时间序列分析作业与第六次实验报告

姓名：康江睿

学号：2018213779

指导老师： 张晓飞

2020 年 12 月 1 日

1 计算理论和样本自相关函数

1.1 函数acfun简介

函数acfun实现了

(1) 对给定参数的AR过程和MA过程的理论自相关函数的计算与可视化。(2) 对给定数据的样本自相关函数的计算与可视化。

代码如下：

```
acfun = function(data = NULL, model = NULL, model.para = NULL,
                 lag.max, plot = TRUE, lag0 = FALSE){
  library(ggplot2)
  if (is.null(model)){
    data = data-mean(data)
    n = length(data)
    SST = t(data)%*%data
    gamma = integer(lag.max)
    for (i in 1:lag.max){
      gamma[i] = t(head(data,-i))%*%tail(data,-i)
    }
    acfval = gamma/as.numeric(SST)
    acfstd = sqrt(1/n*(1+2*cumsum(acfval^2)))
    maxacf = round(10000*max(abs(acfval)))/10000

    if (plot==TRUE){
      if (lag0==TRUE){
        data = data.frame(lags = 0:lag.max, acfval = c(1, acfval))
        acfstd = c(0, acfstd)
        acfplot = ggplot(data, aes(x = lags,y = acfval))+
```

```
21            geom_area(aes(x = lags, y = qnorm(0.975)*acfstd), fill = "#B9CFE7") +
22            geom_area(aes(x = lags, y = -qnorm(0.975)*acfstd), fill = "#B9CFE7") +
23            geom_segment(aes(x = lags, xend = lags, y = 0, yend = acfval))+
24            geom_point(size = 2, color = "red")+
25            scale_x_continuous("滞后数", breaks = seq(0,lag.max,2))+
26            scale_y_continuous("自相关函数值", breaks = seq(-1,1,0.2))+
27            ggtitle("ACF plot")
28        }else{
29          data = data.frame(lags = 1:lag.max, acfval = acfval)
30          acfplot = ggplot(data, aes(x = lags,y = acfval))+
31            geom_area(aes(x = lags, y = qnorm(0.975)*acfstd), fill = "#B9CFE7") +
32            geom_area(aes(x = lags, y = -qnorm(0.975)*acfstd), fill = "#B9CFE7") +
33            geom_segment(aes(x = lags, xend = lags, y = 0, yend = acfval))+
34            geom_point(size = 2, color = "red")+
35            scale_x_continuous("滞后数", breaks = seq(0,lag.max,2))+
36            scale_y_continuous("自相关函数值", breaks = seq(-maxacf,maxacf,maxacf/5))+
37            ggtitle("ACF plot")
38        }
39        result = list(acfval = acfval, acfplot = acfplot)
40      }else{
41        result = list(acfval = acfval)
42      }
43    }else{
44      if (model=="AR"){
45        l = length(model.para)
46        M = matrix(NA, nrow = l, ncol = l+1)
47        for (i in 1:l){
48          v = integer(2*l-1)
49          v[(l+1-i):(2*l-i)] = model.para
50          M[i,] = c(v[l], rev(v[1:(l-1)])+v[(l+1):(2*l-1)], 0)
51        }
52        N = -M[,1]
53        M = M[,2:(l+1)]-diag(x = 1, nrow = l)
54        acfval = t(as.matrix(qr.solve(M, N)))
55        ar = as.matrix(rev(model.para))
56        for (i in (l+1):lag.max){
57          acfval = cbind(acfval, t(tail(t(acfval), l))%*%ar)
58        }
59        acfval = as.vector(acfval)
60        maxacf = round(10000*max(abs(acfval)))/10000
61      }else if (model=="MA"){
62        l = length(model.para)
63        a = 1+sum(model.para^2)
64        rho = integer(lag.max)
65        for (i in 1:lag.max){
66          if (i<l){
67            v = c(-1, model.para[1:(l-i)])
68            u = model.para[i:l]
```

```
69            rho[i] = sum(v*u)/a
70          }else if (i==l){
71            rho[i] = -tail(model.para, 1)/a
72          }else{
73            rho[i] = 0
74          }
75        }
76        acfval = rho
77        maxacf = round(10000*max(abs(acfval)))/10000
78
79      }
80      for (i in 1:(l-1)){
81        if (i==1){
82          para = as.character(model.para[i])
83        }else{
84          para = paste(para, as.character(model.para[i]), sep = ",")
85        }
86      }
87      para = paste(para, as.character(model.para[l]))
88      if (plot==TRUE){
89        if (lag0==TRUE){
90          data = data.frame(lags = 0:lag.max, acfval = c(1, acfval))
91          acfplot = ggplot(data, aes(x = lags,y = acfval))+
92            geom_segment(aes(x = lags, xend = lags, y = 0, yend = acfval))+
93            geom_point(size = 2, color = "red")+
94            scale_x_continuous("滞后数", breaks = seq(0,lag.max,2))+
95            scale_y_continuous("自相关函数值", breaks = seq(-1,1,0.2))+
96            ggtitle(paste(c("ACF plot of"), model, c("["),
97                          as.character(model.para), c("]")))
98        }else{
99          data = data.frame(lags = 1:lag.max, acfval = acfval)
100         acfplot = ggplot(data, aes(x = lags,y = acfval))+
101           geom_segment(aes(x = lags, xend = lags, y = 0, yend = acfval))+
102           geom_point(size = 2, color = "red")+
103           scale_x_continuous("滞后数", breaks = seq(0,lag.max,2))+
104           scale_y_continuous("自相关函数值", breaks = seq(-maxacf,maxacf,maxacf/5))+
105           ggtitle(paste(c("ACF plot of"), model, c("["),
106                         para, c("]")))
107       }
108       result = list(acfval = acfval, acfplot = acfplot)
109     }else{
110       result = list(acfval = acfval)
111     }
112   }
113   return(result)
114 }
```

