时间序列分析 • project

魏嵘 PB19151785

目录

1 引言

1.1 时间序列的概念

在科学研究的诸多领域里，被考察的对象在发展过程中会受到各种偶然因素的影响，往往表现出某种随机性，并后续以一串依时间变化的数据序列的形式被记载下来，而这种序列通常被称为时间序列。与一般的多元统计分析不同的是，时间序列给不同的观测赋予了时间的顺序，我们真正感兴趣的是通过研究当前和过去不同时间点的数据之间的联系，试图建立起此序列变化发展的规律，进而预测它将来的走势，对变量未来的变化进行预报；或者通过调整当前的输入变量，实现对系统发展的控制。

1.2 时间序列的分析方法

最早的时间序列分析往往描述性时间序列分析，通过图表的形式描述和呈现收集到的数据，体现它们的波动情况。后续随着研究领域的不断扩展，对时间序列的分析方法大致可以分成频域分析和时域分析。频域分析借助傅里叶分析的思想，假设无趋势的时间序列可以分解成不同频率的周期波动，是从频域中揭示时间序列的规律；而时域的分析方法是想从自相关的角度在时域中分析序列内部的统计规律性。1970 年，Box 和 Jenkins 在梳理、发展已有成果的基础上，较为系统的阐述的 ARIMA 模型的识别、估计、检验、预测的原理域方法。近些年来，在上述基础上，人们又不断拓展研究方法，转向多变量、异方差、非线性等各种场合，给出了不同的模型去刻画复杂序列的变化过程。

1.3 时间序列建模的基本步骤

从实际数据出发，我们可以将时间序列建模的归纳为五个步骤：

1. 数据预处理

经过数据清洗剔除无效值、数据类型转化使标度统一等步骤后，进行描述性时间序列分析，根据数据做出时序图、相关图等，直观感受数据变化的趋势性和周期性以及发现序列中的一些跳点和拐点，为模型识别的重要参考因素

2. 模型识别

- 判断时间序列的平稳性：通过时序图、自相关图初步判断时间序列的平稳性，并加以统计检验的方法辅助判别，如单位根检验。
- 判断时间序列的随机性：通过平稳性判别后，正如上述所言，我们感兴趣的是序列内部自相关的统计规律，只有序列值之间存在密切相关的序列才值得我们进一步分析建模。因此需要进行纯随机性检验。
- 结合常见的时间序列模型与实际数据展现的特征，初步确定模型结构

3. 模型估计

对于平稳的非白噪声序列，基于样本自相关函数和偏自相关函数的性质选择阶数适当的模型进行拟合，借助数理统计中常用的模型参数估计方法（矩估计、极大似然估计、最小二乘估计等）估计模型中的未知参数的值。

4. 模型检验

由于模型的识别和提出是基于图像的直观的假定，有较强的主观因素，因此需要对模型本身和所估计出的参数的显著性进行检验。

5. 模型优化

有时在一定的置信水平下，能够有效拟合观测值序列的模型并不唯一，因此需要引入一些信息准则来选择统计推断更好的模型。

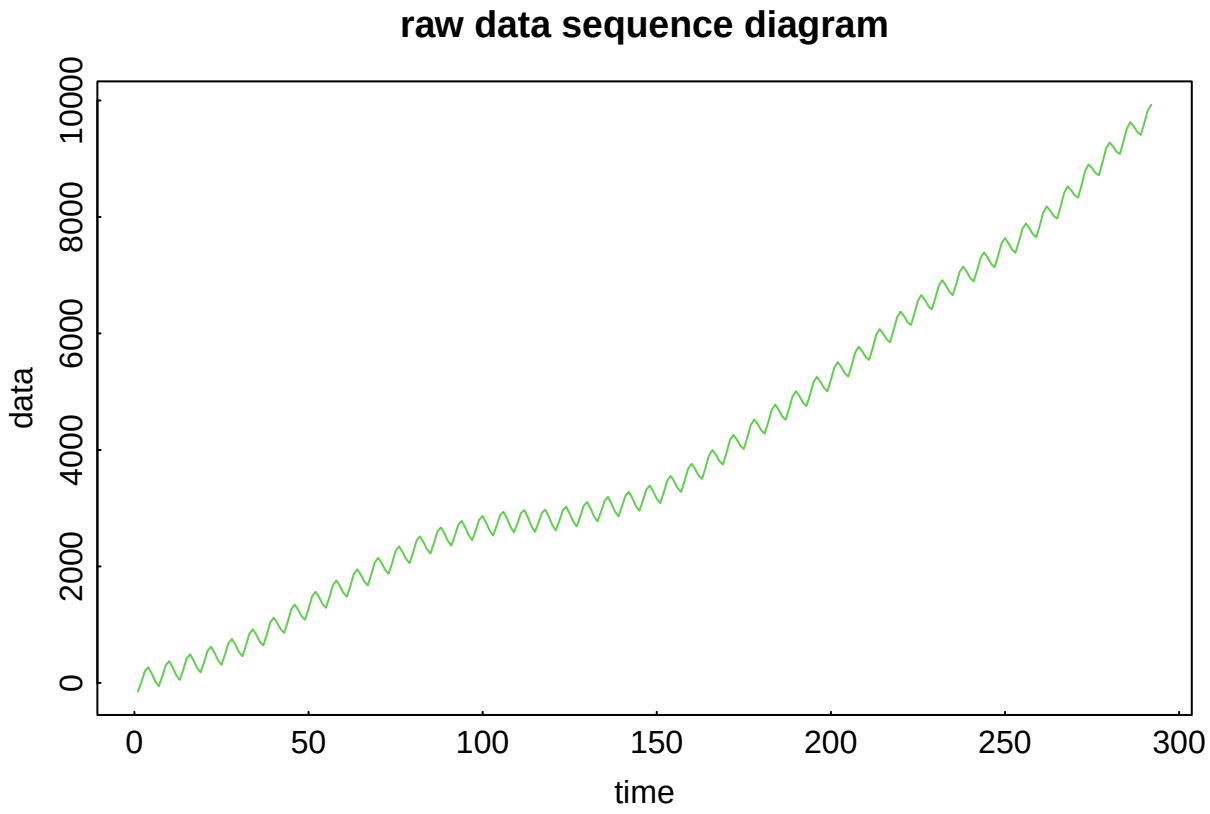
2 构建模型

2.1 数据预处理

```
ts = read.csv("data3.csv") # 读入数据表格
my_ts = data.frame(time =seq(1,ncol(ts)-2),
                   data = as.numeric(ts[ts$学号 == "PB19151785",3:ncol(ts)]))
# 提取时序数据（去除前十个数据）
head(my_ts,6) # 打印数据
```

```
##   time       data
## 1    1  125.90000
## 2    2  177.10000
## 3    3   52.91271
## 4    4  -93.90303
## 5    5 -189.55316
## 6    6  -35.87318
```

```
data = c(my_ts$data)
my_data = ts(data[11:length(data)])
opar = par(mar = c(3,3,3,1),mgp=c(1.5,0.2,0),tck=0.005)
plot(my_data,main="raw data sequence diagram",ylab="data",xlab="time",col=3)
```




```
par(opar)
```

通过时序图，我们可以发现序列表现出

- 长期趋势 (T_t)：长期呈现单调增长的趋势，但中间存在一段较为平稳的平台期
- 循环波动 (S_t)：稳定的周期性的由低到高再由高到低的反复循环波动，且波动的周期为 6
- 随机波动 (R_t)：不能用确定性因素解释的序列的波动

由于每个周期的振幅相对稳定，则说明季节效应没有受到趋势的影响，因此选择加法模型进行因素分解

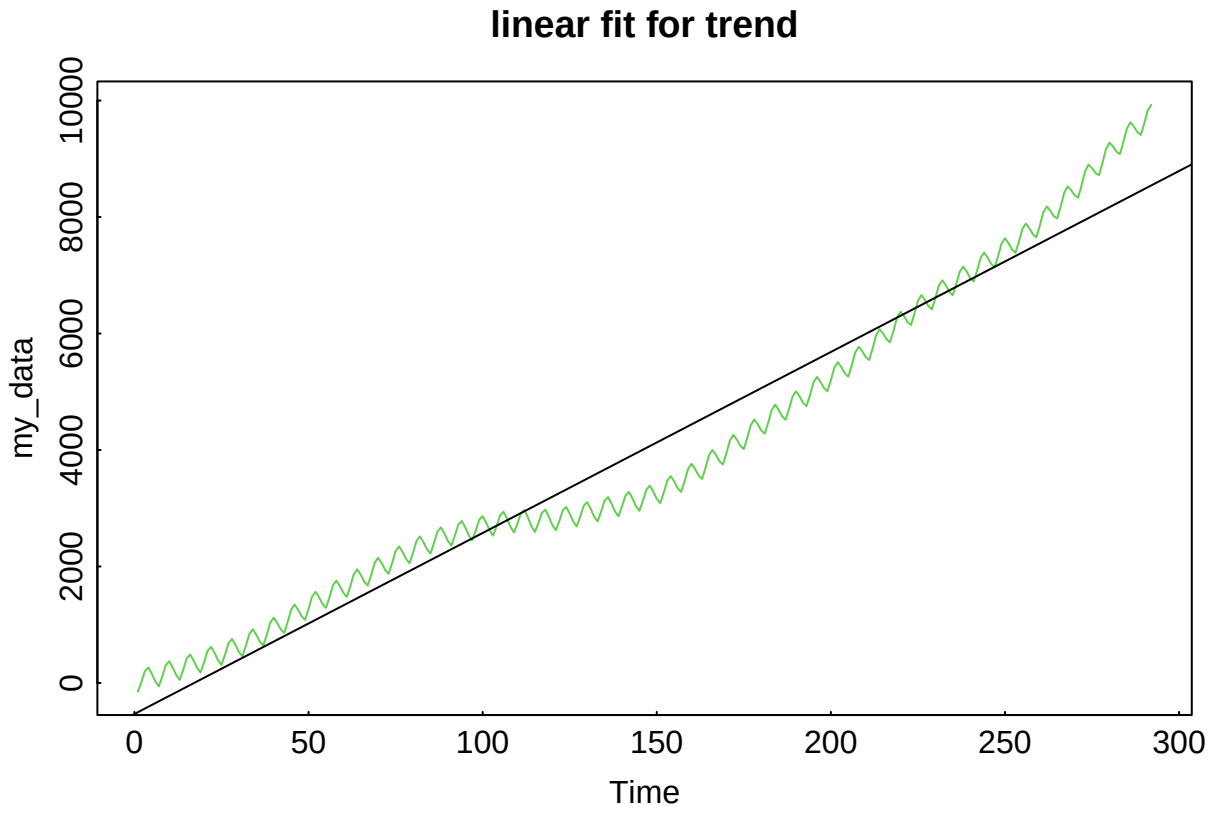
$$x_t = T_t + S_t + R_t \tag{1}$$

2.1.1 趋势效应的提取

对于趋势效应的估计，可以通过：

- 趋势拟合法：将时间看作自变量，将序列观察值看作因变量，建立起序列值-时间的回归模型，其中包含线性拟合 $x_t = a + bt + R_t$ 和曲线拟合。线性拟合可以通过函数 `lm(y~x1+x2+...+xn,data=)` 实现，而非线性回归则需要调用函数 `nls(y~f(x1,x2,...,xn),data=, start=)`。

```
t = seq(1,length(my_data))
opar = par(mar = c(3,3,3,1),mgp=c(1.5,0.2,0),tck=0.005)
plot(my_data,col=3,main="linear fit for trend")
abline(lm(my_data~t))
```



