Chapter 17 – Smoothing Methods
数据平滑方法

Instructor: Zach Zhizhong ZHOU,
Shanghai Jiao Tong University
主讲教师：周志中，上海交通大学

Data Mining for Business Intelligence

Shmueli, Patel & Bruce

Time Series - 1

```
library(fpp)
setwd("C:/BA/TimeSeries")

#时间序列画图
data(melsyd)
plot(melsyd[,"Economy.Class"],
     main="Economy class passengers: Melbourne-Sydney",
     xlab="Year",ylab="Thousands")
data(a10)
plot(a10, ylab="$ million", xlab="Year", main="Antidiabetic
drug sales")
```

Time Series - 2

```
#使用图形展示季节性 Seasonal plot
seasonplot(a10,ylab="$ million", xlab="Year",
           main="Seasonal plot: antidiabetic drug sales",
           year.labels=TRUE, year.labels.left=TRUE, col=1:20,
pch=19)

#Seasonal subseries plots
monthplot(a10,ylab="$ million",xlab="Month",xaxt="n",
          main="Seasonal deviation plot: antidiabetic drug
sales")
#xaxt：x轴坐标先不要画（="n"）

axis(side=1,at=1:12,labels=month.abb)
#side：标注在坐标轴下方，at：标注点是1到12，labels：标注
的标签是月份缩写。
```

Time Series - 3

```
#下面展示时间序列的自相关Autocorrelation。
data(ausbeer)
beer2 <- window(ausbeer, start=1992, end=2006-.1)
# end = 2006-.1确保事件序列取值到2005最后一个quarter，可以尝试end=2006-0.26和end=2006-0.51看有什么结果
beer2
lag.plot(beer2, lags=9, do.lines=FALSE)
#do.lines=FALSE不要把点按时间次序连起来，如果取值为TRUE则连起来。
Acf(beer2)
#Acf: autocorrelation function or ACF.The plot is also known as a correlogram.
```

ACF: Autocorrelation Function

The value of r_k can be written as

$$r_k = \frac{\sum_{t=k+1}^{T} (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{T} (y_t - \bar{y})^2},$$

where T is the length of the time series.

Time Series - 4

```
# Decomposition 分解时间序列
fit <- decompose(ausbeer, type="multiplicative")
plot(fit)

fit <- decompose(ausbeer, type="additive")
plot(fit)
names(fit)
fit$trend
fit$seasonal
fit$random
```

Time Series - 5

```
#使用stl函数（Seasonal Decomposition of Time Series by Loess）分解时间序列
fit <- stl(elecequip, t.window=15, s.window="periodic", robust=TRUE)
#这里的t.window表示trend window，s.window表示seasonal window.
#表示trend和seasonal effect变化的速度有多快，取值越小则变化越快。
#s.window可以取值periodic或者给定一个数值比如4表示每隔4个时间段
plot(fit)
#Forecasting with decomposition 使用分解后的时间序列做预测。
fcast <- forecast(fit, method="naive")
plot(fcast, ylab="New orders index")
```

Simple Exponential Smoothing

We often want something between these two extremes. For example it may be sensible to attach larger weights to more recent observations than to observations from the distant past. This is exactly the concept behind simple exponential smoothing. Forecasts are calculated using weighted averages where the weights decrease exponentially as observations come from further in the past --- the smallest weights are associated with the oldest observations:

$$\hat{y}_{T+1|T} = \alpha y_T + \alpha(1-\alpha)y_{T-1} + \alpha(1-\alpha)^2 y_{T-2} + \cdots, \quad (7.1)$$

where $0 \leq \alpha \leq 1$ is the smoothing parameter. The one-step-ahead forecast for time $T+1$ is a weighted average of all the observations in the series $y_1, \ldots, y_T$. The rate at which the weights decrease is controlled by the parameter α.