1.2 函数pacfun简介

函数acfun实现了

(1) 对给定参数的AR过程和MA过程的理论偏自相关函数的计算与可视化。(2) 对给定数据的样本偏自相关函数的计算与可视化。

代码如下：

```
pacfun = function(data = NULL, lag.max, plot = TRUE, method = NULL,
                  model = NULL, model.para = NULL){
  library(ggplot2)
  if (is.null(model)){
    result.acf = acfun(data = data, lag.max = lag.max, plot = FALSE)
    acfval = result.acf$acfval
    pacfval = integer(lag.max)
    if (is.null(method)|method=="system"){
      for (i in 1:lag.max){
        if (i==1){
          RHO = 1
          pacfval[1] = acfval[1]
        }else{
          RHO = rbind(c(1, acfval[1:(i-1)]), cbind(acfval[1:(i-1)], RHO))
          phi = qr.solve(RHO, acfval[1:i])
          pacfval[i] = tail(phi, 1)
        }
      }
    }else if (method=="iteration"){
      for (i in 1:lag.max){
        if (i==1){
          pacfval[1] = acfval[1]
          PHI = acfval[1]
        }else{
          pacfval[i] = (acfval[i]-PHI%*%rev(acfval[1:(i-1)]))/
            (1-PHI%*%acfval[1:(i-1)])
          PHI = PHI-pacfval[i]*rev(PHI)
          PHI = c(PHI, pacfval[i])
        }
      }
    }
    n = length(data)
    pacfstd = sqrt(1/n)
    maxpacf = round(10000*max(abs(pacfval)))/10000

    if (plot==TRUE){
      data = data.frame(lags = 1:lag.max, pacfval = pacfval)
      pacfplot = ggplot(data, aes(x = lags,y = pacfval))+
        geom_area(aes(x = lags, y = qnorm(0.975)*pacfstd), fill = "#B9CFE7") +
        geom_area(aes(x = lags, y = -qnorm(0.975)*pacfstd), fill = "#B9CFE7") +
```

```
41          geom_segment(aes(x = lags, xend = lags, y = 0, yend = pacfval))+
42          geom_point(size = 2, color = "red")+
43          scale_x_continuous("滞后数", breaks = seq(0,lag.max,2))+
44          scale_y_continuous("偏自相关函数值", breaks = seq(-maxpacf,maxpacf,maxpacf/5))+
45          ggtitle("PACF plot")
46        result = list(pacfval = pacfval, pacfplot = pacfplot)
47      }else{
48        result = list(pacfval = pacfval)
49      }
50    }else{
51      result.acf = acfun(model = model, model.para = model.para,
52                         lag.max = lag.max)
53      acfval = result.acf$acfval
54      pacfval = integer(lag.max)
55      for (i in 1:lag.max){
56        if (i==1){
57          RHO = 1
58          pacfval[1] = acfval[1]
59        }else{
60          RHO = rbind(c(1,acfval[1:(i-1)]), cbind(acfval[1:(i-1)], RHO))
61          phi = qr.solve(RHO, acfval[1:i])
62          pacfval[i] = tail(phi,1)
63        }
64      }
65      maxpacf = round(10000*max(abs(pacfval)))/10000
66      for (i in 1:(l-1)){
67        if (i==1){
68          para = as.character(model.para[i])
69        }else{
70          para = paste(para, as.character(model.para[i]), sep = ",")
71        }
72      }
73      para = paste(para, as.character(model.para[l]))
74
75      if (plot==TRUE){
76        data = data.frame(lags = 1:lag.max, pacfval = pacfval)
77        pacfplot = ggplot(data, aes(x = lags,y = pacfval))+
78          geom_segment(aes(x = lags, xend = lags, y = 0, yend = pacfval))+
79          geom_point(size = 2, color = "red")+
80          scale_x_continuous("滞后数", breaks = seq(0,lag.max,2))+
81          scale_y_continuous("偏自相关函数值", breaks = seq(-maxpacf,maxpacf,maxpacf/5))+
82          ggtitle(paste(c("PACF plot of"), model, c("["),
83                        para, c("]")))
84        result = list(pacfval = pacfval, pacfplot = pacfplot)
85      }else{
86        result = list(pacfval = pacfval)
87      }
88    }
```

```
89
90   return(result)
91 }
```

1.3 函数使用说明

对AR(p)过程，记为：

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + e_t$$

对MA(q)过程，记为：

$$Y_t = e_t - \theta_1 e_{t-1} - \cdots - \theta_q e_{t-q}$$

记$\boldsymbol{\phi} = [\phi_1, \phi_2, \cdots, \phi_p]$，$\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_q]$