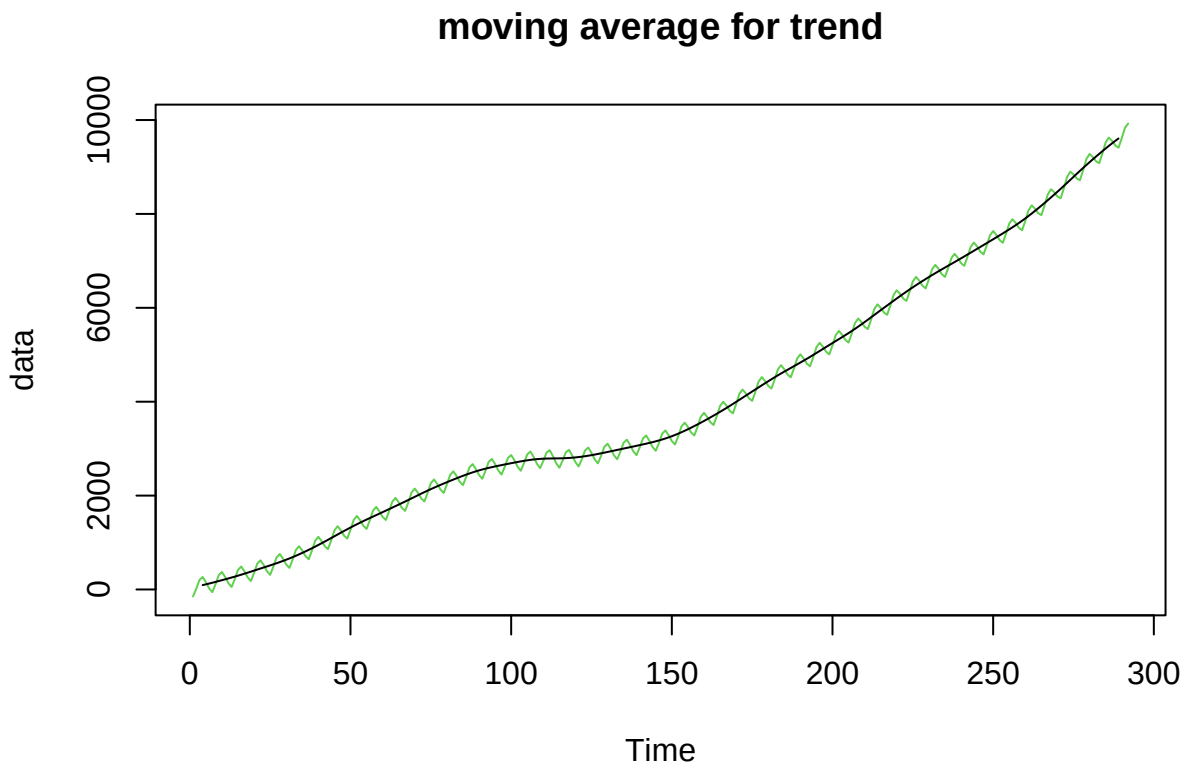

```
par(opar)
```

由于本数据中存在平台期，线性拟合的效果并不好，可以考虑采用非线性函数的方式拟合。

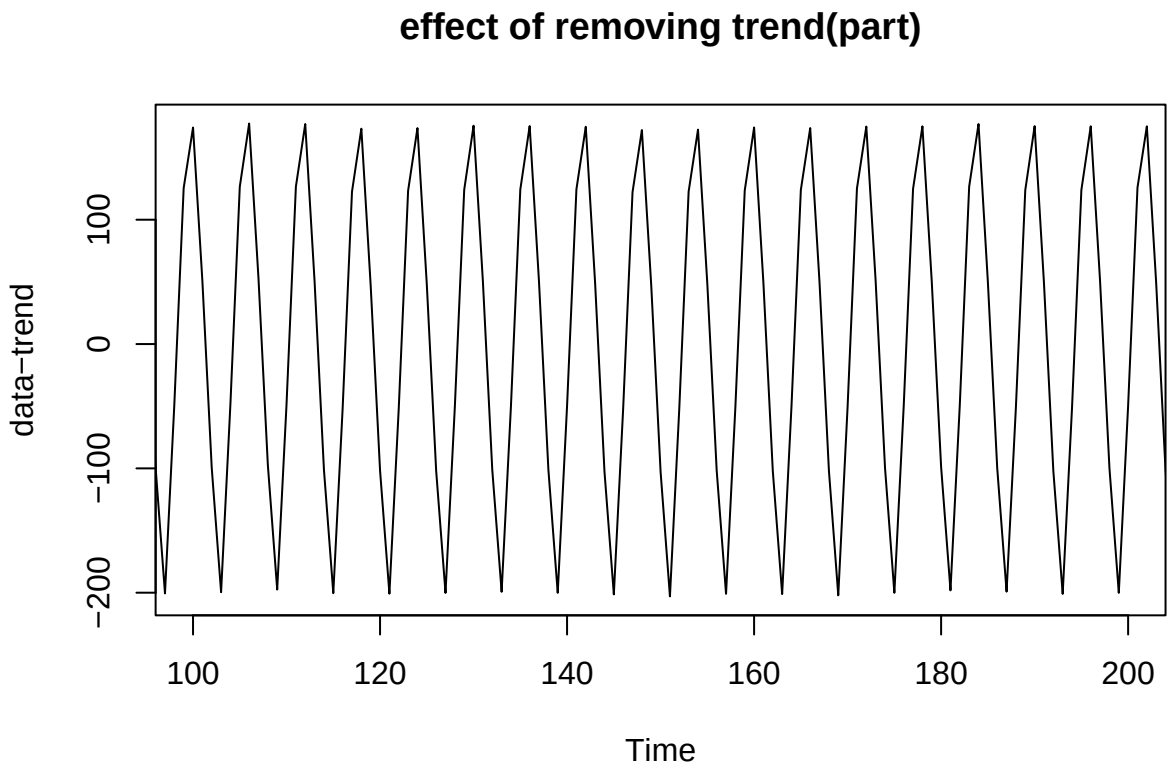
- 平滑法：采用移动平均或者指数平滑的方式削弱短期波动对序列的影响，从而显现出序列的变化趋势。其中移动平均法的平滑系数可以看作对最近 n 期数据的均匀加权，可直接通过函数 `filter` 实现；而指数平滑方法则考虑了近期变化的影响更大，因此定义随时间间隔增大而指数递减的权重，通过 `HolteWinters(data,alpha=,beta=,gamma=,seasonal=)` 实现。

由于本数据具有明显的季节波动性，此处基于滑动平均的思想，采用函数 `filter`, 提取序列的趋势效应。由于移动的平均期数 n 为偶数，故采用 2×4 复合移动平均实现 6 期简单中心移动平均

```
trend = filter(my_data/6,rep(1,6),sides=2)#6 期简单中心移动平均
trend_data = filter(trend/2,rep(1,2),sides=1) #2*6 复合移动平均
plot(my_data,lty=1,col=3,main="moving average for trend",ylab="data")
lines(trend_data) # 黑线为简单中心移动平均的拟合趋势效应
```

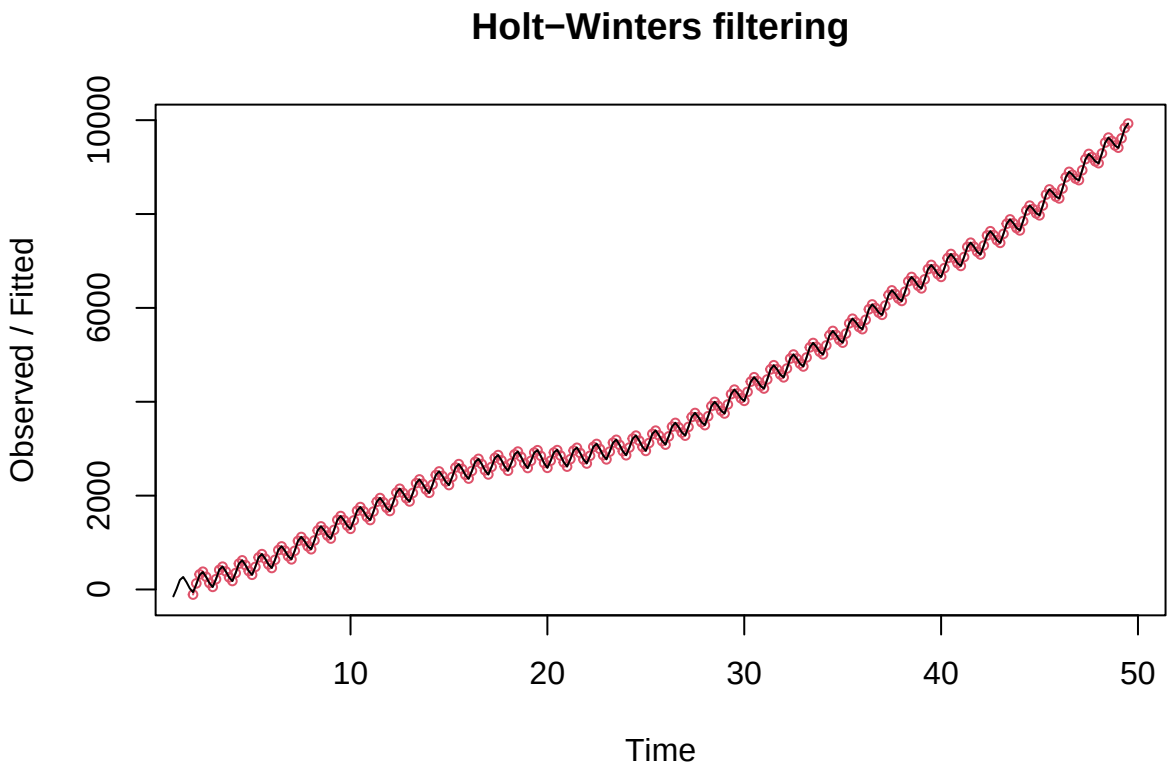



```
plot(my_data-trend_data,
     xlim=c(100,200),
     main="effect of removing trend（part）",
     ylab="data-trend")
```



如果采用指数平滑方法完成上述拟合, 则可细分为：+ 简单指数平滑法：平滑无季节变化、无趋势变化的时序观察值 + Holt 线性指数平滑法：处理拥有比较固定的线性趋势的数据 + Holt-Winters 指数平滑法：同时考虑了有线性趋势和季节变动的时间序列

```
plot(HoltWinters(ts(my_data,frequency=6)),type="o",cex=0.6)
```




```
# 以周期为 6 进行季节项拟合
```

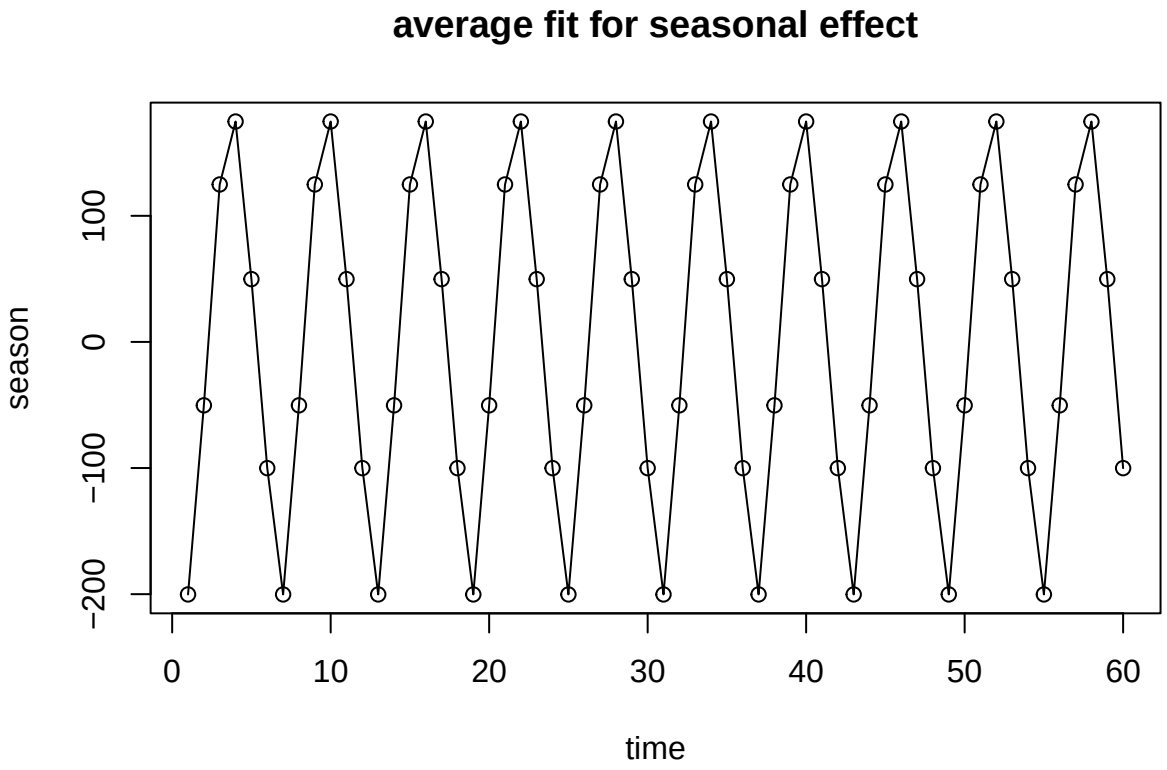
2.1.2 季节效应的提取

针对已经消除趋势效应的序列，加法模型假定每个季度的序列值等于均值加上季节效应，即

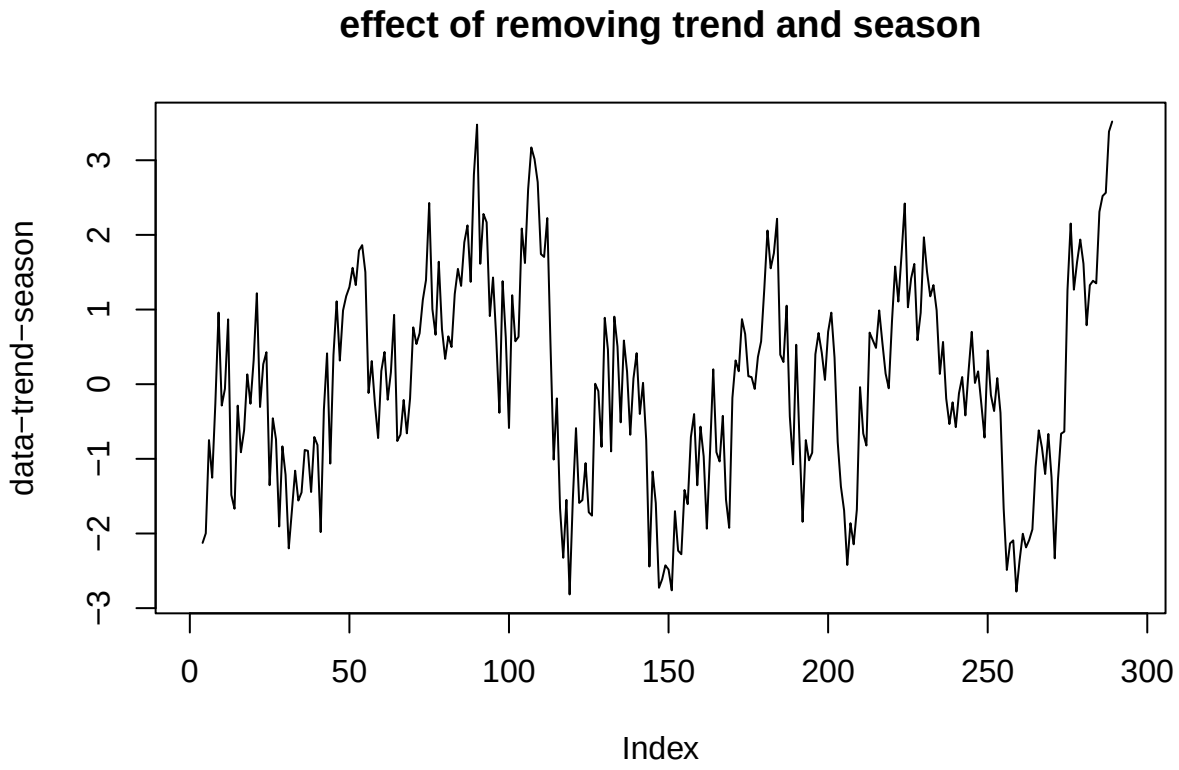
$$y_t = x_t - T_t = S_t + R_t$$

将第 i 个周期的第 j 个季节表示为 $y_{ij} = \overline{y} + S_j + R_{ij}$，满足 $\sum_j S_j = 0$

```
by_season = matrix(c(my_data-trend_data,NA,NA),ncol=6,byrow=TRUE)
season_data = apply(by_season, 2, mean,na.rm=T)# 各季节均值
plot(seq(1,10*6),replicate(10,season_data),
     main="average fit for seasonal effect",
     type = "o",
     ylab="season",xlab="time")
```

