2021 秋季本科时间序列

第 5 次作业答案

11 月 12 日

1. (a)

$$M = \mathbb{E}\begin{bmatrix}1\\X_{t-1}\end{bmatrix}\begin{bmatrix}1 & X_{t-1}\end{bmatrix} = \begin{bmatrix}1 & \mathbb{E}X_{t-1}\\ \mathbb{E}X_{t-1} & \mathbb{E}X_{t-1}^2\end{bmatrix}, det(M) = \text{var}X_{t-1}$$

$$\text{var}X_t = \text{var}(\mu + \phi X_{t-1} + \epsilon) = \phi^2\text{var}X_{t-1} + \sigma_\epsilon^2$$

因为 AR(1) 过程 X_t 平稳，故 $\text{var}X_{t-1} = \text{var}X_t = \frac{1}{1-\phi^2} > 0, M$ 满秩

(b) $\gamma_0 = \frac{1}{1-\phi^2}, \gamma_k = \phi\gamma_{k-1} = \phi^k\gamma_0 = \frac{\phi^k}{1-\phi^2}$

$$S_X(\omega) = \frac{1}{2\pi}\sum_{k=-\infty}^{\infty}\gamma_k e^{-i\omega k}$$

由于常数项不影响谱密度函数，即求 AR(1) 过程 $X_t = \phi X_{t-1} + \epsilon_t$ 的谱密度函数，$X_t - \phi X_{t-1} = \epsilon_t, A(\mathscr{L})X_t = \epsilon_t$，则 $X_t = A^{-1}(\mathscr{L})\epsilon_t, A^{-1}(\mathscr{L}) = \frac{1}{1-\phi\mathscr{L}}$

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 $G(\omega) = \sqrt{C(e^{-i\omega})C(e^{i\omega})}$

则 $S_X(\omega) = G^2(\omega)S_\epsilon(\omega), S_\epsilon(\omega) = \frac{1}{2\pi}$，代入得，

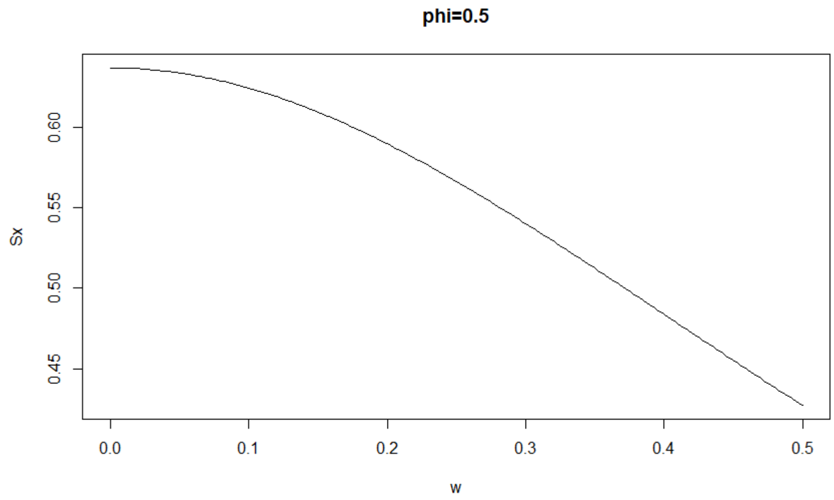
$$S_X(\omega) = \frac{1}{2\pi(1-\phi e^{-i\omega})(1-\phi e^{i\omega})} = \frac{1}{2\pi(1-2\phi\cos\omega + \phi^2)}$$