(1) 函数acfun的输入参数的名称与含义如下表：

输入参数名称	含义
data	给定的（时间序列）数据
model	时间序列模型，”AR”或”MA”
model.para	参数集$\boldsymbol{\phi}$(AR)或$\boldsymbol{\theta}$(MA)
lag.max	最大滞后数
plot	画图选项
lag0	选择是否在图中绘制$\rho_0 = 1$的情况

表 1: 函数acfun的输入参数的名称与含义

(2) 函数pacfun的输入参数的名称与含义如下表：

输入参数名称	含义
data	给定的（时间序列）数据
model	时间序列模型，”AR”或”MA”
model.para	参数集$\boldsymbol{\phi}$(AR)或$\boldsymbol{\theta}$(MA)
lag.max	最大滞后数
plot	画图选项
method	计算样本偏相关函数的方式，”system”（默认）或”iteration”

表 2: 函数pacfun的输入参数的名称与含义

(3) 输入参数plot和lag0都是布尔型变量。

(4) 函数的输出参数都是列表。如果plot为TRUE，则列表包含lag.max阶滞后的（偏）自相关函数值和可视化结果；如果plot为FALSE，则列表只包含lag.max阶滞后的（偏）自相关函数值。

(5) 函数pacfun需要调用函数acfun。

(6) 注意：使用函数前请下载ggplot2程辑包。

1.4 函数功能测试

首先测试acfun函数的功能：

(1a) 计算样本自相关函数的数值：

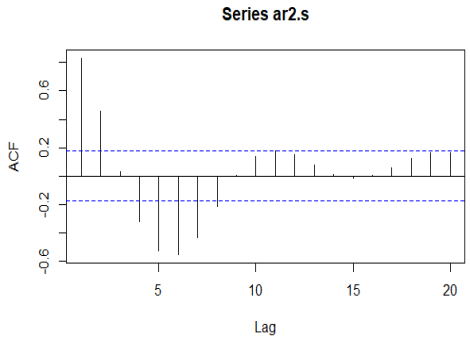
```
> acf(ar1.s,lag.max = 20,plot = F)

Autocorrelations of series 'ar1.' s, by lag

     1      2      3      4      5      6      7      8      9     10     11     12     13     14     15
 0.831  0.719  0.598  0.513  0.405  0.282  0.221  0.122  0.037 -0.055 -0.128 -0.207 -0.272 -0.291 -0.251
    16     17     18     19     20
-0.199 -0.150 -0.167 -0.125 -0.099
> acfun(ar1.s,lag.max = 20,plot = F)
$acfval
 [1]  0.83138203  0.71923267  0.59835829  0.51298864  0.40503456  0.28160185  0.22139516  0.12177077
      0.03676996 -0.05489399 -0.12812391 -0.20730001
[13] -0.27156550 -0.29140294 -0.25107408 -0.19883155 -0.14992094 -0.16743910 -0.12472817 -0.09854198
```

(1b) 计算并绘制样本自相关函数图：

```
> acf(ar2.s,lag.max = 20)
> acfun(ar2.s,lag.max = 20)
```



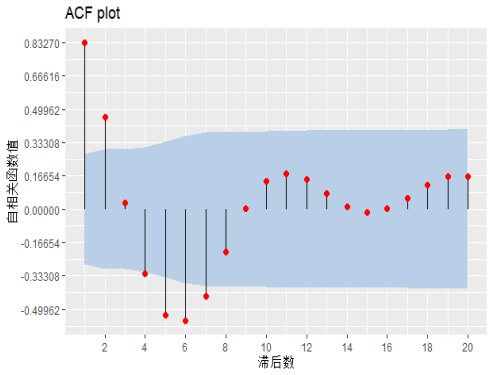


图 1: acf函数绘制的样本自相关函数图　图 2: acfun函数绘制的样本自相关函数图

(1c) 计算AR模型的理论自相关函数：

```
> acfun(model = "AR",model.para = c(1.2,−0.7),lag.max = 20,plot = F)
$acfval
 [1]  0.70588235  0.14705882 −0.31764706 −0.48411765 −0.35858824 −0.09142353
 [7]  0.14130353  0.23356071  0.18136038  0.05413996 −0.06198431 −0.11227915
[13] −0.09134596 −0.03101975 0.02671848 0.05377599 0.04582826  0.01735071
[19] −0.01125892 −0.02565621

> ARMAacf(ar = c(1.2,−0.7),lag.max = 20)
          0           1           2           3           4           5
 1.00000000  0.70588235  0.14705882 −0.31764706 −0.48411765 −0.35858824
          6           7           8           9          10          11
−0.09142353 0.14130353  0.23356071  0.18136038  0.05413996 −0.06198431
         12          13          14          15          16          17
−0.11227915 −0.09134596 −0.03101975 0.02671848 0.05377599 0.04582826
         18          19          20
 0.01735071 −0.01125892 −0.02565621
```