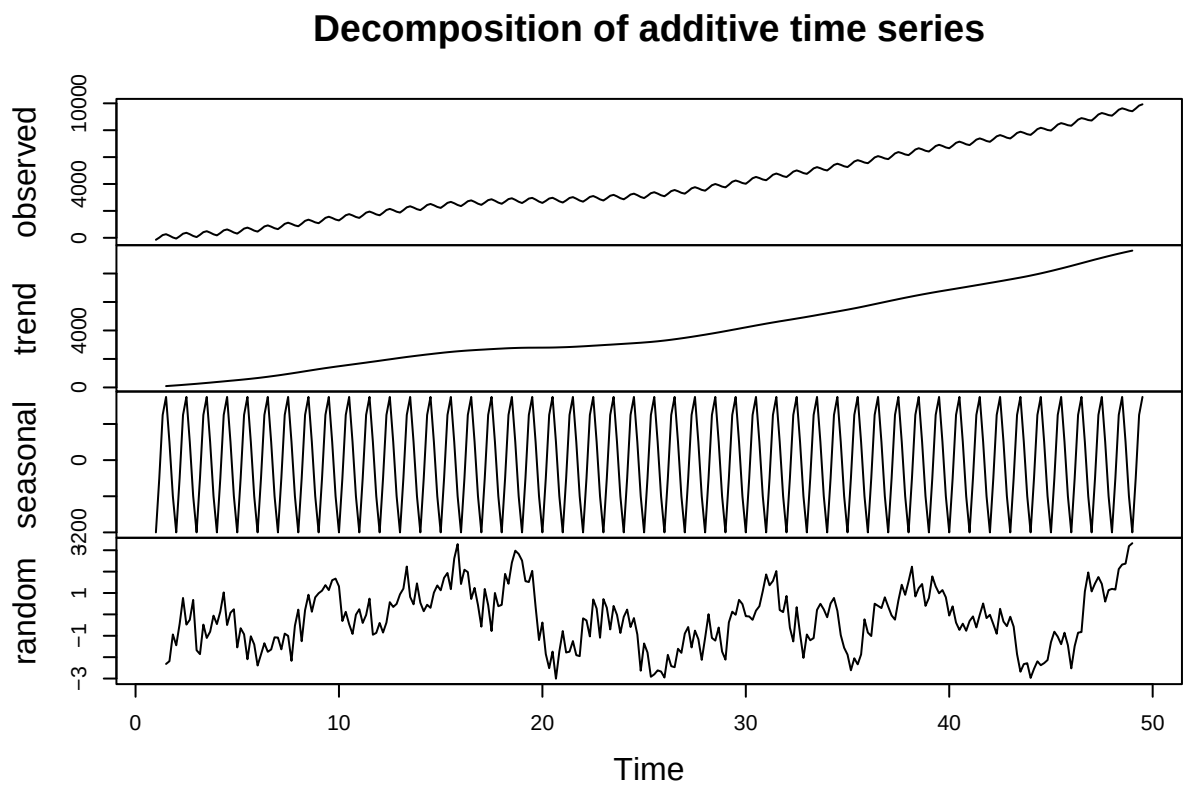



```
plot(c(my_data-trend_data,NA,NA)-season_data,
     main="effect of removing trend and season",
     ylab="data-trend-season",
     type="l")
```



当然，趋势项和季节项的剔除也可以直接使用 `decompose()` 函数（去除趋势项：中心对称滑动平均）或者 `stl()` 函数（去除季节项：loess 局部加权回归估计）实现：

```
r_data = decompose(ts(my_data,frequency=6))
plot(r_data)
```



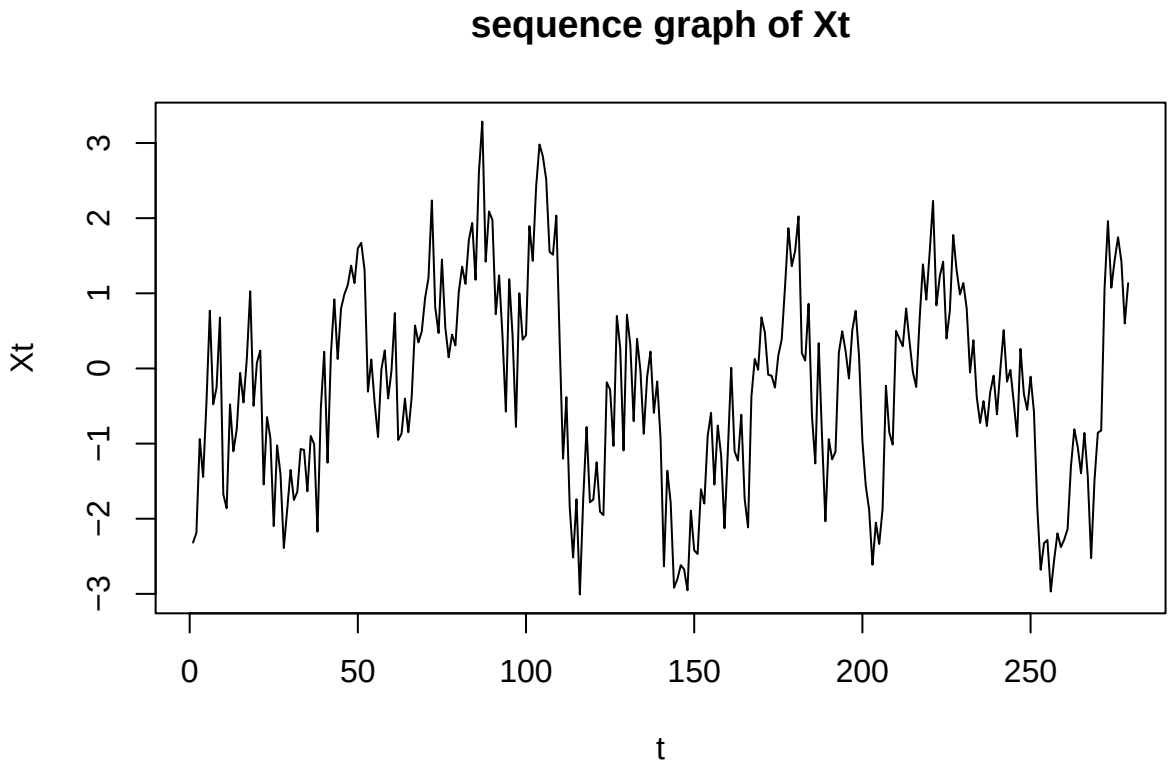
2.2 模型识别

2.2.1 平稳性检验

1. 图检验法

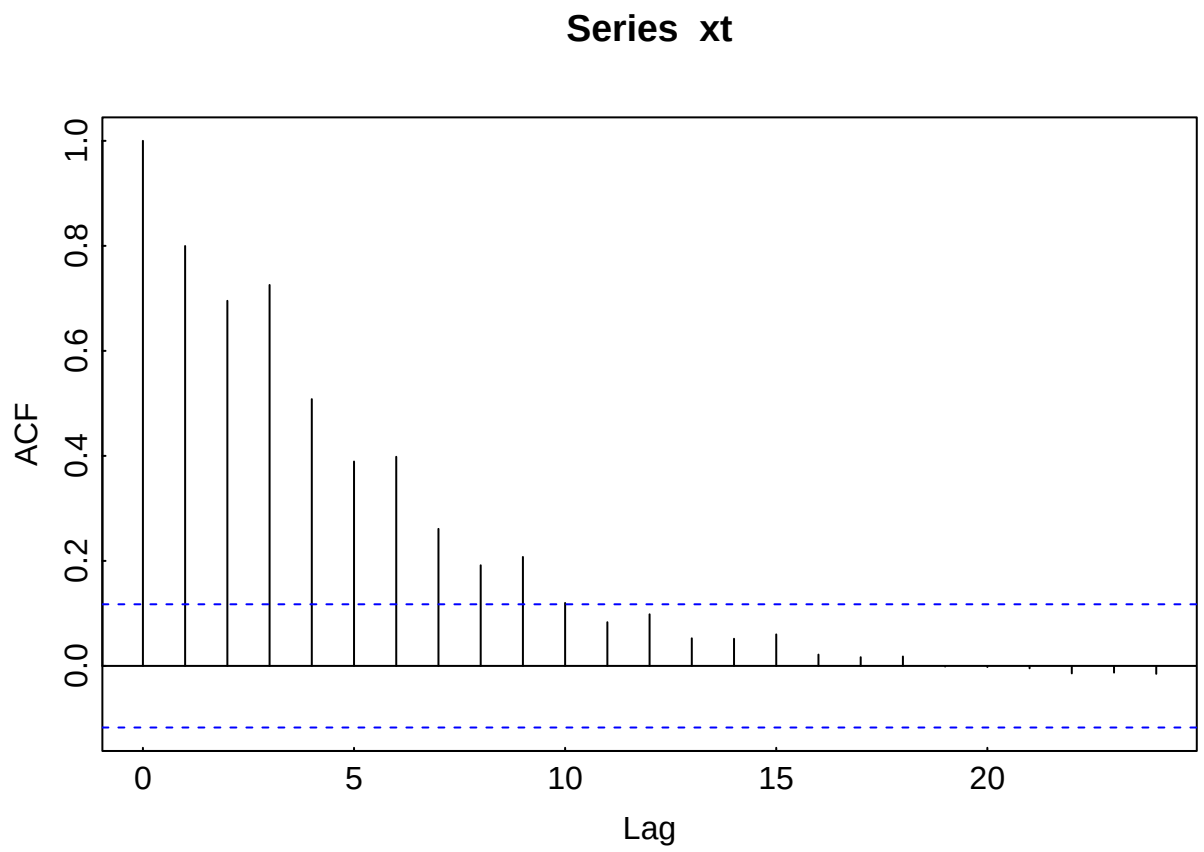
对通过 decompose() 函数得到的随机部分进行序列平稳性检验，首先绘制时序图进行观察

```
xt = r_data$random[4:282]
#decompose 采用滑动平均法，缺少前后 3 个值，去掉后 10 个值用于预测
plot(xt,
     main="sequence graph of Xt",
     ylab="Xt",
     xlab="t",
     type="l")
```



从时序图中可以认为不再有明显的趋势和周期性。由于平稳序列具有短期周期性，自相关系数会快速衰减到0，因此进一步通过自相关图进行检验

```
opar = par(mar = c(3,3,3,1),mgp=c(1.5,0.2,0),tck=0.005)
acf(xt)
```




```
par(opar)
```

相关系数延迟到 10 阶以后落在了两倍标准差之内，可认为序列具有平稳性

2. 统计检验法利用平稳序列的性质：序列的特征根应该在单位圆内，构造出单位根检验：

- 零假设：序列不平稳 ↔ 备择假设：序列平稳
- 检验统计量：比较基础的是 DF 检验 (适用于确定性部分有上一期历史信息决定的 AR(1) 模型的平稳性检验)，应用较广的为 ADF 检验（适用于任意 p 期确定信息的提取）
- 检验方法：可通过 aTSA 包中的 `adf.test` 函数实现

```
library(aTSA)
```

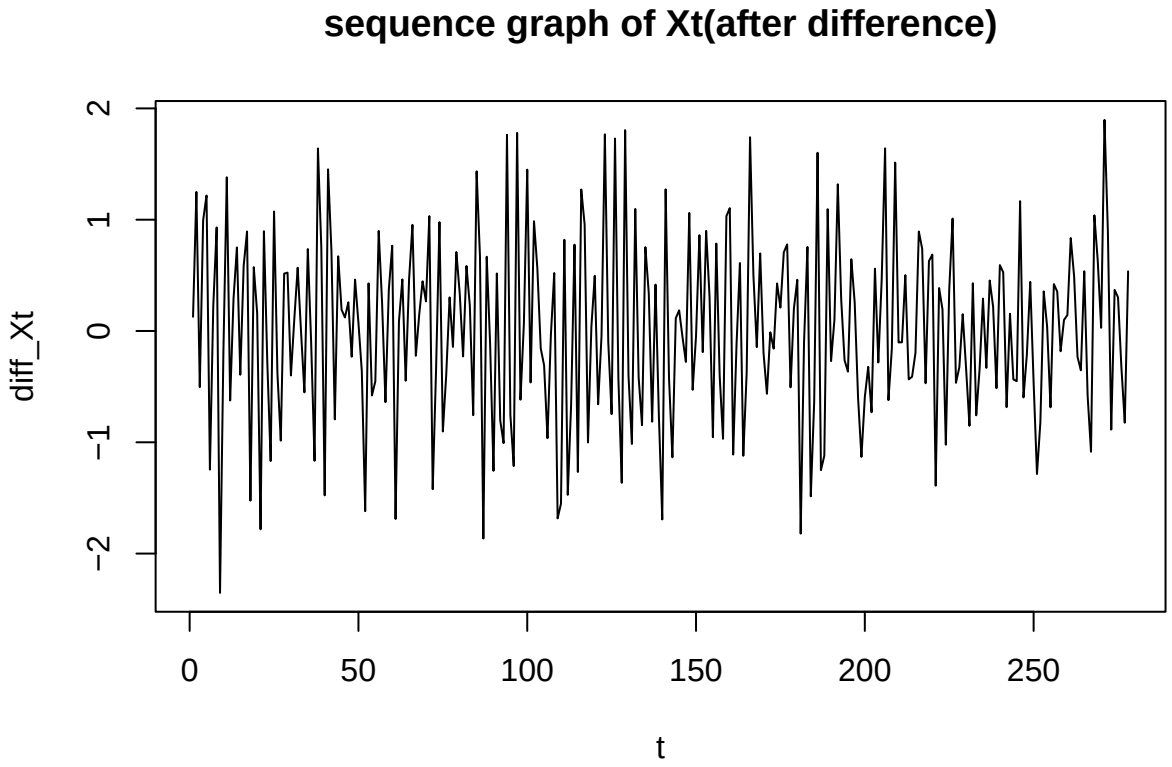
```
##
## 载入程辑包: 'aTSA'

## The following object is masked from 'package:graphics':
##
##     identify
```

```
adf.test(xt)# 不指定 nlag, 系统自动指定延迟阶数
```

```
## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag   ADF p.value
## [1,]   0 -5.48    0.01
## [2,]   1 -4.47    0.01
## [3,]   2 -2.84    0.01
## [4,]   3 -5.31    0.01
## [5,]   4 -4.22    0.01
## [6,]   5 -4.25    0.01
## Type 2: with drift no trend
##      lag   ADF p.value
## [1,]   0 -5.53  0.0100
## [2,]   1 -4.50  0.0100
## [3,]   2 -2.81  0.0609
## [4,]   3 -5.40  0.0100
## [5,]   4 -4.30  0.0100
## [6,]   5 -4.35  0.0100
## Type 3: with drift and trend
##      lag   ADF p.value
## [1,]   0 -5.54   0.010
## [2,]   1 -4.51   0.010
## [3,]   2 -2.82   0.232
## [4,]   3 -5.42   0.010
## [5,]   4 -4.29   0.010
## [6,]   5 -4.33   0.010
## ----
## Note: in fact, p.value = 0.01 means p.value <= 0.01
```