Simple Exponential Smoothing

Weighted average form

The forecast at time $t+1$ is equal to a weighted average between the most recent observation y_t and the most recent forecast $\hat{y}_{t|t-1}$,

$$\hat{y}_{t+1|t} = \alpha y_t + (1-\alpha)\hat{y}_{t|t-1}$$

for $t = 1, \ldots, T$, where $0 \leq \alpha \leq 1$ is the smoothing parameter. The process has to start somewhere, so we let the first forecast of y_1 be denoted by ℓ_0. Then

$$\begin{aligned}
\hat{y}_{2|1} &= \alpha y_1 + (1-\alpha)\ell_0 \\
\hat{y}_{3|2} &= \alpha y_2 + (1-\alpha)\hat{y}_{2|1} \\
\hat{y}_{4|3} &= \alpha y_3 + (1-\alpha)\hat{y}_{3|2} \\
&\vdots \\
\hat{y}_{T+1|T} &= \alpha y_T + (1-\alpha)\hat{y}_{T|T-1}
\end{aligned}$$

Simple Exponential Smoothing

Then substituting each equation into the following equation, we obtain

$$\begin{aligned}
\hat{y}_{3|2} &= \alpha y_2 + (1-\alpha)[\alpha y_1 + (1-\alpha)\ell_0] \\
&= \alpha y_2 + \alpha(1-\alpha)y_1 + (1-\alpha)^2\ell_0 \\
\hat{y}_{4|3} &= \alpha y_3 + (1-\alpha)[\alpha y_2 + \alpha(1-\alpha)y_1 + (1-\alpha)^2\ell_0] \\
&= \alpha y_3 + \alpha(1-\alpha)y_2 + \alpha(1-\alpha)^2 y_1 + (1-\alpha)^3\ell_0 \\
&\vdots \\
\hat{y}_{T+1|T} &= \sum_{j=0}^{T-1} \alpha(1-\alpha)^j y_{T-j} + (1-\alpha)^T \ell_0. \qquad (7.2)
\end{aligned}$$

So the weighted average form leads to the same forecast equation (7.1).

Time Series - 6

```
#Simple Exponential Smoothing
oildata <- window(oil,start=1996,end=2007)
plot(oildata, ylab="Oil (millions of tonnes)",xlab="Year",
     main="Oil production in Saudi Arabia from 1996 to
2007")
fit1 <- ses(oildata, alpha=0.2, initial="simple", h=3)
#ses函数是用于Exponential smoothing forecasts
#alpha=0.2, h=3 表示对未来3期进行预测。
#初始值l0可以针对alpha值进行优化，如果是这样则
initial="optimal"。初始值也可以通过最初几个值进行简单计算得
到，如果是这样则initial="simple"
fit2 <- ses(oildata, alpha=0.6, initial="simple", h=3)
fit3 <- ses(oildata, h=3)
#这里没有给出alpha值，所以会自动计算得到最佳的alpha值使得
预测结果的SSE最小。
```

```
plot(fit1, plot.conf=FALSE, ylab="Oil (millions of tonnes)",
     xlab="Year", main="", fcol="white", type="o")
lines(fitted(fit1), col="blue", type="o")
lines(fitted(fit2), col="red", type="o")
lines(fitted(fit3), col="green", type="o")
lines(fit1$mean, col="blue", type="o")
lines(fit2$mean, col="red", type="o")
lines(fit3$mean, col="green", type="o")
legend("topleft",lty=1, col=c(1,"blue","red","green"),
       c("data", expression(alpha == 0.2), expression(alpha == 0.6),
         expression(alpha == 0.89)),pch=1)
fit3$model
```

#怎么知道fit3对应的最佳alpha值是0.89呢？看它得到的最终model就知道了。

Holt-Winters additive method

The component form for the additive method is:

$$\begin{aligned}
\hat{y}_{t+h|t} &= \ell_t + hb_t + s_{t-m+h_m^+} \\
\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},
\end{aligned}$$

where $h_m^+ = \lfloor (h-1) \mod m \rfloor + 1$, which ensures that the estimates of the seasonal indices used for forecasting come from the final year of the sample. (The notation $\lfloor u \rfloor$ means the largest integer not greater than u.) The level equation shows a weighted average between the seasonally adjusted observation $(y_t - s_{t-m})$ and the non-seasonal forecast $(\ell_{t-1} + b_{t-1})$ for time t. The trend equation is identical to Holt's linear method. The seasonal equation shows a weighted average between the current seasonal index, $(y_t - \ell_{t-1} - b_{t-1})$, and the seasonal index of the same season last year (i.e., m time periods ago).

Holt-Winters multiplicative method

The component form for the multiplicative method is:

$$\begin{aligned}
\hat{y}_{t+h|t} &= (\ell_t + hb_t)s_{t-m+h_m^+}. \\
\ell_t &= \alpha \frac{y_t}{s_{t-m}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\
b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1} \\
s_t &= \gamma \frac{y_t}{(\ell_{t-1} + b_{t-1})} + (1-\gamma)s_{t-m}
\end{aligned}$$

and the error correction representation is:

$$\begin{aligned}
\ell_t &= \ell_{t-1} + b_{t-1} + \alpha \frac{e_t}{s_{t-m}} \\
b_t &= b_{t-1} + \alpha\beta^* \frac{e_t}{s_{t-m}} \\
s_t &= s_t + \gamma \frac{e_t}{(\ell_{t-1} + b_{t-1})}
\end{aligned}$$

where $e_t = y_t - (\ell_{t-1} + b_{t-1})s_{t-m}$.

