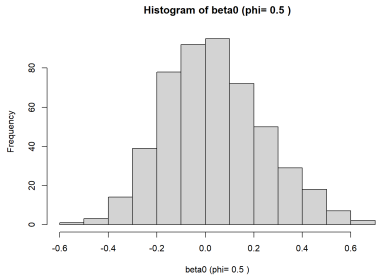


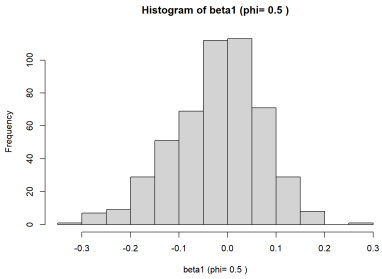
随着 ϕ 的增大，OLS 估计随样本量在 50k 增加时的收敛性逐渐增强。

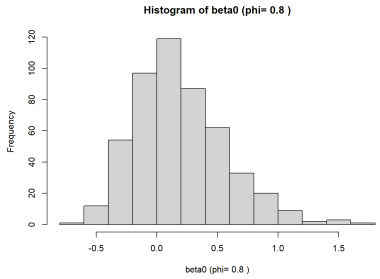
(e) 代码如下：

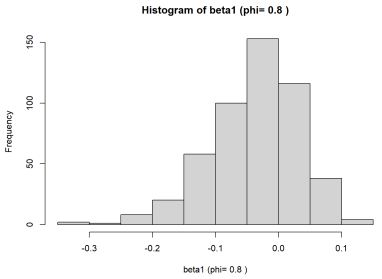
```
# phi = 0.5
X1 <- tibble()
for(i in 1:500){
 X1[1, i] = mu/(1 - phi[1])
 for(j in 1:100){
  X1[j+1, i] = mu + phi[1]*X1[j, i] + rnorm(1,0,1)
 }
}
# phi = 0.8
X2 <- tibble()
for(i in 1:500){
 X2[1, i] = mu/(1 - phi[2])
 for(j in 1:100){
  X2[j+1, i] = mu + phi[2]*X2[j, i] + rnorm(1,0,1)
 }
}
# phi = 0.97
X3 <- tibble()
for(i in 1:500){
 X3[1, i] = mu/(1 - phi[3])
 for(j in 1:100){
  X3[j+1, i] = mu + phi[3]*X3[j, i] + rnorm(1,0,1)
 }
}
X <- list(X1, X2, X3)

beta0 <- tibble()
beta1 <- tibble()
b0 <- tibble()
b1 <- tibble()
sd_beta0 <- vector("double",2)
sd_beta1 <- vector("double",2)
for(i in 1:3){
 for(j in 1:500){
```

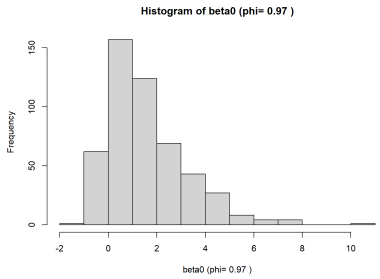
```r
  y <- unlist(X[[i]][2:101, j])
  x <- unlist(X[[i]][1:100, j])
  ols <- coefficients(lm(y ~ x))
  beta0[j, i] <- ols[1]
  beta1[j, i] <- ols[2]
 }
 hist(beta0[[i]]-mu, xlab = paste('beta0 (phi=',
 phi[i],')'), main = paste('Histogram of beta0 (phi=',
 phi[i],')'))
 hist(beta1[[i]]-phi[i], xlab = paste('beta1 (phi=',
 phi[i],')'), main = paste('Histogram of beta1 (phi=',
 phi[i],')'))

 sd_beta0[i] <- sd(beta0[[i]])
 sd_beta1[i] <- sd(beta1[[i]])
}
sd_beta0
## [1] 0.2062898 0.3737770 1.6189055
sd_beta1
## [1] 0.09120952 0.06972700 0.04852232
for(i in 1:3){
 print(ggplot()+geom_histogram(data=NULL,mapping=aes(x=
 beta0[[i]],y=..density..))+xlab(paste('beta0 (phi=',
 phi[i],')'))+geom_function(fun = dnorm,color="red",args=
 list(mean=mu,sd=sd_beta0[i])))
 print(ggplot()+geom_histogram(data=NULL,mapping=aes(x=
 beta1[[i]],y=..density..))+xlab(paste('beta1 (phi=',
 phi[i],')'))+geom_function(fun = dnorm,color="red",args=
 list(mean=phi[i],sd=sd_beta1[i])))
}
```