可以发现类型 1 的 τ 统计量的 P 值均小于显著性水平 $\alpha = 0.05$，因此可以认为减去趋势项和季节项之后的时间序列平稳。但类型 2 和类型 3 的部分 τ 统计量的 P 值较大，说明 `drift` 效应不能忽视，因此对于随机的趋势效应，考虑通过差分的方式去除：

```
diff_xt = diff(xt)
plot(diff_xt,
     main="sequence graph of Xt(after difference)",
     ylab="diff_Xt",
     xlab="t",
     type="l")
```




```
adf.test(diff_xt)
```

```
## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag    ADF p.value
## [1,]   0 -21.26    0.01
## [2,]   1 -20.61    0.01
## [3,]   2  -7.06    0.01
## [4,]   3  -8.27    0.01
## [5,]   4  -7.54    0.01
## [6,]   5  -8.35    0.01
## Type 2: with drift no trend
##      lag    ADF p.value
## [1,]   0 -21.23    0.01
## [2,]   1 -20.58    0.01
## [3,]   2  -7.04    0.01
## [4,]   3  -8.25    0.01
```

```
## [5,]   4  -7.53    0.01
## [6,]   5  -8.33    0.01
## Type 3: with drift and trend
##      lag    ADF p.value
## [1,]   0 -21.19    0.01
## [2,]   1 -20.54    0.01
## [3,]   2  -7.03    0.01
## [4,]   3  -8.24    0.01
## [5,]   4  -7.52    0.01
## [6,]   5  -8.32    0.01
## ----
## Note: in fact, p.value = 0.01 means p.value <= 0.01
```

一次差分之后的序列对于 3 种类型的检验统计量 τ 都小于 0.01，认为差分后的序列更满足平稳性。

2.2.2 纯随机性检验

通过前述可知，我们只关心序列值之间有相关性的序列，因此需要对平稳序列进行纯随机性检验，排除没有分析价值的序列。

- 零假设：延迟阶数 $\leq m$ 期序列值之间相互独立 $\leftrightarrow$ 备择假设：延迟阶数 $\leq m$ 期序列值之间有相关性
- 检验统计量: Q 统计量 (Q_{BP}: Box-Pierce,Q_{LB}: Ljung-Box)
- 检验方法：`Box.test()` 函数

```
Box.test(diff_xt,type = "Ljung-Box",lag = 6)# 延迟阶数 m=6, 利用平稳序列的短期相关性
```

```
##
##  Box-Ljung test
##
## data:  diff_xt
## X-squared = 242.47, df = 6, p-value < 2.2e-16
```

```
Box.test(diff_xt,type = "Ljung-Box",lag = 12)
```

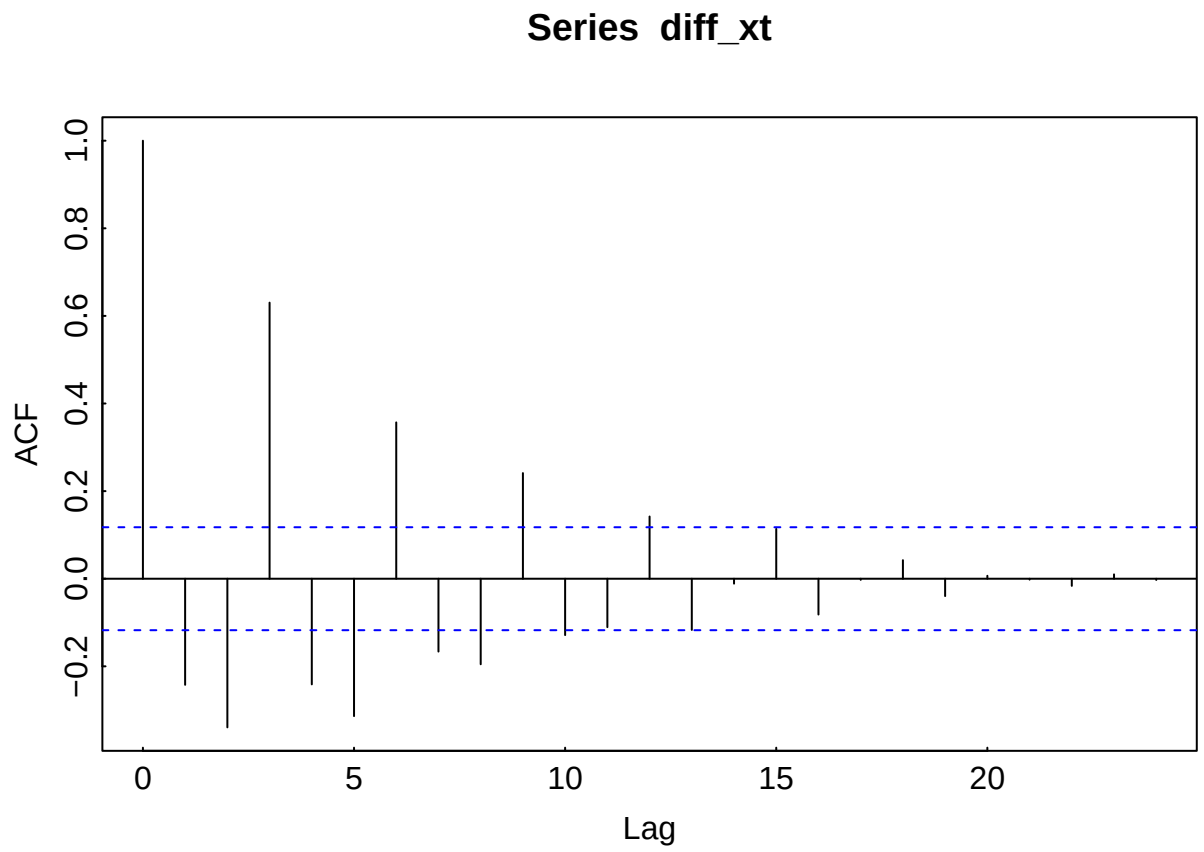
```
##
##  Box-Ljung test
##
## data:  diff_xt
## X-squared = 292.53, df = 12, p-value < 2.2e-16
```

由于 6 阶和 12 阶的 LB 统计量的 P 值显著小于 $\alpha = 0.05$，因此可以拒绝纯随机性的原假设，认为该序列为平稳非白噪声序列。

2.2.3 模型定阶

利用自相关函数和偏自相关函数的截尾性初步定阶:

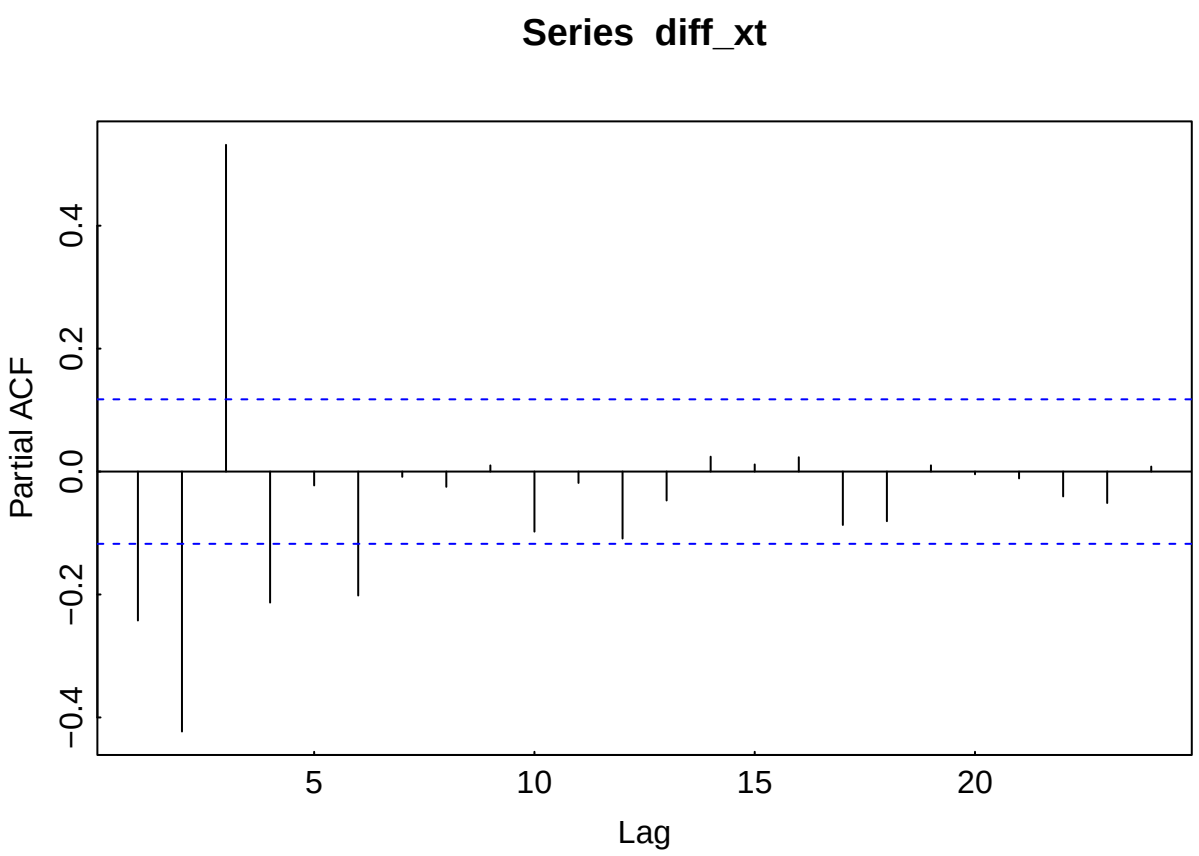
```
opar = par(mar = c(3,3,3,1),mgp=c(1.5,0.2,0),tck=0.005)
acf(diff_xt)
```




```
par(opar)
```

ACF 图具有拖尾的特征（约在 12 期左右落在了两倍标准差之内），再分析 PACF 图：

```
opar = par(mar = c(3,3,3,1),mgp=c(1.5,0.2,0),tck=0.005)
pacf(diff_xt)
```




```
par(opar)
```

偏自相关函数具有 6 阶截尾的特征，则将拟合 diff_xt 的模型定阶为 ARIMA(6,0,0)

2.3 模型估计

- 矩估计：对于 (p+q) 个样本自相关系数的估计，可以通过构造 p+q 个方程组成的 Y-W 方程组通过解方程得到参数值的估计
- 极大似然估计：在序列服从多元正态分布的假设下，最大化似然函数得到参数的估计，往往需要迭代
- 最小二乘估计：最小化残差平方和得到的估计

对 ARMA 模型的参数估计可以通过调用 `arima()` 函数完成：

```
fit1 = arima(diff_xt,order = c(6,0,0),method = "CSS-ML")
# 默认 method 即为最小二乘与极大似然混合法
fit1
```

```
##
## Call:
## arima(x = diff_xt, order = c(6, 0, 0), method = "CSS-ML")
```