Time Series - 8

```
#Holt-Winters Smoothing Methods
aust <- window(austourists,start=2005)
fit1 <- hw(aust,seasonal="additive")
fit2 <- hw(aust,seasonal="multiplicative")
plot(fit2,ylab="International visitor night in Australia
(millions)",
     plot.conf=FALSE, type="o", fcol="white", xlab="Year")
lines(fitted(fit1), col="red", lty=2)
lines(fitted(fit2), col="green", lty=2)
lines(fit1$mean, type="o", col="red")
lines(fit2$mean, type="o", col="green")
legend("topleft",lty=1, pch=1, col=1:3,
       c("data","Holt Winters' Additive","Holt Winters'
Multiplicative"))
```

Time Series - 9

```
states <-
cbind(fit1$model$states[,1:3],fit2$model$states[,1:3])

colnames(states) <-
c("level","slope","seasonal","level","slope","seasonal")

plot(states, xlab="Year")

fit1$model$state[,1:3]

fitted(fit1)

fit1$mean
```

AR(p) Model

8.3 Autoregressive models

In a multiple regression model, we forecast the variable of interest using a linear combination of predictors. In an autoregression model, we forecast the variable of interest using a linear combination of *past values of the variable*. The term *auto*regression indicates that it is a regression of the variable against itself.

Thus an autoregressive model of order p can be written as

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + e_t,$$

where c is a constant and e_t is white noise. This is like a multiple regression but with *lagged values* of y_t as predictors. We refer to this as an **AR(p) model**.

MA(q) Model

8.4 Moving average models

Rather than use past values of the forecast variable in a regression, a moving average model uses past forecast errors in a regression-like model.

$$y_t = c + e_t + \theta_1 e_{t-1} + \theta_2 e_{t-2} + \cdots + \theta_q e_{t-q},$$

where e_t is white noise. We refer to this as an **MA(q) model**. Of course, we do not *observe* the values of e_t, so it is not really regression in the usual sense.

8.5 Non-seasonal ARIMA models

If we combine differencing with autoregression and a moving average model, we obtain a non-seasonal ARIMA model. ARIMA is an acronym for AutoRegressive Integrated Moving Average model ("integration" in this context is the reverse of differencing). The full model can be written as

$$y'_t = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 e_{t-1} + \cdots + \theta_q e_{t-q} + e_t, \quad (8.1)$$

where y'_t is the differenced series (it may have been differenced more than once). The "predictors" on the right hand side include both lagged values of y_t and lagged errors. We call this an **ARIMA(p, d, q) model**, where

$p =$ order of the autoregressive part;
$d =$ degree of first differencing involved;
$q =$ order of the moving average part.

Time Series - 10

```
#Unit Root Test 单根检验判断时间序列是否为平稳时间序列。
data(usconsumption)
USConsump <- usconsumption[,1]
adf.test(USConsump, alternative = "stationary")
#Dickey-Fuller Test
#很小的p值代表时间序列是平稳的，较大的p值（比如5%）说明时间序列是非平稳的。

kpss.test(USConsump)
#KPSS Test
#很小的p值代表时间序列是平稳的，较大的p值（比如5%）说明时间序列是非平稳的。
```

Time Series - 11

#自动选择ARIMA模型中的参数

fit <- auto.arima(USConsump,seasonal=FALSE)

fit

```
Series: usconsumption[, 1]
ARIMA(0,0,3) with non-zero mean

Coefficients:
         ma1     ma2     ma3  intercept
      0.2542  0.2260  0.2695     0.7562
s.e.  0.0767  0.0779  0.0692     0.0844

sigma^2 estimated as 0.3856:  log likelihood=-154.73
AIC=319.46   AICc=319.84   BIC=334.96
```

This is an ARIMA(0,0,3) or MA(3) model:

$$y_t = 0.756 + e_t + 0.254e_{t-1} + 0.226e_{t-2} + 0.269e_{t-3},$$

where e_t is white noise with standard deviation $0.62 = \sqrt{0.3856}$.