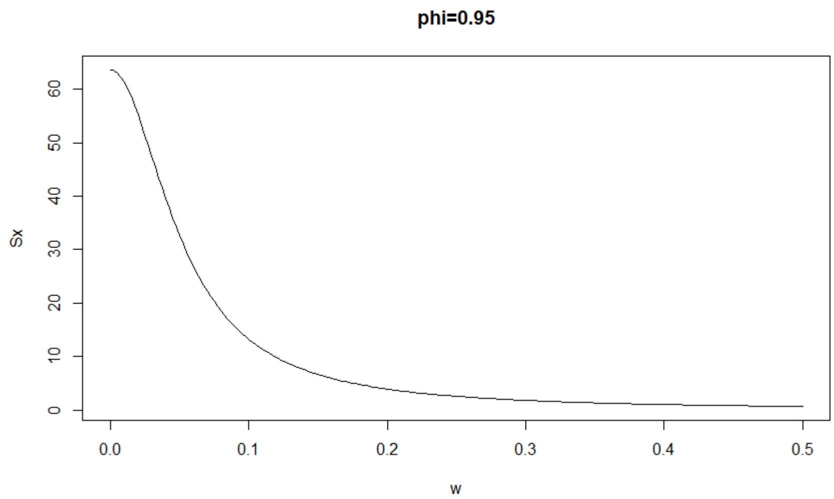
把 $\phi = 0.5, 0.95$ 代入得，$S_X(\omega)|(\phi = 0.5) = \frac{1}{2\pi(1.25-\cos\omega)}, S_X(\omega)|(\phi = 0.95) = \frac{1}{2\pi(1.9025-1.9\cos\omega)}$

代码如下：

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

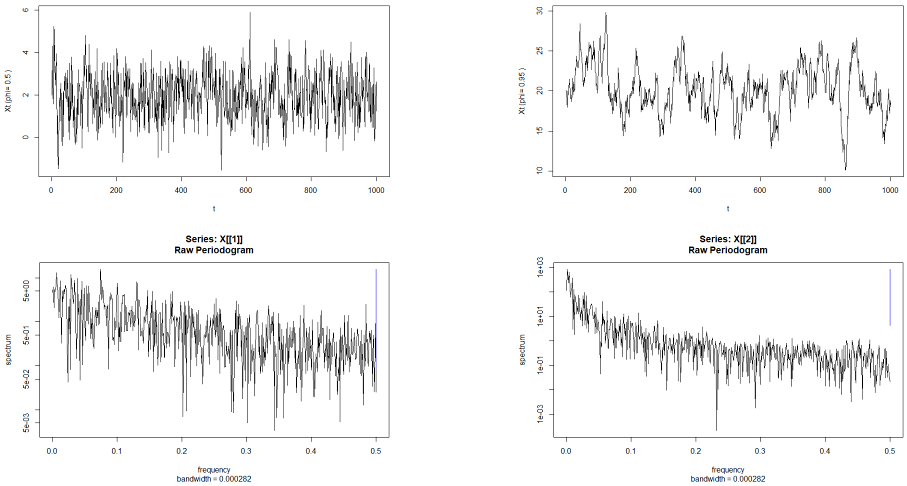
```
  y[i] <- 1/(2*pi*(1+0.95^2-2*0.95*cos(x[i])))
  z[i] <- 1/(2*pi*(1+0.5^2-2*0.5*cos(x[i])))
}
plot(x,z,type="l",main="phi=0.5",xlab="w",ylab="Sx")
plot(x,y,type="l",main="phi=0.95",xlab="w",ylab="Sx")
```





(c) 代码如下：

```
mu <- 1
phi <- c(0.5, 0.95)
X <- tibble()
for(i in 1:2){
  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)
```

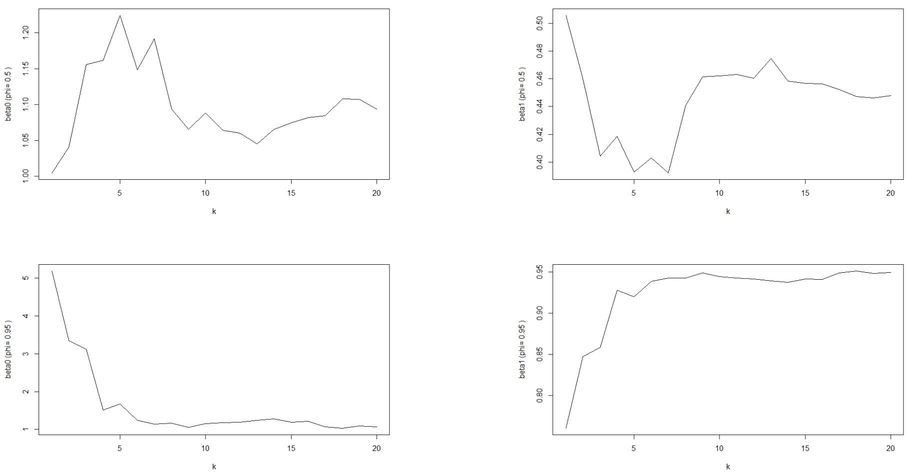



```
  }
  plot(X[[i]], type = "l", xlab = "t",
  ylab = paste('Xt (phi=', phi[i],')'))
}
spec.pgram(X[[1]])
spec.pgram(X[[2]])
```