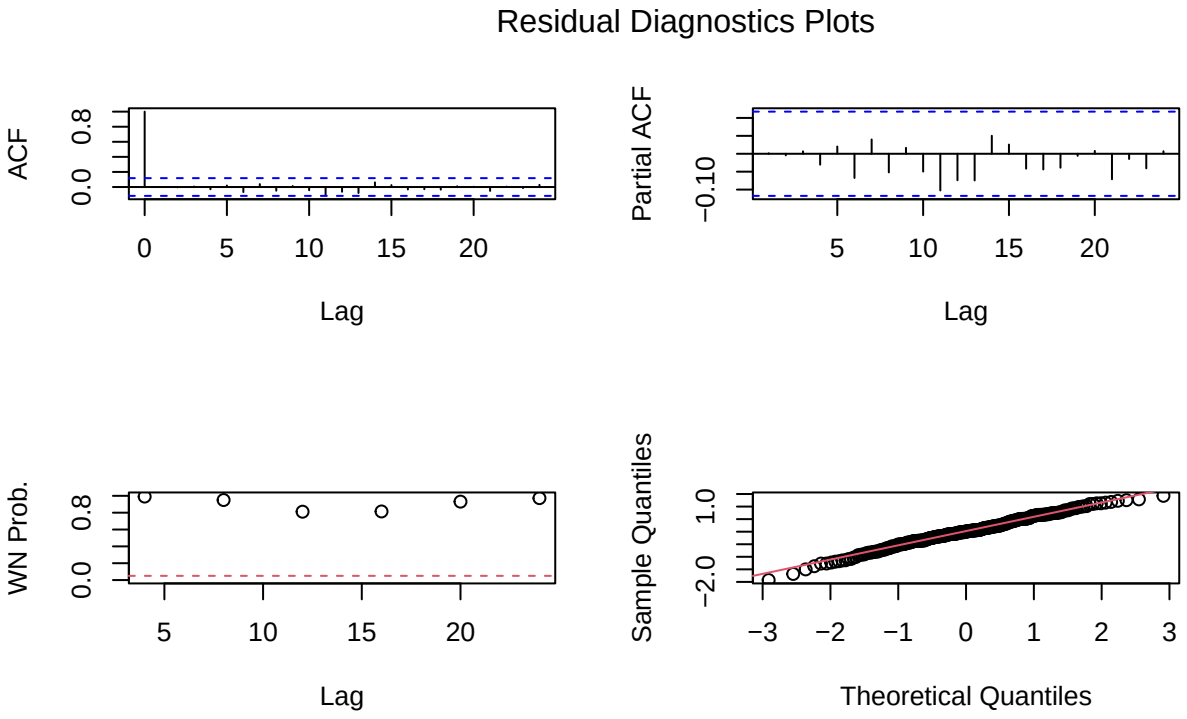
```
##
## Coefficients:
##           ar1      ar2     ar3      ar4      ar5      ar6  intercept
##       -0.0020  -0.3370  0.6103  -0.2879  -0.0145  -0.2104     0.0098
## s.e.   0.0587   0.0591  0.0598   0.0600   0.0592   0.0593     0.0281
##
## sigma^2 estimated as 0.3365:  log likelihood = -244.06,  aic = 504.11
```

2.4 模型检验

检验包括模型的显著性检验和参数的显著性检验；模型的显著性检验是检验所拟合的模型是否能充分地提取原序列中的样本相关信息，也就是检验拟合的残差中是否为白噪声序列。而参数的显著性检验则是判断某些参数是否显著不为 0，从而删去冗余的参数，使得模型尽可能的精简。

- 原假设：残差序列为白噪声（模型拟合有效）
- 检验统计量：LB 统计量
- 检验方法：`ts.diag()` 函数或 `Box.test()` 函数

```
ts.diag(fit1)
```



```
Box.test(fit1$residuals)
```

```
##
##  Box-Pierce test
##
## data:  fit1$residuals
## X-squared = 0.00052044, df = 1, p-value = 0.9818
```

```
t = abs(fit1$coef)/sqrt(diag(fit1$var.coef))
pt(t,length(diff_xt)-length(fit1$coef),lower.tail = F)
```

```
##          ar1          ar2          ar3          ar4          ar5          ar6
## 4.865177e-01 1.529068e-08 3.399660e-21 1.308308e-06 4.033923e-01 2.295952e-04
##    intercept
## 3.634474e-01
```

- 通过 Box.test() 检验得到的 p 值为 0.9818，可以通过模型显著性检验
- 但 ϕ_1, ϕ_5 并不显著，因此可能是拟合阶数过高，下面考虑对模型进行优化

2.5 模型优化

有效的模型不一定是唯一的，基于信息准则可以从多个有效模型中选取拟合地较好地模型。

考虑采用 R 中的自动定阶函数 `auto.arima()`，该函数基于信息量的最小原则，在一定范围内自动识别出最优模型的阶数并给出相应的参数估计：

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

##
## 载入程辑包: 'forecast'

## The following object is masked from 'package:aTSA':
##
##     forecast
```

```
auto.arima(diff_xt) #AIC 准则：输出结果 arima(4,0,3)
```

```
## Series: diff_xt
## ARIMA(4,0,3) with zero mean
##
## Coefficients:
##          ar1      ar2     ar3      ar4      ma1     ma2     ma3
##       0.1344  -0.4656  0.2241  -0.4283  -0.1212  0.1177  0.4811
## s.e.  0.1554   0.1275  0.1336   0.0961   0.1478  0.1191  0.1215
##
## sigma^2 = 0.3437:  log likelihood = -243.5
## AIC=502.99   AICc=503.53   BIC=532.01
```

```
auto.arima(diff_xt,ic="bic") #BIC 准则：输出结果 arima(4,0,0)
```

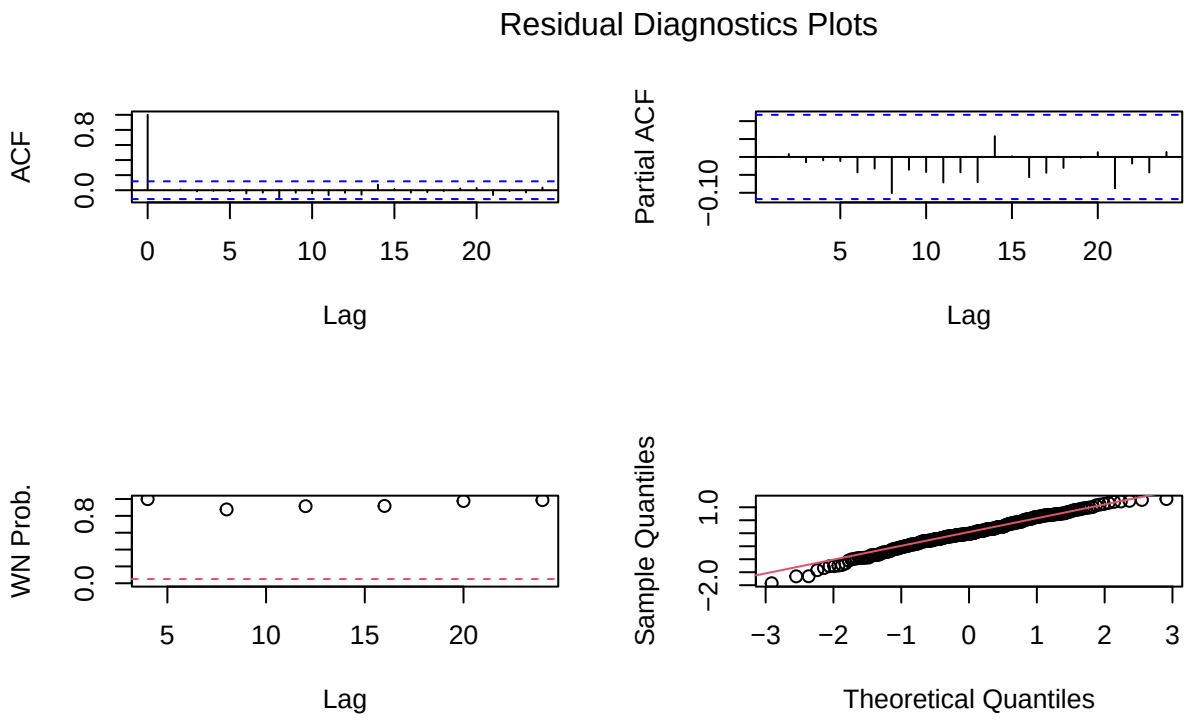
```
## Series: diff_xt
## ARIMA(4,0,0) with zero mean
##
## Coefficients:
##          ar1     ar2     ar3      ar4
##       0.0048  -0.296  0.5091  -0.2268
## s.e.  0.0584   0.050  0.0496   0.0588
##
## sigma^2 = 0.3574:  log likelihood = -250.27
## AIC=510.55   AICc=510.77   BIC=528.68
```

```
auto.arima(diff_xt,ic="aicc") #AICC 准则：输出结果 arima(4,0,3)
```

```
## Series: diff_xt
## ARIMA(4,0,3) with zero mean
##
## Coefficients:
##          ar1      ar2     ar3      ar4      ma1     ma2     ma3
##       0.1344  -0.4656  0.2241  -0.4283  -0.1212  0.1177  0.4811
## s.e.  0.1554   0.1275  0.1336   0.0961   0.1478  0.1191  0.1215
##
## sigma^2 = 0.3437:  log likelihood = -243.5
## AIC=502.99   AICc=503.53   BIC=532.01
```

考虑模型 ARIMA(4,0,3)

```
fit2 = arima(diff_xt,order = c(4,0,3))
ts.diag(fit2)
```




```
fit2
```

```
##
## Call:
## arima(x = diff_xt, order = c(4, 0, 3))
##
## Coefficients:
##          ar1      ar2     ar3      ar4      ma1     ma2     ma3  intercept
##       0.1344  -0.4661  0.2236  -0.4288  -0.1215  0.1179  0.4814     0.0108
## s.e.  0.1555   0.1278  0.1339   0.0960   0.1478  0.1194  0.1216     0.0334
##
## sigma^2 estimated as 0.3349:  log likelihood = -243.44,  aic = 504.89
```

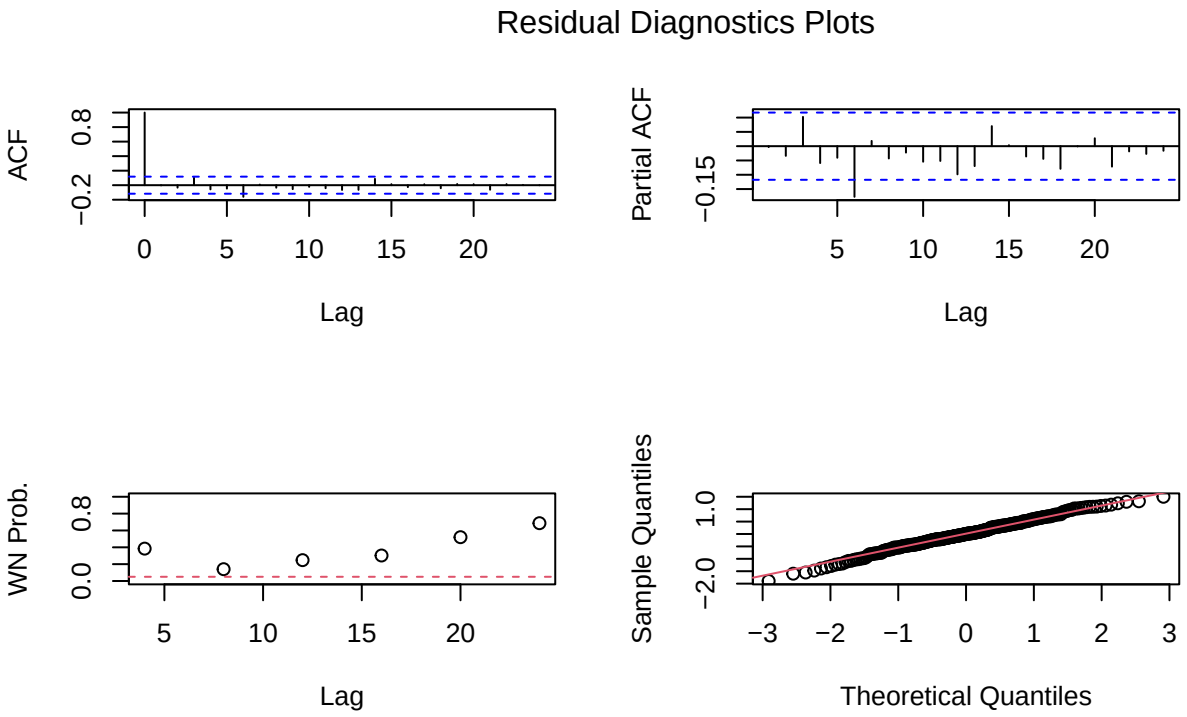
```
t = abs(fit2$coef)/sqrt(diag(fit2$var.coef))
pt(t,length(diff_xt)-length(fit2$coef),lower.tail = F)
```

```
##          ar1          ar2          ar3          ar4          ma1          ma2
## 1.941464e-01 1.592461e-04 4.805360e-02 5.867775e-06 2.058286e-01 1.622257e-01
##          ma3    intercept
## 4.836273e-05 3.738229e-01
```

- 可以通过模型的显著性检验
- 对于参数的显著性检验：$\phi_1, \phi_3, \theta_1, \theta_2$ 和截距项的 P 值都大于 0.05，说明这几个参数并不是显著非 0

可能基于 AIC 准则系统推荐的最优模型阶数偏高，再考虑 BIC 准则得到的模型 ARIMA(4,0,0)