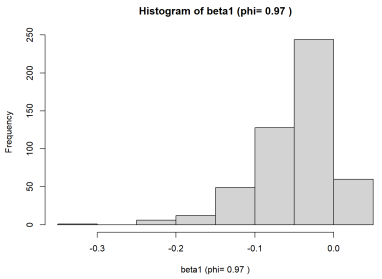

Histogram of beta0 (phi= 0.5)
Frequency
beta0 (phi= 0.5)

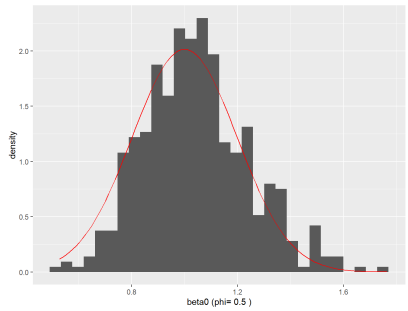

Histogram of beta1 (phi= 0.5)
Frequency
beta1 (phi= 0.5)

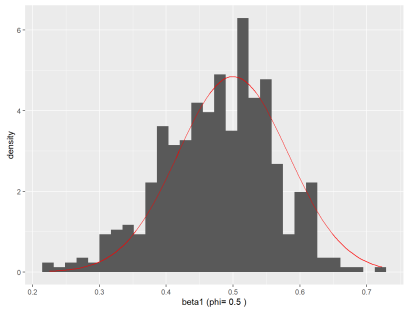

Histogram of beta0 (phi= 0.8)
Frequency
beta0 (phi= 0.8)

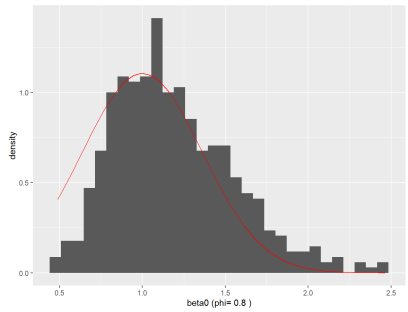

Histogram of beta1 (phi= 0.8)
Frequency
beta1 (phi= 0.8)

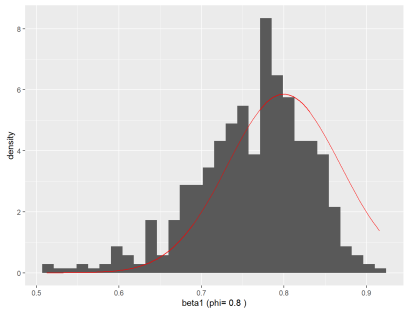

Histogram of beta0 (phi= 0.97)
Frequency
beta0 (phi= 0.97)

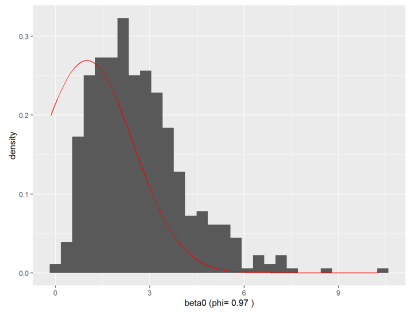

Histogram of beta1 (phi= 0.97)
Frequency
beta1 (phi= 0.97)