(d) 代码如下：

```
beta0 <- tibble()
beta1 <- tibble()
for(i in 1:2){
  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。



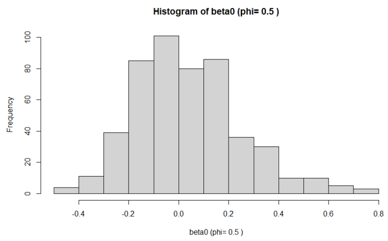
(e) 代码如下：

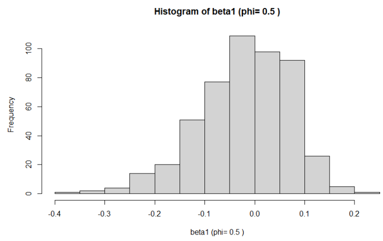
```r
# phi = 0.5
X1 <- tibble()
for(i in 1:500){
  X1[1, i] = mu/(1 - phi[1])
  for(j in 1:100){
    X1[j+1, i] = mu + phi[1]*X1[j, i] + rnorm(1,0,1)
  }
}
# phi = 0.95
X2 <- tibble()
for(i in 1:500){
  X2[1, i] = mu/(1 - phi[2])
  for(j in 1:100){
    X2[j+1, i] = mu + phi[2]*X2[j, i] + rnorm(1,0,1)
  }
}
X <- list(X1, X2)