```
fit3 = arima(diff_xt,order = c(4,0,0))
ts.diag(fit3)
```




```
fit3
```

```
##
## Call:
## arima(x = diff_xt, order = c(4, 0, 0))
##
## Coefficients:
##          ar1      ar2     ar3      ar4  intercept
##       0.0045  -0.2964  0.5087  -0.2269     0.0105
## s.e.  0.0584   0.0500  0.0496   0.0588     0.0352
##
## sigma^2 estimated as 0.3521:  log likelihood = -250.23,  aic = 512.46
```

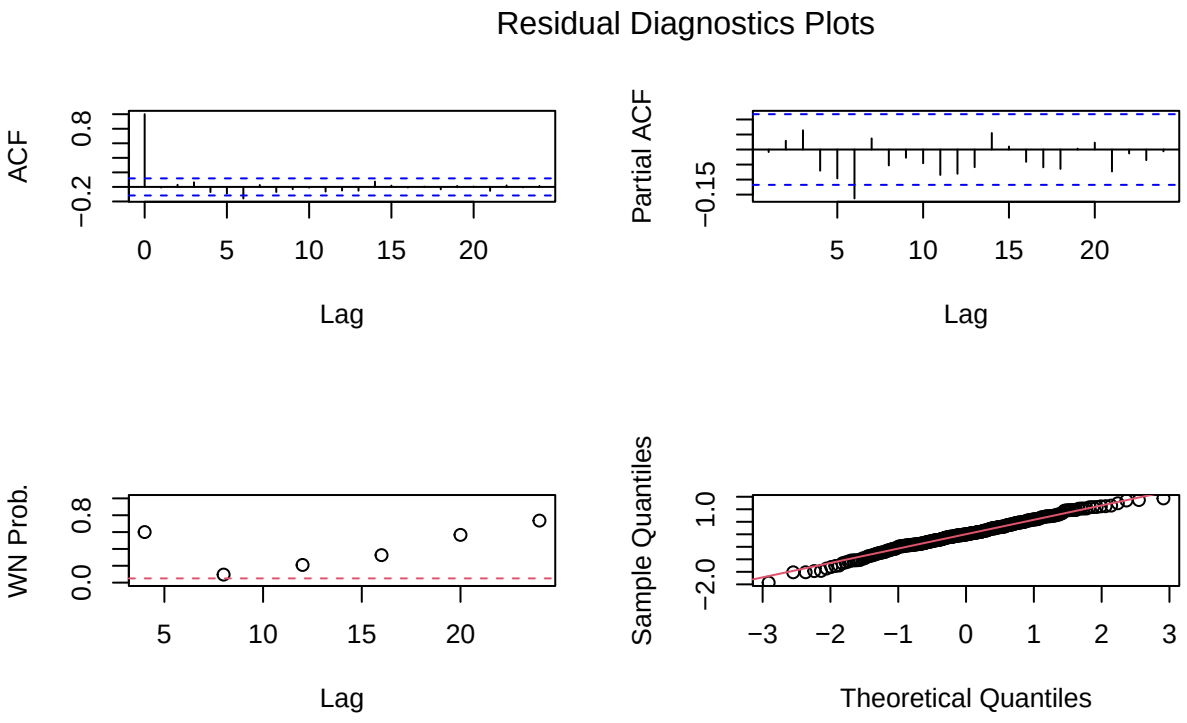
```
t = abs(fit3$coef)/sqrt(diag(fit3$var.coef))
pt(t,length(diff_xt)-length(fit3$coef),lower.tail = F)
```

```
##          ar1          ar2          ar3          ar4    intercept
## 4.693157e-01 4.711142e-09 2.210672e-21 7.087341e-05 3.830163e-01
```

- 可以通过模型的显著性检验
- 部分参数的显著性依然表现的不是很好

由此我们可以发现，直接利用基于信息量 `auto.arima()` 函数进行定阶和参数的选取有时候会出现模型或者不显著的情况，最后还是通过暴力的手动试参数的方法找到了一个有效、系数显著不为 0、且满足较小信息准则的模型 ARIMA(3,0,1)，用于拟合做过时间差分的序列 diff_xt

```
fit4 = arima(diff_xt,order = c(3,0,1))
ts.diag(fit4)
```




```
fit4
```

```
##
## Call:
## arima(x = diff_xt, order = c(3, 0, 1))
```

```
##
## Coefficients:
##           ar1      ar2     ar3     ma1  intercept
##       -0.4329  -0.3525  0.3949  0.4650     0.0112
## s.e.   0.0989   0.0659  0.0759  0.1093     0.0374
##
## sigma^2 estimated as 0.3505:  log likelihood = -249.59,  aic = 511.18
```

```
Box.test(fit4$residuals)
```

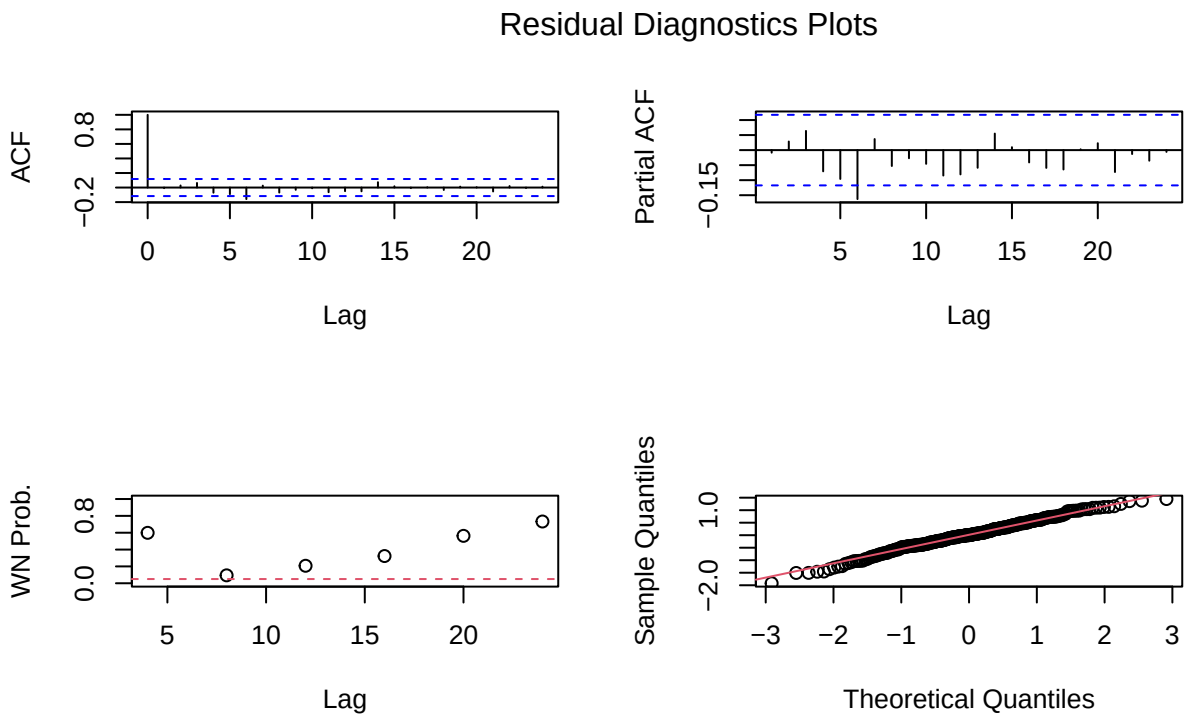
```
##
##  Box-Pierce test
##
## data:  fit4$residuals
## X-squared = 0.018992, df = 1, p-value = 0.8904
```

```
t = abs(fit4$coef)/sqrt(diag(fit4$var.coef))
pt(t,length(xt)-length(fit4$coef),lower.tail = F)
```

```
##          ar1          ar2          ar3          ma1    intercept
## 8.605947e-06 9.433023e-08 1.919711e-07 1.445422e-05 3.821013e-01
```

但同样的也注意到，理论上如果用 ARIMA(3,1,1) 去拟合未做差分的序列效果是等价的。

```
fit5 = arima(xt,order=c(3,1,1))
ts.diag(fit5)
```




```
fit5
```

```
##
## Call:
## arima(x = xt, order = c(3, 1, 1))
##
## Coefficients:
##           ar1      ar2     ar3     ma1
##       -0.4321  -0.3519  0.3955  0.4645
## s.e.   0.0989   0.0659  0.0758  0.1093
##
## sigma^2 estimated as 0.3506:  log likelihood = -249.63,  aic = 509.27
```

```
t = abs(fit5$coef)/sqrt(diag(fit5$var.coef))
pt(t,length(xt)-length(fit5$coef),lower.tail = F)
```

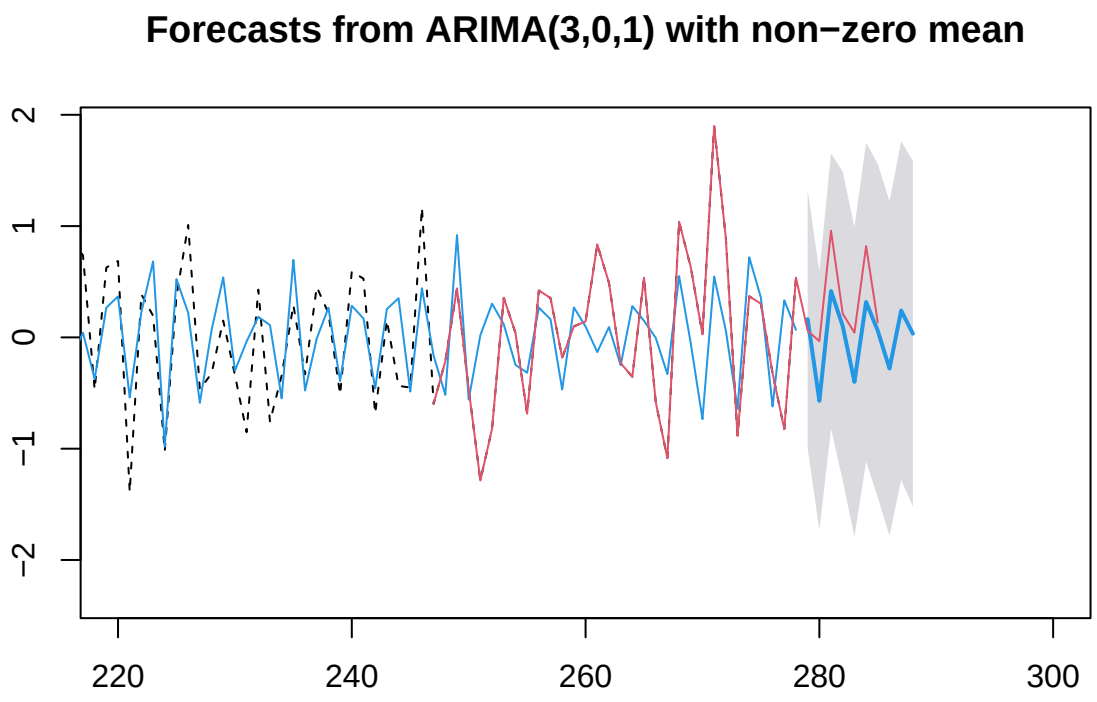
```
##          ar1          ar2          ar3          ma1
## 8.815007e-06 9.642818e-08 1.796173e-07 1.474215e-05
```

3 序列预测

3.1 残差预测

得到了较好的拟合模型后，下一步即对最后手动试出来的模型进行预测

```
fore1 = forecast::forecast(fit4,h=10,level=0.95)
# 黑线为构造模型所用的时间序列（删去了后 10 个值
plot(fore1,lty=2,xlim=c(220,300))
# 蓝线为拟合值
lines(fore1$fitted,col=4)
# 红线为序列真实值
lines(seq(247,286),diff(r_data$random)[250:289],col=2)
```

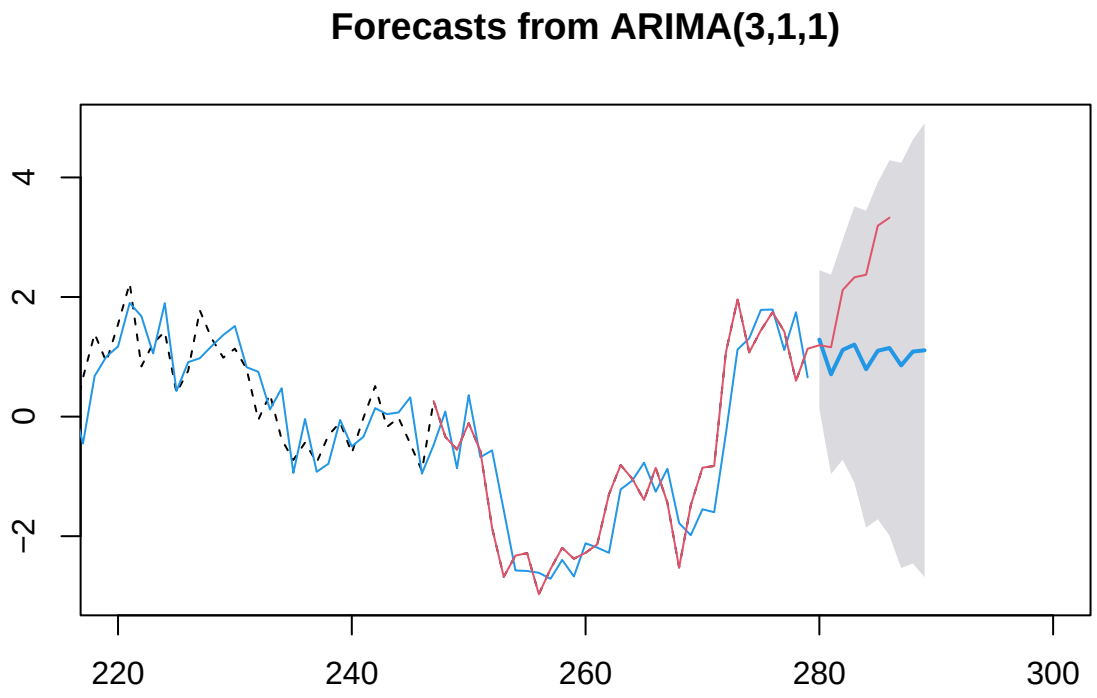


模型

的预测值尚可，预测值略有滞后，但是和真实值相差不大。

但如果直接对未做差分的序列进行拟合