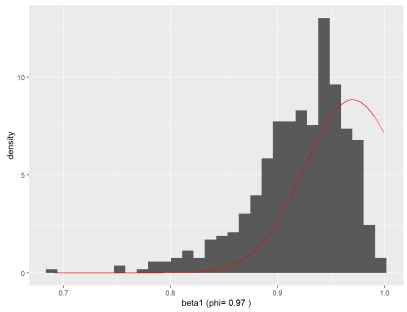

density
beta0 (phi= 0.5)


density
beta1 (phi= 0.5)


density
beta0 (phi= 0.8)


density
beta1 (phi= 0.8)


density
beta0 (phi= 0.97)


density
beta1 (phi= 0.97)

当 ϕ 趋近于 1 时，样本分布逐渐偏离正态分布。

(f) 代码如下：

```
#理论渐近标准误
mu <- 1
sigma <- 1
T_size <- 100
AR1_CM <- function(phi, mu, sigma){
 EX <- mu / (1 - phi)
 EX2 <- (EX)^2 + 1 / (1 - phi^2)
 M <- matrix(c(1, EX, EX, EX2),
 ncol = 2)
 CM <- solve(M) * (sigma ^2) * (1/T_size)
 return(CM)
}

AR1_CM(phi[1], mu = 1, sigma = 1)
#          [,1]     [,2]
#[1,]  0.040  -0.0150
#[2,] -0.015   0.0075
AR1_CM(phi[2], mu = 1, sigma = 1)
#        [,1]     [,2]
#[1,]  0.100  -0.0180
#[2,] -0.018   0.0036
AR1_CM(phi[3], mu = 1, sigma = 1)
#           [,1]       [,2]
#[1,]  0.6666667  -0.019700
#[2,] -0.0197000   0.000591
sd_theory <- vector(mode = "list", length = 3)
for(i in 1:3){
 sd_theory[[i]] <- c(sd_beta0 = sqrt(AR1_CM(phi[i], mu = 1,
  sigma = 1)[1,1]), sd_beta1 =sqrt(AR1_CM(phi[i],
  mu = 1, sigma = 1)[2,2]))
}
names(sd_theory) <- c("phi=0.5","phi=0.8","phi=0.97")
sd_theory
```

```
34
35 # $phi=0.5
36 # sd_bata0     sd_beta1
37 # 0.20000000  0.08660254
38 #
39 # $phi=0.8
40 # sd_bata0    sd_beta1
41 # 0.3162278  0.0600000
42 #
43 # $phi=0.97
44 # sd_bata0     sd_beta1
45 # 0.81649658  0.02431049
46
47 #样本渐近标准误
48 x <- vector("double",101)
49 for (i in 1:3) {
50  x[1] <- mu/(1 - phi[i])
51  for (j in 1:100) {
52   x[j+1] <- mu + phi[i]*x[j] + rnorm(1,0,1)
53  }
54  X <- cbind(rep(1,100), x[2:101])
55  M <- t(X) %*% X / 100    # 求M矩阵
56  CM<- 1/100*1*solve(M)
57  print(CM)
58  sd_sample<-c(sd_beta0=sqrt(CM[1,1]),sd_beta1=sqrt(CM[2,2]))
59  print(sd_sample)
60 }
61 #             [,1]         [,2]
62 # [1,]   0.03842015  -0.01443181
63 # [2,]  -0.01443181   0.00732850
64 # sd_beta0    sd_beta1
65 # 0.19601058  0.08560666
66 #             [,1]          [,2]
67 # [1,]   0.12753886  -0.021741240
68 # [2,]  -0.02174124   0.004021492
```

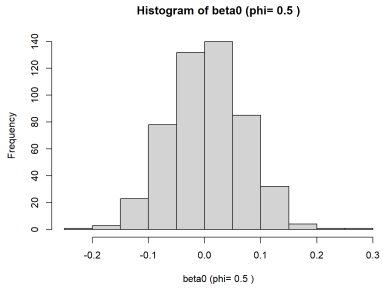
```
# sd_beta0    sd_beta1
# 0.35712583  0.06341523
#             [,1]          [,2]
# [1,]   0.41697764  -0.012744725
# [2,]  -0.01274473   0.000399108
# sd_beta0    sd_beta1
# 0.64573806  0.01997769
```