beta0 <- tibble()
beta1 <- tibble()
b0 <- tibble()
b1 <- tibble()
sd_beta0 <- vector("double",2)
```

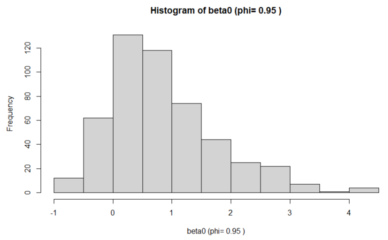
```
sd_beta1 <- vector("double",2)
for(i in 1:2){
  for(j in 1:500){
    y <- unlist(X[[i]][2:101, j])
    x <- unlist(X[[i]][1:100, j])
    ols <- coefficients(lm(y ~ x))
    beta0[j, i] <- ols[1]
    beta1[j, i] <- ols[2]
  }
  hist(beta0[[i]]-mu, xlab = paste('beta0 (phi=', phi[i],')'),
  main = paste('Histogram 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] 0.2147962 1.0858418
sd_beta1
## [1] 0.09592619 0.05361964
for(i in 1:2){
  for(j in 1:10000){
    b0[j,i] <- rnorm(1,mu,sd_beta0[i])
    b1[j,i] <- rnorm(1,phi[i],sd_beta1[i])
  }
  print(ggplot()+geom_histogram(data=NULL,mapping=aes
  (x=beta0[[i]],y=..density..))+xlab(paste('beta0 (phi=',
  phi[i],')'))+geom_density(data=NULL,mapping=aes(b0[[i]])))
  print(ggplot()+geom_histogram(data=NULL,mapping=aes
  (x=beta1[[i]],y=..density..))+xlab(paste('beta1 (phi=',
  phi[i],')'))+geom_density(data=NULL,mapping=aes(b1[[i]])))
}
```

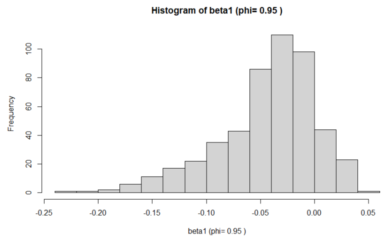
(f) 代码如下:

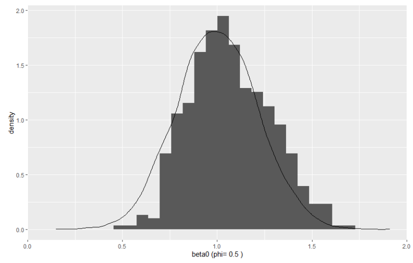
```
#理论渐近标准误
```

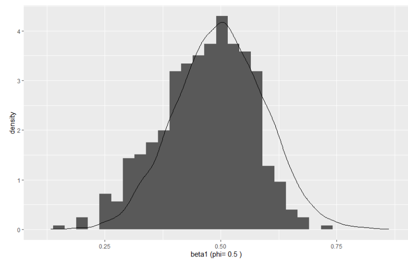

Histogram of beta0 (phi= 0.5)
Frequency
beta0 (phi= 0.5)

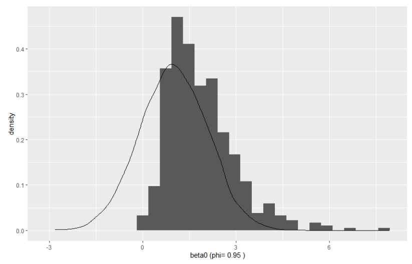

Histogram of beta1 (phi= 0.5)
Frequency
beta1 (phi= 0.5)

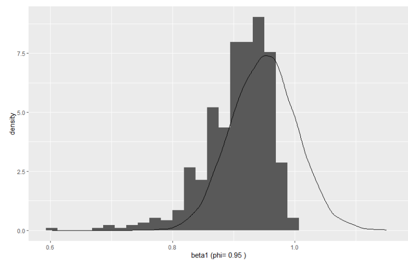

Histogram of beta0 (phi= 0.95)
Frequency
beta0 (phi= 0.95)


Histogram of beta1 (phi= 0.95)
Frequency
beta1 (phi= 0.95)


density
beta0 (phi= 0.5)


density
beta1 (phi= 0.5)


density
beta0 (phi= 0.95)


density
beta1 (phi= 0.95)

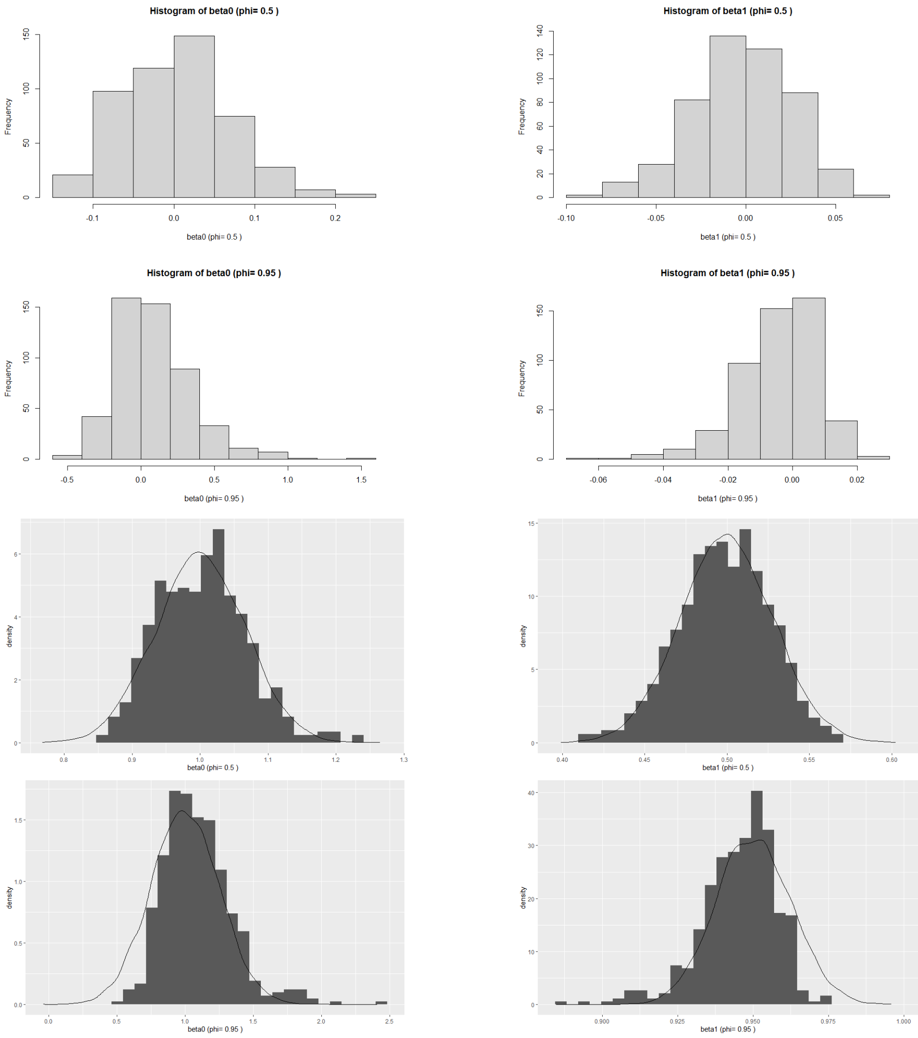
```r
M1  <- cbind(c(1,1/(1-phi[1])),c(1/(1-phi[1]),
  (1/(1-phi[1]))^2+1/(1-phi[1]^2)))
M2  <- cbind(c(1,1/(1-phi[2])),c(1/(1-phi[2]),
  (1/(1-phi[2]))^2+1/(1-phi[2]^2)))
sd_b0 <- vector("double",2)
sd_b1 <- vector("double",2)
sd_b0[1] <- sqrt(1*solve(M1)[1,1]*1/100)
sd_b1[1] <- sqrt(1*solve(M1)[2,2]*1/100)
sd_b0[2] <- sqrt(1*solve(M2)[1,1]*1/100)
sd_b1[2] <- sqrt(1*solve(M2)[2,2]*1/100)
sd_b0
## [1] 0.2000000 0.6324555
sd_b1
## [1] 0.08660254 0.03122499