```
fore2 = forecast::forecast(fit5,h=10,level=0.95)
plot(fore2,lty=2,xlim=c(220,300))
lines(fore2$fitted,col=4)
lines(seq(247,286),r_data$random[250:289],col=2)
```

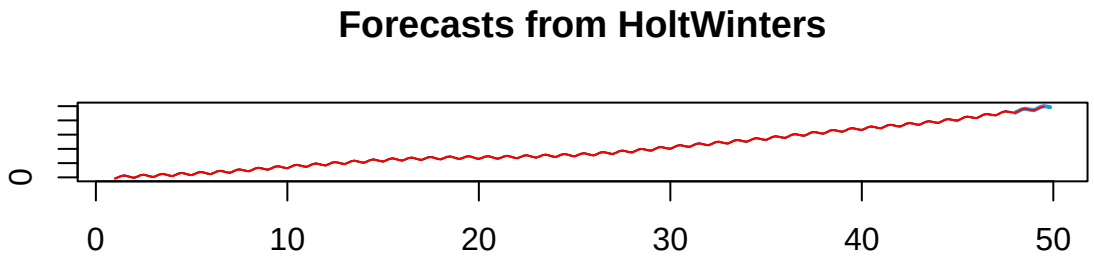


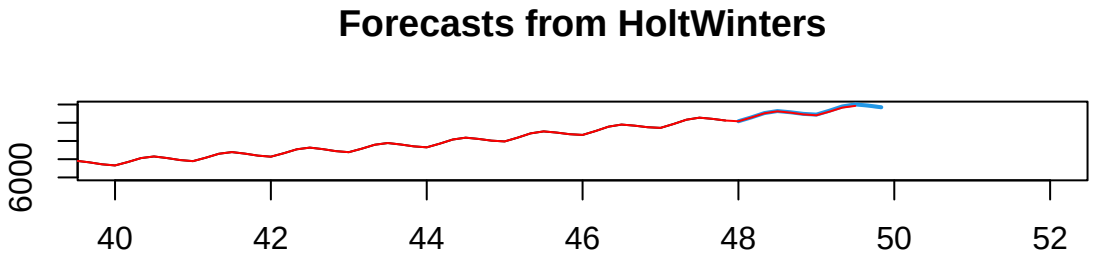
可以看出尽管真实值落在了 95% 的置信区间之内，但是和预测值（蓝线）相差较大，因此尽管 ARIMA(3,1,1) 和对差分后序列做 ARIMA(3,0,1) 拟合从理论上来说是等价的，但对差分序列预测较好并不能推出对原本序列拟合的好坏（差分后去掉了随机性趋势的影响，无法还原为有 drift 效应的原时间序列）。

3.2 整体预测

指数模型是用于预测时间序列未来值的最常用的模型，由上述拟合过程知我们此处可以采用三指数模型（也就是 Holt-Winters exponential smoothing) 拟合水平项、趋势项以及季节效应的时序，再通过 forecast 包对此 holtwinter 拟合得到的模型进行预测

```
fit_whole = HoltWinters(ts(my_data[1:(292-10)],frequency = 6))
fore3 = forecast(fit_whole)
par(mfrow=c(2,1))
# 整体图
plot(fore3)
lines(ts(my_data,frequency = 6),col="red")
# 局部图
plot(fore3,xlim=c(40,52),ylim=c(6000,10000)) # 蓝线表示预测值
lines(ts(my_data,frequency = 6),col="red") # 红线表示真实值
```

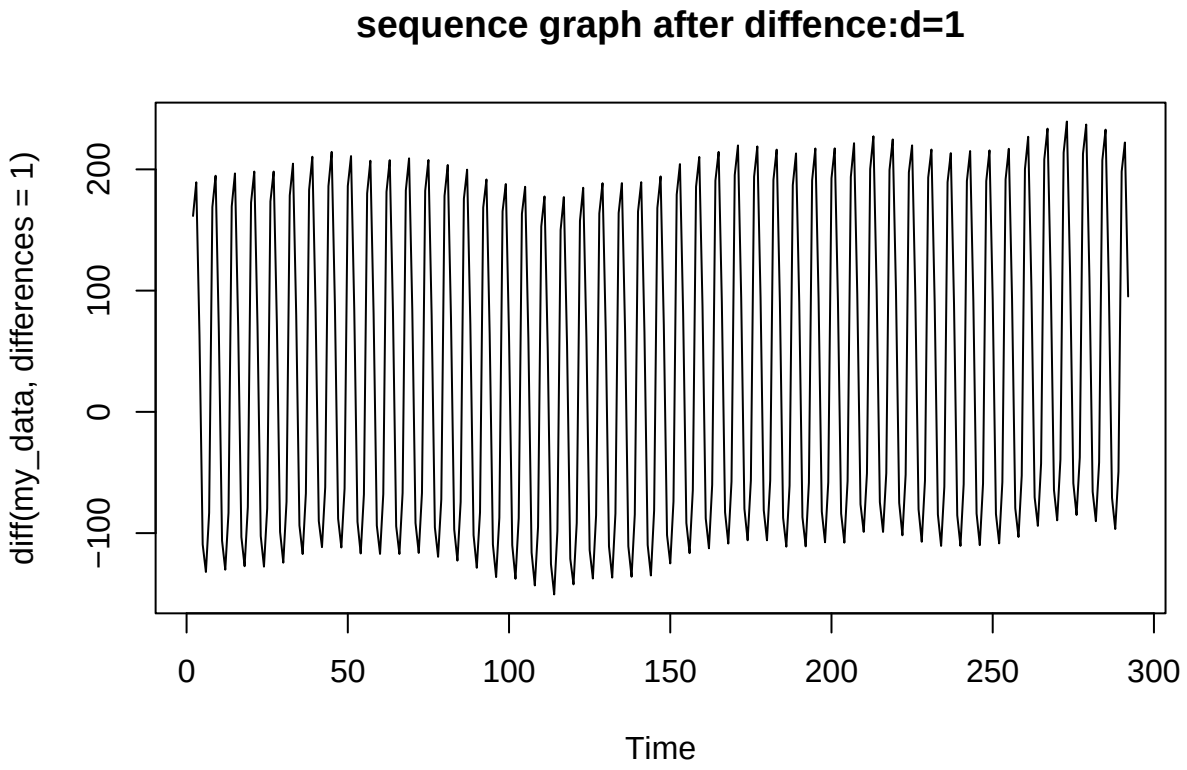




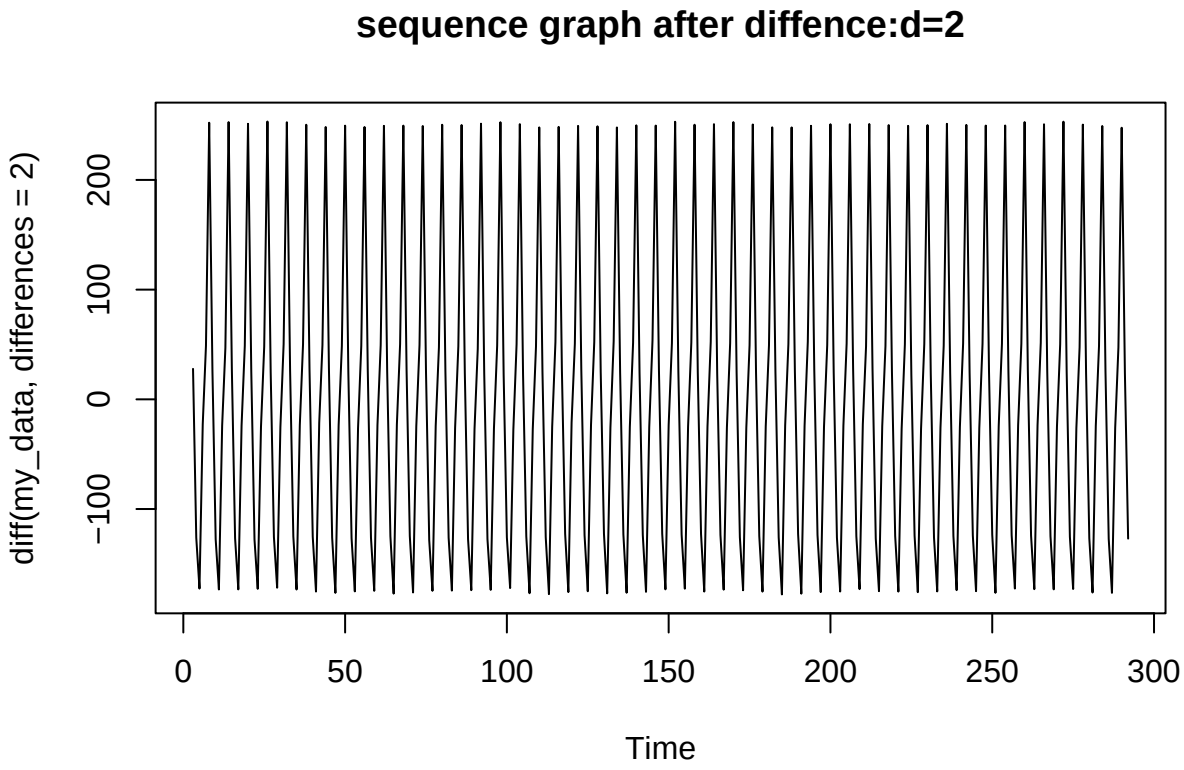
4 SARIMA 模型

通过上述分析可知，通过滑动平均减去确定性趋势项、减去周期项、差分去除随机性趋势项、再拟合 ARMA 模型未免有些复杂和冗余，因此考虑直接用 $SARIMA(p,d,q)\times(P,D,Q)_s$ 来对数据进行拟合

```
#par(mfrow=c(1,1))
#plot(my_data,main="sequence graph of raw data")
plot(diff(my_data,differences = 1),main="sequence graph after diffence:d=1")
```

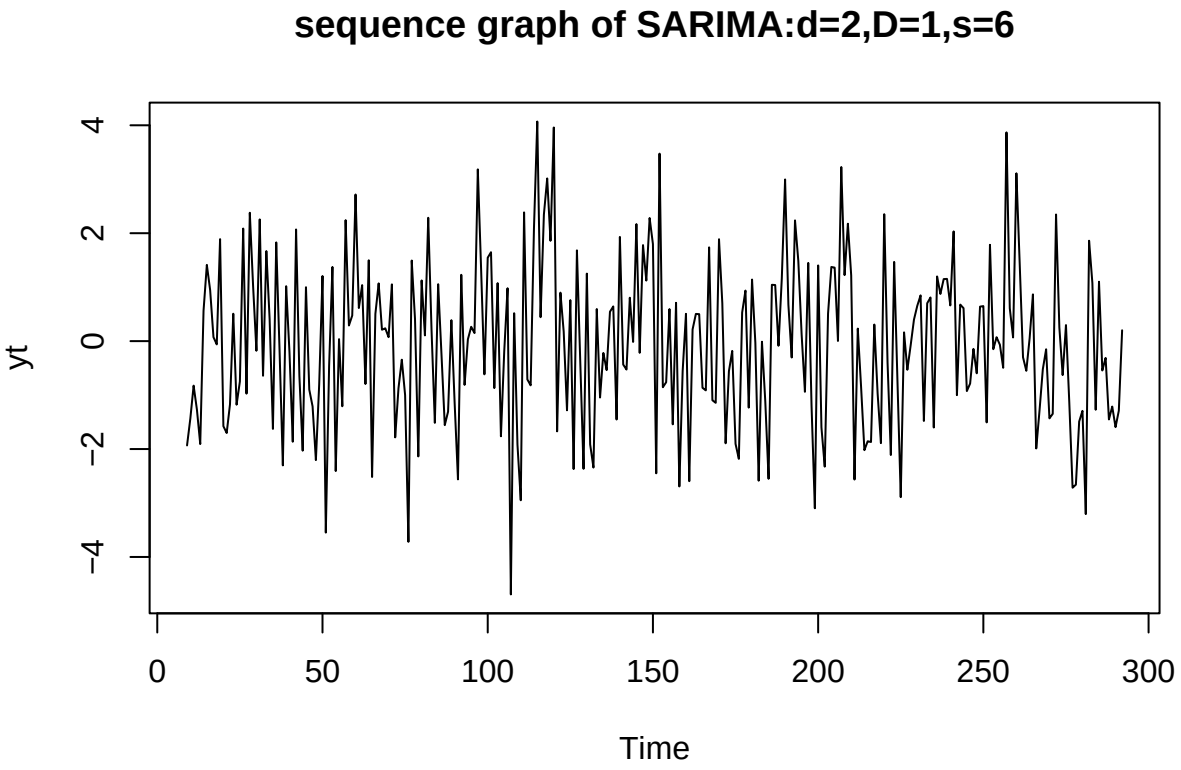



```
plot(diff(my_data,differences = 2),main="sequence graph after diffence:d=2")
```

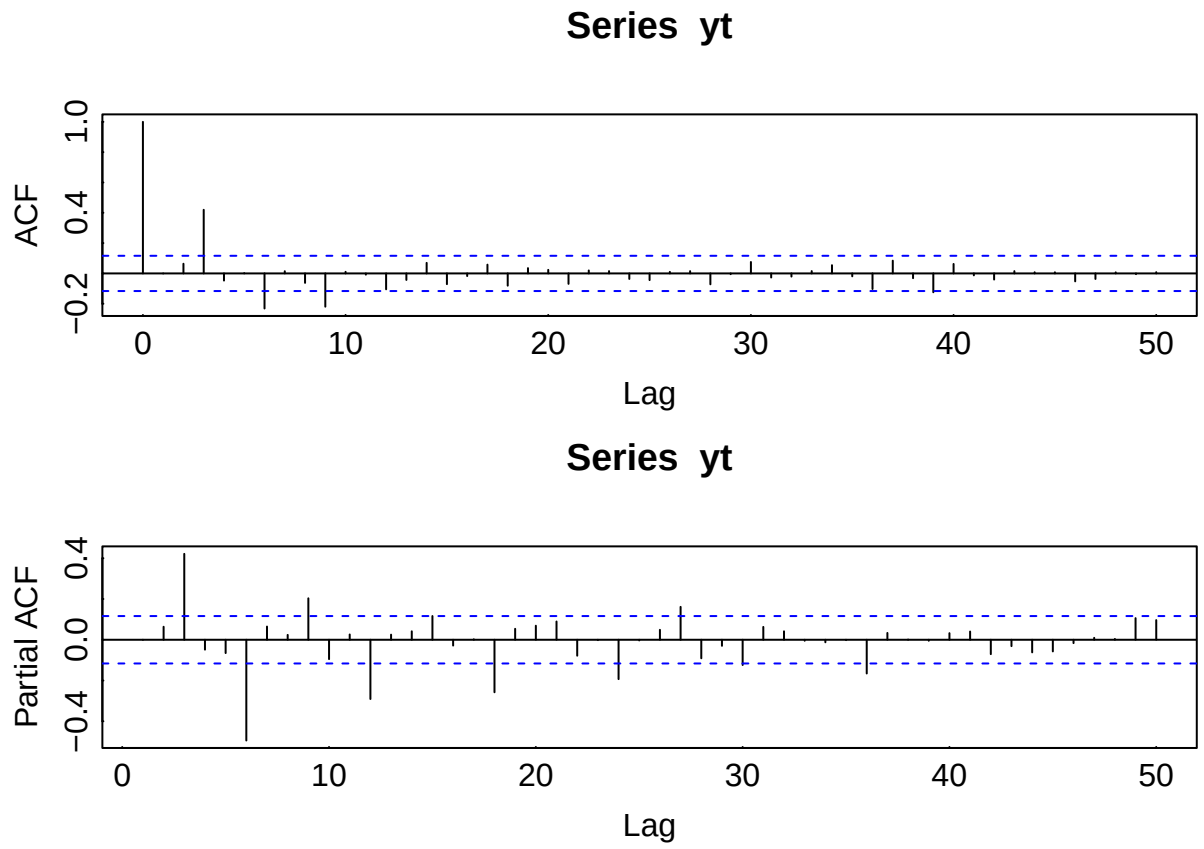


由前述可知，通过二阶差分，序列已无递增趋势，但是具有明显的季节性周期，因此再对一阶差分后的序列做周期为 6 的季节差分.