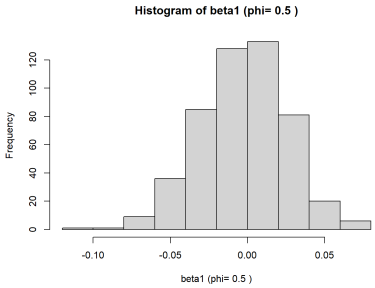
渐进标准误接近样本标准差的值。

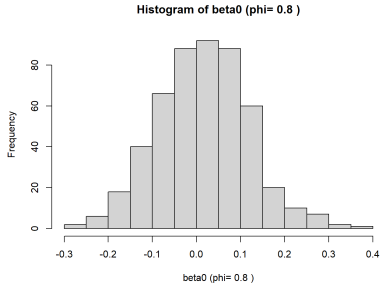
(g) 代码如下：

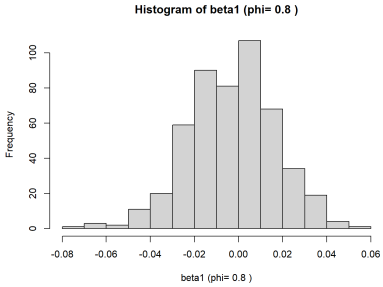
```
# phi = 0.5
X1 <- tibble()
for(i in 1:500){
 X1[1, i] = mu/(1 - phi[1])
 for(j in 1:900){
  X1[j+1, i] = mu + phi[1]*X1[j, i] + rnorm(1,0,1)
 }
}
# phi = 0.8
X2 <- tibble()
for(i in 1:500){
 X2[1, i] = mu/(1 - phi[2])
 for(j in 1:900){
  X2[j+1, i] = mu + phi[2]*X2[j, i] + rnorm(1,0,1)
 }
}
# phi = 0.97
X3 <- tibble()
for(i in 1:500){
 X3[1, i] = mu/(1 - phi[3])
 for(j in 1:900){
  X3[j+1, i] = mu + phi[3]*X3[j, i] + rnorm(1,0,1)
 }
}
X <- list(X1, X2, X3)
beta0 <- tibble()
```

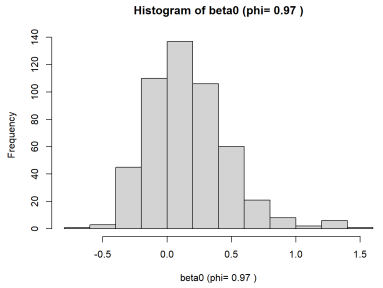
```r
beta1 <- tibble()
b0 <- tibble()
b1 <- tibble()
sd_beta0 <- vector("double",2)
sd_beta1 <- vector("double",2)
for(i in 1:3){
 for(j in 1:500){
  y <- unlist(X[[i]][2:901, j])
  x <- unlist(X[[i]][1:900, j])
  ols <- coefficients(lm(y ~ x))
  beta0[j, i] <- ols[1]
  beta1[j, i] <- ols[2]
 }
 hist(beta0[[i]]-mu, xlab = paste('beta0 (phi=', phi[i],')'),
  main = paste('Histogram of beta0 (phi=', phi[i],')'))
 hist(beta1[[i]]-phi[i], xlab = paste('beta1 (phi=',
  phi[i],')'), main = paste('Histogram of beta1 (phi=',
   phi[i],')'))
 sd_beta0[i] <- sd(beta0[[i]])
 sd_beta1[i] <- sd(beta1[[i]])
}
sd_beta0
## [1] 0.06672679 0.10226909 0.31583850
sd_beta1
## [1] 0.028816909 0.019713237 0.009370077
for(i in 1:3){
 print(ggplot()+geom_histogram(data=NULL,mapping=aes(x=
 beta0[[i]],y=..density..))+xlab(paste('beta0 (phi=',
 phi[i],')'))+geom_function(fun = dnorm,color="red",args=
 list(mean=mu,sd=sd_beta0[i])))
 print(ggplot()+geom_histogram(data=NULL,mapping=aes(x=
 beta1[[i]],y=..density..))+xlab(paste('beta1 (phi=',
 phi[i],')'))+geom_function(fun = dnorm,color="red",args=
 list(mean=phi[i],sd=sd_beta1[i])))
}
```