(1d) 计算MA模型的理论自相关函数：

```
> acfun(model = "MA",model.para = c(0.5,−0.5),lag.max = 5,plot = F)
$acfval
[1] −0.5000000 0.3333333  0.0000000  0.0000000  0.0000000

> ARMAacf(ma = c(−0.5,0.5),lag.max = 5)
          0           1           2           3           4           5
 1.0000000 −0.5000000 0.3333333  0.0000000  0.0000000  0.0000000
```

接下来测试pacfun函数的功能：

(2a) 计算样本偏自相关函数的数值：

```
> pacf(ar2.s,lag.max = 20,plot = F)

Partial autocorrelations of series ‘ar2.’ s, by lag

     1      2      3      4      5      6      7      8      9     10
 0.833 −0.765 −0.037 −0.105 −0.037 0.006 −0.022 0.052 −0.237 −0.042
    11     12     13     14     15     16     17     18     19     20
−0.008 0.005 −0.024 0.040  0.086 −0.065 0.127  0.044  0.041  0.054
> pacfun(ar2.s,lag.max = 20,plot = F)
$pacfval
 [1]  0.832735369 −0.764603375 −0.036765964 −0.105071822 −0.037105344
 [6]  0.005517211 −0.022027182 0.051601004 −0.237199486 −0.041937692
[11] −0.007581464 0.005142360 −0.023581125 0.040143306 0.085986034
[16] −0.064500144 0.126806184 0.043526077  0.040602444 0.053742143
```

(2b) 计算并绘制样本偏自相关函数图：

```
> pacf(ar2.s,lag.max = 20)
```

```
> pacfun(ar2.s,lag.max = 20)
```

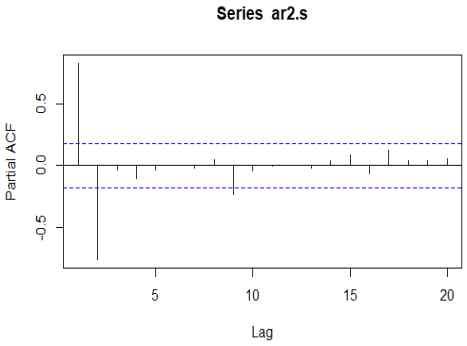


图 3: pacf函数绘制的样本偏自相关函数图

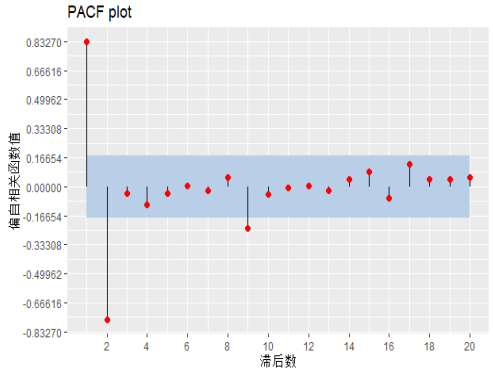


图 4: pacfun函数绘制的样本偏自相关函数图

(2c) 计算AR模型的理论偏自相关函数：

```
> ARMAacf(ar = c(1.2,−0.7),lag.max = 20,pacf = T)
 [1]  7.058824e−01 −7.000000e−01 −6.508204e−16 3.310207e−16 −2.023657e−16
 [6]  2.701562e−16 −2.277871e−16 5.423503e−17 −1.843991e−16 3.525277e−16
[11] −2.982927e−16 9.762306e−17 9.762306e−17 −1.084701e−16 6.508204e−17
[16] −2.440576e−17 −5.423503e−18 6.643792e−17 −5.152328e−17 4.067627e−18
> pacfun(model = "AR",model.para = c(1.2,−0.7),lag.max = 20,plot = F)
$pacfval
 [1]  7.058824e−01 −7.000000e−01 −2.970734e−15 1.146821e−15 −5.260798e−16
 [6]  1.627051e−16 −3.254102e−16 4.338803e−17 1.355876e−16 2.549047e−16
[11]  1.545698e−16 −5.152328e−17 −5.423503e−18 2.440576e−17 −1.084701e−17
[16] −5.965854e−17 5.152328e−17 7.999667e−17 2.440576e−17 −2.643958e−17（显然数值的不同是由舍入误差造成的）
```

(2d) 计算MA模型的理论偏自相关函数：

```
> ARMAacf(ma = c(−0.5,0.5),lag.max = 20,pacf = T)
 [1] −5.000000e−01 1.111111e−01 2.750000e−01 8.722110e−02 −9.104478e−02
 [6] −8.822444e−02 1.012146e−03 4.448743e−02 2.173913e−02 −1.135802e−02
[11] −1.654254e−02 −2.594063e−03 6.973572e−03 4.783702e−03 −1.094839e−03
[16] −2.939236e−03 −9.222047e−04 1.008512e−03 9.653573e−04 −2.157693e−05
> pacfun(model = "MA",model.para = c(0.5,−0.5),lag.max = 20,plot = F)
$pacfval
 [1] −5.000000e−01 1.111111e−01 2.750000e−01 8.722110e−02 −9.104478e−02
 [6] −8.822444e−02 1.012146e−03 4.448743e−02 2.173913e−02 −1.135802e−02
[11] −1.654254e−02 −2.594063e−03 6.973572e−03 4.783702e−03 −1.094839e−03
[16] −2.939236e−03 −9.222047e−04 1.008512e−03 9.653573e−04 −2.157693e−05
```