#样本渐近标准误
x <- vector("double",101)
sd_beta0 <- vector("double",2)
sd_beta1 <- vector("double",2)
for (i in 1:2) {
  x[1] <- mu/(1 - phi[i])
  for (j in 1:100) {
    x[j+1] <- mu + phi[i]*x[j] + rnorm(1,0,1)
  }
  X <- cbind(rep(1,100), x[2:101])
  M <- t(X) %*% X / 100   # 求M矩阵
  sd_beta0[i] <- sqrt(1/100*1*solve(M)[1,1])
  sd_beta1[i] <- sqrt(1/100*1*solve(M)[2,2])
}
sd_beta0
## [1] 0.1907710 0.7808787
sd_beta1
## [1] 0.07786382 0.04222444
```

渐进标准误接近样本标准差的值。

(g) 代码如下：

```r
# 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 = 0.95
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)
  }
}
X <- list(X1, X2)

beta0 <- tibble()
beta1 <- tibble()
b0 <- tibble()
b1 <- tibble()
sd_beta0 <- vector("double",2)
sd_beta1 <- vector("double",2)
for(i in 1:2){
  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]])
}
for(i in 1:2){
  for(j in 1:10000){
    b0[j,i] <- rnorm(1,mu,sd_beta0[i])
    b1[j,i] <- rnorm(1,phi[i],sd_beta1[i])
  }
  print(ggplot()+geom_histogram(data=NULL,mapping=aes
  (x=beta0[[i]],y=..density..))+xlab(paste('beta0 (phi=',
  phi[i],')'))+geom_density(data=NULL,mapping=aes(b0[[i]])))
  print(ggplot()+geom_histogram(data=NULL,mapping=aes
  (x=beta1[[i]],y=..density..))+xlab(paste('beta1 (phi=',
  phi[i],')'))+geom_density(data=NULL,mapping=aes(b1[[i]])))
}
sd_beta0
## [1] 0.06504834 0.25155206
sd_beta1
## [1] 0.02794186 0.01237188

#样本渐近标准误
x <- vector("double",901)
sd_beta0 <- vector("double",2)
sd_beta1 <- vector("double",2)
for (i in 1:2) {
  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 / 100   # 求M矩阵
  sd_beta0[i] <- sqrt(1/100*1*solve(M)[1,1])
  sd_beta1[i] <- sqrt(1/100*1*solve(M)[2,2])
}
```




```
sd_beta0
## [1] 0.06706611 0.23503935
sd_beta1
## [1] 0.02905544 0.01125132
```

此时估计系数的样本标准差近似为 (c) 中的 1/3, 渐进标准误近似为 (d) 中的 1/3。