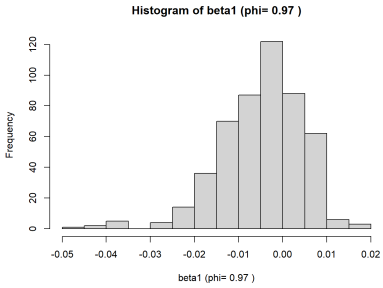

Histogram of beta0 (phi= 0.5)
Frequency
0
20
40
60
80
100
120
140
-0.2
-0.1
0.0
0.1
0.2
0.3
beta0 (phi= 0.5)

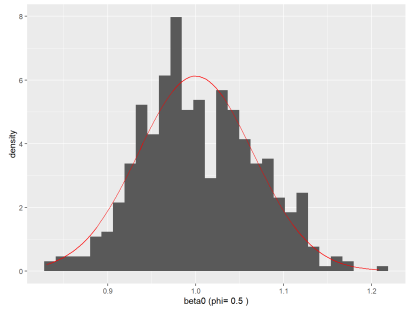

Histogram of beta1 (phi= 0.5)
Frequency
0
20
40
60
80
100
120
-0.10
-0.05
0.00
0.05
beta1 (phi= 0.5)

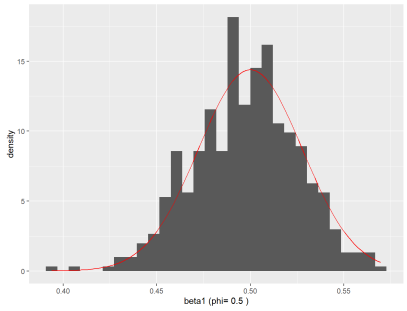

Histogram of beta0 (phi= 0.8)
Frequency
0
20
40
60
80
-0.3
-0.2
-0.1
0.0
0.1
0.2
0.3
0.4
beta0 (phi= 0.8)

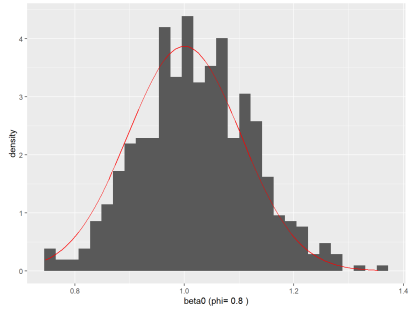

Histogram of beta1 (phi= 0.8)
Frequency
0
20
40
60
80
100
-0.08
-0.06
-0.04
-0.02
0.00
0.02
0.04
0.06
beta1 (phi= 0.8)

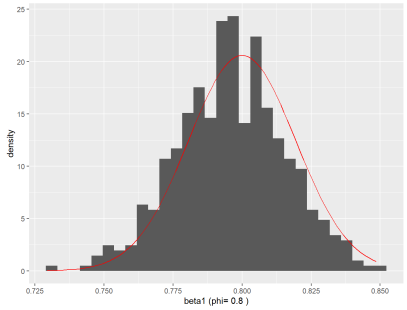

Histogram of beta0 (phi= 0.97)
Frequency
0
20
40
60
80
100
120
140
-0.5
0.0
0.5
1.0
1.5
beta0 (phi= 0.97)

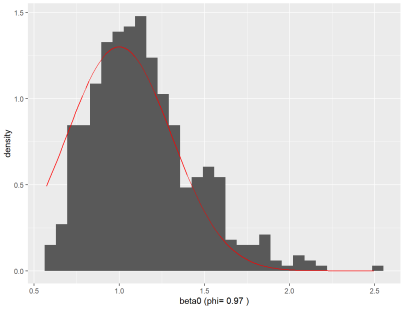

Histogram of beta1 (phi= 0.97)
Frequency
0
20
40
60
80
100
120
-0.05
-0.04
-0.03
-0.02
-0.01
0.00
0.01
0.02
beta1 (phi= 0.97)