```
par(mfrow=c(1,1))
yt = diff(diff(my_data,differences = 2),6,1)
plot(yt,main="sequence graph of SARIMA:d=2,D=1,s=6")
```




```
par(mfrow=c(2,1),mar = c(3,3,3,1),mgp=c(1.5,0.2,0),tck=0.005)
acf(yt,lag=50)
pacf(yt,lag=50)
```



都不具有明显的周期性，且都拖尾

4.1 平稳性检验

```
adf.test(yt)
```

```
## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag    ADF p.value
## [1,]   0 -16.82    0.01
## [2,]   1 -11.16    0.01
## [3,]   2  -5.92    0.01
## [4,]   3  -5.81    0.01
## [5,]   4  -5.89    0.01
## [6,]   5  -9.43    0.01
## Type 2: with drift no trend
##      lag    ADF p.value
## [1,]   0 -16.81    0.01
```

```
## [2,]   1 -11.15    0.01
## [3,]   2  -5.92    0.01
## [4,]   3  -5.81    0.01
## [5,]   4  -5.89    0.01
## [6,]   5  -9.43    0.01
## Type 3: with drift and trend
##      lag    ADF p.value
## [1,]   0 -16.80    0.01
## [2,]   1 -11.15    0.01
## [3,]   2  -5.93    0.01
## [4,]   3  -5.82    0.01
## [5,]   4  -5.90    0.01
## [6,]   5  -9.44    0.01
## ----
## Note: in fact, p.value = 0.01 means p.value <= 0.01
```

通过平稳性检验

4.2 纯随机性检验

```
Box.test(yt)
```

```
##
##  Box-Pierce test
##
## data:  yt
## X-squared = 7.1141e-05, df = 1, p-value = 0.9933
```

p 值较大，显示可能由于过差分导致原序列间的相关性被消除，因此无法进行有效的分析，考虑适当减小差分的阶数，取 d=1,D=1,s=6

```
adf.test(diff(diff(my_data),6,1))
```

```
## Augmented Dickey-Fuller Test
## alternative: stationary
##
## Type 1: no drift no trend
##      lag   ADF p.value
## [1,]   0 -2.58  0.0101
## [2,]   1 -2.51  0.0131
## [3,]   2 -2.68  0.0100
```

```
## [4,]   3 -4.56  0.0100
## [5,]   4 -4.50  0.0100
## [6,]   5 -4.30  0.0100
## Type 2: with drift no trend
##      lag   ADF p.value
## [1,]   0 -2.49  0.1340
## [2,]   1 -2.44  0.1563
## [3,]   2 -2.63  0.0921
## [4,]   3 -4.57  0.0100
## [5,]   4 -4.53  0.0100
## [6,]   5 -4.34  0.0100
## Type 3: with drift and trend
##      lag   ADF p.value
## [1,]   0 -2.49   0.370
## [2,]   1 -2.43   0.396
## [3,]   2 -2.61   0.318
## [4,]   3 -4.55   0.010
## [5,]   4 -4.50   0.010
## [6,]   5 -4.30   0.010
## ----
## Note: in fact, p.value = 0.01 means p.value <= 0.01
```

在 type1 的情形下可以通过平稳性检验

4.3 模型拟合

```
#auto.arima(my_data) 自动定阶的输出是 ARIMA(3,1,0) with drift
auto.arima(diff(diff(my_data),6,1)) # 对做完差分之后的序列再次定阶 ARMA(4,0,0)
```

```
## Series: diff(diff(my_data), 6, 1)
## ARIMA(4,0,0) with zero mean
##
## Coefficients:
##          ar1     ar2    ar3      ar4
##       0.9357  0.0949  0.356  -0.4694
## s.e.  0.0522  0.0733  0.073   0.0527
##
## sigma^2 = 1.805:  log likelihood = -488.2
## AIC=986.39   AICc=986.61   BIC=1004.65
```

因此考虑采用 $SARIMA(4,1,0)\times(0,1,0)_6$ 进行拟合

```
library(astsa)
```

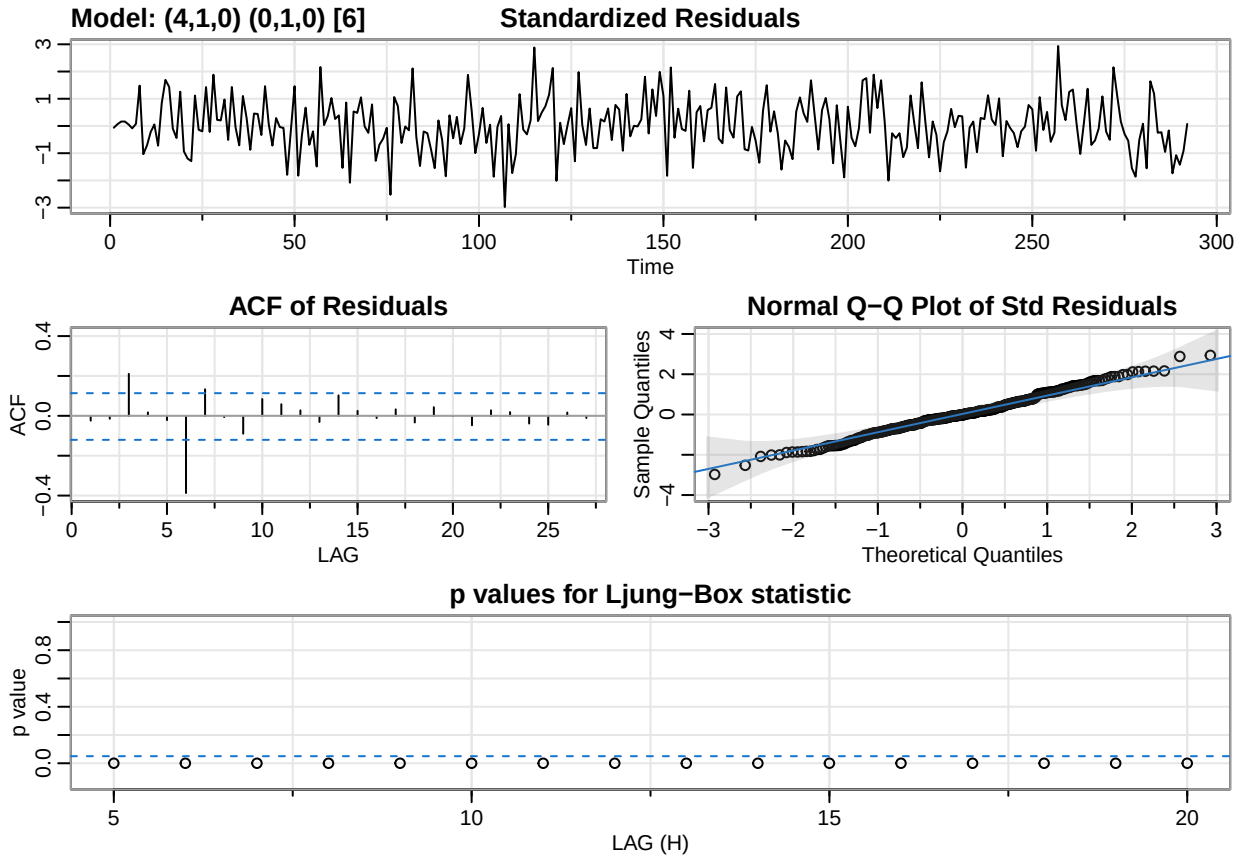
```
## Warning: 程辑包'astsa'是用R版本4.1.3 来建造的
```

```
##
## 载入程辑包: 'astsa'
```

```
## The following object is masked from 'package:forecast':
##
##     gas
```

```
fit_s1 = sarima(my_data,4,1,0,0,1,0,6)
```

```
## initial  value 1.540379
## iter   2 value 0.900815
## iter   3 value 0.793945
## iter   4 value 0.601646
## iter   5 value 0.403997
## iter   6 value 0.324321
## iter   7 value 0.299114
## iter   8 value 0.290333
## iter   9 value 0.288354
## iter  10 value 0.288338
## iter  11 value 0.288337
## iter  11 value 0.288337
## final  value 0.288337
## converged
## initial  value 0.294053
## iter   2 value 0.294036
## iter   3 value 0.294016
## iter   4 value 0.294012
## iter   5 value 0.294009
## iter   6 value 0.294009
## iter   6 value 0.294009
## iter   6 value 0.294009
## final  value 0.294009
## converged
```




```
fit_s2 = arima(my_data,order=c(4,1,0),seasonal = list(order=c(0,1,0),period=6))
fit_s2
```

```
##
## Call:
## arima(x = my_data, order = c(4, 1, 0), seasonal = list(order = c(0, 1, 0), period = 6))
##
## Coefficients:
##          ar1     ar2    ar3      ar4
##       0.9357  0.0949  0.356  -0.4694
## s.e.  0.0522  0.0733  0.073   0.0527
##
## sigma^2 estimated as 1.779:  log likelihood = -488.19,  aic = 986.38
```

结果模型为

$$(1-0.9357B-0.0949B^2-0.356B^3+0.4694B^4)(1-B)(1-B^6)y_t=\epsilon_t$$

对残差做 LB 白噪声检验

```
Box.test(fit_s1$fit$residuals,lag = 12)
```

```
##
##  Box-Pierce test
##
## data:  fit_s1$fit$residuals
## X-squared = 68.006, df = 12, p-value = 7.547e-10
```

```
Box.test(fit_s2$residuals,lag= 12)
```

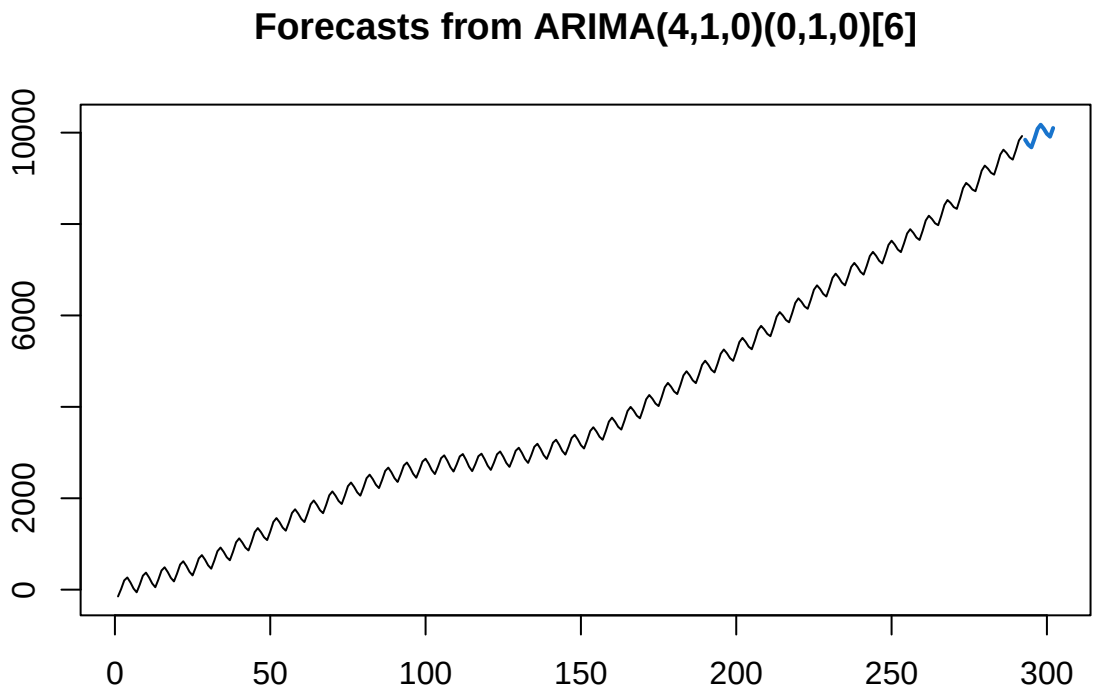
```
##
##  Box-Pierce test
##
## data:  fit_s2$residuals
## X-squared = 68.006, df = 12, p-value = 7.547e-10
```

发现残差不能通过白噪声检验，但通过不断调整参数之后仍无法消除相关性，可能是随机漂移带来的影响【和部分同学交流过，依然没有解决直接用 sarima 拟合的残差检验问题】

4.4 模型预测

不过依然可以进行预测而且似乎效果不错

```
fore_s2 = forecast(fit_s2,h=10)
plot(fore_s2)
```



5 小结

通过对一组时间序列数据的简单处理，大致掌握了时间序列建模的基本方法，实现了对时间序列数据的基本处理，尝试了 R 包中一些估计、检验、预测函数的使用，为后续复杂模型的拟合、多种预测方法的实现等高阶的方法打下了基础。