综上，函数在TSA数据集上的表现和库函数acf,pacf,ARMAacf一致，应该认为是有效的。

2 课后习题(6.24,6.25,6.27,6.28,6.33)

在做习题之前，先导入必要的程辑包和函数：

```
> library(ggplot2)
> source('D:/R Files/TSAcourse/acfun.R', encoding = 'UTF-8')
> source('D:/R Files/TSAcourse/pacfun.R', encoding = 'UTF-8')
> source('D:/R Files/TSAcourse/simARIMA.R')
```

simARIMA是本次习题中用于模拟仿真的函数。

2.1 习题6.24

(a,b,c) 实现的代码如下：

```
v24 = simARIMA(MA.para = 0.7, diff = 0, seq.length = 24)
v60 = simARIMA(MA.para = 0.7, diff = 0, seq.length = 60)
v120 = simARIMA(MA.para = 0.7, diff = 0, seq.length = 120)
rho = acfun(model = "MA", model.para = 0.7, lag.max = 1, plot = F)
r24 = acfun(data = v24, lag.max = 1, plot = F)
r60 = acfun(data = v60, lag.max = 1, plot = F)
r120 = acfun(data = v120, lag.max = 1, plot = F)
```

结果如下：

```
> rho
$acfval
[1] −0.4697987

> r24
$acfval
[1] −0.3494682

> r60
$acfval
[1] −0.5749973

> r120
$acfval
[1] −0.5074666
```

(d) 可以发现，对于越大的n，ρ_1的估计值离理论值越接近。

量化图表如下：

由表可知，随着样本容量的增加，估计的精度逐渐提升。

n	$\sqrt{Var(r_1)}$	$\sqrt{Var(r_k)}, k>1$	$Corr(r_1, r_2)$
24	0.1490	0.2490	-0.84
60	0.0942	0.1575	-0.84
120	0.0666	0.1114	-0.84

表 3: 参数为0.7的MA(1)模型选定的r_k的大样本结果

2.2 习题6.25

实现的代码如下：

```
sim.25 = simARIMA(AR.para = 0.7, diff = 0, seq.length = 36)
theo.25 = acfun(model = "AR", model.para = 0.7, lag.max = 20)
samp.25 = acfun(data = sim.25, lag.max = 20)
theo.p.25 = pacfun(model = "AR", model.para = 0.7, lag.max = 20)
samp.p.25 = pacfun(data = sim.25, lag.max = 20)
```

(a,b) 理论自相关函数和样本自相关函数的可视化结果如下：

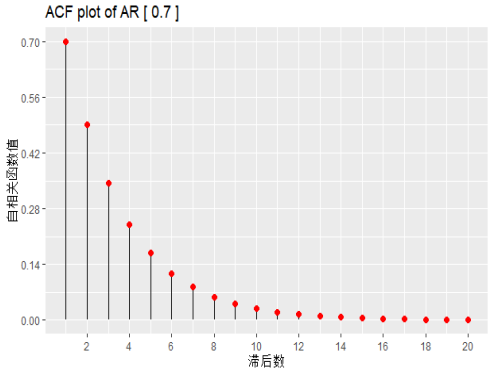


图 5: AR(1)模型的理论自相关函数

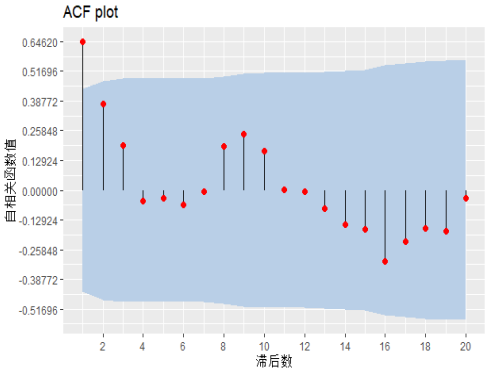


图 6: 模拟AR(1)序列的样本自相关函数

相比之下，两者差异很大，很不匹配。

(c,e) 该模型的理论偏自相关函数仅在一阶滞后不等于0，其取值刚好为模型参数值。表达式如下：

$$\phi_{kk} = Corr(Y_t - \beta_1 Y_{t-1}, Y_{t-k} - \beta_1 Y_{t-k+1})$$

其中β_1为自回归模型参数的最优预测。

理论偏自相关函数和样本偏自相关函数的可视化结果如下：

相比之下，两者差异很大，很不匹配。但是抛去匹配度的问题，从模拟AR(1)序列的样本偏自相关函数中，我们可以看出模拟序列的AR部分定为1阶最合适，故此处的样本偏自相关函数对于模型识别仍然是有效的。