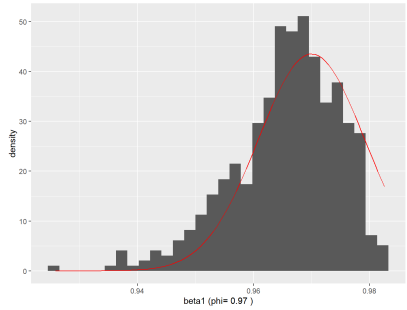

density
0
2
4
6
8
0.9
1.0
1.1
1.2
beta0 (phi= 0.5)


density
0
5
10
15
0.40
0.45
0.50
0.55
beta1 (phi= 0.5)


density
0
1
2
3
4
0.8
1.0
1.2
1.4
beta0 (phi= 0.8)


density
0
5
10
15
20
25
0.725
0.750
0.775
0.800
0.825
0.850
beta1 (phi= 0.8)






```r
#理论渐近标准误
mu <- 1
sigma <- 1
T_size <- 900
AR1_CM <- function(phi, mu, sigma){
 EX <- mu / (1 - phi)
 EX2 <- (EX)^2 + 1 / (1 - phi^2)
 M <- matrix(c(1, EX, EX, EX2),
 ncol = 2)
 CM <- solve(M) * (sigma ^2) * (1/T_size)
 return(CM)
}
AR1_CM(phi[1], mu = 1, sigma = 1)
#              [,1]          [,2]
# [1,]  0.004444444 -0.0016666667
# [2,] -0.001666667  0.0008333333
AR1_CM(phi[2], mu = 1, sigma = 1)
#              [,1]    [,2]
# [1,]  0.01111111 -2e-03
# [2,] -0.00200000  4e-04
AR1_CM(phi[3], mu = 1, sigma = 1)
#               [,1]          [,2]
# [1,]  0.074074074 -2.188889e-03
# [2,] -0.002188889  6.566667e-05
sd_theory <- vector(mode = "list", length = 3)
for(i in 1:3){
 sd_theory[[i]] <- c(sd_beta0 = sqrt(AR1_CM(phi[i], mu = 1,
 sigma = 1)[1,1]), sd_beta1 =sqrt(AR1_CM(phi[i], mu = 1,
```

```
 sigma = 1)[2,2]))
}
names(sd_theory) <- c("phi=0.5","phi=0.8","phi=0.97")
sd_theory
# $phi=0.5
# sd_beta0    sd_beta1
# 0.06666667 0.02886751
#
# $phi=0.8
# sd_beta0   sd_beta1
# 0.1054093 0.0200000
#
# $phi=0.97
# sd_beta0     sd_beta1
# 0.272165527 0.008103497

#样本渐近标准误
x <- vector("double",901)
for (i in 1:3) {
 x[1] <- mu/(1 - phi[i])
 for (j in 1:900) {
  x[j+1] <- mu + phi[i]*x[j] + rnorm(1,0,1)
 }
 X <- cbind(rep(1,900), x[2:901])
 M <- t(X) %*% X / 900    # 求M矩阵
 CM<- 1/900*1*solve(M)
 print(CM)
 sd_sample<-c(sd_beta0=sqrt(CM[1,1]),sd_beta1=sqrt(CM[2,2]))
 print(sd_sample)
}
#              [,1]          [,2]
# [1,]  0.004363026 -0.0017251069
# [2,] -0.001725107  0.0009151513
# sd_beta0    sd_beta1
# 0.06605320 0.03025147
```

```
#                [,1]           [,2]
# [1,]  0.010068250 -0.0018359279
# [2,] -0.001835928  0.0003763067
# sd_beta0    sd_beta1
# 0.10034067 0.01939863
#                [,1]           [,2]
# [1,]  0.078249275 -0.0022643978
# [2,] -0.002264398  0.0000664716
# sd_beta0     sd_beta1
# 0.279730719 0.008153012
```

此时估计系数的样本标准差和渐进标准误均近似为先前的 1/3。