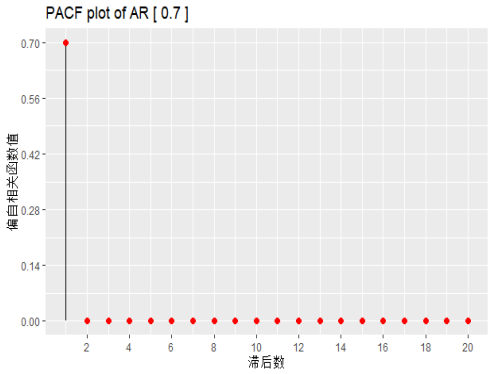


图 7: AR(1)模型的理论偏自相关函数

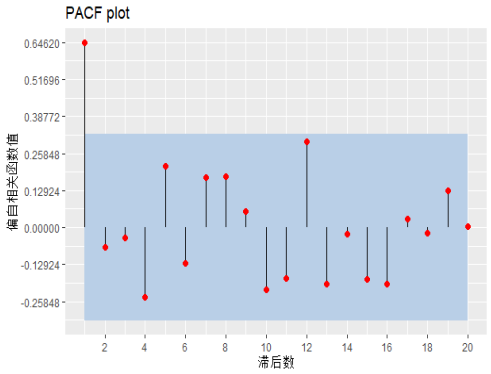


图 8: 模拟AR(1)序列的样本偏自相关函数

(c,d) 用大样本标准误差量化样本自相关函数，那么对于越高的滞后数，估计的样本自相关函数的方差会更大。因此我们可以发现样本自相关函数在前几阶与理论自相关函数的差别是不大的，但是对于较高的滞后（如4阶以后），样本自相关函数在前几阶与理论自相关函数的差别很大。

用大样本标准误差量化样本偏自相关函数，那么估计的样本偏自相关函数的方差将在大样本情形下近似服从$N(0, 1/n)$。图像表明，样本偏自相关函数的取值都没有超出$1/\sqrt{n}$的范围，这印证了上面的结论。

2.3 习题6.27

实现的代码如下：

```
sim.27 = simARIMA(AR.para = c(0.7, −0.4), diff = 0, seq.length = 72)
theo.27 = acfun(model = "AR", model.para = c(0.7, −0.4), lag.max = 20)
samp.27 = acfun(data = sim.27, lag.max = 20)
theo.p.27 = pacfun(model = "AR", model.para = c(0.7, −0.4), lag.max = 20)
samp.p.27 = pacfun(data = sim.27, lag.max = 20)
```

(a,b) 理论自相关函数和样本自相关函数的可视化结果如下：

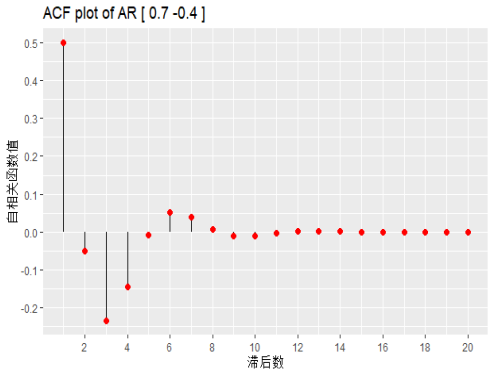


图 9: AR(2)模型的理论自相关函数

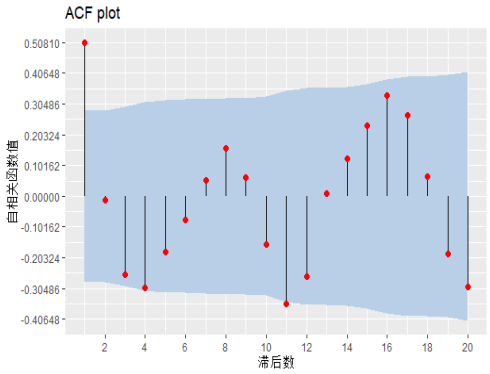


图 10: 模拟AR(2)序列的样本自相关函数

相比之下，两者在前几阶比较接近，但在后几阶差异很大，很不匹配。

(c,e) 该模型的理论偏自相关函数仅在1、2阶滞后上不为0。表达式如下：

$$\phi_{kk} = Corr(Y_t - \beta_1 Y_{t-1} - \beta_2 Y_{t-2}, Y_{t-k} - \beta_1 Y_{t-k+1} - \beta_2 Y_{t-k+2})$$

其中β_1, β_2为自回归模型参数的最优预测。

理论偏自相关函数和样本偏自相关函数的可视化结果如下：

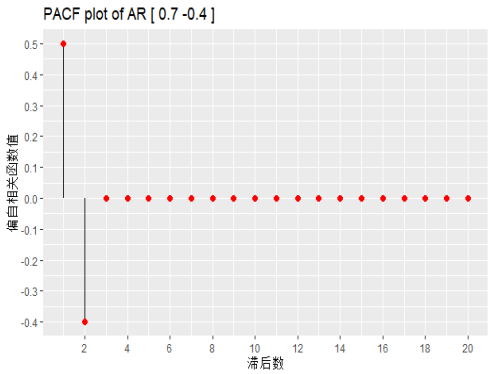


图 11: AR(2)模型的理论偏自相关函数

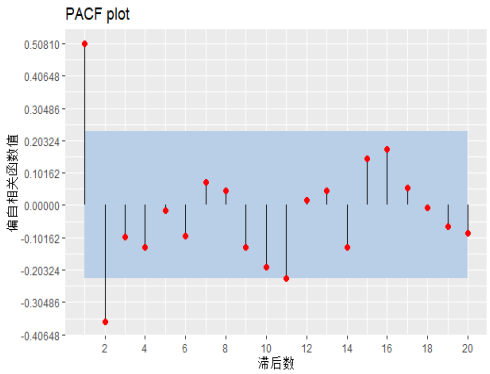


图 12: 模拟AR(2)序列的样本偏自相关函数

相比之下，两者在大于2阶的滞后上差异很大，很不匹配。但是抛去匹配度的问题，从模拟AR(2)序列的样本偏自相关函数中，我们可以看出模拟序列的AR部分定为2阶最合适，故此处的样本偏自相关函数对于模型识别仍然是有效的。

2.4 习题6.28

实现的代码如下：

```r
sim.28 = simARIMA(MA.para = c(0.7, -0.4), diff = 0, seq.length = 3600)
theo.28 = acfun(model = "MA", model.para = c(0.7, -0.4), lag.max = 20)
samp.28 = acfun(data = sim.28, lag.max = 20)
theo.p.28 = pacfun(data = simARIMA(MA.para = c(0.7, -0.4),
                                   diff = 0, seq.length = 50000),
                   lag.max = 20)
samp.p.28 = pacfun(data = sim.28, lag.max = 20)
```

(a,b) 理论自相关函数和样本自相关函数的可视化结果如下：

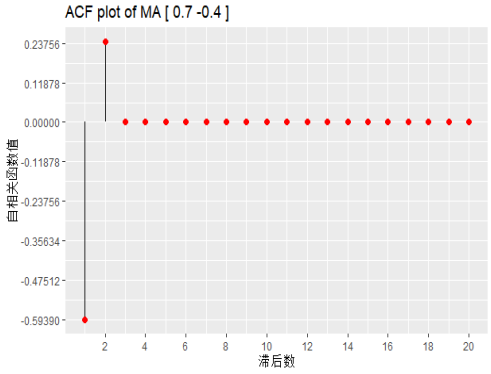


图 13: MA(2)模型的理论自相关函数

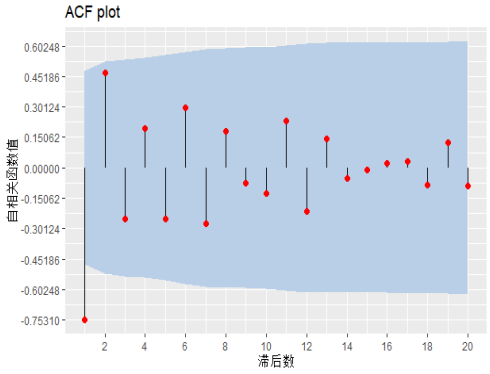


图 14: 模拟MA(2)序列的样本自相关函数

相比之下，两者在前几阶比较接近，但在后几阶差异很大，很不匹配。除此以外，模拟MA(2)序列的样本自相关函数图表明，模拟序列的MA部分定为1阶最合适，与模拟的情形矛盾。故此处的样本偏自相关函数对于模型识别是错误的，这可能要归因于样本量太小。

(c,e) 理论偏自相关函数和样本偏自相关函数的可视化结果如下：

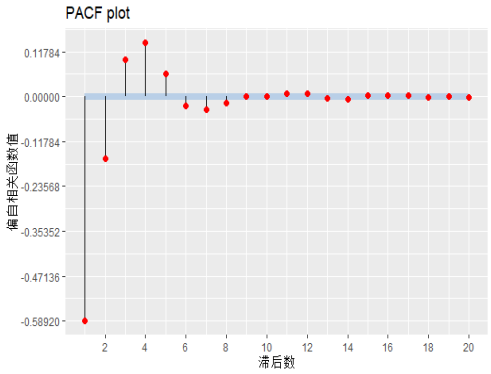


图 15: MA(2)模型的理论偏自相关函数

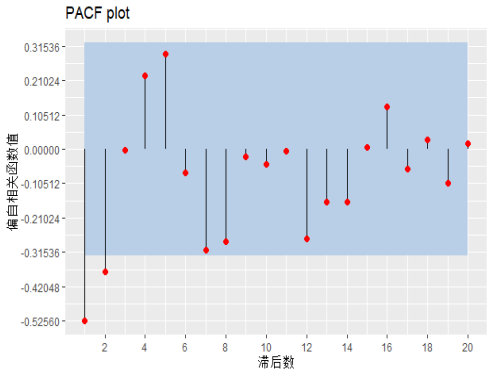


图 16: 模拟MA(2)序列的样本偏自相关函数

相比之下，两者在大于6阶的滞后上差异很大，很不匹配。

2.5 习题6.33

实现的代码如下：

```
data("deere1")
ts33 = plot.ts(deere1, type = "o")
samp.33 = acfun(data = deere1, lag.max = 20)
deere1[which.max(deere1)] = (deere1[which.max(deere1)+1]+
  deere1[which.max(deere1)-1])/2
samp.d.33 = acfun(data = deere1.d, lag.max = 20)
samp.p.33 = pacfun(data = deere1.d, lag.max = 20)
```

(a) 时间序列图如下：

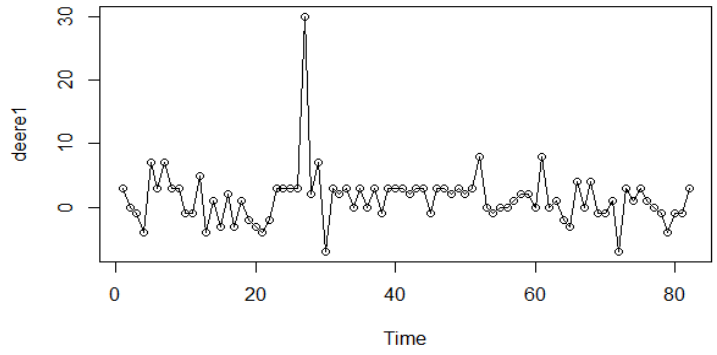


图 17: deere1数据的时间序列图

从图中明显可以看出，有一个点的取值（相比总体情况）非常大，因此认为这是一个异常点，应予以删除或者插值替换。

(b) 计算并可视化序列的样本ACF，结果如下：

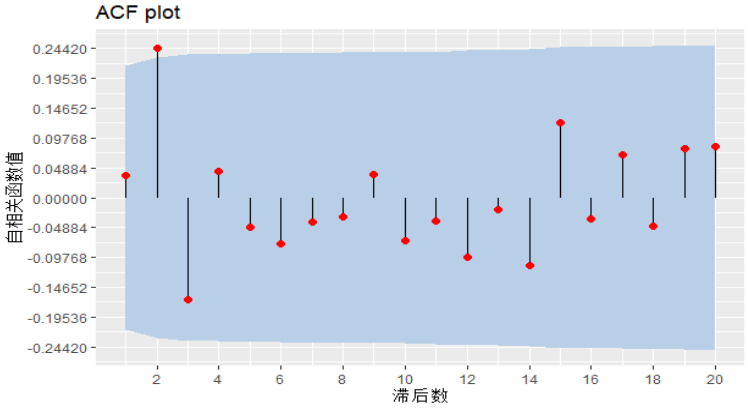


图 18: deere1序列的样本ACF图

序列的样本ACF图显示，其一阶样本自相关函数值很低，但是二阶样本自相关函数显著大于0。故推测可能序列服从AR(p)过程，或者是异常值点影响了样本自相关函数的表现。

(c) 考虑到时间序列并没有表现出明显的趋势，变化也不规律，故考虑直接用序列异常值点前后两个点的观测值均值进行插值替换。对新的数据计算并可视化样本ACF如下：

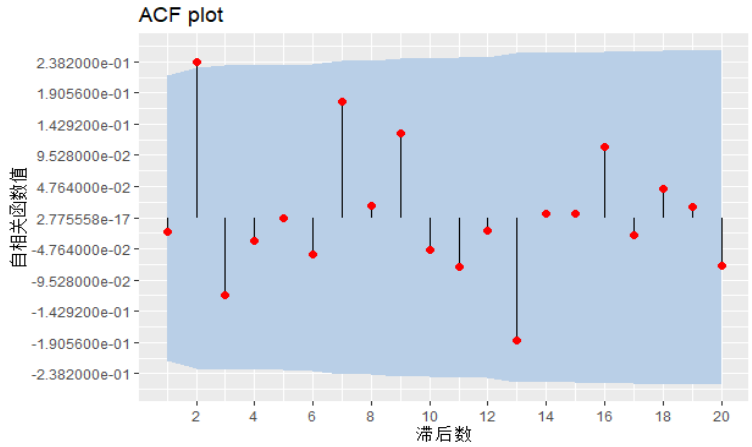


图 19: 插值替换后deere1序列的样本ACF图

结果表明，对异常值的处理仍然不能改善样本ACF图的模型识别能力。故考虑使用PACF来进行模型识别。

(d) 对新的数据计算并可视化样本PACF 如下：

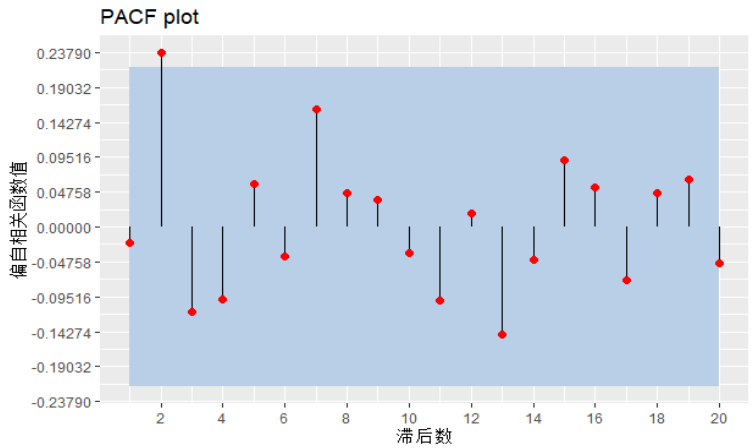


图 20: 插值替换后deere1序列的样本PACF图

结果表明，样本PACF图也无法直接对模型进行定阶，故我们可以考虑设定一些其他的模型来描述数据。由样本自相关图可以发现，样本数据的自相关性

并不是很显著，而且序列数据是平稳的，因此我们甚至可能认为模型是白噪声序列。